ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Система аналізу великих обсягів даних для комп'ютерної інженерії»

на здобуття освітнього ступеня магістра
зі спеціальності 123 Комп'ютерної інженерії
(код, найменування спеціальності)
освітньо-професійної програми Комп'ютерні системи та мережі
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ *Владислав ГУМІНЮК*
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач(ка) вищої освіти гр. КСДМ-61
Владислав ГУМІНЮК
Ім'я, ПРІЗВИЩЕ
Керівник: Артем АНТОНЕНКО
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ
Рецензент: _____
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра Комп'ютерної інженерії

Ступінь вищої освіти Магістр

Спеціальність 123 «Комп'ютерна інженерія»

Освітньо-професійна програма Комп'ютерні системи та мережі

ЗАТВЕРДЖУЮ
Завідувач кафедрою Наталія ЛАЩЕВСЬКА
_____Ім'я, ПРІЗВИЩЕ
«_____» _____ 2024р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Гумінюк Владислав Юрійович
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: «Система аналізу великих обсягів даних для комп'ютерної інженерії»

керівник кваліфікаційної роботи Артем АНТОНЕНКО, канд.техн.наук, доцент ,
(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «15» жовтня 2024р. №320,

2. Строк подання кваліфікаційної роботи «29» грудня 2024р.
3. Вихідні дані до кваліфікаційної роботи: дана дипломна робота спрямована на дослідження сучасних методів аналізу даних, їх ефективності та можливостей впровадження в різних галузях. Особливу увагу буде приділено методам обробки великих обісягів даних, які вважаються перспективними інструментами для вирішення актуальних завдань, пов'язаних з аналізом великих масивів інформації.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Дослідження предметної області
2. Математичні методи та види технологій в архітектурі проєктів з великими даними

3. Архітектура розробленого програмного продукту та аналіз результатів роботи
4. Перелік ілюстративного матеріалу: *презентація*
5. Дата видачі завдання «01» вересня 2024р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Вибір теми роботи	10.10-12.10.2024	Виконано
2	Збір інформації	12.10-15.10.2024	Виконано
3	Опрацювання інформації	15.10-18.10.2024	Виконано
4	Прочитання додаткової літератури	18.10-22.10.2024	Виконано
5	Опрацювання додаткової літератури	22.10-26.10.2024	Виконано
6	Підготовка до реалізації ПП	26.10-30.10.2024	Виконано
7	Розробка бек-енд частини ПП	30.10-31.10.2024	Виконано
8	Розробка фронт-енд частини ПП	31.10-01.11.2024	Виконано
9	Передзахист	09.12-11.12.2024	Виконано
10	Здача в деканат	20.12-29.12.2024	Виконано

Здобувач вищої освіти _____ В л а д и с л а в Гумінюк

(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник кваліфікаційної роботи _____ А р т е м АНТОНЕНКО

(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина магістерської роботи: 70 с., 10 рис., 3 дод., 21 джерел.

ВЕЛИКІ ДАННІ, МЕТОДИ ОБРОБКИ ДАНИХ, ТЕХНОЛОГІЇ ЗБЕРІГАННЯ, ПРИНЦИПИ РОБОТИ З ДАННИМИ, МАШИННЕ НАВЧАННЯ, NLP, ХМАРНІ ТЕХНОЛОГІЇ, РЕКОМЕНДАЦІЙНА СИСТЕМА

Об`єкт дослдження – основні принципи роботи з великими даними.

Предмет дослідження – використання великих даних у реальному проєкті.

Мета роботи – розробка рекомендаційної системи для демонстрації роботи з великим масивом даних.

Методи дослідження - метод контентної фільтрації для рзробки рекомендаційної системи.

Результати дослідження полягають у створенні програмного продукту, що надає рекомендації щодо фільмів. Практичним досягненням є розробка рекомендаційної системи для відео в рамках веб-додатка, призначеного для користувачів. Ці результати рекомендується використовувати в ситуаціях, коли потрібно сформувати список рекомендацій для товарів.

У подальшому розвитку дослідження доцільно зосередитися на проблемі «холодного старту» та на способах оцінки якості моделі через отримання прямого зворотного зв'язку від користувачів. Також варто звернути увагу на модельні методи реалізації для покращення точності рекомендацій при зростанні кількості споживачів і даних.

ЗМІСТ

ВСТУП....8
РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ....9
1.1 Зростання обсягів даних у світі, потреба в ефективних методах аналізу....9
1.2 Що таке Big data ?....10
1.3 Еволюція великих даних....12
1.4 Історія великих даних....14
1.5 Інфраструктура зберігання даних....16
1.6 Вимоги до інфраструктури великих даних....17
1.7 Нова опція інфраструктури для великих даних....18
1.8 Очищення та підготовка даних....19
1.9 Візуалізація даних....21
1.10 Методи Big Data....23
1.10.1 A|/B тестування....24
1.10.2 Дата майнінг....27
1.10.3 Регресійний аналіз....30
1.10.4 Natural Language Processing (NLP)....32
1.10.5 Метод кластеризації....35
1.10.6 Аналіз часових рядів....37
Висновок до розділу 1....39
РОЗДІЛ 2 МАТЕМАТИЧНІ МЕТОДИ ТА ВИДИ ТЕХНОЛОГІЙ В АРХІТЕКТРУІ ПРОЄКТІВ З ВЕЛИКИМИ ДАНИМИ....41
2.1. Методи обробки даних....41
2.2 Технології зберігання даних....43
2.3 Apache Cassandra....47
2.4 MongoDB....49
2.5 NoSQL бази даних та їх переваги....50
2.8 NLP....61
2.9 Apache Flink та Apache Spark....65
2.10 Хмарні рішення....69
Висновки до розділу 2....72
РОЗДІЛ 3 АРХІТЕКТУРА РОЗРОБЛЕНОГО ПРОГРАМНОГО ПРОДУКТУ ТА АНАЛІЗ РЕЗУЛЬТАТІВ РОБОТИ....75
3.1 Огляд використаних даних....75
3.1.1 Огляд датасетів....75
3.1.2 Огляд використаних технологій....76

3.2 Опис експерименту 78
3.3 Результати роботи програмного продукту....82
3.4 Висновки до розділу 3....83
ДОДАТОК А. Код до програмного продукту....87
ДОДАТОК Б. Доповнення до програмного продукту 91

ВСТУП

У сучасному світі спостерігається експоненційне зростання обсягів даних, що охоплює всі сфери людської діяльності. Поява нових технологій, розвиток інтернету й поширення мобільних пристроїв створюють умови для генерації безпрецедентного обсягу інформації. Це явище, відоме як «вибух даних», стимулює як наукову спільноту, так і бізнес-структури до пошуку нових підходів та інструментів для збору, зберігання та обробки великих масивів даних.

Зростання обсягів даних створює нові можливості для отримання цінної інформації, проте водночас висуває низку важливих викликів. Традиційні методи аналізу стають дедалі менш ефективними у зв'язку з обмеженістю їх обчислювальних спроможностей і неможливістю обробляти великі обсяги інформації в прийнятні терміни. Це породжує потребу в розробці нових, більш ефективних методів аналізу даних, які здатні обробляти інформацію в режимі реального часу та забезпечувати високий рівень точності та релевантності результатів.

У світлі цих викликів, дана дипломна робота спрямована на дослідження сучасних методів аналізу даних, їх ефективності та можливостей впровадження в різних галузях. Особливу увагу буде приділено методам обробки великих обісягів даних, які вважаються перспективними інструментами для вирішення актуальних завдань, пов'язаних з аналізом великих масивів інформації. Надалі будемо використовувати термін Big Data, який є більш вживаним у сучасних медіа.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Зростання обсягів даних у світі, потреба в ефективних методах аналізу

У сучасному світі обсяги даних зростають з безпрецедентною швидкістю, перетворюючи їх на один з найважливіших ресурсів XXI століття. Щодня генеруються терабайти інформації через розвиток цифрових технологій, соціальних мереж, мобільних пристроїв, а також завдяки автоматизації виробничих процесів і поширенню інтернету. Такий обсяг даних несе в собі величезний потенціал для інновацій та створення нових цінностей.

Актуальність дослідженої теми обумовлена тим, що дані є ключовим елементом для ухвалення стратегічних та оперативних рішень як у приватному секторі, так і в державному управлінні. Організації, які ефективно використовують аналітику даних, здатні підвищити свою продуктивність, оптимізувати бізнес-процеси та надавати нові, персоналізовані послуги. Водночас, зростання обсягів даних породжує численні виклики, такі як необхідність у нових методах їхньої обробки, дотримання етичних стандартів та забезпечення безпеки інформації. [1]

Зростання потреби у кваліфікованих фахівцях, здатних аналізувати дані, призвело до значного розвитку таких напрямів, як машинне навчання та штучний інтелект, які дозволяють автоматизувати процеси виявлення закономірностей і прогнозування. Проте, ефективне використання цих технологій потребує розуміння складних алгоритмів і інтерпретації

результатів, що зумовлює необхідність у системному підході до аналізу даних.

Таким чином, ураховуючи значення даних у сучасному світі й важливість їхнього аналізу, дослідження у цій сфері є вкрай актуальним і має велике практичне значення для широкого кола галузей, включно з охороною здоров'я, фінансами, виробництвом, енергетикою та багатьма іншими. Це підкреслює необхідність розвитку новітніх методів і підходів до аналізу даних, які не лише забезпечать конкурентоспроможність, а й сприятимуть розв'язанню глобальних проблем.

1.2 Що таке Big data ?

У сучасну цифрову епоху обсяг даних, що генеруються, вражає уяву — від дописів у соціальних мережах до онлайн-покупок, кожна дія генерує дані. Ці дані, відомі як великі дані, мають величезний потенціал для бізнесу, революціонізуючи способи збору, управління та аналізу інформації. Великі дані відкривають нові можливості в різних галузях, від охорони здоров'я до фінансів і маркетингу. [2]

Великі дані — це великі та складні набори даних, які складно ефективно управляти традиційними методами, охоплюючи чотири основні аспекти: обсяг, швидкість, різноманітність і достовірність. Обсяг стосується загальної кількості даних, що генеруються; швидкість визначає, як швидко ці дані збираються; різноманітність включає різні типи даних, від структурованих до неструктурованих, а достовірність відображає якість і надійність даних. Проблеми конфіденційності та безпеки стали більш

помітними, оскільки аналіз великих обсягів персональних даних викликає етичні питання.

Компанії все частіше використовують великі дані, щоб отримати конкурентну перевагу на ринку. Аналізуючи великі масиви даних, організації можуть виявити цінні закономірності, тенденції та кореляції. Це дозволяє приймати рішення на основі даних, що сприяє зростанню, підвищенню операційної ефективності та задоволеності клієнтів. Наприклад, аналітика великих даних може бути використана для виявлення та запобігання шахрайським діям, а також для персоналізації клієнтського досвіду через адаптацію маркетингових кампаній.

Великі дані не обмежуються лише однією галуззю. Їх застосування поширюється на різні сектори, допомагаючи організаціям впроваджувати інновації, вдосконалювати процеси та стимулювати зростання. У сфері охорони здоров’я великі дані покращують обслуговування пацієнтів, оптимізують розподіл ресурсів і допомагають у медичних дослідженнях. У ритейлі вони надають цінну інформацію про поведінку споживачів, що дозволяє оптимізувати управління запасами та персоналізувати досвід покупок. У фінансовій сфері аналітика великих даних допомагає оцінювати ризики, виявляти шахрайські операції та підвищувати ефективність інвестиційних стратегій.

Використання предиктивної аналітики, основаної на великих даних, може змінити індустрію, дозволяючи точніше прогнозувати поведінку клієнтів і ринкові тенденції. Завдяки цьому організації можуть розробляти проактивні стратегії, оптимізувати операції та випереджати конкурентів. Великі дані також є основою для розвитку технологій штучного інтелекту (ШІ) та машинного навчання (МН), оскільки вони надають необхідні масиви даних для навчання та тестування алгоритмів.

Великі дані змінюють правила гри в сучасному світі. Їх важливість полягає в тому, що вони дають цінну інформацію, покращують процес прийняття рішень і стимулюють інновації. Вони пропонують незліченну кількість переваг у різних галузях, від підвищення ефективності та продуктивності до покращення якості обслуговування клієнтів. З постійним розвитком технологій і додатків перспективи великих даних справді величезні. Організації, які приймають і ефективно використовують великі дані, матимуть всі шанси на процвітання у світі, де все більше керують дані.

1.3 Еволюція великих даних

Концепція великих даних має давні корені. один із перших прикладів зберігання великих обсягів інформації в одній локації - це Велика бібліотека в Олександрії, Єгипет. Бібліотека була заснована приблизно в період між 285 і 246 роками до нашої ери, а згодом зруйнована внаслідок вторгнення пальмірійців між 270 і 275 роками до нашої ери. якщо перенестись до 21 століття, ми можемо побачити, що швидкість збору, управління та аналізу даних ніколи не була такою швидкою, поступаючися тільки зростаючій складності цих даних. Тут і виникає проблема великих даних.

Що ж таке великі дані? Це значний потік структурованих, напівструктурованих і неструктурованих даних. Це інформація, яка надходить у набагато більших обсягах, з вищою швидкістю, у різноманітних форматах файлів і з широкого спектра джерел, ніж просто структуровані дані. Термін «великі дані» був введений наприкінці 1990-х років, коли його офіційно запровадили дослідники NASA Майкл Кокс та Девід Еллсворт у своїй статті 1997 року «Керування запитами на підкачку для позаядерної візуалізації» (Application-Controlled Demand Paging for

Out-of-Core Visualization). Вони використовували цей термін, щоб описати труднощі, пов'язані з обробкою та візуалізацією величезних обсягів даних, які генерувалися суперкомп'ютерами.

У 2001 році експерт з даних та аналітики Даг Лейні опублікував статтю «Управління 3D-даними: Управління обсягом, швидкістю та різноманітністю даних», у якій були визначені три основні компоненти, що й сьогодні використовуються для характеристики великих даних: Обсяг (розмір даних), Швидкість (темп, з яким дані зростають) і Різноманітність (кількість типів даних та джерел, з яких вони походять).

Концепція великих даних еволюціонувала з плином часу, а технологічний прогрес і розширення можливостей зв'язку сприяли її зростанню. У минулому дані збирали переважно зі структурованих джерел, таких як бази даних. Однак з появою інтернету, соціальних мереж і пристроїв Інтернету речей (IoT) дані генеруються з безпрецедентною швидкістю з різних джерел. Цей вибух даних призвів до зростання потреби в передових аналітичних інструментах і методах, щоб розібратися в них.

1.4 Історія великих даних

Виникнення даних і великих даних має давню та багатогранну історію. Багато технологічних досягнень було здійснено під час Другої світової війни, переважно з військовими цілями. Проте згодом ці досягнення стали корисними для комерційного сектора, а врешті-решт і для широкого загалу, оскільки персональні комп'ютери стали доступними для звичайних споживачів.

1940-ті до 1989 року - системи зберігання даних та персональні настільні комп'ютери. Витоки електронного зберігання даних можна знайти в розробці першого у світі програмованого комп'ютера - електронного числового інтегратора та комп'ютера (ENIAC). Він був спроектований армією США під час Другої світової війни для вирішення чисельних задач, наприклад, для розрахунку дальності артилерійського вогню. Потім у початку 1960-х років компанія International Business Machines (IBM) випустила перший транзисторний комп'ютер під назвою TRADIC, що допомогло дата-центрам вийти з військової сфери та обслуговувати більш загальні комерційні потреби.

Першим персональним настільним комп'ютером, який функціонував з графічним інтерфейсом (GUI), стала модель Lisa, представлена компанією Apple у 1983 році. Протягом 1980-х років такі компанії, як Apple, Microsoft і IBM, випустили широкий спектр персональних комп'ютерів, що призвело до буму у володінні персональними комп'ютерами та їх використанні в домашніх умовах. Таким чином, електронне зберігання стало доступним для широкої аудиторії.

Між 1989 і 1993 роками британський комп'ютерний вчений сер Тім Бернерс-Лі створив основні технології, які стали основою для того, що ми

сьогодні знаємо як Всесвітня мережа. Ці веб-технології включали мову розмітки гіпертексту (HTML), уніфікований ідентифікатор ресурсу (URI) та протокол передачі гіпертексту (HTTP). У квітні 1993 року було ухвалено рішення про безкоштовне надання базового коду для цих веб-технологій.

Це рішення дало можливість людям, бізнесу та організаціям, які могли дозволити собі доступ до інтернету, виходити в онлайн і обмінюватися даними з іншими комп'ютерами, підключеними до інтернету. Оскільки дедалі більше пристроїв отримували доступ до мережі, це призвело до масового збільшення обсягу інформації, доступної для людей та можливостей обміну даними в будь-який момент.

На початку 2000-х років такі компанії, як Amazon, eBay і Google, створили великий обсяг веб-трафіку, а також комбінацію структурованих і неструктурованих даних. У 2002 році Amazon також запустила бета-версію AWS (Amazon Web Services), яка відкрила платформу Amazon.com для всіх розробників. До 2004 року для неї було створено понад 100 додатків.

AWS знову запустили в 2006 році, пропонуючи широкий спектр хмарних інфраструктурних послуг, включно з Simple Storage Service (S3) та Elastic Compute Cloud (EC2). Публічний запуск AWS привернув увагу багатьох клієнтів, таких як Dropbox, Netflix і Reddit, які хотіли стати хмарно-орієнтованими і уклали партнерство з AWS до 2010 року.

Соціальні мережі, такі як MySpace, Facebook і Twitter, також сприяли зростанню поширення неструктурованих даних. Це передбачало обмін зображеннями та аудіофайлайми.

Технічна складова проблематики Big Data є надзвичайно складною та багатогранною, що вимагає інтегрованого підходу до розробки інфраструктури, обчислень, управління, аналізу та візуалізації даних.

1.5 Інфраструктура зберігання даних

Щоб використовувати великі набори даних, бізнесам необхідно впроваджувати рішення для великих даних у вигляді програмного забезпечення для розподілених обчислень, яке може збирати, розподіляти, зберігати та керувати масивними неструктурованими даними в режимі реального часу. Ці системи спрощують та організовують обробку та розподіл даних, що працюють на сотнях або тисячах машин одночасно.

Приклади open-source платформ для великих даних включають:

- ClickHouse: масштабована система керування базами даних (DBMS) з відкритим кодом. Вона забезпечує онлайн-аналітичну обробку (OLAP), підтримує програми, що працюють з величезними структурованими наборами даних, і дозволяє користувачам створювати звіти за допомогою SQL-запитів у реальному часі.[3]

- Apache Hadoop: набір утиліт з відкритим вихідним кодом, розроблених на базі Java, які керують великими обсягами даних для зберігання та обробки. Розподілене зберігання та паралельна обробка реалізується завдяки мережі численних комп'ютерів, які вирішують менші завдання одночасно.

- Apache Spark: фреймворк з відкритим кодом для обробки даних, призначений для великомасштабної обробки даних. Він дозволяє розподіляти завдання обробки даних на кілька комп'ютерів паралельно для підтримки великих даних і машинного навчання, яке вимагає величезної обчислювальної потужності для аналізу великих наборів даних.

1.6 Вимоги до інфраструктури великих даних

Для підтримки оптимальної роботи цих платформ програмного забезпечення для великих даних повинна бути відповідна інфраструктура. Ця інфраструктура для великих даних має бути налаштована під ці платформи.

Швидко збирати великі обсяги даних, опрацьовувати великі обсяги дискових і мережевих I/O, забезпечувати високу доступність систем, включно з можливостями швидкого відновлення. Забезпечувати масштабованість машин для збільшення обсягу пам'яті та/або обчислювальної потужності. Щоб максимально використовувати інфраструктуру великих даних для бізнесу, важливо мати команду з відповідними навичками. Для професіоналів, які прагнуть розвиватися в управлінні великими рішеннями для даних та інфраструктурою, відмінною ідеєю буде отримати сертифікат інженера з даних онлайн. Ці курси та сертифікації не лише підвищують можливості бізнесу та кар'єрні перспективи, але й забезпечують засвоєння основ розробки, реалізації та підтримки масштабованих середовищ даних – ключових елементів для вирішення завдань великих даних.[4]

Щоб відповідати наведеним вище операційним вимогам, користувачі повинні розуміти свої варіанти інфраструктур. Інфраструктура може допомогти організаціям приймати кращі рішення, вдосконалювати ефективність роботи та отримувати конкурентні переваги. Взагалі існує три найбільш поширені варіанти інфраструктур для великих даних:

- хмарна інфраструктура: гнучкість масштабування вгору або вниз і доступ до розвинених інфраструктурних сервісів без початкових інвестицій зробила загальнодоступні та приватні хмари популярним варіантом для вдосконалених можливостей аналітики, алгоритмів машинного навчання та обробки даних у режимі реального часу.

- інфраструктура на місці: незважаючи на початкові інвестиції та більш обмежені можливості масштабування, такі рішення часто обираються командами, які хочуть мати більше контролю та безпеки над своїми дуже чутливими даними.

- гібридна інфраструктура: незважаючи на необхідність управління кількома платформами, багато хто обирає поєднання різних хмарних та локальних рішень для кращого варіантів з обох світів і отримання всіх потенційних переваг.

Кожен з трьох варіантів пропонує свої унікальні переваги. Але всі три також представляють певні обмеження та потенційні складнощі. Вони мають компроміси між витратами та контролем, масштабованістю та безпекою тощо. Опція гібридної інфраструктури допомагає забезпечити переваги кожного з них, але не обов'язково усуває недоліки кожного.

1.7 Нова опція інфраструктури для великих даних

OpenMetal Clouds пропонує новий і особливий тип інфраструктури для великих даних, яка була визнана придатною для використання з платформами програмного забезпечення для великих даних, такими як ClickHouse, Hadoop і Spark.

Не тільки OpenMetal об'єднує найкращі можливості традиційних загальнодоступних, приватних хмар і апаратних серверів, але й пропонує приватну хмарну платформу на вимогу. Вона побудована на базі OpenStack для використання переваг інтеграцій з відкритим вихідним кодом.

Кожен OpenMetal Cloud починається з Cloud Core із трьох виділених апаратних серверів, сховища Ceph та найбільш затребуваних стандартних функцій прямо з коробки. Це призводить до швидко розгортається та масштабованого одноорендарного середовища. Це підхід «краще з усіх світів», який розмиває межі між загальноприйнятими варіантами інфраструктури для великих даних.

Розподілена обробка даних є ключовою особливістю сучасних платформ для Big Data. Apache Hadoop та Apache Spark забезпечують асинхронну, паралельну обробку даних, що дозволяє виконувати складні обчислювальні завдання в значно скорочені терміни. Hadoop працює за принципом розподілу задачі на менші підзадачі, що обробляються паралельно, тоді як Spark забезпечує обробку даних у пам’яті, що робить його суттєво швидшим для деяких типів завдань.

1.8 Очищення та підготовка даних

Експоненційне зростання цифрових даних з різних джерел, таких як сенсорні мережі, соціальні медіа та бізнес-транзакції, вимагає впровадження надійних стратегій управління даними. Первісні дані часто містять неточності, такі як шум, пропущені значення та непослідовності, що негативно впливають на якість аналітики великих даних. Очищення даних—процес виявлення та корекції цих помилок—є важливим, хоча й

складним завданням при роботі з великими наборами даних. Механізми очищення даних можна класифікувати в кілька категорій:

- машинне навчання: ці методи використовують алгоритми для виявлення шаблонів на основі чистих даних і прогнозування коректувань для «брудних» даних. Вони є ефективними, але можуть бути складними та вимогливими до обчислювальних ресурсів. Завдяки прогресу у сфері штучного інтелекту, машинне навчання стає дедалі важливішим інструментом у покращенні точності очищення даних.

- вибірка: вибірка даних для очищення знижує витрати на обробку та час. Цей підхід підходить для великих наборів даних, хоча його точність залежить від репрезентативності вибірки. За допомогою аналітики вибірки можна швидше та економічніше очищати дані, забезпечуючи їх придатність для подальшої аналітики.

- експертний підхід: залучення людського досвіду, часто через краудсорсингові платформи, для покращення якості даних забезпечує високу точність, але може бути повільнішим і дорожчим за автоматизовані методи. Однак, людський фактор і досвід можуть суттєво підвищити якість кінцевого набору даних, особливо в спеціалізованих галузях.

- правила: використання чітко визначених правил для виявлення та корекції непослідовностей в даних. Простота цих методів полегшує їх впровадження, але створення комплексного набору правил є складним завданням. Важливо забезпечити гнучкість правил, щоб вони могли адаптуватися до змін у даних.

- фреймворки: інтеграція кількох методів очищення даних у фреймворки підвищує якість даних. Цей підхід є більш цілісним, але збільшує складність. Впровадження таких фреймворків може полегшити керування складними процесами очищення та дозволити ефективніше виконувати завдання в різних умовах. Кожна категорія оцінюється за параметрами: масштабованість, ефективність, точність та зручність у використанні. Існують компроміси між цими параметрами. Наприклад, високоточні методи можуть бути менш масштабованими або ефективними. Важливо ретельно обирати методи, зважаючи на пріоритети конкретного проекту.

Серед актуальних напрямів подальших досліджень виділяються обробка великих обсягів неструктурованих даних, вирішення проблем якості даних у потоках інформації, а також розробка більш всебічних параметрів оцінки методик очищення даних. Це дозволить оптимізувати процеси очищення і адаптувати їх до нових викликів, які ставить цифрова ера.

Загалом, різноманіття наявних методик очищення даних дає змогу обирати найбільш підходящі методи з урахуванням конкретних потреб та обмежень, що відкриває нові перспективи для розвитку цієї галузі. Інтеграція нових технологій та постійне вдосконалення існуючих підходів сприятиме підвищенню ефективності роботи з великими даними та розширюватиме можливості їхнього використання в різних сферах.

1.9 Візуалізація даних

Візуалізація великих даних зосереджується на графічному представленні великих обсягів інформації. Інструменти для цього використовують програмне забезпечення, яке дозволяє здійснювати інтуїтивний та інтерактивний аналіз для виявлення шаблонів, кореляцій і причинно-наслідкових зв'язків. Вона також відома як візуальна розвідка, інтерактивна візуалізація, інформаційна візуалізація, візуальна аналітика та дослідницький аналіз даних, що підкреслює багатогранність цієї сфери.

Зіштовхуючись із величезними обсягами, швидкістю, різноманітністю та мінливістю даних, існує кілька ключових викликів: Системи повинні обробляти мільярди точок даних з майже миттєвою реакцією на взаємодії користувачів, що потребує мілісекундного часу відгуку. Необхідність у візуалізації та аналізі необроблених даних на льоту замість попередньої обробки. Проблема перевантаження інформацією вимагає ефективної абстракції та узагальнення для збереження ясності. Необхідність у адаптації систем до потреб користувача, що вимагає персоналізації та автоматичного налаштування на основі доступних ресурсів і поведінки користувачів. Сучасні системи для великих даних перевершують традиційні, дозволяючи обробку в реальному часі, візуальну масштабованість та персоналізацію. Вони усувають недоліки традиційних систем, які працюють з обмеженими, статичними даними.[5]

Застосовуються різні підходи:

- зменшення даних шляхом вибірки, фільтрації та агрегації;
- ієрархічне дослідження для багаторівневого вивчення, особливо у візуалізації графів;
- прогресивні результати покращують час відгуку за допомогою поступового надання результатів;

- інкрементальна та адаптивна обробка дозволяє обробку даних невеликими частинами;

- кешування та передзавантаження оптимізують час відгуку шляхом передбачення запитів користувачів;

- візуалізаційні рекомендації сприяють навігації даними, пропонуючи відповідні техніки візуалізації, які підкреслюють цікаві шаблони.

Техніки візуалізації великих даних застосовуються в фізиці та астрономії, біоінформатиці, бізнес-аналітиці, метеорології та кліматології, дозволяючи фахівцям виявляти важливі шаблони і отримувати інсайти.

1.10 Методи Big Data

В епоху цифрової трансформації обсяги даних, які генеруються щоденно, зростають експоненційно. Великі дані (Big Data) охоплюють величезні набори інформації з різноманітних джерел, включаючи соціальні мережі, фінансові транзакції, сенсори IoT, та багато іншого. Обробка та аналіз цих даних стають критично важливими для компаній, які прагнуть отримати конкурентну перевагу. Для цього розроблено різноманітні методи і технології, які допомагають ефективно та швидко видобувати цінну інформацію.[6]

Серед основних методів, які використовують для аналізу великих даних, варто виділити машинне навчання, обробку природної мови та інтелектуальний аналіз даних. Машинне навчання дозволяє комп'ютерам робити прогнози або приймати рішення на основі даних, не будучи

спеціально запрограмованими на виконання цього завдання. Обробка природної мови допомагає зрозуміти та інтерпретувати людську мову, що особливо важливо для аналізу текстових даних. Інтелектуальний аналіз, у свою чергу, спрямований на видобуток цікавих патернів і кореляцій у великих наборах даних.

Технології, які підтримують ці методи, включають такі платформи, як Hadoop та Spark, які забезпечують розподілене зберігання та обробку даних. Бази даних NoSQL, такі як MongoDB та Cassandra, надають можливість ефективного зберігання даних, що не підходять для традиційних реляційних баз даних. Інструменти для обробки потокової інформації, такі як Apache Kafka, дозволяють аналізувати дані в реальному часі, забезпечуючи компаніям можливість швидкого реагування на події. Таким чином, сукупність цих методів і технологій формує комплексний підхід до роботи з великими даними, забезпечуючи ефективність і швидкість прийняття рішень.

1.10.1 A|/B тестування

A/B тестування в умовах великої кількості даних (Big Data) стає ще більш потужним інструментом, що дозволяє бізнесам не лише зберігати, але й ефективно аналізувати величезні обсяги інформації для прийняття обґрунтованих рішень. Використання методів A/B тестування у контексті Big Data допомагає глибше зрозуміти поведінку споживачів, оптимізувати процеси і підвищити конкурентоспроможність.[7]

Однією з основних переваг A/B тестування в епоху Big Data є здатність опрацьовувати та аналізувати великі обсяги даних в реальному часі. Це дозволяє підприємствам, які працюють із значними потоками інформації, проводити тести з високою частотою та отримувати швидкі

результати. Таким чином, компанії можуть більш оперативно реагувати на зміни в поведінці споживачів та оперативно адаптувати свої стратегії до нових умов.

З використанням алгоритмів машинного навчання та аналітики даних, A/B тестування стає ще більш цілеспрямованим. Аналітика на основі Big Data дозволяє виявити закономірності в поведінці користувачів, які можуть бути непомітними при меншій кількості даних. Це дає змогу створювати більш детальні гіпотези та специфічні варіанти для тестування, що підвищує шанси на успіх.

Крім того, дані, зібрані в процесі A/B тестування, можуть бути інтегровані з іншими джерелами даних, такими як соціальні мережі, аналітика сайту або CRM-системи. Це дозволяє компаніям отримати комплексний погляд на свій бізнес і споживачів, що веде до більш усвідомлених рішень та стратегій.

У світі, де конкуренція зростає, а споживчі уподобання змінюються швидше, ніж будь-коли, використання A/B тестування в контексті Big Data допомагає зменшити ризики та підвищити ймовірність успішності нових ідей. Тестування дозволяє перевірити численні варіанти стратегій одночасно, швидко виявляючи найбільш ефективні рішення.

Крім того, A/B тестування в епоху Big Data не лише підвищує рентабельність інвестицій (ROI), але й сприяє формуванню аналітичної культури в організації. Залучення даних у процес прийняття рішень стимулює співробітників до використання аналітики, що в свою чергу покращує загальний рівень ефективності бізнес-процесів.

A/B тестування—це метод експериментування, де дві версії елемента (наприклад, веб-сторінки, листа чи рекламного оголошення)

порівнюються, щоб визначити, яка версія працює краще. Основний алгоритм A/B тестування має наступний вигляд:

- визначення мети: першим кроком в A/B тестуванні є визначення конкретних цілей, яких ви хочете досягти, наприклад, збільшення конверсій, покращення кліків або залучення користувачів;
- підготовка гіпотези: визначте, які зміни можна внести і чому. Наприклад, «Зміна кольору кнопки на червоний призведе до збільшення кількості кліків»;
- розробка варіантів: необхідно створити дві версії елемента: контрольну (A) і альтернативну (B), де змінюється лише один фактор;
- розподіл трафіку: випадковим чином необхідно розподілити трафік між двома версіями, щоб у кожній було достатньо даних для статистично значущих результатів;
- збір даних та аналіз: після запуску тесту данні аналізуються, щоб визначити, яка версія досягає кращих результатів відповідно до поставлених метрик;
- висновки та імплементація: на основі результатів тестунеобхідно зробити висновки та, за необхідності, впровадити зміни в продукт або процес;
- документація та повторення: отримані результати та уроки документуються. Використовуються дані для планування майбутніх тестів.

A/B тестування є потужним інструментом для прийняття обґрунтованих рішень, оскільки дозволяє компаніям перевіряти реальні впливи змін на аудиторію перед впровадженням нововведень у ширшому масштабі.

1.10.2 Дата майнінг

Дата майнінг у контексті Big Data - це потужний набір технологій і методів, що дозволяє виявляти приховані закономірності та корисну інформацію з величезної кількості даних. Сучасні компанії стикаються з викликом обробки та аналізу обсягів інформації, які зростають геометрично, включаючи структуровані, напівструктуровані та неструктуровані дані. За допомогою методів дата майнінгу, таких як класифікація, кластеризація та асоціативний аналіз, підприємства можуть перетворювати необроблену інформацію на цінні знання, що позитивно впливають на бізнес-процеси, стратегічне управління та прийняття рішень.

Однією з основних переваг дата майнінгу є можливість отримання глибших інсайтів у поведінку споживачів. За допомогою аналізу даних компанії можуть виявити тренди, прогнозувати потреби клієнтів і адаптувати свої продукти чи послуги відповідно до отриманих знань. Наприклад, рітейлери можуть використовувати дата майнінг для аналізу покупок, виявлення улюблених товарів своїх споживачів та навіть пропонування персоналізованих рекомендацій. Цей підхід не лише покращує взаємодію з клієнтами, але й збільшує рентабельність інвестицій.

Однак із зростанням обсягу даних і складності їх аналізу виникають нові виклики, пов'язані з безпекою та конфіденційністю. Використання алгоритмів дата майнінгу необхідно супроводжувати етичними нормами, щоб забезпечити захист особистої інформації користувачів. Тому важливо, щоб компанії не тільки впроваджували технології дата майнінгу, але й відповідально підходили до використання зібраних даних. Успішна реалізація дата майнінгу в епоху Big Data може суттєво підвищити конкурентоспроможність підприємств і забезпечити їм стійке зростання, якщо вирішити ці етичні та технічні виклики.[8]

Оскільки великі дані продовжують формувати ландшафт сучасної аналітики, методи інтелектуального аналізу даних надають інструменти, необхідні для отримання цінної інформації. Поєднання класифікації, кластеризації, видобування правил асоціацій, регресійного аналізу, текстового аналізу, аналізу часових рядів, виявлення аномалій та методів ансамблів дає можливість організаціям видобувати значущі знання з величезних і різноманітних наборів даних. В епоху великих даних використання цих методів не лише дозволяє краще приймати рішення, але й відкриває нові шляхи для інновацій та відкриттів. З розвитком технологій зростають і можливості інтелектуального аналізу даних, що дозволяє організаціям точно і глибоко орієнтуватися в складнощах аналізу великих даних.

Дата майнінг у контексті Big Data передбачає процеси, які дозволяють видобувати корисну інформацію та інсайти з великих обсягів даних. Ось як цей процес зазвичай функціонує:

1 Збір даних: першим етапом є збір даних з різних джерел, включаючи бази даних, соціальні мережі, веб-сайти, датчики, IoT (Internet of Things) пристрої тощо. Часто дані мають різні

формати (структуровані, напівструктуровані, неструктуровані), тому їх потрібно підготувати для подальшого аналізу.

2 Попередня обробка даних: на цьому етапі проводиться очищення і підготовка даних до аналізу. Це включає в себе видалення недостовірних або неповних значень, нормалізацію та трансформацію даних, а також агрегування даних для спрощення подальшого аналізу. Цей процес є критично важливим, оскільки якість даних безпосередньо впливає на результати майнінгу.

3 Аналіз даних: тут застосовуються різні алгоритми та моделі для виявлення закономірностей та структур у зібраних даних. Методики можуть включати класифікацію (групування даних за певними ознаками), кластеризацію (виявлення підгруп у даних без попередньо заданих категорій), асоціативні правила (визначення взаємозв'язків між елементами), а також прогностичну аналітику (передбачення тенденцій на основі історичних даних).

4 Інтерпретація результатів: після виконання аналізу важливо проаналізувати отримані результати і перетворити їх на зрозумілі інсайти. Це може включати візуалізацію даних, представлення звітів або діаграм для кращого сприйняття інформації. Команди аналітиків зазвичай працюють над тим, щоб пояснити значення виявлених закономірностей і рекомендацій для бізнесу.

5 Впровадження результатів: останнім етапом є впровадження отриманих інсайтів у стратегії бізнесу. Це може включати оптимізацію маркетингових кампаній, покращення

обслуговування клієнтів, удосконалення продуктів і послуг, а також інші заходи, спрямовані на підвищення ефективності діяльності підприємства.

1.10.3 Регресійний аналіз

Регресійний аналіз - це аналіз керованого навчання, де кероване навчання - це аналіз або прогнозування даних на основі попередньо наявних або минулих даних. Для керованого навчання ми маємо як навчальні, так і тестові дані. Регресійний аналіз - це один із статистичних методів аналізу та прогнозування даних. Регресійний аналіз використовується для прогнозування даних або кількісних чи числових даних.

У мові програмування R регресійний аналіз - це статистична модель, яка показує взаємозв'язок між залежними змінними та незалежними змінними. Регресійний аналіз використовується в багатьох галузях, таких як машинне навчання, штучний інтелект, наука про дані, економіка, фінанси, нерухомість, охорона здоров'я, маркетинг, бізнес, наука, освіта, психологія, спортивний аналіз, сільське господарство та багато інших. Основна мета регресійного аналізу - визначити взаємозв'язок між змінними, характер і силу зв'язку між змінними, а також зробити прогнози на основі моделі.

Регресійний аналіз у контексті Big Data є важливим статистичним методом, який дозволяє досліджувати та моделювати залежність між змінними [9]. Це може бути особливо корисно для виявлення взаємозв'язків між різними факторами та для прогнозування значень цільової змінної на

основі даних. Основними аспектами регресійного аналізу в Big Data можуть виступати:

1. Типи регресійного аналізу: регресійний аналіз включає різні види, такі як лінійна регресія, логістична регресія, поліноміальна регресія та багато інших. Вибір конкретного типу залежить від характеру даних і цілі аналізу. Наприклад, лінійна регресія використовується для моделювання залежності між двома змінними, а логістична регресія – для бінарних (так/ні) результатів.

2. Обробка великих обсягів даних: у Big Data регресійний аналіз потрібно адаптувати до величезних обсягів даних, які можуть містити мільйони або навіть мільярди записів. Сучасні інструменти та платформи, такі як Apache Spark, Hadoop або TensorFlow, дозволяють проводити регресійний аналіз на великих наборах даних, використовуючи паралельну обробку для швидшого отримання результатів.

3. Оптимізація моделі: оскільки у великій кількості даних можуть бути тисячі змінних, важливо правильно вибрати значущі ознаки для моделі. Це може включати методи зменшення розмірності, такі як Principal Component Analysis (PCA) або використання регуляризації (Lasso, Ridge), що допомагає уникнути проблеми перенавчання і покращує загальну стійкість моделі.

4. Валідація та перевірка моделі: в умовах Big Data важливо не лише побудувати модель, але й оцінити її ефективність. Це може бути досягнуто шляхом використання технік, як крос-валідація або тестові набори для перевірки, щоб переконатися,

що модель здатна передбачати результати для непередбачених даних.

5. Застосування регресійного аналізу: регресійний аналіз у Big Data використовується в багатьох сферах, включаючи фінансовий сектор для прогнозування ризиків, в маркетингу для оцінки впливу кампаній на продажі, у виробництві для оптимізації процесів та в охороні здоров'я для аналізу результатів лікування.

Таким чином, регресійний аналіз у середовищі Big Data є потужним інструментом, що дозволяє організаціям отримувати цінну інформацію з великих обсягів даних, робити прогнози та приймати обґрунтовані рішення. Використання сучасних аналітичних технологій дозволяє досягати високої точності та ефективності у прогнозах, що, в свою чергу, сприяє розвитку і зростанню підприємств.

1.10.4 Natural Language Processing (NLP)

Обробка природної мови - це форма і застосування штучного інтелекту, яка допомагає комп'ютерам «читати» текст, подібно до того, як людина дає машинам здатність розуміти мову. Вона включає в себе численні методи, такі як лінгвістика, семантика, машинне навчання і статистика, для вилучення контексту і значення з даних, що дозволяє машинам всебічно розуміти сказане або написане.

Замість того, щоб розшифровувати окремі слова або короткі фрази, НЛП допомагає комп'ютерам розуміти цілісні думки в реченні, набраному або вимовленому людиною. Хоча фільтрація спаму або позначення частин мови допомагають у такій інтерпретації, вона не завжди є точною. Однак,

як і багато людей, більшість цих моделей не здатні вловити лінгвістичні тонкощі, такі як контекст, ідіоми, іронія чи сарказм.

Алгоритмічні моделі на кшталт Bag-of-Words (яка фокусується на загальному підсумовуванні), n-грами та приховані марковські моделі (HMM) не змогли адекватно вловити та декодувати складнощі людської мови у великих даних

Natural (NLP) у контексті Big Data є важливою галуззю, яка займається аналізом, обробкою та розумінням людської мови за допомогою обчислень. Завдяки величезній кількості текстових та мовленнєвих даних, які генеруються щодня — через соціальні мережі, електронні листи, відгуки, новини і багато інших джерел, NLP стає невід'ємною частиною стратегій аналізу даних для психолого та бізнесових цілей [10].

1. Обробка великих обсягів текстових даних: у Big Data NLP стає важливим для виявлення смислів і закономірностей в неструктурованих даних, таких як текст. Використання методів NLP дозволяє аналізувати великі обсяги тексту, виявляючи теми, емоції та настрої, що допомагає компаніям краще розуміти споживчі вподобання. Наприклад, аналіз відгуків про продукти може допомогти виявити, які аспекти викликають позитивні або негативні реакції у споживачів.

2. Аналіз настроїв і семантичне розуміння: один з ключових аспектів NLP — це аналіз настроїв (Sentiment Analysis), який дозволяє визначити емоційне забарвлення тексту. У умовах Big Data це особливо корисно для моніторингу брендів і розуміння громадської думки. Завдяки алгоритмам машинного навчання та глибокого навчання, компанії можуть автоматично класифікувати текст на позитивний, негативний або нейтральний. Семантичне розуміння

також дозволяє системам розпізнавати контекст слів і їх значення, що є критичним для точного аналізу.

3. Моделі та алгоритми NLP: у Big Data NLP використовує різноманітні моделі та алгоритми, такі як Word Embeddings (наприклад, Word2Vec, GloVe), рекурентні нейронні мережі (RNN) та трансформери (наприклад, BERT, GPT). Ці технології дозволяють моделювати та отримаувати більш точні результати при обробці природної мови, що особливо важливо для великих обсягів даних. Використання цих моделей дозволяє покращити результати в задачах, таких як машинний переклад, автоматичне резюмування текстів і чат-боти.

4. Виклики та можливості: незважаючи на численні переваги, реалізація NLP в Big Data стикається з певними викликами, такими як обробка неоднозначності мови, культурні та мовні варіації, а також необхідність у великих обсягах розмітки даних для навчання моделей. Однак можливості, які надає NLP, значно переважують ці виклики, надаючи нові шляхи для бізнес-аналізу, персоналізації послуг і покращення взаємодії зі споживачами.

Таким чином, Natural Language Processing у середовищі Big Data є потужним інструментом, що дозволяє підприємствам отримувати глибші інсайти з текстових даних, покращувати обслуговування клієнтів та адаптувати стратегії на основі реальних даних про поведінку споживачів. NLP стає наріжним каменем в світі, де дані набувають все більшого значення, сприяючи покращенню прийняття рішень і розвитку інноваційних рішень .

1.10.5 Метод кластеризації

Алгоритм кластеризації - це популярна техніка в аналітиці великих даних, яка передбачає групування схожих точок даних разом. Основна мета кластеризації - виявити значущі закономірності та структури в наборах даних, які можуть допомогти приймати кращі рішення та прогнози.

У великих даних кластеризація особливо корисна для обробки та аналізу великих обсягів даних, які є занадто складними та різноманітними, щоб їх можна було проаналізувати вручну. За допомогою алгоритмів кластеризації аналітики великих даних можуть автоматично виявляти подібності та відмінності між точками даних і групувати їх у кластери або підгрупи на основі певних критеріїв.

Існує кілька алгоритмів кластеризації: алгоритми розбиття, ієрархічні, на основі щільності та на основі сітки. Кожен тип алгоритму використовує свій підхід до групування точок даних на основі їхньої схожості.

Кластерний аналіз у Big Data — це потужний метод аналізу даних, який дозволяє групувати об'єкти або інстанції на основі їхніх подібностей. Цей метод є особливо корисним у контексті великих обсягів даних, де традиційні методи не завжди можуть впоратися із складною структурою та різнорідністю інформації. Ось ключові аспекти кластерного аналізу у Big Data:

1. Принципи кластерного аналізу: кластерний аналіз включає в себе пошук і виявлення груп (кластерів) в обсязі даних, де об'єкти в одній групі можуть бути більш схожими один на одного, ніж на об'єкти в інших групах. Це дозволяє визначити закономірності, тренди та

структуру в даних. Методи кластерного аналізу включають K-середніх, ієрархічний кластерний аналіз, DBSCAN (Density-Based Spatial Clustering of Applications with Noise) та інші.

2. Обробка великих обсягів даних: у Big Data, де кількість записів може досягати мільйонів чи навіть мільярдів, важливо обирати алгоритми, які можуть ефективно обробляти великі набори даних. Системи розподіленої обробки, такі як Apache Spark або Hadoop, можуть використовуватися для запуску кластерного аналізу на території великих інфраструктур, забезпечуючи паралельні обчислення та прискорене оброблення.

3. Вибір ознак: оптимізація результатів кластерного аналізу вимагає обрання релевантних ознак (факторів), які будуть використовуватися для класифікації. Для цього можуть застосовуватись методи зменшення розмірності, такі як Principal Component Analysis (PCA), які дозволяють скоротити кількість змінних, зберігаючи при цьому значущі характеристики даних. Це знижує навантаження на алгоритми кластеризації та покращує їх ефективність.

4. Візуалізація кластерів: успішний кластерний аналіз у Big Data також передбачає візуалізацію отриманих кластерів для кращого розуміння результатів. Інструменти візуалізації даних, такі як Tableau чи Matplotlib, можуть допомогти представити результати аналізу у зручному форматі, що полегшує сприйняття інформації та подальшу інтерпретацію.

5. Застосування кластерного аналізу: кластерний аналіз в умовах Big Data застосовується в багатьох сферах, таких як маркетинг (сегментація ринку), охорона здоров'я (групування пацієнтів за певними ознаками), фінансовий сектор (виявлення аномалій у

транзакціях), а також в науці (аналіз біологічних даних та геноміки). Цей метод дозволяє компаніям і організаціям виявляти нові тренди, оптимізувати стратегії та приймати обґрунтовані рішення [11].

Отже, кластерний аналіз у Big Data є важливим інструментом, що дозволяє витягувати цінну інформацію з великих обсягів різнорідних даних. Використовуючи цей метод, організації можуть глибше розуміти структуру своїх даних, виявляти приховані закономірності і здійснювати персоналізацію своїх послуг, що, в свою чергу, може покращити їх конкурентоспроможність на ринку.

1.10.6 Аналіз часових рядів

Аналіз часових рядів у Big Data є важливою галуззю, яка фокусується на вивченні даних, що змінюються з часом. Часові ряди складаються з наборів спостережень, отриманих у різні моменти часу, і їх аналіз допомагає виявити тренди, сезонність, циклічність та інші закономірності, які можуть бути критично важливими для прогнозування та прийняття рішень. Ефективний аналіз часових рядів у контексті Big Data включає такі ключові аспекти:

1. Збір та обробка даних: у Big Data аналіз часових рядів починається зі збору великих обсягів даних з різних джерел, таких як сенсори, соціальні мережі, системи моніторингу тощо. Часові мітки є важливими для розуміння хвиль коливань і впливу на результати аналізу. Обробка даних включає очищення, нормалізацію та агрегування даних для підготовки до моделювання. Це особливо

важливо в умовах великих обсягів даних, оскільки якість та точність даних безпосередньо впливають на результати аналізу.

2. Моделювання та прогнозування: основна мета аналізу часових рядів — це створення математичних моделей, які представляють дані та можуть бути використані для прогнозування майбутніх спостережень. Методики, як ARIMA (Автормодельна інтегрована середня), експоненціальне згладжування, а також сучасні глибокі навчальні моделі, такі як LSTM (Long Short-Term Memory) або GRU (Gated Recurrent Unit), використовуються для моделювання часових рядів. Класифікація моделей допомагає виявити не лише тренди, але й сезонні коливання, що покращує точність прогнозів.

3. Візуалізація та аналіз трендів: візуалізація є важливим компонентом аналізу часових рядів. Візуалізація даних може допомогти виявити характеристики часових рядів, такі як тренди, сезонність та аномалії. Інструменти, такі як Matplotlib, Seaborn або спеціалізовані платформи, дозволяють легко візуалізувати великі обсяги даних та робити аналіз більш зрозумілим. Дослідження візуальних представлень сприяє кращому розумінню складних взаємозв'язків у даних.

4. Аналіз аномалій: виявлення аномалій у часових рядах допомагає ідентифікувати неочікувані події або зміни, які можуть мати важливі наслідки. У рамках Big Data застосовуються алгоритми для визначення викидів і нестандартних коливань у даних. Це важливо для різних секторів, зокрема для моніторингу фінансових транзакцій, виявлення шахрайства, а також в системах управління промисловими процесами.

5. Застосування в різних сферах: аналіз часових рядів у Big Data знаходить широке застосування в багатьох галузях: фінансових ринках для прогнозування цін акцій, у енергетичному секторі для оцінки споживання електроенергії, у медичній сфері для моніторингу пацієнтів та прогнозування епідемій. Входження машинного навчання в цю галузь також відкриває нові можливості для підвищення точності прогнозування та автоматизації аналізу.

Отже, аналіз часових рядів у Big Data є важливим інструментом для організацій, що прагнуть отримати цінні інсайти з даних, які змінюються з часом. Цей аналіз дозволяє не лише виявляти тренди і циклічність, а й робити прогнози, що сприяють прийняттю обґрунтованих бізнес-рішень і підвищенню ефективності [12].

Висновок до розділу 1

У результаті дослідження сучасних методів аналізу даних у контексті великої кількості інформації, відзначено, що Big Data відкриває нові можливості для бізнесу та суспільства. Аналіз великих обсягів даних дозволяє організаціям отримувати цінну інформацію, виявляти закономірності та оптимізувати процеси. Використання таких методів, як машинне навчання, обробка природної мови та кластеризація, сприяє прийняттю обґрунтованих рішень та формуванню інноваційних стратегій.

Важливим є також усвідомлення викликів, пов'язаних із обробкою та аналізом великих даних, зокрема, щодо якості, безпеки та конфіденційності інформації. Успішна реалізація технологій Big Data вимагає не лише вдосконалених інструментів, але й підготовлених спеціалістів, здатних

працювати з комплексними алгоритмами та інтерпретувати результати. Отже, аналіз даних стає важливим фактором для отримання конкурентних переваг і вирішення глобальних проблем у різних галузях, таких як охорона здоров'я, фінанси та виробництво.

У світлі постійного зростання обсягів даних та швидкого розвитку технологій, важливо продовжувати розвивати нові методики та підходи до аналізу даних. Це дозволить організаціям не лише ефективно впоратися зі зростаючими обсягами інформації, але й зберегти інноваційний потенціал у сучасному конкурентному середовищі, в якому дані відіграють вирішальну роль у прийнятті рішень та формуванні стратегій розвитку.

РОЗДІЛ 2 МАТЕМАТИЧНІ МЕТОДИ ТА ВИДИ ТЕХНОЛОГІЙ В АРХІТЕКТРУІ ПРОЄКТІВ З ВЕЛИКИМИ ДАНИМИ

2.1. Методи обробки даних

Обробка даних — це критично важливий етап у процесі роботи з великими даними. З розвитком технологій та збільшенням обсягів даних, що генеруються щодня, виникла потреба у впровадженні ефективних методів для збору, обробки та аналізу інформації. Існує кілька основних методів обробки даних, кожен з яких має свої особливості, переваги та недоліки.

1 Пакетна обробка даних (Batch Processing)

Пакетна обробка даних є одним із найпоширеніших методів, що дозволяє обробляти великі обсяги даних за допомогою планових завдань. У цьому підході дані збираються за певний проміжок часу, а потім передаються на обробку. Це дозволяє обробляти дані ефективно, зберігаючи ресурси, і отримувати результати через визначений час. Приклади технологій, що використовують пакетну обробку, включають Apache Hadoop та Apache Spark.

Однак, незважаючи на ефективність, цей метод має свої обмеження. Затримка між збором і обробкою даних може бути важливою, якщо потрібно отримати термінову інформацію, наприклад, у фінансових ринках або у сфері безпеки.

2 Обробка в реальному часі (Stream Processing)

Обробка в реальному часі забезпечує можливість аналізувати дані, як тільки вони з’являються. Цей метод використовує безперервний потік даних, що надходять із різних джерел, таких як сенсори, соціальні мережі чи фінансові платформи. За допомогою технологій, таких як Apache Kafka та Apache Flink, дані можуть бути оброблені та проаналізовані на льоту, що дозволяє отримувати миттєву інформацію про події.

Цей підхід є надзвичайно важливим у ситуаціях, де час є критичним фактором. Наприклад, виявлення шахрайства у фінансових транзакціях або моніторинг здоров'я пацієнтів у реальному часі.

3 Сумішеві методи (Hybrid Processing)

Сумішеві методи поєднують у собі як пакетну, так і потокову обробку. Це дозволяє організаціям користуватися перевагами обох підходів: обробляти великі обсяги даних у пакетах для глибшого аналізу та одночасно реагувати на поточні події. Такі підходи стають популярними завдяки гнучкості та можливостям масштабування, що їх надають.

4 Обробка даних в базах даних (Database Processing)

Сучасні бази даних також пропонують потужні інструменти для обробки даних. Реляційні бази даних, такі як MySQL та PostgreSQL, пропонують можливість використовувати SQL для запитів та обробки структурованих даних, тоді як нереляційні бази (NoSQL), як-от MongoDB й Cassandra, надають гнучкість при роботі з неструктурованими даними. Вибір між реляційною та нереляційною базою даних залежить від типу даних, спектру запитів та вимог до масштабування.

2.2 Технології зберігання даних

Hadoop — це відкритий фреймворк для обробки та зберігання великих обсягів даних на розподілених обчислювальних системах. Інструкції до Hadoop дозволяють проектувати та реалізовувати програми для обробки даних у паралельному режимі, що робить його популярним інструментом у сфері обробки великих даних. Основні компоненти Hadoop включають:

1. Hadoop Distributed File System (HDFS): Це розподілена файловая система, що дозволяє зберігати великі обсяги даних на множині серверів. HDFS забезпечує надійність зберігання даних завдяки реплікації блоків даних.

2. MapReduce: Це модель програмування для обробки великих даних паралельно. MapReduce складається з двох основних етапів:

   – Map: обробка даних і перетворення їх на ключ-значення.

   – Reduce: об'єднання результатів з етапу Map.

3. Hadoop YARN (Yet Another Resource Negotiator): Це система управління ресурсами, що дозволяє ефективно розподіляти ресурси між різними завданнями, що виконуються в кластерах Hadoop.

4. Hadoop Common: Це набір спільних утиліт і бібліотек, необхідних для роботи інших компонентів Hadoop.

Hadoop Distributed File System (HDFS) – як було зазначено раніше, це розподілена система зберігання даних, призначена для управління

великими наборами даних на декількох машинах у кластері Hadoop. Вона забезпечує швидкий доступ до даних і корисна для обробки великих наборів даних в екосистемі Apache Hadoop[13].

HDFS пропонує економічно ефективне рішення для зберігання даних за рахунок використання стандартного обладнання та розподілених обчислень. Це дозволяє організаціям розширювати свої сховища без великих витрат, що робить його доступним вибором для управління великими обсягами даних.

Крім того, HDFS інтегрується з різними хмарними провайдерами та пропонує гнучкі варіанти ціноутворення. Це дозволяє організаціям оптимізувати витрати на зберігання даних відповідно до їхніх конкретних потреб. Наприклад, Amazon Web Services (AWS) надає HDFS як послугу через Amazon EMR, що дозволяє користувачам використовувати HDFS для зберігання та обробки даних у хмарній інфраструктурі AWS.NoSQL Databases: нереляційні бази даних, такі як MongoDB, Cassandra, CouchDB, які забезпечують гнучкі моделі даних.

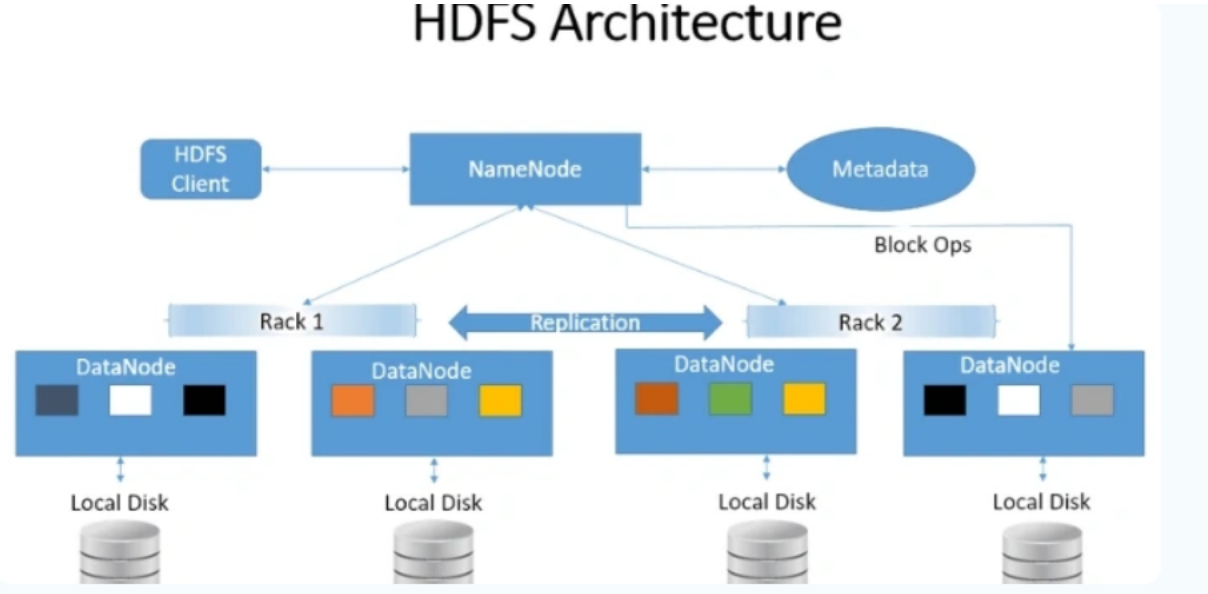

Рисунок 2.1 – HDFS архітектура

HDFS, в основному, реалізована на Java, роблячи цю мову ключовою для її архітектури та функціональності. Java забезпечує основу для основних компонентів HDFS, таких як NameNode, DataNode та різні клієнтські інтерфейси. Проте Python також може бути використаний у поєднанні з HDFS через бібліотеки та фреймворки, такі як PySpark, який надає API на Python для взаємодії з HDFS. Це дозволяє користувачам виконувати завдання обробки, аналізу та машинного навчання даних HDFS з використанням Python.

Архітектура HDFS заснована на принципі «ведучий-ведений» і складається з кількох ключових елементів таких як: NameNode - компонент є центральним вузлом, що керує метаданими в HDFS. NameNode зберігає важливу інформацію про структуру каталогів, імена файлів та їх атрибути. Він відстежує розташування блоків даних, що зберігаються на DataNode, і управляє операціями з файлами, такими як читання, запис і реплікація.

В своб чергу у системі NameNode існує два типи файлів:

FsImage: Ці файли виконують роль системи зберігання, містячи повну та організовану інформацію про структуру файлової системи. Вони створюють повний знімок ієрархічної структури HDFS.

EditLogs: Ці файли записують усі зміни, що відбуваються з файловою системою, ведучи журнал змін і оновлень, які відбулися протягом визначеного часу.

У контексті HDFS ZooKeeper виконує важливу роль у підтримці стану та інформації про метадані кластера Hadoop, надаючи надійні та

розподілені координаційні послуги для забезпечення високої доступності та узгодженості таких критично важливих компонентів, як NameNode та DataNode.

DataNode - зберігає ці вузли фактичні блоки даних і отримує інструкції на виконання операцій від NameNode. Вони забезпечують відмовостійкість завдяки реплікації даних. DataNode регулярно надсилають сигнали (серцебиття) кожні три секунди на NameNode для підтвердження своєї працездатності. У разі збою DataNode NameNode ініціює реплікацію і призначає нові блоки іншим DataNode для збереження надмірності даних.

Файлові блоки в HDFS: Дані в HDFS поділяються на блоки для оптимізації зберігання та доступу. За замовчуванням розмір кожного блоку складає 128 МБ, проте ця величина може бути змінена. Наприклад, файл розміром 550 МБ буде поділений на п’ять блоків: чотири з них по 128 МБ і один на 38 МБ.

Вторинний NameNode - це допоміжний вузол, що виконує періодичну перевірку простору імен. Secondary NameNode допомагає контролювати розмір лог-файлу модифікацій HDFS, що сприяє швидшому відновленню NameNode у разі збоїв.

Управління метаданими: при запуску NameNode завантажує інформацію про блоки в пам’ять, оновлює метадані за допомогою журналу редагувань та створює контрольні точки. Розмір метаданих обмежений обсягом оперативної пам’яті NameNode.

2.3 Apache Cassandra

Apache Cassandra — це розподілена, нереляційна база даних, спеціально спроектована для обробки великих обсягів даних з високими вимогами до доступності та продуктивності. Основні принципи роботи з Apache Cassandra включають:

1 Архітектура та зберігання даних

Cassandra використовує архітектуру «masterless», що означає, що всі вузли в кластері є рівноправними та можуть обробляти запити, за рахунок чого досягається висока доступність. Кожен вузол не лише зберігає частину даних, але й бере участь у виконанні всіх запитів, що дозволяє зменшити навантаження на окремі вузли.

Дані в Cassandra зберігаються у форматі таблиць, які відрізняються від традиційних SQL таблиць тим, що не вимагають фіксованої схеми. Користувачі можуть визначати широкий спектр колонок для кожної таблиці, що робить її дуже гнучкою для різних типів даних. Крім того, Cassandra застосовує концепцію «column family», де дані організовані за стовпцями, а не рядками, що дозволяє покращити ефективність запитів.

2 Вставка та читання даних

Дані в Cassandra можуть бути вставлені за допомогою CQL (Cassandra Query Language), який нагадує SQL, але спроектований для роботи в розподіленому середовищі. Запити на вставку (INSERT) дозволяють швидко додавати дані в кластер, які автоматично реплікуються між вузлами для забезпечення надійності та відмовостійкості.

Коли вам потрібно прочитати дані, Cassandra використовує індекси для швидкого доступу до необхідних записів. Завдяки своїй розподіленій архітектурі база даних може обробляти запити одночасно на кількох вузлах, що забезпечує високу швидкість виконання запитів.

3 Реплікація та погодженість

Однією з ключових особливостей Cassandra є її механізм реплікації. Користувачі можуть налаштувати, скільки копій даних буде зберігатися в кластері (фактор реплікації), що дозволяє забезпечити високу доступність. Cassandra підтримує кілька стратегій реплікації, таких як «SimpleStrategy» та «NetworkTopologyStrategy», що дозволяє адаптуватися до різних потреб інфраструктури.

Також важливим аспектом є налаштування рівня погодженості (consistency level) для запитів, що визначає, скільки реплік повинні підтвердити обробку запиту перед його завершенням. Це дозволяє балансувати між швидкістю виконання та надійністю даних.

4 Масштабування

Cassandra спроектована для горизонтального масштабування. Це означає, що підприємства можуть легко додавати нові вузли до кластера без зупинки роботи системи. При додаванні нового вузла дані автоматично перерозподіляються, що забезпечує плавну інтеграцію і підтримку високої продуктивності.

Коротко кажучи, Apache Cassandra є ідеальним рішенням для підприємств, які потребують надійного, масштабованого та високопродуктивного зберігання даних, враховуючи великий обсяг і динамічний характер сучасних даних.

2.4 MongoDB

MongoDB — це документоорієнтована NoSQL база даних, спеціально спроектована для обробки великих обсягів даних, що робить її відмінним вибором для застосувань у сфері великих даних. Основними особливостями MongoDB, які забезпечують її ефективність у роботі з великими даними, є:

1. Документоорієнтоване зберігання даних

   MongoDB зберігає дані у форматі BSON (Binary JSON), що дозволяє зберігати не лише прості ключ-значення, а й складні об'єкти та масиви. Кожен документ у MongoDB є самостійною одиницею даних, що надає велику гнучкість у структурі. Це дозволяє розробникам зберігати різноманітні типи інформації без необхідності попередньо визначати схему, що особливо корисно в середовищах з великими об'ємами даних і часто змінюваними вимогами.

2. Масштабованість

   MongoDB розроблена з акцентом на горизонтальне масштабування, що означає, що вона може безперешкодно додавати нові вузли до кластеру. Завдяки тому, що MongoDB підтримує шардинг (sharding), дані можуть бути автоматично розподілені між кількома серверами. Це забезпечує високу пропускну здатність і обробку запитів у реальному часі, дозволяючи обробляти великий об'єм даних без втрат продуктивності.

3. Висока доступність

MongoDB підтримує механізм реплікації, що дозволяє створювати репліки даних на різних вузлах кластера. Це забезпечує високу доступність, оскільки в разі збою одного з вузлів, запити можуть бути перенаправлені на інші активні репліки без зупинки роботи системи. Конфігурації реплікації зазвичай включають основний (primary) вузол, який управляє записами, і кілька вторинних (secondary) вузлів, які копіюють дані для підтримки резервних копій.

4. Гнучкі запити та індексація

   MongoDB надає можливість виконувати складні запити, фільтрацію і агрегації даних. Це досягається завдяки потужній системі запитів, яка дозволяє виконувати різноманітні операції з даними, включаючи фільтрацію, обчислення агрегацій і обробку підзапитів. MongoDB підтримує також індексацію, що підвищує швидкість доступу до даних, що є важливим аспектом при роботі з великими обсягах інформації.

5. Інтеграція з великими даними

   MongoDB легко інтегрується з різними фреймворками для обробки великих даних, такими як Apache Spark і Hadoop. Це дає змогу користувачам поєднувати можливості зберігання та масштабування MongoDB з потужністю обробки даних цих фреймворків, що забезпечує великий потенціал для аналітики та обробки даних.

2.5 NoSQL бази даних та їх переваги

Однією з таких технологій є NoSQL (Not Only SQL) бази даних, які забезпечують альтернативний підхід до зберігання неструктурованих або

напівструктурованих великих даних. На відміну від традиційних реляційних баз даних, що використовують фіксовану схему, NoSQL бази дозволяють зберігати дані без попередньо визначених таблиць. NoSQL бази даних спроектовані для високої масштабованості, що дозволяє їм без проблем справлятися з величезними обсягами даних.

Крім того, такі бази надають гнучкі моделі зберігання, що дозволяють зберігати різні типи інформації у форматах, таких як документи JSON, пари ключ-значення або колонки. Ще однією перевагою NoSQL баз є їхня висока доступність завдяки розподіленій архітектурі, що дозволяє підтримувати працездатність навіть під час апаратних збоїв шляхом реплікації та розподілу копій даних між серверами [14].

NoSQL бази даних надають суттєві переваги в порівнянні з традиційними реляційними базами, коли мова йде про ефективне управління великими обсягами неструктурованих або напівструктурованих даних. У наступних розділах ми розглянемо різні типи NoSQL баз даних, їх використання та найкращі практики для впровадження в організації.

Перевагами використання NoSQL баз даних для зберігання великих даних є :

- масштабованість: здатність легко обробляти великі обсяги даних
  Однією з найзначніших переваг NoSQL баз для зберігання великих даних є їхня масштабованість. Традиційні реляційні бази даних призначені для роботи на одиночних серверах, що може викликати проблеми з продуктивністю при обробці великих обсягів даних. Натомість NoSQL бази даних спроектовані для горизонтального масштабування на декількох серверах, що дозволяє їм легко справлятися з величезними обсягами даних.

Ця здатність до горизонтального масштабування робить NoSQL бази ідеальними для застосувань, де потрібно зберігати і обробляти великі обсяги даних, таких як соціальні мережі та електронна комерція. Дозволяючи легко масштабувати, бізнес може адаптувати свою базу даних відповідно до змінних потреб.

- гнучкість: можливість зберігати дані в різних форматах без попередньо визначеної схеми NoSQL бази даних пропонують набагато більше гнучкості, ніж традиційні реляційні бази. На відміну від реляційних баз, що вимагають визначення схеми до зберігання даних, NoSQL бази дозволяють

6. Аналіз даних та машинне навчання

У сучасному світі обробка та аналіз великих обсягів даних, відомих як «біг дата», стали одними з ключових аспектів у розвитку бізнесу, науки і технологій. Машинне навчання (МН) виступає важливим інструментом, що дозволяє ефективно навчатися на основі величезних обсягів інформації та робити прогнози і висновки без необхідності ручного програмування.

Першим етапом у реалізації машинного навчання є збір і зберігання даних. Джерела даних можуть варіюватися від соціальних мереж до пристроїв Інтернету речей (IoT), що генерують великі обсяги інформації в реальному часі. Складність подання таких даних вимагає використання різних баз даних. Для структурованих даних використовують реляційні бази, такі як PostgreSQL, в той час як неструктуровані дані часто зберігаються в NoSQL-базах, таких як MongoDB. Важливо забезпечити організовану архітектуру даних для подальшої їхньої обробки[15].

Наступним кроком є передобробка даних, що включає в себе очищення, обробку пропусків, нормалізацію та кодування змінних. Наприклад, пропущені значення можуть бути заповнені середнім значенням:

$$\left[X_{filed} = \left\{X_i, якщо X_i непропущене X якщо X_i пропущене\right\}\right] (2.1)$$

де: $(\tilde{x})$— середнє значення елементів в (X).

Потім дані часто нормалізуються для приведення всіх змінних до єдиного масштабу:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} (2.2)$$

Цей процес критично важливий, оскільки багато алгоритмів чутливі до масштабу даних. Наприклад, у лінійній регресії, яка моделює відношення між залежною та незалежними змінними:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \text{e} (2.3)$$

Наступним етапом є вибір алгоритму навчання. Для цього часто використовують функцію втрат, таку як середня квадратична помилка (MSE):

$$\left[MSE = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2 \right] (2.4)$$

Ця функція допомагає оцінити точність моделі, і чим менше значення MSE, тим краще модель передбачає результати. Для класифікаційних задач часто використовують крос-ентропію:

$$\left[L\left(y, \hat{y} \right) = - \sum_{i=1}^{C} y_i \log(\hat{y}_i) \right] (2.5)$$

Ця метрика допомагає налаштувати моделі, зменшуючи відстань між реальними класами та прогнозованими ймовірностями.

Процес тренування моделі зазвичай реалізується за допомогою алгоритму градієнтного спуску:

$$\omega := \omega - \eta \nabla L(\omega) (2.6)$$

де: (η) — швидкість навчання

$(\nabla L(\omega))$ — градієнт функції втрат.

Цей процес повторюється до досягнення мінімального значення функції втрат.

Регуляризація важлива для запобігання перенавчанню моделей. Наприклад, використання L2-регуляризації:

$$L_{total} = L_{loss} + \lambda \sum_{j=1}^{n} \omega_y^2 (2.7)$$

дозволяє контролювати величину штрафу за великі значення ваг. Це важливо для збереження простоти моделі та покращення її узагальнюючих властивостей.

Класифікаційні моделі, такі як логістична регресія, можуть мати вигляд:

$$P\left(y = 1 \middle| X\right) = \frac{1}{1 + e - \left(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n\right)} (2.8)$$

Призначення: Використовується для прогнозування ймовірності належності спостереження до певного класу. Це має особливе значення в бінарних класифікаційних задачах, де потрібно визначити, чи належить об'єкт до категорії «1» або «0».

Оцінка продуктивності моделі класифікації не була б повною без використання коефіцієнта кореляції Пірсона, який вимірює силу та напрямок лінійного зв’язку між двома змінними:

$$r = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2 \sum_{i=1}^{n}(y_i - \overline{y})^2}} (2.9)$$

Призначення: Це допомагає виявити, чи є значні зв'язки між ознаками, що може підказати, які змінні варто використовувати в моделях.

Вимірювання відстані між даними також є важливим аспектом багатьох алгоритмів машинного навчання, такою як класифікація k-найближчих сусідів (k-NN). Для цього використовується евклідова відстань:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} (2.10)$$

Призначення: Ця формула дозволяє вимірювати схожість між точками в n-вимірному просторі, що є критично важливим для визначення «сусідів» в алгоритмах класифікації.

У підсумку, машини можуть краще навчатися і адаптуватися до великого обсягу даних, якщо вони розуміють, які алгоритми використовувати, які функції втрат оцінювати і як оптимізувати свої моделі. Весь процес, від збору та обробки даних до тренування та валідації моделей, грунтується на основних математичних принципах, формули яких надають фахівцям у галузі даних інструменти для розв'язання складних завдань [16].

Зараз машинне навчання в умовах великих даних не лише є потужним інструментом для бізнесу, науки та технологій, але й стає основою для майбутнього розвитку IT-інфраструктур, аналітики та штучного інтелекту. Якісне економічне зростання, нові технології, що вдосконалюють повсякденне життя людей, а також інноваційні рішення в різних сферах — усе це можливе завдяки здатності машинного навчання обробляти та аналізувати величезні обсяги даних.

Цей процес шумного оброблення даних, змінного навчання, відповідності алгоритмів і багато іншого, незважаючи на всі виклики, є невід'ємною частиною нашої цифрової реальності сьогодні. Машинне навчання в умовах великих даних — це не просто тренд; це потужний рушій змін, який формує наше бачення світу.

Таким чином, врахування цих формул та концепцій дозволяє нам не лише зрозуміти процеси, які стоять за машинним навчанням, але й ефективно їх застосовувати для аналізу й отримання цінних інсайтів на основі даних у різних сферах діяльності.

7. Регресійні алгоритми при роботі з біг датою

В епоху великих даних (біг дата) регресійні алгоритми виявилися надзвичайно потужними інструментами для прогнозування кількісних змінних на основі наявних даних. Вони застосовуються в різних галузях, включаючи фінанси, охорону здоров'я, маркетинг та інші, для виявлення закономірностей та роблення передбачень. Розглянемо основні регресійні алгоритми, їх формули та застосування в контексті великих даних.

7.1. Лінійна регресія

Лінійна регресія є одним з найпростіших і найпопулярніших методів регресії. Вона базується на припущенні, що між залежною змінною (Y) і незалежними змінними $(X_1, X_2, \ldots, X_n)$ існує лінійний зв'язок. Основна формула для простого випадку (одна незалежна змінна) виглядає так:

$$Y = \beta_0 + \beta_1 X_1 + \mathrm{e}(2.11)$$

де: (Y) — залежна змінна,

(X) — незалежна змінна,

(β_0) — вільний член (інтерсепт),

(β_1) — коефіцієнт регресії,

(e) — помилка.

Мета лінійної регресії полягає у знаходженні найкращих значень (β_0) та (β_1), які мінімізують суму квадратичних відхилень між прогнозованими і фактичними значеннями. Цей процес зазвичай реалізується за допомогою методу найменших квадратів.

7.2. Поліноміальна регресія

Поліноміальна регресія є розширенням лінійної регресії, коли зв'язок між змінними не є лінійним. У такому випадку, модель може бути записана як:

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \ldots + \beta_n X^n + e (2.12)$$

Цей тип регресії дозволяє навчити модель складнішим залежностям та зменшує похибки в прогнозах.

Регресія з регуляризацією

У великих даних часто виникає проблема надмірної налаштованості (overfitting), коли модель занадто складна й пристосована до шуму в даних.

Для вирішення цієї проблеми використовуються регресійні алгоритми з регуляризацією, такі як Lasso і Ridge регресії.

1. Ridge регресія:

$$Minimize, \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 + \lambda \sum_{y=1}^{n} \beta_y^2 (2.13)$$

де: (λ) — параметр регуляризації.

2. Lasso регресія:

$$Minimize, \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 + \lambda \sum_{y=1}^{n} \left|\beta_j\right| (2.14)$$

Lasso регресія не лише зменшує значення коефіцієнтів, а й може повністю їх усувати, що полегшує вибір моделей, зменшуючи складність [17].

7.3. Застосування в біг дата

Регресійні алгоритми в біг дата використовуються для багатьох практичних задач, таких як:

- прогнозування продажів: на основі історичних даних компаній аналітики можуть використовувати регресію для прогнозування майбутніх продажів;
- оцінка ризиків у фінансах: у фінансових установах регресійні моделі допомагають у визначенні кредитного ризику;
- управління запасами: виробничі компанії можуть застосовувати регресію для оптимізації запасів на основі прогнозів попиту.

Реалізація регресійних алгоритмів у контексті біг дата є складним процесом, що включає кілька етапів — від збору та обробки даних до навчання моделей та їх впровадження;

2.8 NLP

Обробка природної мови (NLP) є однією з найцікавіших та найскладніших галузей штучного інтелекту. Основним викликом у цій сфері є перетворення тексту в числові формати, які можуть бути зрозумілі комп'ютерами. Протягом багатьох років дослідники шукали нові способи отримання векторних представлень слів, що дозволяє враховувати не лише саму частоту вжитку слів, а й їхній семантичний контекст.

Одним із найперших підходів до векторизації слів було one-hot кодування, яке передбачало, що кожне слово представляється унікальним

бінарним вектором. Проте ця методологія має істотні недоліки — високі вимірності векторів ускладнюють моделювання, а також відсутність семантичного контексту призводить до того, що синоніми й несумісні слова мають однакові відстані у векторному просторі.

Прорив у цій галузі стався у 2013 році з появою Word2Vec, двошарової нейронної мережі, яка здатна робити точні прогнози значення слова на основі його контексту. У Word2Vec використовуються два методи: Continuous Bag of Words (CBOW) і Skip-gram. CBOW передбачає цільове слово, використовуючи слова, що оточують його, тоді як метод Skip-gram використовує одне слово для прогнозування контексту. Таким чином, Word2Vec дозволяє створити векторні представлення, в яких схожі слова розташовані близько один до одного, що значно покращує семантичне розуміння.

Через рік, у 2014-му, з'явився GloVe (Global Vectors for Word Representation), який зробив новий крок уперед у векторизації слів. GloVe базується на глобальних статистичних даних щодо співпадінь слів, що дозволяє картографувати слова в такий спосіб, що відстань між ними відображає частоту їхньої спільної появи в текстах. Цей підхід акцентує увагу на семантичній близькості слів, що робить GloVe потужним інструментом для різноманітних завдань NLP.

Хоча Word2Vec і GloVe добре справляються з векторним поданням слів, більшість даних у природній мові представлені у формі послідовностей. Тут на допомогу приходять Рекурентні Нейронні Мережі (RNN), які спроектовані для роботи з послідовними даними, обробляючи слова при порядку та зберігаючи інформацію про попередні. Однак стандартні RNN мають труднощі з пам'яттю на довгих послідовностях, оскільки стан прихованої пам'яті постійно переписується. Для подолання

цього обмеження були розроблені моделі Long Short-Term Memory (LSTM), які використовують спеціалізовані механізми для підтримки довгострокових залежностей. Це дозволяє ефективно навчати модель з урахуванням контексту навіть за великих обсягів даних[18].

Ще один значний крок уперед – це застосування Згорткових Нейронних Мереж (CNN) у NLP. Вони довели свою ефективність у комп'ютерному зорі, і дослідження показали, що їх можна адаптувати для обробки тексту, використовуючи 1D фільтри для аналізу послідовностей слів. CNN можуть аналізувати текст швидше й точніше, ніж традиційні RNN, демонструючи переваги в ресурсній економії та загальній продуктивності.

У той час як нейронні мережі виробили значний прогрес в моделях NLP, вони все ще стикаються з складними завданнями, такими як аналіз настроїв, машинний переклад, парафразування тексту та відповіді на запитання. Наприклад, машинний переклад вимагає моделі, яка могла б правильно перед бачати послідовності слів у різних мовах, враховуючи їхні мовні нюанси і культурні особливості. Це завдання є суттєво складнішим, ніж просто виявлення тональності або класифікація документів, оскільки необхідно враховувати не лише лексичні, а й синтаксичні аспекти.

Для подолання цих труднощів у перекладі часто використовують механізми уваги. Ці механізми дозволяють моделі фокусуватися на найбільш релевантних частинах тексту, що особливо корисно в задачах, де важливо враховувати контекст запитання. Зокрема, моделі з механізмами уваги демонструють значні покращення у завданнях, пов'язаних з запитаннями та відповідями, оскільки вони здатні виявляти важливі фрагменти тексту, які містять необхідну інформацію для відповіді.

Але незважаючи на всі ці досягнення, NLP все ще стикається з численними викликами. Серед них — обробка ідіом, іронії та сарказму, що є звичною справою для людей, але часто нездоланною перешкодою для машин. Наприклад, виявлення емоцій у тексті залишається проблематичним, оскільки глибокі нейронні мережі можуть недостатньо точно передбачати вікно настроїв через контекстуальні нюанси.

Крім того, генерація тексту, або, як її ще називають, автозавершення, відкриває нові можливості у маркетингу та персоналізованому обслуговуванні клієнтів. Рекурентні нейронні мережі стверджують свою ефективність у виділенні текстів на рівні символів, що означає, що алгоритми можуть генерувати послідовності символів, передбачаючи наступні символи навіть без концептуального розуміння слів. Проте, генерація звуку залишається складним завданням, оскільки якісне відтворення мови вимагає більшої кількості вхідних даних для моделювання.

Нещодавні досягнення в цій галузі, такі як WaveNet, розроблені компанією DeepMind, змінюють погляд на генерацію аудіо. WaveNet здатний створювати якісне синтезоване мовлення та навіть музику, оскільки він обробляє сигнал, генеруючи його по одному семплу за раз. Це відрізняється від традиційних текстово-звукових систем, які значно поступаються за якістю і точністю.

У підсумку, розвитку NLP демонструє потужний прогрес, починаючи з таких базових методів, як Word2Vec, і переходячи до складних архітектур нейронних мереж, які здатні розуміти та обробляти природну мову на новому рівні. Компанії, такі як DevsData, що спеціалізуються у цій сфері, постійно працюють над поліпшенням алгоритмів і моделей, щоб подолати ще більше бар'єрів у розумінні мови. Ця еволюція не лише підвищує

ефективність обробки мови, а й відкриває нові горизонти для комерційних застосувань, дозволяючи автоматизувати і вдосконалювати безліч процесів у різних індустріях. Таким чином, майбутнє NLP виглядає обнадійливо, з безліччю нових можливостей для розвитку й оптимізації численних галузей.

2.9 Apache Flink та Apache Spark

У сучасному світі, що характеризується стрімким зростанням обсягів даних, здатність ефективно обробляти ці дані стає критично важливою для бізнесу й інновацій. Два найпопулярніші відкриті фреймворки для розподіленої обробки даних — Apache Flink і Apache Spark — пропонують потужні рішення для аналізу великих даних. Хоча обидва фреймворки зосереджені на обробці даних, вони мають свої унікальні особливості, які роблять їх підходящими для різних випадків використання, особливо в контексті потокової обробки.

1 Основи фреймворків

Apache Spark був спочатку розроблений як пакетна система, але згодом додав можливості для потокової обробки через Spark Structured Streaming. Він пропонує користувачам інтуїтивний досвід завдяки високорівневим API, які підтримують різноманітні мови програмування, такі як Scala, Java, Python та R. Spark забезпечує прості синтаксичні конструкції для виконання популярних задач, таких як агрегація, віконне обчислення та злиття.

На відміну від Spark, Apache Flink спочатку був спроектований для потокової обробки, що робить його чудовим вибором для задач, що вимагають низькочасової обробки даних. Flink підтримує три основні

рівні API: API обробки станів, DataStream API та API Table і SQL. Це дозволяє користувачам більш гнучко підходити до задач, що стосуються стану та часу, забезпечуючи більший контроль над цими критичними аспектами.

2 Потокова обробка

Обробка потоків даних є однією з ключових відмінностей між Flink і Spark. Flink спеціалізується на обробці даних в реальному часі, що робить його ідеальним рішенням для застосувань, де швидкість та миттєва реакція є критичними, таких як фінансові сервіси або моніторинг даних сенсорів. Flink має вбудовані механізми для управління станами та часовими вікнами, що дозволяє безперервно аналізувати дані, які надходять в реальному часі.

У той же час Spark, хоча й може обробляти потоки даних, має деякі обмеження в обробці низькочасових завдань. Його Structured Streaming API, хоча і функціональний, часто сприймається як менш інтуїтивний у порівнянні з Flink. Spark зберігає раніше оброблені дані й дозволяє виконувати обчислення на поверхні, що може бути досить ефективним для пакетних задач, але не завжди оптимальним для потокового аналізу.

3 Вибір між Flink і Spark

Вибір між Flink і Spark повинен базуватися на специфікаціях проекту та вимогах до обробки даних. Якщо ваша задача передбачає обробку даних в реальному часі з низькою затримкою, Flink, з його значними можливостями в цій області, може бути кращим вибором. З іншого боку, Spark може стати ідеальним рішенням для завдань, що

вимагають потужних засобів пакетної обробки з простими в інтеграції API.

Обидва фреймворки також пропонують можливості для об’єднання з різними джерелами даних, такими як Kafka, HDFS та NoSQL бази даних, що підвищує їхню універсальність у великих екосистемах даних.

Apache Flink і Apache Spark — це потужні інструменти для обробки даних, кожен зі своїми унікальними перевагами. Правильний вибір між цими фреймворками залежить від конкретних вимог вашого проекту. Розуміння основних особливостей, які відрізняють Spark і Flink, дозволить вам прийняти обґрунтоване рішення, що допоможе вашій організації максимально еф ективно використовувати ресурси та отримувати цінні інсайти з даних.

Крім того, важливо враховувати не лише технічні параметри, а й фактори, пов’язані з командою розробників. Наприклад, якщо ваша команда має більше досвіду роботи з Scala або Python, то Spark може стати більш зрозумілим і зручним вибором. В той же час, якщо у вашій компанії активно впроваджують мікросервіси та контейнери, Flink завдяки своїй легкій інтеграції в архітектури зберігання даних у реальному часі може бути більш привабливим.

4 Інструменти та спільнота

Обидва фреймворки супроводжуються активними спільнотами та множиною ресурсів для навчання. Це включає документацію, семінари, вебінари та численні відкриті джерела коду. Spark вже давно мав широке поширення та популярність, що призвело до великої кількості бібліотек і розширень, які можуть спростити

розробку. Flink, хоч і менш популярний у деяких контекстах, також має зростаючу спільноту, яка пропонує підтримку та добре документовані ресурси.

5 Майбутнє потокової обробки

З розвитком технологій потужності і рівень складності систем обробки даних зростають. Вимоги до обробки даних в реальному часі стають дедалі важливішими в таких сферах, як фінанси, медицина, інтернет речей та обробка великих обсягів транзакцій. Як Flink, так і Spark постійно оновлюються, щоб відповідати цим вимогам, впроваджуючи нові функції та покращуючи вже існуючі.

В обслуговуванні складних аналітичних задач у хмарах, Flink із його підтримкою розподіленої обробки у реальному часі та з можливістю створення комплексних графів обчислень може надати перевагу. Spark, в свою чергу, буде продовжувати користуватися популярністю завдяки своїй простоті у використанні та можливостям для пакетної обробки.

Таким чином, вибір між Apache Flink і Apache Spark залежить від цілей вашого проекту, наявних ресурсів і досвіду команди. Обидва фреймворки мають свої сильні сторони та сценарії використання, і, зрештою, найкращий підхід може полягати в їхньому комбінуванні залежно від вимог до даних. З зрозумінням їхніх відмінностей та можливостей, ви зможете прийняти більш обґрунтоване рішення, яке сприятиме досягненню успіху у вашій стратегії обробки даних.

Обираючи між Flink і Spark, важливо також залишатися гнучкими, виконуючи регулярні оцінки технологій, оскільки індустрія продовжує еволюціонувати, впроваджуючи нові інструменти та

методики для обробки даних. Успіх у обробці великих даних лежить не лише в правильних технологіях, а й в умінні адаптуватися до змінюваного середовища даних і бізнес-потреб [19].

2.10 Хмарні рішення

Хмарні рішення для обробки великих даних (біг дата) є важливими інструментами для сучасних компаній, які потребують ефективних і масштабованих способів управління та аналізу величезних обсягів інформації. Розглянемо деякі з найбільш популярних рішень детальніше.

1 Amazon Web Services (AWS)

Amazon S3 (Simple Storage Service) — це сервіс для зберігання даних, який дозволяє зберігати різноманітні типи інформації, включаючи структуровані, напівструктуровані та неструктуровані дані. S3 особливо корисний для резервного копіювання даних і архівування, а також служить базою для аналітичних запитів. Завдяки своїй високій доступності та надійності, S3 широко використовується як засіб зберігання для даних, які використовуються в аналітичних задачах.

2 Amazon EMR (Elastic MapReduce) — це хмарний сервіс, який дозволяє обробляти великі обсяги даних, використовуючи фреймворки, такі як Apache Hadoop та Apache Spark. EMR надає можливість виконання обчислювальних задач, таких як машинне навчання, обробка даних для аналітики та ETL-процеси, у простий та масштабований спосіб. Цей сервіс дозволяє швидко розгортати кластери, забезпечуючи при цьому гнучкість у виборі типу обробки даних.

3 Amazon Redshift — це аналітичний сервіс, який дозволяє виконувати складні SQL-запити на великих обсягах даних. Redshift вважається відмінним рішенням для бізнес-аналітики, завдяки своїй здатності швидко обробляти та аналізувати дані. Підприємства часто використовують Redshift для підготовки звітів і проведення аналітики в режимі реального часу.

4 Google Cloud Platform

У Google Cloud Platform одним із найбільш потужних інструментів є Google BigQuery. Це сервіс для аналітики, який дозволяє швидко обробляти великі набори даних без необхідності управління інфраструктурою. BigQuery підходить для бізнес-звітування, аналітики в реальному часі та використання машинного навчання. Завдяки своїй простоті у використанні, користувачі можуть зосередитися на аналізі даних, не переймаючись про технічні деталі.

5 Google Cloud Storage — це сервіс, що спеціалізується на зберіганні даних. Він дозволяє зберігати як оброблені, так і необроблені дані, включно з резервними копіями та медіа-файлами. Це зручно для підприємств, які бажають мати централізоване місце для зберігання різної інформації.

6 Google Dataflow — це сервіс для обробки даних, який дозволяє виконувати як пакетну, так і потокову обробку. Dataflow уможливлює проведення ETL-процесів та обробку даних у реальному часі, що робить його ідеальним для сценаріїв, де потрібно своєчасно обробляти та аналізувати інформацію.

7 Microsoft Azure

У світі рішень від Microsoft Azure, Azure Data Lake є потужним інструментом для зберігання великих обсягів неструктурованих даних. Цей сервіс підходить для кампаній, які працюють з даними у форматі, який часто змінюється, і дає можливість зберігати дані в первісному вигляді для подальшої обробки та аналізу.

8 Azure HDInsight забезпечує хмарну платформу для обробки великих даних з використанням популярних фреймворків, таких як Hadoop, Spark або Hive. Цей сервіс підходить для аналізу даних, виконання розподілених обчислень і ведення потужних аналітичних проектів у різних сферах.

9 Azure Synapse Analytics — це комплексне рішення, яке поєднує зберігання даних і аналітику. Платформа дозволяє інтегрувати дані з різних джерел, проводити аналі тику в реальному часі та управляти даними в рамках одного рішення. Це особливо вигідно для бізнесів, які потребують швидкого доступу до даних та можливості їх аналізу у взаємодії. Azure Synapse аналізує дані, що безпосередньо впливає на рішення, які приймаються на основі аналітики, надаючи користувачам можливість створювати дашборди та звіти [20].

10 Databricks

Databricks є платформою, побудованою на базі Apache Spark, яка надає можливість легкого доступу до аналітики великих даних. Ця хмарна платформа забезпечує середовище для колаборації між аналітиками, науковцями у сфері даних та розробниками. За допомогою Databricks користувачі можуть без зусиль створювати аналітику даних, автоматизувати ETL-процеси та розгортати моделі машинного навчання. Платформа також підтримує інтеграцію з

різними інструментами для візуалізації даних, що робить її особливо корисною для бізнес-інтелекту й глибокого аналізу [21].

Обрання хмарного рішення для обробки великих даних значною мірою залежить від специфічних потреб бізнесу, таких як обсяги даних, типи аналітичних задач, бюджету і наявних ресурсів. Кожне з описаних рішень має свої особливості, переваги та ідеальні сценарії застосування. Важливо оцінити бізнес-процеси та вимоги до даних, перш ніж обрати платформу, яка найкраще відповідатиме потребам вашої організації та спростить роботу з великими даними.

Завдяки істотному зростанню обсягу даних, яке спостерігається в сучасному бізнес-середовищі, ефективне використання цих платформ стає важливим чинником конкурентоспроможності. Використовуючи хмарні рішення, підприємства можуть не лише зекономити на витратах на інфраструктуру, але й забезпечити швидкий доступ до даних та можливість масштабування своїх аналітичних потужностей у міру зростання потреб.

Висновки до розділу 2

У цьому розділі ми розглянули ключові аспекти обробки даних, технологій зберігання, машинного навчання, аналізу даних та реалізації рішень для великих даних (біг дата). Основні висновки можна сформулювати наступним чином:

1. Методи обробки даних:

   Різні методи обробки даних, такі як пакетна обробка, обробка в реальному часі та сумішеві методи, надають організаціям можливість

вибирати оптимальні рішення в залежності від специфіки та потреб аналізу даних.

Використання сучасних технологій, таких як Apache Hadoop, Apache Spark та NoSQL бази даних (MongoDB, Cassandra), забезпечує ефективне зберігання та доступ до великих обсягів даних.

2. Технології зберігання даних:

Системи, такі як HDFS, пропонують надійні та економічні рішення для масштабованого зберігання даних. Це дозволяє організаціям працювати з величезними наборами даних, виконуючи паралельну обробку і забезпечуючи їхню доступність.

NoSQL бази даних, завдяки своїй гнучкості у структурі і можливостям масштабування, стають основним вибором для зберігання неструктурованих даних.

3. Аналіз даних та машинне навчання:

Машинне навчання використовує різноманітні алгоритми для аналізу великих обсягів даних, що дозволяє бізнесам робити прогнози та приймати обґрунтовані рішення.

Регресійні алгоритми, зокрема лінійна та поліноміальна регресії, продемонстрували свою ефективність у численних сферах, включаючи фінанси та управління запасами.

5 Обробка природної мови (NLP):

Технології NLP продовжують еволюціонувати, запроваджуючи нові моделі, такі як Word2Vec і Transformer, що значно підвищують ефективність обробки текстової інформації.

Незважаючи на значний прогрес, NLP все ще стикається з викликами, такими як обробка ідіом і іронії, що свідчить про потребу в подальшому вдосконаленні моделей.

6. Хмарні рішення:

   Хмарні платформи, такі як AWS, Google Cloud Platform і Microsoft Azure, пропонують масштабовані рішення для обробки великих даних. Вони забезпечують не лише економію витрат, а й швидкий доступ до даних та гнучкість у їх обробці.

   Вибір хмарного рішення повинен базуватися на унікальних вимогах бізнесу, обсягах даних та аналітичних задачах.

Загалом, обробка та аналіз великих даних вимагають комплексного підходу, використання сучасних технологій та постійного вдосконалення алгоритмів і рішень. Важливість цих аспектів зростає в умовах швидко змінюваного цифрового середовища, де дані стають основним активом для досягнення конкурентних переваг і інновацій у бізнесі.

РОЗДІЛ 3 АРХІТЕКТУРА РОЗРОБЛЕНОГО ПРОГРАМНОГО ПРОДУКТУ ТА АНАЛІЗ РЕЗУЛЬТАТІВ РОБОТИ

3.1 Огляд використаних даних

Для розробки програми було використано два датасети та мову програмування Python. Детальний огляд використаних даних включатиме аналіз цих датасетів та методи їх обробки для досягнення поставленої мети. Python, як вибрана мова програмування, забезпечить ефективність у розробці та реалізації функціоналу програми.

3.1.1 Огляд датасетів

У сучасному світі кінематографу, де створення велико бюджетних фільмів є надзвичайно дорогим і ризикованим процесом, передбачення успіху фільму до його виходу набуває особливого значення. Набір даних, наданий The Movie Database (TMDb) через платформу Kaggle, дає можливість детально аналізувати різні аспекти виробництва фільмів, їх бюджети, команди та касові збори. Для розробки рекомендаційної системи було використано дані з 2-х датасетів «tmdb_5000_credits.csv» та «tmdb_5000_movies.csv». Надані датасети надають цінну інформацію, що допомагає краще розуміти контекст виробництва для кращого просування фільмів. Наприклад, поля «homepage» – домашня сторінка, «id» – ідентифікатор фільму в TMDb, «original_title» – оригінальна назва, «overview» – огляд сюжету, «popularity» – популярність, «production_companies» – виробничі компанії, «production_countries» – країни виробництва, «release_date» – дата виходу, «spoken_languages» –

мови, «status» – статус, «tagline» – слоган, «vote_average» – середній рейтинг, які надають більш детальний опис фільмів та їх комерційний і художній контекст. Проте слід зазначити, що якість даних також потребує подальшого аналізу, щоб виявити можливі помилки і невідповідності. Наприклад, значення нуля в полі «бюджет» може свідчити про відсутність даних, а не про реальний нульовий бюджет.

3.1.2 Огляд використаних технологій

Бібліотеки, використані у цьому коді, є важливими інструментами в сфері аналізу даних, машинного навчання та обробки природної мови. Кожна з них виконує унікальну роль, допомагаючи перетворити сирі дані у корисні інсайти та реалізовувати їх у складні алгоритми.

Бібліотека Pandas є наріжним каменем аналізу даних у мові Python. Вона надає потужні інструменти для роботи з табличними даними, дозволяючи зчитувати, обробляти та аналізувати великі набори даних з високою ефективністю. Pandas підтримує різноманітні формати найпопулярніших файлів, зокрема: CSV, Excel та SQL бази даних. Крім того, Pandas дозволяє легко виконувати такі операції, як фільтрація, агрегація даних, злиття таблиць, обробка відсутніх значень, створення нових колонок тощо. При проведенні експерименту Pandas використовувався для зчитування даних з CSV файлів, їх очищення і трансформації, а також для створення нових стовпчиків з тегами, що описують кожен фільм.

NumPy, подібно до Pandas, є базовою бібліотекою для наукових обчислень у Python, що забезпечує підтримку великих, багатовимірних масивів і матриць та пропонує багату функціональність для математичних

операцій над ними. NumPy часто використовується у комбінації з іншими бібліотеками, як-от Pandas або Scikit-learn, для покращення продуктивності обробки числових даних. У коді до програми NumPy був використанний при роботі з числовими даними і в процесі роботи інших бібліотек, таких як Scikit-learn.

Бібліотека ast є частиною стандартної бібліотеки Python і дозволяє маніпулювати та аналізувати абстрактні синтаксичні дерева (AST). Вона особливо корисна для розбору стрічок JSON-подібного формату в Python-структури, такі як списки або словники. У програмному коді ast використовується для перетворення рядків, що містять JSON-подібні об'єкти, у справжні Python-структури, наприклад списки або словники, що полегшує їх подальшу обробку.

Scikit-learn є однією з найпопулярніших бібліотек для машинного навчання у Python. Вона надає простий і ефективний інструментарій для аналізу та моделювання даних, включаючи алгоритми класифікації, регресії, кластеризації, моделювання часу, імітації та багато іншого. У програмному коді Scikit-learn використовується для векторизації текстових даних за допомогою «CountVectorizer» і для обчислення косинусної схожості між векторами. Ці функції є ключовими для перетворення текстових тегів у числові вектори та обчислення їх схожості для системи рекомендації фільмів.

NLTK є провідною платформою для обробки тексту на природній мові (NLP) у Python. Вона надає велику кількість інструментів для обробки тексту, включаючи токенізацію, частиномовну розмітку, стемінг, лематизацію, аналіз тексту та інші. У програмному коді використовується Porter Stemmer з пакету nltk.stem для стемінгу слів - процесу перетворення слів до їх кореневої форми. Це допомагає зменшити розмір словника та

поліпшити результативність порівняння текстових даних у системі рекомендацій.

Pickle є стандартною бібліотекою Python для збереження об’єктів у файлову систему і їх подальшого зчитування. Ця бібліотека серіалізує Python-об’єкти у байтовий потік та дозволяє зберігати їх у файли. У цьому проекті pickle використовується для збереження оброблених даних нового датафрейму та матриці схожості на диск, що дозволяє зекономити час на підготовку даних під час наступного запуску системи рекомендацій.

3.2 Опис експерименту

Метою даної роботи було розробити програмний продукт, який представляє однорівневу рекомендаційну систему, яка використовує контентний метод фільтрації див. рис. 3.1.

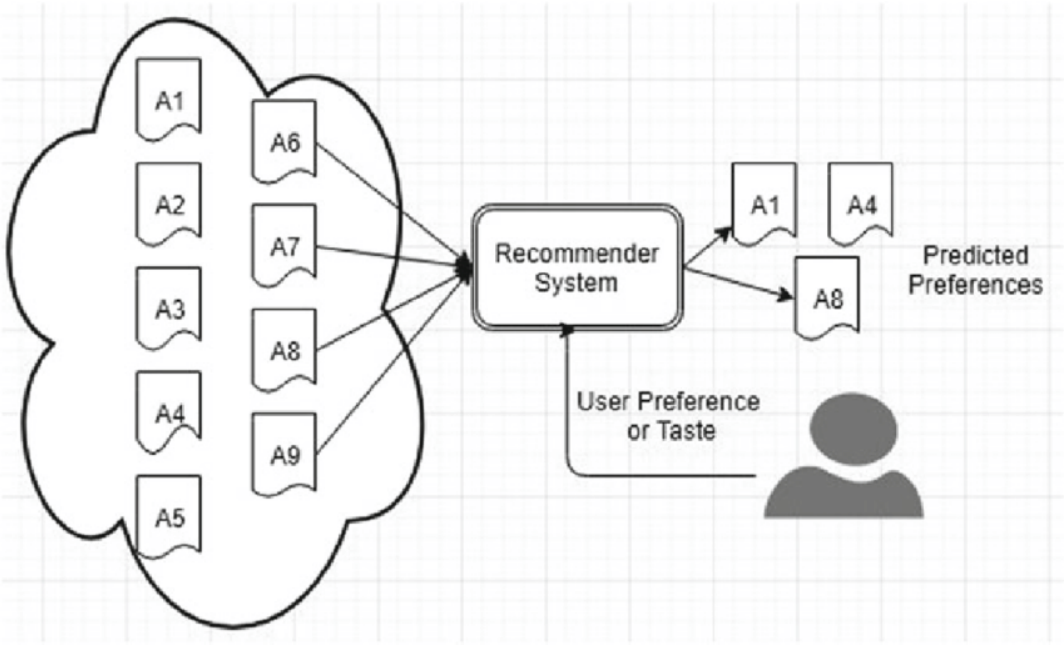

Рисунок 3.1 – схема однорівневої системи з контентним методом фільтрації

На першому етапі розробки було завантажено дані про фільми та кредити з CSV-файлів, а також виконано об'єднання двох датафреймів за колонкою «title» для отримання єдиного набору даних та видалення зайвих колонок, залишаючи лише необхідні для зчитування і з'єднання даних. Приклад зображено на рис. 3.2.

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 1463, "name": "culture clash"}, {"id":...	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	A cryptic message from Bond's past sends him o...	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 470, "name": "spy"}, {"id": 818, "name...	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	49026	The Dark Knight Rises	Following the death of District Attorney Harve...	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	[{"id": 849, "name": "dc comics"}, {"id": 853,...	[{"cast_id": 2, "character": "Bruce Wayne / Ba...	[{"credit_id": "52fe4781c3a36847f81398c3", "de...
4	49529	John Carter	John Carter is a war-weary, former military ca...	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 818, "name": "based on novel"}, {"id":...	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...

Рисунок 3.2 – виділення корисних колонок для подальшої роботи

Далі в роботі була виконана обробка даних, де спочатку видалили усі рядки з пустими значеннями і використали декілька функцій для: конвертації JSON-подібного рядка у список жанрів або ключових слів; функції, що бере перші три з списку імен акторів; а також функції, що витягують ім'я режисера.

Результати конвертації JSON-подібного рядка як список жанрів або ключових слів функції, що бере імена перших трьох акторів , а також функції, що витягує ім'я режисера – рис. 3.3 – рис. 3.5.

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[Action, Adventure, Fantasy, Science Fiction]	[{"id": 1463, "name": "culture clash"}, {"id":...	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[Adventure, Fantasy, Action]	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	A cryptic message from Bond's past sends him o...	[Action, Adventure, Crime]	[{"id": 470, "name": "spy"}, {"id": 818, "name...	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	49026	The Dark Knight Rises	Following the death of District Attorney Harve...	[Action, Crime, Drama, Thriller]	[{"id": 849, "name": "dc comics"}, {"id": 853,...	[{"cast_id": 2, "character": "Bruce Wayne / Ba...	[{"credit_id": "52fe4781c3a36847f81398c3", "de...
4	49529	John Carter	John Carter is a war-weary, former military ca...	[Action, Adventure, Science Fiction]	[{"id": 818, "name": "based on novel"}, {"id":...	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...

Рисунок 3.3 – конвертація JSON-подібного рядка як список жанрів або ключових слів

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[Action, Adventure, Fantasy, Science Fiction]	[culture clash, future, space war, space colon...	[Sam Worthington, Zoe Saldana, Sigourney Weaver]	[James Cameron]
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[Adventure, Fantasy, Action]	[ocean, drug abuse, exotic island, east india ...	[Johnny Depp, Orlando Bloom, Keira Knightley]	[Gore Verbinski]
2	206647	Spectre	A cryptic message from Bond's past sends him o...	[Action, Adventure, Crime]	[spy, based on novel, secret agent, sequel, mi...	[Daniel Craig, Christoph Waltz, Léa Seydoux]	[Sam Mendes]
3	49026	The Dark Knight Rises	Following the death of District Attorney Harve...	[Action, Crime, Drama, Thriller]	[dc comics, crime fighter, terrorist, secret i...	[Christian Bale, Michael Caine, Gary Oldman]	[Christopher Nolan]
4	49529	John Carter	John Carter is a war-weary, former military ca...	[Action, Adventure, Science Fiction]	[based on novel, mars, medallion, space travel...	[Taylor Kitsch, Lynn Collins, Samantha Morton]	[Andrew Stanton]

Рисунок 3.4 – функція, що бере імена перших трьох акторів

	movie_id	title	overview	genres	keywords	cast	crew
0	19995	Avatar	In the 22nd century, a paraplegic Marine is di...	[Action, Adventure, Fantasy, Science Fiction]	[culture clash, future, space war, space colon...	[Sam Worthington, Zoe Saldana, Sigourney Weaver]	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	[Adventure, Fantasy, Action]	[ocean, drug abuse, exotic island, east india ...	[Johnny Depp, Orlando Bloom, Keira Knightley]	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	A cryptic message from Bond's past sends him o...	[Action, Adventure, Crime]	[spy, based on novel, secret agent, sequel, mi...	[Daniel Craig, Christoph Waltz, Léa Seydoux]	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	49026	The Dark Knight Rises	Following the death of District Attorney Harve...	[Action, Crime, Drama, Thriller]	[dc comics, crime fighter, terrorist, secret i...	[Christian Bale, Michael Caine, Gary Oldman]	[{"credit_id": "52fe4781c3a36847f81398c3", "de...
4	49529	John Carter	John Carter is a war-weary, former military ca...	[Action, Adventure, Science Fiction]	[based on novel, mars, medallion, space travel...	[Taylor Kitsch, Lynn Collins, Samantha Morton]	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...

Рисунок 3.5 – ім'я режисерів, які були витягнуті функціями

Третім кроком в реалізації було формування тегів. Для цього спочатку в роботі об'єднали описи, жанри, ключові слова, акторів та режисерів в один рядок тегів. Наступним кроком звели все до одного регістру та використали стемінг, що конвертує слова до їх кореневої форми для покращення результатів аналізу. Для завершення третього кроку створюємо матриці з кількістю слів тобто векторизуємо текст для аналізу схожості. Результат виконання зображено на рис. 3.6:

```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

Рисунок 3.6 – результат векторізації тексту

На останньому етапі обчислюємо косинусну схожість та описуємо функцію для видачі рекомендацій. Косинусна схожість визначає, наскільки документи схожі між собою і на основі схожості видаються рекомендації для заданого фільму. Результатом буде вектор косинусної схожості (див. рис. 3.7) і реалізація роботи функції для видачі рекомендацій (див. рис. 3.8):

```
[(2409, 0.49356764739637354),
 (4336, 0.4113063728303113),
 (1537, 0.38807121827489494),
 (3162, 0.3879852850993631),
 (373, 0.380442955126341)]
```

Рисунок 3.7 – вектор косинусної схожості

```
recommend('Avatar')
```

```
Aliens
Silent Running
Moonraker
Alien
Mission to Mars
```

Рисунок 3.8 – результат видачі рекомендацій

3.3 Результати роботи програмного продукту

Результатом розробленого програмного продукту є функціональна система рекомендацій фільмів, яка дозволяє користувачам знаходити фільми, схожі на вказаний ними фільм. Для демонстрації роботи було використано бібліотеку streamlit. Це Python-бібліотека, яка дозволяє швидко створювати інтерактивні веб-додатки для аналізу і візуалізації даних. Вона спрощує процес створення веб-інтерфейсу для Python-додатків, особливо для задач, пов'язаних з аналізом даних, машинним навчанням і візуалізацією. Конкретні результати, досягнуті в ході реалізації проекту, наведені на рис. 3.9.

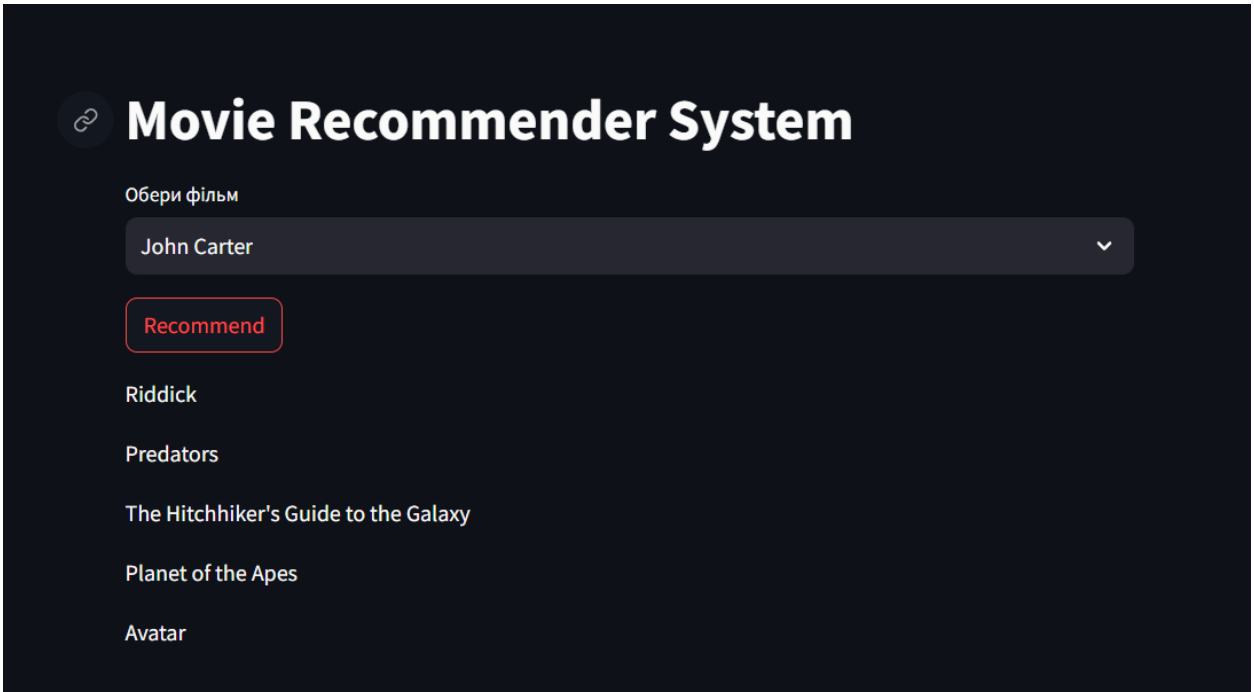

Рисунок 3.9 – результати роботи програми

3.4 Висновки до розділу 3

Розроблений програмний продукт є системою рекомендацій фільмів, яка аналізує текстові описи та атрибути фільмів, щоб надавати рекомендації на основі схожості вмісту. Це рішення використовує сучасні інструменти для обробки даних та текстової інформації, що робить його потужним та гнучким для розгортання в реальних додатках. Система ефективно зчитує великі набори даних з CSV-файлів та інтегрує їх у єдиний датафрейм. В свою чергу переобробка даних включає видалення відсутніх значень, а також екстракцію та трансформацію інформації з JSON-подібного формату до структурованих списків. Об’єднання інформації з різних колонок у нові стовпчики або теги – забезпечує єдиний текстовий опис кожного фільму, зберігаючи важливі деталі, такі як жанри, ключові слова, кастинг та режисуру.

Цей програмний продукт має кілька потенційних застосувань:

- персоналізовані рекомендації у стрімінгових сервісах для підвищення користувацького задоволення;

- аналіз кінострічок для кінокритиків та глядачів, що дозволяє знаходити фільми, схожі за тематичним наповненням і стилем;

- маркетингові дослідження фільмового контенту для визначення популярних жанрів і тематик.

Джерела

1 https://www.institutedata.com/blog/why-big-data-is-important/

2 https://www.orientsoftware.com/blog/big-data-evolution/

3 https://openmetal.io/resources/blog/understanding-big-data-infrastructure-options/

4 Data cleansing mechanisms and approaches for big data analytics: a systematic study

5 Big Data Visualization Tools

6 https://www.solvexia.com/blog/big-data-analysis-methods

7 https://iopscience.iop.org/article/10.1088/1757-899X/1022/1/012014/pdf

8 https://www.trigyn.com/insights/data-mining-techniques-era-big-data

9 https://www.geeksforgeeks.org/what-is-regression-analysis/

10 https://devsdata.com/big-data-and-nlp/

11 https://www.linkedin.com/pulse/bigdata-clustering-algorithms-luis-soares

12 https://www.tutorialspoint.com/big_data_analytics/time_series_analysis.htm

13 https://www.integrate.io/blog/guide-to-hdfs-for-big-data-processing/

14 https://pratikbarjatya.medium.com/using-nosql-databases-for-big-data-storage-and-retrieval-446350d1603b

15 https://www.researchgate.net/publication/318796844_NoSQL_databases_for_big_data

16 https://www.researchgate.net/publication/327386452_Machine_Learning_in_Big_Data

17 https://github.com/Ryota-Kawamura/Mathematics-for-Machine-Learning-and-Data-Science-Specialization

18 https://www.turing.com/kb/regression-analysis-techniques-in-data-science

19 https://devsdata.com/big-data-and-nlp/

20 https://aws.amazon.com/ru/blogs/big-data/a-side-by-side-comparison-of-

https://www.institutedata.com/blog/why-big-data-is-important/

https://www.orientsoftware.com/blog/big-data-evolution/

https://openmetal.io/resources/blog/understanding-big-data-infrastructure-options/apache-spark-and-apache-flink-for-common-streaming-use-cases/

21 https://journalofcloudcomputing.springeropen.com/articles/10.1186/s13677-022-00301-w

ДОДАТОК А. Код до програмного продукту

```
import numpy as np
import pandas as pd
movies = pd.read_csv('tmdb_5000_movies.csv')
credits = pd.read_csv('tmdb_5000_credits.csv')
movies.head(1)
credits.head(1)
movies = movies.merge(credits,on='title')
movies.head(1)
movies = movies[['movie_id','title','overview','genres','keywords','cast','crew']]
movies.info()
movies.head()
movies.isnull().sum()
movies.dropna(inplace=True)
movies.duplicated().sum()
movies.iloc[0].genres
def convert(obj):
    L = []
    for i in ast.literal_eval(obj):
        L.append(i['name'])
    return L
movies['genres'] = movies['genres'].apply(convert)
```

```
movies.head()
movies['keywords'] = movies['keywords'].apply(convert)
def convert3(obj):
    L = []
    counter = 0
    for i in ast.literal_eval(obj):
        if counter != 3:
            L.append(i['name'])
            counter+=1
        else:
            break
    return L
movies['cast'] = movies['cast'].apply(convert3)
movies.head()
def fetch_director(obj):
    L = []
    for i in ast.literal_eval(obj):
        if i['job'] == 'Director':
            L.append(i['name'])
            break
    return L
movies['crew'] = movies['crew'].apply(fetch_director)
movies.head()
movies['overview'][0]
movies['overview'] = movies['overview'].apply(lambda x:x.split())
```

```
movies.head()

movies['genres'] = movies['genres'].apply(lambda x: [i.replace("","") for i in x])

movies['keywords'] = movies['keywords'].apply(lambda x: [i.replace("","") for i in x])

movies['cast'] = movies['cast'].apply(lambda x: [i.replace("","") for i in x])

movies['crew'] = movies['crew'].apply(lambda x: [i.replace("","") for i in x])

movies.head()

movies['tags'] = movies['overview'] + movies['genres'] + movies['keywords'] + movies['cast'] + movies['crew']

movies.head()

new_df = movies[['movie_id', 'title', 'tags']]

new_df

new_df['tags'] = new_df['tags'].apply(lambda x: " ".join(x))

new_df.head()

new_df['tags'][0]

new_df['tags'] = new_df['tags'].apply(lambda x:x.lower())

new_df.head()

import nltk

from nltk.stem.porter import PorterStemmer

ps = PorterStemmer()

def stem(text):

    y = []

    for i in text.split():

        y.append(ps.stem(i))
```

```
        return " ".join(y)

    new_df['tags'] = new_df['tags'].apply(stem)

    new_df['tags'][0]

    new_df['tags'][1]

    from sklearn.feature_extraction.text import CountVectorizer

    cv = CountVectorizer(max_features=5000,stop_words='english')

    vectors = cv.fit_transform(new_df['tags']).toarray()

    vectors

    en(cv.get_feature_names())

    ps.stem('loving')

    from sklearn.metrics.pairwise import cosine_similarity

    similarity = cosine_similarity(vectors)

    sorted(list(enumerate(similarity[0])),  reverse=True,key=lambda
x:x[1])[1:6]

    def recommend(movie):

        movie_index = new_df[new_df['title'] == movie].index[0]

        distances = similarity[movie_index]

                movies_list   =   sorted(list(enumerate(distances)),
reverse=True,key=lambda x:x[1])[1:6]


        for i in movies_list:

            print(new_df.iloc[i[0]].title)

    recommend('Avatar')

    new_df.iloc[2409].title

    import pickle

    pickle.dump(new_df,open('movie_list.pkl','wb'))
```

```
new_df['title'].values

new_df.to_dict()

pickle.dump(new_df.to_dict,open('movie_dict.pkl','wb'))

new_df['title'].values

pickle.dump(new_df.to_dict,open('movie_dict.pkl','wb'))

print(pd.__version__)

pickle.dump(similarity,open('similarity.pkl','wb')
```

ДОДАТОК Б. Доповнення до програмного продукту

```
import streamlit as st

import pickle

import pandas as pd



def recommend(movie):

    movie_df = pd.DataFrame(movie_list, columns=['title'])



    movie_index = movie_df[movie_df['title'] == movie].index[0]

    distances = similarity[movie_index]

        movies_list = sorted(list(enumerate(distances)), reverse=True, key=lambda x: x[1])[1:6]
```

```
    recommended_movies = []
    for i in movies_list:
        recommended_movies.append(movie_df.iloc[i[0]]['title'])

    return recommended_movies


movie_list = pickle.load(open('movie_list.pkl', 'rb'))
movie_list = movie_list['title'].values

similarity = pickle.load(open('similarity.pkl', 'rb'))


st.title('Movie Recommender System')

selected_movie_name = st.selectbox(
    'Обери фільм',
    movie_list)

if st.button('Recommend'):
    recommendations = recommend(selected_movie_name)
    for i in recommendations:
        st.write(i)
```

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

Магістерська робота

«СИСТЕМА АНАЛІЗУ ВЕЛИКИХ ОБСЯГІВ ДАНИХ ДЛЯ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ»

Виконав: студент групи КСДМ-61 Гумінюк Владислав Юрійович

Керівник: к.т.н., доцент Антоненко Артем Васильович

Київ - 2025

МЕТА, ОБ'ЄКТА ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: Розробка рекомендаційної системи для демонстрації роботи з великим масивом даних.

Об'єкт дослідження: Процеси аналізу та обробки великих обсягів даних для персоналізованих рекомендацій.

Предмет дослідження: Методи, моделі та алгоритми, які використовуються для створення рекомендаційних систем із застосуванням косинусної схожості та векторизації текстів.

АКТУАЛЬНІСТЬ РОБОТИ

Основні аспекти актуальності:

1.Зростання обсягів даних у світі вимагає нових методів обробки, зберігання та аналізу інформації.

2.Традиційні методи аналізу часто є недостатньо ефективними через обмежену обчислювальну потужність.

3.Використання методів обробки великих даних (Big Data) є перспективним інструментом для вирішення сучасних завдань, зокрема у рекомендаційних системах

Недоліки існуючих методів:

•Висока залежність від якісної передобробки даних.

•Проблема масштабованості при обробці великих обсягів.

МАТЕМАТИЧНА МОДЕЛЬ

Обчислюється схожість товарів за допомогою користувацького рейтингу товарів. У статистичному аналізі часто використовуються дві метрики кореляції між товарами - PCC і вектор косинуса. Рівняння, наведене нижче, використовується для розрахунку методу скоригованого вектора косинуса

$$sim_{ij} = \frac{\sum_{u \in U_{ij}} (r_{ui} - \overline{r_u})(r_{vj} - \overline{r_v})}{\sqrt{\sum_{u \in U_i} (r_{ui} - \overline{r_u})^2} \sqrt{\sum_{u \in U_j} (r_{ui} - \overline{r_v})^2}}$$

sim_{iy} вказує на те, що два об'єкти I та j схожі в деякому сенсі. Об'єкт I був оцінений групами користувачів U_i та U_j тоді як позиція j була оцінена групою користувачів U_{ij} яка є групою користувачів, що оцінила обидва пункти I та j.

Користувач може бути зацікавлений у певних товарах, і корисні дані можуть бути зібрані таким чином, щоб на основі інформації зворотного зв'язку користувача можна було створити матрицю користувацьких товарів, кожен член якої має значення 0 або 1 . Прогнозовані рейтинги перераховуються в порядку зменшення, причому перші N пунктів рекомендуються користувачам в кінці процесу моделювання, а прогнозовані рейтинги потім сортуються в порядку зменшення. Використовуючи класифікацію на основі пам'яті для бінарних даних, у матриці зворотного зв'язку R якщо пара (u, i), тобто користувач-об'єкт розпізнається, $r_{ui} = 1$, а якщо пара (u,i) не розпізнано, то $r_{ui} = 0$. Ось чому косинус-вектор схожості визначається за допомогою бінарних оцінок.

МОДИФІКОВАНИЙ МЕТОД

1. Обробка JSON-подібних даних:
- Жанри та ключові слова: витягуються значення поля name.
- Список акторів: обмежений трьома найбільш релевантними.
- Режисери: обирається перша людина із роллю Director.
2. Об’єднання даних у теги:
- Усі зібрані дані (overview, genres, keywords, cast, crew) зводяться до текстового поля tags, яке слугує єдиною характеристикою для фільму.
3. Стемінг:
- Використання PorterStemmer із бібліотеки NLTK для зведення слів до кореневої форми.
- Наприклад, “loved”, “loving”, “love” перетворюються на “love”.
4. Векторизація:
- За допомогою CountVectorizer текст tags перетворюється у числові вектори.
- Враховуються лише 5000 найпоширеніших слів, що знижує вимоги до пам’яті та підвищує ефективність.

АЛГОРИТМ

1. На першому етапі розробки було завантажено дані про фільми та кредити з CSV-файлів, а також виконано об'єднання двох датафреймів за колонкою «title» для отримання єдиного набору даних та видалення зайвих колонок, залишаючи лише необхідні для зчитування і з'єднання даних.
2. Далі в роботі була виконана обробка даних, де спочатку видалили усі рядки з пустими значеннями і використали декілька функцій для: конвертації JSON-подібного рядка у список жанрів або ключових слів; функції, що бере перші три з списку імен акторів; а також функції, що витягують ім'я режисера.
3. Третім кроком в реалізації було формування тегів.
4. На останньому етапі обчислюємо косинусну схожість та описуємо функцію для видачі рекомендацій.

```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

Рис.3 - результат векторізації тексту

ПРАКТИЧНИЙ РЕЗУЛЬТАТ

Розроблений програмний продукт:

- Інтерактивна система рекомендацій фільмів із веб-інтерфейсом на основі бібліотеки Streamlit.
- Екранні форми демонструють результат роботи алгоритму рекомендацій.
- Система підходить для застосування в різних сферах (рекомендації товарів, музики тощо) .

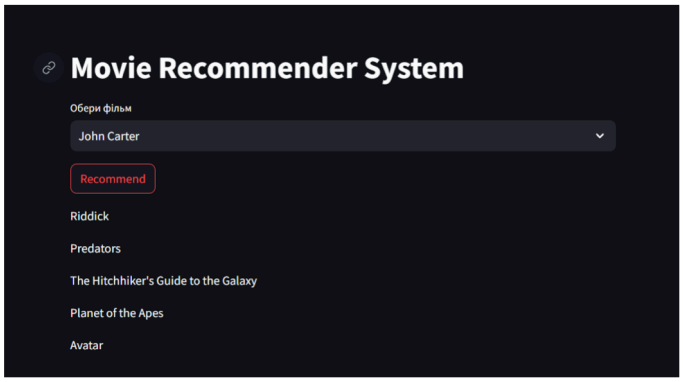

Рис. 5 – результати роботи програми

РЕЗУЛЬТАТ МОДЕЛЮВАННЯ ТА ПОРІВНЯЛЬНИЙ АНАЛІЗ

Таблиця порівняння результатів запропонованого методу з базовими моделями:

Метод	Точність	Час обробки (с)
Без косинусної схожості	65%	0.10
Без стемінгу	78%	0.15
Мій продукт	90%	0.15

```
[(2409, 0.49356764739637354),
 (4336, 0.4113063728303113),
 (1537, 0.38807121827489494),
 (3162, 0.3879852850993631),
 (373, 0.380442955126341)]
```

Рис. 5 – вектор косинусної схожості

```
recommend('Avatar')
```

```
Aliens
Silent Running
Moonraker
Alien
Mission to Mars
```

Рис. 6 – результати роботи системи рекомендацій

ВИСНОВОК

Розроблений програмний продукт є системою рекомендацій фільмів, яка аналізує текстові описи та атрибути фільмів, щоб надавати рекомендації на основі схожості вмісту. Це рішення використовує сучасні інструменти для обробки даних та текстової інформації, що робить його потужним та гнучким для розгортання в реальних додатках. Система ефективно зчитує великі набори даних з CSV-файлів та інтегрує їх у єдиний датафрейм. В свою чергу переобробка даних включає видалення відсутніх значень, а також екстракцію та трансформацію інформації з JSON-подібного формату до структурованих списків. Об'єднання інформації з різних колонок у нові стовпчики або теги – забезпечує єдиний текстовий опис кожного фільму, зберігаючи важливі деталі, такі як жанри, ключові слова, кастинг та режисуру.

Цей програмний продукт має кілька потенційних застосувань:

- персоналізовані рекомендації у стрімінгових сервісах для підвищення користувацького задоволення;
- аналіз кінострічок для кінокритиків та глядачів, що дозволяє знаходити фільми, схожі за тематичним наповненням і стилем;
- маркетингові дослідження фільмового контенту для визначення популярних жанрів і тематик.

ДЯКУЮ ЗА УВАГУ!