Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Методи побудови рекомендаційних систем на основі класифікації об'єктів за їх зображеннями
(тема)

Виконав:
студент 2 курсу, групи СШІм-20-2
Чекалкін П.О.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)
Керівник проф. Рябова Н.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2022

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____20____р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Чекалкіну Павлу Олеговичу
(прізвище, ім'я, по батькові)

1. Тема роботи Методи побудови рекомендаційних систем на основі класифікації об'єктів за їх зображеннями

затверджена наказом університету від 24 березня 2022 р. № 414Ст

2. Термін подання студентом роботи до екзаменаційної комісії 12 травня 2022 р.

3. Вихідні дані до роботи Науково-технічна публікація, дані Інтернет-джерел та відомих наукових проектів щодо розробки та дослідження різних нейромережевих методів побудови рекомендаційних систем на основі класифікації об'єктів за їх зображенням

4. Перелік питань, що потрібно опрацювати в роботі

1) Аналіз предметної галузі

2) Огляд існуючих методів для задачі перетворення

3) Опис обраного методу для вирішення задачі

4) Програмна реалізація та аналіз результатів

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)

Рисунок 2.1 – Приклад колоборативної фільтрації, Рисунок 2.2 – Схожості та відмінності, що використовуються в алгоритмах колаборативної фільтрації, Рисунок 2.4 – Моделі рекомендаційних систем на базі гібридних алгоритмів, Рисунок 2.5 – Етапи функціонування рекомендаційної системи, Рисунок. 2.6 – Узагальнена архітектура рекомендаційної системи, Рисунок 2.7 – Гібридна рекомендаційна модель з урахуванням тональності відгуків, Рисунок 2.8 – Матриця товарних переваг, Рисунок 2.9 – Принцип роботи алгоритму найближчих сусідів, Рисунок 2.10 – Алгоритм SVD, Рисунок 2.11 – Алгоритм роботи факторизаційних машин, Рисунок 3.1 – Найпростіший приклад роботи штучної нейронної мережі, Рисунок 3.2 – Архітектури нейронних мереж: а – повно зв'язна мережа, б – багатошарова мережа з послідовними зв'язками, в – слабо зв'язані мережі, Рисунок 3.3 – Повна архітектура загорткової нейронної мережі, Рисунок 3.4 – Робота шару згортки, Рисунок 3.5 – Приклад максимального пулінгу, Рисунок 3.6 – Повністю зв'язаний шар, Рисунок 3.7 – Принцип роботи градієнтного спуску, Рисунок 3.8 – Принцип роботи регресійного аналізу, Рисунок 3.10 – Структура поліноміального регресійного аналізу, Рисунок 4.1 – Гібридний підхід, Рисунок 4.2 – Шаблон рекомендацій у програмі Create ML, Рисунок 4.3 – модель готова до використання, Рисунок 4.4 – модель готова до використання,

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

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Основна частина	проф. Рябова Н.В.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання	28.03.2022	виконано
2	Аналіз предметної області і постановка задачі	29.03.2022	виконано
3	Аналіз методів щодо вирішення задачі	31.03.2022	виконано
4	Розробка вимог до програмної системи	02.04.2022	виконано
5	Розробка системи	04.04.2022	виконано
6	Програмна реалізація	04.04.2022	виконано
7	Аналіз результатів	12.04.2022	виконано
8	Оформлення пояснювальної записки	15.04.2022	виконано
9	Попередній захист		
10	Захист перед ЕК		

Дата видачі завдання 28 березня 2022 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Рябова Н.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 73 с., 40 рис., 1 дод., 44 джерел.

КЛАСИФІКАЦІЯ ЗОБРАЖЕННЯ, МАШИННЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ, РЕКОМЕНДАЦІЙНА СИСТЕМА.

Об'єкт дослідження – методи побудови рекомендаційних систем на основі класифікації об’єктів за їх зображенням.

Предмет дослідження – дослідження методів побудови рекомендаційних систем на основі класифікації об’єктів за їх зображенням.

Мета роботи – дослідження та аналіз найкращих практик, моделей та методів та підвищення ефективності надання рекомендацій.

Методи дослідження – аналіз технічної літератури з області побудови рекомендаційних систем на основі класифікації об’єктів за їх зображенням.

Проведено теоретичний аналіз різних архітектур нейронних мереж, методів побудови рекомендаційних систем на основі класифікації об’єктів за їх зображенням.

В результаті проведення досліджень вирішено задачу побудови рекомендаційних систем на основі класифікації об’єктів за їх зображенням за допомогою нейронної мережі. Отримані результати мають змогу використовуватися в різних напрямках, наприклад: персональні рекомендації по фотографії, аналіз та підбір доцільної інформації для користувача.

ABSTRACT

Explanatory note: 73 p., 40 fig., 1 ann., 44 sources.

IMAGE CLASSIFICATION, MACHINE LEARNING, NEURAL NETWORKS, RECOMANDATION SYSTEM.

Object of research - methods of building recommendation systems based on the classification of objects by their image.

The subject of research is the study of methods of building recommendation systems based on the classification of objects by their image.

The purpose of the work is research and analysis of best practices, models and methods and increase the effectiveness of providing recommendations..

Research methods – analysis of technical literature in the field of construction of recommendation systems based on the classification of objects by their image.

Theoretical analysis of different architectures of neural networks, methods of construction of recommendation systems based on the classification of objects by their image.

As a result of research, the problem of building recommendation systems based on the classification of objects by their image using a neural network was solved. The obtained results can be used in different areas, for example: personal recommendations on photography, analysis and selection of relevant information for the user.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

КФ – колаборативна фільтрація;

РС – рекомендаційна Система;

ШІ – штучний інтелект;

SVD – Singular Value Decomposition – розкладання сингулярних значень;

CBF – Content–Based Filtering – фільтрація на основі вмісту;

TF-IDF – Term Frequency – Inverse Document Frequency – частота термінів – зворотна частота документів.

ВСТУП

Сьогодні, одним з найбільш цінних ресурсів людства – є інформація. Вона займає одну з центральних ролей у процесі формування сучасного суспільства. З виникненням Інтернету – основним сховищем інформації стала саме Всесвітня Мережа. Але з часом, її об’єми почали зростати значно швидше, ніж обчислювальні можливості, для обробки даних. Кількість інформації в Інтернеті настільки велика, що людина просто не здатна знайти те, що їй дійсно потрібно.

Сьогодні кількість електронних ресурсів важко підрахувати. Кожен з них представляє собою величезний набір даних. Всю цю інформацію можна і потрібно обробляти. Завдяки цьому можна підвищити ефективність системи. Це відбувається шляхом аналізу поведінки користувачів ресурсів, для прогнозування їх побажань, та відображенню прогнозованих рекомендацій окремим користувачам. Даний тип систем називається «рекомендаційними».

Це інформаційні системи, завдання яких, запропонувати користувачеві дії, послуги або товари на основі попередніх переваг його або користувачів, схожих на нього за купленим товарами / послугами. У зв’язку з поширеністю рекомендаційних систем різних типів, тема є актуальною.

Також актуальною є розробка спеціальної системи, яка буде сама пропонувати користувачу деякі елементи, яка вона буде вважати доцільними. Така система буде формувати свої рекомендації на основі поведінки користувачів в минулому та їх вподобань. Це дозволить користувачам зекономити великий об’єм часу для пошуку необхідного контенту. Резюмуючи все вищезазначене, можна зробити висновок що тема досліджень в області рекомендаційних систем є досить популярною. Рекомендаційні системи зараз знаходяться у стані стрімкого розвитку, тобто якість їх роботи постійно збільшується.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Передпроектне обстеження

1.1.1 Опис предметної області

Предмет дослідження рекомендаційної системи на базі нейронної мережі, що дозволить скоротити час проведення тестів та спростити вибір спеціальності, за рахунок автоматизації пошуку, за певним набором характеристик. Для досягнення поставленої мети необхідно вирішити наступні задачі:

а) спроектувати модель предметної області, визначити необхідної вхідну і вихідну інформацію;

б) розробити алгоритм вибору найкращого рішення з використанням методів і засобів штучного інтелекту, який дозволяє вибрати найкращий варіант з усіх можливих рішень і представити рішення в порядку переваги. Для цього необхідно:

− дослідити і проаналізувати основи нейронних мереж;

− дослідити і проаналізувати сутність нейронних мереж;

− розібрати технології проектування нейронних мереж;

в) провести аналіз особливостей і специфіки нейронних систем для підтримки прийняття рішень щодо вибору спеціальності;

г) провести дослідження існуючих методів нейронної мережі, а саме прямих і ітераційних. Використання прямих методів у даній сфері є традиційним і найчастіше полягає у відображенні одержаної системи лінійних алгебраїчних рівнянь на архітектуру типу Хопфілда.

д) порівняти використовувані методи нейронної мережі; 13

е) розробити обрання спеціалізації майбутнього навчання на основі методів:

– метод уточнення результату – даний метод є розвитком результатів робіт. За основу використовується питання закритого типу і у випадку, якщо відповідь буде дана невірно, тестуючому буде дано уточнювальне питання;

– метод редагування помилок – об’єднання методів оцінки відповіді на питання закритого і відкритого типу роботи систем на основі нейронних мереж полягає в розробці програм, які при вирішенні задач, важких для експерта–людини, отримують результати, що не поступаються за якістю та ефективністю рішень, що отримуються експертом.

1.1.2 Проблеми та міркування

У сучасних системах, на практиці, зазвичай застосовують комбінації цих та інших підходів, характер їх поєднання може бути надзвичайно різним. З точки зору алгоритмів, найбільшу поширеність все ж на сьогодні завоював алгоритм матричної факторизації, переможець конкурсу Netflix prize. Це один з алгоритмів колаборативної фільтрації, заснований на формальній моделі розкладу матриць на менші матриці для зменшення кількості обрахунків. Загальною характеристикою алгоритмів, заснованих на моделях, можна вважати такий важливий елемент, як закладена у них здатність до знаходження рішень незалежно від людини, що може привести до цілком неочевидних, але не менш ефективних рішень.

Рекомендаційні системи займають важливе місце для таких відомих компаній як Amazon.com, YouTube, Netflix, Spotify, LinkedIn, Facebook, TripAdvisor, Last.fm, IMDb, Google та Yandex. Більше того, багато медіа компаній зараз розробляють рекомендаційні системи як частину свого сервісу, яку вони надають своїм користувачам. Компанія Netflix, онлайн провайдер потокового відео, нагородила призом в мільйон доларів команду, яка перша змогла значно покращити продуктивність їхньої рекомендаційної

моделі. Відповідно до досліджень проведених компанією McKinsey, більше 75% контенту, який дивились користувачі компанії Netflix, був саме запропонований їхньою рекомендаційною системою.

Система допомагає споживачам знайти цікавий контент, який вони хочуть подивитись, а також допомагає компанії економити на витратах маркетингу [2]. Торгівельний гігант Amazon заявив, що 35% свого доходу він отримав завдяки їхнім рекомендаційним системам. Перш за все, вони показують товари, які часто купують з тими товарами, які у них вже є в кошику. Також вони показують товари, які схожі до тих, які вони щойно переглянули, та товари які є новими версіями до тих, які вони вже купили.

Поява та зростання інтернет магазинів має значний вплив на поведінку покупців, надаючи їм доступ до значної різноманітності товарів та інформації. Поки ця свобода покупок зробила інтернет комерцію в багатомільярдну індустрію, вона також ускладнила для покупців вибір товарів, які найбільше відповідають їхнім потребам. Одним із основних методів вирішення проблеми інформаційного перевантаження є рекомендаційні системи, які забезпечують автоматизоване та персоналізоване пропонування товарів для споживачів [4].

1.2 Аналіз існуючих аналогів.

Інтернет-магазин eBay має досить розвинену програмну платформу, яка об’єднує ряд програмних інтерфейсів (API) та сервісів, які направлені на ефективнішу взаємодію кінцевих користувачів з каталогом товарів eBay. Використання цих переваг дає можливість значно покращити якість роботи рекомендаційної системи, явно чи неявно вплинувши на зростання релевантності пропозицій товар чи послуг:

− виконання пошукових запитів з урахуванням семантики запиту, що дозволяє забезпечувати кращу відповідність результатів пошуку очікуванням користувачів. Для обліку семантики запитів кожен рівень

таксономічного графу (каталог товарів та послуг) асоціюється з набором ключових слів, які забезпечують максимально точне відображення можливих термінів в пошуковому запитів на відповідні розділи каталогу. Підхід, заснований на використанні ключових слів, є досить обмеженим на відміну від використання онтологій, але в контексті інтернет-магазину виглядає цілком виправданим;

− обробка відгуків покупців з метою формування рейтингу продавця (Feedback API). Товари, що пропонуються з «високим» рейтингом матимуть більший пріоритет при формуванні пропозицій для потенційних покупців;

− формування груп взаємопов'язаних товарів, також які можуть пропонуватися «пакетом» (Related Items Management API). Таким чином, продавець має можливість явно вказувати які ще товари можуть бути корисні покупцю;

− створення зв'язків, які описують взаємозв'язки комплектуючих та аксесуарів з продуктами (Product Services). Так покупець при пошуку ноутбука може, наприклад, отримати одразу ж список сумок, що підходять до його моделі ноутбука.

Інший досить крупний інтернет-магазин Amazon (amazon.com) розробив власне високопродуктивне сховище пар «ключ–значення» (Highly Available Key-Value Store) Dynamo, яке використовується рекомендаційною системою Amazon. Dynamo використовує синтез добре відомих технік для досягнення масштабування та високої доступності: дані кластеризуються (partioning) та реплікуються, використовуючи узгоджене хешування (consistent hashing), а корректність даних забезпечується за допомогою версій об’єктів. Для 99% запитів СУБД забезпечує час відгуку на запит не більше, ніж 300 мс. Рекомендаційна система інтернет-магазину Amazon реалізує різні підходи до формування рекомендацій, основною метою яких є максимальний облік інтересів користувачів за допомогою їх залучення до процесу «оцінювання товарів».

Amazon та eBay використовують схожі технології для надання користувачам рекомендацій, але якщо звернути увагу на статистику відгуків про якість рекомендаційної систем цих інтернет-сервісів, Amazon має більш розвинену інфраструктуру та якість систем, що корелюють дані користувачів для надання рекомендацій.

1.2 Постановка задачі

Після проведення аналізу предметної області та існуючих аналогів, необхідно зробити дослідження методів побудови рекомендаційних систем на основі зображень та розробити програмний продукт для демонстрації отриманих результатів, а саме:

– провести дослідження існуючих методів формування рекомендацій;

– розробити власний алгоритму формування рекомендацій на основі дворівневої моделі;

– провести оцінку якості роботи запропонованого алгоритму;

– програмно реалізувати рекомендаційну системи на основі розробленого алгоритму.

2 ОГЛЯД ІСНУЮЧИХ МЕТОДІВ ДЛЯ ЗАДАЧІ ПОБУДОВИ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

2.1 Фундаментальні методи

У більшості рекомендаційних систем використовується один з двох базових підходів: колаборативна фільтрація (collaborative filtering) та контентна фільтрація (content-based filtering). Також існує клас підходів, що базуються на поєднанні двох основних – гібридна фільтрація (hybrid filtering).

Метод прогнозу в рекомендаційних системах, який використовує відомі переваги (оцінки) групи користувачів для прогнозування невідомих переваг (оцінок) іншого користувача. За допомогою цього алгоритму будується певна таблиця користувачів, які групуються за схожістю, та прогнозуються результати для інших користувачів. Колаборативна фільтрація прогнозує рекомендації, засновані на моделі попередньої поведінки користувача [2]. Ця модель може бути побудована виключно на основі поведінки цього користувача або – що більш ефективно – з урахуванням поведінки інших користувачів з подібними характеристиками. У тих випадках, коли колаборативна фільтрація бере до уваги реакцію інших користувачів, вона використовує знання про групу (group knowledge) для вироблення рекомендацій на основі схожості користувачів.

По суті рекомендації базуються на автоматичній взаємодії множини користувачів і на виділені (методом фільтрації) тих користувачів, які демонструють схожі уподобання або шаблони поведінки. Наприклад, при створенні веб-блогу, на якому необхідно запровадити рекомендаційну систему на основі інформації від багатьох користувачів, які підписуються на блоги і читають їх, можна згрупувати цих користувачів за їх інтересами. Можна об'єднати в одну групу користувачів, які читають кілька однакових

блогів і за цією інформацією ідентифікувати найпопулярніші блоги серед тих, які читають учасники цієї групи. Потім конкретному користувачу з 16 цієї групи будуть рекомендуватися найпопулярніший блог з тих, на які він ще не підписаний.

На рисунку 2.1 рядки відповідають блогам, а стовпці – користувачам [6]. Осередок на перетині рядка блогу і строки користувача містить кількість статей, прочитаних цим користувачем в цьому блозі. Кластеризація користувачів на основі читацьких вподобань (наприклад, за допомогою алгоритму найближчих сусідів) дозволяє виділити два кластери, кожен з яких містить по два користувача. Варто звернути увагу на схожість читацьких звичок у членів кожного кластера: Кластер 1 утворюють учасники з іменами Марк і Еліза, кожен з яких прочитав по кілька статей по тематиці «Linux» і за тематикою «Хмарні обчислення». Кластер 2 утворюють учасники з іменами Меган і Джилл, кожен з яких прочитав по кілька статей по тематиці «Javatехнології» і за тематикою «Agile-технології».

Блоги	Марк	Меган	Элиза	Джилл
Linux	13	3	11	-
Продукты с открытым кодом	10	-	-	3
Облачные вычисления	6	1	9	-
Java-технологии	-	6	-	9
Agile-технологии	-	7	1	8
	Количество статей, прочитанных данным пользователем			
Кластер	1	2	1	2

Рисунок 2.1 – Приклад колаборативної фільтрації [6]

Тепер можна ідентифікувати певні відмінності в рамках кожного кластера і сформувати рекомендації. У кластері 1 Марк прочитав 10 статей

з блогу «Продукти з відкритим вихідним кодом», а Еліза не прочитали жодної такої статті; Еліза прочитала одну статтю в блозі з Agile-технологіями, а Марк не прочитав жодної такої статті. Таким чином, виходячи з рисунку 2.1, Елізі можна порекомендувати блог «Продукти з відкритим вихідним кодом». Для Марка неможливо сформувати ніяких рекомендації, оскільки невелика різниця між ним і Елізою за кількістю прочитань блогу по Agile-технологій з великою 17 ймовірністю буде відфільтрована. У кластері 2 Джилл прочитала три статті в блозі «Продукти з відкритим вихідним кодом», а Меган не прочитали жодної такої статті; Меган прочитала 3 статті в блозі по Linux, а Джилл не прочитала жодної такої статті. Таким чином, для учасників кластера 2 можна сформувати наступні дві рекомендації: учаснику Джилл рекомендується блог по Linux, а учаснику Меган – блог «Продукти з відкритим вихідним кодом».

Інший спосіб розгляду цих відносини заснований на їх схожості та відмінності, як ілюструє діаграма Венна на рисунку 2.2 [6]. Схожість визначають, за якими ознаками (за допомогою відповідного алгоритму) слід згрупувати користувачів. Відмінності – це можливості, які можуть бути використані для вироблення рекомендацій – наприклад, за допомогою застосування фільтра популярності.

Хоча на рисунку 2.2 показана спрощена картина (з впливом розрідженості даних через використання лише двох зразків), таке уявлення досить зручне. Переваги: швидка робота алгоритмів (K-based та ін.) – мала кількість ітерацій; прості в реалізації.

Недоліки: не вирішені проблеми холодного старту, шахрайства, нема що рекомендувати новим або нетиповим користувачам; розріджені матриці оцінок (іноді неможливо зробити прогноз).

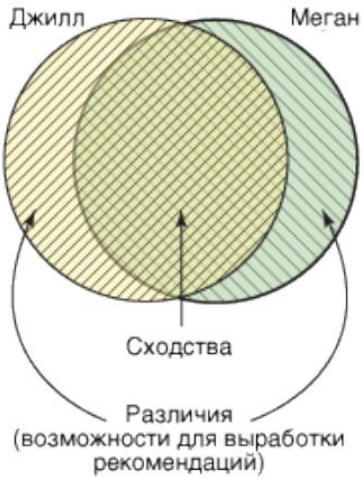

Рисунок 2.2 – Схожості та відмінності, що використовуються в алгоритмах колаборативної фільтрації [6]

Щоб модель рекомендацій працювала належним чином, її потрібно навчити з набором добре представлених і відповідних даних для створення профілю користувача. Існує два основних типи фільтрації інформації в Recommender Systems (рисунок 2.3):

− спільна фільтрація: цей тип фільтрації базується на попередніх оцінках і вподобаннях користувачів зі схожими смаками. По суті, він намагається знайти спільні риси серед користувачів, і коли це робить, він рекомендує нові елементи, які подобалися користувачам зі схожими перевагами в минулому. Дуже хорошим прикладом може бути Netflix, де якщо два дуже схожих користувача дивилися фільми A,B,C, а перший користувач також дивився фільм D, система рекомендує другому користувачеві також переглянути фільм D;

− фільтрація на основі вмісту: цей тип фільтрації заснований на подібності самих елементів. Відповідно до переваг користувача щодо певного елемента в минулому, фільтрація на основі вмісту намагається знайти інші подібні елементи та рекомендує їх користувачеві. Наприклад, якщо користувач раніше купував спортивний одяг, цілком ймовірно, що

наступного разу він запропонує йому кросівки, спортивні сумки та інші речі, пов’язані з спортзалом.

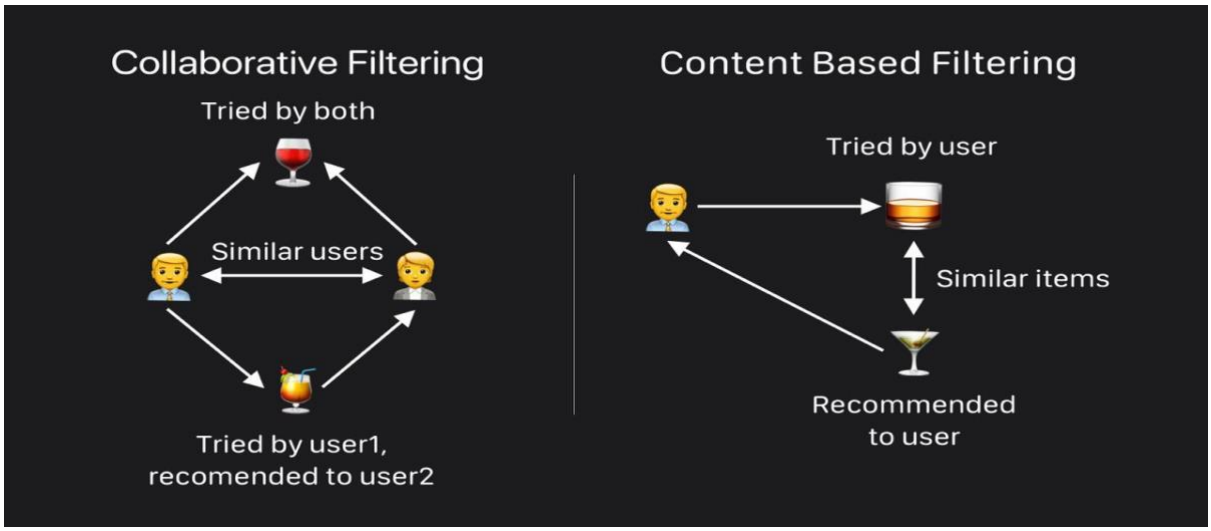

Рисунок 2.3 – Ілюстрація двох основних типів фільтрації інформації

Гібридні підходи, які поєднують колаборативну і контентну фільтрацію, також підвищують ефективність (і складність) рекомендаційних систем. Простий приклад гібридної системи міг би використовувати підходи, що показані на рисунку 2.4. Об'єднання результатів колаборативної і контентної фільтрації потенційно дозволяє підвищити точність рекомендації. Щоб модель рекомендацій працювала належним чином, її потрібно навчити з набором добре представлених і відповідних даних для створення профілю користувача.

Метод прогнозу в рекомендаційних системах, який використовує відомі переваги (оцінки) групи користувачів для прогнозування невідомих переваг (оцінок) іншого користувача. За допомогою цього алгоритму будується певна таблиця користувачів, які групуються за схожістю, та прогнозуються результати для інших користувачів.

Тип	Характеристика
Зважена	Оцінки, отримані методами колаборативної та контентної фільтрації, агрегуються для отримання єдиної рекомендації.
Комутуюча	Система використовує критерії, щоб перемикатися між методами контентної та колаборативної фільтрації.
Каскадна	Рекомендація такої моделі з'являється за рахунок удосконалення рекомендації інших систем.
Комбінаційна	Критерії з різних джерел рекомендацій вкидаються в єдиний рекомендаційний алгоритм.
Нарощувальна	Вихідні дані, отримані з однієї системи, використовуються як вхідні для іншої.
Змішана	Рекомендації з різних рекомендаційних систем показуються одночасно
Мета-рівень	Навчена модель з однієї рекомендаційної системи використовується в якості системних даних для іншої системи.

Рисунок 2.4 – Моделі рекомендаційних систем на базі гібридних алгоритмів

Функціонування рекомендаційної системи складається з наступних етапів (рисунок 2.5) [3], [5], [9]:

− нагромадження інформації про користувачів і предмети;

− навчання;

− прогнозування;

− зворотний зв'язок.

Рисунок 2.5 – Етапи функціонування рекомендаційної системи

На першому етапі відбувається збір і нагромадження інформації про користувачів і предмети для формування профілів користувачів і предметів. Ця інформація включає характеристики поведінки користувача, атрибути користувача, інформацію про ресурси, до яких звертається користувач, 52 інформацію про атрибути предметів. Для коректної роботи рекомендаційної системи потрібна найповніша інформація про користувачів і предмети. Для нагромадження достовірної інформації про уподобання і інтереси користувачів система використовує явний і неявний зворотний зв'язок. На другому етапі використовуються методи і алгоритми навчання для обробки інформації про користувачів, яка отримується за допомогою зворотного зв'язку (рисунок 2.6).

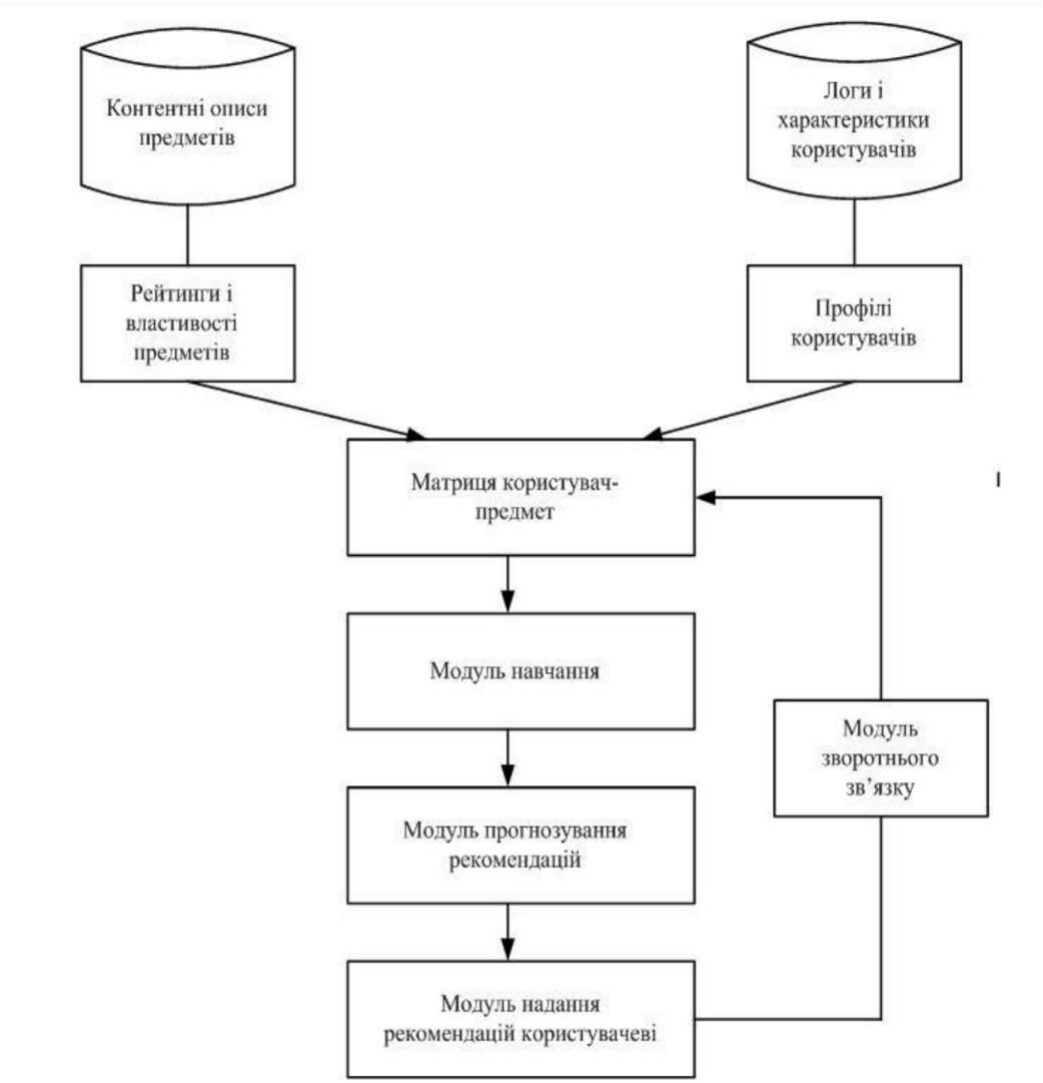

Рисунок 2.6 – Узагальнена архітектура рекомендаційної системи

В рекомендаційних системах використовується зворотний зв’язок двох видів: явний і неявний. Явний зворотний зв’язок полягає в тому, що користувач коректує прогнозовані рейтинги предметів після їх використання. При неявному зворотньому зв’язку система автоматично визначає уподобання користувача шляхом відслідковування різних дій користувача, таких, як історія покупок, історія навігації і час, який користувач проводить на веб-сторінках, посилання за якими здійснює навігацію користувач, послідовність натискання кнопок на веб-сторінках [8].

Узагальнена архітектура рекомендаційної системи наведена на рисунку 2.6. Нехай U − множина векторів профілів користувачів, I – множина профілів предметів, тоді множина R = U×I − множина можливих рекомендацій рекомендаційної системи.

Рекомендаційні системи, які використовують контентну фільтрацію, аналізують множину описів властивостей предметів, які раніше були вибрані користувачами [6], [7], [8], [9]. На основі цього аналізу рекомендаційна система будує профілі інтересів користувачів. Профіль представляє собою структуроване надання інтересу користувача і призначений для прогнозування рекомендацій нових предметів. Процес рекомендації полягає у співставленні профілів користувачів і атрибутів профілів предметів.

Результатом є оцінка рівня релевантності, яка визначає рівень інтересу користувача до предмета. Рекомендаційні системи, які використовують контентну фільтрацію, не вимагають даних про інших користувачів і можуть рекомендувати нові предмети і предмети, які не користуються великою популярністю. До недоліків контентно-орієнтованих систем можна віднести те, що вони використовують явний опис характеристик предметів і не змінюють його в процесі роботи.

Колаборативні рекомендаційні системи. Ідея колаборативної (спільної) фільтрації полягає в тому, що користувачі з подібними профілями (інтересами) виявляють інтерес до подібних предметів, а подібні предмети вибирають подібні користувачі. В методі колаборативної фільтрації використовується множина профілів користувачів і множина профілів предметів [5], [9].

Профіль користувача і профіль предмета містять числові рейтингові оцінки, які користувачі виставили вже вибраним предметам. Числова рейтингова оцінка – це ціле додатнє число, яке може приймати значення із певної множини R ={rmin , rmax } . При цьому rmin = 0, rmax = 5 або 10. Рейтинговою оцінкою може бути також двійкове значення =1, = 0. like

dislike r r Прогноз рекомендацій для активного предмета або активного користувача здійснюється за допомогою векторів профілів подібних користувачів із околу активного користувача або векторів профілів подібних предметів із околу активного предмета.

При цьому активний користувач – це користувач, який звернувся до рекомендаційної системи для отримання рекомендацій, активний предмет – це предмет, для якого рекомендаційна система повинна здійснити прогноз оцінок користувачів. На даний час найбільш широко використовують колаборативні рекомендаційні системи.

До переваг рекомендаційних систем колаборативної фільтрації можна віднести те, що вони не вимагають явного опису характеристик предметів. Такі системи використовують рейтингові числові оцінки предметів, які відповідають дійсним уподобанням користувачів. До недоліків можна віднести велику розмірність матриці користувач-предмет, неможливість прогнозувати рекомендації для нових предметів і нових користувачів.

Демографічні рекомендаційні системи Демографічні рекомендаційні системи класифікують користувачів на основі їх демографічних характеристик (вік, стать, освіта, рід занять тощо). В подальшому демографічні рекомендаційні системи використовують числові рейтингові оцінки користувачів, які вони виставили предметам і які зберігаються в профілях користувачів [2], [6].

Демографічні рекомендаційні системи використовують для прогнозування обчислювальні вирази, які подібні до методу предмет–предмет в колаборативних рекомендаційних системах. Однак, на відміну від методу предмет–предмет в колаборативних рекомендаційних системах, в демографічних рекомендаційних системах використовують коефіцієнти подібності між векторами демографічних профілів користувачів. До переваг демографічних рекомендаційних систем можна віднести відсутність потреби в історії оцінок предметів користувачами. Демографічні рекомендаційні системи рекомендують однакові предмети для користувачів

з однаковими демографічними характеристиками і не враховують рівні інтереси користувачів в однакових демографічних групах, що значно обмежує вибір користувачів.

Системи рекомендацій, які використовують знання Традиційні рекомендаційні системи (контентно-орієнтовані рекомендаційні системи, колаборативні рекомендаційні системи) добре підходять для рекомендацій предметів, які володіють певними властивостями і відповідають певним уподобанням користувачів. Це книги, музика, фільми, новини. Однак, для таких предметів, як автомобілі, комп'ютери, квартири, такий підхід не є вдалим. Наприклад, певну квартиру купують не часто. Це не дозволяє нагромадити достатню кількість оцінок, які необхідні для методу колаборативної фільтрації.

Окрім того, рекомендації не є вдалими, якщо вони використовують оцінки річної давнини чи старіші. До систем рекомендацій, які використовують знання, належать системи рекомендацій на основі прецендентів CBR (case-based recommender systems) і рекомендаційні системи, які побудовані на основі вмісту факторів, які характеризують предметну область (constraint-based recommender systems) [1], [5], [6].

Системи рекомендацій на основі прецендентів прогнозують 46 рекомендації на основі метрик подібності. Рекомендаційні системи, які побудовані на основі вмісту факторів, які характеризують предметну область, прогнозують рекомендації на основі бази знань, яка містить явні правила про те, як пов'язані між собою вимоги користувачів і властивості предметів. Рекомендації на основі прецендентів використовують методології, які застосовуються при побудові експертних систем і які базуються на накопиченому досвіді. На відміну від експертних систем, що діють на основі логічних правил, CBR-системи зберігають успішні рішення ряду реальних проблем, так звані case (приклади, або прецеденти), і при появі нової проблеми знаходять (за певним алгоритмом, найчастіше за допомогою машини логічного висновку, з кількісною оцінкою) найбільш

підходящі (схожі) прецеденти, після чого пропонують відповідно модифіковану комбінацію їх рішень. Якщо нова проблема виявляється таким чином успішно вирішеною, це рішення заноситься (вводиться) в базу прецедентів для підвищення ефективності системи в майбутньому. Недолік CBR-систем в тому, що вони не створюють моделей або правил, які узагальнюють накопичений досвід.

Системи рекомендацій для груп користувачів В багатьох прикладних задачах виникає потреба в прогнозуванні рекомендацій для груп користувачів [4], [6]. До таких задач відносяться задачі прогнозування літературних творів певної тематики для груп користувачів бібліотек, інтернет ресурсів; рекомендації музичних творів для груп слухачів; рекомендації новин для інтернет-користувачів; рекомендації туристичних маршрутів; рекомендації товарів для груп покупців в задачах електронної комерції. Системи прогнозування рекомендацій для груп користувачів на відміну від попередніх, розглянутих вище підходів до побудови рекомендаційних систем, вимагають розв’язання певних специфічних задач, які характерні тільки для цієї предметної області.

До таких задач відносяться:

– виділення груп користувачів з подібними уподобаннями;

– агрегація уподобань групи користувачів в єдине інтегроване уподобання всієї групи користувачів [10];

– прогнозування таких рекомендацій, котрі би задовольняли всіх членів групи, тобто в групі не повинно бути користувачів, ступінь задоволення потреб яких значно відрізнявся би від ступеня задоволення потреб інших членів групи.

У зв’язку із зростанням кількості користувачів Інтернету в розробленні рекомендаційних систем все більше уваги приділяється розробленню рекомендаційних систем для груп користувачів.

Розвиток сучасних рекомендаційних систем вимагає підвищення ефективності аналізу та пошуку інформації у великих сховищах даних,

мережах, Інтернеті [8]. Тому все більш важливе значення в розвитку таких систем займають перспективні інформаційні технології інтелектуального аналізу даних (Data mining) [10], [13], аналізу текстової інформації (Text mining) [5], виявлення і пошуку залежностей і знань у вебданих (Web mining) [12], [15].

У кожній з цих технологій використовуються методи і алгоритми виділення груп і класів об'єктів. Серед них важливе місце займають методи та алгоритми кластерного аналізу, методи і алгоритми пошуку асоціативних правил. Важливою особливістю цих методів є те, що вони не вимагають початкових знань про поділ об'єктів на класи і групи і можуть використовуватися на початкових стадіях аналізу даних. Гібридні моделі і методи прогнозування рекомендацій можуть поєднувати декілька моделей і методів.

Колаборативний аналіз та аналіз на основі контенту підходять для компенсації недоліків один одного, що забезпечує точність та стабільність системи рекомендацій. З одного боку, колаборативна фільтрація може компенсувати відсутність персоналізації в методі на основі контенту.

З іншого боку, метод на основі контенту може компенсувати такий недолік методу колаборації як масштабованість, яка у нього є відносно слабкою. Загалом спосіб гібридної рекомендації виконується на основі даних користувача та даних про фільм, щоб отримати попередній список рекомендацій. Аналіз тональності впроваджується для оптимізації попереднього списку та отримання списку рекомендацій. Крім того, на основі гібридної рекомендаційної платформи це дослідження повністю бере до уваги ефективність системи рекомендацій.

У процесі рекомендації фільмів дослідження зосереджується на відгуках користувачів про фільми. В умовах колективної поведінки користувачі схильні вибирати товари або послуги, яким більшість людей віддає перевагу. Отже, у порівнянні з фільмами, у яких багато негативних відгуків, фільмам з більш позитивними відгуками будуть надаватися

пріоритети для рекомендації користувачам. Після оптимізації, генерується фінальний список рекомендацій, як показано на рисунку 2.7.

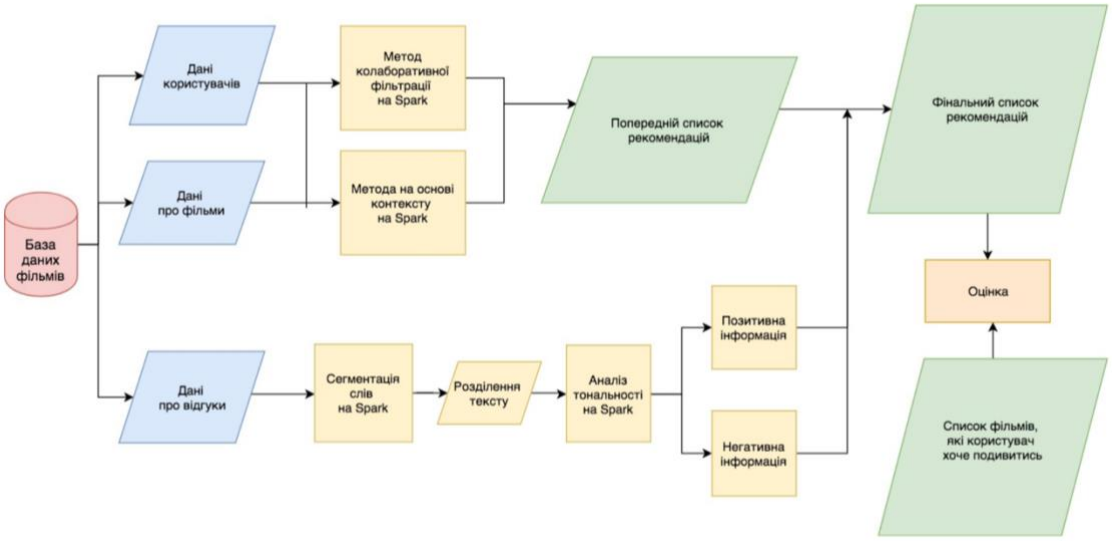

Рисунок 2.7 – Гібридна рекомендаційна модель з урахуванням тональності відгуків

Основний підхід розв'язання задачі побудови рекомендаційних систем – колаборативна фільтрація. Ідея колаборативної фільтрації проста – якщо користувач робив покупки товару або перегляди контенту, знайдемо користувачів зі схожими смаками, і порекомендувати нашому клієнту те, що схожі на нього люди споживали, а клієнт ще ні. Це User-Based підхід.

Аналогічним чином можна подивитися на завдання з погляду товару, і підібрати компліментарні товари до кошика клієнта, підвищивши середній чек, або замінивши товар, що відсутні на складі, на аналог. Це Item-Based підхід. У найпростішому випадку використовується алгоритм пошуку найближчих сусідів (рисунок 2.8). Приклад: Якщо Марії подобається фільм «Титанік» та «Зоряні війни», найближчим до смаку користувач до неї буде Аня, яка дивилася на додаток до цих фільмів ще й «Хатико». Порекомендуємо Марії фільм «Хатико».

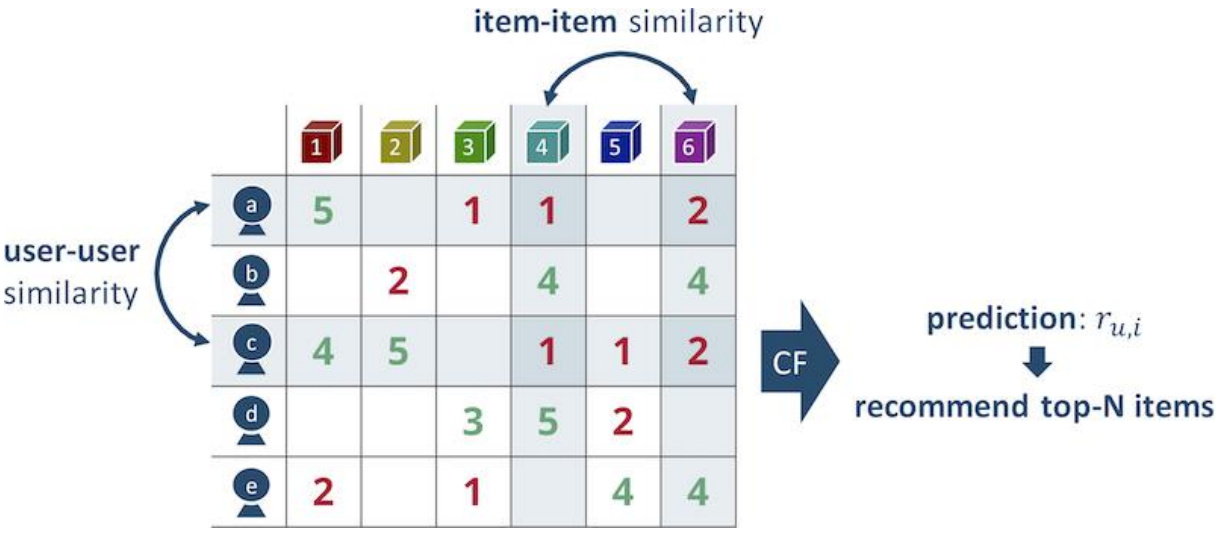

Рисунок 2.8 – Принцип роботи алгоритму найближчих сусідів

Варто уточнити, що зазвичай використовують не одного найближчого сусіда, а кілька із усередненням результатів.

Розглянемо складні алгоритми рекомендаційних систем, що базуються на властивості матриць, а точніше на розкладанні матриць.

Класичний алгоритм SVD – сингулярне розкладання матриць (рисунок 2.9). Сенс алгоритму у цьому, що матриця товарних переваг (матриця, де рядки це користувачі, а стовпці це продукти, із якими користувачі взаємодіяли) є твором трьох матриць (рисунок 2.10).

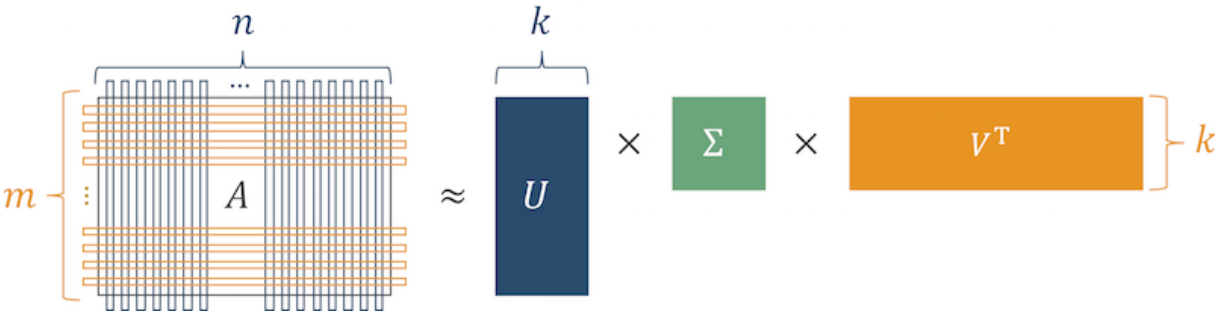

Рисунок 2.9 – Алгоритм SVD

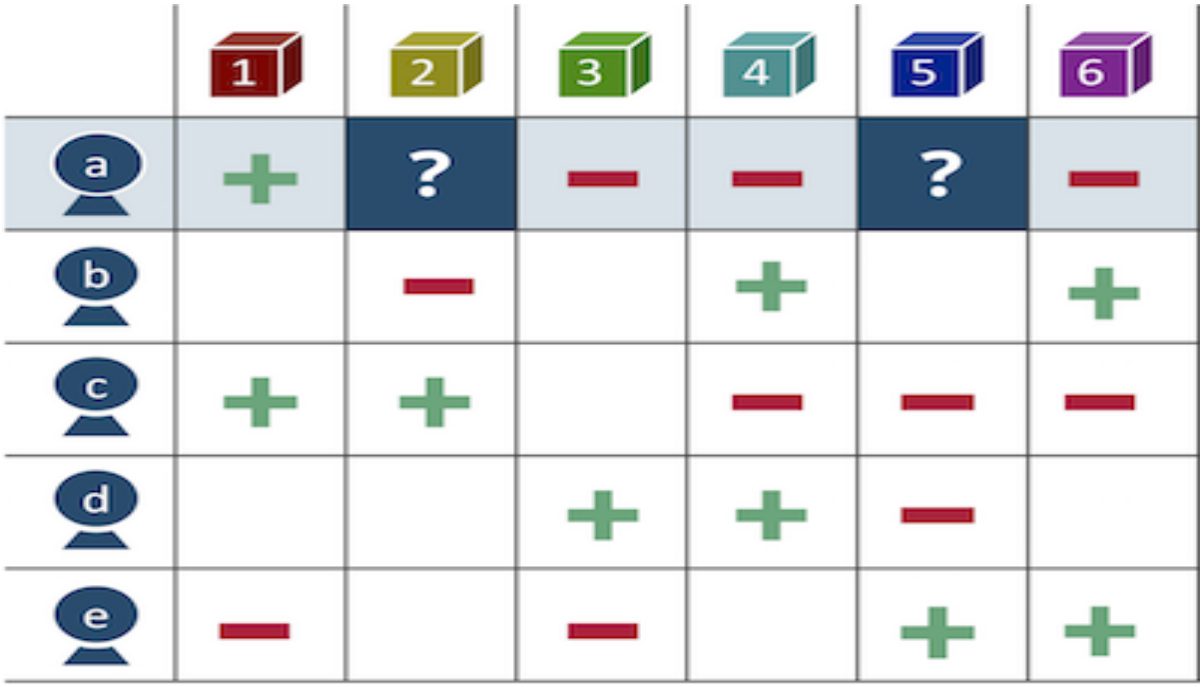

Рисунок 2.10 – Матриця товарних переваг

2.2 Прямі методи T2I

Після відновлення вихідної матриці, клітини, де користувача були нулі, а з'явилися «великі» числа, показують ступінь латентного інтересу до товару. Упорядкуємо ці цифри і отримаємо список товарів, релевантних для користувача. При цій операції у користувача та товару з'являються «латентні» ознаки. Це ознаки, що показують «прихований» стан користувача та товару. Але відомо, що і користувача та у товару крім «латентних» є ще й явні ознаки. Це стать, вік, середній чек покупки, регіон і т.д.

Рішення кейсів у галузі побудови рекомендаційних систем для інтернет магазинів, саме факторизаційні машини дають найкращий результат (рисунок 2.11).

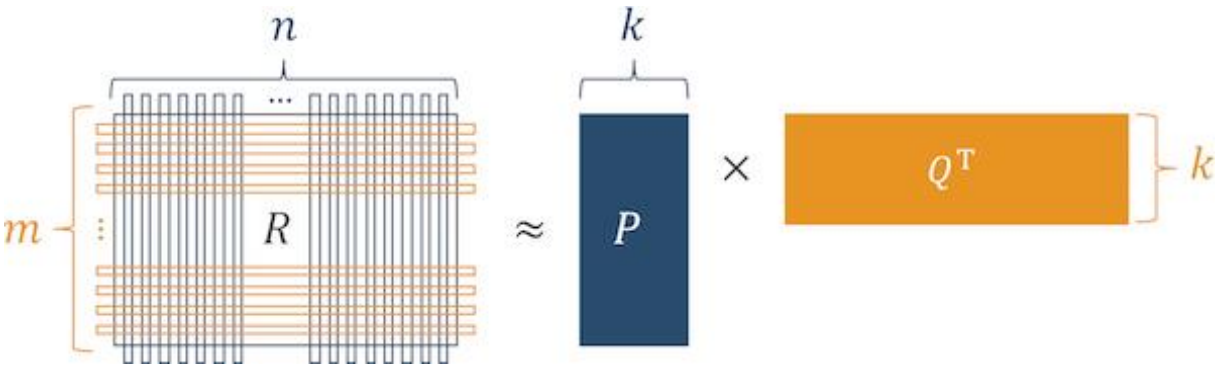

Рисунок 2.11 – Алгоритм роботи факторизаційних машин

Так, факторизаційні машини ми використали для побудови рекомендаційної системи для нашого клієнта компанії KupiVip. Приріст за метрикою RMSE становив 6–7%. Але підходи на основі розкладання матриць мають свої недоліки. Число узагальнюваних патернів взаємної комбінації товарів не велике. Для вирішення цієї задачі доцільно використовувати нейронні мережі. Але для нейронної мережі потрібні обсяги даних, які є тільки у великих компаній.

3 ОПИС ТЕХНОЛОГІЙ РОСПІЗНАВАННЯ ЗОБРАЖЕННЬ

3.1 Штучні нейронні мережі

Нейронні мережі дозволяють нам краще нам класифікувати і кластеризувати. Тобто вони виступають кластерним чи класифікаційним шаром поверх даних, якими ми можемо оперувати і зберігати. Також вони краще помогають групувати невідомі дані відповідно до загальних рис вхідних даних прикладу (рисунок 3.1), тобто класифікують їх, коли у них є набір даних щоб навчання було успішним.

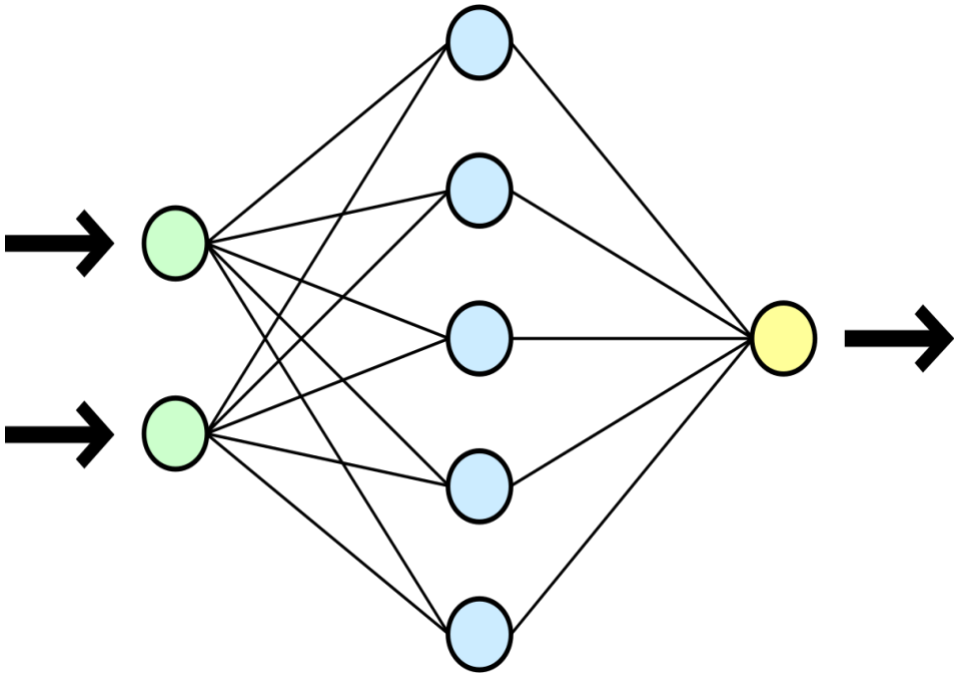

Рисунок 3.1 – Найпростіший приклад роботи штучної нейронної мережі

Класифікація нейронних мереж за топологією [6]:

– повно зв’язні мережі – кожен нейрон якої передає вихідний сигнал іншим нейронам, в тому числі і самому собі (рисунок 3.2, а), всі вхідні сигнали подаються до усіх нейронів;

– багатошарові або шаруваті мережі – зовнішні вхідні сигнали

подаються на входи нейронів 1 шару (нульовий) виступають лише в якості розподільних точок, в проміжних (прихованих) шарах проходить підсумовування і перетворення сигналів, а виходами мережі є вихідні сигнали крайнього шару (рисунок 3.2, б);

– слабо зв’язані мережі – шаруваті мережі з малою кількістю зв'язків (рисунок 3.2, в).

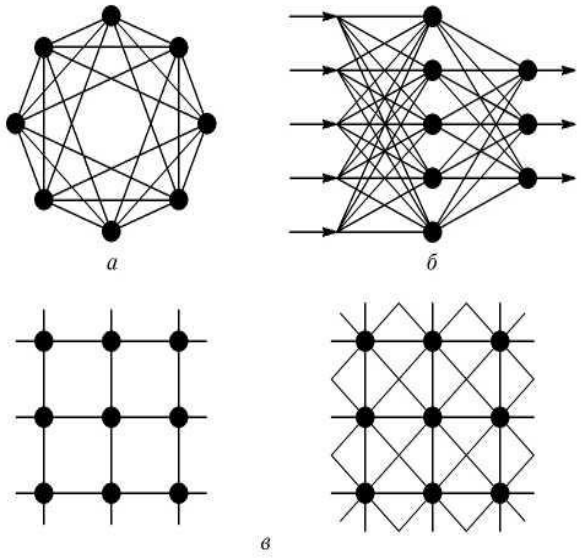

Рисунок 3.2 – Архітектури нейронних мереж:
а – повно зв’язна мережа, б – багатошарова мережа з послідовними зв'язками, в – слабо зв’язані мережі

3.2 Згорткові нейронні мережі

Згорткова нейронна мережа (ConvNets або CNNs) є однією з основних категорій для класифікації зображень, розпізнавання зображень, розпізнавання осіб, виявлення об'єктів, и тому подібне та є алгоритмом Deep Learning.

Архітектура ConvNet (рисунок 3.3) аналогічна архітектурі зв'язаних нейронів в людському мозку і була зроблена завдяки організації візуальної кори. Вихід з кінцевого шару об’єднання, який є сплощеним, є вхідним для

повністю підключеного шару. Процес повного підключення практично працює наступним чином: Нейрони, присутні в шарі з повним зв'язком, виявляють певну функцію і зберігають її значення, а потім передають значення класам собак і кішок, які потім перевіряють функцію і вирішують, чи є ця функція актуальною для них. [9].

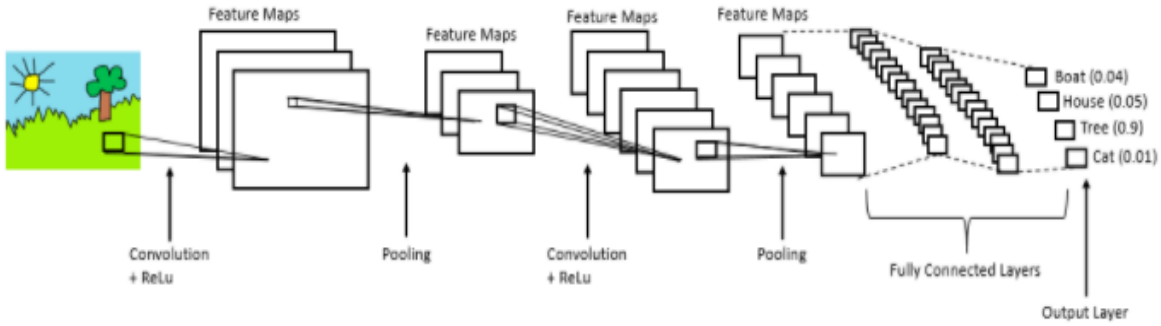

Рисунок 3.3 – Повна архітектура загорткової нейронної мережі

Розглянемо три головні етапи для побудови архітектури ConvNet: згортковий рівень, шар пулу і повністю підключений рівень (в точності як в звичайних нейронних мережах). Ми будемо складати ці шари для формування повної архітектури ConvNet.

Згортка + шар Relu: у цьому шарі разом із операцією згортки застосовуються встановлені фільтри. Згортка — це лінійна операція, яка включає множення вхідної ваги, подібно до традиційної нейронної мережі. Оскільки методика була розроблена для надання двовимірного введення, двовимірний масив ваг, фільтр або ядро перемножуються між масивами вхідних даних. [7]. Це математична операція, яка приймає два входи, такі як матриця зображення і фільтр або ядро (рисунок 3.4).

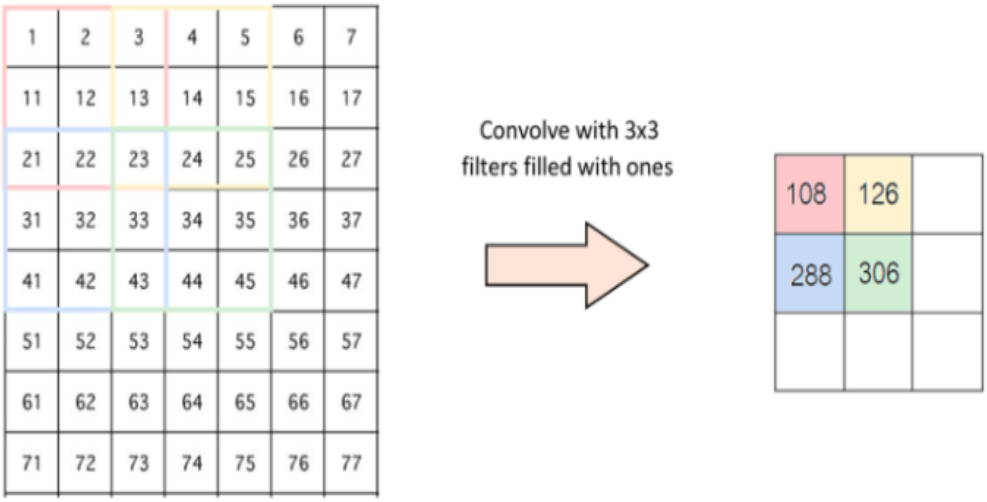

Рисунок 3.4 – Робота шару згортки

Об’єднаний шар: це свого роду нелінійний шар або шар із зниженням дискретизації, який зменшує розмір зображення великого розміру до низькорозмірного. Це полегшує обчислення, а також знижує високі витрати на обчислення [5]. Просторове об'єднання поділяється на декілька типів: максимальний пулінг, середній пул, підсумовування пулу. найбільше об'єднання трапляється коли максимальний елемент остається на мапі об'єктів (рисунок 3.5).

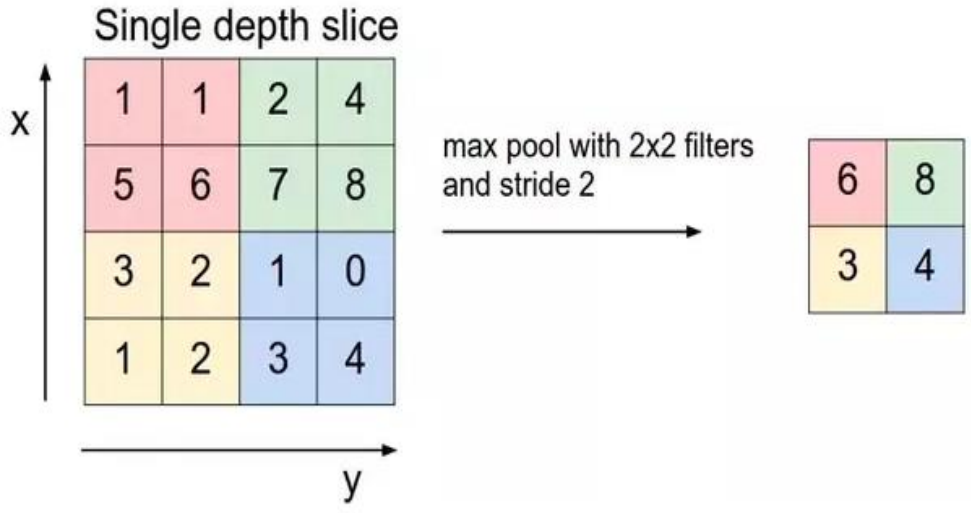

Рисунок 3.5 – Приклад максимального пулінгу

Повністю підключений шар: У цьому шарі кожен вузол підключений до кожного наступного вузла шару. За ним слідував шар softmax або SVM для розподілу ймовірностей. Найвищий розподіл ймовірностей буде розглядатися на вихідному рівні (рисунок 3.6).

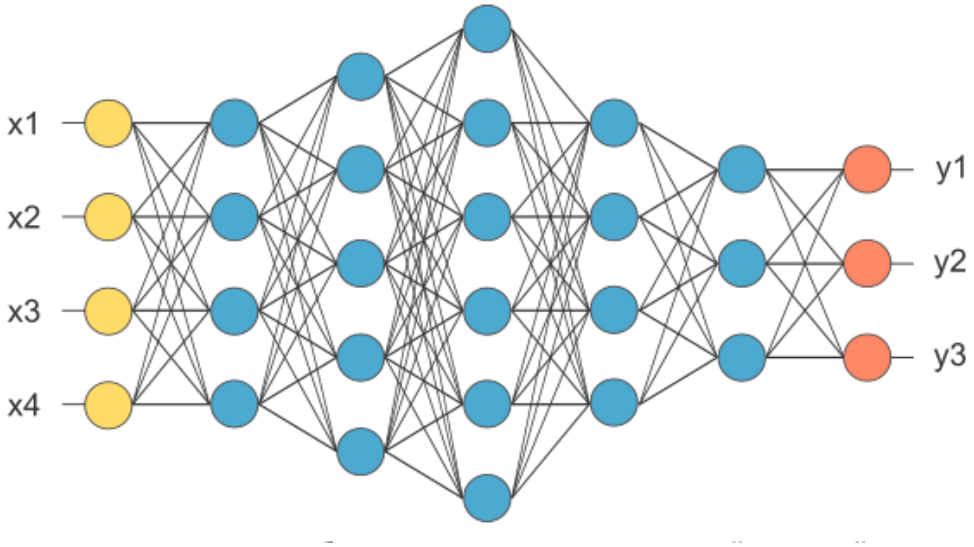

Рисунок 3.6 – Повністю зв’язаний шар

ConvNets оброблює вхідне зображення, шар за шаром, з вхідних значень пікселів у вихідний вектор значень. Буває різні типи шарів. Кожен щар приймає вхідний 3D-об'єм і перетворює його в вихідний 3D-об'єм за допомогою функції, що диференціюється. кожний рівень може мати або не мати параметри, додаткові гіперпараметри.

Найбільш простим і популярним способом (рекомендації) настанова такої нейронної мережі є метод інструкція з учителем (на маркованих даних) – метод зворотного розповсюдження помилки і його модифікації [10].

Опісля навчання нейронна мережа набуває здатності до узагальнення, інакше кажучи починає правильно систематизувати ті напої, які не представлені в навчальній вибірці, інакше кажучи тих, зміст яких ніхто не знає.

3.3 Градієнтний спуск

Градієнтний спуск – алгоритм навчання, він застосовується майже в кожній моделі машинного навчання. Градієнтний спуск - це, по суті, і є те, як навчаються моделі.

Без ГС машинне навчання не було б там, де зараз. Метод градієнтного спуску з деякою модифікацією широко використовується для навчання персептрона та глибоких нейронних мереж , і відомий як метод зворотного розповсюдження помилки. На цьому пості ви знайдете пояснення градієнтного спуску з невеликою кількістю математики. [8].

Метод знаходження мінімального значення функції втрат (є безліч видів цієї функції).

Мінімізація будь-якої функції означає пошук найглибшої западини у цій функції. Майте на увазі, що функція використовується, щоб контролювати помилку у прогнозах моделі машинного навчання. Пошук мінімуму означає отримання найменшої можливої помилки або підвищення точності моделі. Ми збільшуємо точність, перебираючи набір навчальних даних при налаштуванні параметрів нашої моделі (ваг та зсувів).

Отже, градієнтний спуск необхідний мінімізації функції втрат. Суть алгоритму процес отримання найменшого значення помилки. Аналогічно це можна розглядати як спуск у западину у спробі знайти золото на дні ущелини (найнижче значення помилки) (рисунок 3.7).

Коли ми перебираємо всі навчальні дані, продовжуємо додавати значення dJ/dw для кожної ваги. Оскільки втрати залежить від прикладу навчання, dJ/dw також продовжує змінюватися. Потім ділимо зібрані значення кількість прикладів навчання щоб одержати середнього. Потім ми використовуємо це середнє значення (кожної ваги) для налаштування кожної ваги.

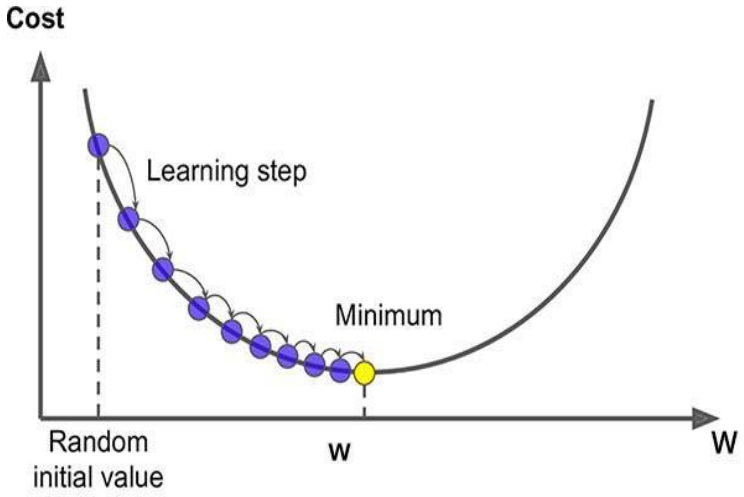

Рисунок 3.7 – Принцип роботи градієнтного спуску

Функція втрат призначена для відстеження помилки з кожним прикладом навчання, тоді як похідна функції відносної однієї ваги – це те, куди потрібно змістити вагу, щоб мінімізувати її для цього прикладу навчання. Ви можете створювати моделі навіть без застосування втрат. Але вам доведеться використовувати похідну щодо кожної ваги (dJ/dw).

Тепер, коли ми визначили напрямок, у якому треба підштовхнути вагу, нам треба зрозуміти, як це зробити. Тут ми використовуємо коефіцієнт швидкості навчання, його називають гіперпараметром.

Гіпер-параметр – значення, необхідне вашою моделлю, про яке ми дійсно маємо дуже невиразне уявлення. Зазвичай ці значення можуть бути вивчені методом спроб та помилок. Тут не так: одне підходить для всіх гіперпараметрів. Коефіцієнт швидкості навчання можна розглядати як «крок у правильному напрямку», де напрямок походить від dJ/dw. Це була функція втрат, побудована на одну вагу. У реальній моделі ми робимо все вище перераховане для всіх ваг, перебираючи всі приклади навчання. Навіть у відносно невеликій моделі машинного навчання у вас буде більш ніж 1 або 2 ваги. Це ускладнює візуалізацію, оскільки графік матиме розміри, які розум не може собі уявити. [7].

Типовим прикладом алгоритмів з коефіцієнтами, які можна

оптимізувати за допомогою градієнтного спуску, є логістична регресія.

3.4 Регресійне дослідження та поліноміальна логістична регресія

Логістична регресія – це статистичний метод для аналізу набору даних, в якому є одна або кілька незалежних змінних, що визначають результат. Результат вимірюється за допомогою дихотомічної змінної (у якій є лише два можливі результати). Він використовується для прогнозування двійкового результату (1/0, Так / Ні, Істина / Брехня) з урахуванням набору незалежних змінних[10].

Ми також можемо розглядати логістичну регресію як особливий випадок лінійної регресії, коли вихідна змінна є категоріальною, де ми використовуємо логарифм шансів як залежну змінну. Простіше кажучи, він передбачає можливість виникнення події шляхом підгонки даних до логіт функція.

Треба пам’ятати, що в деяких випадках залежні змінні можуть мати більше двох результатів, наприклад, у шлюбі / незаміжня / у розлученні, такі сценарії класифікуються як поліноміальна логістична регресія. Хоча вони працюють однаково, щоби передбачити результат (рисунок 3.8).

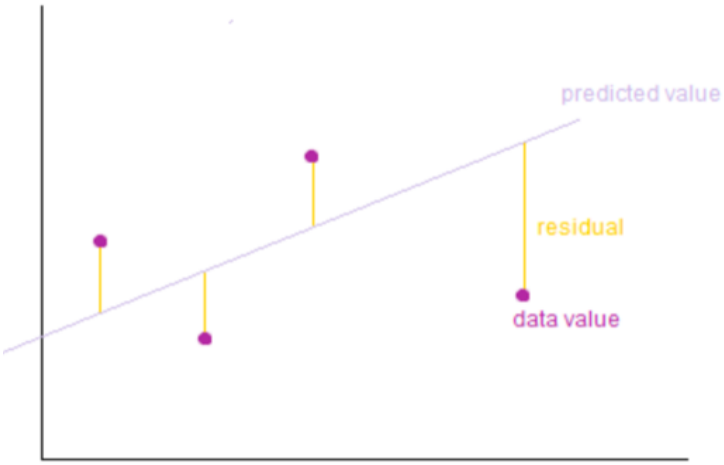

Рисунок 3.8 – Принцип роботи регресійного аналізу

Регресійний аналіз – метод моделювання вимірюваних даних та дослідження їх властивостей. Дані складаються з пар значень залежної змінної (змінної відгуку) та незалежної змінної (що пояснює змінної). Регресійна модель є функція незалежної змінної та параметрів з доданою випадковою змінною . Параметри моделі налаштовуються таким чином, що модель найкраще наближає дані. Критерієм якості наближення (цільовою функцією) зазвичай є середньоквадратична помилка : сума квадратів різниці значень моделі та залежної змінної для всіх значень незалежної змінної як аргумент.

Регресійний аналіз - розділ математичної статистикита машинного навчання . Щодо характеру розподілу цієї величини робляться припущення, які називають гіпотезою породження даних. Для підтвердження чи спростування цієї гіпотези виконуються статистичні тести , які називають аналізом залишків.

При цьому передбачається, що незалежна змінна не містить помилок. Регресійний аналіз використовується для прогнозу , аналізу часових рядів , тестування гіпотез та виявлення прихованих взаємозв'язків у даних [11] (рисунок 3.9).

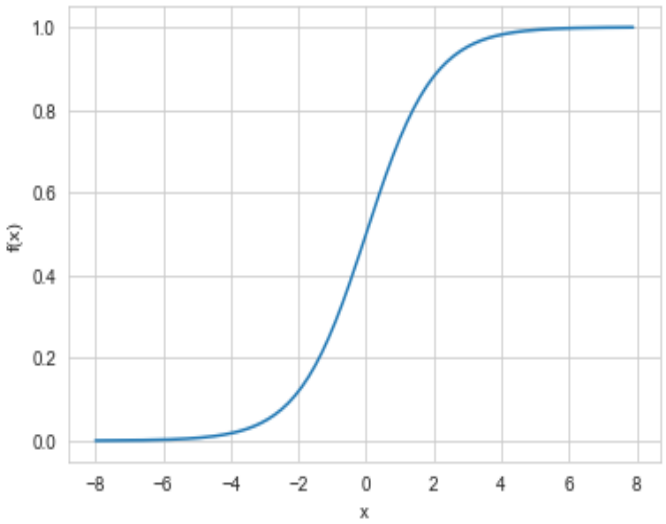

Рисунок 3.9 – Логістична крива

Спробуймо зрозуміти логістичну регресію, розуміючи логістичну модель. Як і у випадку лінійної регресії, давайте представимо нашу гіпотезу (пророцтво залежної змінної) у класифікації. У класифікації наше уявлення гіпотези, яке намагається передбачити двійковий результат або о або 1 [9].

Переваги логістичної регресії:

– широко використовується для задач класифікації;

– не вимагає лінійних відносин між залежними і незалежними змінними, оскільки застосовує нелінійне логарифмічне трансформація до прогнозного відношенню шансів;

– має перспективу використання покрокового методу для оцінки (вимагає великих розмірів вибірки, через те що оцінки максимальної правдоподібності менш ефективні при малих розмірах вибірки).

Представлення логістичної функції, яку ми також називаємо сигмовидною функцією. З наведеного вище візуального уявлення сигмовидної функції ми можемо легко зрозуміти, як ця крива описує багато реальних ситуацій, такі як зростання населення.

На початкових етапах це показує експоненційне зростання, але через деякий час, через конкуренцію за певні ресурси (шийка пляшки), швидкість зростання знижується, поки не досягне тупикової ситуації, і зростання не буде Питання тут у тому, як це логіт (сигмоїдальна функція) допомагає нам визначити можливість класифікації даних за різними класами. Спробуймо зрозуміти, як розраховується наша функція logit, що дасть нам деяку ясність [5].

Передбачається, що співвідношення шансів для будь-яких двох категорій незалежні від усіх інших категорій відповідей. так само для заданого шаблону коваріат, відгуки передбачаються незалежними поліноміальними змінними.

Подібно до бінарної логістичної регресії, поліноміальна логістична

регресія використовує максимальну оцінку ймовірності, щоб оцінити ймовірність категоричного членства, приналежності до якоїсь з груп.

3.5 Алгоритм

У реалізації Image Classifier, для моделювання зв'язків між вхідними та вихідними даними застосовується видозмінена нейронна мережа, в якій відсутній прихований шар. Вимірюється внесок кожного вхідного класу, і в момент закінчення навчання різні ознаки (входи) забезпечуються ваговими коефіцієнтами. Create ML використовує для цього поліноміальну логістичну регресію (рисунок 3.10).

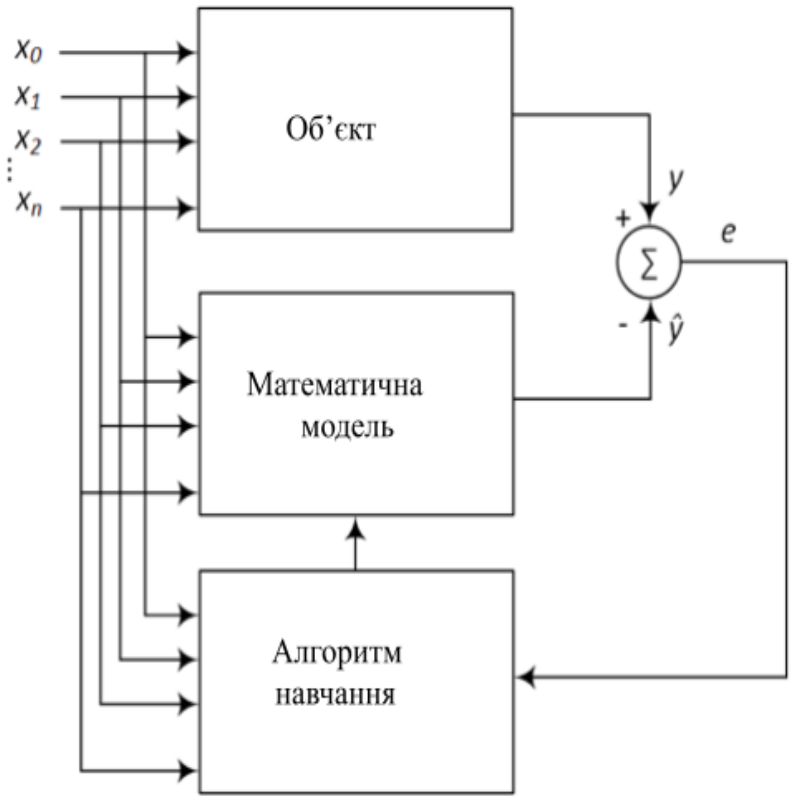

Рисунок 3.10 – Структура поліноміального регресійного аналізу

Тобто в алгоритмі вихідні нейрони використовують сигмоїдальну (логістичну) функцію:

$$\sigma(x) = 1/(1 + exp(-tx)), \quad (3.1)$$

де *t* – параметр функції, що визначає її крутизну.

Різниця між поліноміальною логіт-моделлю та безліччю інших методів, моделей, алгоритмів і т. д. З тією ж базовою настройкою (алгоритм персептрона , машини опорних векторів , лінійний дискримінантний аналіз і т. д.) полягає в процедурі визначення (навчання) оптимальних ваг . / Коефіцієнти та спосіб інтерпретації оцінки. Зокрема, у моделі поліноміального логіту оцінка може бути безпосередньо перетворена на значення ймовірності, що вказує на ймовірність спостереження і, що вибирає результат kз огляду на виміряні характеристики спостереження. Це забезпечує важливий спосіб включення передбачення конкретної поліноміальної логіт-моделі в більшу процедуру, яка може включати кілька таких передбачень, кожне з яких має ймовірність помилки. Без засобів комбінування прогнозів помилки мають тенденцію множитися. [8]. Логістична регресія використовується, тоді як змінна відгуку носить категоріальний характер, при вирішенні задачі класифікації (до якої напої відноситься). Завдання, яке доведеться постановити називається завдання навчання з учителем, складається в пошуку коефіцієнтів a.

Цільова функція такої системи має зовнішність:

$$E(k) = \sum_{k=1}^{N} (e(k))^2 \rightarrow min. \quad (3.2)$$

Даний метод має алгоритм , що представлений в Додатку А.

Робота алгоритму.

Крок 1. Завдання початкових умов для всіх синаптичних ваг мережі у вигляді досить малих випадкових чисел для того, щоб активаційні функції нейронів не ввійшли в режим насичення на початкових стадіях навчання (захист від «паралічу» мережі):

$$w^T = \{0,0..0\}. \quad (3.3)$$

Крок 2. Подаємо на вхід вектор даних про перший вид напою:

$$x(k) = \{1, x_{11}, x_{12}\ldots x_{1n}\}^T. \quad (3.4)$$

Крок 3. Розраховуємо проміжні виходи (строчка на стовпець – скаляр) та локальні помилки:

$$o(k) = w^T x(k), \quad (3.5)$$

$$\delta(k) = -\eth E(k)/\eth u. \quad (3.6)$$

Крок 4. Уточнюємо всі синаптичні ваги:

$$\triangle w = n\delta x. \quad (3.7)$$

Крок 5. Подача на вхід мережі даних наступного класу.

Невідомі параметри в кожному векторі β k зазвичай спільно оцінюються максимальною апостеріорною оцінкою (MAP), яка є розширенням максимальної правдоподібності з використанням регуляризації ваг для запобігання патологічних рішень (звичайно квадрат регулюючої функції, що еквівалентно розміщенню гаусівського апріорного розподілу з ну та інші розподіли). Рішення зазвичай знаходиться з використанням ітераційної процедури, такої як узагальнене ітеративне масштабування, [7] ітеративно зважений метод найменших квадратів (IRLS), [8] за допомогою алгоритму оптимізації на основі градієнта, такі як L-BFGS, [4] або спеціалізовані алгоритми спуску координат .[9]

3.6 Висновки по розділу

Під час огляду існуючих технологій проектування виділили найважливіші такі пункти.

Оптимізація – велика компонент машинного навчання.

Градієнтний спуск – це проста процедура оптимізації, яку ми можемо застосовувати з багатьма алгоритмами машинного навчання.

Регресійний аналіз – це форма техніки прогнозного моделювання, яка досліджує відношення між залежною (цільової) і незалежною змінною (предиктором). Цей метод використовується для прогнозування і формулювання причинно–наслідкового зв'язку між змінними.

Поліноміальний регресійне дослідження зможе гарно робити на нашій кількості напоїв і з достовірною ймовірністю видавати відсоток схожості з різними напоями.

4 ОПИС ОБРАНОГО МЕТОДУ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ ПОБУДОВИ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

Початкова ідея фільтрації інформації була підходом на основі контенту, тобто способом фільтрації на основі смаку страви, який надає перевагу користувачеві. Однак нам довелося створити гібрид фільтрації на основі вмісту та спільної роботи через той факт, що модель Recommender у додатку CreateML більше підходить для останнього типу фільтрації (рисунок 4.1).

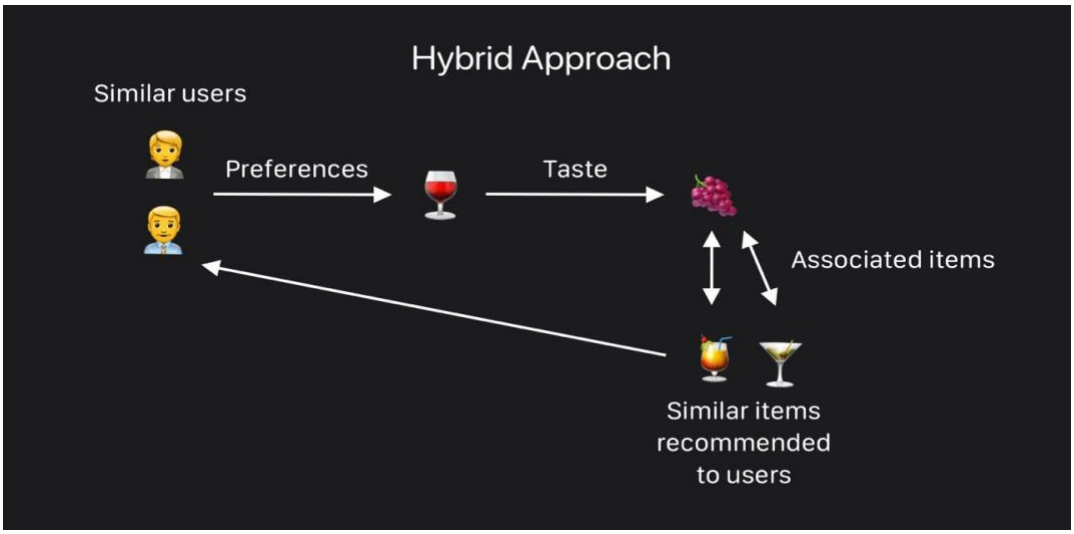

Рисунок 4.1 – Гібридний підхід

Наша перша проблема полягала в тому, що для використання обох методів фільтрації нам довелося «перекласти» дані з подібних елементів на подібні користувачам. У підсумку ми створили власний невеликий набір даних, де ми вирішили замінити звичайного користувача спільної фільтрації категоріями, які представляють смак кожного користувача, так що їхні попередні вподобання коктейлів формуватимуть рекомендаційний стиль фільтрації на основі вмісту.

Для досягнення більшої точності ми оцінили кожну категорію смаку від 1 до 5 на основі того, наскільки кожен конкретний елемент відповідає

оцінці. У цьому прикладі, коли оцінки отримані не від конкретного користувача, дуже важливо зрозуміти, які атрибути ви хочете використовувати у своїй моделі. Наприклад, у нашому проектному навчанні модель, заснована на уподобаннях споживача, була б дуже нечіткою та неефективною.

Перш ніж реалізувати механізм рекомендацій на основі спільної пам’яті, ми повинні спочатку зрозуміти основну ідею такої системи. Для цього механізму кожен елемент і кожен користувач є нічим іншим, як ідентифікаторами. Тому ми не будемо брати до уваги будь-які інші атрибути фільму (наприклад, акторський склад, режисер, жанр тощо) під час створення рекомендацій. Подібність між двома користувачами представлена за допомогою десяткового числа від –1,0 до 1,0. Назвемо це число індексом подібності.

Нарешті, можливість того, що користувачеві сподобається фільм, буде представлено за допомогою іншого десяткового числа від –1,0 до 1,0. Тепер, коли ми змоделювали світ навколо цієї системи, використовуючи прості терміни, ми можемо розкрити кілька елегантних математичних рівнянь, щоб визначити зв’язок між цими ідентифікаторами та числами. У нашому алгоритмі рекомендацій ми будемо підтримувати декілька наборів. Кожен користувач матиме два набори: набір фільмів, які йому подобаються, і набір фільмів, які користувачеві не подобаються. З кожним фільмом також буде пов’язано два набори: набір користувачів, яким фільм сподобався, і набір користувачів, яким фільм не сподобався.

На етапах, на яких генеруються рекомендації, буде створено ряд наборів – переважно об'єднання або перетину інших наборів. Ми також матимемо впорядковані списки пропозицій та подібних користувачів для кожного користувача. Для обчислення індексу подібності ми будемо використовувати варіант формули індексу Жакара. Спочатку відома як «coefficient de communauté» (придумана Полем Жаккардом), формула порівнює два набори і створює просту десяткову статистику від 0 до 1,0:

$$J(A,B) = |A \cap B| \div |A \cup B|. \quad (4.1)$$

Формула передбачає поділ кількості загальних елементів у будь-якому наборі на кількість усіх елементів (врахованих лише один раз) в обох наборах. Індекс Жаккара двох однакових наборів завжди буде 1, тоді як індекс Жаккара двох наборів без спільних елементів завжди дає 0. Тепер, коли ми знаємо, як порівнювати два набори, давайте подумаємо про стратегію, яку ми можемо використовувати для порівняння двох користувачів. Як обговорювалося раніше, користувачі, з точки зору системи, – це три речі: ідентифікатор, набір фільмів, які сподобалися, і набір фільмів, які не сподобалися. Якби ми визначали індекс подібності наших користувачів лише на основі набору фільмів, які їм сподобалися, ми могли б безпосередньо використати формулу індексу Жакара:

$$S(U_1, U_2) = |L_1 \cap L_2| \div |L_1 \cup L_2|. \quad (4.2)$$

Тут U1 і U2 – це два користувача, яких ми порівнюємо, а L1 і L2 – це набори фільмів, які сподобалися U1 і U2 відповідно. Тепер, якщо подумати, двом користувачам, яким подобаються однакові фільми, схожі, то двом користувачам, яким не подобаються ті самі фільми, також повинні бути схожі. Ось де ми трохи модифікуємо рівняння:

$$S(U_1, U_2) = (|L_1 \cap L_2| + |D_1 \cap D_2|) \div |L_1 \cup L_2 \cup D_1 \cup D_2|. \quad (4.3)$$

Замість того, щоб просто розглядати загальні вподобання в чисельнику формули, ми тепер також додаємо кількість поширених лайків. У знаменнику ми беремо кількість усіх елементів, які сподобалися або не сподобалися користувачеві. Тепер, коли ми розглянули як симпатії, так і антипатії в незалежний спосіб, нам слід також подумати про випадок, коли два користувача є полярними протилежностями у своїх

уподобаннях. Індекс подібності двох користувачів, де одному подобається фільм, а іншому не подобається, не повинен бути 0:

$$S(U_1, U_2) = (|L_1 \cap L_2| + |D_1 \cap D_2| - |L_1 \cap D_2| - |L_2 \cap D_1|) \div |L_1 \cup L_2 \cup D_1 \cup D_2|. \quad (4.4)$$

Вона схожа на нашу попередню формулу з невеликою різницею в чисельнику. Тепер ми віднімаємо кількість конфліктуючих лайків і антипатій двох користувачів від кількості їхніх загальних лайків і антипатій. Це призводить до того, що формула індексу подібності має діапазон значень від –1,0 до 1,0. Двоє користувачів, які мають ідентичні смаки, матимуть індекс подібності 1,0, а два користувача, які мають абсолютно протилежні смаки у фільмах, матимуть індекс подібності –1,0.

Тепер, коли ми знаємо, як порівнювати двох користувачів на основі їхніх смаків у фільмах, нам потрібно вивчити ще одну формулу, перш ніж ми зможемо розпочати впровадження нашого саморобного алгоритму рекомендацій:

$$P(U, M) = |Z_L - Z_D| \div (|M_L| + |M_D|). \quad (4.5)$$

Ми маємо на увазі P(U,M)можливість того, що користувачеві Uсподобається фільм M. ZLi ZDє сумою індексів подібності користувача Uз усіма користувачами, яким фільм сподобався чи не сподобався Мвідповідно. |ML|+|MD| представляє загальну кількість користувачів, яким сподобався або не сподобався фільм M. У результаті P(U,M)виходить число від –1,0 до 1,0. Ось і все. У наступному розділі ми можемо використовувати ці формули, щоб почати впроваджувати наш механізм рекомендацій на основі спільної пам’яті.

Ми створили файл CSV з трьома стовпцями: категорія, яка представляє категорію користувачів зі схожим смаком, назва, яка

представляє назву коктейлю, і рейтинг, що показує рейтинг для кожного коктейлю.

Після створення файлу CSV починається власне навчання моделі Recommender. Щоб навчити модель, вам потрібно відкрити Xcode, а потім у меню в лівому верхньому куті вибрати Xcode > Відкрити інструмент розробника > Створити ML, після запуску програми Create ML ви виберете шаблон рекомендацій (рисунок 4.2).

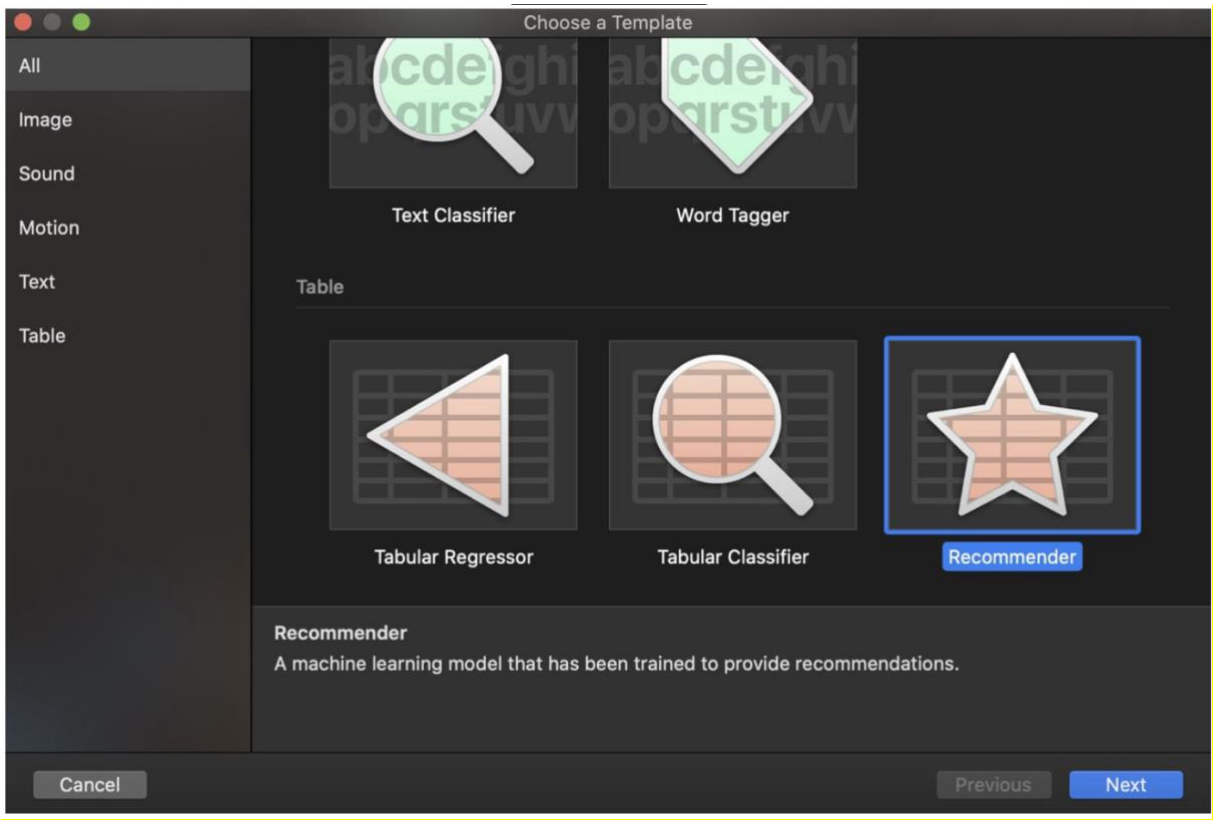

Рисунок 4.2 – Шаблон рекомендацій у програмі Create ML

Далі розміщаємо як вхідний файл CSV, який ви створили – ви навіть можете зробити це за допомогою перетягування – і ви зможете побачити як кількість перевірених елементів, так і загальну кількість прикладів (рисунок 4.3). Потім вам потрібно вибрати ім’я, яке ви використовуєте у своєму файлі для користувача, елемент і, за бажанням, рейтинг.

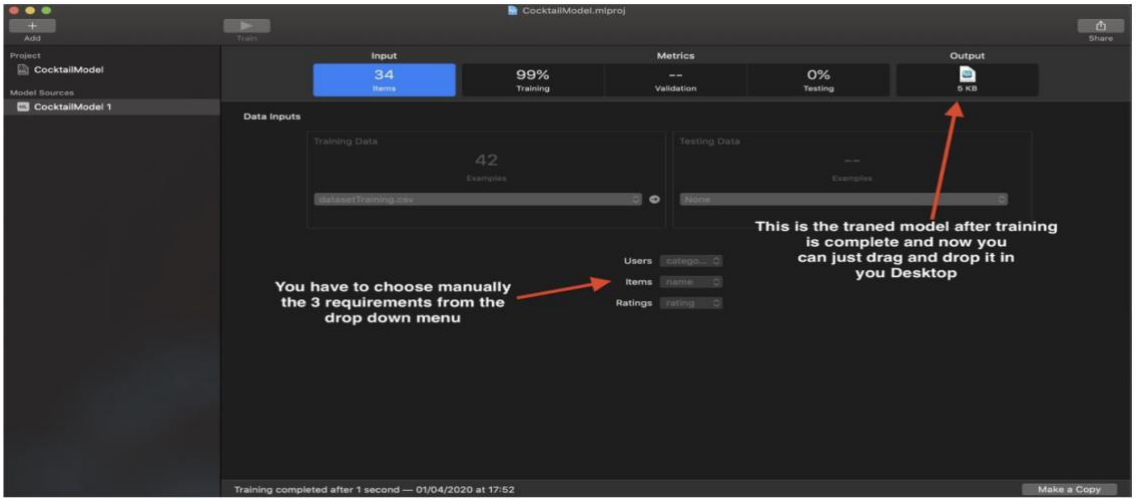

Рисунок 4.3 – Модель готова до використання

Останнім кроком для фактичної перевірки моделі є перехід на панель введення, а потім додавання реального користувача, щоб спробувати різні елементи та перевірити рекомендації, які ви отримуєте (рисунок 4.4).

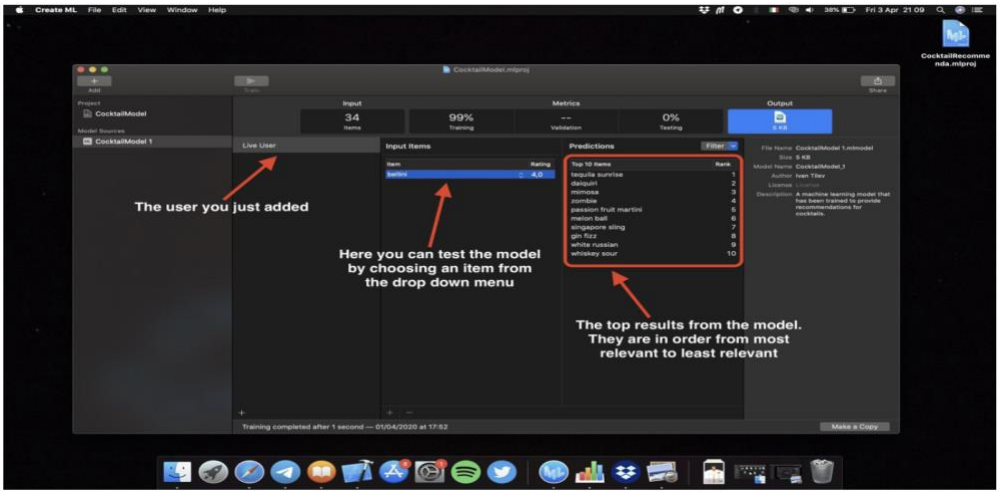

Рисунок 4.4 – Модель готова до використання

5 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

5.1 Технічне завдання

Програма розроблена на мові програмування Swift. Для цього ми використали оточення XCode 13. Також до коду додається датасет за допомогою якого навчалась нейронна мережа. Датасет знаходиться в тій самій папці де знаходиться вихідний код. Для того щоб навчити нейромережу ми використовуємо фреймворк CreateML. Image Classifier у CreateML використовується як шаблон для класифікації картинок за їх змістом. Після навчання моделі, вона має спроможність упізнавати предмет на фото і відносить його до певного класу об’єктів.[32]

Після вибраного з галереї знімка або зробленого на камеру фото, система виводить процент впевненості який є рівнем збігу фото та відомих напоїв.

5.2 Вибір мови програмування та архітектури додатку

Swift – багатопарадигмова компільована мова програмування, розроблена компанією Apple для того, щоб співіснувати з Objective C і бути стійкішою до помилкового коду. Swift була представлена на конференції розробників WWDC 2014[1]. Мова побудована з LLVM компілятором, включеного у Xcode 6 beta. Безкоштовний посібник мови програмування Swift доступний для завантаження у магазині iBooks[2].

Компілятор Swift побудований з використанням технологій вільного проєкту LLVM. Swift успадковує найкращі елементи мов C і Objective-C, тому синтаксис звичний для знайомих з ними розробників, але водночас відрізняється використанням засобів автоматичного розподілу пам'яті і контролю переповнення змінних і масивів, що значно збільшує надійність і безпеку коду.[34]

При цьому Swift-програми компілюються у машинний код, що дозволяє забезпечити високу швидкодію. За заявою Apple, код Swift виконується в 1.3 рази швидше коду на Objective-C. Замість збирача сміття Objective-C в Swift використовуються засоби підрахунку посилань на об’єкти, а також надані у LLVM оптимізації, такі як автовекторизація.

Мова також пропонує низку сучасних методів програмування, таких як замикання, узагальнене програмування, лямбда-вирази, кортежі і словникові типи, швидкі операції над колекціями, елементи функційного програмування. Основним застосуванням Swift є розробка користувацьких застосунків для macOS, iOS, tvOS, watchOS з використанням тулкіта Cocoa і Cocoa Touch. При цьому Swift надає об'єктну модель, сумісну з Objective-C.[14]

Сирцевий код мовою Swift може змішуватися з кодом на C і Objective-C в одному проєкті. Swift щільно інтегровано до власницького середовища розробки Xcode, проте може бути викликано з терміналу, що уможливлює її використання на операційний системах, відмінних від macOS, наприклад, на Linux.[43] Окремо варто відзначити, що Swift від компанії Apple не варто плутати з досить давно розроблюваною скриптовою мовою Swift, націленої на багатонитеве програмування і поставленого під вільною ліцензією Apache.

Рисунок 5.1 – Емблема мови програмування Swift

Apple та IBM об'єднали свої зусилля та разом інвестують у Swift. IBM Cloud надає можливості не тільки розробляти та застосовувати, але також

ділитися ресурсами та використовувати Swift Sandbox для швидких експериментів. Таким чином, Objective-C виявився поза грою. Практично всі зміни, які вносилися до Objective-C останнім часом, робилися для того, щоб спростити імпорт даних у Swift [10]. Swift — компактніша мова програмування. Але це зовсім не означає, що код простий. Іноді він дуже складний. Але в той же час він забезпечує переваги, якими не може похвалитися Objective-C. Синтаксис і будова мови виключають кілька типів помилок, які потенційно можливі у Objective-C. Ця стабільність мови допомагає розробнику уникнути небажаних похибок[21].

Продуктивність Swift наближається до C++. Apple постійно працює над покращенням швидкості Свіфту. Objective-C працює повільніше.

Створений для платформ Apple, Swift потихеньку поширюється на інші платформи. В даний момент він досяг лише Linux. Але прогнозується, що ця мова стане ще портативнішою і переноситиметься на інші платформи. Кажуть, що у планах автора Кріса Латтнера є ідея зробити його сумісним із Android[28].

Інструмент Swift Playgrounds надав нові можливості розробникам. Він дозволяє тестувати код у режимі реального часу без складання великих шматків або всієї програми. Ця функція чудово підходить для експериментів із кодом [7].

Сучасну мову програмування Apple легше зрозуміти розробникам, які не працюють з iOS. Це позитивно впливає на продуктивність та швидкість роботи. Swift надає унікальну можливість - він може бути використаний для всіх цілей програмного забезпечення iOS.

На щорічній конференції розробників WWDC2018 компанія Apple представила інструменти для роботи з машинним навчанням Create ML. Навчена в Create ML модель є наслідком застосування алгоритму машинного навчання набору навчальних даних. Моделі не займають багато місця (близько 3Мб), тому їх можна зберігати у проекті. Спочатку моделі пропонувалося навчати за допомогою Playgrounds у XCode 10 та

підтримувалася робота із зображеннями, текстом та таблицями. При запуску playgrounds проекту необхідно було імпортувати бібліотеку CreateML і запустити MLImageClassifierBuilder (якщо ми навчали модель для класифікації зображень).

MVVM є найновішим з MV(X) патернів, тому будемо сподіватися, що він з'явився з урахуванням усіх проблем, притаманних MV(X). Теоретично Model-View-ViewModel виглядає дуже добре. View і Model вже нам знайомі, як і View Model як посередник. Щоб уникнути проблеми MVC, в світ iOS був введений новий шаблон архітектури Model–View–ViewModel (рисунок 5.2).

MVVM розглядає View Controller як View ; у ньому немає тісного зв'язку між View та Model . Крім того, він робить біндінг як наглядальна версія MVP, але не між View та Model , а між View та View Model . То що таке View Model у середовищі iOS? Це незалежне від UIKit уявлення View та її стану. View Model викликає зміни у Model і самостійно оновлюється з вже оновленою Model . І так як біндінг відбувається між View і View Model , то перша, відповідно, теж оновлюється.

Рисунок 5.2 – Структура шаблону архітектури MVVM

повернемося до нашої оцінки ознак гарної архітектури : розподіл : з нашого крихітного прикладу це неясно, але насправді MVVM View має більше обов'язків, ніж View з MVP.

Тому що перша оновлює свій стан з View Model за рахунок установки

біндингу, тоді як друга спрямовує всі події в Presenter і не оновлює себе (це робить Presenter)

Тестування : View Model не знає нічого про уявлення, це дозволяє нам з легкістю тестувати її. View також можна тестувати, але оскільки вона залежить від UIKit, ви можете просто пропустити це; простота використання : той же обсяг коду, як у нашому прикладі MVP, але в реальному додатку, де вам доведеться направити всі події з View в Presenter та оновлювати View вручну, MVVM буде набагато стрункішою (якщо ви використовуєте біндінги).

MVVM є дуже привабливим патерном, тому що він поєднує в собі переваги вищезгаданих підходів і не вимагає додаткового коду для оновлення View у зв'язку з біндингу на стороні View. Проте тестування все ще знаходиться на хорошому рівні.

5.3 Формування Dataset

Dataset для машинного навчання – це оброблена та структурована інформація у табличному вигляді. Рядки такої таблиці називаються об'єктами , а стовпці – ознаками . Розрізняють 2 види ознак [1] : незалежні змінні - предиктори ; залежні змінні - цільові ознаки , які обчислюються на основі одного або кількох предикторів. Ознакове опис притаманно завдань класифікації, коли є вибірка – кінцеве безліч об'єктів, котрим відомо, яких класів вони ставляться. Класова приналежність інших об'єктів невідома. У процесі машинного навчання будується модель, здатна класифікувати довільний об'єкт із вихідної множини [2] . Практичний сенс завдань класифікації полягає у передбаченні можливих результатів з урахуванням сукупності вхідних змінних, наприклад, діагностика захворювань, попередня оцінка ефективності родовищ з корисними копалинами, кредитний скоринг, розпізнавання мови, прогнозування відтоку клієнтів (Churn Rate) тощо. Залежно від варіанта завдання класифікації, цільова

ознака може виглядати по-різному [1] : один стовпець із двійковими значеннями (1/0, TRUE/FALSE та ін.): двокласова класифікація (binary classification), коли кожен об'єкт належить лише одному класу; кілька стовпців з двійковими значеннями : завдання класифікації з класами, що перетинаються (multi-label classification), коли один об'єкт може належати кільком класам; один стовпець з дійсними значеннями : регресійний аналіз, коли прогнозується одна величина; кілька стовпців з дійсними значеннями : задача множинної регресії, коли прогнозується кілька величин.

Для нашого Dataset ми обрали 10 найрозповсюдженіших коктейлів. Знайшли по 800 фотографій кожного напою, з яких 650 ввійшло в тренувальні показники та 150 – до тестових. Створили десять папок з загальними назвами напоїв англійською мовою та створили з них Dataset (рисунок 5.3).

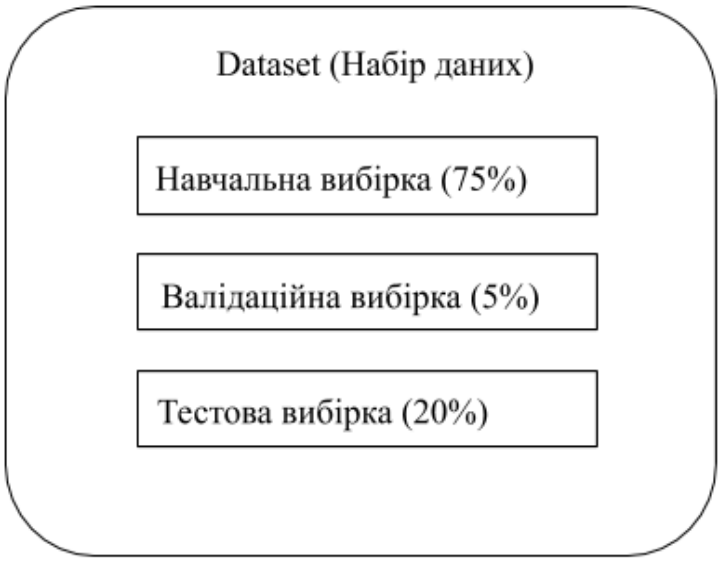

Рисунок 5.3 – Структура набору даних

5.4 Навчання нейронної мережі

Запускаємо інструмент Create ML (XCode > Open Developer Tool > Create ML). Вилазить вікно, де робимо новий проект та вибираємо шаблон

(Image Classifier).Після зберігання проекта ми маємо змогу бачити робоче середовище для роботи з моделлю(рисунок 5.4).

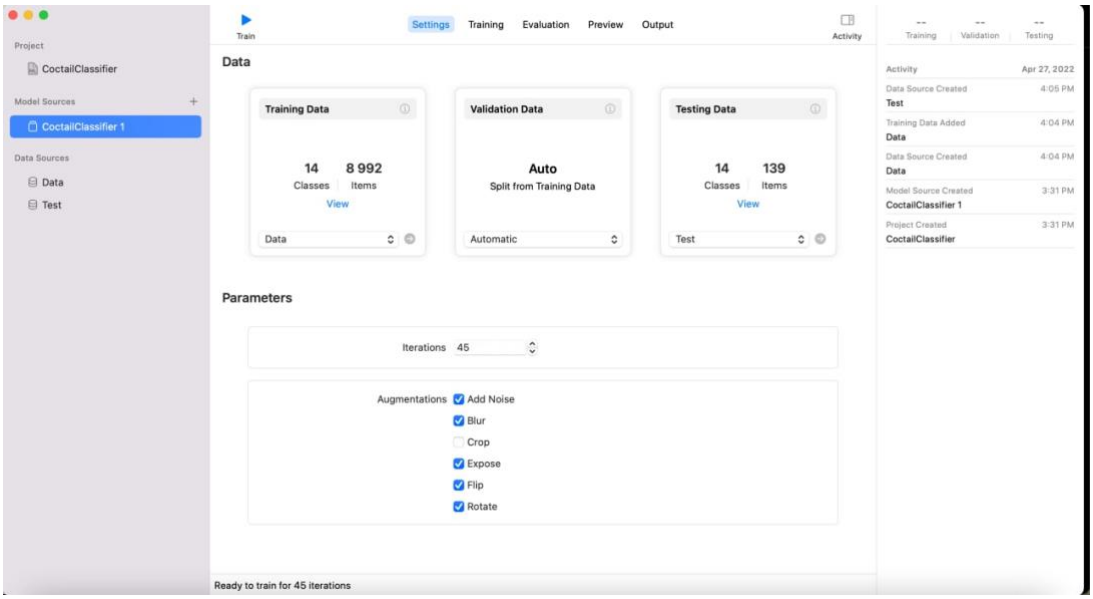

Рисунок 5.4 – Робоче середовище для роботи з навчанням нейронної мережі

В Training Data додаємо зображення для навчання моделі іншими словами це послідовність даних, якими оперує нейронна мережа, Validation Data – комплект зображень для перевірки навчання моделі (залишив Auto), Testing Data – зображення для тестування моделі, які не були задіяні в навчанні моделі.

У розділі Parameters задаємо максимальну частку ітерацій – загальна чисельність тренувальних сетів пройдених нейронною мережею, рівною 25, вироблених над одним зображенням під час навчання моделі. А в розділі Augmentations вибираємо ефекти, що накладаються на зображення та які ми можемо проігнорувати (Blur – розмиття, Flip – віддзеркалене зображення) [14].

Пізніше після навчання оцінюємо, якою мірою модель класифікує зображення з набору Training Data. Оскільки модель навчена на відносно малій кількості зображень, вона не через міру добре класифікує їх. Модель

визначила 82% зображеннь з навчального набору і 74% зображень з перевірочного (рисунок 5.5).

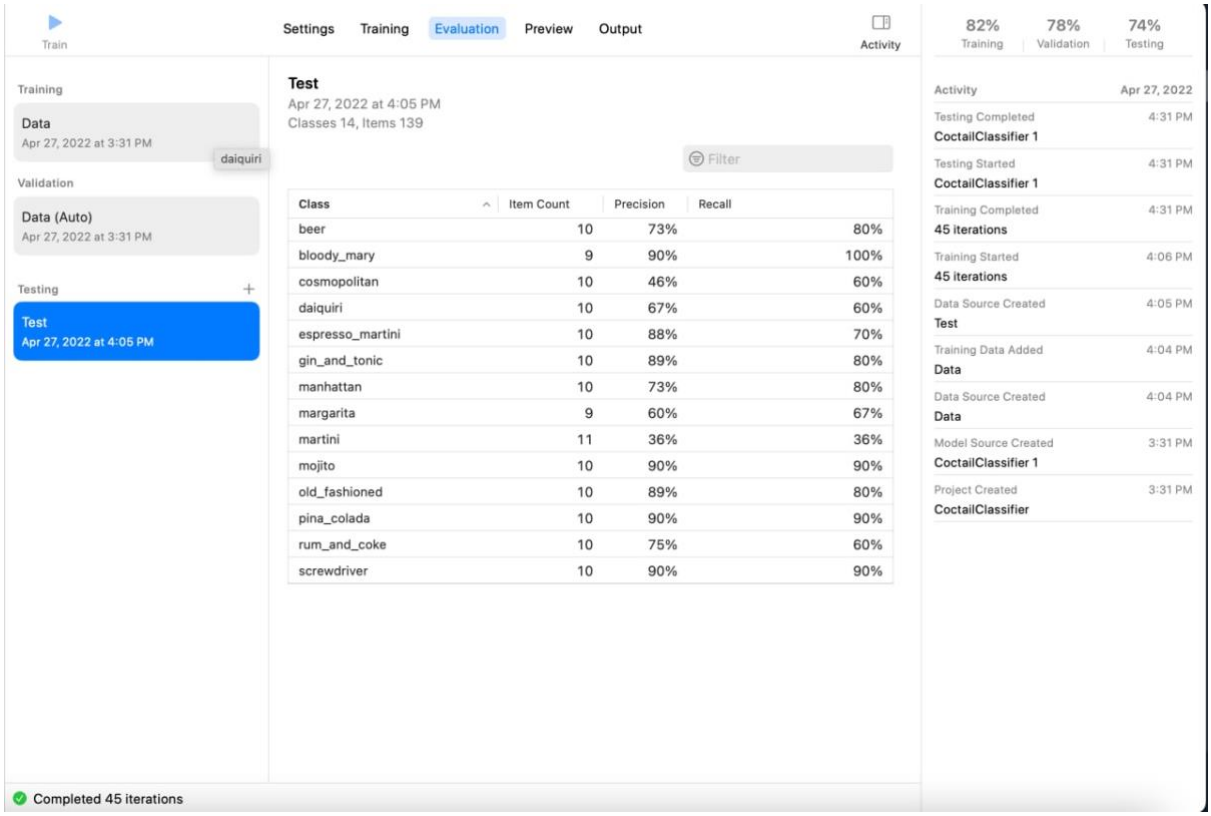

Рисунок 5.5 – Оцінка якості тренувальної моделі

У вкладці Testing натискаємо Test Model. За результатами тестування, модель впоралась на 74% (рисунок 5.6). Бачимо процент успішності розпізнавання різних коктейлів на 150 тестових зображеннях для кожного.

Test
Apr 27, 2022 at 4:05 PM
Classes 14, Items 139

Filter

Class	Item Count	Precision	Recall
beer	10	73%	80%
bloody_mary	9	90%	100%
cosmopolitan	10	46%	60%
daiquiri	10	67%	60%
espresso_martini	10	88%	70%
gin_and_tonic	10	89%	80%
manhattan	10	73%	80%
margarita	9	60%	67%
martini	11	36%	36%
mojito	10	90%	90%
old_fashioned	10	89%	80%
pina_colada	10	90%	90%
rum_and_coke	10	75%	60%
screwdriver	10	90%	90%

Рисунок 5.6 – Оцінка якості моделі після навчання під час тесту

Перейшовши в Output, перевіряємо роботу моделі. Для цього додаємо зображення пива і перевіряємо результат (рисунок 5.7).

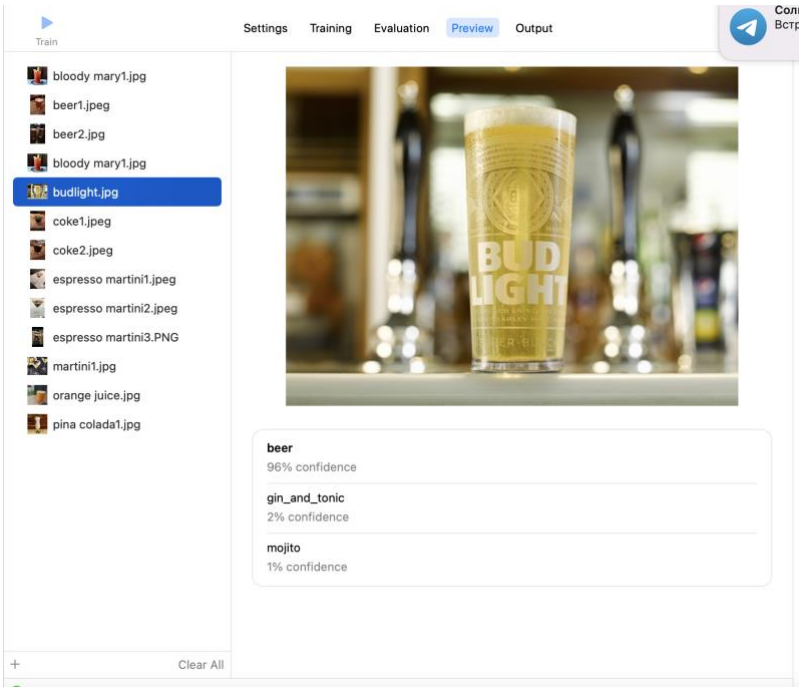

Рисунок 5.7 – Перевірка роботи моделі

Далі інтегруємо нейрону мережу в середовище програмування XCode 11.3.1.

Клас Cocktail відповідає за збіг з категорією коктейлі в нейронній мережі, робимо запит в модель з фотографією і на виході дивимось чи потрапляє результат в наш json, який читаємо з файлу (рисунок 5.8).

```
// The Cocktail Model of the app, that every cocktail instance should have
struct Cocktail {

    var category: String

    var name: String

    var rating: Double

    init(category: String, name: String, rating: Double) {
        self.category = category
        self.name = name
        self.rating = rating
    }
}
```

Рисунок 5.8 – Лістинг коду класу Cocktail

В структурі Prediction (рисунок 5.9) повертається передбачення, яке знаходимо після аналізу напою.

```
// MARK: - Prediction

struct Prediction {

    // MARK: Properties

    let category: String
    let probability: Double

    // MARK: Format Results

    static func topPredictions(_ k: Int, _ prob: [String: Double]) -> String {
        precondition(k <= prob.count)
        let topPredictions = Array(prob.map { x in Prediction(category: x.key, probability: x.value) }
            .sorted(by: { a, b -> Bool in a.probability > b.probability })
            .prefix(through: k - 1))
        return predictionString(predictions: topPredictions)
    }

    static func predictionString(predictions: [Prediction]) -> String {
        var s: [String] = []
        for (i, prediction) in predictions.enumerated() {
            s.append(String(format: "%d: %@ (%3.2f%%)", i + 1, prediction.category, prediction.probability * 100))
        }
        return s.joined(separator: "\n\n")
    }
}
```

Рисунок 5.9 – Лістинг коду структури Prediction

Клас ImageClassifier класифікатор зображень має методи классифікації, що реалізовуються в функціях classifyImageWithVision, classifyImageWithCoreML (рисунок 5.10).

```
15
16  func classifyImageWithVision(image: CGImage, completionHandler: @escaping (String) -> Void) {
17      guard let visionModel = try? VNCoreMLModel(for: studentCatModel.model) else {
18          completionHandler("could not create vision model from coreml model")
19          return
20      }
21
22      let request = VNCoreMLRequest(model: visionModel) { request, error in
23          if let observations = request.results as? [VNClassificationObservation] {
24              let top3 = observations
25                  .prefix(through: 2)
26                  .map { Prediction(category: $0.identifier, probability: Double($0.confidence)) }
27              completionHandler(Prediction.predictionString(predictions: top3))
28          }
29      }
30
31      let handler = VNImageRequestHandler(cgImage: image)
32      try? handler.perform([request])
33  }
34
35  // MARK: Legacy Image Classification
36
37  func classifyImageWithCoreML(image: UIImage, completionHandler: @escaping (String) -> Void) {
38      // NOTE: Do any pre-processing on background thread
39      DispatchQueue.global(qos: .background).async {
40          guard let normalizedImage = image.resize(newSize: CGSize(width: 224, height: 224)), let imageData = normalizedImage.pixelBuffer() else {
41              completionHandler("preprocessing failed")
42              return
43          }
44
45          // NOTE: Format results
46          if let prediction = try? self.studentCatModel.prediction(image: imageData) {
47              completionHandler(Prediction.topPredictions(5, prediction.classLabelProbs))
48          } else {
49              completionHandler("prediction failed")
50          }
51      }
52  }
53 }
```

Рисунок 5.10 – Лістинг коду класу ImageClassifier

3.4 Опис призначеного для користувача інтерфейсу

Додаток реалізовано англійською мовою, бо вірішено що він буде орієнтований на багато країн. Назва додатку – CoctailPhotoRecomendation. Логотип додатку який має такий вигляд як представлено на рисунку 5.11.

Рисунок 5.11 – Іконка мобільного додатку CoctailPhotoRecomendation

Управління стеком навігації проходить за допомогою використання навігаційного контролера, який підтримує перехід між екранами, Отже він керує повність відображенням інформації користувачу.

Коли показується нове вікно, навігаційний бар повідомляє про це користувача, щоб він міг розуміти своє знаходження у ієрархії додатка.

Реалізовані кнопки вперед і назад дають користувачу спроможність керувати навігацією мобільного додатка. Кнопки знаходяться у панелі навігації. (рисунок 5.12).

Рисунок 5.12 – Стек навігації мобільного додатку

Управляючий контролер ClassifyViewController, який відповідає за знаходження напою завдяки знімку камери чи у реальному часі, робить це за допомогою функцій @IBAction func. (рисунок 5.13).

```swift
import UIKit
import Photos

// MARK: - ClassifyVC: UIViewController

class ClassifyViewController: UIViewController {

    @IBOutlet weak var textView: UITextView!
    @IBOutlet weak var imageView: UIImageView!
    // MARK: Properties
    var viewmodel: ClassifyViewModelType!
    var router: NavigationRouter!

    private let classifier = ImageClassifier()

    // MARK: Life Cycle

    override func viewDidLoad() {
        super.viewDidLoad()
    }

    func configure(router: NavigationRouter, viewmodel: ClassifyViewModelType) {
        self.router = router
        self.viewmodel = viewmodel
    }
    override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
        super.prepare(for: segue, sender: sender)
        router.prepare(for: segue, sender: sender)
    }
    @IBAction func videoButtonPressed(_ sender: Any) {
        present(ClassifyVideoVC(), animated: true, completion: nil)
    }
    @IBAction func cameraButtonPressed(_ sender: Any) {
        openImagePicker(sourceType: .camera)
    }
    @IBAction func galleryButtonPressed(_ sender: Any) {
        openImagePicker(sourceType: .photoLibrary)
    }
}
```

Рисунок 5.13 – Лістинг коду класу ClassifyViewController

Функція ImagePickerController відкриває камеру чи галерею зображена на рисунку 5.14.

```swift
// MARK: - ClassifyVC: UIImagePickerControllerDelegate, UINavigationControllerDelegate

extension ClassifyViewController: UIImagePickerControllerDelegate, UINavigationControllerDelegate {
    func imagePickerController(_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [String : Any]) {
        if let image = info[UIImagePickerControllerOriginalImage] as? UIImage, let cgImage = image.cgImage {
            imageView.alpha = 1.0
            imageView.backgroundColor = nil
            imageView.layer.cornerRadius = 0
            imageView.image = image
            textView.text = "Processing image for classification..."
            classifier.classifyImageWithVision(image: cgImage) { [weak self] (results) in
                DispatchQueue.main.async {
                    self?.textView.text = results
                }
            }
        }
        dismiss(animated: true, completion: nil)
    }

    func imagePickerControllerDidCancel(_ picker: UIImagePickerController) {
        dismiss(animated: true, completion: nil)
    }

    // MARK: Helpers

    func openImagePicker(sourceType: UIImagePickerControllerSourceType) {
        let picker = UIImagePickerController()
        picker.sourceType = sourceType
        picker.delegate = self
        present(picker, animated: true, completion: nil)
    }
}
```

Рисунок 5.14 – Лістинг коду функції ImagePickerController

Клас TableViewCell, який відповідає за відображення таблиці з коктейлями (рисунок 5.15) та клас CocktailsTableViewController – єкрану деталей напою.

```swift
        cell.detailTextLabel?.text = "Rating: \(cocktail.rating)"

        return cell
    }

    // Title for the Header of each Section:
    override func tableView(_ tableView: UITableView, titleForHeaderInSection section: Int) -> String? {

        let headerTitles = cocktails[section]

        if section < headerTitles.count {
            return headerTitles[section].category
        }

        return nil
    }

    // MARK: - Navigation

    // Did Select Row At:
    override func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {

        userCocktail = cocktails[indexPath.section][indexPath.row]

        if let viewController = storyboard?.instantiateViewController(identifier: "toRecommender") as?
            ViewController {
            viewController.cocktailName = userCocktail.name
            viewController.cocktailRating = userCocktail.rating
            navigationController?.pushViewController(viewController, animated: true)
        }
    }
```

Рисунок 5.15 – Лістинг коду класу CocktailsTableViewController

Впроваджуючи цей функціонал: ми змогли добитися співпадіння додатку з поставленим технічним завданням, та має увесь набір поставлених функцій.

3.5 Висновки по розділу

В розділі було:

– поставлено технічне завдання;

– вибрано мову програмування Swift з архітектурою MVVM;

– створено Dataset з двома тисячами фотографій десяти різних напоїв;

– інтегровано його в нейронну мережу в середовищі програмування XCode 11.3.1;

– створено інтерфейс додатку мовою Swift з урахуванням всіх критеріїв технічного завдання.

ВИСНОВКИ

В результаті виконання роботи було проаналізовано основні алгоритми та базові підходи для прогнозування рекомендацій. Було описано прикмети переваги та недоліки кожного з них та виконано порівняльний аналіз існуючих підходів. Також було зроблено огляд допоміжних засобів для кореляції результатів, отриманих після вжиття основних алгоритмів.

Алгоритми колаборативної фільтрації дозволяють чимдужче утворювати та інтегрувати рекомендаційні системи, адже вони засновані тільки на пошуку схожих користувачів. Тому колаборативна фільтрація найпоширеніша в інтернет-сервісах, що використовують рекомендаційні системи. Такі системи зазвичай підходять для невеликих блогів, інформаційних порталів тощо, бо в цих випадках характерні проблеми для рекомендаційних систем не матимуть широкого впливу.

Алгоритми контентної фільтрації прогнозують рекомендації завдяки створенню моделей користувачів та товарів. Рекомендаційні системи засновані на цьому підході потребують значно сильніше ресурсів для створення, ніж ті, що використовують колаборативну фільтрацію. Але вони не мають проблем холодного старту, унікальних користувачів, шахрайства тощо. Такі підходи зазвичай використовуються для створення інтернет-магазинів, де рекомендації товарів важливо підбирати під конкретного користувача.

Гібридні алгоритми мають на меті знизити вагу звичних для колаборативної та контентної фільтрації проблем. Але такі системи мають застосовуватися виправдано, оскільки вони громіздкі, на їх побудову необхідно витратити велику кількість часу, коштів та людських ресурсів. Вони 96 зазвичай використовуються в крупних інтернет-магазинах та соціальних мережах.

Були розглянуті шляхи до подолання типових недоліків базових алгоритмів. Для вирішення недоліків колаборативної фільтрації

використовується алгоритм SVD. Адже з ростом кількості користувачів та їх оцінок в будь-якому інтернет-сервісі буде зростати розмірність матриці оцінок. Цей алгоритм дозволяє зменшити розмірність цієї матриці з невеликою втратою точності. Щоб пришвидшити роботу алогоритмів контентної фільтрації зазвичай використовують алгоритм TF–IDF, який дозволяє споруджати рейтингові списки документів великого корпусу документів.

В якості демонстрації проведеного аналізу базових алгоритмів було реалізовано мобільний додаток із інтегрованою рекомендаційною системою на базі колаборативної фільтрації. Платформою для створення інтернет-сервісу було обрано фреймворк Core ML, який дозволяє ефективно створювати проекти високої складності з використанням різних модулів (внутрішніх та зовнішніх). Вбудована рекомендаційна система на основі колаборативної фільтрації має на меті збільшити залученість користувачів до сервісу та збільшити його конверсію.

Отримані результати від створеної рекомендаційної системи виявилися задовільними: прогнозовані рекомендації виявилися релевантними, а алгоритм пошуку схожих користувачів досить точно виявляв групу схожих користувачів. Так як в даному проекті було реалізовано веб-додаток з рекомендаційною системою на базі колаборативної фільтрації, в подальшому можна виготовити сильніше глибоке та детальне вивчення дужче ефективних способів реалізації, а саме – гібридних алгоритмів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Aggarwal Charu C. Recommender Systems: The Textbook. Springer, 2016. 498p.

2. Falk K. Practical Recommender Systems, Manning Publications Co. 2019. 432 p.

3. Recommendation systems: Principles, methods and evaluation URL: https://www.sciencedirect.com/science/article/pii/S1110866515000341 (Last accessed: 22.03.2022).

4. Article Recommendation System with Machine Learning URL: https://thecleverprogrammer.com/2021/11/10/article-recommendation-system-with-machine-learning/ (Last accessed: 16.03.2022).

5. F. Ricci, L. Rokach, S. Bracha. Recommender Systems Handbook. New York: Springer, 2015. 1003 p.

6. Statistical Analysis of K-Nearest Neighbor Collaborative Recommendation URL: https://arxiv.org/pdf/1010.0499.pdf (Last accessed: 20.03.2022).

7. Montaner, M., López, B., de la Rosa, J.L.: A taxonomy of recommender agents on the internet. Artificial Intelligence Review 19(4), 285–330 (2003) (Last accessed: 11.03.2022).

8. Query Dependent Ranking Using K-Nearest Neighbor URL: https://www.andrewoarnold.com/fp025-geng.pdf (Last accessed: 20.03.2022).

9. Снитюк В. Є. Прогнозирование. Модели, методы, алгоритмы: учебное пособие. Київ: Маклаут, 2008. 364 с.

10. Similarity Measures used in Recommender Systems: A Study URL: https://pdfs.semanticscholar.org/943a/e455fafc3d36ae4ce68f1a60ae4f85623e2a.pdf (Last accessed: 26.03.2022).

11. O’Connor M. PolyLens: A Recommender System for Groups of Users URL: http://files.grouplens.org/papers/poly-camera-final.pdf (Last accessed: 25.03.2022).

12. Matrix Factorization Techniques for recommender systems URL: https://datajobs.com/data-science-repo/Recommender-Systems-[Netflix]. pdf (Last accessed: 10.03.2022).

13. A Singularly Valuable Decomposition: The SVD of a Matrix URL: https://datajobs.com/data-science-repo/SVD-[Dan-Kalman].pdf (Last accessed: 04.03.2022).

14. Large-Scale Machine Learning with Stochastic Gradient Descent URL: https://leon.bottou.org/publications/pdf/compstat-2010.pdf (Last accessed: 08.03.2022).

15. Collaborative Filtering for Implicit Feedback Datasets URL: http://yifanhu.net/PUB/cf.pdf (Last accessed: 10.03.2022).

16. Онлайн курс “Навчання на розмічених даних” URL: https://www.coursera.org/learn/supervised-learning?specialization=machi ne-learning-data-analysis (дата звернення: 20.03.2022).

17. A Survey on Decision Tree Algorithms of Classification in Data Mining
URL:https://www.researchgate.net/publication/324941161_A_Survey_on_Decision_Tree_Algorithms_of_Classification_in_Data_Mining (Last accessed: 30.03.2022).

18. XGBoost: A Scalable Tree Boosting System URL: https://arxiv.org/pdf/1603.02754.pdf (Last accessed: 10.03.2022).

19. CatBoost: unbiased boosting with categorical features URL: https://arxiv.org/pdf/1706.09516.pdf (Last accessed: 20.03.2022).

20. Megogo Kaggle Challenge URL : https://www.kaggle.com/c/megogochallenge/overview 05.06.2019). (Last

accessed: 24.03.2022).

21. J. Bobadilla, F. Ortega, A. Hernando, and A. Guti´errez, “Recommender systems survey,” Knowledge-Based Systems, vol. 46, 2013 – P. 109-132.

22. Adomavicius G. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions / G. Adomavicius, A. Tuzhilin // IEEE Transactions on Knowledge and Data Engineering – 2005. – Vol. 17, No6. – P. 734- 749.

23. В.В. Куриленко Разработка вэб-приложения музыкального аудиостриминга с рекомендательной системой // НИ ТГУ – 2017 – P. 7-15.

24. G. Salton Automatic Text Processing // Addison-Wesley. – 1989 – 530 pages.

25. Daniar Asanov (n.d.) Algorithms and Methods in Recommender Systems, Berlin, Germany: Berlin Institute of Technology – 2011 – P. 1-7.

26. Jones, M. T. (2013, December 12). Recommender systems.Introduction to approaches and algorithms. Retrieved November 25, 2017 URL: https://www.ibm.com/developerworks/library/os-recommender1/ (Last accessed: 20.03.2022).

27. Jerold Angelus Grundy Newbrain // Duct Publishing – 2012 – 120 pages.

28. D. Goldberg Using Collaborative Filtering to Weave an Information Tapestry / D. Goldberg, D. Nichols, B.M. Oki, D. Terry // Comm. ACM. – 1992. – Vol. 35, No12. – P. 61-70.

29. Gleb Beliakov, Tomasa Calvo and Simon James. Aggregation of preferences in recommender systems / Gleb Beliakov and Simon James // School of Information Technology, Deakin University – 2008 – P. 705-735.

30. J. Basilico and T. Hofmann. Unifying collaborative and content-based filtering. In Proceedings of the 21th International Conference on Machine

Learning, 2004 – P. 9-16 11. C.C. Aggarwal Horting Hatches an Egg: A New Graph-Theoretic Approach to Collaborative Filtering / C.C. Aggarwal, J.L. Wolf, K-L. Wu, P.S. Yu // Proc. Fifth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining. – 1999 – P. 201-212.

31. P. Resnick GroupLens: An Open Architecture for Collaborative Filtering of Netnews / P. Resnick, N. Iakovou, M. Sushak, P. Bergstrom, J. Riedl // Proc. 1994 Computer Supported Cooperative Work Conf. – 1994 – P. 175-186.

32. U. Shardanand Social Information Filtering: Algorithms for Automating "Word of Mouth" / U. Shardanand, P. Maes // Proc. Conf. Human Factors in Computing Systems. – 1995 – P. 210-217.

33. J.S. Breese Empirical Analysis of Predictive Algorithms for Collaborative Filtering / J.S. Breese, D. Heckerman, C. Kadie // Proc. 14th Conf. Uncertainty in Artificial Intelligence. – 1998 – P. 43-52.

34. B. Sarwar Item-Based Collaborative Filtering Recommendation Algorithms / B. Sarwar, G. Karypis, J. Konstan, J. Riedl // Proc. 10th Int'l WWW Conf. – 2001 – P. 285- 295.

35. J. Delgado Memory-Based Weighted-Majority Prediction for Recommender Systems / J. Delgado, N. Ishii // Proc. ACM SIGIR '99 Workshop Recommender Systems: Algorithms and Evaluation. – 1999 – P. 186-198.

36. A. Nakamura Collaborative Filtering Using Weighted Majority Prediction Algorithms / A. Nakamura, N. Abe //Proc. 15th Int'l Conf. Machine Learning. – 1998 – P. 395-403.

37. I. Soboroff Combining Content and Collaborationin Text Filtering / I. Soboroff, C. Nicholas // Proc. Int'l Joint Conf. Artificial Intelligence Workshop: Machine Learning for Information Filtering. – 1999 – P. 86-91.

38. M. Claypool Combining Content-Based and Collaborative Filters in an Online Newspaper / M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, M. Sartin // Proc. ACM SIGIR '99 Workshop Recommender Systems:

Algorithms and Evaluation, Aug. – 1999.

39. D. Billsus User Modeling for Adaptive News Access / D. Billsus, M. Pazzani // User Modeling and User-Adapted Interaction. – 2000. – Vol. 10, No2, No3. – P. 147-180. 21. A.I. Schein Methods and Metrics for Cold-Start Recommendations / A.I. Schein, A. Popescul, L.H. Ungar, D.M. Pennock // Proc. 25th Ann. Int'l ACM SIGIR Conf. – 2002 – P. 253-260.

40. А. В. Заболеева-Зотова Латентный семантический анализ: новые решения в Internet / А. В. Заболеева-Зотова, А. Ю. Пастухов, П. В. Сердюков, Н. А. Козлова, С. А. Чернов // Информационные технологии. – 2001 – Р. 67-82.

41. M. Pazzani A Framework for Collaborative, Content-Based, and Demographic Filtering // Artificial Intelligence Rev. – 1999. – P. 393-408.

42. M. Balabanovic Fab: Content-Based, Collaborative Recommendation / M. Balabanovic, Y. Shoham // Comm. ACM. – 1997. – Vol. 40, No3. – P.66-72.

43. P. Melville Content-Boosted Collaborative Filtering for Improved Recommendations / P. Melville, R.J. Mooney, R. Nagarajan // Proc. 18th Nat'l Conf. Artificial Intelligence. – 2002 – P. 187–192.

44. Рябова Н.В., Чекалкін П.О. Методи побудови рекомендаційних систем на основі класифікації об'єктів за їх зображеннями. Матеріали XII Міжнародної науково-технічної конференції «Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління». Баку-Харків-Жиліна, 27-28 квітня 2022.