НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ ТЕЛЕКОМУНІКАЦІЙ І ГЛОБАЛЬНОГО ІНФОРМАЦІЙНОГО ПРОСТОРУ

Кваліфікаційна наукова
праця на правах рукопису

Купрін Олексій Миколайович

УДК 004.021

ДИСЕРТАЦІЯ

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ РЕКОМЕНДАЦІЙНОЇ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ

122 – «Комп’ютерні науки»

Галузь знань 12 – Інформаційні технології

Подається на здобуття ступеня доктора філософії

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

О.М. Купрін

Наукові керівники:
Гуляєв Кирило Дмитрович
кандидат технічних наук,
старший науковий співробітник.
Кряжич Ольга Олександрівна
кандидат технічних наук,
старший науковий співробітник.

Київ – 2023

АНОТАЦІЯ

Купрін О.М. Інформаційна технологія рекомендаційної підтримки прийняття рішень. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 122 «Комп'ютерні науки» – Інститут телекомунікацій і глобального інформаційного простору Національна академія наук України, Київ, 2023.

Дисертаційна робота присвячена розробці моделей та методів рекомендаційної підтримки прийняття рішень для збору і обробки інформації із різнорідних джерел з метою створення рекомендації на основі вивчення уподобань споживача.

У вступі обґрунтовано актуальність теми, розглянуто зв'язок роботи з науковими темами та актуальним напрямком наукових досліджень, сформульовані мета та задачі дослідження, розкрито наукову новизну та практичну цінність.

У першому розділі проведено огляд моделей методів та підходів, що використовуються при розробці інформаційних технологій рекомендаційної підтримки рішень. Виконано загальний огляд рекомендаційних систем та їх видів, включаючи аналіз вмісту, колаборативне фільтрування, гібридні та методи на основі популярності. На цій основі поглиблено досліджені Content-based алгоритми та алгоритми колаборативної фільтрації. Це дозволило краще розуміти принципи формування нових підходів до створення моделей рекомендаційних механізмів та їх алгоритмізацію. В результаті дослідження було зроблено висновок щодо необхідності розробки алгоритму, який комбінуватиме моделі різних типів в рекомендаційних системах гібридного типу на основі вмісту та колаборативного фільтрування.

В процесі теоретичного дослідження наукових джерел було виявлено проблему нестабільної продуктивності впроваджених моделей

рекомендаційних систем. На основі цього було запропоновано варіанти вирішення цієї проблеми шляхом використання певних визначних факторів, що приваблюють користувача при виборі якогось продукту, але при тому можуть не бути визначальними, щоб остаточно здійснити покупку.

У підсумку аналізу теоретичних джерел були запропоновані варіанти вирішення проблем нестабільної продуктивності та вдосконалення роботи неоднорідних моделей відкрили шляхи для подальших досліджень та розвитку в цій області, а також дозволили обрати та довести методику дослідження, що відповідає поставленим задачам, вибрати підходи, методи та засади для вирішення поставлених у роботі задач.

У другому розділі проведено дослідження особливостей структурування процесів рекомендаційної підтримки, що базуються на маркерах користувача. Під маркером користувача розуміються певний визначні фактори, які є пріоритетними при виборі окремого продукту споживачем на визначеному відрізку часу. Користувач при виборі продукту має свій особистий інтерес та вирішує певну проблему, яка може бути виражена, як послідовність взаємопов'язаних кроків. Користувач при цьому дбає не стільки про формулювання кінцевого рішення, скільки про те, що пов’язане з цим процесом і у підсумку з нього випливає. Для вирішення проблеми потрібно мати вибір варіантів рішення.

Якщо виходити з визначення рекомендаційної системи, то подібні системи є підкласом систем фільтрації інформації, що дозволяють побудувати певний рейтинг за запитами чи уподобаннями. Аналізуючи процеси підтримки прийняття рішень, можна зрозуміти, що в основу покладено наукові методи оцінки результатів праці, поведінки, успішності та ранжування за рядом показників. І саме це є основою для побудови алгоритмів системи рекомендацій.

Коли користувач багато працює в мережі Інтернет і переглядає занадто великі обсяги інформації, створюється система з обмеженнями за знаннями і часом. Якщо при цьому приходиться вирішувати велику кількість завдань, то

формулюється велика кількість альтернатив, що може призвести до невірного вибору і виникнення критичної ситуації. Тобто, замість пошуку найкращого можливого рішення користувачі повинні перебирати альтернативи тільки до тих пір, поки не виявиться така, яка задовольнить певному прийнятому мінімальному стандарту. Тому на практиці, найчастіше, замість пошуку оптимального рішення користувачі вибирають рішення, яке дозволить зняти проблему.

Для виконання задач роботи проведене структурування процесів рекомендаційної системи при різних інтересах користувачів з використанням n-мірних матриць з метою подальшої реалізації алгоритмів за допомогою опенсорсної бібліотеки TensorFlow. Враховано, що алгоритми повинні передбачати реалізацію ситуації за певними маркерами користувача, тобто, ключових ознаках для користувача за якими він приймає рішення, з врахуванням різноманітних змін, обмежень та персональних дій користувача на визначений момент часу. За підсумками викладеного розроблена лінійна модель рекомендацій за декількома маркерами користувача, при якій створено перехід від вирішення задачі за допомогою теорії ігор до алгоритмізації процесів за допомогою оціночної моделі. Проведений розрахунок довів адекватність моделі, побудованої за двома визначеними маркерами користувача.

В роботі була досліджена система із стаціонарним станом – технічна систему великих торгівельних майданчиків на Інтернет-платформах (Розетка, Prom, OLX; книжкові магазини: Книгарня «Є», YAKABOO, КСД та інші). Особливістю цих торгівельних майданчиків є те, що за умови реєстрації або ID користувача формується база попередніх переглядів, на основі чого рекомендуються аналогічні або супутні товари. У цьому випадку за матрицею інтенсивності переходів користувача формується вибірка за ключовими ознаками стосовно товару, який цікавив користувача раніше. Тобто, у даному випадку, технічна система має n можливих станів, що реалізуються за матрицею інтенсивностей переходів користувача. Було поставлене завдання

знаходження значення деяких змінних, які характеризуватимуть стан системи у певному проміжку часу. На практиці це може проявлятися рекомендацією відвідувачу системи щодо пропозицій товарів та послуг, аналогічних переглянутим раніше. Подібну задачу вирішено методом Рунге-Кутти 4-го порядку для систем диференціальних рівнянь, у яких права частина не залежить від часу. Необхідно чисельно знайти у момент часу t значення перемінних x_1, x_2,...,x_n, що задовольняють деякій системі диференціальних рівнянь при початкових умовах: $x_i(t=0) = a_i$ (I=1,2,…,n). На цій основі була створена модель та реалізовані алгоритми для побудови переходів при формуванні вибору за маркерами користувача для систем, що знаходяться у стаціонарному стані та таких систем, де рекомендація виступає випадковою подією. Також була представлена методологія розробки алгоритму формування рекомендацій за маркерами користувача, у випадку, коли користувач переглядає занадто великі обсяги інформації і важко визначити пріоритетність маркерів. У такому випадку виникає велика кількість альтернатив, що у підсумку може призвести до помилки. Запропонований підхід до мінімізації такої помилки.

У третьому розділі роботи наведені розроблені моделі надання рекомендацій за допомогою правил комп’ютерної логіки.

При реалізації інформаційної технології рекомендаційної підтримки прийняття рішень можна розділити процес на окремі кроки: на першому кроці за маркерами користувача збирається і систематизується інформація, на другому кроці формується рекомендація на основі отриманих вибірок щодо уподобань користувача. Тобто, у цьому випадку можна говорити про дискретний алгоритм, що складається з декількох актів, виконання яких не викликає сумніву. А враховуючи те, що такий підхід реалізує принцип створення управляючого пристрою, коли за діями користувача рішення може приймати лише два значення – одиничне (на основі отриманих маркерів користувачу можна надати рекомендацію) та нульове (маркери не орієнтують, що даний продукт може зацікавити користувача, тобто, рекомендацію щодо

пропозиції товару чи послуги надати неможливо), наведене можна описати комбінаційною схемою за допомогою правил комп'ютерної логіки. Орієнтуючись на підходи до реалізації задач у вебпрограмуванні, технологію рекомендаційної підтримки, яка на вході має інформацію з різнорідних джерел стосовно поведінки користувача, а на виході – інформацію з рекомендаціями, можна представити у вигляді моделі В.М. Глушкова – сукупності керуючого та операційного автоматів.

Можна розглянути поведінку користувача через матрицю, тобто, навести множини продуктів, що будуть суміжними з кожним користувачем та множину користувачів, що суміжні з якимось конкретним продуктом. У цьому випадку поведінку користувача можна розглядати через множини вхідних сигналів, станів, з врахуванням початкового стану та функції переходів між станами, і виходів, які надають результат. Іншими словами, функція переходів δ показує, що автомат S, перебуваючи в деякому стані $a_j \in A$, при появі вхідного сигналу $x_j \in X$ переходить у якийсь стан $a_p \in A$. На цьому базисі розроблено механізм представлення переходів користувача з метою відбору параметрів для надання рекомендацій. Прийнято, що розглядаються кроки у дискретному середовищі, коли за діями користувача рішення може приймати лише два значення – одиничне (на основі отриманих маркерів користувачу можна надати рекомендацію) та нульове (маркери не орієнтують, що даний продукт може зацікавити користувача, тобто, рекомендацію щодо пропозиції товару чи послуги надати неможливо). Деталізовано моделі переходів в рекомендаціях в залежності від зміни уподобань за допомогою створення графів переходів за різними вхідними параметрами, що характеризують дії користувача та використання еквівалентних станів уподобань при формуванні матриць переходів рекомендацій за зміни окремих ознак.

Рекомендаційна система, що пропонується до реалізації, має в основі синтез цифрових автоматів. На цій основі розроблено механізм рекомендацій за ознакою структурної повноти. Розроблено механізм формування тригерів для створення рекомендацій за зміни маркерів користувача. Під тригером у

дані роботі прийнято стійкий стан рекомендації до зміни ключових ознак на вході системи – маркерів користувача. У підсумку, за наведеним математичним базисом, за допомогою мови Python розроблено рекомендаційну систему.

У четвертому розділі роботи представлено опис розробленої інформаційної технології з комбінування методів для побудови рекомендації на основі переваг та відмов користувача.

Особливість підходу, що пропонується в роботі, полягає у реалізації такого механізму, який дозволяє вичленовувати та аналізувати маркери споживача навіть у тих випадках, коли користувач переглядав якусь позицію продукту але не вибрав остаточно. Аналіз такої поведінки та використання показників «не вибору» у якості вхідних параметрів у моделі рекомендаційного механізму дозволить розширити можливий спектр рекомендацій за булевим базисом «І-Або-Ні» та «Або-Ні» через асинхронний RS-тригер.

На практиці подібне реалізується наступним чином: користувач може мати маркер стосовно гучності музичного твору, тоді базис вибірки чітко буде визначений за тригером «Або-Ні», а може не мати такого маркера, тобто, користувачу байдужа потужність музичного твору, і тоді тригер спрацьовує за базисом «І-Або-Ні». В останньому випадку користувачу будуть рекомендовані твори, що враховують інші маркери користувача. Якщо базис «Або-Ні» відповідає умовам відбору, то користувачу буде надана більш точна рекомендація.

В розділі також представлене вирішення задачі логічної еквівалентності в процесі розробки інформаційної технології рекомендаційної підтримки за допомогою підходів, використовуваних в комп’ютерній логіці. Запропонований підхід до мінімізації числа станів. Таке рішення допомагає більш спрощено представити дискретну систему за окремими перевіреними кроками, з контролем щодо виявлення можливих помилок на кожному кроці реалізації у якості вебсервісу.

В роботі наведена розроблена реалізована інформаційна технологія рекомендаційної підтримки прийняття рішень у вигляді веб-додатку, яка дозволяє:

– автоматизувати збір, обробку й завантаження даних з відкритих і конфіденційних джерел (за умови надання дозволу General Data Protection Regulation (EU GDPR));

– виявити реальні або потенційні рекомендації користувачу на основі спостережень і обробки отриманої інформації з різнорідних джерел на основі визначених маркерів користувача;

– формувати внутрішні звіти за результатами аналізу інформації, інформування користувача про проходження процесу аналізу, оперативне інформування про виявлені ситуації щодо неможливості виконання запиту.

Результати проведеної роботи впроваджені у діяльність ТОВ «Евергрін Інтерпрайз» у розробці програмного забезпечення, видавництва «Каяла» (м. Київ) для створення нового сайту книжкового Інтернет-магазину з рекомендаційною системою, ПрАТ «Українсько-Польський навчальний заклад «Центрально-Європейський університет» для використання у навчальному процесі.

Ключові слова: маркер користувача, ключова ознака, TensorFlow, комп’ютерна логіка, метод Рунге-Кутти, тригер, модель переходів, стаціонарний стан, булевий базис.

ABSTRACT

Kuprin O.M. Information technology of recommendation for decision support. – Qualifying scientific work on manuscript rights.

Dissertation for the degree of Doctor of Philosophy in specialty 122 "Computer Science" – Institute of Telecommunications and Global Information Space, National Academy of Sciences of Ukraine, Kyiv, 2023.

The dissertation work is about the devoted to the models and methods for decision support for collecting and processing information from various sources in order to create a recommendation based on the study of consumer preferences.

The paper reviewed models of methods and approaches used in the development of information technologies of recommendation for decision support. A general overview of recommender systems and their types, including content analysis, collaborative filtering, hybrid and popularity-based methods, is performed. On this basis, Content-based algorithms and collaborative filtering algorithms were studied in depth. This made it possible to better understand the principles of the formation of new approaches to the creation of models of recommendation mechanisms and their algorithmization. As a result of the study, a conclusion was drawn regarding the need to develop an algorithm that will combine models of different types in hybrid type recommender systems based on content and collaborative filtering.

In the process of theoretical research of scientific sources, the problem of unstable performance of implemented models of recommendation systems was revealed. Based on this, options for solving this problem were proposed by using certain significant factors that attract the user when choosing a product, but may not be decisive to finally make a purchase.

A research of the peculiarities of the structuring of recommendation support processes based on user token was conducted. The user marker is defined as certain significant factors that are prioritized when a consumer chooses a particular product

in a certain period of time. When choosing a product, the user has his own personal interest and solves a certain problem, which can be expressed as a sequence of interconnected steps. At the same time, the user cares not so much about the formulation of the final decision, but about what is connected with this process and ultimately follows from it. To solve a problem, you need to have a choice of solution options.

Based on the definition of a recommender system, such systems are a subclass of information filtering systems that allow building a certain rating based on requests or preferences. Analyzing decision-making support processes, it can be understood that scientific methods of evaluating work results, behavior, success and ranking according to a number of indicators are the basis. And this is the basis for building algorithms of the recommendation system.

To fulfill the tasks of the work, the structuring of the processes of the recommender system with different interests of users was carried out using n-dimensional matrices with the aim of further implementation of algorithms using the open source TensorFlow library. It is taken into account that the algorithms should provide for the implementation of the situation according to certain user markers, i.e., key features for the user by which he makes decisions, taking into account various changes, restrictions and personal actions of the user at a certain time. Based on the results of the above, a linear model of recommendations based on several user tokens was developed, in which a transition from problem solving using game theory to process algorithmization using an evaluation model was created. The performed calculation proved the adequacy of the model built based on two defined user token.

Developed models of providing recommendations using computer logic rules are given.

When implementing information technology for recommendatory decision-making support, the process can be divided into separate steps: in the first step, information is collected and systematized according to the user's tokens, in the second step, a recommendation is formed based on the received samples regarding the user's preferences. That is, in this case we can talk about a discrete algorithm

consisting of several acts, the execution of which does not raise any doubts. And taking into account the fact that this approach implements the principle of creating a control device, when based on the actions of the user, the decision can take only two values – one (on the basis of the received token, the user can be given a recommendation) and zero (the token do not indicate that this product may interest the user, i.e. , it is impossible to provide a recommendation on the offer of a product or service), the above can be described by a combinational diagram using the rules of computer logic. Focusing on approaches to the implementation of tasks in web programming, the technology of recommendation support, which at the input has information from various sources about user behavior, and at the output – information with recommendations, can be presented in the form of a model of V.M. Hlushkov – a set of control and operating automata.

The recommendation system proposed for implementation is based on the synthesis of digital automata. On this basis, a mechanism of recommendations based on structural completeness was developed. A mechanism for generating triggers for creating recommendations for changing user token has been developed. Under the trigger, the steady state of the recommendation to change the key signs at the system entrance – user token – is adopted in this work. As a result, based on the given mathematical basis, a recommender system was developed using the Python language.

A description of the developed information technology for combining methods for building a recommendation based on user preferences and refusals is presented.

The paper presents the developed and implemented information technology for recommendatory of decision support in the form of a web application, which allows:

– automate the collection, processing and uploading of data from open and confidential sources (provided that General Data Protection Regulation (EU GDPR) permission is granted);

– identify real or potential recommendations to the user based on observations and processing of information received from disparate sources based on defined user tokens;

– to form internal reports based on the results of information analysis, informing the user about the completion of the analysis process, promptly informing about detected situations regarding the impossibility of fulfilling the request.

The results of the work were implemented in the activities of Evergreen Enterprise LLC in the development of software, Kayala Publishing House (Kyiv) for the creation of a new online book store website with a recommendation system, PJSC "Ukrainian-Polish Educational Institution "Central European University" for use in the educational process.

Keywords: user token, key feature, TensorFlow, computer logic, Runge-Kutta methods, trigger, transition model, steady state, Boolean basis.

СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковані основні наукові результати дисертації

Публікації у фахових виданнях:

1. Купрін О.М. Алгоритмізація процесів у рекомендаційних системах / Математичні машини і системи. – 2022. – № 1. – сс. 71 – 80. ISSN 1028-9763**. (кат. Б)**

2. Кряжич О.О., Ющенко К.С., Іцкович В.Є., Купрін О.М. Особливості алгоритмізації процесів мінімізації похибок апроксимації при вирішенні прикладних задач**.** Математичні машини і системи. – 2023 - №1 – сс. 118 – 129. **(кат. Б)**

Публікації у виданнях, що входять до міжнародних науково-метричних баз:

3. Kryazhych, O., Itskovych, V., Iushchenko, K., Kuprin, O. (2023). Features in solving individual tasks to develop service-oriented networks using dynamic programming. Eastern-European Journal of Enterprise Technologies, 1 (4 (121)), 34–40. doi: https://doi.org/10.15587/1729-4061.2023. **(Scopus, Q3)**

4. Гуляєв, К. Д., Ющенко, К. С., Купрін, О. М. (2023). Застосування абстрактних автоматів Мілі та Мура для реалізації алгоритмів рекомендації та вибору. International Scientific Technical Journal "Problems of Control and Informatics", 67(6), с. 14–24. doi: https://doi.org/10.34229/1028-0979-2022-6-2. **(кат. А)**

Наукові праці, які засвідчують апробацію матеріалів дисертації:

5. Купрін О.М. Надання рекомендацій користувачу онлайн-сервісу на підставі обробки даних як базис безпаперової інформатики В. Глушкова // Історія, сучасний стан та тенденції цифрового розвитку суспільства. Матеріали 10-ої Міжнар. наук.-практ. конф. «Глушковські читання», Київ, 2021 р. / Уклад.: Р.М. Богачев, В.Д. Піхорович, А.Ю. Самарський, М.І. Сторожик. – Київ, 2021. – с. 102 – 104.

6. Купрін О.М. Структуризація алгоритмів рекомендаційних систем, що використовуються у фінансово-кредитній сфері / Інформаційно-комунікаційні технології та сталий розвиток // Колективна монографія за матеріалами XXI Міжнародної науково-практичної конференції (Київ, 14-16 листопада 2022 р.) / За заг. ред. С.О. Довгого. – К.: ТОВ «Видавництво «Юстон», 2022. – сс. 195 – 196. ISBN 978-617-7854-76-9

7. Кряжич О.О., Коваленко О.В., Купрін О.М. Використання механізму рекомендацій при вдосконаленні інструментів у комп'ютерній програмі «Випадкова точка»// Інформаційно-комунікаційні технології для перемоги та відновлення / Колективна монографія за матеріалами XXII Міжнародної науково-практичної конференції «Інформаційно-комунікаційні технології та сталий розвиток» (Київ, 14-15 листопада 2023 р.) / За заг. ред. С.О. Довгого. – К.: ТОВ «Видавництво «Юстон», 2023. – сс. 89 – 90. ISBN 978-617-8335-06-9.

8. Кряжич О.О., Купрін О.М. Процес створення рекомендаційного алгоритму. XII Наукова конференція «Наукові підсумки 2023 року». Збірка наукових праць. – Харків, Х.: Технологічний Центр, 2023. – 98 С. (с. 67). e-ISBN 978-617-8360-00-9.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

EU GDPR – General Data Protection Regulation.

ICF – Item-based Collaborative Filtering.

IDF – логарифм оберненої частоти документів, що містять слово term (inverse document frequency).

OLAP – online analytical processing.

TF – кількість входжень слова term в документ d (term frequency).

TF-IDF – перетворення тексту в осмислене представлення чисел (term frequency-inverse document frequency).

UCF – User-based Collaborative Filtering.

ЕП – елементи пам’яті.

ІТ – інформаційна технологія.

КС – комбінаційна схема.

РС – рекомендаційні системи.

СППР – системи підтримки прийняття рішень.

ЦА – цифровий автомат.

ВСТУП

Алгоритми машинного навчання стають невід’ємним напрямом сучасного розвитку інформаційних технологій. Вони поєднують в собі дані, обмеження та моделі, що у підсумку дозволяють створити інформаційну технологію, яка дозволяє виконувати операції безпосередньо машині самостійно генерувати та виконувати складні алгоритми без втручання людини. Рекомендаційні системи є класом алгоритмів машинного навчання, які надають користувачеві релевантні рекомендації на основі взаємодії користувача з аналогічними елементами або на основі вмісту елемента. Вони є новою ланкою сучасних систем підтримки прийняття рішень. У цьому випадку зазначені системи набувають можливості оперування дійсно великими обсягами інформації, обробляючи масиви даних та перетворюючи їх у систематизовані вибірки за запитами кінцевих користувачів. Подібні системи зменшуватимуть інформаційне перевантаження та задовольнятимуть більш широку сферу потреб користувачів он-лайн сервісів.

Рекомендаційні системи, засновані на глибокому навчанні, та останнім часом привертають дедалі більшу увагу як науковців, так і комерційний сектор. І керівним питанням розвитку таких систем є всебічне врахування інтересу користувачів. Проте більшість існуючих моделей рекомендацій враховують вхідні дані послідовності поведінки, що отримані лише при натисканні або купівлі, тобто в результаті взаємодії користувача з активним елементом. У цьому випадку моделі рекомендацій є недостатньо продуктивними, оскільки відбувається ігнорування всього спектру даних про поведінку користувача, пропускаючи те, що отримало інтерес користувача, але у підсумку не було обране.

Визначальним фактором до вдосконалення рекомендації є розуміння еволюції вподобань користувачів. Більшість існуючих моделей послідовних рекомендацій (наприклад, GRU4REC, NARM, SDM, SASRec, Caser) мають у

якості вхідних даних лише виконані натискання, яким присвоюється статус ключових ознак, або маркерів користувача, що визначають його дії. Проте зазначені моделі приділяють мало уваги деталізації впливу окремих маркерів користувача на підсумковий результат, хоча вибір може відбуватися за сукупністю деяких малозалежних ознак. Але ж подібне може доповнювати дані про зацікавлення користувача, створюючи більш точну модель рекомендацій, що підкреслює **актуальність теми дослідження**.

Питанням вдосконалення систем надання рекомендацій на підставі обробки великих даних приділяли увагу такі вчені, як Aggarwal C., Koren Y., Volinsky C., Melville P., Francesco R., Power D. J., Zhang, S.X., Babovic, V., Russell S. J.; Norvig P., Murphy K. P. та інші. Розробці моделей та методів у розробці систем прийняття рішень присвячені роботи українських вчених Трофимчука О.М., Довгого С.О., Морозова А.О., а також закордонних науковців – Sprague R., Power D. J., Zhang, S.X. та інших. Також привертає увагу відома книга із застосування рекомендаційних систем на практиці Falk K., яка представляє собою найбільш фундаментальну працю для розробників систем. В роботі демонструється структура рекомендаційної системи, типовий підхід до алгоритмізації та реалізації у веб-просторі. Проте автор теж не приділяє уваги до питання аналізу ключових ознак стосовно не обраних користувачем товарів та послуг, хоча останнє і відноситься до проблем функціонування сучасних рекомендаційних систем.

Зазначений напрям дослідження – дуже динамічний. Потреба у більш досконалих системах рекомендацій та підтримки прийняття рішень зростає з активізацією цифровізації життя. Тому зазначене питання щодо врахування при розробці рекомендаційних систем та систем прийняття рішень вибраних але не активізованих користувачем маркерів, особливо у сфері, яка стосується фінансового вибору та прийняття рішення, визначає в сукупності **актуальність завдань** дисертаційних досліджень.

Зв’язок роботи з науковими програмами, планами, темами. Наукова спрямованість дисертації відповідає напрямкам науково-технічної політики

України, Закону України «Про наукову і науково-технічну діяльність» (в останній редакції). Окремі дослідження виконувалися в рамках науково-дослідних Інституту телекомунікацій і глобального інформаційного простору НАН України, зокрема: «Розробка засобів інформаційно-аналітичної підтримки завдань забезпечення стійкості об'єктів критичної інфраструктури в регіональній соціоекосистемі за умов зростання природних, техногенних і соціальних загроз» (№ ДР 0121U109216) та «Розробка інформаційної технології моделювання і прогнозування розвитку соціально-еколого-економічних систем в умовах невизначеності, нестаціонарності та ризику» (№ ДР 0121U100132).

Мета й завдання дослідження. Метою роботи є розробка моделей та методів рекомендаційної підтримки прийняття рішень для збору і обробки інформації із різнорідних джерел з метою створення рекомендації на основі вивчення уподобань споживача.

Мета розкривається через наступні завдання:

– *систематизувати* існуючі моделі, методи та підходи, використовувані при розробці технологій прийняття рішень та надання рекомендацій, на основі чого сформулювати методологію дослідження;

– *визначити* особливості та *обґрунтувати* підхід стосовно структурування процесів рекомендаційної системи, що базуються на дослідженні впливу маркерів користувача на визначення сигналів на виході – рекомендацій, з аналізом за критеріями поведінки користувача в мережі Інтернет «переглянув/обрав» та «переглянув/не обрав»;

– *обґрунтувати та розробити* модель рекомендаційної підтримки, формування якої відбувається за правилами комп'ютерної логіки з метою видачі інформаційної вибірки, отриманої шляхом обробки даних з великої кількості джерел, з мінімізацією повторів;

– *представити* програмну реалізацію рекомендаційної системи яка інтегруватиме можливу послідовність взаємодії користувача з онлайн-сервісом у вхідні дані моделей рекомендаційної системи.

Об’єкт дослідження – процеси автоматизації підтримки прийняття рішень з вибору найбільш оптимальної дії за аналізом інформації, що надходить з різнорідних джерел.

Предмет дослідження – моделі та методи підтримки прийняття рішень з надання рекомендацій стосовно інформації на основі поведінкових уподобань користувача Інтернет.

Методи дослідження. В роботі використані окрім загальнонаукових методів, методи та підходи комп’ютерної логіки, математичного моделювання та теорії ймовірності, теорії ігор, асоціативні методи пошуку, методи рішення інженерних задач на етапі проектування інформаційних систем для обґрунтування поведінки користувача в Інтернеті при вирішенні питань, стосовно надання рекомендацій щодо здійснення персонального оптимального вибору.

Для реалізації завдань роботи використана відкрите програмне забезпечення – бібліотека Tensor Flow.

Наукова новизна одержаних результатів міститься в наукових положеннях, що виносяться на захист та у яких:

1. *Уперше* запропонована модель з математичним обґрунтуванням за допомогою методу Рунге-Кутти для системи зі стаціонарним станом та застосуванням закону розподілу ймовірності при наданні рекомендації користувачу, як випадкової події при Інтернет-серфінгу, яка базується на аналізі поведінки користувача в мережі Інтернет, коли користувач переглядає деяку позицію і не обирає її, з проведенням аналогії, коли наслідком перегляду є вибір;

2. *Дістала подальшого розвитку* методологія розробки рекомендаційного алгоритму, яка відрізняється тим, що вхідними параметрами до створення рекомендації виступають маркери користувача, які можуть бути як позитивними, так і негативними, мати свою вагу, а у випадку, коли користувач переглядає занадто великі обсяги інформації і важко

визначити пріоритетність маркерів з аналізом впливу альтернатив, своєчасно виявляти помилки, які можуть призвести до невірної рекомендації;

3. *Уперше* представлена модель надання рекомендацій з використанням правил комп'ютерної логіки, в якій опрацьовано механізм переходів користувача з метою відбору параметрів для надання рекомендацій за кроками у дискретному середовищі, коли за діями користувача рішення може приймати лише два значення – «так» чи «ні»;

4. *Уперше* запропонована програмна реалізація рекомендаційної підтримки прийняття рішень у вигляді вебдодатку, з реалізацією моделі переходів в рекомендаціях в залежності від зміни уподобань за допомогою створення графів переходів за різними вхідними параметрами, що характеризують дії користувача та використання еквівалентних станів при формуванні матриць переходів рекомендацій за зміни окремих ознак.

Достовірність та обґрунтованість результатів. Результати роботи є достовірними, оскільки вони отримані та перевірені за допомогою математичного аналізу та обробки статистичних даних, частина результатів отримана в процесі комп'ютерних експериментів та під час практичної апробації створених алгоритмів та розробленого програмного продукту.

Достовірність основних положень та результатів дисертації доведено:

– використанням апробованих методів математичного аналізу та обробки статистичних даних;

– відповідністю експериментальних досліджень статистичним результатам, отриманим з перевірених джерел;

– використанням сертифікованих комп'ютерних програм та інтегрованих середовищ розробки.

Наукові положення, висновки та рекомендації **обґрунтовані**, тому що базуються на науково-обґрунтованих теоремах, законах та апробованих раніше фундаментальних працях. Розроблені алгоритми були протестовані як на статистичних даних, так і в процесі проведення комп'ютерних експериментів.

Практичне значення отриманих результатів. Особливістю отриманих результатів та вирішених поставлених в роботі задач є їх універсальність щодо застосування у сучасних Інтернет-технологіях, які дозволятимуть користувачу зробити обґрунтований вибір та уникнути зайвих витрат, викликаних спонтанними рішеннями на основі реклами чи пропозицій інших користувачів. Розроблені моделі, алгоритми та методи дозволять розробляти Інтернет-технології з рекомендаційними системами на підставі інтелектуальної обробки даних, отриманих з багатьох джерел як за перевагами користувача, так і за його відхиленнями окремих пропозицій.

Реалізація роботи. Результати досліджень впроваджено:

1) ТОВ «Евергрін Інтерпрайз» у розробці програмного забезпечення;

2) Видавництві «Каяла» (м. Київ) для створення нового сайту книжкового Інтернет-магазину з рекомендаційною системою;

3) ПрАТ «Українсько-Польський навчальний заклад «Центрально-Європейський університет» для використання у навчальному процесі.

Документи, які підтверджують впровадження результатів досліджень, наведені в Додатку А.

Особистий внесок автора в роботи, опубліковані в співавторстві: Автором самостійно отримані головні результати дисертаційного дослідження. В опублікованих у співавторстві наукових працях здобувачем здійснено (згідно «Списку опублікованих праць за темою дисертації»): у роботі [2] – виконання розрахунків з застосуванням підходу до апроксимації помилок стосовно прикладних задач аналізу витрат користувача при користуванні фінансовими сервісами в мережі Інтернет; у статті [3] – моделювання поведінки користувачів у мережевих сервісах, статистична обробка отриманих у дослідженнях результатів; у роботі [4] – апробація використання абстрактних автоматів в реалізації алгоритмів рекомендації; у тезах [7] – апробація рекомендаційного механізму при вирішенні окремої прикладної задачі; у тезах [8] – розробка алгоритму рекомендацій.

Апробація результатів дисертації. Результати дисертації обговорювалися та доповідалися на чотирьох міжнародних науково-практичних конференціях: X Міжнародній науково-практичній конференції «Глушковські читання: Історія, сучасний стан та тенденції цифрового розвитку суспільства» (2022 р.); XXI Міжнародній науково-практичній конференції «Інформаційно-комунікаційні технології та сталий розвиток» (Київ, 14-16 листопада 2022 р.); XXII Міжнародній науково-практичній конференції «Інформаційно-комунікаційні технології та сталий розвиток» (Київ, 14-15 листопада 2023 р.); XII науковій конференції «Наукові підсумки 2023 року» (Харків, 1-2 грудня 2023 р.).

У повному обсязі дисертація доповідалася у 2023 р. на розширеному науково-технічному семінарі Інституту телекомунікацій та глобального інформаційного простору Національної Академії наук України під керівництвом доктора технічних наук, професора, члена-кореспондента НАН України Трофимчука О.М.

Публікації. Основні наукові результати дисертаційної роботи опубліковані у 8 публікаціях у професійних фахових виданнях: одна з них – стаття, що проіндексована в наукометричній базі SCOPUS; одна з них – стаття, що відноситься до журналу, який входить до категорії А у затверджених МОН України виданнях. Загалом чотири статті – у затверджених МОН України виданнях. Серед публікацій, які додатково відображають наукові результати дисертації, є 4 надрукованих доповіді у матеріалах вітчизняних та міжнародних наукових та науково-практичних конференцій.

Структура й обсяг дисертації. Дисертаційна робота містить вступ, чотири розділи, висновки, додатки, список використаних джерел. Загальний обсяг дисертації – 213 сторінок, обсяг основного тексту – 148 сторінки. Робота містить 37 таблиць, 31 рисунок, додатки на 53 сторінках. Список використаних джерел складається зі 109 найменувань.

РОЗДІЛ 1
ОГЛЯД МОДЕЛЕЙ, МЕТОДІВ ТА ПІДХОДІВ, ЩО ВИКОРИСТОВУЮТЬСЯ ПРИ РОЗРОБЦІ ТЕХНОЛОГІЙ РЕКОМЕНДАЦІЙНОЇ ПІДТРИМКИ РІШЕНЬ

1.1 Типи та загальна характеристика рекомендаційних систем

Рекомендаційні системи (РС) виступають складовою систем підтримки прийняття рішень (СППР) [1] для надання персоналізованих рекомендацій [2] та підтримки оптимального вибору на основі спеціальних алгоритмів, що попередньо вивчають уподобання користувача на певних платформах, сервісах, соціальних мережах. Ці алгоритми [3] допомагають зменшити інформаційний шум, забезпечити більш швидкий та ефективний пошук оптимальних варіантів, а також підвищити рівень інформованості та персоналізації, що призводить до кращих результатів і поліпшення користувальницького досвіду.

Система прийняття рішень – це комплексний набір методів, процедур та інструментів, що допомагають приймати обґрунтовані рішення в складних ситуаціях [4]. Така система може включати збір та аналіз даних, моделювання, прогнозування та оцінку ризиків, а також використання різних методів та алгоритмів для підтримки прийняття рішень. У цьому випадку РС виступають програмними інструментами та методами, які надають пропозиції щодо предметів, які, найімовірніше, можуть зацікавити конкретного користувача [5]. Варто враховувати, що пропозиції стосуються різних процесів прийняття рішень, наприклад, які товари купувати, яку музику слухати або які новини в Інтернеті читати. У даному випадку термін «товар» або, інакше, «продукт» – загальний термін, який використовується для позначення того, що система рекомендує користувачам [2].

Функціонування рекомендаційної системи складається з [6]:

– накопичення інформації про користувачів і предмети;

– навчання;

– прогнозування;

– зворотного зв’язку.

Існує два основних типи РС (рис. 1.1) – персоналізовані та неперсоналізовані [7].

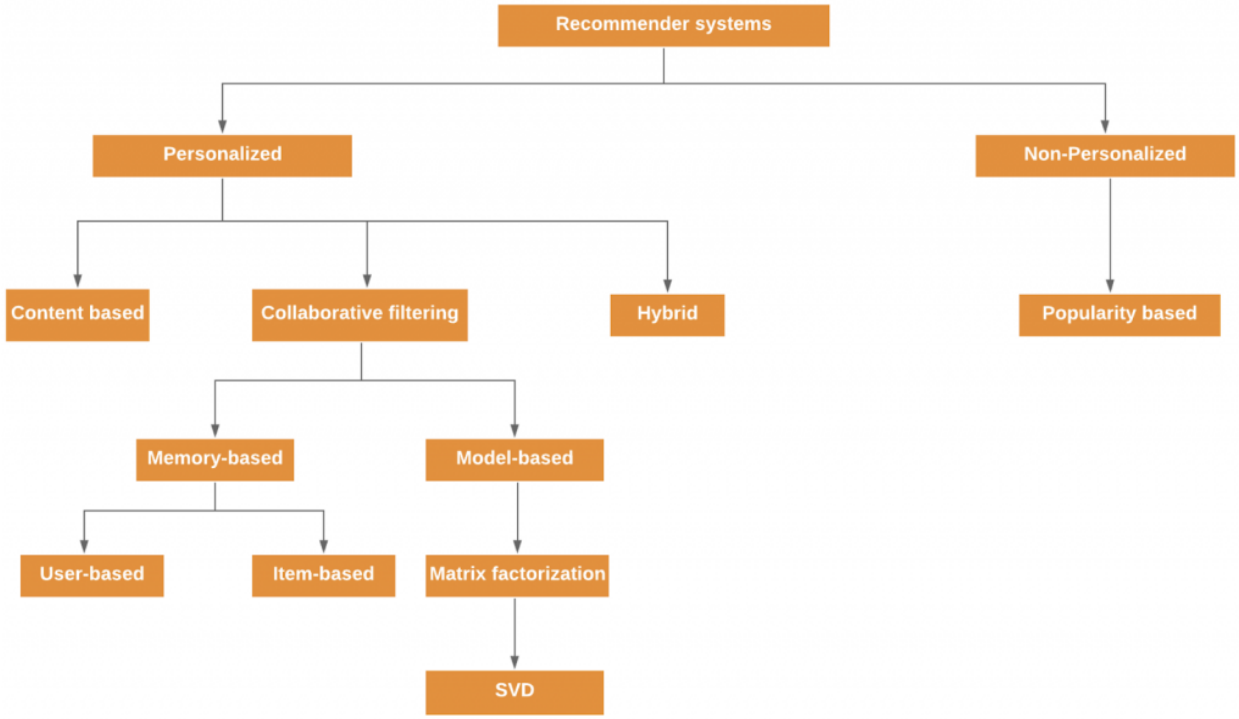


Рисунок 1.1 – Типи рекомендаційних систем

Неперсоніфіковані системи рекомендацій, наприклад, такі як рекомендації на основі популярності, що рекомендують користувачам найпопулярніші товари: топ-10 фільмів, найпопулярніші книги, товари, які найчастіше купують. На відміну від них, персоналізовані системи рекомендацій більш детально аналізують дані користувачів, їх покупки, рейтинг і стосунки з іншими користувачами для того, щоб надати кожному користувачу індивідуальні рекомендації [8].

Методи на основі вмісту (Content based methods) є найбільш використовуваними у сучасних розробках, адже вони використовуються для надання персоналізованих рекомендацій на основі аналізу характеристик або вмісту самого елемента. У фільтрації на основі вмісту для опису елементів використовуються ключові слова, а профіль користувача створюється для того, щоб вказати тип елементів, які подобаються цьому користувачеві [9]. Основна ідея подібного підходу [10] полягає в тому, щоб рекомендувати користувачеві елементи, що подібні до тих, що йому сподобалися раніше. Для цього аналізуються характеристики або властивості елементів [11], такі як ключові слова, теми, жанри, автори, теги тощо.

Процес створення рекомендацій на основі вмісту включає наступні кроки [12]:

1. Видобуток вмісту. Початково необхідно видобути та представити вміст елементів. Це може включати текстову інформацію, зображення, аудіо- або відеодані, які є характерними для кожного елемента.

2. Формування профілів елементів. Далі для кожного елемента створюється його профіль, який містить характеристики та властивості, що описують його вміст. Наприклад, у випадку фільмів це можуть бути жанри, актори, режисери, сюжетні елементи тощо.

3. Визначення вподобань користувача. Для кожного користувача також формується його профіль на основі його вподобань, попередніх виборів або оцінок. Профіль користувача може містити інформацію про те, які властивості елементів йому сподобалися або були оцінені ним позитивно.

4. Відповідність та рекомендації. За допомогою різних алгоритмів, таких як косинусна схожість або TF-IDF (term frequency-inverse document frequency), обчислюються відповідності між профілями користувача та елементів. Чим більша відповідність, тим вища ймовірність, що елемент буде рекомендований користувачу [13].

Переваги використання методів на основі вмісту в рекомендаційних системах [14]:

1. Незалежність від додаткових даних про користувачів. Методи на основі вмісту не потребують великого обсягу інформації про користувачів або їх взаємодію з іншими елементами. Вони можуть працювати навіть з новими користувачами, для яких немає історичних даних.

2. Ефективність в узагальненні рекомендацій. Методи на основі вмісту можуть дати добрі рекомендації, навіть якщо кількість користувачів або елементів велика, оскільки вони зосереджуються на характеристиках самого елемента.

3. Прозорість та пояснюваність рекомендацій. Оскільки методи на основі вмісту використовують характеристики елементів для формування рекомендацій, вони можуть бути більш прозорими та пояснювати, чому певний елемент був рекомендований.

Незважаючи на переваги, методи на основі вмісту також мають свої недоліки [15]:

а) обмежена різноманітність рекомендацій. Методи на основі вмісту зазвичай рекомендують елементи, які мають схожі характеристики з тими, що користувач вподобав раніше. Це може призводити до обмеженості в рекомендаціях та відсутності різноманітності;

б) проблема «холодного старту». Використання методів на основі вмісту може бути складним для нових елементів, для яких немає достатньої інформації про характеристики або контент [16]. У таких випадках, методи на основі вмісту можуть не бути ефективними [17];

в) обмежена увага на контекст. Методи на основі вмісту зосереджуються головним чином на характеристиках самого елемента, а не на контекстуальних факторах, таких як місцезнаходження, час, соціальні аспекти тощо. Це може дещо обмежувати [18] контекстуальну адаптацію рекомендацій;

г) проблема оновлення рекомендацій. Якщо характеристики елементів змінюються в часі, методи на основі вмісту можуть мати проблеми з оновленням рекомендацій із зміненими характеристиками.

Підсумовуючи вищесказане, можна стверджувати, що такі методи є ефективними для рекомендаційних систем, якщо є достатньо характеристик або вмісту елементів, і якщо різноманітність рекомендацій та адаптація до контексту не є критичними факторами.

1.2 Базові підходи до розробки інформаційних технологій рекомендаційної підтримки рішень

1.2.1 Колаборативне фільтрування

Колаборативні методи рекомендаційної підтримки прийняття рішень є підходом, який базується на співпраці між користувачами або елементами для надання персоналізованих рекомендацій [19]. Вони використовують історичні дані про взаємодію між користувачами та елементами (рис. 1.2) для знаходження схожості та використання цієї інформації для формування рекомендацій [20].

Рисунок 1.2 – Схема здійснення колаборативної фільтрації

Основна ідея колаборативних методів [21] полягає у тому, що якщо два користувачі мають подібні вподобання або спільно взаємодіють з елементами, то імовірність того, що їм сподобаються ті самі елементи, висока. Тому, на основі цієї схожості, рекомендаційні системи можуть надати нові елементи користувачам, які мають схожих «сусідів» у відношенні до їхньої взаємодії з елементами.

Колаборативні методи можна поділити на дві основні категорії:

а) Memory-based методи (засновані на пам'яті) – це методи, що використовують безпосередньо історичні дані про взаємодію між користувачами та елементами для надання рекомендацій [22]. В цій категорії виокремлюють два підходи:

– User-Based Collaborative Filtering (фільтрування на основі користувачів). Цей підхід визначає схожість між користувачами на основі їхніх вподобань або історії взаємодії з елементами [23]. Рекомендації надаються, враховуючи вподобання користувачів, які мають схожих «сусідів» у відношенні до їхньої взаємодії з елементами;

– Item-Based Collaborative Filtering (фільтрування на основі елементів). Цей підхід визначає схожість між елементами на основі їхньої взаємодії з користувачами. Рекомендації надаються, враховуючи схожі елементи, які були вподобані користувачами [22];

б) Model-based методи (засновані на моделі) – це методи, що використовують статистичні або машинні навчання моделі для роботи з даними про взаємодію між користувачами та елементами. Ці методи використовують історичні дані для тренування моделі, яка потім може бути використана для надання рекомендацій [24]. Серед моделей даного методу можна виділити наступні:

– Singular Value Decomposition – використовує лінійну алгебру для розкладу матриці оцінок на дві менші матриці, що дозволяє знайти складові, які відображають схожість між користувачами та елементами [25];

– Matrix Factorization – використовує матричний факторизаційний метод для знаходження складових, що описують схожість між користувачами та елементами [26].

Отже, колаборативні методи засновані на припущенні, що подібні користувачі будуть мати схожі вподобання та що подібні елементи будуть подобатися одному й тому ж користувачу. За допомогою алгоритмів схожості, співпраці між користувачами або елементами, ці методи можуть здійснювати передбачення та формувати рекомендації для недоступних елементів або нових користувачів.

Основні переваги колаборативних методів рекомендаційних систем [27]:

– персоналізація. Колаборативні методи дозволяють надавати персоналізовані рекомендації, враховуючи вподобання та інтереси кожного користувача;

– здатність до адаптації. Вони можуть адаптуватися до змін у вподобаннях користувачів та нових елементів шляхом оновлення схожості та перевизначення рекомендацій;

– масштабованість. Колаборативні методи можуть бути застосовані в системах з великою кількістю користувачів та елементів, оскільки їх ефективність не залежить від розміру системи;

– неперсональна інформація. Вони не потребують додаткової інформації про користувачів або елементи, окрім даних про їх взаємодію.

Звісно, окрім переваг колаборативні методи мають також певні недоліки:

– проблема «холодного старту». Для нових користувачів або нових елементів може бути важко надати адекватні рекомендації, оскільки взаємодія з ними ще недостатня;

– розрідженість даних. У великих системах з великою кількістю користувачів та елементів може виникнути проблема розрідженості даних, коли взаємодія обмежена, що може впливати на якість рекомендацій;

– проблема «камераля» рекомендацій. Колаборативні методи можуть обмежувати рекомендації до сфери інтересів конкретної групи користувачів, що може ускладнити надання нових та різноманітних рекомендацій;

– проблема витікання інформації. Взаємодія між користувачами через рекомендації може приводити до витоку конфіденційної або особистої інформації.

Підсумовуючи інформацію щодо колаборативного фільтрування, можна сказати, що вибір між memory-based та model-based підходами залежить від конкретної системи та її вимог. Memory-based методи простіші у реалізації та не вимагають додаткових обчислень, але можуть бути більш чутливими до шуму у даних. Model-based методи можуть забезпечувати кращу точність та здатність до прогнозування, але потребують тренування моделі та обчислень.

1.2.2 Гібридні методи

Гібридна техніка фільтрації поєднує в собі різні методи рекомендацій з метою отримання кращої оптимізації системи, щоб уникнути деяких обмежень і проблем чистих рекомендаційних систем [28]. Вони комбінують переваги різних методів [29], щоб забезпечити більш точні, різноманітні та персоналізовані рекомендації для користувачів.

Ідея гібридних методів полягає в тому, що комбінація алгоритмів забезпечить більш точні та ефективні рекомендації, ніж один алгоритм (рис. 1.3) [30].

Концепція гібридних методів [31] полягає в тому, щоб використовувати силу кількох підходів, оскільки недоліки одного алгоритму можуть бути подолані іншим алгоритмом. Це може бути досягнуто шляхом комбінації різних видів методів, таких як колаборативне фільтрування, методи на основі вмісту та знань про контекст.

Одним з підходів до гібридних методів є об'єднання рекомендацій, отриманих від різних методів, в одну згуртовану рекомендацію [32].

Наприклад, можна комбінувати рекомендації, отримані від колаборативного фільтрування та методів на основі вмісту, шляхом використання ваг або ранжування рекомендацій з різних методів [33].

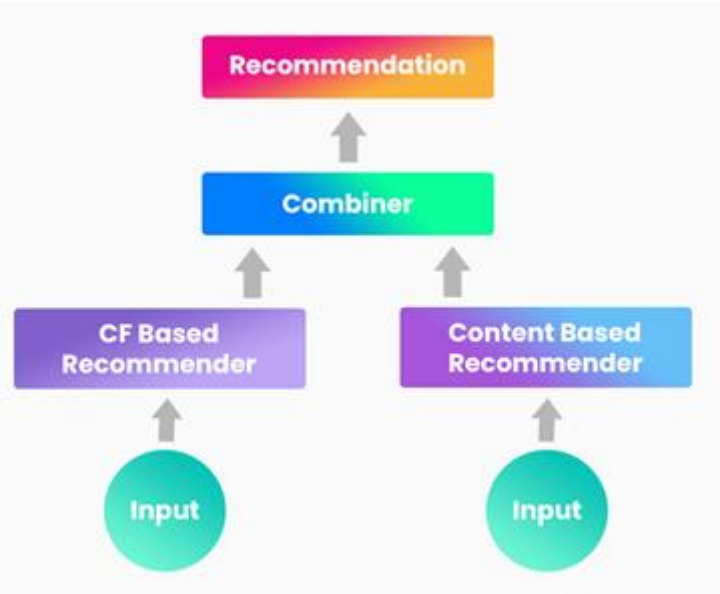


Рисунок 1.3 – Комбінація алгоритмів для реалізації гібридних методів рекомендацій

Інший підхід – це використання моделей машинного навчання, які об'єднують різні підходи в одну модель. Наприклад, можна створити модель, яка поєднує колаборативний фільтринг та методи на основі вмісту, і використовувати цю модель для генерації рекомендацій [34].

Ще одним підходом є використання контекстуальних даних, таких як час, місцезнаходження, демографічні дані тощо, для покращення рекомендацій [35]. Наприклад, можна враховувати географічне положення користувача при формуванні рекомендацій місць для відвідування.

Переваги гібридних методів включають [34]:

– краща точність. Гібридні методи можуть поєднувати сильні сторони різних методів, що дозволяє отримувати більш точні рекомендації;

– більш широке охоплення. Гібридні методи можуть охоплювати різні типи даних, що дозволяє рекомендувати різноманітні елементи, такі як фільми, музика, книги тощо;

– більш персоналізовані рекомендації. Гібридні методи дозволяють більш точно враховувати індивідуальні вподобання та потреби користувачів, що робить рекомендації більш персоналізованими та релевантними.

Незважаючи на переваги, гібридні методи також мають свої недоліки [36]:

– складність реалізації. Гібридні методи можуть бути складнішими у реалізації порівняно з окремими методами. Вони вимагають розробки та налагодження моделей, комбінування різних алгоритмів та управління вагами рекомендацій;

– обчислювальна складність. Гібридні методи можуть вимагати більше обчислювальних ресурсів, оскільки вони включають в себе більше етапів обробки та аналізу даних;

– проблема підбору ваг. Ваги, які використовуються для комбінування рекомендацій з різних методів, можуть впливати на якість рекомендацій. Визначення оптимальних ваг є складним завданням і може вимагати додаткового налаштування та експертного знання.

Необхідно також враховувати контекст, особливості системи та потреби користувачів при виборі та розробці гібридних методів. Це дозволить створити ефективну та чітко працюючу рекомендаційну систему, яка задовольнятиме потреби користувачів та покращить їх досвід.

1.2.3 Методи на основі популярності

Варто також звернути увагу на неперсоналізовані методи [34], а саме методи на основі популярності – це одні з найцікавіших підходів у рекомендаційних системах. Ці методи базуються на ідеї, що популярні або

часто використовувані елементи мають велику ймовірність бути цікавими для користувача.

Концепція методів на основі популярності полягає у використанні агрегованої інформації про популярність елементів для формування рекомендацій. Популярність може бути виміряна за різними метриками, такими як кількість переглядів, продажів, рейтингів, вподобань, загальний час витрачений на елемент, тощо [37].

Процес формування рекомендацій на основі популярності може включати в себе наступні кроки [38]:

а) збір інформації про популярність. Збираються дані про популярність елементів, що можуть бути виміряні за допомогою різних метрик. Ці дані можуть бути отримані з внутрішніх джерел (наприклад, локальна база даних системи) або зовнішніх джерел (наприклад, соціальні медіа, інтернет-магазини тощо);

б) ранжування елементів за популярністю. Елементи ранжуються за відповідними метриками популярності. Це дозволяє визначити, які елементи є найбільш популярними у системі;

в) надання рекомендацій. Найпопулярніші елементи пропонуються користувачам як рекомендації. Це може бути простим списком найпопулярніших елементів або додатково враховувати контекстуальні фактори, інтереси користувача тощо.

Переваги методів на основі популярності [34]:

а) простота реалізації. Методи на основі популярності є простими у реалізації і не вимагають складних алгоритмічних розрахунків або налаштування параметрів;

б) робота з новими користувачами. Ці методи можуть бути ефективними для нових користувачів, для яких немає достатньої інформації про їхні вподобання або історію взаємодії з системою;

в) врахування загальної популярності. Ці методи враховують загальну популярність елементів, що може бути важливим фактором для деяких сценаріїв, наприклад, в медіа- або розважальних системах.

Недоліки методів на основі популярності [39]:

а) відсутність персоналізації. Ці методи не враховують індивідуальні вподобання та потреби користувачів, що може призводити до неперсоналізованих рекомендацій;

б) проблема «фільтру болота». Якщо всі користувачі отримують рекомендації на основі популярності, це може призводити до ситуації, коли популярні елементи стають ще більш популярними, а менш відомі елементи майже не отримують уваги, що зменшує різноманіття рекомендацій;

в) відсутність нових елементів. Методи на основі популярності можуть недостатньо враховувати нові або недавно введені в систему елементи, оскільки вони ще не набрали значну популярність;

г) вплив зовнішніх факторів. Методи на основі популярності можуть підлягати сильному впливову зовнішніх факторів, таких як реклама, маркетингові кампанії або маніпуляції з пропагандою, що може спотворювати рекомендації та погіршувати їх об'єктивність.

У реалізації рекомендаційних систем часто використовуються [40] комбінації методів на основі популярності з іншими підходами, такими як колаборативне фільтрування або методи на основі вмісту, для поліпшення якості та персоналізації рекомендацій. Такі комбіновані методи можуть забезпечити баланс між загальною популярністю та персоналізацією, а також враховувати контекстуальні фактори та індивідуальні вподобання користувачів.

1.3 Базові підходи до реалізації рекомендаційних алгоритмів

Базові рекомендаційні алгоритми мають в основі аналіз характеристик або вмісту елементів, що рекомендуються [41]. Вони використовують інформацію про властивості або описи елементів для визначення ступеня схожості між елементами та вподобанням користувачів. Основна ідея полягає в тому, що якщо користувачам сподобалися певні елементи з певними характеристиками, то ймовірно вони зацікавлені в подібних елементах зі схожими характеристиками.

Content-based алгоритми рекомендаційних систем можна представити на основі вмісту та їхні формули [42]:

а) підготовка даних – збір та підготовка даних про елементи може включати отримання текстових описів, категорій, ключових слів або інших характеристик елементів.

Наприклад, ми маємо таблицю items_data, де маємо інформацію про елементи, їхні назви та описи;

б) векторизація характеристик – кожен елемент представляється у векторній формі, де кожна характеристика має своє числове представлення.

Векторизація текстових описів елементів [43] може бути здійснена за допомогою TfidfVectorizer. Цей метод використовує алгоритм TF-IDF для визначення важливості слів у тексті.

Наприклад, формула для обчислення TF-IDF для слова term у документі *d* може бути представлена так, як на рис. 1.4 [4].

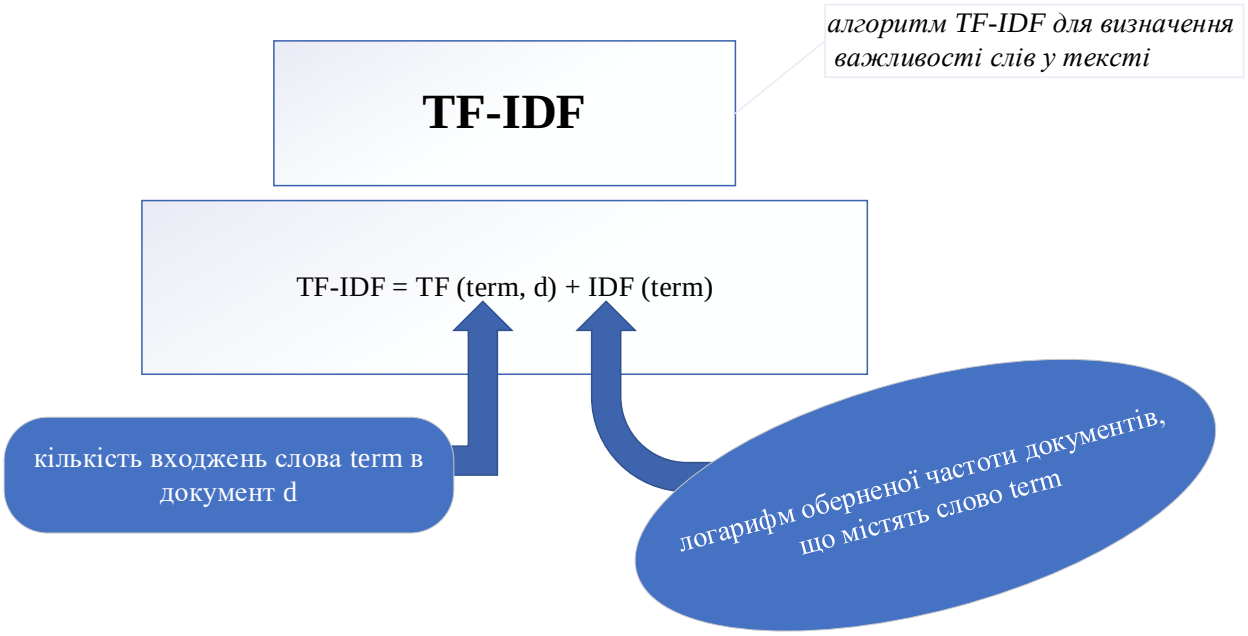


Рисунок 1.4 – Схематичне представлення формули для обчислення TF-IDF

На рис. 1.4 прийняті наступні позначення:

– TF(term, d) – кількість входжень слова term в документ d (term frequency);

– IDF(term) – логарифм оберненої частоти документів, що містять слово term (inverse document frequency).

Безпосередньо формулу для обчислення IDF можна представити наступним чином:

$$IDF(term) = log(N / (1 + DF(term))), \quad (1.1)$$

де:

N – загальна кількість документів,

DF(term) – кількість документів, що містять слово term (document frequency).

За допомогою TfidfVectorizer можна отримати числові вектори, де кожна компонента вектору представляє вагу відповідного слова в описі елемента;

в) розрахунок схожості – використовуючи вектори характеристик, обчислюється схожість між елементами.

Зазвичай [44 – 45] для вимірювання схожості між векторами використовується косинусна схожість (cosine similarity). Формула для обчислення косинусної схожості між векторами A та B виглядає так:

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}, \quad (1.2)$$

де:

A * B – скалярний добуток векторів A та B;

||A|| та ||B|| – норми векторів A та B;

г) визначення вподобання користувача – для кожного користувача обчислюється вподобання для елементів на основі їхньої схожості з вже вподобаними або вибраними користувачем елементами. Це може бути здійснене шляхом зваженого поєднання схожості елемента з вагою, яка відображає ступінь важливості характеристики для користувача. Один з популярних підходів – використання зваженого середнього:

$$preference(item) = \frac{\sum_i similarity(item, liked_item_i) \cdot weight(liked_item_i)}{\sum_i weight(liked_item_i)}, \quad (1.3)$$

де:

similarity(item, $liked_item_i$) – схожість між елементом item та вподобаним елементом $liked_item_i$;

weight($liked_item_i$) – вага вподобаного елемента $liked_item_i$;

д) рекомендації – на основі вподобання користувача можна згенерувати список рекомендованих елементів, відсортованих за спаданням ваги

вподобання [46]. Цей список може бути представлений як ранжований список елементів, які найбільше відповідають інтересам користувача.

Подібний алгоритм можна представити у вигляді коду, з використанням найпоширенішого фреймворку для вирішення задач рекомендаційних систем є Python з бібліотекою scikit-learn та pandas. Варто зазначити, що однією з головних причин вибору Python для гібридної рекомендаційної системи є широка популярність мови в галузі аналітики даних та машинного навчання. Scikit-learn надає широкий вибір алгоритмів машинного навчання та простий інтерфейс для розробки та налаштування моделей. За допомогою pandas, розробники можуть зручно та ефективно маніпулювати та аналізувати дані перед використанням їх у моделях рекомендацій. Такий вибір надає зручне та ефективне середовище для розробки гібридних рекомендаційних систем з різноманітними алгоритмами та обробкою даних.

У випадку описаного алгоритму код буде наступним:

```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity

# 1. Підготовка даних
# Завантаження даних
data = pd.read_csv('data.csv')

# 2. Векторизація характеристик
# Використовуємо TfidfVectorizer для векторизації характеристик
vectorizer = TfidfVectorizer()
feature_vectors = vectorizer.fit_transform(data['features'])

# 3. Розрахунок схожості
# Обчислення матриці схожості на основі косинусної подібності
```

```
similarity_matrix = cosine_similarity(feature_vectors)

# 4. Визначення вподобання користувача
# Виберіть випадкового користувача та отримайте індекс його вподобань
user_index = 0  # Замініть на вибраний індекс користувача
user_preferences = data.iloc[user_index]['preferences']

# 5. Рекомендації
# Обчислення вагованих рейтингів для елементів
weighted_ratings = similarity_matrix.dot(user_preferences)

# Відсортувати рейтинги у порядку спадання
sorted_indices = weighted_ratings.argsort()[::-1]

# Вивести топ-K рекомендованих елементів
K = 5  # Задайте кількість топ-K рекомендацій
top_recommendations = data.iloc[sorted_indices[:K]]['item_name']

print("Рекомендовані елементи для користувача:")
print(top_recommendations)
```

Наведений код показує загальну структуру алгоритму content-based. Застосовані бібліотеки pandas, scikit-learn і TfidfVectorizer допомагають зчитувати дані, векторизувати характеристики, обчислити схожість за допомогою косинусної подібності, визначити вподобання користувача і зробити рекомендації [47].

Загальний алгоритм User-based Collaborative Filtering можна охарактеризувати за наступними етапами реалізації [48]:

а) збір даних. Першим кроком є збір даних про вподобання користувачів. Це можна зробити за допомогою матриці вподобань, що представляє дані у вигляді матриці, де рядки відповідають користувачам, а стовпці – елементам. Кожна комірка матриці містить рейтинг або оцінку, яку користувач поставив елементу. Можна позначити цю матрицю як *R*, де *(Ru; i)* – рейтинг, який користувач *u* поставив елементу *i*;

б) обчислення подібності між користувачами. Після збору даних потрібно виміряти ступінь подібності між користувачами. Це може бути зроблено з використанням різних метрик схожості, таких як коефіцієнт кореляції Пірсона або косинусна схожість. Метрика обчислюється для кожної пари користувачів і вказує на ступінь схожості їх вподобань.

Коефіцієнт кореляції Пірсона вимірює ступінь лінійної залежності між двома користувачами. Формула для обчислення коефіцієнта кореляції Пірсона між користувачами *u* і *v*:

$$sim(u,v) = \frac{\sum_{i \in I_{uv}} (R_{u,i} - \bar{R}_u)(R_{v,i} - \bar{R}_v)}{\sqrt{\sum_{i \in I_{uv}} (R_{u,i} - \bar{R}_u)^2} \sqrt{\sum_{i \in I_{uv}} (R_{v,i} - \bar{R}_v)^2}}, \quad (1.4)$$

де:

I_{uv} – множина елементів, які оцінили як користувач *u*, так і користувач *v*;

$R_{u,i}$ – рейтинг, який користувач *u* поставив елементу *i*;

$\bar{R}_u$ – середній рейтинг, який виставив користувач *u*;

$R_{v,i}$ – рейтинг, який користувач *v* поставив елементу *i*;

$\bar{R}_v$ – середній рейтинг, який виставив користувач *v*.

Значення коефіцієнта кореляції Пірсона може бути в діапазоні від -1 до 1, де наближення до 1 вказує на сильну позитивну кореляцію (схожі вподобання), наближення до -1 вказує на сильну негативну кореляцію (різні

вподобання), а значення, наближене до 0 вказує на відсутність кореляції (незалежні вподобання).

Косинусна схожість вимірює кут між векторами вподобань двох користувачів [34]. Формула для обчислення косинусної схожості між користувачами *u* і *v*:

$$sim(u, v) = \frac{\sum_{i \in I_{uv}} R_{u,i} \cdot R_{v,i}}{\sqrt{\sum_{i \in I_u} R_{u,i}^2} \cdot \sqrt{\sum_{i \in I_v} R_{v,i}^2}}, \quad (1.5)$$

де:

I_{uv} – множина елементів, які оцінили як користувач *u*, так і користувач *v*;

$R_{u,i}$ – рейтинг, який користувач *u* поставив елементу *i*;

I_u – множина елементів, які оцінив користувач *u*;

$R_{v,i}$ – рейтинг, який користувач *v* поставив елементу *i*;

I_v – множина елементів, які оцінив користувач *v*;

в) вибір найближчих сусідів. Після обчислення подібності між користувачами потрібно вибрати «найближчих сусідів» для кожного користувача. Це може бути зроблено, наприклад, шляхом вибору кількох найбільш схожих користувачів за допомогою заданого порогу схожості;

г) рекомендація елементів. Після вибору найближчих сусідів можна перейти до рекомендації елементів. Один з підходів – це врахування ваги оцінок сусідів для кожного елемента, який не був спожитий або оцінений користувачем. Це може бути зроблено шляхом обчислення зваженого середнього значення оцінок сусідів, де ваги відповідають ступеню подібності між користувачами [49].

Формула для розрахунку рекомендованого рейтингу елемента *i* для користувача *u* може мати наступний вигляд:

$$R_{u,i}^{rec} = \bar{R}_u + \frac{\sum_{v \in N(u)} \text{sim}(u, v) \cdot (R_{v,i} - \bar{R}_v)}{\sum_{v \in N(u)} |\text{sim}(u, v)|}, \quad (1.6)$$

де:

$R_{u,i}^{rec}$ – рекомендований рейтинг елемента *i* для користувача *u*;

$\bar{R}_u$ – середній рейтинг, який виставив користувач *u*;

$\text{sim}(u, v)$ – схожість між користувачами *u* і *v*;

$R_{v,i}$ – рейтинг, який користувач *v* поставив елементу *i*;

$\bar{R}_v$ – середній рейтинг, який виставив користувач *v*;

$N(u)$ – множина сусідніх користувачів користувача *u* (користувачів з найвищою схожістю).

У формулі (1.6) для обчислення рекомендованого рейтингу використовується вага, яка відповідає ступеню подібності між користувачами, тобто схожості між користувачем u і сусідніми користувачами. Це означає, що сусіді з більшою схожістю матимуть більший вплив на рекомендований рейтинг елемента і [50].

Використовуючи ці формули та кроки алгоритму, можна реалізувати User-based Collaborative Filtering для рекомендаційних систем [51]. Варто зазначити, що існує багато уточнень та варіацій цього алгоритму, які можуть бути застосовані залежно від потреб і контексту конкретного завдання рекомендаційної системи.

Зазначене можна представити у вигляді наступного лістингу виконання алгоритму:

```
import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity

# Зчитування даних
```

```
data = pd.read_csv('data.csv')  # Припустимо, дані зберігаються у файлі CSV

# Розрахунок подібності між користувачами за допомогою косинусної схожості
similarity_matrix = cosine_similarity(data)

# Функція для рекомендації елементів для користувача
def recommend_items(user_id, k=5):
    user_index = data.index[data['user_id'] == user_id].tolist()[0]  # Отримуємо індекс користувача в даних
    user_similarity = similarity_matrix[user_index]  # Подібність між користувачем та іншими користувачами
    sorted_indices = user_similarity.argsort()[::-1]  # Сортуємо індекси користувачів за спаданням подібності

    recommended_items = []
    seen_items = set(data[data['user_id'] == user_id]['item_id'].tolist())  # Елементи, які вже переглянув користувач

    for index in sorted_indices:
        if len(recommended_items) >= k:
            break
        neighbor_user_id = data.iloc[index]['user_id']
        neighbor_user_items = data[data['user_id'] == neighbor_user_id]['item_id'].tolist()

        # Рекомендації елементів, які були сподобалися сусіду, але не були переглянуті користувачем
```

```
            recommended_items.extend([item for item in neighbor_user_items if
item not in seen_items and item not in recommended_items])

        return recommended_items[:k]

    # Приклад використання
    user_id = 1
    recommended_items = recommend_items(user_id, k=5)
    print(f"Рекомендовані елементи для користувача {user_id}:
{recommended_items}")
```

У цьому коді дані зчитуються з файлу CSV, обчислюється матриця подібності за допомогою косинусної схожості, а потім реалізована функція recommend_items, яка рекомендує елементи для заданого користувача на основі алгоритму User-based Collaborative Filtering. Наприклад використання показано для користувача з ідентифікатором 1, але можна змінити user_id і k (кількість рекомендованих елементів) за своїми потребами.

Цей код є простим прикладом, проте в реальних випадках можуть бути застосовані додаткові оптимізації [52], такі як розрахунок подібності на основі обмеженої кількості найближчих сусідів або використання інших метрик подібності.

Алгоритм Item-based Collaborative Filtering (ICF) є альтернативою до User-based Collaborative Filtering (UCF) для рекомендаційних систем на основі коллаборативного фільтрування. Основна різниця між ICF та UCF полягає в тому, як вони використовують схожість для рекомендацій [34].

В алгоритмі ICF, використовується схожість між елементами, а не між користувачами, як в UCF. Замість того, щоб шукати користувачів зі схожими вподобаннями до активного користувача, ICF шукає елементи, які мають подібні оцінки до елементів, що споживаються активним користувачем.

Таким чином, в ICF спочатку обчислюється схожість між елементами на основі їхніх взаємних оцінок. Потім для рекомендацій використовуються оцінки інших елементів, які мають високу схожість з тими, що споживаються активним користувачем [53].

Окрім цієї основної різниці, основні кроки ICF подібні до UCF, включаючи обчислення схожості, вибір сусідніх елементів та розрахунок рекомендованих рейтингів. Хоча формули можуть бути подібними, вони застосовуються до елементів, а не до користувачів, що відрізняє ICF від UCF.

1.4 Методологія вирішення задач дослідження

1.4.1 Основні проблеми розробки рекомендаційних систем та можливі шляхи їх вирішення

У даному дослідженні розглядається вдосконалення систем прийняття рішень та надання рекомендацій на основі обробки даних з підключених до інтелектуального комплексу джерел інформації. Результати теоретичного огляду показали, що комбінація моделі на основі вмісту та колаборативного фільтрування може ефективно покращити якість рекомендаційних систем.

Продовжуючи цю лінію дослідження, є кілька перспективних напрямків, на які варто звернути увагу. Перш за все, слід вдосконалити роботу неоднорідних моделей, побудованих з врахуванням перегляду користувачем деякого продукту, але не прийняття остаточного вибору. Також, часто на практиці використовуються різні моделі з різними атрибутами, розмірами вбудованих таблиць та різними наборами даних.

Об'єднання неоднорідних моделей [54] може значно покращити якість роботи рекомендаційних систем в цілому. Наприклад, на платформі, подібній до Facebook, можуть використовуватись різні рекомендаційні системи для

стрічки новин, сторіз та дослідження. Інтеграція цих моделей дозволить забезпечити [55] більш точні та персоналізовані рекомендації для користувачів.

Для вдосконалення роботи неоднорідних моделей можна розглянути декілька підходів:

а) Ensemble Learning (Навчання ансамблю моделей). Цей підхід полягає в поєднанні декількох моделей в одну ансамбль моделей [56]. Кожна модель може мати свої сильні та слабкі сторони. Поєднання прогнозів різних моделей може покращити загальну якість рекомендацій. Наприклад, можна використовувати методи багатокласової класифікації, такі як Random Forest або Gradient Boosting, для об'єднання прогнозів різних моделей;

б) Model Selection (Вибір моделі). Замість використання однієї фіксованої моделі можна використовувати алгоритми вибору моделей, що дозволяють динамічно вибирати найкращу модель для кожного конкретного запиту або користувача [57]. Це може залежати від характеристик об'єктів, доступних даних, особистих вподобань користувача тощо. Наприклад, можна використовувати алгоритми вибору моделей, такі як Meta-Learners або Multi-Armed Bandit, для динамічного вибору найкращої моделі залежно від контексту;

в) Model Stacking (Стекінг моделей). Цей підхід полягає в поєднанні вихідних прогнозів різних моделей та використанні їх як вхідні дані для вторинної моделі, яка здійснює фінальне прийняття рішень [58]. Наприклад, можна використовувати стекінг моделей з використанням алгоритмів класифікації, таких як Logistic Regression або Neural Networks, для комбінування прогнозів різних моделей;

г) Model Fusion (Об'єднання моделей). Цей підхід передбачає об'єднання вихідних прогнозів різних моделей з метою отримання більш точних та надійних рекомендацій [59]. Можна використовувати методи фузії моделей, такі як голосування більшості, зважене голосування, або методи зрізу, щоб приймати рішення на основі прогнозів різних моделей.

Дослідження в напрямку гібридних методів для поєднання неоднорідних моделей у системах прийняття рішень та рекомендаційних системах може привести до розробки нових методів та стратегій, які покращать точність та персоналізованість рекомендацій для користувачів.

Крім того, важливо вирішити проблему нестабільної продуктивності впроваджених моделей, що виникає через перенавчання рекомендаційних систем. Перенавчання виникає, коли модель дуже точно «запам'ятовує» тренувальні дані, втрачаючи здатність узагальнювати на нові дані, особливо ті, які «з точки зору» системи є не базовими. Це може призводити до змін в рекомендаціях при незначних змінах у вхідних даних, що погіршує довіру користувачів до системи.

Для вирішення цієї проблеми, необхідно використовувати методи та стратегії, що допоможуть контролювати перенавчання моделей [60]. Ось декілька підходів [34], які будуть використані у роботі:

а) регуляризація. Застосування регуляризаційних методів, таких як L1 або L2 регуляризація, допоможе обмежити ваги моделі та уникнути перенавчання. Це зробить модель більш універсальною та менш залежною від конкретних тренувальних даних;

б) застосування методів підгонки гіперпараметрів. Підгонка гіперпараметрів моделі є важливим процесом, що дозволяє знайти оптимальні значення гіперпараметрів для досягнення кращої продуктивності. Використання автоматичних методів оптимізації гіперпараметрів, таких як Grid Search або Random Search, може допомогти відшукати найкращі налаштування моделі та уникнути перенавчання;

в) застосування регуляризації вихідних даних. Для запобігання перенавчанню можна застосовувати методи регуляризації безпосередньо до вихідних даних, наприклад, шляхом використання методів семплування даних або аргументації даних. Це може допомогти збалансувати нерівномірність даних та зменшити ризик перенавчання;

г) використання ансамблю моделей. Комбінування декількох моделей в ансамбль може знизити ефект перенавчання та покращити стабільність продуктивності. Ансамбль може використовувати різні алгоритми навчання, підмоделі або підходи, що допомагають знизити перенавчання та забезпечити більш стійкі рекомендації;

д) використання аналізу важливості ознак. Аналіз важливості ознак може допомогти виявити та виключити непотрібні або шумові ознаки, що можуть сприяти перенавчанню. Це можна зробити за допомогою методів, таких як аналіз важливості випадкових лісів або методів зведення ознак, що допомагають відібрати найбільш важливі ознаки для моделі.

Ці підходи можуть сприяти стабільній продуктивності рекомендаційних моделей та забезпечити більш надійні та точні рекомендації для користувачів. Продовження дослідження в цьому напрямку має велике значення для подальшого розвитку систем прийняття рішень та рекомендаційних систем, оскільки дозволить покращити якість рекомендацій та забезпечити стабільну продуктивність у реальних умовах застосування.

1.4.2 Теоретична база реалізації технології дослідження

Теоретичною базою дослідження, окрім проаналізованих вище робіт, у дисертаційній роботі виступатимуть:

– наукові праці О.М. Трофимчука [61 – 63], Морозова А.О [64], Sprague R. [65] стосовно математичних основ алгоритмізації процесів, які відбуваються в системах підтримки прийняття рішень;

– дослідження Power D. J. [66], Zhang, S.X.; Babovic, V. [67] в яких розглядаються моделі систем підтримки прийняття рішень у веб-середовищі;

– роботи Russell S. J.; Norvig P. [68], Murphy K. P. [69] з аналізу підходів, алгоритмів та особливостей технологічних рішень у системах машинного навчання.

Основою для теоретичних досліджень стосовно розробки та використанню рекомендаційних систем виступить раніше згадувана робота Falk K. [34], а також дослідження Koren Y. [70], Bhasker B. [71] та роботи інших дослідників за зазначеною тематикою.

Основою для розробки математичних основ рекомендаційного алгоритму обрана комп’ютерна логіка. Роботи з цього напряму дослідників Barwise J. [72], Minato Sh. [73] та частково роботи академіка Глушкова В.М. [74] стосовно особливостей проектування рішень на основі логічних операцій будуть використані для обґрунтування підходів до створення моделей рекомендаційних алгоритмів за використання гібридних методів.

1.4.3 Обґрунтування використання гібридних методів для створення рекомендаційних алгоритмів

Гібридні методи підходять для вирішення питань дослідження, пов'язаних з об'єднанням неоднорідних моделей та вдосконаленням продуктивності рекомендаційних систем.

Один з можливих алгоритмів, на якому можна наочно показати вирішення цих питань, це комбінація моделей на основі вмісту та колаборативного фільтрування в гібридному підході.

Такий вибір можна пояснити наступним чином:

– використовуючи алгоритми на основі вмісту, можна створити профілі користувачів та елементів, враховуючи їх характеристики, атрибути та відношення. Це дозволяє зрозуміти індивідуальні вподобання користувачів та властивості елементів;

– застосування колаборативного фільтрування дозволяє аналізувати взаємодію та спільність між користувачами та елементами на основі їхніх взаємодій, які можуть бути представлені у вигляді рейтингів, відгуків або інших даних. Це допомагає виявити схожість між користувачами та елементами;

– за допомогою гібридного підходу, моделі на основі вмісту та колаборативного фільтрування можуть бути поєднані для отримання кращих рекомендацій. Наприклад, рекомендації на основі вмісту можуть використовуватись для персоналізації рекомендацій залежно від індивідуальних вподобань користувачів, а колаборативне фільтрування може забезпечувати рекомендації на основі спільнот та популярності;

– для вдосконалення продуктивності рекомендаційних систем, можна використовувати різні підходи, такі як кешування, оптимізація обчислювальних процесів або використання розподіленого обчислення. Це дозволяє забезпечити швидку та ефективну роботу системи навіть при обробці великих обсягів даних.

Враховуючи алгоритми [34], можна описати загальну структуру алгоритму, який демонструє інтеграцію моделей на основі вмісту та колаборативного фільтрування, налаштування параметрів, оцінку якості рекомендацій та продуктивності системи:

1) Збір та підготовка даних:

– збереження даних про користувачів, елементи та їх відношення;

– витягнення та попередня обробка характеристик або атрибутів елементів;

2) Реалізація моделі на основі вмісту:

– створення профілю користувачів на основі їх взаємодії та властивостей елементів;

– розробка алгоритму рекомендацій на основі вмісту, враховуючи схожість між характеристиками елементів та індивідуальними вподобаннями користувачів;

3) Реалізація колаборативного фільтрування:

– визначення схожості між користувачами та елементами на основі їх взаємодій або рейтингів;

– розробка алгоритму колаборативного фільтрування для отримання рекомендацій на основі спільності та популярності;

4) Інтеграція моделей:

– розробка механізму комбінування рекомендацій з моделі на основі вмісту та колаборативного фільтрування;

– врахування ваги або рейтингу моделей при формуванні кінцевих рекомендацій;

5) Настроювання параметрів:

– визначення та налаштування параметрів моделей на основі вмісту та колаборативного фільтрування;

– оптимізація параметрів для досягнення кращої точності та ефективності рекомендаційної системи;

6) Оцінка якості та продуктивності:

– використання метрик, таких як точність, покриття, затримка тощо, для оцінки якості рекомендацій;

– вимірювання продуктивності системи, таких як час обробки запитів або завантаження ресурсів.

Варто зазначити, що при реалізації подібного алгоритму відбудеться інтеграція моделей [75], яка здійснюється за допомогою функції hybrid_recommendation. Вона обчислює вектор користувача з моделі на основі вмісту та знаходить найближчих сусідів користувача з моделі колаборативного фільтрування. Після цього розраховується схожість між користувачем і елементами, а рекомендовані елементи сортуються за спаданням схожості.

Завдяки цій комбінації моделей, код може забезпечити персоналізовані рекомендації для користувачів на основі їхніх взаємодій та характеристик елементів. Це відповідає поставленим у дослідженні питанням щодо розробки ефективної рекомендаційної системи та використання моделей на основі вмісту та колаборативного фільтрування для поліпшення якості рекомендацій.

1.5 Висновки за першим розділом

За підсумками виконаного огляду наукових джерел та аналізу існуючих моделей та методів, що застосовуються для розробки алгоритмів для рекомендаційних систем можна зробити наступні висновки:

1) було виконано загальний огляд рекомендаційних систем та їх видів, включаючи аналіз вмісту, колаборативне фільтрування, гібридні та методи на основі популярності;

2) поглиблено досліджені Content-based алгоритми та алгоритми колаборативної фільтрації. Це дозволило краще розуміти принципи формування нових підходів до створення моделей рекомендаційних механізмів та їх алгоритмізацію. В результаті дослідження було зроблено висновок щодо необхідності розробки алгоритму, який комбінуватиме моделі різних типів в рекомендаційних системах гібридного типу на основі вмісту та колаборативного фільтрування;

3) в процесі теоретичного дослідження наукових джерел було виявлено проблему нестабільної продуктивності впроваджених моделей рекомендаційних систем. На основі цього було запропоновано варіанти вирішення цієї проблеми шляхом використання певних визначних факторів, що приваблюють користувача при виборі якогось продукту, але при тому можуть не бути визначальними, щоб остаточно здійснити покупку;

4) запропоновані варіанти вирішення проблем нестабільної продуктивності та вдосконалення роботи неоднорідних моделей відкрили шляхи для подальших досліджень та розвитку в цій області, а також дозволили обрати та довести методику дослідження, що відповідає поставленим задачам, вибрати підходи, методи та засади для вирішення поставлених у роботі задач;

5) Матеріали, що подані в розділі, знайшли своє відображення у науковій статті та представлені у вигляді доповіді на конференції [76 – 77].

РОЗДІЛ 2
ОСОБЛИВОСТІ СТРУКТУРУВАННЯ ПРОЦЕСІВ РЕКОМЕНДАЦІЙНОЇ ПІДТРИМКИ, ЩО БАЗУЮТЬСЯ НА МАРКЕРАХ КОРИСТУВАЧА

2.1 Структуризація процесів рекомендаційної підтримки рішень при різних інтересах користувачів

Під маркером користувача в цій роботі розуміються певний визначні фактори, які є пріоритетними при виборі окремого продукту споживачем на визначеному відрізку часу [78]. Користувач при виборі продукту має свій особистий інтерес та вирішує певну проблему, яка може бути виражена, як послідовність взаємопов'язаних кроків [79 – 80]. Користувач при цьому дбає не стільки про формулювання кінцевого рішення, скільки про те, що пов’язане з цим процесом і у підсумку з нього випливає. Для вирішення проблеми потрібно мати вибір варіантів рішення. Тому, хоча процес вирішення проблеми алгоритмічно можна представити в п’ять етапів [81], фактичне число етапів визначається самою проблемою.

І на кожному етапі повинна бути своя система рекомендацій для обґрунтування вибору наступного кроку у вирішенні поставленої задачі.

Якщо виходити з визначення рекомендаційної системи [71, 82], то подібні системи є підкласом систем фільтрації інформації, що дозволяють побудувати певний рейтинг за запитами чи уподобаннями. Аналізуючи процеси підтримки прийняття рішень, можна зрозуміти, що в основу покладено наукові методи оцінки результатів праці, поведінки, успішності та ранжування за рядом показників. І саме це є основою для побудови алгоритмів системи рекомендацій.

При структуруванні процесів рекомендаційної системи відбувається орієнтація саме на науковий метод обґрунтування управлінських рішень. Тому основні процеси вирішення задачі управління можна розписати таким чином:

1) Спостереження. Для вирішення проблеми, що виникла і за якою слід прийняти рішення, необхідно зібрати повну інформацію про подію, провести її аналіз. Користувач часто не збирає цю інформацію. Для цього значна частина рекомендованої інформації буде надаватися власними підрозділами організацій. Аналіз інформації також проводиться, як правило, сторонніми особами, спеціальними службами організації або сторонніми організаціями, що надають такі послуги на ринку [83].

2) Формулювання гіпотези. На цьому етапі користувач виявляє та розглядає альтернативні варіанти дій, досліджує наслідки, прогнозує перспективний розвиток події при прийнятті того чи іншого рішення.

3) Верифікація. На фазі підтвердження достовірності гіпотези користувач перевіряє гіпотезу, спостерігаючи результати рішень, які були прийняті при вирішенні подібних питань. Наприклад, користувач може збільшити запаси на величину, використовуючи аналіз рішень керівництва країни щодо економічної політики або з рекомендації близьких, які більше володіють господарськими навичками. Якщо при цьому запаси не падають і не зростають понад міру, гіпотезу слід визнати правильною. Якщо все ж виникає нестача продукції із зростанням попиту або запаси зростуть настільки, що витрати за їх утримання стають надмірними, гіпотезу слід визнати недостовірною. Тобто, у цьому випадку потрібна додаткова інформація, яка і може бути надана рекомендаційною системою підтримки прийняття управлінських рішень (рис. 2.1).

Якщо аналізувати наведену схему, то можна зазначити, що організація складається із взаємопов'язаних частин. Тому при розробці рекомендаційної системи та алгоритмізації процесів у ній слід дотримуватися системної орієнтації. А це додає додаткові взаємопов’язані кроки при побудові алгоритму рекомендаційної системи. Для вирішення будь-якого питання

користувач повинен прийняти не одиничне рішення, а здійснити сукупність виборів. Тому фактично число етапів визначається самою проблемою, яку вирішує кожен окремий користувач. Тоді схема алгоритмізації процесів рекомендаційної системи для підтримки прийняття рішень, наведена на рис. 2.1, є лише узагальненим фрагментом деталізації окремого одного кроку роботи рекомендаційної системи такого класу.

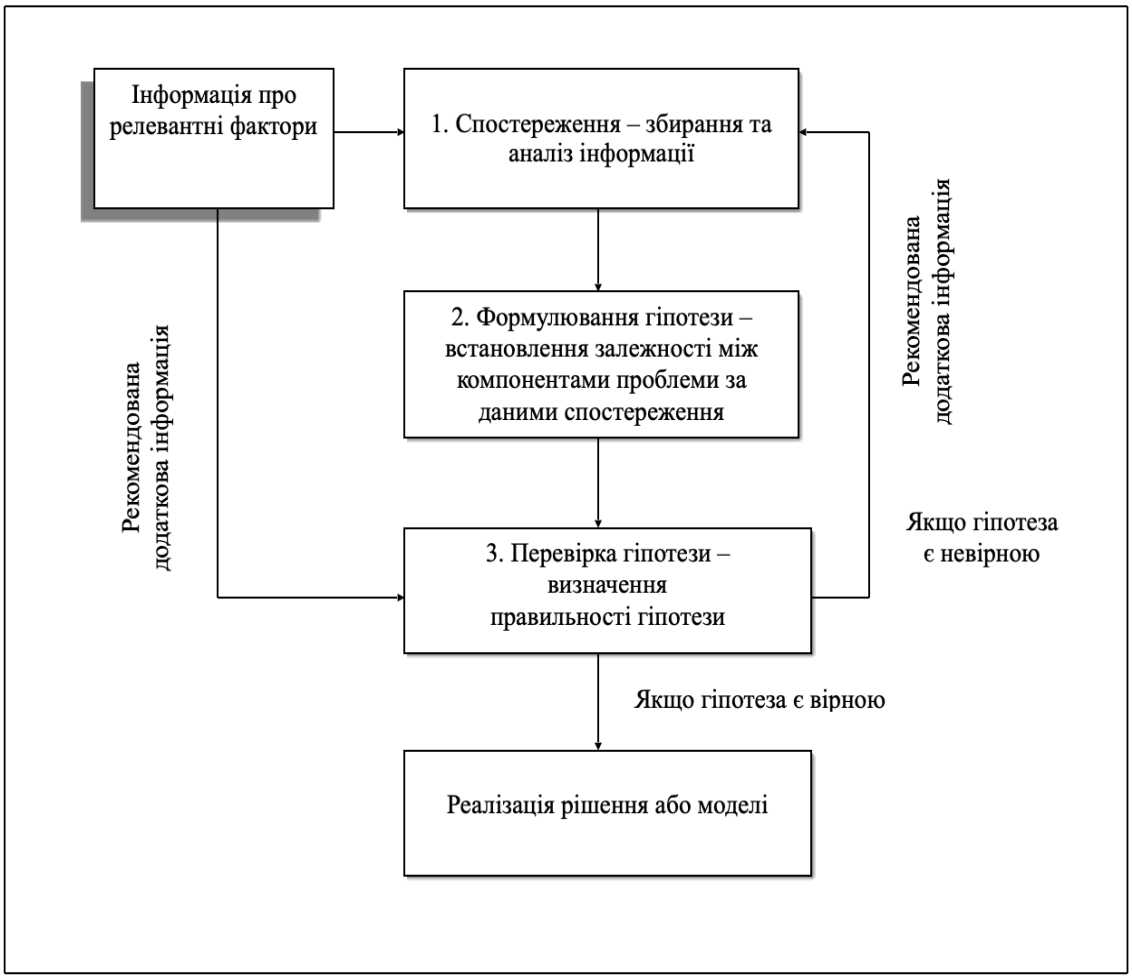


Рисунок 2.1 – Схема алгоритмізації процесів рекомендаційної системи для підтримки прийняття рішень

Якщо абстрагуватися саме на одному кроці алгоритму рекомендаційної системи для автоматизації роботи користувача, а саме – на діагностиці проблемного питання, яке відбувається на кроці 1 рис. 2.1, то можна побачити складність розробки алгоритмів рекомендаційних систем, орієнтованих на

широке коло різноспрямованих користувачів, а не на одного користувача чи групу користувачів, що цікавляться якимось одним продуктом.

Складність полягає в тому, що чітко визначити базовий критерій, або, інакше, маркер користувача, при виборі якогось продукту часто важко: при розробці рекомендацій персоніфіковані дані можуть використовуватися тільки за спеціальним дозволом, а не персоніфіковані дані занадто узагальнюють можливий вибір користувачем товару чи послуги. Крім того, при переборі продуктів користувачем виникають сотні взаємозалежностей, бо різні організації, що надають певні продукти на ринок, між собою пов'язані не будуть, за виключенням зв'язків зовнішнього середовища [83]. Саме тому тільки етап спостереження є окремим алгоритмом процесу з усвідомлення і встановлення перешкод або наявних можливостей. Тобто, відбувається вирішення системи рівнянь, що описують процеси за різними інтересами учасників цих процесів, з декількома обмеженнями для прийняття остаточного рішення за вибором продукту.

Виявлення подібності різних інтересів учасників процесу допомагає користувачу системно розглянути проблему, яка виникла. Але збільшення кількості інформації не обов'язково підвищує якість рішення [84]. Саме тому у процесі спостережень важливо виділяти релевантну інформацію, що стосується саме того питання, яке розглядається, враховуючи максимальну точність і відповідність проблемі. Користувачу може бути непросто отримати вичерпну точну інформацію із проблеми, особливо тієї, що стосується задоволення його особистих різних інтересів. А якщо задача вирішується щодо надання рекомендацій двом різним користувачам, які зацікавлені одним і ми самим продуктом, але маркери у кожного з користувачів є різними, то подібну задачу вже можна розглядати через матричну гру із двома гравцями А і В. Це – парна гра, яка може бути описана заданою функцією Р $\phi(A_i, B_j) = a_{ij}$, $i = 1, \dots, m$, $j = 1, \dots, n$.

$A = \{a_{ij}\}$ – матриця гри. Припустимо, що гравець A вибрав стратегію A_i, тоді в найгіршому варіанті гри його виграш складе мінімум a_{ij}.

Передбачаючи таку можливість, гравець A намагається одержати максимально можливий виграш: $\alpha = \max_i \min_j a_{ij}$.

Стратегія A_{i0}, що забезпечує величину виграшу α, є максимінною. Число α виступатиме нижньою ціною гри. Якщо гравець B зробив вибір B_j, у найгіршому варіанті він програє величину $\max_i a_{ij}$. Передбачаючи це, гравець B намагається зменшити свій можливий програш: $\beta = \min_j \max_i a_{ij}$.

Стратегія B_{j0}, за якою буде досягнута ця величина, виступатиме мінімаксною стратегією з числом β як верхня ціна гри.

Фактично виграш гравця A (програш гравця B) обмежений нижньою і верхньою ціною гри при розумних діях партнера (інтервал $[\alpha\,;\beta]$).

У разі $\alpha = \beta$ спільне дорівнює значенню гри V, тобто, $\alpha = \beta = V$. У цьому випадку гра є цілком визначеною. Для матриці це число називається сідловою точкою $V = a_{i0j0}$.

Але, якщо гра кінцева, то вона має, принаймні, одне рішення [85], яке може знаходитися в області змішаних стратегій:

$$\sum_{i=1}^{m} x_i^* \, a_{ij} \geq V, \;\; \sum_{j=1}^{n} a_{ij} \cdot y_j^* \leq V, \qquad (2.1)$$

де:

X^*, Y^* – оптимальні ймовірності стратегій;

V – значення гри чи ціна гри.

У рамках поставлених у роботі задач можна розглянути найпростіший окремий випадок:

$$n = 2, \;\; m = 2, \;\; A = \begin{Vmatrix} a_{11} & a_{12} \\ a_{21} & a22 \end{Vmatrix}.$$

Якщо сідлової точки немає, необхідно застосовувати змішані стратегії. У цьому випадку можна ввести до розгляду (2.1) два вектори ймовірностей:

$$X = \{x_1 ; x_2\}, \quad Y = \{y_1 ; y_2\}, \tag{2.2}$$

$$\begin{cases} a_{11}x_1 + a_{21}x_2 = V, \\ a_{21}x_1 + a_{22}x_2 = V, \\ x_1 + x_2 = 1, \end{cases} \tag{2.3}$$

де необхідно знайти x_1, x_2, V.

Система (2.3) лінійна і може бути вирішена будь-якими способами. Наприклад:

$$\begin{cases} x_1 = \dfrac{a_{22} - a_{21}}{a_{11} + a_{22} - a_{12} - a_{21}}, \\ x_2 = \dfrac{a_{11} - a_{12}}{a_{11} + a_{22} - a_{12} - a_{21}}, \\ V = \dfrac{a_{11}a_{22} - a_{12}a_{21}}{a_{11} + a_{22} - a_{12} - a_{21}}. \end{cases}$$

Друга система для ймовірностей Y:

$$\begin{cases} a_{11}y_1 + a_{12}y_2 = V, \\ a_{21}y_1 + a_{22}y_2 = V, \\ y_1 + y_2 = 1. \end{cases} \tag{2.4}$$

Розв’язок задачі (2.4):

$$A = \begin{Vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{Vmatrix}. \tag{2.5}$$

Рішення гри (2.5) з матрицею [2 × 2] можна знайти графічно (рис. 2.2) за допомогою наступних побудов.

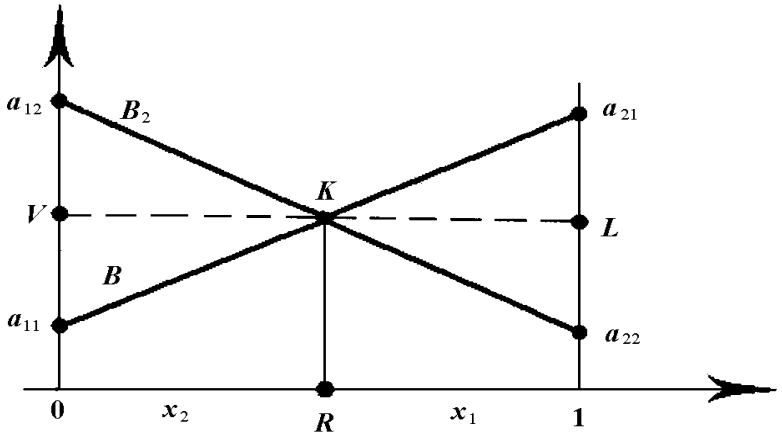


Рисунок 2.2 – Розв’язок гри у змішаних стратегіях визначення переваги

Але розглянутий процес передбачає чіткі варіанти за маркером користувача для кожного гравця. Щоб деталізувати вибір кожного гравця за низкою маркерів варто на осі абсцис відкласти відрізок, що дорівнює одиниці. На цьому відрізку вже передбачається дві можливі точки рішень за різними маркерами користувача. Лівий кінець відрізка (точка x = 0) відповідає стратегії A_1 правий – стратегії A_2. Проміжні точки x відповідають деяким змішаним стратегіям $(x_1;\ x_2)$, де $x_1 = 1 - x,\ x_2 = x$. Тобто, розглядаються гравцями різні варіанти вибору в залежності від того, якими маркерами у даному випадку оперує користувач.

На кінцях вибраного відрізку проводяться прямі, перпендикулярні осі абсцис, на них відкладається виграш при відповідних чистих стратегіях. Якщо гравець В застосовує стратегію B_1, то виграш при використанні чистих стратегій A_1, A_2 становить відповідно $a_{11};\ a_{21}$. Ці точки на прямих позначаються та поєднуються з прямою B_1, B_1. Якщо розглянути випадок, що гравець А застосовує змішану стратегію, то його виграшу відповідає деяка точка М лежить на цій прямій (рис. 2.3).

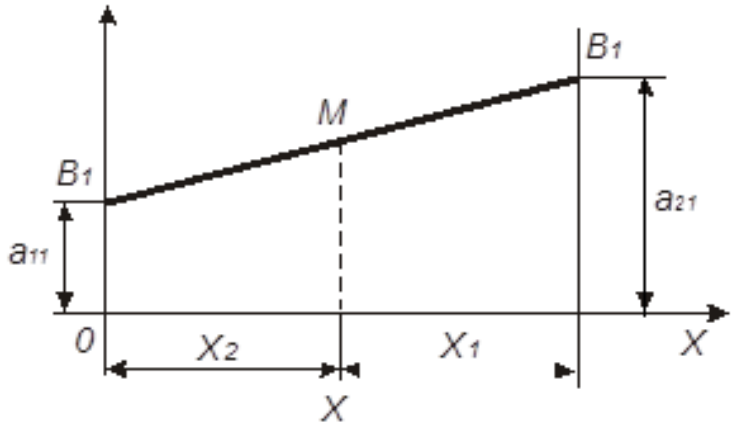


Рисунок 2.3 – Користувач В застосовує стратегію B_1

Аналогічно можна деталізувати пряму B_2 , B_2 , відповідну стратегії B_2 гравця В (рис. 2.4). Ламана В1КВ2 – нижня межа виграшу, одержуваного гравцем А. Точка К, в якій він максимальний, визначає ціну гри і її рішення.

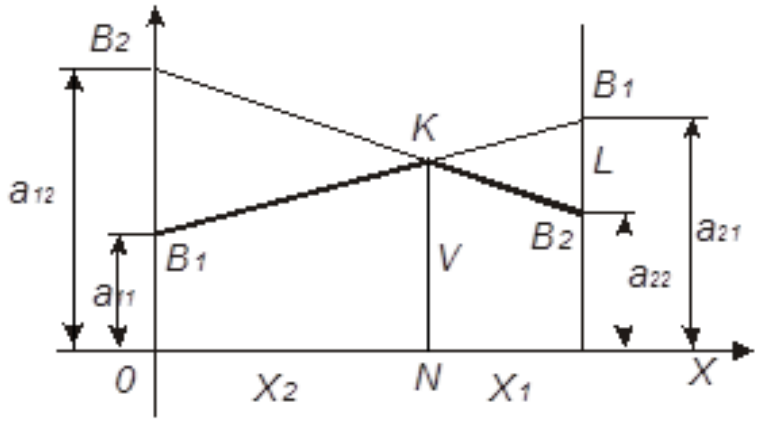


Рисунок 2.4 – Деталізація розв’язку гри двох користувачів

Для знаходження оптимальної стратегії гравця В можна скористатися наступними формулами [85]:

$$y_1 = \frac{LB_2}{LB_2 + LB_1}, \qquad y_2 = \frac{LB_1}{LB_2 + LB_1}. \tag{2.6}$$

У справедливості цих співвідношень можна переконатися, якщо у формули, що виражають y_1 і y_2, підставити замість LB_2, LB_1 їх значення. Маємо:

$$LB_2 = \upsilon - a_{22};\ LB_1 = a_{21} - \upsilon. \qquad (2.7)$$

Виходячи з (2.7) можна розглянути задачу мінімізації верхньої межі виграшу для гравця В, помінявши місцями при вирішенні гравців А і В.

За допомогою наведеного геометричного методу можливо знайти рішення гри [2 × n], де n виступатиме означенням стратегії за кожним маркером користувача та обумовлюватиме вибір рішення. Кожною з n стратегій гравця В виступатиме пряма. Побудував ці прямі, знайдемо нижню межу виграшу гравця А. Точка К, що розташована на нижній межі, для якої величина виграшу найбільша, визначає ціну гри і її рішення. При цьому визначаються активні стратегії гравця В – прямі, які перетинаються в точці К.

Подібна інтерпретація з графічним рішенням задачі не зручна для подальшого викладення за допомогою мов програмування, але вона дає можливість зрозуміти, яким чином будуються стратегії гравців з використанням різних маркерів користувача для отримання найвищої вигоди при виборі певного продукту. Якщо брати до уваги лише активні стратегії гравця, то матриця гри міститиме два рядки і два стовпці, тобто, представляє собою матрицю [2 × 2] (2.5). У цьому випадку подібну задачу з вибору альтернативи і, відповідно, розробці рекомендації щодо цього, можна вирішити за допомогою тензорів [86], розглядаючи їх як багатомірні масиви з n-мірними матрицями [87]. Створюючи за маркерами користувача алгоритм колаборативної фільтрації з результатами, що розміщуються у n-мірних матрицях, за допомогою опенсорсної бібліотеки TensorFlow від Google [88], можна розробити систему за технологією машинного навчання для вирішення задачі прийняття рішень за рекомендаціями. Розглядаючи подібну задачу при

побудові рекомендаційного алгоритму можна зробити висновок щодо обмежень, які варіюються і залежать від ситуації та конкретних дій користувача. У цьому випадку рекомендаційна система може бути зорієнтована саме на оцінці альтернативних варіантів вибору саме за низкою маркерів користувача.

2.2 Побудова лінійної моделі рекомендації за декількома маркерами користувача

Щоб побудувати модель за декількома маркерами користувача можна задати до розгляду наступну ситуацію: є деяке число осіб, наприклад, тридцять, які роблять вибір одного з двох продуктів А та В, характеристики щодо кожного маркеру користувача та часом здійснення вибору за кожним продуктом наведені у таблиці (табл. 2.1).

Приймаємо, що кожен користувач оцінює для себе продукт за трьома маркерами, які визначаються за шкалою від 1 до 10 балів. Для прийняття рішення, користувач може застосувати від 1 до 3 маркерів з цілим значенням за окремим маркером або в долях (з округленням до цілого числа). Крім того, визначається час в секундах, під час якого користувач знаходився на сторінці опису кожного продукту, а також час, коли він здійснював пошук в Інтернеті за назвою продукту відгуків стосовно товару або детальний опис характеристик.

Якщо розглядати продукт А, то за першим маркером користувача (функціональність) присвоюється a_1 балів, другим маркером (надійність в роботі) – a_2 балів, третім маркером (ціна) – a_3 балів. За другим продуктом В b_1, b_2, b_3 балів. Час перегляду сторінки, де надані всі характеристики щодо продукту А витрачено t_1 секунд, продукту В – t_2 секунд, додатковий пошук

інформації про продукти зайняв t_3 секунд. Загальна сума балів за критеріями, яка стала визначальною для вибору користувача за продуктом А становить α балів, а продукту В – β балів.

Таблиця 2.1 – Таблиця оцінки вибору двох продуктів за трьома маркерами користувача

Користувач	a_1	a_2	a_3	b_1	b_2	b_3	t_1	t_2	t_3	α	β
1	5	3	2	10	3	3	505	393	348	7	10
2	7	6	1	3	3	2	1365	1245	650	6	2
3	6	4	3	2	3	4	600	520	600	6	2
4	5	4	3	3	3	4	750	630	700	5	6
5	8	6	3	10	3	2	840	870	560	6	10
6	3	3	2	10	3	5	273	300	380	4	10
7	2	3	3	10	6	7	438	747	812	7	10
8	4	3	2	3	4	6	480	444	546	2	4
9	4	3	3	3	4	10	440	393	450	6	10
10	2	3	2	3	6	8	428	672	672	3	8
11	10	8	5	4	7	9	784	552	567	4	10
12	10	10	3	10	5	6	684	690	558	6	10
13	8	7	4	3	6	9	864	864	945	2	3
14	10	8	5	3	4	3	671	588	423	5	2
15	10	11	9	4	5	10	1095	865	1080	5	10
16	2	3	5	4	2	1	800	600	510	1	4
17	1	2	2	3	5	2	600	1100	560	3	5
18	1	2	3	4	3	4	700	900	1260	1	6
19	2	4	1	10	2	4	600	800	900	2	10
20	3	2	5	2	4	1	600	800	1100	2	2
21	2	1	2	5	3	2	1100	600	600	1	2
22	2	1	3	3	4	4	900	700	1700	4	3
23	4	2	1	10	3	4	800	600	675	3	10
24	6	2	1	2	4	1	500	450	375	6	2
25	6	7	1	3	3	2	1245	1365	650	6	2
26	2	3	4	6	4	3	546	444	480	2	3

Продовження табл. 2.1

Користувач	a_1	a_2	a_3	b_1	b_2	b_3	t_1	t_2	t_3	α	β
27	10	3	10	5	6	3	690	558	684	10	6
28	5	10	8	3	3	4	423	671	588	10	2
29	3	5	2	3	2	3	393	505	348	4	2
30	4	5	3	3	10	4	630	750	700	4	4

Щоб визначити, якому продукту – А чи В надасть перевагу користувач, варто пам’ятати, що маркери користувача у такій задачі виступлять обмеженнями, а найменші витрати часу при здійсненні вибору – перевагою. Для підприємства буде важливим продати деякий товар, що позначений в задачі абстрактним продуктом А чи В, тому слід виявити область припустимих рішень і за маркерами користувача та мінімумом за витратами часу визначити мінімальну точку, яка означатиме, що за цією межею найменша вірогідність здійснення цільової дії.

Можна представити цю задачу на площині [53, 85, 89] із застосуванням двох змінних: x_1 та x_2. Нехай потрібно мінімізувати лінійну функцію поведінки користувача: $F = c_1 x_1 + c_2 x_2$ при обмеженнях:

$$\begin{cases} a_{i1}x_1 + a_{i2}x_2 \le b_i & (i = 1,\ldots,m_1) \\ a_{i1}x_1 + a_{i2}x_2 \ge b_i & (i = m_1+1,\ldots,m_2) \quad x_1, x_2 \ge 0. \\ a_{i1}x_1 + a_{i2}x_2 = b_i & (i = m_2+1,\ldots,m) \end{cases} \tag{2.8}$$

Геометрично перші m_2 обмеження являють собою на півплощини з граничною прямою $a_{i1}x_1 + a_{i2}x_2 = b_i$. Обмеження, що залишилися – прямі. Областю припустимих рішень є їхнє перетинання. Область припустимих рішень може являти собою замкнутий чи відкритий багатокутник, відрізок, точку і т.інш. Нехай область припустимих рішень – обмежений замкнутий багатокутник.

Лініями рівня цільової функції є набір паралельних прямих. Значення функції зростає в напрямку градієнта *grad(F)=(c_1,c_2)* убуває в напрямку антиградієнта *-grad(F)=(-c_1,-c_2).* Далі варто побудувати лінію з нульовим рівнем: $c_1x_1 + c_2x_2 = 0$.

У випадку мінімізації цільової функції за часом прийняття рішень можна пересувати цю лінію паралельно самій собі в напрямку антиградієнта таким чином, щоб вона перетиналася з областю припустимих рішень. Тоді крайнє положення, що займе ця лінія, визначає точку мінімуму. Саме за цією точкою вірогідність здійснення цільової функції за набором маркерів користувача наближається до нуля.

Розглянемо зазначене на прикладі з прогнозуванням того, скільки одиниць продукту А чи В буде придбано у групі користувачів зі 100 осіб за виставленими маркерами. Для цього будуть використані усереднені дані за вибіркою (за табл. 2.1):

а) для продукту А:

функціональність (a_1) – 7 балів,

надійність в роботі (a_2) – 4 бали,

ціна (a_3) – 8 балів;

б) для продукту В:

функціональність (b_1) – 6 балів,

надійність в роботі (b_2) – 9 балів,

ціна (b_3) – 3 бали;

в) на перегляд витрачено часу: t_1 – 864 с., t_2 – 945 с., додатковий пошук t_3 – 864 с.;

г) мінімальна кількість балів за критеріями, яка стала визначальною для вибору користувача: α – 2 бали, β – 3 бали.

За зазначеними умовами можна змоделювати область вірогідності здійснення цільової дії за кожним з продуктів за максимально припустимою

тривалістю витрат часу для пошуку інформації про продукт (при умові надання рекомендації щодо даного продукту).

Позначимо:

x_1 – кількість продукту А;

x_2 – кількість продукту В;

Цільова функція – витрати часу при виборі за наявності рекомендацій щодо продукту А та В:

$$F = 2 \cdot x_1 + 3 \cdot x_2 \to \max .$$

В задачі наявні 3 обмеження – маркери користувача за функціональністю, надійністю та ціною:

$$\begin{cases} 7x_1 + 6x_2 \le 864 \\ 4x_1 + 9x_2 \le 945 \\ 8x_1 + 3x_2 \le 864 \end{cases}, \quad x_1, x_2 \ge 0 .$$

Кожне обмеження геометрично є на півплощиною.

Граничні лінії:

1) $7x_1 + 6x_2 = 864$ $\quad (A_1(123;0); A_2(0;144))$;
2) $4x_1 + 9x_2 = 945$ $\quad (B_1(236;0); B_2(0;105))$;
3) $8x_1 + 3x_2 = 864$ $\quad (C_1(108;0); C_2(0;288))$;
4) $x_1 = 0$;
5) $x_2 = 0$.

Побудуємо на графіку область припустимих рішень (рис. 2.5). Областю припустимих рішень є п’ятикутник OB_2DEC_1. Далі складається вектор $N = grad\,\mathrm{F} = (2, 3)$ з коефіцієнтів цільової функції та будується в точці О.

Проводиться пряма через точку О перпендикулярно вектору $\overline{N}$ (нульова лінія рівня прийняття рішення щодо вибору якогось із рекомендованих продуктів). Лінія з нульовим рівнем: $2x_1 + 3x_2 = 0$ або $x_1 = -1{,}5x_2$.

Виконується паралельне перенесення цієї прямої в напрямку вектору $\overline{N}$ таким чином, щоб вона перетиналася з областю припустимих рішень. Остання спільна точка прямої і многокутника (точка «прощання») визначає розв'язок задачі – точка D (це точка максимуму). Знайдемо її координати, вирішуючи систему рівнянь лінії, що перетинаються в цій точці:

$$\begin{cases} 7x_1 + 6x_2 = 864 \\ 4x_1 + 9x_2 = 945 \end{cases}$$

$$\begin{cases} x_1 = 54 \\ x_1 = 81 \end{cases} \Rightarrow m.D(54;81) - точка\ максимуму .$$

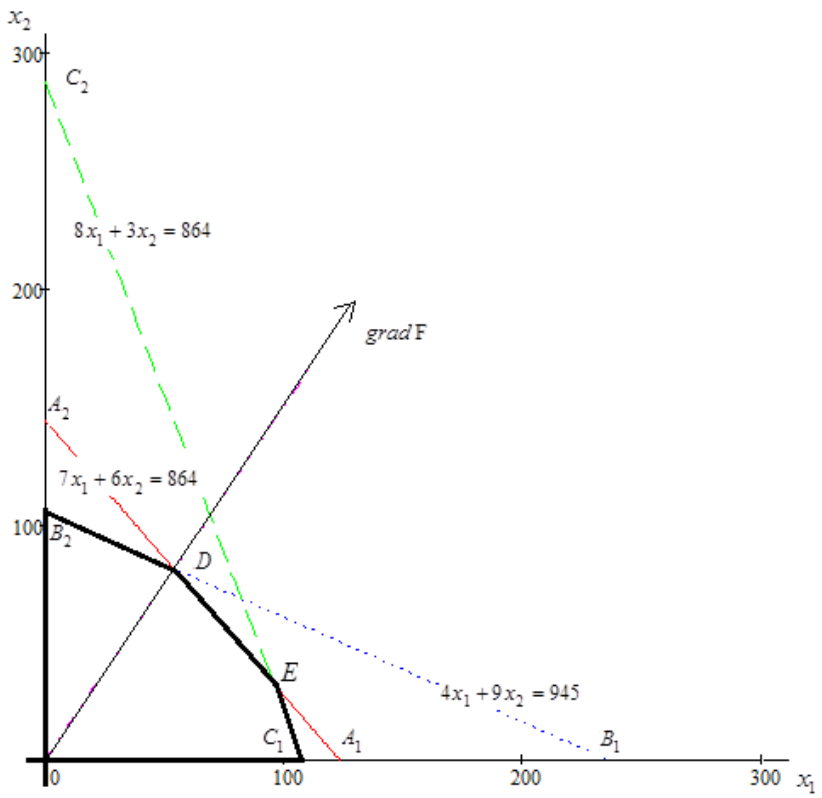


Рисунок 2.5 – Рішення поставленої задачі

У випадку, представленому на рисунку, максимальний час, який готові витратити користувачі при виборі продукту за наявною рекомендацією складе 351 с. або 5,85 хв. Цього часу достатньо, щоб перейти на сторінку, куди направляє рекомендація, прочитати характеристики, натиснути кнопку вибору продукту до кошика та внести всі необхідні дані для доставлення продукту. При цьому зі 100 користувачів з більшою ймовірністю за наявними характеристиками від 54 (мінімальна кількість користувачів) до 81 користувача (максимальна кількість користувачів з вибірки у 100 осіб) обиратиме товар В за маркерами користувача «надійність в роботі» та «ціна».

2.3 Побудова переходів при формуванні вибору за маркерами користувача

2.3.1 Поведінка користувача за умов знаходження у системі зі стаціонарним станом

Під системою зі стаціонарним станом у цій роботі будемо розуміти технічну систему великих торгівельних майданчиків на Інтернет-платформах (Розетка, Prom, OLX; книжкові магазини: Книгарня «Є», YAKABOO, КСД та інші). Особливістю цих торгівельних майданчиків є те, що за умови реєстрації або ID користувача формується база попередніх переглядів, на основі чого рекомендуються аналогічні або супутні товари. У цьому випадку за матрицею інтенсивності переходів користувача формується вибірка за ключовими ознаками стосовно товару, який цікавив користувача раніше. Тобто, у даному випадку, технічна система має n можливих станів, що реалізуються за матрицею інтенсивностей переходів користувача.

Тобто, необхідно знайти значення деяких змінних, які характеризуватимуть стан системи у певному проміжку часу. На практиці це

може проявлятися рекомендацією відвідувачу системи щодо пропозицій товарів та послуг, аналогічних переглянутим раніше.

Подібну задачу можна вирішити методом Рунге-Кутти 4-го порядку [90] для систем диференціальних рівнянь, у яких права частина не залежить від часу. Необхідно чисельно знайти у момент часу t значення перемінних x_1, x_2,...,x_n, що задовольняють системі диференціальних рівнянь:

$$\begin{cases} \frac{dx_1}{dt} = F_1(x_1, x_2, ...x_n); \\ \frac{dx_2}{dt} = F_2(x_1, x_2, ...x_n); \\ .... \\ \frac{dx_n}{dt} = F_n(x_1, x_2, ...x_n). \end{cases} \quad (2.9)$$

при початкових умовах: $x_i(t=0) = a_i$ (I=1,2,…,n).

Метод Рунге-Кута 4-го порядку полягає у русі по часу від 0 до t за m кроків довжиною $h = \frac{t}{m}$ [91]. Основна формула у векторному вигляді [92]:

$$\vec{x}_{j+1} = \vec{x}_j + \frac{1}{6}\left[\vec{K}_1 + 2(\vec{K}_2 + \vec{K}_3) + \vec{K}_4\right], \quad (2.10)$$

де j=0,1,2,…,m, причому, $\vec{x}_0 = \vec{a} = (a_1, a_2, ..., a_n)$;

$$\vec{K}_1 = \vec{F}(\vec{x}_j) \cdot h;$$
$$\vec{K}_2 = \vec{F}(\vec{x}_j + \frac{1}{2} \cdot \vec{K}_1) \cdot h;$$
$$\vec{K}_3 = \vec{F}(\vec{x}_j + \frac{1}{2} \cdot \vec{K}_2) \cdot h;$$
$$\vec{K}_4 = \vec{F}(\vec{x}_j + \vec{K}_3) \cdot h.$$

Розмірність кожного з наведених векторів дорівнює n.

Метод Рунне-Кути відрізняється невеликою погрішністю, що є пропорційною від h^5. Для підвищення точності і контролю досягнення

потрібної точності необхідно виконувати розрахунки при двох значеннях h (друге значення обирають, як правило, у 2 рази менше за перше). Якщо результати розрахунків не відрізняються у межах потрібної точності – рішення вважають досягнутим.

Для ілюстрування наведеного припустимо, що задана наступна матриця інтенсивностей:

$$\begin{Vmatrix} 0 & 2 & 0 & 0 \\ 1 & 0 & 0 & 2 \\ 0 & 4 & 0 & 1 \\ 0 & 0 & 3 & 0 \end{Vmatrix}$$

На основі матриці інтенсивностей можна побудувати граф переходів (рис. 2.6).

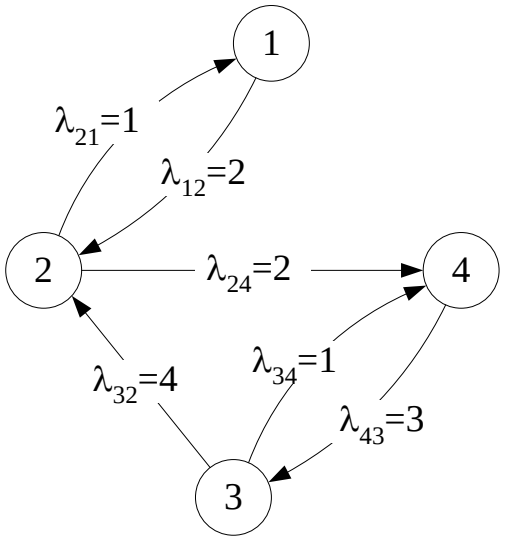


Рисунок 2.6 – Граф переходів за матрицею інтенсивностей переходів на деякому ресурсі, що має стаціонарний стан

Граф станів системи (рис. 2.6) є ергодичним Марківським ланцюгом [93], бо з кожного стану можна попасти до всякого іншого. Для ймовірностей можна використати систему рівнянь Колмогорова, користуючись правилами [94]:

$$\begin{cases} \frac{dp_1}{dt} = -\lambda_{12}p_1 + \lambda_{21}p2 = -2p_1 + p_2; \\ \frac{dp_2}{dt} = \lambda_{12}p_1 - (\lambda_{21} + \lambda_{24})p_2 + \lambda_{32}p_3 = 2p_1 - 3p_2 + 4p_3; \\ \frac{dp_3}{dt} = -(\lambda_{32} + \lambda_{34})p_3 + \lambda_{43}p_4 = -5p_3 + 3p_4; \\ \frac{dp_4}{dt} = \lambda_{24}p_2 + \lambda_{34}p_3 - \lambda_{43}p_4 = 2p_2 + p_3 - 3p_4. \end{cases} \quad (2.11)$$

Після цього укладається система рівнянь у стаціонарному стані для знаходження фінальних ймовірностей. Для цього:

а) усі похідні дорівнюються до 0;

б) замість четвертого рівняння записується умова того, що усі стани створюють повну групу:

$$p_1 + p_2 + p_3 + p_4 = 1. \quad (2.12)$$

Записуємо систему рівнянь для стаціонарного стану у матричному вигляді:

$$\begin{Vmatrix} -2 & 1 & 0 & 0 \\ 2 & -3 & 4 & 0 \\ 0 & 0 & -5 & 3 \\ 1 & 1 & 1 & 1 \end{Vmatrix} \times \begin{Vmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{Vmatrix} = \begin{Vmatrix} 0 \\ 0 \\ 0 \\ 1 \end{Vmatrix}$$

Далі рішення задачі розв’язується у середовищі Excel (Додаток Б). Робочий аркуш наведено на рис. Б1:

а) У комірках (B2:F5) наведено розширену матрицю: матрицю коефіцієнтів з доданням стовпця вільних членів.

б) У комірках (B7:E10) будується зворотна матриця. Формула масиву (B7:E10): =МОБР(B2:E5)

в) У комірках (H7:H10) будується масив фінальних імовірностей шляхом матричного множення. Формула масиву (H7:H10):

=МУМНОЖ(B7:E10;F2:F5).

Програмна реалізація вирішення задачі з метою спрощення процесу тестування створена мовою VBA. Варіант лістинга програми чисельного рішення диференціальних рівнянь методом Рунге-Кута з прив'язкою до Excel наведено на рис. Б2. Лістинг підпрограми розрахунків функцій наведено на рис. Б3. Робочий лист Excel, що зв'язаний з програмою, для тестування запропонованого методу наведено на скріншоті (рис. Б4). У комірках B11:E11 містяться початкові наближення імовірностей. Приймається, що у початковий момент часу система находиться у стані 1 з імовірністю 1, а імовірності інших станів дорівнюють 1. У комірках B12:E12 містяться рішення – значення імовірностей у момент часу t, що міститься у комірці A12. Для його одержання слід пустити програму (для цього створена кнопка).

У комірках A4:D4 наведені довідкові значення фінальних ймовірностей для контролю розрахунків.

Розрахунки проведено таким чином:

а) у комірку A12 уводять значення часу;

б) у програмі встановлюють відносно велике значення кількості кроків m (100);

в) натискують кнопку, одержують рішення;

г) збільшують кількість кроків m=200, одержують нове рішення;

д) якщо різниця між двома рішеннями менше, ніж 0,001 – рішення вважається знайденим. У протилежному випадку збільшують кількість кроків і повторюють;

е) порівнюють, суттєво чи ні, відрізняється одержане рішення від довідкового значення фінальних імовірностей. Якщо різниця – менше за 0,01 – стаціонарний стан не змінено. Як випливає з рис. Б4, через 4 с. можна вважати, що досягнуто стаціонарний розподіл імовірностей і користувач отримав рекомендацію, яка задовольняє його попередні уподобання;

ж) виконуються розрахунки у проміжних точках, результати переносяться до таблиці, будуються графіки (рис. Б4). Виходячи з графіку, уточнюємо період переходу до нового стаціонарного стану (у наведеному комп'ютерному експерименті це – практично 2,5 с.).

2.3.2 Аналіз вибору користувача як випадкової події

Але при навігації мережею Інтернет виникають ситуації (як правило, рекламні [79 – 80]), коли при перегляді якоїсь сторінки виникає пропозиція товару чи послуги, що якимось чином перетинаються з тематикою сторінки, на якій перебуває користувач. Проте, одночасно, пропонований продукт повинен зацікавити користувача, тобто у рекомендації повинен відображатися маркер, притаманний саме цьому користувачу.

У зазначеній ситуації керівну роль може відігравати закон розподілу [95] – множина можливих подій з ймовірностями їх настання.

Припускаємо, що є деяка кількість відвідувачів сайту N1, що містить можливість надання N2 рекомендацій на деякій сторінці. Середній час переходу користувача до потрібної сторінки складає t_1 с. Середній інтервал часу між переходами користувачів складає t_2 с. Середній прибуток від операції, коли спрацьовує рекомендація від одного споживача складає 2 у.о. Коли споживач не зацікавлений у продукті, що пропонується за рекомендацією, він залишає Інтернет-сторінку.

Час, протягом якого споживач в середньому перебував на сайті – 10 хв. (600 с). Із зазначеного виникає питання – чи збільшення кількості рекомендацій дозволить збільшити прибуток від реалізації продукту?

Для цього слід розглянути такі варіанти:

а) час між приходом окремих користувачів і час перебування одного окремого користувача на сайті розподілені за показовим законом;

б) час між приходом окремих користувачів і час перебування одного окремого користувача на сайті за рівномірним законом.

Розрахунки можна провести в середньому за 10 хв.

Для моделювання на прикладі показового розподілу часу приймаються наступні умови: є два користувача та сім можливих рекомендацій за видами продуктів, що згадуються в різних ракурсах на інтернет-сторінці. В середньому один користувач затримується на сторінці на 30 с., а проміжок часу між переходами споживачів до сторінки – 10 с.

Враховуючи, що в середньому на сайті користувач перебуває все ж 10 хв, для підвищення точності розіб'ємо 1 хв на 10 тактів. Тобто, середній час перебування 30 тактів, а проміжок часу між клієнтами – 100 тактів.

Основна ідея рішення такої задачі полягає у знаходженні середньої кількості користувачів протягом 10 хв., кожному з користувачів при цьому видається, наприклад, 2 – 7 рекомендацій від системи. А далі вираховується доля ймовірності, з якою кожен користувач зробить вибір продукту, що пропонується. На цій основі вже можна буде розробити саме такі рекомендації, які з найбільшою долею ймовірності будуть сприйняті користувачем позитивно.

Алгоритм реалізації такої задачі може бути наступним:

1) задається значення перемінної NW – кількість спостережень;

2) визначається масив PER(1 to 7). У цьому масиві у PER(1), PER(2) буде міститись час перебування користувачів (0-якщо їх немає). У PER(3) ÷PER(7) – вказується поточна кількість користувачів, які здійснюють переходи: 0 – немає, 1 – є;

3) визначаються змінні:

NW – кількість спостережень – циклів випадкових подій;

N_C – кількість часу до переходу наступного споживача;

N_Yes – кількість споживачів у поточний момент часу;

N_No – кількість споживачів, що вже прийшли на сайт, але не здійснили перехід до сторінки з рекомендаціями;

N-Per – загальна кількість рекомендацій у системі (7);

N_Barb – кількість користувачів на сторінці з рекомендаціями;

tcl – середній проміжок часу між переходами;

tser – середній проміжок часу передування користувача на сторінці з рекомендаціями;

4) для кожного циклу спостережень у 10 хв. від i=1 до i=NW на «нульовому» кроці перед початком розрахунків:

а) перемінним N_Yes, N_No присвоюється значення «0»;

б) генерується час появи першого користувача N_C;

в) генерується час перебування користувача – PER(1);

5) У циклі випробувань для кожного такту з 1 до 600 виконуються таки дії:

а) перевіряється, чи дорівнює N_C нулю?

б) імітується поведінка системи у наступний такт часу;

б) Якщо перемінна N_C дорівнює нулю:

а) генерується нове значення N_C;

б) визначається змінна NS=0, проводиться перебір змісту масиву PER(i). Якщо поточний елемент більше 0 – зміст NS збільшується на 1;

в) якщо NS=N_PER – немає можливих рекомендацій системи для цього користувача. Зміст N_No збільшується на 1;

г) якщо NS < N_PER – збільшується на 1 зміст N_Yes. Визначається, які елементи є вільними:

– якщо вільним є хоча б один з елементів від 1 до N_Barb, це означає, що на сайті мало користувачів і рекомендації можуть повторюватися. Тоді у елементі масиву PER, генерується запас часу для надання нової рекомендації;

– якщо всі рекомендації хоча б по 1 разу були надані кожному споживачу, а нових споживачів не зафіксовано 0 (відсутні) – знаходиться першій вільний споживач, якому присвоюється значення 1, для нього надається наступна, відмінна від попередньої, рекомендація;

7) для імітацій поведінки системи у наступний момент часу:

а) зменшується на 1 вміст N_C;

б) у комірках PER(1) ÷ PER(N_BARB), що відмінні від нуля, зменшується на 1.

Якщо внаслідок цього вміст поточної комірки дорівнює 0 і наявні комірки черзі, вміст яких дорівнює 1, то обираємо першу з таких комірок і присвоюємо значення «0». Далі генерується час перебування користувача на інтернет-сторінці;

8) по закінченні циклу спостережень тактів (600) на робочий аркуш виводять поточне значення змінної за період, N_Yes, N_No;

Наведений алгоритм можна розширити шляхом введення розрахунків з іншими значеннями вхідних параметрів.

При використанні рівномірного розподілу необхідно ввести зміни у підпрограми-функції програми. При цьому нижня межа часу (a) буде дорівнювати нуля, а верхня межа (b) буде дорівнювати подвоєному значенню середнього часу. Лістинг програми за наведеним алгоритмом надано у Додатку В.

Розрахунки середніх значень відбуваються за допомогою функції СРЗНАЧ() табличного процесору Excel, а стандартних відхилень – СТАНДОТКЛ(), які знаходяться серед статистичних функцій Excel.

На першому етапі роботи було проведено моделювання згідно з програмою, що наведена Додатку В. Моделювання проводилося згідно таких варіантів:

а) кількість рекомендацій 2, кількість користувачів – 7 (2 – на сторінці з рекомендаційною системою);

б) кількість рекомендацій 3, кількість користувачів – 5 (2 – роблять переходи на сторінку рекомендацій, інші присутні там).

Для обох випадків було розраховано середню кількість користувачів, які здійснять перехід на рекомендовану сторінку, середньоквадратичне відхилення цієї величини, а також, середню кількість користувачів, що придбають запропонований продукт, з тим, щоб визначити, чи є результати моделювання сталими. Результати моделювання представлені у табл. 2.2.

Таблиця 2.2 – Вплив кількості випадкових випробувань (NW) на середню кількість переходів користувачів ($N_{пер}$), середньоквадратичне відхилення кількості здійснених переходів ($\sigma_{пер}$), частку користувачів від загальної кількості (P,%) для варіантів а) і б).

NW	Варіант а)			Варіант б)		
	$N_{пер}$	$\sigma_{пер}$	P,%	$N_{пер}$	$\sigma_{пер}$	P,%
100	43	5,0	71	49	5,2	82
500	43	5,3	71	50	5,2	82
1000	43	5,3	71	49	5,1	81
1500	43	5,2	70	49	5,3	82
2000	43	5,1	71	49	5,2	82

З даних табл. 2.2 випливає наступне:

– спостерігається сталість результатів моделювання, що проявляється у незалежності результатів від кількості випробувань;

– при переході від варіанту а) до варіанту б) відбувається збільшення кількості користувачів, що робили вибір за рекомендацією з 43 (у середньому) до 49. Тобто, збільшення складає у середньому 6 переходів за наданою рекомендацією.

Виходячи з результатів моделювання, можна вважати, що реорганізація щодо збільшення кількості рекомендацій на сайті за варіантом б) є економічно вигідною і доцільною.

2.4 Методологія розробки алгоритму формування рекомендацій за маркерами користувача

Як зазначалося в процесі аналізу літературних джерел, в умовах, коли користувач багато працює в мережі Інтернет і переглядає занадто великі обсяги інформації, створюється система з обмеженнями за знаннями і часом [11]. Якщо при цьому приходиться вирішувати велику кількість завдань, то формулюється велика кількість альтернатив, що може призвести до невірного вибору і виникнення критичної ситуації [34]. Тобто, замість пошуку найкращого можливого рішення користувачі повинні перебирати альтернативи тільки до тих пір, поки не виявиться така, яка задовольнить певному прийнятому мінімальному стандарту. Тому на практиці, найчастіше, замість пошуку оптимального рішення користувачі вибирають рішення, яке дозволить зняти проблему.

Розробка інформаційної технології рекомендаційної підтримки прийняття рішень націлена саме на те, щоб знайти оптимальне рішення за обмежений період часу. Поглиблений аналіз складних проблем необхідний для розробки декількох альтернатив, що дійсно суттєво розрізняються, включаючи можливість бездіяльності. Саме для того слід оцінити кожну альтернативу.

Якщо для ілюстрації вирішення подібної задачі у попередньому розділі роботи були використані змішані стратегії двох учасників за двома обмеженнями, то підхід до вирішення зазначеної задачі значно ширший. Коротко подібний підхід можна представити у вигляді простого коду, що модель на основі вмісту і колаборативне фільтрування для надання персоналізованих рекомендацій користувачам на основі їхніх взаємодій та характеристик елементів.

Це можна представити за допомогою наступних прийомів:

```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.neighbors import NearestNeighbors

# Збір та підготовка даних (зазвичай дані імпортуються з файлів, наприклад .csv)
users_data = pd.DataFrame({
    'user_id': [1, 2, 3, 4, 5],
    'user_name': ['John', 'Alice', 'Bob', 'Claire', 'David']
})

items_data = pd.DataFrame({
    'item_id': [1, 2, 3, 4, 5],
    'item_name': ['Item 1', 'Item 2', 'Item 3', 'Item 4', 'Item 5'],
    'description': ['Description 1', 'Description 2', 'Description 3', 'Description 4', 'Description 5']
})

interactions_data = pd.DataFrame({
    'user_id': [1, 1, 2, 2, 3, 3, 4, 5],
    'item_id': [1, 2, 3, 4, 1, 3, 4, 5],
    'rating': [5, 4, 3, 2, 5, 4, 3, 2]
})

# Реалізація моделі на основі вмісту
tfidf = TfidfVectorizer()
item_features = tfidf.fit_transform(items_data["description"]).toarray()
```

```
    # Реалізація колаборативного фільтрування
    user_item_matrix = interactions_data.pivot_table(index="user_id",
columns="item_id", values="rating", fill_value=0)

    knn_model = NearestNeighbors(metric="cosine", algorithm="brute")
    knn_model.fit(user_item_matrix.values)

    # Інтеграція моделей
    def hybrid_recommendation(user_id):
        # Знаходження маркеру користувача в даних про користувачів
        user_index = users_data[users_data["user_id"] == user_id].index[0]
        # Отримання вектору користувача з моделі на основі використаних
раніше маркерів
        user_vector = item_features[user_index]

        # Знаходження найближчих сусідів користувача за маркерами з
моделі колаборативного фільтрування
        _, item_indices =
knn_model.kneighbors([user_item_matrix.loc[user_id].values], n_neighbors=2)

        scores = []
        for item_idx in item_indices.flatten():
            item_vector = item_features[item_idx]
            # Розрахунок схожості між вектором користувача та вектором
елемента
            similarity_score = cosine_similarity([user_vector], [item_vector])[0][0]
            scores.append((items_data["item_id"][item_idx], similarity_score))

        scores.sort(key=lambda x: x[1], reverse=True)
```

```
    recommended_items = [item[0] for item in scores]
    return recommended_items

# Приклад виклику функції рекомендацій для користувача з ID 1
user_id = 1
recommended_items = hybrid_recommendation(user_id)
print(f"Рекомендовані елементи для користувача {user_id}:
{recommended_items}")
```

В цілому, кожна частина коду має своє значення при реалізації алгоритму рекомендації, зокрема:

1) Збір та підготовка даних:

– users_data: DataFrame з даними про користувачів, може включати за згодою [95] їх ідентифікатор та ім'я;

– items_data: DataFrame з даними про елементи, включаючи їх ідентифікатор, назву та опис;

– interactions_data: DataFrame з даними про взаємодії між користувачами та елементами, включаючи ідентифікатори користувачів, елементів та оцінки;

2) Реалізація моделі на основі вмісту:

– tfidf: ініціалізація об'єкту TfidfVectorizer для векторизації текстових описів елементів;

– item_features: побудова матриці ознак на основі текстових описів елементів, використовуючи TfidfVectorizer;

3) Реалізація колаборативного фільтрування:

– user_item_matrix: побудова матриці взаємодій користувачів та елементів, використовуючи взаємодії з interactions_data;

– knn_model: ініціалізація моделі колаборативного фільтрування на основі методу найближчих сусідів (k-NN) з використанням метрики «cosine» та алгоритму «brute». Модель навчається на матриці взаємодій user_item_matrix;

4) Інтеграція моделей:

– hybrid_recommendation: функція для рекомендації елементів користувачу на основі моделей на основі вмісту та колаборативного фільтрування. Приймає ідентифікатор користувача як вхідний параметр;

– отримує індекс користувача в users_data;

– отримує вектор користувача з матриці ознак item_features;

– знаходить найближчих сусідів користувача з використанням моделі колаборативного фільтрування;

– розраховує схожість між вектором користувача та вектором елементів з використанням косинусної схожості;

– сортує елементи за спаданням схожості та повертає рекомендовані елементи;

5) Приклад виклику функції рекомендацій:

– задається ідентифікатор користувача (user_id);

– викликається функція hybrid_recommendation з user_id в якості вхідного параметра;

– Виводяться рекомендовані елементи для заданого користувача.

Підсумовуючи, можна зазначити, що даний алгоритм ефективно поєднує дві моделі рекомендаційних систем: модель на основі вмісту та колаборативне фільтрування. Модель на основі вмісту використовує TfidfVectorizer для векторизації текстових описів елементів та побудови матриці ознак, що дозволяє порівнювати схожість між елементами на основі їхнього вмісту. Колаборативне фільтрування використовує метод найближчих сусідів (k-NN) для знаходження найближчих користувачів до даного користувача.

З наведеного алгоритму можна зробити висновок, що при виявленні нової альтернативи потрібна її оцінка. Тобто, цикл слід повторити, а для зіставлення рішень за кожною альтернативою ще й розширити рекомендаційний алгоритм. Крім того, наведений алгоритм демонструє

необхідність рекомендації альтернатив поведінки в залежності від варіантів зміни умов поведінки користувача.

2.5 Висновок за другим розділом

За підсумками виконання другого розділу роботи можна зробити такі висновки:

1) проведене структурування процесів рекомендаційної системи при різних інтересах користувачів з використанням n-мірних матриць з метою подальшої реалізації алгоритмів за допомогою опенсорсної бібліотеки TensorFlow. Алгоритми повинні передбачати реалізацію ситуації за певними маркерами користувача, тобто, ключових ознаках для користувача за якими він приймає рішення, з врахуванням різноманітних змін, обмежень та персональних дій користувача на визначений момент часу;

2) розроблена лінійна модель рекомендацій за декількома маркерами користувача, при якій створено перехід від вирішення задачі за допомогою теорії ігор до алгоритмізації процесів за допомогою оціночної моделі. Проведений розрахунок довів адекватність моделі, побудованої за двома визначеними маркерами користувача;

3) розроблена модель та реалізовані алгоритми для побудови переходів при формуванні вибору за маркерами користувача для систем, що знаходяться у стаціонарному стані та таких систем, де рекомендація виступає випадковою подією. В розробці застосований метод Рунге-Кутти для системи зі стаціонарним станом, а також застосовано закон розподілу ймовірності при наданні рекомендації користувачу, як випадкової події при Інтернет-серфінгу;

4) представлена методологія розробки алгоритму формування рекомендацій за маркерами користувача, у випадку, коли користувач переглядає занадто великі обсяги інформації і важко визначити пріоритетність

маркерів. У такому випадку виникає велика кількість альтернатив, що у підсумку може призвести до помилки;

5) матеріали, що подані в розділі, знайшли своє часткове відображення у статтях та представлені у вигляді доповіді на конференції [76, 96 – 97].

РОЗДІЛ 3
РОЗРОБКА МОДЕЛІ НАДАННЯ РЕКОМЕНДАЦІЙ ЗА ДОПОМОГОЮ ПРАВИЛ КОМП'ЮТЕРНОЇ ЛОГІКИ

3.1 Представлення переходів користувача з метою відбору параметрів для надання рекомендацій

При реалізації інформаційної технології рекомендаційної підтримки прийняття рішень можна розділити процес на окремі кроки: на першому кроці за маркерами користувача збирається і систематизується інформація, на другому кроці формується рекомендація на основі отриманих вибірок щодо уподобань користувача. Тобто, у цьому випадку можна говорити про дискретний алгоритм [98], що складається з декількох актів, виконання яких не викликає сумніву. А враховуючи те, що такий підхід реалізує принцип створення управляючого пристрою [99], коли за діями користувача рішення може приймати лише два значення – одиничне (на основі отриманих маркерів користувачу можна надати рекомендацію) та нульове (маркери не орієнтують, що даний продукт може зацікавити користувача, тобто, рекомендацію щодо пропозиції товару чи послуги надати неможливо), наведене можна описати комбінаційною схемою за допомогою правил комп'ютерної логіки [72]. На подібне рішення наштовхнула робота [100] щодо практичного застосування дискретних особливостей при розробці інформаційних технологій, а також зазначене дослідження [99]. Орієнтуючись також на підходи до реалізації задач у вебпрограмуванні [72], технологію рекомендаційної підтримки, яка на вході має інформацію з різнорідних джерел стосовно поведінки користувача, а на виході – інформацію з рекомендаціями, можна представити у вигляді моделі В.М. Глушкова [101] – сукупності керуючого та операційного автоматів (рис. 3.1).

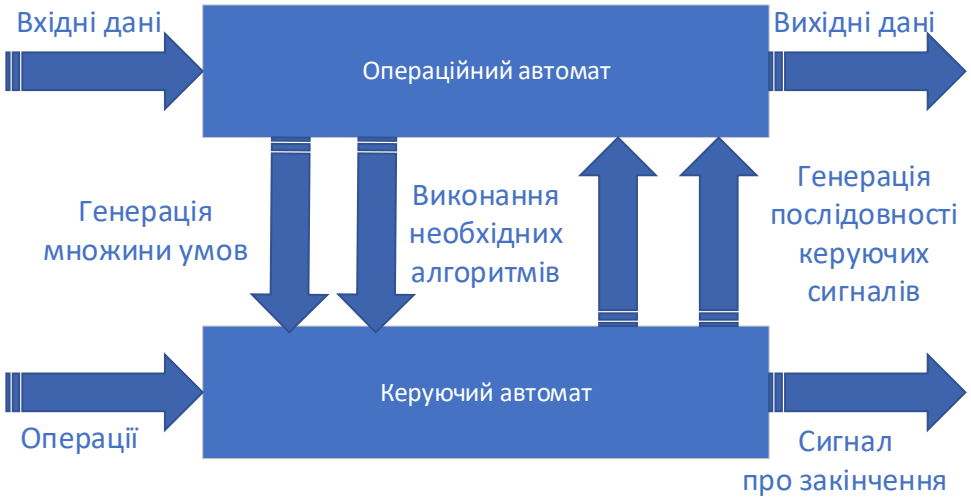


Рисунок 3.1 – Модель цифрового пристрою за В.М. Глушковим

Детальне дослідження [97, 102] застосування цифрових автоматів (ЦА) для створення ІТ рекомендаційної підтримки користувачів в мережі Інтернет дозволили описати поведінку користувачів у мережі розрідженою матрицею, наприклад, як це наведено на рис. 3.2.

	Маркер 1	Маркер 2	Маркер 3	Маркер 4	Маркер 5	Маркер 6	Маркер 7
Користувач 1	**5**	**4**	**5**			**1**	**2**
Користувач 2	**4**		**5**		**4**		
Користувач 3		**3**	**5**	**3**	**4**	**1**	
Користувач 4	**3**		**4**	**2**			**3**

Рисунок 3.2 – Фрагмент таблиці з характеристикою поведінки користувачів

Подібна матриця (рис. 3.2) дозволяє навести множини продуктів, що будуть суміжними з кожним користувачем та множину користувачів, що суміжні з якимось конкретним продуктом. У цьому випадку та з орієнтацією на модель В.М. Глушкова (рис. 3.1), поведінку користувача та її наслідки у

вигляді отриманої рекомендації можна представити за роботою [103] множиною, що складається з шести елементів:

$$S=\{ X,A,Y,\delta,\lambda,a_o\}, \quad (3.1)$$

де:

$X=\{x_1, x_2,... x_n\}$ – множина вхідних сигналів;

$Y=\{y_1, y_2, ... y_m\}$ – множина вихідних сигналів;

$A=\{ a_0,a_1, a_2,... a_N\}$ – множина станів;

a_o – початковий стан ($a_o \in A$);

δ – функція переходів , що задає відображення $(XxA)\rightarrow A$, тобто ставить у відповідність будь-якій парі елементів декартового добутку (ХхА) елемент множини А;

λ – функція виходів, що задає відображення $(XxA)\rightarrow Y$ або відображення $A\rightarrow Y$.

Іншими словами, враховуючи правила [89] функція переходів δ показує, що автомат S, перебуваючи в деякому стані $a_j \in A$, при появі вхідного сигналу $x_j \in X$ переходить у якийсь стан $a_p \in A$.

Це можна записати:

$$a_p=\delta\ (a_i,X_j). \quad (3.2)$$

Функція виходів показує, що автомат S, перебуваючи в деякому стані $a_j \in A$, при появі вхідного сигналу $x_j \in X$ видає вихідний сигнал $y_k \in Y$. Це можна записати:

$$y_k=\lambda(\ a_i,\ X_j). \quad (3.3)$$

У даному випадку поняття стану використовується тому, що є необхідність опису поведінки системи, виходи якої залежать не тільки від стану входів в даний момент часу, але і від деякої передісторії, тобто від сигналів, які надходили на входи системи раніше. Стан як раз і відповідає деякій пам'яті про минуле, дозволяючи усунути час як явну змінну і виразити вихідні сигнали як функцію станів і входів в даний момент часу.

Дана система, як було зазначено, є дискретною, час складає t=0,1,2, але переходи між станами здійснюються миттєво. Кожен момент часу t система знаходиться у стані a(t) з множини A. Початковий момент часу t=0 завжди є відповідним стану a_o. При наявності стану a(t), на вході моделі (рис. 3.1) є інформація $x(t) \in X$, а на виході ця інформація буде сигнал $y(t)=\lambda(a(t), x(t))$. Стан зміниться на $a(t+1)=\delta(a(t), x(t))$. За формування способу виходу інформації відповідають наступні типи автоматів [98]:

– Мілі, що характеризується рівняннями:

$$\begin{aligned} y(t)&=\lambda(a(t), x(t)),\\ a(t+1)&=\delta(a(t), x(t)); \end{aligned} \qquad (3.4)$$

– Мура:

$$\begin{aligned} y(t)&=\lambda(a(t)),\\ a(t+1)&=\delta(a(t), x(t)); \end{aligned} \qquad (3.5)$$

– С-автомат:

$$\begin{aligned} y&=y_1 \cup y_2,\\ y_1(t)&=\lambda_1(a(t),x(t)),\\ Y_2(t)&=\lambda_2(a(t)),\\ a(t+1)&=\delta(a(t), x(t)). \end{aligned} \qquad (3.6)$$

Якщо на вхід абстрактного автомата Мілі або Мура, встановленого в початковий стан a_o, подавати деяку послідовність сигналів x(0), x(1), ... – вхідне слово, то на виході з'явиться y(0), y(1),... – вихідне слово. Для випадку С-автомата на його виходах будуть з'являтися дві послідовності: $y_1(0)$, $y_1(1)$,... и $y_2(0)$, $y_2(1)$,.... В абстрактному С-автоматі вихідний сигнал $y_2(t)= \lambda_2 (a(t))$ видається весь час, поки автомат знаходиться в стані a(t). Вихідний сигнал $y_1(t)=\lambda_1(a(t),x(t))$ видається під час дії вхідного сигналу x(t) при знаходженні С-автомата в стані a(t).

Таким чином функцією ЦА є перетворення вхідної інформацію на вихідну, що відповідає меті поставленої у роботі задачі. Далі задача вирішується шляхом побудови низки таблиць, що є таблицями переходів та виходів. Інакше – будується матриця з'єднань. Наприклад, рис. 3.2 можна за допомогою зазначеного представити, як рис. 3.3.

Стани Вхідні сигнали	Маркер a_1	Маркер a_2		Маркер a_k
Користувач x_1	$\delta (a_1,x_1)$	$\delta (a_2, x_1)$	...	$\delta (a_к, x_1)$
…				
Користувач x_j	$\delta (a_1,x_j)$	$\delta (a_2, x_j)$		$\delta (a_к, x_j)$

Рисунок 3.3 – Приклад формування переходів стосовно поведінки користувачів

У клітинці таблиці переходів, що знаходиться на перетині рядка, зазначеного вхідним сигналом x_i, і стовпця зазначеного станом a_j, ставиться стан a_k, що є результатом переходу автомата зі стану a_j під впливом вхідного сигналу x_i, що визначається виразом $a_k=\delta (a_j, x_j)$. Якщо ситуація описується розрідженою матрицею, як представлено на рис. 3.2, то подібне можна представити як частковий автомат. Тоді в клітинці таблиці його переходів, що

знаходиться, на перетині рядка, зазначеної вхідним сигналом і стовпця зазначеного відповідним станом (за умови, що перехід у цей стан під дією даного вхідного сигналу не визначено) ставиться прочерк, і будь яке вхідне слово, що приводить до зазначеного переходу є забороненим.

Заповнення інших клітин аналогічно випадку повністю визначеного автомата. Вид таблиці переходів не залежить від типу заданого автомата (автомат Мілі, Мура, С-автомат). Таблиці виходів автоматів Мілі, Мура, С-автомата мають відмінності.

Таблиця виходів повністю визначеного автомата Мілі будується наступним чином: ідентифікація стовпців і рядків, а також формат таблиці відповідають таблиці переходів повністю визначеного автомата. У клітинці таблиці виходів, що знаходиться на перетині рядка, зазначеної вхідним сигналом x_j, і стовпця, зазначеного станом $a_к$, ставиться вихідний сигнал y_m, який автомат видає, перебуваючи в стані a_k при наявності вхідного сигналу x_j, що визначається виразом: $y_m = \lambda(a_k, x_j)$.

Запропоноване можна розглянути на прикладі заповнення таблиці виходів деякого абстрактного повністю визначеного автомата Мілі з вхідним алфавітом $X=\{x_1, x_2\}$, алфавітом станів $A=\{a_1, a_2, a_3\}$ та вихідним алфавітом $Y=\{y_1, y_2, y_3\}$ (табл. 3.1) для ситуації щодо поведінки користувача у системі зі стаціонарним станом (див. підр. 2.3.1).

Таблиця 3.1 – Вхідні параметри для ситуації щодо поведінки користувача у системі зі стаціонарним станом

a / x	a_1	a_2	a_3
x_1	y_2	y_3	y_1
x_2	y_3	y_1	y_2

Якщо розглянути для цієї ситуації використання автомату Мура у якості керуючого автомату (рис. 3.1), то таблиця виходів повністю визначеного

автомата Мура будується простіше: кожному стану автомата ставиться у відповідність свій вихідний сигнал. Приклад таблиці виходів автомата Мура з алфавітом станів A={a_1, a_2, a_3} та вихідним алфавітом Y={y_1, y_2, y_3} - представлений в табл. 3.2.

Таблиця 3.2 – Таблиця виходів автомату Мура

A	a_1	a_2	a_3
y	y_1	y_2	y_2

Загальний вигляд таблиці переходів для вказаної ситуації в системі зі стаціонарним станом наведено у табл. 3.3, таблиця переходів автомата Мілі – табл. 3.4, таблиця переходів автомата Мура – табл. 3.5.

Таблиця 3.3 – Загальний вигляд таблиці переходів для вказаної ситуації в системі зі стаціонарним станом

Стани / Вхідні сигнали	a_1	a_2	…	a_k
x_1	$\delta(a_1,x_1)$	$\delta(a_2,x_1)$	…	$\delta(a_k,x_1)$
	$\lambda(a_1,x_1)$	$\lambda(a_2,x_1)$	…	$\lambda(a_k,x_1)$
…	…	…	…	…
x_j	$\delta(a_1,x_j)$	$\delta_{(A_2,X_j)}$	…	$\delta(a_k, x_j)$
	$\lambda(a_1,x_j)$	$\lambda(a_2, x_j)$	…	$\lambda(a_k, x_j)$

Таблиця 3.4 – Таблиця переходів автомата Мілі

A / X	a_1	a_2	a_3
x_1	a_2/y_2	a_3/y_3	a_1/y_3
x_2	a_1/y_3	a_1/y_1	a_2/y_2

Таблиця 3.5 – Таблиця переходів автомата Мура

Y	y_1	y_2	y_3
A / X	a_1	a_2	a_3
x_1	a_2	a_3	a_1
x_2	a_1	a_1	a_2

Крім розглянутих вище таблиць переходів і виходів довільний абстрактний автомат може бути заданий матрицею з'єднань.

Матриця з'єднань є квадратною і містить стільки стовпців (рядків), скільки різних станів містить алфавіт станів даного автомата. Кожен стовпець (рядок) матриці з'єднань позначається літерою стану автомата. У клітинці, що знаходиться на перетині стовпця, поміченого a_j і рядки з позначкою літерою a_s автомата, ставиться вхідний сигнал (або диз'юнкція вхідних сигналів), під впливом якого здійснюється даний перехід.

Для абстрактного автомата Мілі в комірці поруч із станом проставляється також вихідний сигнал, який автомат видає в результаті даного переходу (табл. 3.6) Для автомата Мура вихідний сигнал проставляється в рядку поруч із станом (ці стани відповідають вихідним станам автомата).

Таблиця 3.6 – Зазначення вхідного сигналу для автомата Мілі

A / X		a_1	a_n
x_1	a_1	$x_j(y_k)$	
x_2	a_n		$x_j(y_m)$

У випадку, коли розглядається ситуація, що рекомендація для користувача виникає як випадкова подія (див. підр. 2.3.2), то такий автомат може бути заданий четвіркою об'єктів: S = {X, A, Y, F}, де F задає для кожного стану a_j автомата відображення (XxA) → (AxY). Іншими словами, завдання для кожного стану автомата a_j вказується відображення F_{ai}, що представляє собою безліч всіх трійок a_p, x_m, y_k, і таких, що під впливом вхідного сигналу x_m автомат переходить зі стану a_j в стан a_p, видаючи при цьому вихідний сигнал y_k. Останнє рівнозначно опису функцій δ і λ у відповідності з виразом: $a_p = \delta(a_i, x_m)$, $y_к = \lambda(a_i, x_m)$.

Відображення F_{ai} записується таким чином:

$$F_{ai}\{a_p(X_m/y_k), a_i(X_f/y_z)\ldots\}.$$

Наприклад, для абстрактного автомата Мілі (табл. 3.4) аналітичне завдання матиме наступний вигляд:

$S=\{ X,A,Y,F\}$, $X=\{x_1,x_2\}$, $A=\{a_1, a_2, a_3\}$, $Y=\{y_1, y_2, y_3\}$,

$F_{a1}=\{a_2(x_1/y_2), a_1(x_2/y_3)\}$,

$F_{a2}=\{a_3(x_1/y_3), a_1(x_2/y_1)\}$,

$F_{a3}=\{a_1(x_1/y_3), a_2(x_2/y_2)\}$.

Слід зазначити, що функція F_{ai} завжди записується для вихідного стану.

Розглянемо приклад, коли керуючий автомат представлений автоматом Мілі та заданий таблицями переходів (табл. 3.7) та виходів (табл. 3.8).

Таблиця 3.7 – Таблиця переходів

A / X	a_1	a_2	a_3	a_4
x_1	a_1	a_3	a_3	a_3

x_2	a_2	a_4	a_1	a_1
x_3	a_1	a_2	a_4	a_2

Таблиця 3.8 – Таблиця виходів

X \ A	a_1	a_2	a_3	a_4
x_1	y_1	y_3	y_2	y_2
x_2	y_1	y_3	y_1	y_1
x_3	y_2	y_3	y_3	y_3

Таблиця переходів для заданого автомата може бути представлена наступним чином (табл. 3.9).

Таблиця 3.9 – Таблиця переходів

X \ A	a_1	a_2	a_3	a_4
x_1	a_1 / y_1	a_3 / y_3	a_3 / y_2	a_3 / y_2
x_2	a_2 / y_1	a_4 / y_2	a_1 / y_1	a_1 / y_1
x_3	a_1 / y_2	a_2 / y_2	a_4 / y_2	a_2 / y_2

Матриця з'єднань представлятиме собою квадратну матрицю розміром A x A і вона також об'єднує таблиці переходів і виходів абстрактного автомата та доволі легко виконується при реалізації засобами Tensor Flow. Для заданого автомата вона набуває вигляду, представленого в табл. 3.10.

Таблиця 3.10 – Матриця з'єднань

a(t) / a(t+1)	a_1	a_2	a_3	a_4
a_1	$x_1(y_1) \vee x_3(y_2)$	-	$x_2(y_1)$	$x_2(y_1)$
a_2	$x_2(y_1)$	$x_3(y_3)$	-	$x_3(y_3)$
a_3	-	$x_1(y_3)$	$x_1(y_2)$	$x_1(y_2)$
a_4	-	$x_2(y_3)$	$x_3(y_3)$	-

Аналітично автомат задається четвіркою об'єктів:

$$S = \{X, A, Y, F\},$$

де F задає для кожного стану a_i автомата відображення (X*A) → (A*Y). У зазначеному випадку аналітичне задавання абстрактного автомата Мілі буде виглядати, як наведено у табл. 3.9:

$$X = \{x_1, x_2, x_3\} ; A = \{a_1, a_2, a_3, a_4\}; Y = \{y_1, y_2, y_3\}$$
$$F_{a1} = \{a_1(x_1/y_1), a_2(x_2/y_1), a_1(x_3/y_2)\}$$
$$F_{a2} = \{a_3(x_1/y_3), a_4(x_2/y_3), a_2(x_3/y_3)\}$$
$$F_{a3} = \{a_3(x_1/y_2), a_1(x_2/y_1), a_4(x_3/y_3)\}$$
$$F_{a4} = \{a_3(x_1/y_2), a_1(x_2/y_1), a_2(x_3/y_3)\}.$$

Враховуючи наведене, матрицю з’єднань можна представити у вигляді табл. 3.11.

Таблиця 3.11 – Матриця з'єднань за четвіркою об’єктів

a(t) / a(t+1)	a_1	a_2	a_3	a_4
a_1	x_2	x_1	x_2	-
a_2	x_1	-	x_1	-
a_3	-	-	-	x_2
a_4	-	x_2	-	x_1

Але наведене (табл. 3.9 та табл. 3.11) більше характерно для системи зі стаціонарним станом при незмінних уподобаннях користувача. Коли відбуваються зміни уподобань, відповідно повинні відбуватися зміни в рекомендаціях з реалізацією відповідних переходів.

3.2 Моделі переходів в рекомендаціях в залежності від зміни уподобань

3.2.1 Формування графів переходів за різними вхідними параметрами, що характеризують дії користувача

Розглянемо ситуацію, коли за моделлю (рис. 3.1) коли операційний (S_A) і керуючий (S_B) автомат є еквівалентними, тобто при встановленні їх у початковий стан їх реакції на вхідне слово співпадають. При описанні алгоритмів взаємної трансформації автоматів Мілі і Мура, будемо не приймати до уваги вхідний сигнал автомату Мура, пов'язаний з початковим станом (λ (a_1)).

Проаналізуємо перетворення автомата Мура в автомат Мілі.

Нехай дано автомат Мура: $S_A=\{ X_A, A_A, У_A, \delta_A, \lambda_A, a_{oA}\}$,

де:

$X_A=\{x_1, x_2, \dots x_n\}$; $Y=\{y_1, y_2, \dots y_m\}$; $A=\{ a_0, a_1, a_2, \dots a_N\}$;

$a_{0A} = a_0$ – початковий стан ($a_{0A} \in A$);

δ_A – функція переходів автомата, що задає відображення ($X_A x A_A \rightarrow A_A$);

λ_A – функція виходів автомата, що задає відображення $A_A \rightarrow У_A$.

Побудуємо автомат Мілі: $S_B=\{ X_B, A_B, Y_B, \delta_B, \lambda_B, a_{0B}\}$, у якого $A_B=A_A$; $X_B=X_A$; $Y_B=У_A$; $\delta_B= \delta_A$; $a_{0B}=a_{0A}$.. Функцію виходів λ_B визначимо наступним чином: якщо в автоматі Мура $\delta_A(a_m, x_1)= a_s$ і $\lambda_A (a_s)=y_g$, то в автоматі Мілі $\lambda_B (a_m, x_1)=y_g$.

Перехід від автомата Мура до автомата Мілі при графічному способі завдання ілюструється рис. 3.4. вихідний сигнал y_g записаний поруч з вершиною (a_s), переноситься на всі дуги, що входять в цю вершину.

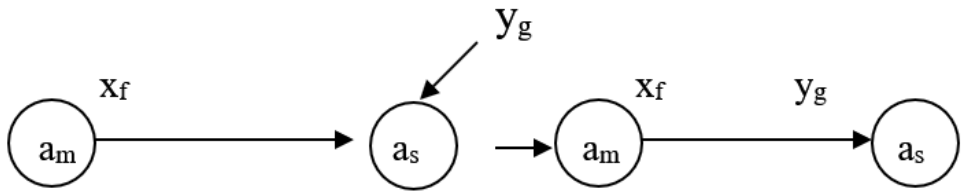


Рисунок 3.4 – Ілюстрація переходу від моделі Мура до моделі Мілі

При табличному способі завдання автомата таблиця переходів автомата Мілі збігається з таблицею переходів вихідного автомата Мура, а таблиця виходів виходить з таблиці переходів заміною символу a_s, що стоїть на перетині рядка x_f і стовпця a_m, символом вихідного сигналу y_g відзначає стовпець a_s в таблиці переходів автомата S_A.

З самого способу побудови автомата Мілі S_B очевидно, що він еквівалентний автомату Мура S_A. За індукції неважко показати, що будь-яке вхідне слово кінцевої довжини, подане на входи автоматів S_A і S_B, встановлених в стан a_m, викличе появу однакових вихідних слів і, отже,

автомати S_A і S_B будуть еквівалентними. Тобто, за будь-яких вхідних параметрах, що є маркерами користувача, при виконанні зазначених умов, на виході буде отримана рекомендація, що чітко відповідатиме поставленим умовам.

Перш ніж розглянути трансформацію автомата Мілі в автомат Мура, накладемо на автомат Мілі наступне обмеження: у автомата не повинно бути перехідних станів. Під перехідним будемо розуміти стан, в який при заданні автомата у вигляді графа не входить жодна дуга, але який має принаймні одну дугу, що виходить.

Отже, нехай задано автомат Мілі:

$$S_A=\{ X_A, A_A, Y_A, \delta_A, \lambda_A, a_{0A}\},$$

де:

$X_A=\{x_1, x_2, \dots x_n\}$; $Y=\{y_1, y_2, \dots y_m\}$; $A=\{ a_0, a_1, a_2, \dots a_N\}$;

$a_{0A} = a_0$ – початковий стан ($a_{0A} \in A$);

δ_A – функція переходів автомата, що задає відображення ($X_A x A_A \rightarrow A_A$);

λ_A – функція виходів автомата, що задає відображення $A_A \rightarrow У_A$.

Побудуємо автомат Мура: $S_B=\{X_B, A_B, Y_B, \delta_B, \lambda_B, a_{0B}\}$, у якого $X_B=X_A$; $Y_B=Y_A$.

Для визначення A_B кожному стану a_s A_A поставимо у відповідність множину A_s всіляких пар виду (a_s, y_g).

Функцію виходів δ_B визначимо наступним чином. Кожному стану автомата Мура S_B, який представляє собою пару вигляду (a_s, y_g), поставимо у відповідність вихідний сигнал y_g. Якщо в автоматі Мілі S_A був перехід $\delta_A(a_m, x_f) = a_s$ і при цьому видавався вихідний сигнал $\lambda_A (a_m, x_f) = y_g$, то в S_B буде перехід з множини станів A_m, породжуваних a_m, в стан (a_s, y_g) під дією вхідного сигналу x_f.

В якості початкового стану a_{0B} можна взяти будь-який з станів множини A_0, яке породжується початковим станом a_0 автомата S_A. При цьому вихідний сигнал в момент часу t = 0 не повинен враховуватися.

Розглянемо приклад. Нехай заданий автомат Мілі (табл. 2.12).

Таблиця 3.12 – Параметри автомата Мілі

X / A	x_1	x_2
a_0	a_2/y_1	a_0/y_1
a_1	a_0/y_1	a_2/y_2
a_3	a_0/y_2	a_1/y_1

Поставимо у відповідність кожній парі a_i/x_k стан b_{ik} (i-номер стану, k-номер вхідного сигналу), з урахуванням b_0.

Складемо таблицю переходів автомата Мура, керуючись наступними правилами:

1) Випишемо з табл. 3.13 стани автомата Мілі.

Таблиця 3.13 – Стани автомата Мілі

X / A	x_1	x_2
a_0 b_0	a_2/y_1 b_{01}	a_0/y_1 b_{02}
a_1	a_0/y_1 b_{11}	a_2/y_2 b_{12}
a_2	a_0/y_2 b_{21}	a_1/y_1 b_{22}

Після цього встановлюємо відповідні кожному з них множини станів автомата Мура (b_{ik}):

$$a_0 = \{b_0, b_{02}, b_{11}, b_{21}\}; \qquad a_1 = \{b_{22}\}; \qquad a_2 = \{b_{01}, b_{12}\};$$

2) Якщо стан автомата Мура b_{ik} входить до множини, яка відповідає стану a_p автомата Мілі, то в рядок таблиці переходів автомата Мура для стану b_{ik} слід записати рядок з таблиці переходів автомата Мілі, який відповідає стану a_p (з табл. 3.12).

3) Функцію виходів автомата Мура визначимо наступним чином: $\lambda_B (b_{ik}) = \lambda_A (a_i, x_k)$. Для початкового стану b_0 значення вихідного сигналу можна вибрати довільно, але породжуваний початковим станом a_0 (з урахуванням поняття еквівалентності станів). Результуюча таблиця переходів і виходів автомата Мура еквівалентного автомату Мілі, заданому табл. 3.12 представлена в табл. 3.14.

Таблиця 3.14 – Результуюча таблиця переходів

	x_1	x_2	Y
b_0	b_{01}	b_{02}	y_1
b_{01}	b_{21}	b_{22}	y_1
b_{02}	b_{01}	b_{02}	y_1
b_{11}	b_{01}	b_{02}	y_1
b_{12}	b_{21}	b_{22}	y_2
b_{21}	b_{01}	b_{02}	y_2
b_{22}	b_{11}	b_{12}	y_1

4) Знайдемо в таблиці 2.3. еквівалентні стани і видалимо їх (замінимо на представника класу еквівалентності). Якщо вихідний сигнал біля b_0 доозначити y_1, то виявиться, що в даній таблиці переходів знаходиться 3 еквівалентних стани (b_0, b_{11}, b_{02}). Замінивши клас еквівалентності одним представником (b_0), отримаємо остаточну таблицю переходів (табл. 3.15

Таблиця 3.15 – Підсумкова таблиця переходів

	x_1	x_2	У
b_0	b_{01}	b_0	y_1
b_{01}	b_{21}	b_{22}	y_1
b_{12}	b_{21}	b_{22}	y_2
b_{21}	b_{01}	b_0	y_2
b_{22}	b_0	b_{12}	y_1

Викладені методи взаємної трансформації автоматів Мілі і Мура показують, що при переході від автомата Мура до автомата Мілі число станів автомата не змінюється, тоді як при зворотному переході число станів в автоматі Мура, як правило, зростає. Це дозволяє розширити варіанти надання рекомендацій, враховуючи ті фактори, які при реалізації рекомендаційних механізмів за традиційними моделями, опускалися, як несуттєві (наприклад, користувач переглядав декілька разів рекомендований продукт, але не придбав і т. інш.).

3.2.2 Врахування еквівалентних станів уподобань при формуванні матриць переходів рекомендацій за зміни окремих ознак

Можливість враховувати еквівалентні стани уподобань при формуванні матриць переходів рекомендацій за зміни окремих ознак можна реалізувати шляхом розбиття станів вихідного абстрактного автомата на класи еквівалентних станів, які попарно не перетинаються, і заміні кожного класу еквівалентності одним станом – представником даного класу.

Два стани автомата a_m і a_s будуть еквівалентними ($a_m \equiv a_s$), якщо $\lambda(a_m,X) = \lambda(a_s,X)$ для всіх можливих вхідних слів довжини X. Якщо a_m і a_s не еквівалентні, вони різні. Більш слабкою еквівалентністю є k-еквівалентність. Стани a_m і a_s k-еквівалентні, якщо $\lambda (a_m, X_k) = \lambda (a_s, X_k)$ для всіх можливих вхідних слів довжини k.

При мінімізації числа внутрішніх станів автомата Мілі $S =\{X, Y, A, \lambda, \delta, a_0\}$ використовується алгоритм Ауфенкампа-Хона [104]:

1) знаходять послідовні розбиття $\pi_1, \pi_2, ..., \pi_k, \pi_{k+1}$, множини A на класи одно-, дво-, ..., k-, (k+1)-еквівалентних станів до тих пір, поки на якомусь (k +1) кроці не виявиться, що $\pi_k = \pi_{k+1}$. У цьому випадку k-еквівалентні стани є еквівалентними. Число кроків k, при якому $\pi_k = \pi_{к+1}$, не перевищує N-1, де N-число внутрішніх станів автомата;

2) у кожному класі еквівалентності π вибирають по одному елементу (представнику класу), які утворюють множини A' станів мінімального автомата S';

3) функцію переходів δ' і виходів λ' автомата S' визначають на множині A'xX. Для цього в таблиці переходів і виходів викреслюють стовпці станів, які не увійшли до множини A', а в решті стовпців таблиці переходів всі стани замінюються на еквівалентні з множини A' на представників;

4) в якості a'_0 вибирається один зі станів, еквівалентних стану a_0. Зокрема, зручно прийняти сам стан a_0.

При мінімізації автомата Мура вводиться поняття 0-еквівалентності станів і розбиття множини станів на 0-класи: 0-еквівалентними називаються будь-які, однаково зазначені вихідними сигналами, стани автомата Мура. При реалізації рекомендаційних систем за цим підходом найбільш точні надані рекомендації будуть підтверджуватися однаково зазначеними вхідними сигналами.

3.3 Створення механізму рекомендацій за ознакою структурної повноти

Наступним етапом абстрактного синтезу автоматів, що закінчується мінімізацією числа станів, слідує етап структурного синтезу, метою якого є побудова схеми, що реалізує автомат з логічних елементів заданого типу. У структурному автоматі враховується структура вхідних і вихідних сигналів автомата, а також його внутрішня побудова на рівні структурних схем. Основним завданням структурної теорії автоматів є знаходження загальних прийомів побудови структурних схем автоматів на основі композиції елементарних автоматів, що належать до заздалегідь заданого кінцевого числа типів.

У структурній теорії як вхідні так і вихідні канали вважаються складаються з елементарних вхідних (вихідних) каналів. По всіх елементарних вхідних (вихідних) каналах можуть передаватися тільки елементарні сигнали.

Набір можливих значень сигналів, що подаються на один зовнішній вхідний (вихідний) вузол, називається структурним вхідним (вихідним) алфавітом автомата. Алфавіт повинен бути обмеженим.

Вхідний і вихідний сигнали задаються обмеженими впорядкованими наборами елементарних сигналів, які називаються векторами, а складові їх елементарні сигнали – компоненти векторів. Число компонент вектору – це розмірність алфавіту.

Наприклад, $X=\{x_1, x_2, x_3, x_4, x_5\}$ – вхідний алфавіт абстрактного автомата.

Структурний вхідний алфавіт, розмірність якого дорівнює трьом:

$x_1 = 000$, $x_2 = 001$, $x_3 = 010$, $x_4 = 011$, $x_5 = 100$.

Векторне представлення вхідних і вихідних сигналів відповідно є структурним вхідним вихідним сигналом.

На етапі структурного синтезу попередньо вибираються елементарні автомати, з яких потім шляхом їх композиції будується структурна схема отриманого на етапі абстрактного синтезу автомата Мілі, Мура або С-автомата. Якщо рішення задачі структурного синтезу існує, то задана система автоматів структурно повна. При цьому застосовується канонічний метод структурного синтезу, при якому використовуються елементарні автомати деякого спеціального виду: автомати з пам'яттю, що мають більше одного стану, і автомати без пам'яті – з одним станом. Автомати першого класу носять назву елементів пам'яті, а автомати другого класу – комбінаційних або логічних елементів.

Теоретичним обґрунтуванням канонічного методу структурного синтезу автоматів є теорема про структурну повноту (теорема Глушкова [101]): усяка система елементарних автоматів, яка містить автомат Мура з нетривіальною пам'яттю, який має повну систему переходів і повну систему виходів, і яку-небудь функціонально повну систему логічних елементів, є структурно повною.

Існує загальний конструктивний прийом (канонічний метод структурного синтезу), що дозволяє в розглянутому випадку звести задачу структурного синтезу довільних автоматів до задачі синтезу комбінаційних схем.

Результатом канонічного методу структурного синтезу є система логічних рівнянь, що виражає залежність вихідних сигналів автомата (функції виходів автомата) і сигналів, що подаються на входи елементів пам’яті, від сигналів, що приходять на вхід всього автомата в цілому, і сигналів, що знімаються з виходу елементів пам'яті (функції збудження елементів пам'яті автомата).

Для правильної роботи схем, очевидно, не можна дозволяти, щоб сигнали на вході елементів пам'яті безпосередньо брали участь в утворенні

вихідних сигналів, які по ланцюгах зворотного зв'язку подавалися б у той же самий момент часу на ці входи. У зв'язку з цим елементами пам'яті повинні бути не автомати Мілі, а автомати Мура.

Таким чином, структурно повна система елементарних автоматів повинна містити хоча б один автомат Мура. У той же час для синтезу будь-яких автоматів з мінімальним числом елементів пам'яті необхідно в якості таких елементів вибирати автомати Мура, які мають повну систему переходів і повну систему виходів – так звані повні автомати.

Повнота системи переходів означає, що для будь-якої пари станів (a_m, a_s) автомата знайдеться вхідний сигнал, який переводить перший елемент цієї пари a_m в другий – a_s тобто в такому автоматі в кожному стовпці таблиці переходів повинні зустрічатися всі стани автомата. Повнота системи виходів автомата Мура полягає в тому, що кожному стану автомата поставлений у відповідність свій особливий вихідний сигнал, відмінний від вихідних сигналів інших станів. Очевидно, що в такому автоматі число вихідних сигналів дорівнює числу станів автомата. Разом з попереднім твердженням це призводить до того, що в автоматі Мура з повною системою виходів можна ототожнити стани автомата з його вихідними сигналами. У зв'язку з цим в автоматах пам'яті ми будемо використовувати одні й ті ж позначення і для станів, і для вихідних сигналів, тобто зазначена таблиця переходів в автоматах Мура з повною системою виходів перетворюється просто в таблицю переходів.

Наявність функціонально повної системи логічних елементів дозволяє реалізувати булеву функцію будь-якого ступеня складності.

Після вибору елементів пам'яті і кодування станів синтез структурного автомата зводиться до синтезу комбінаційної схеми, що реалізує канонічні рівняння.

У канонічному методі структурного синтезу можна виділити кілька основних етапів за наступним алгоритмом:

1) кодування алфавітів автомата;

2) вибір елементів пам'яті;

3) вибір функціонально повної системи логічних елементів;

4) запис і мінімізація канонічних рівнянь;

5) побудова функціональної схеми автомата;

Вихідними даними для початку роботи даного методу є абстрактний цифровий автомат з пам'яттю, заданий таблицею переходів і виходів.

Розглянемо приклад.

Абстрактний автомат Мура задано наступними алфавітами:

$X=\{x_1, x_2, x_3\}$;

$Y=\{y_1, y_2, y_3\}$;

$A=\{a_1, a_2, a_3, a_4, a_5\}$.

Розробимо структурну схему автомата з системою рішень, що описують роботу системи з використанням даного автомату. Для цього визначимо число компонент двійкових векторів, якими можливо закодувати елементи вхідного, вихідного алфавітів та алфавіту станів.

Число компонент двійкового вектору вхідного алфавіту визначається наступним чином:

$$K_{вх} \geq \text{int}(\log_2 |X|);$$

де:

int(X) – округлення до найближчого цілого числа в бік збільшення;

|X| – потужність вхідного алфавіту.

У випадку, що розглядається, це $|X|=3$, тобто $K_{вх} \geq \text{int}(\log_2 |3|)=2$.

Приймаємо $K_{вх} =2$.

Позначимо елементи двійкового вектору вхідного алфавіту структурного автомата, як $Z=\{z_1, z_2\}$.

Аналогічно визначаємо число компонент двійкового вектору вихідного алфавіту структурного автомата:

$К_{вих} \geq int(\log_2 |Y|)$; де $|Y|=3$;

$К_{вих} \geq int(\log_2 |3|)=2$; $К_{вих}=2$.

Позначимо елементи двійкового вектору вихідного алфавіту структурного автомата, як $W=\{w_1, w_2\}$.

Аналогічно визначаємо число компонент двійкового вектору алфавіту станів структурного автомата:

$К_{стан} \geq int(\log_2 |A|)$;

де $|A|=6$;

$К_{стан} \geq int(\log_2 |6|)=3$;

$К_{стан}=3$.

Ця величина визначає кількість елементів пам’яті в структурному автоматі. Їх повинно бути 3.

Позначимо елементи двійкового вектору алфавіту станів структурного автомата, як $A`=\{\alpha_1, \alpha_2, \alpha_3\}$.

Далі, виходячи із узагальненої структурної схеми автомата Мура, будуємо структурну схему автомата, заданого умовами даної задачі (рис. 3.5).

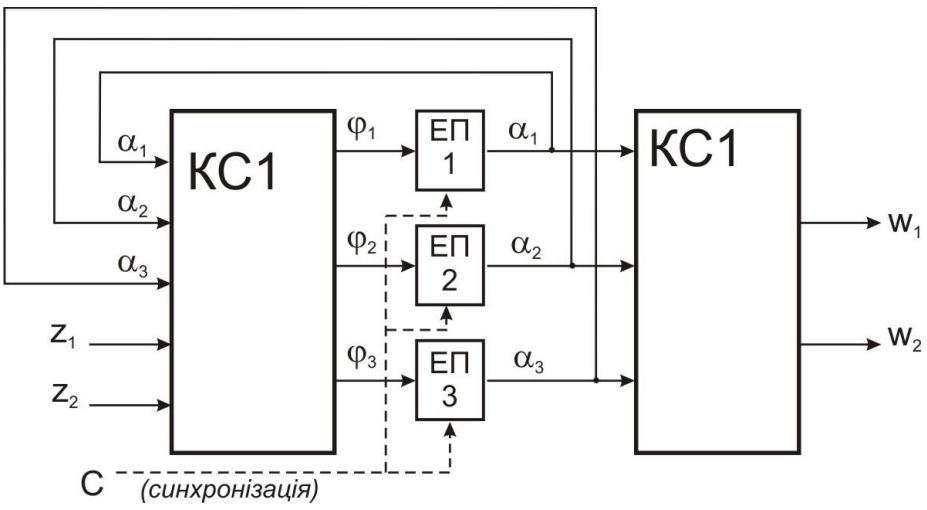


Рисунок 3.5 – Структурна схема автомата Мура

Позначення рис. 3.5 означають:

– КС 1 – комбінаційна схема формування функцій збудження входів;

– КС 2 – комбінаційна схема, яка формує функції виходів;

– ЕП 1, ЕП 2, ЕП 3 – елементи пам’яті автомата, які утримують інформацію про сформовані рекомендації.

Відповідно до наведеної схеми (рис. 3.5) можна представити систему рівнянь:

$$\varphi_1 = \varphi_1(z_1, z_2, \alpha_1, \alpha_2, \alpha_3)$$
$$\varphi_2 = \varphi_2(z_1, z_2, \alpha_1, \alpha_2, \alpha_3)$$
$$\varphi_3 = \varphi_2(z_1, z_2, \alpha_1, \alpha_2, \alpha_3)$$
$$w_1 = w_1(\alpha_1, \alpha_2, \alpha_3)$$
$$w_2 = w_2(\alpha_1, \alpha_2, \alpha_3).$$

Якщо відомі таблиці кодування всіх алфавітів структурного автомата, а також властивості елементів пам'яті, то синтез структурного автомата зводиться до синтезу комбінаційних схем КС1 і КС2, структура яких і визначається системою канонічних рівнянь.

Подібне можна реалізувати і для автомату Мілі, однак у цьому випадку слід зазначити, що в реальних схемах функції елементів пам’яті виконують прості елементи пам’яті, які отримали назву тригери. Елементи памяті, виконані у вигляді тригерів, можуть мати один, два та більше інформаційних входів і, як правило, два виходи (прямий та інверсний). Тому число виходів із КС1 може бути в 2 рази (та навіть більше) більшим. Це дає можливість зміни рекомендацій в залежності від поведінки користувача.

3.4 Механізм формування тригерів для створення рекомендацій за зміни маркерів користувача

Під тригером у дані роботі розуміється стійкий стан рекомендації до зміни ключових ознак на вході системи – маркерів користувача. Тригер, як елемент пам’яті, може бути заданий одним з декількох способів: скороченою таблицею переходів, повною таблицею переходів, характеристичним рівнянням, матрицею переходів. Розглянемо останній спосіб, як такий, що використаний для реалізації задач даної роботи.

Для кожного з 4-х можливих переходів елементарного автомата (00, 01, 10, 11) завжди знайдеться значення вхідного сигналу, що дорівнює 0 або 1, яке викликає даний перехід. Якщо елементарний автомат має 2 або більше входів, то на деякі переходи значення вхідних сигналів, що діють на одному або іншому вході виявляються несуттєвими.

Опишемо закон функціонування елементарного автомата, що має m входів, матрицею переходів:

$$\begin{array}{c|cccccc} & x_1 & x_2 & \dots & x_k & \dots & x_m \\ 00 & b_{00}^{1} & b_{00}^{2} & \dots & b_{00}^{k} & \dots & b_{00}^{m} \\ 01 & b_{01}^{1} & b_{01}^{2} & \dots & b_{01}^{k} & \dots & b_{01}^{m} \\ 10 & b_{10}^{1} & b_{10}^{2} & \dots & b_{10}^{k} & \dots & b_{10}^{m} \\ 11 & b_{11}^{1} & b_{11}^{2} & \dots & b_{11}^{k} & \dots & b_{11}^{m} \end{array}$$

Кількість рядків матриці завжди дорівнює 4 (за кількістю можливих переходів), кількість стовпців дорівнює числу вхідних сигналів. Елемент матриці b_{is}^{k} являє собою значення вхідного сигналу x_k, під дією якого

елементарний автомат перейде зі стану i в стан s. При цьому b_{is}^{k} завжди дорівнює 0 або 1, або невизначений, якщо він не впливає на даний перехід.

Матриця переходів елементарного автомата складається за таблицею переходів.

Розглянемо приклад побудови матриці переходів тригера.

Нехай тригер, в загальному випадку, заданий скороченою таблицею переходів (табл. 3.16 – а). Повна таблиця переходів тригера, побудована за скороченою, представлена в табл.3.16 – б). Із повної таблиці переходів комбінації вхідних сигналів, які відповідають всім можливим переходам представлені у табл. 3.16 – в).

Таблиця 3.16 – Матриця переходів тригера

а)

t		(t+1)
X	У	Q
0	0	0
0	1	Q(t)
1	0	$\overline{Q(t)}$
1	1	1

б)

t			t+1
X	У	Q	Q
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

в)

Q(t)	Q(t+1)	X	У
0	0	0	0
		0	1
0	1	1	0
		1	1
1	0	0	0
		1	0
1	1	0	1
		1	1

Таким чином:

1) для переходу "0-0" X = 0, У може бути 0 або 1;

2) для переходу "0-1" X = 1, У може бути 0 або 1;

3) для переходу "1-0" X може бути 0 або 1, а У = 0;

4) для переходу "1-1" X може бути 0 або 1, а У = 1.

Тоді матриця переходів тригера запишеться у вигляді:

0 - 0	0	b1
0 - 1	1	b2
1 - 0	b3	0
1 - 1	b4	1

де b1, b2, b3, b4 – довільні сигнали (0 або 1)

Як правило, значення двох різних коефіцієнтів b_i і b_s з одного рядка матриці є залежними один від одного і знаходження цієї залежності з ростом числа інформаційних входів ускладнюється. Встановлення точної взаємозалежності між вхідними змінними тригера є обов'язковою умовою, що забезпечує можливість максимального спрощення схем з пам'яттю. В основі методики лежить таблиця, в якій представлені можливі поєднання вхідних змінних та відповідні їм комбінації (табл. 3.17).

Таблиця 3.17 – Таблиця довизначення вхідних сигналів тригерів

		Номер варіанта														
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Вхідні сигнали		X1	0	0	1	1	00	01	01	01	01	11	001	001	011	011	0011
		X2	0	1	0	1	01	00	01	10	11	01	010	011	001	101	0101
Описання взаємозалежності	I	X1	0	0	1	1	-	b1	b1	b1	b1	-	b1	b1	b1	b1	b1
		X2	0	1	0	1	-	0	b1	$\overline{b1}$	1	-	$\overline{b1}\cdot b2$	b1∨b2	b1∨b2	$\overline{b1}\vee b2$	b2
	II	X1	0	0	1	1	0	-	b1	$\overline{b1}$	-	1	$\overline{b1}\cdot b2$	b1∨b2	b1∨b2	$\overline{b1}\vee b2$	b2
		X2	0	1	0	1	b1	-	b1	b1	-	b1	b1	b1	b1	b1	b1

З урахуванням довизначення вхідних змінних матриці переходів деяких стандартних тригерів мають вигляд (табл. 3.18)

Таблиця 3.18 – Матриці переходів стандартних тригерів

Триггер-D			Триггер-T			Триггер-RS				Триггер-JK			
Q(t)	Q(t+1)	D	Q(t)	Q(t+1)	T	Q(t)	Q(t+1)	R	S	Q(t)	Q(t+1)	K	J
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	0	1	1	0	1	0	1	0	1	0	1
1	0	0	1	0	1	1	0	1	0	1	0	1	0
1	1	1	1	1	0	1	1	0	0	1	1	0	0

При побудові таблиць переходів синхронного тригера (додатково до входів X і Y вводиться вхід синхронізації C), слід мати на увазі, що при C = 0 внутрішній стан тригера не змінюється незалежно від станів входів X і Y, тобто Q (t +1) = Q (t), а при C = 1 синхронний тригер функціонує як відповідний асинхронний. З урахуванням цього отримуємо скорочену (табл. 3.19 – а) і повну (табл. 3.19 – б) таблиці переходів синхронного тригера. Із повної таблиці переходів запишемо комбінації вхідних сигналів, які відповідають всім можливим переходам (табл. 3.19 – в).

Таблиця 3.19 – Переходи синхронного тригера

а)

t			(t+1)
C	X	Y	Q
0	0	0	Q(t)
0	0	1	Q(t)
0	1	0	Q(t)
0	1	1	Q(t)
1	0	0	0
1	0	1	Q(t)
1	1	0	$\overline{Q(t)}$
1	1	1	1

б)

t				t+1
C	X	У	Q	Q
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

в)

Q(t)	Q(t+1)	X	У
0	0	0	0
		0	1
0	1	1	0
		1	1
1	0	0	0
		1	0
1	1	0	1
		1	1

Матриця переходів синхронного тригера повторюватиме матрицю переходів аналогічного асинхронного тригера.

Розглянемо наступний приклад.

Словесний опис роботи асинхронного RS-тригера звучить наступним чином: RS-тригер – двухвходовий, який при подачі активного сигналу на S-вхід (S = 1) і неактивного сигналу на R-вхід (R = 0) встановлюється в

одиничний стан (Q = 1). При подачі активного сигналу на R-вхід (R = 1) і неактивного сигналу на S-вхід (S = 0) тригер встановлюється в нульовий стан (Q = 0).

При подачі двох не активних сигналів на S і R входи (S = 0; R = 0) тригер зберігає попередній стан (Q (t +1) = Q (t)). Одночасна подача двох активних сигналів на S і R входи (S = 1; R = 1) заборонена. Якщо ж така ситуація виникає, то тригер вважається байдужим (невизначеним), (Q (t +1) = X).

Розглянемо дію цього тригера.

Скорочена таблиця переходів асинхронного RS-тригера представлена в табл. 3.20.

Таблиця 3.20 – Скорочена таблиця переходів асинхронного RS-тригера

Входи		Вихід	Режим
t		t+1	
R(t)	S(t)	Q(t+1)	
0	0	Q(t)	Зберігання
0	1	1	Установка в 1
1	0	0	Установка в 0
1	1	x	Невизначений стан

На основі скороченою таблиці переходів побудуємо повну таблицю переходів асинхронного RS-тригера (табл. 3.21).

Таблиця 3.21 – Повна таблиця переходів асинхронного RS-тригера

R	S	Q(t)	Q(t+1)	$\overline{Q}$ (t+1)
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	x	x
1	1	1	x	x

Щоб скласти матрицю переходів RS-тригера, спочатку представимо повну таблицю переходів в дещо іншому вигляді (табл. 3.21).

Таблиця 3.21 – Модифікована таблиця переходів

Q(t)	Q(t+1)	R(t)	S(t)
0	0	0	0
		1	0
0	1	0	1
1	0	1	0
1	1	0	0
		0	1

З табл. 3.22 випливає, що для переходу 0 → 0 S(t) = 0, а R(t) може дорівнювати як 0, так і 1, тобто бути невизначеним. Позначимо невизначений стан R(t) символом b1. Перехід від 0→1 відбувається при однозначній комбінації сигналів R(t)=0, а S(t) = 1. Як і перехід 1→0 можливий при однозначній комбінації сигналів R(t) = 1, а S(t) = 0. При переході 1→1 R(t) = 0, а S(t) може бути як 0, так і 1. Позначимо невизначений стан S(t) символом b2. І тоді остаточно матриця переходів асинхронного RS-тригера, у якій показані всі можливі переходи Q(t) → Q(t +1) і відповідні цим переходам значення вхідних сигналів R(t) і S(t) прийме вигляд, представлений у таблиці 3.23.

Таблиця 3.23 – Матриця переходів асинхронного RS-тригера

Q(t)	Q(t+1)	R(t)	S(t)
0	0	b_1	0
0	1	0	1
1	0	1	0
1	1	0	b_2

Матриця переходів RS-тригера буває необхідна при синтезі інших типів тригерів, а також при синтезі автоматів з пам'яттю.

Для запису характеристичного рівняння асинхронного RS-тригера, скористаємося повною таблицею переходів RS-тригера (табл. 3.21), розглядаючи її як табличне завдання функції Q(t +1) = f (R(t), S(t), Q(t)).

Виходячи з таблиці переходів, можна зробити висновок, що дана функція є не повністю визначеною. Тому для її мінімізації скористаємося методом карт Карно [105] для функції 3-х змінних (рис. 3.6).

R \ SQ	00	01	11	10
0		1	1	1
1			*	*

Рисунок 3.6 – Застосування методу карт Карно

В результаті довизначення невизначених наборів «1», отримаємо наступний результат мінімізації:

$$Q(t+1) = \overline{R}Q \vee S \qquad (3.7)$$

Це і є характеристичне рівняння асинхронного RS-тригера.

Граф переходів асинхронного RS-тригера наведено на рис. 3.7.

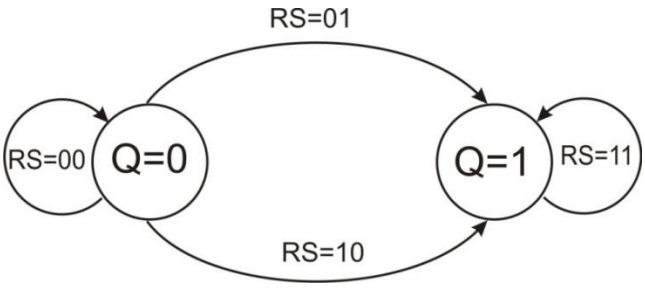


Рисунок 3.7 – Граф переходів асинхронного RS-тригера

З використанням наведеної моделі (3.7) та графа переходів (рис. 3.7) розроблена програмна реалізація надання рекомендацій. Основні фрагменти лістингу коду, створеного на Python, наведено у Додатку Г.

3.5 Висновок за третім розділом

За підсумками виконання третього розділу роботи можна зробити наступні висновки:

1) представлена модель надання рекомендацій з використанням правил комп'ютерної логіки. Розроблено механізм представлення переходів користувача з метою відбору параметрів для надання рекомендацій. Прийнято, що розглядаються кроки у дискретному середовищі, коли за діями користувача рішення може приймати лише два значення – одиничне (на основі отриманих маркерів користувачу можна надати рекомендацію) та нульове (маркери не орієнтують, що даний продукт може зацікавити користувача, тобто, рекомендацію щодо пропозиції товару чи послуги надати неможливо);

2) наведено моделі переходів в рекомендаціях в залежності від зміни уподобань за допомогою створення графів переходів за різними вхідними параметрами, що характеризують дії користувача та використання еквівалентних станів уподобань при формуванні матриць переходів рекомендацій за зміни окремих ознак;

3) рекомендаційна система, що пропонується до реалізації, має в основі синтез цифрових автоматів. На цій основі розроблено механізм рекомендацій за ознакою структурної повноти;

4) розроблено механізм формування тригерів для створення рекомендацій за зміни маркерів користувача. Під тригером у дані роботі прийнято стійкий стан рекомендації до зміни ключових ознак на вході системи – маркерів користувача. У підсумку, за наведеним математичним базисом, за допомогою мови Python розроблено рекомендаційну систему.

5) матеріали, що подані в розділі, були представлені у роботах [97, 102].

РОЗДІЛ 4
РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ З КОМБІНУВАННЯ МЕТОДІВ ДЛЯ ПОБУДОВИ РЕКОМЕНДАЦІЇ НА ОСНОВІ ПЕРЕВАГ ТА ВІДМОВ КОРИСТУВАЧА

4.1 Реалізація механізму рекомендації через тригер за булевим базисом «І-Або-Ні» та «Або-Ні»

Особливість підходу, що пропонується в роботі, полягає у реалізації такого механізму, який дозволяє вичленовувати та аналізувати маркери споживача навіть у тих випадках, коли користувач переглядав якусь позицію продукту але не вибрав остаточно. Аналіз такої поведінки та використання показників «не вибору» у якості вхідних параметрів у моделі рекомендаційного механізму дозволить розширити можливий спектр рекомендацій за булевим базисом «І-Або-Ні» та «Або-Ні» через асинхронний RS-тригер за моделлю (3.7), що можна представити за (рис. 4.1.).

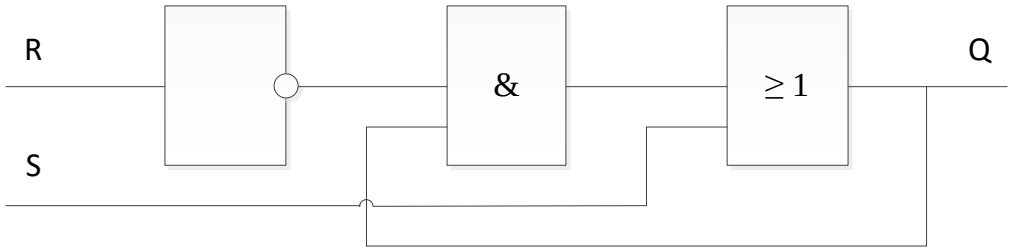


Рисунок 4.1 – Схема асинхронного RS-тригера за булевим базисом «І-Або-Ні» та «Або-Ні»

Однак, найбільш просто схема RS-тригера реалізується в базисі «Або-Ні». Для цього можна перетворити модель (3.7) до виду, зручного для реалізації в базисі «Або-Ні» за правилом Де-Моргана [106]:

$$\overline{Q}^{t+1} = \overline{\overline{\overline{R^t} \cdot Q^t} \vee S^t} = \overline{\overline{(R^t \vee \overline{Q^t})} \vee S^t}. \quad (4.1)$$

У підсумку можна отримати наступну функціональну схему RS-тригера в базисі «Або-Ні» з інверсним виходом. (індекси t та t+1 на схемі не проставлено) (рис. 4.2).

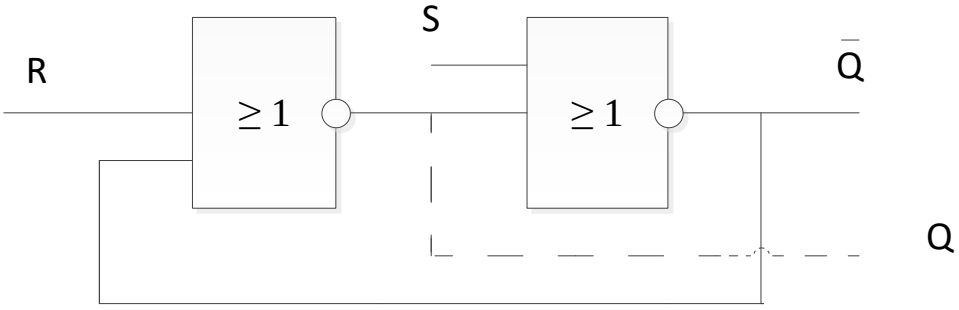


Рисунок 4.2 – Схема RS-тригера в базисі «Або-Ні»

Виконавши перетворення за моделлю (3.7), але для інверсного виходу з врахуванням $\overline{Q}^{t+1}$, можна показати, що:

$$Q^{t+1} = \overline{\overline{(S^t \vee Q^t)} \vee R^t}. \quad (4.2)$$

Зазначене у моделі (4.2) відповідає функціональній схемі RS-тригера з прямим виходом (рис. 4.3).

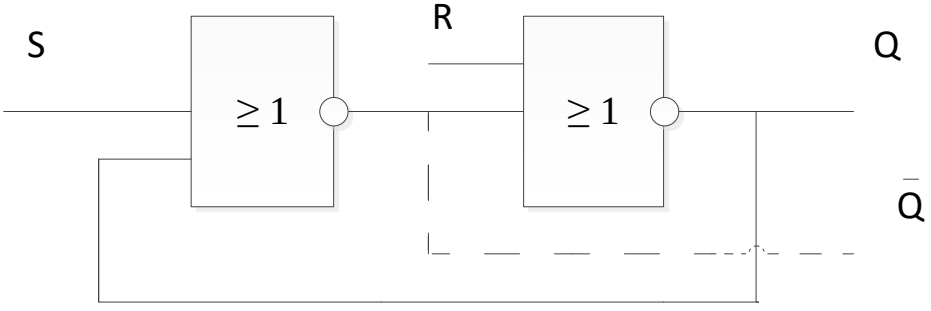


Рисунок 4.3 – Тригер з прямим входом

Порівнюючи схеми з рис. 4.2 і рис. 4.3, можна об’єднати обидві схеми з представленням прямого та інверсійного виходів (рис. 4.4).

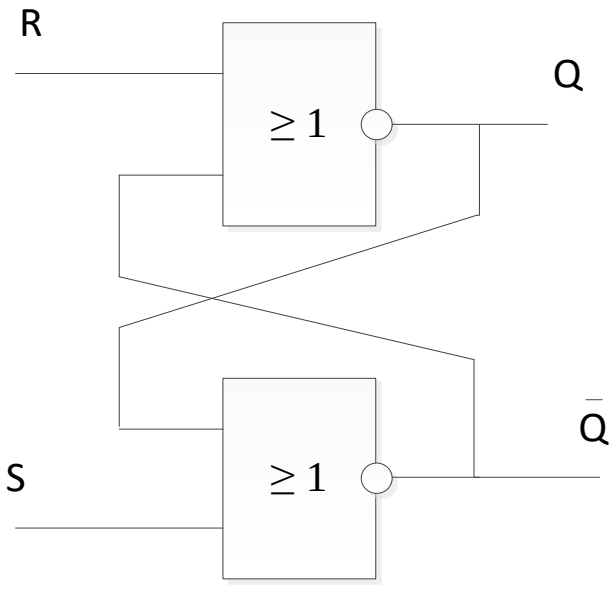


Рисунок 4.4. – Об’єднана схема тригера

У вигляді умовного графічного зображення, зручного для реалізації в процесі програмування, це можна представити у вигляді рис. 4.5.

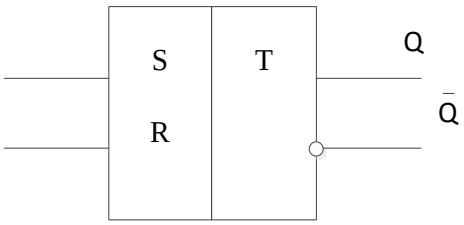


Рисунок 4.5 – Спрощена схема тригера

Часові діаграми роботи подібної схеми при програмній реалізації RS-тригера наведені на рис. 4.6.

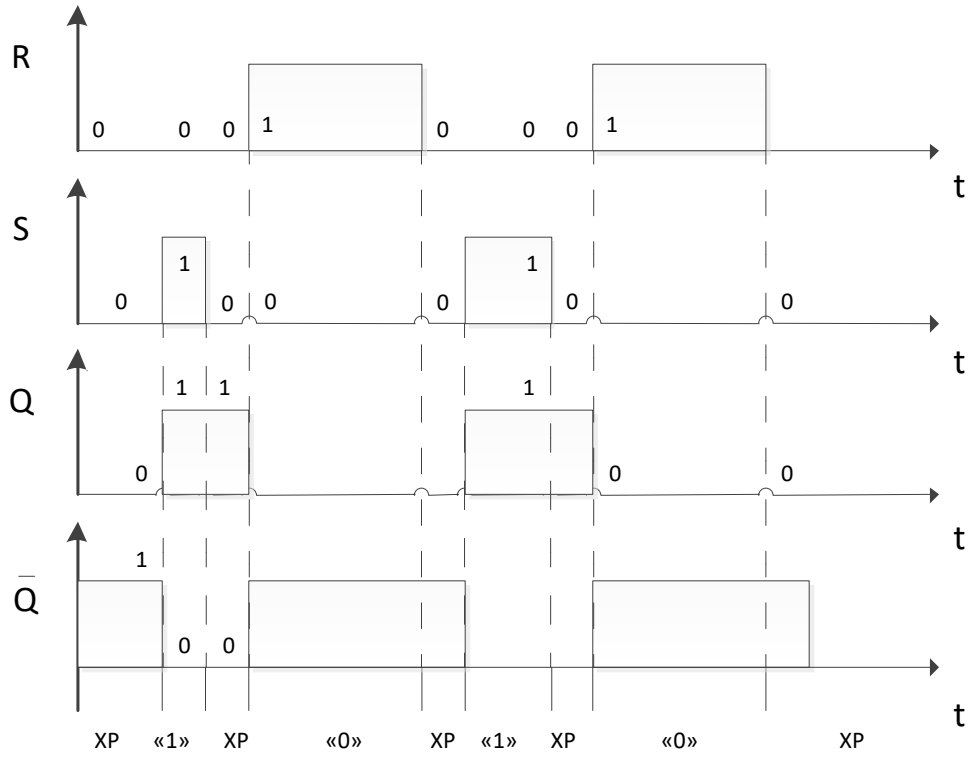


Рисунок 4.6 – Часові діаграми роботи

Часові діаграми за рис. 4.6 можна пояснити наступним чином.

На інтервалі 0 ... t_1, RS = 00 тригер зберігає стан (Q = 0, $\overline{Q} = 1$).

У момент часу t_1, RS = 01, що відповідає режиму встановлення тригера в «1» (Q = 1, $\overline{Q} = 0$).

Режим установки в «1» діє на інтервалі $t_1 – t_2$.

У момент часу t_2, RS = 0, що відповідає режиму переходу тригера в режим зберігання «1», діє на інтервалі $t_2 – t_3$.

У момент часу t_3, RS = 10, що відповідає режиму встановлення тригера в «0» (Q=0, $\overline{Q} = 1$).

Режим установки в «0» діє на інтервалі $t_3 – t_4$.

У момент часу t_4, RS = 00, що відповідає переходу тригера в режим зберігання «0» (Q=0, $\overline{Q} = 1$).

Режим установки в «1» діє на інтервалі $t_4 – t_5$.

У момент часу t_5, RS = 01, тригер встановлюється в «1» (Q=1, $\overline{Q} = 0$).

На інтервалі $t_5 – t_6$ діє режим установки в «1».

У момент часу t_6, RS = 00, і тригер переходить в режим зберігання «1», який триває до моменту t_7.

У момент часу t_7, RS = 10, що відповідає режиму встановлення тригера в «0» (Q=0, $\overline{Q} = 1$).

Режим установки в «0» діє на інтервалі $t_7 – t_8$.

У момент часу t_8, RS = 00, і тригер переходить в режим зберігання «0».

Приклад реалізації інформаційної технології, що враховує комбінування методів для побудови рекомендацій на основі переваг та відмов користувача із застосуванням даного підходу (Додаток Д) дозволив провести сортування уподобань за таким фактором поведінки користувача, як «гучність» стосовно музичних творів у порівнянні з іншими маркерами користувача (рис. 4.7).

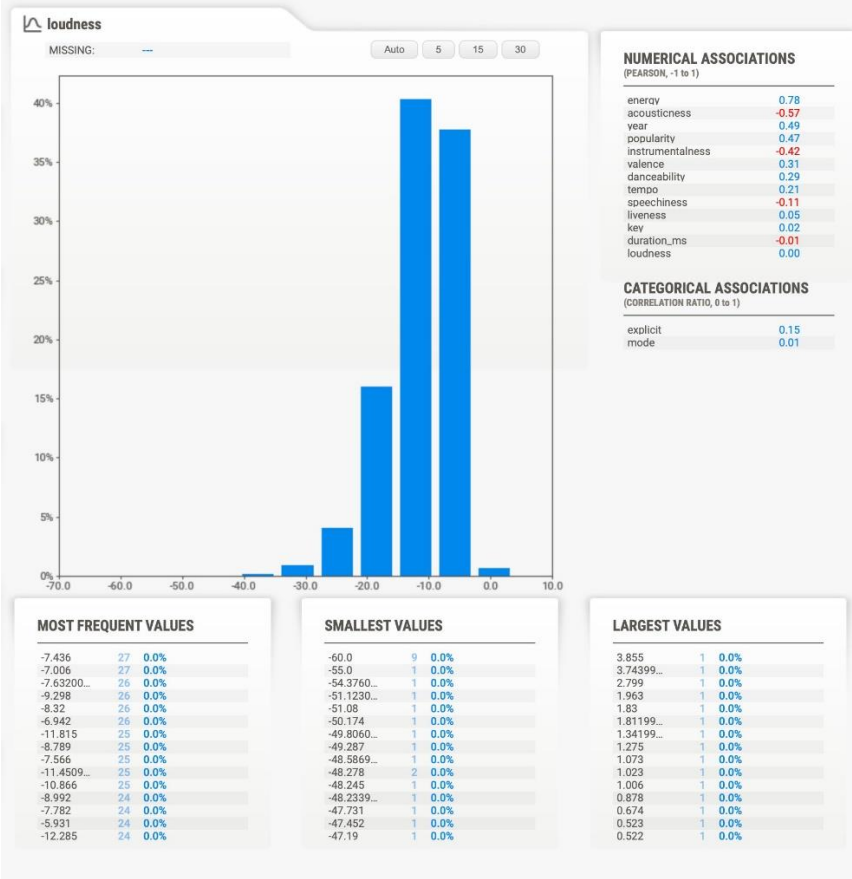


Рисунок 4.6 – Програмне представлення сортування уподобань користувача

На практиці подібне (рис. 4.6) реалізується наступним чином: користувач може мати маркер стосовно гучності музичного твору, тоді базис вибірки чітко буде визначений за тригером «Або-Ні», а може не мати такого маркера, тобто, користувачу байдужа потужність музичного твору (як на рис. 4.6 можна побачити аналіз перебору маркерів та зазначення «0»), і тоді тригер спрацьовує за базисом «І-Або-Ні». В останньому випадку користувачу будуть рекомендовані твори, що враховують інші маркери користувача. Якщо базис «Або-Ні» відповідає умовам відбору, то користувачу буде надана більш точна рекомендація.

Це можна пояснити наступним чином: знову повертаючись до схеми (рис. 3.1) та враховуючи наступні трансформації механізму обробки інформації (рис. 3.2 та 3.3), варто пам’ятати, що на структурному рівні кожна буква вхідного алфавіту x ∈ X, кожна буква вихідного алфавіту y ∈ У і кожна буква алфавіту станів a ∈ A кодується двійковими векторами (двійковими наборами), число компонент яких визначається таким чином:

$$Квх=int(\log_2|X|);\quad Квих=int(\log_2|У|);\quad Кстан=int(\log_2|A|);$$

де:

int – найближче більше ціле число,

| X |, | У |, | A | – потужність алфавіту вхідного, вихідного і станів, відповідно.

Під потужністю алфавіту для випадку вирішення поставлених у роботі задач приймаються, як раніше зазначалося, маркери користувача, у тому числі і ті, які «не спрацювали» у попередньому виборі. Літери алфавіту (X, У, A) у даному випадку є умовними позначеннями маркерів користувача, які у підсумку описуються двійковими векторами. Наприклад, розглянемо інформацію, представлену переходами та виходами (табл. 4.1) щодо поведінки окремого користувача, яка виступає основою для реалізації програмного представлення сортування уподобань користувача (див. рис. 4.6).

Таблиця 4.1 – Таблиця переходів та виходів за поведінкою окремого користувача

A \ X	x_1	x_2
a_1	a_2/y_1	a_3/y_3
a_2	a_1/y_2	a_2/y_1
a_3	a_2/y_2	a_1/y_1

Випишемо алфавіти автомата та визначимо довжини векторів кодів алфавітів:

$X = \{x_1, x_2\}$; Квх=int($\log_2|2|$)= 1,

$У = \{y_1, y_2, y_3\}$; Квих=int($\log_2|3|$)= 2,

$A = \{a_1, a_2, a_3\}$; Кстан=int($\log_2|3|$)= 2.

Таблиці кодування можна навести у вигляді схеми (рис. 4.7), яка показує логічний процес обробки інформації системою.

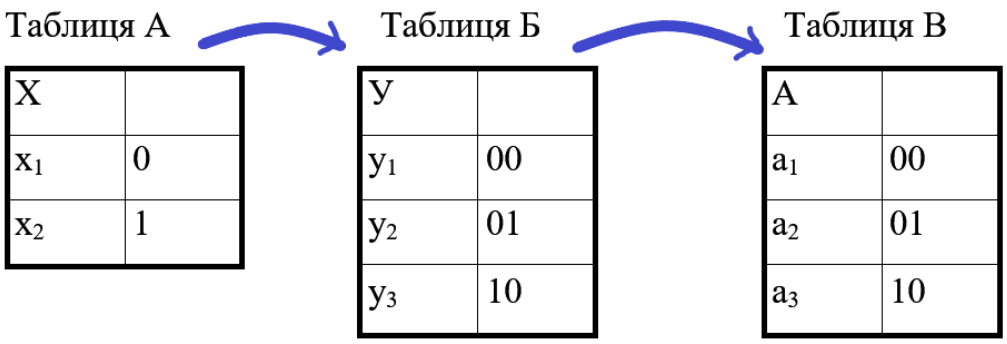


Рисунок 4.7 – Логічний процес обробки інформації

Результуюча таблиця (табл. 4.2) – структурна таблиця переходів і виходів автомата. Отриманням структурної таблиці переходів і виходів

автомата закінчується етап кодування та відбувається перехід до етапу формування рекомендації.

Таблиця 4.2 – Структурна таблиця переходів та виходів автомата

A \ X	0	1
00	01/00	10 /10
01	00 /01	01/00
10	01 / 01	00 / 00

У загальному випадку, кожній кодованій букві може бути присвоєний довільний двійковий вектор, але обов'язково дві різні букви одного алфавіту повинні кодуватися різними векторами. Коди, які відповідають символам різних алфавітів можуть збігатися, в розглянутому прикладі – коди вихідних сигналів і станів. При кодуванні вихідних сигналів з врахуванням деяких маркерів користувача, які не призвели до вибору продукту, використовується ваговий метод (метод вагових коефіцієнтів). У цьому випадку алгоритм рекомендації розширюється наступними кроками:

1) кожному вихідному сигналу y_i ставиться у відповідність ціле число P_i, яке дорівнює числу появ сигналу y_i в таблиці виходів автомата;

2) числа P_i сортуються за зменшенням;

3) вихідний сигнал y_i з найбільшою вагою (P_i max) кодуються кодом, що містить всі 0 (00 ... 00);

4) наступні L вихідних сигналів (де L-число розрядів у двійковому векторі вихідного сигналу) за списком зменшення ваги (крок 2) кодуються кодами, що містять одну 1 (00 ... 01, 00 ... 10, ..., 10 ... 00);

5) для кодування наступних за списком зменшення вихідних сигналів використовуються всі коди, що містять дві одиниці, потім три одиниці і т.д., поки не будуть закодовані всі вихідні сигнали.

В результаті отримується таке кодування, при якому чим частіше зустрічається подібний вихідний сигнал у таблиці виходів, тим менше одиниць міститься в його коді.

4.2 Вирішення задач логічної еквівалентності в процесі розробки інформаційної технології рекомендаційної підтримки

Розглянемо деяке слово, що поступає на вхід автомата Мура, який виступає операційним автоматом в системі рекомендацій. Наприклад, це слово задано наступним чином:

$$X = x_1\, x_2\, x_1\, x_2\, x_1.$$

Реакція автомата Мура (рис. 4.8) на вхідне слово X буде наступною:

Вхідне слово	x_1	x_2	x_1	x_2	x_1		
Стан	a_1	a_2	a_4	a_3	a_1	a_3	a_1
Вихідне слово	y_2	y_1	y_3	y_1	y_2	y_1	y_2

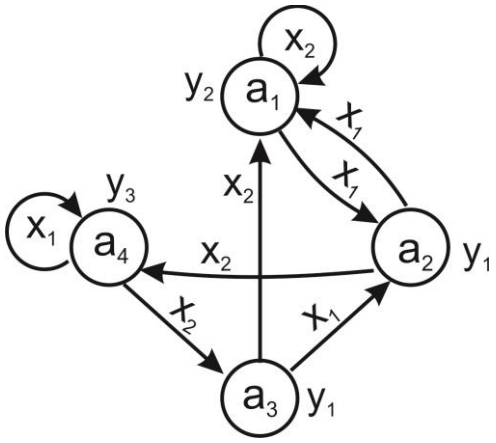


Рисунок 4.8 – Представлення реакції автомата Мура на вхідне слово

При цьому виді перетворення сигнал y_g, записаний поруч з вершиною a_S переноситься на всі дуги, що входять на цю вершину.

Реакція еквівалентного автомата Мілі (рис. 4.9) на вхідне слово X:

Вхідне слово	x_1	x_2	x_1	x_2	x_1		
Стан	a_1	a_2	a_4	a_3	a_1	a_3	a_1
Вихідне слово	y_1	y_3	y_1	y_2	y_1	y_2	

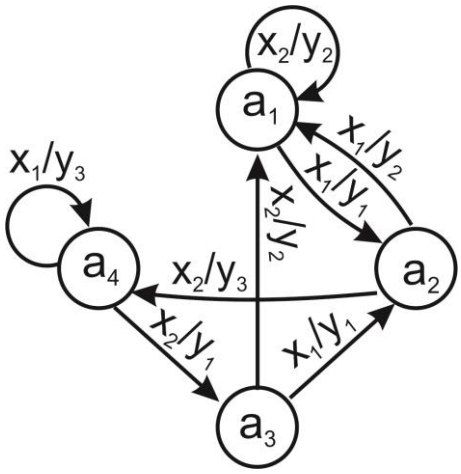


Рисунок 4.9 – Граф автомата Мілі

Без урахування реакції автомата Мура в початковому стані a_1/y_2, так як вона не залежить від x_1 в початковий момент часу, а визначається лише станом автомата в початковий момент часу (a_1), реакції автомата Мура та автомата Мілі є однаковими, тобто, отримано еквівалентні автомати.

У випадку, зазначеному на рис 4.9, матриця з’єднань може бути представлена табл. 4.3.

Таблиця 4.3 – Матриця з’єднань для автомата Мілі

a(t) / a(t+1)	a_1	a_2	a_3	a_4
a_1	x_2/y_2	x_1/y_2	x_2/y_2	-
a_2	x_1/y_1	-	x_1/y_1	-
a_3	-	-	-	x_3/y_1
a_4	-	x_2/y_3	-	x_1/y_3

Далі весь процес надання рекомендацій за підходом логічної еквівалентності відбувається шляхом обробки інформації на вході та виході системи із забезпеченням переходів. Механізм реалізації наведено у Додатку Ж.

Для рішення поставлених задач є важливим також мінімізація числа внутрішніх стану автомату для мінімізації форми запису. Це допомагає отримати скінченний стан автомату, який надасть найбільш точну інформацію на виході.

Розглянемо це на прикладі реалізованої системи, є автомат Мура, заданого таблицею переходів і виходів (табл. 4.4), використовуючи алгоритм Ауфенкампа-Хона, підхід з використанням якого представлений у попередніх розділах роботи.

Таблиця 4.4 – Переходи та виходи автомата Мура

	y_2	y_3	y_4	y_1	y_4	y_2	y_1	y_4	y_2	y_3
A	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
x_1	5	2	9	5	3	1	2	6	5	10
x_2	9	1	6	7	2	3	4	10	1	9
x_3	2	3	8	6	4	9	7	10	10	3

Виконаємо розбиття π_0:

$\pi_0=\{B_1,B_2,B_3,B_4\}$;

$B_1=\{a_4, a_7\}$, $B_2=\{a_1, a_6, a_9\}$, $B_3=\{a_2, a_{10}\}$, $B_4=\{a_3, a_5, a_8\}$.

Побудуємо таблицю розбиття π_0 (табл. 4.5).

Таблиця 4.5 – Розбиття π_0

	B_1		B_2			B_3		B_4		
A	a_4	a_7	a_1	a_6	a_9	a_2	a_{10}	a_3	a_5	a_8
x_1	B_4	B_3	B_4	B_2	B_4	B_3	B_3	B_2	B_4	B_2
x_2	B_1	B_1	B_2	B_4	B_2	B_2	B_2	B_2	B_3	B_3
x_3	B_2	B_1	B_3	B_2	B_3	B_4	B_4	B_4	B_1	B_3

Виконаємо розбиття π_1:

$\pi_1=\{C_1,C_2,C_3,C_4,C_5,C_6,C_7\}$;

$C_1=\{a_4\}$, $C_2=\{a_7\}$, $C_3=\{ a_1, a_9\}$, $C_4=\{ a_6\}$, $C_5=\{a_2, a_{10}\}$, $C_6=\{ a_3\}$, $C_7=\{ a_5\}$, $C_8=\{ a_8\}$.

Результати наведені в табл. 4.6

Таблиця 4.6 – Результуюча таблиця

	C_1	C_2	C_3		C_4	C_5		C_6	C_7	C_8
A	a_4	a_7	a_1	a_9	a_6	a_2	a_{10}	a_3	a_5	a_8
x_1	C_7	C_5	C_7	C_7	C_3	C_5	C_5	C_3	C_6	C_4
x_2	C_2	C_1	C_3	C_3	C_6	C_3	C_3	C_4	C_5	C_5
x_3	C_4	C_2	C_5	C_5	C_3	C_6	C_6	C_8	C_1	C_5

Виконаємо розбиття π_2:

$\pi_2=\{D_1,D_2,D_3,D_4,D_5,D_6,D_7,D_8\}$;

$D_1=\{a_4\}$, $D_2=\{a_7\}$, $D_3=\{a_1,a_9\}$, $D_4=\{a_6\}$, $D_5=\{a_2,a_{10}\}$, $D_6=\{a_3\}$, $D_7=\{a_5\}$, $D_8=\{a_8\}$.

Розбиття π_2 повторює розбиття π_1 – процедуру розбиття завершено.

Можна зробити висновок: $a_1 \equiv a_9$; $a_2 \equiv a_{10}$;

Виберемо довільно з кожного класу еквівалентності D_1, D_2, D_3, D_4, D_5, D_6, D_7, D_8 по одному представнику - в даному випадку з мінімальним номером:

$A'= \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8\}$.

Видаляючи з вихідної таблиці переходів «зайві» стани (a_9,a_{10}), визначаємо мінімальний автомат Мура (табл. 4.7).

Таблиця 4.7 – Мінімальний автомат Мура

	y_2	y_3	y_4	y_1	y_4	y_2	y_1	y_4
A	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
x_1	5	2	1	5	3	1	2	6
x_2	1	1	6	7	2	3	4	2
x_3	2	3	8	6	4	1	7	2

Мінімізацію для автомата Мілі за алгоритмом Ауфенкампа-Хона можна розглянути за наступною таблицею переходів і виходів (табл. 4.8).

Таблиця 4.8 – Таблиця переходів та виходів для автомату Мілі

A	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
x_1	a_1/y_2	a_0/y_1	a_2/y_4	a_3/y_2	a_5/y_1	a_0/y_1	a_1/y_2	a_7/y_4	a_5/y_1
x_2	a_3/ y_3	a_5/ y_2	a_3/ y_1	a_8/ y_3	a_3/ y_2	a_2/ y_2	a_3/ y_3	a_3/ y_1	a_3/ y_2
x_3	a_7/ y_4	a_6/ y_4	a_4/ y_2	a_4/ y_1	a_8/ y_3	a_8/ y_4	a_2/ y_4	a_8/ y_2	a_4/ y_3

Виконаємо розбиття π_1:

$\pi_1=\{B_1, B_2, B_3, B_4, B_5\}$;

$B_1=\{a_0, a_6\}$, $B_2=\{a_1, a_5\}$, $B_3=\{a_2, a_7\}$,. $B_4=\{a_3\}$, $B_5=\{a_4, a_8\}$.

Побудуємо таблицю розбиття π_1 (табл. 4.9).

Таблиця 4.9 – Таблиця розбиття π_1

	B_1		B_2		B_3		B_4	B_5	
A	a_0	a_6	a_1	a_5	a_2	a_7	a_3	a_4	a_8
x_1	B_2	B_2	B_1	B_1	B_3	B_3	B_4	B_2	B_2
x_2	B_4	B_4	B_2	B_3	B_4	B_4	B_5	B_4	B_4
x_3	B_3	B_3	B_3	B_5	B_5	B_5	B_5	B_5	B_5

Виконаємо розбиття π_2:

$\pi_2=\{C_1, C_2, C_3, C_4, C_5, C_6\}$;

$C_1=\{a_0, a_6\}$, $C_2=\{a_1\}$, $C_3=\{ a_5\}$, $C_4=\{a_2, a_7\}$, $C_5=\{a_3\}$, $C_6=\{a_2, a_8\}$

Побудуємо таблицю розбиття π_2 (табл. 4.10).

Таблиця 4.10 – Таблиця розбиття π_2

	C_1		C_2	C_3	C_4		C_5	C_6	
A	a_0	a_6	a_1	a_5	a_2	a_7	a_3	a_4	a_8
x_1	C_2	C_2	C_1	C_1	C_4	C_4	C_5	C_3	C_3
x_2	C_5	C_5	C_3	C_4	C_5	C_5	C_6	C_5	C_5
x_3	C_4	C_4	C_1	C_6	C_6	C_6	C_8	C_6	C_6

Виконаємо розбиття π_3:

$\pi_3=\{D_1, D_2, D_3, D_4, D_5, D_6\}$;

$D_1=\{a_0, a_6\}$, $D_2=\{a_1\}$, $D_3=\{a_5\}$, $D_4=\{a_2, a_7\}$, $D_5=\{a_3\}$, $D_6=\{a_4, a_8\}$

Розбиття π_3 повторює розбиття π_2 – процедуру розбиття завершено.

Можна зробити висновок: $a_0 \equiv a_6$; $a_2 \equiv a_7$; $a_4 \equiv a_8$.

Виберемо довільно з кожного класу еквівалентності $D_1, D_2, D_3, D_4, D_5, D_6$ по одному представнику – в даному випадку з мінімальним номером:

$$A'= \{a_0, a_1, a_2, a_3, a_4, a_5\}.$$

Видаляючи з вихідної таблиці переходів «зайві» стани, визначаємо мінімальний автомат Мілі (табл. 4.11).

Таблиця 4.11 – Мінімальний автомат Мілі

A	a_0	a_1	a_2	a_3	a_4	a_5
x_1	a_1/y_2	a_0/y_1	a_2/y_4	a_3/y_2	a_5/y_1	a_0/y_1
x_2	a_3/ y_3	a_5/ y_2	a_3/ y_1	a_4/ y_3	a_3/ y_2	a_2/ y_2
x_3	a_2/ y_4	a_0/ y_4	a_4/ y_2	a_4/ y_1	a_4/ y_3	a_4/ y_4

У підсумку запропонована мінімізація дозволяє визначати еквівалентність автоматів. Для задач реалізації системи за допомогою веб-

програмування таке рішення допомагає більш спрощено представити дискретну систему за окремими перевіреними кроками, з контролем щодо виявлення можливих помилок на кожному кроці реалізації. Варто зазначити, що для більш складних випадків реалізації рекомендаційної системи класів еквівалентності може бути значно більше, що дозволяє за заданих вхідних параметрів врахувати більше аспектів, які можуть вказувати на можливі вихідних символів (рекомендації).

4.3 Реалізація інформаційної технології рекомендаційної підтримки прийняття рішень у вигляді веб-додатку

Описана інформаційна технологія (ІТ) рекомендаційної підтримки прийняття рішень реалізована у вигляді веб-додатку (Додаток З).

Опис можливостей розробленої технології:

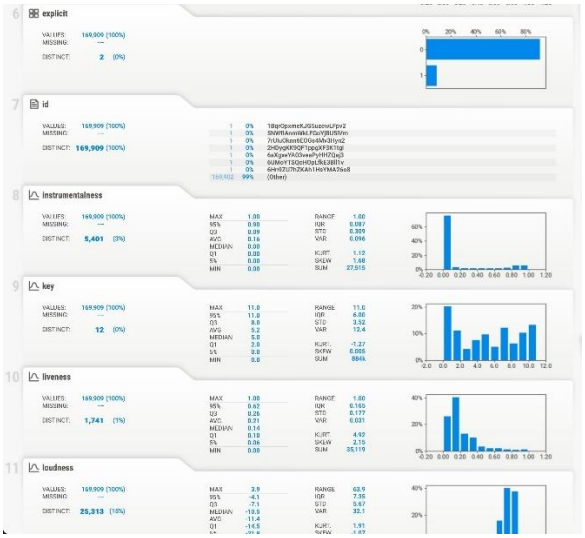
– автоматизація збору, обробки й завантаження даних з відкритих і конфіденційних джерел (за умови надання дозволу General Data Protection Regulation (EU GDPR) [95]);

– виявлення реальних або потенційних рекомендацій користувачу на основі спостережень і обробки отриманої інформації з різнорідних джерел на основі визначених маркерів користувача;

– формування внутрішніх звітів за результатами аналізу інформації, інформування користувача про проходження процесу аналізу, оперативне інформування про виявлені ситуації щодо неможливості виконання запиту.

Інтерфейс розробленого веб-додатку наведено на рис. 4.10

а)



б)

в)

Рисунок 4.10 – Інтерфейс програмної реалізації, де а), б), в) є послідовним представленням інтерфейсу користувача

З лівої частини інтерфейсу перелічено параметри (маркери користувача), за якими відбирається інформація, а з лівої сторони надається звіт щодо результатів обробки. Для аналізу використано 17 маркерів користувача. Для кінцевого формування рекомендацій для кожного користувача можуть бути використані від 1 до 17 параметрів.

Інформаційна технологія рекомендаційної підтримки складається з декількох підсистем. Нижче коротко наведені основні характеристики цих підсистем.

Безпосередньо підсистема збору інформації та формування рекомендацій виконує наступні функції:

– збір даних з відкритих і конфіденційних джерел (за умови надання дозволу EU GDPR) [95]);

– настроювання правил добування інформації із сайтів у мережі Інтернет, блогів, форумів, електронної пошти за стандартними протоколами;

– виконання операцій з добування інформації із джерел в автоматичному режимі з веденням журналу;

– попередня обробка інформації з метою її приведення до універсального формату зберігання;

– завантаження інформації в сховище даних;

– обробка посилань на медіа-контент (відеороліки, аудіозаписи, графічні зображення) з можливістю їхнього завантаження й зберігання;

– перевірка актуальності й вірогідності даних;

– настроювання правил автоматизованого аналізу інформації;

– аналіз інформації з метою виявлення об'єктів (персони, організації, географічні об'єкти) з урахуванням настроєних правил;

– формування значень маркерів користувача;

– оцінка значень маркерів користувача за певними критеріями;

– аналіз тенденцій, що мають місце при застосуванні маркерів користувача;

– прогнозування розвитку процесів для створення рекомендацій;

– візуалізація результатів аналізу інформації в табличному й графічному виді;

– формування звітів за результатами аналізу інформації;

– поточне інформування користувача про стан процесів обробки інформації за діючими регламентами.

Підсистема адміністрування інформаційної технології надання рекомендацій має наступні функції:

– ведення переліку облікових записів користувачів;

– установлення й зміна прав доступу користувачів до функцій і даних системи;

– авторизація користувачів при запуску;

– ведення протоколів дій користувачів;

– обмеження доступу користувачів до функціональних і інформаційних блоків.

Підсистема адміністрування довідкової інформації передбачає виконання:

– забезпечення ведення довідників і класифікаторів, необхідних для функціонування системи, з можливістю імпорту елементів довідників і класифікаторів із зовнішніх файлів і систем;

– пошук елементів довідників і класифікаторів по заданих атрибутах;

– підтримка версійності довідкової інформації;

– забезпечення можливості автоматизованого зіставлення елементів різних довідників і класифікаторів за їхніми атрибутами.

Підсистема адміністрування сховища даних інформаційної технології передбачає:

– ведення опису показників сховища даних;

– зберігання даних за показниками, ведення яких передбачається в ІТ;

– забезпечення можливості уведення даних користувачами з підтримкою статусу уведених даних;

– підтримка версійності збережених даних (при необхідності, наприклад, у частині довідників і класифікаторів);

– пошук інформації в базі даних у режимі реального часу;

– зберігання довідників і класифікаторів, необхідних для роботи системи, і інших метаданих, необхідних для доступу функціональних компонентів ІТ до вихідних даних (у т.ч. розрізи, одиниці виміру й ін.);

– індексація інформації для оптимізації пошукових запитів;

– формування вітрин даних.

Після входу в Інтернет-браузері за адресою веб-додатка системи з'явиться діалогова форма для авторизації користувачів (рис. 4.11).

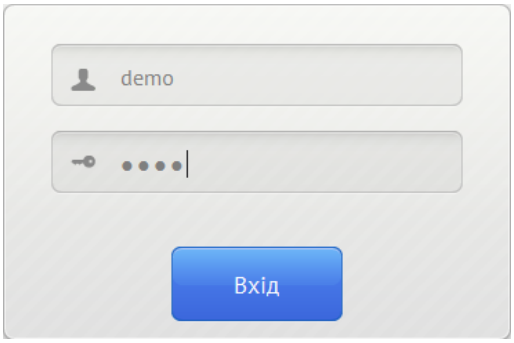
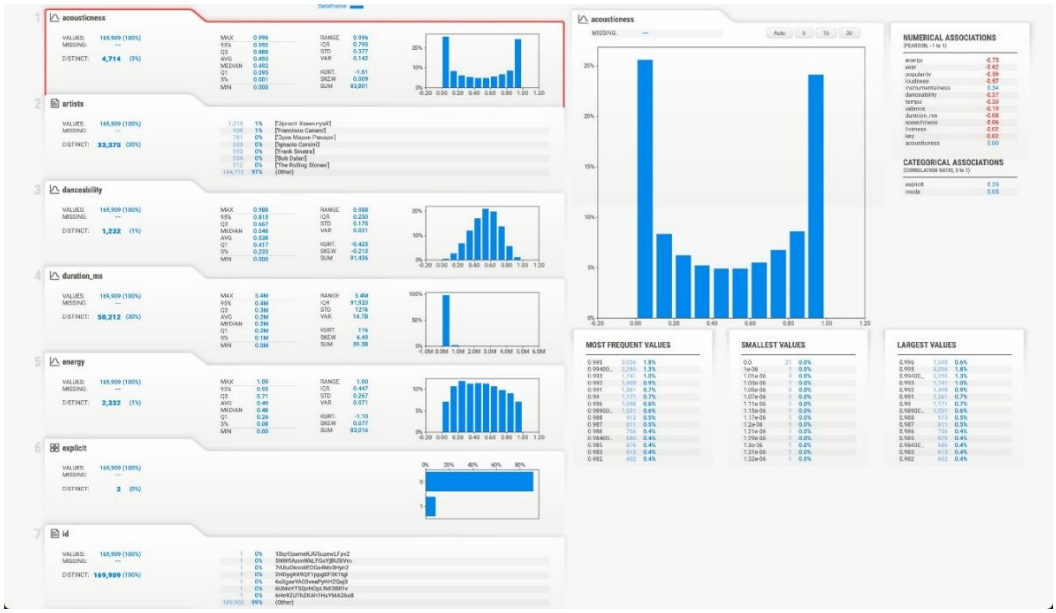


Рисунок 4.11 – Вікно авторизації користувача у веб-додатку

У даній формі необхідно ввести логін і пароль, після чого натиснути кнопку «Вхід». Після успішної авторизації користувача протягом деякого часу у розробленій демо-версії буде проводитися з'єднання із сервером і завантаження веб-додатка, після чого з'явиться вікно інтерфейсу користувача де у правій частині наведено аналіз параметрів щодо конкретного маркеру користувача, а у лівій частині інтерфейсу користувача наведені результати обробки вхідних параметрів та їх візуалізація (рис. 4.12).

На першому рисунку (рис. 4.12 – а) наведений аналіз параметрів щодо маркеру користувача, що стосується акустичних вимог. Зліва , у вікні, відразу формується рекомендація за критерієм асоціації – тобто, вибираються інші маркери, що формують вимоги, які можуть призвести до вірогідності здійснення цільової дії. Тобто, користувач скоріше вибере рекомендоване, ніж відмовиться від вибору.

Те ж саме відбувається і за маркерами щодо вибору виконавців (рис. 4.12 – б). У цьому випадку наведений приклад реалізації рекомендаційної підтримки щодо вибору книги.



а)

б)

Рисунок 4.12 – Структура інтерфейсу користувача за роботою з маркерами

Наведені графіки й діаграми являють собою різні форми графічного відображення табличних даних і призначені для їхнього наочного відображення. Вся інформація в ІТ рекомендаційної підтримки представлена таблично. Діаграма ґрунтується на якій-небудь області таблиці й зберігає

зв'язок з даними. Будь-які зміни даних у таблиці приводять до перемальовування діаграми з урахуванням цих змін.

При переході у ІТ безпосередньо до аналізу даних забезпечується можливість перегляду й аналізу даних у наступних online analytical processing (OLAP) – поданнях, які можна відкрити або з навігатора ліворуч, або вибрати зі списку, що відкривається при виборі пункту навігатора.

Користувачу ІТ рекомендаційної підтримки доступний пошук за ключовими словами інформації, що має деякі особливості для кожного розділу даних:

1) показники. У даному розділі пошук здійснюється як за найменуванням показника, так і за найменуванням набору даних;

2) набори даних. У даному розділі пошук здійснюється за найменуванням набору даних, а також за найменуванням показників у наборах даних;

3) звіти. Пошук здійснюється тільки за найменуванням звітів;

4) документи. Пошук здійснюється за найменуванням й вмісту документів;

Крім пошуку за ключовими словами підтримуються наступні додаткові типи пошукових запитів:

– обрамлення пошукового запиту лапками із двох сторін для пошуку за точним словом або фразою;

– знак мінус (-) перед словами, які потрібно виключити з результатів пошуку;

– знак плюс (+) перед словами, для пошуку слів, які обов'язково включити в результати пошуку.

Особливості:

– не враховуються скорочення й слова-синоніми;

– врахування відмінків у поточній версії не реалізоване;

– пошук за частиною слова не підтримується.

Для реалізації завдань використана відкрита програмна бібліотека для машинного навчання TensorFlow. Завдяки цьому розроблений та опрацьований інструмент аналізу, що призначений для оперування багатомірними масивами даних. Багатомірний масив у загальному випадку може містити необмежене число вимірів. Для подання масиву у вигляді таблиці частина вимірів фіксується – у кожному з них визначається елемент, по якому буде побудований зріз даних. Зріз даних являє собою двовимірну таблицю даних, у якої в шапці й боковині відображаються елементи вимірів масиву, розташовані по стовпцях і по рядках.

Приклад роботи з програмною бібліотекою TensorFlow для реалізації задач роботи наведений на рис. 4.13.

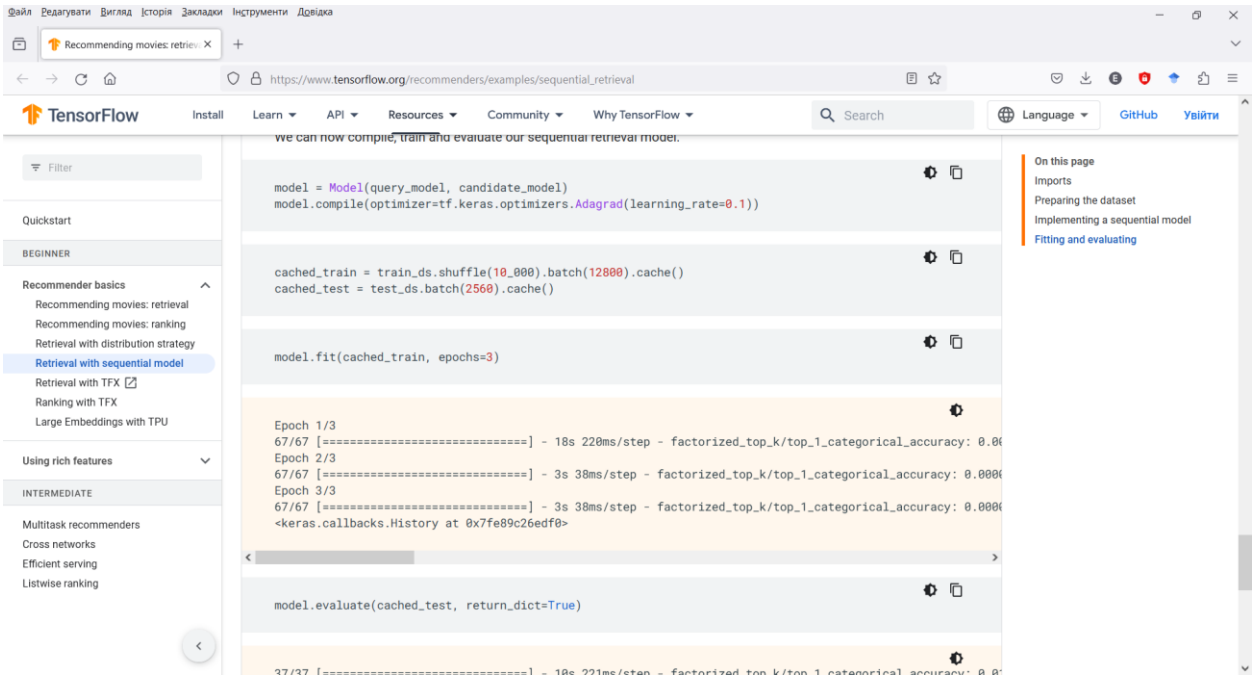


Рисунок 4.13 – Приклад роботи з програмною бібліотекою TensorFlow

Як зазначалося, діаграма в аналізі даних будується на базі зрізу даних, тобто отриманої двовимірної таблиці. При цьому інструмент аналізу IT рекомендаційної підтримки дає можливість оперувати декількома багатомірними масивами одночасно за допомогою TensorFlow. У цьому

випадку будується так званий віртуальний масив, що містить загальні й частки виміру всіх вхідних у нього таблиць з даними.

При виконанні функцій збору, аналізу інформації та формування рекомендації здійснюються наступні операції:

– завантаження в систему масивів статистичних даних і довідкової інформації з використанням інтерфейсу програми обміну статистичними даними, у тому числі операції по завантаженню в систему й розбору файлів xml формату, завантаженню статистичних даних і довідників у буферні структури, операції по перевантаженню даних з буферних структур в оперативний склад даних (база з обміну даними) з урахуванням здійснення перевірок форматно-логічного контролю й забезпечення ведення версійності даних;

– імпорт із системи довідкової інформації з використанням інтерфейсу програмної реалізації;

– формування наборів даних, що полягає в:

а) автоматичному виборі по кожному з показників, використовуваних у методиці створення рекомендацій, найбільш актуальної (по даті завантаження в систему) версії даних;

б) формуванні звітних форм для користувача, у яких він має можливість переглядати дані за актуальною версією, вибирати іншу версію даних, переглядати по ній дані;

– формування вітрин даних по обробленим маркерам користувача з деталізацією за показниками у відповідності зі сформованим набором даних.

Проведене тестування інформаційної технології рекомендаційної підтримки прийняття рішень відбувалося за допомогою порівняння показників розробленої технології з популярними базовими системами за критерієм точності та стабільності отриманих результатів (табл. 4.12).

Таблиця 4.12 – Порівняльна таблиця результатів тестування розробленої технології рекомендаційної підтримки

Ресурси	POP	S-POP	IT рекомендаційної підтримки
Top 100: Ukraine	0,0050	0,2672	0,2518
Hot Hits Україна	0,1061	0,2844	0,1055
book24.ua	0,0412	0,3133	0,0624

Для створення таблиці використано:

а) POP – прогноз популярності, який завжди рекомендує найпопулярніші елементи навчання. Оцінка здійснюється шляхом надання подій сеансу по одному та перевірки рангу елемента наступної події;

б) S-POP – базовий рівень популярності поточного сеансу.

Розроблена технологія порівнюється на платформах найбільш популярних книжок (book24.ua) та аудіосервісів (TOP 100, Hot Hits Україна). Була проаналізована 1 тис. сесій. Оцінка здійснюється шляхом надання подій сеансу та перевірки рангу елемента наступної події. Після завершення сеансу результат скидається в нуль. Обмеженнями даного тестування було відносно невелика кількість сеансів, проте отримані результати показують деякі переваги розробленого підходу до створення системи рекомендаційної підтримки користувачів. Також у використовуваних рекомендаційних системах, орієнтованих на торгівлю, часто використовується попередня фільтрація з урахуванням популярності, а не лише за маркерами користувача.

Отримані показники за розробленою технологією за деякими параметрами мають кращі показники, ніж існуючі базові системи. Але, наприклад, по книжковому порталу це обумовлено тим, що розроблена система адаптована саме під кириличний текст, тому пошук проходить без відносних затримок. В цілому розроблена інформаційна технологія

рекомендаційної підтримки прийняття рішень надає кращий результат, у порівняння з базовими системами, на 0,6 – 5,8%.

4.4 Висновок за четвертим розділом

За підсумками виконання четвертого розділу роботи можна зробити наступні висновки:

1) реалізовано механізм рекомендації через тригер за булевим базисом «І-Або-Ні» та «Або-Ні», який дозволяє вичленовувати та аналізувати маркери споживача навіть у тих випадках, коли користувач переглядав якусь позицію продукту але не вибрав остаточно. Наведені приклади реалізації, функціональні схеми та математичне обґрунтування;

2) представлене вирішення логічної еквівалентності в процесі розробки інформаційної технології рекомендаційної підтримки за допомогою підходів, використовуваних в комп'ютерній логіці. Запропонований підхід до мінімізації числа станів. Таке рішення допомагає більш спрощено представити дискретну систему за окремими перевіреними кроками, з контролем щодо виявлення можливих помилок на кожному кроці реалізації у якості веб-сервісу;

3) реалізована інформаційна технологія рекомендаційної підтримки прийняття рішень у вигляді веб-додатку. Наведений лістинг кодів програмної реалізації. Отримані результати тестування розробленої технології рекомендаційної підтримки дозволяють зробити висновок відносно наявних переваг отриманої технології з існуючими базовими системами на 0,6 – 5,8%;

4) матеріали, що подані в розділі, були частково представлені у роботі [107] та у матеріалах конференції [108 – 109].

ВИСНОВКИ

За підсумками проведеної роботи можна навести наступні висновки стосовно вирішення поставлених задач:

1) Проведений теоретичний аналіз та систематизація існуючих моделей, методів та підходів використовуваних при розробці технологій прийняття рішень з наданням рекомендацій, дозволити сформулювати методологію дослідження, принципи формування нових підходів до створення моделей рекомендаційних механізмів та їх алгоритмізацію, пов'язану з вирішенням питань нестабільності продуктивності існуючих моделей рекомендаційних систем;

2) Визначено особливості та обґрунтовано підхід щодо структурування процесів рекомендаційної системи, який базується на дослідженні впливу маркерів користувача для визначення сигналів на виході – рекомендацій, з аналізом за критеріями поведінки в мережі Інтернет «переглянув/обрав» та «переглянув/не обрав». Для цього розроблено дві моделі: лінійна за декількома маркерами користувача, при якій створено перехід від вирішення задачі за допомогою теорії ігор та модель для систем, що знаходяться у стаціонарному стані, де рекомендація виступає випадковою подією. В розробці застосований метод Рунге-Кутти, а також застосовано закон розподілу ймовірності при наданні рекомендації користувачу, як випадкової події при Інтернет-серфінгу;

3) Розроблена модель рекомендаційної підтримки з її обґрунтуванням. Формування моделі відбувається за правилами комп'ютерної логіки, а метою є створення та надання такої інформаційної вибірки, яка створена за результатами обробки та аналізу даних з великої кількості джерел. Деталізовано процеси переходів у рекомендаціях з матрицями переходів, для цього використані підходи, які використовуються у синтезі цифрових

автоматів. Розроблено механізм формування тригерів для створення рекомендацій за зміни маркерів користувача;

4) реалізована інформаційна технологія рекомендаційної підтримки прийняття рішень у вигляді веб-додатку. Наведений лістинг кодів програмної реалізації. Отримані результати тестування розробленої технології рекомендаційної підтримки дозволяють зробити висновок відносно наявних переваг отриманої технології з існуючими базовими системами на 0,6 – 5,8%.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Zhang, S.X.; Babovic, V. (2011). An evolutionary real options framework for the design and management of projects and systems with complex real options and exercising conditions. Decision Support Systems. 51 (1): 119–129. doi:10.1016/j.dss.2010.12.001. S2CID 15362734.

2. Francesco R., Lior R., Bracha S., Paul K.B. Recommender Systems Handbook. Dordrecht: Springer, 2015. 1009 p.

3. Jiménez, Antonio; Ríos-Insua, Sixto; Mateos, Alfonso (2006). A generic multi-attribute analysis system. Computers & Operations Research. Elsevier BV. 33 (4): 1081–1101. doi:10.1016/j.cor.2004.09.003

4. Wright, A; Sittig, D (2008). A framework and model for evaluating clinical decision support architectures q. Journal of Biomedical Informatics. 41 (6): 982–990. doi:10.1016/j.jbi.2008.03.009.

5. Aggarwal C.C. Recommender Systems: The Textbook. New York: Springer, 2017. 498 p.

6. Drachsler H., Hummel H.G.K., Koper R. Identifying the Goal, User model and Conditions of Recommender Systems for Formal and Informal Learning. Journal of Digital Information. 2009. Vol. 10, N 2: Social Information Retrieval for Technology Enhanced Learning. 17 p. URL: https://journals.tdl.org/jodi/index.php/jodi/article/download /442/279.

7. Introduction to recommender systems [Електронний ресурс]: https://thingsolver.com/introduction-to-recommender-systems/

8. Чалий С. Доповнення вхідних даних рекомендаційної системи в ситуації циклічного холодного старту з використанням темпоральних обмежень типу «next» / S. Chalyi, V. Leshchynskyi, I. Leshchynska // Системи управління, навігації та зв'язку. Збірник наукових праць. – Полтава: ПНТУ, 2019. – Т. 4 (56). – С. 105-109. – doi:https://doi.org/10.26906/SUNZ.2019.4.105.

9. Чалий С.Ф., Лещинський В.О., Лещинська І.О. Моделювання контексту в рекомендаційних системах. Науковий журнал «Проблеми інформаційних технологій», 2018, №. 1(023). С. 21-26.

10. Hu Y., Koren Y. and Volinsky C. (2008), Collaborative filtering for implicit feedback datasets, Data Mining, ICDM'08. Eighth IEEE International Conference on. IEEE, pp. 263–272.

11. Yehuda Koren, Robert Bell, and Chris Volinsky. (2009), Matrix factorization techniques for recommender systems, Computer No.8, pp. 30–37.

12. Linden G. Amazon.com recommendations: Item-to-item collaborative filltering / G. Linden, B. Smith, J. York // IEEE Internet Computing, vol. 7, no. 1, pp. 76–80, 2003.

13. Xiaoyuan Su and Taghi M. Khoshgoftaar. A Survey of Collaborative Filtering Techniques A Survey of Collaborative Filtering Techniques // Hindawi Publishing Corporation, Advances in Artificial Intelligence archive, USA : 2009. – p. 1-19.

14. Jannach D., Zanker M., Felfernig A. Friedrich G. Recommender Systems. An Introduction. New York: Cambridge University Press 32 Avenue of the Americas, 2011. 352 P.

15. Melville P., Sindhwani V. Recommender systems. Encyclopedia of Machine Learning. 2010. p. 30.

16. Christian Desrosiers and George Karypis. A comprehensive survey of neighborhoodbased recommendation methods. In Recommender systems handbook, pages 107–144. Springer, 2011.

17. Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on, pages 263–272. IEEE, 2008.

18. Cantador I., Konstas I., Jose J. Categorising social tags to improve folksonomy-based recommendations // Web Semantics: Science, Services and Agents on the World Wide Web. – 2011. – Vol. 9, no. 1. — P. 1–15.

19. Daniel Lemire and Anna Maclachlan. Slope one predictors for online rating-based collaborative filtering. In SDM, volume 5, pages 1–5. SIAM, 2005.

20. Istv´an Pil´aszy and Domonkos Tikk. Recommending new movies: even a few ratings are more valuable than metadata. In Proceedings of the third ACM conference on Recommender systems, pages 93–100. ACM, 2009.

21. Linden G., Smith B., and York J. Item-to-Item Collaborative Filtering // IEEE Internet Computing, Los Alamitos, CA USA. — 2003. — P. 76 – 80. https://web.archive.org/web/20150618040813/http://www.cs.umd.edu/~samir/498/Amazon-Recommendations.pdf

22. Sammut C., Webb J. (Eds.). Encyclopedia of Machine Learning. — NY, USA : IBM T. J.Watson Research Center, 2010. — T. 1. — C. 829-838. — ISBN 978-0-387-30768-8.

23. Xiaoyuan Su and Taghi M. Khoshgoftaar. A Survey of Collaborative Filtering Techniques A Survey of Collaborative Filtering Techniques // Hindawi Publishing Corporation, Advances in Artificial Intelligence archive, USA. — 2009. — P. 1 - 19. https://web.archive.org/web/20150320063441/http://downloads.hindawi.com/journals/aai/2009/421425.pdf

24. Fleder D., Hosanagar K. Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity // Management Science, Vol. 55, No. 5, May 2009, pp. 697-712. — 2009. — P. 1 - 49. https://web.archive.org/web/20150322175318/http://papers.ssrn.com/sol3/papers.cfm?abstract_id=955984

25. Ghazanfar, Mustansar Ali; Prügel-Bennett, Adam; Szedmak, Sandor (2012). Kernel-Mapping Recommender system algorithms. Information Sciences. 208: 81–104. doi:10.1016/j.ins.2012.04.012. S2CID 20328670

26. He, Xiangnan; Liao, Lizi; Zhang, Hanwang; Nie, Liqiang; Hu, Xia; Chua, Tat-Seng (2017). Neural Collaborative Filtering. Proceedings of the 26th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee. pp. 173–182. arXiv:1708.05031. doi:10.1145/3038912.3052569

27. Recommender Systems – The Textbook | Charu C. Aggarwal | Springer. Springer. 2016. ISBN 9783319296579.

28. Bi, Xuan; Qu, Annie; Shen, Xiaotong (2018). Multilayer tensor factorization with applications to recommender systems. Annals of Statistics. 46 (6B): 3303–3333. doi:10.1214/17-AOS1659

29. Groh G., Ehmig C. Recommendations in taste related domains: collaborative filtering vs. social filtering // Proceedings of the 2007 international ACM conference on Supporting group work / Citeseer. — 2007. — P. 127–136.

30. Гібридні системи надання рекомендацій та їх реалізація для систем електронного навчання / М.П. Горностай // Пробл. програмув. — 2008. — N 2-3. — С. 453-458. — Бібліогр.: 14 назв. — укр. http://dspace.nbuv.gov.ua/handle/123456789/1446

31. Lytvyn V. Design of a recommendation system based on collaborative filtering and machine learning considering personal needs of the user / V. Lytvyn, V. Vysotska, V. Shatskykh, I. Kohut, O. Petruchenko, L. Dzyubyk, V. Bobrivetc, V. Panasyuk, S. Sachenko, M. Komar // Eastern-European Journal of Enterprise Technologies. - 2019. - № 4(2). - С. 6-28. - Режим доступу: http://nbuv.gov.ua/UJRN/Vejpte_2019_4%282%29__2

32. Su, X., Khoshgoftaar, T. M. (2009). A Survey of Collaborative Filtering Techniques. Advances in Artificial Intelligence, 2009, 1–19. doi: https://doi.org/10.1155/2009/421425

33. Burov, Y., Vysotska, V., Kravets, P. (2019). Ontological approach to plot analysis and modeling. Proceedings of the 3rd International Conference on Computational Linguistics and Intelligent Systems (COLINS-2019). Volume I: Main Conference, 2362, 22–31.

34. Kim Falk (2019), Practical Recommender Systems, Manning Publications, P. 432. ISBN 9781617292705.

35. Ge, M., Delgado-Battenfeld, C., Jannach, D. (2010). Beyond accuracy: Evaluating recommender systems by coverage and serendipity. Proceedings of the

fourth ACM conference on Recommender systems - RecSys ‘10, 257–260. doi: https://doi.org/10.1145/1864708.1864761

36. Bobadilla, J., Ortega, F., Hernando, A., Bernal, J. (2012). A collaborative filtering approach to mitigate the new user cold start problem. Knowledge-Based Systems, 26, 225–238. doi: https://doi.org/10.1016/j.knosys.2011.07.021

37. Schafer J.B. E-Commerce Recommendation Applications / J. B. Schafer J. B., J. A. Konstan, J. Riedl // Data Mining and Knowledge Discovery. – 2001. – Vol. 5. – No. 1–2. – P. 115–123.

38. Candillier L. Comparing State-of-the-Art Collaborative Filtering Systems. / L. Candillier, F. Meyer, M. Boullé. // In Proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition, LNCS. – Vol. 4571. – 2007. – P. 548–562.

39. Dietmar Jannach; Markus Zanker; Alexander Felfernig; Gerhard Friedrich (2010). Recommender Systems:An Introduction. CUP. ISBN 978-0-521-49336-9.

40. Ricci, Francesco; Rokach, Lior; Shapira, Bracha (2022). Recommender Systems: Techniques, Applications, and Challenges. In Ricci, Francesco; Rokach, Lior; Shapira, Bracha (eds.). Recommender Systems Handbook (3 ed.). New York: Springer. pp. 1–35. doi:10.1007/978-1-0716-2197-4_1. ISBN 978-1-0716-2196-7.

41. Su X., Khoshgoftaar T. M. A survey of collaborative filtering techniques / X. Su, T. M. Khoshgoftaar // Adv. Artif. Intell. — Vol. 4571. – 2007 – P. 1–19.

42. Isinkaye F. O. Recommendation systems: Principles, methods and evaluation / F. O. Isinkaye F. O., Y. O. Folajimi, B. A. Ojokoh // Egyptian Informatics Journal. – Vol. 16. – 2015. – P. 261–273.

43. Baran, Remigiusz; Dziech, Andrzej; Zeja, Andrzej (2018). "A capable multimedia content discovery platform based on visual content analysis and intelligent data enrichment". Multimedia Tools and Applications. 77 (11): 14077–14091. doi:10.1007/s11042-017-5014-1. ISSN 1573-7721.

44. Resnick P., Varian H. R. Recommender systems / P. Resnick, H. R. Varian // Communications of the ACM. – Vol. 40. – 1997. – P. 56–58.

45. L. Boratto, S. Carta, “State-ofthe-art in group recommendation and new approaches for automatic identification of groups,” / L. Boratto, S. Carta // In Information Retrieval and Mining in Distributed Environments. – vol. 324. – Springer Berlin Heidelberg – 2011. – P. 1–20.

46. Guha S. Rock: A robust clustering algorithm for categorical attributes / S. Guha, R. Rastogi, K. Shim // Information Systems. – vol. 25, No. 5. – 2000. – P. 345–366.

47. ChenHung-Hsuan; ChenPu (2019). "Differentiating Regularization Weights -- A Simple Mechanism to Alleviate Cold Start in Recommender Systems". ACM Transactions on Knowledge Discovery from Data. 13: 1–22. doi:10.1145/3285954. S2CID 59337456.

48. Rubens, Neil; Elahi, Mehdi; Sugiyama, Masashi; Kaplan, Dain (2016). "Active Learning in Recommender Systems". In Ricci, Francesco; Rokach, Lior; Shapira, Bracha (eds.). Recommender Systems Handbook (2 ed.). Springer US. pp. 809–846. doi:10.1007/978-1-4899-7637-6_24. ISBN 978-1-4899-7637-6.

49. Bobadilla, J.; Ortega, F.; Hernando, A.; Alcalá, J. (2011). "Improving collaborative filtering recommender system results and performance using genetic algorithms". Knowledge-Based Systems. 24 (8): 1310–1316. doi:10.1016/j.knosys.2011.06.005.

50. Lytvyn, V., Vysotska, V., Rusyn, B., Pohreliuk, L., Berezin, P., Naum, O. (2019). Textual Content Categorizing Technology Development Based on Ontology. Workshop Proceedings of the 8th International Conference on “Mathematics. Information Technologies. Education”, 2386, 234–254.

51. Elahi, Mehdi; Ricci, Francesco; Rubens, Neil (2016). A survey of active learning in collaborative filtering recommender systems. Computer Science Review. 20: 29–50. doi:10.1016/j.cosrev.2016.05.002.

52. Nathan Srebro, Tommi Jaakkola, et al. Weighted low-rank approximations. In ICML, volume 3, pages 720–727, 2003.

53. G. Adomavicius, A. Tuzhilin Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions / Adomavicius G.,

Tuzhilin A. // IEEE Transactions on Knowledge and Data Engineerin. – Vol. 17. – 2005. – P. 734–749.

54. Matrix factorization and neighbor based algorithms for the netflix prize problem / G. Tak´acs, I. Pil´aszy, B. N´emeth, D. Tikk // Proceedings of the 2008 ACM conference on Recommender systems / ACM. — 2008. — P. 267–274.

55. Beel, J.; Genzmehr, M.; Gipp, B. (2013). A comparative analysis of offline and online evaluations and discussion of research paper recommender system evaluation. Proceedings of the International Workshop on Reproducibility and Replication in Recommender Systems Evaluation. pp. 7–14. doi:10.1145/2532508.2532511. ISBN 978-1-4503-2465-6.

56. Rokach, L. (2010). Ensemble-based classifiers. Artificial Intelligence Review. 33 (1–2): 1–39. doi:10.1007/s10462-009-9124-7

57. Ding, Jie; Tarokh, Vahid; Yang, Yuhong (2018). Model Selection Techniques: An Overview. IEEE Signal Processing Magazine. 35 (6): 16–34. doi:10.1109/MSP.2018.2867638

58. Barton M., Lennox B. (2022). Model stacking to improve prediction and variable importance robustness for soft sensor development. Digital Chemical Engineering. Vol. 3. ISSN 2772-5081. https://doi.org/10.1016/j.dche.2022.100034.

59. Okazaki K., Katsumi I. (2022). Explainable Model Fusion for Customer Journey Mapping. Frontiers in Artificial Intelligence. 5. 10.3389/frai.2022.824197.

60. Rubens, N., Elahi, M., Sugiyama, M., Kaplan, D. (2015). Active Learning in Recommender Systems. Recommender Systems Hand-book, 809–846. doi: https://doi.org/10.1007/978-1-4899-7637-6_24

61. Trofymchuk O. M., Bidyuk P.I. Decision support systems, modeling, forecasting, risk estimation. — LAP LAMBERT Academic Publishing — 2019. — 179 p.

62. Довгий С. О., Бідюк П. І., Трофимчук О. М., Савенков О. І. Методи прогнозування в системах підтримки прийняття рішень. — К.: Азимут-Україна. — 2011. — 608 с.

63. Довгий С. О., Бідюк П. І., Трофимчук О. М. Системи підтримки прийняття рішень на основі статистично-ймовірнісних методів. — К.: Логос. — 2014. — 419 с.

64. Морозов А. А.. Косолапов В. Л. Информационно-аналитические технологии поддержки принятия решений на основе регионального социально-экономического мониторинга. — Киев: Наукова думка, 2002. — 230 с.

65. Sprague R. (1986). Decision support systems : putting theory into practice. Englewood Cliffs, N.J: Prentice-Hall. ISBN 978-0-13-197286-5.

66. Power D. J. Web-based and model-driven decision support systems: concepts and issues. Americas Conference on Information Systems, Long Beach, California, 2000.

67. Zhang, S.X.; Babovic, V. (2011). An evolutionary real options framework for the design and management of projects and systems with complex real options and exercising conditions. Decision Support Systems. 51 (1): 119–129. doi:10.1016/j.dss.2010.12.001

68. Russell S. J.; Norvig P. (2003). Artificial Intelligence: A Modern Approach. 2nd. Upper Saddle River, New Jersey: Prentice Hall. ISBN 0-13-790395-2.

69. Murphy K. P. (2022). Probabilistic Machine Learning: An Introduction. The MIT Press. 944 P. ISBN: 978-0262046824.

70. Koren Y.; Bell R.; Volinsky C. (2009). Matrix Factorization Techniques for Recommender Systems..Computer (IEEE) 42 (8): 30–37.

71. Bhasker B.; Srikumar K. (2010). Recommender Systems in E-Commerce. CUP. ISBN 978-0-07-068067-8.

72. Barwise J.; Etchemendy J; Allwein G.; Barker-Plummer D.; Liu A. (1999). Language, proof, and logic. CSLI Publications. ISBN 978-1-889119-08-3.

73. Minato Sh., Muroga S. (2007). Binary Decision Diagrams. У Wai-Kai Chen. The VLSI handbook (2nd). CRC Press. ISBN 978-0-8493-4199-1.

74. Енциклопедія кібернетики : у 2 т. / за ред. В. М. Глушкова. — Київ : Гол. ред. Української радянської енциклопедії, 1973.

75. Ms. Ashwini A. Chirde, Ms. Urmila K. (2015). Combination of a Cluster-Based and Content-Based Collaborative Filtering Approach for Recommender System. International Journal on Recent and Innovation Trends in Computing and Communication, 3 (7), 4770–4774.

76. Купрін О.М. Алгоритмізація процесів у рекомендаційних системах / Математичні машини і системи. – 2022. – № 1. – сс. 71 – 80. ISSN 1028-9763.

77. Купрін О.М. Структуризація алгоритмів рекомендаційних систем, що використовуються у фінансово-кредитній сфері / Інформаційно-комунікаційні технології та сталий розвиток // Колективна монографія за матеріалами XXI Міжнародної науково-практичної конференції (Київ, 14-16 листопада 2022 р.) / За заг. ред. С.О. Довгого. – К.: ТОВ «Видавництво «Юстон», 2022. – сс. 195 – 196. ISBN 978-617-7854-76-9.

78. Del Giorgio S.F. (2017). Public Benchmarking: contributions for subnational governments and Benchmarking Design. Villa Elisa: FDGS, p. 5. ISBN 978-987-42-6026-0. doi:10.13140/RG.2.2.36285.10722

79. Narver, J.C.; Slater, S.F. (1990). The Effect of a Market Orientation on Business Profitability. Journal of Marketing. 54 (4): 20–34. doi:10.2307/1251757

80. Aspara J.; Grant D. B.; Holmlund M. (2021). Consumer involvement in supply networks: A cubic typology of C2B2C and C2B2B business models. Industrial Marketing Management. 93: 356–369. doi:10.1016/j.indmarman.2020.09.004. ISSN 0019-8501

81. Новітні тенденції розвитку управління підприємствами: монографія / [Федонін О.С., Швиданенко Г.О., Лаврененко В.В. та ін.]. – К.:КНЕУ, 2011. – 257 с.

82. Aggarwal C. C. Recommender Systems: The Textbook. New York: Springer, 2017. 498 p.

83. Michael H. Mescon, Michael Albert, Franklin Khedouri. Management: Individual and Organizational Effectiveness. Harper & Row, 1985 – 756 P.

84. Russell L. Ackoff. Management Misinformation Systems. In: Management Science, 14(4), 1967, 147–156.

85. Дослідження операцій. Ч. 3. Ухвалення рішень і теорія ігор / М. Я. Бартіш, І. М. Дудзяний. – Львів: Видавничий центр Львівського національного університету ім. І.Франка, 2009 . – 277 с. : іл. – Бібліогр.: с.271-272 (36 назв). – ISBN 966-613-496-9.

86. Разумова М.А., Хотяїнцев В. М. Основи векторного і тензорного аналізу: навчальний посібник. – Київ: ВПЦ «Київський університет», 2011.-216с.

87. Kolda T.; Bader B. (2009). Tensor Decompositions and Applications. SIAM Review. 51 (3): 455–500. doi:10.1137/07070111X

88. Hope T.; Resheff Y. S.; Lieder I. (2017). Learning TensorFlow: A Guide to Building Deep Learning Systems (1st ed.). O'Reilly Media. p. 242. ISBN 9781491978504.

89. Івченко І.Ю. Математичне програмування : Навчальний посібник. - К.: Центр учбової літератури,2007-232с.

90. William H. Press, Brian P. Flannery, Saul A. Teukolsky, William T. Vetterling. Numerical Recipes in C. Cambridge, UK: Cambridge University Press, 1988.

91. Dormand, J. R.; Prince, P. J. (1978). New Runge–Kutta Algorithms for Numerical Simulation in Dynamical Astronomy. Celestial Mechanics. **18** (3): 223–232. Bibcode:1978CeMec..18..223D. doi:10.1007/BF01230162

92. Delin T.; Zheng C. (2012). On A General Formula of Fourth Order Runge-Kutta Method. Journal of Mathematical Science & Mathematics Education, **7** (2): 1–10.

93. Gagniuc P.A. (2017). Markov Chains: From Theory to Implementation and Experimentation. USA, NJ: John Wiley & Sons. pp. 1–235. ISBN 978-1-119-38755-8.

94. Малицька Г.П.Системи рівнянь типу Колмогорова . Український математичний журнал. — 2008. — Т. 60, № 12. — С. 1650–1663. — Бібліогр.: 9 назв. — укр. http://dspace.nbuv.gov.ua/handle/123456789/164795

95. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing

Directive 95/46/EC. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:02016R0679-20160504

96. Кряжич О.О., Ющенко К.С., Іцкович В.Є., Купрін О.М. Особливості алгоритмізації процесів мінімізації похибок апроксимації при вирішенні прикладних задач**.** Математичні машини і системи. – 2023 - №1 – сс. 118 – 129.

97. Купрін О.М. Надання рекомендацій користувачу онлайн-сервісу на підставі обробки даних як базис безпаперової інформатики В. Глушкова // Історія, сучасний стан та тенденції цифрового розвитку суспільства. Матеріали 10-ої Міжнар. наук.-практ. конф. «Глушковські читання», Київ, 2021 р. / Уклад.: Р.М. Богачев, В.Д. Піхорович, А.Ю. Самарський, М.І. Сторожик. – Київ, 2021. – с. 102 – 104.

98. Нікольський, Ю.В. Дискретна математика: підручник [Текст] / Ю.В.Нікольський, В.В. Пасічник, Ю.М. Щербина. – Львів: Магнолія 2006, 2007. – 608с.

99. Логическое проектирование дискретных устройств / Глушков В.М., Капитонова Ю.В., Мищенко А.Т. – Киев: Наук. Думка, 1987. – 264 с.

100. Довгий С.О. Алгоритми методу дискретних особливостей для обчислювальних технологій / С.О. Довгий, С.І. Ляшко, Д.І. Черній // Кибернетика и системный анализ. — 2017. — Т. 53, № 6. — С. 147–159. — Бібліогр.: 10 назв. — укр. http://dspace.nbuv.gov.ua/handle/123456789/144816

101. Глушков В.М. Синтез цифровых автоматов. − М.: Физматгиз, 1962. − 476 с., ил.

102. Гуляєв, К. Д., Ющенко, К. С., Купрін, О. М. (2023). Застосування абстрактних автоматів Мілі та Мура для реалізації алгоритмів рекомендації та вибору. International Scientific Technical Journal "Problems of Control and Informatics", 67(6), с. 14–24. doi: https://doi.org/10.34229/1028-0979-2022-6-2.

103. John C. Martin (2011). Introduction to Languages and The Theory of Computation. New York: McGraw Hill. ISBN 978-0-07-319146-1.

104. Aufenkamp, D. D.; Kohn, F. S.: Analysis of sequential machines. IRE Trans. Electr. Comp. EC-6 (1957), pp. 276–285.

105. Holder M. E. (2005). A modified Karnaugh map technique. IEEE Transactions on Education. IEEE. 48 (1): 206–207. doi:10.1109/TE.2004.832879

106. Copi I. M.; Cohen C.; McMahon K. (2016). Introduction to Logic. doi:10.4324/9781315510897

107. Kryazhych, O., Itskovych, V., Iushchenko, K., Kuprin, O. (2023). Features in solving individual tasks to develop service-oriented networks using dynamic programming. Eastern-European Journal of Enterprise Technologies, 1 (4 (121)), 34–40. doi: https://doi.org/10.15587/1729-4061.2023.

108. Кряжич О.О., Коваленко О.В., Купрін О.М. Використання механізму рекомендацій при вдосконаленні інструментів у комп'ютерній програмі «Випадкова точка»// Інформаційно-комунікаційні технології для перемоги та відновлення / Колективна монографія за матеріалами XXII Міжнародної науково-практичної конференції «Інформаційно-комунікаційні технології та сталий розвиток» (Київ, 14-15 листопада 2023 р.) / За заг. ред. С.О. Довгого. – К.: ТОВ «Видавництво «Юстон», 2023. – сс. 89 – 90. ISBN 978-617-8335-06-9.

109. Кряжич О.О., Купрін О.М. Процес створення рекомендаційного алгоритму. XII Наукова конференція «Наукові підсумки 2023 року». Збірка наукових праць. – Харків, Х.: Технологічний Центр, 2023. – 98 С. (с. 67). e-ISBN 978-617-8360-00-9.

ДОДАТКИ

Додаток А

Документи, що підтверджують впровадження результатів дисертації

«ЗАТВЕРДЖУЮ»

Директор ТОВ «Евергрін Ентерпрайз»

_____Кравцов С.В.

«14» грудня 2023 р.

АКТ

про впровадження результатів дисертаційної роботи

Купріна Олексія Миколайовича

за темою «Інформаційна технологія рекомендаційної підтримки прийняття рішень», представленої на здобуття наукового ступеня доктора філософії

Цим актом підтверджується, що результати досліджень, які проводилися в межах роботи «Інформаційна технологія рекомендаційної підтримки прийняття рішень», впроваджені в роботу ТОВ «Евергрін Інтерпрайз», що займається розробкою програмного забезпечення, управління комп'ютерними системами та обробкою інформації.

Отримані в дисертаційній роботі нові науково-технічні та практичні результати були використані:

1) підхід з використанням методу Рунге-Кутти для розробки алгоритму роботи з підказками складних керуючих систем з механізмом адаптації під знання та вміння користувача;

2) лінійну модель рекомендацій, створену за декількома маркерами споживача, апробовано, та частково використано для розробки торгівельної платформи для компанії-замовника;

3) підхід з використанням комп'ютерної логіки для відбору вхідних параметрів за уподобаннями користувача, а також алгоритм, розроблений за цим підходом, використовується для автоматизації внутрішніх процесів стосовно збору інформації про очікування замовників щодо продуктів компанії.

Директор
ТОВ Евергін Ентерпрайз
Кравцов Сергій Володимирович

Кравцов С.В.

АКТ

впровадження результатів досліджень дисертаційної роботи

Купріна Олексія Миколайовича

на тему «Інформаційна технологія рекомендаційної підтримки прийняття рішень»

Акт підтверджує використання матеріалів, розроблених при виконанні роботи «Інформаційна технологія рекомендаційної підтримки прийняття рішень», виконаної на здобуття наукового ступеня доктора філософії Купріним О.М., аспірантом Інституту телекомунікацій і глобального інформаційного простору НАН України (м. Київ), при розробці проекту нового сайту видавництва ФОП Ретівов Тетяна «Каяла» ТМ (код 1997823247, свідоцтво суб'єкта видавничої справи: ДК №5016 від 24.11.2015 р.). На сайті передбачено створення рекомендаційного механізму з позицій, які радяться до перегляду на основі тих маркерів, що визначені при перегляді попередніх книжок користувачем, а також рекомендацій книг, близькими за тематикою до тих, що були придбані користувачем через сайт видавництва раніше.

Керівник: Тетяна Ретівов

ПрАТ «Українсько-Польський вищий навчальний заклад»
«Центрально-Європейський університет»

Юридична адреса: 03680, Київ, вул. Козацька 120/4
Адреса знаходження: 01030, Київ, вул. Рейтарська 19а
E-mail: referent.ceu.info@gmail.com;
www.c-e-u.info
Ліцензія МОН УКРАЇНИ
IV рівень акредитації.

Тел.: (044) 278-24-73
Факс: (044) 278-18-92

АКТ

впровадження результатів досліджень,
отриманих при виконанні дисертаційної роботи
Купріна Олексія Миколайовича
на здобуття наукового ступеня доктора філософії

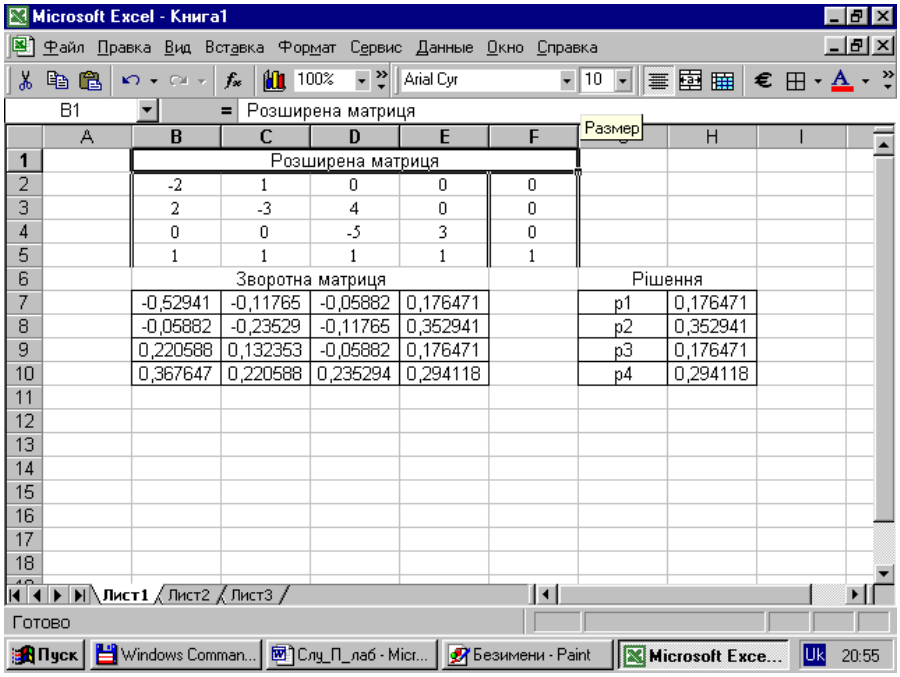
Даний Акт підтверджує впровадження результатів дисертаційної роботи аспіранта Інституту телекомунікацій та глобального інформаційного простору НАН України Купріна О.М. на тему «Інформаційна технологія рекомендаційної підтримки прийняття рішень», у навчальний процес Приватного акціонерного товариства «Українсько-Польський вищий навчальний заклад «Центрально-Європейський університет» для підготовки здобувачів рівня «Бакалавр» за спеціальністю «Маркетинг. Фінанси» з дисциплін «Інтернет маркетинг» та «Вебзастосунки у комерційній діяльності».

Перший проректор Шепелюк О. Ю.

Зав. навчальної частини Лапуста Т. О.

Додаток Б

Розв’язок задачі



	A	B	C	D	E	F	G	H	I
1		Розширена матриця							
2		-2	1	0	0	0			
3		2	-3	4	0	0			
4		0	0	-5	3	0			
5		1	1	1	1	1			
6		Зворотна матриця						Рішення	
7		-0,52941	-0,11765	-0,05882	0,176471		p1	0,176471	
8		-0,05882	-0,23529	-0,11765	0,352941		p2	0,352941	
9		0,220588	0,132353	-0,05882	0,176471		p3	0,176471	
10		0,367647	0,220588	0,235294	0,294118		p4	0,294118	
11									
12									
13									
14									
15									
16									
17									
18									

Рисунок Б1 – Робочій аркуш вирішення задачі

```
Global A(1 To 100), F(1 To 100), X(1 To 100)
Sub Runge()
Dim K1(1 To 100), K2(1 To 100), K3(1 To 100), K4(1 To 100)
m = 100
n = 4
For i = 1 To n
   A(i) = Cells(11, i + 1)
   'MsgBox A(i)
Next i
t = Cells(12, 1)
h = t / m
For i = 1 To m
   For j = 1 To n
      X(j) = A(j)
   Next j
   funct
   For j = 1 To n
     K1(j) = F(j) * h
     X(j) = A(j) + K1(j) / 2
   Next j
   funct
   For j = 1 To n
     K2(j) = F(j) * h
     X(j) = A(j) + K2(j) / 2
    Next j
    funct
    For j = 1 To n
    K3(j) = F(j) * h
     X(j) = A(j) + K3(j)
    Next j
    funct
    For j = 1 To n
     K4(j) = F(j) * h
     A(j) = A(j) + (K1(j) + 2 * (K2(j) + K3(j)) + K4(j)) / 6
    Next j
 Next i
 For i = 1 To n
  Cells(12, i + 1) = A(i)
 Next i
End Sub
```

Рисунок Б2 – Варіант лістинга програми чисельного рішення диференціальних рівнянь методом Рунге-Кута

```
Sub funct()
'
F(1) = -2 * X(1) + X(2)
F(2) = 2 * X(1) - 3 * X(2) + 4 * X(3)
F(3) = -5 * X(3) + 3 * X(4)
F(4) = 2 * X(2) + X(3) - 3 * X(4)

End Sub
```

Рисунок Б3 – Лістинг підпрограми розрахунків функцій

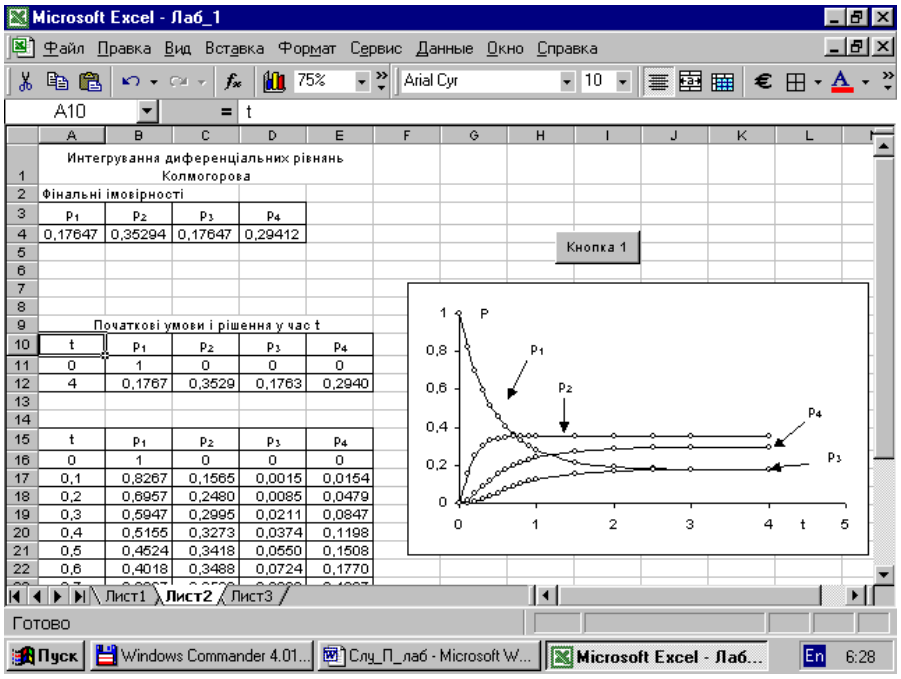


Рисунок Б4 – Робочий лист Excel, зв’язаний з програмою для проведення тестування запропонованого методу

Додаток В

Лістинг програми аналізу вибору користувача як випадкової події

```
Option Explicit
Global N_c, PER(1 To 7), N_Yes, N_No As Integer
Global N_Barb, N_Per, tcl, tser As Integer

Dim i, j, k, m, N_s, NW As Integer

Sub Barber()
Randomize
NW = Cells(7, 2)
N_Per = Cells(3, 2)
N_Barb = Cells(4, 2)
tcl = Cells(6, 2)
tser = Cells(5, 2)
'***Початок випробувань
For i = 1 To NW
    N_No = 0
    N_Yes = 1
    N_c = Time_Cli()
    PER(1) = Time_Ser()
    For j = 2 To N_Per
        PER(j) = 0
    Next j
'MsgBox n_c & "   " & PER(1)
    For j = 1 To 6000
'**Випадок n_c=0
    If N_c = 0 Then

        N_c = Time_Cli
        N_s = 0
        For k = 1 To N_Per
            If PER(k) > 0 Then N_s = N_s + 1
        Next k
        If N_s = N_Per Then
        N_No = N_No + 1
        End If
        If N_s < N_Per Then
            N_Yes = N_Yes + 1
'*** чи є пусті місця у перукарів
            If N_s < N_Barb Then

                For k = 1 To N_Barb
                    If PER(k) = 0 Then
                        PER(k) = Time_Ser()
                        Exit For
                    End If
                Next k
```

```
            End If ' Кінець N_s < N_Barb
            If N_s >= N_Barb Then

                For k = N_Barb + 1 To N_Per
                   If PER(k) = 0 Then
                       PER(k) = 1
                       Exit For
                   End If
                Next k
            End If '**кінець N_s > N_Barb
        End If ' ***Кінець N_s < N_Per
    End If ' ***Кінець n_c=0
'**** Положення у такті i
       N_c = N_c - 1
       For k = 1 To N_Barb
          If PER(k) > 0 Then PER(k) = PER(k) - 1
       Next k
       'm = PER(1)
       For k = 1 To N_Barb
            If PER(k) = 0 Then
               For m = N_Barb + 1 To N_Per
                  If PER(m) = 1 Then
                      PER(m) = 0
                      PER(k) = Time_Ser
                      Exit For
                  End If
               Next m
            End If
       Next k
    Next j
    Cells(i + 3, 4) = i
    Cells(i + 3, 5) = N_Yes
    Cells(i + 3, 6) = N_No
      'For k = 1 To N_Per
           ' Cells(1, 3 + k) = PER(k)
       ' Next k
 Next i
'
End Sub
Function Time_Cli()
Time_Cli = Int(-tcl * Log(Rnd) + 1)
End Function
Function Time_Ser()
Time_Ser = Int(-tser * Log(Rnd) + 1)
End Function
```

Додаток Г

Основні фрагменти лістингу коду програмної реалізації рекомендаційної системи

Задавання основних параметрів системи та маркерів користувача:


```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 1,
   "metadata": {
    "id": "Y07rX5qojyDG"
   },
   "outputs": [],
   "source": [
    "import pandas as pd"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 2,
   "metadata": {
    "id": "Zv-d_MCYj_HD"
   },
   "outputs": [
    {
     "data": {
      "text/html": [
       "<div>\n",
       "<style scoped>\n",
       "    .dataframe tbody tr th:only-of-type {\n",
       "        vertical-align: middle;\n",
       "    }\n",
       "\n",
       "    .dataframe tbody tr th {\n",
       "        vertical-align: top;\n",
       "    }\n",
       "\n",
       "    .dataframe thead th {\n",
       "        text-align: right;\n",
       "    }\n",
       "</style>\n",
       "<table border=\"1\" class=\"dataframe\">\n",
       "  <thead>\n",
       "    <tr style=\"text-align: right;\">\n",
       "      <th></th>\n",
       "      <th>acousticness</th>\n",
```

```
        "      <th>artists</th>\n",
        "      <th>danceability</th>\n",
        "      <th>duration_ms</th>\n",
        "      <th>energy</th>\n",
        "      <th>explicit</th>\n",
        "      <th>id</th>\n",
        "      <th>instrumentalness</th>\n",
        "      <th>key</th>\n",
        "      <th>liveness</th>\n",
        "      <th>loudness</th>\n",
        "      <th>mode</th>\n",
        "      <th>name</th>\n",
        "      <th>popularity</th>\n",
        "      <th>release_date</th>\n",
        "      <th>speechiness</th>\n",
        "      <th>tempo</th>\n",
        "      <th>valence</th>\n",
        "      <th>year</th>\n",
        "    </tr>\n",
        "  </thead>\n",
        "  <tbody>\n",
        "    <tr>\n",
        "      <th>0</th>\n",
        "      <td>0.9950</td>\n",
        "      <td>['Carl Woitschach']</td>\n",
        "      <td>0.708</td>\n",
        "      <td>158648</td>\n",
        "      <td>0.1950</td>\n",
        "      <td>0</td>\n",
        "      <td>6KbQ3uYMLKb5jDxLF7wYDD</td>\n",
        "      <td>0.563000</td>\n",
        "      <td>10</td>\n",
        "      <td>0.1510</td>\n",
        "      <td>-12.428</td>\n",
        "      <td>1</td>\n",
        "      <td>Singende Bataillone 1. Teil</td>\n",
        "      <td>0</td>\n",
        "      <td>1928</td>\n",
        "      <td>0.0506</td>\n",
        "      <td>118.469</td>\n",
        "      <td>0.7790</td>\n",
        "      <td>1928</td>\n",
        "    </tr>\n",
        "    <tr>\n",
        "      <th>1</th>\n",
        "      <td>0.9940</td>\n",
        "      <td>['Robert Schumann', 'Vladimir
Horowitz']</td>\n",
        "      <td>0.379</td>\n",
        "      <td>282133</td>\n",
        "      <td>0.0135</td>\n",
        "      <td>0</td>\n",
        "      <td>6KuQTIu1KoTTkLXKrwlLPV</td>\n",
```

```
        "      <td>0.901000</td>\n",
        "      <td>8</td>\n",
        "      <td>0.0763</td>\n",
        "      <td>-28.454</td>\n",
        "      <td>1</td>\n",
        "      <td>Fantasiestücke, Op. 111: Più tosto
lento</td>\n",
        "      <td>0</td>\n",
        "      <td>1928</td>\n",
        "      <td>0.0462</td>\n",
        "      <td>83.972</td>\n",
        "      <td>0.0767</td>\n",
        "      <td>1928</td>\n",
        "    </tr>\n",
        "    <tr>\n",
        "      <th>2</th>\n",
        "      <td>0.6040</td>\n",
        "      <td>['Seweryn Goszczyński']</td>\n",
        "      <td>0.749</td>\n",
        "      <td>104300</td>\n",
        "      <td>0.2200</td>\n",
        "      <td>0</td>\n",
        "      <td>6L63VW0PibdM1HDSBoqnoM</td>\n",
        "      <td>0.000000</td>\n",
        "      <td>5</td>\n",
        "      <td>0.1190</td>\n",
        "      <td>-19.924</td>\n",
        "      <td>0</td>\n",
        "      <td>Chapter 1.18 - Zamek kaniowski</td>\n",
        "      <td>0</td>\n",
        "      <td>1928</td>\n",
        "      <td>0.9290</td>\n",
        "      <td>107.177</td>\n",
        "      <td>0.8800</td>\n",
        "      <td>1928</td>\n",
        "    </tr>\n",
        "    <tr>\n",
        "      <th>3</th>\n",
        "      <td>0.9950</td>\n",
        "      <td>['Francisco Canaro']</td>\n",
        "      <td>0.781</td>\n",
        "      <td>180760</td>\n",
        "      <td>0.1300</td>\n",
        "      <td>0</td>\n",
        "      <td>6M94FkXd15sOAOQYRnWPN8</td>\n",
        "      <td>0.887000</td>\n",
        "      <td>1</td>\n",
        "      <td>0.1110</td>\n",
        "      <td>-14.734</td>\n",
        "      <td>0</td>\n",
        "      <td>Bebamos Juntos - Instrumental
(Remasterizado)</td>\n",
        "      <td>0</td>\n",
```

```
"      <td>1928-09-25</td>\n",
"      <td>0.0926</td>\n",
"      <td>108.003</td>\n",
"      <td>0.7200</td>\n",
"      <td>1928</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>4</th>\n",
"      <td>0.9900</td>\n",
"      <td>['Frédéric Chopin', 'Vladimir
Horowitz']</td>\n",
"      <td>0.210</td>\n",
"      <td>687733</td>\n",
"      <td>0.2040</td>\n",
"      <td>0</td>\n",
"      <td>6N6tiFZ9vLTSOIxkj8qKrd</td>\n",
"      <td>0.908000</td>\n",
"      <td>11</td>\n",
"      <td>0.0980</td>\n",
"      <td>-16.829</td>\n",
"      <td>1</td>\n",
"      <td>Polonaise-Fantaisie in A-Flat Major, Op.
61</td>\n",
"      <td>1</td>\n",
"      <td>1928</td>\n",
"      <td>0.0424</td>\n",
"      <td>62.149</td>\n",
"      <td>0.0693</td>\n",
"      <td>1928</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>...</th>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"      <td>...</td>\n",
"    </tr>\n",
"    <tr>\n",
```

```
"      <th>169904</th>\n",
"      <td>0.1730</td>\n",
"      <td>['DripReport', 'Tyga']</td>\n",
"      <td>0.875</td>\n",
"      <td>163800</td>\n",
"      <td>0.4430</td>\n",
"      <td>1</td>\n",
"      <td>4KppkflX7I3vJQk7urOJaS</td>\n",
"      <td>0.000032</td>\n",
"      <td>1</td>\n",
"      <td>0.0891</td>\n",
"      <td>-7.461</td>\n",
"      <td>1</td>\n",
"      <td>Skechers (feat. Tyga) - Remix</td>\n",
"      <td>75</td>\n",
"      <td>2020-05-15</td>\n",
"      <td>0.1430</td>\n",
"      <td>100.012</td>\n",
"      <td>0.3060</td>\n",
"      <td>2020</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>169905</th>\n",
"      <td>0.0167</td>\n",
"      <td>['Leon Bridges', 'Terrace Martin']</td>\n",
"      <td>0.719</td>\n",
"      <td>167468</td>\n",
"      <td>0.3850</td>\n",
"      <td>0</td>\n",
"      <td>1ehhGlTvjtHo2e4xJFB0SZ</td>\n",
"      <td>0.031300</td>\n",
"      <td>8</td>\n",
"      <td>0.1110</td>\n",
"      <td>-10.907</td>\n",
"      <td>1</td>\n",
"      <td>Sweeter (feat. Terrace Martin)</td>\n",
"      <td>64</td>\n",
"      <td>2020-06-08</td>\n",
"      <td>0.0403</td>\n",
"      <td>128.000</td>\n",
"      <td>0.2700</td>\n",
"      <td>2020</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>169906</th>\n",
"      <td>0.5380</td>\n",
"      <td>['Kygo', 'Oh Wonder']</td>\n",
"      <td>0.514</td>\n",
"      <td>180700</td>\n",
"      <td>0.5390</td>\n",
"      <td>0</td>\n",
"      <td>52eycxprLhK3lPcRLbQiVk</td>\n",
"      <td>0.002330</td>\n",
```

```
"      <td>7</td>\n",
"      <td>0.1080</td>\n",
"      <td>-9.332</td>\n",
"      <td>1</td>\n",
"      <td>How Would I Know</td>\n",
"      <td>70</td>\n",
"      <td>2020-05-29</td>\n",
"      <td>0.1050</td>\n",
"      <td>123.700</td>\n",
"      <td>0.1530</td>\n",
"      <td>2020</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>169907</th>\n",
"      <td>0.0714</td>\n",
"      <td>['Cash Cash', 'Andy Grammer']</td>\n",
"      <td>0.646</td>\n",
"      <td>167308</td>\n",
"      <td>0.7610</td>\n",
"      <td>0</td>\n",
"      <td>3wYOGJYD31sLRmBgCvWxa4</td>\n",
"      <td>0.000000</td>\n",
"      <td>1</td>\n",
"      <td>0.2220</td>\n",
"      <td>-2.557</td>\n",
"      <td>1</td>\n",
"      <td>I Found You</td>\n",
"      <td>70</td>\n",
"      <td>2020-02-28</td>\n",
"      <td>0.0385</td>\n",
"      <td>129.916</td>\n",
"      <td>0.4720</td>\n",
"      <td>2020</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>169908</th>\n",
"      <td>0.1090</td>\n",
"      <td>['Ingrid Andress']</td>\n",
"      <td>0.512</td>\n",
"      <td>214787</td>\n",
"      <td>0.4280</td>\n",
"      <td>0</td>\n",
"      <td>60RFlt48hm0l4Fu0JoccOl</td>\n",
"      <td>0.000000</td>\n",
"      <td>0</td>\n",
"      <td>0.1050</td>\n",
"      <td>-7.387</td>\n",
"      <td>1</td>\n",
"      <td>More Hearts Than Mine</td>\n",
"      <td>65</td>\n",
"      <td>2020-03-27</td>\n",
"      <td>0.0271</td>\n",
"      <td>80.588</td>\n",
```

```
       "      <td>0.3660</td>\n",
       "      <td>2020</td>\n",
       "    </tr>\n",
       "  </tbody>\n",
       "</table>\n",
       "<p>169909 rows × 19 columns</p>\n",
       "</div>"
      ],
      "text/plain": [
       "        acousticness
artists  danceability  \\\n",
       "0             0.9950                          ['Carl
Woitschach']         0.708   \n",
       "1             0.9940  ['Robert Schumann', 'Vladimir
Horowitz']         0.379   \n",
       "2             0.6040                    ['Seweryn
Goszczyński']         0.749   \n",
       "3             0.9950                       ['Francisco
Canaro']         0.781   \n",
       "4             0.9900  ['Frédéric Chopin', 'Vladimir
Horowitz']         0.210   \n",
       "...              ...
...           ...   \n",
       "169904        0.1730                    ['DripReport',
'Tyga']         0.875   \n",
       "169905        0.0167        ['Leon Bridges', 'Terrace
Martin']         0.719   \n",
       "169906        0.5380                      ['Kygo', 'Oh
Wonder']         0.514   \n",
       "169907        0.0714             ['Cash Cash', 'Andy
Grammer']         0.646   \n",
       "169908        0.1090                         ['Ingrid
Andress']         0.512   \n",
       "\n",
       "        duration_ms  energy  explicit
id  \\\n",
       "0            158648  0.1950         0
6KbQ3uYMLKb5jDxLF7wYDD   \n",
       "1            282133  0.0135         0
6KuQTIu1KoTTkLXKrwlLPV   \n",
       "2            104300  0.2200         0
6L63VW0PibdM1HDSBoqnoM   \n",
       "3            180760  0.1300         0
6M94FkXd15sOAOQYRnWPN8   \n",
       "4            687733  0.2040         0
6N6tiFZ9vLTSOIxkj8qKrd   \n",
       "...             ...     ...       ...
...   \n",
       "169904       163800  0.4430         1
4KppkflX7I3vJQk7urOJaS   \n",
       "169905       167468  0.3850         0
1ehhGlTvjtHo2e4xJFB0SZ   \n",
```

```
       "169906       180700  0.5390         0
52eycxprLhK3lPcRLbQiVk   \n",
       "169907       167308  0.7610         0
3wYOGJYD31sLRmBgCvWxa4   \n",
       "169908       214787  0.4280         0
60RFlt48hm0l4Fu0JoccOl   \n",
       "\n",
       "        instrumentalness  key  liveness  loudness  mode
\\\n",
       "0               0.563000   10    0.1510   -12.428     1
\n",
       "1               0.901000    8    0.0763   -28.454     1
\n",
       "2               0.000000    5    0.1190   -19.924     0
\n",
       "3               0.887000    1    0.1110   -14.734     0
\n",
       "4               0.908000   11    0.0980   -16.829     1
\n",
       "...                  ...  ...       ...       ...   ...
\n",
       "169904          0.000032    1    0.0891    -7.461     1
\n",
       "169905          0.031300    8    0.1110   -10.907     1
\n",
       "169906          0.002330    7    0.1080    -9.332     1
\n",
       "169907          0.000000    1    0.2220    -2.557     1
\n",
       "169908          0.000000    0    0.1050    -7.387     1
\n",
       "\n",
       "                                                    name
popularity  \\\n",
       "0                           Singende Bataillone 1. Teil
0   \n",
       "1           Fantasiestücke, Op. 111: Più tosto lento
0   \n",
       "2                     Chapter 1.18 - Zamek kaniowski
0   \n",
       "3       Bebamos Juntos - Instrumental (Remasterizado)
0   \n",
       "4         Polonaise-Fantaisie in A-Flat Major, Op. 61
1   \n",
       "...                                                 ...
...   \n",
       "169904                  Skechers (feat. Tyga) - Remix
75   \n",
       "169905                Sweeter (feat. Terrace Martin)
64   \n",
       "169906                                How Would I Know
70   \n",
```

```
      "169907                                    I Found You
70   \n",
      "169908                           More Hearts Than Mine
65   \n",
      "\n",
      "       release_date  speechiness    tempo  valence  year
\n",
      "0              1928       0.0506  118.469   0.7790  1928
\n",
      "1              1928       0.0462   83.972   0.0767  1928
\n",
      "2              1928       0.9290  107.177   0.8800  1928
\n",
      "3        1928-09-25       0.0926  108.003   0.7200  1928
\n",
      "4              1928       0.0424   62.149   0.0693  1928
\n",
      "...             ...          ...      ...      ...   ...
\n",
      "169904   2020-05-15       0.1430  100.012   0.3060  2020
\n",
      "169905   2020-06-08       0.0403  128.000   0.2700  2020
\n",
      "169906   2020-05-29       0.1050  123.700   0.1530  2020
\n",
      "169907   2020-02-28       0.0385  129.916   0.4720  2020
\n",
      "169908   2020-03-27       0.0271   80.588   0.3660  2020
\n",
      "\n",
      "[169909 rows x 19 columns]"
     ]
    },
    "execution_count": 2,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "data=pd.read_csv('data.csv.zip',compression='zip')\n",
   "data"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 3,
  "metadata": {
   "id": "uquiLJiSkP1i"
  },
  "outputs": [],
  "source": [
   "data.drop_duplicates(inplace=True,subset=['name'])"
  ]
```

```
   },
   {
    "cell_type": "code",
    "execution_count": 4,
    "metadata": {
     "id": "_QE5-Z9VK_on"
    },
    "outputs": [],
    "source": [
     "name=data['name']"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 5,
    "metadata": {
     "id": "1_KHyoq0kQ7i"
    },
    "outputs": [],
    "source": [
     "from sklearn.cluster import KMeans\n",
     "from sklearn.preprocessing import MinMaxScaler"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 6,
    "metadata": {
     "id": "Im9wkNs7kXfJ"
    },
    "outputs": [],
    "source": [
     "col_features = ['danceability', 'energy', 'valence',
'loudness']\n",
     "X = MinMaxScaler().fit_transform(data[col_features])\n",
     "kmeans = KMeans(init=\"k-means++\",\n",
     "                n_clusters=2,\n",
     "                random_state=15).fit(X)\n",
     "data['kmeans'] = kmeans.labels_"
    ]
   },
   {
    "cell_type": "code",
    "execution_count": 7,
    "metadata": {
     "id": "JQAXWkX_KvTE"
    },
    "outputs": [],
    "source": [
     "data['song_name']=name"
    ]
   },
   {
```

```
"cell_type": "code",
"execution_count": 8,
"metadata": {
 "colab": {
  "base_uri": "https://localhost:8080/",
  "height": 623
 },
 "id": "acbgUfzDQIgp",
 "outputId": "21faa441-20a3-4895-f45c-1a96daa8e1cc"
},
"outputs": [
 {
  "data": {
   "text/html": [
    "<div>\n",
    "<style scoped>\n",
    "    .dataframe tbody tr th:only-of-type {\n",
    "        vertical-align: middle;\n",
    "    }\n",
    "\n",
    "    .dataframe tbody tr th {\n",
    "        vertical-align: top;\n",
    "    }\n",
    "\n",
    "    .dataframe thead th {\n",
    "        text-align: right;\n",
    "    }\n",
    "</style>\n",
    "<table border=\"1\" class=\"dataframe\">\n",
    "  <thead>\n",
    "    <tr style=\"text-align: right;\">\n",
    "      <th></th>\n",
    "      <th>acousticness</th>\n",
    "      <th>artists</th>\n",
    "      <th>danceability</th>\n",
    "      <th>duration_ms</th>\n",
    "      <th>energy</th>\n",
    "      <th>explicit</th>\n",
    "      <th>id</th>\n",
    "      <th>instrumentalness</th>\n",
    "      <th>key</th>\n",
    "      <th>liveness</th>\n",
    "      <th>...</th>\n",
    "      <th>mode</th>\n",
    "      <th>name</th>\n",
    "      <th>popularity</th>\n",
    "      <th>release_date</th>\n",
    "      <th>speechiness</th>\n",
    "      <th>tempo</th>\n",
    "      <th>valence</th>\n",
    "      <th>year</th>\n",
    "      <th>kmeans</th>\n",
    "      <th>song_name</th>\n",
```

```
        "    </tr>\n",
        "  </thead>\n",
        "  <tbody>\n",
        "    <tr>\n",
        "      <th>0</th>\n",
        "      <td>0.9950</td>\n",
        "      <td>['Carl Woitschach']</td>\n",
        "      <td>0.708</td>\n",
        "      <td>158648</td>\n",
        "      <td>0.1950</td>\n",
        "      <td>0</td>\n",
        "      <td>6KbQ3uYMLKb5jDxLF7wYDD</td>\n",
        "      <td>0.563000</td>\n",
        "      <td>10</td>\n",
        "      <td>0.1510</td>\n",
        "      <td>...</td>\n",
        "      <td>1</td>\n",
        "      <td>Singende Bataillone 1. Teil</td>\n",
        "      <td>0</td>\n",
        "      <td>1928</td>\n",
        "      <td>0.0506</td>\n",
        "      <td>118.469</td>\n",
        "      <td>0.7790</td>\n",
        "      <td>1928</td>\n",
        "      <td>0</td>\n",
        "      <td>Singende Bataillone 1. Teil</td>\n",
        "    </tr>\n",
        "    <tr>\n",
        "      <th>1</th>\n",
        "      <td>0.9940</td>\n",
        "      <td>['Robert Schumann', 'Vladimir
Horowitz']</td>\n",
        "      <td>0.379</td>\n",
        "      <td>282133</td>\n",
        "      <td>0.0135</td>\n",
        "      <td>0</td>\n",
        "      <td>6KuQTIu1KoTTkLXKrwlLPV</td>\n",
        "      <td>0.901000</td>\n",
        "      <td>8</td>\n",
        "      <td>0.0763</td>\n",
        "      <td>...</td>\n",
        "      <td>1</td>\n",
        "      <td>Fantasiestücke, Op. 111: Più tosto
lento</td>\n",
        "      <td>0</td>\n",
        "      <td>1928</td>\n",
        "      <td>0.0462</td>\n",
        "      <td>83.972</td>\n",
        "      <td>0.0767</td>\n",
        "      <td>1928</td>\n",
        "      <td>1</td>\n",
        "      <td>Fantasiestücke, Op. 111: Più tosto
lento</td>\n",
```

```
        "    </tr>\n",
        "    <tr>\n",
        "      <th>2</th>\n",
        "      <td>0.6040</td>\n",
        "      <td>['Seweryn Goszczyński']</td>\n",
        "      <td>0.749</td>\n",
        "      <td>104300</td>\n",
        "      <td>0.2200</td>\n",
        "      <td>0</td>\n",
        "      <td>6L63VW0PibdM1HDSBoqnoM</td>\n",
        "      <td>0.000000</td>\n",
        "      <td>5</td>\n",
        "      <td>0.1190</td>\n",
        "      <td>...</td>\n",
        "      <td>0</td>\n",
        "      <td>Chapter 1.18 - Zamek kaniowski</td>\n",
        "      <td>0</td>\n",
        "      <td>1928</td>\n",
        "      <td>0.9290</td>\n",
        "      <td>107.177</td>\n",
        "      <td>0.8800</td>\n",
        "      <td>1928</td>\n",
        "      <td>0</td>\n",
        "      <td>Chapter 1.18 - Zamek kaniowski</td>\n",
        "    </tr>\n",
        "    <tr>\n",
        "      <th>3</th>\n",
        "      <td>0.9950</td>\n",
        "      <td>['Francisco Canaro']</td>\n",
        "      <td>0.781</td>\n",
        "      <td>180760</td>\n",
        "      <td>0.1300</td>\n",
        "      <td>0</td>\n",
        "      <td>6M94FkXd15sOAOQYRnWPN8</td>\n",
        "      <td>0.887000</td>\n",
        "      <td>1</td>\n",
        "      <td>0.1110</td>\n",
        "      <td>...</td>\n",
        "      <td>0</td>\n",
        "      <td>Bebamos Juntos - Instrumental
(Remasterizado)</td>\n",
        "      <td>0</td>\n",
        "      <td>1928-09-25</td>\n",
        "      <td>0.0926</td>\n",
        "      <td>108.003</td>\n",
        "      <td>0.7200</td>\n",
        "      <td>1928</td>\n",
        "      <td>1</td>\n",
        "      <td>Bebamos Juntos - Instrumental
(Remasterizado)</td>\n",
        "    </tr>\n",
        "    <tr>\n",
        "      <th>4</th>\n",
```

```
      "       <td>0.9900</td>\n",
      "       <td>['Frédéric Chopin', 'Vladimir
Horowitz']</td>\n",
      "       <td>0.210</td>\n",
      "       <td>687733</td>\n",
      "       <td>0.2040</td>\n",
      "       <td>0</td>\n",
      "       <td>6N6tiFZ9vLTSOIxkj8qKrd</td>\n",
      "       <td>0.908000</td>\n",
      "       <td>11</td>\n",
      "       <td>0.0980</td>\n",
      "       <td>...</td>\n",
      "       <td>1</td>\n",
      "       <td>Polonaise-Fantaisie in A-Flat Major, Op.
61</td>\n",
      "       <td>1</td>\n",
      "       <td>1928</td>\n",
      "       <td>0.0424</td>\n",
      "       <td>62.149</td>\n",
      "       <td>0.0693</td>\n",
      "       <td>1928</td>\n",
      "       <td>1</td>\n",
      "       <td>Polonaise-Fantaisie in A-Flat Major, Op.
61</td>\n",
      "     </tr>\n",
      "     <tr>\n",
      "       <th>...</th>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "       <td>...</td>\n",
      "     </tr>\n",
      "     <tr>\n",
      "       <th>169901</th>\n",
      "       <td>0.2640</td>\n",
      "       <td>['Meek Mill', 'Roddy Ricch']</td>\n",
```

```
"      <td>0.744</td>\n",
"      <td>167845</td>\n",
"      <td>0.7020</td>\n",
"      <td>1</td>\n",
"      <td>0j2CNrgtalXRGIvHMO2vzh</td>\n",
"      <td>0.000000</td>\n",
"      <td>7</td>\n",
"      <td>0.1200</td>\n",
"      <td>...</td>\n",
"      <td>0</td>\n",
"      <td>Letter To Nipsey (feat. Roddy Ricch)</td>\n",
"      <td>66</td>\n",
"      <td>2020-01-27</td>\n",
"      <td>0.2880</td>\n",
"      <td>91.885</td>\n",
"      <td>0.3380</td>\n",
"      <td>2020</td>\n",
"      <td>0</td>\n",
"      <td>Letter To Nipsey (feat. Roddy Ricch)</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>169903</th>\n",
"      <td>0.2100</td>\n",
"      <td>['LEGADO 7', 'Junior H']</td>\n",
"      <td>0.795</td>\n",
"      <td>218501</td>\n",
"      <td>0.5850</td>\n",
"      <td>0</td>\n",
"      <td>52Cpyvd2dKb6XRn313nH87</td>\n",
"      <td>0.000001</td>\n",
"      <td>8</td>\n",
"      <td>0.1120</td>\n",
"      <td>...</td>\n",
"      <td>1</td>\n",
"      <td>Ojos De Maniaco</td>\n",
"      <td>68</td>\n",
"      <td>2020-02-28</td>\n",
"      <td>0.0374</td>\n",
"      <td>97.479</td>\n",
"      <td>0.9340</td>\n",
"      <td>2020</td>\n",
"      <td>0</td>\n",
"      <td>Ojos De Maniaco</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>169904</th>\n",
"      <td>0.1730</td>\n",
"      <td>['DripReport', 'Tyga']</td>\n",
"      <td>0.875</td>\n",
"      <td>163800</td>\n",
"      <td>0.4430</td>\n",
"      <td>1</td>\n",
"      <td>4KppkflX7I3vJQk7urOJaS</td>\n",
```

```
"      <td>0.000032</td>\n",
"      <td>1</td>\n",
"      <td>0.0891</td>\n",
"      <td>...</td>\n",
"      <td>1</td>\n",
"      <td>Skechers (feat. Tyga) - Remix</td>\n",
"      <td>75</td>\n",
"      <td>2020-05-15</td>\n",
"      <td>0.1430</td>\n",
"      <td>100.012</td>\n",
"      <td>0.3060</td>\n",
"      <td>2020</td>\n",
"      <td>1</td>\n",
"      <td>Skechers (feat. Tyga) - Remix</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>169905</th>\n",
"      <td>0.0167</td>\n",
"      <td>['Leon Bridges', 'Terrace Martin']</td>\n",
"      <td>0.719</td>\n",
"      <td>167468</td>\n",
"      <td>0.3850</td>\n",
"      <td>0</td>\n",
"      <td>1ehhGlTvjtHo2e4xJFB0SZ</td>\n",
"      <td>0.031300</td>\n",
"      <td>8</td>\n",
"      <td>0.1110</td>\n",
"      <td>...</td>\n",
"      <td>1</td>\n",
"      <td>Sweeter (feat. Terrace Martin)</td>\n",
"      <td>64</td>\n",
"      <td>2020-06-08</td>\n",
"      <td>0.0403</td>\n",
"      <td>128.000</td>\n",
"      <td>0.2700</td>\n",
"      <td>2020</td>\n",
"      <td>1</td>\n",
"      <td>Sweeter (feat. Terrace Martin)</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>169906</th>\n",
"      <td>0.5380</td>\n",
"      <td>['Kygo', 'Oh Wonder']</td>\n",
"      <td>0.514</td>\n",
"      <td>180700</td>\n",
"      <td>0.5390</td>\n",
"      <td>0</td>\n",
"      <td>52eycxprLhK3lPcRLbQiVk</td>\n",
"      <td>0.002330</td>\n",
"      <td>7</td>\n",
"      <td>0.1080</td>\n",
"      <td>...</td>\n",
"      <td>1</td>\n",
```

```
       "      <td>How Would I Know</td>\n",
       "      <td>70</td>\n",
       "      <td>2020-05-29</td>\n",
       "      <td>0.1050</td>\n",
       "      <td>123.700</td>\n",
       "      <td>0.1530</td>\n",
       "      <td>2020</td>\n",
       "      <td>1</td>\n",
       "      <td>How Would I Know</td>\n",
       "    </tr>\n",
       "  </tbody>\n",
       "</table>\n",
       "<p>132940 rows × 21 columns</p>\n",
       "</div>"
      ],
      "text/plain": [
       "        acousticness
artists  danceability  \\\n",
       "0             0.9950                          ['Carl
Woitschach']         0.708   \n",
       "1             0.9940  ['Robert Schumann', 'Vladimir
Horowitz']         0.379   \n",
       "2             0.6040                   ['Seweryn
Goszczyński']         0.749   \n",
       "3             0.9950                      ['Francisco
Canaro']         0.781   \n",
       "4             0.9900  ['Frédéric Chopin', 'Vladimir
Horowitz']         0.210   \n",
       "...              ...
...           ...   \n",
       "169901        0.2640              ['Meek Mill', 'Roddy
Ricch']         0.744   \n",
       "169903        0.2100                  ['LEGADO 7',
'Junior H']         0.795   \n",
       "169904        0.1730                    ['DripReport',
'Tyga']         0.875   \n",
       "169905        0.0167        ['Leon Bridges', 'Terrace
Martin']         0.719   \n",
       "169906        0.5380                     ['Kygo', 'Oh
Wonder']         0.514   \n",
       "\n",
       "        duration_ms  energy  explicit
id  \\\n",
       "0            158648  0.1950         0
6KbQ3uYMLKb5jDxLF7wYDD   \n",
       "1            282133  0.0135         0
6KuQTIu1KoTTkLXKrwlLPV   \n",
       "2            104300  0.2200         0
6L63VW0PibdM1HDSBoqnoM   \n",
       "3            180760  0.1300         0
6M94FkXd15sOAOQYRnWPN8   \n",
       "4            687733  0.2040         0
6N6tiFZ9vLTSOIxkj8qKrd   \n",
```

```
       "...              ...     ...       ...
...   \n",
       "169901       167845  0.7020         1
0j2CNrgtalXRGIvHMO2vzh   \n",
       "169903       218501  0.5850         0
52Cpyvd2dKb6XRn313nH87   \n",
       "169904       163800  0.4430         1
4KppkflX7I3vJQk7urOJaS   \n",
       "169905       167468  0.3850         0
1ehhGlTvjtHo2e4xJFB0SZ   \n",
       "169906       180700  0.5390         0
52eycxprLhK3lPcRLbQiVk   \n",
       "\n",
       "        instrumentalness  key  liveness  ...  mode
\\\n",
       "0               0.563000   10    0.1510  ...     1
\n",
       "1               0.901000    8    0.0763  ...     1
\n",
       "2               0.000000    5    0.1190  ...     0
\n",
       "3               0.887000    1    0.1110  ...     0
\n",
       "4               0.908000   11    0.0980  ...     1
\n",
       "...                  ...  ...       ...  ...   ...
\n",
       "169901          0.000000    7    0.1200  ...     0
\n",
       "169903          0.000001    8    0.1120  ...     1
\n",
       "169904          0.000032    1    0.0891  ...     1
\n",
       "169905          0.031300    8    0.1110  ...     1
\n",
       "169906          0.002330    7    0.1080  ...     1
\n",
       "\n",
       "                                                    name
popularity  \\\n",
       "0                           Singende Bataillone 1. Teil
0   \n",
       "1            Fantasiestücke, Op. 111: Più tosto lento
0   \n",
       "2                      Chapter 1.18 - Zamek kaniowski
0   \n",
       "3       Bebamos Juntos - Instrumental (Remasterizado)
0   \n",
       "4         Polonaise-Fantaisie in A-Flat Major, Op. 61
1   \n",
       "...                                                  ...
...   \n",
```

```
       "169901           Letter To Nipsey (feat. Roddy Ricch)
66   \n",
       "169903                                    Ojos De Maniaco
68   \n",
       "169904                  Skechers (feat. Tyga) - Remix
75   \n",
       "169905                Sweeter (feat. Terrace Martin)
64   \n",
       "169906                                    How Would I Know
70   \n",
       "\n",
       "        release_date speechiness    tempo  valence  year
kmeans  \\\n",
       "0               1928      0.0506  118.469   0.7790  1928
0   \n",
       "1               1928      0.0462   83.972   0.0767  1928
1   \n",
       "2               1928      0.9290  107.177   0.8800  1928
0   \n",
       "3         1928-09-25      0.0926  108.003   0.7200  1928
1   \n",
       "4               1928      0.0424   62.149   0.0693  1928
1   \n",
       "...              ...         ...      ...      ...   ...
...   \n",
       "169901    2020-01-27      0.2880   91.885   0.3380  2020
0   \n",
       "169903    2020-02-28      0.0374   97.479   0.9340  2020
0   \n",
       "169904    2020-05-15      0.1430  100.012   0.3060  2020
1   \n",
       "169905    2020-06-08      0.0403  128.000   0.2700  2020
1   \n",
       "169906    2020-05-29      0.1050  123.700   0.1530  2020
1   \n",
       "\n",
       "                                              song_name
\n",
       "0                          Singende Bataillone 1. Teil
\n",
       "1             Fantasiestücke, Op. 111: Più tosto lento
\n",
       "2                       Chapter 1.18 - Zamek kaniowski
\n",
       "3       Bebamos Juntos - Instrumental (Remasterizado)
\n",
       "4         Polonaise-Fantaisie in A-Flat Major, Op. 61
\n",
       "...                                                ...
\n",
       "169901           Letter To Nipsey (feat. Roddy Ricch)
\n",
```

```
       "169903                                  Ojos De Maniaco
\n",
       "169904                   Skechers (feat. Tyga) - Remix
\n",
       "169905                  Sweeter (feat. Terrace Martin)
\n",
       "169906                                  How Would I Know
\n",
       "\n",
       "[132940 rows x 21 columns]"
      ]
     },
     "execution_count": 8,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "data"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 9,
   "metadata": {
    "id": "iTRkAs8AkdzD"
   },
   "outputs": [],
   "source": [
    "cluster=data.groupby(by=data['kmeans'])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 10,
   "metadata": {
    "id": "aPYcUVwvnFwM"
   },
   "outputs": [],
   "source": [
    "y=data.pop('kmeans')\n",

"x=data.drop(columns=['name','artists','id','release_date','song
_name'])\n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 11,
   "metadata": {
    "id": "9O1CvTVyl_8o"
   },
   "outputs": [],
```

```
   "source": [
    "from sklearn.model_selection import train_test_split\n",
"x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.
25)"
   ]
  },
```

Переходи та формування рекомендацій

```
    },
    "metadata": {
     "needs_background": "light"
    },
    "output_type": "display_data"
   }
  ],
  "source": [
   "import lightgbm\n",
   "import matplotlib.pyplot as plt\n",
   "ax = lightgbm.plot_importance(model, max_num_features=10,
figsize=(15,15))\n",
   "plt.show()"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 16,
  "metadata": {
   "id": "1y0iZzU3HfnR"
  },
  "outputs": [],
  "source": [
   "df=cluster.apply(lambda x:
x.sort_values([\"popularity\"],ascending=False))\n",
   "df.reset_index(level=0, inplace=True)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 17,
  "metadata": {
   "colab": {
    "base_uri": "https://localhost:8080/",
    "height": 229
   },
   "id": "NNBzwoH3t6Q8",
   "outputId": "b2b58f9c-0108-4c60-bd49-5a95db121a00"
  },
  "outputs": [],
  "source": [
   "from keras.preprocessing.image import img_to_array\n",
```

```
    "import imutils\n",
    "from keras.models import load_model\n",
    "import numpy as np\n",
    "import cv2"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 26,
   "metadata": {},
   "outputs": [],
   "source": [
    "detection_model_path = 'haarcascade_frontalface_default.xml'\n",
    "emotion_model_path = 'final_model3.h5'\n",
    "face_detection = cv2.CascadeClassifier(detection_model_path)\n",
    "emotion_classifier = load_model(emotion_model_path, compile=False)\n",
    "EMOTIONS = [\"happy\",\"sad\"]"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 39,
   "metadata": {},
   "outputs": [],
   "source": [
    "from keras.preprocessing import image\n",
    "def emotion_testing()::\n",
    "        cap=cv2.VideoCapture(0)\n",
    "        while True:\n",
    "            ret,test_img=cap.read()# captures frame and returns boolean
value and captured image\n",
    "            if not ret:\n",
    "                continue\n",
    "            gray_img= cv2.cvtColor(test_img, cv2.COLOR_BGR2GRAY)\n",
    "\n",
    "            faces_detected = face_detection.detectMultiScale(gray_img,
1.32, 5)\n",
    "\n",
    "\n",
    "            for (x,y,w,h) in faces_detected:\n",
    "
cv2.rectangle(test_img,(x,y),(x+w,y+h),(255,0,0),thickness=7)\n",
    "                roi_gray=gray_img[y:y+w,x:x+h]#cropping region of
interest i.e. face area from  image\n",
    "                roi_gray=cv2.resize(roi_gray,(48,48))\n",
    "                img_pixels = image.img_to_array(roi_gray)\n",
    "                img_pixels = np.expand_dims(img_pixels, axis = 0)\n",
    "                img_pixels /= 255\n",
    "\n",
    "                predictions = emotion_classifier.predict(img_pixels)\n",
    "\n",
    "                #find max indexed array\n",
    "                max_index = np.argmax(predictions[0])\n",
    "                predicted_emotion = EMOTIONS[max_index]\n",
    "\n",
```

```
    "                cv2.putText(test_img, predicted_emotion, (int(x),
int(y)), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255), 2)\n",
    "\n",
    "            resized_img = cv2.resize(test_img, (1000, 700))\n",
    "            cv2.imshow('Facial emotion analysis ',resized_img)\n",
    "\n",
    "\n",
    "            if cv2.waitKey(0) & 0xFF == ord('q'):\n",
    "                    break\n",
    "        cap.release()\n",
    "        cv2.destroyAllWindows\n",
    "        return predicted_emotion"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 46,
   "metadata": {},
   "outputs": [],
   "source": [
    "emotion_word=emotion_testing()\n",
    "if emotion_word=='sad':\n",
    "    emotion_code=0\n",
    "else:\n",
    "    emotion_code=1"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 47,
   "metadata": {
    "id": "QHyMLtLnHsiD"
   },
   "outputs": [],
   "source": [
    "def get_results(emotion_code):\n",
    "  NUM_RECOMMEND=10\n",
    "  happy_set=[]\n",
    "  sad_set=[]\n",
    "  if emotion_code==0:\n",
    "
happy_set.append(df[df['kmeans']==0]['song_name'].head(NUM_RECOMMEND))\n",
    "      return pd.DataFrame(happy_set).T\n",
    "  else:\n",
    "
sad_set.append(df[df['kmeans']==1]['song_name'].head(NUM_RECOMMEND))\n",
    "      return pd.DataFrame(sad_set).T"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 48,
   "metadata": {},
   "outputs": [
```

```
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "                                  song_name\n",
    "87852  ily (i love you baby) (feat. Emilee)\n",
    "87844                             Supalonely\n",
    "87953     Stuck with U (with Justin Bieber)\n",
    "87858                           Dance Monkey\n",
    "87952       Rain On Me (with Ariana Grande)\n",
    "87951                                GOOBA\n",
    "87850                       Don't Start Now\n",
    "87890                           Breaking Me\n",
    "87969                            Boss Bitch\n",
    "87970                         Yo Perreo Sola\n",
    "emotion detected is SAD\n"
   ]
  }
 ],
 "source": [
  "print(get_results(emotion_code))\n",
  "if emotion_word=='sad':\n",
  "    print('emotion detected is SAD')\n",
  "else:\n",
  "    print('emotion detected is HAPPY')"
 ]
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "colab": {
   "base_uri": "https://localhost:8080/",
   "height": 343
  },
  "id": "TJ8QZCIrUC7g",
  "outputId": "75533041-2e43-482c-f3ab-208b69f42114"
 },
 "outputs": [],
 "source": []
},
{
 "cell_type": "code",
 "execution_count": null,
 "metadata": {
  "id": "QoXSslE8uyZY"
 },
 "outputs": [],
 "source": []
}
```

Додаток Д

Приклад реалізації інформаційної технології, що враховує комбінування методів для побудови рекомендацій на основі переваг та відмов користувача (фрагменти лістингу коду)

```
-->
<opencv_storage>
<cascade type_id="opencv-cascade-classifier"><stageType>BOOST</stageType>
  <featureType>HAAR</featureType>
  <height>20</height>
  <width>20</width>
  <stageParams>
    <maxWeakCount>93</maxWeakCount></stageParams>
  <featureParams>
    <maxCatCount>0</maxCatCount></featureParams>
  <stageNum>24</stageNum>
  <stages>
    <_>
      <maxWeakCount>6</maxWeakCount>
      <stageThreshold>-1.4562760591506958e+00</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 0 1.2963959574699402e-01</internalNodes>
          <leafValues>
            -7.7304208278656006e-01 6.8350148200988770e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 1 -4.6326808631420135e-02</internalNodes>
          <leafValues>
            5.7352751493453979e-01 -4.9097689986228943e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 2 -1.6173090785741806e-02</internalNodes>
          <leafValues>
            6.0254341363906860e-01 -3.1610709428787231e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 3 -4.5828841626644135e-02</internalNodes>
          <leafValues>
            6.4177548885345459e-01 -1.5545040369033813e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 4 -5.3759619593620300e-02</internalNodes>
          <leafValues>
            5.4219317436218262e-01 -2.0480829477310181e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 5 3.4171190112829208e-02</internalNodes>
          <leafValues>
            -2.3388190567493439e-01 4.8410901427268982e-
01</leafValues></_></weakClassifiers></_>
    <_>
      <maxWeakCount>12</maxWeakCount>
      <stageThreshold>-1.2550230026245117e+00</stageThreshold>
      <weakClassifiers>
```

```
        <_>
          <internalNodes>
            0 -1 6 -2.1727620065212250e-01</internalNodes>
          <leafValues>
            7.1098899841308594e-01 -5.9360730648040771e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 7 1.2071969918906689e-02</internalNodes>
          <leafValues>
            -2.8240481019020081e-01 5.9013551473617554e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 8 -1.7854139208793640e-02</internalNodes>
          <leafValues>
            5.3137522935867310e-01 -2.2758960723876953e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 9 2.2333610802888870e-02</internalNodes>
          <leafValues>
            -1.7556099593639374e-01 6.3356137275695801e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 10 -9.1420017182826996e-02</internalNodes>
          <leafValues>
            6.1563092470169067e-01 -1.6899530589580536e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 11 2.8973650187253952e-02</internalNodes>
          <leafValues>
            -1.2250079959630966e-01 7.4401170015335083e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 12 7.8203463926911354e-03</internalNodes>
          <leafValues>
            1.6974370181560516e-01 -6.5441650152206421e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 13 2.0340489223599434e-02</internalNodes>
          <leafValues>
            -1.2556649744510651e-01 8.2710450887680054e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 14 -1.1926149949431419e-02</internalNodes>
          <leafValues>
            3.8605681061744690e-01 -2.0992340147495270e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 15 -9.7281101625412703e-04</internalNodes>
          <leafValues>
            -6.3761192560195923e-01 1.2952390313148499e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 16 1.8322050891583785e-05</internalNodes>
          <leafValues>
            -3.4631478786468506e-01 2.2924269735813141e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 17 -8.0854417756199837e-03</internalNodes>
          <leafValues>
            -6.3665801286697388e-01 1.3078659772872925e-
01</leafValues></_></weakClassifiers></_>
    <_>
      <maxWeakCount>9</maxWeakCount>
```

```
      <stageThreshold>-1.3728189468383789e+00</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 18 -1.1812269687652588e-01</internalNodes>
          <leafValues>
            6.7844521999359131e-01 -5.0045782327651978e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 19 -3.4332759678363800e-02</internalNodes>
          <leafValues>
            6.7186361551284790e-01 -3.5744878649711609e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 20 -2.1530799567699432e-02</internalNodes>
          <leafValues>
            7.2220700979232788e-01 -1.8192419409751892e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 21 -2.1909970790147781e-02</internalNodes>
          <leafValues>
            6.6529387235641479e-01 -2.7510228753089905e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 22 -2.8713539242744446e-02</internalNodes>
          <leafValues>
            6.9955700635910034e-01 -1.9615580141544342e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 23 -1.1467480100691319e-02</internalNodes>
          <leafValues>
            5.9267348051071167e-01 -2.2097350656986237e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 24 -2.2611169144511223e-02</internalNodes>
          <leafValues>
            3.4483069181442261e-01 -3.8379558920860291e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 25 -1.9308089977130294e-03</internalNodes>
          <leafValues>
            -7.9445719718933105e-01 1.5628659725189209e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 26 5.6419910833938047e-05</internalNodes>
          <leafValues>
            -3.0896010994911194e-01 3.5431089997291565e-
01</leafValues></_></weakClassifiers></_>
    <_>
      <maxWeakCount>16</maxWeakCount>
      <stageThreshold>-1.2879480123519897e+00</stageThreshold>
      <weakClassifiers>
        <_>
          <internalNodes>
            0 -1 27 1.9886520504951477e-01</internalNodes>
          <leafValues>
            -5.2860701084136963e-01 3.5536721348762512e-01</leafValues></_>
        <_>
```


***


```
        <_>
          <internalNodes>
            0 -1 1050 -5.7094299700111151e-04</internalNodes>
```

```
  <leafValues>
    -1.4076329767704010e-01 1.0287419706583023e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 1051 -2.2152599412947893e-03</internalNodes>
  <leafValues>
    1.6593599319458008e-01 -8.5273988544940948e-02</leafValues></_>
<_>
  <internalNodes>
    0 -1 1052 -2.8084890916943550e-02</internalNodes>
  <leafValues>
    2.7022340893745422e-01 -5.5873811244964600e-02</leafValues></_>
<_>
  <internalNodes>
    0 -1 1053 2.1515151020139456e-03</internalNodes>
  <leafValues>
    4.2472891509532928e-02 -3.2005849480628967e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 1054 -2.9733829433098435e-04</internalNodes>
  <leafValues>
    1.6177169978618622e-01 -8.5115589201450348e-02</leafValues></_>
<_>
  <internalNodes>
    0 -1 1055 -1.6694780439138412e-02</internalNodes>
  <leafValues>
    -4.2858770489692688e-01 3.0541609972715378e-02</leafValues></_>
<_>
  <internalNodes>
    0 -1 1056 1.1982990056276321e-01</internalNodes>
  <leafValues>
    -1.6277290880680084e-02 7.9846781492233276e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 1057 -3.5499420482665300e-04</internalNodes>
  <leafValues>
    1.5935939550399780e-01 -8.3272881805896759e-02</leafValues></_>
<_>
  <internalNodes>
    0 -1 1058 -1.8226269632577896e-02</internalNodes>
  <leafValues>
    1.9527280330657959e-01 -7.3939889669418335e-02</leafValues></_>
<_>
  <internalNodes>
    0 -1 1059 -4.0238600922748446e-04</internalNodes>
  <leafValues>
    7.9101808369159698e-02 -2.0806129276752472e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 1060 4.0892060496844351e-04</internalNodes>
  <leafValues>
    1.0036630183458328e-01 -1.5128210186958313e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 1061 9.5368112670257688e-04</internalNodes>
  <leafValues>
    -7.3011666536331177e-02 2.1752020716667175e-01</leafValues></_>
<_>
  <internalNodes>
    0 -1 1062 4.3081799149513245e-01</internalNodes>
  <leafValues>
    -2.7450699359178543e-02 5.7061582803726196e-01</leafValues></_>
<_>
```

```
          <internalNodes>
            0 -1 1063 5.3564831614494324e-04</internalNodes>
          <leafValues>
            1.1587540060281754e-01 -1.2790560722351074e-01</leafValues></_>
        <_>
          <internalNodes>
            0 -1 1064 2.4430730263702571e-05</internalNodes>
          <leafValues>
            -1.6816629469394684e-01 8.0449983477592468e-02</leafValues></_>
        <_>
          <internalNodes>
            0 -1 1065 -5.5345650762319565e-02</internalNodes>
          <leafValues>
            4.5338949561119080e-01 -3.1222779303789139e-
02</leafValues></_></weakClassifiers></_></stages>
  <features>
    <_>
      <rects>
        <_>
          0 8 20 12 -1.</_>
        <_>
          0 14 20 6 2.</_></rects></_>
    <_>
      <rects>
        <_>
          9 1 4 15 -1.</_>
        <_>
          9 6 4 5 3.</_></rects></_>
    <_>
      <rects>
        <_>
          6 10 9 2 -1.</_>
        <_>
          9 10 3 2 3.</_></rects></_>
    <_>
      <rects>
        <_>
          7 0 10 9 -1.</_>
        <_>
          7 3 10 3 3.</_></rects></_>
    <_>
      <rects>
        <_>
          12 2 2 18 -1.</_>
        <_>
          12 8 2 6 3.</_></rects></_>
    <_>
      <rects>
        <_>
          8 6 8 6 -1.</_>
        <_>
          8 9 8 3 2.</_></rects></_>
    <_>
      <rects>
```


***


```
        <_>
          6 10 2 1 -1.</_>
        <_>
          7 10 1 1 2.</_></rects></_>
    <_>
      <rects>
```

```
          <_>
            1 1 18 16 -1.</_>
          <_>
            10 1 9 16 2.</_></rects></_>
      <_>
        <rects>
          <_>
            14 4 3 15 -1.</_>
          <_>
            15 4 1 15 3.</_></rects></_>
      <_>
        <rects>
          <_>
            19 13 1 2 -1.</_>
          <_>
            19 14 1 1 2.</_></rects></_>
      <_>
        <rects>
          <_>
            2 6 5 8 -1.</_>
          <_>
            2 10 5 4 2.</_></rects></_></features></cascade>
</opencv_storage>
```

Додаток Ж

Приклади забезпечення логічної еквівалентності

Використовуючи граф автомата Мілі (рис. 4.9) і матрицю з'єднань (табл. 4.3) можна скласти таблиці переходів і виходів, а також таблицю переходів із забезпеченням еквівалентності для двох автоматів.

Це можна реалізувати наступним чином.

Таблиця Ж.1 – Таблиця переходів

A / X	a_1	a_2	a_3	a_4
x_1	a_2	a_1	a_2	a_4
x_2	a_1	a_4	a_1	a_3

Таблиця Ж.2 – Таблиця виходів

A / X	a_1	a_2	a_3	a_4
x_1	y_1	y_2	y_1	y_3
x_2	y_2	y_3	y_2	y_1

Таблиця Ж.3 – Таблиця переходів із забезпеченням еквівалентності

A / X	a_1	a_2	a_3	a_4
x_1	a_2 / y_1	a_1 / y_2	a_2 / y_1	a_4 / y_3
x_2	a_1 / y_2	a_4 / y_3	a_1 / y_2	a_3 / y_1

Припускаємо, що автомат Мілі заданий зазначеною таблицею переходів (табл. Ж4).

Таблиця Ж.4 – Таблиця переходів

A_A / X_A	a_0	a_1	a_2
x_1	a_2 / y_1	a_0 / y_1	a_0 / y_2
x_2	a_0 / y_1	a_2 / y_2	a_1 / y_1

Побудуємо еквівалентний автомат Мура. Для цього поставимо у відповідність кожній парі a_i/x_k стан b_{ik} (i-номер стану, k-номер вхідного сигналу), з урахуванням b_0 (див. табл. Ж5).

Таблиця Ж.5 – Таблиця відповідностей

A_A / X_A	a_0 b_0	a_1	a_2
x_1	a_2 / y_1 b_{01}	a_0 / y_1 b_{11}	a_0 / y_2 b_{21}
x_2	a_0 / y_1 b_{02}	a_2 / y_2 b_{12}	a_1 / y_1 b_{22}

Складемо таблицю переходів автомата Мура, керуючись наступними правилами:

1) Випишемо з табл. Ж5 стани автомата Мілі і відповідні кожному з них множини станів автомата Мура (b_{ik}).

$$a_0 = \{b_0, b_{02}, b_{11}, b_{21}\};\ a_1 = \{b_{22}\};\ a_2 = \{b_{01}, b_{12}\}\ .$$

2) Якщо стан автомата Мура b_{ik} входить в множину a_p автомата Мілі, то в стовпчик таблиці переходів автомата Мура для стану b_{ik} слід записати стовпчик з таблиці переходів автомата Мілі, який відповідає стану a_p (з табл. Ж5).

3) Функцію виходів автомата Мура визначимо наступним чином $\lambda_B=(b_{ik})=\lambda_A(a_i,x_k)$. Для початкового стану b_0 значення вихідного сигналу можна вибрати довільно, але породжуваний початковим станом a_0 (з урахуванням поняття еквівалентності стану).

Результуюча таблиця переходів і виходів автомата Мура еквівалентного автомату Мілі, заданому в табл. Ж5, представлена в табл. Ж6.

Таблиця Ж.6 – Результатуюча таблиця

Y	y_1	y_1	y_1	y_1	y_2	y_2	y_1
A_B / X	b_0	b_{01}	b_{02}	b_{11}	b_{12}	b_{21}	b_{22}
x_1	b_{01}	b_{21}	b_{01}	b_{01}	b_{21}	b_{01}	b_{11}
x_2	b_{02}	b_{22}	b_{02}	b_{02}	b_{22}	b_{02}	b_{12}
	*		*	*			

4) Знайдемо в табл. Ж6 еквівалентні стани і видалимо їх (замінимо на представника класів еквівалентності).

Якщо вихідний сигнал біля b_0 довизначити y_1, то виявиться, що в даній таблиці переходів знаходиться три еквівалентних стани: $b_0 \equiv b_{02} \equiv b_{11}$. Замінивши клас еквівалентності одним представником (b_0), отримаємо остаточну зазначену таблицю переходів автомата Мура (табл. Ж7). При цьому стовпці, що відповідають еквівалентним станам (b_{02} і b_{11}) видаляємо з таблиці, а в інших стовпцях замінюємо їх представником класу еквівалентності, тобто b_0.

Таблиця Ж.7 – Остаточна таблиця переходів автомата Мура

Y	y_1	y_1	y_2	y_2	y_1
A_B / X	b_0	b_{01}	b_{12}	b_{21}	b_{22}
x_1	b_{01}	b_{21}	b_{21}	b_{01}	b_0
x_2	b_0	b_{22}	b_{22}	b_0	b_{12}

Додаток З

Лістинг коду веб-додатка інформаційної технології рекомендаційної підтримки прийняття рішень

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <script>
    /*! jQuery v3.4.1 | (c) JS Foundation and other contributors |
jquery.org/license */
    !function(e, t) {
        "use strict";
        "object" == typeof module && "object" == typeof module.exports ?
module.exports = e.document ? t(e, !0) : function(e) {
            if (!e.document)
                throw new Error("jQuery requires a window with a document");
            return t(e)
        } : t(e)
    }("undefined" != typeof window ? window : this, function(C, e) {
        "use strict";
        var t = [],
            E = C.document,
            r = Object.getPrototypeOf,
            s = t.slice,
            g = t.concat,
            u = t.push,
            i = t.indexOf,
            n = {},
            o = n.toString,
            v = n.hasOwnProperty,
            a = v.toString,
            l = a.call(Object),
            y = {},
            m = function(e) {
                return "function" == typeof e && "number" != typeof
e.nodeType
            },
            x = function(e) {
                return null != e && e === e.window
            },
            c = {
                type: !0,
                src: !0,
                nonce: !0,
                noModule: !0
            };
        function b(e, t, n) {
            var r,
                i,
                o = (n = n || E).createElement("script");
            if (o.text = e, t)
                for (r in c)
                    (i = t[r] || t.getAttribute && t.getAttribute(r)) &&
o.setAttribute(r, i);
            n.head.appendChild(o).parentNode.removeChild(o)
        }
        function w(e) {
```

```
            return null == e ? e + "" : "object" == typeof e || "function" ==
typeof e ? n[o.call(e)] || "object" : typeof e
        }
        var f = "3.4.1",
            k = function(e, t) {
                return new k.fn.init(e, t)
            },
            p = /^[\s\uFEFF\xA0]+|[\s\uFEFF\xA0]+$/g;
        function d(e) {
            var t = !!e && "length" in e && e.length,
                n = w(e);
            return !m(e) && !x(e) && ("array" === n || 0 === t || "number" ==
typeof t && 0 < t && t - 1 in e)
        }
        k.fn = k.prototype = {
            jquery: f,
            constructor: k,
            length: 0,
            toArray: function() {
                return s.call(this)
            },
            get: function(e) {
                return null == e ? s.call(this) : e < 0 ? this[e +
this.length] : this[e]
            },
            pushStack: function(e) {
                var t = k.merge(this.constructor(), e);
                return t.prevObject = this, t
            },
            each: function(e) {
                return k.each(this, e)
            },
            map: function(n) {
                return this.pushStack(k.map(this, function(e, t) {
                    return n.call(e, t, e)
                }))
            },
            slice: function() {
                return this.pushStack(s.apply(this, arguments))
            },
            first: function() {
                return this.eq(0)
            },
            last: function() {
                return this.eq(-1)
            },
            eq: function(e) {
                var t = this.length,
                    n = +e + (e < 0 ? t : 0);
                return this.pushStack(0 <= n && n < t ? [this[n]] : [])
            },
            end: function() {
                return this.prevObject || this.constructor()
            },
            push: u,
            sort: t.sort,
            splice: t.splice
        },
        k.extend = k.fn.extend = function() {
            var e,
                t,
                n,
                r,
```

```
                i,
                o,
                a = arguments[0] || {},
                s = 1,
                u = arguments.length,
                l = !1;
            for ("boolean" == typeof a && (l = a, a = arguments[s] || {},
s++), "object" == typeof a || m(a) || (a = {}), s === u && (a = this, s--); s
< u; s++)
                if (null != (e = arguments[s]))
                    for (t in e)
                        r = e[t],
                        "__proto__" !== t && a !== r && (l && r &&
(k.isPlainObject(r) || (i = Array.isArray(r))) ? (n = a[t], o = i &&
!Array.isArray(n) ? [] : i || k.isPlainObject(n) ? n : {}, i = !1, a[t] =
k.extend(l, o, r)) : void 0 !== r && (a[t] = r));
            return a
        },
        k.extend({
            expando: "jQuery" + (f + Math.random()).replace(/\D/g, ""),
            isReady: !0,
            error: function(e) {
                throw new Error(e)
            },
            noop: function() {},
            isPlainObject: function(e) {
                var t,
                    n;
                return !(!e || "[object Object]" !== o.call(e)) && (!(t =
r(e)) || "function" == typeof (n = v.call(t, "constructor") && t.constructor)
&& a.call(n) === l)
            },
            isEmptyObject: function(e) {
                var t;
                for (t in e)
                    return !1;
                return !0
            },
            globalEval: function(e, t) {
                b(e, {
                    nonce: t && t.nonce
                })
            },
            each: function(e, t) {
                var n,
                    r = 0;
                if (d(e)) {
                    for (n = e.length; r < n; r++)
                        if (!1 === t.call(e[r], r, e[r]))
                            break
                } else
                    for (r in e)
                        if (!1 === t.call(e[r], r, e[r]))
                            break;
                return e
            },
            trim: function(e) {
                return null == e ? "" : (e + "").replace(p, "")
            },
            makeArray: function(e, t) {
                var n = t || [];
                return null != e && (d(Object(e)) ? k.merge(n, "string" ==
typeof e ? [e] : e) : u.call(n, e)), n
```

```
            },
            inArray: function(e, t, n) {
                return null == t ? -1 : i.call(t, e, n)
            },
            merge: function(e, t) {
                for (var n = +t.length, r = 0, i = e.length; r < n; r++)
                    e[i++] = t[r];
                return e.length = i, e
            },
            grep: function(e, t, n) {
                for (var r = [], i = 0, o = e.length, a = !n; i < o; i++)
                    !t(e[i], i) !== a && r.push(e[i]);
                return r
            },
            map: function(e, t, n) {
                var r,
                    i,
                    o = 0,
                    a = [];
                if (d(e))
                    for (r = e.length; o < r; o++)
                        null != (i = t(e[o], o, n)) && a.push(i);
                else
                    for (o in e)
                        null != (i = t(e[o], o, n)) && a.push(i);
                return g.apply([], a)
            },
            guid: 1,
            support: y
        }),
        "function" == typeof Symbol && (k.fn[Symbol.iterator] =
t[Symbol.iterator]),
        k.each("Boolean Number String Function Array Date RegExp Object Error
Symbol".split(" "), function(e, t) {
            n["[object " + t + "]"] = t.toLowerCase()
        });
        var h = function(n) {
            var e,
                d,
                b,
                o,
                i,
                h,
                f,
                g,
                w,
                u,
                l,
                T,
                C,
                a,
                E,
                v,
                s,
                c,
                y,
                k = "sizzle" + 1 * new Date,
                m = n.document,
                S = 0,
                r = 0,
                p = ue(),
                x = ue(),
                N = ue(),
```

```
                A = ue(),
                D = function(e, t) {
                    return e === t && (l = !0), 0
                },
                j = {}.hasOwnProperty,
                t = [],
                q = t.pop,
                L = t.push,
                H = t.push,
                O = t.slice,
                P = function(e, t) {
                    for (var n = 0, r = e.length; n < r; n++)
                        if (e[n] === t)
                            return n;
                    return -1
                },
                R =
"checked|selected|async|autofocus|autoplay|controls|defer|disabled|hidden|ism
ap|loop|multiple|open|readonly|required|scoped",
                M = "[\\x20\\t\\r\\n\\f]",
                I = "(?:\\\\.|[\\w-]|[^\0-\\xa0])+",
                W = "\\[" + M + "*(" + I + ")(?:" + M + "*([*^$|!~]?=)" + M +
"*(?:'((?:\\\\.|[^\\\\'])*)'|\"((?:\\\\.|[^\\\\\"])*)\"|(" + I + "))|)" + M +
"*\\]",
                $ = ":(" + I +
")(?:\\(((('((?:\\\\.|[^\\\\'])*)'|\"((?:\\\\.|[^\\\\\"])*)\")|((?:\\\\.|[^\\\
\()[\\]]|" + W + ")*)|.*)\\)|)",
                F = new RegExp(M + "+", "g"),
                B = new RegExp("^" + M + "+|((?:^|[^\\\\])(?:\\\\.)*)" + M +
"+$", "g"),
                _ = new RegExp("^" + M + "*," + M + "*"),
                z = new RegExp("^" + M + "*([>+~]|" + M + ")" + M + "*"),
                U = new RegExp(M + "|>"),
                X = new RegExp($),
                V = new RegExp("^" + I + "$"),
                G = {
                    ID: new RegExp("^#(" + I + ")"),
                    CLASS: new RegExp("^\\.(" + I + ")"),
                    TAG: new RegExp("^(" + I + "|[*])"),
                    ATTR: new RegExp("^" + W),
                    PSEUDO: new RegExp("^" + $),
                    CHILD: new RegExp("^:(only|first|last|nth|nth-last)-
(child|of-type)(?:\\(" + M + "*(even|odd|(([+-]|)(\\d*)n|)" + M + "*(?:([+-
]|)" + M + "*(\\d+)|))" + M + "*\\)|)", "i"),
                    bool: new RegExp("^(?:" + R + ")$", "i"),
                    needsContext: new RegExp("^" + M +
"*[>+~]|:(even|odd|eq|gt|lt|nth|first|last)(?:\\(" + M + "*((?:-\\d)?\\d*)" +
M + "*\\)|)(?=[^-]|$)", "i")
                },
                Y = /HTML$/i,
                Q = /^(?:input|select|textarea|button)$/i,
                J = /^h\d$/i,
                K = /^[^{]+\{\s*\[native \w/,
                Z = /^(?:#([\w-]+)|(\w+)|\.([\w-]+))$/,
                ee = /[+~]/,
                te = new RegExp("\\\\([\\da-f]{1,6}" + M + "?|(" + M +
")|.)", "ig"),
                ne = function(e, t, n) {
                    var r = "0x" + t - 65536;
                    return r != r || n ? t : r < 0 ? String.fromCharCode(r +
65536) : String.fromCharCode(r >> 10 | 55296, 1023 & r | 56320)
                },
                re = /([\0-\x1f\x7f]|^-?\d)|^-$|[^\0-\x1f\x7f-\uFFFF\w-]/g,
```

```
                ie = function(e, t) {
                    return t ? "\0" === e ? "\ufffd" : e.slice(0, -1) + "\\"
+ e.charCodeAt(e.length - 1).toString(16) + " " : "\\" + e
                },
                oe = function() {
                    T()
                },
                ae = be(function(e) {
                    return !0 === e.disabled && "fieldset" ===
e.nodeName.toLowerCase()
                }, {
                    dir: "parentNode",
                    next: "legend"
                });
            try {
                H.apply(t = O.call(m.childNodes), m.childNodes),
                t[m.childNodes.length].nodeType
            } catch (e) {
                H = {
                    apply: t.length ? function(e, t) {
                        L.apply(e, O.call(t))
                    } : function(e, t) {
                        var n = e.length,
                            r = 0;
                        while (e[n++] = t[r++])
                            ;
                        e.length = n - 1
                    }
                }
            }

***

            }
            function ve(a) {
                return le(function(o) {
                    return o = +o, le(function(e, t) {
                        var n,
                            r = a([], e.length, o),
                            i = r.length;
                        while (i--)
                            e[n = r[i]] && (e[n] = !(t[n] = e[n]))
                    })
                })
            }

***

                        return r
                    }
                    return o
                }, b.find.CLASS = d.getElementsByClassName && function(e, t)
{
                    if ("undefined" != typeof t.getElementsByClassName && E)
                        return t.getElementsByClassName(e)
                }, s = [], v = [], (d.qsa = K.test(C.querySelectorAll)) &&
(ce(function(e) {
                    a.appendChild(e).innerHTML = "<a id='" + k +
"'></a><select id='" + k + "-\r\\' msallowcapture=''><option
selected=''></option></select>",
                    e.querySelectorAll("[msallowcapture^='']").length &&
v.push("[*^$]=" + M + "*(?:''|\"\")"),
```

```
                    e.querySelectorAll("[selected]").length || v.push("\\[" +
M + "*(?:value|" + R + ")"),
                    e.querySelectorAll("[id~=" + k + "-]").length ||
v.push("~="),
                    e.querySelectorAll(":checked").length ||
v.push(":checked"),
                    e.querySelectorAll("a#" + k + "+*").length ||
v.push(".#.+[+~]")
                }), ce(function(e) {
                    e.innerHTML = "<a href='' disabled='disabled'></a><select
disabled='disabled'><option/></select>";
                    var t = C.createElement("input");
                    t.setAttribute("type", "hidden"),
                    e.appendChild(t).setAttribute("name", "D"),
                    e.querySelectorAll("[name=d]").length && v.push("name" +
M + "*[*^$|!~]?="),
                    2 !== e.querySelectorAll(":enabled").length &&
v.push(":enabled", ":disabled"),
                    a.appendChild(e).disabled = !0,
                    2 !== e.querySelectorAll(":disabled").length &&
v.push(":enabled", ":disabled"),
                    e.querySelectorAll("*,:x"),
                    v.push(",.*:")
                })), (d.matchesSelector = K.test(c = a.matches ||
a.webkitMatchesSelector || a.mozMatchesSelector || a.oMatchesSelector ||
a.msMatchesSelector)) && ce(function(e) {
                    d.disconnectedMatch = c.call(e, "*"),
                    c.call(e, "[s!='']:x"),
                    s.push("!=", $)
                }), v = v.length && new RegExp(v.join("|")), s = s.length &&
new RegExp(s.join("|")), t = K.test(a.compareDocumentPosition), y = t ||
K.test(a.contains) ? function(e, t) {
                    var n = 9 === e.nodeType ? e.documentElement : e,
                        r = t && t.parentNode;
                    return e === r || !(!r || 1 !== r.nodeType ||
!(n.contains ? n.contains(r) : e.compareDocumentPosition && 16 &
e.compareDocumentPosition(r)))
                } : function(e, t) {
                    if (t)
                        while (t = t.parentNode)
                            if (t === e)
                                return !0;
                    return !1
                }, D = t ? function(e, t) {
                    if (e === t)
                        return l = !0, 0;
                    var n = !e.compareDocumentPosition -
!t.compareDocumentPosition;
                    return n || (1 & (n = (e.ownerDocument || e) ===
(t.ownerDocument || t) ? e.compareDocumentPosition(t) : 1) || !d.sortDetached
&& t.compareDocumentPosition(e) === n ? e === C || e.ownerDocument === m &&
y(m, e) ? -1 : t === C || t.ownerDocument === m && y(m, t) ? 1 : u ? P(u, e)
- P(u, t) : 0 : 4 & n ? -1 : 1)
                } : function(e, t) {
                    if (e === t)
                        return l = !0, 0;
                    var n,
                        r = 0,
                        i = e.parentNode,
                        o = t.parentNode,
                        a = [e],
                        s = [t];
                    if (!i || !o)
```

```
                        return e === C ? -1 : t === C ? 1 : i ? -1 : o ? 1 :
u ? P(u, e) - P(u, t) : 0;
                    if (i === o)
                        return pe(e, t);
                    n = e;
                    while (n = n.parentNode)
                        a.unshift(n);
                    n = t;
                    while (n = n.parentNode)
                        s.unshift(n);
                    while (a[r] === s[r])
                        r++;
                    return r ? pe(a[r], s[r]) : a[r] === m ? -1 : s[r] === m
? 1 : 0
                }), C
            }, se.matches = function(e, t) {
                return se(e, null, null, t)
            }, se.matchesSelector = function(e, t) {
                if ((e.ownerDocument || e) !== C && T(e), d.matchesSelector
&& E && !A[t + " "] && (!s || !s.test(t)) && (!v || !v.test(t)))
                    try {
                        var n = c.call(e, t);
                        if (n || d.disconnectedMatch || e.document && 11 !==
e.document.nodeType)
                            return n
                    } catch (e) {
                        A(t, !0)
                    }

***

            },
            undelegate: function(e, t, n) {
                return 1 === arguments.length ? this.off(e, "**") :
this.off(t, e || "**", n)
            }
        }),
        k.proxy = function(e, t) {
            var n,
                r,
                i;
            if ("string" == typeof t && (n = e[t], t = e, e = n), m(e))
                return r = s.call(arguments, 2), (i = function() {
                    return e.apply(t || this, r.concat(s.call(arguments)))
                }).guid = e.guid = e.guid || k.guid++, i
        },
        k.holdReady = function(e) {
            e ? k.readyWait++ : k.ready(!0)
        },
        k.isArray = Array.isArray,
        k.parseJSON = JSON.parse,
        k.nodeName = A,
        k.isFunction = m,
        k.isWindow = x,
        k.camelCase = V,
        k.type = w,
        k.now = Date.now,
        k.isNumeric = function(e) {
            var t = k.type(e);
            return ("number" === t || "string" === t) && !isNaN(e -
parseFloat(e))
        },
```

```
        "function" == typeof define && define.amd && define("jquery", [],
function() {
            return k
        });
        var Qt = C.jQuery,
            Jt = C.$;
        return k.noConflict = function(e) {
            return C.$ === k && (C.$ = Jt), e && C.jQuery === k && (C.jQuery
= Qt), k
        }, e || (C.jQuery = C.$ = k), k
    });
    </script>
    <script>
    let g_snapped = "";
    // let g_lastHovered = "";

    function hideAllDetails()
    {
        $(".container-feature-detail").hide();
        $(".container-df-associations").hide();
        $("span.bg-tab-summary-rollover").hide();
    }

    // GLOBAL EVENTS
    // ------------------------------------------------------------------------
----------------------------------------------------
    // EVENT: [ANYWHERE] RIGHT-CLICK REMOVES SELECTION
    // $(document).contextmenu(function() {
    //     if (g_snapped != "")
    //     {
    //         g_snapped = "";
    //         hideAllDetails();
    //     }
    //     if (g_lastHovered != "")
    //     {
    //         $(g_lastHovered).show();
    //         //alert("#"+g_lastHovered);
    //     }
    //     return false;
    // });

    $("span.bg-tab-summary-rollover").hide();
    hideAllDetails();

    $(document).ready(function() {
        // INITIALIZATION
        // ----------------------------------------------------------
        hideAllDetails();
        $("span.bg-tab-summary-rollover").hide();

        // Make the detail column the same height, so the floating element
has room
        //$("#col2").height($("#col1").height());
        $("#col1").height(g_height);
        $("#col2").height(g_height);
        //alert($("#col1").height());

        // SUMMARY AREA
        // ----------------------------------------------------------
        // EVENT: SUMMARY ROLLOVER
        // $(".selector, .container-feature-summary-target").hover(
        $(".selector").hover(// ENTER function
        function(event) {
```

```
            if (g_snapped == "")
            {
                // Rollover start!
                $(".container-feature-detail").hide();
                $("span.bg-tab-summary-rollover").hide();
                $("#" + $(this).data("detail-div")).show();
                $("#" + $(this).data("rollover-span")).removeClass("bg-tab-
summary-rollover-locked");
                $("#" + $(this).data("rollover-span")).addClass("bg-tab-
summary-rollover");
                $("#" + $(this).data("rollover-span")).show();
            }

        }, // g_lastHovered = "#" + $(this).data("detail-div");
        // EXIT function
        function(event) {
            if (g_snapped == "")
            {
                // Rollover end!
                hideAllDetails();

            }
        }
        );
        //FBFB        $("#" + $(this).data("detail-div")).hide();
        // EVENT: SUMMARY CLICK
        // $(".container-feature-summary, .container-feature-summary-
target").click(function(event) {
        $(".selector").click(function(event) {
            // alert($(this).parent().attr('id'));
            let this_to_snap = $(this).parent().attr('id');

            if (g_snapped == this_to_snap)
            {
                // "Unselect"
                $("#" + $(this).data("rollover-span")).removeClass("bg-tab-
summary-rollover-locked");
                $("#" + $(this).data("rollover-span")).addClass("bg-tab-
summary-rollover");
                g_snapped = "";
            }
            else if (g_snapped == "")
            {
                // "Select"
                $("#" + $(this).data("rollover-span")).removeClass("bg-tab-
summary-rollover");
                $("#" + $(this).data("rollover-span")).addClass("bg-tab-
summary-rollover-locked");
                g_snapped = $(this).parent().attr('id');



            }
            else //$("#" + $(this).data("detail-div")).show();
            //$(g_lastHovered).show();
            // alert(this.parent().id);
            if (g_snapped !== this_to_snap)// implied
            {
                // "Select" while another was previously selected
                $("#" + $("#" + g_snapped).children().data("rollover-
span")).removeClass("bg-tab-summary-rollover-locked");
                $("#" + $("#" + g_snapped).children().data("rollover-
span")).addClass("bg-tab-summary-rollover");
```

```
                $(".container-feature-detail").hide();
                $("span.bg-tab-summary-rollover").hide();
                $("#" + $(this).data("detail-div")).show();

                $("#" + $(this).data("rollover-span")).removeClass("bg-tab-
summary-rollover");
                $("#" + $(this).data("rollover-span")).addClass("bg-tab-
summary-rollover-locked");
                $("#" + $(this).data("rollover-span")).css("display",
"inline");
                g_snapped = $(this).parent().attr('id');
            }
        }
        );

        // SPECIFIC BUTTONS
        // ------------------------------------------------------------------
-------------------------------------------------------
        // SUMMARY: ASSOCIATIONS
        $("#button-summary-associations-source, #button-summary-associations-
compare").hover(// ENTER function
        function()
        {
            if (g_snapped == "")
            {
                hideAllDetails();
                $("#" + $(this).data("detail-div")).show();


            }

        }, // $("#df-assoc").show();
        //$("#df-assoc").show();
        // g_lastHovered = "#df-assoc";
        // EXIT function
        function()
        {
            if (g_snapped == "")
            {
                hideAllDetails();
            }
        });
        // ASSOCIATIONS CLICK
        $("#button-summary-associations-source, #button-summary-associations-
compare").click(function(event) {
            let this_to_snap = this.id;
            if (g_snapped == this_to_snap)
            {
                // "Unselect"
                g_snapped = "";
            }
            else if (g_snapped == "")
            {
                // "Select"
                //$(".container-feature-detail").hide();
                //alert("#" + this.id+" GS:"+g_snapped);
                //$("#df-assoc").show();
                g_snapped = this.id;
            }
            else
            {
                // "Select" while another was previously selected
```

```
                $("#" + $("#" + g_snapped).children().data("rollover-
span")).removeClass("bg-tab-summary-rollover-locked");
                $("#" + $("#" + g_snapped).children().data("rollover-
span")).addClass("bg-tab-summary-rollover");
                hideAllDetails();
                g_snapped = this.id;
                $("#" + $(this).data("detail-div")).show();
            }
        });

        // DETAIL GRAPH BUTTONS
        $(".button-bin").click(function() {
            which_id = $(this).attr('data-target');
            $("#" + which_id).attr('class', $(this).attr('data-new_class') +
" pos-detail-num-graph");
        });


    }); // $(document).ready(...
    </script>
    <style>
    @font-face
    {
        font-family: RobotoBoldCond;
        font-style: normal;
        font-weight: normal;
        src: url ……
```