Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____

Центр післядипломної освіти

Кафедра_____Програмної інженерії_____

АТЕСТАЦІЙНА РОБОТА

Пояснювальна записка

_____другий (магістерський)_____

Дослідження методів аналізу даних поведінки користувачів веб орієнтованих систем

Виконав: студент 2 курсу, групи ІПЗ мзд-17-1_____
спеціальності 121 – Інженерія програмного забезпечення
спеціалізації Інженерія програмного забезпечення_____
_____Іванов О.В._____
Керівник к.т.н, доц. каф. ПІ Назаров О.С.

Допускається до захисту
Зав. кафедри, проф. _____ З.В.Дудар
2019 р.

Харківський національний університет радіоелектроніки

Центр післядипломної освіти

Кафедра _Програмної інженерії_____

Рівень вищої освіти _____другий (магістерський)_____

Спеціальність _121 – Інженерія програмного забезпечення_____

Спеціалізація _Інженерія програмного забезпечення_____

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

«_____»_____ 20___ р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові _Іванову Олегу Васильовичу_____

1. Тема роботи Дослідження методів аналізу даних поведінки користувачів веб орієнтованих систем

затверджена наказом по університету від “___” _____20 ___ р № _____

2. Термін подання студентом роботи до екзаменаційної комісії

07 червня 2019 р.

3. Вихідні дані до роботи *методи аналізу даних поведінки користувачів, пояснювальна записка, програмна система*

4. Перелік питань, що потрібно опрацювати в роботі *мета роботи, аналіз проблемної галузі і постановка задачі, огляд методів аналізу даних поведінки користувачів*

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) *Мета завдання, обґрунтування доцільності розроблення, постановка задачі, об'єктна модель системи, базові моделі, методи й алгоритми, структура бази даних, структурно-логічна схема взаємодії даних, інтерфейс програмної системи, результати тестування програмної системи, демонстраційні матеріали*

6. Консультанти розділів роботи

Найменування розділу	Консультант	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	к.т.н, доц. каф. ПІ Назаров О.С.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка*
1.	Аналіз предметної галузі	11 лютого 2019 р.	
2.	Огляд існуючих методів	21 лютого 2019 р.	
3.	Методи аналізу даних поведінки користувачів	25 лютого 2019 р.	
4.	Підготовка пояснювальної записки	07 березня 2019 р.	
5.	Спецчастина	11 квітня 2019 р.	
6.	Підготовка презентації та доповіді	21 травня 2019 р.	
7.	Попередній захист	30 травня 2019 р.	
8.	Нормоконтроль, рецензування	03 червня 2019 р.	
9.	Занесення диплома в електронний архів	04 червня 2019 р.	
10.	Допуск до захисту у зав. кафедри	05 червня 2019 р.	

Дата видачі завдання _____ _____2019 р.

Студент _____

Керівник роботи _____ к.т.н, доц. каф. ПІ Назаров О.С.

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до атестаційної роботи: 70 с., 21 рис., 2 табл., 3 додатки, 18 джерел.

АНАЛІТИКА, ПОВЕДІНКА КОРИСТУВАЧІВ, NODE JS, WEB–САЙТ.

Об’єктом дослідження є методи аналізу даних поведінки користувачів веб-орієнтованих систем.

Метою роботи є дослідження, виявлення особливостей та вдосконалення існуючих методів аналізу даних поведінки користувачів.

Методи розробки базуються на технології NODE JS.

У результаті роботи здійснена програмна реалізація системи дослідження поведінки користувачів. Запропоновано вдосконалений метод прогнозування поведінки користувачів веб-орієнтованих систем.

Explanatory note to the project: 70 p., 21 images, 2 tables, 3 adds, 18 sources.

ANALYTICS, BEHAVIOR OF USERS, NODE JS, WEB SITE.

The object of research is analysis methods of the user behavior data of the web oriented systems.

The aim – research, identifying features and improving existing analysis methods of the user behavior data.

Methods of developing technology based on NODE JS.

As a result of the work – the program realization of the user behavior data analysis system. Proposed the improved method of user behavior prediction.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

WWW	World Wide Web
URL	Uniform Resource Locator
IP	Internet Protocol (address)
TLPM	Two-level prediction model
TPM	Transition Probability Matrix
FPMM	Frequency-pruned Markov model
CPMM	Confidence-pruned Markov model
EPMM	Error-pruned Markov model
SMM	Selective Markov Models
HMM	Hidden Markov Models

ВСТУП

Актуальність теми. З самого початку виникнення мережі інтернет створення оптимальної структури і якісного змісту веб-сайту для залучення максимуму інтересу відвідувачів тісно пов'язана з розумінням цілей користувача та його мотивації для відвідування веб-сайту. Зрозуміло, що, маючи більш точні знання про інтереси та переваги веб-користувачів, кращий зміст і структуру можна запропонувати.

Аналіз поведінки людини проводився в таких різноманітних дисциплінах, як психологія, соціологія, економіка, лінгвістика, маркетинг та інформатика. Звідси доступні широкі теоретичні рамки, з високим потенціалом для застосування в інших галузях знань, наприклад в особливостях аналізу поведінки веб-користувачів. Ці попередні дисципліни використовують опитування та експериментальні вибірки для тестування та калібрування їх теоретичних моделей. Для Інтернету користувачем основним джерелом даних є веб-журнали, які зберігають дії кожного відвідувача в Інтернеті сайту. Такі файли можуть містити мільйони регістрів залежно від трафіку веб-сайту, що становить головне джерело даних про поведінку людини.

Оскільки серверні журнали призначені здебільшого для програмістів і являють собою величезні обсяги текстових даних, то інтернет-маркетологу буде не особливо зручно з цими даними працювати безпосередньо. І тут на допомогу приходить спеціальне програмне забезпечення за допомогою якого нескінченні потоки цифр можна перетворюватися в доступні для розуміння маркетолога графічні формати – таблиці, діаграми, графіки.

З точки зору електронної комерції, поведінкова аналітика також дозволяє передбачити, які саме продукти будуть зацікавлені ваші клієнти в певний час дня або пори року. Прогнозування того, які товари будуть цікаві різним користувачам з різних місць у світі також може бути корисно для поведінкової сегментації та прогнозування конверсії. Продавці використовують аналітику даних для кращого

ознайомлення з їхніми клієнтами, оптимізації процесу онлайн-покупок та збільшенні продажів.

Аналітика поведінки користувачів допомагає виявляти внутрішні загрози, цільові атаки і фінансові шахрайства. Технологія також здатна виявляти шаблони які зображують підозрілу поведінку, з інсайдера або хакера. На жаль, технологія не здатна зупинити атаки у системах, але вона може виявити діяльність і звести до мінімуму будь-який збиток, який був би викликаний.

Зв'язок роботи з програмами наукових досліджень кафедри ПІ. Атестаційна робота виконувалась згідно з тематичним планом науково-дослідницьких робіт кафедри ПІ.

Метою дослідження є виявлення особливостей та вдосконалення існуючих методів аналізу даних поведінки користувачів веб-орієнтованих систем.

Задачами досліджень є:

− аналіз особливостей відомих теоретичних та експериментальних результатів;

− виявлення шляхів і засобів удосконалення існуючих та запропонування нових методів аналізу даних поведінки веб-користувачів;

− удосконалення методу прогнозування поведінки веб-користувачів з використанням моделей Маркова;

− оцінка ефективності та перевірка якості запропонованого методу шляхом проведення аналітичного та експериментального дослідження.

Об’єктом дослідження є серверні логи, процес обробки та методи їх аналізу.

Предметом дослідження є моделі, методи і алгоритми для виявлення та визначення поведінки веб-користувачів на основі серверних логів.

Методи дослідження. Теоретичні дослідження базуються на: загальних методах функціонального аналізу. Для тестування запропонованого методу використовувалась система написана на технології NodeJS.

Наукова новизна одержаних результатів. Удосконалено метод прогнозування поведінки користувачів веб-орієнтованих систем з використанням моделей Маркова.

Практичне значення одержаних результатів полягає у створенні нових методів аналізу даних поведінки користувачів та впровадження їх у системи аналітики веб-орієнтованих систем.

Публікації. Основні результати роботи опубліковані в науково практичних конференціях:

− Іванов О.В. Дослідження основних джерел, процесів і методів обробки даних поведінки користувачів веб-орієнтованих систем. Новини науки: дослідження, наукові відкриття, високі технології: зб. наук. праць «ΛΌГОΣ» з матеріалами міжнар. наук.-практ. конф., м. Харків, 31 березня, 2019 р. Харків : ГО «Європейська наукова платформа», 2019. – С. 15-17. ISBN 978-617-7171-80-4;

− Іванов О.В. Дослідження методів аналізу даних поведінки користувачів веб-орієнтованих систем. Збірник наукових робіт «ADVANCED OF SCIENCE» (Карлові Вари, Чехія) з матеріалами міжнар. наук.-практ. конф. «Discovery Science», м. Київ, м. Карлові Вари, 5 квітня 2019 р. – С. 213-218. ISBN 978-80-7534-078-8;

− Іванов О.В. Прогнозування поведінки користувачів веб-орієнтованих систем на основі аналізу серверних логів. Міжнародна наукова інтернет-конференція "Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 37)" / Збірник тез доповідей: випуск 37 (м. Тернопіль, 2 квітня 2019 р.). – Тернопіль, 2019. – С. 23-25. ISSN 2522-932X;

− Іванов О.В. Попередня обробка даних для аналізу поведінки користувачів веб-орієнтованих систем. Прикладні наукові розробки та теоретичні дослідження XXI століття: зб. наук. праць «ΛΌГОΣ» з матеріалами міжнар. наук.-практ. конф., м. Вінниця, 15 квітня, 2019 р. Вінниця: ГО «Європейська наукова платформа», 2019. – С 72-74. ISBN 978-617-7171-80-4.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз галузі та сфери використання

З ранніх часів цивілізації людство зіткнулося з проблемою розуміння самого себе. Трейдери передбачають потреби людей, політики розраховують кроки з найкращим політичним результатом, а генерали вирішують позицію армії. Люди живуть разом у суспільствах, що складають складні системи взаємозалежності. Одним з етапів для побудови кращого суспільства є наявність достатніх знань про людську поведінку [3].

Соціальні науки нещодавно використовували багато сучасних інструментів, таких як динамічні та стохастичні системи для опису структур соціальних змін. Кожна соціальна система, природно, описувалася як дуже складна мережа взаємодії, яку вважали неможливим математично відобразити. Проте, за допомогою абстракції багато моделей були розроблені для пояснення певних соціальних явищ. Наприклад, додатки до політики виявляються шляхом моделювання динаміки масової думки. Модель динаміки виборців [4] може допомогти зрозуміти результати масового голосування.

У бізнес-середовищі маркетинг є сукупністю процесів, які допомагають визначити переваги клієнтів і попит на продукти та послуги, рекомендуючи стратегії продажів і комунікацій. Google – це компанія, яка базує свій основний бізнес на вибраній рекламі на сторінці результатів пошуку. Маркетингова стратегія Google базується на рейтингах веб-користувачів для веб-сторінок. Проста стохастична модель для перегляду веб-сторінок (алгоритм Page Rank [5]) генерує стаціонарні ймовірності для розрахунку цього рейтингу. Тому Інтернет став новою межею у сфері маркетингу, що обіцяє комерційний успіх, але ціною необхідності мати точні знання про веб-користувача [3].

Можна згадати деякі успішні приклади в електронній комерції, такі як Amazon.com. Amazon є американською компанією, яка в основному розробляється як книжковий інтернет-магазин, але також перейшла на торгівлю електронними

пристроями, меблями та іншими товарами. Amazon.com вважається першим промоутером інтернет-магазинів з успішними бізнес-результатами. Її річний чистий прибуток склав близько 902 мільйонів доларів США у 2019 році покладається на онлайнові рекомендації клієнтам відповідно до виявленого шаблону профілювання. Такі технології засновані на прогнозуванні переваг веб-користувача на основі його спостережуваної навігаційної поведінки.

Netflix є ще однією компанією з дот-комом з чистим прибутком у 719 млн. Доларів США у 2018 році, який зосереджується на оренді фільмів DVD. У 2006 році ця компанія запропонувала конкурс на один мільйон доларів для поліпшення на десять відсотків ефективності свого алгоритму, що рекомендує кіно. Проте, виграш отримав лише у вересні 2009 року після чотирьох років спроб світових конкурентів. Переможець використовував специфічний алгоритм інтелектуального аналізу даних, подібний до багатьох інших. Головний висновок, який можна зробити, це те, що моделювання поведінки людини є дуже складною проблемою.

Для вивчення поведінки користувача для навігації в Інтернеті важливо спочатку пояснити деякі основні поняття про веб-систему (див. рис. 1.1).

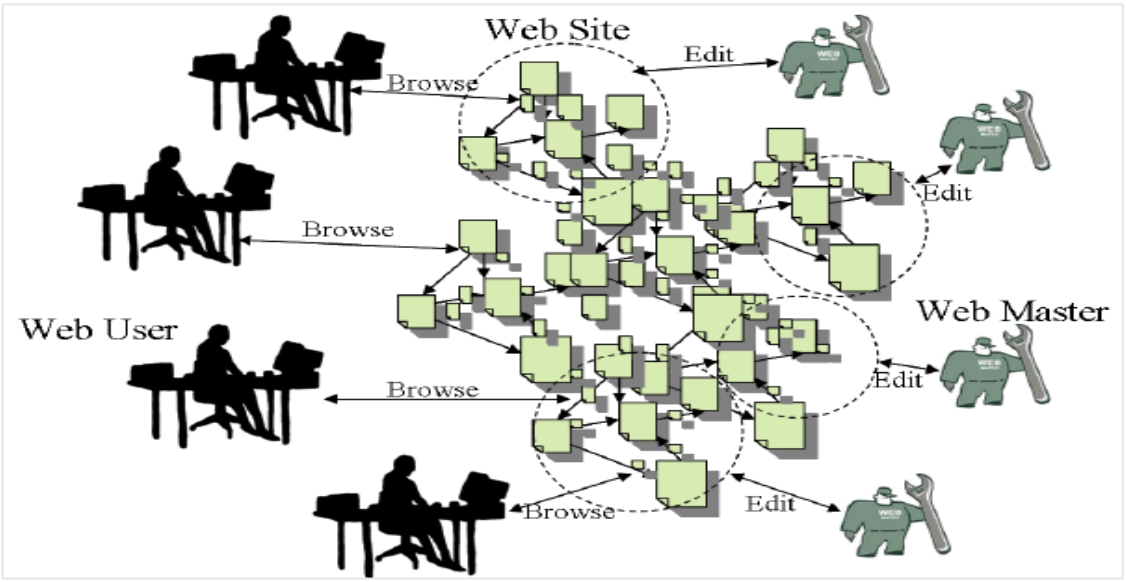

Рисунок 1.1 – Опис першого рівня веб-системи користувача сайту

Веб-користувачі вважаються людськими суб'єктами, які за допомогою веб-браузера отримують доступ до інформаційних ресурсів у гіпер медіапросторі під

назвою World Wide Web (WWW). Загальними цілями веб-користувачів є пошук інформації (пошук інформації про щось), діяльність у соціальних мережах (наприклад, Facebook), транзакції електронної торгівлі (наприклад, Amazon Shopping), банківські операції тощо.

З іншого боку, гіпермедійний простір організований на веб-сторінках, які можна охарактеризувати як компактні субодиниці під назвою «веб-об'єкти». Дизайн веб-сторінок створюється "веб-майстрами", які відповідають за групу сторінок, що називаються "веб-сайтом". Тому WWW складається з великого сховища взаємопов'язаних веб-сайтів для різних цілей.

На другому рівні абстракції веб-сторінки поширюються за допомогою "веб-серверів", а "веб-браузери" використовуються для запуску механізму. Веб-сторінка являє HTML-кодований документ, який містить гіперпосилання на інші сторінки. Вміст веб-сторінки відповідає візуальному тексту та мультимедіа, який веб-користувач сприймає, коли браузер інтерпретує код HTML.

Веб-структура відповідає графіку сторінок, пов'язаних гіперпосиланнями. Насправді, як зміст, так і структура були дуже динамічними, оскільки додаток Web 2.0 почав персоналізувати веб-сайти до поточного користувача, наприклад Facebook або Amazon.

У цьому контексті система, що складається з веб-користувачів, які здійснюють навігацію по веб-сайтах, має дуже складний опис.

Перше застосування моделі поведінки людини в мережі, подібно алгоритму Page Rank. Цей алгоритм мав дуже велику репутацію, оскільки його використовувала пошукова система Google. PageRank – це числова величина, що характеризує важливість веб-сторінки. Чим більше посилань на сторінку, тим вона важливіша. Таким чином, PageRank – це метод обчислення ваги сторінки шляхом підрахунку важливості посилань на неї.

Алгоритм відвідування користувачем будь-якої посилання на сторінці з однаковою ймовірністю. Ця стохастична модель також охоплює ймовірність перезапуску процесу, повторюючи навігацію на іншій рівномірно розподіленій сторінці, що призводить до ергодичної ланцюга Маркова.

Простий стохастичний процес дотримується стаціонарних правил ймовірності. Найбільш важливі або цікаві сторінки мають найбільшу ймовірність того, що випадковий користувач відвідає його.

Цей процес створює рейтинг для сторінок, які використовуються у пошукових системах. Подальші уточнення цієї ідеї були використані у новій парадигмі для ранжирування сторінок у всесвітній павутині [1], де веб-користувачі розглядаються як потоки через мережу гіперпосилань.

Інші підходи стосуються моделей оцінки якості навігації веб-сайтів. Зокрема [2], мають в собі загальні дії перегляду, які виконує веб-користувач, наприклад припинення сеансу, перехід до, повернення, перебування та перехід до. Кожне з цих дій стосується величини ймовірності, яка включена в ланцюгову модель Маркова.

1.2 Постановка задачі

Постановка задачі дослідження може бути сформульована наступним чином:

− проаналізувати особливості відомих теоретичних та експериментальних результатів;

− виявити шляхи і засоби удосконалення існуючих та запропонувати нові методи аналізу даних поведінки веб-користувачів;

− удосконалити метод прогнозування поведінки веб-користувачів з використанням моделей Маркова;

− оцінити ефективність та перевірити якість запропонованого методу шляхом проведення аналітичного та експериментального дослідження;

− визначити можливі напрями подальшого вдосконалення роботи та поліпшення точності прогнозування.

1.3 Обгрунтування мети дослідження

Прогнозування майбутньої поведінки користувачів у мережі є ключовим питанням у поведінковому націлюванні. Поведінкова орієнтація – це динамічно розвивається сфера веб-видобутку, що стосується оптимізації онлайн реклами на основі аналізу поведінки користувачів Інтернету. Підхід до аналізу користувачів нещодавно став центром інтересу в он-лайн рекламі та має великий потенціал у підвищенні продуктивності систем обслуговування об'яв, що підтверджено останніми експериментами [10]. Таким чином, питання є проблемою що має велике значення для індустрії та водночас має високу складність.

Існує більше зв'язків між нашою проблемою та онлайновою рекламою. Наприклад, у динамічно зростаючому ринку пошукової реклами часто ставиться завдання визначити якість реклами, щоб класифікувати їх на сторінці результатів пошуку. Якість повинна оцінюватися з історичних даних або шляхом дослідження популярності (виміряної за кількістю кліків) конкретної реклами в минулому, або шляхом вилучення особливостей зі змісту реклами та використання одного з методів машинного навчання.

1.4 Аналіз стану вирішення проблеми за матеріалами закордонних публікацій

Agrawal R. та Srikant R. (1994) запропонували метод побудови першого і другого порядків марківських моделей прогнозування доступу до веб-сайту на основі минулої поведінки відвідувача і порівняння техніки асоціативних правил. У цих підходах послідовності запитів користувачів збираються методом ідентифікації сеансу, який відрізняє запити для однієї веб-сторінки в різних браузерах.

Трилок Натх Панді, Ранджіта Кумарі Даш, Алака Нанда Трипаті, Барналі Саху Пітков J і Піроллі П (1999) запропонували інтегрувати марківську модель з кластеризацією (IMC) для прогнозування майбутніх запитів користувачів. У даній роботі автор представив удосконалення точності марківської моделі шляхом групування веб-сесій у кластери. Веб-сторінки у користувальницьких сесіях спочатку розподіляються на категорії відповідно до веб-служб які функціонально найважливіші. Потім алгоритм кластеризації k-means реалізується з використанням найбільш відповідного числа кластерів і вимірювання відстані.

Дослідники використовували різні моделі передбачення, включаючи такі як k-найближчий сусід (kNN), ANNs (Nasraoui.O і Krishnapuram.R 2002), нечіткий висновок (fuzzy inference) (Nasraoui.O і Petenes.C 2003) SVM, байєсівська модель, марківська модель та інші.

Mobasher використовував техніку ARM (associative rule mining) і запропонував графік частого набору елементів для відповідності активної сесії користувача з частими наборами елементів і передбачити наступну сторінку, яку користувач може відвідати. Тим не менш, ARM страждає від обмежень таких як масштабованість і ефективність.

Дворівнева модель прогнозування (TLPM), Chu-Hui Lee (2011). На першому рівні марківська модель використовується для прогнозування наступної можливої категорії, яка буде переглянута користувачем.

На другому рівні теорема Байеса використовується для прогнозування наступної можливої сторінки, яка належить до передбаченої категорії першого рівня.

Результат експерименту доводить, що TLPM може підвищувати ефективність прогнозування шляхом виявлення важливої категорії на першому рівні і зменшення сторінки-кандидата, встановленої на другому рівні.

Joachims та інші. запропонували WebWatcher, який базується на шляху користувача рекомендаційною моделлю, заснованою на методі k-найближчий сусід (kNN) та навчанні. Використовується поєднання попередніх турів подібних користувачів та навчання для створення рекомендацій.

Sujatha та Punithavalli (2012) запропонували прогнозування шаблонів навігації користувачем за допомогою кластеризації та класифікації (PUCC) з даних веб-журналу. На першому етапі PUCC зосереджується на відокремленні потенційних користувачів у даних веб-журналу, а на другому етапі процес кластеризації використовується для групування потенційних користувачів з подібним інтересом, а на третьому етапі використовуються результати класифікації та кластеризації для прогнозування майбутніх запитів користувачів.

2 ШЛЯХИ ВДОСКОНАЛЕННЯ ІСНУЮЧИХ ТА ЗАПРОПОНУВАННЯ НОВИХ МЕТОДІВ

2.1 Репрезентативні змінні

Поведінка перегляду веб-користувачів може бути описана трьома видами даних [15]: веб-структурою, веб-контентом і веб-сеансом користувача. Перший безпосередньо пов'язаний з навколишнім середовищем. Третій описує потік кліків який виконує кожен веб-користувач під час свого відвідування веб-сайту.

Веб-структура. Веб-сайт може бути представлений (див. рис. 2.1) у вигляді спрямованого графа G (N, V, T), що складається з набору n вузлів N = {1, ..., n} і вершин V = {(i, j)) / точка веб-посилання від і до j} з вмістом тексту T = {T_i}. Вузол і з G відповідає веб-сторінці з текстовим вмістом T_i.

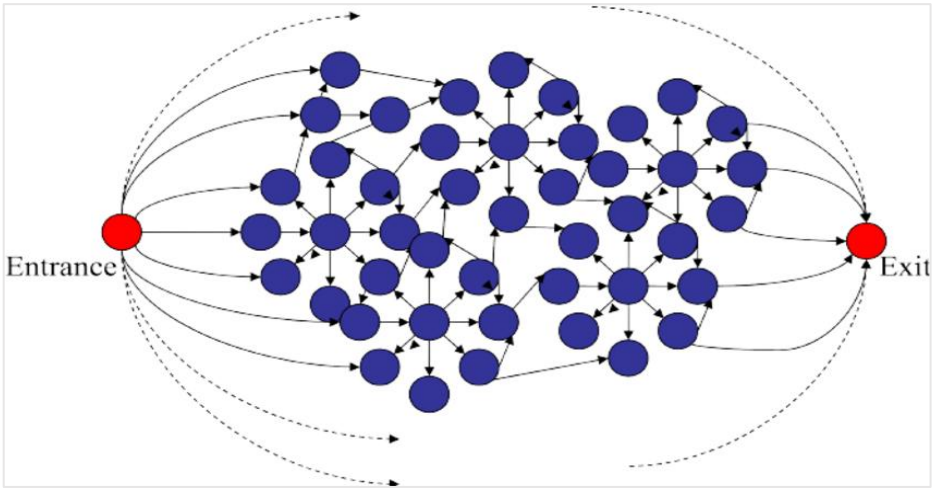

Рисунок 2.1 – Статичне представлення структури веб-сайту

Два спеціальних вузла повинні бути індивідуалізовані, оскільки вони не мають відповідності з будь-якою реальною сторінкою.

Це пояснюється тим, що вони представляють вихід / вхід до веб-сайту і кожен вузол складається з посилання на вузол "вихід або вхід". Це уявлення має перевагу явно включати весь перехід між вузлами, що корисно для стохастичних описів процесів.

Однак, цей опис веб-сайту можна розглядати як перше наближення реальної структури гіперпосилання. Поняття веб-сторінки, що складається зі статичного вмісту та унікальної URL-адреси, не може відповідати динамічному випадку. Веб-сайти постійно оновлюються, постійно змінюючи посилання та вміст, включаючи ті, які залежать від адаптаційних змін веб-сайту до веб-профілю користувача. На статичних сторінках з фреймами ця концепція також оскаржується, оскільки сторінка є складовою.

Розглядаючи сайти Web 2.0, остання модель здається застарілою. Проте, це наближення діє при деяких загальних обставинах. Інформаційні сайти, що регулярно оновлюються, подібно до тих, що подаються у газеті, можуть бути представлені в цьому графічному поданні протягом певного періоду часу. Блоги також можуть бути представлені як такі з урахуванням того, що відповіді все частіше додаються через те, що їхні вузли складаються з повідомлення та відповіді.

Взагалі, графічна структура залежних від часу різноманітних веб-об'єктів та їх асоціацій може бути визначена для сайтів загального призначення. Для застосування аналіз повинен бути пристосований до більш простої моделі залежно від конкретного веб-сайту.

Веб-контент. Відповідає семантичній інформації сприйняття користувачем веб-сайту кожної відвідуваної сторінки з веб-сайту. На більш ранній мережі Інтернет ці дані в основному відповідали змісту тексту. Сьогодні сайти Web 2.0 представляють набагато складнішу картину. Веб-контент є більш динамічним і складається з різноманітних медіа (текст, зображення, відео, вбудовані програми тощо). Веб-сторінки є композитами веб-об'єктів, які мають семантичні значення.

Обробка природної мови для семантичного вилучення була великою темою дослідження, коли почалися інформаційно-пошукові системи, і це залишається великою невирішеною проблемою для надійної та автоматичної операційної системи. Незважаючи на його обмеження, деякі наближення до проблеми були запропоновані на основі спеціального представлення тексту, а заходи подібності часом мають обґрунтовані результати. Замість вилучення точної семантики, поняття подібної семантики між текстами продемонструвало більш розумний

підхід. Крім того, динамічний контент передбачає залежність семантики від часу та користувача.

Оскільки веб-програми стають більш складними, ніж стандартне представлення вмісту, семантика стає більш неточною. Конкретний семантичний контекст, який групує різноманітність контенту, повинен бути пристосований для кожного конкретного застосування.

Сесія веб-користувача. Веб-користувач відвідує веб-сайт, представлений траєкторіями перегляду, що класифікується як сеанс.

Сеанс s – це послідовність $s = [(i_1, \tau_1), ..., (i_L, \tau_L)] \in S$ сторінок $i_k \in N$ та часу $\tau_k \in R^+$, витрачених web користувачем. Сеанс можна вважати контейнером для дій, які користувач виконує на вашому сайті.

Розмір $L = ||s||$ сеансу відповідає кількості вузлів без урахування одержувача даних та джерела. У цьому поданні час, пов'язаний як з вузлами джерела, так і з вузлами одержувача даних і тривалістю сеансу $T = P\ k\ \tau_k$ є сумою часу відвідування всіх відвідувачів сайту.

Проте, якщо сеанси не вказані явно, вони повинні бути реконструйовані з інших джерел, таких як веб-журнали. Коли вони спеціально витягуються, виникають певні проблеми конфіденційності, які ускладнюють його реалізацію. З іншого боку отримання сеансу з веб-журналів має менше проблем з конфіденційністю, оскільки дані зберігаються анонімно.

2.2 Основні джерела даних

Інтелектуальний аналіз даних є процесом вилучення неявної інформації та виявлення змістовних шаблонів, профілів і тенденцій з великих баз даних. Цей ітеративний процес виявляється цінною стратегією для розуміння активності користувачів в інтернеті.

Головним чином, існують чотири типи джерел даних, в яких дані про використання записуються на різних рівнях: рівень клієнта, рівень браузера, збір на рівні сервера та проксі.

Збір клієнтського рівня. На цьому рівні дані збираються разом за допомогою скриптів. Ці дані показують поведінку одного користувача на одному сайті. Збір даних на стороні клієнта вимагає участі користувача для ввімкнення скриптів або аплетів. Перевагою збору даних на стороні клієнта є те, що цей збір може захопити всі кліки, включаючи натискання кнопки назад або перезавантаження сторінки.

Колекція рівня браузера. Другий спосіб збору даних – це модифікація браузера. Він показує поведінку одного користувача на декількох сайтах. Можливості збору даних розширюються шляхом зміни вихідного коду існуючого браузера. Вони надають набагато більш різноманітні дані, оскільки розглядають поведінку одного користувача на декількох сайтах.

Збірка рівня сервера. Журнал веб-сервера зберігає поведінку декількох користувачів по одному сайту. Ці файли журналів можуть зберігатися в загальному форматі журналу або розширеному форматі журналу. Журнали сервера не можуть зберігати кешовані перегляди сторінок. Іншим методом, що використовується для збору даних на рівні сервера, є перехоплення пакетів TCP/IP. Пакетні переглядачі працюють шляхом моніторингу трафіку мережевої роботи і безпосереднього отримання даних про використання.

Журнали веб-сервера – це текстові файли (ASCII) і незалежні від сервера. Існують деякі відмінності між серверним програмним забезпеченням, але традиційно існує кілька типів серверних журналів: журнал доступу та журнал помилок. Запис в ці журнали ведеться в певних форматах. Common Log Format (CLF) створено для відстеження запитів, які відбуваються на веб-сайті в хронологічному порядку. Він містить IP-адресу клієнта, ім'я хоста, ім'я користувача, позначку часу, ім'я файлу та розмір файлу.

Збір на рівні проксі: проксі-сервери використовуються провайдером послуг інтернету для надання клієнтам доступу до World Wide Web. Ці сервери зберігають поведінку декількох користувачів на декількох сайтах. Такі серверні функції, як

кеш-сервер, можуть створювати кешовані перегляди сторінок. Прогнозуючи схему використання відвідувача інтелектуальний аналіз даних покращує якість послуг електронної комерції, персоналізує інтернет або підвищує продуктивність веб-структури та веб-сервера.

Дані, що знаходяться в лог-журналах, не можуть використовуватися для аналізу в тому вигляді в якому вони існують і повинні бути оброблені спеціальним чином.

Процес отримання даних про поведінку веб-користувачів можна розділити на три категорії. Перша – це попередня обробка, в процесі якої сесії веб-користувачів виводяться з джерел даних, якими є веб-журнали, які зберігають дії кожного відвідувача сайту. Такі файли можуть містити мільйони записів залежно від трафіку веб-сайту це і є головне джерело даних про поведінку людини.

Друга категорія це пошук шаблонів в даних. Цей процес здійснюється за допомогою стандартних методів інтелектуального аналізу даних, таких як пошук асоціативних правил або послідовних шаблонів.

На третьому етапі інформаційні фільтри які базуються на знаннях домену і структурах веб-сайту застосовуються до шаблонів аналізу в пошуках цікавих шаблонів. Зв'язки між сторінками та подібність між вмістом сторінок свідчать про те, що сторінки пов'язані між собою.

Фаза попередньої обробки [14] даних виконується з використанням перетвореного файлу журналу, який був очищений шляхом видалення всіх непотрібних, нерегулярних і відсутніх даних з оригінального загального файлу журналу. Після початкової попередньої обробки фільтр веб-сеансу використовують до перетвореного файлу журналу для вилучення ознак. Метою фільтра є агрегування всіх запитів користувачів у сесії в єдиний набір змінних.

Для виявлення цікавих шаблонів застосовуються статистичні методи, а також методи інтелектуального аналізу даних – аналіз шляхів, правило асоціації, послідовні структури та правила кластерів та класифікації.

Фаза видобування даних включає дві підфази: дескриптивний аналіз і аналіз за допомогою штучного інтелекту. Використовується описовий аналіз підбивання

підсумків, методів кластеризації та асоціативних правил для генерування набору даних, отримання уявлення про характеристики користувачів і описати основні шаблони поведінки користувачів. Аналіз за допомогою штучного інтелекту [18] використовується для прогнозування.

Фаза аналізу моделей включає інтерпретацію даних та оцінку результатів. Цей етап необхідний для визначення значущих результатів з результатів фази аналізу даних.

2.3 Попередня обробка даних

Дані для аналізу поведінки веб-користувачів отримані з джерел таких як наприклад журнал веб-сервера, потребують попередньої обробки.

На етапі очищення даних спочатку видаляються глобальні та локальні шуми. Глобальні шуми включають дзеркальні сайти, дубльовані веб-сторінки, попередні версії веб-сторінок та шумні слова. Місцеві шуми [11] включають нерелевантні пункти на веб-сторінці, такі як банерна реклама, навігаційна довідка, прикраси, графічні та відео формати.

Код статусу HTTP розглядається в наступному процесі очищення. Перевіряючи поле стану кожного запису в журналі, записи з кодом стану понад 299 або менш як 200 видаляються, оскільки дають не успішну відповідь серверу.

Видалення записів, що не відображають активність користувача. Веб-боти в автоматичному режимі переглядають безліч різних сторінок в мережі [12]. Їх поведінка сильно відрізняється від людського, і вони не представляють інтересу з точки зору аналізу використання веб-ресурсів.

Можна застосувати фільтрацію рядків користувача-агента, часто в їх ім'я може бути включений URL або e-mail адресу. Це, мабуть, найпростіший, але найменш надійний спосіб виявити, чи є він користувачем чи ні. Багато ботів як правило, підміняють агенти користувачів, а деякі роблять це з поважних причин

(тобто вони хочуть лише сканувати мобільний контент), а інші просто не хочуть бути ідентифіковані як боти. Ще гірше, деякі боти підміняють легітимних ботів, такі як агенти користувача Google, Microsoft, Lycos та інших сканерів, які зазвичай вважаються ввічливими. Далі, якщо швидкість перегляду перевищує поріг, ці запити також видаляються. За допомогою реалізації вищезазначених методів очищаються оригінальні файли журналів. Близько 50-60% невідповідних записів видаляються, що сприяє більш якісному результату аналізу поведінки веб-користувачів надалі.

Визначення кожного окремого користувача. Більшість порталів в мережі Інтернет доступні анонімним користувачам. Можна застосовувати інформацію про зареєстрованих користувачів, доступні файли cookie для визначення кожного користувача.

Ідентифікація сеансу користувача [17] – це процес сегментації журналу активності кожного користувача на сеанси, кожен з яких являє собою один візит на сайт. Мета евристики сеансу полягає в тому, щоб відновити з даних потоку кліків фактичну послідовність дій виконаних одним користувачем під час одного відвідування сайту.

Ідентифікація користувача за IP-адресою. Використовується для присвоєння унікальної адреси пристроям (комп'ютеру, принтерам і т.д.), що беруть участь у мережі. IP-адреса записується в журнал, коли користувач потрапляє на сторінку. Цю адресу можна використовувати для розрізнення різних користувачів. Але у випадку проксі-сервера, коли багато користувачів запитують певну сторінку, сервер веб-сайту реєструє той самий IP-адресу (IP-проксі-сервер) у лог-файл. Практично різні користувачі отримують доступ до цієї сторінки. Кешування також створює проблему для ідентифікації унікального користувача. Всякий раз, коли користувач намагається отримати доступ до попередньо переглянутої сторінки, сторінки браузера відображаються з локального кешу, і в журнал не входить запис.

Сеанс може бути ідентифікований атрибутом реферера в розширеному форматі журналу. Припустимо, що X і Y є двома запитами на послідовні сторінки одним і тим самим користувачем і (сеансом), якщо посилання на Y було викликано

раніше в цій сесії S, тоді Y буде додано в сеанс S, в іншому випадку нова сесія створюється з Y як перша запитувана сторінка.

Ідентифікація користувача топологією сайту. Цей метод використовує структурну топологію веб-сайту [13] для ідентифікації унікального користувача. Припустимо, що користувач запитує сторінку яка не доступна через попередньо запитані сторінки, тоді він розглядається як новий користувач. Це можна зробити, використовуючи атрибут referrer розширеного формату журналу та інформацію про посилання з топології сайту. Деякі ситуації коли цей підхід призводить до плутанини це якщо користувач робить запит використовуючи сторінки з закладками які не підключені через посилання.

Розглянуті методи предобробки специфічні виключно для даних веб-логів. Однак це не означає, що відомості вже готові до використання і побудови моделей. Далі необхідно провести звичайні кроки обробки, а саме: оцінка якості даних, відновлення пропущених значень, виявлення аномальних значень, нормалізація.

2.4 Моделі Маркова

Акт переглядання веб-сайту користувачем зазвичай моделюється шляхом спостереження за набором відвідуваних сторінок. Веб-сеанс (W) – це набір сторінок, які користувач відвідує, і він представлений послідовністю сторінок W (P1, P2,…, Pl), які були доступні. У цій послідовності Pi являє собою i-th сторінку у веб-сесії користувача.

Модель передбачення побудована на наборі веб-сеансів, які відносяться до навчального набору. Дана модель буде оцінена на точність на раніше невизначеному наборі веб-сесій, званих тестовим набором.

Існує декілька показників продуктивності, які слід враховувати для порівняння різних методів, заснованих на марківській моделі, для вирішення проблеми прогнозування поведінки користувача.

Точність моделі є першим показником. Точність вимірює прогностичну здатність моделі і визначається за допомогою окремого набору веб-сеансів, які не використовувалися під час навчання. Одним із способів цього є приховування останньої сторінки в кожній з веб-сесій тестового набору і використання моделі для прогнозування результуючої обрізаної веб-сесії.

Точність визначається як відношення кількості веб-сесій, для яких модель здатна правильно передбачити приховану сторінку загальної кількості веб-сесій у тестовому наборі.

Другим показником ефективності, який слід враховувати, є кількість станів моделі (див. рис. 2.2), яка вимірює простір і складність навчання, а також застосування моделі.

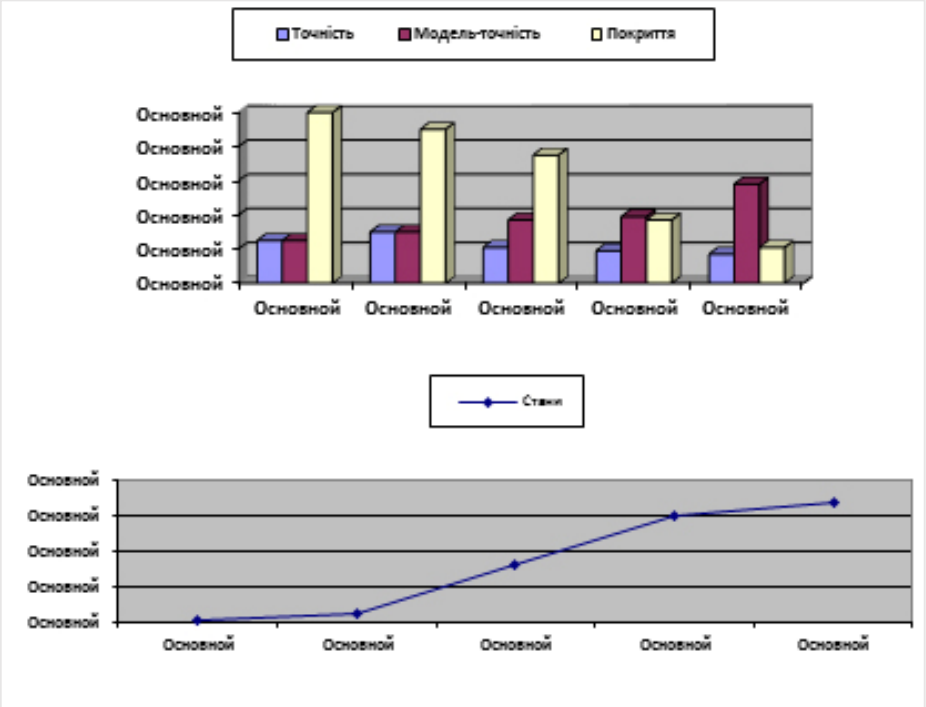

Рисунок 2.2 – Порівняння точності, покриття, точності моделі та розміру моделі з порядком марківської моделі

Модель, яка вимагає великої кількості станів, може істотно обмежити застосовність марківських моделей для додатків, в яких швидкі прогнози є критичними для продуктивності в реальному часі або для додатків з обмеженнями

пам'яті. Число станів марківської моделі визначається як загальна кількість станів, для яких марківська модель оцінила найбільш ймовірну сторінку, до якої слід звернутися.

По-третє, охоплення моделі, яка вимірює кількість разів, коли модель Маркова була в змозі обчислити прогноз, не вдаючись до передбачення за замовчуванням. Охоплення марківської моделі оцінюється на тестовому наборі. Вона визначається як відношення кількості веб-сесій, стан яких, необхідний для внесення прогнозу, було знайдено в моделі до загальної кількості веб-сесій у тестовому наборі.

Останньою метрикою є точність моделі, яка визначається як точність частини тестового набору, для якої модель змогла визначити стан, необхідний для прогнозування.

2.5 Обмеження традиційних марківських моделей

Традиційні марківські моделі використовуються для прогнозування наступної веб-сторінки, якою користувач, швидше за все, отримає доступ, зіставляючи поточну послідовність доступу користувача з історичними послідовностями веб-доступу користувача. Використовуючи моделі, дослідники порівнюють елементи максимальних префіксів кожної історичної послідовності веб-доступу з елементами з суфіксів однотипної поточної послідовності доступу користувача та отримують історичні послідовності з найбільшою ймовірністю відповідних елементів.

Властивості моделей:

− нехай $X = X_1X_2\ldots X_L$ це послідовність сторінок;

− марківська модель 0 порядку безумовна ймовірність базової швидкості. Таким чином, ймовірність прогнозування наступної сторінки не залежить від будь-яких (нульових) сторінок, що передують цьому;

− pr (X_i | X_1X_2… X_i-1) = Pr (Xi);

− марківська модель першого порядку розглядає ймовірності переходу між сторінками pr (X_i | X_1X_2… Xi-1) = Pr (X_i | X_i-1);

− модель послідовності Маркова з K-th порядком розглядає умовну ймовірність того, що користувач переходить на і-ту сторінку з урахуванням своїх попередніх відвідувань k = i - 1 сторінок;

− pr (X_i | X_1X_2… Xi-1) = pr (Xi | X_i-1X_i-2… Xi-k).

Для прикладу, марківська модель 0-порядку має 5 станів (тобто (p1), (p2), (p3), (p4), (p5)), тоді як марківська модель першого порядку має 8 стану (тобто (p1, p2), (p1, p3), (p2, p1), (p3, p1), (p2, p3), (p3, p4), (p3, p5), (p4, p3))).

На жаль, марківські моделі нижчого порядку (наприклад, перше та / або другого порядку) не дуже точні у прогнозуванні майбутнього доступу до веб-сторінок, оскільки ці моделі не виглядають досить далеко в минуле щоб правильно дискримінувати поведінкові режими користувачів.

Таким чином, гарні прогнози вимагають моделей вищого порядку (наприклад, третій, четвертий порядок).

Модельні стани вищого порядку є різними комбінаціями дій, що спостерігаються в наборі даних, тому число станів має тенденцію до зростання в геометричній прогресії при збільшенні порядку моделювання. Таким чином, моделі високого порядку є надзвичайно складними завдяки їхній великій кількості станів, що підвищує вимоги до простору та часу виконання.

Можна побачити, що по мірі збільшення порядку моделі точність моделі зменшується, що супроводжується зменшенням покриття. Однак точність моделі продовжує зростати. Це вказує на те, що хоч більш висока модель може знаходити стани лише для невеликого набору веб-сесій, вони можуть передбачати ці веб-сесії з більшою точністю, ніж моделі нижчого порядку. Варто відзначити, що по мірі збільшення порядку моделі кількість станів, що використовуються для моделі, також різко зростає. Це пояснюється тим, що стани моделей вищого порядку є не що інше, як різні комбінації дій, що спостерігаються в наборі даних.

Це різке збільшення кількості станів може істотно обмежити застосовність марківських моделей для додатків, в яких швидкі прогнози є критичними для продуктивності в реальному часі або в додатках, де обмеження пам'яті є жорсткими (наприклад, вбудовані моделі для кешування веб-доступу, в тестовому наборі може бути багато прикладів, які не мають відповідних станів у більш високій марковській моделі, таким чином, зменшуючи їх покриття. У таких сценаріях моделі вищого порядку повинні робити прогноз за замовчуванням, що може призвести до зниження точності.

Щоб краще зрозуміти ці недоліки, проведемо експеримент на одному з веб-наборів даних. Порівнюючи різні марківські моделі вищого порядку, починаючи з першого шляху до марківської моделі п'ятого порядку.

У наведених результатах (див. рис. 2.3) порядок моделі збільшує точність, що супроводжується зменшенням покриття.

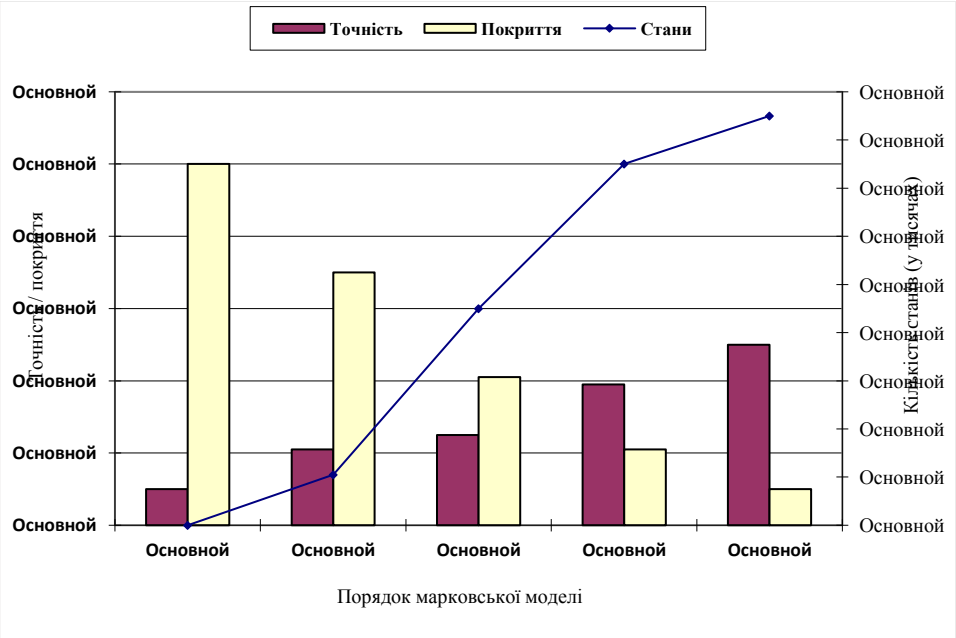

Рисунок 2.3 – Графік порівняння точності, покриття та розміру моделі з порядком марківської моделі

Якщо модель вищого порядку не змогла зробити прогноз для певної веб-сесії, вона була проігнорована з розрахунків точності (тобто точність обчислюється тільки на тих прикладах, на яких було зроблено прогноз).

У той же час, коли порядок моделі збільшується, кількість станів, що використовуються для моделі, також різко зростає.

Одним із способів подолання проблеми низького охоплення на тестовому наборі є підготовка марківських моделей різного порядку, а потім їх об'єднання для прогнозування. У цій схемі для кожного тестового екземпляра для прогнозування використовується марківська модель найвищого порядку, що охоплює примірник. Ця схема називається марківською моделлю «All-Kth-порядку». Незважаючи на те, що ця марківська модель вирішує проблему низького охоплення, на жаль, вона посилює проблему розміру моделі, оскільки стани всіх марківських моделей різного порядку тепер є частиною моделі.

Оскільки число станів у марківських моделях вищого порядку збільшує число екземплярів навчальних наборів, що використовуються для обчислення ймовірностей переходу стану-дії для кожного з станів, має тенденцію до зменшення. Наслідком цього зменшення підтримки окремих станів є те, що деякі з оцінених ймовірностей переходу між станами і діями не є точними.

Гібридна деревоподібна марківська модель (див. рис. 2.4). Як згадувалося раніше, одним з недоліків марківських моделей вищого порядку для прогнозування є те, що їх розмір швидко зростає з їхнім порядком. Цю проблему можна вирішити за допомогою деревоподібної марківської моделі (TMM).

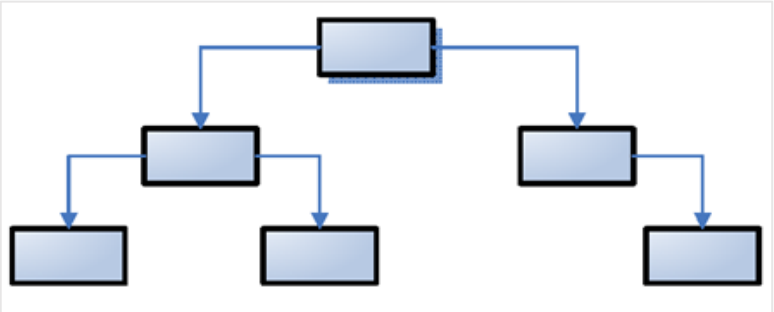

Рисунок 2.4 – Структура деревоподібної марківської моделі

K-порядок TMM є багатокористувацьким деревом, сформованим з історичної бази даних послідовності доступу до Web DsK+1 +, де кожен вузол являє собою відвідану сторінку, і кожна гілка є шлях до вузла.

Кожен вузол записує підрахунок, тобто кількість разів, коли користувач відвідав вузол по тому ж маршруту. Завдання K-порядку TMM полягає в реєстрації всієї інформації веб-доступу. Таким чином, його висота (k + 2), а її ширина не більше, ніж кількість послідовностей в базі даних. Таким чином, існуючий K-порядок TMM не може генерувати велику кількість вузлів.

Як результат, загальна точність, досягнута марківськими моделями вищого порядку, іноді може бути меншою, ніж досягається відповідними марківськими моделями нижнього порядку.

Для подолання обмежень марківських моделей вищого порядку навчаються марківські моделі різного порядку, а потім комбіновані для прогнозування.

Деревоподібні марківські моделі (HTMM) із гібридним порядком пропонують два методи поєднання моделей для прогнозування поведінки користувача.

Перший – це метод точності голосування, а другий – метод, що використовується для стиснення даних.

Метод точності голосування складається з трьох кроків:

− налаштування 1~N-порядку TMM з веб-журналів;

− витяг 1~N-суфіксів з поточної послідовності доступу користувача (тобто 1~N послідовність) до 1~N-порядкового TMM для прогнозування. Вибір сторінки прогнозування з максимальним рахунком як результат прогнозування, а потім отримання набору прогнозів PS = {P1, P2,…, PN}, де Pk ($1 \le k \le N$) є результатом прогнозування K-порядку TMM;

− обчислення параметру прогнозування сторінок у PS за допомогою (2.1). Вага прогнозування обчислюється з використанням для кожного результату прогнозування TMM різного порядку.

Метод змішування моделі. Змішування можна розглядати як рекурсивну процедуру знизу вгору для обчислення суміші, де суміш є в основному середньозваженим кількох оцінок ймовірностей. Представлена точність між HTMM і традиційними марківськими моделями (див. рис. 2.5).

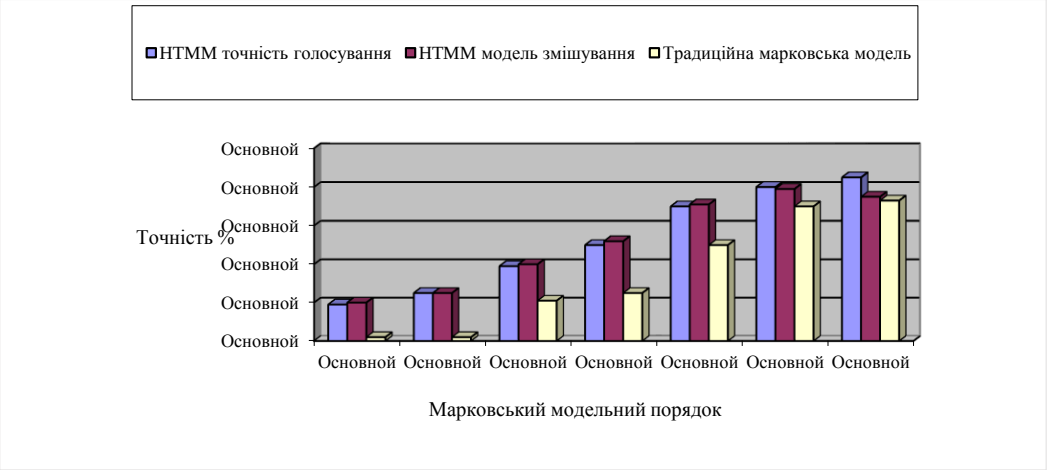

Рисунок 2.5 – Порівняння точності між HTMM і традиційними марківськими моделями

В кінцевому результаті, сторінки сортуються за параметрами прогнозування.

2.6 Вибіркові моделі Маркова

У випадку селективних марківських моделей (SMM) відправною точкою є також марківська модель All-Kth-Order, отримана шляхом побудови послідовності зростаючих порядкових моделей Маркова.

Замість того, щоб використовувати цю модель для прогнозування у цьому випадку для усунення певних станів у різних марківських моделях використовується ряд методів.

Сукупність станів, які вижили на цьому кроці, потім стають кінцевою моделлю, яка використовується для прогнозування. Метою цієї ліквідації станів є зменшення складності стану та підвищення точності прогнозування отриманої моделі.

Для SMM алгоритм прогнозування аналогічний алгоритму Марківської моделі All-Kth-Order. Для даної послідовності сторінок, спочатку ми повинні ідентифікувати всі стани в моделі, які можна використовувати для прогнозування

наступної сторінки, а потім використовувати серед них найвищий стан для виконання прогнозування. Ключовим кроком у алгоритмі прогнозування SMM є схема, що використовується для визначення потенційної точності конкретного стану.

Для розширення рівня складності були розроблені три різні схеми. Перша схема, названа частотно-обрізаною марківською моделлю (FPMM), виключає стани що мають дуже низьку частоту зустрічальності.

Друга схема, модель впевненого обрізання Маркова (CPMM), використовує техніку для ідентифікації станів, де умовні ймовірності двох найбільш помітних сторінок істотно не відрізняються. При цьому такі стани обрізаються.

Нарешті, третя схема, модель з помилковим обрізанням (EPMM), використовує підхід на основі оцінювальної помилки для усунення станів з низькою точністю прогнозування.

Частотно-обрізана марківська модель (FPMM). Основним завданням є спостереження за станами, які відбуваються з низькою частотою в навчальному наборі. Таким чином, ці стани мають тенденцію до низького прогнозування.

Кількість усунених станів у схемі FPMM перевіряється параметром φ, який називається частотним порогом. Використовуючи цей параметр, FPMM виключає всі стани різних марківських моделей k-го порядку для k>1, які відбуваються в меншій кількості, ніж екземпляри навчальних наборів.

Важливо помітити деякі зауваження щодо FPMM:

− той самий частотний поріг використовується для всіх моделей незалежно від їхнього порядку;

− політика обрізання, швидше за все, скорочуватиме стани високого порядку, оскільки більш високі стани відбуваються з меншою частотою;

− пороговий параметр частоти φ визначає фактичну кількість екземплярів навчальних наборів, які повинні підтримуватися кожним станом, а не частки екземплярів навчальних наборів, як це часто робиться в контексті виявлення правила асоціації.

Одним з обмежень схеми FPMM є те, що він не охоплює всі параметри, які впливають на точність стану (тобто розподіл ймовірностей вихідних сторінок зі стану повністю ігнорується).

Наступна представлена модель розглядає не тільки частоту виникнення стану, але й зважує розподіл ймовірностей сторінок перед прийняттям рішень обрізки.

Марківська модель впевненого обрізання (CPMM) визначає для кожного стану, якщо ймовірність найбільш часто використовуваної сторінки істотно відрізняється від ймовірності інших сторінок, які можуть бути доступні з цього стану.

Якщо ймовірнісні відмінності не є значущими, то цей стан навряд чи дасть високу точність і його обрізають. На відміну від цього, якщо вірогідні відмінності є значними, стан зберігається.

Якщо різниця ймовірностей (2.1) між найбільш ймовірною сторінкою і другою найбільш ймовірною сторінкою перевищує певний поріг – поріг довіри φc, стан зберігається. На відміну від цього, якщо різниця ймовірностей між двома найбільш ймовірними сторінками нижче порогу довіри, стан зменшується.

$$\Phi_c = P - Z_{\alpha/2}\, p(1-p)/\, n \qquad (2.1)$$

де P – ймовірність найбільш ймовірної сторінки;

$z_{\alpha/2}$ – верхня $_{\alpha/2}$ процентна точка стандартного нормального розподілу;

n – частота стану Маркова.

Ступінь усунення станів в CPMM контролюється zα/2(коефіцієнт довіри). При збільшенні значення $z_{\alpha/2}$ зменшується розмір довірчого порогу φc, що призводить до збільшення обрізання.

Важливо відзначити, що навіть якщо різниця в ймовірностях між двома найбільш ймовірними сторінками є відносно невеликою, стан, швидше за все, буде збережено. Це означає, що в тих випадках, коли стан має велику кількість вихідних

сторінок, навіть невеликі уподобання щодо однієї з цих сторінок передає важливу інформацію.

Модель Маркова з помилковим обрізанням (EPMM). Крок перевірки є широко використовуваним підходом для оцінки помилки, пов'язаної з кожним станом. Під час етапу перевірки всю модель перевіряють, використовуючи частину навчального набору, який не використовувався під час етапу побудови моделі.

У EPMM кінцеві прогнози обчислюються за допомогою лише станів моделі, які мають найменшу оцінену помилку. Незважаючи на те, що EPMM також використовує марківську модель All-Kth-Order, як FPMM та CPMM, вона відрізняється від цих двох схем у способі обрізки.

В обох схемах FPMM та CPMM один параметр (φ чи φc) обчислюються і вся марківська модель скорочується за допомогою цього параметра. Проте у випадку ЕПММ стан вищого порядку скорочується шляхом порівняння його частоти помилок з частотою помилок його станів нижчого порядку.

Розроблено дві стратегії обрізки на основі помилок. Обидва ці методи слідують аналогічному підходу обрізання, але відрізняються тим, як обчислюється рівень помилок для кожного стану.

У першій схемі, яка називається загальним обрізанням помилок, кожне стан Маркова нижнього порядку має єдине значення швидкості помилки, яке обчислюється по всьому набору перевірок. У другій схемі, відомій як обрізання індивідуальної помилки, кожна з нижніх станів має багато частот помилок, пов'язаних з нею, по одному для кожного з відповідних станів вищого порядку.

2.7 Приховані моделі Маркова

Приховані марківські моделі (HMM) є розширенням ланцюгів Маркова, в яких стан ланцюга не є безпосередньо доступним і існують лише неточні стохастичні спостереження (див. рис. 2.6), які надають інформацію про стан.

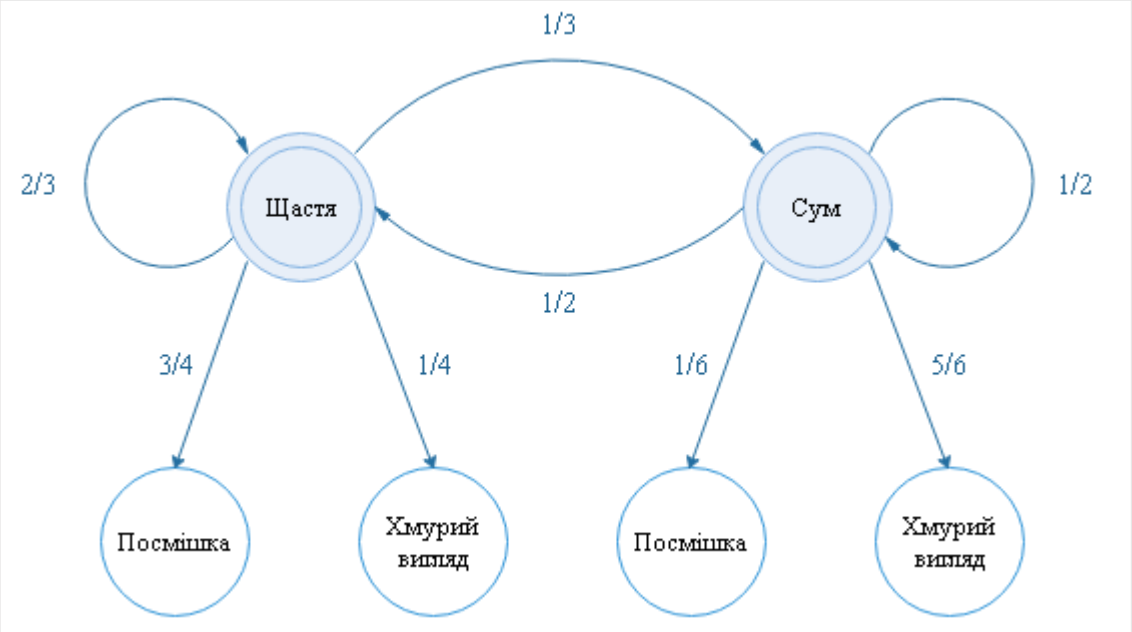

Рисунок 2.6 – Приклад прихованих моделей Маркова

Приховані марківські моделі (ПММ) можуть бути використані для моделювання більш широких наборів проблем, ніж моделі Маркова, оскільки вони не припускають, що ми знаємо держави, що більше схоже на те, що ми зустрічаємо в реальному світі. Тепер ми розглянемо основні поняття і алгоритми, пов'язані з ПММ, на основі класичного підручника від Rabiner.

ПММ складається з N станів. На кожному дискретному часу t процес знаходиться в певному стані. Через марківську властивість стан на етапі часу t визначається стохастичне на основі стану на попередньому етапі часу t - 1, але не залежить від будь-яких станів перед t - 1.

Існує розподіл ймовірностей переходу, що стосується попереднього стану до поточного. Існує спостереження Ot на кожному кроці часу, який залежить від стану Q_t. Для кожного можливого стану існує розподіл ймовірностей, який генерує спостереження.

ПММ можна уявити як процеси з двома «рівнями». Існує ланцюг Маркова (перший рівень), і кожен стан генерує випадкові емісії. Ключовим є те, що ланцюг Маркова не спостерігається, але емісії спостерігаються. Розглянемо це на прикладі

з «емоціями»: визначимо стан в ланцюгу Маркова подвійними колами, а емісії – як окремі кола.

Ребра, як звичайно, нотуються з вірогідністю, що ми слідуємо за кожним краєм. Тут ми маємо «емоційні стани» – щастя і сум і «реакції» посмішку і нахмурювання, як емісії. На підставі встановленого HMM ми говоримо, що емоційні стани не спостерігаються, але реакції спостерігаються.

Тобто, ми можемо побачити, чи людина посміхається або хмуриться, але ми не можемо спостерігати, якщо вони щаслива або сумна. У цьому конкретному прикладі ймовірності інтуїтивні: щаслива людина, швидше за все, посміхається, ніж нахмурилася (ймовірності 3/4 і 1/4), і так само сумна людина, швидше за все, похмуриться, ніж посмішка (ймовірність 5/6 і 1/6).

Використовуємо наступні позначення:

- T: кількість кроків часу в послідовності спостережень;
- N: кількість можливих станів;
- M: кількість можливих спостережень;
- S: безліч N можливих станів;
- V: безліч M можливих спостережень;
- Qt: випадкова величина, що позначає стан на етапі часу t;
- Ot: випадкова величина, що позначає спостереження на етапі часу T;
- A: матриця ймовірності переходу розміром N × N, де aij = P (Qt + 1 = j | Qt = i) - ймовірність переходу з i-го стану в j-й стан, з i та j між 1! і N;
- B: матриця ймовірностей спостереження розміром N × M, де bjk = P (Ot = k | Qt = j) - ймовірність k-го спостереження в j-му стані, з k = 1. . . M ij = 1. . . N;
- π: початковий розподіл стану, який є вектором розміру N, з πi = P (Q1 = i) = O1. . . OT: послідовності спостережень;
- Q = Q1. . . Qt: послідовність станів.

ПММ складається з кортежу λ=(A,Bπ). ПММ генерує спостереження наступним чином. Для кроку часу t=1 початковий стан Q_1 виводиться з початкового розподілу стану π.

У кожному кроці часу t спостереження O_t породжується розподілом ймовірностей спостереження для стану Qt відповідно до відповідного рядка B. Стани для кожного етапу часу t>1 відбираються з розподілу ймовірності переходу для попереднього стану, як зазначено у відповідному рядку матриці A.

Існують три основні проблеми для ПММ. Перша проблема полягає у знаходженні ймовірності P (O) спостереження.

Другою проблемою є знаходження найбільш ймовірної послідовності станів для наведеної послідовності спостережень: $\arg\max_Q P(Q|O)$.

Третя проблема полягає у виборі параметрів моделі, які максимізують ймовірність набору послідовностей спостережень.

2.8 Опис проведених теоретичних і експериментальних досліджень

Застосування марківської моделі прийнято Mukund Deshpande та George Karypis (2004) до веб-сторінок для обчислень і передбачення мети, а також були широко використані для прогнозування дії, яку користувач зробить наступним, з урахуванням послідовності дій, які він чи вона вже виконував. Корисним також є внесок Peter Pirolli (1996).

Для цього типу досліджень марківські моделі представлені трьома параметрами:

− набір всіх можливих дій A, які може виконувати користувач;

− це множина всіх можливих станів S, для яких побудована марківська модель;

− матриця ймовірності переходу (Transition Probability Matrix TPM) T, де кожен запис Tij відповідає ймовірності виконання дії j, коли процес знаходиться в стані i.

Простір стану марківської моделі залежить від кількості попередніх дій, що використовуються при прогнозуванні наступної дії. Найпростіша марківська

модель передбачає наступну дію, лише дивлячись на останню дію, виконану користувачем.

У цій моделі, також відомої як марківська модель першого порядку, кожна дія, яка може виконуватися користувачем, відповідає стану в моделі.

Дещо складніша модель обчислює прогнози шляхом перегляду останніх двох дій, виконаних користувачем. Це називається другою порядковою марківською моделлю, а її стани відповідають всім можливим парам дій, які можуть виконуватися в послідовності.

Цей підхід узагальнено до марківської моделі Kth-порядку, яка обчислює передбачення, розглядаючи останні K-дії, що виконуються користувачем, приводячи до простору стану що містить всі можливі послідовності дій K. Припустімо, що прогнозування наступної сторінки, доступної користувачеві на веб-сайті, є проблемою.

Вхідні дані для побудови марківських моделей складаються з веб-сесій, де кожен сеанс складається з послідовності сторінок, доступ до яких здійснюється користувачем під час його відвідування сайту.

У цій задачі дії для моделі Маркова відповідають різним сторінкам веб-сайту, а стани відповідають всім послідовним сторінкам довжини K, які спостерігалися в різних сесіях.

У випадку моделей першого порядку, стани будуть відповідати одній сторінці, у випадку моделей другого порядку, стани будуть відповідати всім парам послідовних сторінок і так далі.

Після виявлення станів марківської моделі можна обчислити матрицю ймовірностей переходу. Є багато способів, за допомогою яких TPM може бути побудований.

Найбільш часто використовуваний підхід полягає у використанні навчального набору дій-послідовностей і оцінки кожного T_{ji}-запису на основі частоти події, що дія ai слідує за станом sj.

Наприклад, розглянемо веб-сесію WS2 (P3; P5; P2; P1; P4), (див. рис. 2.7).

1st Order	P1	P2	P3	P4	P5
S1={P1}	0	0	0	2	1
S2={P2}	4	0	0	0	1
S3={P3}	0	1	0	1	1
S4={P4}	0	1	0	0	2
S5={P5}	0	3	0	2	0

2nd Order	P1	P2	P3	P4	P5
{P1;P4}	0	1	0	0	0
{P1;P5}	0	0	0	1	0
{P2;P1}	0	0	0	1	1
{P2;P5}	0	0	0	1	0
{P3;P2}	1	0	0	0	0

2nd Order	P1	P2	P3	P4	P5
{P2;P5]	0	1	0	0	0
{P2;P4}	0	0	0	0	1
{P4;P5}	0	2	0	0	0
{P5;P2}	3	0	0	0	0
{P3;P4}	0	0	0	0	1

Рисунок 2.7 – Зразок веб-сесій з відповідними матрицями ймовірностей переходу 1-го і 2-го порядків

Якщо використати марківську модель першого порядку, то кожен стан складається з однієї сторінки, тому перша сторінка P3 відповідає до стану s3. Оскільки сторінка p5 слідує за станом s3, запис t35 в TPM буде оновлено. Аналогічно, наступний стан буде s5, а запис t52 буде оновлено в TPM. У випадку моделі вищого порядку кожен стан буде складатися з декількох дій, тому для моделі другого порядку перший стан для веб-сесії WS2 складається зі сторінок {P3; P5} і оскільки сторінка P2 слідує за станом {P3; P5} в веб-сеансі запис TPM, відповідний стану {P3; P5} і сторінка P2 буде оновлена.

Як тільки матриця ймовірності переходу побудована, прогноз для веб-сеансів є прямим. Наприклад, розглянемо користувача, який має доступ до сторінок {P1; P5; P4}. Якщо ми хочемо передбачити наступну сторінку, до якої користувач звернеться далі, використовуючи модель першого порядку, спочатку визначимо стан s4, який пов'язаний зі сторінкою P4, і шукаємо TPM, щоб знайти сторінку pi, яка має найвищий рівень ймовірності і передбачити її. У нашому прикладі прогноз буде на сторінці P5.

Веб-сесії:

– WS1: {P3; P2; P1};

– WS2: {P3; P5; P2; P1; P4};

– WS3: {P4; P5; P2; P1; P5; P4};

– WS4: {P3; P4; P5; P2; P1};

– WS5: {P1; P4; P2; P5; P4}.

Кластеризація (див. рис. 2.8) є однією з найважливіших проблем в аналізі.

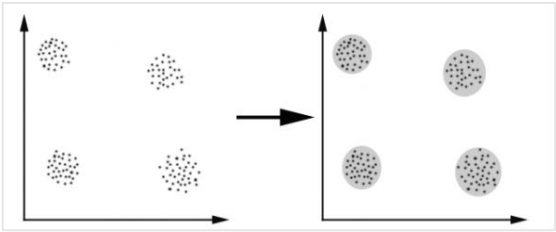

Рисунок 2.8 – Представлення кластера

Отже, просте визначення кластеризації [16] може бути «процесом організації об'єктів у групи, члени яких певною мірою подібні».

Таким чином, кластер є сукупністю об'єктів, які «подібні» між собою і «несхожі» на об'єкти інших кластерів. Ми можемо показати це простим прикладом.

Оскільки веб-сторінки мають вміст, тому кожна сторінка має ідентичність таку як кількість конкретних ключових слів і користувач намагається переміщатися відповідно до вмісту. Наприклад, якщо користувач прочитав певний абзац на веб-сторінці, то наступна сторінка, яку він відвідує, може бути пов'язана зі змістом ключових слів поточної сторінки. Таким чином, ключові слова сторінки діють як особливість веб-сторінки. Іншою особливістю веб-сторінки є сеанс користувача як шаблон для досягнення будь-якої сторінки. Ці функції використовуються для прогнозування наступної веб-сторінки, бо з точки зору ключових слів які користувач знаходить на попередніх сторінках, він може також шукати деякі відносні сторінки.

Таким чином, збір ключових слів здійснюється шляхом зчитування вмісту веб-сторінок в Bag of Words (BOW), а потім попередньої обробки даних шляхом видалення стоп-слова. Для коректної роботи необхідний фільтр, оскільки BOW може мати велику кількість непотрібних слів які видаляються, встановлюючи поріг

частоти слова, що з'являється на веб-сторінці. Треба видалити всі слова які знаходяться за межами діапазону. Таким чином, за допомогою цього фільтра BOW нарешті містить набір ключових слів.

Запропонований підхід. Модуль 1. Сесія веб-користувача збирається у вигляді вектора сеансу з веб-сайту (див. рис. 2.9), прогнозування веб-сторінок якого необхідно зробити.

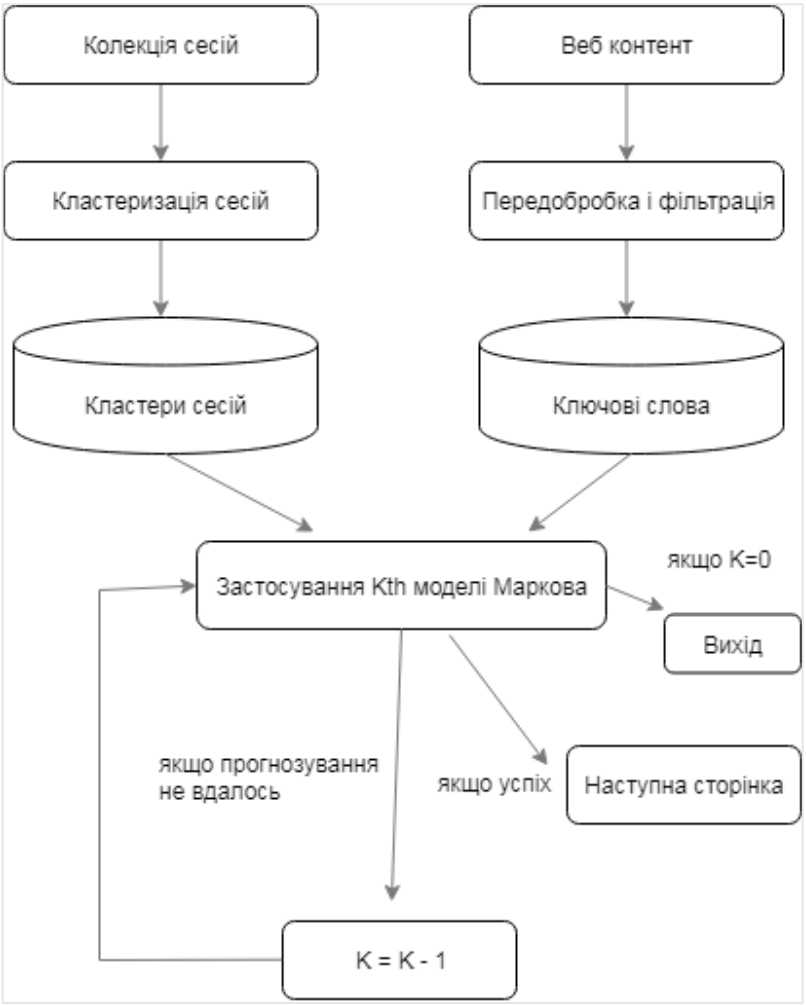

Рисунок 2.9 – Схема алгоритму прогнозування веб-сторінок

Оскільки веб-сеанс перебуває в порядку в якому був переглянутий користувачем або тимчасовим порядком. Таким чином, виконується кластеризація сесії ієрархічним підходом, оскільки використовується послідовність веб-сторінок. Для того щоб підтримувати марківську модель, один кластер виконується як одна і та ж перша сторінка сеансу, потім другий створюється як одна і та ж перша сторінка

сеансу, після чого третій кластер створюється у вигляді однієї першої трьох сторінок сеансу. Четвертий кластер створюється за тією ж логікою.

Ці кластеризації зменшать завдання пошуку даних, оскільки кількість сеансів розділяється в кожному кластері, тому загальний час порівняння зменшується, а робота буде більш тонка, оскільки кластер продовжуватиме більше подібний сеанс для прогнозування.

Модуль 2. Для поліпшення ймовірності веб-прогнозування введена ще одна особливість, яка являє собою набір ключових слів для кожної сторінки веб-контенту, який буде підтримувати прогнозування наступної сторінки з вмістом веб-користувача. Спочатку треба зібрати вміст веб-сторінки, а потім зробити попередню обробку, оскільки вона значно зменшує розмір вхідних текстових документів. Попередня обробка містить в собі такі дії, як визначення кордону аналізу, усунення стоп слів специфічних для даної мови. Стоп-слова – це функціональні слова не корисні для класифікації, які часто зустрічаються у тексті. На цьому етапі читаємо весь проект і вкладаємо всі слова у вектор. Тепер знову прочитаємо файл, що містить стоп-слова, після чого видалимо подібні слова з вектора. Після того, як дані будуть попередньо оброблені, це буде набір слів, які можуть бути в списку онтологій. Наприклад, приймемо один папір класу зображення, а його текстовий вектор $\{a_1,f_1,s_1,a_2,s_2,a_3,a_4,f_2,...\}$, а колекція стоп-слів $\{a_1,a_2,a_3,...,a_x\}$. Тоді вектор отримують після попередньої обробки $\{f_1,s_1,s_2,f_2,...,f_x\}$.

Далі передаємо цей вектор з фільтра, щоб уточнити зміст вектора, встановивши верхню межу і нижню межу частоти слів, які з'являються в змісті. Таким чином вектор має колекцію ключових слів по сторінках.

Сеанс користувача який необхідно передбачити використовується для збору слів користувачів, що представляють для нього інтерес. Це є збіркою слів які отримуються з попередніх сторінок. Наступним кроком сторінки, які передбачені за допомогою марківською моделі, фільтруються по сторінці, на якій більшість слів до яких є інтерес у користувача. Таким чином, ця функція підтримує прогнозування веб-сторінки.

Модуль 3. У цьому модулі, всі етапи попередньої обробки, групування вже зроблено в попередніх модулях. Ми генеруємо всі порядки марківських моделей і використовуємо їх колективно в прогнозі. Зауважимо, що передбачається, що функція predict (x, mk) передбачає наступну відвідану сторінку сесії x за допомогою kth порядку моделі Маркова mk. Якщо mk виходить з ладу, mk – 1 розглядається з використанням нового сеансу x' довжини k–1, де x' обчислюється шляхом видалення ID першої сторінки в x. Цей процес повторюється до тих пір, поки не буде отримано прогнозування або прогнозування не відбудеться.

Наприклад, з урахуванням сесії користувача x = <P1, P5, P6>, передбачення моделі Kth виконується шляхом використання моделі Маркова третього порядку. Якщо прогнозування з використанням моделі Маркова третього порядку не вдається, то марківська модель другого порядку розглядається на сесії x_ = x – P1 = <P5, P6>. Цей процес повторюється до досягнення марківської моделі першого порядку. Тому, на відміну від основної марківської моделі, марківська модель all-Kth-order досягає кращого прогнозування, і це лише неможливо коли всі порядки основних моделей Маркова зазнають невдачі і не можуть передбачити.

Алгоритм прогнозування веб-сторінок:

− сесія веб-користувача збирається у вигляді вектора сеансу з веб-сайту;

− виконується кластеризація сесії ієрархічним підходом;

− введення особливості – набір ключових слів для кожної сторінки веб-контенту. Збирається зміст кожної веб-сторінки на предмет ключових слів;

− попередня обробка і фільтрація отриманого контенту;

− встановлюємо верхню межу і нижню межу частоти слів, які з'являються в змісті. Таким чином вектор має колекцію ключових слів по сторінках;

− генеруємо всі порядки марковських моделей і використовуємо їх колективно в прогнозі;

− якщо прогнозування з використанням моделі Маркова вищого порядку не вдається, то марківська модель розглядається з використанням нового сеансу довжини K–1. Тобто зниженням порядку моделі;

− процес повторюється до досягнення марківської моделі першого порядку або коли прогнозування не відбувається.

Алгоритм Predict_markov приймає сеанс і модальне число як вхідні дані, потім знаходить найчастішу сторінку. Якщо він згенерує більше однієї сторінки, тоді буде передбачено другу функцію для вибору сторінки, яка є ключовими словами, витягнутими з веб-сторінок. Існує подібна функція key_vector, яка є колекцією ключових слів, яка отримується з попередньої сторінки сеансу, потім порівнюються ключові слова сторінок у V вектор. Найбільш подібна сторінка буде наступною цільовою сторінкою сесії. Ця сторінка повертається до функції.

Результати експерименту. Для того, щоб передбачити нову сторінку веб-сесії користувача, я розглянув набори даних та попередню їх обробку. Було розглянуто два набори даних, а саме, набір даних NASA, який створюється штучно веб-сайтом для створення цієї роботи.

На додаток до багатьох інших елементів, попередня обробка набору даних містить в собі наступне: групування сеансів, визначення початку і кінця кожного сеансу, призначення унікального ідентифікатора сеансу для кожної сесії, а також фільтрація нерелевантних записів деталей набору даних (див. табл. 2.1).

Таблиця 2.1 – Резюме набору даних

	NASA	Штучні
Всього сесій	50,000	20,000
Середня тривалість сесії	6.4	4.5
Кількість сторінок	2266	16
Час датасета	серпень 1995	свій

У цьому експерименті кроки попередньої обробки та методи ідентифікації сеансу виконуються у двох модулях кластеризації та створення векторів ознак ключових слів. Реалізація як вектора Маркова так і ключових слів для моделей прогнозування веб-сторінок виконується коли веб-сесії згруповані в різні групи, а потім збираються ключові слова з веб-контенту, щоб зробити ключові слова

вектором. Після того, як цей крок зроблено, тренують модель з різним співвідношенням набору даних, наприклад 60% для навчання і 40% для тестування, або 67% для навчання і 33% для тестування модального.

Щоб виміряти точність, потрібно розділити кожен набір даних випадковим чином на тренувальний набір (дві третини вихідного набору) і тестовий набір (одна третина початкового набору). Точність узагальнення є стандартною процедурою, яка широко використовується для вимірювання точності моделей прогнозування щодо нових прикладів, які, можливо, не спостерігалися під час навчання. Я проводив кожен експеримент 10 разів, застосовуючи випадкове розбиття та вибірку на набір даних для кожного різного критерію. Іншими словами, кожна точка в будь-якій з представлених кривих є середньою величиною 10 прогонів тих же параметрів, але різного секціонування та вибірки.

Тут точність є вимірювальним параметром, який приймається як на наборі даних з різним набором навчань. У випадку набору даних NASA (див. табл. 2.2), результати отримуються без вмісту контенту, оскільки набір даних містить деякі з журналів зображення, так що вилучення вмісту цієї сторінки неможливе з цієї причини.

Таблиця 2.2 Усі значення K-порядку для 60% розміру набору даних

	Штучний набір даних		NASA DataSet
Порядок	Журнали	Журнали + вміст	Журнали
Перший	0.062	0.085	0.91
Другий	0.1213	0.132	0.088
Третій	0.564	0.543	0.014
Четвертий	0.4102	0.208	0.138
Точність	0.1953	0.2050	0.086

Штучний набір даних вводиться і використовується для обох випадків веб-функцій.

Враховуючи сеанс тестування (t) довжини L, ми проводимо прогнозування за допомогою моделі Маркова (L–1) і отримуємо прогноз для оцінки точності моделі. Нагадаємо, що остання сторінка t є кінцевим результатом, за яким ми оцінюватимемо правильність режиму проти; отже, ми використовуємо L1. У випадку, якщо t довший за найвищий N, використаний в експерименті, ми застосовуємо ковзне вікно розміром L на t. Наприклад, припустимо, t=p1, p2, p3, p4, p5, якщо використовувати марківську модель третього порядку, то розбиваємо t на p1, p2, p3, p4 та p2, p3, p4, p5.

Оскільки набори даних відрізняються, тому результати точності мають таку різницю, але метод для обох наборів даних для функції журналу є однаковим.

Для різних навчальних наборів спостерігається підвищення тренувальних даних, а також підвищення їх точності, але на дуже малу частку. Оскільки після 50% даних встановлюється точність розміру зміни за малим значенням в різних значеннях марківського порядку.

Всесвітня павутина вимагає від користувачів використовувати автоматизовані інструменти для пошуку потрібних інформаційних ресурсів, а також для того, щоб слідувати та оцінювати їхню модель використання. Попереднє завантаження веб-сторінок широко використовується для зменшення проблеми затримки доступу користувачів до Інтернету, його успіх головним чином спирається на точність прогнозування веб-сторінок.

Марківська модель є найбільш часто використовуваною моделлю прогнозування через її високу точність. Низький порядок марківських моделей має більш високу точність і менший покриття. Моделі вищого порядку мають ряд обмежень, пов'язаних з вищою складністю стану, зменшеним охопленням, іноді навіть гіршою точністю прогнозування. Кластеризація є одним з кращих рішень для розв'язання проблеми гіршого прогнозу точність моделі Маркова. Це потужний метод організації сеансів користувачів у кластери відповідно до їх подібності. У цій роботі розроблені методики подолання питань прогнозування веб-сторінок. Проте дослідження прогнозування веб-сторінок відбувається лише на самому початку і потребує глибшого розуміння.

3 ОПИС РОЗРОБЛЕНОЇ ПРОГРАМНОЇ СИСТЕМИ

3.1 Архітектура системи

Існує досить багато підходів для побудови складних систем з хорошою архітектурою. Незважаючи на невеликі відмінності цих підходів, у них багато спільного. Зрозуміло, вони все задають способи розбиття програми на окремі модулі. При цьому в кожній системі як мінімум є модулі, що містять бізнес-логіку програми (див. рис. 3.1) і модулі для відображення даних.

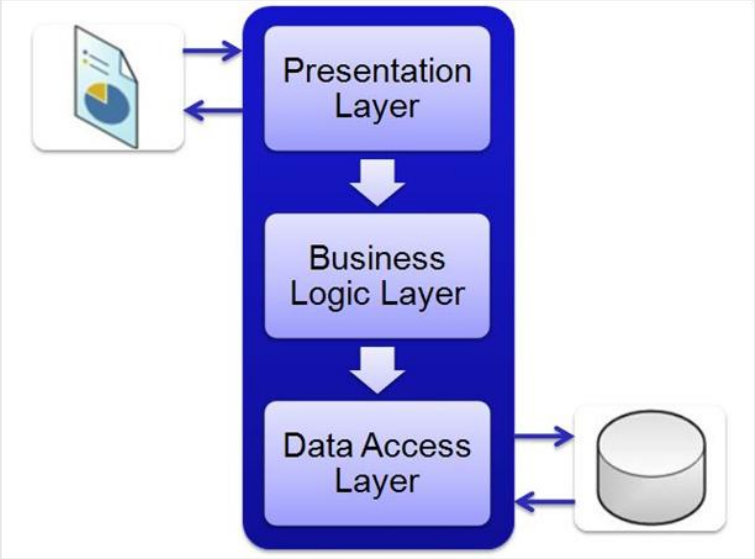

Рисунок 3.1 – Архітектура системи

Кожен підхід в підсумку дозволяє побудувати систему, яка задовольняє наступним принципам:

− архітектура повинна бути незалежна від різних фреймворків. Зрозуміло, в сучасному світі ми не можемо обходитися без якихось бібліотек, які дозволяють вирішувати завдання набагато швидше та частіше ефективніше, ніж це зробили б ми в разі самостійної реалізації. Але тут важливо розуміти, що бібліотека повинна вбудовуватися в вашу архітектуру, а не архітектура повинна підлаштовуватися під обрану бібліотеку. При використанні бібліотеки, яка змушує вас переробляти архітектуру програми, ми завжди будемо стикатися з певними обмеженнями цієї

бібліотеки та не зможемо перебудовувати архітектуру за потрібне вам чином. Потрібно використовувати бібліотеки тільки у якості допоміжних інструментів;

− система повинна бути протестована. При цьому ви повинні мати можливість тестувати як модулі системи окремо, так і тестувати взаємодію цих модулів між собою та інтеграцію їх в системі. Крім того необхідно тестувати систему без UI, реального сервера і роботи з базою даних, тобто архітектура повинна бути незалежна від оточення;

− з попередніх пунктів плавно випливають і такі принципи, які говорять про те, що наш додаток має бути незалежно від усього: від інтерфейсу користувача, від роботи з базою даних, від роботи сервера і від інших елементів оточення. Незалежність архітектури від оточення дуже важлива, оскільки це дозволяє змінювати різні компоненти оточення без зміни самої архітектури. Що мається на увазі під зміною компонентів архітектури. Це може бути зміна у виборі бази даних (або ж взагалі відмова від неї) або ж зміна в UI-частини програми (наприклад, потрібно змінити зовнішній вигляд екрану). Якщо для внесення таких змін вимагає від нас зміни архітектури, то, можливо, вам варто переглянути архітектуру.

Вибір підходу для реалізації серверної частини. Для можливості обробки великої кількості запитів слід проаналізувати який з підходів дозволить виконувати обробку найбільш оптимально.

Багатопотокова модель. У даній можливої реалізації додаток створює кілька потоків, передаючи кожному з них надійшов від клієнта запит. Потоки виконуються паралельно. Якщо потоки не мають загальних даних, то не буде накладних витрат на синхронізацію, що робить роботу досить швидкої. Після завершення роботи потік не видаляється, а залишається в пулі, очікуючи наступного запиту. Це дозволяє позбутися від накладних витрат на створення і видалення потоків. Припустимо потоків досить велика кількість, повільні запити забирають потік надовго, швидкі – обробляються майже миттєво, звільняючи потік для іншої роботи. Це не дозволяє повільним запитам забирати все процесорний час, змушуючи чекати швидкі запити. Може виникнути ситуація, коли на сервер прийде дуже велика кількість повільних запитів, наприклад працюють з БД. Такі запити

заберуть собі всі потоки, що унеможливить виконання інших запитів. Це можна вирішити збільшенням числа потоків, щоб вони могли обробити досить велика кількість повільних запитів, але чим більше потоків створюється, тим більше накладних витрат на їх обробку і тим менше процесорного часу виділяється кожному потоку.

Асинхронна модель побудована на черзі подій (event-loop) (див. рис. 3.2).

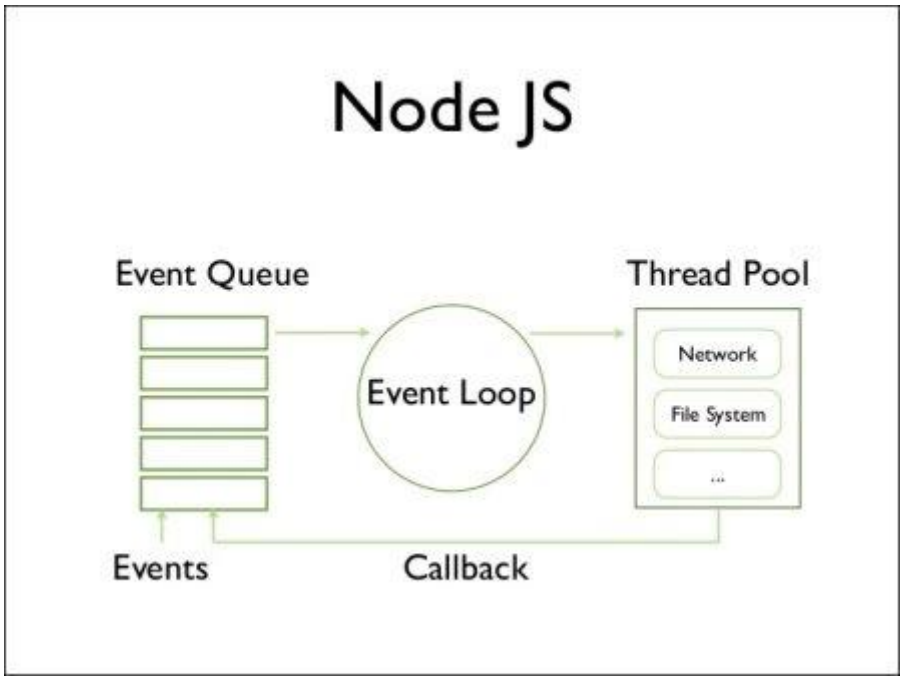

Рисунок 3.2 – Подійний цикл Node.js

Коли на сервер приходить черговий запит, він поміщається в кінець черги. Потік, який обробляє цю чергу, бере запит з початку черги, і виконує пов'язаний з ним код. Поки черга не порожня, процесор буде зайнятий роботою. За такою схемою працює Node.js. В даному випадку є єдиний потік, що обробляє чергу подій. Використання блокують операцій не рекомендується.

Розберемо на прикладі деякого запиту. З черги повідомлень береться запит. Node.js його обробляє і при цьому витрачає 1 мс. Далі робиться асинхронний запит до бази даних і управління відразу ж передається далі. Node.js бере з черги наступний запит. Припустимо після цього приходить відповідь бази даних на найперший запит. Подія, пов'язане з ним поміщається в чергу. Якщо в черзі нічого

не було він відразу ж виконається і дані відправляться клієнту. Якщо в черзі щось є то подія буде чекати обробку інших подій. Процесор завжди буде зайнятий корисною роботою. При цьому на обробку черги та перехід від події до події витрачається набагато менше часу, ніж на перемикання між потоками в багатопотоковій системі.

Технологія Node.js як засіб серверної розробки було вибрано, тому що має ряд переваг:

− досить імовірно, що клієнтські частини веб-додатків написані на JavaScript. В цьому випадку універсальність коду в вживаному стеку технологій – це важливий плюс використання JS і на сервері, про який варто пам'ятати;

− інструменти webpack допомагають в повторному використанні коду і на клієнті, і на сервері, що веде до його одноманітності на всіх рівнях системи;

− застосовуючи JS на клієнті та на сервері, можна створювати веб-додатки, які можна використовувати і в браузері, і на сервері. При цьому такі системи зазвичай працюють дуже чітко і зрозуміло;

− поява в Node конструкції async await повністю змінило підхід до написання асинхронного коду. Тепер такий код нагадує звичайний синхронний код, і за зовнішнім виглядом, і по поведінці. Механізм async / await підтримується в Node починаючи з версії 7.6. Він, зокрема, є рішенням сумнозвісної проблеми пекла коллбеков.

У якості веб-сервера був обраний фреймворк Express, який являє собою мінімальний та гнучкий інструмент для веб-додатків на node.js, що забезпечує надійний набір функцій для створення одно та багатосторінкових і гібридних веб-додатків». Основні характеристики:

Мінімальний. Це один з найбільш привабливих аспектів Express. Багато разів розробники забувають, що зазвичай «менше – більше». Це не означає, що він не є надійним, або він не має достатньо корисних функцій. Це означає, що він потрапляє у ваш шлях менше, дозволяючи вам повною мірою висловити свої ідеї, водночас надаючи щось корисне.

Гнучкий. Іншим ключовим аспектом філософії Express є те, що Express є розширюваним. Експрес надає вам мінімальні рамки, і ви можете додати в різних частинах експрес-функціональності за потреби, замінюючи все, що не відповідає вашим потребам. Це подих свіжого повітря. Так багато фреймворків дають вам все, залишаючи вас роздутим, таємничим і складним проектом, перш ніж ви навіть написали один рядок коду. Найчастіше перше завдання полягає в тому, щоб витрачати час на обробку непотрібних функціональних можливостей або заміну функціональності, яка не відповідає вимогам. Експрес приймає протилежний підхід, дозволяючи додати те, що вам потрібно, коли це потрібно.

Багатосторінкові веб-програми є більш традиційним підходом до веб-сайтів. Кожна сторінка на веб-сайті забезпечується окремим запитом до сервера. Тільки тому, що цей підхід є більш традиційним, це не означає, що це не без заслуг або що односторінкові додатки якось краще. Зараз є просто більше варіантів, і ви можете вирішити, які частини вашого вмісту повинні бути доставлені як односторінкові програми, і які частини повинні бути доставлені за індивідуальними запитами. "Гібрид" (див. рис. 3.3) описує сайти, які використовують обидва ці підходи.

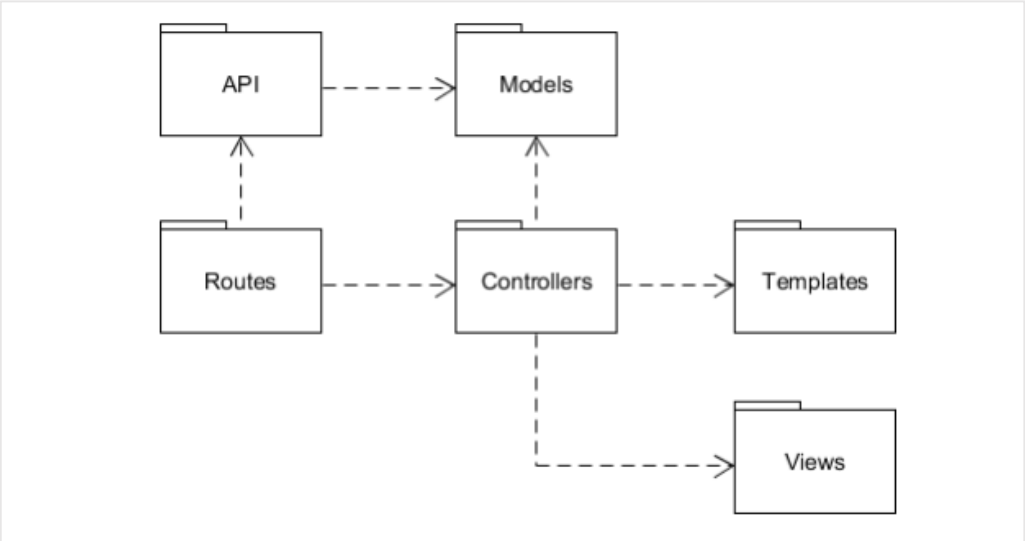

Рисунок 3.3 – Діаграма пакетів серверної частини веб-додатків

Односторінкові веб-програми є відносно новою ідеєю. Замість веб-сайту, який вимагає мережевого запиту кожного разу, коли користувач переходить на

іншу сторінку, веб-додаток з однією сторінкою завантажує весь сайт (або хороший фрагмент) до веб-переглядача клієнта. Після початкового завантаження навігація відбувається швидше, оскільки зв'язок із сервером практично відсутній. Односторінкові розробки додатків полегшуються використанням популярних фреймворків, таких як Angular або Ember, які Express раді обслуговувати.

Така архітектура дозволить чітко розділити функції кожного з модулів та виключити появу плавучих функціональних багів, які б блокували нормальну роботу кожного із компонентів системи.

Основні сервіси:

− сервіс імпорту лог файлу в базу даних. Кожне відвідування користувачем веб-сайту зазначене в балці витягується і переноситися як окремий запис в базу mysql;

− сервіс попередньої обробки. Імпортовані дані фільтруються за багатьма параметрами;

− сервіс прогнозування.

MySQL – дуже швидка, надійна система керування базами даних. База даних дозволяє ефективно зберігати, шукати, сортувати та отримувати дані. Сервер MySQL управляє доступом до даних, дозволяючи працювати з ними одночасно декільком користувачам, забезпечує швидкий доступ до даних і гарантує надання доступу тільки мають на це право користувачам. Отже, MySQL є багатокористувацьким та багатопотоковим сервером. Він застосовує SQL (Structured Query Language – мова структурованих запитів), що використовується по всьому світу стандартна мова запитів в бази даних. В даний час пакет MySQL доступний як програмне забезпечення з відкритим вихідним кодом, але в разі потреби можна отримати і комерційні ліцензії.

Основні переваги MySQL:

− багатопоточність, підтримка декількох одночасних запитів;

− оптимізація зв'язків з приєднанням багатьох даних за один прохід;

− записи фіксованої та змінної довжини;

− ODBC драйвер;

- гнучка система привілеїв і паролів;
- гнучка підтримка форматів чисел, рядків змінної довжини та міток часу;
- інтерфейс з мовами C і Perl, PHP;
- швидка робота, масштабованість;
- безкоштовна в більшості випадків;
- хороша підтримка з боку провайдерів послуг хостингу;
- швидка підтримка транзакцій через механізм InnoDB.

UML проектування. В області розробки програмного забезпечення використовується спеціальна мова для графічного опису об'єктного моделювання мова UML (Unified Modeling Language).

При опису роботи програми створюється абстрактна модель системи або підсистеми, яка називається UML моделлю.

Діаграма прецедентів (див. рис. 3.4) є графом, що складається з множини акторів, прецедентів (варіантів використання) обмежених границею системи (прямокутник), асоціацій між акторами та прецедентами, відношень серед прецедентів, та відношень узагальнення між акторами. Діаграми прецедентів відображають елементи моделі варіантів використання.

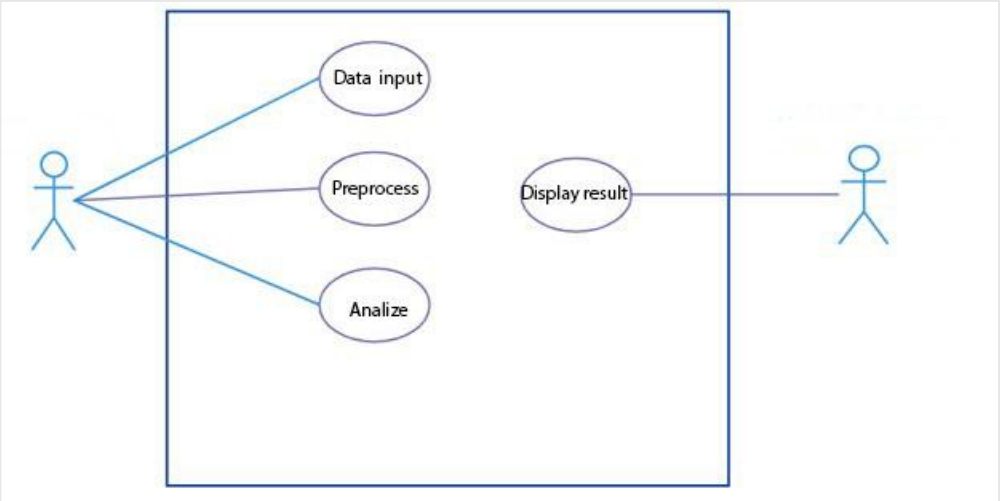

Рисунок 3.4 – Діаграма прецедентів

Суть даної діаграми полягає в наступному: проектована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із

системою за допомогою так званих варіантів використання. Варіант використання використовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система при діалозі з актором. При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.

На діаграмі послідовностей (див. рис. 3.5) зображена взаємодія компонентів системи, які відповідають за отримання користувачем даних розрахунку.

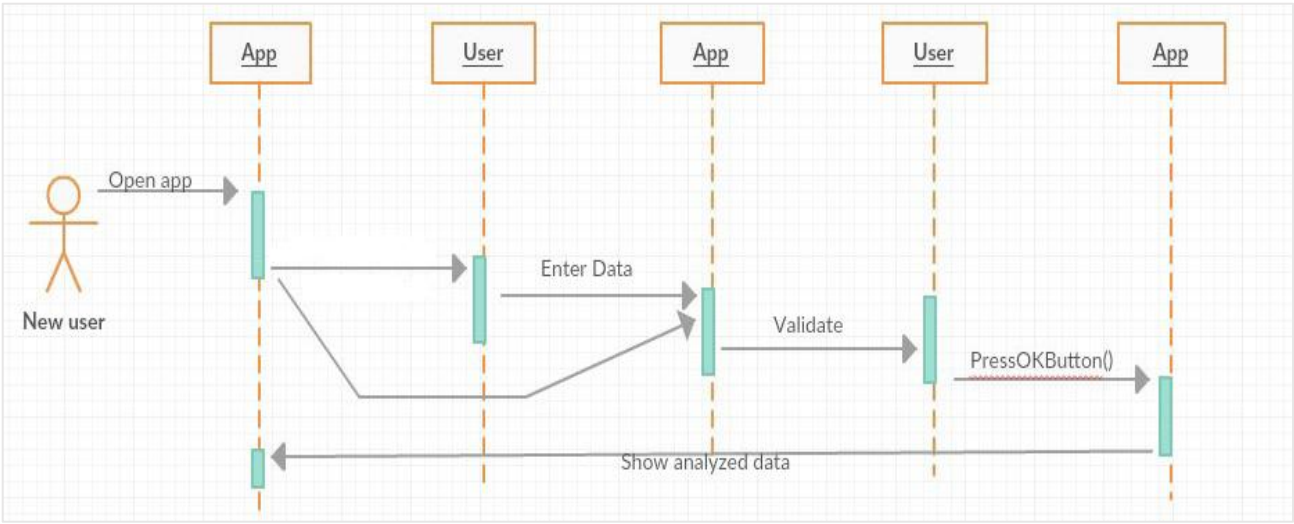

Рисунок 3.5 – Діаграма послідовностей

Діаграма послідовностей відображає взаємодію об’єктів у часі. На ній присутні тільки ті об’єкти які беруть участь у процесі взаємодії. Ключовий момент це динаміка взаємодії об’єктів у часі.

В UML діаграма послідовності має виміри. Перший зліва направо у вигляді вертикальних ліній, кожна з яких зображує лінію життя окремого об'єкта, який бере участь у взаємодії. Крайнім зліва на діаграмі зображується об'єкт, який є ініціатором взаємодії. Праворуч зображується інший об'єкт, який безпосередньо взаємодіє з першим. Таким чином, всі об'єкти на діаграмі послідовності утворюють деякий порядок, який визначається черговістю або ступенем активності об'єктів при взаємодії один з одним.

Другим виміром діаграми послідовності є вертикальна тимчасова вісь, спрямована зверху вниз. Початкового моменту часу відповідає сама верхня частина діаграми. Взаємодії об'єктів реалізуються за допомогою повідомлень, які надсилаються одними об'єктами іншим. Повідомлення зображуються у вигляді горизонтальних стрілок з ім'ям повідомлення, а їх порядок визначається часом виникнення. Тобто, повідомлення, розташовані на діаграмі послідовності вище, ініціюються раніше тих, які розташовані нижче.

На діаграмі станів (див. рис. 3.6) зображений процес отримання користувачем автоматичного розрахунку даних поведінки користувача.

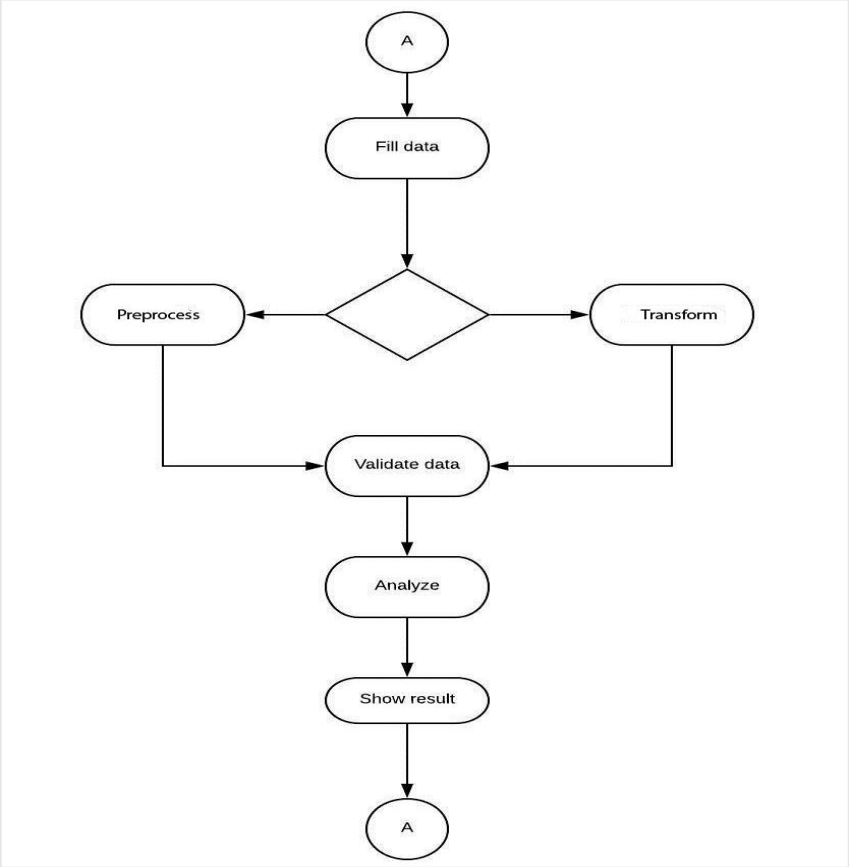

Рисунок 3.6 – Діаграма станів

Процес починається з запуску додатку. Коли схема обрана послідовність дій користувача буде дуже пряма та проста: йому необхідно заповнити дані для аналізу, після чого додаток автоматично розрахує прогноз.

Діаграма станів показує, як об'єкт переходить з одного стану в інший. Очевидно, що діаграми станів служать для моделювання динамічних аспектів системи (як і діаграми послідовностей, кооперації, прецедентів і, як ми побачимо далі, діаграми діяльності). Від інших діаграм діаграма станів відрізняється тим, що описує процес зміни станів тільки одного примірника певного класу – одного об'єкта, причому об'єкту реактивного, тобто об'єкта, поведінка якого характеризується його реакцією на зовнішні події.

Поняття життєвого циклу може бути застосовано як раз до реактивних об'єктів, даний стан (і поведінку) яких обумовлено їх минулим станом. Але діаграми станів важливі не тільки для опису динаміки окремого об'єкта. Вони можуть використовуватися для конструювання виконуваних систем шляхом прямого і зворотного проектування.

Уніфікована мова моделювання (UML) є стандартним інструментом для створення "креслень" програмного забезпечення. За допомогою UML можна візуалізувати, уточняти, конструювати та документувати артефакти програмних систем. Візуальні моделі забезпечують ясність представлення обраних архітектурних рішень і дозволяють зрозуміти, що розробляється у всій її повноті.

UML може бути застосовано на всіх етапах життєвого циклу аналізу бізнес-систем і розробки прикладних програм. Різні види діаграм які підтримуються UML, і найбагатший набір можливостей представлення певних аспектів системи робить UML універсальним засобом опису як програмних, так і ділових систем.

Діаграми дають можливість представити систему у такому вигляді, щоб її можна було легко перевести в програмний код. Основною причиною використання мови UML є спілкування розробників між собою.

Крім того, UML спеціально створювалася для оптимізації процесу розробки програмних систем, що дозволяє збільшити ефективність їх реалізації у кілька разів і помітно поліпшити якість кінцевого продукту.

UML прекрасно зарекомендувала себе в багатьох успішних програмних проектах. Засоби автоматичної генерації кодів дозволяють перетворювати моделі

мовою UML у вихідний код об’єктно-орієнтованих мов програмування, що ще більш прискорює процес розробки.

UML необхідний:

− керівникам проектів, які керують розподілом завдань і контролем за проектом;

− проектувальникам інформаційних систем які розробляють технічні завдання для програмістів;

− бізнес-аналітикам, які досліджують реальну систему і здійснюють інжиніринг і реінжиніринг бізнесу компанії;

− програмістам які реалізовують модулі інформаційної системи;

− при модифікації системи об'єктний підхід дозволяє легко включати в систему нові об'єкти і виключати застарілі без істотної зміни її життєздатності. Використання побудованої моделі при модифікаціях системи дає можливість усунути небажані наслідки змін, оскільки вони не ламають структури системи, а тільки змінюють поведінку об'єктів.

3.2 Проектування UI системи

Додаток що розробляється буде дійсно дуже простий у взаємодії з користувачем, адже він не перевантажений інформацією та запроектований чітко для виконання однієї цілі – розраховування та прогнозування поведінки користувачів веб-орієнтованих систем.

Дизайни відповідних станів програми є дуже простими, адже йому лише потрібно обрати специфічні дані через форму.

У 2014 році на конференції I/O була представлена нова дизайн-система, підхід, який отримав назву Material Design.

Нова дизайн-система дозволяє створювати сумісний користувальницький досвід на всіх екранах: десктоп, смартфон, планшети, годинник, телевізори, машини.

Для Android-додатків Material Design являє собою еволюцію візуальної мови Holo і дизайн-гайдлайни. У багатьох сенсах це більш гнучка система, яка створювалася з урахуванням того, що користуватися нею будуть інші дизайнери – Google був лише першим користувачем.

Material Design ґрунтується на чотирьох основних принципах.

Тактильні поверхні. У Material Design інтерфейс складається з відчутних шарів так званого «цифрового паперу». Ці шари розташовані на різній висоті і відкидають тіні один на одного, що допомагає користувачам краще розуміти анатомію інтерфейсу і принцип взаємодії з ним.

Поліграфічний дизайн. Якщо вважати шари шматками «цифрового паперу», то в тому, що стосується «цифрового чорнила» (всього того, що зображується на «цифровому папері»), використовується підхід з традиційного графічного дизайну: наприклад, журнального і плакатного.

Осмислена анімація. У реальному світі предмети не виникають нізвідки і не зникають в нікуди – таке буває тільки в кіно. Тому в Material Design ми весь час думаємо про те, як за допомогою анімації в шарах і в «цифрових чорнилах» давати користувачам підказки про роботу інтерфейсу.

Адаптивний дизайн. Йдеться про те, як ми застосовуємо попередні три концепції на різних пристроях з різними дозволами і розмірами екранів.

Material Design дозволяє більш об'єктивно підходити до прийняття дизайн-рішень: як щось виглядає, як щось працює, як здійснюється анімація і так далі. Вона задає розумні рамки, але не зайві обмеження.

Google Material Design являє собою комплексну концепцію створення візуальних, що рухаються і інтерактивних елементів для різних платформ і пристроїв.

3.3 Тестування розробленого програмного забезпечення

Написання автоматизованих тестів (див. рис. 3.7) є дуже важливим у будь-якому реальному проекті.

При написанні функції ми зазвичай уявляємо, що вона повинна робити, яке значення на яких аргументах видавати.

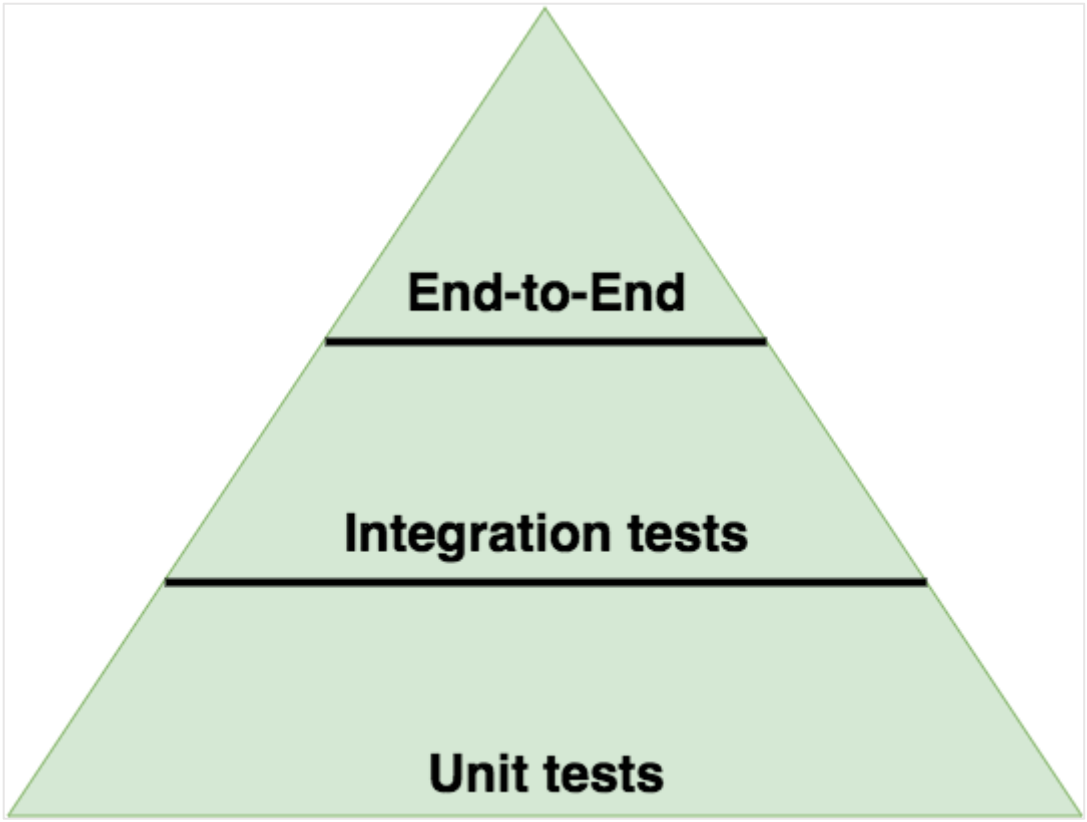

Рисунок 3.7 – Піраміда тестування

У процесі розробки ми час від часу перевіряємо, чи правильно працює функція. Найпростіший спосіб перевірити – це запустити її, наприклад в консолі, і подивитися результат.

Кожен модульний тест має таку структуру:

- налаштування тесту;
- виклик тестованого методу;
- asserting.

Кожен модульний тест повинен перевіряти лише одну проблему.

Для модульного тестування ми будемо використовувати наступні модулі:

– mocha ця бібліотека містить об'ємні функції для тестування, включаючи опис і її;

– chai це бібліотека підтримує різноманітні функції для перевірки. Є різні «стилі» перевірки результатів, з якими ми познайомилися з ним, на поточний момент ми будемо використовувати лише твердження;

– sinon для емуляції та хитрої підміни функцій «заглушка».

Ці бібліотеки дозволяють тестувати JS не тільки в браузері, але і на сервері NodeJS.

Jest є інструментом тестування з Facebook, що дозволяє легко виконувати модульне тестування в JavaScript.

Для того, щоб тести були підібрані Jest автоматично, вони або повинні бути розміщені в каталозі __tests__ проекту, або включати тест слова або spec в імені файлу. Це можна легко налаштувати у параметрах Jest.

```
describe('POST /api/v1/import', () => {
  it('should respond with a success, (done) => {
    chai.request(server)
    .post('/api/v1/import')
    .send(data)
    .end((err, res) => {
      // there should be no errors
      should.not.exist(err);
      // there should be a 201 status code
      // (indicating that something was "created")
      res.status.should.equal(201);
      // the response should be JSON
      res.type.should.equal('application/json');
      // the JSON response body should have a
      // key-value pair of {"status": "success"}
      res.body.status.should.eql('success');
      // the JSON response body should have a
      // key-value pair of {"data": 1 user object}
      res.body.data[0].should.include.keys(
        'id', 'username', 'email', 'created_at'
      );
      done();
    });
  });
```

Інтеграційне тестування – вид тестування, при якому відповідно до вимог перевіряється інтеграція модулів, їх взаємодія між собою, а також інтеграція підсистеми в одну систему. Для інтеграційного тестування використовуються компоненти, які вже проведені за допомогою модульного тестування, які групуються в множинах. Дані множини перевіряються у відповідності з планом тестування, складеним для них, і об'єднуються вони через свої інтерфейси.

ВИСНОВКИ

Аналіз поведінки веб-користувачів має гарну перспективу розвитку у майбутньому. З огляду на затребуваності в сфері електронної комерції і зростання популярності системи веб-аналітики в різних сферах діяльності, можна зробити висновок, що виконана робота є актуальною.

В ході виконання науково-дослідної атестаційної роботи, було проведено детальне дослідження існуючих методів аналізу даних поведінки користувачів веб-орієнтованих систем, виявлення протиріч відомих теоретичних та експериментальних результатів.

Також було встановлено що концепції і підходи до аналізу даних поведінки були придумані вже давно проте набули популярності тільки нещодавно, оскільки з’явились інструменти та технології що дозволяють значно зменшити великий об’єм ресурсів що необхідний для технічної реалізації та значно спростити процес аналізу.

Результатом теоретичних та експериментальних досліджень є удосконалений метод прогнозування поведінки користувачів веб-орієнтованих систем на основі використання марківської моделі і ключових слів. Марківська модель є найбільш часто використовуваною моделлю прогнозування через її високу точність. Низький порядок марківських моделей має більш високу точність і менше покриття. Моделі вищого порядку мають ряд обмежень, пов'язаних з вищою складністю стану, зменшеним охопленням, іноді навіть гіршою точністю прогнозування.

Оглянуті типові особливості архітектури веб-додатку відображують тенденцію до адаптації вже відомих шаблонів проектування та рішень що використовуються при розробці. Для тестування запропонованого методу використовувалась система написана на технології NodeJS.

Набуті в ході даної роботи результати можуть використовуватися для підвищення якості та достовірності звітів систем веб-аналітики. Прогнозування того, які товари будуть цікаві різним користувачам з різних місць у світі також

може бути корисно для поведінкової сегментації та прогнозування конверсії. Продавці використовують аналітику даних для кращого ознайомлення з їхніми клієнтами, оптимізації процесу онлайн-покупок та збільшенні продажів.

Можливими напрямами подальшого вдосконалення роботи можна вважати поліпшення точності прогнозування за допомогою нових підходів застосування марківської моделі та попередньої обробки даних, що дасть більшу точність прогнозування в цілому.

Апробація роботи. Основні положення даного дослідження доповідалися на міжнародних науково-практичних конференціях:

- «Discovery Science», м. Карлові Вари, Чехія;
- «Прикладні наукові розробки та теоретичні дослідження XXI століття», м. Вінниця, Україна;
- «Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 37)», м. Тернопіль, Україна;
- «Новини науки: дослідження, наукові відкриття, високі технології», м. Харків, Україна.

Вважаю мету поставленої науково-дослідної атестаційної роботи досягнутою.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

Видання

1. Tomlin, J.A.: A new paradigm for ranking pages on the world wide web. In: WWW 2003, Budapest, Hungary, May 20-24 (2003); In: Computer Networks and ISDN Systems, pp. 107–117 (1998)

2. Zhou, Y., Leung, H., Winoto, P.: Mnav: A markov model-based web site navigability measure. IEEE Trans. Softw. Eng. 33(12), 869–890 (2007)

3. Velasquez, J.D., Palade, V.: A knowledge base for the maintenance of knowledge extracted from web data. Knowledge Based Systems Journal 20(3), pp. 238–248 (2007)

4. Schneider-Mizell, C.M., Sander, L.M.: A generalized voter model on complex networks.Technical Report arXiv:0804.1269, Department of Physics, University of Michigan, 15 pages, 3 figures (April 2008)

5. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: Computer Networks and ISDN Systems, pp. 107–117 (1998)

6. Kosala, R., Blockeel, H.: Web mining research: A survey. SIGKDD Explorations: Newsletters of the Special Interest Group (SIG) on Knowledge Discovery and Data Mining 1(2), 1–15 (2000)

7. Anderson, C.: Wired Magazine, Editorial (June 2008)

8. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. 34(1), 1–47 (2002)

9. Abraham, A., Ramos, V.: Web usage mining using artificial ant colony clustering and genetic programming. In: Procs. of the 2003 IEEE Congress on Evolutionary Computation (CEC 2003), pp. 1384–1391 (2003)

10. Jaworska, J., Sydow, M.: Behavioural Targeting in On-line Advertising: An Empirical Study, 2008, Accepted for the 9th International Conference on Web Information Systems Engineering (Wise 2008), Auckland, New Zealand, September 1-4, 2008 (to be printed in LNCS, Springer).

11. Nithya.P and Dr.P.Sumathi., 2012, “Novel PreProcessing Technique for Web Log Mining by Removing Global Noise and Web Robots”, 2012 National Conference on Computing and Communication Systems 978-1-4673-1953-9/12 © 2012 IEEE:

12. P.-N. Tan, V. Kumar (2000) Modeling of web robot navigational patterns, in: WEBKDD Web Mining for Ecommerce Challenges and Opportunities, Second International Workshop.

13. Jaideep Srivastava, Robert Cooley, Mukund Deshpande, Pang-Ning Tan (2000), Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data, Vol.1 Page(s): 12-23.

Статті, конференції, тези доповідей

14. Іванов О.В. Дослідження основних джерел, процесів і методів обробки даних поведінки користувачів веб-орієнтованих систем. Новини науки: дослідження, наукові відкриття, високі технології: зб. наук. праць «ΛΌГОΣ» з матеріалами міжнар. наук.-практ. конф., м. Харків, 31 березня, 2019 р. ГО «Європейська наукова платформа». ISBN 978-617-7171-80-4. С. 15-17;

15. Іванов О.В. Дослідження методів аналізу даних поведінки користувачів веб-орієнтованих систем. Збірник наукових робіт «ADVANCED OF SCIENCE» з матеріалами міжнар. наук.-практ. конф. «Discovery Science», м. Київ, м. Карлові Вари, 5 квітня 2019 р.. ISBN 978-80-7534-078-8. С. 213-218;

16. Іванов О.В. Прогнозування поведінки користувачів веб-орієнтованих систем на основі аналізу серверних логів. Міжнародна наукова інтернет-конференція "Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 37)" / Збірник тез доповідей: випуск 37 (м. Тернопіль, 2 квітня 2019 р.). ISSN 2522-932X. С. 23-25;

17. Іванов О.В. Попередня обробка даних для аналізу поведінки користувачів веб-орієнтованих систем. Прикладні наукові розробки та теоретичні дослідження XXI століття: зб. наук. праць «ΛΌГОΣ» з матеріалами міжнар. наук.-практ. конф., м. Вінниця, 15 квітня, 2019 р. Вінниця: ГО «Європейська наукова платформа», 2019. ISBN 978-617-7171-80-4. С 72-74;

Електронний ресурс

18. [Електронний ресурс] – Режим доступу до ресурсу: https://www.i-scoop.eu/artificial-intelligence-cognitive-computing/