Міністерство освіти і науки України
Державний університет "Одеська політехніка"
Навчально-науковий інститут комп'ютерних систем
Кафедра системного програмного забезпечення

Жупікова Вікторія Євгенівна,
студент групи АС-161

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

Програмний засіб для допомоги у виборі готелів з урахуванням аналізу тональності відгуків

Спеціальність:
121 – Інженерія програмного забезпечення

Освітня програма:
Інженерія програмного забезпечення

Керівник:
Комлева Наталія Олегівна,
канд. техн. наук, доцент

Одеса – 2021

ЗМІСТ

Міністерство освіти і науки України
Державний університет "Одеська політехніка"
Навчально-науковий інститут комп'ютерних систем
Кафедра системного програмного забезпечення

Рівень вищої освіти: другий (магістерський)
Спеціальність: 121 – Інженерія програмного забезпечення
Освітня програма: Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Любченко В.В.
«___» _____ 20__ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Жупікової Вікторії Євгенівни, група АС-161

1. Тема роботи: Програмний засіб для допомоги у виборі готелів з урахуванням аналізу тональності відгуків
Керівник роботи: Комлева Наталія Олегівна,канд. техн. наук, доцент
затверджені наказом ректора від «25» жовтня 2021 р. № 374-в.
2. Зміст роботи: проведення критичного аналізу існуючих рішень для допомоги у виборі готелів з урахуванням аналізу тональності відгуків; розробка моделі для рекомендацій, вибір алгоритмів; специфікація вимог до програми; проектування програмного засобу; вибір та обґрунтування інструментів розробки, особливості підключення бібліотек до програмної системи; програмна реалізація системи; тестування моделі класифікатора текстових відгуків, тестування функціональності системита визначення властивостей програмного засобу.
3. Перелік ілюстративного матеріалу: згідно зі слайдами презентації.

4. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

5. Дата видачі завдання: «___» _____ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання	Примітка
1	Аналіз існуючих рішень	2.09.2021 – 16.09.2021	вик.
2	Специфікація вимог до програмної системи	17.09.2021 – 30.09.2021	вик.
3	Проектування програмної системи	01.10.2021 – 17.10.2021	вик.
4	Програмна реалізація системи	18.10.2021 – 13.11.2021	вик.
5	Тестування системи	14.11.2021 – 24.11.2021	вик.
6	Оформлення пояснювальної записки та графічного матеріалу	25.11.2021 – 30.11.2021	вик.

Здобувач вищої освіти _____ В.Є. Жупікова

Керівник роботи _____ Н. О. Комлева

АНОТАЦІЯ

Метою роботи є скорочення часу для отримання рекомендаційних оцінок для вибору готелів на підставі відгуків користувачів та іншої інформації щодо умов та послуг з використанням методів аналізу тональності та машинного навчання. Використано методи обробки текстових даних, методи класифікації за допомогою згорткової нейронної мережі та верифікації роботи моделі. Методи програмної розробки базуються на мові програмування Python з використанням бібліотек spaCy, minibatch, compounding та конвеєра nlp. Як результат роботи розроблено програмний засіб, що дозволяє створювати комплексні рекомендаційні оцінки для допомоги при виборі готелів.

Ключові слова: аналіз тональності, машинне навчання, обробка текстових даних, згорткова нейронна мережа,Python, spaCy, minibatch, compounding, nlp.

ABSTRACT

The aim of the work is to reduce the time for obtaining recommendation assessments for the selection of hotels based on user feedback and other information on conditions and services using sentiment analysis and machine learning methods. Methods of text data processing, methods of classification using a convolutional neural network and verification of the model are used. Software development methods are based on the Python programming language using spaCy, minibatch, compounding libraries and nlp pipeline. As a result, a software tool that allows you to create comprehensive recommendation assessments to help you choose hotels has been developed.

Keywords:sentiment analysis,machine learning, text processing, convolutional neural network, Python, spaCy, minibatch, compounding,nlp.

ВСТУП

В останні десятиліття інтерес багатьох розробників та користувачів був націлений на автоматизацію виконання різних задач незалежно від того, до якої предметної області вони належать. Сбір та попередня обробка інформації, обчислення та ефективне представлення результатів, виконання ланцюжку логічних висновків, перетворення форматів даних – все це та багато іншого виконується за допомогою автоматизованих програмних засобів.

Метою автоматизації є полегшення роботи кінцевого користувача: зменшення часу на пошук чи обробку інформації, підвищення зручності сприйняття поданих даних, надання результатів логічних висновків як додаткової інформації для спрощення формулювання власної думки з приводу певних ситуацій та завдань. Також автоматизовані програмні інструменти дозволяють підвищити точність отримання результатів вирішення завдань. Все це повною мірою виправдовує зусилля розробників, які витрачаються на розвиток інструментівавтоматизації.

В даний час, незважаючи на важке епідеміологічне становище як в Україні, так і в усьому світі, важливе значення в розвитку е-індустрії мають туристичні подорожі та готельні ланцюги. Вони підтримують просування на світовому ринкувизначених стандартів обслуговування туристів, а також сприяють розповсюдженню інформації щодо рівня якості обслуговування туристів у готелях. Лідерами готельних ланцюгів можна вважати такі як «Hospitaliti Franchise System Blanstone Part.», «Holiday Inn Worldwide», «Best Western International», «Accor», «Choice Hotel International», «Marriott International», «ITT Sheraton Corp.»,«Promus Corp.», «Hilton Hotel Corp.». Але коло інтересів туристів не повинно бути обмеженим тільки найвідомішими готелями. Для туриста є важливим отримати актуальну інформацію щодо великої сукупності готелей, застосувати для неї інструменти, що дозволяють виконати автоматичну обробку, та проаналізувати отримані результати. Саме ці результати можуть стати підставою для прийняття власних рішень щодо вибору потрібного готеля.

«Будь-який готель має свої власні стандарти для обслуговуючого персоналу, що визначаються класом готелю, сегментом ринку, в якому він працює, спектром додаткових послуг, фактором сезонності, конкуренцією, системою управління персоналом, що здійснює не тільки підбір кадрів, але і вибір моделі управління підприємством» [1].

Підставою для вибору готелю можуть стати як об’єктивні аспекти (наявність зон відпочинку, власного пляжу, розміщення на території готелю аквапарку чи атракціонів), так і суб’єктивні, що визначаються оцінюванням туристами певних послуг за категоріальною шкалою (зазвичай це шкала Лайкерта зі значеннями до п’яти упорядкованих категорій) та створенням детальних текстових відгуків. Загальна кількість об’єктивних аспектів може бути досить великою, а текстові відгуки – вельми розгорнутими. Це веде до підвищення складності аналізу інформації,поданої в необробленому вигляді. Якщо все це помножити на загальну кількість готелів, ситуацію по яким бажано проаналізувати, то стає зрозумілим, що без застосування спеціалізованих інструментів зробити це неможливо.

Впровадження комп'ютерних систем обробки та аналізу інформації щодо готелей сприяє забезпеченню ефективності та своєчасності приймання рішень та підвищенню якості таких рішень. З усього сказаного випливає, що розробка рекомендаційної системи для вибору готелів на підставі відгуків з використанням методів машинного навчання є актуальним завданням.

Метою роботи є скорочення часу для отримання рекомендаційних оцінок для вибору готелів на підставі відгуків користувачів та іншої інформації щодо умов та послуг з використанням методів аналізу тональності та машинного навчання.

Для досягнення цієї мети вирішено наступні завдання:

– розглянуто особливості роботи готельного бізнесу та визначено характеристики, за якими можна оцінювати готелі;

– проаналізовано принципи роботи рекомендаційних систем та визначено тип системи, відповідний до вирішуваного завдання;

– проведено аналіз існуючих аналогів та визначено множину вимог до розроблюваної системи;

– створено модель оцінювання з урахуванням аспектно-орієнтованого підходу;

– вивчено принципи роботи згорткової нейронної мережі із застосуванням до класифікації відгуків;

– проведено деталізацію вимог до функціональних та нефункціональних характеристик розроблюваного програмного застосування;

– виконано проєктування архітектури програми з використанням шаблонів проєктування, визначено відповідні структури даних;

– виконано програмну реалізацію системи згідно з обраною мовою та інструментами програмування;

– виконано тестування роботи програмної системи.

1 КРИТИЧНИЙ АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

1.1Аналіз особливостей роботи готельного бізнесу

Сьогодні в світі туристичного та готельного бізнесу можна спостерігати жорстку конкурентну ситуацію, при якій постачальники послуг повинні боротися за своїх клієнтів і в обов'язковому порядку конструктивно реагувати на зміни в їх зовнішніх запитах. Незважаючи на важку епідеміологічної ситуацію, пов'язану з розповсюдженням вірусу COVID-19, готельно-туристичний бізнес продовжує займати своє місце на ринку і шукає нові форми розвитку. Розвитку такого бізнесу сприяє введення актуальних послуг і підтримка виконання існуючих послуг на високому рівні.

Готельний бізнес не існує сам по собі, він активно задіює інші напрямки і сприяє їх розвитку. Серед таких напрямків можна виділити бізнес, торгівлю, будівництво та, не в останню чергу, інформаційні технології. Саме розвиток інформаційних технологій дозволяє перевести багато процесів в онлайн-режим і налагодити безперебійну роботу багатьох процесів навіть в умовах карантинних обмежень. Готельний бізнес надає працівникам робочі місця, число таких місць залежить від рівня готелю (числа "зірок") і ідеології самого готелю, або ж мережі готелів, якщо готель не є самостійним, а належить до великої готельної мережі.

В останні роки готельний бізнес став залучати підприємців ще й через можливості організації комфортної та ефективної роботи в умовах складних епідеміологічних ситуацій.

З іншого боку, цей бізнес є привабливим і для підприємців, тому що має такі особливості:

– для початку ведення бізнесу не завжди потрібні великі інвестиції, завжди є можливість почати з малого, а потім розвивати бізнес у випадку позитивного результату;

– є можливість гнучко управляти прибутками і витратами, формуючи спеціальні пакети послуг і пропозицій;

– специфіка розташування готелів дозволяє ефективно включитися в потік туристичного попиту;

– термін окупності вкладень в готельний бізнес зазвичай невисокий;

– для підвищення привабливості готелю можна використовувати сезонну специфіку місця його розміщення, а також бути відкритими до розміщення учасників різних широко відомих заходів, форумів, конкурсів, чемпіонатів.

Розглянемо основні фактори, які є настільки суттєвими, що впливають на конкурентоспроможність певного готелю. В першу чергу це - географічне положення, тобто країна або місцевість розташування, а також віддаленість від певних геопозицій, які цікавлять туриста (океан, море, пам'ятки, торгові точки, міська інфраструктура, транспортна інфраструктура, парки розваг, місця відпочинку і т.д.). Далі враховується матеріально-технічне оснащення номерів кожного рівня якості по відношенню до вартості номерів за рівнями (люкс, напівлюкс і ін.).Крім цього враховується кваліфікованість персоналу, якість роботи точок харчування, місць видачі інвентарю і т.д. Кожен клієнт сам вибудовує свою схему пріоритетів, відповідно до якої вибирає той чи інший готель, що йому сподобався.

Відповідно до принципів маркетингу, що надаються постачальником, товари і послуги повинні відповідати запитам споживача. Однак, за відсутності єдиної думки споживачів, повинні бути продумані механізми, що дозволяють адаптувати кожну послугу під сегмент користувачів зі схожими потребами. Така сегментація дозволяє опрацьовувати маркетингові стратегії, звертаючись цільовим чином до того чи іншого конкретного сегменту користувачів. Все це дозволяє ефективно розширяти цільову аудиторію користувачів, що належать різним сегментам.Іншим поширеним підходом є орієнтація готелю (особливо в тому випадку, коли він є невеликим і масштабування найближчим часом не передбачається) на конкретний сегмент користувачів. Це дозволяє спростити багато процесів, включаючи логістику.При цьому дуже важливим є дотримання вірної цінової політики. Помилки в процесі формування собівартості готельних

послуг можуть привести до зниження ефективності ведення бізнесу або, в гіршому випадку, до його провалу.

Корисним інструментом для розуміння ступеня успішності впровадження будь-якої інновації, або оцінки якості по певній цікавій позиції (послуги або товару) є зворотний зв'язок від користувачів. Для випадку готелей такими користувачами є туристи, які можуть залишати відгуки з оцінкою якості послуг. Чим більш детальними є відгуки, тим більше інформації можна отримати з них.

Найбільшу ступеня гнучкості використання надають текстові відгуки туристів, які можуть містити досить великі обсяги тексту, поданого в довільному вигляді. Також корисними є оцінки, які туристи можуть виставляти по відповідним позиціям готельних послуг.

Отже, для наочного узагальнення усього вищесказаного, створено інтелект-карту програмної та організаційної частини розробки рекомендаційної системи для вибору готелів (рис. 1.1). Карта деталізує відповіді на питання «хто», «що», «як» та «чому». Для розробки інтелект-карти використано сучасну програмуCanva.

При відповіді на питання «хто?» визначаються категорії осіб, для яких була б корисною дана рекомендаційна система.

Такими особами можуть бути, по-перше, звичайні користувачі, що є потенційними туристами та бажають обрати готель для власного відпочинку. Система надає таким користувачам рекомендаційну інформацію, яка полегшує процес власного вибору. В ідеальному випадку такі рекомендації було б бажано отримувати для будь-якого списку готелів, але для цього потрібна підготовлена інформація щодо оцінок цих готелів користувачами. У разі ускладнення надавання такої інформації рекомендаційна система не буде мати підстав для висновку.

По-друге, особами можуть бути менеджери готелів, які націлені на контроль та поліпшення усіх процесів та націлені на перемогу у конкурентній боротьбі.

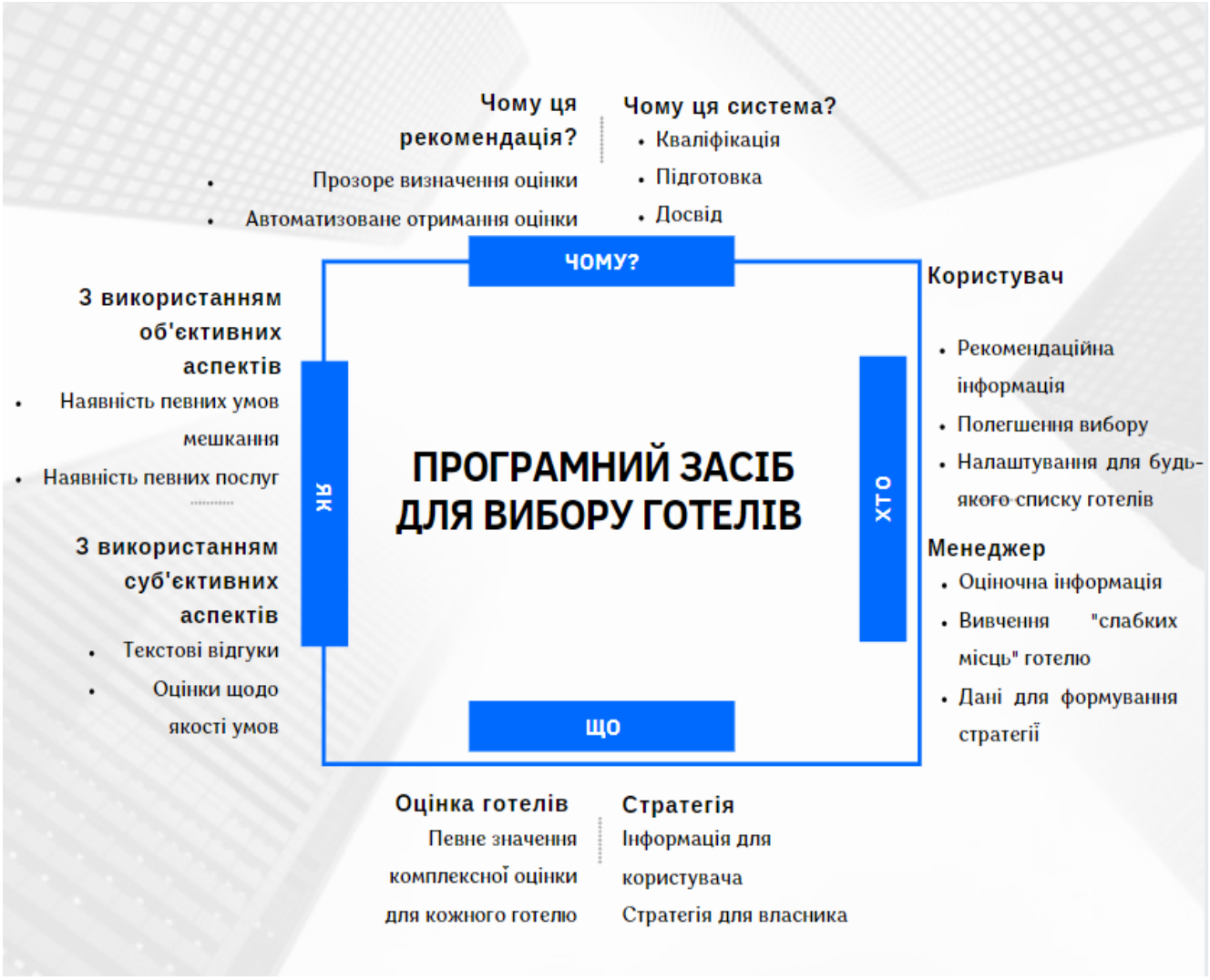

Рисунок 1.1 – Інтелект-карта програмної та організаційної частини розробки рекомендаційної системи для вибору готелів

Їм також на користь буде оціночна інформація задля вивчення «слабких місць» власного готелю та аналізу даних, що мають стати підставою для формування оновленої успішної стратегії.

При відповіді на питання «що?» визначаються сутності, якими можна оперувати як під час користування рекомендаційною системою, так і пізніше. Отримані оцінки готелів мають певні значення та комплексно оцінюють якість готельних послуг для обраного користувачем готелю. Ці оцінки є додатковою інформацією для туристів, які обирають готель, та для власників (менеджерів) з метою оновлення стратегії розвитку готелю.

Питання «як?» розкриває внутрішні механізми, за допомогою яких працює рекомендаційна система. Першим ефективним механізмом є використання об’єктивних аспектів оцінювання, які включають урахування наявності чи відсутності певних умов мешкання (наприклад, чи має готель власний пляж, чи має він дитячу кімнату) чи певних послуг (наприклад, ремонт одягу, масаж чи послуги перукаря). Другий ефективний механізм – це використання суб’єктивних аспектів оцінювання, що стосується оцінки рівня якості тих послуг, які надаються. Оцінки можуть бути надані за категоріальною шкалою:

1 – дуже погано;

2 – погано;

3 – нормально;

4 – добре;

5 – дуже добре.

Також можуть бути задіяні текстові відгуки. Розпізнавати та аналізувати їх складніше, ніж оцінки за категоріями, але вони надають можливість висловити розширений відгук стосовно будь-якого боку проблеми.

Нарешті, питання «чому?» пояснює чому саме розроблювана рекомендаційна система повинна бути взятою до уваги. По-перше, визначення оцінки щодо якості роботи готелю є прозорим, тому що будується на відкритих даних та зрозумілих правилах. Також оцінювання виконується за допомогою спеціально розроблено програмного забезпечення, що дозволяє максимально автоматизувати цей процес. По-друге, при розробці системи у кваліфікаційній роботі застосовано принципи інженерії програмного забезпечення та використано спеціальні компетентності, отримані при вивченні відповідних освітніх компонент.

1.2 Основні принципи аналізу тональності відгуків

Під тональністю будь-якого тексту розуміють вид емоційного забарвлення цього тексту, який показує ставлення автора до якої-небудь події, об'єкту або процесу. Аналіз тональності проводиться на основі аналізу емоційно забарвленої

лексики і її складових. Аналіз тональності тексту відноситься до класу методів контент-аналізу, робота яких може бути автоматизована і організована таким чином, щоб підвищувати інформованість про рівень думок про певні об'єкти або процеси.

Використання методів сентимент-аналізу дозволяє вирішувати такі цікаві завдання, як визначення емоційного стану автора під час створення тексту та відношення автора до певного об’єкту, про який йдеться у тексті. При цьому теоретичні визначення характеристик сентимент-аналізу мають досить конкретні практичні застосування на практиці.

Найбільш розповсюдженим практичним прикладом заохочення методів сентимент-аналізу є оцінка якості послуг чи / та певних товарів на підставі текстових відгуків користувачів. Важливість такої інформації, отриманої користувачем стосовно певного товару чи послуги коливається в залежності від ступеня значущості послуги чи товару для користувача. Тому у випадку необхідності прийняття складного рішення для користувача є доцільним отримання рекомендаційно-довідкової інформації. При цьому різкий зріст кількості товарів та послуг, а також осіб, що ними користуються, призводить до неможливості ручної обробки масивів даних. Тому такі процеси повинні бути автоматизованими.

Не менш важливим є вирішення задачі моніторингу соціальних мереж задля виявлення заборонених та опасних повідомлень, що можуть бути пов’язані з ситуаціями, які впливають на порядок у країні.

Окремим цікавим завданням є визначення авторства для певного тексту. Це може мати відношеня до багатьох областей: література, криміналістика та ін.

Новим напрямком застосування методів сентимент-аналізу став репутаційний маркетинг, що має зв’язок з вивченням та визначенням впливу доповідей на потенційну аудиторію. Також застосування ідей цього маркетингу сприяє створенню освітніх технологій, навчальних матеріалів та інструментів їх подання з можливістю адаптації під аудиторію з певними узагальненими характеристиками.

Задача сентимент-аналізу складається з трьох етапів (рис. 1.2).

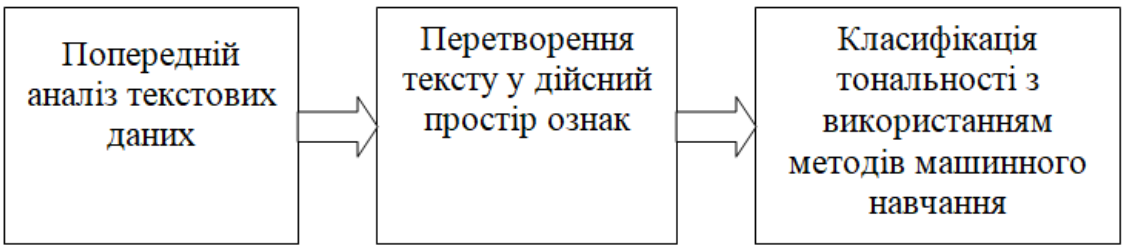

Рисунок 1.2 – Загальна схема виконання сентимент-аналізу

Під попереднім аналізом текстових даних розуміється:

- видалення стоп-слів;
- сегментація;
- приведення слів до єдиної норми;
- маркування частин мови.

Для перетворення тексту в дійсний простір ознак найчастіше використовуються два методи:

- мішок слів (англ. - bag of words);
- Word2Vec.

Обидві ці моделі застосовують для роботи методи, що засновані на статистичній інформації про слова тексту. При цьому підході для кожного об’єкта створюється вектор з довжиною, яка дорівнює кількості слів, що використані в усіх аналізованих текстах.

На останньому кроці сентимент-аналізу обирається та застосовується найбільш придатний метод машинного навчання. Саме він виконує класифікацію, за результатами якої отримається значення класу (емоційний окрас щодо визначення ставлення, думки автора), до якого належить текстове повідомлення:

- позитивна думка;
- нейтральна думка;
- негативна думка.

Методів класифікації існує досить багато. До найбільш розповсюджених методів класифікації відносять метод опорних векторів, градієнтний бустинг, наївний Байєс та ін.

1.3 Огляд існуючих програмних аналогів

Існує досить багато аналогів, що виконують аналіз тональності тексту. Розглянемо найбільш вагомі з них.

DiscoverText– надає «десятки багатомовних функцій, інтелектуальний аналіз тексту, науку про дані, анотації для людей і функції машинного навчання. DiscoverText пропонує широкий спектр простих і просунутих хмарних програмних засобів, що дозволяють користувачам швидко і точно оцінювати великі обсяги текстових даних. Користувачі працюють з допомогою графічного призначеного для користувача інтерфейсу "точка і клацання" в веб-браузерах для сортування неструктурованого вільного тексту, поширеного в дослідженнях ринку, а також пов'язаних метаданих, які також можна знайти в платформах зворотного зв'язку з клієнтами, CRM, чатах, електронній пошті, великомасштабних кадрових або інших відкритих відповідях на опитування, публічних коментарях державним установам, Twitter, RSS-канали та інші форми текстових даних.» [2]

Clarabridge – це «платформа управління клієнтським досвідом. Вона витягує і аналізує текст з чатів, платформ для опитувань, блогів, форумів і сайтів оглядів. Користувачі також можуть отримувати інформацію з електронних листів, заміток співробітників і операторів, записів розмов і опитувань з інтерактивним голосовою відповіддю: система може перетворювати їх у текст. Вони також забезпечують прослуховування в соціальних мережах. Система враховує галузь і джерело, розуміючи значення і контекст кожного коментаря. Результати аналізу тональності відображаються за 11-бальною шкалою. При необхідності користувачі можуть змінювати оцінки настроїв, щоб вони були більш специфічними для бізнесу.» [3]

InMoment«пропонує п'ять продуктів, які разом складають платформу оптимізації клієнтського досвіду. Один з них, «Голос клієнта», дозволяє підприємствам збирати і аналізувати відгуки клієнтів в текстовій, відео- і голосовий формах. Кількість джерел даних достатня і включає опитування, соціальні мережі, CRM і т.д. Розробники надають користувачам текстові повідомлення в реальному часі, що настроюються інформаційні панелі і різні варіанти звітності.» [4]

SAS Sentiment Analysis - це частина SAS Text Analytics - лінійки продуктів, що забезпечують «всебічний лінгвістичний і статистичний аналіз неструктурованою інформації. Sentiment Analysis аналізує розміщеного контенту джерела, включаючи вебсайти і соціомедіаресурси, а також внутрішні текстосодержащіе ресурси, і будує звіти, в яких відображаються думки споживачів, клієнтів і конкурентів із зазначенням динаміки в часі. У продукті інтегровані засоби машинного навчання і лінгвістичні технології, що збільшує достовірність результатів аналізу. Крім того, можна відзначити можливості динамічного аналізу, зручний інтерфейс і підтримку різних мов.» [5]

Lithium – дає можливість «будувати і використовувати соціальні спільноти на базі як Інтернету, так і мобільного телефону. Інструмент дозволяє відокремити сигнал від шуму, виділити, що саме ваші клієнти говорять про вас, зрозуміти, що їх хвилює, і дає можливість ознайомитися з їх думкою. Додаток здатний показати майданчики найбільш активного обговорення і визначити найбільш пристрасних ораторів. Lithium Social Media Monitoring знаходить нові ідеї щодо вдосконалення продуктів і звільняє їх від недоліків, виявляє нові можливості і шляхи вдосконалення послуг, а також дозволяє донести їх до співробітників компанії, щоб використовувати всі вигоди від розуміння соціальної інформації, отриманої від клієнтів.Customer Intelligence Center покликаний конвертувати розмови клієнтів і дані про їх поведінкових профілях в бізнес-переваги корпорації. Інструмент дозволяє збирати інформацію про соціальну поведінку клієнтів і забезпечувати поінформованість впливових прихильників, щоб підтримувати інтерес соціальної спільноти до бренду.» [6]

Аналіз наведених ресурсів та сервісів наданий у табл. 1.1.

Таблиця 1.1 – Порівняльна характеристика програмних продуктів

Назва продукту	Властивості продукту				
	Сентимент-аналіз	Доступна ціна	Можливість інтеграції	Робота з сайтами	Відкритий код
DiscoverText	+	+	-	+	+
Clarabridge	+	+	+	+	+
InMoment	+	+	+	-	-
SAS Sentiment Analysis	+	-	-	+	-
Lithium	+	-	-	+	-
Власний програмний засіб	+	+	+	+	+

Як можна побачити, більшість програмних ресурсів вимагають власної платформи для їх використання та значна частина з них є коштовними, тому їх незручно використовувати для вирішення поставленого завдання, бо це не дасть можливість аналізувати інші параметри готелів крім текстових відгуків.

Проведений аналіз доказує необхідність розробки власного програмного засобу для допомоги у виборі готелів з урахуванням аналізу тональності відгуків.

1.4 Висновки до розділу

У першому розділі виконано аналіз особливостей роботи готельного бізнесу. Розглянуто основні принципи аналізу тональності відгуків. Проведено огляд існуючих програмних аналогів та обґрунтовано актуальність розробки власного програмного засобу.

2 РОЗРОБКА ПРОГРАМНОГО ЗАСОБУ

2.1 Побудова моделі оцінювання готелю

Особливістю розроблюваного програмного засобу є використання власної моделі оцінювання готелів. Відповідно до оцінювання застосовано аспектно-орієнтований підхід. Він дозволяє включити у модель набори об’єктивних Oa та суб’єктивних Sa аспектів:

$$H = (<Oa, \{Sa\} \epsilon Oa>) \quad (2.1)$$

Під об’єктивними аспектами розуміється наявність чи відсутність певної складової готелю: ресторан, дитяча розважальна кімната, басейн, тощо.

Як можна побачити, для кожного об’єктивного аспекту ставиться у відповідність множина суб’єктивних аспектів. Під цими суб’єктивними аспектами розуміються фактично характеристики Oa, які можуть бути оцінені користувачами за певною шкалою. Тут, як і при використанні інших методів машинного навчання, якість значень характеристик визначає достовірність результату [7 – 9].

Окремими Sa є текстові відгуки користувачів, які аналізуються з визначенням певного емоційного окрасу відгуку.

Так як об’єктивні аспекти оцінювання відповідають наявності чи відсутності складових готелю, вони визначаються на двійковій шкалі: true, якщо складова є наявності чи false, якщо фізично складова не передбачена.

Якщо складова існує, то користувач може працювати з нею у двох режимах:

- у режимі туриста, який має опит мешкання у готелі, та може створювати оцінки за характеристиками цієї складової (суб’єктивними аспектами);
- у режимі туриста, який ще обирає готель, та може переглядати усереднені оцінки стосовно характеристик обраної складової.

На рис. 2.1 показано алгоритм роботи з об’єктивними аспектами оцінювання.

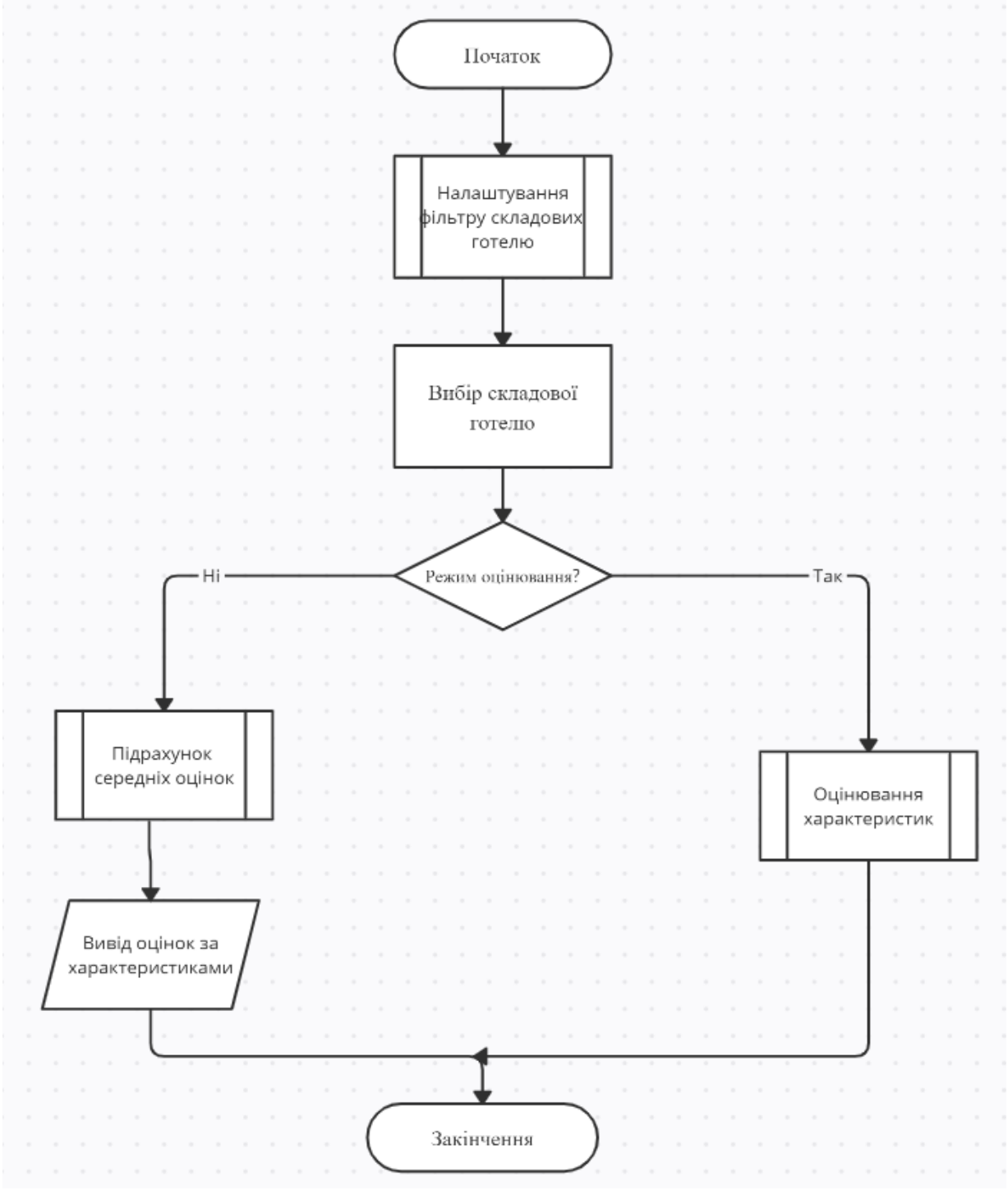

Рисунок 2.1 – Режими роботи зі складовими готелю

Згідно з алгоритмом, користувач застосовує фільтр, використання якого дозволяє обрати ті складові готелю, що його цікавлять. Потім згідно з фільтром обираються складові готелю. Якщо користувач бажає оцінити складову, він обирає режим оцінювання та відповідно до заданої шкали оцінювання надає власну оцінку складової.

Якщо користувач бажає переглянути вже існуючі оцінки за обраною складовою готелю, виконується підрахунок середніх оцінок за усіма характеристиками складової. Результати підрахунку надаються користувачеві.

Розроблюване програмне забезпечення бере за основу ту інформацію, яку можна отримати з відповідного сайту готелей. Тому набори конкретних складових готелю (об'єктивних аспектів) та їх характеристик (суб'єктивних аспектів) залежить цілком від структури та організації відповідних сайтів. В залежності від того, яка інформація взагалі передбачена для користувачів, і будуть формуватись аспекти оцінки готелів. Тому у подальшому можна говорити лише про типові набори аспектів.

Розглянемо деякі з них.

Для об'єктивного аспекту «ресторан» можна визначити наступні суб'єктивні аспекти:

- якість ресторанної їжі;
- якість обслуговування;
- швидкість обслуговування;
- привітність персоналу ресторану;
- рівень цін (якщо вартість не входить до основної вартості путівки);
- рівень шуму у ресторані.

Кожна з цих характеристик може бути оцінена за 5-бальною шкалою.

Визначимо ще декілька об'єктивних та пов'язаних з ними суб'єктивних аспектів:

- власна берегова пляжна зона: зовнішній вигляд території, стан зони, рівень чистоти, зручність під'їзду/підходу;
- зона відпочинку: рівень завантаженості, рівень чистоти, наявність шезлонгів та інших меблів для відпочинку, якість обслуговування у зоні відпочинку, стан навісів;
- автостоянка (парковка): зручність під'їзду, паркувальні апарати, якість покриття, рівень завантаженості;

– приміщення та кухонне обладнання місць для самостійного приготування і прийому їжі: зовнішній вигляд, рівень чистоти, укомплектованість сучасним кухонним обладнанням, рівень завантаженості, рівень пожежної безпеки, цілодобовий доступ.

На рис. 2.2 наведено узагальнену схему алгоритму для процедури оцінювання туристами характеристик складових готелю.

Рисунок 2.2 – Оцінювання роботи складових готелю

2.2Алгоритм векторизації текстових відгуків

У даній роботі для визначення емоційного забарвлення відгуків використовується метод навчання з вчителем, тобто для роботи класифікатора застосовуються попередньо розмічені дані [10 – 12].

Для того, щоб текст можна було подати на вхід класифікатору, потрібно виконати його векторизацію. У ході векторизації, в залежності від обраної технології, тексту ставиться у відповідність сукупність признаків, поданих у вигляді чисел. Для проведення векторизації можна використовувати наступні алгоритми: Topicmodeling, GloVe, One-hotencoding, SVD, FastTextта інші.

Але найбільш розповсюдженим є алгоритм Word2vec,що надає можливість визначення ступеня близькості значень слів аналізованого тексту в залежності від контексту слів. Для визначення ступеня близькості слів, представлених векторами, використовується косинусна схожість, яка для двох векторів A та B обчислюється за стандартною формулою:

$$similarity = \frac{(A, B)}{|A||B|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (2.2)$$

У ході навчання нейромережі завданням алгоритму Word2vec є максимізація косинусної схожісті між векторами тих слів, що знаходяться у схожих за сенсом контекстах, та, навпаки, мінімізація косинусної відстані між словами, що не розташовані поряд у контексті.

Нейромережа алгоритму Word2vecснавчається за методом backpropagation (зворотного поширення помилки), тобто корегуються ваги схованого слою W’, а потім вхідного слою W. Результатом роботи Word2vec є координати векторів для певних слів. На рис. 2.3 показано схему нейромережі, що визначає вектор для слова «y»за його «h»сусідами: x1, …, xk, …, xh.

Зазначимо, що реалізація алгоритму Word2vec вже давно відома, та немає необхідності самостійно програмувати нейромережу для проведення векторизації слів тексту. Для обраної у роботі мови програмування PythonWord2vec реалізовано у бібліотеці spaCy.

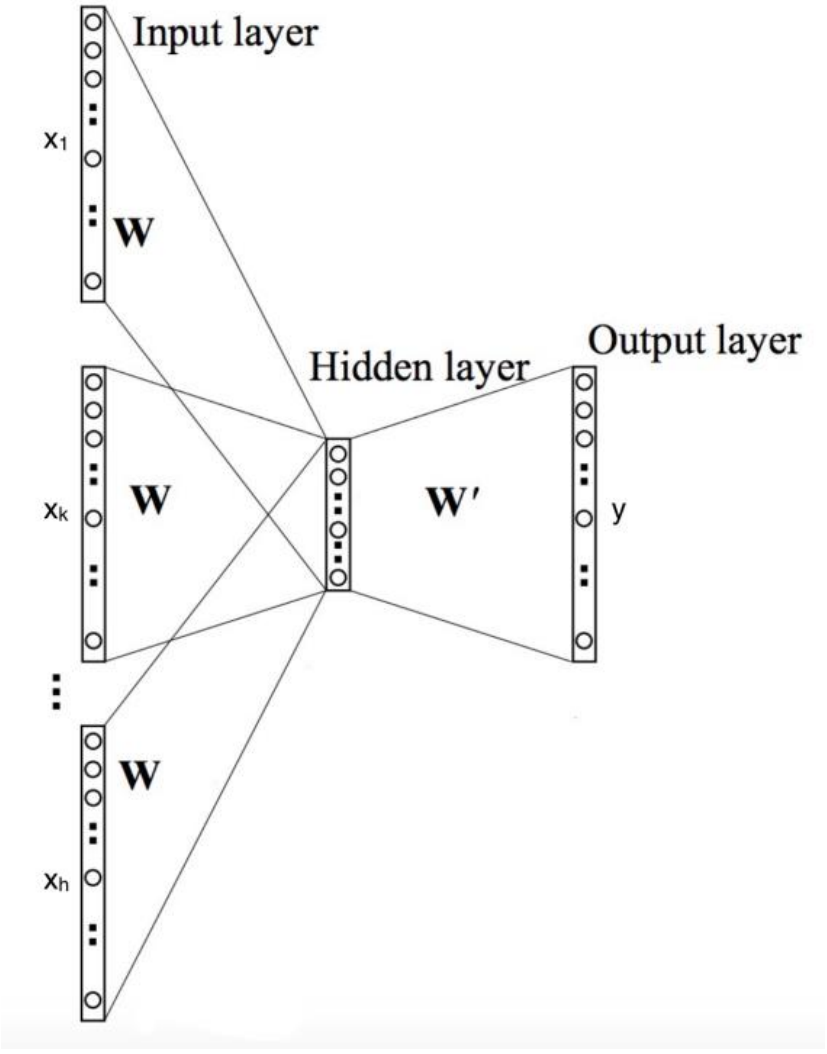

Рисунок 2.3 – Нейромережа роботи алгоритму Word2vec

У загальному випадку розмір словника може сягати великої кількості слів. Зрозуміло, що при цьому робота алгоритму буде вимагати багато часу, тому що метод backpropagation потребує обчислення градієнту на двох кроках. Саме тому у методи бібліотеці spaCy закладено способи оптимізації процесу векторизації слів для можливості швидкого використання великих словників (понад декількох сотен тисяч слів).

2.3 Застосування згорткової нейронної мережі для аналізу тональності відгуків

Використовувана у кваліфікаційній роботі бібліотека spaCy містить функції аналізу тональності текстів, засновану на застосуванні згорткової нейронної мережі [13].

Спочатку згорткові нейромережі застосовувались для розпізнавання зображень, але далі почали активно використовуватись у задачах прогнозування, класифікації та модифікації.

Архітектура згорткової нейромережі є одно направленою (тільки прямий напрям поширення сигналів активації), функції активації обираються за вимогою користувача. Нейромережа має багато шарів. Для навчання використовується метод backpropagation [14].

Відмінною рисою архітектури є операція згортки, яка виконує множення частини фрагменту матриці векторизованого тексту на ядро згортки елемент за елементом (рис. 2.4). Отриманий результат підраховується шляхом сумування та розміщується у відповідну комірку вихідних даних.

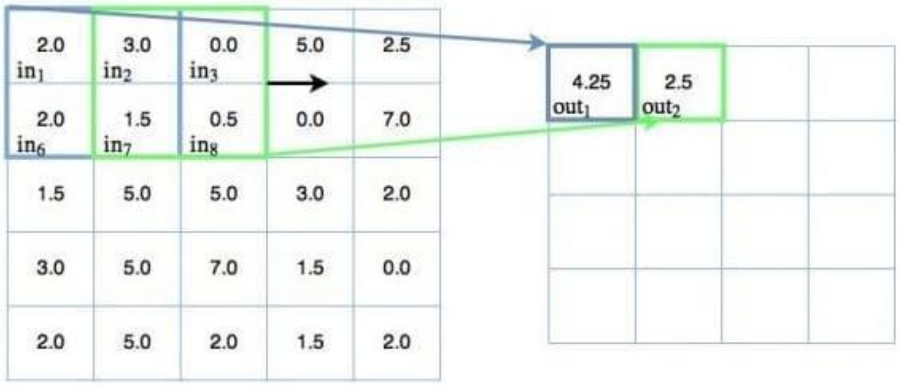

Рисунок 2.4 – Діаграма отримання вихідного значення

Приклад обчислення вихідних значень:

$$
\begin{aligned}
out_1 &= 0.5in_1 + 0.5in_2 + 0.5in_6 + 0.5in_7 \\
&= 0.5 \times 2.0 + 0.5 \times 3.0 + 0.5 \times 2.0 + 0.5 \times 1.5 \\
&= 4.25 \\
out_2 &= 0.5in_2 + 0.5in_3 + 0.5in_7 + 0.5in_8 \\
&= 0.5 \times 3.0 + 0.5 \times 0.0 + 0.5 \times 1.5 + 0.5 \times 0.5 \\
&= 2.5
\end{aligned}
$$

Для вибору певної архітектури згорткової нейронної мережі потрібно визначити тип вирішуваного завдання, існуючі обмеження, які можуть бути зазначені у системі (вимоги до точності результата, обмеження за рівнем технічного обладнання, швидкість отримання результату), а також визначити розмірність вхідних даних (розмір векторизованого текстового повідомлення), вихідних даних (кількість класів для визначення тональності тексту) та кількість внутрішніх шарів. Застосування різних функцій активації допомагає налаштувати потрібну точність результатів. Результатом стає діаграма вузлів нейромережі з певною кількістю вхідних вузлів, внутрішніх шарів та вихідних вузлів. Приклад фрагменту діаграми зображений на рис. 2.5.

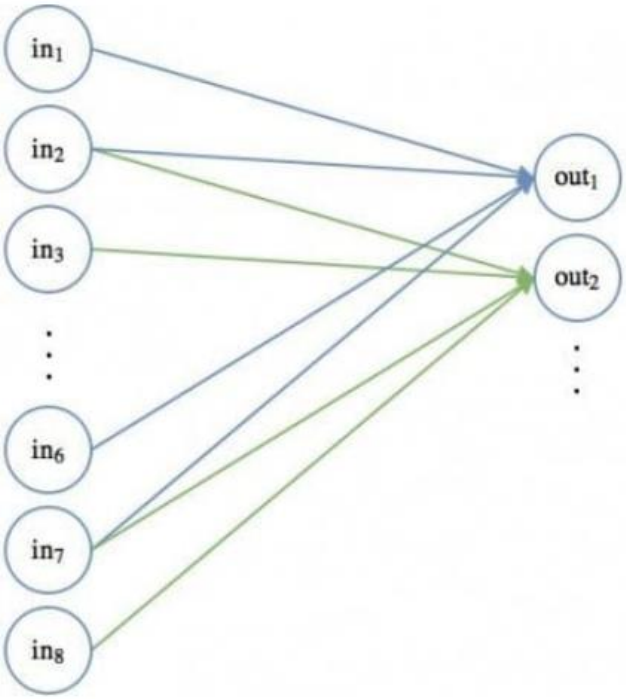

Рисунок 2.5 – Приклад фрагменту діаграми вузлів нейромережі

Складним завданням для побудови архітектури нейромережі є визначення кількості внутрішніх шарів. Додавання надлишкової кількості шарів приводить до забування мережею градієнта та різкого зростання кількості вагів, що потрібні для тренування мережі. Проблему забування можна вирішити з використанням функцій активації сімейства ReLU (Rectified Linear Unit - зрізаний лінійний вузол) функцій. Проблема зростання кількості вагів для тренування призводить до зростання часу на тренування та можливості перенавчання моделі мережі. Для вирішення цієї проблеми використовується визначена кореляція між значеннями єлементів векторизованого тексту, яка дозволяє не створювати повнозв'язну нейромережу та, тим самим, зменшити кількість використовуваних для тренування вагів.

2.4 Висновки до розділу

У другому розділі побудовано модуль оцінювання готелю, яка включає у себе об'єктивні та суб'єктивні аспекти. За об'єктивні аспекти оцінювання беруться складові готелю, які можуть бути фізично присутні чи відсутні (недосяжні) для туристів-користувачів. Під суб'єктивними аспектами розуміються оцінки характеристик складових готелю, які виставляються користувачами.

Крім того, враховується тональність текстового відгуку, створеного користувачами стосовно готелю взагалі. Для аналізу тональності запропоновано застосування Python-бібліотеки spaCy, що використовує згорткову нейронну мережу. Для можливості роботи spaCy попередньо потрібно провести векторизацію текстів відгуків, яку запропоновано проводити з використанням алгоритму Word2vec.

3 СПЕЦИФІКАЦІЯ ВИМОГ ДО ПРОГРАМНОГО ЗАСОБУ

3.1 Варіанти використанняпрограмного засобу

Для того, щоб формалізувати вимоги до розроблюваної програмної системи, був проведен аналіз інформації, наданої майбутніми потенційними користувачами програмного продукту. Користувачі розроблюваної системи поділяються на такі наступні категорії:

– «Абстрактний Користувач» (AbstractUser) – користувач розроблюваного програмногопродукту, актор найвищого рівня абстракції.

– «Менеджер» (Manager) – користувач, що працює з програмою у режимі підговки до роботи з певним сайтом та навчання нейромережі на основі парсингу його даних;

– «Користувач» (User) –звичайний користувач-турист, що може користуватись вже поставленими оцінками щодо роботи готелів, представлених на сайті, та залишати власні оцінки.

Формальний опис функціональних вимог представлено далі на рис. 3.1 у вигляді діаграми варіантів використання. На діаграмі наведено актори та виконувани ними прецеденти.

Так як розроблювальний програмний застосунок призначений для можливості використання будь-яким звичайним користувачем, які на слабкомурівні володіють обчислювальною технікою та не мають спеціальних навочок роботи з програмним забезпеченням, то цей застосунок повинен бути простим у використанні, тобто мати інтиутивно зрозумілий, зручнийта дружній інтерфейс. Наочність та простота інтерфейсу допоможе користувачеві сконцентрувати увагу саме на роботі з оцінками обраних ними готелів.

Для роботи з певним сайтом представлена на ньому інформація щодо готелів та їх оцінок повинна бути попередньо підготовлено, це здійснюється завдяки роботі парсеру. Парсер може бути реалізований програмним образом, або ж можна задіяти готові парсери web-сторінок.

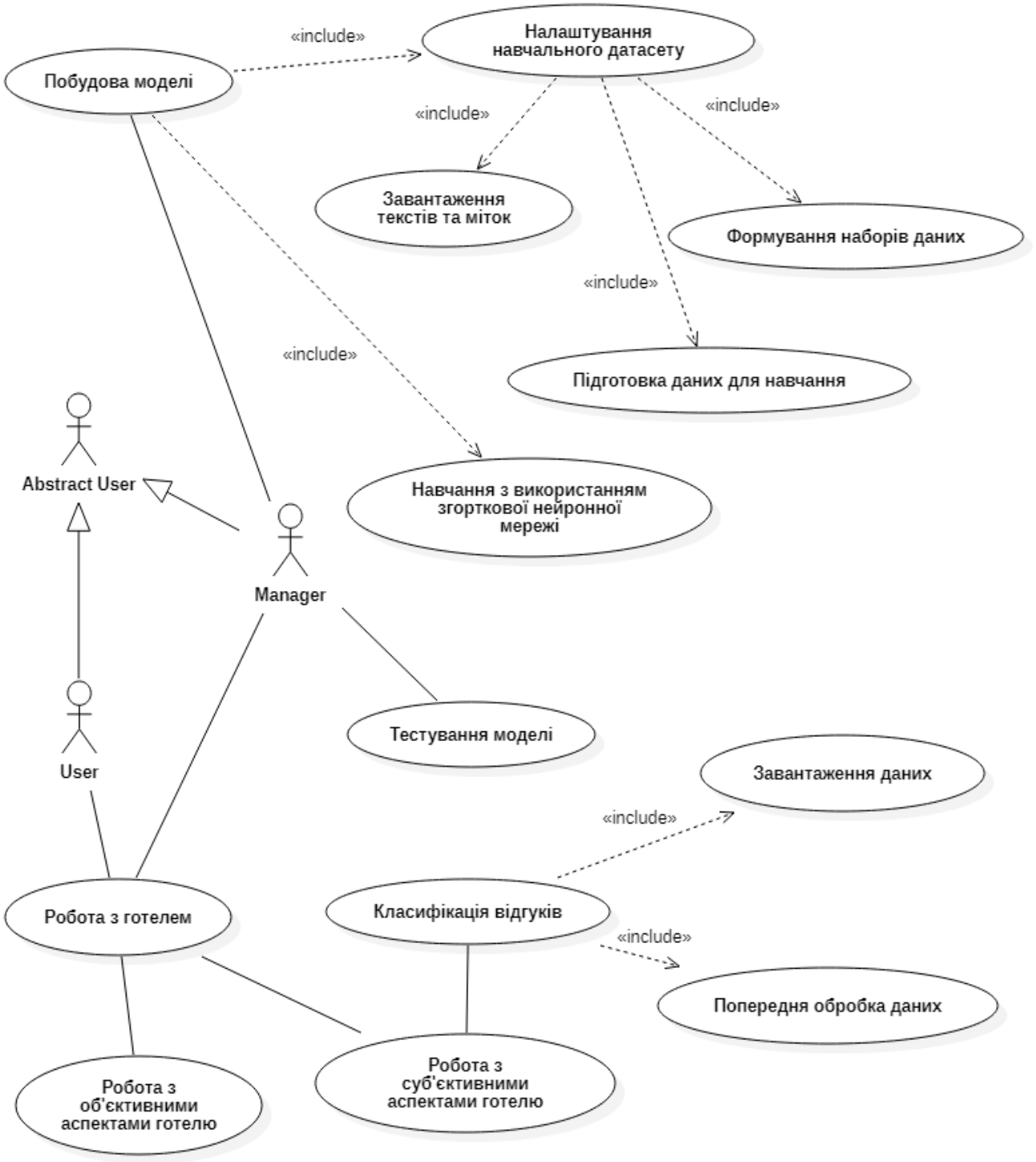

Рисунок 3.1 – Діаграма варіантів використання системи

Виконання основних дій у за стосунку має виконуватись за допомогою команд меню, кнопок піктограм, основним діям повинні відповідати підказки. У разі помилкових дій користувачеві потрібно надавати відповідні пояснення.

Далі наведемо специфікацію прецедентів, вказуючи у якості основних пунктів кроки основного сценарію, а у якості підпунктів – відповідні кроки альтернативного сценарію.

Специфікація прецедентів (ВВ – варіантів використання):

1. ВВ «*Формування наборів даних*»

Передумова: Програма запущена.

Тригер: Менеджер бажає налаштувати навчальний датасет.

1. Менеджер обирає сайт, за якого буде завантажуватись інформація.
2. Система перевіряє сайт на наявність усіх потрібних даних.
   2а. Дані щодо оцінок готелів не знайдно.
   2а.1 Система повідомляє про помилку.
3. Система виконує парсинг сайту даних.
4. Менеджер бажає зберегтиотримані дані.
   4а. Помилка збереження даних.
   4а.1 Система повідомляє пропомилку.
5. Система зберегла дані. Успіх.

2. *ВВ «Завантаження текстів та міток»*

Передумова: Дані успішно сформовані.

Тригер: Менеджер бажає налаштувати навчальний датасет.

1. Менеджер обирає режим завантаження розмічених текстів з відгуками.
2. Програма перевіряє цілісність файлів з даними.
   2а. Файл пошкоджено.
   2а.1 Перехід до ВВ 1.
3. Програма перевіряє чи всі тексти розмічені.
   3а. Є нерозмічені тексти.
   3а.1 Видаляються нерозмічені тексти.
4. Програма перевіряє коректність розмітки.

4а. Знайдені порушення розмітки текстів.

4а.1 Видалити тексти з некоректною розміткою.

5. Програма завантажує розмічені тексти як навчальний датасет для аналізу тональності відгуків.Успіх.

3. *ВВ «Підготовка даних для навчання»*

Передумова: Дані успішно завантажені.

Тригер: Менеджер бажає налаштувати навчальний датасет.

1 Програма визначає параметри моделі.

1а. Параметри помилкові.

1а.1 Вивід повідомлення про помилку.

2 Програма виконує перемішування даних для навчання.

3 Програма розділяє дані на певні частини для навчання-верифікації-тестування.

4 Програма виводить повідомлення про успіх.

4. *ВВ «Налаштування навчального датасету»*

Передумова: Набори даних сформовані та завантажені, дані підготовлені для навчання.

Тригер: Менеджер бажає налаштувати навчальний датасет.

1. Програма виконує формування наборів даних, для цього викликає виконання ВВ1.

2. Програма завантажує розмічені тексти відгуків, для цього викликає виконання ВВ2.

3. Програма виконує підготовку даних для навчання моделі мережі, для цього викликає виконання ВВ3.

4. Програма фіксуєналаштування датасету, завантажує його у систему тана носій.

4а. Помилка запису на носій.

4а.1 Вивід повідомлення про помилку.

5. Програма виводить повідомлення про успіх.

5. *ВВ «Навчання з використанням згорткової нейронної мережі»*

Передумова: Навчальний датасет налаштований.

Тригер: Менеджер бажає визначити параметри моделі класифікатора.

1. Менеджер обирає режим навчання моделі.
2. Системастворює новий сеанс навчання.
3. Система виконує навчання моделі з корегуванням її параметрів з використанням зворотного поширення помилки.
4. Системаперевіряє значення помилки, у разі перевищення припустимого значення помилки корегування вагів повторюється, для цього відбувається перехід до п. 3.
5. Система зберігає результат.

6. *ВВ «Побудова моделі»*

Передумова: Навчальний датасет налаштований.

Тригер: Менеджер бажає побудувати модель класифікатора.

1. Менеджер обирає режим побудови моделі.
2. Системастворює новий сеанс роботи з моделлю.
3. Система завантажує параметри моделі на основі згорткової нейронної мережі.

3а. Помилка у завантаженні.

3а.1 Повідомлення про помилку.

3а.2 Перехід до п. 2.

4. Системаперевіряє значення параметрів.

4а. Помилка у параметрах чи параметрів недостатньо.

4а.1 Повідомлення про параметричні помилки.

4а.2 Перехід до п. 2.

5. Система створює модель.
6. Система зберігає результат.

7. *ВВ «Тестування моделі»*

Передумова: Модель побудовано.

Тригер: Менеджер бажає провести тестування моделі.

1. Менеджер обирає модель для тестування.

   1а. Помилка при завантаженні моделі.

   1а.1 Повідомлення про помилку завантаження.

2. Менеджер обирає параметри тестування.

   2а. Помилка в параметрах тестування.

   2а.1 Повідомлення про помилку.

   2а.2 Перехід до п.1.

3. Система виконує тестування та визначає точність моделі.

   3а. Помилка в даних, повідомлення про помилку.

   3а.1 Перехід до п.1.

4. Система надає результати менеджеру.

5. Система зберігає дані щодо тестування.

8. *ВВ «Робота з об'єктивними аспектами готелю»*

Передумова: Дані щодо готелю є в наявності на сайті.

Тригер: Користувач чи менеджер бажає працювати зі складовими готелю, який представлений на сайті.

1. Користувач виконує налаштування фільтру складових готелю. У фільтрі він обирає ті складові, які представляють для нього інтерес та впливають на здійснення вибору.

2. З зазначених у фільтрі складових користувач обирає певну складову.

3. Користувач обирає режим роботи: режим оцінювання складової чи отримання інформації про складову.

4. Якщо обрано режим отримання інформації про складову, то система завантажує оцінки по характеристикам обраної складової готеля та обчислює середні оцінки.

4а. Помилка при завантаженні інформації про складову.

4а.1 Вивід повідомлення. Перехід до п. 9.

5. Система виводить розраховані середні оцінки за розглянутими характеристиками.

6. Перехід до п. 9.

7. Виконується оцінювання складової за її характеристиками. Для цього програма викликає виконання ВВ12.

8. Збереження даних. Система повідомляє про успіх.

8а. Збереження неможливе.

8а.1 Вивід повідомлення про неможливість збереження даних про складові готелю.

9. Якщо залишились нерозглянуті складові – перехід до п. 2.

10. Завершення роботи.

9. ВВ «*Попередня обробка даних*»

Передумова: Обрано готель, на сайті готелю містяться відгуки.

Тригер: Користувач чи менеджер бажає визначити тональність відгуків до готелю.

1. Система завантажує web-сторінку з відгуками.

2. Система визначає дані щодо оцінок готелю та текстові відгуки.

2а. Дані неповні чи помилкові.

2а.1 Вивід повідомлення про помилки.

2а.2Перехід до п. 4.

3. Система виконує парсинг даних.

3а. Помилки у ході парсингу оцінок.

3а.1 Вивід повідомлення про неможливість визначення даних про оцінки складові готелю.

3б. Помилки у ході парсингу текстових відгуків.

3б.1 Вивід повідомлення про неможливість визначення даних про текстові відгуки.

4. Завершення роботи.

10. ВВ «*Завантаження даних*»

Передумова: Дані щодо оцінок та текстових відгуків успішно оброблені (проведено парсинг).

Тригер: Користувач чи менеджер бажає визначити тональність відгуків до готелю.

1. Система перевіряє дані, отримані після парсингу.
2. Система завантажує отримані дані у формат xml.

2а. Помилка завантаження.

2а.1 Система відміняє завантаження.

3. Система створює файл з оцінками та відгуками. При необхідності користувач чи менеджер можуть користуватись інформацією з цього файлу.
4. Завершення роботи.

11. ВВ «*Класифікація відгуків*»

Передумова: Дані щодо оцінок та текстових відгуків успішно оброблені та завантажені.

Тригер: Користувач чи менеджер бажає визначити тональність відгуків до готелю.

1. Система завантажує дані у форматі xml, отримані після парсингу.
2. Система виконує векторизацію даних з застосуванням алгоритму Word2vec.

2а. Помилка визначення similarity.

2а.1 Система вважає визначення близькості векторів невдалим.

3. Система визначає ваги нейромережі за методом backpropagation.

3а. Помилка перевищує припустиму.

3а.1 Повернення до п. 2.

4. Система вимагає зменшення часу на обчислення градієнту.

4а. Зменшення є критичним.

4а.1 Повернення до п. 2.

4б. Для зменшення потрібно змінити ядро згортки.

4б.1 Обрано інше ядро згортки.

4б.2 Аналіз часу, що потенційно потрібен для обчислення градієнту.

4б.3Якщо зменшення доси є критичним, повернення до п. 2.

5. Визначення точності результата.
6. Визначення обмеження за рівнем технічного обладнання.

6а. Потрібно покращити технічне обладання.

6а.1 Повідомлення про зміни обладання.

7. Визначення швидкість отримання результату.
8. Визначення розміру векторизованого текстового повідомлення.
9. Визначення кількість класів для визначення тональності тексту.

9б. Потрібно змінити кількість класів.

9б.1 Повідомлення про зміни.

10. Визначення кількості внутрішніх шарів.
11. Завершення роботи.

12. ВВ «*Робота з суб'єктивними аспектами готелю»*

Передумова: Готель, для якого потрібно працювати з оцінками характеристик (суб'єктивними аспектами), визначений.

Тригер: Користувач чи менеджер бажає працювати з оцінками характеристик певного готеля.

1. Система запускає виконання ВВ8, який визначає перелік складових готелю. Система завантажує для кожної складової перелік характеристик.
2. Незалежно від встановлених складових система завантажує текстові відгуки користувачів.

2а. Помилка завантаження даних.

2а.1 Система повідомляє про помилку завантаження.

3. Система надає користувачеві перелік характеристик для оцінювання.
4. Система завантажує шкали оцінювання для характеристик складових.

5. Система надає можливість створити оцінки та збережує їх.

5а. Оцінювання не зроблено.

5а.1 Система повідомляє про помилку оцінювання.

5а.2Повертання до п. 3.

5б. Помилка збереження даних.

5б.1 Система повідомляє про помилку.

5б.2Перехід до п. 6.

6. Завершення роботи.

13. ВВ «*Робота з готелем*»

Передумова: Готель визначений. Його складові та їх характеристики визначені.

Тригер: Користувач чи менеджер бажає здійснити роботу з готелем.

1. Система запускає роботу ВВ8 для роботи з складовими готелю.
2. ВВ8 визначає роботу ВВ12.
3. Система зберігає дані користувачів для готелю.
4. Завершення роботи.

3.2 Вимоги до нефункціональних характеристиксистеми

Обмеження

Впровадження системи має займати понад 3 місяців. Користування веб-застосунком не потребує спеціальних апаратно-програмних ресурсів.

Показники якості

Застосованість

– Час, необхідний навчання користувачів – 1 робочий день (10-15 хвилин), навчання менеджера – 1 робочий день (1година).

– Час відгуку для встановлення власної оцінки – не більше 3 секунд, для отримання узагальнених оцінок – не більше 5 секунд.

Функціональність

Здатність до взаємодії – веб-застосунокповинен взаємодіяти зі сховищем даних, для отримання інформації про сутності системи.

Захищеність – дані у БД захищені розділенням прав користувачів.

Точність – повинна бути достатньою для збереження значень характеристик готелю.

Переносимість

Адаптованість – програмний продукт є веб-застосуванням, він підтримує кроссплатформеність, його можна буде запускати на будь-якій платформі, яка має вихід до мережі Інтернет. Веб-застосунокповинен працювати однаково вбраузерах Google Chrome, Mozilla Firefox,Safari.

Зручність використання

Зручність роботи – веб-застосунокповинен мати простий і зрозумілий інтерфейс.

Ефективність

Часові характеристики – збереження інформації про дані користувачів веб-застосункуповинно виконуватися менше, ніж 3 секунди, а збереження змін повинно виконуватися менше, ніж за 1 секунду.

Використання ресурсів – веб-застосунокне повиненвикористовувати більше 1024 МБ оперативної пам’яті під час роботи.

Надійність

– Доступність – час, який витрачається на обслуговування веб-застосунку, не повинен перевищувати 5% від загального часу роботи.

– Середній час безвідмовної роботи – 90 робочих днів.

– Максимальна норма помилок або дефектів – 30 помилок на 10500 рядків коду.

3.3 Загальні системні вимоги

Стандарти, що застосовуються

Система повинна відповідати всім стандартам сучасних веб-застосувань.

Системні вимоги

Мінімальні системні вимоги:

– 8GB RAM
– 32GB вільного місця
– Процесор із тактовою частотою 1.6 ГГц або вище

Експлуатаційні вимоги

Веб-застосунокповинен бути здатним підтримувати щонайменше 1000 одночасно працюючих користувачів, пов'язаних із загальною базою даних і мати можливість збільшити їх кількість до 10000 на випадок збільшення навантаження.

Вимоги до документації

– Інструкція користувача

У системі мають бути представлені посібники користувачів (за типами користувачів – у даній системи 2 типи користувачів: звичайний користувач та менеджер).

Вони повинні містити розшифровку всіх термінів, що використовуються, опис загальних можливостей системи, опис основних варіантів використання, включаючи альтернативні сценарії, докладний огляд інтерфейсу веб-застосування, а також опис відомих помилок та можливих шляхів щодо їх усунення.

Інтерактивна довідка

Інтерактивна довідка необхідна для вирішення питань, що виникли під час роботи. У довідці також має бути реалізована можливість пошуку інформації за ключовими словами, а також варіант подання інформації щодо окремих позицій меню програми. Довідка повинна містити максимально повну та детальну інформацію щодо роботи веб-застосування.

3.4 Висновки до розділу

У третьому розділі роботи визначено специфікації до веб-застосунку. За допомогою варіантів використання формально описано функціональні. Усього визначено 13 варіантів використання веб-застосунку.

Створено вимоги до нефункціональних характеристик системи, а саме: обмеження, переносимість, зручність використання, ефективність, надійність та інші.

У загальних системних вимогах визначено застосовані стандарти, системні вимоги, експлуатаційні вимоги та вимоги до документації.

4 ПРОЕКТУВАННЯ ЗАСОБУ ДЛЯ ДОПОМОГИ У ВИБОРІ ГОТЕЛІВ

4.1 Проектування архітектури системи

Дана програмна система являє собою веб-застосування і серверну частину, що взаємодіє з базою даних, що в сукупності надають гнучкі інструменти для роботи з готовими оцінками готелів, проведення оцінювання і подальшого аналізу отриманих даних з використанням методів обробки даних.

Високорівнева структура системи складається з трьох частин, а саме: Web Application, Server і Database. Зв’язки між ними показані на рис. 4.1.

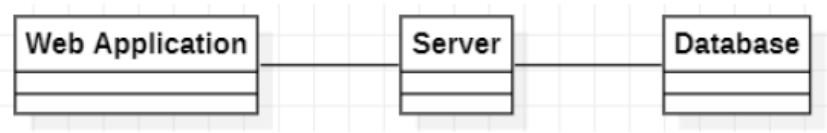

Рисунок 4.1 – Високорівнева архітектура системи

Як видно з діаграми, структура розроблюваного продукту є клієнт-серверною, в якій клієнтом є веб-застосування.

Частина Web Application має засоби управління всім функціоналом користувача, містить інтерфейс, з яким взаємодіє користувач, виводить необхідні дані користувачу і реалізує зв’язок з сервером.

Частина Server реалізує обмін даними між базою даних і веб-застосуванням, а також виконує розрахунки результатів оцінювання після завершення процедури надання оцінок.

Частина Database є базою даних, в якій зберігаються всі дані системи: інформація про користувачів, оцінки і узагальнені результати по обраним готелям.

Веб застосування побудоване за технологією SPA (односторінкове застосування).

На відміну від традиційного підходу, коли на кожну дію користувача відбувається перезавантаження і чергове виведення сторінки, в односторінковому застосуванні завантаження програми в браузер відбувається один раз і далі все взаємодія з сервером йде у фоновому режимі, у даному випадку за допомогою

технології Virtual DOM. Тобто, коли користувач працює з елементами управління, не відбувається повного перезавантаження сторінки. Необхідна порція інформації підвантажується і оновлюються конкретні елементи сторінки (вузли DOM-дерева).

Різниця в підходах зображена на рис. 4.2 (традиційний підхід) та 4.3 (підхід SPA).

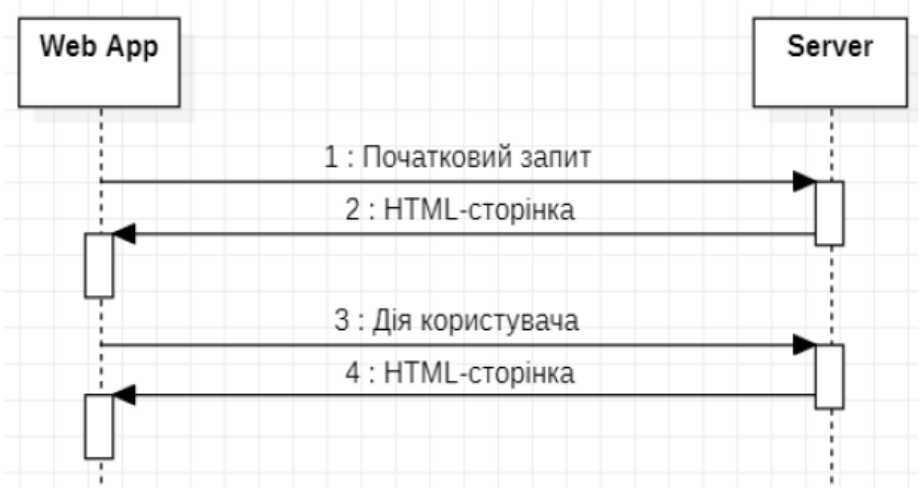

Рисунок 4.2 – Традиційний підхід до взаємодії веб-застосувань з сервером

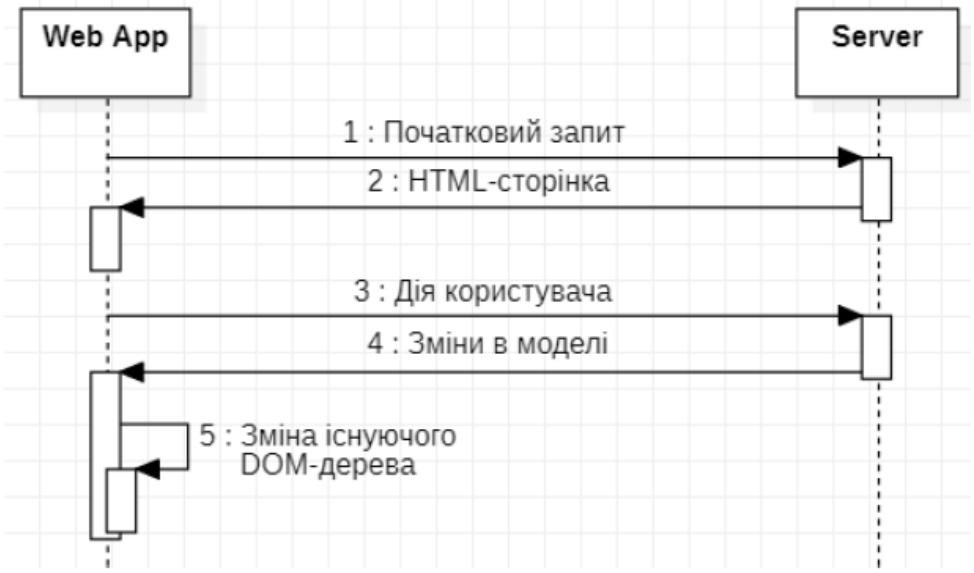

Рисунок 4.3 – Підхід SPA

Таким чином, при SPA оновлюються окремі вузли існуючої Document Object Model.

4.2 Основні діаграми роботи системи

На рис. 4.4 та 4.5 наведені діаграми послідовності для роботи з текстовими відгуками туристів щодо готелю.

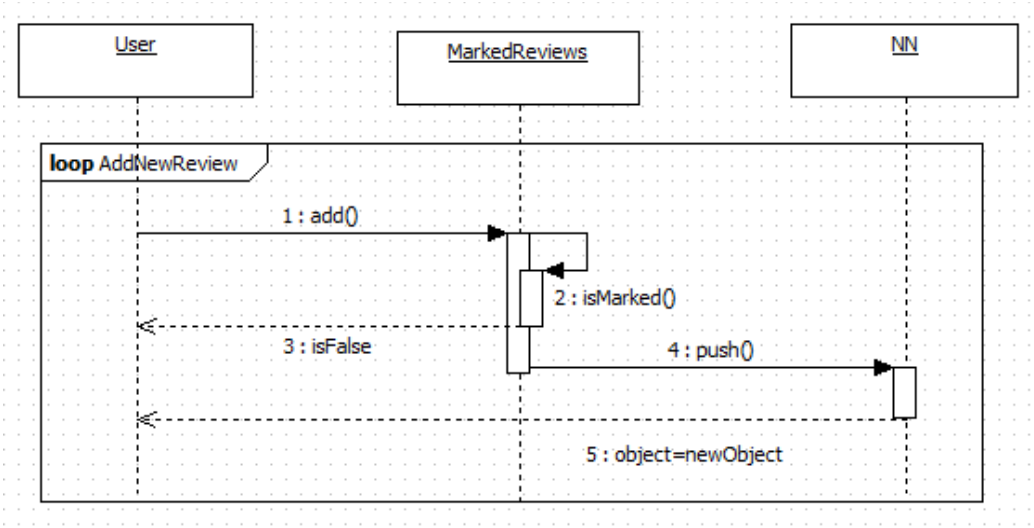

Рисунок 4.4 – Діаграма послідовності для додавання розмічених відгуків

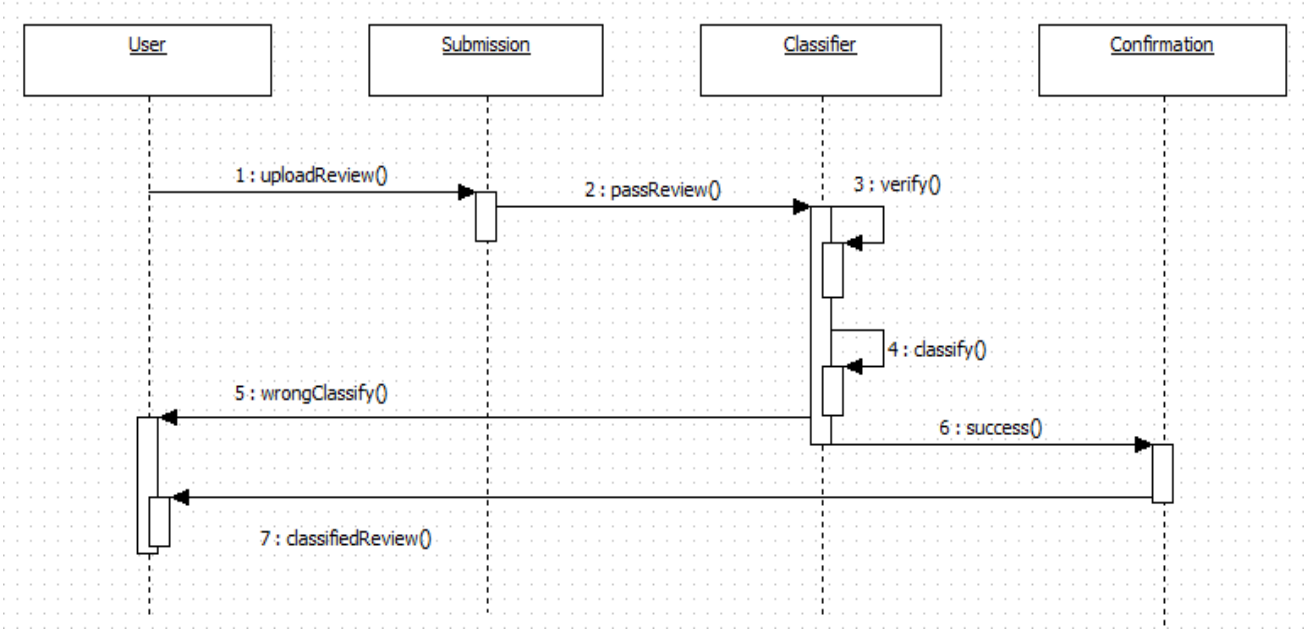

Рисунок 4.5 – Узагальнена діаграма послідовності для класифікації відгуку

На рис. 4.6 наведена більш детальна схема процесу визначення тональності відгуків.

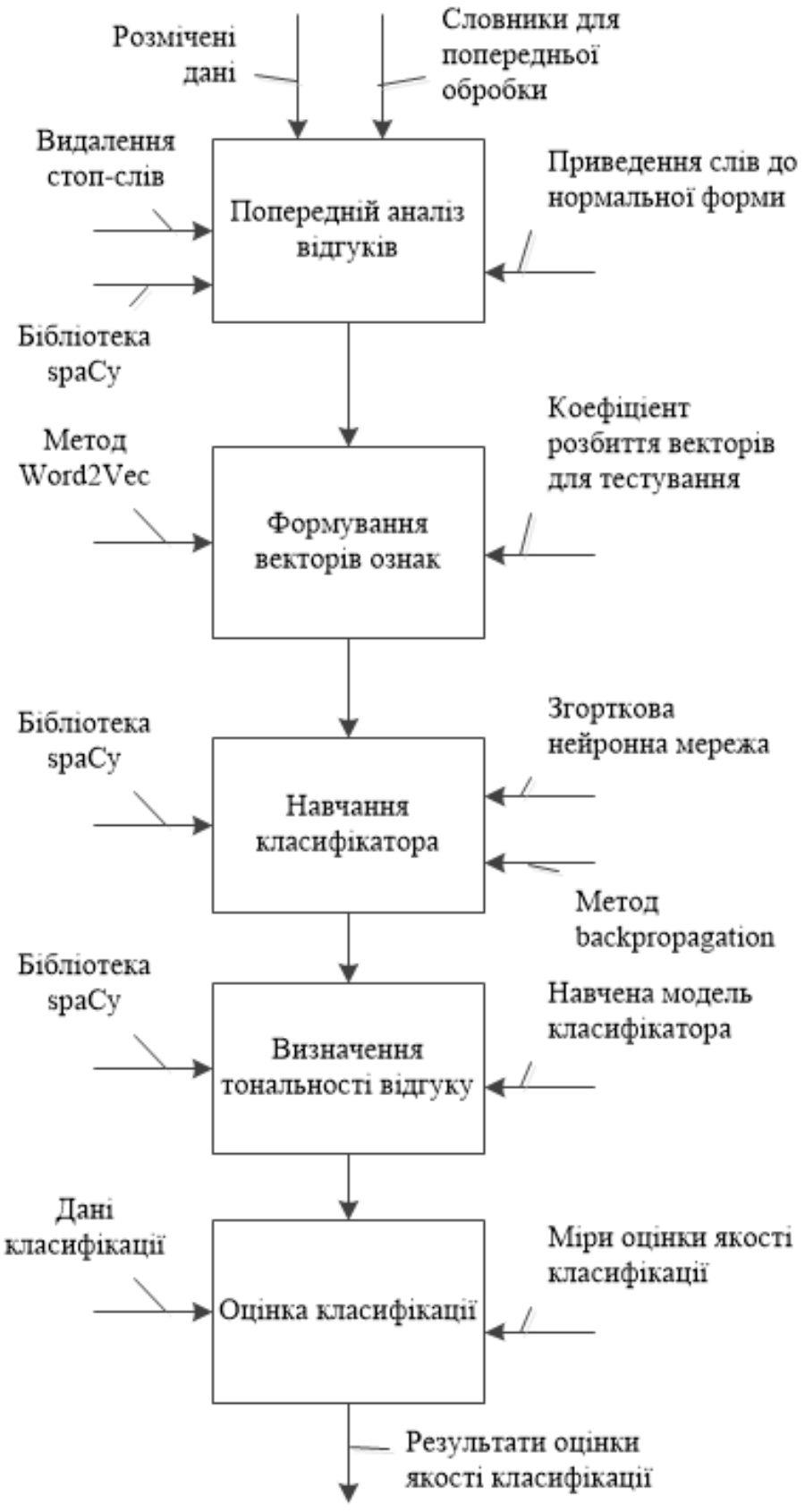

Рисунок 4.6– Схема процесу визначення тональності відгуків

Як зазначено на схемі, спочатку проводиться попередній аналіз відгуків. При цієї обробці на вхід блоку потрапляють розмічені дані, тобто відгуки, помічені як позитивні чи негативні, та словники. Словники використовуються для того, щоб у подальшому виконувати приведення слів до нормальної форми та видаляти стоп-слова. Стоп-словами звуться такі слова, які при тому, що мають важливе значення у спілкуванні людей, не надають корисної інформації при аналізі тональності тексту. Для виконання попереднього аналізу відгуків використовується бібліотека spaCy.

Бібліотека spaCy постачається зі списком стоп-слів за замовчуванням (рис. 4.7). Цей список при необхідності можна налаштувати.

['whence', 'here', 'show', 'were', 'why', 'n't', 'the', 'whereupon', 'not', 'more', 'how', 'eight', 'indeed', 'i', 'only', 'via', 'nine', 're', 'themselves', 'almost', 'to', 'already', 'front', 'least', 'becomes', 'thereby', 'doing', 'her', 'together', 'be', 'often', 'then', 'quite', 'less', 'many', 'they', 'ourselves', 'take', 'its', 'yours', 'each', 'would', 'may', 'namely', 'do', 'whose', 'whether', 'side', 'both', 'what', 'between', 'toward', 'our', 'whereby', "'m", 'formerly', 'myself', 'had', 'really', 'call', 'keep', "'re", 'hereupon', 'can', 'their', 'eleven', 'm', 'even', 'around', 'twenty', 'mostly', 'did', 'at', 'an', 'seems', 'serious', 'against', "n't", 'except', 'has', 'five', 'he', 'last', "'ve", 'because', 'we', 'himself', 'yet', 'something', 'somehow', 'm', 'towards', 'his', 'six', 'anywhere', 'us', "'d", 'thru', 'thus', 'which', 'everything', 'become', 'herein', 'one', 'in', 'although', 'sometime', 'give', 'cannot', 'besides', 'across', 'noone', 'ever', 'that', 'over', 'among', 'during', 'however', 'when', 'sometimes', 'still', 'seemed', 'get', "'ve", 'him', 'with', 'part', 'beyond', 'everyone', 'same', 'this', 'latterly', 'no', 'regarding', 'elsewhere', 'others', 'moreover', 'else', 'back', 'alone', 'somewhere', 'are', 'will', 'beforehand', 'ten', 'very', 'most', 'three', 'former', "'re", 'otherwise', 'several', 'also', 'whatever', 'am', 'becoming', 'beside', 's', 'nothing', 'some', 'since', 'thence', 'anyway', 'out', 'up', 'well', 'it', 'various', 'four', 'top', 's', 'than', 'under', 'might', 'could', 'by', 'too', 'and', 'whom', "'ll", 'say', 'therefore', "'s", 'other', 'throughout', 'became', 'your', 'put', 'per', "'ll", 'fifteen', 'must', 'before', 'whenever', 'anyone', 'without', 'does', 'was', 'where', 'thereafter', "'d", 'another', 'yourselves', 'n't', 'see', 'go', 'wherever', 'just', 'seeming', 'hence', 'full', 'whereafter', 'bottom', 'whole', 'own', 'empty', 'due', 'behind', 'while', 'onto', 'wherein', 'off', 'again', 'a', 'two', 'above', 'therein', 'sixty', 'those', 'whereas', 'using', 'latter', 'used', 'my', 'herself', 'hers', 'or', 'neither', 'forty', 'thereupon', 'now', 'after', 'yourself', 'whither', 'rather', 'once', 'from', 'until', 'anything', 'few', 'into', 'such', 'being', 'make', 'mine', 'please', 'along', 'hundred', 'should', 'below', 'third', 'unless', 'upon', 'perhaps', 'ours', 'but', 'never', 'whoever', 'fifty', 'any', 'all', 'nobody', 'there', 'have', 'anyhow', 'of', 'seem', 'down', 'is', 'every', "'ll", 'much', 'none', 'further', 'me', 'who', 'nevertheless', 'about', 'everywhere', 'name', 'enough', "'d", 'next', 'meanwhile', 'though', 'through', 'on', 'first', 'been', 'hereby', 'if', 'move', 'so', 'either', 'amongst', 'for', 'twelve', 'nor', 'she', 'always', 'these', 'as', "'ve", 'amount', "'re", 'someone', 'afterwards', 'you', 'nowhere', 'itself', 'done', 'hereafter', 'within', 'made', 'ca', 'them']

Рисунок 4.7 – Стоп-слова, що видаляються з системи

Наступним етапом виконується векторизація текстових відгуків з формуванням векторів ознак. Для цього, як було зазначено у розділі 2, використовується метод Word2Vec. На цьому ж етапі визначаються коефіцієнти розбиття векторів для тестування. За умовчанням на навчання припадає 80% даних, на тестування та перевірку якості результату класифікації – 20%.

Далі виконується навчання класифікатора. Як було вказано вище, для цього використовується згорткова нейронна мережа. Застосування методом зворотного розповсюдження помилки слугує для навчання багатошарового перцептрону

завдяки ітеративному градієнтному алгоритму. Він дозволяє знизити помилки роботи нейромережі та отримати задовільні результати класифікації.

Навчена модель класифікатора використовується для визначення тональності відгуку, для цього задіяні теж функції бібліотеки spaCy.

Результати класифікації оцінюються згідно з визначеними мірами оцінки якості класифікації. Традиційно визначається точність класифікації.

Система розподілена на два окремих модуля, що доповнюють одне одного та інтегровані у проект SignatureScanTool: підсистема класифікації та підсистема навчання, яка створена для генерації класифікаторів, що підключаються до модуля класифікації.

На основі зазначених діаграм та схем створено діаграму концептуальних класів (рис. 4.8), що потім була деталізована до діаграми програмних класів.

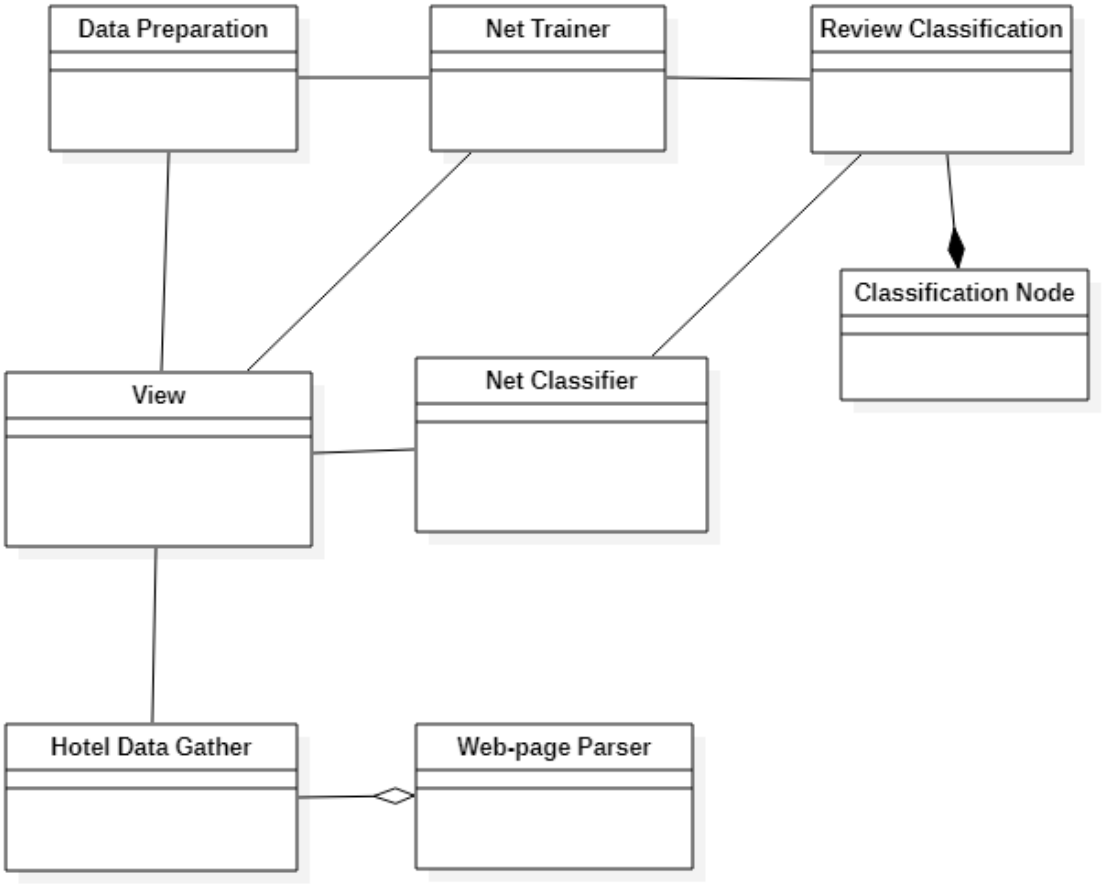

Рисунок 4.8 – Діаграма концептуальних класів

Наведемо короткий опис концептуальних класів:

DataPreparation –попередня обробка даних для подальшої класифікації;

NetTrainer – допоміжній клас для тренування нейромережі;

ReviewClassification – основний клас для проведення класифікації та визначення точності результатів;

NetClassifier – визначення тональності відгуку на основі навченої моделі класифікатора;

ClassificationNode – клас для управління конвеєром для обробки певного відгуку;

View – клас для забезпечення взаємодії з користувачами;

HotelDataGather – клас для збирання з сайту інформації щодо обраного готелю;

Web-pageParser – парсер веб-сторінки з готелями.

4.3 Визначення структур даних

Усі дані, яківикористовує веб-застосунок, представляються у форматі JSON (JavaScript Object Notation). Це полегшує роботу з веб-сторінками та дозволяє створювати ефективні запити до даних.

Загальна структура даних JSON, що описує готель, виглядає наступним чином:

```
{
  "name": String,
  "URL": String,

  "address": {
     "city": String,
     "postalCode": Int,
     "country": String
  },
```

```
  "phoneNumbers": [
      String,
     …
      String
  ],
"reviews": [
      String,
     …
      String
  ],
  "hotelComponents":[
 {
  "componentName": String,
"componentCharacteristics": [
{
"componentCharacteristicName": String,
"componentCharacteristicValue": [
{
 "scale": Int,
"value": Int
}
…
"scale": Int,
"value": Int
},
  …
,
{
"componentCharacteristicName": String,
"componentCharacteristicValue": Int
```

```
        }
       ]
       },
       …
       {
       "componentName": String,
       "componentCharacteristics": [
       {
       "componentCharacteristicName": String,
       "componentCharacteristicValue": [
       {
 "scale": Int,
       "value": Int
       }
       …
"scale": Int,
       "value": Int
       },
         …
       {
       "componentCharacteristicName": String,
       "componentCharacteristicValue": Int
        }
       ]
       }
}
```

Наведемо короткий опис формату:

name – назва готелю;

URL – інтернет-адреса;

address – фізична адреса готелю;

phoneNumbers – номера телефонів для зв’язку;

reviews – відгуки туристів;

hotelComponents – складові готелю;

componentName – назва складової;

componentCharacteristics – характеристики складової;

componentCharacteristicName – назва характеристики складової;

componentCharacteristicValue – оцінки характеристики складової(за певною шкалою);

scale – шкала оцінювання (кількість рівнів, наприклад, для оцінок за варіантами «дуже погано», «погано», «нормально», «добре», «дуже добре» значення scale=5);

value – оцінка характеристики складової конкретною особою.

Для навчання класифікатора використовується JSON-файл з наступною структурою:

```
{
  "documents": [
    {
      "id": Int,
      "language": String,
      "text": String,
       "value": Int
    },
…
{
      "id": Int,
      "language": String,
      "text": String,
      "value": Int
    }
  ]
```

}

Опис наведеного формату:

id – ідентифікатор певного відгука;

language – мова відгука;

text – текст наданого відгука;

value – тональність текстового відгука.

4.4 Проектування структури класів програмного засобу

На рис. 4.9 наведена діаграма програмних класів, яка містить класи: DataPreparation, MainClassifier, ReviewClassification, ClassificationNode, NetTrainer, View, HotelValueGather, HotelReviewsGather, WebPageParser.

Наведемо зв’язки між вказаними класами.

Класи MainClassifier та ReviewClassification пов’язані між собою відношенням узагальнення (generalization), тому що клас ReviewClassification наслідує атрибути та методи класа MainClassifier.

Класи ReviewClassification та ClassificationNode пов’язані між собою відношенням композиції, тому що ClassificationNode входить у ReviewClassification як необхідна для функціонування частина, що не може існувати без наявності ReviewClassification.

Клас HotelValueGather пов’язаний з класом WebPageParser відношенням агрегації, тому що WebPageParser потрібен як складова для роботи HotelValueGather, але може існувати й без нього.

Аналогічно клас HotelReviewsGather пов’язаний відношенням агрегації з класом WebPageParser.

Всі інші відношення між класами є асоціативними.

Розглянем більш детально атрибути та методи визначених програмних класів.

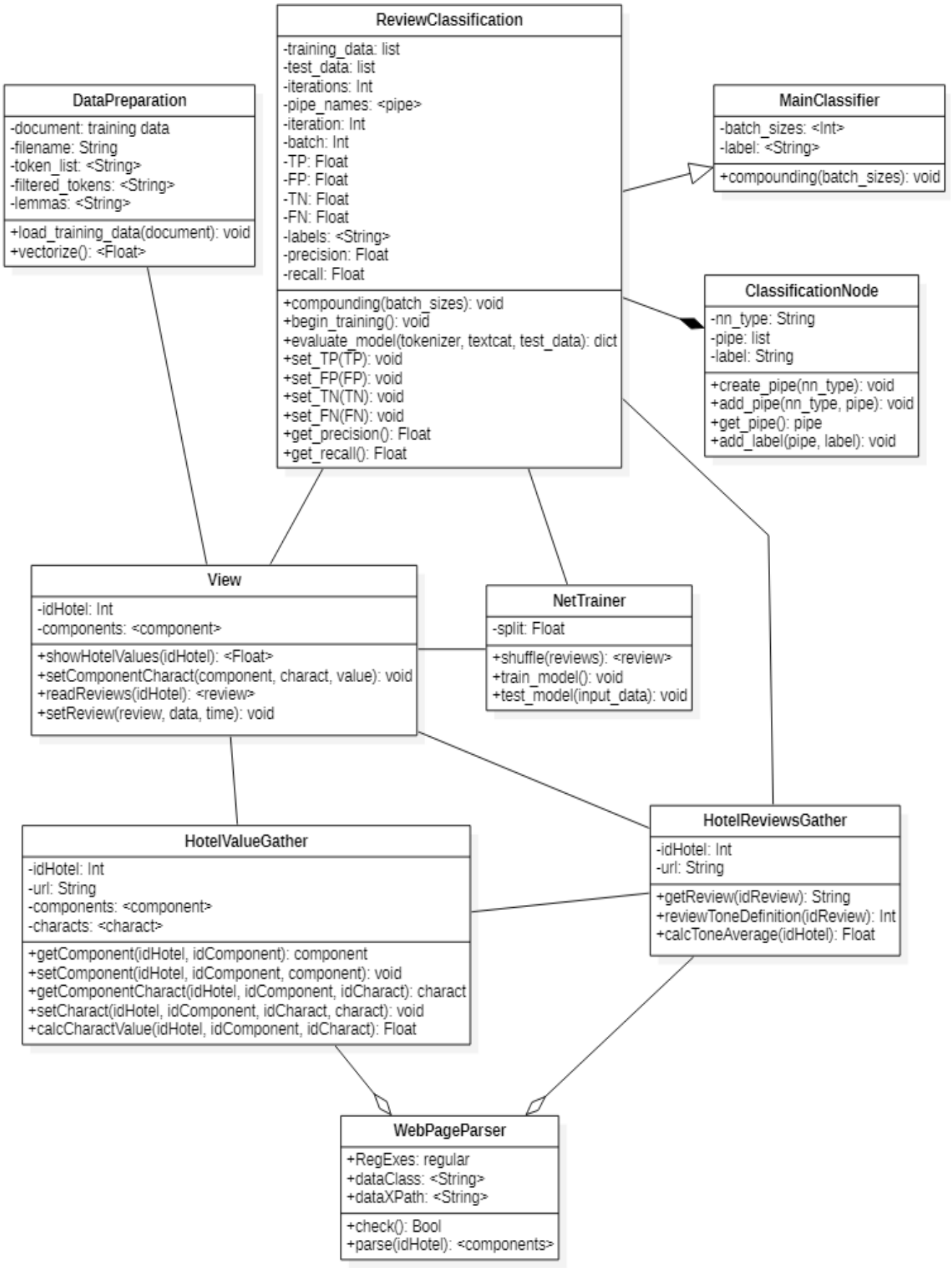

Рисунок 4.9 – Діаграма програмних класів

Клас DataPreparation:

document: training data – документ, що містить дані для тренування (у форматі JSON-файлу);

filename: String – повний шлях до файлу document;

token_list: <String> - список токенів (окремих слів), визначених з текстового відгуку;

filtered_tokens: <String> - токени, відфільтровані від стоп-слів;

lemmas: <String> - список лем – відфільтрованих токенів, приведених до нормальної форми слів;

load_training_data(document): void – метод завантаження тренувальних даних у класифікатор;

vectorize(): <Float> - метод векторизації лем з отриманням числового масива перед тренуванням класифікатора.

Клас MainClassifier:

batch_sizes: <Int> - розміри батчей (пакетів даних), які приймають участь у навчанні нейромережі;

label: <String> - припустимі мітки оцінки тональності відгуку, у нашому випадку labelприймає значення з набору ["pos", "neg"];

compounding(batch_sizes): void – метод для пришвидшення навчання класифікатора з використанням бібліотеки spaCy.

Клас ReviewClassification:

training_data: list – частина даних, що використовується для тренування класифікатора;

test_data: list – частина даних, що використовується для тестування роботи тренованого класифікатора;

iterations: Int – ітерації навчання нейромережі;

pipe_names: <pipe> - список компонентов для конвеєра для навчання нейромережі;

iteration: Int – поточна ітерація;

batch: Int – поточна порція даних;

TP: Float – true positives - істинно позитивні: кількість позитивних відгуків, які модель класифікатора правильно передбачила як позитивні;

FP: Float – false positives – хибно позитивні: кількість негативних відгуків, які модель передбачила як позитивні, хоча це було помилкою;

TN: Float – true negatives - істинно негативні: кількість негативних відгуків, які модель класифікатора правильно передбачила як негативні.

FN: Float – false negatives –хибно негативні: кількість позитивнихвідгуків, які модель передбачила як негативні, хоча це було помилкою;

labels: <String> - мітки відгуків;

precision: Float – точність класифікації,що визначається як відношення істинно позитивних результатів (TP) ко всім результатам, що були спрогнозовані як позитивні (TP+FP);

recall: Float – полнота класифікації, що визначається як відношення істинно позитивних результатів (TP) ко всім результатам, що були спрогнозовані як істинно позитивні та хибно негативні (TP+FN).

compounding(batch_sizes): void – перевантажений метод для пришвидшення навчання класифікатора з використанням бібліотеки spaCy.

begin_training(): void – метод для початку тренування мережі.

evaluate_model(tokenizer, textcat, test_data): dict – функція, що виконує розділ відгуків та їх міток, подальшу токенізацію відгуків та перебір токенізованих елементів;

set_TP(TP): void – встановлення початкового значення true positives;

set_FP(FP): void – встановлення початкового значення false positives;

set_TN(TN): void–встановлення початкового значення true negatives;

set_FN(FN): void – встановлення початкового значення false negatives;

get_precision(): Float – отримання значення точності;

get_recall(): Float – отримання значення повноти.

Клас ClassificationNode:

nn_type: String – тип нейромережі;

pipe: list – компонент для конвеєра;

label: String – мітка для навчання класифікатора;

create_pipe(nn_type): void – створення вбудованого компонента конвеєра «textcat»;

add_pipe(nn_type, pipe): void – додавання компонента «textcat» до мережі nn_type;

get_pipe(): pipe – отримання поточного компонента «textcat»;

add_label(pipe, label): void – додавання мітки до тексту.

Клас NetTrainer:

split: Float –співвідношення навчальних та тестових даних;

shuffle(reviews): <review> - перемішуванняданих;

train_model(): void –метод для завантаження вбудованого конвеєра та перевірки наявності компонента «textcat», у разі відсутності «textcat» він створюється та розміщується у кінці конвеєра;

test_model(input_data): void – створення моделі для генерації прогнозу, прогнозне значення отримаємо у атрибуті «cats», що належить до змінної «parsed_text».

Клас View:

idHotel: Int – ідентифікатор готелю, з інформацією до якого працює користувач;

components: <component> - складові готелю;

showHotelValues(idHotel): <Float> - метод відображення характеристик складових обраного готелю;

setComponentCharact(component, charact, value): void – метод встановлення значень характеристик складових готелю;

readReviews(idHotel): <review> - метод, що дозволяє перегляд відгуків;

setReview(review, data, time): void – метод, що дозволяє користувачеві створити власний відгук.

Клас HotelValueGather:

idHotel: Int – ідентифікатор готелю;

url: String – електронний ресурс сайта з інформацією про готелі;

components: <component> - перелік складових готелю;

characts: <charact> - перелік характеристик складової готелю;

getComponent(idHotel, idComponent): component – метод для отримання певної складової idComponent для обраного готелюidHotel;

setComponent(idHotel, idComponent, component): void – метод для встановлення певного значення componentскладової idComponentдля обраного готелюidHotel;

getComponentCharact(idHotel, idComponent, idCharact): charact – метод для отримання значення певної характеристики idCharact складової idComponent для готеля idHotel;

setCharact(idHotel, idComponent, idCharact, charact): void – метод длявстановленняпевного значення charactхарактеристики idCharact складової idComponent для готеля idHotel;

calcCharactValue(idHotel, idComponent, idCharact): Float – метод обчислення усередненої оцінки для характеристики idCharact складової idComponent для готеля idHotel.

Клас HotelReviewsGather:

idHotel: Int – ідентифікатор готелю;

url: String – електронний ресурс сайта з інформацією про готелі;

getReview(idReview): String – метод отримання відгуку;

reviewToneDefinition(idReview): Int – метод визначення тональності відгуку за ідентифікатором idReview;

calcToneAverage(idHotel): FloatidHotel– метод обчислення узагальненої оцінки по відгукам для готелю idHotel.

Клас WebPageParser:

RegExes: regular – регулярний вираз для пошуку елементів сайту;

dataClass: <String> - клас, до якого належить певний елемент сайту;

dataXPath: <String> - XPath для посилання на певний елемент сайту;

check(): Bool – перевірка коректності даних сайту;

parse(idHotel): <components> - парсинг даних сайту з отриманням компонент певного готелю.

4.5 Висновки до розділу

У розділі «Проектування засобу для допомоги у виборі готелів» визначено архітектуру розроблюваної системи та створено основні діаграми, що відобраєують її структуру та динаміку роботи.Розроблено схему процесу визначення тональності відгуків. Визначено перелік стоп-слів, що видаляються з системи.

Визначено структури даних у форматі JSON: загальна структура даних, що описує готель, та структура даних для навчання класифікатора.

Виконано проектування та наведено опис концептуальних та програмних класів системи.

5 ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАСТОСУВАННЯ

5.1 Особливості створення програми за допомогою обраних інструментів розробки

У даному розділі опишемо особливості використання обраної мови програмування Python та середовища розробки.

На сьогоднішній день існує дуже велика кількість мов програмування, але саме Python можна вважати найпопулярнішим та універсальним середовищем розробки програмного забезпечення [15, 16]. З використанням Python можнавирішувати різноманітні типи завдань: скрипти, веб-розробки, ігри та ін.

Python є кросплатформеною мовою, що інтерпретується; інтерпретатори Python існують для багатьох платформ [17]. Тому не виникає труднощів з запуском Python на будь-якій операційній системі.

До переваг Python можна віднести наступні [18]:

– Python є простою мовою та підходить як перша мова програмування;

– існує великий набір середовищ для розробки програм на Python, вони мають можливість підключати та завантажувати спеціальні бібліотеки для спрощення реалізації багатьох функцій при робробці власних проектів;

– мови Python є затребуваною на ринку праці, та вона постійно оптимізується та розвивається з релізом кожною нової версії, покращення стосуються як оптимізації старих функцій, так і появі нових програмних можливостей;

– Python потребує менше часу на розробку програмного забезпечення, ніж може знадобитись при використанні іншій мови програмування.

Для програмування системи було використано IDE PyCharm, що надає зручний інтерфейс роботи з Python. Але у наступному підрозділі для наочності демонстрації основних етапів обробки текстових відгуків, а також у шостому розділі при тестуванні моделі класифікатора, було використано інтерпретатор jupyter-ноутбук та показано які саме команди Python виконуються.

5.2 Підключення та використання бібліотеки spaCy

Open-source бібліотека spaCy використовується для обробки природної мови та надає спеціальне програмне забезпечення для можливості розробки проектів.

Для підключення бібліотеки використовуються наступні команди (рис. 5.1):

```
pip install spacy
python -m spacy download en_core_web_sm
```

Рисунок 5.1 – Інсталяція spaCy

На рис. 5.2 показано використання вбудованого завантажувача spaCy для попередньо навченої статистичної мовної моделі для роботи з текстом англійською мовою. Використання методу nlp (Natural language processing) дозволяє виконати токенізацію тексту [19].

```
In [2]: import spacy
nlp = spacy.load("en_core_web_sm")

text = """
I booked this hotel because the price was reasonable and there was a pool.
After reading a few negative reviews, I thought it would not matter
to me as I am quite calm and we planned not to be in the hotel most
of the time. However, the problems were not long in coming.
I booked bed and breakfast, but the lady at the front desk said
that breakfast was not included. Since she was not very helpful,
I had to figure it out on my own. The indoor pool smelled and looked
like it hadn't been cleaned since it opened! The air conditioner
leaked when we left it on, and every time there was a puddle
of water on the floor. Hot water pressure practically did not exist.
I had to ask twice to be looked at. Needless to say, I was very upset.
Wouldn't recommend staying there.
"""

doc = nlp(text)
token_list = [token for token in doc]

print(token_list)

[
, I, booked, this, hotel, because, the, price, was, reasonable, and, there, was, a, pool, .,
, After, reading, a, few, negative, reviews, ,, I, thought, it, would, not, matter,
, to, me, as, I, am, quite, calm, and, we, planned, not, to, be, in, the, hotel, most,
, of, the, time, ., However, ,, the, problems, were, not, long, in, coming, .,
, I, booked, bed, and, breakfast, ,, but, the, lady, at, the, front, desk, said,
, that, breakfast, was, not, included, ., Since, she, was, not, very, helpful, ,,
, I, had, to, figure, it, out, on, my, own, ., The, indoor, pool, smelled, and, looked,
, like, it, had, n't, been, cleaned, since, it, opened, !, The, air, conditioner,
, leaked, when, we, left, it, on, ,, and, every, time, there, was, a, puddle,
, of, water, on, the, floor, ., Hot, water, pressure, practically, did, not, exist, .,
, I, had, to, ask, twice, to, be, looked, at, ., Needless, to, say, ,, I, was, very, upset, .,
, Would, n't, recommend, staying, there, .,
]
```

Рисунок 5.2 – Приклад токенізації відгуку на готель

Коли spaCy розбиває відгук на токени за допомогою nlp, він отримує при цьому об'єкт Doc, який складається з набору об'єктів класу Token та іншій інформації [20]. Для визначення та подальшого видалення з токенізованих відгуків стоп-слів використовується конструкція token.is_stop (рис. 5.3):

```
In [3]: filtered_tokens = [token for token in doc if not token.is_stop]
        print(filtered_tokens)

[
, booked, hotel, price, reasonable, pool, .,
, reading, negative, reviews, ,, thought, matter,
, calm, planned, hotel,
, time, ., ,, problems, long, coming, .,
, booked, bed, breakfast, ,, lady, desk, said,
, breakfast, included, ., helpful, ,,
, figure, ., indoor, pool, smelled, looked,
, like, cleaned, opened, !, air, conditioner,
, leaked, left, ,, time, puddle,
, water, floor, ., Hot, water, pressure, practically, exist, .,
, ask, twice, looked, ., Needless, ,, upset, .,
, recommend, staying, .,
]
```

Рисунок 5.3 – Приклад відгуку, очищеного від стоп-слів

Після цього spaCy приводить токени, що залишились у відгуку, до первинної форми з використанням стандартної процедури лематизації (рис. 5.4).

```
In [4]: lemmas = [
            f"Token: {token}, lemma: {token.lemma_}"
            for token in filtered_tokens
        ]

        print(lemmas)

['Token: \n, lemma: \n', 'Token: booked, lemma: book', 'Token: hotel, lemma: hotel', 'Token: price, lemma: price', 'Token: reas
onable, lemma: reasonable', 'Token: pool, lemma: pool', 'Token: ., lemma: .', 'Token: \n, lemma: \n', 'Token: reading, lemma: r
ead', 'Token: negative, lemma: negative', 'Token: reviews, lemma: review', 'Token: ,, lemma: ,', 'Token: thought, lemma: thin
k', 'Token: matter, lemma: matter', 'Token: \n, lemma: \n', 'Token: calm, lemma: calm', 'Token: planned, lemma: plan', 'Token:
hotel, lemma: hotel', 'Token: \n, lemma: \n', 'Token: time, lemma: time', 'Token: ., lemma: .', 'Token: ,, lemma: ,', 'Token: p
roblems, lemma: problem', 'Token: long, lemma: long', 'Token: coming, lemma: come', 'Token: ., lemma: .', 'Token: \n, lemma:
\n', 'Token: booked, lemma: book', 'Token: bed, lemma: bed', 'Token: breakfast, lemma: breakfast', 'Token: ,, lemma: ,', 'Toke
n: lady, lemma: lady', 'Token: desk, lemma: desk', 'Token: said, lemma: say', 'Token: \n, lemma: \n', 'Token: breakfast, lemma:
breakfast', 'Token: included, lemma: include', 'Token: ., lemma: .', 'Token: helpful, lemma: helpful', 'Token: ,, lemma: ,', 'T
oken: \n, lemma: \n', 'Token: figure, lemma: figure', 'Token: ., lemma: .', 'Token: indoor, lemma: indoor', 'Token: pool, lemm
a: pool', 'Token: smelled, lemma: smell', 'Token: looked, lemma: look', 'Token: \n, lemma: \n', 'Token: like, lemma: like', 'To
ken: cleaned, lemma: clean', 'Token: opened, lemma: open', 'Token: !, lemma: !', 'Token: air, lemma: air', 'Token: conditioner,
lemma: conditioner', 'Token: \n, lemma: \n', 'Token: leaked, lemma: leak', 'Token: left, lemma: leave', 'Token: ,, lemma: ,',
'Token: time, lemma: time', 'Token: puddle, lemma: puddle', 'Token: \n, lemma: \n', 'Token: water, lemma: water', 'Token: floo
r, lemma: floor', 'Token: ., lemma: .', 'Token: Hot, lemma: hot', 'Token: water, lemma: water', 'Token: pressure, lemma: pressu
re', 'Token: practically, lemma: practically', 'Token: exist, lemma: exist', 'Token: ., lemma: .', 'Token: \n, lemma: \n', 'Tok
en: ask, lemma: ask', 'Token: twice, lemma: twice', 'Token: looked, lemma: look', 'Token: ., lemma: .', 'Token: Needless, lemm
a: needless', 'Token: ,, lemma: ,', 'Token: upset, lemma: upset', 'Token: ., lemma: .', 'Token: \n, lemma: \n', 'Token: recomme
nd, lemma: recommend', 'Token: staying, lemma: stay', 'Token: ., lemma: .', 'Token: \n, lemma: \n']
```

Рисунок 5.4 – Приклад лематизації токенів відгуку

Далі лематизовані токени перетворюються в унікальні числові значення, тобто виконується векторизація, обчислюється по вектору на кожен токен. У бібліотеці spaCy вектора є щільними, тобто не створюються нульові порожні значення, що дозволяють прискорити обробку нерозрідженого масиву. Для векторизації застосовується метод nlp(), що обчислює вектора з використанням атрибуту vector. На рис. 5.5 застосовано атрибут .vector для другого токена у списку filter_tokens. У наборі відповідно до відгуку (см. рис. 5.3) це є лемою «book» (англ. – «замовляти» дієслово)

```
In [5]: filtered_tokens[1].vector

Out[5]: array([-0.17592722,  0.06095813, -1.4293706 , -0.18826613,  0.84499943,
                1.4389664 , -1.3379372 , -0.5899961 ,  0.1628955 ,  0.29713255,
               -0.19300655,  0.7718655 ,  1.571498  , -0.3328234 , -0.19415773,
                0.93991953,  0.2785424 , -0.01855116, -0.9492949 , -0.8256474 ,
                1.0163538 ,  0.03457074,  1.1341486 , -0.32692268, -0.2122661 ,
               -0.21722402, -0.7426927 ,  0.20807073, -0.01519477, -0.9966668 ,
                0.68437046,  0.02990232, -0.80066985, -0.86858577, -0.950673  ,
               -0.46122164,  0.8290591 ,  0.82245594, -0.4408849 , -0.68238837,
                0.6057755 , -0.05789411,  0.03246182, -0.01307476, -0.06263418,
               -0.04667979, -0.45334956,  0.3654241 ,  0.06148691,  0.14956266,
                0.17141461,  0.42302227,  1.022785  ,  1.8514408 ,  0.4565221 ,
               -0.64137393, -0.1291435 , -0.0490946 , -0.14588568,  0.41110134,
                0.38033962,  0.87609226, -0.57277554, -0.09274889,  1.3609868 ,
               -0.6406636 , -0.2974056 , -0.14703968, -0.301465  ,  0.4007138 ,
                0.06845524,  0.41050386,  0.03420545,  0.8215891 , -0.38295692,
                0.811915  , -0.6548831 , -0.49080163, -0.57124   , -0.72643524,
               -0.1546426 , -0.8904324 , -0.3976854 , -0.5175861 , -0.01561791,
               -1.1672353 ,  0.12356131, -0.2144843 , -0.564918  , -0.58439386,
               -0.2111254 ,  1.5473728 ,  0.40290937, -0.45929417,  0.53584266,
               -1.0877639 ], dtype=float32)
```

Рисунок 5.5 – Приклад векторизації лематизованих токенів відгуку по готелю

На рис. 5.6 показано перехід до каталогу з даними для навчання та перевірки класифікатора.

```
In [6]: import os
        import tarfile

        os.chdir(os.path.relpath('../../../Datasets/'))

        fname = 'aclImdb_v1.tar.gz'
        with tarfile.open(fname, "r:gz") as tar:
            tar.extractall()
            tar.close()
```

Рисунок 5.6 – Застосування розмічених даних для навчання класифікатора

Після цього за допомогою методу load_training_data завантажуються розмічений текст та відповідні йому мітки. Дані перемішуються та розділяються на два набори – навчальний набір та тестовий, – після чого ці набори повертаються як результат роботи методу (рис. 5.7).

```
In [7]: def load_training_data(
            data_directory: str = "aclImdb/train",
            split: float = 0.8,
            limit: int = 0
        ) -> tuple:
```

Рисунок 5.7 – Формування навчального та тестового наборів даних класифікатора

Далі набори даних завантажуються у список та фактично створюється структура каталогів файлів даних (рис. 5.8). Потім до списку відгуків на готелі додається кортеж вмісту та, крім того, словник тегів (вимога формату моделі spaCy під час навчання). HTML-теги «br/» перетворюються на символи нового рядка та видаляються несуттєві символи пробілів на початку та наприкінці рядків.

```
In [8]: # одиночными символами решетки здесь и далее помечен код,
        # добавленный или изменившийся в сравнении с предыдущим кодом
        import os                                                          #

        def load_training_data(
            data_directory: str = "aclImdb/train",
            split: float = 0.8,
            limit: int = 0) -> tuple:
            # Загрузка данных из файлов
            reviews = []                                                   #
            for label in ["pos", "neg"]:                                   #
                labeled_directory = f"{data_directory}/{label}"            #
                for review in os.listdir(labeled_directory):               #
                    if review.endswith(".txt"):                            #
                        with open(f"{labeled_directory}/{review}") as f:   #
                            text = f.read()                                #
                            text = text.replace("<br />", "\n\n")          #
                            if text.strip():                               #
                                spacy_label = {                            #
                                    "cats": {                              #
                                        "pos": "pos" == label,             #
                                        "neg": "neg" == label              #
                                    }                                      #
                                }                                          #
                                reviews.append((text, spacy_label))        #
```

Рисунок 5.8 – Формування списку відгуків reviews

Після того, як файли з відгуками вже завантажені, виконується їх тасування для усунення впливу порядку завантаження відгуку на результат класифікації (рис. 5.9).

```
In [9]: import random

        random.shuffle(reviews)

        if limit:
            reviews = reviews[:limit]
        split = int(len(reviews) * split)
        return reviews[:split], reviews[split:]
```

Рисунок 5.9 – Тасування відгуків reviews

Після цього об'єкт Doc обробляється за деяку кількість кроків (це є «pipeline»). Зазвичай модель pipeline вміщує компоненти «tagger», «parser» та «entity recognizer».

При цьому компоненти pipeline приймають на вхід Doc-об'єкт та додають йому атрибути (рис. 5.10). Для цього додається компонент textcat та вказуються мітки даних: «pos» для позитивних відгуків щодо готелю та «neg» для негативних відгуків. Якщо компонент textcat не є доступним, то з використанням методу create_pipe() такий компонент створюється, додається у кінець конвеєра та отримує словник з відповідною конфігурацією.

```
In [10]: import os
         import random
         import spacy

         def train_model(
             training_data: list,
             test_data: list,
             iterations: int = 20
         ) -> None:
             # Строим конвейер
             nlp = spacy.load("en_core_web_sm")
             if "textcat" not in nlp.pipe_names:
                 textcat = nlp.create_pipe(
                     "textcat", config={"architecture": "simple_cnn"}
                 )
                 nlp.add_pipe(textcat, last=True)
```

Рисунок 5.10 – Додавання компоненту textcat

Якщо компонент «textcat» є доступним, то додаються мітки відгуків: позитивна «pos» та негативна «neg» (рис. 5.11).

```
In [11]: import os
         import random
         import spacy

         def train_model(
             training_data: list,
             test_data: list,
             iterations: int = 20
         ) -> None:
             # Строим конвейер
             nlp = spacy.load("en_core_web_sm")
             if "textcat" not in nlp.pipe_names:
                 textcat = nlp.create_pipe(
                     "textcat", config={"architecture": "simple_cnn"}
                 )
                 nlp.add_pipe(textcat, last=True)
             else:                                    #
                 textcat = nlp.get_pipe("textcat")    #

             textcat.add_label("pos")                 #
             textcat.add_label("neg")                 #
```

Рисунок 5.11 – Додавання міток при існуванні «textcat»

Щоб навчання класифікатору потрібно налаштувати конвеєр для навчання компонента «textcat», для цього генеруються пакети даних та використовуються функції compounding() та minibatch() з пакету spacy.util (рис. 5.12). Власне, батч – це невелика частина таких даних, що використовуються у процесі навчання.

Також в методі створюємо загальний список таких компонентів цього конвеєру, які зараз не є компонентами «textcat», та налаштовуємо розміри батчі за допомогою compounding(). Це фактично є генератором нескінченої послідовності для вхідних даних, що використовуються для навчання та перевірки класифікатора.

Після цього викликається метод nlp.begin_training(), що повертає саме початковий оптимізатор, який використовується nlp.update() у якості оновлення вагів для базової моделі.

Далі застосовується метод compounding(), який створює генератор та задає послідовність розмірів batch_sizes, які поступають на вхід методу minibatch() бібліотеки spaCy (рис. 5.13).

In [12]:
```
    # Обучаем только textcat
    training_excluded_pipes = [
        pipe for pipe in nlp.pipe_names if pipe != "textcat"
    ]
    with nlp.disable_pipes(training_excluded_pipes):
        optimizer = nlp.begin_training()
        # Итерация обучения
        print("Начинаем обучение")
        batch_sizes = compounding(
            4.0, 32.0, 1.001
        )  # Генератор бесконечной последовательности входных чисел
```

Рисунок 5.12 – Налаштування для навчання «textcat»

In [13]:
```
from spacy.util import minibatch, compounding
        for i in range(iterations):
            loss = {}
            random.shuffle(training_data)
            batches = minibatch(training_data, size=batch_sizes)
            for batch in batches:
                text, labels = zip(*batch)
                nlp.update(
                    text,
                    labels,
                    drop=0.2,
                    sgd=optimizer,
                    losses=loss
                )
```

Рисунок 5.13 – Додавання навчання на батчах

На рис. 5.14 представлений метод evaluate_model() для класифікації текстів з набору даних (валідаційного) на недонавчені моделі. При цьому результати моделі порівнюються з мітками, пов’язаними з вихідними даними.

In [14]:
```
def evaluate_model(tokenizer, textcat, test_data: list) -> dict:
    reviews, labels = zip(*test_data)
    reviews = (tokenizer(review) for review in reviews)
    # Указываем TP как малое число, чтобы 0 знаменателе
    # не оказался 0
    TP, FP, TN, FN = 1e-8, 0, 0, 0
    for i, review in enumerate(textcat.pipe(reviews)):
        true_label = labels[i]['cats']
        score_pos = review.cats['pos']
        if true_label['pos']:
            if score_pos >= 0.5:
                TP += 1
            else:
                FN += 1
        else:
            if score_pos >= 0.5:
                FP += 1
            else:
                TN += 1
    precision = TP / (TP + FP)
    recall = TP / (TP + FN)
    f_score = 2 * precision * recall / (precision + recall)
    return {"precision": precision, "recall": recall, "f-score": f_score}
```

Рисунок 5.14 – Оцінка моделі класифікатора текстових відгуків щодо готелів

Далі ця оцінка та true_label застосовується для визначення моделей – ложних та істинних. Вони використовуються для подальшого розрахунку значень точності, F-міри та повноти (рис. 5.15).

```
In [15]:            with textcat.model.use_params(optimizer.averages):
                        evaluation_results = evaluate_model(
                            tokenizer=nlp.tokenizer,
                            textcat=textcat,
                            test_data=test_data
                        )
                        print(f"{loss['textcat']:9.6f}\t\
{evaluation_results['precision']:.3f}\t\
{evaluation_results['recall']:.3f}\t\
{evaluation_results['f-score']:.3f}")

                # Сохраняем модель
                with nlp.use_params(optimizer.averages):
                    nlp.to_disk("model_artifacts")
```

Рисунок 5.15 – Визначення істинних та ложних моделей

Далі виконується навчання моделі (рис. 5.16) та отримання результату (рис. 5.17).

```
In [16]: train, test = load_training_data(limit=5000)
         train_model(train, test, iterations=10)
```

Рисунок 5.16 – Фрагмент програми для навчання моделі класифікатора відгуків

```
Начинаем обучение
Loss            Prec.   Rec.    F-score
13.758302       0.809   0.776   0.792
 1.080611       0.827   0.784   0.805
 0.264118       0.833   0.776   0.804
...
 0.005302       0.833   0.776   0.804
```

Рисунок 5.17 – Результати навчання моделі класифікатора відгуків

5.3 Приклади інтерфейсу користувача

У цьому підрозділі наведено основні приклади інтерфейсу для демонстрації роботи системи. Для отримання оцінок, які залишили туристи для певного готеля, потрібно ввести назву та URL готелю (рис. 5.18).

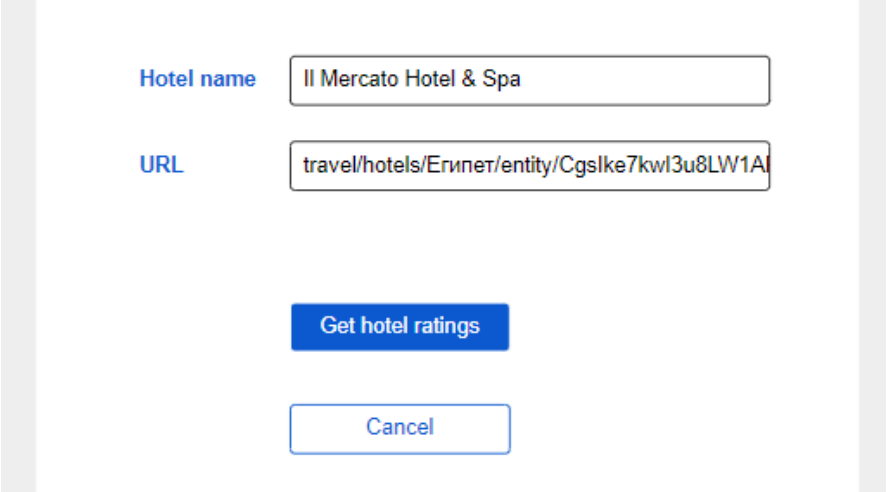

Рисунок 5.18 – Вікно для отримання оцінок щодо готелю

У разі, якщо за зазначеною адресою оцінок для вказаного готелю не знайдено, або відсутній парсер для вказаної веб-сторінки, видається повідомлення про помилку.

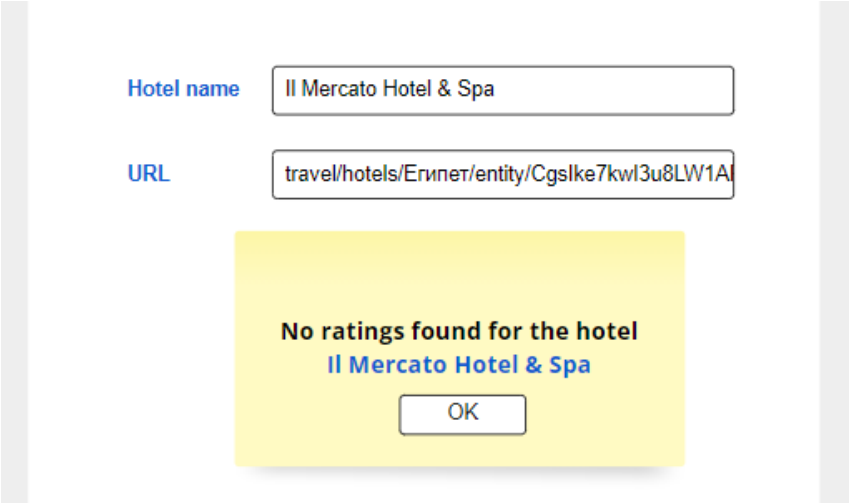

Рисунок 5.19 – Сповіщення про помилку при отриманні оцінок щодо готелю

Для отримання оцінок характеристик готелю користувачеві надається форма (рис. 5.20). Форма містить блоки, кожен блок відповідає певній складовій готелю, всередині блоку містяться характеристики відповідної складової. Якщо користувач відмітив складову (поставив «галочку» зліва від блоку), він може далі обирати характеристики з обраного блоку (відмічати «галочками» характеристики).

Рисунок 5.19 – Вибір складових та характеристик готелю

Це є фрагментов загальної форми. У разі, якщо для певного готелю на його сайті не передбаче оцінювання будь-якої складової, то недосяжний блок, що відповідає цій складовій, відображується жовтим кольором, та користувач не може його обрати.

На рис. 5.20 показано приклад форми для перегляду оцінок по пляжу та береговій зоні готелю. Позначки від 1 до 5 відображають усереднену оцінку по кожній характеристиці. Як можна побачити, зовнішній вигляд території, стан зони та рівень чистоти оцінени на високому рівні, а оцінка зручністі під'їзду/підходу до берегу є досить низькою.

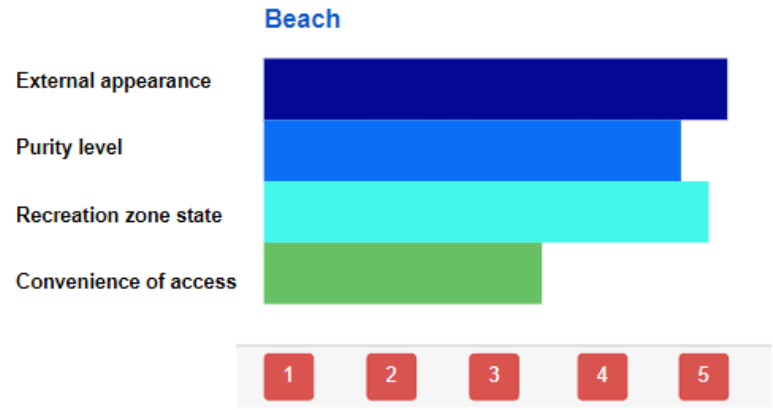

Рисунок 5.20 – Форма для перегляду оцінок берегової зони готелю

На рис. 5.21 наведено оцінювання роботи персоналу, вказано дані тільки для тих характеристик, які були обрані користувачем (відповідно до рис. 5.19).

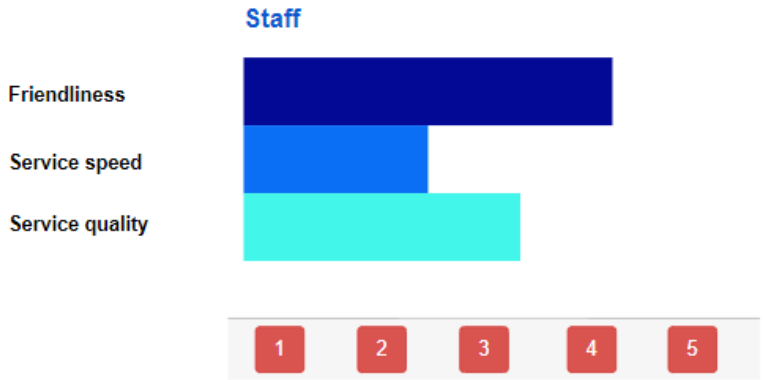

Рисунок 5.21 – Форма для перегляду оцінок якості роботи персоналу

Текстові відгуки щодо готелю користувач може переглянути у списку (рис. 5.22). Поряд з кожним відгуком встановлено оцінку, отриману з використанням автоматичного аналізу тональності відгуків, завдяки тренуванню нейромережі на попередньо розмічених відгуках з відповідними оцінками. Під списком розміщено усереднену оцінку по всім відгукам, що є у системі на певний готель. Текстова оцінка «very good» відповідає числовим оцінкам, які перевищують «4».

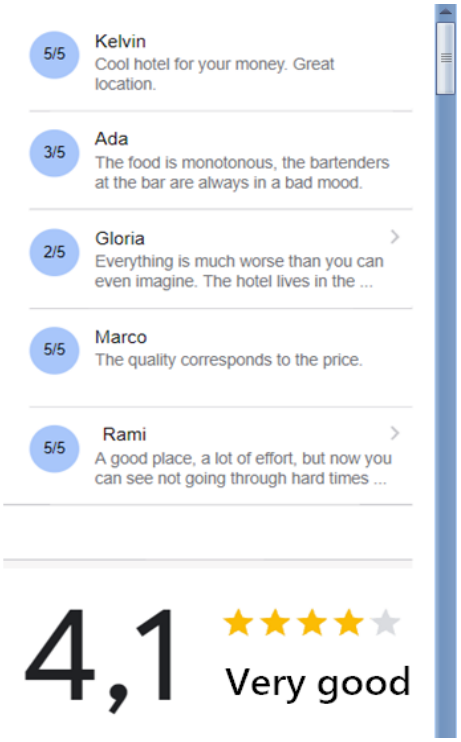

Рисунок 5.22 – Форма для перегляду оцінок тональностей відгуків

5.4 Висновки до розділу

У розділі описано створення програми за допомогою обраної мови програмування Python та open-source бібліотеки spaCy для обробки природної англійської мови з метоб аналізу тональностей текстових відгуків. Описано етапи перетворення текстових відгуків для отримання оцінки тональності. Наведено приклади інтерфейсу користувача для перегляду оцінок обраних характеристик готелю та результатів обробки текстових відгуків.

6 ВИЗНАЧЕННЯ ВЛАСТИВОСТЕЙ ПРОГРАМНОГО ЗАСТОСУНКУ

6.1 Тестування моделі класифікатора текстових відгуків

Тестування точності роботи класифікатора є невід'ємною частиною тестування функціональності застосунку (рис. 6.1), так як при низькій точності було б неможливим вважати систему придатною для використання.

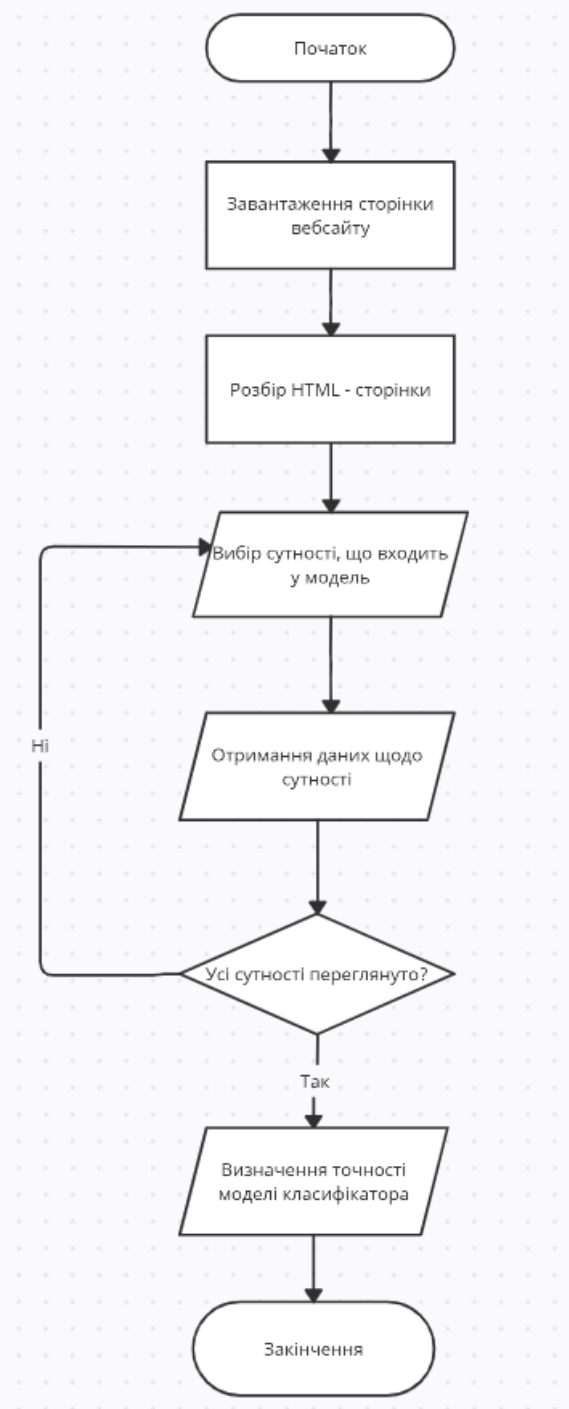

Рисунок 6.1 – Алгоритм тестування програмного застосунку

На рис. 6.2 до методу test_model передається текст нерозміченого відгуку для генерації та відображення користувачеві прогнозу по тональності відгуку.

```
In [17]: def test_model(input_data: str):
             # Загружаем сохраненную модель
             loaded_model = spacy.load("model_artifacts")
             parsed_text = loaded_model(input_data)
             # Определяем возвращаемое предсказание
             if parsed_text.cats["pos"] > parsed_text.cats["neg"]:
                 prediction = "Положительный отзыв"
                 score = parsed_text.cats["pos"]
             else:
                 prediction = "Негативный отзыв"
                 score = parsed_text.cats["neg"]
             print(f"Текст обзора: {input_data}\n\
         Предсказание: {prediction}\n\
         Score: {score:.3f}")
```

Рисунок 6.2 – Визначення прогнозу тональності нерозміченого відгуку

Текст відгуку показано на рис. 6.3. Результати прогнозу – на рис. 6.4.

```
In [19]: test_model(input_data=TEST_REVIEW)

         Текст обзора:
         Beautiful view, very clean, very friendly and helpful staff.
         I've stayed here twice now, and the first time we had
         breakfast with monkeys! Second time, we didn't, which was
         unfortunate, but not at the fault of the hotel.
         Overall, it's been a great experience every time.
```

Рисунок 6.3 – Приклад відгуку для визначення прогнозу

```
Предсказание: Положительный отзыв
Score: 0.912
```

Рисунок 6.4 – Результати прогнозування тональності

Для об’єднання методів завантаження, навчання та тестування моделі використано Python-інструкцію (рис. 6.5):

```
In [20]: if __name__ == "__main__":
             train, test = load_training_data(limit=2500)
             train_model(train, test)
             print("Testing model")
             test_model()
```

Рисунок 6.5 – Об’єднання методів роботи з моделлю класифікатора

6.2 Експериментальне визначення часу на отримання оцінок готелю

Як було зазначено, метою роботи є скорочення часу для отримання рекомендаційних оцінок для вибору готелів на підставі відгуків користувачів та іншої інформації щодо умов та послуг з використанням методів аналізу тональності та машинного навчання. Тому для проведення тестування потрібно обрати декілька готелів, та визначити оцінки їх характеристик без використання засобу, та з ним, а потім порівняти ці значення.

Мною було обрано 5 готелів, та спочатку визначено усі доступні оцінки їх характеристик без власного засобу, а потім отримано ті ж самі оцінки за допомогою нього. У ході експерименту визначався затрачений час у секундах, результати зведені у табл. 6.1.

Таблиця 6.1 – Результати експерименту

№	Без використання засобу при наявності на сайті оцінок		З використанням засобу при наявності на сайті оцінок		З використанням засобу без оцінок на сайті	
	Час, сек.	Відхилення від середнього значення (по модулю), сек.	Час, сек.	Відхилення від середнього значення (по модулю), сек.	Час, сек.	Відхилення від середнього значення (по модулю), сек.
1	28	3	3	0,8	13	2
2	31	6	5	1,2	14	1
3	24	1	3	0,8	19	4
4	23	2	4	0,2	13	2
5	19	6	4	0,2	16	1
Середнє значення	25		3,8		15	

Середнє значення часу на визначення оцінок без використання засобу склало 25 сек., а з використанням засобу – 3,8 сек. Таким чином, час знижено у середньому в 6 разів.

Значення середньоквадратичного відхилення без використання засобу склало приблизно 5 %, а з використанням – приблизно 8 %. Ці результати означають, що розсіювання значень визначеного часу находиться у припустимому діапазоні та експериментальні висновки можна вважати достовірними.

При відсутності оцінок, у тому числі визначення тональності відгуків, на сайті, отримати їх без використання засобу практично неможливо, тому що це означає самостійне вивчення користувачем багатьох текстових записів, тому такий експеримент не проводився. З використанням запису були проведені експерименти для інших 5 сайтів готелів, що містили до тисячі відгуків, та було отримано оцінки у середньому за 15 секунд. При цьому значення середньоквадратичного відхилення значень витраченого часу склало приблизно 3 %.

6.3 Висновки до розділу

У шостому розділі виконано тестування моделі класифікатора, результати показали високу точність прогнозування тональності відгуків.

Поставлено та проведено експеримент з метою визначення часу, що потребує визначення оцінок по готелям. Результати експерименту показали, що з використанням розробленого програмного засобу час скоротився більш ніж у 6 разів, тобто мета проекту досягнута.

ВИСНОВКИ

У кваліфікаційній роботі розроблено програмний засіб для допомоги у виборі готелів з урахуванням аналізу тональності відгуків. Засіб дозволяє прискорити отримання оцінок, що надали туристи готелю, та визначити тональність текстових відгуків для подальшого використання цієї інформації як рекомендаційної. З використанням розробленого програмного засобу час для отримання оцінок скоротився більш ніж у 6 разів.

У кваліфікаційній роботи було опрацьовано наступні етапи.

У першому розділі виконано аналіз існуючих рішень та проведено огляд аналогів.

У другому розділі створено модель оцінювання готелю та алгоритм векторизації текстових відгуків. Обгрунтовано застосування згорткової нейронної мережі для аналізу тональності відгуків.

У третьому розділі визначено специфікації вимог до програмного засобу – функціональних, нефункціональних та системних.

Четвертий розділ присвячено проєктуванню застовунку: визначена архітектура програми, створено основні UML-діаграми, визначено структури даних та виконано проектування програмних класів.

П’ятий розділ містить реалізацію засобу з описом особливостей роботи з мовою Python та його бібліотеками. Також наведено приклади інтерфейсу користувача

Шостий розділ присвячений тестуванню моделі класифікатора текстових відгуків та проведенню експерименту з визначення часових характеристик при отриманні оцінок готелю.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Паклин Н.Б. Бизнес–аналитика: от данных к знаниям: Учебное пособие. [Текст] / Н.Б. Паклин, В.И. Орешков — СПб: Питер, 2013. — с. 428–472.

2. DiscoverText [Електронний ресурс]: – Режим доступу: https://discovertext.com/. – Загл. з екрану: дата звернення 17.10.2021.

3. Clarabridge [Електронний ресурс]: – Режим доступу: https://www.clarabridge.com/customer-experience-dictionary/sentiment-analysis/. – Загл. з екрану: дата звернення 17.10.2021.

4. InMoment [Електронний ресурс]: – Режим доступу: https://asu-analitika.ru/analiz-nastroenij-tipy-instrumenty-i-scenarii-ispolzovanija/. – Загл. з екрану: дата звернення 17.10.2021.

5. Сентимент-аналіз і просування в соціальних медіа [Електронний ресурс]: – Режим доступу: https://compress.ru/article.aspx?id=23115#3/ Загл. з екрану: дата звернення 17.10.2021.

6. Куртукова А.В., Романов А.С., Васильева М.И., Мещеряков Р.В. Анализ тональности текстов с использованием методов машинного обучения // Proceedings of the R. Piotrowski's Readings in Language Engineering and Applied Linguistics. - Saint Petersburg, Russia, November 27, 2017. - Pp. 86-95. [Электронный ресурс]. - URL: http://ceur-ws.org/Vol- 2233/. Загл. з екрану: дата звернення 17.10.2021.

7. Krisilov, V. A. & Komleva, N. O. (2019).“Analysis and Evaluation of Competence of Information Sources in Problems of Intellectual Data Processing”. [Analiz i ocenka kompetentnosti istochnikov informacii v zadachah intellektual'noj obrabotki dannyh]. Problemele Energeticii Regionale, Vol. 1-1(40), pp. 91-104 DOI: 10.5281/zenodo.3239184 (in Russian).

8. Komleva, N. O., Liubchenko, V. V. & Zinovatnaya S. L. “Methodology of Information Monitoring and Diagnostics of Objects Represented by Quantitative

Estimates Based on Cluster Analysis". Applied Aspects of Information Technology. Publ. Nauka i Tekhnika. Odessa: Ukraine. 2020; Vol. 3 No. 1: 376–392. DOI: https://doi.org/10.15276/aait.01.2020.1

9. Комлевая Н.О. Построение системы диагностических признаков с использованием метода дискриминантного анализа в офтальмологических исследованиях. – Радіоелектронні і комп'ютерні системи. – Харків «ХАІ», 2010. – Вип. 6 (47). – С. 250 – 253.

10. Комлевая Н.О., Комлевой А.Н., Тимченко Б.И. Сравнительный анализ двух подходов при решении задачи классификации. – Науко-технічний журнал "Радіоелектронні і комп'ютерні системи". – Харьков, 2014. – № 6(70). – С. 115 – 119.

11. Liubchenko, V., Komleva, N., Zinovatna, S. & Pysarenko, K. "Framework for Systematization of Data Science Methods". Applied Aspects of Information Technology. Publ. Nauka i Tekhnika. Odessa: Ukraine. 2021; Vol.4 No.1: 80–90. DOI: https://doi.org/10.15276/aait.01.2021.7

12. Komleva N.O., Cherneha K.S., Tymchenko B.I., Komlevoy O.M. Intellectual approach application for pulmonary diagnosis. Proceedings of the 2016 IEEE 1st International Conference on Data Stream Mining and Processing, DSMP, 2016, Article № 7583505, pp. 48-52.

13. Элементарное введение в технологию нейронных сетей с примерами программ [Текст] / Р. Тадеусевич, Б. Боровик, Т. Гончаж, Б. Леппер, переклад: Рудинський Д.В. — М.: Горячая линия, 2011. — 408 с.

14. Хайкин С. Нейронные сети: полный курс [Текст] / С.С. Хайкін — М.: Вильямс, 2006. — 1104 с.

15. Васильєв, О. Програмування мовою Python / О. Васильєв. – Київ: Навчальна книга – Богдан, 2019. – 504 с.

16. Naomi Ceder. The Quick Python Book 3rd Edition / Naomi Ceder. – NY: Manning Publications Co., 2018. – 432 p.

17. Kenneth A. Lambert. Fundamentals of Python: first programs / Kenneth A. Lambert. – NY: Cengage Learning, 2018. – 476 p.

18. Devpractice Team. Python. Визуализация данных. Matplotlib. Seaborn. Mayavi / Devpractice Team, М.И. Абдурахманов. 2020. – 412 с.

19. Мэтиз Эрик. Изучаем Python: программирование игр, визуализация данных, веб-приложения. 3-е изд. – СПб.: Питер, 2020. – 512 с.

20. Златопольский, Д. М. Основы программирования на языке Python. – М.: ДМК Пресс, 2017. – 284 с.

21. Комлева Н.О., Жупікова В.Є. Програмний засіб для допомоги у виборі готелів з урахуванням аналізу тональності відгуків // Topical issues of modern science, society and education. Proceedings of the 5th International scientific and practical conference. SPC “Sci-conf.com.ua”. Kharkiv, Ukraine. 2021. Pp. 500-506. URL: https://sci-conf.com.ua/v-mezhdunarodnaya-nauchno-prakticheskaya-konferentsiya-topical-issues-of-modern-science-society-and-education-28-30-noyabrya-2021-goda-harkov-ukraina-arhiv/