НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

«На правах рукопису»
УДК 004.89

До захисту допущено

В. о. завідувача кафедри ММСА

_____ О.Л.Тимощук

«___»_____2018 р.

Магістерська дисертація

на здобуття ступеня магістра за спеціальністю 124 Системний аналіз
на тему: «Система рекомендацій для користувачів на основі аналізу тональності тексту відгуків»

Виконала:
студентка ІІ курсу, групи КА-71 мп
Мулява Галина Ярославівна _____

Керівник: в. о. завідувача кафедри ММСА
кандидат технічних наук, доцент, Тимощук О. Л. _____

Рецензент: завідувач кафедри загально-інженерних дисциплін та теплоенергетики Таврійського національного університету
доктор технічних наук, професор
Медведєв М. Г. _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.

Студент (-ка) _____

Київ

2018

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

Рівень вищої освіти — другий (магістерський)
Спеціальність (спеціалізація) — 124 «Системний аналіз» («Системний аналіз і управління»)

ЗАТВЕРДЖУЮ
В. о. завідувача кафедри ММСА
О. Л. Тимощук
«___» _____ 2018 р.

ЗАВДАННЯ

на магістерську дисертацію студентці Муляві Галині Ярославівні

1. Тема дисертації: «Система рекомендацій для користувачів на основі аналізу тональності тексту відгуків», науковий керівник дисертації Тимощук Оксана Леонідінв, кандидат технічних наук, доцент, затверджені наказом по університету від «07» листопада 2018 р. № 4121-с

2. Термін подання студентом дисертації: _____

3. Об’єкт дослідження: рекомендаційні системи, математичні алгоритми машинного навчання.

4. Предмет дослідження: аналіз тональності тексту з використанням машинного навчання.

5. Перелік завдань, які потрібно розробити:
1) Огляд технічної літератури за темою роботи;
2) Дослідження актуальності обраної теми;
3) Вибір методів для моделювання;
4) Збір вхідних даних;
5) Виконання обчислювальних експериментів та їх аналіз;
6) Проведення аналізу ринкових можливостей запуску стартап-проекту;
7) Підготовка ілюстративного матеріалу;
8) Оформлення пояснювальної записки.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу:
1) Постановка завдання дослідження;
2) Методи аналізу даних;
3) Оцінка моделі;
4) Наукова новизна результатів.

7. Орієнтовний перелік публікацій:

(1) Системний підхід у прийнятті рішення для задач класифікації // Міжнародний науковий журнал "Інтернаука".

8. Дата видачі завдання: _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Отримання завдання на магістерську дисертацію	07.09.2018 – 09.09.2018	
2.	Огляд технічної літератури за темою	10.09.2018 – 30.09.2018	
3.	Дослідження актуальності обраної теми	01.10.2018 – 07.10.2018	
4.	Вибір методів для моделювання і прогнозування	08.10.2018 – 14.10.2018	
5.	Збір вхідних даних	15.10.2018 – 21.10.2018	
6.	Виконання обчислювальних експериментів	22.10.2018 – 28.10.2018	
7.	Аналіз результатів моделювання і прогнозування	29.10.2018 – 04.11.2018	
8.	Проведення аналізу ринкових можливостей запуску стартап-проекту	05.11.2018 – 11.11.2018	
9.	Підготовка ілюстративного матеріалу	12.11.2018 – 18.11.2018	
10.	Оформлення пояснювальної записки	19.11.2018 – 26.11.2018	

Студентка Г. Я. Мулява

Науковий керівник дисертації О. Л. Тимощук

РЕФЕРАТ

Магістерська дисертація: 86 с., 9 рис., 15 табл., 1 додатки, 10 джерел.

Об’єкт дослідження: рекомендаційні системи, математичні алгоритми машинного навчання, аналіз тональності тексту.

Предмет дослідження: аналіз тональності тексту з використанням машинного навчання.

Мета магістреської дисертації: дослідження та реалізація алгоритмів машинного навчання для розробки системи рекомендацій на основі структурованих та неструктурованих даних.

Результати:

- запропоновано власний алгоритм формування рекомендацій на основі структурованих та неструктурованих даних, присутніх в обраній предметній області;
- практичним результатом роботи є розробка архітектури рекомендаційної системи для вибору розважальних бізнесів на основі запропонованого алгоритм аналізу тональності тексту у вигляді прикладної програми;

Напрямок подальших досліджень: впровадження алгоритмів нейронних мереж для оцінки вподобань.

РЕКОМЕНДАЦІЙНІ СИСТЕМИ, АНАЛІЗ ТОНАЛЬНОСТІ ТЕКСТУ, МАШИННЕ НАВЧАННЯ, ОБРОБКА ПРИРОДНОЇ МОВИ.

ABSTRACT

Master's thesis: 86 p., 9 figures, 15 tables, 1 application, 10 sources.

Object of research: recommendation system, mathematical algorithms of machine learning, sentiment analysis of the text.

Subject of research: sentiment analysis of text using machine learning.

The purpose of the master's thesis: research and implementation of machine learning algorithms for developing a system of recommendations based on structured and unstructured data.

Results:

- the own algorithm of the recommendations formation on the basis of structured and unstructured data present in the chosen subject area is offered;

- the practical result of the work is the development of the advisory system architecture for the selection of films on the basis of the proposed algorithm in the form of an application;

Direction of further researches: introduction of algorithms of neural networks for the estimation of preferences.

RECOMMENDATION SYSTEMS, SENTIMENT ANALYSIS, MACHINE LEARNING, NATURAL LANGUAGE PROCESSING.

ЗМІСТ

ПЕРЕЛІК ПРИЙНЯТИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

РС – Рекомендаційна Система

МШ – Машинне навчання

ВСТУП

Метою Рекомендаційної Системи (РС) є створення корисних рекомендацій для користувачів про предмети або продукти, які можуть представляти для них інтерес. Ця нова область досліджень набуває все більшого значення в основному завдяки впливам широкого використання соціальних мереж. Більшість існуючих систем та ресурсів адаптовано до англійських або інших європейських мов. Незважаючи на те, що різноманітні методи створення рекомендацій наявні в джерелах, ті підходи, що використовують обробку природної мови знаходяться на етапі активного дослідження.

Аналіз думок стосується почуттів та емоцій, виражених у тексті. Сьогодні він розвивається швидко через широке використання веб-сайтів та соціальних мереж для вираження дуже великої кількості думок. Ця система перетворює огляди текстів на чисельну оцінку і використовує їх для знаходження можливих інтересів користувача.

Машинне навчання (Machine learning) як підгалузь штучного інтелекту досліджує вивчення та побудову алгоритмів, які можуть навчатися з даних. У сучасному світі технлогій – це набір інструментів для аналізу даних. Цей підхід є альтернативним до аналітики, яка заснована на правилах, коли розробник повинен визначити чіткі правила, що потребує високого знання предметної області. За допомогою машинного навчання можливо будувати системи для аналізу даних при незначній експертизі у предметній області. Системи на основі цього підходу в більшості прикладних напрямків показують кращу ефективність та результативність, аніж системи, які засновані на чітко-визначених правилах.

Людині буквально на кожному кроці доводиться розпізнавати, приймати рішення і вчитися. Нейромережевий підхід виник з прагнення зрозуміти, яким чином мозок вирішує такі складні завдання, і реалізувати ці принципи в автоматичних пристроях. Поки що штучні нейронні мережі (artificial neural networks) є лише гранично спрощеними аналогами природних нейронних мереж. Нервові системи тварин і людини є набагато складнішими тих пристроїв, які можна створити за допомогою сучасних технологій. Однак для успішного вирішення багатьох практичних завдань виявилося цілком достатньо використати лише загальні принципи функціонування нервової системи. Деякі різновиди штучних мереж представляють математичні моделі, що мають лише віддалену подібність з нейрофізіології, що аж ніяк не перешкоджає їх практичного застосування.

Разом з теорією, було запропоновано інструменти аналізу для задач класифікації. Було реалізовано багатошарову перцептронну мережу прямого поширення для задачі класифікації тексту. Визначено функцію втрат, оптимальна кількість прихованих шарів та їх кількість нейроннів, функції активації. Було обрано метод стохастичного градієнтоного спуску і його параметри. Обрано параметри регуляризації моделі, що дає високу точність.

За допомогою запропонованих методів можна ефективно розв'язувати задачі машинного навчання.

РОЗДІЛ 1 РЕКОМЕНДАЦІЙНІ СИСТЕМИ НА ОСНОВІ МАШИННОГО НАВЧАННЯ

1.1 Загальні відомості про рекомендаційні системи.

Системні рекомендаційні системи є інструментами для взаємодії з великими та складними інформаційними просторами. Вони забезпечують індивідуальний перегляд таких простору, визначаючи пріоритетність елементів, які можуть представляти інтерес для користувача. Поле, яке було закріплено в 1995 році, надзвичайно зросло в різноманітності проблем, що розглядаються, і в застосуванні прийомів, і в практичних цілях. Дослідження систем рекомендацій включає в себе широкий спектр методів штучного інтелекту, включаючи машинне навчання, видобування даних, моделювання користувачів, випадкове обґрунтування та задоволення від обмежень, серед іншого. Персоналізовані рекомендації є важливою частиною багатьох онлайнових програм електронної комерції, таких як Amazon.com, Netflix та Pandora. Це багатство практичного досвіду застосування дало натхнення дослідникам, щоб розширити сферу застосування рекомендаційних систем у нових та складних областях. Мета цього спеціального питання полягає в оцінці поточного ландшафту дослідження систем рекомендацій та визначення напрямків, які зараз здійснюється в галузі. Цей розділ містить огляд поточного стану галузі та вводить різні статті в спеціальному випуску. Систему рекомендацій опишемо наступним чином. У типовій системі рекомендацій люди надають рекомендації в якості вхідних даних, які система потім агрегатує та спрямовує до відповідних одержувачів. У деяких випадках первинна трансформація знаходиться в агрегації; в інших

цінність системи полягає у його здатності робити хороші матчі між рекомендаційниками та тими, хто шукає рекомендацій.

Слід зауважити, що це визначення робить акцент на системах, що рекомендують, як на підтримку співпраці між користувачами. Пізніше дослідники розширили визначення, щоб включити системи, які пропонують цікаві предмети, незалежно від того, як виготовляються ці рекомендації: "будь-яка система, яка виробляє індивідуальні рекомендації як висновок, або може керувати користувачем персоналізованим способом для цікавих або корисних об'єктів у це великий простір можливих варіантів. Система рекомендованих призначена для того, щоб допомогти користувачеві вибрати один із дискретних варіантів. Як правило, предмети вже відомі заздалегідь, а не створені спеціально. Особливості персоналізації систем, що рекомендують, сильніше виділяють цю лінію досліджень з того, що зазвичай розуміється як дослідження в пошукових системах та інших додатках для пошуку інформації. У пошуковій системі або іншій інформаційно-пошуковій системі ми очікуємо набір результати, відповідні конкретному запиту, повинні бути однаковими, незалежно від того, хто його опублікував. Багато систем, що рекомендують, досягають персоналізації, підтримуючи профілі діяльності користувача (довгострокові або короткочасні) або заявлені переваги. Інші досягають персоналізованого результату завдяки розмовній. Системне дослідження "Recommender System Typology Recommender System" характеризується загальною проблемою, а не загальною технологією або підходом. Експертиза останніх чотирьох конференцій ACM Recommender System показує, що до проблеми систем рекомендацій застосовується широкий спектр дослідницьких підходів, починаючи від статистичних методів до онтологічних міркувань, і вирішуючи широкий спектр проблем - від вибору споживацьких товарів до знаходження друзі та коханці. Один із уроків, який вивчався протягом останніх років дослідження системних систем, полягає в тому, що область застосування сильно впливає на типи методів, які можуть бути успішно застосовані. Характеристика домену, як збереження корисності

користувача, має великий вплив: наприклад, смак користувачів у музиці може змінюватися повільно, але його інтерес до новин про знаменитості може коливатися набагато більше. Таким чином, надійність уподобань, зібраних у минулому, може бути різною. Подібним чином деякі предмети, такі як книги, доступні для рекомендації та споживання протягом тривалого періоду часу - часто років. З іншого боку, в технологічній сфері, такі як мобільні телефони або камери, старі продукти швидко старіють і не можуть бути корисними. Це також стосується тих областей, де актуальність актуальна, як новини та культурні події. Тому не дивно, що існує безліч напрямів досліджень у системах, що рекомендують, оскільки дослідники займаються різними областями рекомендацій. Щоб уніфікувати ці різноспрямовані підходи, корисно розглянути аспекти рекомендацій AI, зокрема, основи знань, що лежать в основі системи рекомендацій. Джерела знань Кожна система IC використовує один або більше джерел знань для виконання своєї роботи. Наприклад, контрольована машинна система навчання матиме мітку набору даних як основного джерела знань, але алгоритм та його параметри можна вважати іншим неявним видом знань, який притягується до завдання класифікації. Алгоритми рекомендацій також можуть бути класифіковані відповідно до джерел знань, які вони використовують. Існує три основних типи знань: соціальні знання про базу користувачів в цілому, індивидуальні знання про конкретного користувача, для якого потрібні рекомендації (і, можливо, знання про конкретні вимоги, які ці рекомендації повинні відповідати), і, нарешті, знання контенту про предмети що рекомендується, починаючи від простих списків функцій до більш складних онтологічних знань та засобів, що дають змогу зрозуміти, як продукт може відповідати потребам користувача [1].

Різні методи рекомендацій спираються на різні частини цього спектру джерел знань. Умови конкурсу Netflix Prize дали доступ лише до думок у формі рейтингів, але ніяких вимог або демографічної інформації про користувачівю Знання хорошого домену надзвичайно важко зібрати в цій області через

складність уявлення та обгрунтування наративного змісту, директорійного стилю тощо. Таким чином, проблема зупинилася на техніці математичного наближення, що працює виключно з рейтингів як соціальних, так і індивідуальних. На відміну від цього, проблема рекомендацій щодо варіантів інвестування, може мати користь від детальних знань про доходи та фінансовий стан клієнта, інших елементах їх портфеля та їх ставлення до ризику. Думки та вибір інших користувачів можуть бути корисними, але недостатніми для створення якісних рекомендацій у цій галузі. Питання дослідження в рекомендаційній системі

1.2 Типові архітектури рекомендаційних систем

Найбільш відомим методом у рекомендації є спільна рекомендація. Основне розуміння цієї методики - це своєрідна спадкоємність у сфері смаку - якщо користувачі мають ту ж саму утиліту для елементів від 1 до k, то шансів, що вони матимуть одна і та сама утиліта для пункту k + 1. Зазвичай ці утиліти базуються на рейтингах, наданих користувачами для елементів, з якими вони вже знайомі. Ключовою перевагою спільної рекомендації є його простота. Проблема обчислювальної утиліти перетворюється на проблему екстраполяції пропущених значень у матриці оцінок, рідкісну матрицю, де кожен користувач є рядком, кожен елемент - стовпчик, а значення - відомими рейтингами. Це розуміння можна застосувати кількома способами. Спочатку технології найближчих сусідів застосовувались для пошуку сусідів однодумців-однолітків. Проте матрична факторизація та інші методи зменшення розмірності тепер визнані вищими за

точність. Деякі проблеми з рекомендаціями щодо спільної роботи є визнаними. Нові елементи не можна рекомендувати, не покладаючись на якийсь додатковий джерело знань. Екстраполяція залежить від наявності деяких значень, з яких потрібно проектувати. Дійсно, пункти з обмеженим рейтингом в цілому становлять проблему, оскільки в системі відсутня інформація про те, на якій основі можна прогнозувати. Точно так само користувачі, які поставили кілька оцінок, отримають шумніші рекомендації, ніж ті, що мають більш суттєві історії. Проблеми нових користувачів та нові рейтинги колективно називають проблемою холодного запуску в спільній рекомендації. Розподіл оцінок та уподобань користувачів у багатьох галузях смаку для споживачів досить концентрований: невелика кількість "блокбастерів" приділяють велику увагу, і їх багато, дуже рідко оцінюються. Зловмисні користувачі можуть створювати велику кількість псевдонімів профілів і використовувати їх, щоб так чи інакше змішувати рекомендації системи.

Існує багато алгоритмічних досліджень, зосереджених на проблемах спільної рекомендації: більш точні та ефективні оцінки матриці оцінок, краща робота нових користувачів та нових елементів, а також розширення базової ідеї спільної рекомендації до нових типів даних в тому числі багатовимірні рейтинги і теги, створені користувачами, серед інших. Рекомендація, орієнтована на зміст До появи рекомендацій щодо спільної роботи в 1990-х роках, раніше дослідження в області персоналізованого доступу до інформації зосереджувались на поєднанні знань про елементи з інформацією про переваги користувачів, з тим щоб знайти відповідні елементи. Цей підхід, через те, що він залежить від джерела знань вмісту, зокрема елементів, стало відомо як рекомендація, основана на вмісті. Рекомендації, основані на вмісті, тісно пов'язані з наглядом за машиною. Ми можемо розглянути проблему як вивчення набору користувальницьких класифікаторів, де класи корисні для користувача X і не корисні для користувача X. Одним із ключових проблем рекомендації на основі вмісту є якість об'єкта. Об'єкти, що рекомендуються, повинні бути описані

таким чином, щоб мати місце значне вивчення користувацьких уподобань. В ідеалі кожен об'єкт буде описаний на тому ж рівні деталізації, і набір функцій міститиме дескриптори, які співвідносяться з дискримінаціями, зробленими користувачами. На жаль, це часто не так. Описи можуть бути частковими, або деякі частини простору об'єкта можуть бути описані більш детально, ніж інші. Співпраця між набором функцій та функцією корисності користувача також має бути хорошою. Одна з переваг популярного потокового музичного сервісу Pandora полягає в тому, що набір функцій, який він використовує для музичних виділень, вручну вибирається музичними підручниками. Автоматична обробка музики ще не є достатньою, щоб надійно витягувати такі функції. На додаток до розробки та застосування нових алгоритмів навчання для завдання рекомендації, дослідження в рекомендаціях, що містять зміст, також розглядає проблему вилучення функцій у різних областях. Подальший підтип рекомендацій на основі вмісту - це рекомендація, основана на знаннях, у якій поширюється покладання на елементи об'єктів на інші види знань про продукти та їхні потенційні утиліти для користувачів [2]. Прикладом такої системи є згаданий вище рекомендатор з інвестицій, який повинен знати про профілі ризиків та податкові наслідки різних інвестицій і як вони взаємодіють з фінансовим становищем інвестора. Як і в інших системах, що знаходяться на знаннях, основними проблемами є придбання, підтримка та перевірка знань. Крім того, оскільки рекомендації, що базуються на знаннях, можуть використовувати детальні вимоги від користувача, вивчення інтерфейсу користувача є надзвичайно важливим у розробці рекомендацій, що базуються на знаннях, які не створюють надмірного навантаження для користувачів.

1.3 Класифікація відомих методів аналізу тональності.

При проведенні досліджень чи прийнятті рішень кожного дня ми часто шукаємо думки інших людей. Ми ведемо консультації з політичними дискусійними форумами під час політичного голосування на виборах, читаємо споживацькі звіти при покупці техніки, запрошуємо друзів запропонувати вечерню ресторан. Тепер Інтернет дозволив з'ясувати думки мільйонів людей про все, починаючи з останніх прикладів політичних світоглядів. Останнє дослідження Pew в Інтернеті та громадянських зв'язках означає, що лише у п'яти користувачів Інтернету (19%) опублікували матеріали про політичні чи соціальні проблеми або використовували сайт соціальних мереж для певної форми громадянської чи політичної участі. Інше дослідження показує, що третина (33%) користувачів Інтернету читають блоги, причому 11% це робить щодня. Інтернет все частіше стає як форумом для обговорення, так і джерелом інформації для все більшої кількості людей. Реалізація наявного в думках тексту створила нову область в текстовому аналізі, розширюючи предмет вивчення з традиційно факто-інформаційно-орієнтованого перегляду тексту, щоб дозволити почуття -захисні програми. Впродовж останнього десятиліття витягання настроїв з тексту привертає велику увагу як у промисловості, так і в наукових сферах. Підприємства все більше усвідомлюють важливість інтернет-користувачів щодо їх продукту та послуг. Цей документ є огляд сфери аналізу настроїв, який стосується суб'єктивних текстів. Наше перше завдання - визначити почуття і визначити його відношення до тексту. Одним із завдань аналізу настроїв є визначення об'єктів дослідження - думки та суб'єктивність. Суб'єктивність часто вживається в розмові, вона дуже контекстно-чутлива, і її вираз часто властивий кожній людині.

Почуття також має декілька унікальних властивостей, які відрізняють його від інших кваліфікацій, які ми хочемо відстежити в тексті. Часто ми хочемо класифікувати текст по темі, який може включати в себе обробку всіх таксономічних тем. Класифікація почуттів, з іншого боку, зазвичай стосується двох класів (позитивних чи негативних), аранжування полярності (наприклад, рейтинг зірок для кінофільмів) або навіть діапазон міцності. Ці класи охоплюють багато тем та користувачів, а також документи. Хоча вирішення лише кількох класів може здатися легшим завданням, ніж стандартний аналіз тексту, він не може бути далі від істини. Як область дослідження, вона тісно пов'язана з (чи може розглядатися як частина) ком'ютаційною лінгвістикою, обробкою природної мови та виведення тексту. Виходячи з вивчення афективного стану (психології) та судження (теорії оцінки), це поле прагне відповісти на питання, доведені до вивчення в інших областях дискурсу, використовуючи нові інструменти, що забезпечуються інтелектуальною інформацією та обчислювальною лінгвістикою. Аналіз сенсу має безліч назв. Часто це називається аналізом ассуб'єктивності, виявлення думок та вилученням оцінки, з деякими зв'язками до ефективного обчислення (комп'ютерне визнання та вираження емоцій. Поле, як правило, вивчає суб'єктивні елементи, визначені як лінгвістичні вирази приватних держав у контексті. Вони, як правило, одиничні слова, фрази або речення. Іноді ці документи були засвоєні як одиниця , але зазвичай вони погоджуються, що таке почуття існує в менших лінгвістичних одиницях. Полярність почуттів - особлива особливість тексту. Вона зазвичай дихотомізована двом - позитивним і негативним - але полярність також можна розглядати як діапазон. Примітка, що містить кілька твердих висловлювань, матиме змішану полярність, яка буде відрізнятися від відсутності полярності (бути об'єктивною). Крім того, необхідно провести відмінність між полярністю настроїв та її стійкістю. Можна сильно відчувати про те, що продукт є задовільним, не особливо хорошим або поганим; або слабко про те, що продукт дуже хороший (тому що, можливо, хтось володіє нею занадто короткий час для

формування сильної думки). Іншою важливою частиною настроїв є його ціль - об'єкт, поняття, людина, що завгодно. Більшість робіт було зроблено на огляді продуктів і фільмів, де це легко визначити тему тексту. Для вирішення цілей, описаних вище, використовується широкий спектр інструментів та методів. Зазначино деякі з найбільш поширених та цікавих. Обговорюватиметься позначення машинного навчання та тестування частиною мови, оскільки це дуже потужні інструменти, які найчастіше використовуються в аналізі почуттів. Потім будуть розглянуті конкретні технології та підходи до вирішення кожного з завдань.

Багато завдань аналізу настроїв можна розглядати як класифікацію. Машинне навчання пропонує безліч алгоритмів, призначених для цього саме так, але це завдання класифікації тексту відповідно до його настроїв дає багато унікальних викликів. Традиційні системи пошуку інформації вже давно підкреслюють важливість термінальної частоти. Знаменита міра TF-IDF (частота слова- обернена частота документа) широко використовується у моделювальних документах. Навчання в тому, що терміни, які часто відображаються у документі, але рідко в цілому, є більш інформативними стосовно того, що таке документ у порівнянні з умовами, що згадуються лише один раз. У галузі аналізу настроїв ми знаходимо, що, незважаючи на увагу на найчастіші терміни, більш вигідно шукати найбільш унікальні. Представлення документа, що підкреслює термін присутності, містять 1, якщо термін відображається в документі принаймні один раз, 0 в іншому випадку. Використання n-грам також спостерігається в даній галузі. Позиції в темі також важливі у поданні документів для аналізу аналізі. Позиція термінів визначає, а іноді й повертає, полярність фрази. Таким чином, інформація про позицію іноді кодується в вектор функції вибирає n грамів (n = 1,2,3,4), виходячи з точності, розрахованої з використанням анотованих документів. Потім грами - словотвірна пара, частина мови.

Як згадувалося раніше, було визначено, що прикметники є хорошими показниками почуттів у і в останнє десятиліття вони широко використовувались в аналізі сміху. Це справедливо для інших полів при аналізі тексту, оскільки теги частин мови можуть розглядатися як груба форма розсіювання.

Синтаксична інформація також використовується в наборах функцій, хоча все ще існує дискусія щодо заслуг цієї інформації в класифікації відчуттів. Ця інформація може включати важливі текстові функції, такі як заперечення, підсилювачі та зменшувачі, використовується підстроковий алгоритм підсилення з функціями на основі дерева залежностей для класифікації полярності та показав, що він перевищує базову лінію баг-слово.

Негативні відомості були давно відомі як невід'ємна частина аналізу настроїв. Звичайне подання тексту з тексту розриває всі слова, і розглядає такі погляди, як "Мені подобається ця книга" та "Мені не подобається ця книга", дуже схожі, оскільки лише одне слово відрізняє один від одного. Але, говорячи про сентименти, заперечення перекриває полярність цілісної фрази. Погашення часто розглядаються в пост-обробці результатів, тоді як оригінальне подання тексту ігнорує його. Можливо, що спільне розташування може бути надто сировиною. Було б неправильно заперечувати почуття в такому реченні, як "Недарма кожен любить це". Для обробки таких випадків, використовують специфічні шаблони міток мови для ідентифікації негативів, пов'язаних із полярністю почуття фраз.

Одним з найпростіших завдань в аналізі почуттів є визначення семантичноїорієнтації (полярності та об'єктивності) слова. Були використані різноманітні методики, які можна грубо класифікувати у наступному: • використання alexicon, побудованого вручну або автоматично; • використовуючи аналогічні технічні засоби, такі як пошук збігів слова зі словом відомої полярності; • використання тренінгу документів, позначених або не позначених, як джерело знань про полярність умов всередині колекції.

Визначення семантичної орієнтації речень і фраз важливик крок в наліі чутливості. Оскільки визначено семантичну орієнтацію окремих слів, є бажано поширити його на фразу або речення, слово з'являється в. Одним з найбільш простих способів здійснення цього є прийняття середньогополярності слова в реченні. Візьмемо припущення. Якщо позитивний / негативний висновок переважає, речення думок розглядається як позитивний / негативний. У тому випадку, коли число позитивних та негативних слів є однаковим, вони приймають орієнтацію найближчого вислову думки. У даному випадку обрано Наївний Байесівський класифікатор, використовуючи речення та документи, позначені як виражені або фактичні як зразки двох категорій. Функції включають слова, біграми і триграми, а також частини мови в кожному реченні. Вони також використовують наявність слова з відомими полярності в реченні як ознака того, що вирок є суб'єктивним. І вони враховують ефект негативних слів, таких як "ні", "не", що з'являються у вікні з 5 слів навколо слів у питанні. Незважаючи на те, що це спрощене, цей евристичний інструмент виявився ефективним для більшості випадків. Ще більш складне поєднання етикетки сприйняття є можливим завдяки перевагам синтаксичних зв'язків між словами. Використовують маркування релаксації, що не підкоряється класифікації, що розширює етикетку, приписану слову, до того, що воно з'являється. Цей підхід, крім усього іншого, включає модифікатори відхилення.

У новинному застосуванні методів машинного перекладу описано система перекладу з японського на англійську мову, дослідники змогли побудувати дерева пропозицій, а потім застосувати співставлення шаблонів, щоб виявити орієнтацію пропозицій на строк. Через складність методу вони змогли включити в цей процес багато мовних сигналів, включаючи негативи, застосування яких було обговорено раніше. Визначення семантичної орієнтації документів, хоча більша частина роботи проводиться при визначенні семантичного орієнтація словесних та фраз, деякі завдання, такі як узагальнення та пошук тексту, можуть вимагатимемантичне маркування всього документа. Це може не мати сенсу для

датування довгих документів, таких як статті чи книги, які були ключовою формою внутрішнього пошуку інформації. Але в епоху соціальних мереж та інтернет-торгівлі ми бачимо значно більшу кількість і різноманітність коротких документів, часто містить лише кілька речень. Це можуть бути відгуки на продукти, електронні листи, публікації в блозі тощо. Багато в чому подібні підходи для визначення семантичної орієнтації слів, документи також варіюються від простих статистичних до тих, що використовують розробку структур знань для керівництва процесом. Одне з найбільш популярних і простих методів є лінійною комбінацією всіх полярностей, наприклад, використовують усереднення для визначення полярності документів.

Видобуток показників об'єкта є важливим кроком. Тепер ми переходимо до іншої важливої частини почуття - її ціль. У більш коротких, більш цілеспрямованих документах часто можна припустити, що автор розмовляє лише з темою документа. Наприклад, огляди продуктів, як правило, містять інформацію про цей продукт, а огляди фільмів говорять про фільми, про які йде мова. Проте часто це недостатньо для ознайомлення з загальною темою написання. Сприятливий продукт, звичайно, хоче знати не тільки те, що люди думають про цей продукт в цілому, але які особливості вони особливо подобаються або не подобаються. Таким чином, задача вилучення ознак (де функція може бути будь-якою ціллю вираженого висловлювання) набирає популярність у сфері аналізу настрою. Загальний підхід полягає у використанні тегів частин мови (part of speech) для побудови тематичних таблиць про те, як почуття відчуття застосовується до об'єктів.

Розглянемо комерційне використання. Незважаючи на те, що поле аналізу настроїв є відносно молодим, існують численні підприємства, які використовують технології, розроблені в цій галузі, для клієнтів, зацікавлених у відстеженні бренда та сприйнятті ринку. Наприклад, як частина своїх анти-підробок та онлайн-служб збуту брендів, послуги з аналізу аналізу, такі як

моніторинг, вимірювання та аналіз зворотного зв'язку споживачів, з тим щоб їх клієнти були краще інформовані для розуміння потреб ринку, цільових сегментів клієнтів, і позиція проти конкурентів . Зокрема, це види діяльності, які можуть бути задіяні:

• відстеження колективної думки користувачів та оцінки продуктів та послуг

• аналіз споживчих тенденцій, конкурентів та ринкових змін

• вимірювання відповідей на події та інциденти, пов'язані з компанією

• моніторинг критичних проблем для запобігання негативні ефекти вірусу

• оцінка зворотного зв'язку на багатьох мовах.

Описуючи, оцінюючи та інтерпретуючи дані, що знаходяться на веб-сайтах, надають статистику та рекомендації, а також прогнозують тенденції щодо продукту та бренду. Виявлення лідерів думки допомагає компаніям виявити свої сили. Ці послуги також можуть бути корисними для державної розвідки. Моніторинг комунікацій для спайків у негативному настрої може бути корисним для агентств. Але, крім компаній та державних установ, генеральні веб-користувачі можуть скористатися інструментами, що знають про себе. Є кілька орієнтованих на думку орієнтованих пошукових систем, доступних в Інтернеті. За попередньою міткою веб-сторінок та блогів ці служби забезпечують кластеризоване представлення результатів, що полегшує розуміння користувачами результатів. Теми не потребують обмеження в огляді продуктів. Це можуть бути політичні питання або думки про кандидатів для роботи в офісі. Нарешті, відкриття думки може бути корисним підкомпонентом іншої служби. Рекомендовані системи можуть значно виграти від вилучення користувацьких рейтингів від тексту. Інформаційно-пошукові системи також можуть застосовувати заходи суб'єктивності при роботі з певним типом інформації, наприклад, коли об'єктивність бажана в науково-популярній літературі7.

Відкриті напрямки досліджень відносно нової області, у своїй короткій історії аналіз сентиментальності використовував природні Обробка мовної обробки даних, виведення даних та інструменти для пошуку тексту для вирішення проблеми викладання думок з тексту.

Висновки до розділу

У даному розділі розглянуто опис загальних теоретичних аспектів рекомендаційних систем, їх роль та місце в науці як міждисциплінарної області. Були описані визначення, характеристики та властивості даного типу задач, а також розглянуті основні підходи до їх розв’язання.

Всупереч розмитому визначенню почуттів та складності його прояву в тексті аналіз тональності тексту знайшов своє місце у обчислювальній лінгвістиці та науці про дані. Було описано найпоширеніші методи даної задачі.

Здатність машинного навчання у обробці природної мови навчатися на даних за допомогою вчителя або без такого забезпечує їх важливу властивість, що має важливе теоретичне і практичне застосування, що зробило їх неоціненним інструментом у різноманітних галузях застосування.

РОЗДІЛ 2 МОДЕЛЬ КЛАСИФІКАЦІЇ ТОНАЛЬНОСТІ ТА АЛГОРИТМ ПІДБОРУ РЕКОМЕНДАЦІЙ

2.1 Загальна постановка задачі рекомендацій

Розглянемо формальну постановку задачі. X – множина об’єктів. Y – множина допустимих відповідей. Існує цільова функція $y^*: X \to Y$, значення якої $y_i = y^*(x_i)$ відомі тільки на скінченній підмножинні об’єктів: $X^\ell = (x_i, y_i)_{i=1}^\ell$ (називається навчальної вибіркою) [3].

Задача навчання полягає в тому, щоб за вибіркою X^ℓ відновити залежність y^*. Тобто побудувати вирішальну функцію $a: X \to Y$, яка наближала б цільову функцію, причому не тільки на об'єктах навчальної вибірки, а й на всій множині X (називається алгоритмом). Саме тому задача цієї роботи відноситься до категорії навчання з учителем (кероване навчання, англ. supervised learning): Комп'ютерові представляють приклади входів та їхніх бажаних виходів, задані «вчителем», і метою є навчання загального правила, яке відображає входи на виходи.

Ознака f_j об'єкта X - це результат вимірювання деякої характеристики об'єкта $j = 1, \ldots, n$. Відомо, що часто найважливішим при вирішенні завдання є вміння правильно відібрати і навіть створити ознаки. В англомовній літературі це називається Feature Engineering, що є досить творчим процесом і покладаэться більше на інтуїцію і експертні знання. Вдалий вибір ознак є запорукою мінімальної помилки результуючої моделі. Формально ознака називається

відображення $f : X \to D f$, де $D f$ - множина допустимих значень ознаки. Зокрема, будь-який алгоритм $a: X \to Y$, також можна розглядати як ознаку.

Маємо матрицю об'єктів-ознак:

$$F = \|f_j(x_i)\|_{\ell \times n} = \begin{pmatrix} f_1(x_1) & \dots & f_n(x_1) \\ \dots & \dots & \dots \\ f_1(x_\ell) & \dots & f_n(x_\ell) \end{pmatrix}, \qquad (2.1)$$

де f_j - ознака об'єкта x_i , $i = 1, \dots l$

В залежності від природи безлічі Df ознаки діляться на кілька типів. Якщо $Df = \{0, 1\}$, то f - бінарна ознака;

Якщо Df -скінченна множина, то f - номінальний ознака;

Якщо Df - скінченна впорядкована множина, то f - порядкова ознака;

Якщо $Df = R$, то f - кількісний ознака.

Залежно від природи множини допустимих відповідей Y завдачі навчання по прецедентах діляться на наступні типи. Якщо $Y = \{1,\dots, M\}$, то це завдання класифікації (classification) на M неперетинаючих класів. У цьому випадку вся множина об'єктів X розбивається на класи $Ky = \{x \in X: y^*(x) = y\}$, і алгоритм $a(x)$ повинен давати відповідь на питання «якому класу належить x?». У деяких додатках класи називають образами і говорить про задачі розпізнавання образів (pattern recognition).

Сформуємо задачу класифікації на 2 класи: $Y = \{-1, +1\}$

Функціонал якості $\mathscr{L}(a, x)$ - функція втрат, величина помилки алгоритму a на об’єкті x.

Наприклад (для задач класифікації) $\mathscr{L}(a, x) = [a(x) \neq y(x)]$

Емпіричний ризик – функціонал якості алгоритму a на X^ℓ:

$$Q(a, X^\ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} \mathscr{L}(a, x_i) \quad , \qquad (2.2)$$

де $\mathscr{L}(a, x)$ - функція втрат алгоритму a на об'єкті x.

Функція втрат, приймаюча лише значення 0 і 1, називається бінарною. В цьому $\mathscr{L}(a, x) = 1$ означає, що алгоритм припускається помилки на об'єкті x, а функціонал Q називається частотою помилок алгоритму на вибірці X^ℓ.

Найбільш часто використовуються наступні функції втрат, при $Y \subseteq R$:

$\mathscr{L}(a, x) = [a(x) \neq y(x)]$ - індикатор помилки, зазвичай застосовується в задачах класіфікації;

$\mathscr{L}(a, x) = | a(x) - y^*(x) |$ - відхилення від правильної відповіді; Q функціонал називається середньою помилкою алгоритму на вибірці X^ℓ;

$\mathscr{L}(a, x) = (a(x) - y^*(x))^2$ - квадратична функція втрат; Q функціонал називається середньою квадратичної помилкою алгоритму на вибірці X^ℓ; зазвичай використовуэться в задачах регресії.

Класичний метод навчання, так звана мінімізація емпіричного ризику (empirical risk minimization , ERM), полягає в тому, щоб знайти в заданому моделі A алгоритму a, що доставляє мінімальне значення функціоналу якості Q на заданому навчальній вибірці X^ℓ:

$$\mu(X^\ell) = \arg\min_{a \in A} Q(a, X^\ell) \qquad (2.3)$$

де $X^\ell = (x_i, y_i)_{i=1}^{\ell}$ - навчальна вибірка, a – алгоритм навчання.

У завданнях навчання по прецедентах елементи множини X - це не реальні об'єкти, а лише доступні дані про них. Дані можуть бути неточними, оскільки вимірювання значень ознак f_j об'єкта x і цільової залежності $y^*(x)$ зазвичай виконуються з похибками. Дані можуть бути неповними, оскільки вимыряють не всі мислимі ознаки, а лише фізично доступні для вимірювання. В результаті одному й тому ж самому опису x може відповідати різні об'єкти і різні відповіді. У такому випадку $y^*(x)$, строго кажучи, не є функцією. Усунути цю некоректність дозволяє імовірнісна постановка задачі.

Замість існування невідомої цільової залежності $y^*(x)$ припустимо існування невідомого ймовірного розподілу на множини $X \times Y$ з щільністю $p(x,y)$, з якого випадково і незалежно вибираються обмежена кількість спостережень . Такі вибірки називається прості або випадковими однаково розподіленими (independent identically distributed, i.i.d.).

Розглянемо функція втрат для класифікації. У машинному навчанні та математичної оптимізації, функції втрат для класифікації є обчислювально здійсненні функції втрат, що представляють ціну, сплачену за неточність прогнозів в задачах класифікації. З огляду на X як векторний простір всіх можливих входів, і $Y = \{-1, +1\}$ як векторний простір всіх можливих результатів, ми хочемо знайти функцію $F: X \mapsto R$, який найкраще відображає x в y. Проте, через неповної інформації, шуму в вимірі, або імовірнісних компонентів в основний процес, можна за те ж саме x, щоб генерувати інший y.

Логістична функція втрати визначаються як сигмоъда з t - це параметр функції, що визначає її крутизну. Коли t прямує до нескінченності, функція вироджується в порогову. При $t = 0$ сигмоїда вироджується в постійну функцію із значенням 0,5. Область значень даної функції знаходиться в інтервалі (0,1). Важливою перевагою цієї функції є простота її похідної:

$$\frac{d\sigma(x)}{dx} = tf(x)(1 - f(x)) \quad , \qquad (2.4)$$

де $f(x)$ –ознака об’єкта , $\sigma(x)$ - функція втрат, t – час.

Ця функція показує аналогічну швидкість збіжності як і функція втрат петлі (hinge loss function) , і, оскільки вона є неперервною, градієнтні методи спуску може бути використані. Однак функція логістичної втрати не привласнює нульовий штрафу будь-яких точок. Замість того, функції, які правильно класифікувати точки з високим ступенем достовірності (тобто з високими значеннями | *f (x)* | штрафуються менше. Ця структура призводить логістичну функцію втрат, щоб бути чутливим до викидів в даних. Мінімізатор $I\ [\ f\ ]$ для функції логістичної втрати[3].

Те, що похідна цієї функції може бути виражена через її значення, полегшує використання цієї функції при навчанні мережі за алгоритмом зворотного поширення. Особливістю нейронів з такою передавальною характеристикою є те, що вони посилюють сильні сигнали істотно менше, ніж слабкі, оскільки області сильних сигналів відповідають пологим ділянках характеристики. Це дозволяє запобігти насиченню від великих сигналів

Ця функція не визначена, коли p (1 | x) = 1 або p (1 | x) = 0 (прямуючи до ∞ і -∞ відповідно), але прогнозує плавну криву, яка росте, коли p (1 | x) збільшується і дорівнює 0, коли p (1 | x) = 0,5

Етапи розв'язку задач машинного навчання:

1) Розуміння задачі та даних
2) Первинна оброка даних та обробка ознак
3) Побудова моделі
4) Приведення навчання до оптимізації

5) Розв’язок проблем оптимізації і перенавчання
6) Оцінка якості
7) Впровадження та експлуатація

2.2 Алгоритм навчання моделі класифікації тексту

Нейронна мережа = суперпозиція нейронів з нелінійною функцією активації

$$a(x, w) = \sigma\big(\langle w, x\rangle\big) = \sigma\left(\sum_{j=1}^{n} w_j f_j(x) - w_0\right), \qquad (2.5)$$

де $f_j(x)$ - ознака об'єкта x, $w_j \in \mathbb{R}$ - ваги ознак, $\sigma(x)$- функція активації.

Вибір параметрів моделі :

Отримуємо задачу $w \equiv \left(w_{jh}, w_{hm}\right) \in \mathbb{R}^{H(n+M+1)+M}$ оптимізації:

$$Q(w) := \sum_{i=1}^{\ell} \mathscr{L}(w, x_i, y_i) \to \min_{w}, \qquad (2.6)$$

1. Ініціалізація мережі: вагові коефіцієнти і зсуви мережі приймають малі випадкові значення.

2. Визначення елемента навчальної множини: (вхід - вихід).

Входи $(x_1, x_2, \dots x_n)$, повинні розрізнятися для всіх прикладів навчальної множини.

3. Обчислення вихідного сигналу: $y_{im} = f(S_{jm})$

$i_m = 1, \dots, N_m, m = 1, \dots, L$

де S - вихід суматора, w - вага зв'язку, y - вихід нейрона, b - зсув, i - номер нейрона, N - число нейронів у прошарку, m - номер прошарку, L - число прошарків, f- передатна функція.

4. Налаштування синаптичних ваг:

$$w_{ij}(t+1) = w_{ij}(t) + r\, g_j\, x_i' , \qquad (2.7)$$

де w_{ij} - вага від нейрона *i* або від елемента вхідного сигналу *i* до нейрона *j* у момент часу t, x_i'- вихід нейрона *i*, r - швидкість навчання, g_j - значення похибки для нейрона *j*.

Якщо нейрон з номером j належить останньому прошарку, тоді

$$g_j = y_j(1 - y_j)(d_j - y_j) , \qquad (2.8)$$

де d_j - бажаний вихід нейрона j, y_j - поточний вихід нейрона *j*.

Якщо нейрон з номером j належить одному з прошарків з першого по передостанній, тоді k пробігає всі нейрони прошарку з номером на одиницю більше, ніж у того, котрому належить нейрон j.

Зовнішні зсуви нейронів b налаштовуються аналогічним чином.

Тип вхідних сигналів: цілі чи дійсні.

Тип вихідних сигналів - дійсні з інтервалу, заданого передатною функцією нейронів. Тип передатної функції - сигмоїдна. Сигмоїдні функції є монотонно зростаючими і мають відмінні від нуля похідні по всій області визначення. Ці характеристики забезпечують правильне функціонування і навчання мережі. Області застосування – це розпізнавання образів, класифікація, прогнозування. Недоліки - це низька швидкість навчання [4]. Переваги. Ефективний та

популярний алгоритм для вирішення численних практичних задач. Модифікації. Модифікації алгоритму зворотного поширення зв'язані з використанням різних функцій похибки, різних процедур визначення напрямку і величини кроку.

Вибір числа шарів. Якщо в конкретному завданні гіпотеза про лінійну роздільність класів виглядає правдоподібно, то можна обмежитися одношаровим персептроном. Двошарові мережі дозволяє представляти звивисті нелінійні границі, і в більшості випадків цього вистачає. Тришарові мережі мають сенс бути використаними для представлення складних багатозв'язних областей. Чим більше шарів, тим багатший клас функцій реалізує мережу, але тим гірше сходяться градієнтні методи, і тим важче її навчити.

Вибір числа нейронів в прихованому шарі H виробляє різні способи, але жоден з них не є найкращим.

1. Візуальний спосіб. Якщо межа класів (або крива регресія) занадто згладжена, значить, мережа занадто спрощена, і необхідно збільшувати число нейронів в прихованому шарі. Якщо межа класів (або крива регресії) відчуває занадто різкі коливання, на тестових даних спостерігаються великі викиди, ваги мережі приймають великі по модулю значення, то мережа переускладнена, і прихований шар слід скоротити. Недолік цього способу в тому, що він підходить тільки для задач з низькою розмірністю простору (невеликим числом ознак).

2. Оптимізація *H* по зовнішньому критерію, наприклад, за критерієм сковзного контролю або середньої помилки на незалежній контрольній вибірці $Q(X^k)$. Залежність зовнішніх критеріїв від параметра складності, яким є H, звичайно має характерний оптимум. Недолік цього способу в тому, що доводиться багато раз заново будувати мережу при різних значеннях параметра H, а в разі змінного контролю - ще й при різних розбитті вибірки на навчальну і контрольну частини.

Розглянемо метод стохастичного градієнтного спуску.

Вхід: вибірка X^ℓ темп навчання η, параметр λ

Вихід: $w \equiv \left(w_{jh}, w_{hm}\right) \in \mathbb{R}^{H(n+M+1)+M}$

Недоліки- для кожного об’єкту рахуємо функцію втрат

Кількість вагових коефіцієнтів $N = (n+1)H + (H+1)M$

Складність алгоритму: $O(N^2)$

Градієнтний спуск працює в просторах з будь-яким числом вимірів, навіть у нескінченновимірних. В останньому випадку простір пошуку зазвичай є простором функцій, і для визначення напрямку спуску здійснюється обчислення похідної Гато функціоналу, який мінімізують.

Якщо кривина заданої функції дуже різниться в різних напрямках, то градієнтному спускові може знадобитися багато ітерацій для обчислення локального мінімуму з потрібною точністю. Для таких функцій повільне збігання лікується передобумовлюванням, яке змінює геометрію простору так, щоби надати множинам рівнів функції форми концентричних кіл. Проте побудова та застосування передобумовлювання можуть бути обчислювально витратними.

Градієнтний спуск можна поєднувати з лінійним пошуком, який на кожній ітерації шукає локально оптимальний розмір кроку γ . Виконання лінійного пошуку може бути витратним за часом. З іншого боку, застосування незмінного малого γ може давати погану збіжність.

Кращими альтернативами можуть бути методи на основі методу Ньютона та обернення гессіану із застосуванням методик спряжених градієнтів. Загалом такі методи збігаються за меншу кількість ітерацій, але вартість кожної ітерації є вищою. Прикладом є метод БФГШ, який складається з обчислення на кожному кроці матриці, на яку множиться вектор градієнту, щоби йти в «найкращому» напрямку, поєднаного зі складнішим алгоритмом лінійного пошуку для

знаходження «найкращого» значення γ. Для надзвичайно великих задач, у яких домінують питання комп'ютерної пам'яті, замість БФГШ або найшвидшого спуску повинні застосовуватися методи з обмеженою пам'яттю, такі як О-БФГШ (LBFGS) [4].

Розглянемо швидкий проксимальний градієнтний метод. Інше розширення градієнтного спуску виникло завдяки Юрієві Нестерову 1983 року, і було згодом узагальнене. Він пропонує просту видозміну цього алгоритму, яка уможливлює швидке збігання для опуклих задач. А саме, якщо функція F є опуклою, а ∇F є ліпшицевою, і немає припущення, що Fє сильно опуклою, то похибку цільового значення, породжувану методом градієнтного спуску на кожному кроці k, буде обмежено O(1 / k). Із застосуванням методики прискорення Нестерова похибка знижується до $O(1 / k^2)$ [3].

Ще одним розширенням, яке знижує ризик застрягнути в локальному мінімумі, а також істотно прискорює збіжність у випадках, коли інакше процес би сильно зиґзаґував, є метод імпульсу (англ. momentum method), який використовує член імпульсу по аналогії з «масою ньютонових частинок, які рухаються в'язким середовищем у консервативному силовому полі». Цей метод часто використовують як розширення алгоритмів зворотного поширення, що застосовуються для тренування штучних нейронних мереж. Метод імпульсу з'явився в основоположному документі Рамелхарта, Хінтона і Вільямса на навчанні зворотного поширення (backpropagation learning). Стохастичний градієнтний спуск з імпульсом запам'ятовує оновленя Δ w на кожній ітерації і визначає наступне оновлення у вигляді випуклої(лінійної) комбінації градієнта і попереднього оновлення.

$$w := w - \eta \nabla Q_i(w) + \alpha \Delta w \quad , \tag{2.9}$$

де $Q_i(w)$ - функція втрат, що мінімізується, параметр w, за яким мінімізується - η - довжина кроку (іноді званої темпом навчання в машинному навчанні).

Назва імпульс походить від аналогії з імпульсом у фізиці: вектор вагових коефіцієнтів, розглядати як частку, що рухається через простір параметрів, тягне за собою прискорення від градієнта втрати («сила»). На відміну від класичного стохастичного градієнтного спуску, він прагне тримати рух в одному напрямку, запобігаючи коливання. Momentum був успішно використаний протягом декількох десятиліть [3].

У стохастичному (або «он-лайн») градієнтном спуску, істинний градієнт $Q_i(w)$апроксимується градієнтом по одному об'єкті.

Алгоритм стохастичного градієнтного спуску може бути представлений таким чином:

- Виберіть вихідний вектор параметрів w та темп навчання (learning_rate) η
- Повторити до тих пір, поки не буде отримано приблизне мінімальне.
- Випадково перетасувати об’єкти в навчальному наборі

Компроміс між обчислення істинного градієнта і градієнта для одного об'єкту - це обчислення градієнта для більш, ніж одного навчального об'єкту (так званої міні-порції (mini-batch)) на кожному кроці. Виконання стає значно краще, ніж істинний стохастичні градієнтний спуск, так як при програмуванні можна використовувати бібліотеки векторизації, а не обчислення кожного кроку окремо. Це також може привести до більш гладенькою збіжності, так як градієнт на кожному кроці при обчисленні використовує все більше об'єктів з навчальної вибірки [5].

Збіжність стохастичного градієнтного спуску була проаналізована за допомогою теорії опуклої мінімізації та стохастичної апроксимації. Якщо коротко, коли темп навчання η зменшується з швидкістю, обраною належним чином і при відносно помірних (поблажливих) припущеннях, стохастичний градієнтний спуск майже напевно збігається до глобального мінімуму, коли цільова функція є опуклою або псевдоопуклою, а в іншому випадку сходиться майже напевно до локального мінімуму. Це насправді є наслідком теореми Роббінса-Зигмунда.

Розглянемо метод зворотного поширення.

Вхід: вибірка X^ℓ темп навчання η,

Параметр λ, H - число нейронів прихованого шару

Вихід: $w \equiv \left(w_{jh}, w_{hm}\right) \in \mathbb{R}^{H(n+M+1)+M}$

Складність алгоритму - $O(3N)$

Переваги Методу зворотного поширення.

• Досить висока ефективність. У разі двошарової мережі прямий хід, зворотний хід і обчислення градієнта вимагають порядку O (Hn + HM) операцій.

• Через кожен нейрон проходить інформація тільки про зв'язкових з ним нейронах. Тому back-propagation легко реалізується на обчислювальних пристроях з паралельною архітектурою.

• Високий ступінь спільності. Алгоритм легко записати для довільного числа шарів, довільної розмірності виходів і виходів, довільної функції втрат і довільних функцій активації, можливо, різних у різних нейронів. Крім того, back-propagation можна застосовувати спільно з різноманітними градієнтними методами оптимізації: методом якнайшвидшого спуску, спряжених градієнтів, Ньютона-Рафсона і ін [6].

Недоліки методу зворотного поширення.

• Метод успадковує відомі недоліки градієнтного налашатування ваг в одношаровому перцептрони. Тут також виникають проблеми повільної збіжності

2.3 Методи вибору параметрів

З метою оптимізації моделі прямого поширення використовують ряд методів. Найбільш відомі з них: попередня зупинка, регуляризація, усереднення, відсікання та нарощування.

Розглянемо метод попередньої (ранньої) зупинки. Ретельна підгонка параметрів моделі на фіксованій навчальній вибірці може призвести до надто точного наслаштування на особливості конкретних даних, що неминуче викличе збільшення реальної похибки. Очевидним виходом із цієї ситуації є зупинка процесу навчання до того моменту, доки реальна похибка не почне зростати з причини надлишкового регулювання. Такий підхід, що отримав назву методу попередньої зупинки, часто застосовується на практиці і має різні модифікації, пов’язані з використанням різноманітних методик визначення моменту зупинки процесу навчання.

У найпростішому випадку регулярно через певні інтервали проводять обчислення навчальної та реальної похибки. Зупинку виконують за наявності тенденції зростання реальної похибки. Але таке визначення моменту зупинки має небезпеку попадання у локальний мінімум, якщо у вхідному сигналі присутній шум або значна нелінійність. Для того, щоб уникнути попадання в

локальний мінімум, оцінку реальної похибки виконують на підмножині параметрів. Існують також більш складні методики, що базуються на встановленні порога чутливості, перевищення якого реальною похибкою є умовою попередньої зупинки. Конкретну величину такого порога визначають у результаті практичної роботи з моделі, оскільки він залежить як від характеру вхідних даних, так і від архітектури нейронної мережі [7].

Розглянемо регуляризацію. Методи, які називають регуляризацією, були започатковані у межах нових підходів до вирішення некоректно поставлених задач. Реальну похибку представляють у вигляді суми середньоквадратичної похибки та деякої функції *R(W)*, що задає попередній стан моделі:

$$\boldsymbol{E} = \frac{1}{N}\sum_{n=1}^{N}\left(\boldsymbol{y}^{(n)} - \tilde{\boldsymbol{y}}^{(n)}\right)^2 + k\boldsymbol{R}(\boldsymbol{W}), \qquad (2.10)$$

де *R(W)* – функція, що задає попередній стан мережі, k — коефіцієнт регуляризації, який задає ступінь впливу *R(W)* на реальну похибку, *N* - кількість об'єктів навчальної вибірки, $y^{(n)}$ – результат на об'єкті n,

Мінімізація *E* прямо відповідає вирішенню дилеми відхилення або дисперсії, оскільки середньоквадратична похибка задає статистичне відхилення, а функція *R(W)* – величину дисперсії. Коефіцієнт регуляризації k відіграє роль параметра, що формує співвідношення між статистичним відхиленням та дисперсією при вирішенні дилеми відхилення або дисперсії і змінюється в діапазоні $0 < k < \frac{2\sigma^2}{\boldsymbol{W}^{\mathrm{T}}\boldsymbol{W}}$.

Його зростання призводить до збільшення відхилення, оскільки частка середньоквадратичної похибки у виразі () зменшується. Відповідно, зменшення коефіцієнта k збільшує вплив середньоквадратичної складової, що призводить до зростання дисперсії. Головна ідея регуляризації полягає в представленні проблеми оптимізації моделі у вигляді некоректно поставленої задачі. Розв'язок

даної задачі одержують, виходячи з варіаційного принципу, при реалізації якого використовують вхідний набір даних та попередню інформацію про гладкість функції. Гладкість визначається за допомогою функціонала гладкості *R[y]* таким чином, що менші його значення відповідають більшій гладкості функції. Отже, регуляризація моделі може бути проведена шляхом формування вихідних даних у відповідності до функції.

2.4 Оцінка моделі та системи рекомендацій

Реальною похибкою називають похибку, з якою функціонує моделі в умовах реальних даних, на відміну від навчальної похибки, що визначається при роботі з навчальною вибіркою. Як правило, реальна похибка завжди перевищує навчальну. Тому для визначення ступеня коректності роботи моделі дуже важливо мати оцінку реальної похибки. Очевидний підхід, що широко застосовується для визначення реальної похибки, полягає у використанні оцінки на незалежній підмножині. Незалежна підмножина повинна включати діапазон зміни даних, який планується використовувати в умовах реального функціонування моделі, бути сформованою до проведення навчання моделі і не повинна використовуватися в майбутньому. Такий підхід характерний для моделі, які працюють з неперервними потоками звукової, відео- або телекомунікаційної інформації. В цьому випадку не виникає труднощів при формуванні незалежної підмножини даних.

Коли вхідний масив даних обмежений, використовують метод багаторазової оцінки. Цей метод полягає в реалізації об’єднаної процедури навчання моделі та визначення її реальної похибки.

Емпіричні оцінки узагальнюючої здатності застосовуються в тих випадках, коли не вдається скористатися теоретичними.

Перехресна перевірка (англ. cross-validation) — метод оцінювання достовірності математичної моделі з метою перевірки, наскільки результати статистичного аналізу узагальнюються на незалежному наборі даних. Переважно використовується в задачах прогнозування для оцінювання точності моделі.

Нехай дана вибірка $X^\ell = (x_i, y_i)_{i=1}^\ell$. Розіб'ємо її *N* різними способами на дві неретинаючі підвибірки - навчальну X_n^l довжини *l* і контрольну X_n^k довжини $k = L - l$. Для кожного розбиття *n = 1,. . . , N* побудуємо алгоритм $a_n = \mu(X_n^\ell)$ і обчислимо значення $Q_n = Q(a_n, X_n^k)$. Середнє арифметичне значень Q_n за всіма розбиття називається оцінкою ковзного контролю (cross-validation, CV):

$$\mathrm{CV}(\mu, X^L) = \frac{1}{N}\sum_{n=1}^{N} Q(\mu(X_n^\ell), X_n^k), \qquad (2.11)$$

де $\mu(X_n^l)$ – алгоритм моделі, X_n^l навчальна вибірка, X_n^k - контрольна вибіка

Одноразова перехресна перевірка передбачає розбиття вибірки на взаємодоповнювані підвибірки з метою проведення аналізу на одній частині (що називається навчальним набором, англ. training set) і перевірки аналізу на іншій частині (що називається контрольним, або тестовим набором, англ. validation set, testing set). Для зниження дисперсії здійснюється багаторазова перехресна

перевірка із застосуванням різного розбиття, і результати цих перевірок усереднюються.

Недоліками змінного контролю є: обчислювальна неефективність, висока дисперсія, неповне використання наявних даних для навчання через скорочення довжини навчальної вибірки з L до l.

Висновки до розділу

У розділі розглянуто базові математичні поняття загальної постановки задачі машинного навчання. Представлено формальний опис рекомендаційних систем та їх структури. Описано методологію навчання текстового класифікатора, пошуку оптимальної кількості структурних одиниць, зокрема кроку Навчання та параметрів регуляризації.

Також приведено алгоритм пошуку вагових коефіцієнтів за допомогою методу стохастичного градієнтного спуску та детальний опис вибору параметрів для даної задачі оптимізації. Розглянуто способи оцінки побудованої моделі.

Перевагами цих методів є їх паралелізм типових процесів для ефективного і ресурсно незатратного розв’язку єдиною глобальної задачі, здатність навчатися, що веде до універсальності, можливість вибору параметрів. Дані методи широко використовуються спеціалістами у галузі аналізу даних протягом останнього десятиріччя

РОЗДІЛ 3 ТЕСТУВАННЯ ТА ПРАКТИЧНА РЕАЛІЗАЦІЯ СИСТЕМИ РЕКОМЕНДАЦІЙ

3.1 Вибір програмної платформи

Python - високорівнева мова програмування загального призначення, орієнтована на підвищення продуктивності розробника і читання коду. Синтаксис ядра Python мінималістичний. У той же час стандартна бібліотека включає великий обсяг корисних функцій.

Ця мова підтримує кілька парадигм програмування, в тому числі структурний, об'єктно-орієнтоване, функціональне, імперативне і аспектно-орієнтоване. Основні архітектурні риси - динамічна типізація, автоматичне керування пам'яттю, повна інтроспекція, механізм обробки виключень, підтримка багатопотокових обчислень і зручні високорівневі структури даних. Код в Python організовується у функції та класи, які можуть об'єднуватися в модулі (вони в свою чергу можуть бути об'єднані в пакети).

Anaconda - дистрибутив мови програмування Python, що включає в себе набір бібліотек для наукових і інженерних розрахунків, зокрема модуль Scikit-Learn, менеджер пакетів Конда, інтерактивну оболонку IPython.

Scikit- Learn - є безкоштовним програмним забезпеченням(модуль) для машинного навчання для мови програмування Python. Описані нижче переваги:

- прості та ефективні інструменти для аналізу даних;
- доступність для всіх, і можливість перевикористання в різних контекстах;
- відкритий вихідний код, комерційно використовується, є ліцензія.

Наведемо важливі модулі з Scikit- Learn та їх опис (таблиця 3.1)

Таблиця 3.1 - Назва модулів та їх опис

Назва модуля	Опис
NumPy	Дозволяє зручно працювати з векторами і матрицями, при цьому реалізація всіх операцій з ними ретельно оптимізована
SciPy	Зібрані більш високорівневі алгоритми, наприклад, методи оптимізації, побудований над NumPy
Scikit- learn	Містить алгоритми машинного навчання, побудований над SciPy
Matplotlib	Візуалізація результатів
Pandas	Робота з табличними даними

3.2 Вибір вхідних даних для аналізу, початкові перетворення

Дані, що обули відібрані для дослідження – розмічена та нерозмічена відгуків вибірки, взяті з відкритого доступу. Вхідний масив інформації включає деякі показники, що є виміряними характеристиками об’єктів (таблиця 3.2).

Таблиця 3.2 - Якісний опис вхідних даних

Вибірка	Контент	Опис
Розмічена вибірка	Відгук	Використовується для навчання моделі
	Клас (позитивний / негативний)	
Вибірка про користувачів	Відгук	На основі вибірки створюєтья алгоритм підбору рекомендацій
	Розважальний бізнес	
	Місто	
	Категорії	

Перейдемо до первинної обробки даних. Для цього використаємо sklearn.preprocessing: Preprocessing and Normalization

Модуль включає

- масштабування;
- централізація;
- нормалізація;
- бінаризація;
- кодування категоріальні ознак;
- вставлення пропущених даних.

Пакет sklearn.preprocessing пропонує кілька поширених допоміжних функцій і трансформуючі класи для обробки сирих вихідних векторів ознак у вигляд, який є більш зручним для наступного потоку оцінок.

Стандартизація наборів даних є загальною вимогою для багатьох оцінок машинного навчання, реалізовані в scikit learn; вони можуть вести себе погано, якщо індивідуальні особливості хоча б більш-менш не схожі на стандартно нормально розподілені дані: Gaussian з нульовим середнім і одиничною дисперсією. На практиці ми часто ігноруємо форму розподілу і просто перетворюємо дані для центрування, виділивши середнє значення кожної функції, а потім масштабувати його шляхом ділення неконстантних особливостей їх стандартного відхиленням [8].

Наприклад, багато елементів, які використовуються в цільової функції алгоритму навчання (наприклад, RBF ядра опорних векторів або l1 і l2 регуляризатора лінійних моделей) Припустимо, що всі функції зосереджені навколо нуля і мають дисперсію в тому ж порядку. Якщо функція має дисперсію, що на порядки більше, ніж інші, він може домінувати в цільову функцію і зробити оцінювач не може дізнатися з інших функцій коректно, як очікувалося.

Метод масштаб забезпечує швидкий і простий спосіб для виконання цієї операції на одному масив типу набору даних (таблиця 3.3).

Таблиця 3.3 - Опис методу масштабування

sklearn.preprocessing.scale		
Параметри	Тип	Опис
X	array-like, sparse matrix	Масив даних для масштабування
with_mean	boolean	Для центрування
with_std	boolean	Для одиничної дисперсії

Нормалізація являє собою процес масштабування окремих зразків, щоб мати одиничну норму. Цей процес може бути корисним, якщо ви плануєте використовувати квадратичну форму, такі як точку продукт або будь-яке інше ядро, щоб кількісно оцінити схожість будь-якої пари зразків. Це припущення є основою моделі векторного простору часто використовуються в тексті класифікації та кластеризації контекстів [9]. Метод нормалізує, забезпечує швидкий і простий спосіб для виконання цієї операції на одному масив, як набір даних, або за допомогою L1 або L2 норми (таблиця 3.4).

Таблиця 3.4 - Опис методу нормалізації

sklearn.preprocessing.normalize		
Параметри	Тип	Опис
X	Array-like, sparse matrix, shape [n_samples, n_features]	Матриця ознак
Norm	‘L1’, ‘L2’, або ‘max’	Норма

Часто ознаки або показники не даються у вигляді безперервних значень, але категоріально. Наприклад, людина може мати ознаки [«чоловік», «жінка»]. Такі ознаки можуть бути ефективно закодовані у вигляді цілих чисел, наприклад може бути виражено як [0, 1], для цього використаємо модуль sklearn.preprocessing.OneHotEncoder.

3.3 Метод класифікації тексту

Багатошаровий перцептроний класифікатор (Multi-layer Perceptron classifier) у модулі називається sklearn.neural_network.MLPClassifier. Ця модель оптимізує логарифмічну функцію втрат з використанням LBFGS або стохастичний градієнтний спуск.

BFGS з обмеженою пам'яттю (L-BFGS або LM-BFGS) являє собою алгоритм оптимізації в сімействі квазіньютоновских методів, аппроксимуючий алгоритм Бройде-Флетчер-Гольдфарб-Шанно (BFGS), використовуючи обмежену кількість пам'яті комп'ютера. Це популярний алгоритм для оцінки параметрів в машинному навчанні.

MLPClassifier навчається ітеративно, оскільки на кожен крок часу часткові похідні функцій втрат по параметрах моделі обчислюються для поновлення параметрів. Він також може мати доданок регуляризації доданий до функції втрат, яка не дає збільшуватись параметрам моделі для запобігання перенавчання. Ця реалізація працює з даними, представленими у вигляді щільних масивів Numpy або розріджених SciPy масивів значень з плаваючою точкою. Розглянемо вхідні параметри (таблиця 3.5) та методи (таблиця 3.6)[10].

Таблиця 3.5 - Опис параметрів модуля методу класифікації

sklearn.neural_network.MLPClassifier			
Параметр	Тип	За замовчуванням	Опис
hidden_layer_sizes	tuple кортеж	(100,)	Довжина масиву = кількість щарів – 2; і-ий елемент - це число нейронів у і-тому прихованому шарі
activation	Елемент з множини {‘identity’, ‘logistic’, ‘tanh’, ‘relu’}	relu	Функція активації для прихованих шарів нейронної мережі
solver	Елемент з множини {‘lbfgs’, ‘sgd’, ‘adam’}	adam	Метод для розв’язку задачі оптимізації - знаходження значень вектора ваги

Продовження таблиці 3.5

Параметр	Тип	За замовчуванням	Опис
alpha	float	0.0001	Параметр регуляризації, Штраф L2
batch_size	int	auto	Розмір міні-порції (minibatches) для стохастичних оптимізаторів. Якщо оптимізатор «lbfgs», класифікатор не використовуватиме minibatch. batch_size = min (200, n_samples)
learning_rate	Елемент з множини {‘constant’, ‘invscaling’, ‘adaptive’},	constant	Використовується тільки при solver = «sgd»
early_stopping	bool	False	Раннє припинення/ зупинка, півищує рівень регуляризації тільки solver=’sgd’ or ‘adam’

Розглянемо параметр activation (функція активаціїї для прихованих шарів нейронної мережі)

- «identity» - функція активації, корисна для реалізації задач з лінійнуими залежностями, повертає f(x) = x;
- « logistic» - логістична сигмовидна (sigmoid) функція;
- « tanh » - гіперболічний тангенс;
- «relu» - функція-випрямляч, повертає f(x) = max (0, x). Також називають випрямлений лінійний юніт (a rectified linear unit ReLU);
- розглянемо параметр solver (метод оптимізації для навчання);
- «lbfgs» є оптимізатор в сімействі квазіньютоновскіх методів;
- «sgd» - стохастичний градієнтний спуск (stochastic gradient descent);
- «adam» відноситься до оптимізаторів, заснованих на методі стохастичного градієнта запропонованого науковцями Кінгма, Дидерік та Джиммі Ба.

Примітка: метод за замовчуванням «адам» працює дуже добре на відносно великих наборах даних (з довжиною навчальної вибірки порядку тисячі або більше) з точки зору часу навчання і оцінка перевірки. Однак, для невеликих наборів даних, «lbfgs» може збігатися швидше і краще.

Параметр alpha – парметр регуляризації. У математиці і статистиці, а також в задачах машинного навчання і обернених задачах, регуляризація означає додавання деякої додаткової інформації, щоб знайти рішення некоректно сформованної задачі, або щоб уникнути перенавчання. Наприклад, це може бути обмеження гладкості результуючої функції або обмежень на норму векторного простору.

Розглянемо парметр batch_size. Компроміс між обчислення істинного градієнта і градієнта для одного об'єкту - це обчислення градієнта для більш, ніж одного навчального об'єкту (так званої міні-порції (mini-batch)) на кожному кроці. Виконання стає значно краще, ніж істинний стохастичні градієнтний спуск, так як при програмуванні можна використовувати бібліотеки

векторизації, а не обчислення кожного кроку окремо. Це також може привести до більш гладенькою збіжності, так як градієнт на кожному кроці при обчисленні використовує все більше об'єктів з навчальної вибірки.

Розглянемо learning_rate. learning_rate_init - ініціалізований темп навчання, переданому в метод:

– Constant - постійний темп навчання, дорівнює learning_rate_init
– Invscaling - зворотне масштабування темпу начання, поступово зменшується з часом t (на кожній ітерації методу СГС).

Обчислюэться за формулою

$$\text{learning\_rate} = \frac{\text{learning\_rate\_init}}{\sqrt{t}} \tag{3.1}$$

Adaptive - адаптивний тримає темп навчання постійним, дорівнює learning_rate_init до тих пір, поки функція втрат (при навчанні) продовжує знижуватися, кожного разу як за дві послідовні епохи (новий крок подання всього набору об’єктів) не зменшується функція втрат навчання, хоча б на tol (за замовчуванням 0,0001) або не збільшується валідаційний рахунок (validation score) хоча б на tol (за замовчуванням 0,0001), або рання зупинка (early_stopping), то поточний темп навчання навчання ділиться на 5.

Зробимо огляд early_stopping**.** Параметр показує чи слід використовувати раннє припинення навчання, коли результат (валідаційний рахунок) (validation score) не поліпшується. Якщо встановлено значення істинно, то він автоматично відкидає 10% навчальних об’єктів для подальшої перевірки і припиняє навчання, коли валідаційний рахунок не поліпшується, хоча б на tol протягом двох послідовних епох. Діє тільки при solver = "sgd " або "adam".

Таблиця 3.6 - Опис методів модуля методу класифікації

Назва методу	Опис
fit (X, y)	Навчання мережі
get_params([deep])	Отримати параметри для цього оцінювача
predict(X)	Передбачити за допомогою багатошарового перцептронного класифікатора
predict_log_proba(X)	Повертає логарифм імовірнісних оцінок.
predict_proba(X)	Оцінка ймовірності
score(X, y[,sample_weight])	Повертає точність, визначену на наведеній тестовій даних і labels.
set_params(params)	Встановити параметри цього оцінювача

3.4 Тестування і застосування методу в рекомендаційній системі

Побудуємо систему рекомендацій. За допомогою класу мови програмування Python і дистрибутиву Anaconda, який пропонує модуль Нейронні мережі для задач класифікації (class sklearn.neural_network.MLPClassifier). Цей алгоритм використовує логістичну функцію втрат, оскільки вона є неперервною, тому градієнтні методи спуску можуть бути використані.

Маємо один прихований шар мережі з трьох (перший – вхідний, третій - вихідний). Оберемо кількість нейронів у прихованому щарі (hidden_layer_sizes = 100). Функцією активації нейронів під час виконання кожного проходу буде логістична (activation = 'log'). Вибір пояснюється тим, що похідна цієї функції може бути виражена через її значення, полегшує використання цієї функції при навчанні мережі за алгоритмом зворотного поширення. Метод оптимізації функції втрат (solver = 'sgd'). Розглянемо параметр регуляризації, що допомагає запобігати перенавчанню, візьмемо його за замовчуванням (alpha = 0.0001). При навчанні будемо розбивати всі об’єкти на порції, які є характерні для нашої моделі(batch_size = 'auto'). Зміну параметру темп навчання зробимо постійним (learning_rate = 'constant'). Ініціалізование темп навчання (learning_rate_init) дорівнює 0.001. Оберемо масимальну кількість ітерацій після якої метод оптимізації ваг нейронної мережі припине свою роботу max_iter = 200. Функцію перемішування ініціалізуємо як так (shuffle = True), бо це має значний вплив на кінцеві показники навчання мережі. Ще один параметр-константа, що використовується для запобігання перенавчання (tol = 0.0001). Інерцію для методу стохастичного градієнтного спуску оберемо momentum = 0.9, оскільки можливі різноманітні дані та коливання функції втрат, яку ми оптимізуємо, а цей

показник приводить до уникнення стрімкого зниження збіжності у так званих ярах, особливо якщо це локальний мінімум, а не глобальний. Оберемо показник nesterovs_momentum = False. Вхідний параметр для методу з назвою рання зупинка оберемо ні (early_stopping = True). Узагальному випадку це часто впливає негативно на затратність виконання бо ускладнює алгоритм, але при достатній потужності серверів – це найкращий, найефективніший спосіб запобігання перенавчання, що дає найбільш якісний результат. Відсоток даних для валідації (validation_fraction = 0.1).

Оберемо функцію активації для такої багатошарової перцептронної нейронної мережі прямого поширення з кількістю вхідних нейронів 7, вихідних 2, кількістю прихованих шарів 3, у кожному по 5 нейронів, метод оптимізації – стохастичний градієнтний спук та функція втрат - логістична. Розглянемо результат (таблиця 3.7).

Таблиця 3.7 - Оцінка функції активації

Функція активації	Точність
f(x) = x	46%
Логістична f(x) = 1 / (1 + exp(-x))	72%
f(x) = max(0, x)	64%

3.5 Оцінка результатів

Метод стохастичного градієнтного спуску, будемо використовувати нейронну мережу з трьома обробними шарами: вхідний шар 7 нейрона, перший обробляє шар 10 нейронів з активацією РЕЛУ.

Другий шар 50 нейронів з активацією РЕЛУ, третій шар - 2 нейрона (по числу класів) з активацією SoftMax. Мережа навчилася за 2500 епохи, результати представлені нижче (рисунки 3.1 - 3.2).

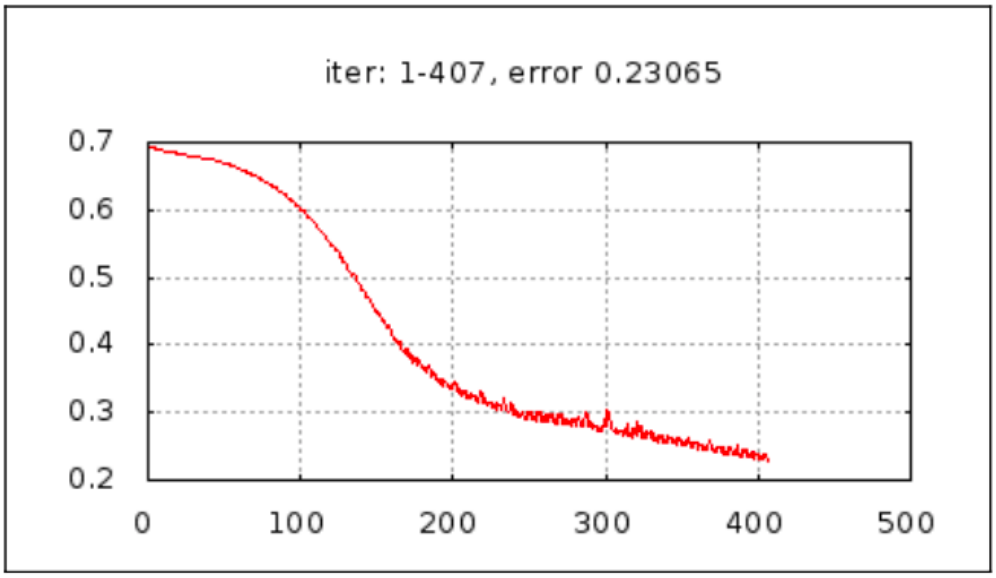

Рисунок 3.1 - Залежність функії втрат від кількості епох на моделі 1

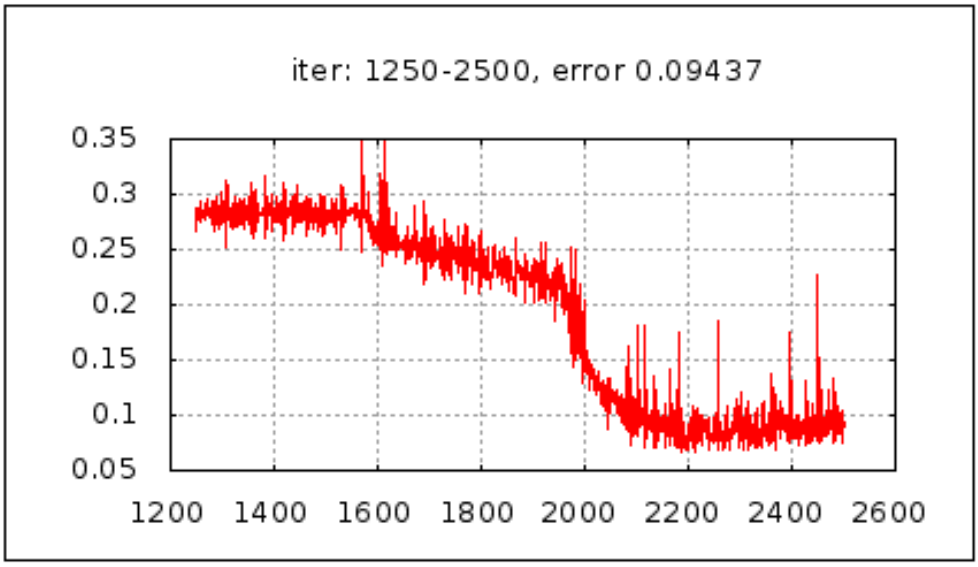

Рисунок 3.2 - Залежність функії втрат від кількості епох на моделі 1 приближено

Метод стохастичного градієнтного спуску, будемо використовувати нейронну мережу з п’ятьма обробними шарами: вхідний шар 7 нейронів, Другий, третій і четвертий шари мають 50 нейронів з активацією РЕЛУ, третій шар - 2 нейрона (по числу класів) з активацією SoftMax. Мережа навчилася за 1000 епох, результати представлені нижче (рисунки 3.3 -3.4).

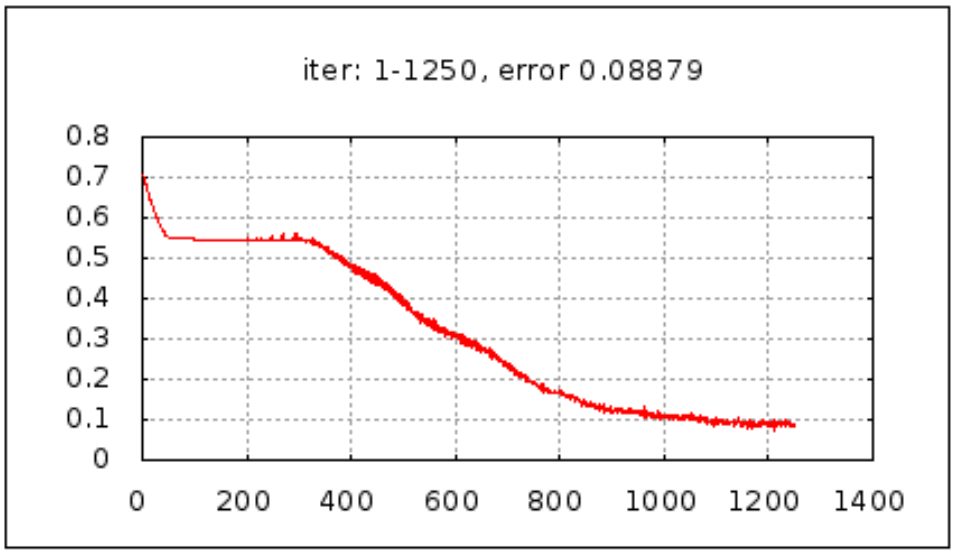

Рисунок 3.3 - Залежність функії втрат від кількості епох на моделі 2

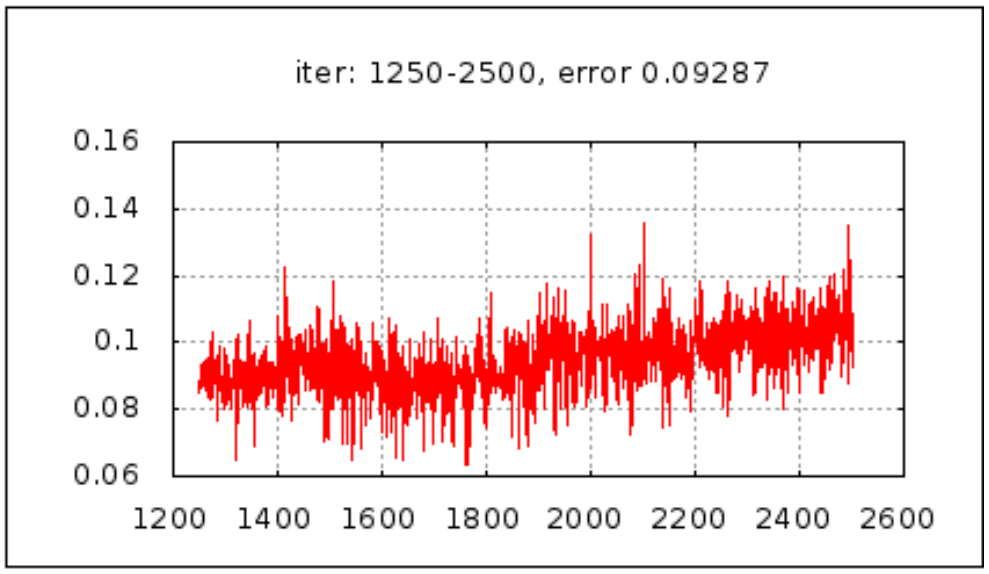

Рисунок 3.4 - Залежність функії втрат від кількості епох на моделі 2 приближено

Висновки до розділу

У даному розіділ була обрана платформа як інструмент для прогрманої реалізації нашої задачі, наступним кроком був збір даних, що виконувався зі сховища даних за допомогою аналітичних міркувань. Первинна обробка даних відбулася за допомогою масштубування, що значно поліпшила опрацювання даних у подальшому.

Було реалізовано байєсівьський наївний класифікатор для задачі розбиття вибірки документів на позитивні та негативні, для навчання розбито розмічену вибірку та тренування та тест. Застосовано паралелізований алгоритм для

знаходження рекомендацій. Більш того, відбувся вибір суттєвих параметрів даного типу системи, таких як: метод оптимізації та його параметри (темп навчання, параметр регуляризації, момент інерції), розмір міні-порцій, показник перемішування, параметр регуляризації (раннє припиненння).

Наведено пояснення до всіх модулів, методів, класів коду, а також графіки й таблиці для наочності результатів.

РОЗДІЛ 4 РОЗРОБКА СТАРТАПУ

4.1 Опис ідеї проекту

Опис ідеї стартам проекту описано в таблиці 4.1.

Таблиця 4.1 - Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Розробка моделі для прогнозування нелінійних нестаціонарних процесів з різними вхідними параметрами	Аудит підприємств	Зменшення корумпованості підприємства
	Організаційне управління	Правильна організаційна структура

Сильні, слабкі та нейтральні характеристики ідеї проекту зображено в таблиці 4.2.

Таблиця 4.2 - Визначення характеристик ідеї проекту

№ п/п	Техніко-економічні характеристики ідеї	Потенційні товари/концепції конкурентів			W (слабка сторона)	N (нейтр. сторона)	S (сильна сторона)
		SNI	lLeasoft	PAnYst			
1.	Відсутність прив’язки до формату даних	-+		-	+		
2.	Моделювання і прогнозування	-		-			+
3.	Ймов. аналіз	+		+			+
5.	АВС-аналіз	+		+		+	

5.	Відомість бренду	-	+	-	+		

4.2 Технологічний аудит ідеї проекту

Технологічний аудит ідеї проекту наведений у таблиці 4.3.

Таблиця 4.3 - Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Ігровий підхід	NetLogo	Технологія наявна і не потребує змін. Потрібно реалізувати алгоритм.	Технологія загальнодоступна
2	Ймовірностний аналіз	NetLogo	Необхідно реалізувати алгоритм	Технологія загальнодоступна
3	Моделювання та аналіз результатів	NetLogo	Необхідно реалізувати розроблені моделі	Технологія загальнодоступна
Обрана технологія реалізації ідеї проекту: для реалізації проекту обрана мова програмування NetLogo.				

4.3Аналіз ринкових можливостей запуску стартап-проекту

Характеристика потенційного ринку стартап-проекту наведена у таблиці 4.4.

Таблиця 4.4 - Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн/ум.од	2500 ум.од
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі (або по ринку), %	30 %

Характеристика потенційних клієнтів стартап-проекту наведена в таблиці 4.5.

Таблиця 4.5 - Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Аудит організаційної структури підприємств	Середній та великий бізнес, що застосовують системи керування ресурсів підприємства, державні підприємтва, аудиторські компанії	ERP система підприємства, розміри оброблюваних даних, технічні обмеження, бюрократичні обмеження	Ефективність прогнозування Швидка обробка даних Оптимальне використання ресурсів

Можливі загрози для стартап-проекту наведені у таблиці 4.6.

Таблиця 4.6 - Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Невпорядкованість і неповнота даних	Клієнтські бази можуть містити невпорядковані дані і також певні дані можуть бути відсутніми	Додавання модуля попередньої обробки даних
2	Нестача технічних ресурсів	Клієнти можуть мати обмежені локальні технічні ресурси, недостатні для повноцінної роботи системи	Винесення модуля обчислення на сервери компаній-партнерів

Фактори можливостей наведені у таблиці 4.7.

Таблиця 4.7 - Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Хмарні обчислення	Можливість виконання усіх обчислень на віддалених серверах	Пристосування модулів обчислення для роботи на сервері
2	Коригування прогнозу	Можливість коригування прогнозу в режимі реального часу на основі власної бази даних та спорідненої інформації з інтернету	Розробка модулів інтеграції з обліковими системами підприємств.

Проведений ступеневий аналіз конкуренції на ринку зображено у таблиці 4.8.

Таблиця 4.8 - Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Вказати тип конкуренції - олігополія	Присутня невелика кількість фірм. Більшість ринку контролюють фірми-гіганти	Впровадження технологічних інновацій. Кооперація з дослідницькими центрами. Розширення функціоналу та задоволення потреб клієнтів.
2. За рівнем конкурентної боротьби - глобальний	Продукція не залежить від країни чи локалізації клієнта	
3. За галузевою ознакою внутрішньогалузева	Продукт спрямований на роздрібну торгівлю	
4. Конкуренція за видами товарів: - за бажанням	Полягає у випередженні задоволення бажань клієнта	
5. За характером конкурентних переваг - неціновa	Переваги передбачають собою ефективність та різноманіття функціоналу	
6. За інтенсивністю - не марочна	Торгова марка майже немає впливу	

Проведений аналіз конкуренції в галузі зображено у таблиці 4.9.

Таблиця 4.9 - Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти
	SNI, Leasoft, PAnYst	SAPPAN
Висновки	Контролюють велику частину ринку, мають узагальнені рішення	Спрямовані на малий бізнес, не мають локалізацій для більшості країн Європи

Фактори конкурентоспроможності та їх обґрунтування наведені в таблиці 4.10.

Таблиця 4.10 - Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	Інновації	Інноваційні рішення мають забезпечити перевагу нашим клієнтам над конкурентами
2	Функціонал	Функціонал повинен покривати вирішення необхідних задач клієнтів
3	Цінова політика	Вартість продукту відіграє велику роль при виборі системи клієнтом
4	Ресурсоємність	Великі затрати технічних ресурсів можуть спровокувати необхідність залучення додаткових коштів

Порівняльний аналіз сильних та слабких сторін проекту відображено у таблиці 4.11.

Таблиця 4.11 - Порівняльний аналіз сильних та слабких сторін RFS

№ п/п	Фактор конкурентоспроможності	ББали 1-20	Рейтинг товарів-конкурентів у порівнянні з RFS						
			–3	–2	–1	0	+1	+2	+3
1	Інновації	18				+			
2	Функціонал	12	+						
3	Цінова політика	16			+				
4	Ресурсоємність	3						+	

SWOT-аналіз проекту наведено в таблиці 4.12.

Таблиця 4.12 - SWOT-аналіз стартап-проекту

Сильні сторони: розумна цінова політика, функціонал забезпечує рішення більшості задач клієнта	Слабкі сторони: відсутність співпраці з інноваційними центрами,
Можливості: впровадження інноваційних рішень, оптимізація роботи продукту	Загрози: неточність результатів

Альтернативи ринкового впровадження проекту розглянуто в таблиці 4.13.

Таблиця 4.13 - Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Спеціалізовані рішення	Висока	1-3 місяців
2	Хмарний сервіс	Висока	3-6 місяців
3	Узагальнення рішення, вихід на нові сфери ринку	Середня	6-12 місяці

1.3 Розроблення ринкової стратегії проекту

Опис та вибір цільових груп потенційних клієнтів зображено в таблиці 4.14.

Таблиця 4.14 - Вибір цільових груп потенційних споживачів

№ п/п	Опис цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Малий бізнес	Абсолютна готовність в розгляді подібних рішень.	Високий попит	Середня	Вхід в сегмент складний
22	Середній бізнес	Середня готовність. В залежності від виду бізнесу, готовність різниться.	Середній попит	Вище середньої	Вхід в сегмент достатньо складний
33	Великий бізнес	Низький рівень, оскільки у великому бізнесі важче переналаштувати свій організаційний устрій.	Низький	Середня	Вхід в сегмент складний
Які цільові групи обрано: 1,2					

В таблиці 4.15 зображено вибір базової стратегії розвитку.

Таблиця 4.15 - Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
	Розробка та створення додаткових функціональних модулів	Таргетні пропозиції бізнесу, проведення презентації функціональних рішень на ярмарках та конференціях	Відсутність аналогічних до новостворених функціональних модулів у конкурентів	Розробка та удосконалення існуючих модулів на основі потреб ринку та інформації від клієнтів

В таблиці 4.16 наведено визначення базової стратегії конкурентної поведінки.

Таблиця 4.16 - Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1	Так	Можливі обидва варіанти	Стандартні функціональні модулі будуть виконувати схожі функції.	Унікальна цінова політика, функціональні інновації, сучасні технології

В таблиці 4.17 наведено визначення стратегії позиціонування.

Таблиця 4.17 - Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	Висока якість прогнозування в клієнтській сфері застосування	Розробка та удосконалення існуючих модулів на основі потреб ринку та інформації від клієнтів	Спеціалізовані рішення, хмарні сервіси	Прогнозування, передбачення, аналіз

1.4 Розроблення маркетингової програми стартап-проекту

В таблиці 4.18 представлені ключові переваги концепції потенційного товару.

Таблиця 4.18 - Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1	Широкий функціонал	Вирішення задач	Забезпечує вирішення більшої кількості задач бізнесу
2	Спеціалізовані рішення	Вирішення задач	Забезпечує більш ефективне вирішення задач у звуженій сфері застосування

3	Технічні ресурси	Хмарні сервіси	Дозволяє користуватись рішенням за рахунок віддалених технічних потужностей

Опис трьох рівнів моделі товару відображено у таблиці 4.19.

Таблиця 4.19 - Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
I. Товар за задумом	Обробка, аналіз даних. Прогнозування та передбачення потреб споживача		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	Швидкодія Ефективність Користувацький інтерфейс	Нм Нм Нм	Тх/Тл/Е Тх/Тл Е
	Якість: стандарти відповідні до законодавства. Створені функціональні скріпти.		
	Пакування: Власний сайт		
	Марка: CM Solutions		
III. Товар із підкріпленням	До продажу: застосунок для інтеграції в існуючі системи керування підприємством для прогнозування та передбачення потреб споживачів на основі великих масивів даних		
	Після продажу: Швидкодія, ефективність, легкість у користуванні		
Закритий код. Захищений від можливості декомпіляції.			

Визначення меж встановлення ціни показано в таблиці 4.20.

Таблиця 4.20 - Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	-	200$/міс	Рівень доходів підприємств надзвичайно високий	150-200$/міс

Формування системи збуту зображено в таблиці 4.21.

Таблиця 4.21 - Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Таргетні пропозиції для компаній	Презентації функціоналу	-	-

Концепція маркетингових комунікацій відображена у таблиці 4.22.

Таблиця 4.22 - Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Середній бізнес – оптимальні рішення за невисоку ціну	Соціальні мережі, внутрішньо ринкова комунікація	Прогнозування покупок споживача	Короткий опис переваг продукту, заохочення дізнатись більше	Передбачення покупок споживачів
2	Великий бізнес – повноцінні рішення для покращення продажів	Таргетні дзвінки до клієнтів	Прогнозування покупок споживача	Донести інформацію про оптимальність рішення для бізнесу клієнта	Передбачення покупок споживачів

2.5 Висновки до розділу

Отже, відповідно до вищенаведених результатів, можна стверджувати про наявність попиту на запропоновану систему. Варто зауважити, що присутня мала

конкуренція, оскільки рішення нове, тож інноваційна складова продукту дозволяє суттєво збільшити конкурентоспроможність проекту.

ВИСНОВКИ ПО РОБОТІ І РЕКОМЕНДАЦІЇ ДО ПОДАЛЬШИХ ДОСЛІДЖЕНЬ

У даній роботі було описано новий підхід у створенні системи рекомендацій. Окрім числових даних користувачів брались до уваги їх відгуки, тобто текстові дані. На розміченій вибірці була натренована модель аналізу тональності тексту. Загальний базис включає в себе підходи обробки природної мови, статистику TF-IDF та наївний байєсівський класифікатор. Для нерозміченої вибірки yelp про користувачів було до відгуків поставлено у відповіднсть клас позитивний чи негативний. Було створено алгоритм знаходження рекомендацій, значною перевагою якого є можливість бути розпаралелізованим, оскільки розроблений на технологіях великих даних та раціонально створений. Для реалізації он-лайн алгоритму, було збережено натреновану модель офлайн. У реальному часі класифікатор надає мітки вхідному тексту.

Продумана та створена архітектура додатку. Дані зберігаються у колекціях розподіленої NoSQL бази даних, для потокового алгоритму було налаштовано розподілена потокова платформа Kafka. Обидві одиниці архітектури є докерізованими (працюють як окремі контейнери програмного забезпечення). Обчислення та керування відбуваються у Spark – аналітичному рушію, встановленому локально. Модель була натренована з чіткою структурою Spark MLlib.

Отримана система має значну кількість застосувань в галузі, де присутні тексти та великі набори даних. Наступним етапом дослідження можливе використання нейронних мереж для аналізу тональності тексту.

ПЕРЕЛІК ПОСИЛАНЬ

1. Bobadilla J. Recommender systems survey / Bobadilla J., Ortega F., Hernando A., Gutierrez A. // Knowledge-Based Systems – 2013. – pp. 109-132.
2. Adomavicius G. Toward the next generation of recommender systems: A survey of state-of-the-art and possible extensions / Adomavicius G., Tuzhilin A. // IEEE Transactions on Knowledge and Data Engineering. – 2005. – 6p.
3. Воронцов К. В. Математические методы обучения по прецедентам [Електронний ресурс]. – 2009. – Режим доступу до ресурсу: http://www.machinelearning.ru/wiki/images/6/6d/Voron-ML-1.pdf
4. Kohavi R. A. study of cross-validation and bootstrap for accuracy estimation and model selection/ Kohavi R. A. // 14th International Joint Conference on Artificial Intelligence, Palais de Congres Montreal, Quebec, Canada. — 1995. — pp. 1137–1145.
5. Вапник В. Н. Восстановление зависимостей по эмпирическим данным / Вапник В. Н. — М.: Наука, 1979. - 448 с.
6. Шурыгин А. М. Прикладная стохастика: робастность, оценивание, прогноз / Шурыгин А. М. — М.: Финансы и статистика, 2000. – 224 с.
7. Hastie T. The Elements of Statistical Learning / Hastie T., Tibshirani R., Friedman J.— Springer, 2001. — 533 p.
8. Дрейпер Н. Прикладной регрессионный анализ / Дрейпер Н., Смит Г. – М.: Финансы и статистика, 1986. – 366 с.
9. Оссовський С. Нейронні мережі для обробки інформації / Станіслав Осовського. Пер. з польського І.Д. Рудинський. - М.: Фінанси і статистика, 2002. - 344 с.

10. Spark with the Dataset and DataFrame API [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://spark.apache.org

ДОДАТОК А

MAIN

```
package nl.rug.sc.app

import org.apache.spark.sql.expressions.Window
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._

object SparkLocalMain extends App {
  private val master = "local[*]" // local[*] means a local cluster with * being the amount of workers, *
= 1 worker per cpu core. Always have at least 2 workers (local[2])

  val sparkSession = SparkSession // Usually you only create one Spark Session in your
application, but for demo purpose we recreate them
    .builder()
    .appName("spark-bootcamp")
    .master(master)
    .config("spark.mongodb.input.uri", "mongodb://127.0.0.1:27017/OlyaDB.reviews")
    .config("spark.mongodb.output.uri", "mongodb://127.0.0.1:27017/OlyaDB.reviews")
    .getOrCreate()

  val sc = sparkSession.sparkContext

  val myMongo = new interactionWithMongoDB(sparkSession)

  // Loading Reviews dataset from mongodb for training the Sentiment Analysis model
  val dfReviewLab = myMongo.loadReviews()
  .withColumnRenamed("review", "text")



//Not using while developing, reading this result from Mongo
//  Main model
  val mySA = new getSAmodel(sparkSession)
//    1) get a model by  training
//    training dataframe
//    val model = mySA.getSentimentAnalysisModel(dfReviewLab)

//  2) get a model by loading the same saved model
  val model = mySA.loadSentimentAnalysisModel()

//KAFKA
    val myKafka = new kafkaStreaming(sparkSession)
    myKafka.listeningToTopic(model, myMongo)


  // Loading yelp_review dataset from mongodb
  val dfYelpReview  = myMongo.loadFromCollectionToDataFrame("yelp_review")
    .select("user_id", "business_id", "text")
```

```
  // Loading yelp_business dataset from mongodb
  val dfYelpBusiness = myMongo.loadFromCollectionToDataFrame("yelp_business")
    .select("city", "categories", "business_id", "name")


//  Apply model for our new dataset
//  val dfYelpReviewLabeled0= model.transform(dfYelpReview)


  //Creating a dataframe to work with and to save to mongoDB
  //Split one VectorType Column of probalities into two
//  val first = udf((v: org.apache.spark.ml.linalg.Vector) => v.toArray(0))
//  val second = udf((v: org.apache.spark.ml.linalg.Vector) => v.toArray(1))
//  val dfYelpReviewLabeled=dfYelpReviewLabeled0.select("business_id", "text", "user_id",
"prediction", "probability")
//    .withColumn("negProb", first(dfYelpReviewLabeled0("probability")))
//    .withColumn("posProb", second(dfYelpReviewLabeled0("probability")))
//    .drop("probability")

//    myMongo.saveDataFrameToMongo(df = dfYelpReviewLabeled, collectionName =
"yelp_predictions_new") //"yelp_predictions"


// While developing it is faster not to apply a model but just load previously saved result
  val dfYelpReviewLabeled = myMongo.loadFromCollectionToDataFrame("yelp_predictions_new")
//"yelp_predictions"

  val dfJoined = dfYelpReviewLabeled //dfYelpReviewLabeled
    .select("prediction", "posProb", "user_id", "business_id")
    .join(dfYelpBusiness,
      Seq("business_id"), "inner")


  //Find the  maximum reviewed city for each user (both positive and negative)
  val dfUserCity0= dfJoined
    .groupBy("user_id", "city")
    .count()

  val w_city = Window.partitionBy("user_id").orderBy(col("count").desc)

  val dfUserCity_2 = dfUserCity0
    .withColumn("rank", dense_rank().over(w_city))
    .select("user_id", "city")
    .where("rank <= 1")


  //Find the maximum positive reviewed category(ies) for each user (only positive)
 val dfCategory =  dfJoined
   .select(col("city"),
     col("business_id"),
     col("name"),
     col("categories"),
     col("user_id"),
     col("posProb"),
```

```
//  one row -> 3 rows OR the ListType column to multiple rows
     explode(col("categories")).as("category"))
   .drop("categories")

  val dfCategory2= dfCategory
    .filter(dfJoined("prediction")
      .equalTo(1.0) )
    .groupBy("user_id", "category")
    .count()

  val w = Window.partitionBy("user_id").orderBy(col("count").desc)

  val dfUserCat = dfCategory2
    .withColumn("rank", dense_rank().over(w))
    .select("user_id", "category").where("rank <= 2")
  //      .drop("rank")


//Joining for each user target cities and categories
  val dfUserCityCat = dfUserCat
    .join(dfUserCity_2, Seq("user_id"))

  //For each city and each category find top 1 (three) places based on all positive reviews
  dfCategory.select("city", "category", "business_id","name", "posProb")
    .createOrReplaceTempView("category")

  val dfCityCatBus = sparkSession.sql("select city, category, business_id, name, " +
    "percentile_approx(posProb, 0.5, 10000) as medianPosProb " +
    "from category " +
    "group by city, category, business_id, name")
    .withColumn("rank", rank().over(Window.partitionBy("city",
"category").orderBy(col("medianPosProb").desc)))
    .filter(col("rank") <= 1) //<=3
    .drop("rank", "medianPosProb")
    .groupBy("city", "category")
    //    3 rows -> 1 row, from multiple rows get ListType row
    .agg(concat_ws(", ", collect_list("name")).as("businesses"))


//RESULT
  val dfUser_Buss = dfUserCityCat
    .join(dfCityCatBus, Seq("category", "city"))
    .select("user_id", "city", "category", "businesses")
      .orderBy("user_id")

  dfUser_Buss.show(100, false)

//  myMongo.saveDataFrameToMongo(df = dfUser_Buss, collectionName =
"yelp_recommendation")


//  //KAFKA
//      val myKafka = new kafkaStreaming(sparkSession)
//      myKafka.listeningToTopic(model, myMongo)
```

```
  sparkSession.stop()
  println("\n Done")

        }
```

DB

```
package nl.rug.sc.app

import com.mongodb.spark.config.ReadConfig
import org.apache.spark.sql.{DataFrame, SparkSession}
import com.mongodb.spark._
import org.apache.spark.sql.functions.udf

class interactionWithMongoDB(sparkSession: SparkSession) {
  case class Review(id: String, review: String, category: String)
  val sc = sparkSession.sparkContext

  def loadFromCollectionToDataFrame(collectionName: String): DataFrame =
    sc
      .loadFromMongoDB(ReadConfig(Map("uri" -> ("mongodb://127.0.0.1:27017/OlyaDB."+
collectionName))))
      .toDF()

  def loadReviews(): DataFrame ={

    val dfReview = MongoSpark.load[Review](sparkSession)  // Uses the SparkSession

    //adding a field 'label' from 'category'
    //the rule: positive -> 1, negative -> 0
    val coder: (String => Int) = (arg: String) => {if (arg == "positive") 1 else 0}
    val sqlfunc = udf(coder)
    val dfReviewLab= dfReview.withColumn("label", sqlfunc( dfReview.col("category")))
    return dfReviewLab
  }

  def saveDataFrameToMongo(df: DataFrame, collectionName: String): Unit = {
    //save to mongo    uses the WriteConfig
    MongoSpark.save(df.write.option("collection", collectionName).mode("overwrite"))
  }

  def appendDataFrameToMongo(df: DataFrame, collectionName: String): Unit = {
    //append to mongo    uses the WriteConfig
    MongoSpark.save(df.write.option("collection", collectionName).mode("append"))
  }


}
```

SA MODEL

```
package nl.rug.sc.app

import org.apache.spark.ml.classification.{NaiveBayes, NaiveBayesModel}
```

```
import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator
import org.apache.spark.ml.feature.{HashingTF, IDF, StopWordsRemover, Tokenizer}
import org.apache.spark.sql.{DataFrame, SparkSession}
import org.apache.spark.ml.Pipeline
import org.apache.spark.ml.tuning.CrossValidatorModel

class getSAmodel(sparkSession: SparkSession) {
  private val pathToModel =
"/Users/galinamylyava/Documents/RUG/ScaCom/2018_group_3_s3605361_s3605345/src/main/sc
ala/nl/rug/sc/app/SAmodel"
  private val numFeatures = 500


  def getSentimentAnalysisModel(dfTrain: DataFrame): CrossValidatorModel = {

//    val rescaledData = getfeatures(df, "review")

    val tokenizer = new Tokenizer().setInputCol("text").setOutputCol("words")
//    val dfwords = tokenizer.transform(df)
    //wordsData.show(5)

    val remover = new StopWordsRemover()
      .setInputCol("words")
      .setOutputCol("wordsfiltered")

//    Maps a sequence of terms to their term frequencies using the hashing trick.
    val hashingTF = new HashingTF()
      .setInputCol("wordsfiltered").setOutputCol("rawFeatures").setNumFeatures(numFeatures)


    val idf = new IDF().setInputCol("rawFeatures").setOutputCol("features")

    val NB = new NaiveBayes()
//      .setThresholds(Array(0.1, 0.8))

    val pipeline = new Pipeline().setStages(Array(tokenizer, remover, hashingTF, idf, NB))

    // Split data into training (60%) and test (40%).
    val splits = dfTrain.randomSplit(Array(0.8, 0.2), seed = 11L)
    val training = splits(0)
    val test = splits(1)

    val model = pipeline.fit(training)

    val trainPredictions = model.transform(training)
    val testPredictions = model.transform(test)

    import org.apache.spark.ml.evaluation.BinaryClassificationEvaluator
    val evaluator = new BinaryClassificationEvaluator().setMetricName("areaUnderROC")

    import org.apache.spark.ml.param.ParamMap
    val evaluatorParams = ParamMap(evaluator.metricName -> "areaUnderROC")

    val areaTrain = evaluator.evaluate(trainPredictions, evaluatorParams)
//    print("areaTrain: ", areaTrain)
```

```
    val areaTest = evaluator.evaluate(testPredictions, evaluatorParams)
//    print("areaTest: ", areaTest)

    // That gives all the combinations of the parameters
    import org.apache.spark.ml.tuning.ParamGridBuilder
    val paramGrid = new ParamGridBuilder()
      .addGrid(hashingTF.numFeatures, Array(20, 1000))
//        .addGrid(NaiveBayes.thresholds, )
      .build


    import org.apache.spark.ml.tuning.CrossValidator
    import org.apache.spark.ml.param._
    val cv = new CrossValidator()
      .setEstimator(pipeline)
      .setEstimatorParamMaps(paramGrid)
      .setEvaluator(evaluator)
      .setNumFolds(10)

    val cvModel = cv.fit(training)
    val cvPredictions = cvModel.transform(test)

    val theBestArea = evaluator.evaluate(cvPredictions, evaluatorParams)

//    val bestModel = cvModel.bestModel
    //save the pipeline
    cvModel.write.overwrite.save(pathToModel)

    return cvModel

  }

    def loadSentimentAnalysisModel(): CrossValidatorModel = {

        val sameCVModel = CrossValidatorModel.load(pathToModel)
        sameCVModel
    }

          }
```

STREAMING

```
package nl.rug.sc.app

import org.apache.spark.streaming.kafka010.{ConsumerStrategies, KafkaUtils, LocationStrategies}
import org.apache.spark.streaming.{Seconds, StreamingContext}
import org.apache.spark.ml.tuning.CrossValidatorModel
import org.apache.kafka.common.serialization.StringDeserializer
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._


class kafkaStreaming (sparkSession: SparkSession){
```

```
  //  Created for ssc: StreamingContext
  private val batchDuration = Seconds(10)

  var sc = sparkSession.sparkContext

  def listeningToTopic(model : CrossValidatorModel, myMongo: interactionWithMongoDB): Unit = {
    val ssc = new StreamingContext(sc, batchDuration)

    val kafkaParams = Map("bootstrap.servers" -> "localhost:9092",
      "key.deserializer" -> classOf[StringDeserializer],
      "value.deserializer" -> classOf[StringDeserializer],
      "group.id" -> "use_a_separate_group_id_for_each_stream")
    //  Names of topics
    val topics = List("test").toSet

    val messages = KafkaUtils.createDirectStream[String, String](
        ssc,
        LocationStrategies.PreferConsistent,
        ConsumerStrategies.Subscribe[String, String](topics, kafkaParams))

    val lines = messages.map(_.value)

    lines.foreachRDD { rdd =>
      import sparkSession.implicits._
      val df = rdd.toDF()

      val dfToModel = df.withColumn("user_id", split(col("value"), "\\|").getItem(0))
        .withColumn("business_id", split(col("value"), "\\|").getItem(1))
        .withColumn("text", split(col("value"), "\\|").getItem(2))
        .drop("value")
//        .show()

      val dfLabeled = model.transform(dfToModel)
        .select("user_id", "business_id", "text", "prediction")

      dfLabeled.show()

      //append to mongo
      myMongo.appendDataFrameToMongo(dfLabeled, "streaming_sa_model_result")
    }

    // Start the computation
    ssc.start()
    ssc.awaitTermination()

  }

}
```

KAFKA

```
import time
from kafka import KafkaProducer

producer = KafkaProducer(bootstrap_servers='localhost:9092',
                         # key_serializer=str.encode,
```

```
                    value_serializer=str.encode)

topic_name = "test"
print("sending messages to topic: [%s]" % topic_name)

while 1:
    with open('push_to_kafka.txt') as f:
        for line in f:
            print(line)
            producer.send('test', line)
            time.sleep(1)

producer.flush()
package nl.rug.sc

import org.apache.spark.sql.{Row, SparkSession}
import org.apache.spark.sql.types.{IntegerType, StructField, StructType}
import org.apache.spark.{SparkConf, SparkContext}

case class Person(id: Int, name: String, grade: Double) // For the advanced data set example, has
to be defined outside the scope

class SparkExample(sparkSession: SparkSession, pathToCsv: String) {
  private val sparkContext = sparkSession.sparkContext

  /**
    * An example using RDD's, try to avoid RDD's
    */
  def rddExample(): Unit = {
    val data = List(1, 2, 3, 4, 5)
    val rdd = sparkContext.parallelize(data)


    rdd
      .map(x => x + 1) // Increase all numbers by one (apply the transformation x => x + 1 to every
item in the set)
      .collect() // Collect the data (send all data to the driver)
      .foreach(println) // Print each item in the list

    printContinueMessage()
  }

  /**
    * An example using Data Frames, improvement over RDD but Data Sets are preferred
    */
  def dataFrameExample(): Unit = {
    import sparkSession.implicits._ // Data in dataframes must be encoded (serialized), these
implicits give support for primitive types and Case Classes
    import scala.collection.JavaConverters._

    val schema = StructType(List(
      StructField("number", IntegerType, true)
    ))

    val dataRow = List(1, 2, 3, 4, 5)
```

```
      .map(Row(_))
      .asJava

    val dataFrame = sparkSession.createDataFrame(dataRow, schema)

    dataFrame
      .select("number")
      .map(row => row.getAs[Int]("number")) // Dataframe only has the concept of Row, we need to
extract the column "number" and convert it to an Int
      .map(_ + 1) // Different way of writing x => x + 1
      .collect()
      .foreach(println)

    dataFrame.printSchema() // Data frames and data sets have schemas

    printContinueMessage()
  }

  /**
    * An example using Data Sets, improvement over both RDD and Data Frames
    */
  def dataSetExample(): Unit = {
    import sparkSession.implicits._ // Data in datasets must be encoded (serialized), these implicits
give support for primitive types and Case Classes

    val dataSet = sparkSession.createDataset(List(1, 2, 3, 4, 5))

    dataSet
      .map(_ + 1) // Different way of writing x => x + 1
      .collect()
      .foreach(println)

    dataSet.printSchema()

    printContinueMessage()
  }

  /**
    * Advanced data set example using Scala's Case Classes for more complex data, note that the
Case Class is defined at the top of this file.
    */
  def dataSetAdvancedExample(): Unit = {
    import sparkSession.implicits._

    val dataSet = sparkSession.createDataset(List(
      Person(1, "Alice", 5.5),
      Person(2, "Bob", 8.6),
      Person(3, "Eve", 10.0)
    ))

    dataSet
      .show() // Shows the table

    printContinueMessage()
```

```
    dataSet
      .map(person => person.grade) // We transform the Person(int, string, double) to a double
(Person => Double), extracting the person's grade
      .collect() // Collect in case you want to do something with the data
      .foreach(println)

    printContinueMessage()

    // Even cleaner is
    dataSet
      .select("grade")
      .show()

    dataSet.printSchema()

    printContinueMessage()
  }

  /**
    * In your case, you will be reading your data from a database, or (csv, json) file, instead of
creating the data in code as we have previously done.
    * We use a CSV containing 3200 US cities as an example data set.
    */
  def dataSetRealisticExample(): Unit = {
    val dataSet = sparkSession.read
      .option("header", "true") // First line in the csv is the header, will be used to name columns
      .option("inferSchema", "true") // Infers the data types (primitives), otherwise the schema will
consists of Strings only
      .csv(pathToCsv) // Loads the data from the resources folder in src/main/resources, can also be
a path on your storage device

    dataSet.show() // Show first 20 results

    printContinueMessage()

    dataSet
      .sort("name") // Sort alphabetically
      .show()

    printContinueMessage()

    import sparkSession.implicits._ // For the $-sign notation

    dataSet
      .filter($"pop" < 10000) // Filter on column 'pop' such that only cities with less than 10k
population remain
      .sort($"lat") // Sort remaining cities by latitude, can also use $-sign notation here
      .show()

    printContinueMessage()

    dataSet
      .withColumn("name_first_letter", $"name".substr(0,1)) // Create a new column which contains
the first letter of the name of the city
      .groupBy($"name_first_letter") // Group all items based on the first letter
```

```
      .count() // Count the occurrences per group
      .sort($"name_first_letter") // Sort alphabetically
      .show(26) // Returns the number of cities in the US for each letter of the alphabet, shows the
first 26 results

    printContinueMessage()

    import org.apache.spark.sql.functions._ // For the round  (...) functionality

    dataSet
      .withColumn("pop_10k", (round($"pop" / 10000) * 10000).cast(IntegerType)) // Create a column
which rounds the population to the nearest 10k
      .groupBy("pop_10k") // Group by the rounded population
      .count() // Count the occurences
      .sort("pop_10k") // Sort from low to high
      .show(100) // Returns the number of cities in 10k intervals

    dataSet.printSchema()

    printContinueMessage()
  }

  private def printContinueMessage(): Unit = {
    println("Check your Spark web UI at http://localhost:4040 and press Enter to continue. [Press
Backspace and Enter again if pressing enter once does not work]")
    scala.io.StdIn.readLine()
    println("Continuing, please wait....")
  }
}
```