ВІННИЦЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ МИХАЙЛА КОЦЮБИНСЬКОГО

Факультет математики, фізики, комп'ютерних наук і технологій

Кафедра математики та інформатики

Д И П Л О М Н А Р О Б О Т А

на тему: «Проектування інтелектуальних інформаційних систем»

Студента 2 курсу МСОІ групи
Освітньої програми Середня освіта. Інформатика, математика.
Спеціальності 014 Середня освіта (Інформатика)
Галузі знань 01 Освіта/Педагогіка
Ступеня вищої освіти Магістр
Бойчука Дмитра Юрійовича

Науковий керівник Клочко Оксана Віталіївна
доктор педагогічних наук, професор

Розширена шкала _____

Кількість балів: _____ Оцінка: ECTS _____

Голова комісії _____ _____
(підпис) (ініціали, прізвище)

Члени комісії _____ _____
(підпис) (ініціали, прізвище)
_____ _____
(підпис) (ініціали, прізвище)
_____ _____
(підпис) (ініціали, прізвище)
_____ _____
(підпис) (ініціали, прізвище)

м. Вінниця – 2019 рік

ЗМІСТ

ВСТУП

Мета дослідження – полягає у теоретичному обґрунтуванні та проектуванні інтелектуальних інформаційних систем.

Об'єктом дослідження є процеси проектування інтелектуальних інформаційних систем.

Предмет дослідження є інформаційні технології проектування інтелектуальних інформаційних систем.

Відповідно до об'єкта, предмета і мети визначено **завдання дослідження**:

1. З'ясувати науково-теоретичний і практичний стан проблеми проектування інтелектуальних інформаційних систем.

2. Розробити модель інтелектуальної інформаційної системи.

3. Спроектувати інтелектуальну інформаційну систему.

4. Реалізувати проект інтелектуальної інформаційної системи.

5. Визначити ефективність функціонування спроектованої інтелектуальної інформаційної системи.

Актуальність дослідження підтверджується сферою застосування інтелектуальних інформаційних систем. Через експоненційне збільшення розмірів даних алгоритми машинного навчання отримують все більше застосування у різних сферах нашого життя на сьогоднішній день такий підхід дозволяє заощадити фінансові ресурси та відмовитись від людської участі в багатьох процесах. За допомогою інтелектуальних інформаційних систем можна вирішувати такі завдання:

- Прогнозування: попиту, обсягу продажів, наповнення складу, завантаження устаткування і інших ресурсів, подальшого розвитку підприємства і т.п.
- Виявлення: тенденцій, прихованих взаємозв'язків, аномалій, повторюваних елементів і т.п.
- Розпізнавання: фото-, відео-, аудіоконтенту, спроб шахрайства, брехні, внутрішніх загроз, зовнішніх атак на систему безпеки і т.п.
- Автоматизація: роботи операторів в онлайн-чатах, телефонних

операторів і т.п.

- Класифікація: аналіз складу покупців, клієнтів, замовників і сегментація їх за різними параметрами.
- Кластеризація: класифікація за параметрами, які з самого початку не були відомі.
- Розробка: питально-відповідальних систем та чат-ботів.

Роботу було *апробовано* під час виступу на Всеукраїнському методологічному семінарі для молодих учених «Інформаційно-комунікаційні технології в освіті та наукових дослідженнях» 06 грудня 2018 р. в Інститут інформаційних технологій і засобів навчання НАПН України (Додаток А) та участі в конкурсі студентських наукових робіт «Інформаційно-комунікаційні технології в освіті».

Публікації. Результати дослідження викладено в опублікованій праці, статті.

1. Бойчук Д., Клочко О. Проектування SMART-середовища навчальної дисципліни на основі штучних нейронних мереж / Д. Бойчук, О. Клочко // Збірник матеріалів VI Всеукраїнської науково-практичної конференції молодих учених «Наукова молодь-2018» (16 листопада 2018 р., м. Київ) [Електронний ресурс] / за ред. Спіріна О.М. та Яцишин А.В. – К.: ІІТЗН НАПН України, 2018. – с. 122-125. – Режим доступу до ресурсу: http://lib.iitta.gov.ua/715444/.

Зв'язок роботи з науковими програмами, планами, темами. Дослідження пов'язане з темами науково-дослідних робіт Вінницького Державного педагогічного університету імені Михайла Коцюбинського, у виконані яких брав активну участь автор даного дослідження:

1) Методичні основи розроблення SMART-комплексів для повідомлення кваліфікованих робітників аграрної, будівельної та машинобудівної галузі (Державний реєстраційний номер 0118U003223);
2) Наукова співпраця з підприємством «ABI-Solutions» (м. Лугано, кантон Тічино, Швейцарська Конфедерація).

З метою розв’язування поставлених завдань використано *методи дослідження:* аналіз робіт в галузі інтелектуальних інформаційних систем; аналіз математичних основ проектування та розробки інтелектуальних інформаційних систем; спостереження, тестування; а також синтез, класифікація, порівняння, системний аналіз, абстрагування, моделювання та ін.

Наукова новизна одержаних результатів полягає у тому, що: обґрунтовано, спроектовано й розроблено інтелектуальну інформаційну систему, проаналізовано математичні основи проектування та розробки інтелектуальних інформаційних систем.

Практичне значення результатів дослідження полягає у тому, що: розглянуті в ньому положення дають можливість визначити підходи до вирішення низки теоретичних і практичних проблем, пов’язаних з розробкою інтелектуальних інформаційних систем. Сформульовані у дослідженні як теоретичні, так і практичні результати, висновки, пропозиції та рекомендації можуть бути використані: у науково-дослідних цілях – для подальших наукових розробок у зазначеній сфері; у процесі розробки проектів створення інтелектуальних інформаційних систем.

Проведене дослідження має *практичну цінність* не лише у можливості застосування описаного підходу проектування інтелектуальної інформаційної системи до задач класифікації текстових даних, а й для розробки інтелектуальних інформаційних систем спрямованих для розв’язання інших завдань

Структура і обсяг дослідження. Робота містить вступ, три розділи, висновки до кожного розділу, загальні висновки, список використаних джерел і додатки. Сторінок 89, додатків 3, рисунків 13.

РОЗДІЛ 1. ТЕОРІЯ І ПРАКТИКА ПРОЕКТУВАННЯ ІНТЕЛЕКТУАЛЬНИХ ІНФОРМАЦІЙНИХ СИСТЕМ

1.1. Аналіз дефініцій категоріального апарату дослідження

Інтелектуальні інформаційні технології - це одна з найбільш перспективних та прогресивних наукових та прикладних областей інформатики, що робить істотний вплив на усі наукові і технологічні напрямки, які пов'язані з використанням комп'ютерів, й вже сьогодні дає суспільству те, що воно очікує від науки та зокрема інформатики. Практично значущі результати, більшість з яких сприяють якісним змінам в сферах їх застосування. Завданнями інтелектуальних інформаційних технологій є, розширення кола завдань, які вирішуються за допомогою комп'ютерів, підвищення рівня інтелектуальної інформаційної підтримки сучасного фахівця.

Переробка інформації – найважливіша функція, без якої неможлива цілеспрямована діяльність будь-якої соціально-економічної, організаційно-виробничої системи (підприємства, організації, і т.п.). Систему, що реалізує функції збору, зберігання, обробки і передачі інформації, називають інформаційною системою (ІС). Найважливішими функціями цих системам є прогнозування, планування, облік, аналіз, контроль і регулювання. Технології виконання таких функцій називаються інформаційними технологіями. Найбільш широко інформаційні системи та технології використовуються у виробничій, управлінській і фінансовій діяльності, хоча почалися зсуви у свідомості людей, зайнятих в інших сферах, щодо необхідності впровадження й активного застосування таких систем і технологій. Робота інформаційної системи пов'язана з обміном інформацією між її компонентами, а також системи з навколишнім середовищем. У процесі роботи персонал одержує відомості про стан системи в кожний момент часу, про досягнення (або не досягненні) заданої мети для того, щоб впливати на систему й забезпечити виконання управлінських рішень. Інформаційна система – це сукупність внутрішніх і зовнішніх потоків прямого і зворотного інформаційного зв'язку певних об'єктів, засобів, фахівців, що беруть участь у процесі обробки інформації та виробленні управлінських рішень.

Практично всі інформаційні системи в наш час засновані на основі засобів автоматики й обчислювальної техніки (комп'ютерів).

Автоматизована інформаційна система – сукупність інформації, економіко-математичних методів і моделей, технічних, програмних, технологічних засобів і фахівців, призначених для автоматизованої обробки інформації та прийняття управлінських рішень.

Згідно із прогнозами багатьох політологів, на сучасному етапі розвитку цивілізації відбувається поступовий перехід від індустріального до постіндустріального або інформаційного суспільства. Одне з його визначень таке: інформаційне суспільство – теоретична концепція постіндустріального суспільства; історична фаза можливого розвитку цивілізації, у якій основними продуктами виробництва є інформація й знання. Головні риси такого суспільства:

– збільшення ролі інформації та знань;

– зростання числа людей, зайнятих інформаційними комунікаціями та виробництвом інформаційних продуктів і послуг;

– створення глобального інформаційного простору, що забезпечує ефективну інформаційну взаємодію людей, їхній доступ до світових інформаційних ресурсів і задоволення їхніх потреб в інформаційних продуктах і послугах.

Термін «інформаційне суспільство» і проекти створення такого суспільства вперше з'явилися в останній третині XX століття в США та країнах Західної Європи. Відтоді поняття інформаційного суспільства міцно зайняло своє місце не тільки в лексиконі фахівців з інформатики, але й у мові політичних діячів, економістів, вчених інших спеціальностей. Як правило, це поняття асоціюється з розвитком інформаційних технологій і засобів телекомунікацій, що дозволяють на базі громадянського суспільства (або, принаймні, декларованих його принципів) здійснити новий еволюційний стрибок, щоб ввести цивілізацію в XXI століття вже у вигляді інформаційного суспільства або, принаймні, його початкового етапу. Згідно з вищевказаним, 27 березня 2006 року генеральна Асамблея ООН прийняла резолюцію під номером A/RES/60/252, що проголошує 17 травня Міжнародним

днем інформаційного суспільства. З огляду на глибину й розмах технологічних і соціальних наслідків комп'ютеризації і цифрового зв'язку в різних сферах громадського життя, процеси інформатизації нерідко називають комп'ютерною або інформаційною революцією. Більше того, західна суспільно-політична думка висунула різні варіанти концепції інформаційного суспільства, що мають своєю метою пояснити явища, породжені новим етапом науково-технічного прогресу. Про значимість і зростаючу популярність цієї концепції на Заході свідчить наростаючий потік публікацій на цю тему. В наш час у громадській думці концепція інформаційного суспільства поступово висувається на те ж саме місце, яке в 1970-х роках займала теорія постіндустріального суспільства. Необхідно відзначити, що ряд західних політологів й економістів схиляються до того, щоб провести різку грань, що відокремлює концепцію інформаційного суспільства від пост індустріалізму. Однак, хоча концепція інформаційного суспільства покликана замінити теорію постіндустріального суспільства 1970-х років, її прихильники повторюють і далі розвивають ряд найважливіших положень технократизму, пост індустріалізму та традиційної футурології. Характерно, що ряд провідних дослідників, що сформулювали теорію постіндустріального суспільства, у наш час виступають як прихильники концепції інформаційного суспільства. Концепція інформаційного суспільства стає своєрідним новим етапом розвитку теорії постіндустріального суспільства. Ключовим елементом інформаційного суспільства є системи зберігання, обробки й передачі цифрової інформації (у тому числі телекомунікації). Не випадковим є той факт, що інформаційне суспільство затвердилося спочатку в тих країнах (Японія, США, Західна Європа), у яких в 1960-70-х роках сформувалися розвинені телекомунікаційні мережі. Говорячи про інформаційне суспільство, його варто розуміти не в буквальному сенсі, а розглядати як орієнтир, тенденцію змін у сучасній цивілізації. У цілому ця модель орієнтована на майбутнє, але в розвинених капіталістичних країнах уже зараз можна назвати цілий ряд викликаних інформаційною технологією змін, які підтверджують концепцію інформаційного суспільства:

– структурні зміни в економіці, особливо в сфері розподілу робочої сили;

- зросле усвідомлення важливості інформації;
- зростаюче розуміння необхідності комп'ютерної грамотності;
- широке поширення інформаційних технологій;
- підтримка урядом розвитку комп'ютерної мікро та наноелектронної технології й телекомунікацій.

Якісно новим моментом є можливість керування більшими комплексами організацій і виробничих систем, що вимагає координації діяльності сотень тисяч і навіть мільйонів людей. Іде бурхливий розвиток нових наукових напрямків, таких як інформаційна теорія, кібернетика, теорія прийняття рішень, теорія ігор і т.д., тобто напрямків, пов'язаних саме з проблемами організаційних множин. Говорячи про зміни й зрушення, що сприяють переходу сучасного суспільства в якісно нову стадію, прихильники розглянутої концепції (інформаційного суспільства) опираються на об'єктивні процеси розвитку наукомістких, енергозберігаючих і працезберігаючих галузей економіки, процеси роботизації виробництва, комп'ютеризації й інформатизації найважливіших сфер суспільного й політичного життя. Дійсно, у цей час від новітніх наукомістких й енергозберігаючих компонентів техніки залежить вирішення таких життєве важливих питань, як економічне зростання, зайнятість, підвищення життєвого рівня і т.д. Вони торкаються основних принципів функціонування й життєдіяльності сучасного суспільства, порушуючи кардинальні питання щодо соціальних і політичних змін, які несе із собою впровадження інформаційної технології. Це впливає на перспективу суспільно-історичного розвитку людства, на долю людини, його місце й роль у цьому процесі. Інформатизація й комп'ютеризація вимагають від людей нових навичок, нових знань і нового мислення, покликаних полегшити адаптацію людини до умов і реалій комп'ютеризованого суспільства й забезпечити йому гідне місце в цьому суспільстві. Тому не можна не погодитися з тим, що інформатизація впливає на образ і якість життя всіх членів суспільства, як на індивідуальному, так і на організаційному рівні, на робочому місці та у побуті. Добре це чи погано, але вона є силою, що не просто трансформує життя цілих співтовариств, але сприяє перебудові самого процесу відносин між людьми.

У наш час є безліч різних тлумачень терміна «інформація». Тому наведемо кілька визначень даному поняттю.

Інформація (від латинського informatio – роз'яснення, виклад) – відомості, передані людьми усним, письмовим або іншим способом (за допомогою умовних сигналів, технічних засобів і т.д.). Із середини XX століття – загальнонаукове поняття, що включає обмін відомостями між людьми, людиною й автоматом, автоматом та автоматом, обмін сигналами у тваринному й рослинному світі, передачу ознак від клітки до клітки, від організму до організму; одне з основних понять кібернетики. Інформація, одна з основних універсальних властивостей предметів, явищ, процесів об'єктивної дійсності, людини, що полягає в здатності сприймати внутрішній стан і впливи навколишнього середовища й зберігати певний час результати, перетворювати отримані відомості й передавати результати обробки (перетворення) іншим предметам, явищам, процесам, машинам, людям. Інформація (для процесу обробки даних), будь-які знання про предмети, факти, поняття і т.п. проблемного середовища, якими обмінюються користувачі системи обробки даних. Як видно, інформація одержала настільки широке поширення в житті цивілізації, що для неї недостатньо одного визначення. Аналогічна ситуація має місце із другим ключовим поняттям – «система». В інформатиці цей термін широко розповсюджений і має безліч змістових значень. Найчастіше він використається стосовно набору технічних засобів і програм. Системою може називатися апаратна частина комп'ютера або безліч програм, призначених для вирішення конкретних прикладних завдань, доповнених процедурами ведення документації й керування розрахунками. Система (від грецького systema – ціле, складене із частин; з'єднання), безліч елементів, що перебувають у відносинах і зв'язках один з одним, що утворять певну цілісність, єдність. Виділяють матеріальні й абстрактні системи. Перші розділяються на системи неорганічної природи (фізичні, геологічні, хімічні, технічні й ін.) і живі системи (біологічні системи – клітини, тканини, організми, популяції, види, екосистеми); особливий клас систем – соціальні системи (від найпростіших соціальних об'єднань до соціальної структури суспільства). Абстрактні системи – поняття, гіпотези, теорії,

наукові знання про систему, лінгвістичні (мовні), формалізовані, логічні системи та ін. У сучасній науці дослідження систем різного роду проводиться в рамках системного підходу, загальної теорії системи, різних спеціальних теорій систем, у кібернетиці, системотехніці, системному аналізі і т.д. У державних стандартах два розглянутих терміни поєднуються в єдине поняття – «інформаційна система».

Інформаційна система, система, що аналізує пам'ять і маніпулює інформацією про проблемну область.

Технологія (від грецького techne – мистецтво, майстерність, уміння), сукупність методів, способів і прийомів одержання, обробки або переробки сировини й напівфабрикатів з метою отримання готової продукції; наукова дисципліна, що вивчає механічні, фізичні, хімічні й інші зв'язки й закономірності, що діють у технологічних процесах. Іноді технологією називають також самі заходи щодо видобутку, обробки, транспортування, зберігання, контролю, що є частиною загального виробничого процесу.

Інтелектуальна інформаційна технологія, прийоми, способи й методи виконання функцій збору, зберігання, обробки, передачі й використання знань.

Інтелектуальна інформаційна система (ІІС) — це один з видів автоматизованих інформаційних систем, інколи ІІС називають системою, засновану на знаннях. ІІС є комплексом програмних, лінгвістичних і логіко-математичних засобів для реалізації основного завдання: здійснення підтримки діяльності людини і пошуку інформації в режимі розширеного діалогу на природній мові.

Класифікація ІІС:

– експертні системи;

– власне експертні системи;

– інтерактивні банери;

– запитально-відповідальна система;

– інтелектуальні пошукові системи.

Експертною системою (ЕС) називають систему підтримки прийняття рішень, яка містить знання з певної вузької предметної області, а також може пропонувати

користувачу рішення проблем з цієї галузі і обґрунтовувати їх. Експертна система складається з бази знань, механізму логічного виводу і підсистеми обґрунтувань.

Експертна система акумулює професійні знання керівників і фахівців, використовуючи їх для формування бази знань, яка містить набір взаємопов'язаних правил. При прийнятті рішень стає можливим аналіз наслідків різних рішень у вигляді питань "що буде, якщо...", не витрачаючи часу на трудомісткий процес програмування.

Створення експертних систем - це спроба значного розширення області застосування комп'ютерної техніки і суттєвого збільшення її можливостей як допомоги людині у її інтелектуальній роботі.

Забезпечення роботи ІІС:

– Математичне;

– Лінгвістичне;

– Програмне;

– Технічне;

– Технологічне;

– Кадрове.

Дані, інформація, оформлена у формалізованому вигляді, зручному для пересилання, інтерпретації або обробки за участю людини або автоматичних засобів.

База даних, сукупність взаємозалежних даних, організованих згідно зі схемою бази даних так, щоб з ними міг працювати користувач.

Система управління базами даних, сукупність програмних та мовних засобів, які забезпечують керування базами даних.

Знання, сукупність фактів, закономірностей, відносин і евристичних правил, що відображає рівень поінформованості про проблеми деякої предметної області.

База знань, упорядкована сукупність правил, фактів, механізмів виводу й програмних засобів, що описує деяку предметну область і призначена для роботи з накопиченими в ній знаннями.

Система управління базою знань, сукупність програмних та апаратних засобів для організації й ведення бази знань.

Процеси, що забезпечують роботу інформаційної системи будь-якого призначення, складаються з таких основних етапів:

– введення інформації із зовнішніх або внутрішніх джерел;

– обробка вхідної інформації й структуризація її в зручному виді;

– вивід інформації для передачі споживачам або в іншу систему;

– повторне введення або корекція незадовільно зібраної вхідної інформації. Інформаційна система визначається наступними властивостями:

– будь-яка інформаційна система може бути піддана аналізу, побудована й керована на основі загальних принципів побудови систем;

– інформаційна система є динамічною й такою, що розвивається;

– при побудові інформаційної системи необхідно використати системний підхід;

– вихідною продукцією інформаційної системи є інформація, на основі якої приймаються рішення;

– інформаційну систему варто сприймати як змішану (людино- комп'ютерну) систему обробки інформації.

У наш час переважна більшість інформаційних систем реалізується за допомогою обчислювальної техніки. Однак у загальному випадку інформаційну систему можна побудувати й у нецифровому варіанті. Щоб розібратися в роботі інформаційної системи, необхідно зрозуміти суть проблем, які вона вирішує, а також організаційні процеси, у які вона включена. Так, наприклад, при визначенні можливості комп'ютерної інформаційної системи для підтримки прийняття рішень варто враховувати:

– структурованість розв'язуваних управлінських завдань;

– рівень ієрархії керування організацією, на якому рішення повинне бути прийняте;

– приналежність розв'язуваного завдання до тієї або іншої функціональної сфери бізнесу;

– вид використовуваної інформаційної технології.

Технологія роботи в комп'ютерній інформаційній системі в загальному випадку доступна для розуміння фахівцем некомп'ютерної області й може бути успішно використана для контролю процесів професійної діяльності й керування цими процесами. Розрізняють три типи задач, для яких створюються інформаційні системи:

– структуровані (такі, що формалізуються);

– неструктуровані (такі, що не формалізуються);

– частково структуровані.

Структурована (така, що формалізуються) задача – це задача, для якої відомі всі її елементи та взаємозв'язки між ними. Неструктурована (така, що не формалізуються) задача – для якої неможливо виділити елементи та установити між ними зв'язки. У структурованій задачі вдається виразити її зміст у формі математичної моделі, що має точний алгоритм рішення. Подібні задачі звичайно доводиться вирішувати багаторазово, і вони носять рутинний характер. Метою використання інформаційної системи для рішення структурованих задач є повна автоматизація їхнього рішення, тобто зведення ролі людини до нуля.

1.2. Концептуальні основи проектування інтелектуальних інформаційних систем

Характерною властивістю сьогодення є процеси інформатизації, що інтенсивно розвиваються та впроваджуються практично у всіх сферах людської діяльності. Вони призвели до формування нової інформаційної інфраструктури, яка пов’язана з новим типом суспільних відносин та використанням нових інформаційних технологій. Одним із напрямків розвитку штучного інтелекту є розробка комп’ютерних інтелектуальних систем, здатних виконувати функції, що традиційно вважаються інтелектуальними, – розуміння мови, логічний висновок, використання накопичених знань, навчання, розпізнавання образів, а також навчатися і пояснювати свої рішення. На сьогодні інтелектуальні інформаційні системи є перспективними у своєму розвитку.

До основних понять, які використовуються у штучному інтелекті, відносять поняття штучного інтелекту, інтелекту, алгоритму, інтелектуальної задачі. Наведемо їх визначення.

Інтелектом ми будемо вважати здатність мозку вирішувати інтелектуальні задачі шляхом придбання, запам'ятовування та цілеспрямованого перетворення знань у процесі навчання, отримання життєвого досвіду та адаптації до різноманітних зовнішніх та внутрішніх обставин. У цьому визначенні «інтелекту» під терміном «знання» ми маємо на увазі не тільки ту інформацію, яка поступає в мозок через органи чуття; зазначена інформація важлива, але недостатня для інтелектуальної діяльності. Річ у тому, що об'єкти навколишнього середовища володіють властивістю не тільки впливати на органи чуття, але і знаходитися один з одним в певних відносинах. Ясно, що для того, щоб здійснювати в навколишньому середовищі інтелектуальну діяльність (або хоч би просто існувати), необхідно мати знання щодо моделі цього світу. У цій інформаційній моделі навколишнього середовища реальні об'єкти, їх властивості та відносини між ними не тільки відображаються і запам'ятовуються, але і можуть подумки цілеспрямовано перетворюватися. При цьому істотне те, що формування моделі зовнішнього середовища відбувається «у процесі навчання, отримання життєвого досвіду та адаптації до різноманітних обставин». Отже, інтелектом вважається спроможність мозку до мисленнєвої діяльності, тобто до оперування знаннями для прийняття певних рішень стосовно конкретної задачі.

Штучний інтелект (ШІ) – наука та технологія створення інтелектуальних машин (програмних комплексів), здатних брати на себе окремі функції інтелектуальної діяльності людини (наприклад, вибирати та приймати оптимальні рішення на основі раніше отриманого досвіду і раціонального аналізу зовнішніх впливів). З іншого визначення, під штучним інтелектом розуміється науковий напрямок, у межах якого ставляться та вирішуються завдання апаратного або програмного моделювання тих видів людської діяльності, які традиційно вважаються інтелектуальними. Саме у цьому сенсі термін штучного інтелекту ввів Джон Маккарті в 1956 р. на конференції в Дартмунтском університеті.

Під алгоритмом розуміється точне розпорядження щодо виконання у певному порядку системи операцій для вирішення будь- якого завдання з деякого класу (множини) завдань. Термін «алгоритм» вийшов від імені узбецького математика Аль-Хорезмі, який ще в IX столітті запропонував прості арифметичні алгоритми. У математиці та кібернетиці клас завдань певного типу вважається вирішеним, коли для його вирішення встановлений певний алгоритм. Знаходження алгоритмів є природною метою людини при рішенні їм різноманітних класів завдань. Відшукання алгоритму для завдань деякого типу пов'язане з тонкими та складними міркуваннями, що вимагають винахідливості та високої кваліфікації. Прийнято вважати, що подібного роду діяльність вимагає участі інтелекту людини. Завдання, пов'язані з відшуканням алгоритму рішення класу завдань певного типу можна назвати інтелектуальними. До речі, цікавий план імітації мислення, запропонований А. Тюрінгом. «Намагаючись імітувати інтелект дорослої людини, – пише Тюрінг, – ми вимушені багато роздумувати про той процес, у результаті якого людський мозок досяг свого справжнього стану. Чому б нам замість того, щоб намагатися створити програму, що імітує інтелект дорослої людини, не спробувати створити програму, яка імітувала б інтелект дитини? Адже, якщо інтелект дитини отримує відповідне виховання, він стає інтелектом дорослої людини. Наші роздуми полягають у тому, що пристрій може бути легко запрограмований. Таким чином, ми розчленуємо нашу проблему на дві частини: на завдання побудови «програми-дитини» та завдання «виховання» цієї програми». Слід додати, що саме цей шлях використовують практично всі системи ШІ. Адже зрозуміло, що практично неможливо закласти усі знання в достатньо складну систему. Крім того, на цьому шляху будуть відбиватися перераховані вище ознаки інтелектуальної діяльності (накопичення досвіду, адаптація та ін.).

Інтелектуальна задача – це процес знаходження алгоритму рішення певного класу задач. Інтелектуальними задачами є задачі, для розв'язання яких немає чітко заданого алгоритму, який завжди приводить до потрібного результату, а інтелектуальною діяльністю можна назвати процес вирішення інтелектуальних задач. Прикладом інтелектуальної задачі є розпізнавання образів, тобто визначення

належності об’єкта, що спостерігається, до однієї із заздалегідь визначених категорій. Інтелектуальним задачам властиві неповнота, неточність та суперечливість знань, а також велика розмірність простору рішень, що не дає змоги розв’язувати їх простим перебором. У таких задачах часто немає чітких критеріїв для вибору оптимального рішення, а сама задача не завжди цілком формалізується.

Основними властивостями інтелектуальних задач є:

– символьне подання умов задачі;

– відсутність чіткої постановки задачі;

– відсутність прийнятного для практичного використання алгоритму рішення, який завжди забезпечує правильний результат;

– неповнота, неточність та суперечливість знань;

– відсутність чітких однозначних критеріїв вибору оптимального рішення;

– велика розмірність простору рішень.

Постійне збільшення обсягів професійних знань та високий ступінь розвитку інформаційно-комунікаційних технологій актуалізує проблему розробки високоякісних програмних продуктів з елементами штучного інтелекту, призначених для задоволення інформаційних потреб користувачів.

Інтелектуальні системи (ІС) – це системи, що дозволяють будувати програми доцільної діяльності при вирішенні поставлених перед ними завдань на підставі конкретної ситуації, що складається на даний момент у навколишньому середовищі. При цьому інтелектуальні системи повинні зберігати працездатність при непередбачених змінах властивостей керованого об’єкту, цілей управління або навколишнього середовища (бути адаптивними). Робота будь-якої інтелектуальної системи передбачає обробку бази знань, яка складає її основу та має властивість поступово накопичувати нові знання. Інтелектуальні системи повинні накопичувати інформацію, узагальнювати її, вчитися на своїх помилках, використовуючи базу знань. Для ІС, орієнтованих на генерацію алгоритмів рішення задач, характерні наступні ознаки:

– розвинені комунікаційні здібності;

– уміння вирішувати складні, погано формалізовані завдання;

– здібність до самонавчання.

Комунікаційні здібності ІС характеризують спосіб взаємодії кінцевого користувача з системою – тобто можливість формулювання довільного запиту в діалозі системи. Складні, погано формалізовані завдання – завдання, що погано формалізуються та вимагають побудови оригінального, нестандартного алгоритму рішення, залежно від конкретної ситуації, для якої можуть бути характерні невизначеність і динамічність. Здібність до самонавчання – можливість автоматичного витягання знань для вирішення завдань з раніше накопиченого досвіду конкретних ситуацій. Оскільки в основі інтелектуальних систем, що створюються як додаток теорії несилової взаємодії, лежить обробка різних дій з виробленням адекватною цим діям реакції, то такі системи отримали назву рефлекторних (по аналогії з поведінкою об’єктів живої природи). За критерієм ступеня інтелектуалізації, який у першому наближенні може характеризуватися обсягом інформації, що обробляється, інтелектуальні системи можна поділити на:

– системи перебору варіантів рішень згідно встановленої пріоритетності для наперед змодельованих ситуацій;

– ІС, які приймають рішення за детермінованими вирішальними правилами без навчання;

– інтелектуальні системи, що реалізують алгоритми компараторного розпізнавання за еталонами;

– експертні системи, що з метою прийняття ефективних рішень маніпулюють спеціальними знаннями, накопиченими фахівцями-експертами у конкретній предметній сфері знань;

– інтелектуальні системи, що здатні самонавчатися;

– знаннє-орієнтовані ІС, що здатні утворювати та використовувати базу знань.

Інтелектуальні системи, що здатні самонавчатися, можна поділити на такі основні класи:

1. ІС, що розв’язують задачу розпізнавання образів за апріорно-класифікованою навчальною матрицею (навчання з “учителем”).

2. ІС, що реалізують алгоритми факторного кластер-аналізу.

3. ІС, що реалізують алгоритми кластер-аналізу при незмінному словнику ознак і за апріорно некласифікованими навчальними матрицями, тобто за умови неповної апріорної інформації про функціональний стан системи (навчання без “учителя”).

4. ІС, які реалізують алгоритми автоматичної класифікації за апріорно некласифікованими навчальними матрицями та здатні оптимізувати параметри словника ознак розпізнавання.

5. Відмовостійкі ІС, що здатні самостійно діагностувати свій функціональний стан і відновлювати свою функціональну спроможність при виникненні відмов.

6. Адаптивні ІС, що здійснюють класифікаційне самонастроювання та самоорганізацію.

7. ІС, що вирішують проблему шкалювання, яка полягає у побудові для шкал з різними мірами виміру зведеної шкали, координати якої можуть бути обернено відображені на відповідні вихідні шкали.

8. Сенсорні ІС, що моделюють чуттєві функції людини та базуються на “образному” комп’ютері, наділеному властивостями “технічного зору”, усномовного розпізнавання, розпізнавання пахощів тощо.

9. Гібридні ІС, які поєднують різні алгоритми та методи автоматичної класифікації.

До знаннє-орієнтованих інтелектуальних систем відносять:

1. Системи, що ґрунтуються на інструктивних знаннях (rulebased reasoning).

2. Системи, що ґрунтуються на автоматичному доведенні теорем (automatic theorem-proving techniques).

3. Системи, що ґрунтуються на автоматичному породженні гіпотез (automatic hypothesizing).

4. Системи, що ґрунтуються на доведенні за аналогією (analogical reasoning).

5. Об’єктно-орієнтовані інтелектуальні системи (object-oriented intelligent systems).

6. Об’єктно-логічні інтелектуальні системи, що поєднують окремі переваги об’єктно-орієнтованих систем з системами автоматичного доведення теорем і використовують об’єктно-логічні мови, фреймові логіки (F-logics), логіки транзакції (transaction logics) та інше.

Зрозуміло, що наведена класифікація не є досить повною, оскільки відбувається неперервне розширення номенклатури ІС як за призначенням, так і за принципами функціонування.

Інтелектуальною називається система, здатна цілеспрямовано, залежно від стану інформаційних входів, змінювати не тільки параметри функціонування, а й сам спосіб своєї поведінки, який залежить не тільки від поточного стану інформаційних входів, але і від попередніх станів системи. Інтелектуальні системи є класом автоматизованих систем обробки інформації, які моделюють розумові процеси, притаманні людині при прийнятті рішень у різних галузях соціально-економічної сфери суспільства. Конструктивно ІС складається з програмно-апаратної частини та об’єкта, процесу або явища, що досліджуються (далі замість цих понять будемо використовувати переважно узагальнюючий термін “процес”). Під проектуванням інтелектуальної системи розуміється процес створення технічного, інформаційного, програмного та організаційного забезпечення ІС для досягнення поставленої мети її функціонування. *Метою проектування* будь-якої системи є конкретизація та визначення таких значень і співвідношень її категорій, які дозволяють системі функціонувати із визначеною ефективністю. Загальна ефективність системи визначає ступінь відповідності її складових виконанню системою свого призначення згідно критерію мети. Важливою складовою загальної ефективності є функціональна ефективність, яка визначає ступінь відповідності

функціонування системи за її робочим алгоритмом виконанню поставленої перед нею задачі згідно критерію мети. Формування критерію мети спрямовано на уникнення невизначеності в описі системи. Оскільки мірою невизначеності є кількість інформації, то критерій мети має інформаційну природу, а інформаційний критерій функціональної ефективності (КФЕ) є його важливою складовою, яка визначає асимптотичні характеристики класифікатора. Залежно від вхідних даних і уявлень про побудову та функціонування системи, задачі синтезу поділяють на три класи:

– інформаційний синтез, який передбачає оптимізацію функціональної ефективності системи;

– структурний синтез, який спрямований на оптимізацію складу, конфігурації, внутрішніх і зовнішніх зв'язків системи при заданих алгоритмах її функціонування;

– комбінований синтез структури та алгоритмів функціонування, пов'язаний з розподілом функцій за елементами системи та визначенням їх оптимального складу.

На сьогодні можна виділити такі основні підходи до аналізу і синтезу інформаційних систем: алгебраїчний; геометричний; теоретико-інформаційний; теоретико-статистичний; структурний (лінгвістичний); біонічний (нейромережний); мережний; нечіткий; теоретико-ігровий та інші. Незважаючи на те, що наведені підходи відрізняються один від одного рівнем і видом математичної формалізації слабо формалізованих процесів прийняття рішень, між ними не існує чіткої межі, а самі підходи часто доповнюють один одного. Необхідною умовою синтезу системи є наявність вхідного математичного опису, який є адекватною моделлю вхідних даних, що відбивають основні характеристики та властивості функціонального стану системи. Під функціональним станом розуміються основні характеристики системи у визначений момент або інтервал часу її функціонування у заданому режимі, які залежать від технічного стану системи та середовища, що впливає на неї через контрольовані і неконтрольовані фактори. Головна задача проектування полягає у формуванні переліку вимог, яким повинна задовольняти

інтелектуальна система, та їх реалізації на етапах апріорного та апостеріорного проектування.

Етап *апріорного проектування* здійснюється за умов відсутності експериментальних даних функціонування ІС, що проектується. Тобто, це початковий етап проектування ІС з властивостями, які відсутні в природі або невідомі наперед розробнику.

Етап *апостеріорного проектування* здійснюється за умов наявності експериментальних даних, одержаних як у результаті випробувань ІС, так і в процесі її експлуатації.

При реалізації головної задачі проектування необхідно враховувати такі фактори:

– складність проектування, яка визначається повнотою апріорної інформації про ІС, що проектується;

– необхідність прийняття компромісних рішень;

– розбіжність з вимогами практики;

– можливі ризики.

Функціональна ефективність ІС, що проектується, значною мірою визначається якістю розробки технічного завдання (ТЗ), що здійснюється за результатами маркетингу та включає формування мети, вимог, оцінку можливості технічної реалізації та процедуру узгодження між замовником і виконавцем щодо вимог та характеристик проекту. Етап технічної пропозиції здійснюється з метою визначення принципів реалізації системи, структури, програмного забезпечення, що задовольняє технічне завдання. На цьому етапі проводиться патентний та інформаційний пошук прототипів та аналогів системи, що проектується. Етапи ескізного та технічного проектування належать до стадії дослідно-конструкторського розроблення ІС. На етапі ескізного проектування розв’язуються задачі аналізу та синтезу, здійснюється детальне опрацювання технічного та інформаційного забезпечення відповідно до ТЗ. Особливістю проектування ІС є необхідність проведення на цьому етапі програмної реалізації розроблених алгоритмів з метою фізичного моделювання системи, що характерно для

евристичних методів аналізу і синтезу. На етапі технічного (робочого) проектування на основі ТЗ та результатів ескізного проектування розробляється повний комплект технічної документації, який містить такі компоненти:

– електричні схеми;

– графічну документацію у вигляді необхідних креслень та специфікацій до них;

– програмну документацію: специфікацію, текст програми, опис програми, формуляр, опис застосування, інструкції для оператора, системного програміста та інше;

– текстову документацію: загальні технічні умови на систему, частинні технічні умови на комплектуючі, технічний опис, технічні формуляри, паспорти, інструкції на настроювання системи та вузлів, інструкції з експлуатації та інше.

На етапі технічного проектування випускається технічна документація, необхідна для виготовлення дослідної партії системи у заводських умовах, яка включає: технологічні інструкції; технологічні (маршрутні) карти; креслення на технологічне оснащення та необхідне технологічне обладнання. Невід'ємною частиною проектування ІС на всіх етапах є проведення випробувань системи, які мають таку класифікацію:

– випробування на рівні прийому та здачі ІС, які полягають у встановленні відповідності системи та її складових технічному завданню;

– лабораторні випробування, які відбуваються на етапах ескізного та технічного проектування і полягають в оцінці правильності функціонування, оцінок точності, стійкості, стабільності, надійності роботи з метою забезпечення технічних умов;

– сумісні випробування, які проводяться проектувальниками та замовниками за програмою лабораторних випробувань, але, як правило, за більш жорстких умов;

– натурні випробування, які проводяться у присутності замовника як завершальний етап перед здачею системи за умов, максимально наближених до заводських.

Під *функціонуванням системи* розуміється процес виконання системою заданого робочого алгоритму при використанні системи за призначенням, тобто за критерієм мети створення системи. ІС можуть функціонувати у двох режимах: автоматичному (тобто без участі в контурі функціонування системи людини – особи, що приймає рішення) або автоматизованому. Використання ІС, що функціонують в автоматичному режимі, є виправданим для розв'язання задач контролю та керування формалізованими процесами, моделювання яких можливе за традиційними математичними методами. Функціонування інтелектуальної системи можна описати як постійне прийняття рішень на основі аналізу поточних ситуацій для досягнення певної мети.

Природно виділити окремі етапи, які утворюють типову схему функціонування інтелектуальної системи:

1. Безпосереднє сприйняття зовнішньої ситуації; результатом є формування первинного опису ситуації.

2. Зіставлення первинного опису зі знаннями системи і поповнення цього опису; результатом є формування вторинного опису ситуації в термінах знань системи. Цей процес можна розглядати як процес розуміння ситуації або як процес перекладу первинного опису на внутрішню мову системи. При цьому можуть змінюватися внутрішній стан системи та її знання. Вторинний опис може бути не єдиним, і система може вибирати між різними вторинними описами. Крім того, система в процесі роботи може переходити від одного вторинного опису до іншого.

3. Планування цілеспрямованих дій та прийняття рішень, тобто аналіз можливих дій та їхніх наслідків і вибір тієї дії, яка найкраще узгоджується з метою системи. Це рішення формулюється деякою внутрішньою мовою (свідомо або підсвідомо).

4. Зворотна інтерпретація прийнятого рішення, тобто формування робочого алгоритму для здійснення реакції системи.

5. Реалізація реакції системи; наслідком є зміна зовнішньої ситуації і внутрішнього стану системи.

Не слід вважати, що вказані етапи є повністю розділеними у тому розумінні, що наступний етап починається тільки після того, як повністю закінчиться попередній. Навпаки, для функціонування інтелектуальної системи характерним є взаємне проникнення цих етапів. Наприклад, ті чи інші рішення можуть прийматися уже на етапі безпосереднього сприйняття ситуації. Насамперед, це рішення про те, на які зовнішні подразники слід звертати увагу, а на яке не обов'язково. Зовнішніх подразників так багато, що їхнє сприйняття повинно бути вибірковим. Суттєвою частиною системи штучного інтелекту є база знань, яка є результатом узагальнення досвіду експлуатації даної системи в певних конкретних умовах. Це означає, що програмістом може бути розроблена тільки «порожня оболонка» системи штучного інтелекту, яка перетворюється в працездатну систему у результаті процесу навчання, який, таким чином, є необхідним технологічним етапом створення подібних систем. Можна провести аналогію між такою системою і дитиною: дитина не може йти працювати, тому йому для цього попередньо потрібно тривале навчання в школі, а потім часто і в вузі, щоб він зміг виконувати певні види робіт.

Для тих, хто вивчає ШІ програмування служить таким же засобом, яким є математика для тих, хто вивчає старіші області науки. Під інтелектуальними системами розуміють будь-які біологічні, штучні або формальні системи, що проявляють здатність до цілеспрямованої поведінки. Останнє включає властивості спілкування, накопичення знань, прийняття рішень, адаптації і т. д. В даний час існує стійка тенденція інтелектуалізації комп'ютерів і їх програмного забезпечення (ПО). Основні функції майбутніх комп'ютерів - рішення задач все більшою мірою необчислювальних характеру, в тому числі логічний висновок, управління базами знань (БЗ), забезпечення інтелектуальних інтерфейсів і ін. Інтелектуалізація комп'ютерів здійснюється за рахунок розробки як спеціальної апаратури

(наприклад, нейрокомп'ютери), так і ПО (експертні системи, бази знань, вирішувачі завдань і т. д.). Робоче визначення поняття «інтелектуальна система» запропоновано в [39]. Система вважається інтелектуальної, якщо в ній реалізовані наступні три базові функції.

1. Подання та обробки знань. Інтелектуальна система повинна бути здатна накопичувати знання про навколишній світ, класифікувати і оцінювати їх з точки зору прагматики і несуперечності, ініціювати процеси отримання нових знань, співвідносити нові знання зі знаннями, що зберігаються в базі знань.

2. Інтелектуальна система повинна бути здатна формувати нові знання за допомогою логічного висновку і механізмів виявлення закономірностей в накопичених знаннях, отримувати узагальнені знання на основі приватних знань і логічно планувати свою діяльність.

3. Інтелектуальна система повинна бути здатна спілкуватися з людиною мовою, близькою до природної мови (ПМ) і отримувати інформацію через канали, аналогічні тим, які використовує людина при сприйнятті навколишнього світу. Відомі чотири підходи до синтезу програм:

1) дедуктивний - побудова програми виконується на основі докази, що рішення задачі існує;

2) індуктивний - програма будується за прикладами, кожен з яких визначає відповідь для деякого підкласу вихідних даних;

3) трансформаційний - програма синтезується шляхом перетворення вихідного опису завдання по правилам, сукупність яких представляє знання про її вирішенні;

4) утилітарний - програма будується з практичних потреб на основі приватних закономірностей і прийомів.

Машинне навчання (Machine Learning) — обширний підрозділ штучного інтелекту, який вивчає методи побудови алгоритмів, здатних навчатися. Машинне навчання знаходиться на стику математичної статистики, методів оптимізації та класичних математичних дисциплін, але має і власну специфіку, пов'язану з

проблемами обчислювальної ефективності та перенавчання. У книзі «Машинне навчання» [19] Томас Мітчелл дає таке означення «навченості»:

Комп'ютерна програма навчається в міру накопичення досвіду відносно деякого класу задач T і цільової функції P, якщо якість вирішення цих завдань (щодо P) поліпшується з отриманням нового досвіду

Багато методів індуктивного навчання розроблялися як альтернатива класичним статистичним підходам. Багато методів тісно пов'язані з вилученням інформації та інтелектуальним аналізом даних. Найбільш теоретичні розділи машинного навчання об'єднані в окремий напрям, теорію обчислювального навчання. Машинне навчання – не тільки математична, а й практична, інженерна дисципліна. Чиста теорія, як правило, не призводить відразу до методів і алгоритмів, які можуть застосовуватися на практиці. Щоб змусити їх добре працювати, доводиться винаходити додаткові механізми, що компенсують невідповідність зроблених в теорії припущень до умов реальних завдань. Практично жодне дослідження в машинному навчанні не обходиться без експерименту на модельних або реальних даних, що підтверджує практичну працездатність методу.

Два основні класи задач машинного навчання - це завдання навчання з учителем (supervised learning) і навчання без вчителя (unsupervised learning). При навчанні з учителем на вхід подається набір тренувальних прикладів, який зазвичай називають навчальним або тренувальним набором даних (training set або training sample - тренувальна вибірка), і завдання полягає в тому, щоб продовжити вже відомі відповіді на новий досвід, виражений зазвичай в вигляді тесотового набору даних (test set, test sample). Основне припущення тут в тому, що дані, доступні для навчання, будуть чимось схожі на дані, на яких потім доведеться застосовувати навчену модель, інакше ніяке узагальнення буде неможливо.

Завдання навчання з учителем зазвичай діляться на завдання класифікації і регресії. У задачі класифікації потрібно поданий на вхід об'єкт визначити в один з (зазвичай кінцевого числа) класів, наприклад, розділити фотографії тварин на

кішок, собак, коней і «все інше»; або по фотографії людського обличчя зрозуміти, хто з ваших друзів у соціальній мережі на ній зображений.

рис. 2.1

А в задачі регресії потрібно передбачити значення якоїсь функції, у якій зазвичай може бути нескінченно багато різних значень. Наприклад, по зросту людини передбачити його вагу, зробити прогноз завтрашньої погоди, передбачити ціну акції або, скажімо, виділити на фотографії прямокутник, в якому знаходиться людське обличчя - зробити це необхідно, щоб потім ці прямокутники подати на вхід згаданому вище класифікатором. Розподіл на регресію і класифікацію, звичайно, дуже умовний, можна легко придумати «проміжні» приклади. Але зазвичай ясно, яке завдання ми вирішуємо, і цей поділ має змістовний сенс: змінюються цільові функції і, як наслідок, процес навчання. В пошукових системах часто зустрічається завдання навчання ранжування (learning to rank). Воно ставиться так: за наявними даними (в пошуковій системі це будуть тексти документів і минулу поведінку користувачів) впорядкувати наявні об'єкти в

порядку спадання цільової функції (в пошуковій системі вона називається релевантність: наскільки даний документ підходить, щоб видати його у відповідь на даний запит). Це завдання чимось схоже на завдання регресії - нам так чи інакше потрібно передбачати безперервну цільову функцію, ту саму релевантність. Але при цьому значень самої функції в даних немає, та вони нам і не важливі. Мають значення тільки результати порівняння цієї функції на різних об'єктах. Це призводить до ряду цікавих і специфічних методів навчання. Якщо ж розміченого набору даних, відповідного конкретному завданні, немає, а є просто дані, в яких треба «знайти якийсь сенс», то виникають завдання навчання без учителя.

Типовий приклад завдання навчання без вчителя - це кластеризація (clustering): потрібно розбити дані на заздалегідь невідомі класи по деякій мірі схожості так, щоб точки, віднесені до одному і тому ж кластеру, були якомога ближче один до одного, як можна більш схожі, а точки з різних кластерів були б якомога далі один від одного, як можна менш схожі. Наприклад, вирішивши завдання кластеризації, можна виділити сімейства генів з послідовностей нуклеотидів, або кластеризувати користувачів вашого веб-сайту і персоналізувати його під кожен кластер, або сегментувати медичний знімок. Ще одне завдання навчання без вчителя - це зниження розмірності (dimensionality reduction), коли вхідні дані мають велику розмірність (наприклад, якщо у вас на вході розбитий на слова текст, розмірність буде обчислюватися десятками тисяч, а якщо фотографії - мільйонами), а завдання полягає в тому, щоб побудувати уявлення даних меншої розмірності, яке буде досить повно відображати вихідні дані. Тобто, наприклад, за поданням меншої розмірності можна буде досить успішно реконструювати вихідні точки великої розмірності. Це можна розглядати, як окремий випадок загальної задачі виділення ознак (feature extraction). І нарешті, третій і найбільш загальний клас задач навчання без учителя - задача оцінки щільності: нам дано точки даних і, можливо, якісь апріорні уявлення про те, звідки взялися ці точки, а хочеться оцінити розподіл p(x), з якого вони вийшли. Це дуже загальна постановка задачі, до неї можна багато чого звести, і нейронні мережі теж її і вирішують. Нерідко в житті виникає щось середнє між навчанням з учителем і навчанням без учителя.

Так зазвичай виходить тоді, коли нерозмічену приклади знайти дуже легко, а розмічені отримати складно. Наприклад, у прикладі з синтаксичним розбором, набрати скільки завгодно нерозмічену текстів не представляє ніякої складності, а ось вручну намалювати навіть одне дерево нелегко. Або, скажімо, задумайтеся про те, що таке «розмічені дані» для рас пізнавання мови. Просто встановити відповідність між звуковим файлом з реченням і текстом, особливо якщо тексти досить довгі, тут може бути недостатньо. По-справжньому розмічені дані для розпізнавання - це звукові файли, в яких вручну відзначені (або хоча б перевірені) межі кожної фонеми, кожного звуку людської мови. Це пекельна праця, зазвичай делегованих молодшим науковим співробітникам, але навіть ціла армія лінгвістів-фонології буде рухатися досить повільно. А як немарковані звуків живої людської мови ви можете записати скільки завгодно, просто включивши диктофон на запис. Таку ситуацію іноді називають навчанням з частковим залученням вчителя, або напів контрольованим навчанням (англійський термін semi-supervised learning. Навчанням з під- кріпленням (reinforcement learning), виникає коли агент, перебуваючи в якійсь середовищі, виконує ті чи інші дії і отримує за це нагороди. Мета агента - отримати якомога більшу нагороду з плином часу, а для цього непогано б зрозуміти, які дії в кінцевому рахунку ведуть до успіху. Так можна навчитися грати в різні ігри або, наприклад, зробити відмінний протокол для A/B-тестування.

Формально найпростіша модель навчання з підкріпленням складається з:

– безлічі станів оточення S;

– безлічі дій A;

– безлічі "виграшів".

Загалом навчання із підкріпленням працює таким чином:

– На агента приходить стан;

– Агент вибирає дію;

– Середовище посилає агенту нагороду і наступний стан;

– Агент опрацьовує нагороду і коректує свої дії [47].

1.3. Стан досліджуваної проблеми у вітчизняній та зарубіжній теорії і практиці.

Ключовим компонентом наукового фундаменту інтелектуальних інформаційних технологій є штучний інтелект (ШІ). Для створення і розвитку ШІ як наукового напрямку багато зробили Ф. Розенблат, І. Вінер, У. Маккаллох, У. Піттс, Д. Маккарті (який вперше ввів термін «artificial intelligence»), А. Сазерленд, М. Мінський, Р. Шенк, С. Пейперт, А. Ньюелл, Г. Саймон, Дж. Шоу, Е. Фейгенбаум, А. Кольмерое, Н. Хомський, Т. Виноград, М. Куілліан, І. Кільсон, П. Уїнстон, Л. заде, Р. Редді, Д. Ленат, Дж. Хінтон, Дж. Андерсон, Ж.-Л. Лорьер і багато інших. Щоб отримати уявлення про основні технології ШІ, необхідно вивчити, як його найважливіші концепції втілюються в програмних рішеннях. Програми дозволяють будувати ясні описи різноманітних процесів. Їх структури можуть відображати структури тих завдань, для вирішення яких вони призначені. В наш час переваги в конкурентній боротьбі вже не визначаються ні розмірами країни, ні її природними ресурсами. Тепер все вирішують рівень освіти і обсяг знань, накопичених суспільством. В майбутньому процвітати будуть держави, що зуміють перевершити інші в створенні і освоєнні нових знань. Особливу роль в цьому відіграють нові інформаційні технології (НІТ), а у них - методи і засоби штучного інтелекту (ШІ).

Сьогодні сфері інтелектуального аналізу тексту приділяється неабияка увага як серед наукової спільноти, так і серед інженерів програмного забезпечення у передових ІТ компаніях світу. Досить часто ці спільноти перетинаються і першокласні науковці застосовують на практиці свої знання на посаді Data scientist в тій, чи інший ІТ компанії. Звичайно, що такий симбіоз науки та бізнесу тільки сприяє загальному розвитку сфери інтелектуального аналізу тексту. Яскравим прикладом такого симбіозу є робота працівника ІТ-гіганта Google Томаса Міколова «Efficient Estimation of Word Representations in Vector Space»[8] в якій було вперше описані техніки word2vec із різними варіантами реалізації архітектури мережі (Continuous Bag of Words та Skip-gram model). Принципи, описані в роботі, сколихнули стрімкий розвиток у сфері інтелектуального аналізу тексту,

стимулювали покращення й оптимізацію описаних методів та створення альтернативних груп алгоритмів. Серед таких алгоритмів варто виокремити GloVe та fastText. Група алгоритмів GloVe або Global Vectors є дітищем лабораторії комп'ютерної лінгвістики Стенфордського університету, яка описана у статті «Global Vectors for Word Representation»[23]. Ця група алгоритмів використовує сингулярний розклад матриці та принципи машинного навчання із учителем, подібні до принципів word2vec. Ще одним значним алгоритмом інтелектуального аналізу тексту для створення векторів word embeddings є результат роботи групи Facebook AI Research Lab (FAIR) – бібліотека fastText. Принципи роботи бібліотеки описані у публікації «Bag of Tricks for Efficient Text Classification»[3] одним із співавторів якої є вже вищезгаданий Томас Міколов. На відміну від word2vec та GloVe алгоритм fastText використовує принципи машинного навчання без учителя (unsupervised learning).

Останні роки state-of-the-art результатів у багатьох сферах NLP досягли архітектури траснформерів після публікації [2]. Доцільно розглянути одного з представників, а саме алгоритм BERT(Bidirectional Encoder Representations from Transformers). BERT – це технологія обробки природної мови (NLP), заснована на нейронних мережах, яка пройшла попередню підготовку в корпусі Wikipedia. Повна абревіатура звучить: «Двонаправлені уявлення кодувальників від трансформаторів». Ми розглянемо детальніше цей підхід у 2 розділі.

Висновки до першого розділу

У цьому розділі ми з'ясували науково-теоретичний і практичний стан проблеми проектування інтелектуальних інформаційних систем, провели аналіз дефініцій категоріального апарату дослідження. Описали концептуальні основи проектування інтелектуальної інформаційної системи. Проаналізували роль та місце інформаційних технологій у сучасному суспільстві. Провели класифікацію інформаційних систем. Значну увагу було приділено означенню термінів, які часто зустрічаються у нашому досліджені. Розглянули типи інформаційних систем та задач, які вони можуть вирішувати. Дослідили сучасний стан машинного навчання, та детально дослідили його класифікацію. Довели актуальність теми дослідження.

При аналізі стану досліджуваної проблеми у вітчизняній та зарубіжній теорії і практиці, було виокремлено сучасні підходи до проектування інтелектуальних інформаційних систем. Під час роботи над цим розділом, через надзвичайно велику сферу застосування інтелектуальних інформаційних систем, було прийнято рішення сконцентрувати увагу на інформаційних системах, які працюють з текстовими даними.

РОЗДІЛ 2. МОДЕЛЮВАННЯ ІНТЕЛЕКТУАЛЬНИХ ІНФОРМАЦІЙНИХ СИСТЕМ

2.1. Сучасні підходи до моделювання інтелектуальних інформаційних систем

На основі первинного аналізу та розгляду предметної області було визначено етапи, які неминуче доведеться затронути на шляху досягнення поставлених цілей. Для рішення даної задачі було визначено, що основними являються процеси обробки вхідних даних, який включає певну попередню обробку тексту, виділення з нього певних характеристик, які будуть якимось чином описувати об’єкт (текст); процес безпосередньо класифікації та процес представлення результату. Останній буде залежати від вибраного методу для моделі класифікації.

Для самої ж класифікації існує два способи виконувати подібну класифікацію: з учителем і без вчителя. У методі з учителем модель спочатку тренується на спеціально визначених даних і тільки після цього натреновану модель можна використовувати для присвоювання певних класів з новими даними. Класифікація без вчителя вирішує проблему без тестових розмічених даних і заздалегідь визначених класів, тільки на підставі вмісту самого тексту. Використовуючи такий підхід можна автоматично розподіляти схожі документи по групах. Так, в нашому випадку їх буде всього дві. Після того, як дані перетворені в табличне представлення, їх можна використовувати для навчання моделей. Табличне представлення є найбільш поширеним представленням даних в сфері машинного навчання та інтелектуального аналізу даних. Рядки таблиці є окремими об'єктами (спостереженнями). Стовпці таблиці, містять незалежні змінні (ознаки), що позначаються X, і залежні (цільові) змінні, що позначаються y. Залежно від класу завдання, цільові змінні можуть бути представлені як одним, так і кількома стовпцями.

Насправді, розуміти просту людську мову для комп'ютера – це досить складне завдання. Занадто багато неоднозначностей, що залежать від контексту. Потрібно справлятися з неологізмами, назвами різних сутностей і ідіомами. Крім того, для розуміння природної мови, комп'ютеру потрібно мати деяку подобу здорового глузду. Тому, текстові документи представляють як n-розмірний вектор

з набором характеристик які описують конкретні об'єкти. Найчастіше це набори слів, які згруповані по унікальності з вказаним значенням повторюваності в документі. При такому підході порядок слів ігнорується і враховується тільки кількість. Розбивка тексту на окремі слова називається токенізацією (tokenization). Але для підвищення точності потрібно виконати ще ряд кроків. Один з найважливіших кроків - це видалення стоп слів (stop word removal). Нам необхідно видалити слова, що зустрічаються занадто часто, а також слова без значного семантичного сенсу. В англійській мові це «at», «which», «and», «so», «a», «an» і ще багато подібних. Ще потрібно провести нормалізацію всіх слів, виконати стемінг (stemming) і привести всі слова до їх нормальної форми. Такий підхід дозволить об'єднати різні форми слів, наприклад, слова "працювати", "працював", "працювали" будуть приведені до слова "праця". Тут потрібно врахувати, що для класифікації всі ці слова об’єднуються, і частота буде вважатися як для одного слова. До речі, основа слова не завжди збігається з морфологічним коренем слова. Можна використовувати лемматизацію (lemmatization). З її допомогою можна знаходити правильні морфологічні основи, що дозволить зберегти зміст слова. Але для лемматизації потрібно використовувати словники, а це значно більш витратний процес, ніж стемінг. І останнє - не варто забувати про приведення всіх слів до одного регістру. Для побудови класифікатора необхідно визначити, які параметри впливають на прийняття рішення про те, до якого класу належить зразок. При цьому можуть виникнути дві проблеми. По-перше, якщо кількість параметрів мала, то може виникнути ситуація, при якій один і той же набір вихідних даних відповідає прикладам, які перебувають в різних класах. Тоді виявиться неможливим навчити модель, і система не буде коректною. Вихідні дані обов'язково повинні бути несуперечливі. Для вирішення цієї проблеми необхідно збільшити розмірність простору ознак (кількість компонент вхідного вектора, відповідного зразку). Але при збільшенні розмірності простору ознак може виникнути ситуація, коли число прикладів може стати недостатнім для навчання системи, і вона замість узагальнення просто запам'ятає приклади з навчальної

вибірки і не зможе коректно функціонувати. Таким чином, при визначенні ознак необхідно знайти компроміс з їх кількістю.

Далі необхідно визначити спосіб представлення вхідних даних для нейронної мережі, тобто визначити спосіб нормування. Нормування необхідне, оскільки системи працюють з даними, зазвичай представленими числами в діапазоні 0..1, а вихідні дані можуть мати довільний діапазон або взагалі бути нечисловими даними. При цьому можливі різні способи, починаючи від простого лінійного перетворення в необхідний діапазон і закінчуючи багатовимірним аналізом параметрів і нелінійною нормалізацією в залежності від впливу параметрів один на одного. Характеристика тексту, як правило, представляє собою деяку величину, яку є можливість обрахувати для даного тексту. Як приклад можна розглядати розподілення довжин слів у виразі. Існує велика кількість задач аналізу тексту, що потребують попереднього визначення його характеристик. До їх числа відноситься, наприклад, задача визначення авторства тексту. Ефективність використання даної характеристики оцінювалась експериментально.

В якості характеристик тексту для визначення його авторства в різні часи пропонувалось використовувати наступні:

– Розподіл довжин слів, середня довжина слова [4];

– Розподіл сполучень символів деякої довжини (n – буквені n-грами) [21];

– Розподіл частин тексту та їх позиції в реченні;

– Розглядався розподіл частин на декількох перших позиціях та в кінці речення;

– Розподіл часто вживаних в мові слів. Існують частотні словники, де вказується кількість появ певного слова на мільйон інших в колекції текстів; [21]

– Словниковий запас [4]. В роботі [26] також запропоновані показники для тексту як:

$$V_1 = \frac{V}{L}; \quad V_2 = \frac{V}{\sqrt{L}}; \quad V_3 = \frac{\log V}{\log L}; \quad V_4 = \frac{\log V}{\log\log L}, \tag{2.1}$$

V – число унікальних слів в тексті; L – їх загальна кількість в тексті.

Пізніше виявилось, що більшість з них підходять і для рішення задачі на визначення емоційної забарвлення та наявність образливих зворотів в тексті. Існують також дещо інші характеристики, що можна використовувати, якщо мова йде коментарі з соціальних мереж:

– Наявність слів, написаних повністю заголовними літерами[9];

– Наявність слів, що оточені різними астерисками (знаками) [18];

– Наявність смайлів, посилань, хеш тегів тощо [9, 18, 36].

Моделювання можна розглядати як заміщення реального об’єкта його умовним еквівалентом, що називають моделлю, що забезпечує близьку до реальної поведінку в рамках допустимих обмежень. Це дозволяє зменшити складність реальних об’єктів, так як модель передає лише необхідні та важливі властивості в даній ситуації.

У більшості випадків ключовими параметрами моделей, що представляють тексти, являються слова, що містяться в тексті. Однією з таких являється векторна модель (VSM – vector space model). Задача описується наступним чином. Нехай дана колекція текстів $D = (d1, d2, \dots, dm)$, та заданий словник з термінів цієї колекції $T = (t1, t2, \dots, tn)$, . Дана модель представляє кожний текст колекції за допомогою цього словника T як n- мірний вектор, координатами якого являються частоти входження термінів словника в даний текст.

В деяких модифікаціях такого підходу частоти входження замінюються вагами, які, по суті, являються тими же частотами, тільки не абсолютними, а відносними. Існує декілька стандартних методик зважування термів. Однією з таких являється метод TF-IDF [35]. Далі наведені деякі пояснення: TF (term frequency — частота слова) — відношення числа входжень обраного слова до загальної кількості слів документа. Таким чином, оцінюється важливість слова ti в межах обраного документа:

$$TF = \frac{n_i}{\sum_k n_k} \quad (2.2)$$

де ni – число входжень слова в документ; $\Sigma k\ nk$ — загальна кількість слів в документі.

IDF (inverse document frequency — обернена частота документа) — інверсія частоти, з якою слово зустрічається в документах колекції [35].

Використання IDF зменшує вагу широковживаних слів.

$$IDF = \log \frac{|D|}{\left|\left(d_i \subset t_i\right)\right|} \quad (2.3)$$

де $|D|$— кількість документів колекції; $\left|\left(d_i \subset t_i\right)\right|$ — кількість документів, в яких зустрічається слово ti коли $ni \neq 0$.

Вибір основи логарифму у формулі не має значення, адже зміна основи призведе до зміни ваги кожного слова на постійний множник, тобто вагове співвідношення залишиться незмінним.

Іншими словами, показник TF-IDF це добуток двох множників: TF та IDF:

$$TF - IDF = TF \cdot IDF \quad (2.4)$$

Більшу вагу TF-IDF отримають слова з високою частотою появи в межах тексту та низькою частотою вживання в інших документах колекції.

Первинний аналіз предметної області дає розуміння, що дана задача відноситься до категорії класифікації текстів на основі виділених з них ознак. Одним із способів вирішення таких, коли рішення приймається на основі лінійного оператора над вхідними даними. Він підходить до класу задач, що володіють властивістю лінійної сепарабельності. Кажучи простими словами, операцію лінійної класифікації для двох класів можна представити як відображення об’єктів в багатовимірному просторі на гіперплощину, в якій об’єкти, що попали по одну сторону розділяючої лінії відносяться до умовно першого класу, а об’єкти по іншу – до другого класу. Визначення для сказаного формулюється наступним чином. Нехай вектор x з дісних чисел представляє собою вхідні дані, а на виході класифікатора визначається показник y за формулою:

$$y = f\left(\vec{w} \cdot \vec{x}\right) = f\left(\sum_j \left(w_j x_j\right)\right) \quad (2.5)$$

де $\vec{w}$ – дійсний вектор ваг, f – функція перетворення скалярного добутку.

Другими словами, вектор ваг $\vec{w}$ – коваріантний вектор або лінійна форма відображення x в R. Значення ваг для вектору $\vec{w}$ визначається в ході машинного навчання на прикладах. Функція f зазвичай являє собою просту порогову функцію, що відділяє один клас від іншого. В більш складних випадках для неї має місце бути вірогідність для того чи іншого рішення. Існує два основних підходи для визначення параметрів лінійного класифікатора $\vec{w}$ – породжувальні та розрізнювальні моделі. Перша використовує умовний розподіл $P(x|class)$. Сюди можна віднести:

– Лінійний дискримінантний аналіз (або лінійний дискримінант Фішера) (ЛДА, англ. *LDA*) — передбачає ґаусові моделі умовної густини.

– Наївний баєсів класифікатор з поліноміальними або багатовимірними моделями подій Бернуллі.

Розрізнювальні моделі (їх ще називають дискримінативні) прагнуть покращити якість вхідних даних на наборі прикладів для навчання. До таких можна віднести:

– Логістичну регресію – алгоритм, за яким не відбувається передбачення значення числової змінної виходячи з вибірки вхідних значень. Натомість, значенням функції являється вірогідність того, що дане вихідне значення належить до певного класу.

– Простий Перцептрон – алгоритм корекції усіх помилок на вхідному наборі прикладів;

– Метод опорних векторів – алгоритм, який максимізує розділення між гіперплощиною рішення та зразками тренувального набору.

– Алгоритми з градієнтним підсиленням.

В ході досліджень наявних моделей машинного навчання було визначено що найпопулярнішими являються: Naive Bayes (наївний баєсів класифікатор) [20] , SVM (метод опорних векторів) [34], та Logistic Regression (логістична регресія) [15] для класифікації текстів, LightGBM/xgBoost.

На основі аналізу предметної області було виділено ряд методів, що використовуються для класифікації на основі виділених ознак з тексту. У галузі машинного навчання метою статистичної класифікації є використання характеристик об'єкту для ідентифікації класу (або групи), до якої він належить.

Лінійний класифікатор (англ. linear classifier) досягає цього ухваленням рішення про класифікацію на основі значення лінійної комбінації цих характеристик. Характеристики об'єкту відомі також як значення ознак, і зазвичай представляються машині у векторі, що називається вектором ознак. Такі класифікатори добре працюють для таких практичних задач, як класифікація документів, і, загальніше, для задач із багатьма змінними (ознаками), досягаючи рівнів точності, порівнянних з нелінійними класифікаторами, у той же час беручи менше часу на тренування та застосування. Далі будуть розглянуті декілька підходів, що відносяться до даної групи.

Метод опорних векторів відноситься до категорії алгоритмів навчання з учителем. Він являється одним із найпопулярніших в ній. Запропонований В. Н. Вапником і в англомовній літературі відомий під назвою SVM (Support Vector Machine)[34]. Зазвичай його відносять до бінарних класифікаторів, хоча існують випадки, коли його адаптують і змушують працювати для мультикласифікації. Особливою властивістю методу опорних векторів являється неперервне зменшення емпіричної помилки класифікації та збільшення зазору, тому його також називають як «метод класифікатора з максимальним зазором». Основна ідея полягає в переводі векторів до простору більш великої розмірності та пошуку розділяючої гіперплощини з максимальним зазором в даному просторі. Дві паралельні гіперплощини будуються по обидві сторони гіперплощини, що розділяє класи. В ролі останньої буде така гіперплощина, що максимізує відстань до двох паралельних гіперплощин. Алгоритм працює в припущенні, що чим більша різниця між ними, тим менша буде середня похибка класифікатора. Формальне описання методу та його застосування наведено далі. Нехай $Y = \{1, -1\}$.

Метод опорних векторів будує класифікатор наступного вигляду:

$$a\left(x, w, w_0\right) = sign\left(\sum_{i=1}^{n} w_i \xi_i - w_0\right) \quad (2.6)$$

де $w = \left(w_1, ..., w_n\right) \in \Re^n$, $w_0 \in \Re$ – параметри алгоритму.

Рівняння $\langle w, x\rangle = w_0$ описує гіперповерхню в просторі $\Re n$. Ідея методу заключається в побудові оптимальною в деякому розумінні гіперповерхні, що буде розділяти класи y=1 та y=−1. Вибірка лінійно подільна, якшо існує така гіперповерхня, що об’єкти різних класів находяться по різні сторони даної поверхні. Оптимальною називається така розділяюча гіперповерхня, що відстань від неї до найближчого об’єкта максимальна у порівнянні с іншими розділяючими поверхнями. Вибірки на практиці зазвичай являються лінійно нероздільні. Метод узагальнюється на цей випадок шляхом дозволу похибок на об'єктах.

У випадку коли вибірка лінійно неподільна зазвичай використовується «ядровий трюк» (*kernel trick*). Його суть наступна. Нехай дано відображення

$\varphi : \Re^n \to H$, де H – простір із скалярним добутком $\left\langle a_1, a_2 \right\rangle_H$.

Якщо цей простір має більш високу розмірність, то вельми ймовірно, що вибірка в ньому буде роздільна..

На практиці безпосередньо відображення нас не цікавить, достатньо знати ядро – функцію.

Приклади ядер:

1) $K\left(x_1, x_2\right) = \left\langle \left(x_1\right), \left(x_2\right) \right\rangle_{\Re n}$ - лінійне ядро, розділяюча поверхня матиме вид гіперповерхні;

2) $K\left(x_1, x_2\right) = \left(\left\langle \left(x_1\right), \left(x_2\right) \right\rangle_{\Re n} \right)^d$ – поліноміальне ядро, розділяюча поверхня буде мати вид поверхні порядку d;

3) $K\left(x_1, x_2\right) = e^{\frac{\left|x_1 - x_2\right|^2}{2\sigma^2}}$ – гауссове ядро с параметром σ;

4) $K\left(x_1, x_2\right) = th\left(k_0 + k_1\left\langle\left(x_1\right),\left(x_2\right)\right\rangle_{\Re n}\right)$ – сигмоїдне ядро.

Існують правила, за якими можемо побудувати свої ядра, проте в більшості випадків на практиці використовують одні з вказаних вище ядер. Для модифікації алгоритму с урахуванням ядер достатньо всюди в програмі зробити заміну звичайного скалярного добутку на функцію ядра. Даний метод опорних векторів широко використовують в різноманітних задачах машинного навчання, в яких він показує гарні результати. Можливість варіювати ядро K и параметр C забезпечує методи необхідною гнучкістю. В загальному випадку тимчасова складність алгоритму достатньо висока, а саме поліноміальна від вибірки. Для лінійного ядра існують методи, що дозволяють найти близьку до оптимальної гіперповерхню за лінійний проміжок часу. Якщо тренувальні дані є лінійно роздільними, то ми можемо обрати дві паралельні гіперплощини, які розділяють два класи даних таким чином, що відстань між ними є якомога більшою. Область, обмежена цими двома гіперплощинами, називається «розділенням» (англ. "margin"), а максимально роздільна гіперплощина є гіперплощиною, яка лежить посередині між цими двома.

Зразки на межах називаються опорними векторами. Даний метод являється одним з найпопулярніших в задачах бінарної класифікації та показує гарні результати.

Логістична регресія (Logistic regression) – метод побудови лінійного класифікатора, що дозволяє оцінювати апостеріорні ймовірності приналежності об'єктів класів. Логістична регресія - це різновид множинної регресії, загальне призначення якої полягає в аналізі зв'язку між декількома незалежними змінними (також званими регресорами або предикторами) і залежною змінною. Бінарна логістична регресія, як випливає з назви, застосовується в разі, коли залежна змінна є бінарною (тобто може приймати тільки два значення). Іншими словами, за допомогою логістичної регресії можна оцінювати вірогідність того, що подія настане для конкретного випробуваного (хворий/здоровий, повернення кредиту/дефолт і т.д.). Як відомо, регресивні моделі можуть бути записані у вигляді формули:

$$y = F(x_1, x_2, \dots, x_n) \quad (2.7)$$

Наприклад, в множинній лінійній регресії передбачається, що залежна змінна є лінійною функцією незалежних змінних, тобто:

$$y = a + b_1x_1 + b_2x_2 + \cdots + b_nx_n \quad (2.8)$$

Її, можна, використовувати для завдання оцінки ймовірності результату події, обчисливши стандартні коефіцієнти регресії. Наприклад, якщо розглядається варіант визначення чи являється короткий текст образливим, чи ні, задається змінна y зі значеннями 1 і 0, де 1 означає позитивну відповідь (текст являється образливим), а 0, що поданий текст – нейтральний. Однак тут виникає проблема: множинна регресія не «знає», що змінна відгуку бінарна за своєю природою. Це неминуче призведе до моделі з прогнозуючими значеннями більшими за 1 і меншими 0. Але такі значення взагалі не припустимі для початкової задачі. Таким чином, множинна регресія просто ігнорує обмеження на діапазон значень для y. Для вирішення проблеми завдання регресії може бути сформульована інакше: замість передбачення бінарної змінної, ми будемо прогнозувати безперервну змінну зі значеннями на відрізку [0,1] при будь-яких значеннях незалежних змінних. Це досягається застосуванням наступного регресійного рівняння (логіт-перетворення):

$$p = \frac{1}{1+e^{-y}}, \quad (2.9)$$

де p - ймовірність того, що відбудеться подія, що нас цікавить; e - основа натуральних логарифмів; y - стандартне рівняння регресії.

Фактично це є розподілом Бернулі. Для підбору параметрів $\theta 0, \dots, \theta n,$ необхідно скласти навчальну вибірку, що складається з наборів значень незалежних змінних і відповідних їм значень залежної змінної y. Формально, це множина пар $(x(1), y(2)), \dots, (x(m), y(m))$, де $x(i) \in Rn$ – вектор значення незалежних змінних, а $y(i) \in \{0, 1\}$ – відповідне їм значення y. Кожна така пара називається навчальним прикладом.

Зазвичай використовується метод максимальної правдоподібності, згідно з яким вибираються параметри θ, що максимізують значення функції правдоподібності на навчальній вибірці:

$$\hat{\theta} = \arg\max_{\theta} L(\theta) = \arg\max_{\theta} \prod_{k=1}^{n} P\left\{ y = y^{(i)} \mid x = x^{(i)} \right\} \quad (2.10)$$

Максимізація функції правдоподібності еквівалентна максимізації її логарифма:

$$\ln L(\theta) = \sum_{i=1}^{m} \log P\left\{ y = y^{(i)} \mid x = x^{(i)} \right\} =$$

$$= \sum_{i=1}^{m} y^{(i)} \ln f\left(\theta^T x^{(i)} \right) + \left(1 - y^{(i)} \right) \ln\left(1 - f\left(\theta^T x^{(i)} \right) \right). \quad (2.11)$$

$$\text{де } \theta^T x^{(i)} = \theta_0 + \theta_1 x_1^{(i)} + \ldots + \theta_n x_n^{(i)}.$$

Для максимізації цієї функції може бути застосований, наприклад, метод градієнтного спуску. На практиці також застосовують метод Ньютона та стохастичний градієнтний спуск. Для поліпшення узагальнюючої здатності отриманої моделі, тобто зменшення ефекту перенавчання, на практиці часто розглядається логістична регресія з регуляризацією, яка полягає в тому, що вектор параметрів θ розглядається як випадковий вектор з деякою заданою апріорної щільністю розподілу $p(\theta)$. Для навчання моделі замість методу найбільшої правдоподібності при цьому використовується метод максимізації апостеріорної оцінки.

Ця модель часто застосовується для вирішення завдань класифікації - об'єкт x можна віднести до класу y = 1, якщо передбачена моделлю ймовірність $P\{y=1 \mid x\} > 0{,}5$ і до класу $y = 0$ в іншому випадку. Отримувані при цьому правила класифікації є лінійними класифікаторами.

Баєсів підхід до класифікації заснований на теоремі, яка стверджує, що коли щільність розподілу кожного з класів відома, то шуканий алгоритм можна виписати в явному аналітичному вигляді. Більше того, даний алгоритм оптимальний, тобто

володіє мінімальною вірогідністю помилок. На практиці щільності розподілу класів, як правило, не відомі, тож їх доводиться оцінювати по навчальній вибірці. В результаті баєсів алгоритм перестає бути оптимальним, адже відновити щільність по вибірці можливо тільки з деякою похибкою. Чим коротше вибірка, тим вище шанси підігнати розподіл під конкретні дані і зіткнутися з ефектом перенавчання. І хоча даний алгоритм є одним з найстаріших, аналіз показує, що він втримує стійкі позиції і сьогодні. Цей підхід лежить в основі й багатьох інших достатньо успішних баєсових методів класифікації. До їх числа відносяться: лінійний дискримінант Фішера, метод парзенового вікна, метод радіальних базисних функцій та інші.

Naive Bayes (Наївний Баєсів класифікатор) – класифікатор, що є одним з обширної сукупності баєсових методів – алгоритмів заснованих на принципі максимума апостеріорної вірогідності. Для об’єкта, що піддають класифікації обраховують так звані функції правдоподібності кожного з класів, по ним обраховують апостеріорні вірогідності цих класів. Після чого об’єкт буде віднесено до того класу, для якого апостеріорна вірогідність максимальна. Безпосередньо сам Наївний баєсів алгоритм (далі НБА) - це алгоритм класифікації, заснований на теоремі Баєса з припущенням про незалежність ознак. Іншими словами, НБА передбачає, що наявність якої-небудь ознаки в класі не пов'язано з наявністю будь-якої іншого ознаки. Наприклад, фрукт може вважатися яблуком, якщо він червоний, круглий і його діаметр становить близько 8 сантиметрів. Навіть якщо ці ознаки залежать один від одного або від інших ознак, в будь-якому випадку вони вносять незалежний внесок у ймовірність того, що цей фрукт є яблуком. У зв'язку з таким припущенням алгоритм називається «наївним».

Теорема Баєса (за роботою [53]) дозволяє розрахувати апостеріорну ймовірність $P\ (y \mid x)$ на основі $P(y)$, $P(x)$ і $P\ (x \mid y)$:

$$\mathrm{P(y \mid x)} = \frac{\mathrm{P(x \mid y)\ P(y)}}{\mathrm{P(x)}} \quad (2.12)$$

де $P(y|x)$ – posterior probability – апостеріорна ймовірність, що дане значення ознаки x говорить про належність до класу y, саме її нам треба розрахувати; $P(x|y)$

– likelihood – ймовірність зустріти ознаку x серед всіх ознак що належать класу y (правдоподібність, тобто ймовірність даного значення ознаки при даному класі); $P(y)$ – class prior probability – безумовна (апріорна) ймовірність зустріти документ класу y в корпусі документів; $P(x)$ – predictor prior probability – безумовна (апріорна) ймовірність даного значення ознаки.

Простіше для розуміння буде переписати теорему в інших позначеннях:

$$\mathrm{P}(\theta|\mathrm{D}) = \frac{\mathrm{P}(\mathrm{D}|\theta)\,\mathrm{P}(\theta)}{\mathrm{P}(D)} \quad (2.13)$$

Якщо співвідносити це з типовим завданням машинного навчання, то тут D - це дані, то, що ми знаємо, а θ – це параметри моделі, які ми хочемо навчити. Наприклад, в моделі SVD [52] дані – це ті рейтинги, які ставили користувачі продуктам, а параметри моделі – фактори, які ми навчаємо для користувачів і продуктів. Кожна з ймовірностей теж має свій сенс. Так, $P(\theta|D)$ – це те, що ми хочемо знайти, розподіл ймовірностей параметрів моделі після того, як ми прийняли до уваги дані. Це називається апостеріорною ймовірністю (posterior probability). Цю ймовірність, як правило, безпосередньо не знайти, і тут як раз і потрібна теорема Байеса. Значення $P(D|\theta)$ – це так звана правдоподібність (likelihood), ймовірність даних за умови зафіксованих параметрів моделі. Знайти її зазвичай легко, власне, конструкція моделі зазвичай в тому і полягає, щоб задати функцію правдоподібності. Апріорна ймовірність $P(\theta)$ (prior probability) – є математичною формалізацією нашої інтуїції про предмет, формалізацією того, що ми знали раніше, ще до всіляких експериментів. Сьогодні баєсів підхід вважають скоріше як розгляд вірогідностей з позиції «ступеня довіри». Для кращого розуміння можна скористатись прикладом використання даного класифікатора при категоризації текстів. Так, якщо необхідно сортувати потік новин по темам на основі вже існуючої бази даних з темами, ми будемо використовувати так звану «bag-of-words» модель [5]: представляти документ (мульти)множиною слів, які в ньому містяться. В результаті кожен тестовий приклад x приймає значення із множини категорій V та описується атрибутами $\langle a1, a2, \dots, an\rangle$.

Сама теорема Баєса тут допомагає нам поміняти місцями причину і наслідок. Знаючи з якою вірогідністю причина приводить до деякої події, вона дозволяє розрахувати вірогідність того, що саме ця причина призвела до спостережуваної події. Ціль класифікації полягає в тому, щоб зрозуміти до якої категорії належить дана новина, тому нам потрібна не сама новина, а найбільш вірогідна категорія. Як було вже зазначено, для її визначення Баєсовий класифікатор використовує оцінку апостеріорного максимума.

Навчити окремі $P(a_i|x=v)$ набагато простіше: достатньо підрахувати статистику зустрічальності слів у категоріях. Слід зауважити, що наївний баєсовий класифікатор робить досить сильне припущення: в класифікації текстів ми припускаємо, що різні слова в тексті на одну і ту ж тему з'являються незалежно один від одного. Таке припущення в хоч і виглядає дещо нелогічним, проте це на практиці дає гарні результати.

Насправді наївний баєсовий класифікатор набагато кращий, ніж здається. Його оцінки ймовірностей оптимальні, звичайно, тільки в разі справжньої незалежності, але сам класифікатор оптимальний в куди більш широкому класі задач, і ось чому. По-перше, атрибути, звичайно, залежні, але їх залежність однакова для різних класів і «взаємно скорочується» при оцінці ймовірностей. Граматичні та семантичні залежності між словами одні й ті ж самі, що в тексті на вільну тему, що в тексті про Баєсове навчання. По-друге, для оцінки ймовірностей наївний Байес дуже поганий, але як класифікатор набагато кращий (звичайно, якщо навіть насправді але класифікація при цьому буде частіше правильна)

$$P(x = v_0|D) = 0.51;$$

$$P(x = v_1|D) = 0.49,$$

наївний Байес дає результат

$$P(x = v_0|D) = 0.99;$$

$$P(x = v_1|D) = 0.01,$$

Метричні методи класифікації, засновані на обчисленні оцінок подібності між об'єктами. Найпростішим метричним класифікатором є метод найближчих сусідів, в якому об’єкт, що класифікується буде віднесеним до того класу, якому

належить більшість схожих з ним об'єктів. Для формалізації поняття подібності вводиться функція відстані між об'єктами $\rho(x, x')$. Як правило, жорстокої вимоги, щоб ця функція була метрикою не пред'являється; зокрема, нерівність трикутника цілком може і порушуватись.

Метод k-найближчих сусідів (k-nearest neighbours, k-NN)- найпростіший метричний класифікатор, заснований на оцінюванні подібності об'єктів. Класифікується об'єкт відноситься до того класу, якому належать найближчі до нього об'єкти навчальної вибірки. Математично його можна сформулювати наступним чином. Нехай задана навчальна вибірка пар «об'єкт-відповідь».

Нехай на множина об'єктів задана функція відстані $\rho(x, x')$. Ця функція повинна бути досить адекватною моделлю подібності об'єктів. Чим більше значення цієї функції, тим менш схожими є два об'єкти (x, x'). Для довільного об'єкта u розташуємо об'єкти навчальної вибірки xi в порядку зростання відстаней до u:

$$\rho(u, x_1; u) \le \rho(u, x_2; u) \le \cdots \le \rho(u, x_m; u), \quad (2.14)$$

де через $xi;u$ позначається той об'єкт навчальної вибірки, який є i-м сусідом об'єкта u. Аналогічне позначення введемо і для відповіді на i-му сусіді: $yi;u$. Таким чином, довільний об'єкт u породжує свою нумерацію вибірки.

Природно вважати, що ця функція невід’ємна та не зростає по i. По-різному задаючи вагову функцію, можна отримувати різні варіанти методу найближчих сусідів.

1) $w(i,u) = [i = 1]$ – найпростіший метод найближчого сусіда;
2) $w(i,u) = [i \le k]$ – метод k-найближчих сусідів;
3) $w(i,u) = [i \le k]q^i$ – метод k-експоненціально зважених найближчих сусідів, де передбачається $q < 1$;
4) $w(i,u) = K\left(\frac{\rho\left(u, x_{i;u}\right)}{h}\right)$ – метод парзенового вікна фіксованої ширини h;

5) $w(i,u) = K\left(\frac{\rho\left(u, x_{i;u}\right)}{\rho\left(u, x_{k+i;u}\right)}\right)$ – метод парзенового вікна змінної ширини.

У вище наведених прикладах $K(r)$ – задана невід'ємна монотонно незростаюча функція на [0, +∞), ядро згладжування.

Основні проблеми з якими можна зіткнутись під час використання:

– Вибір числа сусідів k. При k = 1 алгоритм найближчого сусіда нестійкий до шумових викидів: він дає помилкові класифікації не тільки на самих об'єктах–викидах, а й на найближчих до них об'єктах інших класів. При $k = m$, навпаки, алгоритм надмірно стійкий і вироджується в константу. Таким чином, крайні значення k небажані. На практиці оптимальне значення параметра k визначають за критерієм змінного контролю, найчастіше – методом виключення об'єктів по одному (leave–one–out cross–validation) [14].

– Відсіювання шуму (викидів). Зазвичай об'єкти навчання не є рівноцінними. Серед них можуть знаходитися типові представники класів – еталони. Якщо об'єкт для класифікації близький до ідеалу, то, швидше за все, він належить до того ж класу. Ще одна категорія об'єктів – неінформативні або периферійні. Вони щільно оточені іншими об'єктами того ж класу. Якщо їх видалити з вибірки, це практично не позначиться на якості класифікації. Нарешті, до вибірки може потрапити кілька шумових викидів – об'єктів, що знаходяться «в гущі» чужого класу. Як правило, їх видалення тільки покращує якість класифікації. Відкидання з вибірки шумових і неінформативних об'єктів дає кілька переваг одночасно: підвищується якість класифікації, скорочується обсяг збережених даних і зменшується час класифікації, що витрачається на пошук найближчих еталонів.

При використанні даного підходу існує проблема вибору метрики. Це найбільш складна з усіх проблем. У практичних завданнях класифікації рідко зустрічаються такі «ідеальні випадки», коли заздалегідь відома хороша функція відстані $\rho(x, x')$. Якщо об'єкти описуються числовими векторами, часто беруть евклідову метрику. Цей вибір, як правило, нічим не обґрунтований – просто це

перше, що спадає на думку. При цьому необхідно пам'ятати, що всі ознаки мають бути виміряні «в одному масштабі», а найкращий випадок коли вони будуть нормовані. В іншому випадку ознака з найбільшими числовими значеннями буде домінувати в метриці, тоді як інші ознаки, фактично, враховуватися не будуть. Однак і нормування є вельми сумнівною евристикою, так як залишається питання: «Невже всі ознаки однаково значущі і повинні враховуватися приблизно з однаковою вагою?». Якщо ознак занадто багато, а відстань обчислюється як сума відхилень за окремими ознаками, то виникає проблема «прокляття розмірності». Суми великого числа відхилень з великою ймовірністю мають дуже близькі значення (відповідно до закону великих чисел). Виходить, що в просторі високої розмірності всі об'єкти приблизно однаково далекі один від одного; вибір k-найближчих сусідів стає практично довільним. Проблема вирішується шляхом відбору відносно невеликого числа інформативних ознак (features selection). В алгоритмах обчислення оцінок будується множина різних наборів ознак (так званих «опорних множин») [33], для кожного будується своя функція схожості, потім по всім функціям схожості проводиться голосування.

Методи на основі правил ухвалення рішень також можна віднести до більш загальної категорії «Логічних алгоритмів класифікації» (за К.В. Воронцовим) [41], що застосовують в класифікаторах, що побудовані на основі визначальних правил, коли простір даних моделюється набором правил, на лівій стороні якого знаходиться умова на набір ознак, а на правій – мітка класу. Набір же самих правил генерується з навчальної вибірки. Для даного тестового об’єкта ми отримуємо набір правил, для котрих він задовольняє їх умові на лівій стороні. Після чого визначається мітка класу як функція від отриманих міток класу правил. В більш загальному вигляді, ліва сторона правила являється логічною умовою, що виражена зазвичай в диз’юнктивній нормальній формі (ДНФ). Тим не менш, в більшості випадків, умова на лівій стороні значно простіша і представляє собою набір термів, всі з яких повинні знаходитись у вхідному тексті, для задоволення умови. Одним із основних принципів при побудові набору правил являється те, що простір відповідей повинен бути покритим хоча б одним правилом. В більшості випадків,

це досягається побудовою різноманітних наборів правил для кожного класу та одного правила «за замовчуванням», яке покриватиме всі інші екземпляри.

Дерево ухвалення рішень (Decision tree) – розв'язання задачі навчання з учителем, засноване на тому, як розв'язує завдання прогнозування людина. У загальному випадку - це k-те дерево з вирішальними правилами в нелистових вершинах (вузлах) і деякому висновку про цільову функцію в листових вершинах (прогнозом) (Додаток Б). Вирішальне правило - деяка функція від об'єкта, що дозволяє визначити, в яку з дочірніх вершин потрібно помістити даний об'єкт. У листових вершинах можуть перебувати різні об'єкти: клас, який потрібно присвоїти об'єкту, який туди потрапив (в задачах класифікації), ймовірності класів (в задачах класифікації), безпосередньо значення цільової функції (задачі регресії). Деревом називається кінцевий зв'язний граф с множиною вершин V, що не містить циклів та має виділену вершину $v_0 \in V$, до котрої не входить ні одне ребро. Така вершина називається коренем, а вершина, що не має вихідних ребер називається термінальною або листом. Решта вершин звуться внутрішніми. Найчастіше на практиці використовуються двійкові вирішальні дерева, що також звуться бінарними. Дерево називається бінарним, якщо з будь-якої внутрішньої вершини виходить рівно два ребра. Вихідні ребра зв'язують кожну внутрішню вершину v з лівою дочірньою вершиною Lv та з правою дочірньою вершиною Rv.

Бінарне дерево ухвалення рішень – це алгоритм класифікації, що задається бінарним деревом, в якому кожній внутрішній вершині $v \in V$ приписаний предикат $\beta v : X \to \{0,1\}$, кожній термінальній вершині $v \in V$ приписане ім'я класу $c_v \in Y$.

Об'єкт x входить до вершини v тоді і тільки тоді, коли виконується кон'юнкція що складається із усіх предикатів, прописаних внутрішніми вершинами дерево на шляху від кореня v_0 до вершини v. Нехай T – множина усіх термінальних вершин дерева.

Множина об'єктів $\Omega v = \{x \in X: Kv\ (x) = 1\}$, що виділені термінальними кон'юнкціями $v \in T$, попарно не пересікаються, а їх об'єднання співпадає з усім простором X (це доводиться індукцією по числу вершин дерева). Звідси слідує, що дерево рішень ніколи не відмовляється від класифікації, на відміну від списку

рішень [41]. Звідси також слідує, що алгоритм класифікації a: $X \to Y$, що реалізований бінарним деревом рішень, можна представити у вигляді простого голосування кон'юнкцій:

$$a(x) = \arg\max_{y \in Y} \sum_{v \in T} [c_v = y] K_v(x) \quad (2.15)$$

При чому для любого $x \in X$ один і тільки один доданок в усіх цих сумах рівний одиниці. Замість додавання можна було б використовувати і диз'юнкцію. Як зазначає К. В. Воронцов, вимога максимізації інформативності кон'юнкції Kv (x) означає, що кожна з них повинна виділяти як можна більше навчальних об'єктів, допускаючи при цьому як можна менше похибок.

Штучні нейронні мережі. Мережі з прямим зв'язком є універсальним засобом апроксимації функцій (Додаток В), що дозволяє їх використовувати в рішенні задач класифікації. Як правило, нейронні мережі виявляються найбільш ефективним способом класифікації, тому що фактично генерують велике число регресійних моделей (які використовуються в рішенні задач класифікації статистичними методами). На жаль, в застосуванні нейронних мереж в практичних завданнях виникає ряд проблем. По-перше, заздалегідь не відомо, якої складності (розміру) може знадобитися мережа для досить точної реалізації відображення. Ця складність може виявитися надмірно високою, що потребує складної архітектури мереж. Так Мінський довів, що найпростіші одношарові нейронні мережі здатні вирішувати тільки лінійно роздільні завдання. Це обмеження можна подолати при використанні багатошарових нейронних мереж. У загальному вигляді можна сказати, що в мережі з одним прихованим шаром вектор, відповідний вхідному зразку, перетворюється прихованим шаром в якийсь новий простір, який може мати іншу розмірність, а потім гіперплощини, відповідні нейронам вихідного шару, поділяють його на класи. Таким чином мережа розпізнає не тільки характеристики вихідних даних, але і "характеристики характеристик", сформовані прихованим шаром. Завдання класифікації при наявності двох класів може бути вирішена на мережі з одним нейроном у вихідному шарі, який може приймати одне з двох значень 0 або 1, залежно від того, до якого класу належить зразок. При наявності

декількох класів виникає проблема, пов'язана з представленням цих даних для виходу мережі. Найбільш простим способом представлення вихідних даних в такому випадку є вектор, компоненти якого відповідають різним номерам класів. При цьому і-а компонента вектору відповідає *і*-му класу. Всі інші компоненти при цьому встановлюються в 0. Тоді, наприклад, другому класу буде відповідати «1» на 2 виході мережі і 0 на інших. При інтерпретації результату зазвичай вважається, що номер класу визначається номером виходу мережі, на якому з'явилося максимальне значення. Наприклад, якщо в мережі з трьома виходами ми маємо вектор вихідних значень (0.2, 0.6, 0.4), то ми бачимо, що максимальне значення має друга компонента вектору, значить клас, до якого належить цей приклад – клас 2. При такому способі кодування іноді вводиться також поняття впевненості мережі в тому, що приклад відноситься до цього класу. Найбільш простий спосіб визначення впевненості полягає у визначенні різниці між максимальним значенням виходу і значенням іншого виходу, яке є найближчим до максимального. Наприклад, для розглянутого вище прикладу впевненість мережі в тому, що приклад відноситься до другого класу, визначиться як різниця між другою і третьою координатою вектору і дорівнює *0.6 - 0.4 = 0.2*. Відповідно чим вище впевненість, тим більша ймовірність того, що мережа дала правильну відповідь. Цей метод кодування є найпростішим, але не завжди найоптимальнішим способом представлення даних.

2.2 Формалізований підхід до розробки моделей класифікації текстів.

Серед інтелектуального аналізу тексту виділяють ряд задач, до яких найчастіше відносять такі: Інформаційний пошук (information retrieval) – або ідентифікація корпусів тексту (text corpus), що полягає у збиранні та накопичені множини текстових матеріалів із глобальної мережі інтернет, баз даних або інших джерел (наприклад при оцифруванні друкованих носіїв інформації). Метою інформаційного пошуку є підготовка текстової інформації до аналізу. Інтелектуальний аналіз тексту частково застосовується у сфері розпізнавання природної мови (natural language processing), а саме для синтаксичного аналізу та розмітки мови (part of speech tagging) Розпізнавання іменованих сутностей (named

entity recognition) – використовується журналістами або статистиками для автоматизованого визначення іменованих величин, таких як імена відомих людей, найменування фірм та організацій, відомих місць тощо. Розпізнавання спеціальних шаблонів – напрям, подібний до named entity recognition, однак замість іменованих сутностей використовуються інші спеціалізовані сутності, такі як номера телефонів, адреси електронної пошти, штрих-коди та інші спеціальні фрази. Визначення пов'язаних пар слів (кореферентність – coreference) – ідентифікація іменників та іменникових словосполучень, що позначають один і той самий предмет. Емоційний аналіз слів – визначення за поданим текстом його емоційного забарвлення, внутрішнього стану автора, його бачення, думки, експресивності тощо.

Одним із найважливіших завдань інтелектуального аналізу тексту є вибір способу представлення тексту в пам'яті комп'ютера. Чим оптимальніше вибране представлення, тим успішнішою та ефективнішою буде робота того, чи іншого алгоритму аналізу тексту та більш репрезентативними будуть результати його роботи. Мабуть, чи не першою ідеєю, що спадає на думку для представлення тексту на комп'ютері є кодування слів числами за порядком їх розміщення у словнику. Ідея є досить продуктивною у своїй простоті – натуральний ряд є нескінченним і можна занумерувати всі необхідні слова не турбуючись про виникнення колізій. Однак у цієї ідеї є один, досить суттєвий недолік: слова у словнику зазвичай ідуть в алфавітному порядку і при добавленні нового слова потрібно буде перенумерувати більшу частину слів словника. Але найбільшим мінусом цього підходу є повна втрата семантичного значення слова. Наприклад слова «пес», «собака» та «цуценя» суттєво різняться у лексичному сенсі та ще й стоять далеко один від одного у словнику. Однак, у той же час, вони означають самця, самку та дитинча одного і того самого виду тварин тому є близькими у семантичному сенсі. Для того, щоб отримати можливість зберегти семантичну близькість слів було запропоновано використовувати word embeddings, тобто поставити у відповідність кожному слову деякий вектор, що відображає його у певному «просторі сенсів слів». Найпростіше представити слово у вигляді вектора можна наступним чином.

Візьмемо простір векторів розмірності, що рівна кількості слів у словнику. Тоді кожне слово буде однозначно представлене у вигляді вектора, якщо навпроти його позиції в словнику поставити одиницю у відповідній координаті вектора. Таке представлення слів отримало назву One-hot encoding (OHE).

Однак one-hot encoding все ще не може похвалитись представленням семантичного сенсу слова. Проте OHE може бути використане для певного ряду задач інтелектуального аналізу тексту, коли значення одного конкретно взятого слова не таке вже й важливе. Наприклад, візьмемо декілька наборів слів або текстів. Якщо потрібно представити тексти у векторній формі то можна застосувати OHE для кожного окремого слова й отримані вектори додати. Тобто на виході отримаємо підрахунок кількості слів у тексті в одному векторі. Такий підхід називається «мішок слів» (Bag of Words), тому що втрачається вся інформація про взаємозв'язок слів у тексті. Однак використовуючи модель Bag of Words можна вже порівнювати тексти (наприклад розрахувати косинусну відстань між векторами для двох текстів). Також можна удосконалити модель та представити корпус (набір текстів) у вигляді матриці «слово-документ» (termdocument). Таку матрицю часто називають «зворотним індексом» (inverted index) у тому сенсі, що звичайний/прямий індекс виглядає як «документ-слово» і є досить незручним для швидкого пошуку. Матриця «слово-документ» використовується у тематичному аналізі текстів, де її намагаються представити у вигляді добутку матриць «слово-тема» та «тема-документ». В найпростішому випадку із матриці «слово-документ» за допомогою сингулярного розкладу отримують представлення слів через теми і документів через теми. Word embeddings Вищеописані підходи були (і залишаються) оптимальними для часів (та сфер) в яких кількість текстів є невеликою, а словник обмеженим. Однак із поширенням всесвітньої мережі інтернет у відкритому доступі з'явилось надзвичайно великі об'єми текстової інформації з якими традиційні підходи втратили свою актуальність. У 2013 році чеським аспірантом Томасом Міколовим був запропонований новий метод для кодування слів у вектори (word embeddings). Цей метод базується на дистрибутивній гіпотезі, запропонованій ще у 1954 році Гаррісом. Згідно цієї

гіпотези слова, що зустрічаються в однакових контекстах є семантично близькими. Під контекстом тут розуміється сусідні слова та словосполучення для даного слова. Наприклад у реченні «Quick brown fox jumps over the lazy dog» контекстом для слова «fox» є словосполучення «Quick brown» та «jumps over». Міколов запропонував для кожного слова із словника (причому словник може складатись із значної кількості слів) ставити у відповідність не one-hot encoding вектори (розмірність яких рівна кількості слів у словнику), а певні вектори меншої розмірності (зазвичай від 50 до 1000). Ключовим моментом стало правило: близькі за значенням слова (тобто слова у які зустрічаються у подібних контекстах) мають відповідати подібним векторам (подібність визначається як косинус кута між векторами). Для того, щоб побудувати правило із описаними властивостями був застосований апарат штучних нейронних мереж. У своїй роботі [8] Міколов описав техніку перетворення слів у вектор, яку так і назвав – word2vec. Окрім того Томас запропонував дві архітектури штучних нейронних мереж (Continuous Bag of Words та Skip-gram model) для своєї техніки. Таким чином був запропонований новий спосіб представлення текстової інформації для інтелектуального аналізу тексту, для якого великі об’єми вхідних корпусів перетворились із непосильної перешкоди на ефективну перевагу. Адже при більших об’ємах тренувальної множини тільки покращується якість навчання нейронної мережі, що виступає моделлю при такому підході. Окрім того, новий спосіб містив деякі цікаві «побічні ефекти». Оскільки кожному слову відповідає вектор, то для слів стають доступними всі можливі операції над векторами. Класичним (та канонічним) прикладом являється наступний вираз, запропонований самим Міколовим:

$$[\text{king}] - [\text{men}] + [\text{woman}] \approx [\text{queen}]$$

Тобто, стає можливим виконувати «додавання та віднімання слів» та отримувати осмисленні результати! Як видно із прикладу, якщо від векторного представлення «короля» відняти «чоловіка» та додати векторне представлення «жінки» то отримаємо вектор, що відповідає слову «королева». Про подібні результати можна навіть і не мріяти, використовуючи традиційні методи представлення тексту.

Запропонована Міколовим техніка word2vec для створення векторного представлення слів (word embeddings) містила аж два різні варіанти реалізації. Ці варіанти різняться архітектурою штучної нейронної мережі, котра власне і реалізує «функцію перетворення слів у вектори». Обидві нейронні мережі є звичайними мережами прямого поширення із одним вхідним шаром (input layer), одним прихованим шаром (hidden layer) та вихідним шаром (output layer). Мережі різняться представленням вхідного та вихідного векторів, що у свою чергу впливає на навчання та архітектуру. Принципом роботи мережі Continuous Bag of Words (CBOW) є передбачення слова за його контекстом, а для мережі Skip-gram навпаки – передбачення контексту за вхідним словом.

Розглянемо кожну архітектуру детальніше. CBOW – це звичайна «модель мішка слів», процес тренування якої полягає у поданні на вхід контекстів (зазвичай це чотири слова-сусіди: двоє попередніх та двоє наступних) без урахування порядку слідування у тексті. Виходом мережі є найбільш ймовірне слово при даному контексті. Skip-gram модель, яку ще називають k-skip-n-gram моделю, це послідовності довжини n, в яких елементи знаходяться на відстані не більшій ніж k один від одного. Наприклад маємо речення «fox jumps over the dog». Для нього 1-skip-2-gram будуть виглядати як набори біграм («fox jumps», «jumps over», «over the», «the dog») та наступні послідовності: «fox over», «jumps the», «over dog» – тобто пропускаємо одне (1-skip) слово, а із того, що залишилось справа і зліва від пропущеного створюється біграми (2-gram). Риторичним залишається питання: для яких же випадків краще використовувати ту, чи іншу архітектуру нейронної мережі? Власне сам Міколов у своїй роботі рекомендує використовувати CBOW для великих корпусів текстів (сто мільйонів, мільярд та навіть більше) оскільки навчання швидше і краще працює із більш частотними словами. Для Skip-gram краще використовувати невеликі корпуси текстів (менше за сто мільйонів слів), оскільки ця модель краще враховує рідкісніші слова та працює повільніше.

Попередня підготовка мовної моделі показала свою ефективність для вдосконалення багатьох завдань з обробки природних мов. До них відносяться завдання рівня речення, такі як умовивід про природну мову та перефразовування,

які мають на меті передбачити зв’язки між реченнями, аналізуючи їх цілісно, а також завдання на рівні лексеми, такі як визнання сутності та відповіді на запитання, де для отримання моделей потрібно виробляти дрібнозернистий вихід на рівні лексеми. Існують дві існуючі стратегії застосування попередньо підготовлених представлень мови до завдань нижче: основана на функції та точна настройка.

Підхід на основі функцій, такий як ELMo, використовує архітектури, орієнтовані на завдання, як додаткові функції. Підхід до тонкої настройки, такий як Генеральний попередньо підготовлений трансформатор OpenAI GPT, вводить мінімальні параметри, що відповідають певним завданням, і навчається для завдань нижче за допомогою простої настройки всіх перевірених параметрів. Два підходи поділяють однакову цільову функцію під час попередньої підготовки, коли вони використовують однонаправлені мовні моделі для вивчення загальних уявлень про мову.

Ми стверджуємо, що сучасні методи обмежують владу попередньо підготовлених представлень, особливо для підходів до тонкої настройки. Основне обмеження полягає в тому, що стандартні мовні моделі є однонаправленими, і це обмежує вибір архітектур, які можуть бути використані під час попередньої підготовки.

Наприклад, у OpenAI GPT автори використовують лівосторонню архітектуру, де кожен маркер може відвідувати лише попередні лексеми в шарах уваги трансформатора. Такі обмеження є неоптимальними для завдань рівня речень і можуть бути дуже шкідливими при застосуванні підходів, заснованих на точковому налаштуванні, до завдань рівня лексеми, таких як відповіді на питання, де важливо включити контекст з обох напрямків.

BERT знімає раніше згадане обмеження однонаправленості, використовуючи попередню навчальну мету «модель маскованої мови» MLM. Маска в мові випадково приховує деякі лексеми з вхідних даних, а мета – передбачити, оригінальний ідентифікатор маски, слово, засноване лише на його контексті. На відміну від попередньо тренувальної моделі ліво-орієнтовної мови, мета MLM дозволяє представництву об’єднати лівий і правий контекст, що дозволяє нам шукати глибокий двонаправлений трансформатор. На додаток до маскованої мовної моделі, ми також використовуємо завдання "передбачення наступного речення", яке спільно шукає подання текстових пар.

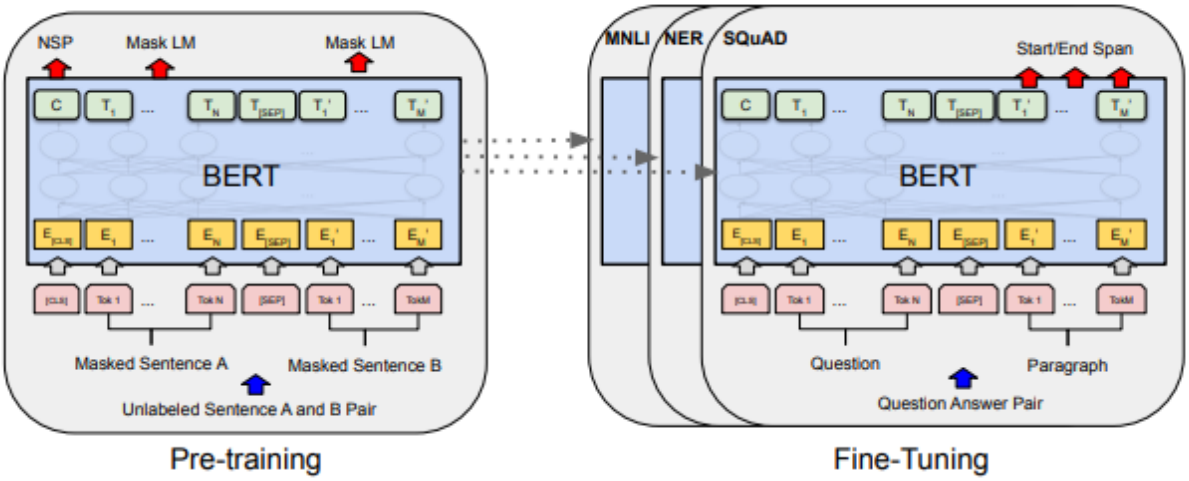


рис. 2.2

На рисунку представлено схема попереднього навчання та точного налаштування моделі BERT.

Щоб довчити BERT під конкретну задачу, необхідно поверх нього додати один-два шари простий Feed Forward мережі, і довчити тільки її, не чіпаючи основну мережу BERT. Це можна зробити або на TensorFlow, або через оболонку Keras BERT. Таке тонке налаштування під конкретний домен відбувається дуже швидко і повністю аналогічно Fine Tuning в згортальних мережах. Так, під задачу SQuAD можна довчити нейромережу на одному TPU всього за 30 хвилин (в порівнянні з 4 днями на 16 TPU для навчання самого BERT).

2.3 Моделювання класифікатора текстових даних

Для розв’язку задачі класифікації текстових даних, на основі попереднього аналізу, було прийнято рішення провести порівняльний аналіз різних підходів за такою загальною схемою:

- Збір даних:
  - Використання готових наборів даних(Kaggle, Google Dataset Search, Machine Learning Repository, VisualData, DATA USA);
  - Створення своїх наборів даних;
- Очищення даних:
  - Видалення нерелевантних символів;
  - Токенізація;
  - Видалення нерелевантних слів;
  - Нормалізація;
  - Зміщення слів(різне написання, синоніми);
  - Лематизація/стемінг;
- Представлення даних:
  - One-hot-encoding(bag of words);
  - TF;
  - TF – IDF;
  - Word2Vec;
  - GloVe;
  - CoVe;
  - BERT;
- Вибір класифікатора:
  - Метод опорних векторів;
  - Логістична регресія;
  - Наївний Баєєсів класифікатор;
  - Методи на основі правил ухвалення рішень;
  - Штучні нейронні мережі;
  - Transfer Learning;

- Валідація та інтерпретація результатів

Через велику кількість варіантів вибору на кожному кроці наведеного вище підходу, доцільніше реалізувати як найбільше варіантів моделі, та на основі експериментальних даних приймати рішення, щодо вибору найкращого підходу.

Висновки до другого розділу

У цьому розділі ми проаналізували сучасні підходи до моделювання інтелектуальних інформаційних систем. Описали формалізований підхід до розробки моделей класифікатора текстових даних. Розробили модель інтелектуальної інформаційної системи. Навели загальну схему проектування класифікатора текстових даних.

В ході дослідження було проаналізовано еволюцію методів представлення текстової інформації в галузі інтелектуального аналізу тексту від традиційних статистичних методів (One-hot encoding, Bag of Word) до сучасної техніки word embeddings. Детально розглянуто основи техніки представлення тексту word embeddings та описано тонкощі її реалізації. Окрім того, у статті було детально досліджено принципи роботи методу перетворення слів у вектори word2vec та двох його реалізацій CBOW та Skip-gram, що різняться архітектурою нейронних мереж. Виявлено таку суттєву перевагу word embeddings над статистичними методами, як «семантичне додавання/віднімання слів». Також, було досліджено сучасний підхід до інтелектуального аналізу тексту на основі трансформаторів на прикладі попередньо навченої моделі BERT.

РОЗДІЛ 3. ПРОЕКТУВАННЯ ІНТЕЛЕКТУАЛЬНИХ ІНФОРМАЦІЙНИХ СИСТЕМ

3.1. Аналіз наборів даних для навчання текстового класифікатора

Для обраної задачі класифікації текстових даних було обрано набори даних «Spam SMS» [32], «Quora Insincere Questions Classification» [27] та «Real or Not? NLP with Disaster Tweets» [29] із ресурсу Kaggle.

Spam SMS вважається одним з класичних для дослідження спаму. Набір даних Spam SMS - це набір SMS повідомлень, зібраних для дослідження спаму. Він містить 5574 SMS повідомлень англійською мовою позначених як “ham” (законний) або “spam”. Цей корпус був зібраний з безкоштовних джерел.

Quora Insincere Questions Classification. Проблема будь-якого великого веб-сайту сьогодні - це поводження з токсичним та розбіжним вмістом. Quora хоче вирішити цю проблему наперед, щоб зберегти свою платформу місцем, де користувачі можуть почувати себе безпечно ділитися своїми знаннями з усім світом.

Quora - це платформа, яка дає можливість людям вчитися один у одного. У Quora люди можуть задавати питання та спілкуватися з іншими, хто надає унікальну інформацію та якісні відповіді. Ключовим викликом є вилучення нещирих питань - тих, що ґрунтуються на помилкових приміщеннях, або які мають намір зробити заяву, а не шукати корисних відповідей.

У цьому змаганні завдання розробити моделі, які визначають і позначають нещирі питання. На сьогоднішній день Quora використовує як машинне навчання, так і ручний огляд для вирішення цієї проблеми. З вашою допомогою вони можуть розробити більш масштабовані методи виявлення токсичного та оманливого вмісту.

Загальний опис даних. У цьому змаганні завдання полягає у тому, що потрібно визначити чи щире питання, задане на Quora, чи ні.

Нещире запитання визначається як питання, призначене зробити заяву, а не шукати корисних відповідей. Деякі характеристики, які можуть означати, що питання нещире:

– Має нейтральний тон.

– Має перебільшений тон, щоб підкреслити думку про групу людей.

– Є риторичним і мається на увазі твердження про групу людей.

– Є зневажливим чи запальним.

– Пропонує дискримінаційну ідею щодо певного класу людей або шукає підтвердження стереотипу.

– Робить зневажливі напади/образи щодо конкретної людини чи групи людей.

– Ґрунтується на расистських уявленнях про групу людей.

– Відхилення від характеристики, яка не піддається фіксації та не піддається вимірюванню.

– Не обґрунтовано насправді.

– Засновується на неправдивій інформації або містить абсурдні припущення.

– Використовує сексуальний вміст для шокового впливу, а не пошуку справжніх відповідей.

Дані навчання включають запитання, яке було задано, та чи було воно ідентифіковане як нещире.

Real or Not? NLP with Disaster Tweets. Twitter став важливим каналом зв'язку в надзвичайні ситуації. Повсюдна здатність смартфонів дозволяє людям оголосити про надзвичайну ситуацію, яку вони спостерігають у режимі реального часу. Через це все більше агентств зацікавлене в тому, щоб систематично контролювати Twitter (тобто організації, що допомагають внаслідок аварій та інформаційні агенції). Але не завжди зрозуміло, чи насправді слова людини сповіщають про катастрофу.

У цьому змаганні поставлено завдання створити модель машинного навчання, яка передбачає, які повідомлення стосуються реальних катастроф, а які - ні. Ми маємо доступ до набору даних 10 000 повідомлень, які були класифіковані вручну.

3.2. Реалізація проекту та вибір середовища розробки

Для реалізації додатку, що зможе задовільнити поставлені до нього вимоги, необхідно визначитись з мовою програмування, на якій власне і буде він написаний. Постає питання вибору найзручнішої, що в рамках даного проекту. Очевидно, що простіше всього буде використовувати наявні бібліотеки, які вже мають реалізовані методи та підходи в собі, що повинно суттєво скоротити час, що буде затрачений на написання програми. В цьому розділі буде описаний короткий аналіз та обґрунтування до вибору мови програмування, існуючих бібліотек з готовими рішеннями, вибір середовища розробки та інших додаткових засобів. Всі висновки з приводу вибору тих чи інших рішень базуватимуться на основі огляду популярності, відгуків, певних кількісних показниках та перспективах розвитку та подальшого потенційного вдосконалення.

У новачків в Data Science часто виникає питання про те, яку мову програмування вибрати основною - специфічний, створений спеціально для обробки даних «R» [28], або популярний і в інших сферах та універсальний «Python» [25]. Обидві мови підтримуються open-source ліцензіями (на відміну від комерційних інструментів SAS і SPSS або пропрієтарного MATLAB) та традиційно розглядаються як найбільш затребувані. Стрімкий розвиток Data Science призводить до швидкої зміни позицій фаворитів. Далі буде коротко проаналізовано тенденції в протистоянні Python і R на початок 2018 року. Було прийнято рішення не розглядати інші мови програмування такі як Java, C# або інші «.net framework-s languages» в причину відсутності гідних бібліотек-претендентів з безкоштовною ліцензією для використання в додатках, написаних вказаними мовами (детальніше питання вибору бібліотек та інших засобів розробки описано в наступних розділах). Аналіз пошукових запитів Google Trends і даних Kaggle показують, що мова програмування загального призначення Python, очевидно, більш затребувана ніж R, і за версією Stack Overflow входить в п'ятірку найбільш популярних мов програмування. Тому порівнювати самі по собі Python і R некоректно - потрібно розглядати їх щодо нашої предметної області. Аналізуючи статистику Google Trends з січня 2012-го року по січень 2018 го по запитам про застосування Python і

R в машинному навчанні, видно, що протягом останніх двох років інтерес до Python став переважати і на початок 2018 року число запитів суттєво перевершує аналогічні звернення, пов'язані з «R»

Виходячи з реальних відгуків спеціалістів, звичайних користувачів та огляду даного сегменту можна виділити переваги та недоліки кожної з мов. Переваги «R»:

– Мова створювався спеціально для аналізу даних: запис конструкцій мови зрозуміла багатьом фахівцям в області;

– Багато функцій, необхідні для аналізу даних, є вбудованими функціями мови. Перевірка статистичних гіпотез часто займає лише кілька рядків коду;

– Установка IDE (RStudio) і необхідних пакетів обробки даних гранично спрощена;

– Зручний репозиторій пакетів і велика кількість готових тестів практично під всі методи Data Science і машинного навчання;

– Ефективна робота з векторами і матрицями

Недоліки «R»:

– Низька продуктивність;

– Специфічність в порівнянні зі стандартними мовами програмування, так як мова вузькоспеціалізована;

– «R» прекрасний інструмент для статистики і відповідних stand-alone додатків, але просідає в тих областях, де традиційно застосовуються мови загального призначення;

– Є можливість виконати один і той же функціонал різними способами. Синтаксис для розв’язування деяких завдань не зовсім очевидний;

– В силу великої кількості бібліотек, документація більшості, менш популярних з них, не є повною.

Переваги «Python»:

– Практично повністю володіє усіма перевагами мови «R», за деякими винятками;

– Універсальна багатоцільова мова: можна здійснювати не тільки обробку даних, але також їх пошук і використання результату обробки навіть у веб-додатку;

– Є однією з тих мов, які відмінно підходять на роль першої мови програмування. У разі втрати інтересу до машинного навчання чи Data Science набуті навички знадобляться в інших прикладних областях - як при використанні самого Python, так і при освоєнні споріднених мов.

Недоліки «Python»:

– Відсутність загального сховища та брак альтернатив для багатьох бібліотек R. Однак ситуація значно покращилася за останні роки: аналітики, які брали раніше кілька різних мов для різних завдань, відзначають, що набір інструментів змістився за останні два роки в бік бібліотек на Python. Крім того, входження новачків полегшують такі збірки як Anaconda.

– Являється мовою з динамічною типізацією. Це істотно прискорює розробку програм, але заодно ускладнює пошук деяких важко відстежуваних помилок, пов'язаних з неправильним привласненням різних даних одним і тим же змінним.

Отже, з огляду на вище описані аспекти було прийнято рішення використовувати мову Python як більш зрозумілу та перспективну.

Для використання у власному додатку якихось готових рішень у вигляді бібліотек, логічним стає пошук таких серед тих, які розповсюджуються за безкоштовною ліцензією. Так як мовою програмування вибрано Python, то остаточний аналіз існуючих бібліотек було проведено після її вибору. Звичайно, початкові обстеження були зроблені під час вирішення питання мови програмування. Далі зроблено огляд на найпопулярніші готові рішення, на які варто звернути увагу. Для обробки мови та виділення характеристик з текстів було виділено два інструменти.

NLTK (natural language toolkit) - пакет бібліотек і програм для символьного і статистичної обробки природної мови, написаних на мові програмування Python. Містить графічні уявлення і приклади даних. Супроводжується великої

документацією, включаючи книгу з поясненням основних концепцій, що стоять за тими завданнями обробки природної мови, які можна виконувати за допомогою даного пакету. NLTK добре підходить для студентів, які вивчають комп'ютерну лінгвістику або близькі предмети, такі як емпірична лінгвістика, когнітивістики, штучний інтелект, інформаційний пошук і машинне навчання. NLTK з успіхом використовується як навчальний посібник, як інструмент індивідуального навчання і в якості платформи для прототипування і створення науково-дослідних систем. NLTK є вільним програмним забезпеченням.

Gensim – популярний інструмент для автоматичної обробки мови, заснований на машинному навчанні і використовуваний як комерційними компаніями, так і академічними дослідниками. У Gensim реалізовані алгоритми дистрибутивної семантики word2vec і doc2vec, він дозволяє вирішувати завдання тематичного моделювання (topic modeling) і виділяти основні теми тексту або документа. Цільова аудиторія є обробка природної мови (НЛП) і IR співтовариство. У Gensim реалізовані популярні алгоритми НЛП, наприклад, word2vec. Більшість реалізацій алгоритмів вміє використовувати кілька ядер.

Для використання готових рішень при роботі в галузі Machine Learning було виділено два інструменти.

Theano - це розширення мови Python, що дозволяє ефективно обчислювати математичні вирази, що містять багатовимірні масиви. Бібліотека надає базовий набір інструментів для конфігурації нейромереж і їх навчання. Найбільше визнання Theano отримала в задачах машинного навчання при вирішенні завдань оптимізації. Бібліотека дозволяє використовувати можливості GPU без зміни коду програми, що робить її незамінною при виконанні ресурсоємних завдань.

TensorFlow - це бібліотека для глибинного навчання, і її зв'язок з Google допомогли проекту TensorFlow залучити багато уваги, деякі його унікальні деталі заслуговують більш глибокого вивчення:

– Основна бібліотека підходить для широкого сімейства технік машинного навчання, а не тільки для глибинного навчання.

– Лінійна алгебра та інші нутрощі добре видно зовні.

– На додаток до основної функціональності машинного навчання, TensorFlow також включає власну систему логування, власний інтерактивний візуалізатор логів і навіть потужну архітектуру з доставки даних.

– Модель виконання TensorFlow відрізняється від scikit-learn мови Python і від більшості інструментів в R.

Scikit-learn - пакет бібліотек для вирішення завдань машинного навчання. Написаний переважно на мові Python, основні алгоритми реалізовано на Cython для досягнення продуктивності. Бібліотека добре документована, містить приклади використання і покрокові інструкції аналізу даних. SciKit - включає засоби для візуалізації даних, що аналізуються і їх оцінки основними популярними методами. Далі наведено інші загальні бібліотеки, що полегшують процес розробки програмного забезпечення, наприклад, при роботі з масивами, даними і тд.

NumPy - це бібліотека мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих математичних функцій для операцій з цими масивами. Математичні алгоритми, 84 реалізовані на Python, часто працюють набагато повільніше тих же алгоритмів, реалізованих на компільованих мовах (наприклад, Фортран, Сі, Java). Бібліотека NumPy надає реалізації обчислювальних алгоритмів (у вигляді функцій і операторів), оптимізовані для роботи з багатовимірними масивами. В результаті будь-який алгоритм, який може бути виражений у вигляді послідовності операцій над масивами (матрицями) і реалізований з використанням NumPy, працює так само швидко, як еквівалентний код, що виконується в MATLAB.

SciPy - це відкрита бібліотека високоякісних наукових інструментів для мови програмування Python. SciPy містить модулі для оптимізації, інтегрування, спеціальних функцій, обробки сигналів, обробки зображень, генетичних алгоритмів, рішення звичайних диференціальних рівнянь та інших задач, зазвичай вирішуються в науці і при інженерної розробці. Бібліотека розробляється для тієї ж аудиторії, що MATLAB і Scilab. Для візуалізації при використанні SciPy часто застосовують бібліотеку Matplotlib, що є аналогом засобів виведення графіки MATLAB. В даний час SciPy поширюється під ліцензією BSD і його розробники

спонсоруються Enthought. На основі первинного огляду було обрано наступні бібліотеки: SciPy для роботи з матрицями; Scikit-learn – для модуля машинного навчання, а також і для виділення ознак з тексту; Numpy – для додаткових математичних операцій та структур даних.

Для Python можна використовувати різні середовища розробки, але однією з найпопулярніших є середовище PyCharm, створена компанією JetBrains. Це середовище динамічно розвивається, постійно оновлюється і доступна для найбільш поширених операційних систем - Windows, MacOS, Linux. Правда, вона має одне важливе обмеження. А саме вона доступна в двох основних варіантах: платний випуск Professional і безкоштовний Community. Багато базових можливостей доступні і в безкоштовному випуску Community. У той же час ряд можливостей, наприклад, веб-розробка, доступні тільки в платному Professional. Іншим середовищем розробки, яке дозволяє працювати з Python, є Visual Studio. Перевагою даної IDE в порівнянні, скажімо, з PyCharm, слід відзначити перш за все те, що в її безкоштовною редакції VS 2017 Community безкоштовно доступні ряд функцій і можливостей, які в тому ж PyCharm доступні тільки в платній версії Professional Edition. У той же час розробка на Python в Visual Studo доступна поки тільки в версії для Windows.

Третій варіант середовища розробки, це хмарне середовище розробки Kaggle Notebook має такі переваги:

- Повністю безкоштовне.
- Простота роботи з наборами даних, які доступні на платформі Kaggle.
- Наявні всі необхідні бібліотеки.
- Надається GPU з 4 GB пам'яті для навчання глибоких нейронних мереж.
- Надається 16 GB RAM пам'яті.
- Можливість працювати з різних пристроїв.
- Для роботи достатньо навіть смартфону з доступом до інтернету.

Четвертий варіант середовища розробки, це хмарне середовище розробки Colab, яке має такі переваги:

- Повністю безкоштовне.

- Простота роботи з наборами даних, які розміщуються на Google Drive.
- Наявні всі необхідні бібліотеки.
- Надається на вибір GPU або TPU для навчання глибоких нейронних мереж.
- Надається 16 GB RAM пам'яті.
- Можливість працювати з різних пристроїв.
- Для роботи достатньо навіть смартфону з доступом до інтернету.

Для реалізації додатку, на основі аналізу можливих середовищ розробки, було прийнято рішення працювати з Kaggle Notebook та Colab.

3.3. Аналіз ефективності запропонованих класифікаторів

Для класифікація повідомлень використовувалися такі алгоритми:

– Support Vector Classification;

– k-nearest neighbours;

– Multinomial Naive Bayes;

– Decision tree classifier;

– Logistic regression;

– Random forest classifier;

– Ada boost classifier;

– Bagging classifier;

– Extra trees classifier.

Для оцінювання моделей використовувалася метрика accuracy.

Більшість алгоритмів показали точність більшу за 96%, на валідаційному наборі, з векторизатором one-hot-encoding рис 3.1.

	Score_OHE
SVC	0.831340
KN	0.861244
NB	0.981459
DT	0.967105
LR	0.982057
RF	0.982057
AdaBoost	0.977273
BgC	0.971890
ETC	0.979665

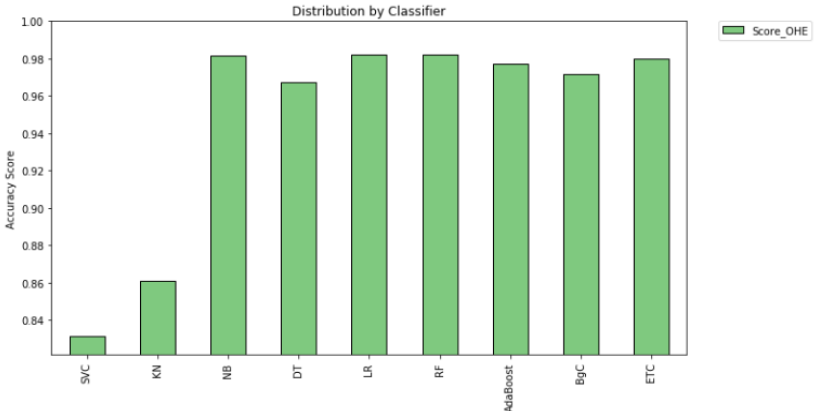


рис. 3.1

На рис 3.2 представлено результати при застосуванні векторизатора TF-IDF.

	Score_OHE	Score_TF-IDF
SVC	0.831340	0.978469
KN	0.861244	0.924641
NB	0.981459	0.984450
DT	0.967105	0.958732
LR	0.982057	0.943182
RF	0.982057	0.970096
AdaBoost	0.977273	0.967105
BgC	0.971890	0.967105
ETC	0.979665	0.977273

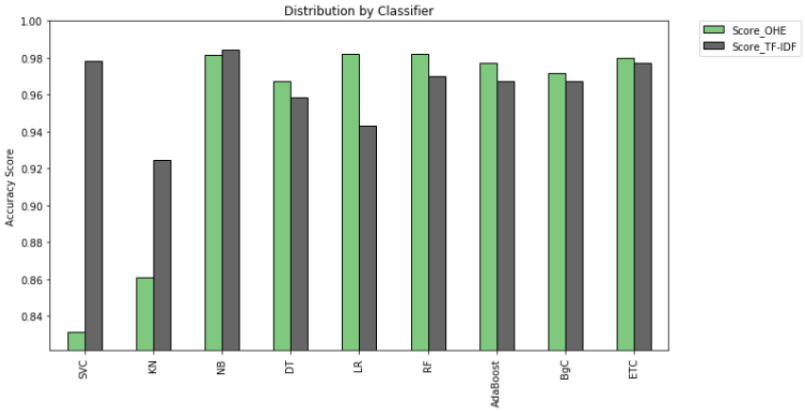


рис. 3.2

На основі аналізу набору даних, було прийнято рішення додати ознаку, яка відповідає довжині повідомлення. На рис. 3.3 та 3.4 зображено результат роботи моделей після додавання описаної ознаки.

	Score_OHE	Score_TF-IDF	Score_add_feature_length
SVC	0.831340	0.978469	0.861244
KN	0.861244	0.924641	0.881579
NB	0.981459	0.984450	0.981459
DT	0.967105	0.958732	0.961722
LR	0.982057	0.943182	0.948565
RF	0.982057	0.970096	0.974880
AdaBoost	0.977273	0.967105	0.967703
BgC	0.971890	0.967105	0.959330
ETC	0.979665	0.977273	0.971890

рис. 3.3

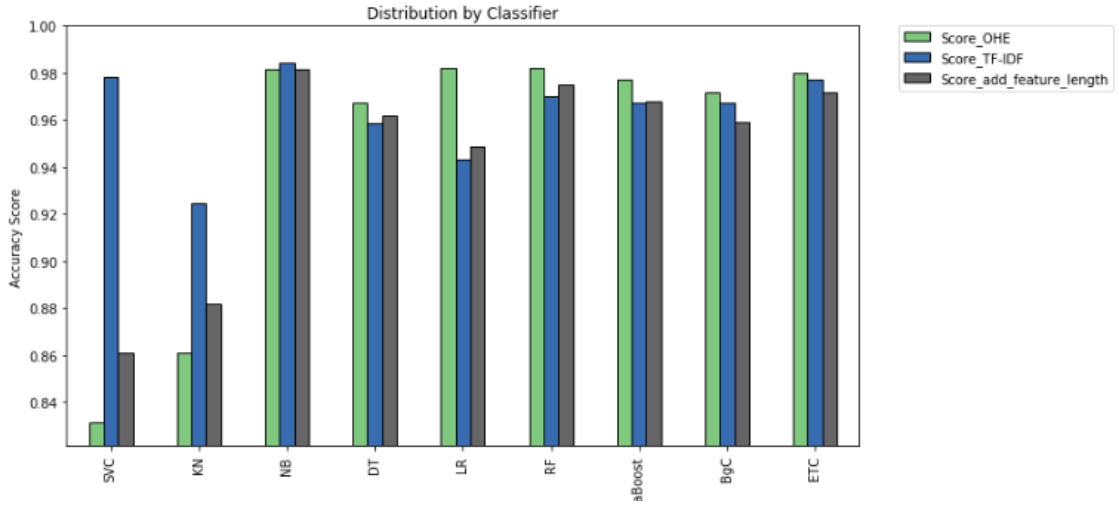


рис. 3.4

Для побудови нейромереж використовувалася бібліотека Keras з TensorFlow backend. При експериментальному порівняні нейромережевих архітектур найкраще себе показала архітектура RNN. У результаті її застосування було досягнено результату 98.6% точності. При розв'язані задачі класифікації повідомлень найкращі результати показали Naive Bayes та RNN.

Для набору даних Quora, було прийнято рішення навчати рекурентну нейронну мережу через значне збільшення набору вхідних даних.

При досліджені набору даних Quora Insincere Questions Classification було прийнято рішень використовувати різні векторні представлення слів для навчання рекурентної нейронної мережі.

Використовувалися такі представлення слів:

- GoogleNews-vectors-negative300.
- glove.840B.300d.
- paragram_300_sl999.
- wiki-news-300d-1M.

Розподіл даних у наборі питань представлено на рис 3.5.

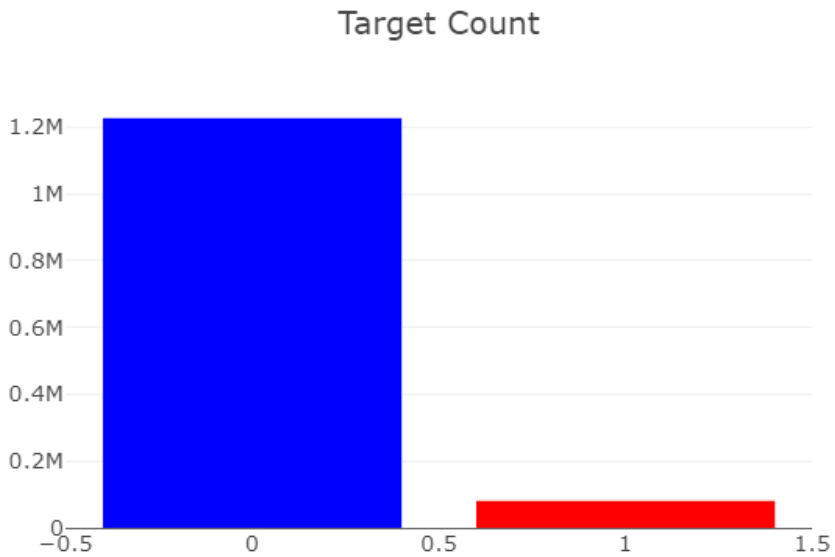


рис. 3.5

Хмара слів зазвичай використовується для опису ключових слів (тегів) на веб-сайтах, або для представлення неформатованого тексту. Ключові слова найчастіше являють собою окремі слова, і важливість кожного ключового слова позначається

розміром шрифту або кольором. Таке уявлення зручно для швидкого сприйняття найвідоміших термінів і для розподілу термінів за популярністю щодо один одного. Хмара слів для набору даних Quora Insincere Questions Classification представлена на рис. 3.6.

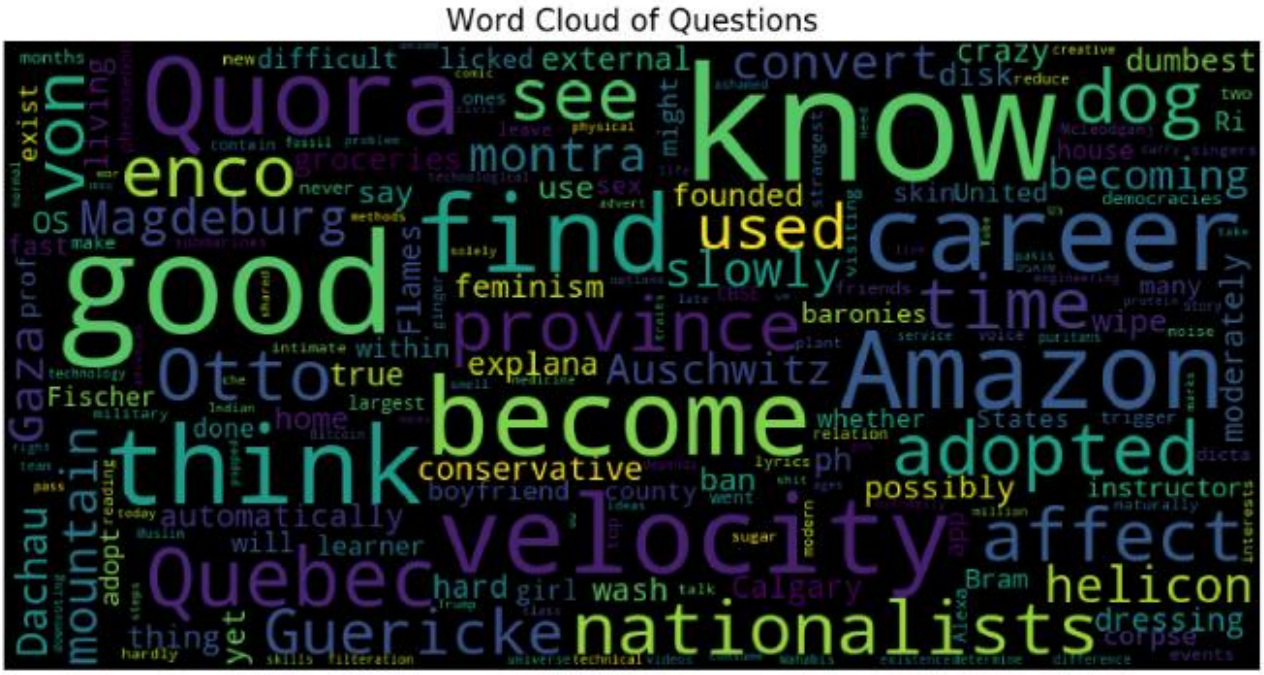


рис. 3.6

Для наочного представлення набору даних, використаємо підхід відображення n-gram які найчастіше зустрічаються у наборі. В області обробки природної мови, N-грами використовується в основному для передбачення на основі імовірнісних моделей. N-грамна модель розраховує ймовірність останнього слова N-грами, якщо відомі всі попередні. При використанні цього підходу для моделювання мови передбачається, що поява кожного слова залежить тільки від попередніх слів.

Інше застосування N-грам є виявлення плагіату. Якщо розділити текст на кілька невеликих фрагментів, представлених N-грамами, їх легко порівняти один з одним, і таким чином отримати ступінь подібності контрольованих документів. N-грами часто успішно використовуються для категоризації тексту та мови. Крім того, їх можна використовувати для створення функцій, які дозволяють отримувати знання з текстових даних. Використовуючи N-грами, можна ефективно знайти кандидатів, щоб замінити слова з помилками правопису.

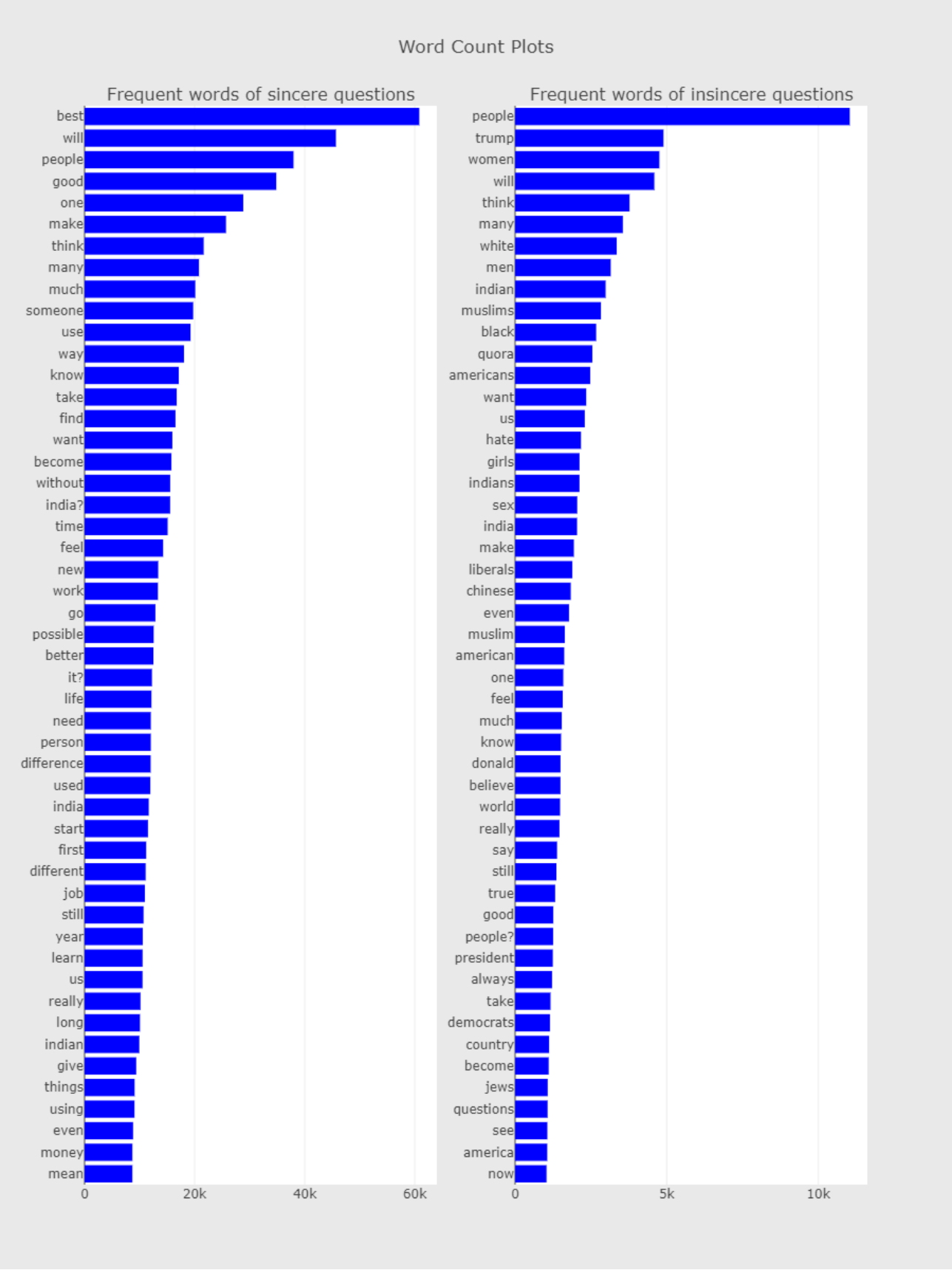


рис. 3.7

Відзначаємо, що до найпопулярніших слів увійшли слова: best, will, people, trump, women, good, think.

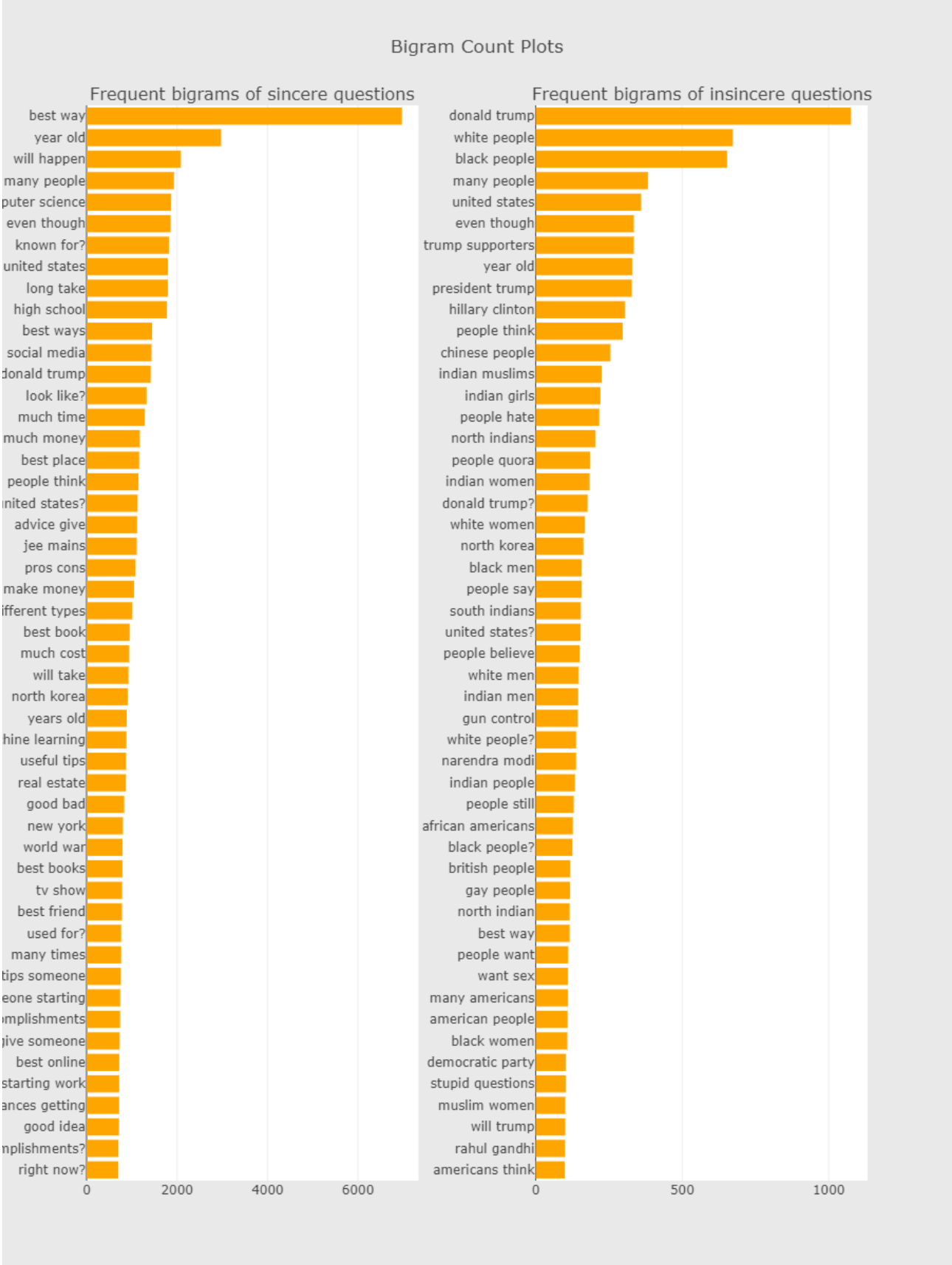


рис. 3.8

Для оцінювання якості моделі використовувалася F1 метрика. У результаті були отримані результати представлено на рис.3.9:

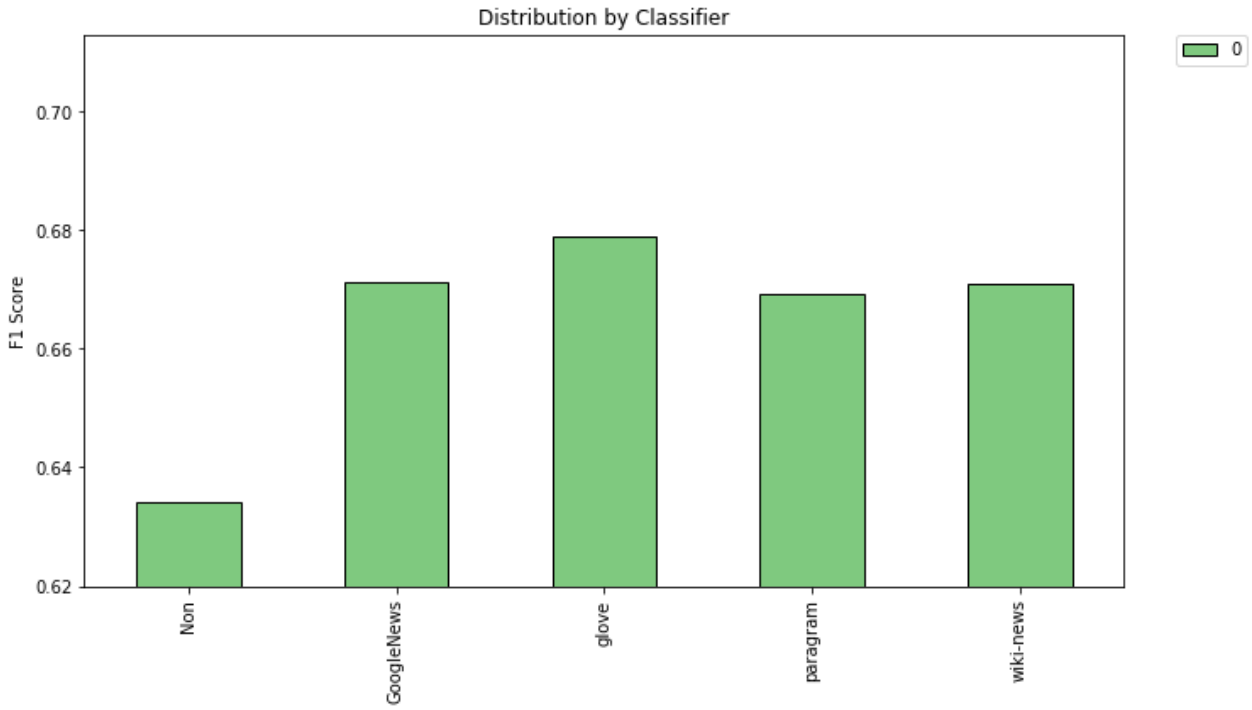


рис. 3.9

Результати підтверджують наше припущення, щодо необхідності використання векторних представленнях слів, через врахування у них семантичного значення слів. Найкращий результат продемонструвала модель, яка навчалася на векторних представленнях GLOVE.

Для набору даних Real or Not? NLP with Disaster Tweets розподіл даних представлено на рис. 3.10.

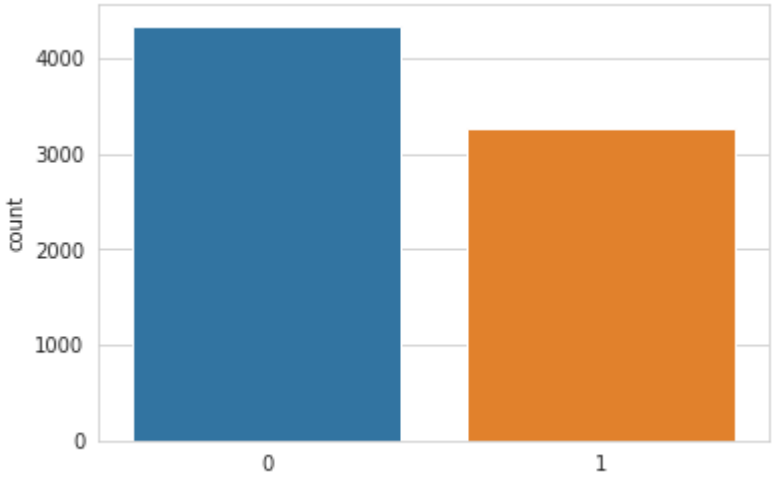


рис. 3.10

При застосуванні до цієї задачі раніше використовуваних нами алгоритмів, ми отримали такі результати:

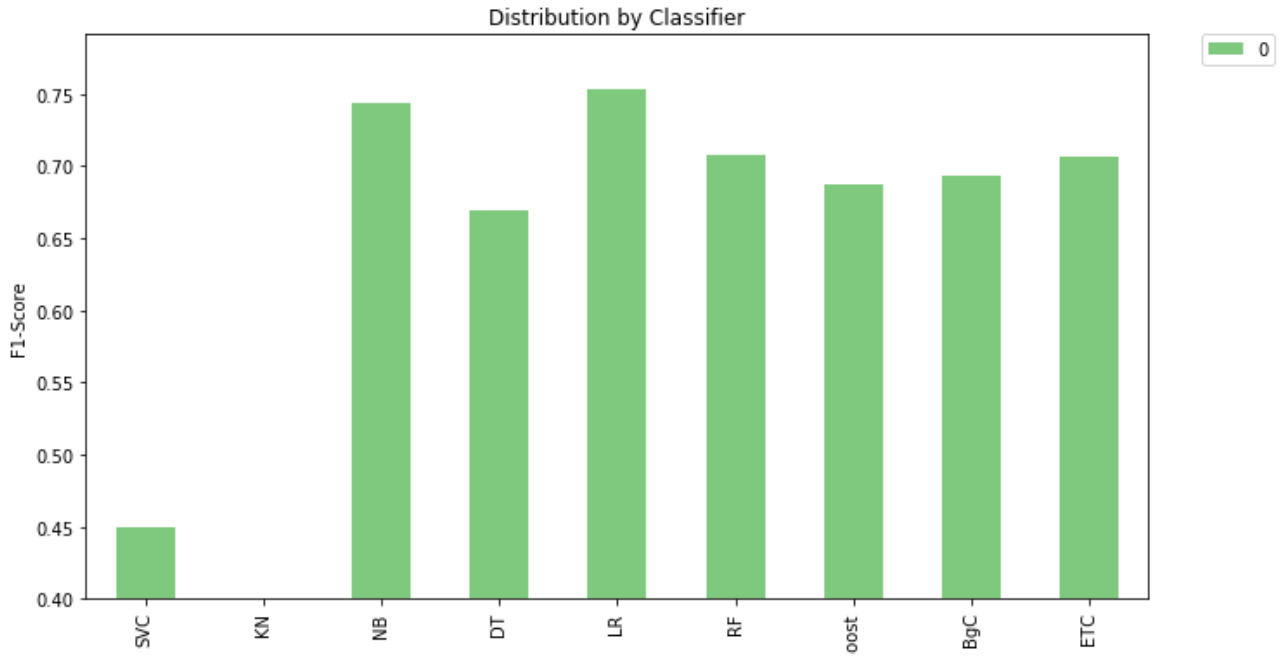


рис. 3.11

Через низьку ефективність було прийнято рішення для розв'язання цієї задачі, використати сучасний підхід, а саме тонке налаштування попередньо навченої нейронної мережі на архітектурі трансформера – BERT. Для цього використаємо бібліотеки transformers та FastAI, та підхід описаний у [6]. При застосувані вказаного підходу час навчання, незначно зріс, але було отримано результат *F1-score = 0.84*, що на 10% краще ніж в моделі з Наївним Баєсовим класифікатором, яка стала найкращою під час попереднього експерименту.

З описаних вище експериментів та їх результатів випливає:

- Вибір векторизатора та класифікатора повинен базуватися на наборі даних;

- Серед класичних класифікаторів гарні результати показали наївний баєсів класифікатор та логістична регресія;

- Побудова статистичних показників покращує результат роботи класифікаторів, але векторні представлення слів (word2vec, GLOVE) дають кращі результати;

- Застосування тонкого налаштування попередньої моделі BERT підтвердило статус state-of-the-art моделі у всіх сучасних завданнях обробки текстових даних.

Висновки до третього розділу

У цьому розділі ми спроектували інтелектуальну інформаційну систему для обробки та класифікації текстових даних. Провели порівняння сучасних середовищ розробки та мов програмування та здійснили вибір на основі цього порівняння. Продемонстрували роботу та перевірили ефективності системи на двох обраних наборах даних. При оцінювані результатів, прийшли до висновку, що ефективність системи досягає результатів систем сучасного рівня.

В ході дослідження було проаналізовано шляхи підвищення ефективності. Для оцінювання системи використовувалися оцінка точності та метрика F1. Результати було оцінено в порівнянні з таблицею результатів на ресурсі Kaggle. Для візуалізації даних використовувався підхід побудов n-gram та хмари слів.

ВИСНОВКИ

У результаті виконання магістерської роботи відповідно до об’єкта, предмета, мети та завдань дослідження нами зроблені такі висновки:

1. З’ясовано науково-теоретичний і практичний стан проблеми проектування інтелектуальних інформаційних систем. Опрацьовано сучасну вітчизняну та зарубіжну літературу з галузі дослідження. Проведено класифікацію інформаційних систем. Значну увагу приділено означенню термінів, які часто зустрічаються у нашому досліджені. Розглянуто типи інформаційних систем та задач, які вони можуть вирішувати. Досліджено сучасний стан машинного навчання, та детально досліджено його класифікацію.
2. Розроблено модель інтелектуальної інформаційної системи. Описано формалізований підхід до розробки моделей класифікатора текстових даних. Проаналізовано еволюцію методів представлення текстової інформації в галузі інтелектуального аналізу тексту від традиційних статистичних методів. Розглянуто процес моделювання таких систем із застосуванням алгоритмів машинного навчання. Розроблено модель інтелектуальної інформаційної системи. Наведено загальну схему проектування класифікатора текстових даних. Досліджено сучасний підхід до інтелектуального аналізу тексту на основі трансформаторів на прикладі попередньо навченої моделі BERT.
3. Реалізовано проект інтелектуальної інформаційної системи. Проведено порівняння сучасних середовищ розробки та мов програмування та здійснено вибір на основі цього порівняння. Розроблена модель написана на мові програмування Python 3.7, з використанням таких бібліотек, як Tensorflow, Sklearn, Pandas, Numpy, NLTK, matplotlib та іншію. Під час дослідження використовувалися такі векторні представлення слів, як word2vec та GLOVE. Продемонстровано роботу системи на двох обраних наборах даних.
4. Визначено ефективність функціонування спроектованої інтелектуальної

інформаційної системи. Прийшли до висновку, що ефективність системи досягає результатів систем сучасного рівня. Для оцінювання використовувалися метрики accuracy та F1.

До перспектив подальших досліджень відносяться використання алгоритмів підсилення градієнтного спуску на основі дерев (LightGBM, XG-Boost), та тонке налаштування попередньо навченої моделі ALBERT, ROBERTA.

У результаті виконання роботи було досягнуто поставленої мети, ти виконані всі поставлені завдання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations [Електронний ресурс] / [Z. Lan, M. Chen, S. Goodman та ін.] // arXiv. – 2019. – Режим доступу до ресурсу: https://arxiv.org/pdf/1909.11942.pdf. – (дата звернення 02.10.2019). – Назва з екрана.

2. Attention Is All You Need [Електронний ресурс] / [A. Vaswani, N. Shazeer, N. Parmar та ін.] // arXiv. – 2017. – Режим доступу до ресурсу: https://arxiv.org/pdf/1706.03762.pdf. – (дата звернення 02.10.2019). – Назва з екрана.

3. Author identification: Using text sampling to handle the class imbalance problem [Електронний ресурс] / Stamatatos E. // Information Processing and Management: an International Journal. – 2008. – № 44. – С.790–799. – Режим доступу: https://dl.acm.org/citation.cfm?id=1347518. – (дата звернення 18.09.2019). – Назва з екрана.

4. Authorship attribution [Електронний ресурс] / Juola P. // Foundations and Trends in Information Retrieval. – 2006. – № 1. – С. 233—334. – Режим доступу: https://books.google.com.ua/books/about/Authorship_Attribution.html?id=_B2zDLdqe60C&redir_esc=y. – (дата звернення 11.04.2017). – Назва з екрана.

5. Bag-of-words model [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Bag-of-words_model.
– (дата звернення 16.09.2019). – Назва з екрана.

6. Bert Fastai [Електронний ресурс] – Режим доступу до ресурсу: https://www.kaggle.com/kpriyanshu256/bert-fastai.
– (дата звернення 24.12.2019). – Назва з екрана.

7. Binder, D.A. Bayesian cluster analysis. - Biometrika, 1978. - 65, 31–38.

8. Efficient Estimation of Word Representations in Vector Space [Електронний ресурс] / T.Mikolov, K. Chen, G. Corrado, J. Dean – Режим доступу до ресурсу: https://arxiv.org/abs/1301.3781. – (дата звернення 16.09.2019). – Назва з екрана.

9. Experiments with mood classification in blog posts [Електронний ресурс] / Mishne G. // 1st workshop on stylistic analysis of text for information access. – 2005. – Режим доступу:

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.111.2693&rep=rep1&type=pdf. – (дата звернення 10.04.2019). – Назва з екрана.

10. HandyCache [Електронний ресурс] – 2018. – Режим доступу: https://ru.wikipedia.org/wiki/HandyCache. – (дата звернення 02.03.2019). – Назва з екрана.

11. Jerome H. Friedman Elements of Statistical Learning [Електронний ресурс] / Robert Tibshirani // Режим доступу:URL: https://web.stanford.edu/~hastie/Papers/ESLII.pdf – (дата звернення 16.09.2019). – Назва з екрана.

12. Kaggle. Your Home for Data Science [Електронний ресурс] – 2018. – Режим доступу: https://www.kaggle.com/. – (дата звернення 02.03.2019). – Назва з екрана.

13. Laplacian smoothing [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Laplacian_smoothing. – (дата звернення 02.03.2019). – Назва з екрана.

14. Leave-one-out cross-validation [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Cross-validation_(statistics)#Leaveone-out_cross-validation. – (дата звернення 02.03.2019). – Назва з екрана.

15. Logistic regression [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Logistic_regression. – (дата звернення 02.03.2019). – Назва з екрана.

16. MachineLearning.ru [Електронний ресурс]. – Режим доступу: http:// www.machinelearning.ru – (дата звернення 16.09.2019). – Назва з екрана.

17. Mercurial SCM [Електронний ресурс] – 2018. –Режим доступу: https://www.mercurial-scm.org/. – (дата звернення 10.03.2019). – Назва з екрана.

18. Mining newsgroups using networks arising from social behavior [Електронний ресурс] / Agrawal R., Rajagopalan S., Srikant R., Xu Y. // Proceedings of the 12th international conference on World Wide Web. – 2003. – С. 529–535. – Режим доступу: https://dl.acm.org/citation.cfm?id=775227. – (дата звернення 03.05.2019). – Назва з екрана.

19. Mitchell T. M. Machine Learning, 1 edition, New York, NY, USA: McGraw-Hill, Inc., 1997.

20. Naive Bayes classifier [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Naive_Bayes_classifier. – (дата звернення 02.03.2019). – Назва з екрана.

21. N-gram-based author profiles for authorship attribution [Електронний ресурс] / Keselj V., Peng F., Cercone N., Thomas C. – Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.9.7388&rep=rep1&type=pdf. – (дата звернення 11.10.2019). – Назва з екрана.

22. Owen Kaser and Daniel Lemire, Tag-Cloud Drawing: Algorithms for Cloud Visualization. In proceedings of Tagging and Metadata for Social Information Organization (WWW 2007) [Електронний ресурс] / https://arxiv.org/pdf/cs/0703109.pdf - (дата звернення 17.08.2019). – Назва з екрана.

23. Pennington J. GloVe: Global Vectors for Word Representation [Електронний ресурс] / J. Pennington, R. Socher, C. Manning – Режим доступу до ресурсу: http://www.aclweb.org/anthology/D14-1162.
– (дата звернення 16.09.2019). – Назва з екрана.

24. Proxomitron [Електронний ресурс] – 2018. – Режим доступу: https://ru.wikipedia.org/wiki/Proxomitron. – (дата звернення 02.03.2019). – Назва з екрана.

25. Python (programming language) [Електронний ресурс] – 2018. – Режим доступу: https://ru.wikipedia.org/wiki/Python. – (дата звернення 02.03.2019). – Назва з екрана.

26. Quantitative authorship attribution: An evaluation of techniques [Електронний ресурс] / Grieve J. // Literary and Linguistic Computing. – 2005. – №22. – С.251-270. – Режим доступу: https://academic.oup.com/dsh/articleabstract/22/3/251/951481?redirectedFrom=fulltext. – (дата звернення 05.10.2018). – Назва з екрана.

27. Quora Insincere Questions Classification [Електронний ресурс] – Режим доступу до ресурсу: https://www.kaggle.com/c/quora-insincere-questions-classification. – (дата звернення 16.09.2019). – Назва з екрана.

28. R (programming language) [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/R_(programming_language). – (дата звернення 02.03.2019). – Назва з екрана.

29. Real or Not? NLP with Disaster Tweets [Електронний ресурс] – Режим доступу до ресурсу: https://www.kaggle.com/c/nlp-getting-started.
– (дата звернення 20.12.2019). – Назва з екрана.

30. Rule-based approach to sentiment analysis at ROMIP 2011 [Електронний ресурс] / Kan D. // AlphaSense Inc.– 2011. – Режим доступу: http://www.dialog-21.ru/media/1393/138.pdf. – (дата звернення 16.11.2019). – Назва з екрана.

31. Russell S. Artificial Intelligence: A Modern Approach / S. Russell, P. Norvig., 2009. – 1 c. – (Third Edition).

32. SMS Spam Collection Dataset [Електронний ресурс] – Режим доступу до ресурсу: https://www.kaggle.com/uciml/sms-spam-collection-dataset.
– (дата звернення 16.09.2019). – Назва з екрана.

33. Support function [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Support_function. – (дата звернення 16.09.2019). – Назва з екрана.

34. Support vector machine [Електронний ресурс] – 2018. – Режим доступу: https://en.wikipedia.org/wiki/Support_vector_machine. – (дата звернення 02.03.2019). – Назва з екрана.

35. TF-IDF [Електронний ресурс] – 2018. – Режим доступу: https://uk.wikipedia.org/wiki/TF-IDF. – (дата звернення 02.11.2019). – Назва з екрана.

36. Twitter sentiment analysis: The good the bad and the omg! [Електронний ресурс] / Kouloumpis E., Wilson T., Moore J. // Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media. – 2011. – С. 538–

541. – Режим доступу: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/download/2857/3251. – (дата звернення 10.09.2019). – Назва з екрана.

37. Using kullback-leibler distance for text categorization [Електронний ресурс] / Bigi B. // Proceedings of the 25th European conference on IR research. – 2003 – С. 305–319. – Режим доступу: https://goo.gl/igNQ5P – (дата звернення 11.01.2019). – Назва з екрана.

38. Аналіз методів машинного навчання для прогнозування відтоку клієнтів [Електронний ресурс] / В. В. Уштаніт, А. А. Папа, А. А. Яровий, О. П. Прозор // Матеріали XLVII науково-технічної конференції підрозділів ВНТУ. – Режим доступу: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2018/paper/view/5317. – (дата звернення 10.10.2019). – Назва з екрана.

39. Грицунов О. В. Інформаційні системи та технології. Навчальний посібник. — Х.: ХНАМГ, 2010. — 222 с.

40. Коцовський В. М. Методи та системи штучного інтелекту / В. М. Коцовський. – Ужгород, 2016. – 75 с.

41. Лекции по логическим алгоритмам классификации [Електронний ресурс] / Воронцов К. В. – 2010. – С. 17-21. – Режим доступу: http://www.machinelearning.ru/wiki/images/3/3e/Voron-ML-Logic.pdf – (дата звернення 17.09.2019). – Назва з екрана.

42. Маслов В.П. Інформаційні системи і технології в економіці : Посібник для студ. вузів/ В.П. Маслов; М-во освіти і науки України. -К.: Слово, 2005. -263 с.

43. Математические методы обучения по прецедентам (теория обучения машин) [Електронний ресурс] / Воронцов К. В. // Курс лекций «Машинное обучение» – Режим доступу: http://www.machinelearning.ru/wiki/images/6/6d/Voron-ML-1.pdf. – (дата звернення 11.09.2019). – Назва з екрана.

44. Машинне навчання. Типи навчання. [Електронний ресурс]. – Режим доступу: https://courses.prometheus.org.ua/courses/IRF/ML101/2016_T3/about – (дата звернення 16.09.2019). – Назва з екрана.

45. Методи та засоби мультимедійних інформаційних систем : навч. посіб. / Т. М. Басюк, П. І. Жежнич; Нац. ун-т "Львів. політехніка". - Львів : Вид-во Львів. політехніки, 2015. - 426 с. - Бібліогр.: с. 413-416.

46. Науменко Т.О. Використання онтологій для зберігання та обміну занннями в мультиагентних системах / Міжнародний науковий журнал «Інтернаука» / № 6 (28), 2017. – с.50-52.

47. Николенко С. Глубокое обучение. Погружение в мир нейронных сетей / С. Николенко, Е. Кадурин, Е. Архангельская. – Питер: СПб, 2018. – 480 с.

48. Общий взгляд на машинное обучение: классификация текста с помощью нейронных сетей и TensorFlow [Електронний ресурс] – Режим доступу: https://tproger.ru/translations/text-classification-tensorflow-neural-networks.
– (дата звернення 10.09.2019). – Назва з екрана.

49. Определение авторства тексту с использованием буквенной и грамматическй информации [Електронний ресурс] / Кукушкина О. В., Поликарпов А. А., Хмелев Д. В. // Проблемы передачи информации. – 2001. – №2. – С. 96–109. – Режим доступу:
http://www.philol.msu.ru/~lex/articles/grco_r.htm. – (дата звернення 01.04.2019).
– Назва з екрана.

50. Основи інформаційних систем : Навч. посіб./ Віктор Ситник, Тамара Писаревсь-ка, Ніна Єрьоміна та ін.; За ред. В.Ф.Ситника; М-во освіти України. Київський нац. еко-ном. ун-т. -2-е вид., перероб. і доп.. -К.: КНЕУ, 2001. - 420 с.

51. Персептроны [Електронний ресурс] / Минский М., Пейперт С. – 1971. – Режим доступу: https://sheba.spb.ru/delo/perseptrony-1969.htm – (дата звернення 17.09.2017). – Назва з екрана.

52. Рекомендательные системы: SVD, часть I [Електронний ресурс] / Николенко С. // Блог компании Surfingbird. – 2012. – Режим доступу: https://habr.com/company/surfingbird/blog/139863/ – (дата звернення 17.08.2019). – Назва з екрана.

53. Рекомендательные системы: теорема Байеса и наивный байесовский классификатор [Електронний ресурс] / Николенко С. // Блог компании Surfingbird. – 2012. – Режим доступу: https://habr.com/company/surfingbird/blog/150207/ – (дата звернення 18.08.2019). – Назва з екрана.

ДОДАТКИ

Додаток А

Національна академія педагогічних наук України
Інститут інформаційних технологій
і засобів навчання

СЕРТИФІКАТ

учасника Всеукраїнського методологічного семінару
для молодих учених
«Інформаційно-комунікаційні технології в освіті
та наукових дослідженнях»
виданий

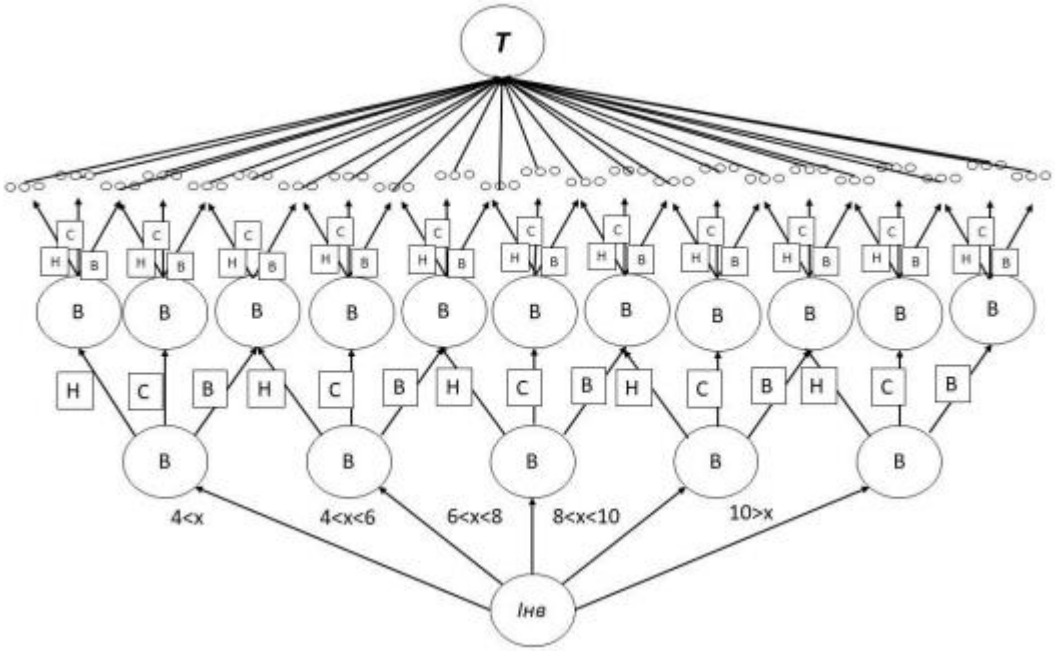
Бойчуку Дмитру Юрійовичу

Директор,
заслужений діяч науки
і техніки України,
дійсний член НАПН України,
доктор технічних наук,
професор — В. Ю. Биков

м. Київ
6 грудня 2018 року

Видавництво «Преса України». Тел. (044) 454-81-70

Додаток Б



Додаток В