Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту (роботи)

магістр
(освітній рівень)

на тему: *«Визначення авторства документу з допомогою методів інтелектуального аналізу тексту»*

Виконав: студент (ка) *VI* курсу, групи *СБм-61*

Спеціальності:
125 «Кібербезпека»
(шифр і назва напряму підготовки, спеціальності)

Вівчарик В. М.
підпис (прізвище та ініціали)

Керівник *Карпінський М.П.*
підпис (прізвище та ініціали)

Нормоконтроль *Кареліна О.В.*
підпис (прізвище та ініціали)

Рецензент
підпис (прізвище та ініціали)

м. Тернопіль – 20*19*

АНОТАЦІЯ

Визначення авторства документу з допомогою методів інтелектуального аналізу тексту// Дипломна робота ОР «Магістр» // Вівчарик Володимир Михайлович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБм-61 // Тернопіль, 2019 // С. 111, рис. – , табл. – , кресл. – , додат. – .

Ключові слова: ІДЕНТИФІКАЦІЯ, ВИЗНАЧЕННЯ АВТОРА, МАШИННЕ НАВЧАННЯ, КЛАСИФІКАЦІЯ, НАЇВНИЙ КЛАСИФІКАТОР БАЙЄСА, СТОП-СЛОВА, СТЕМІНГ, ЛЕМАТИЗАЦІЯ.

Метою роботи – здійснити програмну реалізацію алгоритму мульти-класифікації для визначення автора невідомого текстового документу на підставі наявної прокласифікованої навчальної вибірки, дослідити вплив параметрів класифікаційної моделі та методів нормалізації текстових документів на точність класифікації.

Основні результати роботи: в роботі досліджено основні завдання та можливі сфери застосування задачі визначення авторства деякого документу, обґрунтовано вибір моделі класифікації та програмного середовища Python для практичної реалізації методу визначення автора документу. Проведено тестування імплементованої класифікаційної моделі наївного Байєса для реальних даних, здійснено порівняння основних показників точності моделі для різного розміру простору ознак, проведено аналіз впливу на точність різних методів нормування текстових документів для задач класифікації.

У першому розділі описується, проблема визначення авторства текстового документу, її підвиди та можливі застосування для практичних задач. Наводиться також класифікація методів дослідження стилю написання

текстів та короткі теоретичні відомості про базові методи класифікації, які можуть бути використані для ідентифікації автора документу.

У другому розділі розглядаються особливості побудови та оцінки якості класифікаційних моделей визначення авторства документу.

Третій розділ – практична частина. У ньому описано деталі налаштування необхідних бібліотек Python, наведено лістинги основних етапів алгоритмів підготовки, опрацювання, нормалізації даних та власне класифікації. Проведено оцінку точності для різної розмірності простору ознак та різних методів нормалізації тексту.

В четвертому розділі наведено основні типи та структури даних Python, які використовувались в практичній реалізації запропонованої моделі.

У розділі "Обґрунтування економічної ефективності" обчислено собівартість та термін окупності проекту.

У шостому розділі описано інструкції з охорони праці при роботі з комп’ютером та фактори виробничого середовища і їх вплив на життєдіяльність людини.

В розділі "Екологія" описано питання зниження енергоємності та енергозбереження та індексний метод в екології.

У результаті виконання дипломної роботи обґрунтовано класифікаційну модель у вигляді моделі наївного Байєса, яку реалізовано у вигляді програмного забезпечення на мові програмування Python, проведено експериментальну оцінку точності побудованої моделі для різної кількості вхідних атрибутів, здійснено оцінку методів нормалізації текстових документів.

ANNOTATION

A document authorship identification using the text mining methods // Thesis of the Master degree // Vivcharyk Volodymyr // Ternopil Ivan Puluj National Technical University, Department of Computer Information Systems and Software Engineering, Department of Cybersecurity // Ternopil, 2019 // P. 111, Tables – , Fig. – , Diagrams – , Annexes. – , References – .

Keywords: IDENTIFICATION, AUTHOR DETERMINING PROBLEM, MACHINE LEARNING, CLASSIFICATION, NAIVE BAYES CLASSIFIER, STOP-WORDS, STEMMING, LEMMATIZATION.

The purpose of the work is to implement a software of the multi-classification algorithm to identify the author of an unknown text document on the basis of an available classified training dataset; to research the influence of the parameters of the classification model and methods of normalization of text documents on the accuracy of classification.

Main results of the thesis: the main tasks and possible application areas of author of a document determining problem are investigated; the choice of the classification model and the Python software environment for practical implementation of the method of the document authorship identification is substantiated. Tests of the implemented naive Bayes classification model of for real data were carried out. The basic criteria of model accuracy for different size of feature vector space were compared, the influence of different methods of text documents normalization on accuracy for classification problems was analyzed.

The first section describes the problem of authorship of a text document identification, its types, and possible applications for practical tasks. It also provides a classification of text-writing style research methods and brief theoretical

information on basic classification methods that can be used to identify the author of a document.

The second section discusses the peculiarities of constructing and evaluating the quality of classification models for document authorship identification.

The third section is the practical part. It describes the details of setting up the required Python libraries, code listings of the basic steps of the algorithms for preparation, processing, data normalization and classification are provided. Accuracy for different dimensions of feature space and different methods of text normalization has been estimated .

The fourth section describes the basic types and structures of Python data that were used in the practical implementation of the proposed model.

In the section "Justification of economic efficiency" the cost and payback period of the project are calculated.

The sixth section describes safety instructions while operating computer and the factors of the work environment and their impact on human life.

The section "Ecology" describes the issues of reducing energy consumption and energy conservation and the index method in ecology.

As a result of the diploma thesis, a classification model as a naive Bayes model has been implemented in the form of software in Python programming language, the experimental estimation of the accuracy of the constructed model for different number of input attributes has been performed, the methods of normalization of text documents have been evaluated.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

DM – Data Mining (інтелектуальний аналіз даних)

KNN- K-nearest neighbors (метод k-найближчих сусідів)

ML -Machine Learning (машинне навчання)

NLTK - Natural Language Toolkit (інструмент обробки природньої мови)

SVM – Support Vector Machine (метод опорних векторів)

ШІ – штучний інтелект

ВСТУП

Як відомо, проблема аутентифікації повідомлення, тобто підтвердження автора повідомлення, є одним з наріжних каменів криптографії, яка традиційно розв'язується алгоритмами цифрового підпису. Проте існує кардинально інша проблема ідентифікації та встановлення автору документа шляхом аналізу контенту документу. Ця проблема є менш поширеною, проте не менш важливою, адже задача ідентифікації автора розв'язується для боротьби з плагіатом, встановлення авторства анонімних текстів, програмного коду, шкідливого програмного забезпечення, експертизи та встановлення особистості в криміналістиці, запобігання злочинів та багатьох інших застосувань.

Дослідження "авторського стилю" може здійснюватися на різних рівнях: пунктуаційному, орфографічному, синтаксичному, лексико-фразеологічному та стилістичному. Найбільший інтерес дослідників представляє аналіз трьох останніх рівнів. Існує доволі багато методів аналізу стилю. В цілому їх можна розділити на дві групи: експертні та формальні. Експертні методи аналізу є доволі трудомісткими. Формальні методи базуються на алгоритмах статистичного аналізу, машинного навчання, нейронних мереж, Text mining та ін. Останнім часом зростає популярність методів вбудовування в мережі (embedded networks). Інформація моделюється як мережа, що складається з вузлів та зв'язків між ними. Зважаючи на широкий спектр методів та труднощів в задачах визначення авторства, актуальною науково-практичною задачею є аналіз та практична реалізація цих методів, що можуть бути використані для створення автоматизованих систем визначення авторства.

Отже, метою роботи є створення програмного забезпечення для визначення автора документу на основі обґрунтованої моделі мульти-класифікації, дослідження точності моделі для різного розміру простору

векторних ознак та визначення впливу на точність прогнозування різних попередніх етапів для нормалізації вхідних даних.

Для досягнення поставленої мети, необхідно виконати низку наступних задач:

- провести огляд літературних джерел в області дослідження;
- дослідити основні завдання та сфери практичного застосування задачі визначення авторства;
- обґрунтувати вибір моделі для задачі мульти-класифікації визначення авторства;
- сформувати навчальні та тестові набори реальних даних;
- розробити програмне забезпечення для реалізації обраної моделі;
- провести порівняльний аналіз точності класифікаційної моделі для різної розмірності простору ознак;
- дослідити вплив на якість прогнозної моделі різних методів нормалізації текстових документів.

Об'єкт дослідження – процес визначення авторства для текстових документів.

Предмет дослідження – моделі та методи для ідентифікації автора тексту.

Методи дослідження: загальнонаукові методи пізнання як порівняльний та системний аналіз, методи машинного навчання та математичної статистики

Наукова новизна. В роботі проведено порівняльний аналіз точності прогнозування автору документу в залежності від частки обраних ключових слів від загальної кількості слів та досліджено вплив на точність таких методів нормування тексту, як видалення стоп-слів та стемінгу.

Апробація результатів роботи. Окремі результати роботи доповідались на VII науково-технічній конференції «Інформаційні моделі, системи та технології», Тернопіль, ТНТУ, 11 – 12 грудня 2019 р.

1 ФОРМАЛЬНІ МЕТОДИ АНАЛІЗУ ДАНИХ, НА ЯКИХ БАЗУЄТЬСЯ ВИЗНАЧЕННЯ АВТОРСТВА

1.1 Основні завдання та застосування науки про визначення авторства

Мистецтво та наука розмежовування авторів за авторським стилем їх письма шляхом ідентифікації характеристик особистості авторів на основі вивчення авторських статей називаються аналіз авторства. Авторський стиль -власний письмовий стиль автора, який підсвідомо використовується автором при написанні текстів. Стилеметрія –це вивчення мовного стилю, як правило, до письмової мови, що спирається на те, що кожен автор має незамінні письмові особливості, які не можна імітувати.

Основними завданнями визначення авторства є:

- Ідентифікація автора (Author Attribution or Identification) - це задача знаходження найбільш ймовірного автора статті, тексту чи документу [9]. В термінах машинного навчання це задача мульти-класифікації, покликана дати відповідь на запитання, хто з даної множини можливих авторів написав заданий текст.

- Перевірка авторства (Author Verification) – дає відповідь на запитання, чи був даний текст написаний певним автором чи ні [5]. Фактично, це задача бінарної класифікації, що розв’язується для того, щоб визначити наскільки висока ймовірність написання визначеним автором заданого тексту.

- Визначення профілю автора (Author Profiling) Це ідентифікація параметрів особи (для прикладу статі, віку, рідної мови, когнітивних психологічних ознак, соціально-культурного рівня) шляхом вивчення "наявної інформації" в її текстах [1,8]. Це проблема мульти-класової класифікації а також і кластеризації.

- Виявлення плагіату або знаходження подібності між двома текстами (Similarity detection). Це задача визначення подібності між двома текстами, при цьому необов'язково з визначенням та вказанням авторства [13].
- Виявлення стилістичних невідповідностей – стосується пошуку відмінностей у написанні спільної роботи.

Незважаючи на те, що визначення авторства є вузькоспеціалізованою задачею, проте воно має низку практичних застосувань:

- Цивільне право (вирішення суперечок щодо авторства творів, програмного коду).
- Розвідка (наприклад, присвоєння повідомлень відомим терористам)
- Кримінальне право (наприклад, виявлення авторів образливих чи погрожуючих повідомлень).
- Комп’ютерна криміналістика (підроблені профілі в соціальних мережах, фальшиві огляди ботів, визначення розробників шкідливого програмного забезпечення.
- Літературні дослідження (наприклад, приписування анонімних літературних творів відомим авторам).
- Вивчення ринку споживачів. (вивчення характеристик автора анонімних відгуків. Завдання аналізу авторства допомагають створити профіль споживача, виявити підроблені відгуки та провести сегментацію клієнтів).
- Експертиза психічного здоров'я.

1.2 Історія виникнення науки стилеметрії

Вперше дослідження стилю тексту з метою атрибуції було зроблено ще в XV столітті. Італійський філолог Лоренцо Валла опублікував трактат «Міркування про фальшивість так званої дарчої грамоти Костянтина», в якому

на основі різних, у тому числі стилістичних критеріїв доводилося, що даний текст є підробкою.

Історія сучасної статистичної стилістики починається в середині XIX ст., коли англійський математик Аугустус де Морган в 1851 р висловив припущення, що різні автори можуть бути визначені за допомогою прихованих статистичних характеристик. Розглядаючи проблеми грецької прози, Морган стверджував, що середня довжина слів у творі автора може бути характерною рисою авторського стилю. Проте, наскільки нам відомо, сам де Морган ніяких обчислень не робив.

У середині XIX ст. також існувала група вчених, що розробляє так званий метод «стилеметрики» (Ф.Г. Фріарі, Дж. К. Інграм, Ф.У. Фурнівал). Вони підраховували кількість повторень певного слова і зміну розміру у віршах. Головним результатом їх роботи було відкриття повільного, але постійного зміни стилю Шекспіра протягом 22-х років.

Термін «стилеметрія» був винайдений німецьким філологом Вільгельмом Діттенбергером (1880), який зробив спробу вирішити проблему атрибуції та хронології діалогів Платона. Він досліджував частоту вживання слів, особливо службових, в текстах Платона, реалізація яких не залежить від тематики тексту. Пізніше його дослідження на різних матеріалах продовжили Є. Зеллер (1887), Ф. Чаду (1901), Ц. Ріттер (1903).

У 20-і рр. XX ст. можна назвати тільки кілька серйозних дослідників по стилестатистиці, таких як Р.Е. Паркер (1925), З.Е. Чендлер (1928), М. Перрі (1928), і, особливо, А. Бусман (1925), автора так званого співвідношення дієслово-прикметник.

У 30-і рр. XX ст. було зроблено новий крок у застосуванні статистичних методів у стилістиці такими лінгвістами, як Дж. В. Флетчер (1934), який розглядав розвиток стилю Спенсера, Г.М. Боллінг (1937), з критичним есе по статистичному дослідженню мови Гомера, Дж. Б. Керролл (1938), що піднімав проблему різноманітності словника, і У.Г. Юл (1938), перший досліджував дистрибуцію довжини пропозицій як статистичну характеристику стилю.

Саме з нього починається застосування сучасних статистичних методів у стилістиці. З цього періоду застосування статистичних методів у дослідженні стилю поширюється по всьому світу. Різко зріс інтерес до статистичної лінгвістиці, особливо в 1960-70 рр. (Дж. Б. Керролл, Г. Хердан, Х.Х. Сомерс, Ч. Мюллер, Б. Келман, Л.Т. Милик, Дж. Містрік, Л. Долежела, К.Б. Вільямс, Б.Н. Головін , Й. Краус, М.Н. Кожина та ін.). Саме в цей період виникають і розвиваються різноманітні ідеї аналізу авторського стилю.

З появою комп'ютерів у 60-ті рр. XX ст. стало можливим їх застосування в лінгвістичних дослідженнях, а отже і нових методів у вирішенні проблеми авторства.

1.3 Класифікація методів дослідження стилю написання текстів

Оскільки задача визначення авторства документу зустрічається в різних областях і представляє інтерес не лише для філологів чи літературознавців, а й для істориків, криміналістів, юристів, то дослідження існуючих та створення нових методів визначення авторства є актуальною науковою задачею.

Дослідження "авторського стилю" може здійснюватися на різних рівнях: пунктуаційному, орфографічному, синтаксичному, лексико-фразеологічному та стилістичному. Найбільший інтерес дослідників представляє аналіз трьох останніх рівнів.

Існує доволі багато методів аналізу стилю. В цілому їх можна розділити на дві групи: експертні та формальні. Експерти можуть ідентифікувати авторів невідомого тексту або визначити належність твору іншу автору за допомогою характерних мовних особливостей, стилістичних прийомів. Проте експертні методи аналізу є доволі трудомісткими. Зважаючи на можливості аналізу з використанням інформаційних технологій, розглянемо більш детально формальні методи аналізу тексту, що можуть бути використанні для створення автоматизованих систем визначення авторства.

Загалом сучасна проблема визначення авторства документу лежить на перетині на перетині штучного інтелекту, машинного навчання, глибокого навчання, психології, лінгвістики та інших наук (рис. 1.1)

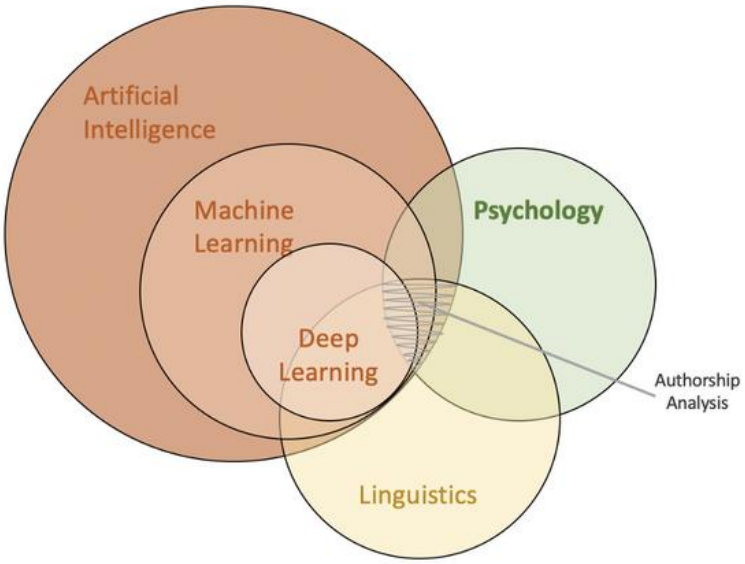

Рисунок 1.1 – Місце задачі ідентифікації автора в розрізі інших суміжних наук.

Формальні методи базуються на алгоритмах статистичного аналізу, машинного навчання, нейронних мереж, алгоритмах Text mining, та ін. [7] Останнім часом зростає популярність методів вбудовування в мережі (embedded networks). Розглянемо більш детально основні ідеї та завдання алгоритмів кожної з галузей, на яких базуються сучасні методи ідентифікації автора.

1.4 Штучний інтелект

Штучний інтелект (ШІ) - це людино-подібний інтелект, що демонструється машинами або програмним забезпеченням. Це також окрема галузь науки. Основні дослідники та підручники ШІ визначають поле як "вивчення та проектування інтелектуальних агентів", де інтелектуальний агент - це система, яка сприймає своє оточення та вживає дій, що максимізують його шанси на успіх. Джон Маккарті, який ввів термін ШІ у 1955 році, визначає

його як "науку та інженерію виготовлення розумних машин". Дослідження ШІ є високотехнологічними та спеціалізованими та можуть бути розділені на напрями, які часто не в змозі взаємодіяти між собою. Частково це пов'язано із соціальними та культурними факторами: деякі алгоритми виросли навколо певних установ та роботи окремих дослідників. Дослідження ШІ також розділені на кілька технічних питань. Деякі напрями зосереджуються на вирішенні конкретних проблем. Інші акцентують увагу на одному з кількох можливих підходів, або на використанні певного інструменту, або на виконанні конкретних застосувань. До центральних проблем (або цілей) дослідження ШІ належать міркування, знання, планування, навчання, обробка природної мови (спілкування), сприйняття та здатність рухати та маніпулювати предметами. Загальний інтелект (або "сильний ШІ") все ще серед довгострокових цілей галузі. В даний час популярними підходами є статистичні методи, обчислювальна розвідка та традиційний символьний ШІ. Існує велика кількість інструментів, що використовуються в ШІ, включаючи версії пошуку та математичної оптимізації, логіки, методи, засновані на ймовірності та економічності, та багато інших. Галузь ШІ є міждисциплінарною, яка знаходиться на перетині низки наук і професій, включаючи інформатику, психологію, лінгвістику, філософію та нейронауку, а також інші спеціалізовані галузі, такі як штучна психологія.

Галузь була заснована на твердженні, що центральна властивість людини, інтелект - sapience Homo sapiens - "може бути настільки точно описана, що можна створити машину для його симуляції". Це піднімає філософські питання про природу розуму та етику створення штучних істот, наділених людиноподібним інтелектом, питання, які з античності розглядалися міфом, художньою літературою та філософією. Штучний інтелект був предметом величезного оптимізму, але також зазнав приголомшливих невдач. Сьогодні він став важливою частиною технологічної галузі, забезпечуючи підґрунтя для багатьох найскладніших проблем інформатики.

За 50 років досліджень ШІ розробив велику кількість інструментів для вирішення найскладніших проблем інформатики. Кілька найбільш загальних із цих методів - це нейромережа, пошук та оптимізація, логіка, теорія управління, машинне навчання.

1.4.1 Нейронна мережа

Вивчення штучних нейронних мереж розпочалося в десятиліття до того, як було засновано дослідницьку галузь ШІ у роботі Вальтера Пітса та Уоррена Маккаллоу. Іншими важливими ранніми дослідниками були Френк Розенблат, який винайшов персептрон, та Пол Вербос, який розробив алгоритм зворотного поширення помилки .

Основними категоріями мереж є ациклічні або нейронні мережі прямого поширення (де сигнал проходить лише в одному напрямку) і рекурентні нейронні мережі (які дозволяють отримати зворотний зв'язок). Серед найпопулярніших мереж прямого поширення- перцептрони, багатошарові перцептрони та мережі радіальних базисних функцій. Серед рекурентних мереж найвідомішою є мережа Хопфілда, форма аттракторної мережі, яку вперше описав Джон Хопфілд у 1982 р. Нейронні мережі можна застосувати до проблеми інтелектуального управління (для робототехніки) або навчання, використовуючи такі прийоми, як Геббійське навчання та конкурентне навчання.

1.4.2 Пошук та оптимізація

Багато проблем ШІ можна вирішити теоретично шляхом розумного пошуку багатьох можливих рішень: логічне обґрунтування може бути зведене до виконання пошуку. Наприклад, логічне підтвердження можна розглядати як пошук шляху, який веде від припущень до висновків, де кожним кроком є застосування правила логічного виведення. Планування алгоритмів пошуку через дерева цілей і підцілей, спроба знайти шлях до кінцевої мети, процес, який називається аналізом "кінцевих цілей". Алгоритми робототехніки для

переміщення кінцівок та захоплення об’єктів використовують локальний пошук у просторі конфігурації. Багато алгоритмів навчання використовують алгоритми пошуку, засновані на оптимізації. Простий пошук методом перебору є рідко успішним для більшості проблем у реальному світі: простір пошуку (кількість місць для пошуку) швидко зростає до астрономічних чисел. Результатом є пошук, який занадто повільний або ніколи не завершується. Рішення для багатьох проблем полягає у використанні "евристики" або "великих" правил, які усувають вибір, який навряд чи призведе до досягнення мети (так звана "обрізка дерева пошуку"). Евристика забезпечує програму «найкращою здогадкою» для шляху, на якому лежить рішення. Евристика обмежує пошук рішень меншим розміром вибірки.

Зовсім інший вид пошуку став відомим у 90-х роках, що базується на математичній теорії оптимізації. Для багатьох проблем можна починати пошук з якоїсь форми здогадки, а потім поступово уточнювати здогадки до тих пір, поки подальші уточнення будуть неможливими. Ці алгоритми можна візуалізувати як сліпе сходження на пагорб: ми починаємо пошук у випадковій точці пейзажу, а потім, стрибками чи кроками, продовжуємо рухати здогадки в гору, поки не досягнемо вершини. Інші алгоритми оптимізації - це імітація відпалу, пошук променя та випадкова оптимізація.

1.4.3 Логіка

Логіка використовується для представлення знань та вирішення проблем, але вона може бути застосована і до інших проблем. Наприклад, satplan алгоритм використовує логіку для планування, а індуктивне логічне програмування як метод навчання. У дослідженнях ШІ використовується кілька різних форм логіки. Пропозиційна або сентенційна логіка - це логіка висловлювань, яка може бути істинною або помилковою. Логіка першого порядку також дозволяє використовувати квантори та предикати і може висловлювати факти про об'єкти, їх властивості та відносини один з одним. Нечітка логіка - це версія логіки першого порядку, яка дозволяє представити

істинність твердження як значення між 0 і 1, а не просто True (1) або False (0). Нечіткі системи можна використовувати для невизначених міркувань, вони широко застосовуються в сучасних системах контролю промислових і споживчих товарів. Суб'єктивна логіка моделює невизначеність по-іншому і більш чітко, ніж нечітка логіка: пене біноміальне твердження за задовольняє рівняння довіра + недовіра + невизначеність = 1 у бета-розподілі. За цим методом невігластво можна відрізнити від імовірнісних тверджень, які агент робить з високою впевненістю.

Логіка за замовчуванням, немонотонна логіка та обмеження опису - це форми логіки, розроблені для допомоги у міркуванні за замовчуванням та кваліфікаційній проблемі. Кілька розширень логіки були розроблені для обробки конкретних областей знань, таких як: логіка опису; обчислення ситуації, обчислення подій та текуче числення (для відображення подій та часу); каузальне числення; обчислення довіри; і модальна логіка.

1.4.4 Теорія управління

Теорія управління, онука кібернетики, має багато важливих застосувань, особливо в робототехніці. Тут доречно коротко прокоментувати значення слова "інтелектуальний" у "інтелектуальному управлінні". Зауважимо, що точне визначення поняття "інтелект" ухиляється від людства вже тисячі років. Останнім часом цим питанням займалзаймалися такі дисципліни, як психологія, філософія, біологія та, звичайно, штучний інтелект (ШІ); зауважимо, що ШІ визначається як вивчення розумових здібностей за допомогою обчислювальних моделей. Досі не існує єдиної думки щодо того, що становить інтелект. Суперечка навколо широко використовуваних тестів на IQ також вказує на той факт, що ми долекі від повного розуміння проблеми. Існують декілька характеристик інтелектуальних систем, які здаються корисними при спробі вирішення складних проблем управління. Інтелектуальні контролери можна розглядати як машини, що імітують розумові здібності людини, такі як адаптація та навчання, планування в умовах

великої невизначеності, впорядкування великої кількості даних тощо з метою ефективного контролю складних процесів; і це є обґрунтуванням використання терміну "інтелектуальний" в інтелектуальному управлінні, оскільки ці розумові здібності вважаються важливими атрибутами інтелекту людини. Альтернативний термін - це "автономний (інтелектуальний) контроль"; він підкреслює той факт, що інтелектуальний контролер, як правило, прагне досягти більш високого ступеня самостійності у досягненні і навіть постановці контрольних цілей, а не наголошує на (розумній) методології, яка досягає цих цілей. Ми повинні пам’ятати, що «інтелектуальний контроль» - це лише ім’я, яке сьогодні видається корисним. Таким же чином "сучасний контроль" 60-х років тепер став "звичайним (або традиційним) контролем", оскільки він став частиною мейнстріму. Те, що сьогодні називається інтелектуальним управлінням, можна буде назвати просто "управлінням" у не такому вже далекому майбутньому. Що більш важливо, ніж використовувана термінологія, - це поняття та методологія, і чи зможуть область управління та інтелектуальний контроль задовольнити постійно зростаючі потреби в нашому технологічному суспільстві.

1.5 Машинне навчання

Машинне навчання (МН, Machine Learning, ML) - великий підрозділ штучного інтелекту, що вивчає методи побудови алгоритмів, здатних навчатися. Існують і інші визначення. Машинне навчання - це «розділ, який досліджує методи, що дозволяють комп'ютерам покращувати свої характеристики на основі отриманого досвіду»

Першу програму на основі алгоритмів, здатних самонавчатися, розробив Артур Самуель (Arthur Samuel) в 1952 році, призначена вона була для гри в шашки. Самуель дав і перше визначення терміну «машинне навчання»: це «область досліджень розробки машин, які не є заздалегідь запрограмованими». Більш точне визначення терміну «навчання» дав набагато

пізніше Т. М. Мітчелл: кажуть, що комп'ютерна програма навчається на основі досвіду E по відношенню до деякого класу задач T і заходи якості P, якщо якість вирішення завдань з T, виміряний на основі P, поліпшується з набуттям досвіду E.

Вже в 1957 році була запропонована перша модель нейронної мережі, що реалізує алгоритми машинного навчання, схожі на сучасні. В даний час ведеться розробка самих різних систем машинного навчання, призначених для використання в таких технологіях майбутнього, як Інтернет Речей, Промисловий Інтернет Речей, в концепції "розумне місто", при створенні безпілотного транспорту і в багатьох інших.

У найзагальнішому випадку розрізняють два типи машинного навчання: навчання по прецедентах, або індуктивне навчання, і дедуктивне навчання. Оскільки останнє прийнято відносити до області експертних систем, то терміни «машинне навчання» і «навчання по прецедентах» можна вважати синонімами.

Класифікація алгоритмів машинного навчання:

- контрольоване (supervised learning) алгоритм генерує функцію, яка відображає вхід функції в потрібні результати. Одним із стандартних формулювань контрольованого навчання є задача класифікації: алгоритм зобов’язаний навчити (наблизити поведінку) функції, яка відображає вектор в один із декількох класів, переглянувши кілька прикладів входу-виходу функції;

- неконтрольоване (unsupervised learning) - моделює набір входів: приклади з мітками недоступні. Завдання машини при неконтрольованому навчанні - знайти зв'язок між окремими даними, виявити закономірності, підібрати шаблони, упорядкувати дані або описати їх структуру, виконати класифікацію даних;

- Навчання з підкріпленням (reinforcement learning) - алгоритм вивчає політику, як діяти за умови спостереження за світом. Кожна дія має певний вплив на навколишнє середовище, а навколишнє середовище

забезпечує зворотний зв'язок, який керує алгоритмом навчання. Таке навчання є окремим випадком контрольованого навчання, але вчителем в даному випадку є «середовище». Фактично машина і середовище утворюють систему зі зворотним зв'язком.

1.6 Класифікація

Завданнями, які вирішуються за допомогою керованого навчання навчання з учителем є, наприклад, класифікація і регресія.

Задача класифікації - формалізована задача, яка містить множину об'єктів (ситуацій), поділених певним чином на класи. Задана скінченна множина об'єктів, для яких відомо, до яких класів вони належать. Ця множина називається вибіркою. До якого класу належать інші об'єкти невідомо. Необхідно побудувати такий алгоритм, який буде здатний класифікувати довільний об'єкт з вихідної множини. Класифікувати об'єкт — означає, вказати номер (чи назву) класу, до якого належить цей об'єкт.

Оскільки задача визначення авторства документу належить до задач мультикласифікації, то розглянемо коротко найбільш відомі моделі класифікації.

1.6.1 Метод k найближчих сусідів

Метод k найближчих сусідів (kNN - k-Nearest Neighbors) - це алгоритм лінивої класифікації. Це означає, що в процесі навчання він не робить нічого, а тільки зберігає тренувальні дані. Він починає класифікацію тільки тоді, коли з'являються нові немарковані дані. Активний же класифікатор створює класифікаційну модель в процесі навчання. Коли вводяться нові дані, такий класифікатор «згодовує» дані класифікаційної моделі.

KNN також вважається непараметричним алгоритмом навчання, адже він не має жодних припущень про базові дані.

Коли з'являються нові нерозмічені дані, kNN проходить по 2 базовим крокам:

- Спочатку він шукає k найближчих розмічених точок даних - іншими словами, k найближчих сусідів.

- Потім, використовуючи класи сусідів, kNN вирішує, як краще класифікувати нові дані.

На рисунку 1.2 проілюстровано випадок, коли k=5.

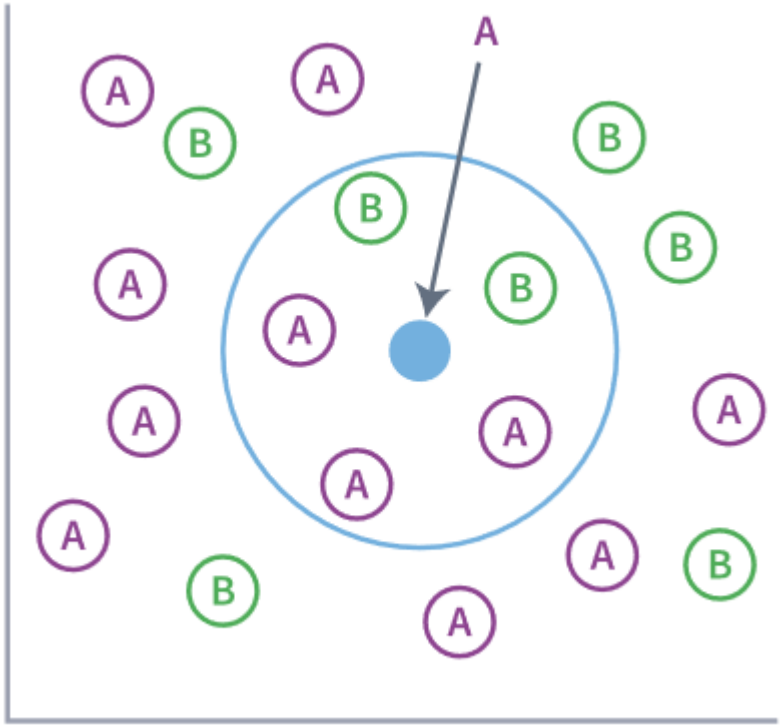

Рисунок 1.2 – Ілюстрація роботи методу k-найближчих сусідів (k=5)

В процесі навчання алгоритм просто запам'ятовує всі вектори ознак і відповідні їм мітки класів. При роботі з реальними даними, тобто спостереженнями, мітки класу яких невідомі, обчислюється відстань між вектором нового спостереження і зареєстрованими раніше розміткованими даними. Потім вибирається k найближчих до нього векторів, і новий об'єкт відноситься до класу, якому належить більшість з них. Більш детально роботу алгоритму можна описати наступним чином:

Крок 1 - Для реалізації алгоритму нам потрібен набір промітковаих навчальних даних, а також необхідно завантажити тестові дані.

Крок 2 - Далі нам потрібно вибрати значення K, тобто кількість найближчих точок до даної, за якими ми будемо приймати рішення. K може бути будь-яким цілим числом.

Крок 3 - Для кожної точки в тестових даних потрібно зробити наступне:

- Розрахувати відстань між тестовою точкою і кожним рядком навчальних даних з допомогою будь-якого з методів, а саме: Евклідової, Манхеттенської або Хеммінговскої відстані. Найбільш часто використовується метод розрахунку відстані Евкліда.
- Тепер, грунтуючись на значенні відстані, потрібно відсортувати їх в порядку зростання.
- Далі алгоритм вибере верхні k рядків з відсортованого масиву.
- Алгоритм призначить клас контрольній точці на основі найбільш вживаного класу цих рядків

Вибір параметра k суперечливий. З одного боку, збільшення його значення підвищує достовірність класифікації, але при цьому кордони між класами стають менш чіткими. На практиці хороші результати дають евристичні методи вибору параметра k, наприклад, перехресна перевірка. Існує рекомендація не вибирати k кратним числу класів, щоб запобігти ситуації, коли вибір класу буде неможливо здійснити.

До недоліків KNN методу можна віднести його ресурсо-затратність (повільний час роботи), чутливість до вибору навчальних даних та параметру k.

1.6.2 Класифікатор наївного Байєса

Наївний класифікатор Байєса - це простий імовірнісний класифікатор, заснований на застосуванні теореми Байєса з сильними (наївними) припущеннями незалежності атрибутів. Більш зрозумілим терміном для основної моделі ймовірності буде "Модель незалежних ознак". Наївний

класифікатор Байєса передбачає, що значення певної ознаки не пов'язане з наявністю або відсутністю будь-якої іншої ознаки. Наприклад, фруктом можна вважати яблуко, якщо воно червоне, кругле та діаметром близько 5 см. Наївний класифікатор Байєса вважає, що кожна з цих ознак незалежно сприяє ймовірності того, що цей фрукт є яблуком, незалежно від наявності чи відсутності інших особливостей. Для деяких типів імовірнісних моделей, наївні класифікатори Байєса можуть дуже ефективно навчатись на навчальному датасеті.

В основі алгоритму лежить теорема Байєса, що дозволяє розрахувати апостеріорну ймовірність P (c|x) на основі P (c), P (x) і P (x|c):

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \quad (1.1)$$

де P (c|x) - апостеріорна ймовірність даного класу c (тобто даного значення цільової змінної) при даному значенні ознаки x.

P (c) - апріорна ймовірність даного класу.

P (x|c) - ймовірність даного значення ознаки при даному класі.

P (x) - апріорна ймовірність даного значення ознаки.

Апостеріорна ймовірність для кожного класу c в методі класифікації наївного Баєса обчислюється за формулою:

$$P(c|x) = P(x_1|c)P(x_2|c) \dots P(x_n|c)P(c) \quad (1.2)$$

Виграє клас з максимальним значенням ймовірності (1.2)

У багатьох практичних програмах для визначення параметрів для наївних моделей Байєса використовується метод максимальної правдоподібності. Незважаючи на наївний дизайн та, мабуть, спрощені припущення, наївні класифікатори Байєса досить добре працювали у багатьох складних реальних ситуаціях. У 2004 році аналіз проблеми байєсівської

класифікації показав, що існують обґрунтовані теоретичні причини очевидної неправдоподібної ефективності наївних класифікаторів Байєса. Тим не менш, всебічне порівняння з іншими алгоритмами класифікації у 2006 році показало, що класифікація Байєса перевершує інші підходи, наприклад, дерева рішень або випадкові ліси.Перевагою наївного Байєса є те, що для оцінки необхідних для класифікації параметрів (середніх та відхилень змінних) йому потрібен лише невеликий обсяг навчальних даних. Оскільки приймаються незалежні змінні, потрібно визначати лише дисперсії змінних для кожного класу, а не всю матрицю коваріації.

Позитивні сторони алгоритму:

- Класифікація, в тому числі мультикласова, виконується легко і швидко.

- Коли припущення про незалежність виконується, мето перевершує інші алгоритми, такі як логістична регресія (logistic regression), і при цьому вимагає менший обсяг навчальних даних.

- Метод наївного Байєса краще працює з категорійними ознаками, ніж з неперервними. Для неперервних ознак передбачається нормальний розподіл, що є досить сильним допущенням.

Негативні сторони:

- Якщо в тестовому наборі даних є певне значення категорійної ознаки, яке не зустрічалося в навчальному наборі даних, тоді модель присвоїть нульову ймовірність цього значення і не зможе зробити прогноз. Це явище відоме під назвою «нульова частота» (zero frequency). Дану проблему можна вирішити за допомогою згладжування. Одним з найпростіших методів є згладжування по Лапласу (Laplace smoothing).

- Хоча класифікатор Байєса є хорошим класифікатором, значення прогнозованих ймовірностей не завжди є достатньо точними.

- Ще одним обмеженням методу є припущення про незалежність ознак. В реальності набори повністю незалежних ознак зустрічаються вкрай рідко.

Застосунки наївного байєсівського алгоритму

- Класифікація в режимі реального часу. Метод дуже швидко навчається, тому його можна використовувати для обробки даних в режимі реального часу.

- Мультикласова класифікація. Метод наївного Байесу забезпечує можливість мультикласової класифікації. Це дозволяє прогнозувати ймовірність для безлічі значень цільової змінної.

- Класифікація текстів, фільтрація спаму, аналіз тональності тексту. При вирішенні завдань, пов'язаних з класифікацією текстів, класифікатор Байєса перевершує багато інших алгоритмів. Завдяки цьому, даний алгоритм знаходить широке застосування в області фільтрації спаму (ідентифікація спаму в електронних листах) і аналізу тональності тексту (аналіз соціальних медіа, ідентифікація позитивних та негативних думок клієнтів).

- Рекомендаційні системи. Наївний Байєсівский класифікатор в поєднанні з колаборативною фільтрацією (collaborative filtering) дозволяє реалізувати рекомендаційну систему. В рамках такої системи за допомогою методів машинного навчання та інтелектуального аналізу даних нова для користувача інформація фільтрується на підставі прогнозованої думки цього користувача про неї.

Наведені вище переваги методу та хороші результати його роботи в текстовій класифікації і обумовили вибір цього методу для практичної реалізації методу визначення автора текстового документу.

1.6.3 Дерева рішень

Дерева рішень (decision trees) є одним з найбільш популярних методів вирішення завдань класифікації та прогнозування. Дерева рішень дозволяють візуально і аналітично оцінити результати вибору різних рішень і використовуються в галузі статистики та аналізу даних для прогнозних

моделей. Вперше були запропоновані Ховілендом і Хантом (Hoveland, Hunt) наприкінці 50-х років минулого століття.

Дерево рішень, подібно його «прототипу» з живої природи, складається з гілок з атрибутами (від них залежить результат - клас), листів зі значеннями класів (кінцеві вершини - результат вибору певного значення атрибута), а також вузлів - випадкових вершин, в яких визначаються можливі варіанти розвитку подій з певного моменту . «Зростає» дерево до тих пір, поки альтернативні варіанти не почнуть повторюватися.

Метою процесу побудови дерева прийняття рішень є створення моделі, за якою можна було б класифікувати випадки і вирішувати завдання вибору класу, маючи на вході кілька змінних.

У найбільш простому вигляді дерево рішень - це спосіб представлення правил «Якщо, тоді» в ієрархічній, послідовній структурі. Основа такої структури - відповіді "Так" або "Ні" на ряд питань.

На рисунку 1.3 наведено приклад дерева рішень, завдання якого - відповісти на питання: "Чи грати в футбол?" Для прийняття рішення, чи грати в футбол, слід віднести поточну ситуацію до одного з відомих класів (в даному випадку - "грати" або "не грати"). Для цього потрібно відповісти на ряд питань, які знаходяться у вузлах дерева, починаючи з його кореня.

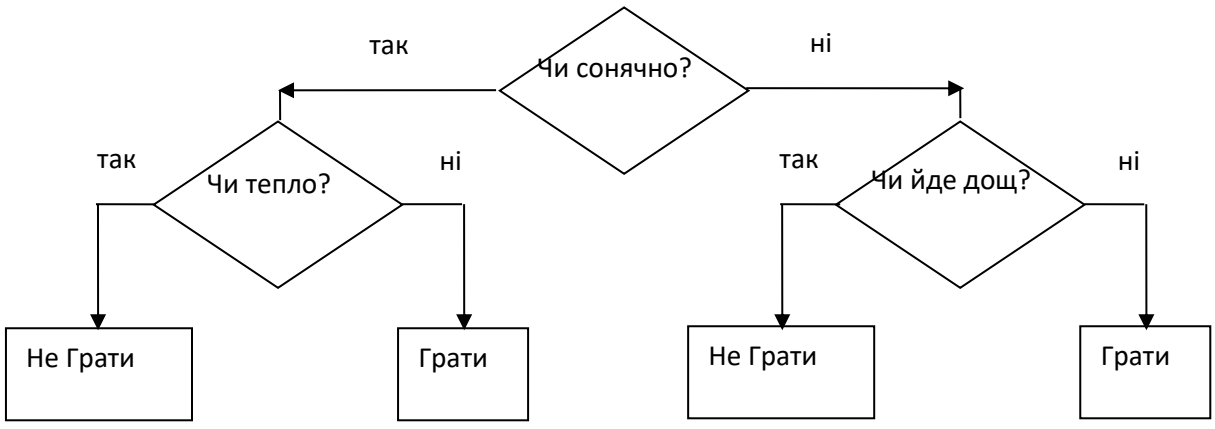

Рисунок 1.3- Дерево рішень "Грати чи в футбол?"

Розглянуте завдання класифікації відноситься до стратегії навчання з учителем.

Спочатку збирається статистична інформація про об'єкти, для яких вже відомо їх клас, - набір цих об'єктів називається навчальною вибіркою. Потім до цих даних застосовується алгоритм навчання, в результаті чого виходить навчений класифікатор, або модель. Тобто модель = алгоритм + навчальна вибірка.

Перед тим як застосовувати модель на реальних даних, як правило, проводять тестування її продуктивності. Найбільш поширеним методом тестування є так звана перехресна перевірка (cross-validation). Її суть полягає в тому, що навчальна вибірка ділиться на n частин. Класифікатор навчається на (n-1) частинах, а одну частину використовують безпосередньо для перевірки результату. Ця процедура повторюється n разів, а результатом вважається середнє арифметичне результатів окремих тестів.

Алгоритми конструювання дерев рішень складаються з етапів "побудова" або "створення" дерева (tree building) і "скорочення" дерева (tree pruning). У ході створення дерева вирішуються питання вибору критерію розгалуження і зупинки навчання (якщо це передбачено алгоритмом). В ході етапу скорочення дерева вирішується питання відсікання деяких його гілок.

В [4] наведено застосування лісів для ідентифікації автора

1.6.4 Метод опорних векторів (SVM)

У разі методу опорних векторів точка в просторі розглядається як вектор розмірності n. Ми хочемо дізнатися, чи зможемо ми розділити дані точки гіперплощиною розмірністю (n-1). Відділяюча гіперплощина є математичною сутністю, що відділяє між собою класи об’єктів з однаковими ознаками. Її назвемо лінією класифікатора. Наприклад, так як це показано на рис.1.4, де у тривимірному просторі площина відділяє кульки світлого кольору від темних кульок.

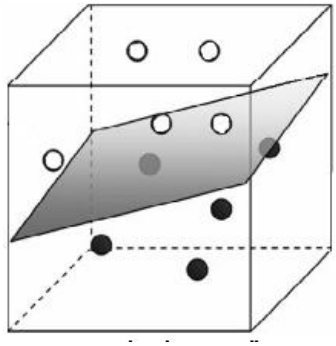

Рисунок 1.4 – Приклад відділяючої площини

Дані можуть бути класифіковані за допомогою різних гіперплощин. Краща гіперплощина - гіперплощина, при побудові якої поділ і різниця між 2 класами максимальні. Двовимірні об'єкти - дані на площині.

Розглянемо дані на площині, гіперплощина в даному випадку - це пряма (рис.1.4). Проведемо будь-яку пряму, яка розділить точки на 2 множини.

Виберемо пряму, максимально далеко проходить від точок. Відстань від неї до найближчої точки з кожного боку максимальна.

Якщо така пряма існує, то її називають гіперплощиною максимальної різниці, і лінійний класифікатор, її визначає, відповідно класифікатор максимальної різниці або персептрон оптимальної стабільності (стійкості).

Опорні вектори - це точки, для яких відстань до гіперплощини мімімальна (рис.1.5). Вони є ефективними елементами на навчальній вибірці.

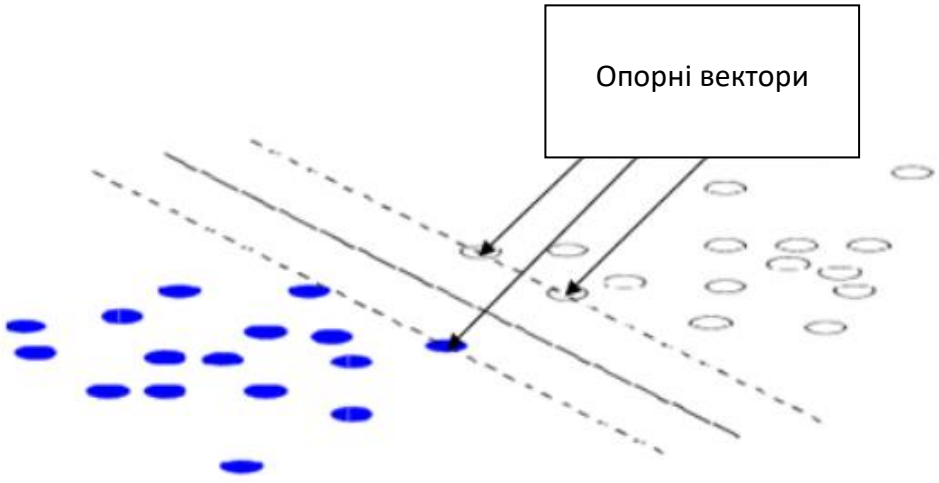

Рисунок 1.5 - Ілюстрація роботи методу опорних векторів

На додаток до лінійної класифікації, SVM можуть ефективно виконувати нелінійну класифікацію, використовуючи, що називається, трюк ядра, неявно відображаючи вхід у багатовимірний простір ознак. В такому випадку всі вектори навчальної вибірки вкладаються простір більш високої розмірності з допомогою спеціального відображення таким чином, щоб в новому просторі вибірка була лінійно подільна.

1.7 Висновки до першого розділу

В даному розділі наведено виклад основних теоретичних засад для задачі ідентифікації автора за текстовим документом, а саме:

- сформульовано основні завдання науки про визначення авторства;
- описано історію виникнення науки стилеметрії;
- наведено класифікацію методів дослідження проблеми ідентифікації автора;
- описано методи класифікації, як базові методи машинного навчання, що використовуються при визначенні автора документу.

2 ОСОБЛИВОСТІ ПОБУДОВИ ТА ОЦІНКИ ЯКОСТІ КЛАСИФІКАЦІЙНИХ МОДЕЛЕЙ ТЕКСТОВИХ ДОКУМЕНТІВ

Класифікація текстових документів, такі як статті в газетах, наукові праці в журналах або, можливо, реферати доповідей, є важливою задачею і використовується в тому числі для визначення авторства документу. Метою класифікації є використання набору попередньо класифікованих документів для класифікації тих, які ще не були віднесені до певного класу. Це стає все більш важливою практичною проблемою, оскільки обсяг друкованих матеріалів у багатьох випадках постійно збільшується, і навіть у спеціалізованих сферах знайти відповідні документи може бути дуже складно.

У принципі ми можемо використовувати будь-який із стандартних методів класифікації (наївний Баєса, найближчих сусідів, дерева рішень тощо) для цього завдання, але набори текстових документів мають ряд специфічних особливостей у порівнянні з наборами даних, які ми бачили досі , які потребують окремого пояснення.

2.1 Мульти-класифікація

Однією з особливостей, що відрізняє класифікацію тексту від інших завдань класифікації, є можливість належності одного об'єкту до декількох класів (мультикласифікація). За звичай, в задачах класифікації вважають, що існує безліч взаємовиключних категорій і що кожен об'єкт неминуче повинен належати лише до однієї з них.

Класифікація тексту є досить різною. Загалом у нас може бути N категорій таких як медицина, бізнес, фінанси, історичні, біографічні, управлінські та освітні науки, і цілком можливо, що документ може бути входити у декілька цих категорій, можливо, навіть у всі чи, можливо, не належати до жодної.

Замість того, щоб розширювати використовуване до цього часу визначення класифікації, задачу мультикласифікації тексту вважають N окремими завданнями бінарної класифікації, наприклад.

- Чи є документ про медицину? Так-ні
- Чи є документ про бізнес? Так-ні
- Чи є документ про фінансування? Так-ні

і так далі. Необхідність виконання N окремих завдань щодо класифікації значно збільшує час, який потребує ця форма класифікації, що навіть для однієї класифікації зазвичай обчислювально дорого.

2.2 Представлення текстових документів для аналізу

Для "стандартних" завдань аналізу даних (data mining) дані подаються на вхід системи у стандартній формі (Тип даних може бути чисельним, стрічковим, символьним чи логічним). Існує за звичай фіксована кількість атрибутів (або функцій), які були обрані до початку збору даних. Для аналіз тексту набір даних зазвичай включає в себе самі документи, а атрибути витягуються з документів автоматично на основі їх вмісту до того, як буде застосований алгоритм класифікації. Як правило, існує дуже велика кількість атрибутів, більшість з них зустрічаються лише рідко, з високою часткою шумових та невідповідних ознак.

Існує кілька способів перетворення документів з простого тексту в об’єкти з визначеною кількістю атрибутів у навчальному наборі. Наприклад, ми можемо порахувати, скільки разів виникають задані фрази, або, можливо, будь-яка комбінація двох послідовних слів, або ми можемо порахувати виникнення двох або трьох комбінацій символів (відомих як біграми та триграми відповідно). Для цілей цієї роботи будемо вважати, що використовується просте представлення на основі слів, відоме як репрезентація "сумки слів" (bag-of-words). При такому поданні документ вважається просто сукупністю слів, які в ньому зустрічаються хоча б один раз.

Порядок слів, їх комбінації, структурування абзаців, пунктуація та звичайно значення слів ігноруються. Отже, в такому випадку, документ можна розглядати як набір слів, розміщених у довільному порядку, скажімо, за алфавітом разом із підрахунком, скільки разів трапляється кожне з них, або якась інша міра важливості кожного слова.

Якщо в даному документі сказано 100 різних слів, ми не можемо просто використовувати представлення з 100 атрибутами. Інші документи в наборі даних можуть використовувати інші слова, можливо, вони накладаються зі 100 у поточному екземплярі, але це не обов'язково. У нових документах, які ми хочемо класифікувати, можуть бути слова, які не використовуються в жодному з навчальних документів. Очевидним, але вкрай поганим підходом було б виділити стільки атрибутів, скільки потрібно для забезпечення всіх можливих слів, які можуть бути використані в будь-якому можливому документі. Якщо мова документів - англійська, кількість можливих слів становить приблизно один мільйон, що є безнадійно непрактичною кількістю атрибутів, які слід використовувати.

Набагато кращий підхід полягає в обмеженні набору слів, які насправді зустрічаються у навчальних документах. Це все-одно може привести до величезних наборів атрибутів. Нижче розглянемо способи скорочення інформативного простору ознак. Ми розміщуємо всі слова, що вживаються принаймні один раз, у «словнику» та виділяємо по одній позиції атрибутів у кожному рядку нашого навчального набору для кожного. Порядок, у якому ми це робимо, є довільним, тому ми можемо вважати це алфавітом.

Представлення даних як "сумки слів" по суті є надзвичайно надмірним. Цілком імовірно, що для будь-якого конкретного документа більшість атрибутів / ознак (тобто слів) не з’являться. Наприклад, словник може містити 10000 слів, але конкретний документ може містити всього 200 різних слів. Якщо це так, його представлення як екземпляр у навчальному наборі матиме 9 800 з 10 000 атрибутів із значенням нуль, що вказує на відсутність подій, тобто невикористаних.

Якщо є декілька класифікацій, є дві можливості побудови словника слів для колекції навчальних документів. Незалежно від того, який із них використовується, словник, ймовірніше за все буде великим.

Перший - це локальний словниковий підхід. Ми формуємо різний словник для кожної категорії, використовуючи лише ті слова, які містяться в документах, віднесених до категорії. Це дає змогу кожному словнику бути порівняно невеликим за необхідності побудови N з них, де існує N категорій.

Другий підхід полягає в побудові глобального словника, який включає всі слова, які зустрічаються хоча б один раз у будь-якому з документів. Потім він використовується для класифікації у кожній з N категорій. Побудова глобального словника буде, безумовно, набагато швидшою, ніж побудова N локальних словників, але ціна створення є ще більш надмірною. Очевидно, , що використання локального словника має кращі результати, ніж використання глобального словника.

2.3 Зупинка слів (stopwords)

При "сумки слів" можливе виникнення десятків тисяч різних слів у досить невеликому наборі документів. Багато з них не важливі для задачі класифікації, і їх використання може істотно погіршити результативність. Обов’язково настільки, наскільки можливо, зменшити розмір простору ознак (тобто скоротити набір слів, що входять до словника). Це можна розглядати як варіант способів підготовки даних та очищення даних.

Один з широко застосовуваних підходів полягає у використанні списку поширених слів, які, ймовірно, будуть марними для класифікації, відомих як стоп-слова, та видаленні цих слів перед створенням словника. Немає визначеного списку стоп-слів, який би використовувався універсально. Список очевидно відрізняється від мови до мови, але в англійській мові очевидним вибором буде "a", "an", "the", "is", "I", "you" і "of". Слід заначити, що вивчення частоти та поширення таких слів може бути дуже корисним для

стилістичного аналізу, тобто вирішення, хто з кількох можливих авторів написав роман чи п’єсу тощо, але для класифікації документа на такі категорії, як медицина, фінанси тощо це очевидно непотрібні слова. Університет Глазго має список із 319 англійських стоп-слів, що починаються з, about, above, across, after, afterwards and ending with yet, you, your, yours, yourself, yourselves. До тих пір, чим довший перелік стоп-слів, тим краще, єдиний ризик - можлива втрата корисної класифікуючої інформації, якщо список стане надмірним.

Існує три відомих методи видалення стоп-слів:

- Словниковий: використовує готовий перелік стоп-слів.
- Статистичний: статистично на основі достатньо великої кількості текстів з даної області визначаються найбільш вживані слова. Перевагою методу є те, що його можна автоматизувати. Недоліком є те, що можуть бути видалені значущі слова.
- За Y-інтерпретацією закону Бредфорда ділить слова на три групи: неважливі (стоп-слова); слова, що ідентифікують текст; слова, що рідко зустрічаються.

2.4 Стемінг (stemming)

Ще одним дуже важливим способом зменшення кількості слів у представленні тексту є використання стемінгу.

Це ґрунтується на спостереженні, що слова в документах часто мають багато морфологічних варіантів. Наприклад, ми можемо використовувати слова computing, computer, computation, computes, computational, computable and computability у одному документі. Ці слова явно мають однаковий мовний корінь. Якщо скласти їх так, ніби вони зустрічалися з одним словом, то ймовірно було б чітко вказати на зміст документа, тоді як кожне слово окремо не могло.

Стемінг - це зведення кількох спільнокореневих слів до їх спільного кореня. Корінь слова -це не ідентичне поняття до морфологічного кореня на основі словника, він просто є рівною або меншою формою слова.

Алгоритми стемінгу зазвичай засновані на правилах. Їх можна розглядати їх як евристичний процес, який начебто відриває кінці та початки слів. Слово переглядається і проходить через ряд умов, які визначають, як його скоротити. Наприклад, у нас може бути правило суфікса, яке спирається на список відомих суфіксів. В англійській мові у нас є суфікси типу "-ed" та "-ing", які можуть бути корисними для того, щоб відрізати для того, щоб зіставити слова “cook,” “cooking,” and “cooked” все до одного кореня “cook.

Метою визначення є розпізнавання наборів слів, таких як "обчислення" та "обчислення" або "застосовано", "застосувати", "застосовується" та "застосувати", які можна трактувати як рівнозначні. Існує безліч алгоритмів, що створюють основні принципи, які були розроблені для зменшення слова до його кореневої форми, за допомогою якого воно замінюється. Наприклад, "обчислення" та "обчислення" можуть бути пов'язані з "комп’ютером", а "застосувати" тощо до "застосунку".

Однак, оскільки стемінг, як правило, базується на евристиці, він далеко не ідеальний. Насправді, зазвичай, цей метод страждає двома протилежними недоліками, зокрема: перевиконання та недовиконання. Перевиконання виникає, коли занадто багато слів скорочено. Це може призвести до безглуздих коренів, де все значення цього слова втрачається або заплутується. Або це може призвести до того, що слова прив'язуться до одним коренів, хоча вони, мабуть, і не повинні бути.

Візьмемо чотири слова "universal", "university", "Universe", "universities". Стемінговий алгоритм, який скорочує ці чотири слова до "univers", перевиконаний (overstemmed). Хоча може бути непоганим, щоб "universal" і "Universe" зв'язалися разом, а "university" і "universities" зійшли разом, всі чотири не підходять. Краще рішення може мати перші два рішення

"univers", а останні два - "universi". Проте створення правил, які приведуть до такого результату - це окрема наукова задача.

Недовиконання (understemming)- протилежне питання. Воно походить, коли у нас є кілька слів, які насправді є формами один одного. Було б непогано, щоб усі вирішили однин і той же корінь, але, на жаль, цього не роблять.Це можна побачити, якщо у нас є алгоритм, що визначає слова "data" (дані) і "datum" до "dat" і "datu". Було б добре скорити їх до "dat". Але як тоді поступити з "date" (датою)? І чи є добре загальне правило? Або ми просто застосовуємо дуже специфічне правило для дуже конкретного прикладу?

Ці питання швидко стають проблемами, коли мова йде про стемінг. Застосування нових правил та евристики може швидко вийти з-під контролю. Вирішення однієї або двох проблем, що виникають в результаті недовиконання або перевиконання стемінгу, можуть призвести до появи ще двох! Скласти хороший стемінговий алгоритм- це важка робота.

Використання стемінгу може бути дуже ефективним способом зменшення кількості слів у словнику до відносно керованого числа. Однак, як і у випадку стоп-слів, не існує стандартного алгоритму, що ідеальним і може призвести до вилучення важливих слів. Наприклад, слово "appliqu´e" в документі може бути важливим атрибутом щодо його класифікації, але може бути зменшене, переходячи на "application", що має такий же корінь.

Відомими алгоритмами стемінгу є:

- Стемер Портера (Porter stemmer): Цей стемінговий алгоритм є найстарішим. Він походить з 1980-х, і головним його завданням є усунення загальних відомих закінчень до слів, щоб їх можна було звести до загальної форми. Це не надто складно, і розвиток на ньому застиг. Як правило, це хороший початковий базовий стемер, але його не рекомендується використовувати для будь-якого прикладного застосування. Натомість він займає своє місце в дослідженні як хороший, базовий алгоритм, що може гарантувати відтворюваність. Це також дуже м'який алгоритм, порівняно з іншими.

- Снігова куля (Snowball stemmer): Цей алгоритм також відомий як алгоритм Porter2. Він майже загальновизнаний як кращий, ніж його попередник. Цей алгоритм є також більш агресивний, ніж стемер Портера. Дуже багато речей, що додали до визначення кореня в «Сніговій кулі», були пов’язані із проблемами, поміченими в Porter stemmer. Існує приблизно 5% різниці в тому, як "Снігова куля" застосовується проти алгоритму Портера
- Ланкастерський стемер (Lancaster stemmer): Це найагресивніший алгоритм стемування в групи. Однак якщо використовувати стемування NLTK, можна легко додати свої власні власні правила до цього алгоритму. Це хороший вибір для цього. Одна з скарг навколо цього алгоритму, що полягає в тому, що він іноді є надто агресивним і може перетворити слова на дивні корені.

2.5 Лематизація (Lemmatization)

Лематизація зазвичай стосується правильної роботи із використанням словника та морфологічного аналізу слів, як правило, спрямована на вилучення лише неважливих закінчень та повернення основної чи словникової форми слова, яка відома як лема. Щоб привести деяке слова до його початкової форми необхідно, насамперед визначити, до якої частини мови воно належить. Для цього потрібна додаткова обчислювальна лінгвістика, наприклад, частина мовлення. Це дозволяє йому робити більш якісні рішення (як, наприклад, вирішення "is" і "am" до слова "be").

Ще одна річ, яку слід зазначити про лематизацію, полягає в тому, щоо створювати лематизатор новою мовою важче, ніж стемінговий алгоритм. Оскільки лематизаторам потрібно набагато більше знань про структуру мови, це набагато більш тривалий процес, ніж просто спроба налаштувати евристичний стемінговий алгоритм.

На щастя, якщо працювати з англійською мовою, можна швидко скористатися лематизацією через NLTK як і увипадку зі стемінгом. Однак для

отримання найкращих результатів вам доведеться подати частину мовних тегів лематизатору, інакше він може не зменшити всі слова до бажаних лем. Більше того, це базується на базі даних WordNet (яка схожа на мережу синонімів чи тезаурус), тому якщо там немає хорошого посилання, то ми все одно не отримаєм правильну лему.

Щоб продемонструвати різницю між стемінгом і лематизацією розглянемо наступний приклад для слова " caring" Стемінг повинен забрати стандартне закінчення "ing", в той час як лематизація привести до початкової форми дієслова " care".

‘Caring’ -> Lemmatization -> ‘Care’

‘Caring’ -> Stemming -> ‘Car’

Лематизацію доцільно проводити перед стемінгом.

2.6 Використання корисної інформації для зменшення розмірності ознак

Навіть після вилучення стоп-слів з документа та заміни кожного слова, що залишилося його коренем, кількість слів, що представляють набір документів, може бути дуже великою.

Одним із способів зменшення кількості слів для даної категорії документів C_k є побудова навчального набору, де кожен екземпляр містить частоту кожного слова (або якусь подібну міру) разом із значенням класифікації C_k, яке повинно бути двійковим так / немає значення [9].

Як відомо ентропія обчислюється за формулою:

$$\boldsymbol{H(x) = -\sum_{i=1}^{K} P(x_i) \cdot log\, P\,(x_i)},\ \text{i = 1, K,} \qquad (2.1)$$

де $X = x_1, x_2, x_3, \ldots x_k$ – алфавіт повідомлення.

Наприклад, якщо 10% навчальних документів належать до категорії C_k, ентропія становить $-0.1 \times log_2\ 0.1 - 0.9 \times log_2\ 0.9 = 0.47$

Використання методу, такого як методика таблиці частот, тепер ми можемо обчислити інформаційний приріст, якщо це класифікує документ як належний до категорії C_k або іншим чином, що буде результатом знань значення кожного з атрибутів по черзі. Зробивши це, ми можемо вирішити використовувати лише функції з найвищими (скажімо) 20, 50 або 100 значеннями коефіцієнта отримання інформації при класифікації документів, належать вони до категорії C_k чи ні.

2.7 Представлення текстових документів: побудова моделі векторного простору

Тепер будемо вважати, що ми вирішили використовувати локальний або глобальний словник і обрали представлення, яке замінює кожен документ за низкою атрибутів. Для "bag of words" представлення кожна функція - це одне слово, але для іншого представлення, це може бути щось інше, наприклад, фраза. Далі ми припустимо, що кожний атрибут є свого роду терміном.

Як тільки ми визначили, що загальна кількість ознак дорівнює N, ми можемо представляти слова у словнику в деякому довільному порядку як t_1, t_2, ...,t_N. Потім ми можемо представити і-й документ як упорядкований набір N значень, який ми будемо називати N-вимірним вектором і записувати як (X_{i1}, X_{i2}, ..., X_{iN}). Ці значення є лише значеннями атрибутів у стандартному форматі навчального датасету. Запис значень у вигляді N-розмірних векторів (тобто у вигляді N значень, розділених комами та взятих у круглі дужки) - просто більш звичайний спосіб перегляду даних у цій галузі обміну даними. Повну множину векторів для всіх документів називають векторною просторовою моделлю (vector space model VSM) [3].

До цього часу ми припускали, що значення, що зберігаються для кожної ознаки (атрибуту), - це кількість разів, скільки кожне слово вживається у відповідному документі. Однак це не повинно бути так. Загалом можна

сказати, що значення X_{ij} - це вага, що вимірює важливість j-го члена t_j в і-му документі.

Одним із поширених способів обчислення ваг є підрахунок частоти зустрічань кожної ознаки в даному документі (відомий як частота слова term frequency - TF). Інша можливість полягає у використанні двійкового подання, де 1 вказує на присутність, а 0 вказує на відсутність терміна в документі.

Більш складний спосіб обчислення ваг називається TF-IDF, який означає термін "Частота слова -Обернена частота документа" (Term Frequency Inverse Document Frequency). Це поєднує частоту слова з мірою рідкості зустрічання терміна в повному наборі документів. Повідомлялося, що це призводить до підвищення продуктивності порівняно з іншими методами.

Даний показник використовується з різними варіантами обчислення TF і IDF. Існує значна кількість робіт в цьому питанні, проте є одна особливість - датасет не повинен змінюватися під час обчислень. Це ускладнює обчислення, якщо потрібно провести обрахування даних в реальному часі.

Найчастіше, значення TF-IDF ваги X_{ij} обчислюється як добуток двох значень, що відповідають терміну частоти та оберненій частоті документа відповідно.

Перше значення - це просто частота j-го члена. Використовуючи це значення, як правило, роблять слова, які часто зустрічаються в даному (одному) документі, є важливішими, ніж інші. Частота слова (Term Frequency або скорочено TF) - відношення числа входжень обраного слова до загальної кількості слів документу.

$$TF(d_i, t_j) = \frac{n_j}{n_i} \tag{2.2}$$

де n_j – число входження j-ого слова t_j в i-ому документі, n_i. – загальна кількість слів в i-ому документі.

Значення IDF (оберненої частоти документа) найчастіше рахують за формулою:

$$IDF(t_j) = log_2 \frac{|D|}{|t_j \in d_i|} \quad (2.3)$$

де $|D|$ - загальна кількість документів в датасеті, а $|t_j \in d_i|$ - кількість документів, що містять термін t_j.

Використання цього значення, як правило, дозволяє присвоїти більшу важливість до слів, більш рідкісних для колекції документів. Якщо в кожному слово (термін) зустрічається у кожному документі, то обернене значення частоти документа дорівнює одиниці. Якщо він зустрічається лише в одному документі з кожних 16, його обернене значення частоти документа рівне log_2 16 = 4. Отже, чим частіше слово зустрічається у всіх документах колекції тим нижчим буде його IDF.

Відомі й інші підходи для виділення термінів простору ознак тексту. Наприклад, можна розглядати не окремі слова, а знаходити n-слівні поєднання по заданих частотних характеристиках. Це можуть бути значення абсолютних або відносних частот для даних словосполучень або значення деякої статистичної міри. Потім можна встановити деяке порогове значення і відсікати по ньому.

2.8 Лексичні методи вилучення ключових слів

В попередньому пункті описано статистичний підхід до виділення ключових слів. На противагу статистичним методам існують ще лексичні методи побудови простору ознак. Як і випадку зі статистичним методами, не існує ідеального способу виділення ключових слів. Спроби створення універсального лінгвістичного методу виділення ключових слів не мали успіху. Класифікацію лексичних підходів можна провести за певними лінгвістичним напрямками [12].

Лінгвістичні методи, що ґрунтуються на значеннях слів, використовують онтологію і семантику слова. Недоліком цих методів є їх ресурсозатратність і трудомісткість: розробка онтологій вимагає великих затрат. Кріт того, людський фактор при виконанні операцій лінгвістичного аналізу текстів є джерелом значної кількості помилок і неточностей. Тому необхідно автоматизувати процес аналізу текстів документів, а це далеко нетривіальна наукова задача.

Існує метод автоматизованого визначення ключових термінів з текстових документів, заснований на мірі семантичної близькості слів, обчисленої з використанням бази даних Вікіпедії, побудові семантичного графа, виборі ключових слів за допомогою алгоритму Гірвана-Ньюмана. Перевагою цього методу є відсутність необхідності в попередньому навчанні, так як працює безпосередньо з базою даних.

Ефективність методу підтверджена експериментальними оцінками точності і повноти вилучення з тексту ключових термінів.

2.9 Нормалізація ваг

Перш ніж використовувати набір N-розмірних векторів, нам спочатку потрібно нормалізувати значення ваг з причин, аналогічних необхідності нормалізації значення безперервних атрибутів. В ідеальному випадку кожне значення знаходиться в межах від 0 до 1 включно, тобто щоб на значення, які використовуються, не надто впливала загальна кількість слів у вихідному документі.

Проілюструвати суть нормалізації на спрощеному прикладі. Припустимо, у нас є словник із лише 6 членами, і припустимо, що використовувані ваги - це лише значення частот (кількості використань) кожного слова в документів. Тоді типовий вектор був би (0, 3, 0, 4, 0, 0). У відповідному документі другий термін з’явився 3 рази, четвертий термін відбувся 4 рази, а інші чотири терміни взагалі не з’явилися. Загалом у

документі з’явилося лише 7 термінів, після видалення стоп-слів, стемінгу тощо.

Припустимо, ми створимо ще один документ, розмістивши точний дублікат його вмісту в кінці першого документу. Розглянемо також випадок, коли в зв’язку з певними причинами вміст оригіналу був надрукований 10 разів, а то й сто разів в документі

У цих трьох випадках векторами будуть (0, 6, 0, 8, 0, 0), (0, 30, 0, 40, 0, 0) і (0, 300, 0, 400, 0, 0). Схоже на те, що ці вектори не мають нічого спільного з першим початковим вектором, який був (0, 3, 0, 4, 0, 0). Це незадовільно. Очевидно, чотири документи повинні бути класифіковані точно так само, і векторне представлення простору повинно відображати це.

Метод, який зазвичай використовується для нормалізації векторів, акуратно вирішує цю проблему. Ми обчислюємо довжину кожного вектора, визначену як квадратний корінь суми квадратів його значень компонентів. Для нормалізації значень ваг ділимо кожне значення на довжину. Отриманий вектор має властивість, що його довжина завжди дорівнює 1.

Для наведеного вище прикладу довжина (0, 3, 0, 4, 0, 0) становить $\sqrt{3^2 + 4^2}$= 5, тому нормалізований вектор дорівнює (0, 3/5, 0, 4/5, 0, 0), що має довжину 1. Зверніть увагу, що нульові значення не грають ніякої ролі у розрахунках.

Розрахунки для інших трьох наведених векторів наступні:

- для вектора (0, 6, 0, 8, 0, 0) обчислена довжина дорівнює $\sqrt{6^2 + 8^2}$= 10, тому нормалізований вектор дорівнює

(0, 6/10, 0, 8/10, 0, 0) = (0, 3/5, 0, 4/5, 0, 0).

- Для вектора (0, 30, 0, 40, 0, 0) довжина рівна $\sqrt{30^2 + 40^2}$= 50, тому нормалізований вектор дорівнює

(0, 30/50, 0, 40/50, 0, 0) = (0, 3/5, 0, 4/5, 0, 0).

- Для вектора (0, 300, 0, 400, 0, 0) довжина рівна $\sqrt{300^2 + 400^2}$= 500, тому нормалізований вектор дорівнює

(0, 300/500, 0, 400/500, 0, 0) = (0, 3/5, 0, 4/5, 0, 0).

У нормалізованому вигляді всі чотири вектори однакові, як і повинно бути.

2.10 Вимірювання відстані між двома векторами

Перед використанням n-вимірного простору ознак, необхідно нормалізувати вагові коефіцієнти. Однією з важливих перевірок на відповідність нормованого представлення векторної просторової моделі документів, є те, що можна ввести чітке визначення відстані між двома векторами. Очевидно, що було б добре якбивідстань між двома однаковими векторами дорівнювала нулю, відстань між двома векторами, максимально різними, дорівнювала 1, а відстань між будь-якими іншими векторами була десь посередині.

Стандартне визначення відстані між двома одиничними векторами, відповідає цим критеріям. Визначимо скалярний добуток двох одиничних векторів одного розміру як суму добутків відповідних пар значень координат.

Наприклад, якщо взяти два ненормалізованих вектора (6, 4, 0, 2, 1) і (5, 7, 6, 0, 2), нормалізувати їх, перетворивши значення в (0,79, 0,53, 0, 0,26, 0,13) та (0,47, 0,66, 0,56, 0, 0,19).

Скалярний добуток в такому випадку дорівнює 0,79 x 0,47 + 0,53 x 0,66 + 0 x 0,56 + 0,26 x 0 + 0,13 x 0,19 = 0,74.

Якщо відняти це значення від 1, отримаємо міру відстані між двома значеннями, яка дорівнює 1 -0,74 = 0,26.

Що станеться, якщо обчислити відстань між двома однаковими одиничними векторами? Скалярний добуток дає суму квадратів значень, яка повинна бути дорівнює 1, оскільки довжина одиничного вектора дорівнює 1 за означенням. Віднімання цього значення від 1 дає відстань нуль.

Якщо ми візьмемо два одиничні вектори, що не мають спільних значень (що двом оригінальним документам, які не мають спільних термінів), скажімо (0,94, 0, 0, 0,31, 0,16) і (0, 0,6, 0,8, 0, 0) скалярний добуток рівний 0,94 x 0 + 0 x

0,6 + 0 x 0,8 + 0,31 x 0 + 0,16 x 0 = 0. Віднімання цього значення від 1 дає міру відстані 1, що є максимальним значенням відстані.

2.11 Оцінка текстового класифікатора

Після того як ми перетворили навчальні документи в нормалізовану векторну форму, ми зможемо побудувати навчальний набір для кожної категорії C_k. Ми можемо перетворити набір тестових документів у тестовий набір об'єктів для кожної категорії так само, побудувати навчальний датасет, і, будь-який обраний алгоритм класифікації, який ми обрали для навчальних даних застосувати для класифікації даних тестового набору.

Для кожної категорії C_k ми можемо побудувати матрицю такого виду.

		Прогнозовані класи	
		C_k	not C_k
Фактичні класи	C_k	*a*	*c*
	not C_k	*b*	*d*

Рисунок 2.1 - Матриця конфігурування для категорії C_k

На рисунку 2.1 значення a, b, c і d є кількостями істинно позитивних (True Positive), хибнопозитивних (False Positive), помилково негативних (False Negative) та справжніх негативних (True Negative) класифікацій відповідно. Для досконалого класифікатора і b, і c повинні були б бути рівними нулю.

Одним з базових методів оцінки є його достовірність (accuracy). Достовірність визначається як кількість правильно класифікованих елементів певного датасету до загальної кількості елементів в цьому датасеті. Достовірність прогнозу можна порахувати за формулою:

$$Accuracy = \frac{True\ Positive + True\ Negative}{Total} = \frac{a+d}{\text{a + b + c + d}} \quad (2.4)$$

Однак, для текстової класифікації тексту, більш звичайним є використання деяких інших показників ефективності класифікатора. Іншими методами оцінки якості моделі є точність (precision) та повнота (recall)

Повнота визначається як частка документів у категорії C_k, які правильно прогнозуються.

$$Recall = \frac{True\ Positive}{Predicted\ results} = \frac{True\ Positive}{True\ Positive + False\ Negative} = \frac{a}{a+c} \quad (2.5)$$

Точність визначається як частка документів, які, за прогнозами, належать до категорії C_k і які, фактично, належать до цієї категорії.

$$Precision = \frac{True\ Positive + True\ Negative}{Actual\ results} = \frac{True\ Positive}{True\ Positive + False\ Positive} = \frac{a}{a+b} \quad (2.6)$$

Загальноприйнятою практикою є поєднувати точність та повноту в єдиний показник продуктивності, який називається показником F1, який визначається формулою:

$$F1 = \frac{\text{2 x Precision x Recall}}{\text{Precision + Recall}} \quad (2.7)$$

Це лише добуток Precision і Recall, поділений на їх середнє значення.

Створивши матриці конфігурування для кожної з N бінарної класифікації, ми можемо їх поєднати декількома способами. Один з методів називається мікро-усередненням. N матриць конфігурування додаються разом по елементам для формування єдиної матриці, з якої можна обчислити Recall, Precision, F1 та будь-які інші бажані заходи.

2.12 Висновки до розділу 2

В даному розділі наведено особливості розв'язання текстової класифікації, зокрема:

- описано процес попередньої підготовки даних, їх очищення;
- наведено схеми їх нормалізації (стемінг, лематизація);
- проведено аналіз формування простору ознак з текстових документів;
- запропоновано підхід для вимірювання різниці (відстані) між двома документами для даної задачі класифікації;
- проведено аналіз існуючих оцінок точності для моделі текстового класифікатора.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ МЕТОДУ ІДЕНТИФІКАЦІЇ АВТОРА

3.1 Набір даних

При розв’язанні задач машинного навчання велике значення має набір даних (датасет). В цій роботі використано датасет з репозитарію машинного навчання UCI, розміщеному за посиланням https://archive.ics.uci.edu/ml/datasets/Victorian+Era+Authorship+Attribution#. [3]

Для зменшення упередженості та створення надійного набору даних про атрибуцію авторів було вибрано наступні критерії для фільтрації авторів у базі даних Gdelt: автори, що пишуть англійською мовою, автори, у яких є достатня кількість книг (принаймні 5), автори 19 століття. За цими критеріями було відібрано 50 авторів, а їхні книги отримали запит через базу даних Big Query Gdelt. Потім провели очищення набору даних через проблеми з читанням OCR у вихідному сировинному вигляді. Для цього спочатку всі книги просканували, щоб отримати загальну кількість унікальних слів і частоти кожного слова. Під час сканування текстів було видалено перші 500 слів і останні 500 слів, щоб вилучити такі особливості, як ім’я автора, назва книги та інші особливості слова, які могли б полегшити завдання класифікації. Після цього кроку вибрали 10 000 найпопулярніших слів, які мали місце у 50 корпусах текстових даних. Слова, які не входять до перших 10000 слів, було видалено, залишаючи недоторканою решту структури речення. Вся книга розбита на фрагменти тексту з 1000 слів кожен. Текстові сегменти, що містять менше 1000 слів, були заповнені нулями, щоб зберегти їх у наборі даних. 1000 слів складають приблизно 2 сторінки написання, що є достатньо довгим, щоб отримати з документа різноманітні функції. Кожен об'єкт навчального набору складається з текстового фрагмента з 1000 слів та доданого ідентифікатора автора. У тестовому наборі є лише текстовий фрагмент з 1000 слів для атрибуції авторства. Навчальні дані складаються з 45 авторів, а дані для тестування - 50 авторів. Загалом датасет містить 93600 записів.

3.2 Вибір програмного середовища

Для практичної реалізації методу ідентифікації автора використано високо-рівневу інтерпретовану мову програмування Python.

Python був головним чином розроблений для акцентування на читабельності коду, а його синтаксис дозволяє програмістам виражати концепцію програми в меншій кількості рядків коду. Python - мова програмування, яка дозволяє швидко працювати та ефективніше інтегрувати системи.Є дві основні версії Python - Python 2 та Python 3. Обидві досить різні.

Особливостями даної мови програмування є те, що вона:

- Інтерпретована. Немає окремих етапів компіляції та виконання, таких як в C і C ++. Програма безпосередньо виконується з вихідного коду. Внутрішньо Python перетворює вихідний код у проміжну форму, яку називають bytecodes, яка потім перекладається на рідну мову конкретного комп'ютера для її запуску. Не потрібно турбуватися про зв’язування та завантаження бібліотек тощо.

- Платформенно-незалежна. Програми Python можна розробляти та виконувати на декількох платформах операційних систем. Python можна використовувати в Linux, Windows, Macintosh, Solaris та багатьох інших.

- Безкоштовна та Open Source; Дистрибутивна

- Мова високого рівня. У Python не потрібно дбати про деталі низького рівня, такі як управління пам'яттю, якою користується програма.

- Проста. Ближча до англійської мови; Легка у навчанні. Більше акценту на вирішенні проблеми, а не на синтаксисі.

- Вбудована. Python можна використовувати в програмі C / C ++ для надання скриптових можливостей для користувачів програми.

- Стійка. Виняткові функції керування. Вбудовані Методи управління пам’яттю.

- Багата підтримка бібліотек. Стандартна бібліотека Python дуже велика. Відома філософія Python як "batteries included"; Можна робити різні

речі, пов'язані з регулярними виразами, генерацією документації, тестуванням додатків, базами даних, веб-браузерами, CGI, email, XML, HTML, файлами WAV, криптографією, графічним інтерфейсом та багатьма іншими функціями.

Машинне навчання - найгарячіша тенденція сучасності. За даними Forbes, патенти на машинне навчання постійно зростають, і це лише будуть зростати в майбутньому. А Python - це основна мова програмування, яка використовується для більшості досліджень та розробок у машинному навчанні [11]. Python є навіть головною мовою програмування для машинного навчання згідно Github. За даними Google Trends, інтерес до Python для машинного навчання піднявся на все новий максимум, коли інші мови машинного навчання, такі як R, Java, Scala, Julia тощо, значно відстають (рис. 3.1).

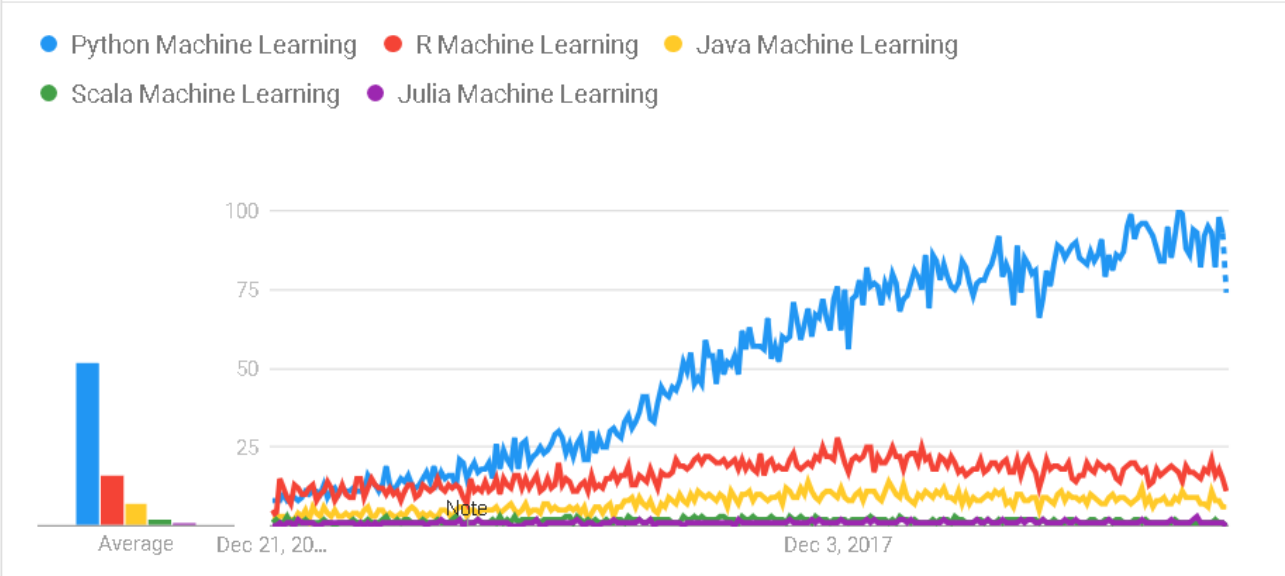

Рисунок 3.1 – Дані Google Trends про використання мов програмування для машинного навчання

Причинами зростання популярності Python для машинного навчання є наступні:

1. Python простий у використанні.

Ніхто не любить надмірно складні речі, тому простота використання Python є однією з головних причин, чому він настільки популярний для

машинного навчання. Простий синтаксис, який легко читається, робить його улюбленим як досвідченими розробниками, так і студентами-новачками. Простота Python означає, що розробники можуть зосередитись на фактичному вирішенні проблеми машинного навчання, а не витрачати весь свій час і енергію на розуміння лише технічних нюансів мови.

Окрім цього, Python також є надзвичайно ефективним. Це дозволяє розробникам реалізовувати алгоритми, використовуючи менше рядків коду. Код Python також легко зрозумілий людям, що робить його ідеальним для реалізації моделей машинного навчання.

2. У Python є кілька бібліотек та фреймворків.

Python вже досить популярний, а отже, він має сотні різних бібліотек та фреймворків, якими можуть користуватися розробники. Ці бібліотеки та фреймфорки дійсно корисні для економії часу, що, в свою чергу, робить Python ще більш популярним.

Існує багато бібліотек Python, які спеціально розроблені для штучного інтелекту та машинного навчання.

Деякі з них наведені нижче:

Keras - бібліотека з відкритим кодом, яка особливо зосереджена на експерименті з глибокими нейронними мережами.

TensorFlow - це безкоштовна бібліотека програмного забезпечення, яка використовується для багатьох алгоритмів машинного навчання, таких як нейронні мережі.

Scikit-learn - це безкоштовна бібліотека програмного забезпечення для машинного навчання, пов'язана з різними алгоритмами класифікації, регресії та кластеризації. Також Scikit-learn можна використовувати разом із NumPy та SciPy.

3. Python має спільноту та корпоративну підтримку.

Python існує з 1990 року, і це достатньо часу для створення спільноти його підтримки. Завдяки цій підтримці початківці можуть легко вдосконалити свої знання машинного навчання, що лише призводить до зростання

популярності Python. Крім того, в Інтернеті існує багато ресурсів для просування машинного навчання в Python, починаючи від навчальних посібників, онлайн-курсів та окремих постів для машинного навчання до YouTube-лекцій, які є великою допомогою для початківців.

Крім того, корпоративна підтримка є дуже важливою частиною успіху Python для машинного навчання. Багато кращих компаній, таких як Google, Facebook, Instagram, Netflix, Quora і т.д., використовують Python для своїх продуктів. Насправді Google несе відповідальність за створення багатьох бібліотек Python для машинного навчання, таких як Keras, TensorFlow тощо.

4. Python є портативним та розширюваним.

Це також важлива причина, чому Python настільки популярний у машинному навчанні. Багато крос-мовних операцій можна легко виконувати на Python через його портативний та розширюваний характер. Є багато науковців, які вважають за краще використовувати GPU для навчання своїх моделей на власних машинах, а портативний характер Python цілком підходить для цього.

Також багато різних платформ підтримують Python, такі як Windows, Macintosh, Linux, Solaris тощо. Крім цього, Python також може бути інтегрований з Java, .NET компонентами або бібліотеками C / C ++ через свою розширюваність.

3.2.1 Бібліотека Scikit-learn

Scikit-learn - це бібліотека з відкритим кодом Python, яка реалізує алгоритми машинного навчання, попередньої обробки, перехресної перевірки та візуалізації за допомогою уніфікованого інтерфейсу [14].

Особливостями бібліотеки scikit-learn є:

- Прості та ефективні інструменти для пошуку даних та аналізу даних. Бібліотека містить різні алгоритми класифікації, регресії та кластеризації, включаючи метод опорних векторів, випадкові ліси, збільшення градієнта, k-середні тощо.

- Доступна для всіх і може використовуватись в різних контекстах.
- Це надбудова до NumPy, SciPy та matplotlib.
- Відкритий код, комерційно доступна - ліцензія BSD. scikit-learn ліцензований за спрощеною ліцензією BSD і поширюється під багатьма дистрибутивами Linux, заохочуючи академічне та комерційне використання.

Scikit-learn забезпечує цілий ряд з вчителем та без вчителя алгоритмів навчання через послідовний інтерфейс у Python.

Бібліотека надбудована над SciPy (Scientific Python), який повинен бути встановлений перед тим, як використовувати scikit-learn.

Цей стек включає:

- NumPy: Базовий пакет n-мірних масивів.
- SciPy: Фундаментальна бібліотека для наукових обчислень.
- Matplotlib: побудова 2D / 3D графіків.
- IPython: вдосконалена інтерактивна консоль.
- Sympy: символьна математика.
- Pandas: Структура та аналіз даних.

Розширення або модулі для SciPy умовно названі SciKits. Як такий, модуль надає алгоритми навчання і називається scikit-learn.

Бачення бібліотеки - це рівень надійності та підтримки, необхідних для використання у виробничих системах. Це означає глибокий фокус на таких проблемах, як простота у використанні, якість коду, співпраця, документація та продуктивність. Хоча інтерфейсом є Python, c-бібліотеки є перевагою для використання numpy для масивів та матричних операцій, LAPACK, LibSVM та cython.

Деякі популярні групи моделей, які надає scikit-learn, включають:

- Кластеризація: для групування нерозмічених даних, такими методами як KMeans.
- Перехресна перевірка: для оцінки продуктивності контрольованих моделей для майбутніх наборів даних.

- Набори даних: для тестування наборів даних та для генерування наборів даних із специфічними властивостями для дослідження поведінки моделі.

- Зменшення розмірності: для зменшення кількості атрибутів у даних для узагальнення, візуалізації та вибору ознак, таких як аналіз основних компонентів (PCA).

- Ансамблеві методи: для поєднання прогнозів декількох контрольованих моделей.

- Вилучення атрибутів: для визначення атрибутів у зображенні та текстових даних.

- Вибір функцій: для визначення значущих атрибутів, з яких можна створити контрольовані моделі.

- Налаштування параметрів: для отримання максимальної користі від контрольованих моделей.

- Manifold Learning: для узагальнення та зображення складних багатовимірних даних.

- Контроловані моделі (з вчителем): величезний масив, не обмежений узагальненими лінійними моделями, дискримінаційним аналізом, нметодом наївного Байєса, ледачими методами, нейронними мережами, методами опорних векторів та деревами прийняття рішень.

3.2.2 Бібліотека Pandas

Pandas - це бібліотека Python з відкритим кодом, що забезпечує високопродуктивний інструмент для обробки та аналізу даних, використовуючи потужні структури даних. Назва Pandas походить від слова Panel Data - економетрика з багатовимірних даних. Pandas побудований поверх пакету Numpy, тобто Numpy необхідний для роботи з Pandas.

У 2008 році розробник Уес Маккінні почав розробляти Pandas, коли виникла потреба високопродуктивного, гнучкого інструменту для аналізу даних.

До Pandas Python в основному використовувався для обробки та підготовки даних. Це був малий вклад у аналіз даних. Бібліотека Pandas вирішила цю проблему. Використовуючи Pandas, ми можемо здійснити п’ять типових кроків в обробці та аналізі даних, незалежно від походження даних - завантажувати, підготовлювати, маніпулювати, моделювати та аналізувати.

Python з Pandas використовується в широкому спектрі галузей, включаючи наукові та комерційні сфери, включаючи фінанси, економіку, статистику, аналітику тощо.

Основні особливості Pandas:

- Швидкий та ефективний об'єкт DataFrame з спеціалізованим індексуванням за замовчуванням.
- Інструменти для завантаження даних в об'єкти даних пам'яті з різних форматів файлів.
- Вирівнювання даних та інтегрована обробка відсутніх даних.
- Переформатування та зміна датасетів.
- Нарізання, індексація та виділення підмножин великих наборів даних на основі міток.
- Стовпці зі структури даних можна видалити або вставити.
- Групування даних для агрегації та перетворення.
- Висока продуктивність об'єднання та з'єднання даних.
- Функціональність часових рядів.

Pandas має дві структури даних для їх обробки

1. Series, що визначається як одновимірний масив, який здатний зберігати різні типи даних. Мітки рядків називаються індексом. Ми можемо легко перетворити список, кортеж та словник у серії за допомогою методу "series". Серія не може містити декілька стовпців. У неї є один параметр. Дані: це може бути будь-який список, словник або скалярне значення.

2. DataFrame. Це широко використовувана структура даних Pandas, що працює з двовимірним масивом з міченими осями (рядки та стовпці). DataFrame визначається як стандартний спосіб зберігання даних і має два різні

індекси, тобто індекс рядків та індекс стовпців. Він складається з таких властивостей: Стовпці можуть бути різнорідними типами, такими як int, bool тощо. Його можна розглядати як словник структури Series, де індексуються і рядки, і стовпці. Позначається як "стовпці" у разі стовпців та "індекс" у разі рядків.

3.2.3 Бібліотека NLTK

NLTK (Natural Language Toolkit) - це провідна платформа для побудови програм Python для роботи з даними людської мови [6]. Вона надає прості у користуванні інтерфейси для багатьох корпоративних та лексичних ресурсів. Крім того, вона містить набір бібліотек обробки текстів для класифікації, токенізації, стемінгу, тегування, парсингу та семантичного аналізу. І що найголовніше, NLTK - це безкоштовний проект з відкритим кодом.

Основні модулі NLTK

- токени: класи для представлення та обробки окремих елементів тексту, таких як слова та речення;
- ймовірність: імовірнісна інформація;
- дерево: ієрархічні структури над текстом;
- cfg: граматики без контексту;
- fsa: скінченні автомати;
- теггер: тегування кожного слова з частиною мови, почуттям і т.д.;
- аналізатор: побудова дерев над текстом – діаграмою, фрагментом;
- класифікатор: класифікує текст на категорії;
- візуалізація: візуалізує структури та процеси NLP.

3.3 Підготовка середовища

Python – це така собі екосистема з великою кількістю бібліотек, які полегшують програмування в тій чи іншій області. Завдяки існуванню таких

бібліотек програмісти можуть використати безліч готових функцій замість написання довгих сторінок коду.

Тому нас спершу необхідно встановити бібліотеки, необхідні для роботи з великими масивами даних та обробкою тексту.

В пунктах 3.2.1, 3.2.2 та 3.2.3 наведено основні відомості про бібліотеки scikit-learn, pandas, nltk, які нам потрібні. Крім того необхідно буде встановити бібліотеку wordcloud та інші. В Лістингу 3.1 наведено підключення необхідних бібліотек

Лістинг 3.1 – Підключення необхідних бібліотек

```
import nltk
import os
import glob
import math
import re
import pandas as pd
from pathlib import Path
from nltk.tokenize import sent_tokenize, word_tokenize

from nltk.stem import PorterStemmer, SnowballStemmer, LancasterStemmer
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_selection import SelectPercentile, f_classif

from sklearn.naive_bayes import GaussianNB

from sklearn.metrics import accuracy_score, recall_score, precision_score, f1_score

from matplotlib import pyplot as plt
```

Pip – це система управління пакетами, що містять всі необхідні файли бібліотек, написаних на Python.

В Лістингу 3.2 наведено перевірку підключення необхідних бібліотек через систему pip.

Лістинг 3.2 – Перевірка підключення бібліотек

```
!pip install nltk
!pip install pandas
!pip install scikit-learn
```

Крім того, бібліотеки потрібно конфігурувати під свої потреби. В Лістингу 3.3 наведено завантаження моделей та наборів даних, які в подальшому будуть використані в роботі.

Лістинг 3.3 – Завантаження наборів даних для токенізації

```
nltk.download("punkt", download_dir=NLTK_DATA_DIR)
nltk.download("wordnet", download_dir=NLTK_DATA_DIR)
nltk.download("stopwords", download_dir=NLTK_DATA_DIR)
```

Здійснивши попереднє налаштування системи, можемо переходити до завантаження даних.

3.4 Завантаження даних

В п.3.1 описано датасет, який ми використали для даної дипломної роботи. Тож для завантажимо спершу навчальну вибірку, яка міститься в папці train. Ця папка містить підлеглі папки з іменами авторів творів (AlexanderSmith, BenjaminKangLim і т.д.) В папці з іменем автора міститься по 50 документів з уривками творів відповідного автора. Кожний текстовий документ містить орієнтовно 1000 слів.

В Лістингу 3.4 наведено частину програмного коду, що відповідає за завантаження даних

Лістинг 3.4 – Завантаження даних

```
    # zero or more directories and subdirectories.
    files = [(Path(f).parent.name, f) for f in
glob.glob("train/**/*.txt", recursive=True)]
    # Sort the files by relative file path
    files.sort(key=lambda tup: tup[1])
    for author, file in files:
        print(author, file)
```

Далі необхідно створити об'єкт Dataframe з метою подальшого використання функцій pandas для його обробки. Створюємо датафрейм з наступною структурою: він буде містити лише два стовпці, де в першому стовпці потрібно поставити автора документу (ім'я папки), а в другий стовпець завантажуєм вміст текстових файлів з текстами, відповідних авторів. Крім того, необхідно посортувати дані за відносних шляхом до файлу. В лістингу 3.5 наведено код функцій для створення потрібного датафрейму.

Лістинг 3.5 – Функції для створення датафрейму

```
def load_data(data_dir):
    authors = []
    texts = []
    df = pd.DataFrame ({'Author': authors, 'Text': texts})
    files = [(Path(f).parent.name, f) for f in
glob.glob("train/**/*.txt", recursive=True)]
    files.sort(key=lambda tup: tup[1])
    df.append?
    for author, file in files:
        with open(file, 'r') as theFile:
            data = theFile.read()
        df = df.append({"Author": author, "Text":
data},ignore_index=True)

    return df

def fix_data_types(df):
    df['Author'] = df.Author.astype(str)
    df['Text'] = df.Text.astype(str)
    return df

def load_train_data():
    return fix_data_types(load_data('train'))

def load_test_data():
    return fix_data_types(load_data('test'))
```

В Лістингу 3.5 сформували структуру датасетів для навчальних та тестових даних.

В лістингу 3.6 код для завантаження даних

Лістинг 3.6 – Завантаження навчальних даних

```
train_df = load_train_data()
train_df.info()
train_df.head(10)
```

Результати роботи коду з лістингу 3.6 зображено на рисунку 3.2

	Author	Text
0	AaronPressman	The Internet may be overflowing with new technology but crime in cyberspace is still of the old-fashioned variety.\nThe National Consumers League ...
1	AaronPressman	The U.S. Postal Service announced Wednesday a plan to boost online commerce by enhancing the security and reliability of electronic mail traveling...
2	AaronPressman	Elementary school students with access to the Internet learned more than kids who lacked access, an indepedent research group concluded after cond...
3	AaronPressman	An influential Internet organisation has backed away from a proposal to dramatically expand the number of addresses available on the global comput...
4	AaronPressman	An influential Internet organisation has backed away from a proposal to dramatically expand the number of addresses available on the global comput...
5	AaronPressman	A group of leading trademark specialists plans to release recommendations aimed at minimizing disputes over Internet address names.\nThe Internati...
6	AaronPressman	When a company in California sells a book to a consumer in Canada from a Web site hosted on a computer in England, what laws govern the transactio...
7	AaronPressman	U.S. laws governing the trillion dollar futures markets could be rocked by the Supreme Court's interpretation of the word "in" in a case to be arg...
8	AaronPressman	Supreme Court justices Wednesday sharply questioned rules governing so-called derivative investments and foreign currency trading in a case that c...
9	AaronPressman	The Internet continued to grow in leaps and bounds this year while online services found it much harder to add new customers, a survey said Friday...

Рисунок 3.2 – Датафрейм з навчальною вибіркою авторів та їх творів

Аналогічно завантажуємо тестові дані

```
test_df = load_test_data()
test_df.info()
test_df.head(10)
```

Навчальна та тестова вибірки містять по 2500 записів.

Необхідно перевірити, чи є даний датасет збалансованим. Нагадаємо, що датасет називається збалансованим якщо кількість запитів в для кожної мітки є приблизно однаковою. Наступна стрічка коду здійснює перевірку на збалансованість (лістинг 3.7).

Лістинг 3.8 – Перевірка датасету на збалансованість

```
train_df.groupby("Author").count()
```

Результатом цієї операції є кількість документів для кожного автора (мітки) у даному датасеті (рисунок 3.3). Очевидно, що датасет є збалансованим і містить одинакову кількість записів для кожної мітки

Author	Text
AaronPressman	50
AlanCrosby	50
AlexanderSmith	50
BenjaminKangLim	50
BernardHickey	50
BradDorfman	50
DarrenSchuettler	50
DavidLawder	50
EdnaFernandes	50
EricAuchard	50
FumikoFujisaki	50
GrahamEarnshaw	50
HeatherScoffield	50
JanLopatka	50
JaneMacartney	50
JimGilchrist	50
JoWinterbottom	50
JoeOrtiz	50
JohnMastrini	50

Рисунок 3.3 – Результат виконання лістингу 3.8

Якщо датасет збалансований, то можна використовувати практично будь-який алгоритм машинного навчання. Якщо датасет розбалансований, спочатку необхідно його додатково обробити (збалансувати), або використовувати алгоритми, що вміють працювати з незбалансованими датасетами.

3.5 Очищення та нормалізація даних

Для виділення ключових слів та побудови простору ознак необхідно спершу провести очищення та нормалізацію текстових даних.

Спершу необхідно очистити текст від зайвих символів, таких як новий рядок, знаки пунктуації та інші (лістинг 3.9)

Лістинг 3.9 – Функція для очищення тексту від зайвих символів та приведення до одного регістру

```
def normalize_text(text):

    replace=['\n',',','.','-','$','!','"','?',"'"]
    for elem in replace :
        if elem in text :
            text = text.replace(elem, " ")

    normalized_text = text.lower()
    return normalized_text
```

Наступна функція (лістинг 3.10) робить поділ тексту на окремі слова

Лістинг 3.10 – Функція для поділу тексту на слова

```
def tokenize_text(text):
    """
    Splits the input string into a list of lexical units
    """
    tokenized = []
    tokenized=word_tokenize(text)
    return tokenized
```

Стрічка коду `words = pd.Series(tokenize_text(normalize_text (combined_text)))` створює структуру даних Series під назвою words з використанням попередньо описаних функцій та змінної `combined_text`, яка, фактично є об'єднанням усіх текстів датасету. words – це, фактично, список усіх слів, які зустрічаються в тексті усіх документів. У даному датасеті виявилось 1287667 слів та 34764 унікальних слів. Зрозуміло, що такий розмір простору інформативних ознак є занадто великим і потребує скорочення.

Доцільно було б взяти лише якусь прийнятну частину ключових слів як атрибутів для задачі класифікації.

Для проведення класифікації перетворюємо стовпець Text в навчальному та тестовому датасетах зі стрічки у масив слів, використовуючи функцію normalize_text (лістинг 3.9) та tokenize_text (лістинг 3.10), як це показано в лістингу 3.11

Лістинг 3.11 – Перетворення та очищення даних

```
train_df['Text'] = train_df['Text'].apply(lambda x: list(tokenize_text(normalize_text(x))))
test_df['Text'] = test_df['Text'].apply(lambda x: list(tokenize_text(normalize_text(x))))
```

Результати виконання лістингу 3.11 наведені на рисунку 3.4.

	Author	Text
0	AaronPressman	[the, internet, may, be, overflowing, with, new, technology, but, crime, in, cyberspace, is, still, of, the, old-fashioned, variety, the, national...
1	AaronPressman	[the, u, s, postal, service, announced, wednesday, a, plan, to, boost, online, commerce, by, enhancing, the, security, and, reliability, of, elect...
2	AaronPressman	[elementary, school, students, with, access, to, the, internet, learned, more, than, kids, who, lacked, access, an, indepedent, research, group, c...
3	AaronPressman	[an, influential, internet, organisation, has, backed, away, from, a, proposal, to, dramatically, expand, the, number, of, addresses, available, o...
4	AaronPressman	[an, influential, internet, organisation, has, backed, away, from, a, proposal, to, dramatically, expand, the, number, of, addresses, available, o...
5	AaronPressman	[a, group, of, leading, trademark, specialists, plans, to, release, recommendations, aimed, at, minimizing, disputes, over, internet, address, nam...
6	AaronPressman	[when, a, company, in, california, sells, a, book, to, a, consumer, in, canada, from, a, web, site, hosted, on, a, computer, in, england, what, la...
7	AaronPressman	[u, s, laws, governing, the, trillion, dollar, futures, markets, could, be, rocked, by, the, supreme, court, 's, interpretation, of, the, word, in...
8	AaronPressman	[supreme, court, justices, wednesday, sharply, questioned, rules, governing, so-called, derivative, investments, and, foreign, currency, trading, ...

Рисунок 3.4 – Результат виконання лістингу 3.11

До 30 найбільш вживаних слів в датасеті належать 'the', 'to', 'of', 'a', 'in', 'and', 'said', "'s", 'for', 'on', 'that', 'is', 'it', 'with', 'be', 'by', 'its', 'as', 'at', 'was', 'from', 'he', 'will', 'but', 'has', 'have', 'would', 'percent', 'are', 'million'.

Очевидно ці слова важко назвати ключовими при визначенні авторства. Тому перед скороченням простору слів, необхідно спершу видалити стоп-слова та обчислити значення атрибутів, що залишились, як показник TF-IDF.

3.5.1 Скорочення простору атрибутів

Проведемо видалення стоп-слів словниковим методом. Наступний код дозволяє вивести кількість стоп-слів в англійській мові та переглянути, які зі слів належать до неключових слів побудови простору ознак, визначених вбудованою функцією `stopwords.words:`

```
print(len(stopwords.words('english')))
stopwords.words('english')
```

Виявилось, що до цього списку належать 179 слів, таких як 'i', 'me', 'my', 'myself', 'we', 'our', 'ours', і тому подібні.

Для видалення стоп-слів з обох датасетів (навчального та тестового) використано код, наведений в лістингу 3.12

Лістинг 3.12 – Видалення шумових слів

```
def delete_stopwords(l,s):
    i=0
    while i < len(l):
        if l[i] in s:
            del l[i]
        else:
            i+=1
    return l

st = set(stopwords.words('english'))
func = lambda x: delete_stopwords(x, st)
train_df['Text'] = train_df['Text'].apply(func)
test_df['Text'] = test_df['Text'].apply(func)
train_df
```

Після цього необхідно обчислити показники TF-IDF для слів, присутніх в датасеті, встановити порогове значення і відібрати, скажімо 30% слів для проведення класифікації. Для цього створюємо об'єкт класу TfidfVectorizer та конфігуруємо його з наступними параметрами:

- sublinear_tf=True
- max_df=0.5
- stopwords='english'

3.5.2 Лематизація

Імплементувати лематизацію можна в наступних пакетах Python

1. Wordnet Lemmatizer
2. Spacy Lemmatizer
3. TextBlob
4. CLiPS Pattern
5. Stanford CoreNLP
6. Gensim Lemmatizer
7. TreeTagger

В роботі проведено лематизацію даних з використанням WordNetLemmatizer(). На рисунку 3.5 приведено результати роботи даного лематизатора.

```
In [141]: wnl = WordNetLemmatizer()
          print(wnl.lemmatize('getting', 'v'))
          print(wnl.lemmatize('rabbits', 'n'))
          print(wnl.lemmatize('quickly', 'r'))
          print(wnl.lemmatize('slowly', 'r'))
          # KeyError! - Doesn't work on non-words!
          print(wnl.lemmatize('xyzing', ''))

          get
          rabbit
          quickly
          slowly

          ---------------------------------------------------------------------------
          KeyError                                  Traceback (most recent call last)
          <ipython-input-141-1f313a6f6b78> in <module>
                5 print(wnl.lemmatize('slowly', 'r'))
                6 # KeyError! - Doesn't work on non-words!
          ----> 7 print(wnl.lemmatize('xyzing', ''))

          ~/anaconda3/lib/python3.7/site-packages/nltk/stem/wordnet.py in lemmatize(self, word, pos)
               39
               40     def lemmatize(self, word, pos=NOUN):
          ---> 41         lemmas = wordnet._morphy(word, pos)
               42         return min(lemmas, key=len) if lemmas else word
               43

          ~/anaconda3/lib/python3.7/site-packages/nltk/corpus/reader/wordnet.py in _morphy(self, form, pos, check_exceptions)
             1879         #    find a match or you can't go any further
             1880
          -> 1881         exceptions = self._exception_map[pos]
             1882         substitutions = self.MORPHOLOGICAL_SUBSTITUTIONS[pos]
             1883

          KeyError: ''
```

Рисунок 3.5 – Результати роботи вбудованого лематизатора

Очевидно, що вбудований в Python лематизатор доволі непогано визначає частини мови та приводить слова до їх початкової форми. Для безмістовного слова xyzing він видає помилку.

3.5.3 Стеммінг

Відповідно до описаних в пункті 2.4 моделей існують різні фунції Python, що відповідають цим моделям:

```
stemmers = [PorterStemmer(), SnowballStemmer("english"), LancasterStemmer()]
```

Відповідно до документації Python, SnowballStemmer – це удосконалена модель PorterStemmer, де виправлено декілька проблем з суфіксами.

В роботі використовували стемер Постера. В лістингу 3.13 наведено код програми, що виконує стемування документів.

Лістинг 3.13 – Стемування

```
def normalize_words(words):
    stemmer = PorterStemmer()
    for i in range(len(words)):
        words[i] = stemmer.stem(words[i])
    return words
train_df['Text'] = train_df['Text'].apply(lambda x: normalize_words(x))
test_df['Text'] = test_df['Text'].apply(lambda x: normalize_words(x))
train_df
```

На рисунку 3.6 наведено результат виконання алгоритму стемування

	Author	Text
0	AaronPressman	[internet, may, overflow, new, technolog, crime, cyberspac, still, old-fashion, varieti, nation, consum, leagu, said, wednesday, popular, scam, in...
1	AaronPressman	[u, postal, servic, announc, wednesday, plan, boost, onlin, commerc, enhanc, secur, reliabl, electron, mail, travel, internet, plan, busi, consum,...
2	AaronPressman	[elementari, school, student, access, internet, learn, kid, lack, access, indeped, research, group, conclud, conduct, compar, studi, seven, urban,...
3	AaronPressman	[influenti, internet, organis, back, away, propos, dramat, expand, number, address, avail, global, comput, network, internet, societi, help, devel...
4	AaronPressman	[influenti, internet, organis, back, away, propos, dramat, expand, number, address, avail, global, comput, network, internet, societi, help, devel...
5	AaronPressman	[group, lead, trademark, specialist, plan, releas, recommend, aim, minim, disput, internet, address, name, intern, trademark, associ, work, white,...
6	AaronPressman	[compani, california, sell, book, consum, canada, web, site, host, comput, england, law, govern, transact, ?, sale, kind, good, internet, continu,...
7	AaronPressman	[u, law, govern, trillion, dollar, futur, market, could, rock, suprem, court, 's, interpret, word, case, argu, wednesday, legisl, solut, of, issu,...
8	AaronPressman	[suprem, court, justic, wednesday, sharpli, question, rule, govern, so-cal, deriv, invest, foreign, currenc, trade, case, could, wide, repercuss, ...
9	AaronPressman	[internet, continu, grow, leap, bound, year, onlin, servic, found, much, harder, add, new, custom, survey, said, friday, estim, , million, adult, ...
10	AaronPressman	[hewlett-packard, co, unveil, new, plan, monday, boost, sale, comput, encod, technolog, consid, key, compon, global, commun, develop, commerc, int...
11	AaronPressman	[internet, continu, grow, leap, bound, year, onlin, servic, found, much, harder, add, new, custom, new, survey, say, estim, , million, adult, unit...
12	AaronPressman	[grow, busi, busi, internet, pose, major, challeng, tax, collector, special, tax, impos, cyberspac, govern, said, report, releas, thursday, treasu...

Рисунок 3.6 – Результати виконання алгоритму стемування

На рисунку 3.6 видно, що частина слів залишились без змін, а частина скоротились до певного кореня. Слід зазначити, що під коренем, в даному випадку, маємо на увазі не корінь, що визначається в результаті морфемного розбору слова за будовою, а деяку спільну частину для споріднених слів.

3.6 Побудова моделі класифікації

В основу класифікаційної моделі покладено модель наївного Байєса. В Python моделі обробки даних зазвичай мають методи fit_transform та transform. Метод fit_transform використовується одночасно для навчання моделі на даних та їх перетворення, а метод transform виключно для перетворення даних (з використанням попередньо навченої моделі).

Моделі класифікації даних зазвичай мають методи fit та predict. Метод fit передбачає навчання моделі, а метод predict - використання вже навченої моделі для отримання прогнозних значень. Тож необхідно використовувати методи fit та fit_transform виключно з навчальними даними, а для тестових даних використовувати tranform та predict. В лістингу 3.14 наведено код програми з налаштуваннями класифікаційної моделі.

Лістинг 3.14 – Код програми побудови класифікаційної моделі

```
	vectorizer = TfidfVectorizer(sublinear_tf=True, max_df=0.5,
stop_words='english')
	train_df['Text'] = train_df['Text'].apply(lambda x: '
'.join(x))
	test_df['Text'] = test_df['Text'].apply(lambda x: '
'.join(x))
	features_train_transformed =
vectorizer.fit_transform(train_df['Text'])
	features_test_transformed =
vectorizer.transform(test_df['Text'])
	selector = SelectPercentile(f_classif, percentile = 20)
	selector.fit(features_train_transformed, train_df['Author'])
	features_test_transformed_optimized =
selector.transform(features_test_transformed)
	features_train_transformed_optimized =
selector.transform(features_train_transformed)
	clf = GaussianNB()
	clf.fit(features_train_transformed_optimized.toarray(),
train_df['Author'])
	predicted =
clf.predict(features_test_transformed_optimized.toarray())
```

Функція GaussianNB вказує на те, що для класифікації використовували гаусівську модель наївного Байєса.

3.7 Оцінка точності методу класифікації

Для оцінки точності побудованого класифікатора використано метрики, запропоновані в пункті 2.11, а саме: достовірність, точність та повнота моделі.

Оцінка моделі проводилась для різних значень частки ключових слів з загального простору ознак, що використовувались для класифікації. В таблиці 3.1 наведено порівняння критеріїв якості моделі для різної кількості коефіцієнтів.

Таблиця 3.1 – Порівняльний аналіз точності класифікаційної моделі для різних квантилів кількості термінів в словнику ознак

Квантиль / Критерій	10%	30%	40%	50%
Достовірність (accuracy)	0,577	0,584	0,585	0,594
Точність (Precision)	0,61	0,627	0,637	0,643
Повнота (Recall)	0,577	0,584	0,585	0,594

Загальний показник обчислювався як середньо-зважений відповідно до кількості правдивих випадків для кожного класу. Результати наведені в таблиці підтверджують припущення про те, що з числом ознак зростає точність моделі. Проте зростання точності настільки мізерне в порівнянні до кількості потрібних ресурсів, що не видається доцільним.

В таблиці 3.2 наведено порівняння показників точності для різних налаштувань класифікаційної моделі.

Таблиця 3.2 – Порівняння показників точності моделі для різних налаштувань моделі для 30%-квантиля ознак, що використовувались для класифікації

Налашту-вання / Критерій	Без нормалізації	Stop-words	Stemming
Достовірність (accuracy)	0,584	0.582	0,558
Точність (Precision)	0,627	0.628	0,614
Повнота (Recall)	0,584	0.582	0,558

Всупереч загально-прийнятій думці, що додаткова нормалізація датасету призводить до покращених результів в даному випадку, вилучення шумових слів фактично не вплинуло на точність, а після стемінгу точність навіть дещо погіршилась. Це ще раз довоить факт, що проблема ідентифікації автора за текстовим документом є надзвичайно складною науковою задачею, яка потребує подальших досліджень.

3.8 Висновки до розділу 3

В третьому розділі:

- описано вибір програмного середовища для задачі ідентифікації автора за текстовим документом;
- описано всі етапи практичної реалізації підготовки даних та налаштувань класифікаційної моделі;
- проведено тестування моделі на реальних даних;
- проведено порівняльний аналіз точності класифікаційної моделі для різних параметрів.

4 СПЕЦІАЛЬНА ЧАСТИНА

4.1 Основні типи даних в Python

В Python кожне значення має тип, але немає необхідності оголошувати тип змінної. Покладаючись на початково присвоєне змінній значення, Python визначає якому типу воно належить і запам’ятовує його самостійно. Таку типізацію називають неявною.

Python має багато вбудованих типів даних. Ось деякі найважливіші:

- Логічні (Булеві) змінні приймають значеня True або False.
- Числа можуть бути цілими (1 і 2), з десятковими дробами (1.1 і 1.2), звичайними дробами (1/2 and 2/3), чи навіть комплексними.
- Рядки (послідовності символів (наприклад 'To be or not to be' або ж цілий HTML документ)
- Байти та масиви байтів (наприклад зображення в форматі JPEG)
- Списки (впорядковані послідовності значень).
- Кортежі (впорядковані незмінні послідовності значень).
- Множини (невпорядковані набори значень).
- Словники (невпорядковані набори пар ключ-значення).

Для початку розглянемо самі базові типи.

4.1.1 Булевий тип (bool)

Булеві змінні приймають лише істинне чи хибне значення. Python має для цих значень дві відповідні константи: True та False, які можна використовувати для присвоєння значень змінним. Окрім констант, булеві значення можуть приймати вирази:

```
>>> is_file_empty = True
>>>
```

4.1.2 Числа

Python підтримує як цілі, так і дробові числа. Вказувати тип числа явно не потрібно, Python сам визначить його за наявністю чи відсутністю десяткової крапки.

Цілі числа (int)

Числа в Python 3 нічим не відрізняються від звичайних чисел. Вони підтримують набір самих звичайних математичних операцій:

Також слід зазначити, що цілі числа в Python, на відміну від багатьох інших мов програмування, підтримують довгу арифметику.

Дробові числа (float)

У дробових числах ціла частина від дробової відділяється знаком крапки.

Дробові числа підтримують ті ж операції, що й цілі. Однак через особливості представлення чисел у пам'яті комп'ютера дробові числа неточні, це може спричиняти помилки при обчисленнях:

Для високої точності використовують інші засоби (наприклад Decimal і Fraction). Також дробові числа не підтримують довгу арифметику.

Рядок (str)

Символьний рядок (або просто рядок) — це упорядкована послідовність символів яку використовують для зберігання та представлення текстової інформації.

Щоб створити рядок символи треба заключити у лапки або апострофи. Якщо рядок містить, наприклад, лапки, тоді його треба заключити в апострофи і навпаки:

Інтерпретатор виводить значення рядкових змінних саме так, як ми їх визначили, тобто разом з лапками у які ми заключили символи.

Щоб Python показав "справжній" рядок, лише символи які у ньому містяться, необхідно написати print і зразу ж за цим словом у круглих дужках вказати рядок чи змінну яка на нього посилається:

4.1.3 Екрановані послідовності

Екрановані послідовності дозволяють включити у рядок символи, які важко ввести з клавіатури.

Екранована послідовність виглядає так: символ бекслеш (зворотній слеш) і зразу за ним один або декілька символів. Ось деякі такі послідовності:

- \n – новий рядок;
- \r – початок рядка;
- \t – табуляція;
- \a - Дзвінок";
- \' – апостроф;
- \" – лапки;
- \ \ - бекслеш.

Коли ми вказуємо інтерпретатору ім'я рядкової змінної, то екрановані послідовності виводяться саме так, як ми їх вказали. Щоб "задіяти" екрановані послідовності треба скористатись print.

Рядки у потрійних лапках чи апострофах

Якщо треба визначити блок тексту який складається з декількох рядків, тоді такі послідовності символів заключають у потрійні лапки чи апострофи. Причому ми маємо змогу вільно користуватись звичайними лапками чи апострофами усередині таких блоків:

4.1.4 Список (list)

Списки — це як масиви у інших мовах програмування, але набагато крутіше!

Список — це упорядкована структура, яка може містити у собі об'єкти, причому об'єкти різних типів.

Створити список дуже просто — потрібно помістити список значень, розділених комою, в квадратні дужки.

До окремих об'єктів списку можна дістатись вказавши його індекс, або порядковий номер якщо простіше. Нумерація завжди починається з нуля.

Від’ємні індекси в масиві задають відповідні по порядку елементи, якщо рахувати справа наліво. Останнім елементом непорожнього списку a_list завжди є a_list[-1].

Списки - дуже потужний і гнучкий інструмент. Щоб дізнатись довжину списка, тобто кількість елементів у ньому, використовують вбудовану у Python функцію len.

4.1.5 Кортеж (tuple)

Кортеж - це незмінюваний список. Він завжди залишається таким, яким його створили.

Кортеж записується майже так само, як список, але замість квадратних дужок використовуються круглі.

Елементи кортежу мають визначений порядок, так само, як і в списку. Так само індексація починається з нуля. Від'ємні індекси діють так само, як у списків.

Основна відмінність між списками та кортежами: кортежі не можна змінювати. Технічно — вони незмінні. Практично — вони не мають засобів, які б дозволили вам змінити їх.

4.1.6 Словник (dict)

Словники — це невпорядковані набори пар "ключ - значення". Коли ви додаєте до словника новий ключ, завжди також повинні додати туди і значення (яке можна буде змінити пізніше). Словники оптимізовані для отримання значення, якщо ви знаєте відповідний ключ, але ніяк не навпаки.

Щоб створити словник, треба у фігурних дужках перерахувати пари "ключ - значення". Якщо таких пар більше однієї, їх розділяють комою. У парі ключ від значення відділяють двокрапкою.

Щоб отримати зі словника значення по його ключу вказують словник і у квадратних дужках ключ. Можна використовувати тільки ті ключі, які вже містяться у словнику.

Словники не мають попередньо заданого обмеження в розмірі. Ви в будь-який час можете додавати до словника нові пари "ключ-значення" чи змінювати значення чинної пари. Словник не може містити дублікатів ключів. Присвоєння значення чинному ключу затре старе значення. Словник — це невпорядкована структура.

4.2 Структури даних Pandas

Бібліотека pandas надає дві структури: Series і DataFrame для швидкої і зручної роботи з даними (насправді їх три, є ще одна структура - Panel, але в даний момент вона перебуває в статусі deprecated і в майбутньому буде виключена зі складу бібліотеки pandas). Series - це маркована одновимірна структура даних, її можна уявити, як таблицю з одним рядком. З Series можна працювати як зі звичайним масивом (звертатися за номером індексу), і як з асоційованим масивом, коли можна використовувати ключ для доступу до елементів даних. DataFrame - це двовимірна маркована структура. Ідейно вона дуже схожа на звичайну таблицю, що виражається в способі її створення і роботи з її елементами. Panel - про який було сказано, що він незабаром буде виключений з pandas, являє собою тривимірну структуру даних. Зупинимося на питаннях створення і отримання доступу до елементів даних структур Series і DataFrame.

4.2.1 Структура даних Series

Конструктор класу Series виглядає наступним чином:

```
pandas.Series (data = None, index = None, dtype = None, name = None, copy = False, fastpath = False)
```

де:

- data - масив, словник чи скалярне значення, на базі якого буде побудований Series;
- index - список міток, який буде використовуватися для доступу до елементів Series. Довжина списку повинна бути дорівнює довжині data;
- dtype - об'єкт numpy.dtype, що визначає тип даних;

- copy - створює копію масиву даних, якщо параметр дорівнює True в іншому випадку нічого не робить.

У більшості випадків, при створенні Series, використовують тільки перші два параметра. Розглянемо різні варіанти як це можна зробити.

Найпростіший спосіб створити Series - це передати в якості єдиного параметра в конструктор класу список Python

```
s1 = pd.Series([1, 2, 3, 4, 5])
```

Для доступу до елементів Series, в даному випадку, можна використовувати тільки позитивні цілі числа .

Можна спробувати використовуватися більше можливостей з тих, що пропонує pandas, для цього передамо в якості другого елементу список рядків (в нашому випадку - це окремі символи). Такий крок дозволить нам звертатися до елементів структури Series не тільки за чисельним індексу, а й по мітці, що зробить роботу з таким об'єктом, схожою на роботу зі словником.

```
s2 = pd.Series([1, 2, 3, 4, 5], ['a', 'b', 'c', 'd', 'e'])
```

Можна створити Series з масиву із numpy:

```
ndarr = np.array([1, 2, 3, 4, 5])
numpy.ndarray
s3 = pd.Series(ndarr, ['a', 'b', 'c', 'd', 'e'])
```

4.2.2 Структура даних DataFrame

Якщо Series являє собою одновимірну структуру, яку для себе можна уявити як таблицю з одним рядком, то DataFrame - це вже двовимірна структура - повноцінна таблиця з безліччю рядків і стовпців.

Конструктор класу DataFrame виглядає так:

```
class pandas.DataFrame (data = None, index = None, columns = None, dtype = None, copy = False)
```

де:

- data - масив ndarray, словник (dict) або інший DataFrame;
- index - список міток для записів (імена рядків таблиці);
- columns - список міток для полів (імена стовпців таблиці);
- dtype - об'єкт numpy.dtype, що визначає тип даних;

- copy - створює копію масиву даних, якщо параметр дорівнює True в іншому випадку нічого не робить.

Структуру DataFrame можна створити на базі:

- словника (dict) в якості елементів якого повинні виступати: одномірні ndarray, списки, інші словники, структури Series;
- двовимірні ndarray;
- структури Series;
- структуровані ndarray;
- інші DataFrame.

4.3 Висновки до розділу 4

В даному розділі описано основні типи та структури даних Python та бібліотеки Pandas, які використовувались в практичній реалізації методу визначення авторства документу. В розділі також наведено базові інструкції для створення та характеристик основних типів даних та структур.

5 ОБҐРУНТУВАННЯ ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ

Метою дипломної роботи є розробка програмного забезпечення для розв’язання задачі мультикласифікації –ідентифікації автора документу та проведення оцінки точності методу для реальних даних. Середовищем розробки було обрано було обране відкрите (Open Source) програмне забезпечення Python, більшість бібліотек якого є також безкоштовними.

5.1 Розрахунок норм часу на виконання науково-дослідної роботи

Ефективне використання часу має велике значення тому, що коефіцієнт корисної дії залежить від оптимального використання часу.

Реалізація та тестування моделі визначення авторства можна поділити на декілька етапів, що дозволяє полегшити і структурувати виконання поставленого завдання.

Основні етапи такі:

1. Пошук літературних джерел з області дослідження.
2. Обґрунтування моделі класифікації.
3. Вибір середовища розробки програмного забезпечення.
4. Розробка програми.
5. Тестування розробленої моделі на реальному датасеті.
6. Дослідження впливу на точність розмірності простору ключових слів та деяких етапів нормалізації текстових даних.

Для оцінки тривалості виконання окремих робіт використовують нормативи часу.

Виконавцем усіх операцій по розробці програмного забезпечення є інженер-програміст.

Витрати часу по окремих операціях технологічного процесу відображені в таблиці 5.1.

Таблиця 5.1 – Операції технологічного процесу та їх час виконання

№ п/п	Назва операції (стадії)	Виконавець	Середній час виконання операції, год.
1.	Пошук літературних джерел з області дослідження.	Інженер-програміст	25
2.	Обґрунтування моделі класифікації.	Інженер-програміст	16
3.	Вибір середовища розробки програмного забезпечення.	Інженер-програміст	5
4.	Розробка програми.	Інженер-програміст	52
5.	Тестування розробленої моделі на реальному датасеті.	Інженер-програміст	18
6.	Дослідження впливу на точність розмірності простору ключових слів та деяких етапів нормалізації текстових даних.	Інженер-програміст	10
Разом			126

Загальні затрати часу на реалізацію даної роботи становить 126 години, найбільш трудомістким є власне розробка програмного забезпечення – 52 години.

5.2 Визначення витрат на оплату праці та відрахувань на соціальні заходи

Відповідно до Закону України “Про оплату праці” заробітна плата – це “винагорода, обчислена, як правило, у грошовому виразі, яку власник або уповноважений ним орган виплачує працівникові за виконану ним роботу”.

Розмір заробітної плати залежить від складності та умов виконуваної роботи, професійно-ділових якостей працівника, результатів його. Заробітна плата складається з основної та додаткової оплати праці.

Основна заробітна плата нараховується за виконану роботу за тарифними ставками, відрядними розцінками чи посадовими окладами.

Додаткова заробітна плата – це складова заробітної плати працівників, до якої включають витрати на оплату праці, не пов'язані з виплатами за фактично відпрацьований час. Нараховують додаткову заробітну плату залежно від досягнутих і запланованих показників, кваліфікації виконавців. Джерелом додаткової оплати праці є фонд матеріального стимулювання, який створюється за рахунок прибутку.

При розрахунку заробітної плати кількість робочих днів у місяці слід в середньому приймати – 24,5 дні/міс., або ж 196 год./міс. (тривалість робочого дня – 8 год.).

Місячний оклад кожного працівника слід враховувати згідно існуючих на даний час тарифних окладів. Згідно закону України «Про Державний бюджет України на 2019 рік», зокрема статтею восьмою мінімальна заробітна плата у погодинному розмірі становить 25,13 грн. Згідно з Єдиною тарифною сіткою розрядів та коефіцієнтів з оплати праці працівників та організацій окремих галузей бюджетної сфери установ розроблені рекомендовані тарифні розряди, що приблизно відповідають наступним межам погодинної оплати: керівник дипломної роботи – 30,00...50,00 грн./год., інженер-програміст першої категорії– 25,13...30,00 грн./год., консультант – 25,13...30,00 грн./год.

Основна заробітна плата розраховується за формулою:

$$З_{осн.} = Т_с \cdot К_г \,, \qquad (5.1)$$

де $Т_с$ – тарифна ставка, грн.; $К_г$ – кількість відпрацьованих годин.

Оскільки всі види робіт в виконує розробник-програміст, то основна заробітна плата буде розраховуватись тільки за однією формулою

$$З_{осн.} = 25{,}13 \cdot 126 = 3166{,}38 \text{ грн.}$$

Додаткова заробітна плата становить 10–15 % від суми основної заробітної плати.

$$З_{дод.} = З_{осн.} \cdot К_{допл.}, \quad (5.2)$$

де $К_{допл}$ – коефіцієнт додаткових виплат працівникам, 0,1–0,15 (візьмемо його рівним 0,15).

$$З_{дод} = 3166{,}38 \cdot 0{,}15 = 474{,}65 \text{ грн.}$$

Звідси загальні витрати на оплату праці ($В_{о.п.}$) визначаються за формулою:

$$В_{о.п.} = З_{осн.} + З_{дод.} \quad (5.3)$$

$$В_{о.п.} = 3166{,}38 + 474{,}65 = 3641{,}03 \text{ грн.}$$

Крім того, слід визначити відрахування на соціальні заходи:

- єдиний соціальний внесок ЄСВ (прибутковий податок) – 22%;
- військовий збір – 1,5%.

У сумі зазначені відрахування становлять 23,5 %.

Отже, сума відрахувань на соціальні заходи буде становити:

$$В_{с.з.} = Ф_{оп} \cdot 0{,}235 \quad (5.4)$$

де $Ф_{оп}$ – фонд оплати праці, грн.

$$В_{с.з.} = 3641{,}03 \cdot 0{,}235 = 855{,}64 \text{ грн.}$$

Проведені розрахунки витрат на оплату праці наведено у таблицю 5.2.

Таблиця 5.2 – Розрахунки витрат на оплату праці

з/п	Категорія працівників	Основна заробітна плата, грн.			Додаткова заробітна плата, грн.	Відрахування $Ф_{ОП}$, грн.	Всього витрати на плату праці, грн. (6=3+4+5)
		Тарифна ставка, грн.	Кількість відпрацьованих год.	Фактично нарах. з/пл., грн.			
А	Б	1	2	3	4	5	6
1.	Програміст (розробник)	25,13	126	3166,38	474,65	855,64	4496,67

З таблиці розрахунки витрат на оплату праці видно що всього витрати на плату праці становить 4496,67грн.

5.3 Розрахунок матеріальних витрат

Матеріальні витрати визначаються як добуток кількості витрачених матеріалів та їх ціни:

$$M_{вi} = q_i \cdot p_i \,, \tag{5.5}$$

де: q_i – кількість витраченого матеріалу і–го виду; p_i – ціна матеріалу і–го виду.

Звідси, загальні матеріальні витрати можна визначити:

$$З_{м.в.} = \sum M_{вi} \,. \tag{5.6}$$

Розрахунки занесемо у таблицю 5.3.

Таблиця 5.3 – Розрахунки матеріальних витрат

Найменування матеріальних ресурсів	Один. виміру	Норма витрат	Ціна за один., грн.	Затрати матер., грн.	Транс-портно-заготівель-ні витрати, грн.	Загальна сума витрат на матер., грн.
1. Основні матеріали						
Використання мережі Internet	години	120	–	120	–	120
2. Допоміжні витрати						
Папір формату А4	шт.	160	0,3	48	–	48
Разом:						168

Загальні матеріальні витрати на Internet і Папір формату А4 становить 168 грн.

5.4 Розрахунок витрат на електроенергію

Затрати на електроенергію 1–ці обладнання визначаються за формулою:

$$З_в = W \cdot T \cdot S\,, \qquad (5.7)$$

де W – необхідна потужність, кВт; T – кількість годин на реалізацію розробки; S – вартість кіловат-години електроенергії.

Вартість кіловат-години електроенергії слід приймати згідно існуючих на даний час тарифів. Отже, 1 кВт з ПДВ коштує 1,68грн.

Потужність комп’ютера для створення дипломної роботи – 80 Вт, кількість годин роботи обладнання згідно таблиці 5.1 –126 години.

Тоді,

$$З_в = 0{,}08 \cdot 126 \cdot 1{,}68 = 16{,}93 \text{ грн.}$$

Згідно формули затрати на електроенергію де необхідна потужність множиться на кількість годин на реалізацію розробки і множиться на вартість кіловат-години електроенергії що в висновку дорівнює 16,93 грн.

5.5 Розрахунок суми амортизаційних відрахувань

Характерною особливістю застосування основних фондів у процесі виробництва є їх відновлення. Для відновлення засобів праці у натуральному виразі необхідне їх відшкодування у вартісній формі, яке здійснюється шляхом амортизації.

Амортизація – це процес перенесення вартості основних фондів на вартість новоствореної продукції з метою їхнього повного відновлення.

Для визначення амортизаційних використовується формула:

$$A = \frac{Б_В \cdot H_A}{100\%}, \tag{5.8}$$

де A – амортизаційні відрахування за звітний період, грн.; $Б_В$ – балансова вартість групи основних фондів на початок звітного періоду, грн.; H_A – норма амортизації.

Комп’ютери та оргтехніка належать до четвертої групи основних фондів. Для цієї групи річна норма амортизації дорівнює 60 % (квартальна – 15 %).

Для даної дипломної роботи засобом розробки є комп’ютер. Його вартість становить 24000 грн. Отже, амортизаційні відрахування будуть рівні:

$$A = 24000 \cdot 5\% / 100\% = 1200{,}00 \text{ грн.}$$

Оскільки робота виконувалась 126 години, а в місяць є 196 робочих годин, то амортизаційні відрахування будуть становити:

$$A = 1200{,}00 \cdot 126 / 196 = 771{,}43 \text{ грн.}$$

Згідно формули для визначення амортизаційних де $Б_В$ множиться $Н_А$ і ділиться на 100% амортизація розробки становить 771,43 грн.

5.6 Обчислення накладних витрат

Накладні витрати пов’язані з обслуговуванням виробництва, утриманням апарату управління спілкою та створення необхідних умов праці.

В залежності від організаційно-правової форми діяльності господарюючого суб’єкта, накладні витрати можуть становити 20–60 % від суми основної та додаткової заробітної плати працівників.

$$Н_в = В_{о.п.} \cdot 0{,}2\ldots0{,}6\ , \qquad (5.9)$$

де $Н_в$ – накладні витрати.

Отже, накладні витрати:

$$Н_в = 4496{,}67 \cdot 0{,}2 = 897{,}33 \text{ грн.}$$

Накладні витрати згідно розрахунку формули, становить 897,33 грн.

5.7 Складання кошторису витрат та визначення собівартості науково-дослідницької роботи

Результати проведених вище розрахунків зведемо у таблицю 5.4.

Таблиця 5.4 – Кошторис витрат на НДР

Зміст витрат	Сума, грн.	В % до загальної суми
Витрати на оплату праці $В_{о.п}$	3641,03	57,3%
Відрахування на соціальні заходи $В_{с.з}$	855,64	13,5%
Матеріальні витрати $З_{м.в}$	168,00	2,6%
Витрати на електроенергію $З_в$	16,93	0,3%
Амортизаційні відрахування $А$	771,43	12,1%
Накладні витрати $Н_в$	897,33	14,1%
Собівартість $С_в$	6350,36	100,00

Собівартість ($С_в$) роботи розрахуємо за формулою:

$$С_в = В_{о.п.} + В_{с.з.} + З_{м.в.} + З_в + А + Н_в \, . \quad (5.10)$$

Отже, собівартість роботи дорівнює:

$С_в$ = 3641,03+855,64+168+16,93+771,43+897,33= 6350,36 грн.

Загальний кошторис витрат та визначення собівартості науково-дослідницької роботи становить 6350,36 грн.

5.8 Розрахунок ціни науково-дослідної роботи

Ціну науково-дослідної роботи можна визначити можна визначити за формулою:

$$Ц = С_В \cdot (1 + Р_{рен}) \cdot (1 + ПДВ) \quad (5.11)$$

де $Р_{рен.}$ – рівень рентабельності, 30 %, *ПДВ* – ставка податку на додану вартість, (20 %).

Звідси ціна на роботу складе:

$$Ц = 6350,36 \cdot (1 + 0,3) \cdot (1 + 0,2) = 9906,56 \text{ грн.}$$

Загальний розрахунок ціни програмного продукту становить 9906,56 грн.

5.9 Визначення економічної ефективності і терміну окупності капітальних вкладень

Ефективність виробництва – це узагальнене і повне відображення кінцевих результатів використання робочої сили, засобів та предметів праці на підприємстві за певний проміжок часу.

Економічна ефективність ($Е_р$) полягає у відношенні результату виробництва до затрачених ресурсів:

$$Е_р = \frac{П}{С_В}, \tag{5.12}$$

де *П* – прибуток; $С_В$ – собівартість.

Плановий прибуток ($П_{пл}$) знаходимо за формулою:

$$П_{пл} = Ц - С_в . \tag{5.13}$$

Розраховуємо плановий прибуток:

$$П_{пл} = 9906,56 - 6350,36 = 3556,20 \text{ грн.}$$

Отже, формула для визначення економічної ефективності набуде вигляду:

$$E_p = \frac{П_{пл}}{C_в}. \quad (5.14)$$

Тоді,

$$E_p = 3556{,}2 / 6350{,}36 = 0{,}56.$$

Поряд із економічною ефективністю розраховують термін окупності капітальних вкладень (T_p):

$$T_p = \frac{1}{E_p}, \quad (5.15)$$

Термін окупності дорівнює:

$$T_p = 1 / 0{,}56 = 1{,}8 \text{ р.}$$

Згідно формул плановий прибуток від розробки становить 3556,2 грн., економічна ефективність дорівнює 0,56 а термін окупності становить 1,8 роки що вважається доцільним та економічно вигідним.

5.10 Висновки до розділу 5

У розділі обчислено собівартість проекту, плановий прибуток, термін окупності та економічну ефективність.

6 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

6.1 Охорона праці

При розробці програмного забезпечення для визначення авторства важливо дотримуватись інструкцій з охорони праці при роботі на комп'ютерах [16-18]. Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці. В законі України «Про охорону праці» визначається, що охорона праці - це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності.

На всіх підприємствах, в установах, організаціях повинні створюватися безпечні і нешкідливі умови праці. Забезпечення цих умов покладається на власника або уповноважений ним орган (далі роботодавець). Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про 86 охорону праці. Роботодавець повинен впроваджувати сучасні засоби техніки безпеки, які запобігають виробничому травматизмові, і забезпечувати санітарно-гігієнічні умови, що запобігають виникненню професійних захворювань працівників. Він не має права вимагати від працівника виконання роботи, поєднаної з явною небезпекою для життя, а також в умовах, що не відповідають законодавству про охорону праці. Працівник має право відмовитися від дорученої роботи, якщо створилася виробнича ситуація,

небезпечна для його життя чи здоров'я або людей, які його оточують, і навколишнього середовища.

Перед початком роботи на комп’ютерах потрібно перевірити надійність встановлення апаратури на робочому столі. ВДТ не повинен стояти на краю столу. Повернути ВДТ так, щоб було зручно дивитись на екран – під прямим кутом (а не збоку) і трохи згори вниз, а екран повинен бути нахиленим під невеликим кутом (нижній край ближче до оператора). Перевірити загальний стан апаратури, справність проводки електромережі, з’єднувальних шнурів, штепсельних вилок та розеток. Відрегулювати освітленість робочого місця. Відрегулювати і зафіксувати висоту крісла, зручний для користувача нахил його спинки. Під’єднати до системного блока необхідну апаратуру (принтер, сканер і т.п.). Всі кабелі, що з’єднують системний блок з іншими пристроями слід вставляти і виймати тільки при вимкнутому електричному живленні. Увімкнути апаратуру ПК вмикачами на корпусах в такій послідовності : стабілізатор (або блок безперервного живлення), ВДТ, системний блок, принтер (якщо передбачається друкування). Відрегулювати яскравість світіння екрану ВДТ, фокусировку, контрастність. Не слід робити яскравість екрану занадто великою, тому що це втомлює очі.

Під час виконання роботи необхідно стійко розташувати всі складові пристрої на столі, в тому числі і клавіатуру. Разом з тим повинна бути передбачена можливість переміщення клавіатури. ЇЇ розташування і кут нахилу повинні відповідати побажанням користувача ПК. Якщо в конструкції клавіатури не передбачений простір для опору долонь, то її слід розташовувати на відстані не менше 100 мм від краю столу в оптимальній зоні моніторного поля. При роботі на клавіатурі слід 95 сидіти прямо, не напружуватись. Для зменшення несприятливого впливу на користувача пристроїв типу "миша" (вимушена поза, необхідність постійного контролю за якістю дій) слід забезпечити вільною більшу площу поверхні столу для переміщення "миші" і зручного упору ліктьового суглоба. Періодично при вимкнутому ПК слід видаляти злегка зволоженою мильним розчином хлопко-

паперовою салфеткою пил з поверхонь апаратури. Екран і захисний екран протирають спеціальною серветкою. Не дозволяється використовувати рідинні або аерозольні засоби чистки поверхонь ПК. ЗАБОРОНЯЄТЬСЯ: самостійно ремонтувати апаратуру; класти будь-які речі на апаратуру ПК, бутерброди та напої на клавіатуру або поруч з нею; затуляти вентиляційні отвори в апаратурі, це може призвести до її перегріву і виходу з ладу; для зняття статичної електрики рекомендується час від часу торкатися до металевих поверхонь (наприклад батарей центрального опалення і т.п.); для зниження напруженості праці на ПК необхідно рівномірно чергувати характер робіт в залежності від їх складності; для зменшення негативного впливу на стан здоров'я працівників різних факторів ризику, пов'язаних з роботою на ПК, необхідно кожні дві години відволікатися і робити перерву на 15 хвилин, під час якої доцільно виконати фізичні вправи.

При закінчення роботи на ПК потрібно закінчити і зберегти в пам'яті ПК файли, які знаходились у роботі. Виконати всі дії для коректного завершення роботи в оперативній системі. Вимкнути принтер та інші периферійні пристрої, вимкнути ВДТ і системний блок. Вимкнути стабілізатор (або блок безперервного живлення при його наявності). Штепсельні вилки витягнути з розеток. Накрити клавіатуру кришкою для попередження попадання в неї пилу. Навести порядок на робочому місці. Вимкнути освітлення та електроприлади.

При аварійних ситуаціях, при раптовому припиненні подачі електричної енергії вимкнути всі пристрої ПК в такій послідовності: периферійні пристрої, ВДТ, системний 96 блок, стабілізатор (або блок безперервного живлення). Витягнути вилки з розеток. При наявності ознак горіння (дим, запах горілого) необхідно вимкнути всі пристрої ПК, знайти місце загоряння і виконати всі можливі заходи для його ліквідації, попередивши терміново про це керівництво. У випадку виникнення пожежі негайно попередити про це пожежну частину та керівництво, виконати усі

можливі заходи по евакуації людей з приміщення і розпочати гасіння пожежі первинними засобами пожежогасіння.

6.2 Безпека в надзвичайних ситуаціях

6.2.1 Фактори виробничого середовища і їх вплив на життєдіяльність людини

Деякі фактори праці, умови і види зайнятості (тривалість робочого дня, тижня, ступінь важкості праці, поєднання декількох видів зайнятості) носять постійний характер при впливі на людину і пов'язані з його фізичним і психічним здоров'ям. Вони можуть впливати на здоров'я поряд з іншими соціальними чинниками.

Трудовий процес здійснюється в певних умовах виробничого середовища, що характеризуються сукупністю елементів та факторів матеріально-виробничого середовища, що впливають на працездатність та стан здоров'я людини в процесі роботи. Виробнича середовище й фактори трудового процесу становлять в сукупності умови праці.

На здоров'я людини, її життєдіяльність великий вплив мають небезпечні і шкідливі фактори.

Небезпека - це наслідок такої дії деяких факторів на людину, яке при їх невідповідності фізіологічним характеристикам останнього зумовлює феномен самої небезпеки. Небезпечний фактор - це дія на людину, що в певних

умовах призводить до травми, а в окремих випадках - до раптового погіршення здоров'я або до смерті.

Шкідливий фактор - це дія на людину, яке в певних умовах призводить до захворювань або зниження працездатності.

Небезпечні та шкідливі фактори, що впливають на людину, діляться на три групи: активні, пасивно-активні і пасивні.

До активних належать фактори, що можуть вплинути на людину, впливаючи своєю енергією:

- механічні, що характеризуються кінетичною і потенціальною енергією і механічним впливом на людину; до них відносяться кінетична енергія рухомих елементів, потенційна енергія; шум; вібрація; прискорення; гравітаційне тяжіння; невагомість; статичне напруження; дим, туман, пил в повітрі; аномальний барометричний тиск та ін .;
- термічні, що характеризуються тепловою енергією та аномальною температурою; до них належать температура нагрітих і охолоджених предметів і поверхонь, температура відкритого вогню і пожежі, температура хімічних реакцій і інших джерел; до цієї групи належать також аномальні мікрокліматичні параметри - вологість, температура і рухомість повітря, що призводять до порушення терморегуляції організму;
- електричні: електричний струм, статичний електричний заряд, електричне поле, аномальна іонізація повітря;
- електромагнітні: радіохвилі, видиме світло, ультрафіолетові та інфрачервоні промені, іонізуючі випромінювання, магнітні поля;
- хімічні: їдкі, отруйні речовини, а також порушення природного газового складу повітря, наявність шкідливих домішок у повітрі;
- біологічні: небезпечні властивості мікро- і макроорганизмов, продукти життєдіяльності людей і інших біологічних об'єктів;
- психофізіологічні: стрес, втома та ін.

До пасивно-активної групи належать фактори, що активізуються за рахунок енергії, носіями якої є людина або обладнання: гострі нерухомі

предмети, малий коефіцієнт тертя, нерівність поверхні, по якій переміщується людина і машина, а також нахил і підйом.

До пасивних належать ті фактори, які впливають опосередковано, небезпечні властивості яких пов'язані з корозією матеріалів, накипом, недостатньою міцністю конструкцій, та ін. Формою прояву цих факторів є руйнування, вибухи та інші види аварій.

Істотне значення для продуктивності праці і охорони здоров'я мають спрямованість виробничої діяльності, конкретні виробничі операції, знаряддя праці, форми організації праці та ін. Кожен з цих показників вимагає певних фізичних і психофізіологічних якостей.

Наприклад, для роботи на малих обчислювальних машинах і комп'ютерах необхідна тонка координація пальців рук, витривалість зорових аналізаторів. При колективній роботі необхідні розвинені комунікативні здібності і т.д.

Продуктивність праці, стан здоров'я та рівень працездатності людини значною мірою залежать від впливу факторів зовнішнього виробничого середовища.

Ці фактори окремо і особливо в комплексі можуть надавати несприятливий вплив на організм людини в процесі виробничої діяльності. До них, зокрема, відносяться метеорологічні умови (мікроклімат), шум, вібрація, заколисування, радіаційне випромінювання, освітленість робочого місця, психологічна напруженість, режим праці та ін.

Метеорологічні фактори характеризуються поєднанням температури, відносної вологості і швидкості руху повітря. Систематичні відхилення від нормального (комфортного) метеорологічного режиму у виробничих приміщеннях призводять до хронічних простудних захворювань, захворювань суглобів, теплових ударів, судом, стресових станів. Порушується тепловий

баланс, знижується здатність до розумової та фізичної роботі, коли змінюється температура зовнішнього середовища.

Фізичне тренування і загартування підвищують стійкість організму людини до різко мінливих погодних умов, до зміни мікроклімату, значно скорочують період акліматизації і сприяють більш швидкому відновленню розумової та фізичної працездатності після втоми. Різка зміна барометричного тиску, наприклад, може супроводжуватися порушенням функції вестибулярного апарату і середнього вуха, втратою координації рухів. Негативний вплив на органи слуху і нервову систему надає також високий рівень шуму. Під впливом вібрації може розвиватися так звана вібраційна хвороба, коли знижується гострота зору, тактильна, теплова та больова чутливість, уражаються кровоносні судини, відбуваються небажані зміни в суглобах і т.д.

Фізична підготовленість набуває великого значення при необхідності адаптуватися до вібрації і закачування, які можуть істотно знижувати продуктивність праці і навіть приводити до повної втрати працездатності.

Освітлення робочого місця - один з найважливіших факторів трудової діяльності. Головні проблеми, пов'язані з органами зору, на виробництві стосуються адекватності і зручності освітлення. Достатня (оптимальна) освітленість робочого місця позитивно впливає на органи зору, знижує втому. Незадовільне освітлення викликає передчасне стомлення, очні хвороби, головні болі і може бути причиною травматизму.

6.2.2 Підвищення стійкості роботи промислового підприємства в умовах впливу ЕМІ ядерних вибухів

За природою ЕМІ з деякими припущеннями можна порівняти з електромагнітним полем біля блискавки, що створює перешкоди для радіоприймачів. Виникає ЕМІ в основному в результаті взаємодії гамма-випромінювання, що утворюється під час вибуху, з атомами навколишнього середовища. При взаємодії гамма-квантів з атомами середовища останнім

передається імпульс енергії, невелика частка якої витрачається на іонізацію атомів, а основна – на передачу поступального руху електронам і іонам, що утворився в результаті іонізації.

ЕМІ безпосередньої дії на людину не надає. Приймачі енергії ЕМІ – провідники електричного струму: всі повітряні і підземні лінії зв'язку, лінії управління, сигналізації електропередачі, металеві щогли і опори, повітряні і підземні антенні пристрою, наземні і підземні трубопроводи, металеві дахи та інші конструкції, виготовлені з металу. У момент вибуху в них на долі секунди виникає імпульс електричного струму і з'являється різниця потенціалу відносно землі. Під дією цих напруг може відбуватися: пробій ізоляції кабелів, пошкодження вхідних елементів апаратури, підключеної до антен, повітряним і підземним лініях (пробій трансформаторів зв'язку, вихід з ладу розрядників, запобіжників, псування напівпровідникових приладів, а також вигорання плавких вставок, включених в лінії для захисту апаратури. Високі електричні потенціали відносно землі, що виникають на екранах, жилах кабелів, дротяних лініях зв'язку можуть становити небезпеку для осіб, які обслуговують апаратуру.

Оцінювання стійкості до ЕМІ проводиться в такій послідовності: визначається очікувана ЕМІ-обстановка, що характеризується наявністю ЕМІ-сигналів при ядерному вибуху і параметрами: часом наростання і спаду електромагнітного поля, напруженістю полів; визначаються можливі значення токів і струмів в елементах системи, що наведені від впливу ЕМІ визначається чутливість апаратури і її елементів до ЕМІ, тобто межові значення наведених струмів, коли робота системи ще не порушується; електротехнічна й електронна система розподіляється на окремі ділянки, які аналізуються з виділенням основних, від яких залежить робота; визначається коефіцієнт безпеки кожної ділянки системи, а також межа стійкості системи в цілому. Одержані результати розрахунків аналізуються й, оцінюються а потім слід зробити висновки, в яких потрібно відмітити: найбільш уразливі ділянки, ступінь стійкості системи до впливу ЕМІ, які необхідно провести організаційні

й інженерно-технічні заходи спрямовані на підвищення стійкості уразливих окремих ділянок і системи в цілому.

При розробці інженерно-технічних заходів, спрямованих на підвищення стійкості електротехнічних і електронних систем, мають бути застосовані способи боротьби з наслідками впливу ЕМІ або захист від проникнення імпульсів – не допустити наведені струми до чутливих вузлів і елементів устаткування.

Основна мета захисних пристроїв від ЕМІ – не допустити наведені токи до чутливих вузлів. Найбільш простим способом захисту є укладання обладнання повністю або окремих вузлів у захисні струмопровідні заземлені екрани і установка спеціальних захисних пристроїв на всіх лініях, трубопроводах, отворах і вікнах, які з’єднують внутрішні приміщення з обладнанням і зовнішнім середовищем. Ефективним буде заземлення окремих монтажних контурів (незалежно від заземлення екранів), застосування скручених пар проводів, провідних зв’язків усередині обладнання за деревовидною схемою. Для захисту провідних ліній або антен доцільно послідовно з грозовим розрядником встановлювати смугові фільтри.

6.3 Висновки до розділу 6

У підрозділі "Охорона праці" розглянуто правила охорони праці при роботі на комп’ютерах. У підрозділі "Безпека життєдіяльності" описано фактори виробничого середовища і їх вплив на життєдіяльність людини. Також наведено інформацію про підвищення стійкості роботи промислового підприємства в умовах впливу ЕМІ ядерних вибухів.

7 ЕКОЛОГІЯ

7.1 Зниження енергоємності та енергозбереження

Серед актуальних проблем, що стоять перед сучасними підприємствами різних галузей промисловості України, можна виділити високу енергоємність виробничих процесів та нераціональне використання енергоресурсів. За даними Державного агентства з енергоефективності та енергозбереження Україна щорічно споживає близько 210 млн тон паливно-енергетичних ресурсів і належить до енергодефіцитних країн. На сьогоднішній день держава покриває свої потреби в енергоспоживанні приблизно на 50 % за рахунок власних джерел. Україна імпортує 75 % необхідного обсягу природного газу та 85% сирої нафти і нафтопродуктів. Така структура енергоресурсів економічно неспроможна. Вона породжує залежність економіки України від країн-експортерів нафти і газу та є загрозою її енергетичної та національної безпеки.

Аналіз витрат в сфері виробництва, розподілу і споживання електроенергії показує, що більша частина втрат (до 90 %) припадає на сферу енергоспоживання, тоді як втрати при передачі електроенергії складають лише 9-10 %. Тому основні зусилля з енергозбереження в Україні повинні бути зосереджені саме в сфері споживання електроенергії. Таким чином, ключовим фактором підвищення енергоефективності виробництва є розробка та комплексна реалізація організаційних, технологічних, техніко-економічних та інших механізмів раціонального використання енергетичних ресурсів в рамках єдиної стратегії, спрямованої на енергозбереження [20]

Висока енергоємність ВВП в Україні є наслідком суттєвого технологічного відставання більшості галузей економіки від рівня розвинутих країн, незадовільної галузевої структури національної економіки, негативного впливу „тіньового” сектора, зокрема, імпортно-експортних операцій, що об’єктивно обмежує конкурентоспроможність національного виробництва і

лягає важким тягарем на економіку – особливо за умов її зовнішньої енергетичної залежності. На відміну від промислово розвинутих країн, де енергозбереження є елементом економічної та екологічної доцільності, для України - це питання виживання в ринкових умовах та входження в європейські та світові ринки. Для цього підлягає розв'язанню проблема збалансованого платоспроможного попиту як на внутрішньому так і зовнішньому ринках, а також диверсифікації імпорту паливно-енергетичних ресурсів.

В Україні питанням енергоефективності приділяється важливе значення, яке знаходить своє відображення в чинному законодавстві. Енергетична стратегія України розроблена на період до 2030 року та визначає пріоритетні напрями та обсяги енергозбереження Дослідженню питань енергозбереження в Україні, а також формування механізмів реалізації енергозберігаючих заходів промислових підприємств присвячено праці вітчизняних і зарубіжних учених: О. Амоші, В. Геєця, В. Джеджули, К. Рідле, Р. Тоуд, П. Неміша, І. Михайленка, Т. Афонченкової, В. Бевза, К. Докуніної, Т. Сердюк, Ю. Чистова, Ю.Вовк, І. Іполітової та інших. Для забезпечення системності, узгодженості та контролю заходів, які реалізуються в рамках енергополітики підприємства, механізми управління процесами енергозбереження та підвищення енергоефективності повинні бути чітко формалізовані та враховувати певну сукупність чинників. Однак незважаючи на накопичений досвід (як вітчизняний, так і закордонний) впровадження енергозберігаючих проектів та формування нормативно-правової бази управління процесами енергоефективності промислових підприємств носить ситуаційний характер.

Основним бар'єром для ефективного використання енергії є відсутність робочих ринків енергоресурсів, що пов'язано із надмірним регулюванням. Для України характерні надмірне державне втручання як зі сторони попиту, так і зі сторони пропозиції на енергетичних ринках. З боку попиту, субсидії на енергоносії є досі прийнятними на політичному рівні, як засіб здійснення соціальної політики. В результаті, низькі ціни

на енергоносії створюють недостатню мотивацію для ощадливого споживання енергії або інвестицій в енергозберігаюче обладнання. Більш того, монополістичні структури з боку пропозиції, державна власність та неякісне управління означають, що неконкурентоспроможні компанії з неефективними виробничими технологіями визначають портрет українського енергетичного сектору. Низькі ціни на енергоносії використовуються для перехресного субсидування енергоємних компаній, що призводить до подальшого використання неконкурентоспроможних виробничих технологій. Держава наразі зазнала невдачі у прийнятті законодавчих змін, які б чітко визначали право власності у секторі будівництва та уможливили б інвестиції в енергозбереження. В цілому, ці обмеження створюють недостатню мотивацію для раціонального використання енергоносіїв, гальмують інвестиції у підвищення енергоефективності та створюють значні витрати для українського суспільства. Тому, усунення цих викривлень є передумовою для будь-якої спроби підвищення енергоефективності.

7.2 Індексний метод в екології

Статистична практика при вивченні екологічних явищ широко використовує індекси. Знання методології побудови індексів значно розширює аналітичні можливості дослідника, збагачує результативну інформацію досліджень [21].

За допомогою індексів можна характеризувати зміну в часі і просторі найрізноманітніших показників: обсяги викидів в атмосферу, скидів шкідливих речовин у водне середовище, інтенсивність забруднень і т. д. Їх поділяють на дві групи: до першої належать об'ємні (сумарні) показники (наприклад, обсяг викидів та скидів кількість забруднювачів, площа забрудненої території та ін.), які виражаються абсолютними величинами; до другої — показники, розраховані на певну одиницю (наприклад, викиди в

розрахунку на одиницю земельної площі або на одного жителя, працівника і т.д.). Останні умовно можна назвати якісними показниками, і виражаються вони у вигляді середніх величин.

Обчислення загальних індексів, що дають змогу співвіднести між собою показники за складними сукупностями, являє собою особливий прийом дослідження, який називається індексним методом.

Індексний метод має свою термінологію та символіку. Її дотримання є обов'язковою умовою в індексному аналізі.

Для побудови статистичного індексу необхідно мати вихідну інформацію, як мінімум, за два періоди. Один з таких періодів називається базисним, другий — поточним. Базисний — це період, з яким порівнюють досліджувані явища, поточний — період, що порівнюється. При побудові статистичних індексів насамперед необхідно вирішити такі питання:

- який набір різнорідних елементів досліджуватиметься;
- які показники виступатимуть індексованими величинами;
- які величини виступатимуть сумірниками (вагами).

Серед багатьох видів індексів найбільш простими, елементарними індексами є індивідуальні індекси.

Індивідуальні індекси характеризують зміну в динаміці або відображають співвідношення в просторі якогось одного показника, наприклад, обсягу викидів певного виду шкідливої речовини чи токсиканта.

Зведені індекси - це співвідношення рівнів показника, до складу якого входять різнорідні елементи. Такими елементами є окремі сфери навколишнього середовища, окремі види природних ресурсів, окремі види забруднень середовища тощо. Якщо сукупність, що вивчають, складається з декількох груп, то в цьому випадку можна визначити зведені групові індекси і зведений індекс по всій сукупності, тобто загальний індекс. Так, прикладом загального індексу може бути індекс динаміки забруднення повітря, води, земель, індексу ресурсного потенціалу, індексу еколого-ресурсного

потенціалу тощо. Зведені індекси забруднення визначені для кожної із сфер, називають груповими.

За своєю формою загальні індекси поділяють на агрегатні і середньозважені.

Агрегатний індекс вважається основною формою загального індексу. Його застосовують для вивчення складних суспільних явищ, які містять у собі різнойменні елементи. Особливу групу становлять індекси середніх величин (індекси змінного та фіксованого складу, індекс структурних зрушень).

Екологічні явища і показники, можуть бути порівнянними, якщо вони мають якусь спільну міру, і непорівнянними, якісними і об'ємними. Так показники забруднення атмосфери і показники забруднення гідросфери або літосфери непорівнянні і безпосередньо підсумовувати їх не можна. Непорівнянність зумовлюється тим, що окремі види забруднень мають різні одиниці виміру. В той же час різні види викидів у повітря є порівнянними і загальну кількість їх можна підсумувати. Тому перш ніж будувати той чи інший загальний індекс, слід привести різні види забруднення до порівнянного виду. Це можна здійснити за допомогою таких коефіцієнтів - сумірників. Перемноживши обсяг викидів кожного виду на відповідний сумірник, дістанемо показники, які можна підсумувати, а отже, і порівняти їх у цілому по сукупності.

Одним з важливих положень побудови і застосування загальних індексів є класифікація факторів-співмножників. У кожному конкретному випадку слід визначити суть кожного з них. Серед двох факторів-співмножників виділяють об'ємний (екстенсивний) і якісний (інтенсивний).

Коли при побудові індексу необхідно один з факторів залишати незмінним (фіксованим), то слід дотримуватись правила, яке прийняте в статистичній практиці:

- якісні фактори-співмножники фіксуються на рівні базисного періоду,
- об'ємні - на рівні поточного.

Кожний із незмінних співмножників при побудові індексів відіграє різну роль. Якщо незмінним є об'ємний показник, то він виступає в ролі ваги, а якщо якісний - то в ролі сумірника.

Таке розмежування показників необхідне лише при побудові загальних індексів і саме тоді, коли індекс має характеризувати зміну якогось складного явища за рахунок окремого фактора:

У кожному з названих загальних індексів один із співмножників є величина індексована, а другий - фіксована, що умовно залишається незмінною.

Агрегатним індексом називається загальний індекс, одержаний зіставленням підсумків, які виражають величину складного показника у звітному та базисному періодах, за допомогою сумірників (незмінних).

Такі індекси дають змогу дати порівняльну характеристику рівнів складного явища, до якого входить ряд різнорідних елементів.

Агрегатна форма індексів дозволяє розв'язати ряд конкретних завдань екологічного аналізу. Проте в окремих випадках неможливо вивчити динаміку складного екологічного явища на основі безпосередньо цієї форми індексу. Тому виникає потреба у використанні інших форм зведених індексів - найчастіше арифметичного чи гармонійного. Вибір тієї чи іншої форми індексу залежить від мети, з якою він визначається, і вихідних даних.

В екологічному аналізі нерідко доводиться порівнювати такі якісні показники, як середні викиди забруднюючих речовин в атмосферу, у водойми, грунт. В даному разі йдеться про середні, що обчислені. По-перше, на основі групових середніх, по-друге, по однойменним викидам, фізичний обсяг яких можна підсумувати. Так, в області рівень різних якісних показників, що функціонально пов'язані з фізичним обсягом викидів в окремих районах, буває неоднаковий.

Середній рівень викидів буде залежати як від викидів в окремих підприємствах, так і від частки окремих підприємств у загальній їх кількості

по області. Розрахунок середнього рівня викидів по області в базисному і поточному періодах можна здійснити за такими формулами:

$$x = X\, x_o\, w_o ; \qquad x = X\, x_i\, w_i$$

де x_0 і X_i - середні викиди по окремих районах в базисному і поточному періодах; x_0 і x - середні викиди по області в цілому; w_o і w_i - кількість підприємств де є викиди.

Отже, зміна середнього рівня якісного показника зумовлена впливом тих факторів, від яких залежить сама середня.

Аналіз динаміки середнього рівня здійснюють на основі побудови системи співзалежних індексів. Індекс, що характеризує зміну середнього рівня якісного показника за рахунок зміни всіх факторів в цілому, дорівнює добутку індексів-співмножників, кожний з яких характеризує зміну лише одного фактора і тим самим вплив цієї зміни на динаміку середньої.

7.3 Висновки до розділу 7

В розділі Екологія описано питання зниження енергоємності та енергозбереження на підприємствах та загальну стратегію енергозбереження в Україні загалом та індексний метод в екології.

ВИСНОВКИ

У дипломній роботі розроблено програмне забезпечення для визначення авторства документу.

Загалом було вирішено наступні задачі:

- Проведено огляд літературних джерел в області дослідження.
- Досліджено основні завдання та сфери практичного застосування задачі визначення авторства.
- Обґрунтувано вибір моделі наївного класифікатора Байєса для задачі мульти-класифікації визначення авторства.
- Навчальний та тестовий набір даних взято з репозитарію машинного навчання UCI.
- Розроблене програмне забезпечення для реалізації обраної моделі з використанням мови програмування Python та бібліотек Pandas, Scikit-learn та NLTK/
- Проведено порівняльний аналіз точності класифікаційної моделі для різної розмірності простору ознак. Виявилось, що збільшення розмірності простору ознак не значно впливає на точність, але в значній мірі сповільнює алгоритм.
- Досліджено вплив на якість прогнозної моделі різних методів нормалізації текстових документів. Всупереч загально-прийнятій думці, що стемінг повинен значно покращити результати текстової класифікації, у даному випадку відбулось мізерне зростання точності.

Крім того у роботі опрацьовано розділи "Обгрунтування економічної ефективності", "Охорона праці та безпека в надзвичайних ситуаціях" та "Екологія".

БІБЛІОГРАФІЯ

1. Argamon, S., Koppel, M., Pennebaker, J.W., Schler, J.: Automatically profiling the author of an anonymous text. Communications of the ACM 52(2), 2009. – P.119–123
2. Choi, F.Y.: Advances in Domain Independent Linear Text Segmentation// Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference. 2000.- pp. 26–33.
3. Gungor, Abdulmecit, Benchmarking Authorship Attribution Techniques Using Over A Thousand Books by Fifty Victorian Era Novelists, Purdue Master of Thesis, 2018-04
4. M. Khonji, Y. Iraqi and A. Jones, "An evaluation of authorship attribution using random forests," 2015 International Conference on Information and Communication Technology Research (ICTRC), Abu Dhabi, 2015, pp. 68-71. doi: 10.1109/ICTRC.2015.7156423
5. Luyckx, K., Daelemans, W.: Authorship attribution and verification with many authors and limited data. In: Proceedings of the 22nd International Conference on ComputationalLinguistics-Volume 1, 2008- pp. 513–520.
6. Natural Language Toolkit [Електронний ресурс]. – Режим доступу: https://www.nltk.org/
7. Ramyaa, Congzhou He, Khaled Rasheed. Using Machine Learning Techniques for Stylometry [Електронний ресурс]. – Режим доступу: https://www.cs.nmt.edu/~ramyaa/publications/ml_techniques_Stylometry.pdf
8. Reddy, T. Raghunadha, B. Vishnu Vardhan, and P. Vijaypal Reddy. “A survey on authorship profiling techniques.” International Journal of Applied Engineering Research 11.5 (2016): 3092–3102.
9. Stamatatos, E.: A Survey of Modern Authorship Attribution Methods. Journal of theAmerican Society for Information Science and Technology 60, 2009.- 538–556
10. Stamatatos, E.: Authorship attribution using text distortion. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics:Volume 1, Long Papers. 2017- pp. 1138–1149.
11. Steven Bird Natural Language Processing with Python Analyzing Text with the Natural Language Toolkit / Steven B., Ewan K., Edward L // Sebastopol: O`REILLY. – 2010. – P. 504 – 512.
12. Vysotska V., Kanishcheva O., Hlavcheva Y. Authorship Identification of the Scientific Text in Ukrainian with Using the Lingvometry Methods //2018 IEEE

13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT). – IEEE, 2018. – Т. 2. – С. 34-38.

13. Каніщева О. В. Визначення стилю автора для виявлення плагіату в академічному середовищі / О. В. Каніщева, Ю. Н. Главчева, В. А. Висоцька // Системний аналіз та інформаційні технлогії : матеріали 19-ї Міжнар. наук.-технічн. конф. SAIT 2017, 22-25 травня 2017. — Київ : ННК "ІПСА" НТУУ "КПІ" ім. Ігоря Сікорського, 2017. — С. 78-79.
14. Луис Педро Коэльо, Вилли Ричарт. Построение систем машинного обучения на языке Python /М., ДМК Пресс. – 2016. – 302 с.
15. Марченко О.О. Система визначення авторства тексту / О.О. Марченко, А.О. Никоненко, Т.В. Россада, Є.А. Мельников // Штучний інтелект. — 2016. — № 2. — С. 77-85.
16. Грибан В. Г., Негодченко О. В. Охорона праці : навчальний посібник . -2-е видання. Київ: Центр учбової літератури, 2018.- 280 с, ISBN 978-966-364-832-3
17. Запорожець О. І., Протоєрейський О. С., Франчук Г. М., Боровик І. М. Основи охорони праці підручник Київ: Центр учбової літератури, 2017. - с.264, ISBN 978-617-673-423-9
18. М. С. Одарченко, А. М. Одарченко, В. І. Степанов, Я. М. Черненко. Основи охорони праці: підручник/ – Х. : Стиль-Издат, 2017. – 334 с. ISBN 966-7885-84-4
19. Величко С.П., Царенко І.Л., Царенко О.М. Методика викладання безпеки життєдіяльності : навчальний посібник, Київ: КНТ, 2008. - 318 с, ISBN 978-966-373-425-5
20. Енергетична стратегія України на період до 2035 року “Безпека, енергоефективність, конкурентоспроможність”, затверджена розпорядженням Кабінету міністрів України від 18 серпня 2017 р. № 605-р.
21. Тарасова В.В. Екологічна статистика // Київ: «Центр учбово\ї літератури», 2008 ро.-391с.
22. Бедрій Я. І.; Джигирей В. С.; Кидисюк, А. І. та ін. Основи екології та охорона навколишнього природного середовища: навч. посіб. для студ. вищих навч. закладів // за ред. В. С. Джигирей ; Український держ. лісотехнічний ун-т, Львівський електротехнікум зв'язку. - Л. : [б.в.], 1999. - 239 с. Альтернативна назва : Екологія та охорона природи. - ISBN 5-7763-2641-9.
23. В. М. Єнколо. Основи екології та соціоекології навч.посібник. / Львівський електротехнікум зв'язку; за ред. В. М. Єнколо. Львів: Афіша, 1998. - 210с. ISBN 966-95023-5-7.

ДОДАТКИ