МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

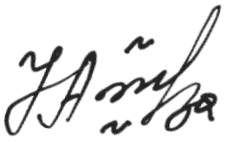
«До захисту допущено»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)
червня 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

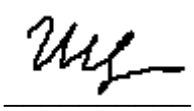
на здобуття освітнього ступеня бакалавр

зі спеціальності 122 – Комп'ютерні науки,
освітньо- професійної програми «Інформатика»
на тему: «ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ РОЗПІЗНАВАННЯ ЗМІСТОВНИХ КОНСТРУКЦІЙ В УКРАЇНСЬКОМУ ТЕКСТІ»
здобувача групи ІН-04р Джумагелдієва Айгул

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.



Айгул ДЖУМАГЕЛДІЄВА
(підпис)

Керівник
доцент,
кандидат технічних наук Сергій ШАПОВАЛОВ 
(підпис)

Суми – 2024

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавра

зі спеціальності 122 - Комп'ютерні науки, освітньо-професійної програми «Інформатика»
здобувача групи ІН-04р Джумагелдієва Айгул

1. Тема роботи: «ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ РОЗПІЗНАВАННЯ ЗМІСТОВНИХ КОНСТРУКЦІЙ В УКРАЇНСЬКОМУ ТЕКСТІ»
затверджую наказом по СумДУ від *«22» квітня 2024 р. №0414-VI*
2. Термін здачі здобувачем кваліфікаційної роботи *до 09 червня 2024 року*
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Аналіз можливостей сучасних фреймворків NLP. 2)Аналіз актуальних моделей інтелектуального аналізу природньої мови. 3) Розробка інтелектуальної компоненти розпізнавання українського тексту. 4) Аналіз та виклад результатів
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проєкту (роботи), із значенням розділів проєкту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ (підпис) Керівник _____ (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз можливостей сучасних фреймворків NLP*		
2	*Аналіз актуальних моделей інтелектуального аналізу природньої мови*		
3	*Розробка інтелектуальної компоненти розпізнавання українського тексту*		
4	*Аналіз та виклад результатів*		
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*		

Здобувач вищої освіти _____ (підпис) Керівник _____ (підпис)

АНОТАЦІЯ

Записка: 37 стр., 10 рис., 1 додаток, 22 використаних джерел.

Обґрунтування актуальності теми роботи – Розроблена інтелектуальна компонента є актуальною, адже дозволяє оцінювати резюме кандидатів зі сторони п’яти психо-емоційних складових характеру людини: результативність в роботі, мотивація, потреби, особливості взаємодії з оточуючими та емоції. Річ у тім, що при розгляданні резюме кандидата роботодавець враховує не лише компетенцію в прикладних навичках, якими рецензент і так повинен володіти, якщо претендую на ту чи іншу позицію, а й його психологічний портрет. Це важливо, щоб оцінити наскільки гарно людина зможе інтегруватися в соціальну екосистему конкретної команди.

Об’єкт дослідження – процес автоматизації інформаційної підтримки.

Мета роботи – розробка інтелектуальної NLP компоненти для оцінювання резюме кандидатів за 5 основними психо-емоційними характеристиками.

Методи дослідження – моделі та методи інтелектуального аналізу природньої мови.

Результати – розроблено класифікатор, який здатний маркувати український текст згідно детермінованого алфавіту класів розпізнавання.

NLP, ІНТЕЛЕКТУАЛЬНА СИСТЕМА, ІНФОРМАЦІЙНА ПІДТРИМКА, МАШИННЕ НАВЧАННЯ, СЕМАНТИЧНИЙ АНАЛІЗ

ЗМІСТ

ВСТУП

Актуальність теми дослідження. Однією з унікальних особливостей людини є її схильність надавати речам абстрактні імена. Фактично даючи назви предметам ми закарбовуємо їх властивості, форму, розміри в своїй пам’яті через їх метафізичний ключ-ім’я. Більшість нейробіологів вважають, що людство змогло досягти поточного рівня свого розвитку лише за допомогою винайдення мови, адже це дозволило передавати інформацію й, як наслідок, навчатися.

Оскільки слова – це не фізичний об’єкт, а більше метаконструкція, яка дозволяє описати світ, то є деякі проблеми з її інтерпретацією за допомогою математичних моделей. Ба більше не зважаючи на уніфікованість та детермінованість значень слів, люди все рівно сприймають їх по різному, а це накладає додатковий шар проблем для реалізації комп’ютеризованої системи розпізнавання речень. Кожен окремий індивід, враховуючи власний досвід, надає додатковий зміст своїм словам. Тим не менш системи обробки природньої мови є достатньо поширеним відгалуженням сучасних досліджень в сфері машинного навчання [1-15].

Хоч на поточний момент часу вдалося значно просунутися в цьому напрямку, проте ця задача все ще залишається не до кінця вирішеною. По-перше, навіть людям не до кінця відомо як наш мозок навчився інтерпретувати мета-конструкції, такі як «слово». Алгоритми, що намагаються відтворити цей механізм, не здатні до нечіткої абстракції. По-друге, сучасні розробки, наприклад, chatGPT більше нагадують експертні системи. Тобто алгоритм, який не здатний вийти за межі власної навчальної вибірки. На відмінну від людини, яка навіть при мінімальній кількості знань може відтворити всю відому людству інформацію. Окрім цього системи обробки природної мови, у більшості випадках, є вузько спеціалізованими. Тобто навченими і розробленими для конкретних задач.

Тим не менш, сучасні інформаційні технології, у деякому сенсі вимагають використання алгоритмів NLP, адже кількість доступної інформації збільшується

експоненційно з кожним роком й без відповідного інструментарію працювати з нею вкрай важко. Ця проблема доповнюється тим фактом, що текстова інформація, як правило, переповнена словесними конструкціями, які не впливають на зміст, але збільшують об’єм роботи.

Як би суспільство до цього не відносилось, але впровадження інтелектуальних алгоритмів, у тому числі обробки тексту, є невід’ємною складовою сучасного світу.

Об’єкт дослідження – процес автоматизації інформаційної підтримки.

Предмет дослідження – інтелектуальна NLP компонента для оцінювання резюме кандидатів за 5 основними психо-емоційними характеристиками.

Гіпотеза дослідження полягає в тому, що розроблений класифікатор буде здатний маркувати український текст згідно детермінованого алфавіту класів розпізнавання.

Наукова новизна. Досліджено проблему розпізнання змістових україномовних конструкцій. Під час дослідження було створено класифікатор, який здатний маркувати україномовні конструкції.

Структура. Кваліфікаційна бакалаврська робота складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатку, викладених на 37 сторінках.

1 ІНФОРМАЦІЙНИЙ ОГЛЯД

1.1 Сучасний стан застосування інтелектуальних систем розпізнавання природної мови.

Оскільки математичні моделі не можуть на пряму працювати з абстрактними об’єктами, то перед безпосередньою роботою з реченнями, необхідно їх векторизувати [1]. Зазвичай вхідний текст проходить декілька рівнів підготовки. Перший з них - це очистка тексту від всіх не алфавітних символів. Звісно, що у ряді задач необхідно знати структуру речення. Яке з них підрядне, яке головне. Або залишити цифри, які також можуть впливати на інтерпретацію, але, у більшості випадках цього не потрібно робити. Замість цього ми зосереджуємося на самому змісті. Наступним етапом позбуваються, так званих, стоп-слів [2]. Як правило це прислівники, які збагачують та структурують речення, але не впливають на його зміст. Для деяких задач можна позбутися ще й прикметників.

Також невід’ємною частиною перетворення тексту в математичний об’єкт є операція токенізації. Токеном, у рамках NLP, називають окремою одиницею речення. Найчастіше – це слово.

Останнім, і за сумісництвом найважчим етапом підготовки вхідних даних, є їх нормалізація. Річ у тім, що для кожної мови характерні різні форми одного й того самого слова. А для коректної роботи майбутньої математичної моделі необхідно, щоб усі її елементи відповідали деяким уніфікованим правилам. У рамках NLP це досягається за рахунок стеммінгу та лемматизації [3]. Перший метод передбачає знаходження основи слова за деяким алгоритмом, який використовується для кожного токену. Тобто усі елементи речення будуть нормалізовані за одним і тим самим набором правил. Тим не менш, основа слова за стеммінгом не завжди є його корнем, в морфологічному змісті, але для математичної моделі це і не занадто важливо. У свою чергу лемматизація це приведення слова до його початкової морфологічної форми. Наприклад: сердитий -> сердитися; навчаюсь -> навчання; велосипеди -> велосипед. Це

набагато більш трудомісткий механізм, який можна виконати лише за допомогою окремо розробленої інтелектуальної компоненти. При цьому варто розуміти, що вона повинна бути навчена лематизувати речення конкретної мови. Хоч лемматизатор і мінімізує неоднозначність в тексті, але слід враховувати, що він потребуватиме для роботи значних витрат ресурсів й не найкращим чином підходить для систем, які працюють в реальному часі.

Одним із найпростіших способів реалізації стеммеру є алгоритм усічення закінчень. По суті, це набір правил для конкретної мови, який дозволяє, враховуючи форму слова, відсікти його частини, щоб залишилася лише основа. Тим не менш, для цього потрібен експерт, який добре орієнтується в лінгвістиці, адже у кожної мови є ряд обмежень та винятків, які можуть вплинути на результат стеммінгу. Так, наприклад, далеко не у всіх мовах є суфікси.

Останнім етапом роботи з текстом є його безпосередня векторизація. Найпопулярніший метод – це «мішок слів» [4, 5]. Документи описуються входженнями слів, при цьому повністю ігнорується інформація щодо відносного їх стану у тексті. За допомогою цього алгоритму знаходять кількість появ кожного слова у всьому тексті. Тобто це матриця розмірність якої дорівнює кількості усіх унікальних слів в навчальній вибірці помноженної на кількість документів. У свою чергу елементами матриці є числа, які представляють з себе лічильник входження слова в текст.

	adult	anxiety	asleep	avoid	care	caused	coming	couldn	day	delaying	...	stuffed	thing	think	unemployment	utterly	ve	wa	wasn	work	working
0	0	1	1	1	0	0	1	0	1	0	...	0	1	0	0	0	0	0	0	0	0
1	1	0	0	0	1	1	0	1	0	1	...	1	1	1	1	1	2	2	1	1	1

Рисунок 1.1 – Приклад векторизації документу

Також популярним методом для векторизації тексту є алгоритм TF-IDF [6], який є статистичним заходом оцінки важливості слова у документі. У тексті великого обсягу деякі слова можуть бути дуже часто зустрічатися, але при цьому не нести ніякої значущої інформації про фактичний зміст тексту (документу). Якщо такі дані передавати безпосередньо класифікатору, такі часті терміни

можуть затінювати частоти більш рідкісних, але при цьому більш цікавих термінів. Для того, щоб цього уникнути, достатньо розділити кількість вживань кожного слова в документі на загальну кількість слів у документі, тобто TF – частота терміна. Термін IDF (inverse document frequency) означає зворотну частоту терміну (інверсія частоти) з якою деяке слово зустрічається у документах. IDF дозволяє виміряти безпосередню важливість терміна.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$, де

$tf_{i,j}$ – це частота елементу i в тексті j.

df_i – це кількість документів, що містять i

N – це загальна кількість текстів

Векторизований текст є повноцінним математичним об’єктом, що дозволяє його використовувати майже з будь-яким алгоритмом машинного навчання. Наприклад: лінійна регресія, дерева рішень, метод опорних векторів, класифікатор Байеса, нейромережі і т.д. Тим не менш, у рамках NLP, існує ряд специфічних задач, які є набагато більш комплекснішими ніж проста класифікація об’єктів.

Аналіз настроїв [8] — один із найпопулярніших методів NLP, який включає вибір фрагмента тексту (наприклад, коментаря, огляду або документа) і визначення того, чи є дані позитивними, негативними або нейтральними. Він має безліч застосувань у охороні здоров'я, обслуговуванні клієнтів, банківській справі тощо.

Розпізнавання іменованих об'єктів (NER) - це метод, що використовується для виявлення та класифікації іменованих об'єктів у тексті за категоріями, такими як особи, організації, місцезнаходження, вирази часу, кількості, грошові значення, відсотки і т. д. Він використовується для оптимізації алгоритмів пошукових систем, систем рекомендацій, клієнтів. підтримка, класифікація контенту тощо.

Також одним із специфічних різновидів NLP є, так зване, моделювання тем. Це група методів неконтрольованої класифікації тексту, які подібні до класичної кластеризації, тобто до алгоритмів навчання без учителя. Цей підхід може бути корисним при організації великих масивів даних та уряді специфічних задач NLP:

- виявлення прихованих класів у датасеті;
- організація документів по темам;
- автоматична анотація тексту;

Одним із алгоритмів моделювання тем – є лінійний дискримінантний аналіз (LDA) [9], який використовує ймовірносне тематичне моделювання для знаходження такого алфавіту класів, який як найкраще опише наявний датасет. Умовно цей процес можна описати наступним чином:

- деяке слово *w* випадковим чином присвоюється одній з *K* тем;
- розрахунок апостеріорної вірогідності кожного класу, враховуючи розподіл всіх інших слів по темам;
- випадковим чином перевизначити слово w новій темі з вірогідністю, пропорційної апостеріорної вірогідності кожної теми;
- процес продовжити поки не виконається збіжність слів темам;

Оскільки будь-яка інтелектуальна компонента потребує числове представлення даних, то варто розібрати і декілька підходів до векторизації текстів. Наприклад, модель Bert [10] дозволяє не лише конвертувати семантичні конструкції в числове представлення, а й не втрачати їх значення. Річ у тім, що будь-яка людина не свідомо створює асоціативно-синонімічні ряди для кожного слова, які його характеризують. Щоб відтворити цю властивість людської свідомості не достатньо просто перетворити слова в набір цифр, необхідно встановити між ними взаємозв’язок. Цим і займається Bert.

На практиці кожному слову ставиться у відповідність не одне число, а цілий вектор. Тоді щоб зрозуміти наскільки одна семантична конструкція схожа на іншу ці вектори розміщуються у відповідному n-мірному просторі, де і рахується відстань між ними. Основна ідея моделі Bert полягає у створенні попередньої інтелектуальної компоненти, яка навчається на частково замаскованих даних. Таким чином на вхід подаються речення, у яких відсутньо 15% інформації, а система повинна вгадати і заповнити ці пропуски правильними, в змістовному сенсі, відповідниками.

Wordcloud є популярним методом, який допомагає нам ідентифікувати ключові слова в тексті. У хмарі слів слова, що часто зустрічаються, мають більший і жирний шрифт, тоді як менш часто зустрічаються слова мають менший або більш тонкий шрифт. Це простий візуальний метод аналізу тексту.

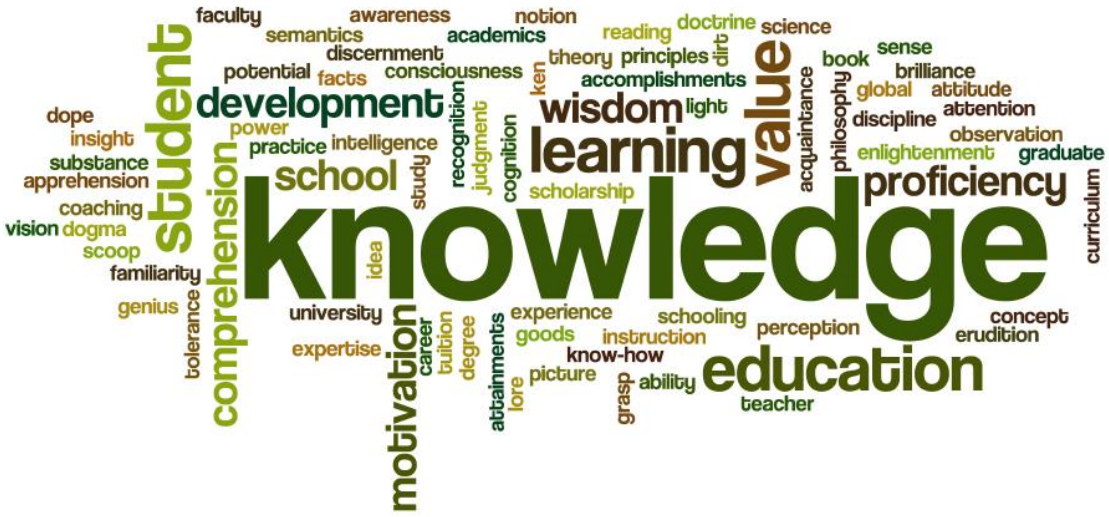


Рисунок 1.2 – Приклад Wordcloud

Останнім часом набуло більшого поширення алгоритми вилучення ключових слів, які можуть бути зручними при початковому аналізі тексту. Усе це робиться для узагальнення та допомозі при роботі з великими об’ємами даних, зберіганні, пошуку та відновленні змісту. Реалізації алгоритмів цієї групи достатньо різноманітні. Деякі вилучають лише слова з тексту, деякі цілі речення чи словосполучення. У деяких випадках аналізується весь текст, а іноді лише окремі розділи. Наступні алгоритми є найбільш популярними:

- ***TextRank***. Алгоритм обчислює ваги між реченнями, дивлячись, які слова збігаються.. Google використовує цей метод для оцінки важливості різних веб-сайтів в Інтернеті;
- **Term Frequency**. Зворотна частота документів (TF-IDF): намагається краще визначити важливість терміну в документі. Крім того, враховуються зв'язки між текстами з одного корпусу;
- **RAKE**. Здатний витягти ключові слова та ключові фрази з вмісту одного документа, не беручи до уваги інші документи в тій же колекції;

Knowledge graphs (рис. 1.3) є ще одним достатньо потужним методом глибокого аналізу великих об'ємів тексту. Як правило це впорядкований набір елементів: суб'єкті, предикатів та сутностей, які пояснюють спосіб формування речень. У деякому сенсі – це один із способів впорядкування інформації з неструктурованих документів у, так звані, графи обізнаності. Це можу бути корисним для аналізу товарів чи послуг, але потребує потужних попередньо навчених моделей.

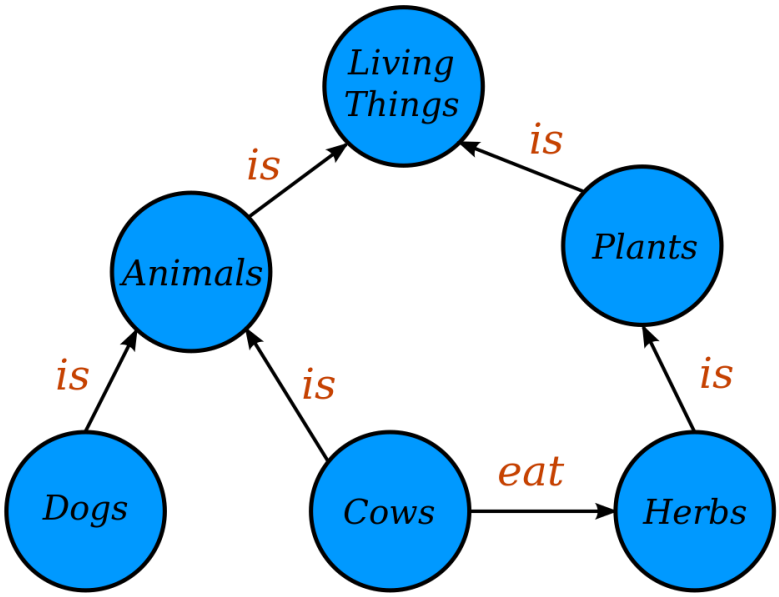


Рисунок 1.3 – Приклад Knowledge graphs

Усе частіше можна зустріти алгоритми автономного конспектування або резюмування тексту (рис.1.4). По своїй суті, ці алгоритми займаються

стисканням текстової інформації до адекватних розмірів, відкидаючи надлишковість.

Резюмування тексту може здійснюватися двома способами: вилученням і реферуванням:

- видалення бітів з тексту, вилученням коротких списків найбільш валідних компонентів інформації;
- абстрогування, де через виявлення найбільш характерних частин тексту створюється новий, який передає суть оригінального змісту.

Для підсумовування тексту можна використовувати різні алгоритми NLP, такі як LexRank, TextRank і Latent Semantic Analysis. Наприклад, при використанні LexRank алгоритм ранжує речення на основі їх подібності. Коли речення максимально схожу на інші в межах документу, то воно отримує вищу оцінку.

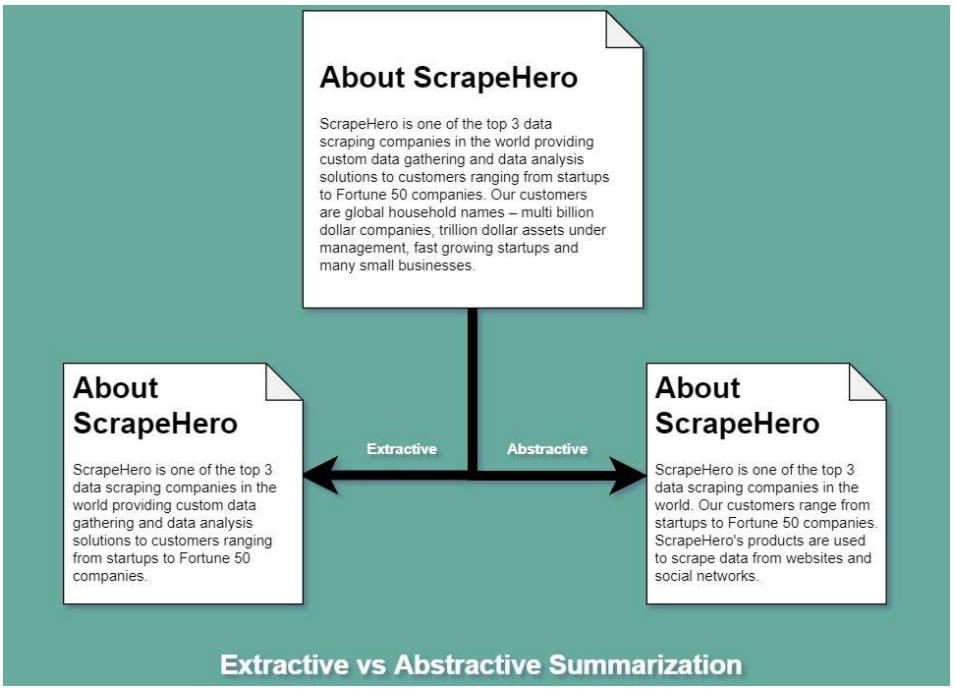


Рисунок 1.4 – Приклад Text summarization

Отже існує безліч способів використання NLP на практиці. Більше того, сучасні інформаційні технології, у деякому сенсі, вимагають від користувача

застосовувати методи NLP, адже вони дозволяють зменшити обсяг великих об’ємів текстової інформації.

1.2 Огляд існуючих фреймворків для роботи з NLP

Особливістю готових рішень для NLP є їх націленість на конкретну мову. На поточний час не існує повністю універсального NLP фреймворку, який би покривав лінгвістичні особливості усіх відомих мов. Тим не менш, для індоєвропейського сегменту ринку існує достатня кількість готових рішень, бібліотек та датасетів.

NLTK [12] - це найбільш класична і часто вживана бібліотека для обробки природньої мови на Python. Вона проста, має розгорнуту документацію і за допомогою неї можна реалізувати більшість задач NLP. У ряді випадках, для її коректної роботи потрібно створювати власні tagger та налаштовувати їх.

Stanford CoreNLP [13] – бібліотека зі схожим функціоналом на NLTK, але розроблена під Java додатки. При цьому у деяких реалізаціях використовуються обидві ці бібліотеки, щоб досягти максимальної ефективності розроблювальної інтелектуальної компоненти.

SpaCy [14] – це бібліотека, а скоріше набір готових моделей для вирішення задач NLP. На відмінну від NLTK, де розробнику надається інструментарій-конструктор для створення власних інтелектуальних компонент, SpaCy надає готові рішення. Широкий вибір модулей для візуалізації та багато реалізованих «з коробки» функцій зробило SpaCy достатньо популярним для вирішення бізнес задач.

Flair [15]– NLP бібліотека, яка більше підходить для дослідницьких задач, ніж для розробки програмного забезпечення прикладного рівня. По-перше, моделі flair потребують налаштування зі сторони розробника, адже вони не здатні одразу «з коробки» давати таку точність, як, наприклад, SpaCy. По-друге, на практиці цей фреймворк використовується для доповнення вже існуючих програмних NLP рішень.

Stanza [16] – у деякому сенсі більш продвинута версія NLTK, яка здатна виконувати задачі NLP з більшою точністю, адже бібліотека побудована на основі високоточних, попередньо навчених інтелектуальних компонент.

fastText [17] – окрема бібліотека, яка використовується для векторизації датасету. Хоч більшість наведених фреймворків мають власні інструменти для конвертації лексики в математичні об'єкти, проте fastText вважається більш досконалим рішенням для створення взаємозв'язаних векторів.

Більшість фреймворків NLP надають готові і попередньо навчені моделі, яких буде достатньо для об'єктивної більшості бізнес-задач. Проте, варто зауважити, що все ще існує мультилінгвістична проблема, адже навіть якщо інтелектуальна компонента добре справляється з однією мовою, то не факт, що можна буде досягти такого ж результату з іншою. Адже кожен лінгвістичний апарат використовує свої власні семантичні конструкції та унікальні для певної культури синонімічно-асоціативні ряди.

1.3 Постановка задачі

На базі аналітичного огляду та джерел інформації поставимо основне завдання досліджень – створити додаток, який розізнає змістовні конструкції української мови і дозволить спростити оцінки резюме. Замість детальної вичитки тексту система одразу розіб'є речення на попередньо визначенні класи розпізнавання, які представляють цінність для бізнесу. Це пришвидшить роботу будь-якого HR-відділу.

Перше з чого варто почати моделювання лінгвістичної системи – це визначення класів розпізнавання. У нашому випадку вони повинні демонструвати деякі характеристики кандидата, які можуть бути корисними для бізнесу:

- ***Результативність в роботі***. Характеристика, яка відповідає за психо-емоційні особливості, які можуть бути в нагоді при роботі. Наприклад: перфікціонізм, вмотивованість, концентрація і т.д.

- ***Мотивація***. Окремо потрібно виділити мотивацію рецензента, оскільки від неї може залежати його відношення до роботи. Наприклад, людина, яка прагне керувати буде схильна до перевлаштування команди з середини, а це може призвести до не передбачуваних проблем.
- ***Потреби***. В залежності від типу потреб можна сказати наскільки зацікавленим буде претендент в роботі, а також зробити припущення стосовно того наскільки ймовірним буде передчасне звільнення.
- ***Особливості взаємодії з людьми***. Оскільки будь-якому співробітнику доведеться працювати в команді, то необхідно визначити такі його характеристики, які стануть у нагоді при роботі з колективом, або навпаки передчасно виявити ті, що можуть цей колектив зруйнувати.
- ***Емоції***. Емоційні характеристики, які дозволять оцінити загальний характер людини і чи підходить такий архетип конкретній компанії або керівнику.

Поки що ми зупинимося на цих п’яти класах. Звісно, що в подальшому алфавіт можна розширити, але це буде залежати від специфіки компанії, де буде працювати ця система. На даному етапі достатньо цих п’яти базових класів-розпізнавання.

За своєю постановкою задача відноситься до слабоформалізованих, тому її розв’язок детермінованими методами, наприклад частотним аналізом, є не доцільним. Насамперед, це пов’язано з тим, що кожна людина має власний унікальний стиль письма, який може включати в себе довільну інтерпретацію деяких слів та словосполучень. Крім цього слова за своєю природою є нечіткою множиною. Одна людина будь-який свій понурий настрій опише словом «депресія», інша ж буде використовувати цей термін лише для опису психологічного розладу. Для когось +15 – це «холодно», інший же використає слово «тепло». Саме тому для алгоритмічного розв’язку задачі не достатньо лише знати кількість входжень того чи іншого слова в реченні, необхідно щоб система визначала контекст.

Оскільки у кожної людини є індивідуальний стиль формування речень, то для початкового навчання інтелектуальної системи не найкращим рішенням буде формувати навчальну базу даних з результатів опитування контрольної групи. Замість цього доцільніше використання рафінованованого, формалізованого тексту. У нашому випадку, який сформований експертом на основі методологічних посібників з відповідної предметної області. Такі як:

- ***Акцент.*** визначає, які риси особистості посилені, які ослаблені, які знаходяться на середньому значенні. Зі поєднання слабких і сильних рис і складається, по суті, наша унікальність і несхожість на інших.
- ***16PF (КЕТЕЛ).*** забезпечує вимірювання особистості і може також використовуватися психологами та іншими фахівцями в галузі психічного здоров'я як клінічний інструмент для діагностики психічних розладів, а також для допомоги в прогнозуванні та терапії. 16PF також може надати інформацію, що відноситься до клінічного процесу та процесу консультування, таку як здатність людини до розуміння, самооцінка, когнітивний стиль, інтерналізація стандартів, відкритість до змін, здатність до емпатії, рівень міжособистісної довіри, якість уподобань, міжособистісні потреби, до влади, реакція на динаміку влади, толерантність до фрустрації та стиль подолання. Таким чином, прилад 16PF дозволяє клініцистам вимірювати тривогу, пристосування, емоційну стабільність та поведінкові проблеми в межах норми. Клініцисти можуть використовувати результати 16PF для визначення ефективних стратегій створення робочого альянсу, розробки терапевтичного плану та вибору ефективних терапевтичних втручань або способів лікування. Його також можна використовувати в інших галузях психології, таких як кар'єра та професійний відбір.
- ***СОНДІ-ТЕСТ.*** Невербальний, проективний, особистісний тест, метою якого є виявлення психічних відхилень. Базується на положенні, що типологічно різні особистісні структури можуть бути представлені

поєднаннями 8 основних потягів. Кожен з яких виявляє (за допомогою тесту Сонді) ту чи іншу патологію чи проблему особистості. В обґрунтування свого тесту, Сонді висловлює припущення, що найбільш виражену силу та психодіагностичне значення мають портрети, які відповідають найбільш значним потребам індивіда та відповідають його генетично обумовленим нахилам.

Кожна із наявних методик має в собі значну кількість рафінованої документації з поясненням кожного питання і результату. Ми використаємо її як навчальний математичний опис, а вже під час реального функціонування системи обов’язково необхідно розширювати базу даних текстом написаним рецензентами. Це пов’язано з тим, що синтаксис та форма речень більшості людей відрізняється від формалізованого, методичного тексту.

2 ПІДГОТОВКА ВХІДНОГО МАТЕМАТИЧНОГО ОПИСУ

2.1 Формування датасету

Для початку сформуємо датасет на якому будемо навчати майбутню математичну модель. У нашому випадку вхідними даними будуть речення, які описують характеристики кандидата згідно трьох психологічних методологій, що в сумі дають нам 1432 унікальних речення, які вже були розмічені експертом. Кожна методологія характеризує різний рівень свідомості рецензента. Нижче наведений приклад речень з якими ми будемо працювати:

```
Виражене почуття внутрішнього дискомфорту (незручності,
незатишності)

Відчуття втоми (тяжкості) від близьких контактів

Почуття безвиході (приреченість, безвихідь) і безнадійності
(розпачу) зусиль, що робляться

Потреба зберігати (підтримувати) в оточуючих позитивну (позитивну)
думку про себе, навіть якщо це обмежує (обмежує) власні інтереси

Виражена потреба у завоюванні (захоплення, взяття) свого місця у
системі соціальних (суспільних) відносин

Виражена потреба бути у контакті з об'єктом (партнером)

…
```

Особливістю датасету є те, що у нас є як оригінальні речення на англійській мові так й їх офіційний переклад. Це важливо, оскільки достатньо невелика кількість бібліотек й фреймворків NLP працюють з українською мовою, тому при роботі ми можемо перевірити точність розпізнавання семантичних структур одразу для обох умов.

Також варто звернути увагу, що, по-перше, датасет сформований з декількох джерел інформації, які у нашому випадку представлені як психологічні методології. Це означає, що речення за своїм вокабуляром та стилем викладання будуть різні. По-друге, присутня деяка рафінованість інформації в датасеті. Речення, здебільшого, є однозначними. Тобто мають лише одне трактування. Ця особливість обумовлена тим, що вони написані в академічному стилі.

Технічні особливості:

- Оскільки перше навчання система виконує на рафінованих даних, які були зібрані з методичних посібників, то під час реального функціонування системи обов’язково необхідно розширювати базу даних текстом написаним рецензентами. Це пов’язано з тим, що синтаксис та форма речень більшості людей відрізняється від формалізованого, методичного тексту.
- Довільний текст може містити в собі орфографічні помилки. Тим не менш, на етапі нормалізації за допомогою лематизації можна виправити їх

2.2 Формування списку стоп-слів

Оскільки основною бібліотекою для реалізації задачі буде NLTK, то слід звернути увагу, що в ній за замовчуванням немає набору стоп-слів для української мови. Тобто таких семантичних конструкцій, які не несуть змістовне навантаження і можуть бути без проблем видаленні з тексту при його нормалізації. Провівши лінгвістичний аналіз був створений список з наступних стоп-слів для української мови: *безумовно, видно, очевидно, здається, щоправда, на диво, на жаль, на щастя, як навмисне, безперечно, без сумніву, мабуть, либонь, сподіваюсь, правда, певна річ, може, правду кажучи, щоправда, на диво, на сором, як на зло, чого доброго, дивна річ, по-моєму, кажуть, на мою думку, по-вашому, мовляв, як відомо, за словами, за думкою, бачу, по-твоєму, по-друге, нарешті, до речі, між іншим, крім того, навпаки, отже, наприклад, інакше, таким чином, значить, коротше кажучи, однак, чуєте, бачиш, уяви, зверніть увагу, послухайте, майте на увазі, зрозумійте, даруйте, скажімо, однак, певно, звичайно, їй-богу, як відомо, зауважте, звісно, ймовірно, смію запевнити, гадаю, соромно казати, як на сміх, як на гріх, прошу вас, між нами кажучи, хвалити долю, погодьтеся, пробачте, з одного боку, крім того, одним словом, страшно сказати, повірте, як говорять, вибачте, будь-ласка, прошу вас, дозвольте, даруйте, на здоров’я, нібито, немовби, все-таки, адже, от, тільки, принаймні,*

навіть, між тим, за традицією, буквально, якби, майже, при тому, при цьому, часом, тим часом, до того ж, приблизно, якраз, як-не-як.

Звісно, що виділені слова не всі є характерними для академічного стилю, в якому написані речення з датасету, але, тим не менш, це найбільш поширені стоп-слова в українській мові.

2.3 Вибір математичної моделі

Враховуючи особливості датасету було виділено чотири математичні моделі, які варто перевірити на ефективність розпізнавання іменованих сутностей заданого тексту.

Перша – це ***лінійна регресія***. Один із найпростіших способів побудови інтелектуального класифікатору, який обчислює лінійний зв'язок (рис. 2.1) між залежною змінною та однією чи кількома незалежними ознаками. Теоретично для поставленої задачі цього може бути достатньо, але що більш важливе в дослідженні – лінійна регресія надає можливості для прозорої інтерпретації моделі, а в перспективі її можна масштабувати для задачі мультикласової ідентифікації.

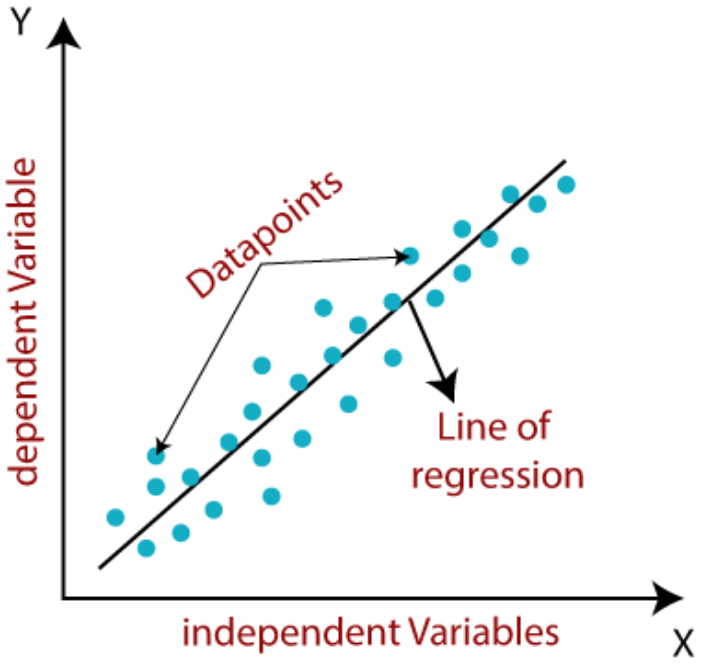


Рисунок 2.1 – Приклад лінійної регресії

Середній показник точності для тестової вибірки склав: 0.63%. Результат не найгірший, але не достатній, адже при такому покажчику система більше вгадує, ніж ідентифікує.

Другою моделью розглянемо ***«наївний баєвський класифікатор»*** (рис.2.2)**,** який заснований на однойменній теоремі розподілу ймовірностей, яка дозволяє сформувати вірогідність настання гіпотези, коли доступні апріорні знання.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)},$$

де $P(A|B)$ - апостеріорна ймовірність, тобто ймовірність гіпотези А, якщо подія В відбудеться. $P(B|A)$ - це ймовірність правдоподібності, тобто ймовірність доказів того, що гіпотеза А вірна. $P(A)$ - це попередня ймовірність, тобто ймовірність гіпотези до спостереження доказів, а *P(B)* — гранична ймовірність, тобто ймовірність доказів.

На практиці цей алгоритм частіше зустрічається в задачах визначення настрою тексту, його емоційного тону, тощо. Тим не менш його використання цілком виправдане й в задачах класифікації іменованих сутностей, хоча можливі проблеми при масштабуванні й не кожний датасет підійде для навчання.

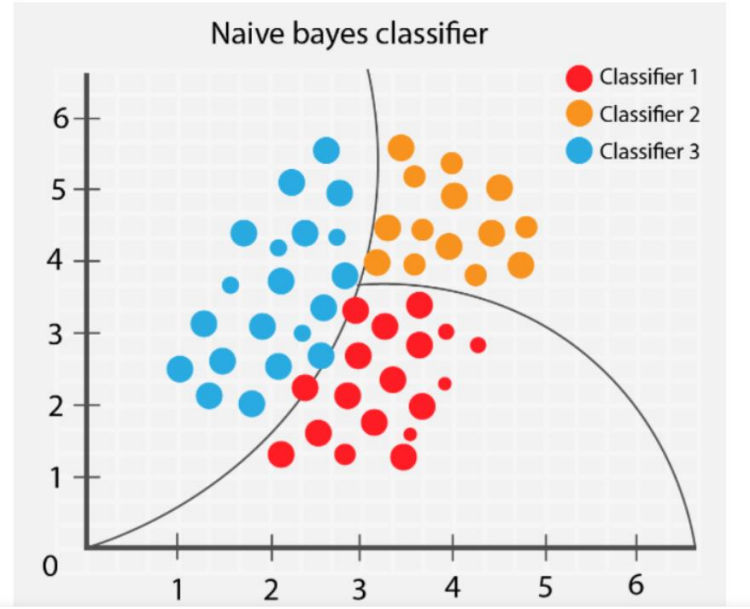


Рисунок 2.2 – Приклад баєвського класифікатору

При тестуванні навчальної вибірки була отримана точність ідентифікації – 0.83 %. У більшості випадках цього може бути достатньо, але є ризик, що цей покажчик зменшиться для баєвського класифікатору, якщо розширити датасет менш формальними реченнями.

Метод опорних векторів (SVM) — це керований алгоритм машинного навчання, який можна використовувати як для класифікації, так і для регресії. Під час навчання, алгоритм малює кожен елемент даних як точку в n-вимірному просторі (де n – кількість ознак), а значення кожної ознаки має конкретні координати. У результаті можна виконати класифікацію, знаходячи гіперплощину, яка дуже добре розрізняє класи.

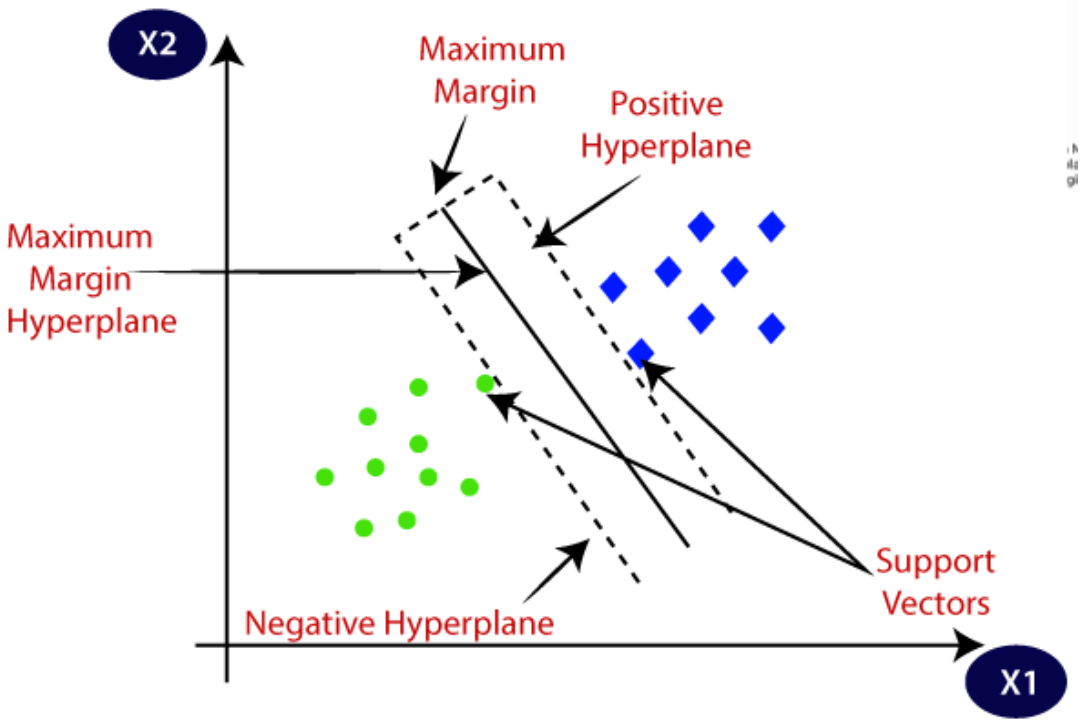


Рисунок 2.3– Приклад методу опорних векторів

Хоч, у деякому сенсі, алгоритм опорних векторів нагадує лінійну регресію, проте отримана середня точність класифікації зросла і становить: 0,82%. Не поганий результат, але метод SVM більше підходить для інших задач NLP. Наприклад, розпізнавання емоційного забарвлення тексту, аніж ідентифікація сутностей.

Останнім розглянутим метод машинного навчання інтелектуальної компоненти буде ***логістична регресія***. Цей тип статистичної моделі часто використовується для класифікації та прогнозної аналітики. Логістична регресія оцінює ймовірність події, наприклад проголосував або не проголосував, на основі заданого набору даних незалежних змінних. Оскільки результат є ймовірністю, залежна змінна обмежена між 0 і 1. У логістичній регресії до шансів застосовується, так зване, логічне-перетворення, тобто ймовірність успіху, поділена на ймовірність невдачі. Це також широко відомо як логарифм шансів, або натуральний логарифм шансів, і ця логістична функція представлена такими формулами:

$$logreg(\pi) = \frac{1}{1 + \exp(-\pi)}$$

$$\ln\left(\frac{\pi}{1-\pi}\right) = ß_1 X_1 + \cdots + ß_k X_k$$

У цьому рівнянні логістичної регресії $logreg(\pi)$ є залежною змінною або змінною відповіді, а X_k є незалежною змінною. Бета-параметр, або коефіцієнт, у цій моделі зазвичай оцінюється за допомогою оцінки максимальної правдоподібності (MLE). Цей метод перевіряє різні значення бета-версії за допомогою кількох ітерацій для оптимізації найкращого підходу логарифмічних коефіцієнтів. Усі ці ітерації створюють логарифм правдоподібності, а логістична регресія прагне максимізувати цю функцію, щоб знайти найкращу оцінку параметра. Після того, як оптимальний коефіцієнт (або коефіцієнти, якщо існує більше однієї незалежної змінної) знайдено, умовні ймовірності для кожного спостереження можна обчислити, зареєструвати та підсумувати, щоб отримати прогнозовану ймовірність. Для двійкової класифікації ймовірність, менша за 0,5, передбачить 0, а ймовірність, більша за 0, передбачить 1 (рис.2.4). Після обчислення моделі найкраще оцінити, наскільки добре модель прогнозує залежну змінну.

Рисунок 2.4– Приклад логістичної регресії

На тестовій вибірці логістична регресія отримала 0.93% точності, що є найкращим результатом серед розглянутих методів. Також важливо помітити, що цей метод повинен без особливих проблем мастабуватися.

2.4 Візуальний аналіз датасету

Оскільки кожне речення конвертується у вектор ознак, то ми можемо розмістити їх у відповідному n-мірному просторі для візуального аналізу датасету. Це дозволить, як мінімум, оцінити розподіл класів розпізнавання або ж виявити нові кластери.

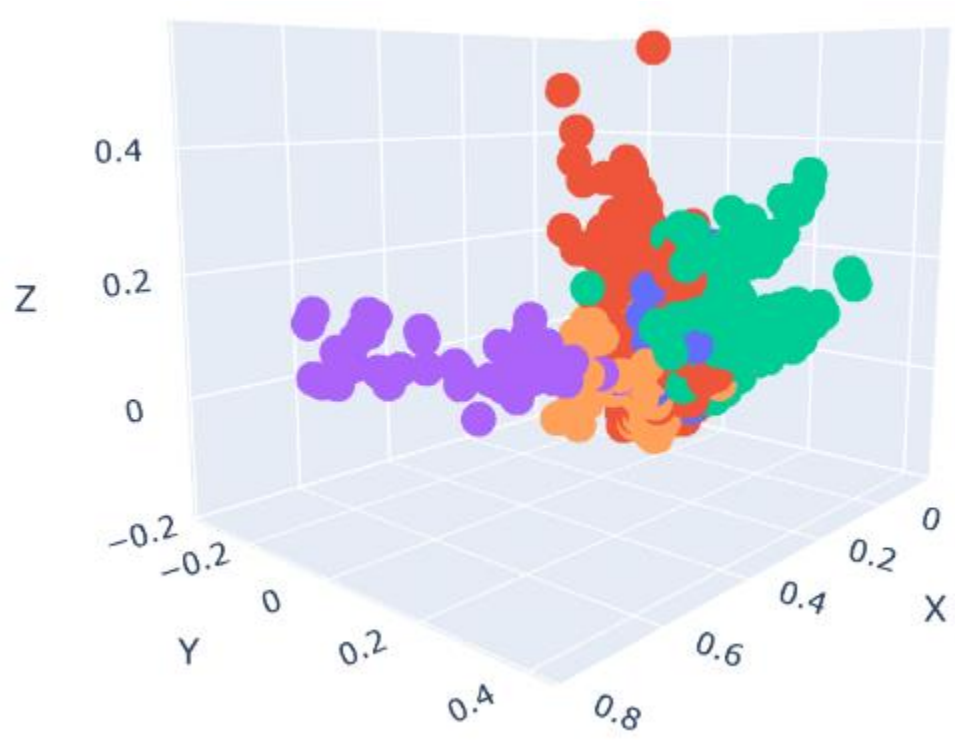


Рисунок 2.5. 3Д візуалізація простору ознак

Оскільки ми працюємо з n-мірним простором, де n – кількість ознак (у межах задачі кількість унікальних слів в датасеті), то не можливо відобразити подібий графік, але можна через метод головних компонент [19] звузити простір ознак до зрозумілої 3-х вимірної візуалізації.

На рисунку 2.5. кожна точка відповідає за одне речення, а її колір за клас до якого воно відноситься. Можна побачити, що алфавіт ознак створює комето-подібну фігуру з трьома хвостами, кожен з яких відповідає за окремий клас. Це означає, по-перше, що класи розпізнавання були обрано правильно, адже їх перетин мінімальний. По-друге, що в просторі ознак не має аномалій. Таких речень, які утворюють поодинокі конгломерати деінде.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Нормалізація датасету

Будь-який текст містить в собі не інформативні ознаки. Такі як: знаки пунктуації, прийменники, вставні слова, сполучники тощо. У більшості випадках вони не впливають на зміст, а лише роблять текст більш привабливим. На початковому етапі навчання від цих частин мови можна позбутися.

Усі слова повинні бути приведені до деякої нормальної форми. Найраціональніше використати один із двох підходів:

Стеммінг. Процес скорочення слова до його основи шляхом відкидання допоміжних частин, таких як закінчення чи суфікс. Приклад: выраженное -> выражен; мотивация-> мотивац; определяется -> определя; окружающего -> окружа;

Лематизація. Процес приведення слова до його початкової форми. Застосовується окрема попередньо навчена інтелектуальна система Приклад: выраженное -> выраженный; мотивация-> мотивация; определяется -> определяться; окружающего -> окружающий;

В незалежності від обраного підходу ми використаємо бібліотеку для python NLTK. Вона підтримує функції класифікації, токенізації, виділення коренів, тегів, синтаксичного аналізу та семантичного маркування. Також там є модулі, які дозволяють графічно продемонструвати роботу з синтаксичними даними.

На даному етапі можна виділити ряд можливих алгоритмічних проблем:

1) Розподіл розмічених класів в навчальній вибірці може бути не рівномірний. Умовно кажучи, до «Мотивація» експерт відніс 250 речень, а до «Особливості результативності» лише 30. У такому випадку, теоретично, система може не навчитися розпізнавати деякі класи із-за не достатньої кількості апріорної інформації про них. Тоді необхідно спробувати доповнити базу даних.

2) Може статися, що деякі з класів мають достатньо сильний перетин своїх ознак. Наприклад, речення, що відносяться до «Особливості результативності» можуть бути подібними до «Особливості взаємодії з іншими людьми». У такому випадку можна спробувати або розширити навчальну вибірку, щоб у системи було достатньо інформації для розрізнення цих класів. Або експерту переосмислити класи (зменшити їх кількість; узагальнити). Або виконувати класифікацію за ієрархією. Створити деякі узагальнені класи-віхи на які спочатку будуть розбиватися речення, а потім вже серед них шукати під класи

Як було зазначено раніше нам потрібний токенізатор, який би розбив текст на речення, а їх вже на слова (токени). Оскільки NLTK розроблялась для роботи з англомовним текстом, то потрібно ввести деякі зміни. Наприклад реалізувати власний токенізатор для української мови.

Особливістю структури обраних речень є наявність синонімів, які знаходятся в дужках. На поточному етапі дослідження вони не представляють цінність, тому видалимо їх.

```
sentenses = []

for sub in classes.text:
    string = re.sub('\(.*?\)', '', sub)
    sentenses.append(string)

data['NoSynonyms'] = sentenses
```

Наступним кроком є видалення стоп-слів, знаків пунктуації і чисел. Припускається, що ця інформація не впливає на зміст речення.

```
import string
def remove_punctuation(text):
    return "".join([ch if ch not in string.punctuation else ' '
for ch in text])

def remove_numbers(text):
    return ''.join([i if not i.isdigit() else ' ' for i in text])

import re
def remove_multiple_spaces(text):
     return re.sub(r'\s+', ' ', text, flags=re.I)
```

```
from nltk.stem import *
from nltk.corpus import stopwords
from pymystem3 import Mystem
from string import punctuation

stopwords = []
stopwords.extend(['…', '«', '»', '...'])

prep_text =
[remove_multiple_spaces(remove_numbers(remove_punctuation(text.low
er()))) for text in tqdm(data["NoSynonyms"])]

data['text_prepNoSynonyms'] = prep_text
```

Останнім кроком був процес нормалізації даних через стемінг речень.

```
from nltk.tokenize import word_tokenize

def makeStemmed(texts):
    stemmed_texts_list = []
    for text in tqdm(texts):
        tokens = word_tokenize(text)
        stemmed_tokens = [stemmer.stem(token) for token in tokens
if token not in stopwords]
        text = " ".join(stemmed_tokens)
        stemmed_texts_list.append(text)
    return stemmed_texts_list

data['text_stem'] = makeStemmed(data['text_prepNoSynonyms'])
```

Отриманий нормалізований датасет є повністю придатним для подальшої розробки інтелектуальної компоненти.

3.2 Результати машинного навчання NLP моделі

Оскільки, наш датасет вже промаркований, то для машинного навчання та подальшої верифікації результатів розіб’ємо його на навчальну та тестову вибірку.

```
X = data['text_stem'].values.astype('U')
y = data['mark']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.4, random_state = 42)
```

Серед базових алгоритмів машинного навчання була обрана логістична регресія, яка дозволила, при мінімальному часі виконання, досягти високої достовірності прийняття класифікаційних рішень на тестовій вибірці.

accuracy 0.924956369982548

	precision	recall	f1-score	support
0	0.88	0.92	0.90	53
1	0.92	0.96	0.94	293
2	0.96	0.93	0.94	100
3	0.94	1.00	0.97	17
4	0.93	0.83	0.87	110
accuracy			0.92	573
macro avg	0.93	0.93	0.93	573
weighted avg	0.93	0.92	0.92	573

Рисунок 3.1 – Оцінка результату навчання NLP моделі

Як видно з рисунку 3.1 точність моделі достатньо висока й дозволяє ідентифікувати текст тестової вибірки з точністю 92%. Це говорить про високу достовірність розробленої моделі. Як варіант удосконалення алгоритму – пропонується реалізувати модуль, який би відповідав за перенавчання під час функціонування системи через розширення словника ознак та алфавіту класів розпізнавання.

У перспективі треба додати можливість мультикласової ідентифікації речень, щоб один і той самий об’єкт міг відноситись до декількох класів. Це підвищить інтерпретованість результату ідентифікації.

ВИСНОВКИ

В бакалаврській кваліфікаційній роботі досліджено та реалізовано класифікатор, який здатний маркувати український текст згідно детермінованого алфавіту класів розпізнавання. При цьому враховані як особливості навчального датасету, так й унікальні маркери української мови. Основними результатами є:

1. Проведений детальний аналіз словника ознак і його відповідність до класів розпізнавання. Таким чином виявлено, що більшість електронних листів-резюме можна розділити на наступні частини:

- перелік навичок та талантів, які можуть бути корисними при виконанні очікуваної роботи;
- психо-емоційна складова кандидата, яка не завжди проявляється у явному вигляді;

2. Розроблена інтелектуальна компонента дозволяє оцінювати резюме кандидатів зі сторони п’яти психо-емоційних складових характеру людини: результативність в роботі, мотивація, потреби, особливості взаємодії з оточуючими та емоції. Річ у тім, що при розгляданні резюме кандидата роботодавець враховує не лише компетенцію в прикладних навичках, якими рецензент і так повинен володіти, якщо претендую на ту чи іншу позицію, а й його психологічний портрет. Це важливо, щоб оцінити наскільки гарно людина зможе інтегруватися в соціальну екосистему конкретної команди.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Egger, Roman. "Text Representations and Word Embeddings: Vectorizing Textual Data." Applied Data Science in Tourism: Interdisciplinary Approaches, Methodologies, and Applications. Cham: Springer International Publishing, 2022. 335-361.
2. Christian, B. (2021) The alignment problem machine learning an human values. W. w. Norton&Company, 2021. - 477 p.
3. Bao, Y., Quan, C., Wang, L., & Ren, F. (2014). The role of pre-processing in twitter sentiment analysis. In Intelligent Computing Methodologies: 10th International Conference, ICIC 2014, Taiyuan, China, August 3-6, 2014. Proceedings 10 (pp. 615-624). Springer International Publishing.
4. Кряжич, О. О., Трофимчук, О. М. (2022) Метод обробки неструктурованої інформації на веб-ресурсах, International Scientific Technical Journal "Problems of Control and Informatics", 67(4), с. 106–115. doi:10.34229/2786-6505-2022-4-8.
5. Ko, Y. (2012, August). A study of term weighting schemes using class information for text classification. In Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval (pp. 1029-1030).
6. Биков, М. М. Основи інтелектуальних технологій. Частина 1. Технології розпізнавання : електронний навчальний посібник комбінованого (локального та мережного) використання [Електронний ресурс] / Биков М. М., Ковтун В. В., Гаврилюк В. О. – Вінниця : ВНТУ, 2023. – 229 с.
7. Dovbysh, A., Shelehov, I., Romaniuk, A., Moskalenko, R., & Savchenko, T. (2023). Decision-making support system for diagnosis of oncopathologies by histological images. Journal of Pathology Informatics, 14, 100193.
8. Majumdar, Srijita, et al. "Sarcasm analysis and mood retention using NLP techniques." International Journal of Information Retrieval Research (IJIRR) 12.1 (2022): 1-23.

9. Jelodar, H., Wang, Y., Yuan, C., Feng, X., Jiang, X., Li, Y., & Zhao, L. (2019). Latent Dirichlet allocation (LDA) and topic modeling: models, applications, a survey. Multimedia Tools and Applications, 78, 15169-15211.
10. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
11. Савченко, Т. Р. Інформаційна технологія персоналізованого діагностування ранньої стадії раку передміхурової залози. MS thesis. Сумський державний університет, 2023.
12. Hardeniya, N., Perkins, J., Chopra, D., Joshi, N., & Mathur, I. (2016). Natural language processing: python and NLTK. Packt Publishing Ltd.
13. Butler, A. (2020). From discourse to logic with stanford CoreNLP and treebank semantics. In New Frontiers in Artificial Intelligence: JSAI-isAI International Workshops, JURISIN, AI-Biz, LENLS, Kansei-AI, Yokohama, Japan, November 10–12, 2019, Revised Selected Papers 10 (pp. 182-196). Springer International Publishing.
14. [Електронний ресурс]. A short introduction to NLP in Python with spaCy. https://towardsdatascience.com/a-short-introduction-to-nlp-in-python-with-spacy-d0aa819af3ad
15. Akbik, A., Bergmann, T., Blythe, D., Rasul, K., Schweter, S., & Vollgraf, R. (2019, June). FLAIR: An easy-to-use framework for state-of-the-art NLP. In Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics (demonstrations) (pp. 54-59).
16. [Електронний ресурс]. Stanza – A Python NLP Package for Many Human Languages. https://stanfordnlp.github.io/stanza/
17. Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., & Mikolov, T. (2016). Fasttext. zip: Compressing text classification models. arXiv preprint arXiv:1612.03651.

18. Dovbysh, A. S., Shelehov, I. V., Romaniuk, A. M., Moskalenko, R. A., & Savchenko, T. R. (2023). Decision-Making Support System for Diagnosis of Breast Oncopathologies by Histological Images. Cybernetics and Systems Analysis, 1-10.
19. Maćkiewicz, A., & Ratajczak, W. (1993). Principal components analysis (PCA). Computers & Geosciences, 19(3), 303-342.
20. Савченко, Т. Р. (2021). Інформаційна технологія розпізнавання об’єктів. Машинне навчання бортової системи розпізнавання наземних об’єктів.
21. Naumenko, Igor, et al. "Information-Extreme Machine Learning of an On-board Ground Object Recognition System with a Choice of a Base Recognition Class." (2022).
22. Denysenko, A. P., Savchenko, T. R., Dovbysh, A. S., Romaniuk, A. M., & Moskalenko, R. A. (2022). Artificial Intelligence Approach in Prostate Cancer Diagnosis: Bibliometric Analysis.

ДОДАТОК

```
import pandas as pd

import re

data = pd.read_csv('dataset.csv')

sentenses = []

for sub in classes.text:
    string = re.sub('\(.*?\)', '', sub)
    sentenses.append(string)

data['NoSynonyms'] = sentences

import numpy as np

from tqdm.auto import tqdm, trange

import nltk

import string

def remove_punctuation(text):
    return "".join([ch if ch not in string.punctuation else ' '
for ch in text])

def remove_numbers(text):
    return ''.join([i if not i.isdigit() else ' ' for i in text])

import re

def remove_multiple_spaces(text):
     return re.sub(r'\s+', ' ', text, flags=re.I)

from nltk.stem import *

from nltk.corpus import stopwords

from pymystem3 import Mystem

from string import punctuation

stopwords = []

stopwords.extend(['…', '«', '»', '...'])

prep_text =
[remove_multiple_spaces(remove_numbers(remove_punctuation(text.low
er()))) for text in tqdm(data["NoSynonyms"])]
```

```
data['text_prepNoSynonyms'] = prep_text

data

from nltk.stem.snowball import SnowballStemmer

from nltk.tokenize import word_tokenize

nltk.download('punkt')

from nltk.tokenize import word_tokenize

def makeStemmed(texts):
    stemmed_texts_list = []
    for text in tqdm(texts):
        tokens = word_tokenize(text)
        stemmed_tokens = [stemmer.stem(token) for token in tokens
if token not in stopwords]
        text = " ".join(stemmed_tokens)
        stemmed_texts_list.append(text)
    return stemmed_texts_list

data['text_stem'] = makeStemmed(data['text_prepNoSynonyms'])

X = data['text_stem'].values.astype('U')

y = data['mark']

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.4, random_state = 42)

from sklearn.pipeline import Pipeline

from sklearn.feature_extraction.text import TfidfTransformer

from sklearn.feature_extraction.text import CountVectorizer

from sklearn.metrics import accuracy_score

from sklearn.metrics import classification_report

from sklearn.linear_model import LogisticRegression

logreg = Pipeline([('vect', CountVectorizer()),
                ('tfidf', TfidfTransformer()),
                ('clf', LogisticRegression(n_jobs=1, C=1e5)),
               ])

%%time
```

```
logreg.fit(X_train, y_train)

%%time

y_pred = logreg.predict(X_test)

print('accuracy %s' % accuracy_score(y_pred, y_test))

print(classification_report(y_test, y_pred))

from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.decomposition import PCA

import plotly.express as px

markersList = ["Емоції", "Особливості взаємодії з людьми",
"Потреби", "Мотивація", "Результативність в роботі"]

pipeline = Pipeline([
  ('vect', CountVectorizer()),
  ('tfidf', TfidfTransformer())])

vectorizer = TfidfVectorizer(sublinear_tf=True)

X = pipeline.fit_transform(data.text_stem)

pca = PCA(n_components=3)

data3D = pca.fit_transform(X.toarray())

index = 0

for mark in markersList:
    dataFrame = data.replace(index, mark)
    index += 1

d = {'X': data3D[:,0], 'Y': data3D[:,1], 'Z':data3D[:,2], 'text':
dataFrame.text, 'mark': dataFrame.mark}

df = pd.DataFrame(data=d)

#log_x=True
fig = px.scatter_3d(df, x="X", y="Y", z='Z', color='mark',
                 hover_name="text")

fig.show()
```