МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Михайлишин Владислав Юрійович

Кваліфікаційна наукова
праця на правах рукопису
УДК 004.652

Система підтримки та прийняття рішень аналізу рекрутингової діяльності на основі великих даних

122 – комп’ютерні науки

Дисертація на здобуття наукового ступеня
доктора філософії

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

_____/В. Ю. Михайлишин/

Науковий керівник

Бойко Наталія Іванівна

кандидат економічних наук, доцент

Львів – 2024

АНОТАЦІЯ

Михайлишин В. Ю. Система підтримки та прийняття рішень аналізу рекрутингової діяльності на основі великих даних. – Кваліфікаційна наукова праця на правах рукопису. Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 122 “Комп’ютерні науки”. – Національний університет «Львівська політехніка», Львів, 2024.

Зміст анотації. Дисертаційна робота присвячена побудові методів та засобів для підтримки прийняття рішень у рекрутинговій діяльності на основі великих даних. У сучасному світі великих даних процеси рекрутингу можуть бути значно покращені за допомогою методів та засобів, які досліджуються в роботі через розробку системи підтримки прийняття рішень та оптимізації процесів рекрутингу, а також розробку нових підходів та методів для оптимізації процесів відбору та оцінки кандидатів.

У першому розділі «Аналіз методів обробки великих даних рекрутингової сфери» досліджуються методи, що використовуються для обробки великих даних у рекрутингу, з особливим акцентом на гетерогенність даних. У ньому детально розглядаються характеристики, пов'язані з рекрутинговими даними, а також різні методи обробки, такі як вибірка, трансформація, зменшення шуму та вилучення ознак, які можна використовувати при роботі з гетерогенними наборами даних. Також розглядаються методи роботи з гетерогенними даними.

Розглядаються методи природних мовних обробників для автоматичної категоризації резюме та вакансій для полегшення процесу пошуку та відбору кандидатів. Охоплено методи обробки природної мови, які використовуються для автоматизації категоризації резюме та вакансій за допомогою методів обробки природної мови. Аналізуються різні статистичні та лінгвістичні підходи, методи опорних векторів та умовних випадкових полів, а також методи токенізації та приховані марковські моделі, метод TF-IDF.

У другому розділі «Розробка алгоритму обробки даних з використанням методів аналізу природної мови та кластеризації» проаналізовано різні методи кластеризації даних, такі як K-means, K-medians, DBSCAN, C-means. Випробувано їх на досліджуваному наборі даних.

Для розглянутих методів проведено оцінювання результатів роботи за допомогою індексів Силуета та Данна.

Прийнято рішення про використання ансамблю методів кластеризації через нестабільність результатів на різних наборах даних.

Розроблено власний метод паралельної кластеризації даних з використанням ансамблю існуючих методів, та вибору кінцевого кластера для визначених точок за допомогою голосування.

У третьому розділі «Розробка методу побудови системи підтримки прийняття рішень та використання зворотнього зв’язку для покращення рекомендацій» проведено аналіз методів регресії, таких як дерева рішень, випадковий ліс, XGBoost, виконано оцінку їх роботи за допомогою статистичних показників, вибрано найкращий метод для застосування в системі. Розроблено метод визначення рівня задоволеності користувача та врахування зворотнього зв’язку для корекції оцінки. Розроблено алгоритм системи підтримки прийняття рішень з застосуванням обраних методів.

У четвертому розділі «Розробка архітектури і апробація результатів» описано розроблення системи збору, обробки, аналізу та оцінки відгуків кандидатів для покращення процесів рекрутингу. Для цього використано поглиблений аналіз даних зворотного зв'язку, зокрема оцінку їхньої якості та корисності, аналіз тенденцій, визначення ключових сфер для вдосконалення та виявлення трендів. Об’єднано ці результати з наявними системними даними для підвищення точності та ефективності систем підтримки прийняття рішень з рекрутингу.

Також в даному розділі представлено розроблену архітектуру системи, обгрунтовано обрані технології та технічні рішення під час реалізації системи.

Визначено набір сервісів та додаткової інфраструктури для роботи системи. Представлено готовий додаток системи підтримки прийняття рішень. Здійснено аналіз отриманих результатів.

У висновка йдеться про результати, що були отримані в ході дослідження, де вони можуть використані та як до них дійшли.

Список літературних джерел наведений після висновків, що включає 101 літературне джерело.

У додатку А представлено акти впровадження результатів дисератційної роботи. Запропоновані методи і моделі впроваджені у навчальний процес Національного університету «Львівська політехніка» при викладанні дисципліни «Об'єктно-орієнтоване програмування» та «Проектування інформаційних систем» (підтверджено актом впровадження). Також результати дисертаційної роботи впроваджено в ТзОВ «Палетний сервіс» (підтверджено актом впровадження). Впровадження в ДБ «Технологія опрацювання мультимодальних українськомовних наборів даних для визначення рівня стресу» (№ держ. реєстру 0123U100231).

Основні наукові результати дисертації опубліковано в 6 працях, зокрема: 1 праці в науковому фаховому виданні інших держав, 4 публікаціях в наукових фахових виданнях України, та 1 матеріалах конференцій.

Ключові слова: методи кластеризації, методи обробки великих даних, великі дані, системи підтрими прийняття рішень, рекрутингова діяльність, мовні обробники, неоднорідні дані, регресія.

ABSTRACT

Mykhailyshyn V. Y. Decision support system of recruiting activities analysis based on big data. Dissertation for the degree of Doctor of Philosophy in specialty 122 "Computer Science." - Lviv Polytechnic National University, Lviv, 2024.

Abstract content. The dissertation is devoted to the development of methods and tools for decision support in recruitment activities based on big data. In today's world of big data, recruitment processes can be significantly improved with the help of such methods and tools. This is explored in the paper through the development of a decision support system and optimisation of recruitment processes, as well as the development of new approaches and methods to optimise the selection and evaluation of candidates.

The first chapter, "Analysing big data processing methods in recruitment" explores the methods used to process big data in recruitment, with a particular focus on data heterogeneity. It discusses in detail the characteristics associated with recruitment data, as well as various processing techniques such as sampling, transformation, noise reduction, and feature extraction that can be used when dealing with heterogeneous data sets. Methods for working with heterogeneous data are also covered.

Natural language processing methods for automatic categorisation of resumes and vacancies to facilitate the process of searching and selecting candidates are considered. Natural language processing techniques used to automate the categorisation of resumes and job postings using natural language processing techniques are covered. Various statistical and linguistic approaches, support vector machines and conditional random fields, as well as tokenisation methods and hidden Markov models, TF-IDF method are analysed.

In the second chapter "Development of a data processing algorithm using natural language analysis and clustering methods", we analyse various data clustering methods, such as K-means, K-medians, DBSCAN, C-means. They are tested on the studied data set.

For the considered methods, the results were evaluated using the Silhouette and Dunn indices.

The decision was made to use an ensemble of clustering methods due to the instability of the results on different data sets.

An own method for parallel data clustering using an ensemble of existing methods and selecting the final cluster for the defined points by voting is developed.

The third chapter "Development of a Method for Building a Decision Support System and Using Feedback to Improve Recommendations," analyses regression methods such as decision trees, random forest, and XGBoost, evaluates their performance using statistical indicators, and selects the best method for use in the system. A method for determining the level of user satisfaction and taking into account feedback to correct the assessment is developed. An algorithm for a decision support system using the selected methods was developed.

The fourth chapter, " Developing the architecture and testing the results" describes the development of a system for collecting, processing, analyzing, and evaluating candidate feedback to improve recruiting processes. This involves in-depth analysis of feedback data, including quality and usefulness assessment, trend analysis, identification of key areas for improvement, and trend detection. These results are integrated with existing system data to enhance the accuracy and efficiency of decision support systems for recruiting.

The conclusion refers to the results obtained during the research, where they can be used and how they were arrived at.

The list of literary sources is given after the conclusions, which includes 101 literary sources.

Appendix A presents the acts of implementation of the results of the dissertation work. The proposed methods and models are implemented in the educational process of the National University «Lviv Polytechnic» when teaching the discipline «Object-oriented programming» and «Designing information systems» (confirmed by the act of

implementation). Also, the results of the dissertation were implemented in the «Палетний сервіс» LLC (confirmed by the act of implementation). Implementation in DB «Technology for processing multimodal Ukrainian-language data sets to determine the level of stress» (state register number 0123U100231).

The main scientific results of the thesis are published in 6 papers, including: 1 paper in the scientific professional edition of foreign country, 4 publications in scientific professional editions of Ukraine and 1 conference papers.

Keywords: clustering methods, big data processing methods, big data, decision support systems, recruitment activities, language processors, heterogeneous data, regression.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Статті у виданнях інших держав:

1. Shakhovska, N., Kaminskyy, R., Khudoba, B., Mykhailyshyn, V., Helzhynskyi, I. A Novel Methodology Analyzing the Influence of Micro-Stresses on Human-Centric Environments. Computation, 2023, 11(11), 224. https://doi.org/10.3390/computation11110224 (квартиль Q2 у НМБД Scopus).

Статті у фахових виданнях України:

2. Бойко Н. І., Михайлишин В. Ю. Алгоритм класифікації текстового контенту соціальних мереж для визначення емоційного тону. Вісник Херсонського національного технічного університету, № 2(85) (2023): с. 133-140. https://doi.org/10.35546/kntu2078-4481.2023.2.18
3. Бойко Н. І., Михайлишин В. Ю. Оцінка ефективності рекурсивного процесу розподілу набору даних з використанням алгоритму CART. Вісник Хмельницького національного університету. Серія: «Технічні науки», №4, 2023, с. 25-35. https://www.doi.org/10.31891/2307-5732-2023-323-4-25-35
4. Boyko N. I., Mykhailyshyn V. Yu. K-NN'S NEAREST NEIGHBORS METHOD FOR CLASSIFYING TEXT DOCUMENTS BY THEIR TOPICS. Радіоелектроніка, інформатика, управління. 2023. № 3. (Radio Electronics, Computer Science, Control. 2023. № 3) pp. 83-97. https://www.doi.org/10.15588/1607-3274-2023-3-9 (WOS)
5. Бойко Н. І., Шаховська Н. Б., Михайлишин В. Ю. Розроблення методу класифікації користувачів за рівнем стресостійкості з використанням модифікованої автоасоціативної нейронної мережі // Вісник Хмельницького національного університету, № 6, 2021 (303), с. 64-68. https://www.doi.org/10.31891/2307-5732-2021-303-6-64-68

Матеріали конференцій:

6. Boyko N., Mykhailyshyn V. Methods of Searching for Associative Rules for Inhomogeneous Data in Semantic Networks. Proceedings of the 3rd International Workshop on Intelligent Information Technologies & Systems of Information Security Khmelnytskyi, Ukraine, March 23–25, 2022, pp. 54-71.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

СППР – система підтримки та прийняття рішень;

СППРКД – система підтримки та прийняття рішень керована даними;

ВД – великі дані;

AWS – Amazon web services;

API – application programming interface;

RF - random forest;

DB – database;

TF-IDF - Term Frequency Inverse Document Frequency of records;

XGBoost - eXtreme Gradient Boosting.

ЗМІСТ

ВСТУП

Актуальність теми. Управління ресурсами лежить в основі будь-якої успішної організації. Однак, оскільки ринок праці продовжує зростати і ставати більш доступним з вражаючою швидкістю, рекрутингова діяльність повинна адаптуватися до нових викликів і можливостей, щоб залишатися ефективною та актуальною. Великі дані відкривають нові можливості для більш точного пошуку, оцінки та відбору кандидатів, а також для більш комплексного аналізу ринків праці та більш ефективного підбору персоналу.

Останні глобальні події, такі як епідемія Covid-19, світова економічна криза внаслідок великої кількості збройних конфліктів, зокрема російсько-Українська війна, конфлікт в Секторі Газа, привели до значного (5% - 40% від усієї кількості робочих контрактів) скорочення, а потім перерозподілу робочих місць, та росту вимог до рівня кандидатів як наслідку [7].

Зі зростанням конкуренції за таланти та необхідністю оптимізації витрат на рекрутинг, система підтримки прийняття рішень на основі аналітики великих даних стала вкрай важливою. Такий підхід дозволяє більш ефективно відбирати кандидатів і прогнозувати їхній вплив та внесок в успіх бізнесу. Крім того, такий підхід сприяє об'єктивному підходу до підбору персоналу, з меншою кількістю людських факторів та упереджень, що впливають на рішення, які приймаються під час рекрутингового процесу.

Зокрема, можна виділити наступні аспекти актуальності такої системи:

Складний ринок праці: Сучасний ринок праці є динамічним і складним, а технологічні інновації створюють нові виклики для стратегій підбору персоналу. Великі дані дають інструменти для кращого розуміння цих змін, виявлення тенденцій та відповідної оптимізації стратегій підбору персоналу.

Ефективність та оптимізація процесів: Традиційні процеси підбору персоналу часто вимагають значних витрат часу та ресурсів для ефективного виконання. Використовуючи системи на основі великих даних, ви можете

автоматизувати та оптимізувати багато аспектів процесу підбору персоналу, заощаджуючи витрати, одночасно підвищуючи ефективність та скорочуючи час, що витрачається.

Покращення відбору кандидатів: Аналітика великих даних виявилася ефективним способом покращити відбір кандидатів, швидко і точно визначаючи тих, хто найкраще підходить для виконання конкретних ролей на основі ретельного вивчення їхніх компетенцій і потенціалу. Це допомагає зменшити кількість невдалих наймів, одночасно підвищуючи загальну продуктивність працівників.

Передбачуваність та адаптивність: Великі дані не лише дозволяють аналізувати поточну ситуацію, але й дають змогу передбачити майбутні тенденції та кадрові потреби, що дозволяє організаціям бути більш адаптивними та оперативно реагувати на зміни на ринку праці.

Дотримання етичних і правових норм: При рекрутингу необхідно також брати до уваги етичні та правові міркування, захист даних; розроблення методів підтримки прийняття рішень, які відповідають цим вимогам, повинно бути важливим елементом дослідження.

Дослідження та розробка такої системи має величезне практичне та теоретичне значення, оскільки відповідає сучасним тенденціям діджиталізації бізнес-процесів, створюючи більш ефективні та інноваційні підходи до управління людськими ресурсами.

Тому задача дисертаційної роботи розроблення методів та засобів підтримки прийняття рішень для рекрутингової діяльності на основі великих даних є актуальною.

Зв’язок роботи з науковими програмами, планами і темами. Дисертація виконувалася відповідно до пріоритетних напрямків науково-дослідних робіт Національного університету “Львівська політехніка”, відповідно до координаційних планів Міністерства освіти і науки України.

Мета дослідження полягає у розробці методів машинного навчання і засобів підтримки та прийняття рішень аналізу рекрутингової діяльності на основі великих даних для покращення процесу пошуку та відбору кандидатів, підвищення ефективності найму.

Для досягнення поставленої мети в роботі потрібно розв’язати такі **задачі**:

1. Провести аналіз методів опрацювання неоднорідних рекрутингових даних і природних мовних обробників для попередньої підготовки та автоматичної категоризації резюме.
2. Розробити алгоритм для обробки даних, що включає попереднє опрацювання вхідної інформації з використанням методів аналізу природної мови та кластеризації.
3. Розробити систему автоматичної рекомендації кандидатів на основі аналізу неоднорідних структурованих та неструктурованих даних про резюме та профілі кандидатів.
4. Розробити систему автоматичного аналізу та прогнозування зворотного зв'язку від кандидатів з метою постійного вдосконалення рекрутингового процесу.
5. Розробити архітектуру системи і програмні модулі та апробувати їх щодо підтримки прийняття верифікованих рішень на основі великих даних.

Об’єкт дослідження – процеси аналізу рекрутингових даних та застосування в системах підтримки та прийняття рішень.

Предмет дослідження – методи та засоби підтримки та прийняття рішень, методи опрацювання неоднорідних великих даних.

Методи дослідження. У процесі виконання роботи використано методи опрацювання природної мови, методи кластеризації, методи оцінки якості кластеризації, статистичні методи, методи регресії, методи об’єктно-орієнтованого проектування, методи проектування інформаційних системи, методи паралельних та розподілених обчислень.

Наукова новизна одержаних результатів полягає в тому що:

• *вперше розроблено* метод кластеризації неоднорідних напівструктурованих даних на основі ансамблю методів кластеризації, що відрізняється від існуючих застосуванням попередньої зваженої токенізації та пакетним розподіленим опрацюванням даних, що дає змогу підвищити якість кластеризації;

• *вперше розроблено* метод побудови систем підтримки та прийняття рішень на основі даних з використанням алгоритму градієнтного бустингу та застосуванням попередньої підготовки даних для усунення проблеми перенавчання та отримання зворотного зв'язку стосовно результатів роботи, що дає змогу обчислити індексу впливу на оцінку, що дозволяє коригувати її в залежності від отриманого зворотнього зв'язку, а відтак підвищити швидкість та ефективність процесу рекрутингу;

• *отримав подальший розвиток* комбінований метод зменшення розмірності даних рекрутингової діяльності, який відрізняється від існуючих поєднанням відсіювання аномальних даних та об'єктів, а також обмеження кількості токенів, що дає змогу збільшити точність та швидкість подальшого опрацювання таких даних.

Практичне значення одержаних результатів. Практична цінність роботи полягає у доведенні отриманих наукових результатів до конкретних технологій, методик, алгоритмів та програмних продуктів. На основі методів було розроблено архітектуру системи з орієнтуванням на веб-застосунок. Для застосування алгоритмів у режимі реального часу запропоновано використовувати лямбда-архітектуру, що забезпечує масштабованість пам'яті та зменшує навантаження на сервер у 2 рази. Розроблено мобільний додаток, у якому реалізовано усі наукові результати.

Запропоновані методи і моделі, які впроваджені у навчальний процес Національного університету «Львівська політехніка» при викладанні дисциплін «Проектування інформаційних систем» та «Об'єктно-орієнтоване програмування»

Також результати дисертаційної роботи були використані для розроблення системи підтримки та прийняття рішень у рекрутингу в ТзОВ “Палетний сервіс” м. Львів.

Особистий внесок здобувача. Основні положення та результати дисертаційної роботи одержані автором самостійно. Особисто здобувачеві належать наступні наукові результати: розроблено модель кластеризації неоднорідних даних ринку праці [2, 5], вдосконалено комбінований метод зменшення розмірності даних рекрутингової діяльності [1, 3, 4], систему прийняття рішень на основі даних з використанням алгоритму XGBoost [6], розроблено архітектуру розподіленого опрацювання інформації.

Апробація результатів дисертації. Результати дисертаційної роботи доповідались на конференції: Proceedings of the 3rd International Workshop on Intelligent Information Technologies & Systems of Information Security Khmelnytskyi 2022. Також результати доповідались на семінарах кафедри систем штучного інтелекту «Національного університету «Львівська політехніка».

РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ ОБРОБКИ ВЕЛИКИХ ДАНИХ РЕКРУТИНГОВОЇ СФЕРИ

У розділі проведено огляд особливостей даних рекрутингової сфери та джерел їх отримання. Проаналізовано методи обробки великих даних, наведено опис та варіанти використання основних їх видів. Проведено аналіз методів опрацювання неоднорідних даних, продемонстровано роботу окремих методів на практиці.

Результати розділу опубліковано у працях автора [1, 6]

1.1. Особливості даних рекрутингової сфери та джерел їх отримання.

Дані про рекрутинг різноманітні та об'ємні, що вимагає аналітичної стратегії для управління їхньою обробкою. До первинних джерел належать резюме кандидатів, профілі в соціальних мережах, оголошення про вакансії, бази даних компаній, а також відгуки та рейтинги як роботодавців, так і кандидатів[8].

Ці джерела надають широкий спектр даних, включаючи не лише кількісну інформацію, таку як досвід роботи та освіта, а й якісні аспекти, такі як навички, відгуки та рекомендації. Стандартизація та уніфікація цих неоднорідних даних становить значний виклик для ефективної аналітики та процесів рекрутингу[9].

Дані з різних джерел – соціальних мереж, професійних порталів, корпоративних баз даних і самих кандидатів - використовуються для побудови комплексних профілів працівників, які дозволяють нам точно відбирати та оцінювати потенційних співробітників.

Якість даних також відіграє важливу роль у рекрутингових системах. Неякісні дані для персоналу часто містять неповну, застарілу або нечітку інформацію, яка потребує аналізу за допомогою складних алгоритмів очищення та нормалізації, їх аналіз може значно покращити системи підбору персоналу, відбираючи кандидатів, які найкраще відповідають потребам організації[10]. Таким чином, щоб виділити лише якісні дані потрібно відкинути:

1) Застарілу інформацію, яка давно не оновлювалась.

2) Неповну інформацію, яка не є достатньою для прийняття рішення (наприклад відсутні дані про освіту та попереднє місце праці).
3) Інформацію, яка не відповідає критеріям підбору (резюме лікарів при пошуку програмістів).

Після такого очищення аналізувати дані буде значно простіше.

Етичні міркування, пов'язані зі збором та обробкою персональних даних, також відіграють вирішальну роль. Оскільки все більше уваги приділяється конфіденційності та захисту даних для того, щоб організації дотримувалися правових та етичних вимог при зборі та обробці персональної інформації про кандидатів[11]. Це не лише забезпечує захист інформації про кандидатів, але й зміцнює довіру між кандидатами та організаціями в їхніх системах підбору персоналу.

Таким чином можна значно покращити процес і результати аналізу даних, якщо їх попередньо опрацювати. При цьому через великі обсяги даних методи обробки повинні бути досить простими і швидкими, але і результативними.

1.2. Аналіз методів обробки великих даних

Методи обробки великих даних часто містять різноманітні техніки, що дозволяють зменшити розмір даних, підвищити якість інформації та вилучити корисні зразки для аналізу. Використання цих методів дає позитивний результат для точності та швидкості подальшого аналізу, проте їх ефективність залежить від самих даних, оскільки для інформації максимальної якості попередня обробка практично не змінить кінцевого результату та не пришвидшить процес аналізу[1].

1.2.1. Метод вибірки даних

Вибірка даних – це практика відбору репрезентативної вибірки з усієї сукупності з метою проведення аналізу та отримання висновків без необхідності попередньої обробки всієї сукупності.

Важливим аспектом відбору даних є забезпечення того, щоб вибірка точно представляла генеральну сукупність, щоб аналіз або висновки, зроблені на основі аналізу, не були упередженими або неправильними[12]. Для великих обсягів даних обсяг інформації мінімізує ймовірність вибрати в більшості неправильні дані, тому така вибірка описується формулою 1.1:

$$P = \frac{n}{N}, \tag{1.1}$$

де P – ймовірність вибору елементу, N – загальна кількість елементів у наборі даних, n – розмір вибірки, яку ми хочемо отримати.

За допомогою такої вибірки можна оцінити загальну якість даних у всьому наборі, порівнюючі середні показники вибірки з бажаними показниками за формулою 1.2:

$$Q = \frac{(\overline{x})}{T}, \tag{1.2}$$

де Q — якість даних, $\overline{x}$ — середнє значення даних вибірки, T — цільове бажане значення.

Таким чином при отриманні значення Q наближеного до 1 можна зробити висновок про хорошу якість вибірки і як наслідок усього набору даних.

Найпоширенішими методами вибірки даних є[13]:

Випадкова вибірка:

- Випадковий вибір підмножини даних з усього набору даних.
- Кожна точка даних має рівні шанси бути обраною.

Розшарована вибірка:

- Поділ даних на різні шари або групи на основі певної характеристики, а потім випадкова вибірка з кожної групи.
- Забезпечує репрезентативність кожної групи.

Систематична вибірка:

- Вибір кожної k-ї точки даних із відсортованого набору даних.
- Початкова точка зазвичай вибирається випадково.

Кластерна вибірка:

- Поділ набору даних на кластери, а потім випадковий вибір цілих кластерів.
- Всі точки даних у вибраних кластерах включаються до вибірки.

Зручна вибірка:

- Вибір точок даних, до яких найлегше отримати доступ.
- Не така репрезентативна, але корисна для попереднього аналізу.

Вибірка снігової кулі:

- Використовується переважно в соціальних науках, де існуючі суб'єкти дослідження набирають майбутніх суб'єктів серед своїх знайомих.
- Корисна для певних груп населення.

Квотна вибірка:

- Забезпечення того, щоб вибірка представляла певні характеристики пропорційно до їхньої присутності в популяції.
- Подібна до стратифікованої вибірки, але невипадкова.

Вибірка на основі суджень (цілеспрямована вибірка):

- Відбір точок даних на основі суджень дослідника.
- Корисний, коли дослідник орієнтується на конкретні критерії.

Перевибірка:

- Навмисна вибірка більшої кількості точок даних з класів для усунення дисбалансу.
- Часто використовується в машинному навчанні для покращення продуктивності моделі.

Недовибірка:

- Зменшення кількості точок даних з одиночних класів для усунення дисбалансу.
- Збалансовує набір даних, роблячи розподіл класів більш рівномірним.

Повторювальна вибірка:

- Створення декількох вибірок шляхом випадкової вибірки із заміною.
- Корисно для оцінки вибіркового розподілу статистики.

Адаптивна вибірка:

- Модифікація процесу вибірки на основі проміжних результатів.
- Часто використовується в клінічних випробуваннях та адаптивних опитуваннях.

Методи вибірки даних пропонують значні переваги швидкодії та застосованості, що робить їх важливими інструментами в аналізі даних. Однак вони також пов'язані з ризиками та проблемами, зокрема, упередженістю, помилкою вибірки та репрезентативністю[14]. Метод вибірки повинен бути ретельно продуманий для конкретних потреб аналізу та характеристик даних. Належним чином розроблені та впроваджені стратегії вибірки можуть надати очікувану інформацію, зменшуючи ймовірність отримання упереджених даних[15].

1.2.2. Метод трансформації даних

Трансформація даних означає зміну формату, структури або значень даних з метою підготовки їх до аналізу. Це передбачає використання спеціального трансформаційного фільтру та може включати нормалізацію шкал, перетворення категоріальних даних у числові або роботу з відсутніми значеннями[16] - важливі кроки в обробці даних, оскільки вони допомагають зменшити складність і підвищити точність подальших аналітичних процедур (Формула 1.3):

$$Y = F(X), \tag{1.3}$$

де: Y —дані після трансформації, F — функція фільтру, яка застосовується до даних, X — вхідні дані перед трансформацією.

Виділяють наступні види трансформації даних (Таблиця 1.1.):

Таблиця 1.1. Види трансформації даних

Вид трансформації	Опис	Варіант використання
Нормалізація[17]	Масштабування даних відповідно до певного діапазону, зазвичай [0, 1]	У випадках функцій з різними масштабами
Стандартизація[18]	Масштабування даних для отримання середнього значення 0 і стандартного відхилення 1	Випадки даних різних масштабів з необхідністю нормалізації
Логарифмічне перетворення[19]	Застосування натурального логарифма до кожної точки даних	Для зменшення асиметрії та обробки експоненціальних моделей зростання
Перетворення степеня[20]	Застосування степеневої функції для стабілізації дисперсії та більш нормального розподілу даних	Для дисперсії випадкової величини
Одночасне кодування[21]	Перетворення категоріальних змінних у серію двійкових стовпчиків	Для обробки категоріальних даних у моделях машинного навчання
Кодування міток[22]	Перетворення категоріальних міток у цілочисельні значення	Для використання з порядковими категоріальними даними

Продовження Таблиці 1.1

Вид трансформації	Опис	Варіант використання
Біннінг[23]	Перетворення неперервних змінних у дискретні біни	Для обробки викидів та спрощення моделей
Функції полінома[24]	Генерація полінома та функцій взаємодії	Для виявлення нелінійних взаємозв'язків даних
Масштабування функцій[25]	Налаштування масштабу функцій для покращення продуктивності модел	Для обробки викидів
Аналіз головних компонент (PCA)[26]	Зменшення розмірності даних шляхом проектування їх на головні компоненти.	Для зменшення складності та мультиколінеарності

Метод трансформації необхідний до застосування, адже неоднорідні дані отримані з різних джерел повинні бути максимально уподібнені для застосування інших методів обробки з оптимальним результатом (Рис. 1.1).

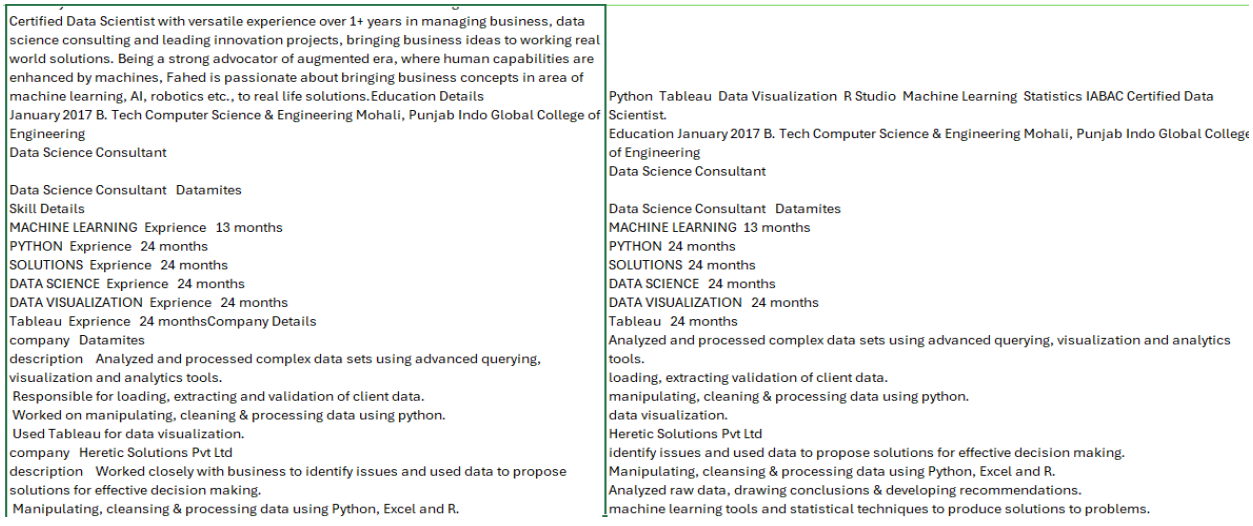


Рис. 1.1. Дані до (зліва) та після (справа) трансформації

Як видно з Рис. 1.1. трансформація допомагає забрати зайві слова (сполучники, займенники, прикметники, неважливі дані для подальшого опрацювання, тощо).

1.2.3. Метод усунення шуму

Зменшення шуму – це виявлення та видалення нерелевантних або помилкових даних, які спотворюють результати аналізу і призводять до неточних висновків (Рис. 1.2). Аналіз великих даних використовує шум для подальшого дослідження, який, у свою чергу, може спотворити результати аналізу і призвести до неправильних висновків з аналізу[27]. Методи зменшення шуму включають фільтрацію, виявлення статистичних аномалій, методи очищення даних, наприклад використання медіанного фільтру (Формула 1.4):

$$x_i' = median\left(x_{\{i-n\}}, \dots, x_i, \dots, x_{\{i+n\}}\right), \quad (1.4)$$

де x_i' - нове значення елемента після фільтрації, $x_{\{i-n\}}$ до $x_{\{i+n\}}$ - набір значень навколо елемента x_i.

Можна виділити наступні методи для усунення шуму (Таблиця 1.2.):

Таблиця 1.2. Методи усунення шуму

Методи усунення шуму	Опис	Варіант використання
Згладжування	Методи, які зменшують шум шляхом усереднення точок даних з їхніми сусідами	Дані часових рядів
Фільтрація	Використовує математичні фільтри для видалення шуму з даних	Обробка сигналів, обробка зображень

Продовження Таблиці 1.2

Методи усунення шуму	Опис	Варіант використання
Вейвлет-перетворення	Розкладає дані на різні частотні складові та встановлює порогові значення коефіцієнтів для зменшення шум	Обробка сигналів, згладжування зображень
Аналіз головних компонент	Зменшує шум шляхом перетворення даних у простір нижчої розмірності та реконструкції з використанням лише найбільш значущих компонентів	Багатовимірні дані, обробка зображень
Медіанна фільтрація	Замінює кожну точку даних медіаною її сусідів	Обробка зображень, часових рядів даних
Гауссова фільтрація	Застосовує ядро Гауса для згладжування даних, зменшуючи шум. Варіант використання: обробка зображень	Обробка зображень
Фільтр Савицького-Голея	Застосовує поліноміальний згладжуючий фільтр до даних	Обробка сигналів, часових рядів даних

Продовження Таблиці 1.2

Методи усунення шуму	Опис	Варіант використання
Згладжування K-найближчих сусідів (KNN)	Усереднює значення k-найближчих сусідів для зменшення шуму	Обробка зображень, часових рядів
Байєсівська фільтрація	Використовує байєсівську ймовірність для оновлення оцінки стану системи по мірі надходження нових даних	Прогнозування часових рядів, робототехніка
Автокодери	Моделі нейронних мереж, які навчаються стискати дані в простір нижчої розмірності та реконструювати їх, ефективно видаляючи шум	Обробка зображень, шумозаглушення сигналів

Такі методи можуть застосовуватись для виділення шумів з даних, або відкидання документів з надмірним рівнем шумів (Рис. 1.2.)

Education Details
MCA YMCAUST, Faridabad, Haryana
Data Science internship

Skill Details
Data Structure Exprience Less than 1 months
C Exprience Less than 1 months
Data Analysis Exprience Less than 1 months
Python Exprience Less than 1 months
Core Java Exprience Less than 1 months
Database Management Exprience Less than 1 months
company Itechpower

Рис. 1.2. Приклад зашумлених даних (досвід роботи < 1 місяця)

Як помітно з прикладу (Рис. 1.2), у кандидата дуже невеликий (практично відсутній) досвід роботи з потрібними навичками та технологіями, тому таке CV буде визначено як шум.

1.2.4. Методи вилучення ознак

Вилучення ознак – це перетворення даних у формат для аналізу та моделювання, який можна легко використовувати, часто шляхом вибору з них лише найбільш інформативних або релевантних ознак[6]. Виділення особливостей відіграє особливо важливу роль у машинному навчанні або предиктивній аналітиці, оскільки воно підвищує точність і ефективність моделей. Такий процес можна описати формулою 1.5:

$$Filtered\ Data\ = \{x\ \in Data \mid Condition(x)\}, \quad (1.5)$$

де, *Filtered Data* – відфільтровані вибрані дані потрібного формату, *x* – дані вхідного набору, *Condition(x)* – умова вибору даних та вилучення необхідних ознак з них.

1.3. Аналіз методів опрацювання неоднорідних даних

Для сфери рекрутингу притаманна велика кількість неоднорідної інформації (Рис 1.3), яку потрібно опрацювати для подальшого використання.

Category	Resume	C
Web Designing	Technical Skills Web Technologies: Angular JS, HTML5, CSS3, SASS, Bootstrap, Jquery, Javascript. Sof	
Web Designing	Education Details January 2016 B.Sc. Information Technology Mumbai, Maharashtra University of Mu	
Web Designing	IT SKILLS Languages: C (Basic), JAVA (Basic) Web Technologies: HTML5, CSS3, Bootstrap, JavaScript, j	
Web Designing	Technical Skills Web Technologies: Angular JS, HTML5, CSS3, SASS, Bootstrap, Jquery, Javascript. Sof	
Web Designing	Education Details B.C.A Bachelor Computer Application Pune, Maharashtra Pune University H.S.C.	
Web Designing	Technical Skills Web Technologies: Angular JS, HTML5, CSS3, SASS, Bootstrap, Jquery, Javascript. Sof	
Web Designing	Education Details January 2016 B.Sc. Information Technology Mumbai, Maharashtra University of Mu	
Web Designing	IT SKILLS Languages: C (Basic), JAVA (Basic) Web Technologies: HTML5, CSS3, Bootstrap, JavaScript, j	
Mechanical Engineer	Education Details May 1999 to September 2002 Diploma Mechanical Engg Mumbai, Maharashtra Inst	
Mechanical Engineer	SKILLS: â¢ Knowledge of software / computer: Auto CAD (Included Diploma Academic Syllabus) â¢ M	
Mechanical Engineer	Education Details January 2018 Bachelor's of Engineering Engineering Mumbai, Maharashtra MGM C	
Mechanical Engineer	Education Details June 2014 to June 2018 BE Mechanical Engineering Pune, Maharashtra Savitribai P	
Mechanical Engineer	* I'm hard working person. * I'm self confident and can mould myself to all work environments.Edu	
Mechanical Engineer	Education Details May 1999 to September 2002 Diploma Mechanical Engg Mumbai, Maharashtra Inst	
Mechanical Engineer	SKILLS: â¢ Knowledge of software / computer: Auto CAD (Included Diploma Academic Syllabus) â¢ M	
Mechanical Engineer	Education Details January 2018 Bachelor's of Engineering Engineering Mumbai, Maharashtra MGM C	
Mechanical Engineer	Education Details June 2014 to June 2018 BE Mechanical Engineering Pune, Maharashtra Savitribai P	
Mechanical Engineer	* I'm hard working person. * I'm self confident and can mould myself to all work environments.Edu	
Mechanical Engineer	Education Details May 1999 to September 2002 Diploma Mechanical Engg Mumbai, Maharashtra Inst	
Mechanical Engineer	SKILLS: â¢ Knowledge of software / computer: Auto CAD (Included Diploma Academic Syllabus) â¢ M	
Mechanical Engineer	Education Details January 2018 Bachelor's of Engineering Engineering Mumbai, Maharashtra MGM C	
Mechanical Engineer	Education Details June 2014 to June 2018 BE Mechanical Engineering Pune, Maharashtra Savitribai P	
Mechanical Engineer	* I'm hard working person. * I'm self confident and can mould myself to all work environments.Edu	

Рис. 1.3. Неоднорідні дані резюме

Як помітно з Рис. 1.3. дані з набору не мають чіткої структури і представляють собою хаотично сформовані стрічки з різнотиповою інформацією. Для подальшої роботи з такими даними їх необхідно попередньо опрацювати виділивши в них певні правила та залежності[28].

Для такої задачі можна використати наступні методи опрацювання неоднорідних даних:

1. Метод пошуку асоціативних правил передбачає аналіз взаємозв'язків між різними елементами, знайденими в гетерогенних наборах даних[29], за допомогою формул, які їх аналізують (Формула 1.6):

$$Support(X) = \frac{Transactions\ containing\ X}{Total\ Transactions}$$
$$Confidence(X \rightarrow Y) = \frac{Support(X \cup Y)}{Support(X)} \quad (1.6)$$
$$Lift(X \rightarrow Y) = \frac{Confidence(X \rightarrow Y)}{Support(Y)},$$

де X та Y - різні елементи або набори елементів даних.

2. Нормалізація - це процес приведення всього набору даних до єдиного стандарту або шкали та використання цієї формули (Формула 1.7):

$$Normalized\ Value\ = \frac{Original\ Value\ -\ Minimum\ Value}{Maximum\ Value\ -\ Minimum\ Value}, \quad (1.7)$$

Це дозволяє порівнювати дані, які були виміряні по-різному, наприклад оцінювання рівня знань чи вмінь кандидата за різними числовими шкалами.

3. Метод змішування даних - це процес об'єднання різних наборів даних в одну базу даних за допомогою алгоритмів злиття та очищення даних[30]. Не існує конкретної формули, якої слід дотримуватися при виконанні цього кроку, кожен проект відрізняється від інших.
4. Метод інтеграції даних передбачає створення єдиної бази даних з декількох джерел і форматів, але основною метою має бути створення єдиного комплексного ресурсу. З цією метою існує кілька методів об'єднання розрізнених наборів інформації в централізоване джерело, наприклад як у формулі 1.8.:

$$\text{Інтеграція} = \text{Об}'\text{єднання}(\text{Джерело } 1, \text{Джерело } 2, \dots, \text{Джерело } N) \quad (1.8)$$

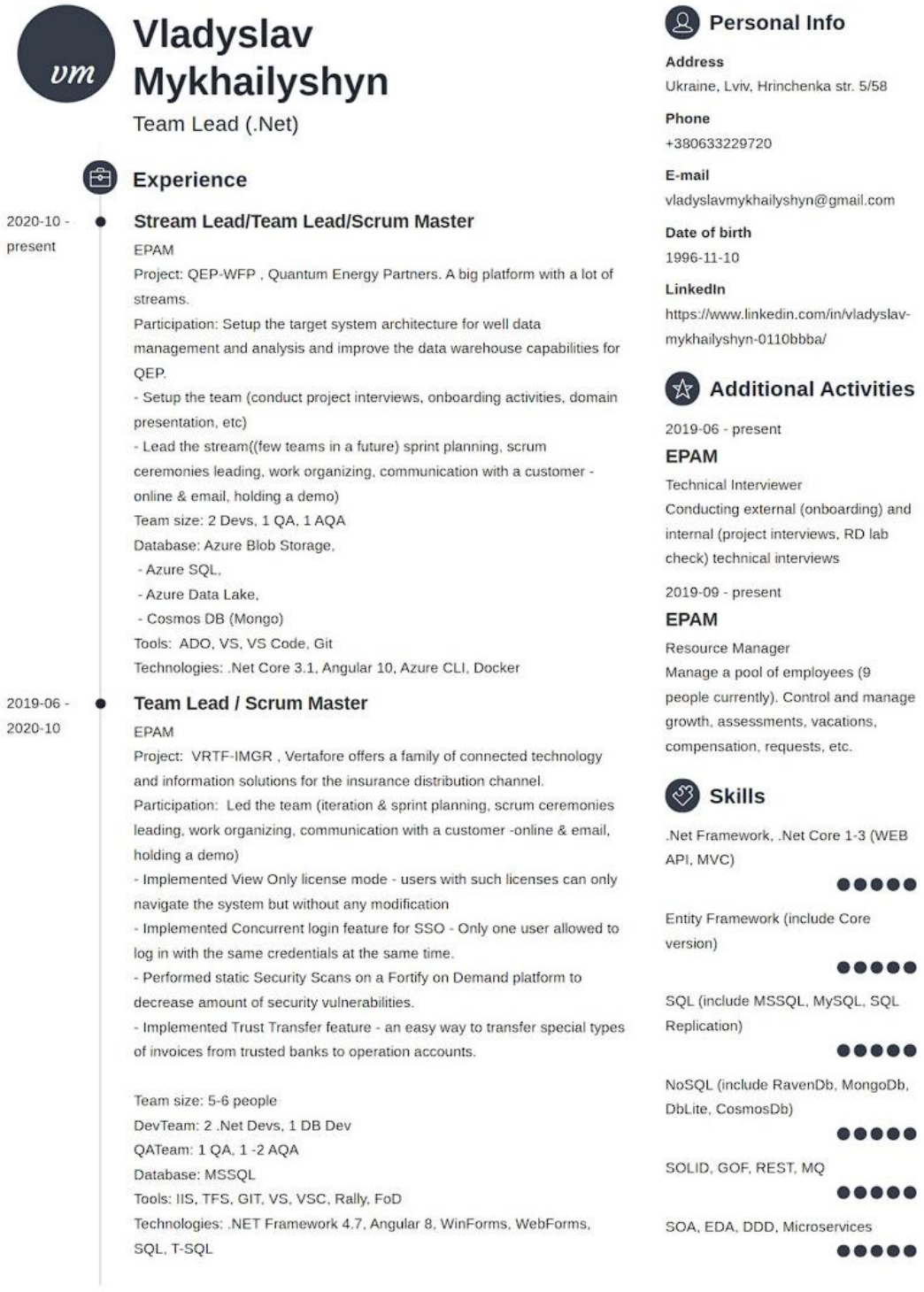
де кожне *Джерело* представляє окремий набір даних.

Кожен з цих методів відіграє важливу роль у забезпеченні якості та цілісності даних, що є необхідними елементами для ефективного аналізу рекрутингу та прийняття рішень.

1.4 Дослідження методів мовних обробників

Основним джерелом даних про кандидатів рекрутингової сфери є їх резюме. Такі дані в першу чергу потрібно розглядати як набір даних природньої мови, оскільки немає загальноприйнятого стандарту резюме, що дозволяє кандидатам

оформляти їх як завгодно (Рис. 1.4.). Це спонукає застосовувати для аналізу та опрацювання резюме методи мовних обробників.



Vladyslav Mykhailyshyn

Team Lead (.Net)

Experience

2020-10 - present

Stream Lead/Team Lead/Scrum Master

EPAM

Project: QEP-WFP , Quantum Energy Partners. A big platform with a lot of streams.

Participation: Setup the target system architecture for well data management and analysis and improve the data warehouse capabilities for QEP.

- Setup the team (conduct project interviews, onboarding activities, domain presentation, etc)
- Lead the stream((few teams in a future) sprint planning, scrum ceremonies leading, work organizing, communication with a customer - online & email, holding a demo)

Team size: 2 Devs, 1 QA, 1 AQA

Database: Azure Blob Storage,

- Azure SQL,
- Azure Data Lake,
- Cosmos DB (Mongo)

Tools: ADO, VS, VS Code, Git

Technologies: .Net Core 3.1, Angular 10, Azure CLI, Docker

2019-06 - 2020-10

Team Lead / Scrum Master

EPAM

Project: VRTF-IMGR , Vertafore offers a family of connected technology and information solutions for the insurance distribution channel.

Participation: Led the team (iteration & sprint planning, scrum ceremonies leading, work organizing, communication with a customer -online & email, holding a demo)

- Implemented View Only license mode - users with such licenses can only navigate the system but without any modification
- Implemented Concurrent login feature for SSO - Only one user allowed to log in with the same credentials at the same time.
- Performed static Security Scans on a Fortify on Demand platform to decrease amount of security vulnerabilities.
- Implemented Trust Transfer feature - an easy way to transfer special types of invoices from trusted banks to operation accounts.

Team size: 5-6 people

DevTeam: 2 .Net Devs, 1 DB Dev

QATeam: 1 QA, 1 -2 AQA

Database: MSSQL

Tools: IIS, TFS, GIT, VS, VSC, Rally, FoD

Technologies: .NET Framework 4.7, Angular 8, WinForms, WebForms, SQL, T-SQL

Personal Info

Address

Ukraine, Lviv, Hrinchenka str. 5/58

Phone

+380633229720

E-mail

vladyslavmykhailyshyn@gmail.com

Date of birth

1996-11-10

LinkedIn

https://www.linkedin.com/in/vladyslav-mykhailyshyn-0110bbba/

Additional Activities

2019-06 - present

EPAM

Technical Interviewer

Conducting external (onboarding) and internal (project interviews, RD lab check) technical interviews

2019-09 - present

EPAM

Resource Manager

Manage a pool of employees (9 people currently). Control and manage growth, assessments, vacations, compensation, requests, etc.

Skills

.Net Framework, .Net Core 1-3 (WEB API, MVC)

Entity Framework (include Core version)

SQL (include MSSQL, MySQL, SQL Replication)

NoSQL (include RavenDb, MongoDb, DbLite, CosmosDb)

SOLID, GOF, REST, MQ

SOA, EDA, DDD, Microservices

Рис. 1.4. Приклад резюме

Як помітно з Рис. 1.4, резюме складається з великої кількості різноформатних даних – списків, переліків, змістовних речень та абзаців прямої мови. Такий довільний формат даних в межах одного резюме робить неможливим виділення чіткої структури, тому такі дані найпростіше розглядати саме як природню мову.

1. Статистичний метод є одним з основних способів обробки тексту є статистичні методи для аналізу великих обсягів текстових даних і виявлення шаблонів і закономірностей у них, наприклад, шляхом обчислення ймовірності появи певних слів або фраз у контексті - наприклад, *n*-грамові моделі можуть допомогти передбачити майбутні слова на основі наявного контексту[31] (Формула 1.9) :

$$P(W_n|W_{n-1}) = \frac{Count(W_{n-1},W_n)}{Count(W_{n-1})}, \tag{1.9}$$

де $P(W_n|W_{n-1})$ - ймовірність слова W_n, знаючи попереднє слово W_{n-1}.

2. Лінгвістичні методи використовують розуміння мовної структури та граматики для аналізу текстів. Вони використовують синтаксичний і семантичний аналіз, де такі компоненти, як підмети, присудки та об'єкти, розбиваються на складові частини і аналізуються на предмет значення і взаємозв'язків. Один з таких підходів до аналізу використовує дерева залежностей, які відображають структурні зв'язки між словами:

$$S \rightarrow NP\ VP$$

$$VP \rightarrow V\ NP \mid V\ NP\ PP$$

$$NP \rightarrow Pronoun \mid Proper-Noun \mid \boldsymbol{Det}\ Nominal$$

$$Nominal \rightarrow Noun \mid Nominal\ Noun \mid Nominal\ PP$$

$$PP \rightarrow Preposition\ NP$$

3. Умовні випадкові поля (УВП) - це статистичні моделі, що використовуються для передбачення міток послідовностей. Вони моделюють умовну ймовірність вихідної послідовності міток за наявності вхідної послідовності спостережень. На відміну від прихованих марковських моделей (HMM), CRF враховують залежності

між вихідними мітками, що дозволяє досягти більш точних результатів у задачах, таких як обробка природної мови, розпізнавання образів та біоінформатика. CRF є потужним інструментом для побудови складних моделей, що враховують взаємозв'язки між даними.

УВП можна використовувати в задачах розпізнавання об'єктів, а також у будь-яких задачах, де контекст є важливим (Формула 1.10):

$$\frac{P(y|x) = \exp\left(sum\left(f_i(y_i - 1, y_i\ , x, i)\right)\right)}{Z(x)} \quad (1.10)$$

де $Z(x)$ - нормалізаційний фактор, а f_i - функції оцінки.

4. Токенізація - це процес розбиття тексту на окремі слова, фрази, символи або інші значущі елементи, які називаються токенами[32]. Це перший крок у багатьох завданнях обробки мови.

Можна виділити такі основні методи токенізації:

- Проста токенізація: Розбиття тексту на слова за допомогою пробілів та пунктуації.
- Токенізація за допомогою регулярних виразів: Використання регулярних виразів для більш гнучкого визначення токенів.
- Використання бібліотек NLP: Застосування спеціалізованих NLP інструментів для токенізації, таких як NLTK або spaCy[33].

Застосувавши токенізацію Рис. 1.6. можна отримати набір токенів, які в подальшому можуть бути перетворені в числові показники за допомогою статистичних методів, що дозволить кластеризувати такий набір даних для навчання та використання в СППР[34].

В подальшому ці токени можуть бути використані для опису основних характеристик тексту[35].

5. Приховані марковські моделі (HMM) - це статистичні моделі, які припускають, що будь-яку систему можна представити як марковський процес з невідомими параметрами, і намагаються виявити ці невідомі значення за

допомогою методів тестування та навчання. HMM стали популярними інструментами в програмах обробки мови, таких як розпізнавання мови та рукописного тексту. Така модель описується як (Формула 1.11):

$$P(Q|O) = P(O|Q) * \frac{P(Q)}{P(O)}, \quad (1.11)$$

де Q - послідовність станів, O - послідовність спостережень.

1.4. Вибір та оптимізація методів мовних обробників для даних рекрутингової діяльності

Для вибору методів опрацювання отриманої інформації потрібно проаналізувати тип цієї інформації (Рис. 1.5.)

Рис. 1.5. Вхідні дані рекрутингової діяльності

Як можна помітити – текстові дані являють собою не зв’язний однорідний текст, а швидше набір тез та назв технологій, закладів освіти, сертифікатів, тощо.

При виборі методів оптимізації мовних обробників можна одразу відкинути лінгвістичний метод, оскільки він підходить для зв’язного тексту, а дані кандидатів в основному окремі слова, або короткі фрази. З тих самих міркувань не підійде метод умовних випадкових полів. Прихована марковська модель більше підходить для розпізнавання тексту, тому теж не зможе допомогти опрацювати отримані дані.

Ефективним буде застосування токенізації, за допомогою якої, можна буде виділити основні технології, навички та програми якими володіє кандидат (Рис. 1.6.)

Рис 1.6. Токенізований набір даних з резюме

Як помітно з Рис. 1.6 внаслідок токенізації набору резюме було виділено набір токенів з яких воно складається. Однак також помітно що велика кількість токенів є не інформативними (розділові знаки). Такий набір даних потребує подальшого опрацювання перед застосуванням.

Наступним етапом обробки є визначення токенів, які найчастіше статистично зустрічаються разом (пов’язані навички, наприклад Java і SQL, оскільки Java розробники в більшості також працюють і з базами даних). Для цього потрібно вилучити усі беззмістовні токени (розділові знаки, сполучники, загальні слова, які часто зустрічаються в резюме (Рис.1.7)), після чого провести статистичний аналіз тексту (Рис 1.8.)

Визначимо текст (T) як послідовність символів. Токенізація тексту (T) може бути описана як функція (f), що перетворює (T) на множину токенів (S) (Формула 1.12):

$$f(T) = S = \{t_1, t_2, \dots, t_{n\}}, \tag{1.12}$$

де кожен (t_i) є послідовністю символів, відділеною від інших послідовностей пробілами, пунктуацією або іншими границями.

Після цього потрібно видалити зайві токени Рис.1.7. і розділові знаки, спецсимволи тощо.

Нехай (S) є множиною токенів. Ми визначаємо предикатну функцію ($g(t)$), що повертає `true`, якщо токен (t) має бути збережений (Формула 1.13):

$$g(t) = \begin{cases} true & t \notin \text{непотрібні токени} \wedge t \notin \text{спецсимволи} \\ false & \text{інакше} \end{cases} \tag{1.13}$$

Тоді множина фільтрованих токенів (S') буде:

$$S' = \{t \in S \mid g(t) = true\}.$$

Наступним кроком є визначення частоти зустрічання токенів, щоб вибрати найбільш значущі.

Нехай (S') є множиною фільтрованих токенів. Предикат ($h(t,k)$) повертає `true`, якщо частота токена (t) більша або дорівнює (k), де (k) є пороговим значенням (Формула 1.14):

$$h(t,k) = (freq(t) \geq k), \tag{1.14}$$

де ($freq(t)$) визначає частоту токена (t) в (S'). Множина токенів, які найчастіше зустрічаються (M) має вигляд (Формула 1.15):

$$M = \{t \in S' \mid h(t,k) = true\}, \quad (1.15)$$

де (k) вибирається так, щоб відображати значущу частоту зустрічання токенів.

```
trash = ["Experience", "Project", "Name", "Duration", "Base", "months",
# Plotting
```

Рис. 1.7. Список непотрібних токенів

На Рис. 1.7. представлено список непотрібних токенів – “trash”, який використовується для вилучення з результатів токенізації цих токенів, які також є присутніми в цьому списку.

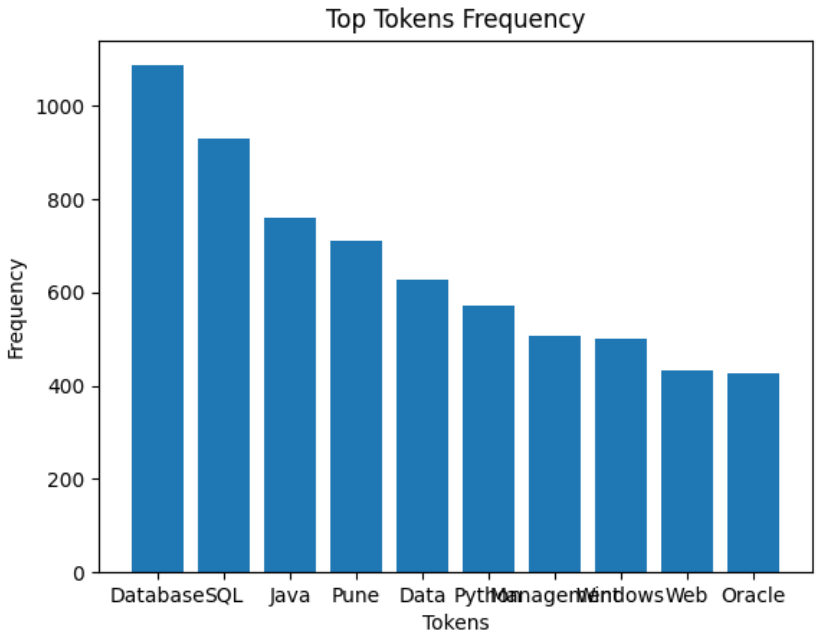


Рис. 1.8. Результати статистичного аналізу

Результати статистичниго аналізу (Рис. 1.8.) набору даних представлених на Рис. 1.5. дозволяють побачити частоту токенів в кожному резюме, але все ще недостатньо точні. Оскільки загальні токени можуть бути схожими в окремих професіях, тому цифрове значення статичстичного показника для цього документу буде наближене до не відповідних йому. Наприклад Java розробник може мати багато схожих токенів опису резюме до SQL розробника, оскільки обидві професії передбачають роботу з базами даних, хоча по своїй сутті вони досить відмінні.

Ці результати можна покращити за допомогою підходу TF-IDF.

Термін "частота, обернена до частоти документа" (TF-IDF) – це статистичний показник, що використовується для оцінки важливості слова в документі, який є частиною речення. Концепція TF-IDF полягає в тому, що важливість слова зростає пропорційно до кількості входжень слова в документі, але компенсується частотою слова в корпусі.

Основні складові методу:

- Частота терміну (TF): Показує, як часто термін зустрічається в документі. Обчислюється як кількість разів, коли термін t з'являється в документі d, поділена на загальну кількість термінів у цьому документі (Формула 1.16):

$$TF(t,d) = \frac{n_{t,d}}{\sum k\left(n_{k,d}\right)}, \tag{1.16}$$

де $n_{t,d}$ є кількістю разів, коли термін t з'являється в документі d, а знаменник – сума кількостей всіх термінів у документі d.

- Обернена частота документа (IDF): Вимірює важливість терміна. При обчисленні TF всі терміни вважаються однаково важливими[36]. Однак деякі терміни, такі як "is", "of" і "that", можуть з'являтися багато разів, але мати невелику важливість. Тому потрібно зменшити вагу поширених термінів і збільшити вагу тих, які не часто зустрічаються, розрахувавши наступний показник.

Ця міра часто використовується в інформаційному пошуку та текстовому аналізі як ваговий коефіцієнт під час аналізу тексту, а також є ключовим

компонентом в алгоритмах машинного навчання для кластеризації текстів[37] (Формула 1.17).

$$IDF(t, D) = \log\left(\frac{|D|}{1 + |\{d \in D: t \in d\}|}\right), \quad (1.17)$$

де:

- $|D|$ є загальною кількістю документів у корпусі D,
- $|\{ d \in D : t \in d \}|$ є кількістю документів, де термін t з'являється (не підраховуючи документи, де термін не з'являється). Додавання 1 у знаменнику забезпечує уникнення ділення на нуль.

Нарешті, щоб отримати оцінку для терміна в документі, потрібно перемножити оцінки TF та IDF[38]. Формула 1.18 для TF-IDF має вигляд:

$$TF - IDF(t, d, D) = TF(t, d) \times IDF(t, D), \quad (1.18)$$

Обрахувавши такий статистичний показник дозволить в майбутньому без проблем кластеризувати належність кожної групи токенів (Рис. 1.9.).

Рис. 1.9. Результат роботи TF-IDF

Такий показник для кожного токену дозволить зробити точний вистовок про важливість токену в загальному наборі інформації, що значно спростить роботу з вибору обмеженої кількості найбільш актуальних токенів[39].

Після опрацювання даних природньої мови таким методом, отримані результати можна використати для їх подальшої кластеризації.

1.6. Висновки

В цьому розділі було проаналізовано особливості даних рекрутингової діяльності та їх можливого опрацювання.

Було розглянуто ключові методи, які використовуються для обробки неоднорідних та великих даних у рекрутинговій діяльності. Завдяки цим методам можна позбутись зайвих та недостовірних даних, а також пришвидшити подальшу роботу з вибіркою за рахунок оптимізації та уподібнення даних. Проведено аналіз методів опрацювання природньої мови таких як статистичний, лінгвістичний методи, метод умовних випадкових полів, токенізація та прихована марковська модель.

Для окремих методів було продемонстровано результати роботи на тестовому наборі даних.

Розглянуто варіант покращення результатів аналізу за рахунок врахування важливості кожного окремого токену за допомогою методу TF-IDF.

РОЗДІЛ 2. РОЗРОБКА АЛГОРИТМУ ОБРОБКИ ДАНИХ З ВИКОРИСТАННЯМ МЕТОДІВ АНАЛІЗУ ПРИРОДНОЇ МОВИ ТА КЛАСТЕРИЗАЦІЇ

У розділі розроблено метод опрацювання даних з використанням методів токенізації, статистичного аналізу та TF-IDF. Проведено аналіз методів кластеризації даних та розроблено алгоритм обробки даних з використанням методів аналізу природної мови та кластеризації з використанням паралельних обчислень.

Результати розділу опубліковано у працях автора [4,5]

2.1. Розроблення методу опрацювання природної мови

Вхідні дані проектованої системи являють собою неоднорідну та неструктуровану інформацію. Перед тим як їх використовувати, дані повинні бути опрацьовані та стандартизовані з використанням методів розглянутих у Розділ 1.

Спочатку потрібно провести етап підготовки даних, на якому необхідно виділити токени у резюме та очистити їх від зайвих значень (розділові знаки, символи, сполучники, тощо)

Наступним етапом буде здійснення аналізу даних з використанням статистичних методів таких як статистичний аналіз та TF-IDF.

Таким чином загальний процес опрацювання тексту і перетворення його в стандартизовані дані для подальшого опрацювання можна подати схематично наступним чином Рис. 2.1.

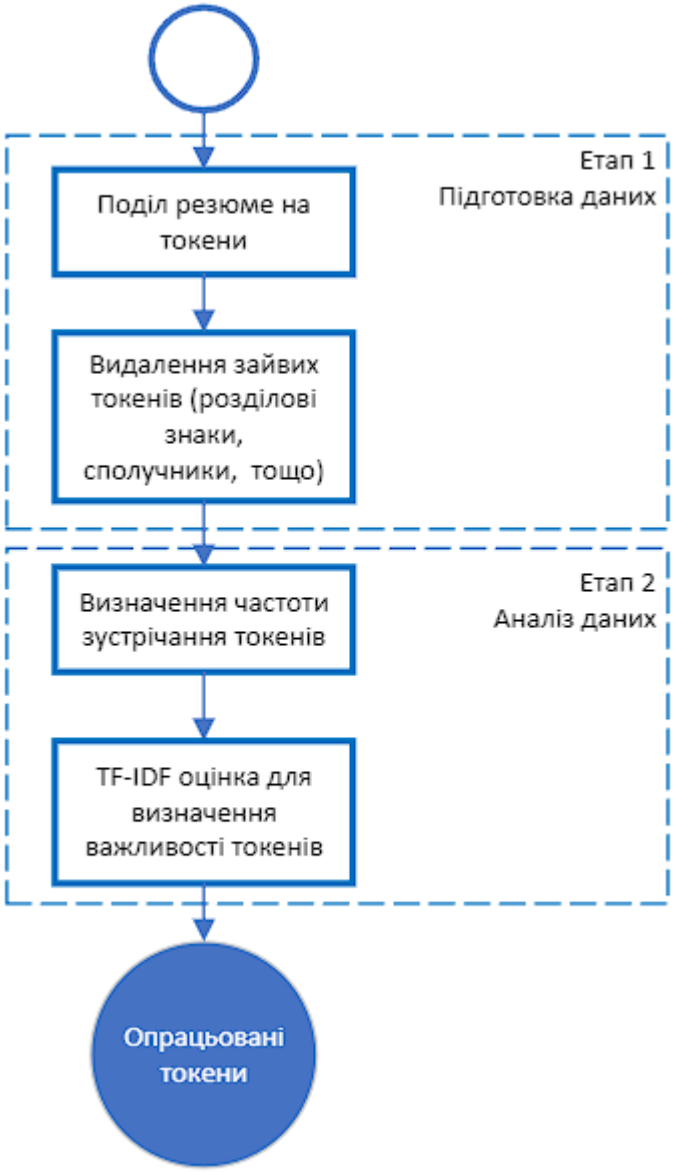


Рис. 2.1. Схема методу опрацювання природньої мови

Отже, загальний процес опрацювання даних природної мови (резюме, профілі в соціальних мережах, тощо) можна розбити на 4 основні етапи (Рис. 1.10.):

1) Поділ резюме на токени – відбувається поділ загального набору тексту на окремі текстові та числові одиниці інформації (Рис. 1.6.)
2) Видалення зайвих токенів – вилучаються зайві токени, які не мають змістового навантаження (розділові знаки, сполучники, займенники), або знаходяться в чорному списку (Рис. 1.7.)

3) Визначення частоти зустрічання токенів – статистичний аналіз для оцінки частоти зустрічання токенів (Рис. 1.8.)
4) Використання TF-IDF методу для визначення важливості токена серед загального набору токенів (Рис. 1.9.)

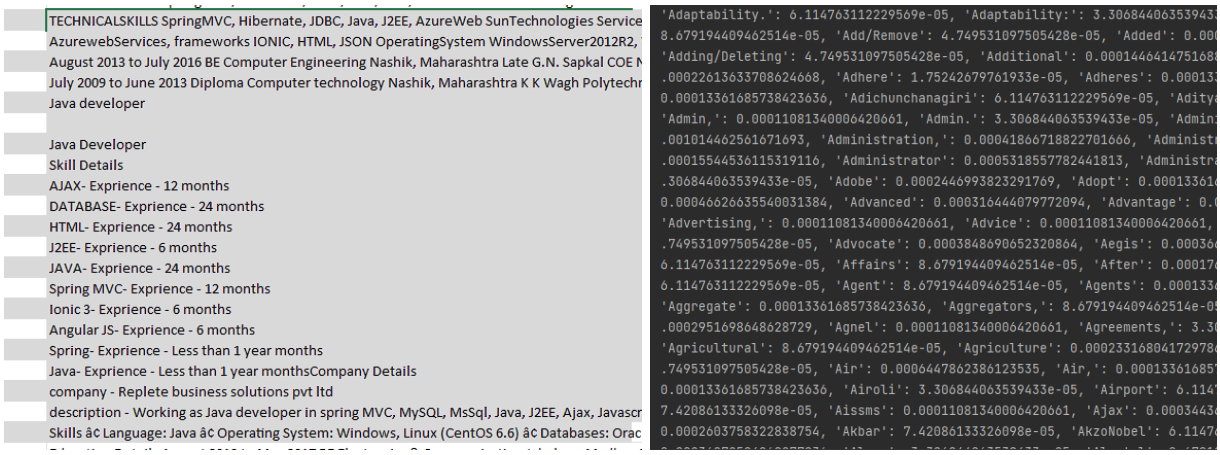


Рис. 2.2. Вхідна інформація (зліва) та вихідні оцінені токени (справа)

В результаті роботи методу (Рис. 2.2.) для кожного резюме буде виділено набір оцінених токенів для подальшого використання у системі.

2.1. Аналіз існуючих методів кластеризації

Другим кроком опрацювання даних буде їх поділ на кластери для виділення груп резюме, які мають найбільшу кількість схожих токенів, що дозволить виділити аномалії та резюме, які складно категоризувати та оцінити[5].

Для цього потрібно проаналізувати різні методи кластеризації, кожен з яких має унікальні характеристики та застосування в обробці великих обсягів даних. Перевіряється їх можливе застосування для аналізу текстових даних на поточній вибірці.

При кластеризації слів або документів, текстові дані спочатку перетворюються в числові форми, такі як вектори TF-IDF, які представляють кожне слово або документ окремо (Рис.2.3.)

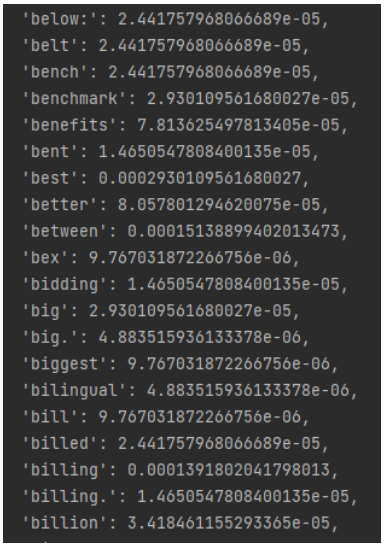


Рис. 2.3. Трансформація слів набору даних в числові дані

Таким чином відбувається визначення ваги кожного окремого токену в результаті роботи методів опрацювання природньої мови (Рис. 2.3).

Потім кластеризація за обраним методом застосовує алгоритм, групуючи кожен вектор у відповідний кластер на основі того, наскільки близьким до нього він є. Таким чином, кожне кластеризоване слово або документ має певний ступінь приналежності до кластера, що вказує на те, наскільки близько або далеко він знаходиться від іншого кластера[4].

2.1.1 Метод K-середніх

K-середніх кластеризує дані, намагаючись мінімізувати варіації в межах кластерів та максимізувати відстань між ними[40]. Основними кроками є:

1. Ініціація: Спочатку визначається число *K* — кількість кластерів, які потрібно сформувати. Центри кластерів (центроїди) вибираються випадково серед спостережень або методом більш складної ініціації[401].

2. Призначення до кластерів: Кожен об'єкт у наборі даних призначається до кластера, центроїд якого знаходиться найближче до нього. Відстань зазвичай вимірюється за допомогою евклідової метрики.
3. Перерахунок центроїдів: Центроїд кожного кластера перераховується як середнє значення всіх точок, що були призначені до цього кластера.
4. Ітерація: Кроки 2 і 3 повторюються до тих пір, поки призначення точок до кластерів не перестануть змінюватися або до досягнення заданого критерію зупинки, як-от задана кількість ітерацій або мінімальне зміщення центроїдів.

Відповідно, процес кластеризації можна описати наступним чином[42]:

Дано набір точок даних $\{x_1, x_2, \dots, x_n\}$

1. Ініціація. Вибрати *K* початкових центроїдів кластерів $(\mu_1, \mu_2, \dots, \mu_K)$
2. Призначення до кластерів. Для кожної точки даних x_i привласнюємо її найближчому центроїду (Формула 2.1.):

$$c_i = \arg\min_k \|x_i - \mu_k\|^2, \quad (2.1.)$$

де c_i - це призначення кластера для точки даних x_i;

μ_k - центроїд кластера *k*.

3. Перерахунок центроїдів. Оновлення центроїдів на основі призначених точок (Формула 2.2.)[93]:

$$\mu_k = \frac{1}{|C_k|}\sum_{x_i \in C_k} x_i, \quad (2.2.)$$

де μ_k – новий центроїд кластеру *k;*

C_k – набір точок присвоєних до кластеру *k;*

$|C_k|$ - кількість точок в кластері *k*.

4. Ітерація. Повторювання кроків 2 і 3 до збіжності, тобто до моменту, коли розподіл кластерів більше не змінюватиметься.

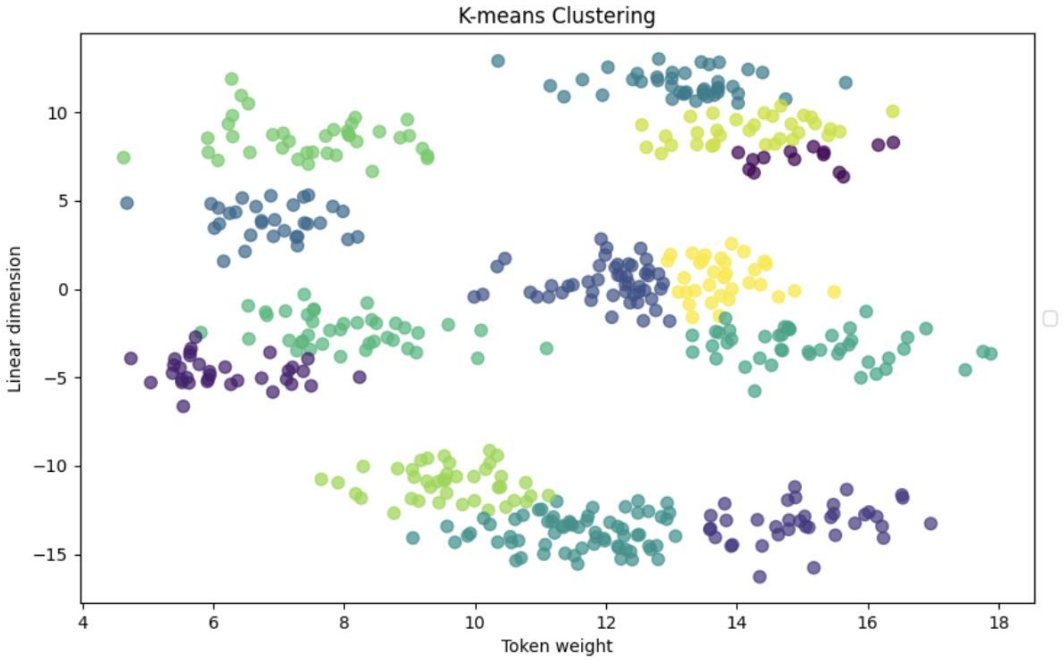


Рис. 2.4. Метод К-середніх для тексту

На Рис. 2.4. зображено кластеризацію методом К-середніх набору даних розміром 500 елементів. Візуально результати показують виділення чітких кластерів, індекс якості кластеризації буде визначено пізніше при виборі методу.

2.1.2 Метод К-медіани

Метод К-медіан – це варіант алгоритму кластеризації, схожий на більш відомий алгоритм К-середніх. Він використовується для групування даних у кластери і є особливо корисним, коли середнє значення схильне до викидів або коли розподіл даних не є нормальним[43].

Основна відмінність між К-середніх і К-середніх полягає у виборі центральної метрики тенденції, яка представляє кластери: К-медіан використовує медіанне значення замість середнього.

Як працює К-медіан:

1) Ініціалізація: спочатку з набору даних вибирається випадкова точка як початкова медіана кластера.

2) Віднесення точок до кластерів: кожна точка даних відноситься до кластера з найближчою медіаною, використовуючи обрану метрику відстані (зазвичай Манхетенську відстань).
3) Оновлення медіани: після того, як точки були віднесені до кластера, обчислюється нова медіана. Для кожного кластера новою медіаною є точка, яка мінімізує суму відстаней до всіх інших точок кластера.
4) Повторення: повторення кроків 2 і 3 до тих пір, поки медіана не перестане змінюватися, що вказує на стабілізацію кластера.

Переваги K-медіан[44]:

- Толерантність до викидів. Оскільки медіана менш чутлива до викидів, ніж середнє значення, K-медіан є більш ефективним, коли в даних є шум або викиди.
- Корисність для нечислових даних: для даних, які можна впорядкувати, але не виміряти чисельно (наприклад, ранжовані дані), K-медіан може бути більш доречним.

Недоліки K-медіан[45]

- Обчислювальна складність: обчислення медіани вимагає більше обчислювальних ресурсів, ніж обчислення середнього значення, особливо для великих наборів даних.
- Вибір початкової медіани: як і у випадку з K-середніми, результати кластеризації можуть залежати від початкового вибору медіани.

Відповідно, процес кластеризації можна описати наступним чином[46]:

Дано набір точок даних $\{x_1, x_2, \dots, x_n\}$

1. Ініціація. Вибрати *K* початкових центроїдів кластерів $(\mu_1, \mu_2, \dots, \mu_K)$.
2. Призначення до кластерів. Для кожної точки даних x_i привласнюємо її найближчому центроїду (Формула 2.3.):

$$c_i = \arg \min_k \|x_i - \mu_k\|^2, \qquad (2.3.)$$

де c_i - це призначення кластера для точки даних x_i;

μ_k - центроїд кластера k.

3. Перерахунок центроїдів. Оновлення центроїдів на основі призначених точок (Формула 2.4.):

$$\mu_k = \text{median}\{x_i : c_i = k\}, \quad (2.4.)$$

де μ_k – новий центроїд кластеру k;

median – медіана точок, призначених до кластеру k.

4. Ітерація. Повторювання кроків 2 і 3 до збіжності, тобто до моменту, коли розподіл кластерів більше не змінюватиметься.

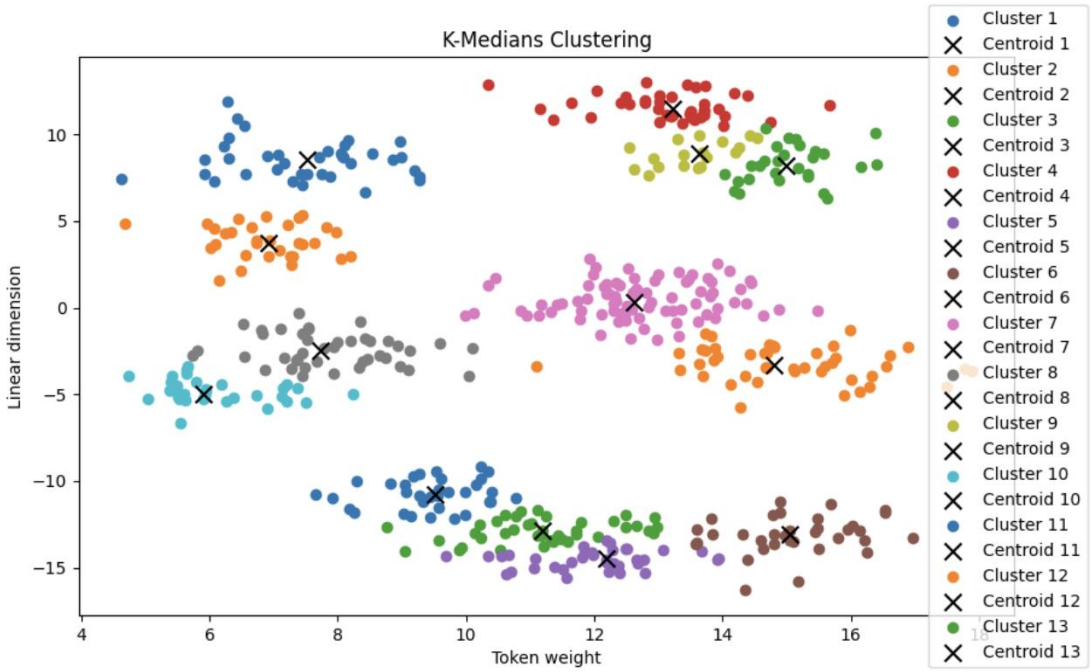


Рис. 2.5. Кластеризація методом К-медіан для тексту

На Рис. 2.5. зображено результати кластеризації того ж набору вхідного даних, який використовувася для демонстрації роботи методу К-середніх (Рис. 2.4.). Помітно, що метод К-медіан сформував кластери дещо інакше. Оцінка його роботи буде проведена пізніше.

2.1.3 DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) є одним з найпопулярніших алгоритмів кластеризації, який використовується у машинному навчанні та аналізі даних. DBSCAN відрізняється від інших алгоритмів кластеризації тим, що він може виявляти кластери різної форми і величини, і не вимагає від користувача заздалегідь вказувати кількість кластерів[47].

Основні характеристики DBSCAN:

- Щільність кластера визначається кількістю точок (або "сусідів") у певному радіусі ((ϵ)).
- Основні точки: Якщо точка має достатню кількість сусідів у радіусі (ϵ), вона вважається основною точкою.
- Межові точки: Якщо точка має менше сусідів, ніж потрібно, але є в радіусі дії від основної точки, вона вважається межовою.
- Шумові точки: Точки, які не є ні основними, ні межовими.

Процес кластеризації в DBSCAN[48]:

1. Визначення параметрів:
   - $\epsilon(eps)$: Максимальна відстань між двома точками, щоб одна з них вважалася сусідньою.
   - Мінімальна кількість точок (MinPts): Мінімальна кількість точок, які повинні знаходитися в радіусі (ϵ), щоб точка вважалася основною.

2. Пошук сусідів:

Для кожної точки в даних алгоритм перевіряє, скільки сусідів лежать в межах відстані (ϵ).

3. Визначення основних та межових точок:
   - Якщо точка має достатньо сусідів (≥ MinPts), вона стає основною.
   - Якщо точка має менше сусідів, ніж MinPts, але вона є сусідом основної точки, вона вважається межовою.
   - Всі інші точки, які не відповідають критеріям вище, вважаються шумом.

4. Формування кластерів:

Основні точки та їхні сусіди об'єднуються в кластери, при цьому межові точки можуть служити мостами, що з'єднують кластери, але вони не можуть ініціювати нові кластери.

Переваги DBSCAN:

- Відмінно підходить для даних із складною структурою і різною щільністю.
- Не потребує заздалегідь визначеної кількості кластерів.
- Здатен визначити і відокремити шум.

Недоліки DBSCAN:

- Вибір параметрів (є) і MinPts може бути складним; поганий вибір може значно вплинути на результати кластеризації.
- Неефективний для дуже великих наборів даних або даних із високою розмірністю без оптимізації.

DBSCAN є потужним інструментом для кластеризації, особливо коли структура даних є нерегулярною або коли присутній шум. Це робить його популярним вибором у багатьох областях, включаючи статистичний аналіз, обробку зображень і біоінформатику (Рис. 2.6.).

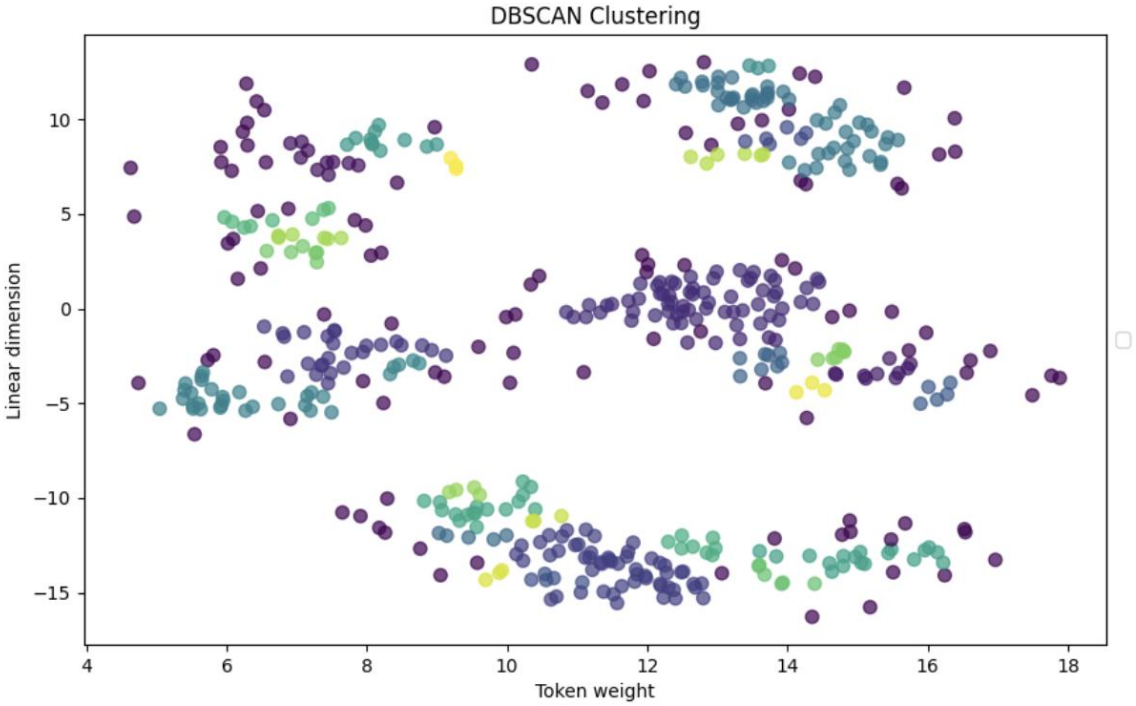


Рис. 2.6. DBSCAN кластеризація текстових даних

Як видно з Рис. 2.6, DBSCAN визначив кластери відмінно від двох попередніх методів, використовуючи той самий набір даних. Поки неможливо однозначно оцінити методи кластеризації.

2.1.5. C-means

Кластеризація за середнім арифметичним, також відома як нечітка кластеризація за середнім арифметичним. Це м'який метод кластеризації, в якому кожна точка даних частково належить до кількох кластерів з різним ступенем приналежності[49]. При застосуванні до текстових даних, таких як слова з документів або речення з документів, алгоритм C-середніх може допомогти згрупувати схожі слова на основі таких ознак, як оцінки TF-IDF.

Метою алгоритму C-середніх є мінімізація цільової функції, яка вимірює відстані між точками даних і центрами кластерів, зважені за ступенем

приналежності. Конкретно, цільова функція (J) може бути сформульована наступним чином (Формула 2.5).

$$J(U,V) = \sum_{i=1}^{n}\sum_{j=1}^{c} u_{ij}^{m}|x_i - v_j|^2, \tag{2.5}$$

де n є кількістю елементів;

c є кількістю кластерів;

x_i є i-м елементом даних;

v_j є центром j-го кластера;

u_{ij} є ступенем належності елемента x_i до кластера j;

m є параметром нечіткості, який контролює рівень нечіткості кластеризації; зазвичай m>1;

$\|x_i - v_j\|^2$ є евклідовою відстанню між x_i і v_j.

Для обрахування алгоритму потрібно пройти 4 кроки:

1) Ініціація: Вибрати кількість кластерів c і випадково ініціювати матрицю належностей U так, щоб сума належностей для кожного елемента до всіх кластерів дорівнювала 1.
2) Оновити центри кластерів на основі ступенів належності та даних, за допомогою Формули 2.6.

$$v_j = \frac{\sum_{i=1}^{n} u_{ij}^{m} x_i}{\sum_{i=1}^{n} u_{ij}^{m}} \tag{2.6}$$

3) Оновити матрицю залежності за формулою 2.7:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{|x_i - v_j|}{|x_i - v_k|}\right)^{\frac{2}{m-1}}} \tag{2.7}$$

4) Повторювати кроки 2 та 3 до тих пір, поки зміни в матриці належностей не стануть мінімальними або не буде досягнуто заданої кількості ітерацій.

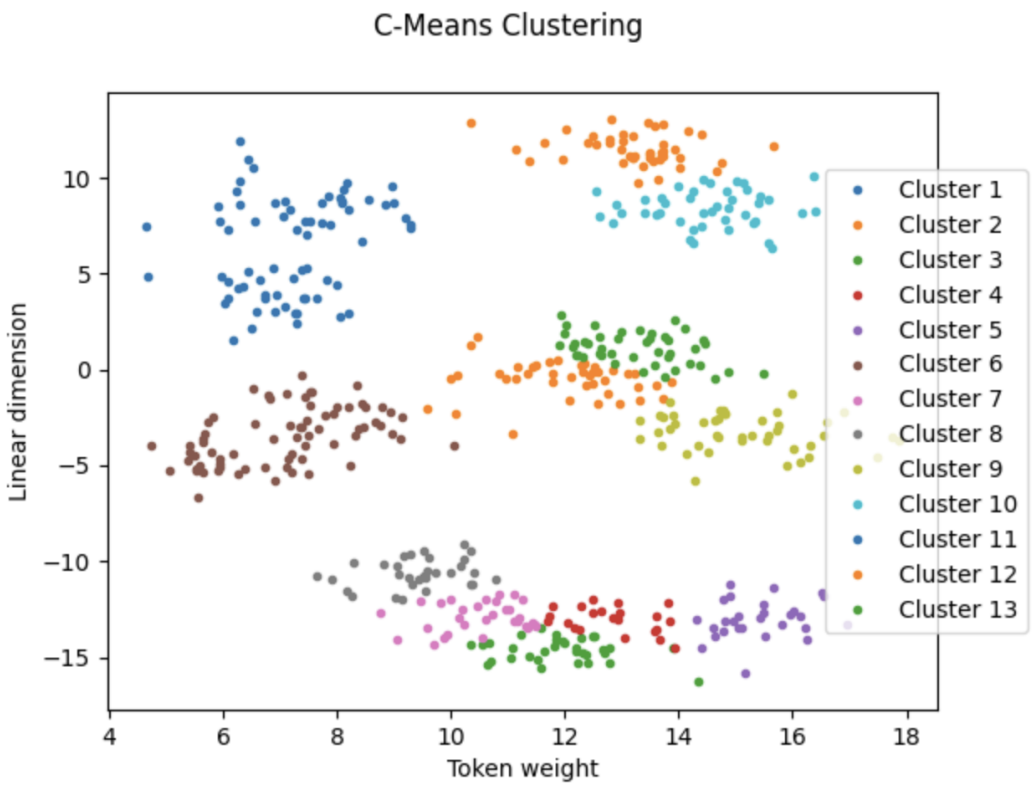


Рис. 2.7. Кластеризація С-середніх

Рис. 2.7. демонструє що метод С-середніх має певні відмінності у визначенні приналежності кластерів, але при цьому немає явних ознак того, що кластери визначені погано. Тому наступним завданням є вибір одного з описаних методів кластеризації.

2.2. Створення та оптимізація методу кластеризації вхідних даних

Проаналізувавши роботу різних методів кластеризації неможливо однозначно стверджувати, який з них працюватиме краще. Для цього необхідно оцінити результати їх роботи. Для такого оцінювання можна застосувати індекси Силуету та Данна.

Індекс Силуету

Індекс силуету вимірює, наскільки добре кластер відокремлений від інших та наскільки щільно згруповані його елементи. Це середнє значення силуетних

коефіцієнтів для кожної точки[50]. Коефіцієнти можуть варіюватися від -1 до 1, де вищі значення вказують на кращу кластеризацію. Індекс силуету для кожного об'єкта розраховується окремо, а потім усереднюється по всім об'єктам для отримання загального індексу[51]. Формула 2.8. обчислює такий індекс для окремого об'єкта:

$$s(i) = \frac{b(i)-a(i)}{\max(a(i),b(i))}, \tag{2.8}$$

де *a(i)* є середньою відстанню від об'єкта *i* до всіх інших об'єктів у тому ж кластері;

b(i) є найменшою середньою відстанню від об'єкта *i* до об'єктів іншого кластеру, тобто відстанню до найближчого кластеру.

Індекс Данна

Індекс Данна визначає співвідношення між найменшою відстанню, яка розділяє кластери, та найбільшою відстанню всередині кластера. Високе значення індексу Данна вказує на кращу кластеризацію, де кластери є добре відокремленими та компактними[101]. Такий індекс обчислюється Формулою 2.9.

$$D = \min_{1\le i\le k}\left(\min_{i\neq j}\frac{\delta(c_i,c_j)}{\max\limits_{1\le l\le k}\Delta(c_l)}\right), \tag{2.9}$$

де $\delta(c_i,c_j)$ визначає відстань між кластерами c_i та c_j;

$\Delta(c_l)$ визначає внутрішню відстань кластера c_l;

k є кількістю кластерів.

Застосувавши обидва індекси для оцінювання результатів кластеризації розглянутими методами (Табл. 2.1.)

Таблиця 2.1. Результати оцінювання кластеризації

Метод кластеризації	Індекс Силуету	Індекс Данна
K-means	0.4765	0.0297
K-medians	0.4460	0.0288
DBSCAN	-0.0104	0.0167
C-means	0.4867	0.0365

Результати дозволяють зробити висновок, що C-means працює найкраще, оскільки у нього найвищі показники індексів Силуету та Данна.

Проте, запустивши ті ж самі методи кластеризації на новому наборі даних, отриманому з іншого джерела (Таблиця. 2.2.) можна помітити, що індекс Данна для K-medians є вищим ніж у C-means

Таблиця 2.2. Результати оцінювання повторної кластеризації

Метод кластеризації	Індекс Силуету	Індекс Данна
K-means	0.5365	0.2177
K-medians	0.5460	0.3138
DBSCAN	0.3454	0.1567
C-means	0.5167	0.3365

На основі показників, отриманих при повторному запуску методів кластеризації, але уже для іншого вхідного набору даних (Табл. 2.2.), можна сформулювати нову проблему. Для різних вхідних наборів даних оптимальний метод кластеризації може відрізнятись.

Така різниця логічно пояснюється тим, що джерела отримання даних рекрутингової діяльності передбачають набори інформації, яка абсолютно відрізняється структурою, форматуванням, кількістю даних, в залежності від

способу збирання інформації (відкрита база даних, профіль LinkedIn, резюме, тощо)[52].

Для вирішення такої проблеми пропонується використання ансамблю методів кластеризації. Такий підхід дозволить не обмежуватись одним методом, який не завжди може показувати оптимальний результат, а використовувати різні кластеризатори для отримання найбільш точного визначення кластерів[53].

Ансамблевий кластерний аналіз – це передовий метод, який поєднує сильні сторони декількох алгоритмів кластеризації для підвищення точності, стабільності та надійності результатів кластеризації. Замість того, щоб покладатися на один алгоритм, який може мати обмеження та упередження, ансамблева кластеризація інтегрує різні методи, такі як K-середні, ієрархічна кластеризація, DBSCAN та інші. Цей колективний підхід пом'якшує індивідуальні недоліки кожного алгоритму і більш ефективно охоплює різні структури даних.

Процес ансамблевої кластеризації:

- Різноманітні методи кластеризації: застосування кількох алгоритмів кластеризації до одного набору даних, генеруючи різноманітні кластерні рішення.
- Досягнення консенсусу: об'єднання результатів різних алгоритмів, використовуючи такі методи, як голосування, матриці ко-асоціацій або функції консенсусу, щоб сформувати єдине рішення для кластеризації.
- Агрегація результатів: оцінка узгодженості та відповідності між різними результатами кластеризації для отримання остаточних кластерів.

Переваги:

- Підвищена точність: завдяки використанню декількох алгоритмів, ансамблева кластеризація часто дає більш точні та надійні результати.
- Надійність: ансамблеві методи менш чутливі до недоліків окремих методів, таких як чутливість до початкових умов у K-середніх або припущень про щільність у DBSCAN[54].

- Комплексне представлення даних: Захоплює різноманітні шаблони і структури в даних, що робить його придатним для складних наборів даних з різною щільністю і рівнем шуму[55].

Для поставленої задачі Такий підхід передбачає наступні кроки:

- Виконання кластеризації з кожним з алгоритмів: K-means, K-medians, DBSCAN і C-means.
- Об'єднати результати кластеризації.
- Використати алгоритм голосування на основі індексів Силуету та Данна для визначення кластеру для кожної точки даних.

Загальний метод кластеризації при цьому виглядатиме наступним чином:

- Вхідна інформація ділиться на групи по 500 записів.
- Кожна група паралельно опрацьовується розглянутими методами кластеризації з ансамблю.
- Виділені кластери об’єднуються.
- Проводиться голосування по результатах роботи кластеризаторів за допомогою індексу Силуету та Данна.
- Для кожної точки вибирається кластер за результатами голосування.
- Результати кластеризації усіх паралельно опрацьованих пакетів об’єднуються і зберігаються в сховище даних.

Опрацювавши набір даних за допомогою запропонованого методу, отримаємо результат ансамблевої кластеризації (Рис. 2.8.)

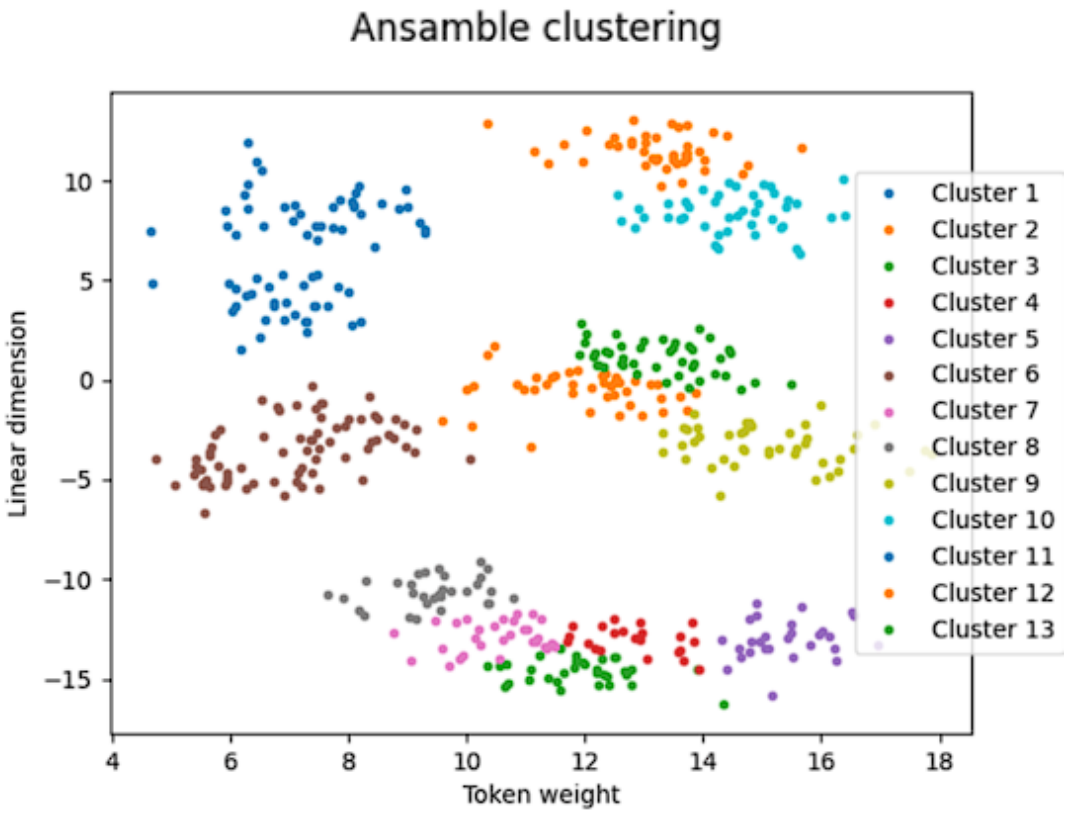


Рис. 2.8. Результати ансамблевої кластеризації

Рис. 2.8. демонструє визначені кінцеві кластери точок набору даних отримані за результатами голосування в ансамблевому методі.

Перевіривши роботу методу на практиці та підтвердивши його функціональність можна побудувати загальну схему його алгоритму (Рис 2.9).

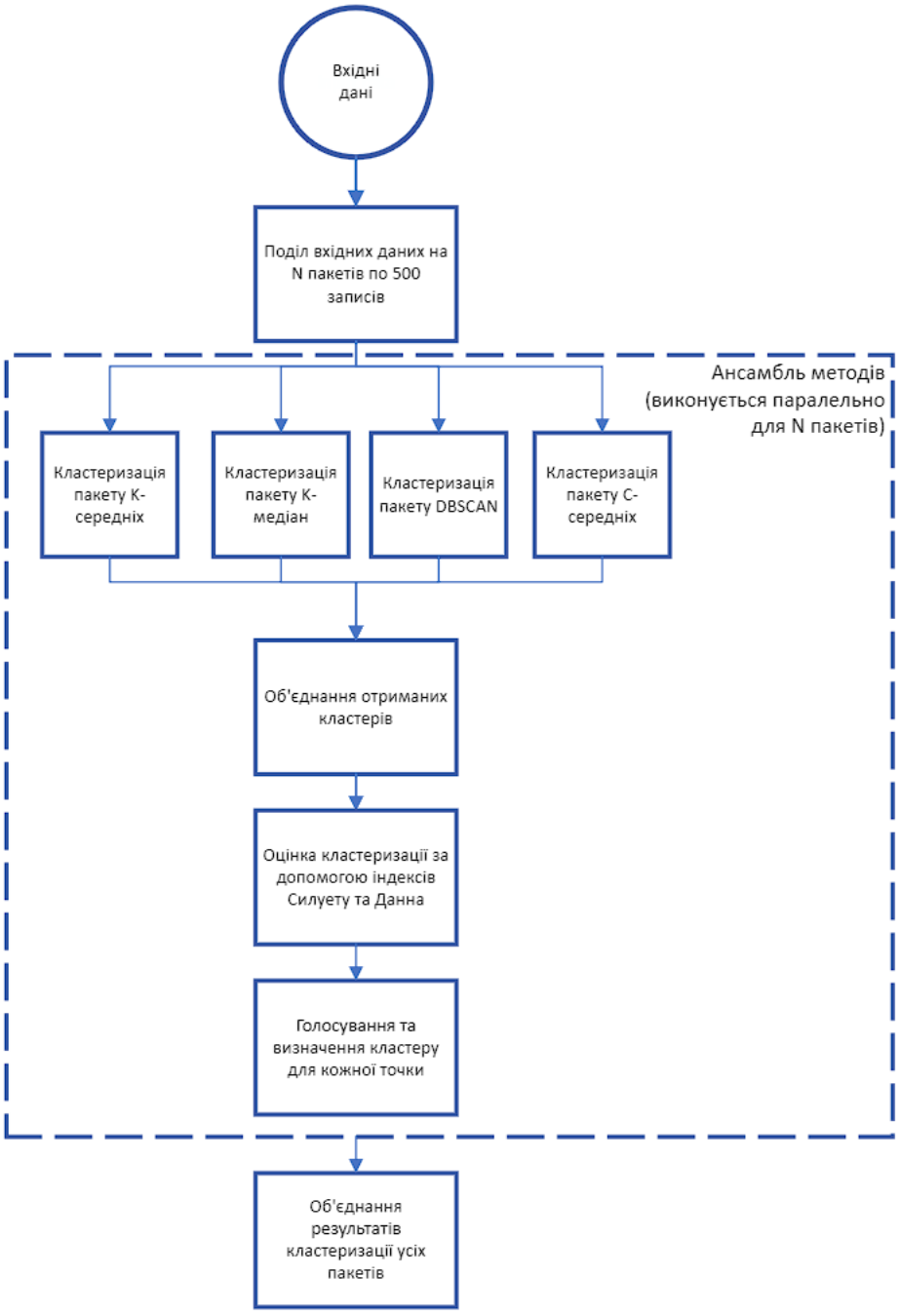


Рис. 2.9. Розроблений метод кластеризації вхідних даних

Вхідна інформація на Рис.2.9 являє собою неоднорідні дані, отримані з джерел, які не дозволяють встановити окремо приналежність одиниці інформації (резюме) до певного класу, тому для визначення таких класів проводиться кластеризація.

Уся сукупність вхідних даних розбивається на набори по 500 записів - кожен.

Виконується паралельне опрацювання усіх пакетів ансамблевим методом кластеризації. З визначених кластерів для кожної з точок за допомогою голосування з використанням індексів Силуету та Данна вибирається кінцевий. Останнім етапом є об’єднання визначених кластерів з їх даними та зберігання для подальшого опрацювання.

2.3. Висновок

Проаналізовано різні методи кластеризації даних, такі як K-means, K-medians, DBSCAN, C-means. Випробувано їх на досліджуваному наборі даних.

Для розглянутих методів проведено оцінювання результатів роботи за допомогою індексів Силуета та Данна.

Прийнято рішення про використання ансамблю методів кластеризації через нестабільність результатів на різних наборах даних.

Розроблено власний метод кластеризації даних з використанням ансамблю існуючих методів та виділення кінцевого кластера за допомогою голосування на основі значень індексів Данна та Силуету з використанням паралельних обчислень.

РОЗДІЛ 3. РОЗРОБКА МЕТОДУ ПОБУДОВИ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ТА ВИКОРИСТАННЯ ЗВОРОТНЬОГО ЗВ'ЯЗКУ ДЛЯ ПОКРАЩЕННЯ РЕКОМЕНДАЦІЙ

У розділі проведено аналіз методів кластеризації даних, найбільш перспективні випробувані на реальних даних та зроблено висновок про їх ефективність для вирішення поставленої задачі. Розроблено метод визначення рівня задоволеності користувача та врахування зворотнього зв'язку для корекції оцінки. Розроблено алгоритм системи підтримки прийняття рішень з застосуванням обраних методів.

Результати розділу опубліковано у працях автора [2,3]

3.1. Проектування системи підтримки прийняття рішень

Системи підтримки та прийняття рішень на основі даних (СППР) – це комп'ютерні системи, призначені для допомоги при прийнятті бізнес-рішення в складних і неструктурованих ситуаціях, шляхом надання бізнес-аналітики, даних та інструментів моделювання для підтримки процесу прийняття рішень[56]. Їх основна мета – підвищити ефективність і якість прийняття рішень, полегшуючи доступ до великих обсягів даних. Наведемо основні компоненти і характеристики СППР:

1. База даних: складається з інформації, яка підтримує прийняття рішень. Вона може включати як внутрішні дані компанії (наприклад, фінансові звіти або показники продажів), так і зовнішню інформацію, таку як ринкові тенденції або дані конкурентної розвідки.
2. Моделювання та аналітика: інструменти та методи для аналізу даних, прогнозування результатів, моделювання різних сценаріїв і оцінювання потенційного впливу варіантів рішень[57].

3. Інтерфейс користувача: надає користувачам легкий доступ до даних та аналітичних інструментів. Цей інтерфейс може включати інтерактивні звіти, інформаційні панелі, графіки та інструменти візуалізації.
4. Інтеграція системи: DSS потрібен зручний механізм інтеграції з іншими інформаційними системами, наприклад управління взаємовідносинами з клієнтами (CRM) і планування ресурсів підприємства (ERP).
5. DSS може допомогти у прийнятті складних рішень: її ефективність є ще більш очевидною в сценаріях, де необхідно враховувати багато змінних і де традиційні методи прийняття рішень можуть бути недостатніми[58].
6. Гнучкість і адаптивність: системи підтримки та прийняття рішень повинні бути достатньо гнучкими, щоб адаптуватися до змін як у бізнес-середовищі, так і у вимогах користувачів[59].

Таким чином системи підтримки та прийняття рішень на основі аналізу даних є невід'ємною частиною сучасного бізнес-середовища, де швидке та обґрунтоване прийняття рішень може надати компаніям перевагу.

Загалом, задачу для такої системи підтримки та прийняття рішень можна сформулювати наступним чином, як на основі загальних даних ринку рекрутингу та інформації про поточних працівників компанії оцінити нового кандидата.

Вважаючи, що поточні працівники компанії добре їй підходять, необхідно побудувати певну систему рекомендацій, яка б оцінювала нових кандидатів виходячи з схожої комбінації показників. Така задача є задачею регресійного аналізу, на якому і базуватиметься робота системи.

3.1 Аналіз методів регресії

Регресія – це статистичний метод аналізу даних, який використовується для вивчення взаємозв'язку між однією або кількома незалежними (пояснювальними)

змінними та залежною (змінною відгуку)[60]. Цей метод дозволяє визначити, які змінні впливають на інші змінні та в якій мірі.

Основна мета регресійного аналізу – зрозуміти взаємозв'язок між змінними та передбачити значення залежної змінної на основі значень незалежних змінних[61]. У простій лінійній регресії є одна незалежна змінна, яка використовується для прогнозування значень залежної змінної, тоді як у множинній лінійній регресії може бути кілька незалежних змінних.

Регресійний аналіз складається з кількох етапів[62]:

- Збір даних: починаючи зі збору даних, дослідник визначає залежні та незалежні змінні, які він планує вивчати.
- Побудова моделі: дослідник обирає математичну модель, яка найкраще відповідає його даним, наприклад, просту лінійну модель або більш складну нелінійну модель.
- Оцінка параметрів моделі: за допомогою статистичних методів, таких як метод найменших квадратів, оцінюються параметри моделі, які найкраще відповідають даним.
- Перевірка гіпотез: після оцінки параметрів проводиться статистичний аналіз для визначення статистичної значущості моделі та її параметрів.
- Використання моделі для прогнозування: після того, як модель побудована, її можна використовувати для прогнозування значень залежної змінної на основі нових значень незалежних змінних.

Регресійний аналіз допомагає зрозуміти взаємозв'язки між різними факторами, що може бути корисним для прийняття рішень у бізнесі, наукових дослідженнях та інших сферах. Цей метод дозволяє виявити та спрогнозувати вплив різних факторів на результати і діяти відповідно до них[63].

Оскільки таких методів існує велика кількість потрібно обрати саме той метод, який найкраще підійде для нашої моделі даних та поставлених вимог.

3.2. Аналіз методів регресії для застосування у системі підтримки та прийняття рішень для рекрутингової діяльності

1. Лінійна регресія моделює зв'язок між однією або декількома незалежними змінними та залежною змінною за допомогою лінійного рівняння[64](Формула 3.1).

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n, \quad (3.1)$$

де $\hat{y}$ - прогнозоване значення залежної змінної;

β_0 - попередній член (зсув);

$\beta_1,\beta_2,\ldots,\beta_n$ - коефіцієнти регресії;

$x_1,x_2,\ldots,x_n$ - значення незалежних змінних.

Переваги: легко зрозуміти та реалізувати, вимагає невеликих обчислювальних ресурсів.

Недоліки: передбачає лінійний зв'язок між змінними, що недосяжно для таких даних, як вміст резюме та навички кандидатів.

Таким чином цей метод не підходить для розроблюваної системи. Немає сенсу пробувати його на практиці.

2. Гребенева регресія вводить штраф на великі значення коефіцієнтів регресії, який додається до функції втрат. Цей штраф виражається через суму квадратів коефіцієнтів (β_j), помножену на параметр λ, який вибирається перед аналізом та визначає рівень регуляризації (Формула 3.2).

$$\hat{\beta}_{ridge} = (X^T X + \lambda I)^{\{-1\}} X^T y, \quad (3.2)$$

де $\hat{\beta}_{ridge}$ - оцінка параметрів моделі гребеневої регресії;

X - матриця ознак;

y - вектор відгуків;

λ - параметр регуляризації;

I - одинична матриця.

Переваги: ефективний, коли змінні сильно корельовані, зменшує складність моделі.

Недоліки: потребує вибору параметра регуляризації, що може потребувати додаткового налаштування.

Оскільки система працюватиме на неоднорідних даних і не вимагатиме від користувача розуміння процесів, вибір параметру регуляризації може бути проблемою. Немає сенсу розглядати практичні результати.

3. Регресія на основі дерева рішень – це метод машинного навчання, який використовується для прогнозування числових значень змінної відгуку. В її основі лежить ідея побудови дерева рішень, де кожен вузол відповідає тестовій умові, яка розділяє дані на підгрупи, а кожен лист відповідає прогнозованому значенню (Формула 3.3.).

$$\hat{y} = \sum_{i=1}^{N} w_i I(x \in R_i), \tag{3.3}$$

де $\hat{y}$ - це прогнозоване значення регресії для вхідного зразка x;

N - кількість листків у дереві рішень;

w_i - значення, яке присвоєно кожному листку;

R_i - область (регіон), що відповідає листку i;

I $(x \in R_i)$ - індикаторна функція, яка дорівнює 1, якщо вхідний зразок x попадає в область R_i, і 0 в іншому випадку.

Ключовими аспектами та перевагами регресії на основі дерева рішень є:

1. Побудова дерева рішень: алгоритм побудови дерева рішень починається з кореневого вузла, який представляє всі дані. Потім вузли поділяються на дві або більше гілок залежно від обраної умови поділу, наприклад, "чи є x більшим за?" або "чи є у меншим за?". Цей процес повторюється рекурсивно, поки не досягне встановленого критерію зупинки.
2. Прогнозування числових значень: у випадку регресії на кожному листі дерева рішень встановлюється прогнозне числове значення. Це може бути

середнє або медіанне значення змінної відгуку в підгрупі даних, які потрапили в цей лист.

3. Гнучкість та інтерпретованість: дерева рішень можуть легко адаптуватися до складних залежностей у даних, що робить їх досить гнучкими. Крім того, вони добре піддаються інтерпретації, оскільки дерево можна візуалізувати і легко зрозуміти.
4. Стійкість до викидів: регресія на основі дерева рішень є стійкою до викидів і шуму в даних, оскільки дерево може робити розбиття, оптимізовані для врахування тенденцій і розподілу даних.
5. Необхідність підтримки гілок: однак дерева рішень можуть бути схильні до перенавчання, особливо якщо їм дозволити занадто розростися. Цю проблему можна вирішити, обмеживши глибину дерева або використовуючи методи регуляризації, такі як обрізка.

Переваги: нелінійний підхід; може моделювати складні залежності; інтуїтивно зрозумілий.

Недоліки: схильний до перенавчання, особливо при великій глибині дерева.

Загалом цей метод регресії, не потребує лінійної залежності даних, але можуть виникнути проблеми з перенавчанням.

4. Регресія випадкового лісу – це ансамблевий метод машинного навчання, який використовує багато дерев рішень для вирішення проблем регресії[65]. Кожне дерево в ансамблі навчається на випадково вибраній підмножині навчальних даних, а результати всіх дерев об'єднуються для визначення остаточного прогнозу (Формула 3.4).

$$\hat{y} = \frac{1}{N}\sum_{j=1}^{N} T(x, \theta_j), \tag{3.4}$$

де $\hat{y}$ - це прогнозоване значення регресії для вхідного зразка x;

N - кількість дерев у випадковому лісі;

$T(x,\theta_j)$ - прогноз, що зроблений j-им деревом з параметрами θ_j.

Переваги: добре обробляє великі набори даних і може ефективно обробляти набори даних з великою кількістю змінних. Забезпечує важливість змінних, що може допомогти в розумінні того, які характеристики мають найбільший вплив на результат. Менш схильні до надмірної підгонки, ніж окремі дерева рішень.

Недоліки: може зайняти багато часу для навчання, особливо з великими наборами даних. Менш інтуїтивно зрозумілі та складніші для інтерпретації, ніж окреме дерево рішень.

Порівняно з лінійною регресією та її різновидами (Ridge, Lasso, Elastic Net), випадковий ліс може бути більш ефективним у випадках, коли зв'язки між змінними є нелінійними. Він також добре підходить для ситуацій, коли є багато даних і велика кількість ознак (наприклад, детальні аспекти резюме).

Коли випадковий ліс використовується для аналізу резюме, цей метод може ефективно обробляти та аналізувати різноманітні характеристики кандидатів, виявляючи складні взаємозв'язки між різними аспектами їхнього досвіду роботи та навичками. Однак важливо забезпечити об'єктивність і відсутність упередженості в навчальних даних.

Таким чином цей метод виглядає перспективним для застосування в СППР рекрутингової діяльності.

5. Градієнтний бустинг – це потужний метод машинного навчання, який використовується як для класифікації, так і для регресійних задач. Цей метод належить до сімейства ансамблевих методів, де модель будується шляхом об'єднання результатів декількох слабких моделей для отримання більш точного прогнозу.

Основна ідея градієнтного бустінгу полягає в тому, що він використовує градієнтний спуск для оптимізації функції втрат. Модель будується ітеративно шляхом додавання нових базових моделей (зазвичай дерев рішень) до ансамблю,

причому кожна нова модель має на меті виправити помилки, допущені попередніми моделями (Формула 3.5).

$$F(x) = \sum_{m=1}^{M} \gamma_m h_m(x), \quad (3.5)$$

де $F(x)$ - це ансамбль моделей, який будується за допомогою градієнтного бустингу;

M - кількість базових моделей в ансамблі;

γ_m - швидкість навчання (learning rate) для кожної базової моделі $h_m(x)$;

$h_m(x)$ - базова модель, яка додається до ансамблю на кожному кроці.

Основні переваги градієнтного бустінгу:

- Сильна прогнозована здатність: градієнтний бустінг може забезпечити дуже точні прогнози, оскільки він поєднує результати багатьох слабких моделей.
- Стійкість до надмірного припасування: використання градієнтного бустінгу допомагає уникнути надмірного припасування, постійно покращуючи модель на кожному кроці.
- Можливість працювати з різними типами даних: градієнтний бустінг можна використовувати як для задач класифікації та регресії, так і для роботи з даними різної природи.

Незважаючи на ці переваги, градієнтний бустінг має деякі недоліки, наприклад, навчання може зайняти багато часу, особливо з великою кількістю базових моделей, і може вимагати налаштування гіперпараметрів.

Зважаючи на результати аналізу методу, він теж виглядає багатообіцяюче для застосування в СППР, оскільки він працює з різними даними та дає дуже точні прогнози.

3.3. Розробка методу регресії для системи підтримки та прийняття рішень

Отже, на основі аналізу, проведеному в попередньому розділі, було виділено три методи, які можуть показати найкращий результат регресії, що дозволить

отримати максимальну точність оцінки для системи підтримки та прийняття рішень.

Вхідними даними системи для навчання будуть дані поточних працівників умовного підприємства та їх оцінка надана керівництвом, що дасть змогу максимально підлаштувати результати роботи системи до потреб конкретного підприємства.

Також цим методом регресії аналізуватиметься усереднине резюме відповідного кластеру з бази даних рекрутингового ринку, а її результати застосуваватимуться для оцінки кореляції: якщо оцінка усередненого резюме буде вища, ніж оцінка кандидати – вона буде зменшена на 20%, оскільки в середньому на ринку є кращі спеціалісти.

3.3.1. Побудова дерева рішень

Дерево рішень будується шляхом рекурсивного поділу навчального набору даних на підмножини з його кореневого вузла на основі певних критеріїв.

Кожен вузол у дереві розглядається як незалежний запис, який можна розбити на розділи. Процес розбиття триває до тих пір, поки не будуть досягнуті певні умови зупинки, такі як максимальна глибина дерева або мінімальна кількість спостережень у вузлі.

Після створення дерева, правила в його гілках використовуються для прогнозування значень для нових спостережень, що потрапляють у відповідний вузол листка. Ці прогнозовані значення представляють середнє значення всіх спостережень, які потрапляють у цей листовий вузол (Рис 3.1.).

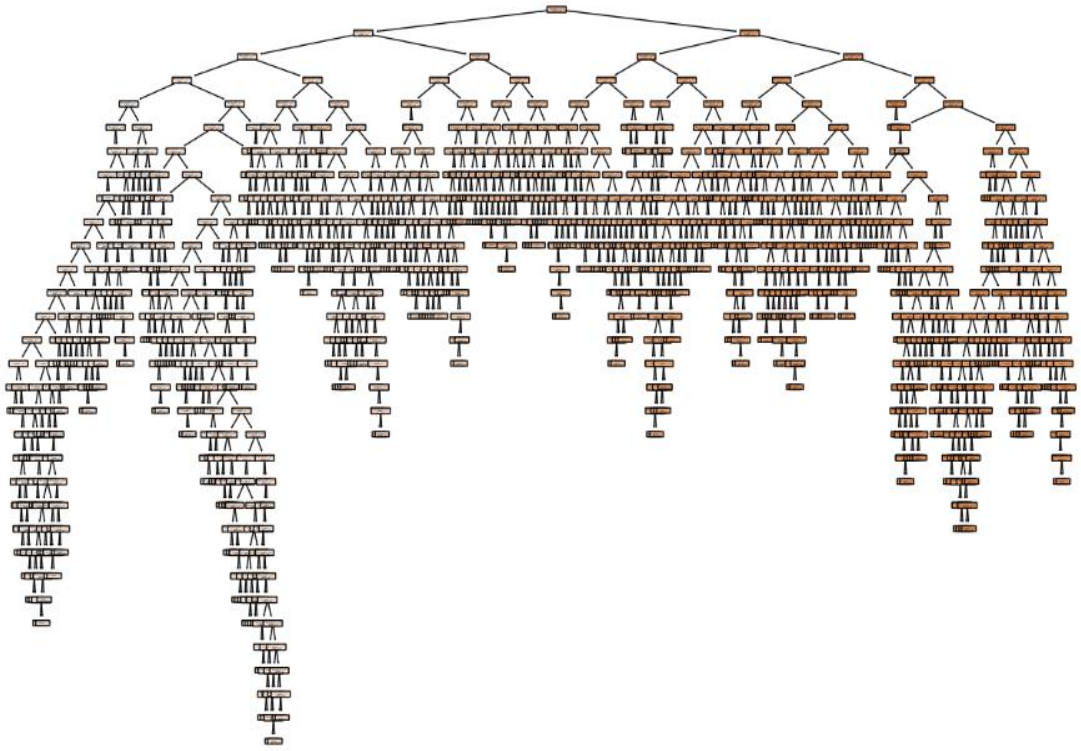


Рис. 3.1. Побудоване дерево рішень

Так виглядає отримане дерево рішень, де в кожному вузлі зберігаються дані про кількість спостережень, які у нього потрапили, їх значення та середньоквадратичне відхилення.

Проаналізувавши дані з використанням отриманого дерева побачимо наступний результат (Рис. 3.2.)

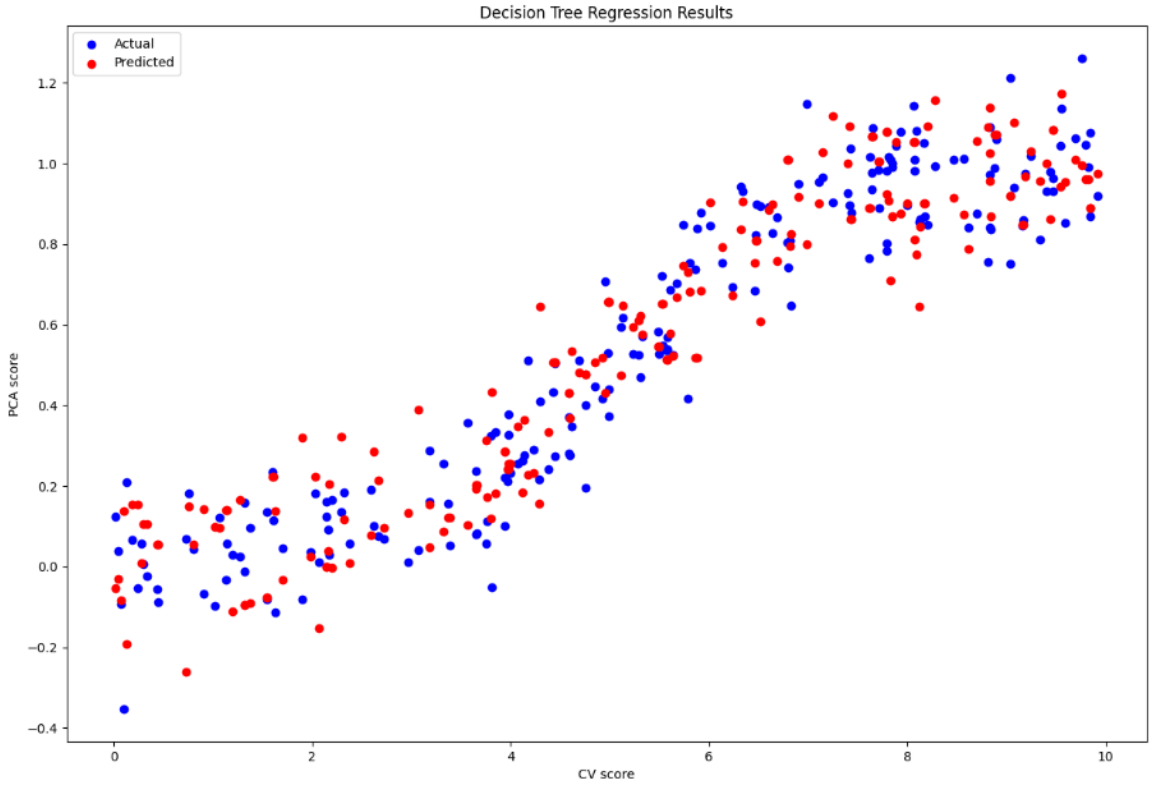


Рис. 3.2. Результат регресії дерева рішень

Рис. 3.2. показує результат регресії за допомогою дерева рішень. Як помітно, загальний масив прогнозованих оцінок досить сильно накладається, що говорить про високу точність регресії. Проте, для остаточного вибору потрібно проаналізувати інший метод.

3.3.2. Побудова випадкового лісу

Метод Random Forest використовує бутстрапінг для випадкового вилучення декількох підмножин навчальних даних, які потім використовуються для незалежної побудови дерев рішень.

Для кожної підмножини даних будується дерево рішень, використовуючи випадково вибрані ознаки як розгалужувачі вузлів для формування його вузлів у

деревоподібній структурі, без необхідності обрізання для підтримки максимального зростання розміру кожної гілки протягом її життя.

Випадковий ліс об'єднує прогнози з окремих дерев рішень у глобальний прогноз. При використанні цього методу для регресії, означатиме усереднення прогнозів з кожного дерева.

При цьому Random Forest може оцінювати важливість ознак, використовуючи середню втрату точності або середнє зниження нечистоти.

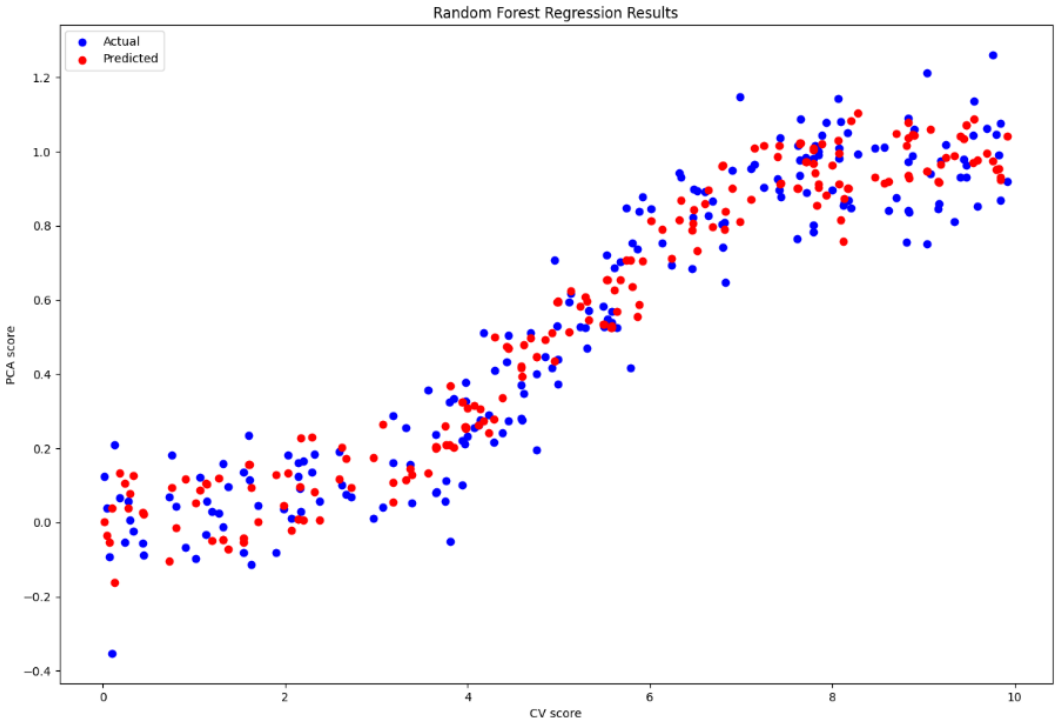


Рис. 3.3. Результат регресії випадкового лісу

Рис. 3.3. демонструє результати регресії випадкового лісу на тому ж наборі даних, що і використовувався для дерева рішень (Рис. 3.2.). Візуально, точність теж виглядає високою, необхідно порівнювати конкретні показники.

3.3.3. Побудова градієнтного бустингу

Градієнтний бустінг будує модель поетапно і узагальнює її, дозволяючи оптимізувати довільну диференційовану функцію втрат.

Для побудови методу градієнтного бустингу важливо знати основні його елементи.

Бустінг: це ансамблева техніка, яка об'єднує прогнози від декількох базових оцінок (слабких учнів) для підвищення надійності та точності. На відміну від методів мішків, таких як випадкові ліси, де дерева будуються незалежно, бустінг будує дерева послідовно, причому кожне нове дерево намагається виправити помилки, допущені попередніми[66].

Слабкий учень: в контексті бустінгу, слабкий учень – це модель, яка працює трохи краще, ніж випадкове вгадування. Пеньки рішень (дерева з одним розщепленням) є поширеним прикладом слабкого навчання.

Градієнтний спуск: градієнтний бустінг використовує градієнтний спуск для мінімізації функції втрат. Кожна нова модель навчається передбачати залишки (помилки) попередніх моделей, рухаючись у напрямку від'ємного градієнта функції втрат.

Процес градієнтного спуску:

- Ініціалізація моделі: починається з простої моделі, як правило, з постійної величини, яка мінімізує функцію втрат (наприклад, середнє значення для регресії).
- Підготовка слабкого учня: навчання слабкого учня на залишках прогнозів попередньої моделі.
- Оновити модель: додавання нового слабкого учня до ансамблевої моделі зі швидкістю навчання, яка контролює внесок кожного учня.
- Ітерація: повторення кроків 2 і 3 для заданої кількості ітерацій або до збіжності.

Хорошим рішенням буде застосування не звичайного методу градієнтного бустингу, а його покращеного варіанту – XGBoost.

Ключові особливості XGBoost[67]:

- Регуляризація: XGBoost включає в себе регуляризацію L1 (Lasso) та L2 (Ridge), щоб запобігти надмірному пристосуванню.
- Паралельна обробка: XGBoost може використовувати багатоядерні процесори для швидшого навчання.
- Обрізка дерева: XGBoost використовує техніку під назвою "максимальна глибина", щоб уникнути перенавчання.
- Обробка відсутніх значень: XGBoost має вбудовану підтримку для обробки відсутніх значень.
- Зважений квантильний ескіз дозволяє алгоритму більш ефективно обробляти зважені дані.
- Перехресна перевірка: XGBoost включає вбудовану перехресну перевірку на кожній ітерації, що дозволяє зупиняти алгоритм на ранній стадії.

Алгоритм XGBoost послідовно будує ансамбль дерев, при цьому кожне нове дерево виправляє помилки попередніх[68]. Він використовує алгоритм градієнтного спуску для мінімізації функції втрат, додаючи регуляризацію для запобігання надмірному пристосуванню.

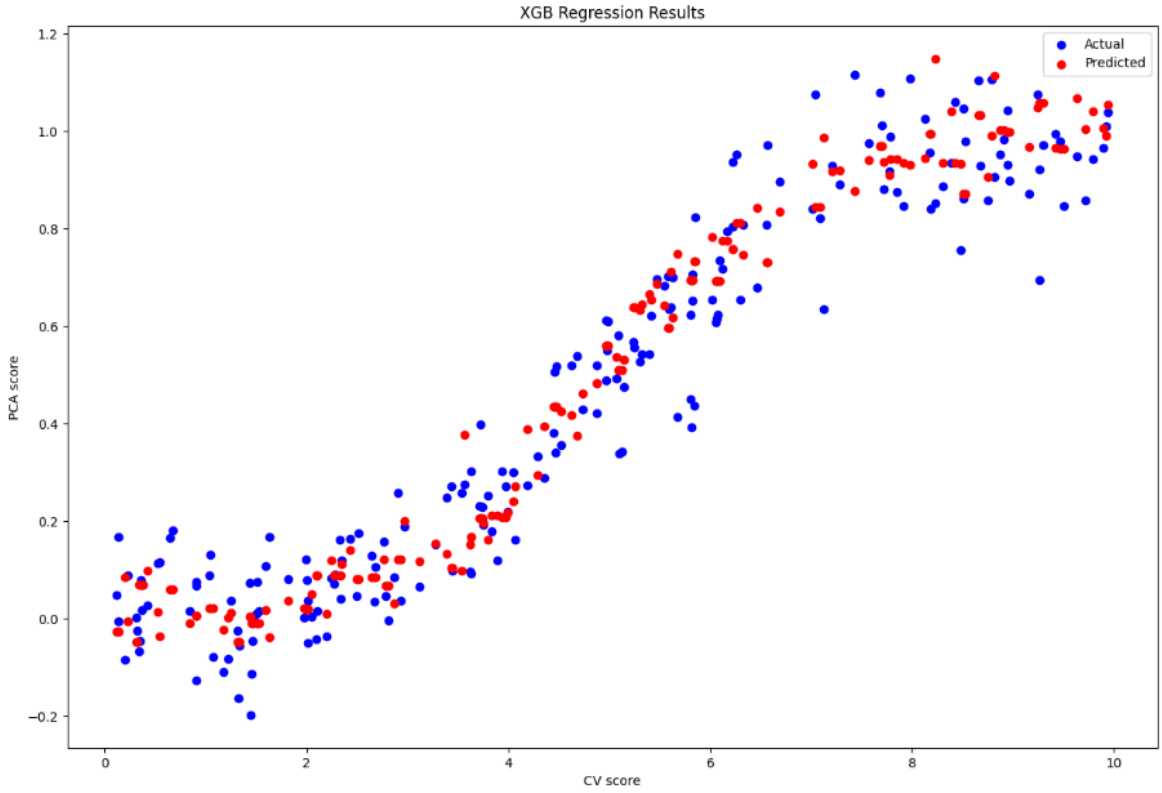


Рис. 3.4. Результат методу XGBoost

Рис. 3.4. показує результати регресії XGBoost на такому ж наборі даних, що і використовувався для попередніх методів регресії. Потрібно визначити статистичні показники для прийняття рішення стосовно використання конкретного методу регресії.

3.3.4. Порівняння методів регресії

Після реалізації усіх методів можна порівняти отримані результати та визначити більш оптимальний.

Усі три методи були реалізовані використовуючи одну і ту ж технологію (Python), натреновані на одному і тому ж наборі даних розміром 50000 записів та перевірені на тому ж тестовому записі (Рис. 3.5.).

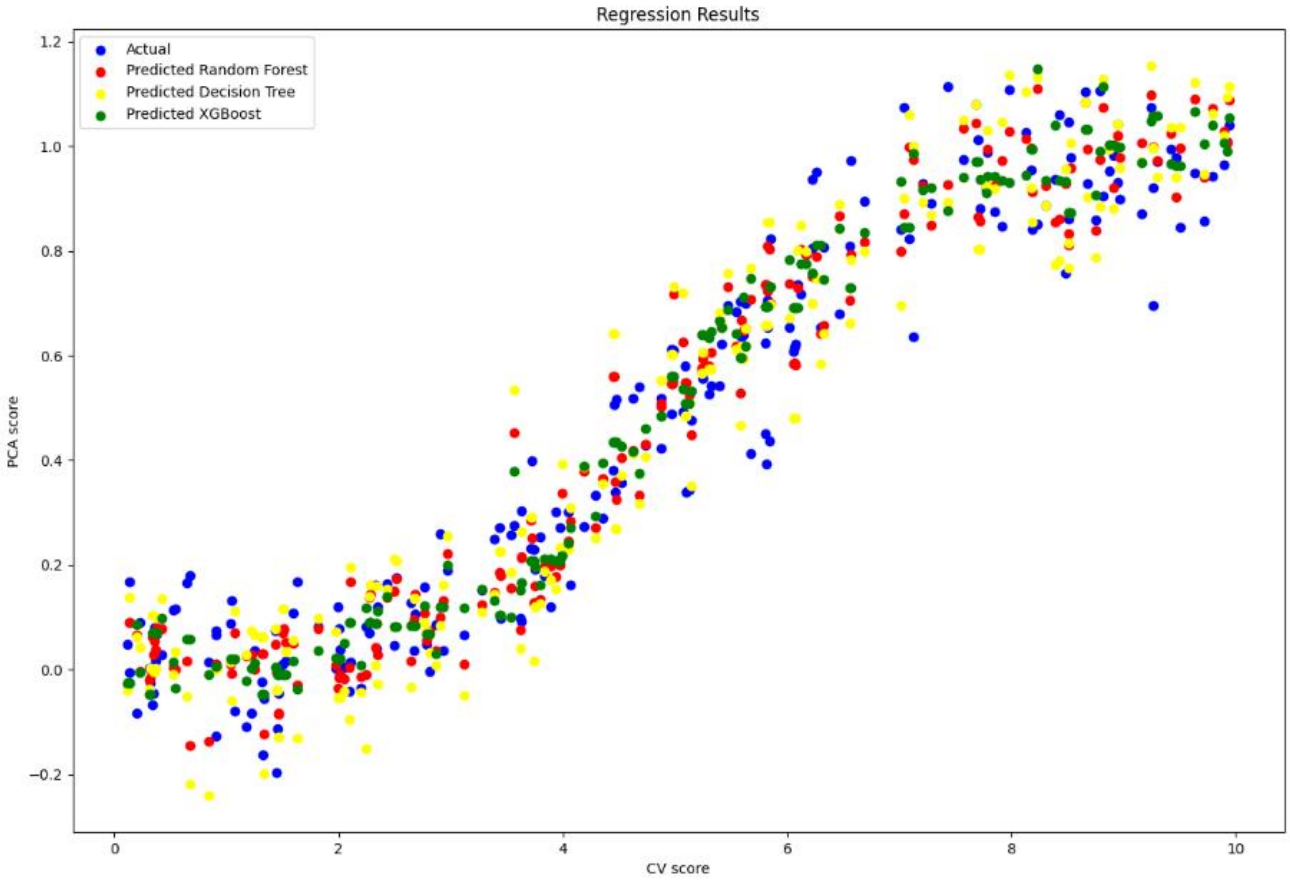


Рис. 3.5. Порівняння результатів регресії розглянутих методів

На Рис. 3.5. зображено результати визначення залежності оцінки від вхідного набору ознак кандидатів. Помітно, що різні методи регресії дали дещо відмінні результати, однак усі вони є досить наближені до цільових значень.

Для порівняльного оцінювання результатів регресії було вибрано наступні 4 показники[96]:

1. Середня абсолютна помилка (MAE) вимірює середню абсолютну відстань між прогнозованими та спостережуваними значеннями. Вона менш чутлива до викидів, ніж MSE.
2. Коефіцієнт детермінації (R^2) вимірює відстань між спостережуваними значеннями та значеннями, передбаченими моделлю. R^2 може приймати значення від 0 до 1, де 1 означає ідеальне прогнозування.

3. Середня абсолютна відсоткова помилка (MAPE) вимірює середнє відхилення прогнозів від фактичних значень у відсотках. Вона дозволяє оцінити точність прогнозування відносно величини фактичних значень.
4. Корельована коефіцієнт Пірсона визначає ступінь кореляції між фактичними та прогнозованими значеннями.

Результат (Таблиця 3.1.) показав, що метод Градієнтного бустингу(XGBoost) має дещо кращі показники регресії, ніж дерево рішень та випадковий ліс, хоча різниця і не значна.

Таблиця 3.1. Оцінка роботи трьох методів регресії

	Дерево рішень	**Випадковий ліс**	**XGBoost**
Mean Absolute Error	0.11096037580	0.09857872933	0.09187863903
R^2 Score	0.87865839266	0.90389408999	0.91580770206
Mean Absolute Percentage Error	1.54335260689	1.34695710708	1.14517005012
Pearson Correlation Coefficient	0.94126559086	0.95239509996	0.95821153786

Таблиця 3.5. показує основні параметри трьох регресій, з яких можна зробити висновок про незначну перевагу в точності регресії у методу XGBoost, який і буде використовуватись для СППР.

Для кожної з регресій ми також отримали потенційну оцінку кандидата (Рис. 3.6).

```
Decision Tree Score: [6.80285338]
Random Forest Score: [6.83275636]
XGB Pearson Score: [6.87039943]
```

Рис. 3.6. Оцінки кандидата для різних методів

Як бачимо, оцінки досить близькі, що відповідає близьким результатам точності, але оскільки найвищу точність показав градієнтний бустинг, то в подальшому будемо використовувати саме цей метод та його оцінювання.

3.4. Застосування зворотнього зв’язку кандидата для покращення рекомендацій

Відгуки користувачів – це думки, пропозиції та скарги, надані користувачами щодо їхнього досвіду роботи з інформаційною системою. Ці відгуки можна збирати різними способами, такими як опитування, форми зворотного зв'язку, взаємодія зі службою підтримки, соціальні мережі та пряме спілкування.

Типи зворотного зв'язку з користувачами:

- Опитування та анкети. Структуровані форми, які користувачі заповнюють, щоб висловити свою думку про різні аспекти системи.
- Відгуки та рейтинги користувачів. Коментарі та оцінки, надані користувачами на платформах або безпосередньо в системі.
- Сповіщення підтримки та взаємодія зі службою підтримки. Проблеми, про які користувачі повідомляють через канали підтримки клієнтів.
- Тестування застосовуваності. Спостереження та відгуки, зібрані під час сесій користувацького тестування.
- Соціальні мережі та форуми. Обговорення та повідомлення на платформах соціальних мереж або форумах користувачів.

- Прямий зворотній зв'язок. Коментарі та пропозиції, надані безпосередньо команді розробників, або через форми зворотного зв'язку в системі.

Варіанти застосування зворотнього зв’язку:

1. Забезпечення задоволеності користувачів.

Утримання клієнтів: Задоволені користувачі з більшою ймовірністю продовжуватимуть користуватися системою, що призводить до вищого рівня утримання клієнтів.

Позитивні відгуки: Задоволені користувачі з більшою ймовірністю залишать позитивні відгуки та порекомендують систему іншим, що може залучити нових користувачів.

2. Прийняття рішень на основі даних.

Обґрунтовані рішення: Відгуки надають конкретні дані, якими можна керуватися в процесі прийняття рішень, гарантуючи, що зміни та вдосконалення ґрунтуються на реальних потребах та вподобаннях користувачів.

Аналіз змін: Зворотній зв'язок дозволяє зрозуміти важливість нових функцій і змін, забезпечуючи основу для постійного вдосконалення.

3. Покращення користувацького досвіду (UX).

Визначення больових точок: відгуки користувачів допомагають виявити проблеми з використанням, помилки та функціонал, в якому користувачі відчувають труднощі. Ця інформація має вирішальне значення для покращення загального користувацького досвіду.

Покращення функцій: розуміючи, що користувачам подобається і не подобається в системі, розробники можуть покращити існуючі функції та додати нові, які відповідають потребам користувачів.

4. Залучення користувачів.

Побудова мережі користувачів: активний пошук та реагування на відгуки користувачів свідчить про те, що компанія цінує своїх користувачів, сприяючи формуванню почуття спільноти та лояльності.

Заохочення участі: коли користувачі бачать, що їхні відгуки враховуються, вони з більшою ймовірністю продовжуватимуть надавати цінну інформацію.

Для застосування зворотнього зв’язку відповідно до розглянутих варіантів в першу чергу потрібно отримати інформацію від користувача. Існує велика кількість різних способів її зібрати:

- Опитування та анкетування: розповсюдження опитування та анкет для збору структурованого зворотнього зв'язку щодо конкретних аспектів системи.
- Форми зворотнього зв'язку: додавання до системи форм зворотнього зв'язку, щоб користувачі могли легко надсилати свої думки та пропозиції.
- Інтерв'ю з користувачами та фокус-групи: проведення інтерв'ю та обговорення у фокус-групах для збору детального зворотнього зв'язку.
- Аналітика та відстеження поведінки користувачів: використання інструментів аналітики для відстеження поведінки користувачів та отримання зворотнього зв'язку на основі моделей використання.
- Моніторинг соціальних мереж: відстеження каналів соціальних мереж на предмет коментарів та обговорень користувачів про систему.
- Взаємодія зі службою підтримки: аналіз звернень до служби підтримки та взаємодії з клієнтами, щоб виявити загальні проблеми та області для покращення.

Виходячи з вимог до розроблюваної системи підтримки прийняття рішень та особливостей рекрутингового процесу оптимальним способом збору зворотнього зв’язку буде спеціальна форма, в якій кандидат виставляє свої оцінки (Рис. 3.7.)

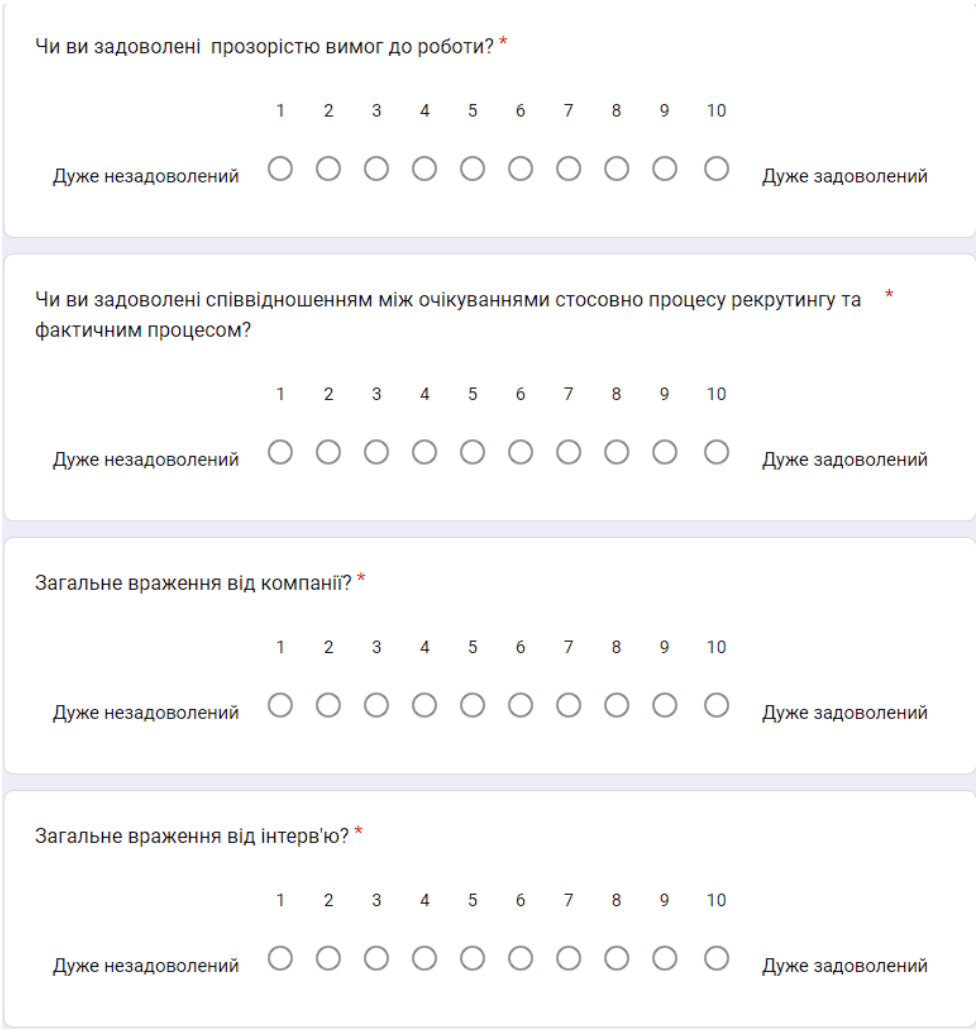


Рис. 3.7. Форма зворотнього зв’язку

Така форма дозволяє зібрати основний відгук користувача стосовно процесу інтерв’ю, його очікувань та реальних результатів, що в майбутньому дозволить зробити висновок про точність рекомендації цього кандидата.

3.4.1. Об'єднання даних зворотного зв'язку з системними даними

Після отримання результатів за 10 бальною шкалою знаходиться їх середнє значення, після чого індекс впливу на оцінку обраховується за формулою 3.6:

$$I = \frac{S - 5}{5}, \tag{3.6}$$

де I – індекс впливу на оцінку;

S – середнє значення оцінки з форми зворотнього зв’язку.

Така формула дозволяє конвертувати оцінку з 10-бальної шкали в шкалу [-0.8;1] після чого обрахований таким чином індекс сумується з оцінкою кандидата визначеною системою.

Інтеграція даних зворотного зв'язку з системними даними дозволяє використовувати дані про ефективність процесу рекрутингу, включаючи інтеграцію показників успішності закриття вакансій, часових рамок відбору кандидатів, оцінки кандидатами процесу, тощо. Якщо кандидат надає негативний зворотній зв’язок – є велика ймовірність, що цей кандидат був помилково запропонований системою, що стало причиною такої оцінки, відповідно зменшення його оцінки в системі зменшить також і вплив його даних на оцінювання інших кандидатів на цю позицію.

3.5. Розроблення алгоритму роботи СППР

Тепер потрібно об’єднати усі проаналізовані та обрані методи в цілісну систему.

Основними задачами є повернення зрозумілих для користувача рекомендації, а також врахування даних ринку при їх формуванні та даних зворотнього зв’язку при зберіганні результатів для подальшої оптимізації роботи.

Таким чином алгоритм роботи системи підтримки та прийняття рішень виглядатиме наступним чином (Рис. 3.8.):

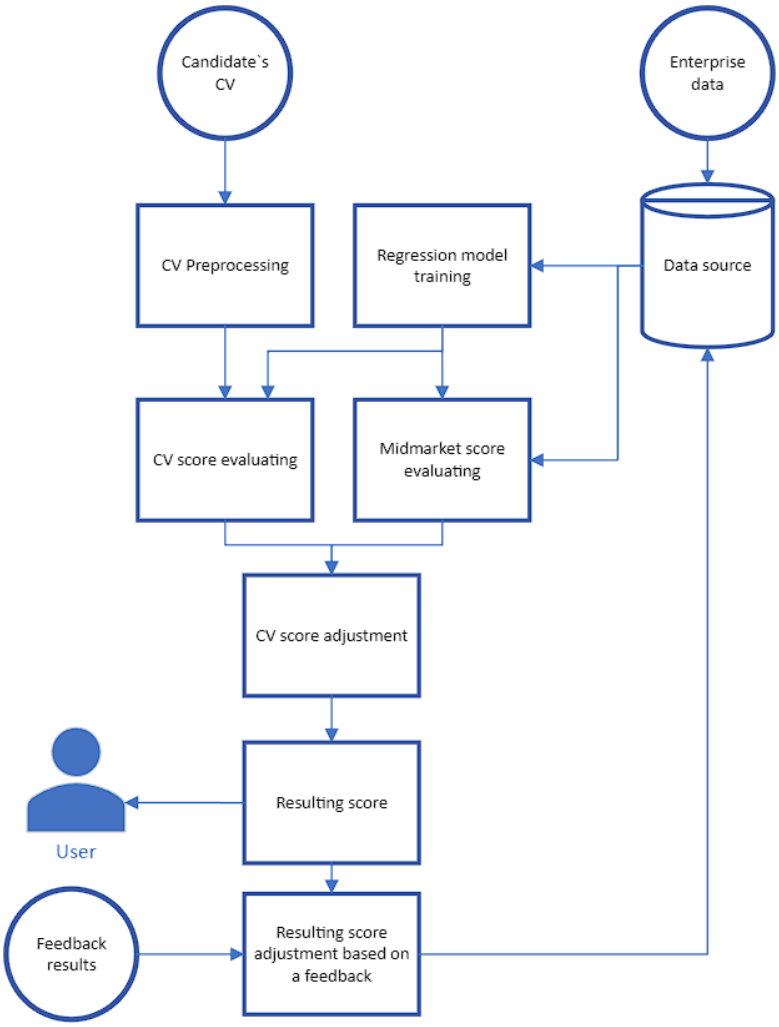


Рис. 3.8. Алгоритм роботи СППР

Можна виділити наступні кроки алгоритму роботи системи підтримки та прийняття рішень:

1) Метод градієнтного бустингу тренується на наборі даних конкретного підприємства (при відсутності або обмеженній кількості даних, застосовуються дані зібрані системою з відкритих джерел).
2) Резюме кандидата проходить попередню підготовку, як дані з публічних джерел, однак не потребує кластерного аналізу, оскільки кластер (позиція) заздалегідь відомі.
3) За допомогою регресії XGBoost визначається оцінка кандидата.

4) Визначається оцінка для усереднених значень даних зібраних системою автоматично з відкритих джерел.
5) Оцінка кандидата збільшується, або зменшується на коефіцієнт визначений підприємством в залежності від того, чи краща вона за оцінку середнього ринкового кандидата.
6) Оцінка демонструється користувачеві як рекомендація СППР.
7) Після проведення інтерв'ю оцінка корегується та зберігається в базі для майбутнього до\перетренування.

3.6. Висновки

У цьому розділі було розглянуто концепцію систем підтримки та прийняття рішень, проаналізовано різні види регресій, як механізму для формування оцінки кандидата на основі даних про інших спеціалістів цієї сфери. Було проведено порівняння три найбільш підходящих видів регресії – дерева рішень, випадкового лісу та градієнтного бустингу, з яких на основі кількох параметрів оцінювання було вибрано ансамблевий метод градієнтного бустингу, як той, який показав кращу точність оцінки.

Було описано створення ефективної системи збору та обробки зворотного зв'язку є ключовим для підвищення швидкості та ефективності процесу рекрутингу. За допомогою розробленого алгоритму було створено процес обрахування індексу впливу на оцінку, що дозволяє коригувати її в залежності від отриманого зворотнього зв'язку.

На основі проаналізованих методів, розроблено алгоритм роботи системи підтримки та прийняття рішень.

РОЗДІЛ 4. РОЗРОБКА АРХІТЕКТУРИ І АПРОБАЦІЯ РЕЗУЛЬТАТІВ

У розділі проведено розроблення архітектури системи, обрано лямбда архітектуру для забезпечення швидкого і ефективного застосування зібраних даних, проведено огляд аналогів, апробовано результати роботи системи підтримки та прийняття рішень.

4.1. Розроблення архітектури системи

Розроблена система орієнтована на використання у форматі веб-застосунку, або інтеграції в існуючі рекрутингові системи за допомогою відкритого API. Такий підхід дозволить використовувати її як на десктоп так і на мобільних пристроях. Сама система не передбачає великої кількості даних, які вводяться через інтерфейс користувача тому буде комфортною і зручною для застосування в будь-якому вигляді[9].

Загальну взаємодію системи із зовнішнім середовищем показано на діаграмі контексту (Рис. 4.1.)

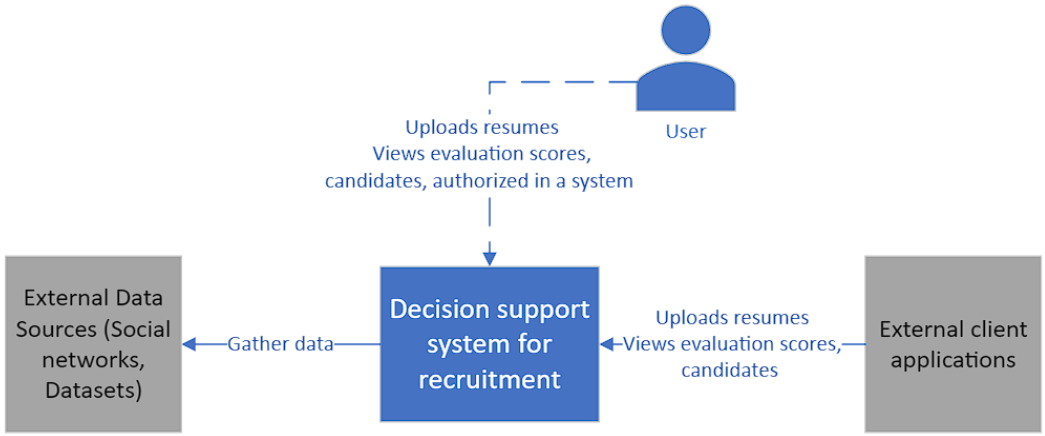


Рис. 4.1. Діаграма контексту

Отже, веб-застосунок – це основний спосіб взаємодії користувача з системою. У ньому користувач зможе завантажити резюме нового кандидата, переглянути його оцінку, запропоновану системою, порівняти її з оцінками інших кандидатів, які знаходяться в базі даних, або додати специфічні налаштування для експертної системи.

В свою чергу веб-застосунок з'єднується з API, яке передає усю необхідну інформацію у веб-застосунок, а також дозволяє використовувати публічні API ендпоінти для взаємодії з інших додатків.

Також частиною API сервісу є моделі системи підтримки та прийняття рішень, відтак для розробки сервісу була обрана технологія Python, яка чудово підходить для обох задач.

Першим кроком буде отримання даних з різних джерел, таких як вебсайти, соціальні мережі, відкриті бази даних тощо.

Вилучення даних із соціальних мереж і веб-сайтів – це метод, який використовується для автоматичного вилучення великих обсягів даних з онлайн-платформ[69]. Цей процес передбачає використання програмних інструментів, відомих як веб-скрепери, для збору такої інформації, як текст, зображення та метадані з веб-сторінок[70].

Етапи вилучення даних[71]:

- Визначення цільових даних, які необхідно отримати з соціальних мереж або веб-сайтів, наприклад, профілі користувачів, пости, коментарі або інформацію про продукт.
- Розробка скрипту для вилучення даних: використання мови програмування, такі як Python, з бібліотеками, такими як BeautifulSoup, Scrapy або Selenium, для створення скриптів, які здійснюють навігацію по веб-сторінках і витягують потрібні дані.
- Обробка структуру веб-сторінок: скрепери аналізують структуру HTML веб-сторінок, щоб знайти відповідні поля даних.

- Автоматизація збору даних: сценарії можуть автоматизувати процес збору даних, багаторазово звертаючись до веб-сторінок для збору оновленої інформації з плином часу.
- Збереження та аналіз даних: витягнуті дані часто зберігаються в базах даних або CSV-файлах для подальшого аналізу за допомогою інструментів і методів обробки даних.

Незважаючи на ці проблеми, скрепинг даних є цінним методом збору великих масивів даних для досліджень, бізнес-аналітики та аналізу ринку, що дозволяє глибше зрозуміти тенденції та поведінку користувачів у соціальних мережах і на веб-сайтах.

API сервісу отримує усю необхідну інформацію з двох сховищ даних – Amazon RDS для даних про користувачів, авторизаційної інформації та працівників, та Amazon Redshift для потокових даних рекрутингової діяльності.

Amazon Relational Database Service (RDS) – це хмарний сервіс баз даних, що надається Amazon Web Services (AWS). Amazon RDS дозволяє легко розгортати, експлуатувати та масштабувати реляційні бази даних у хмарі. Його особливості та переваги:

1. Amazon RDS підтримує декілька систем керування базами даних: Amazon RDS забезпечує підтримку декількох популярних систем управління реляційними базами даних, включаючи Aurora, PostgreSQL, MySQL, MariaDB, Oracle Database та Microsoft SQL Server.

2. Спрощене управління: Amazon RDS спрощує адміністрування баз даних, автоматизуючи багато адміністративних завдань, таких як забезпечення, конфігурація обладнання, резервне копіювання, масштабування та виправлення.

3. Масштабованість: Amazon RDS дозволяє легко масштабувати обчислювальні ресурси та ресурси зберігання відповідно до мінливих вимог робочого навантаження на базу даних.

4. Висока доступність та надійність: Amazon RDS забезпечує багатозонне розгортання та реплікацію для високої доступності та довговічності даних.

5. Безпека: сервіс забезпечує надійне шифрування даних у стані спокою та під час передачі, контроль доступу на основі ролей та надійний захист конфіденційних файлів паролем.

6. Автоматичне резервне копіювання: Amazon RDS автоматично створює резервні копії баз даних і зберігає резервні копії для відновлення даних у разі їх втрати.

7. Моніторинг та сповіщення: сервіс надає інструменти для моніторингу стану бази даних та створення сповіщень для швидкого виявлення та вирішення проблем у міру їх виникнення.

Таким чином він ідеально підходить для зберігання даних користувачів з постійною і незмінною структурою.

Amazon Redshift – це хмарний сервіс зберігання даних від Amazon Web Services (AWS), призначений для зберігання та аналізу великих обсягів даних. Як можливий варіант пропозицій хмарних обчислень, AWS надає повне управління Amazon Redshift.

Ключові особливості Amazon Redshift:

- Redshift пропонує високу продуктивність та масштабованість: технологія сховища даних Redshift використовує стовпчасте зберігання даних і технології паралельної обробки запитів для швидкого аналізу великих обсягів інформації, від гігабайт до петабайт, що робить її придатною для великих наборів даних.
- Економічно ефективні рішення: Amazon Redshift пропонує економічно ефективні рішення для зберігання даних, які дозволяють клієнтам оплачувати лише ті ресурси, які вони використовують.

– Redshift пропонує розширені функції безпеки даних, включаючи шифрування даних у стані спокою та під час передачі, а також контроль доступу на основі ролей.
– Redshift легко інтегрується з іншими сервісами AWS: Redshift без проблем працює з такими сервісами AWS, як S3, Data Pipeline і Lambda, щоб допомогти користувачам створювати складні аналітичні рішення.
– SQL-інтерфейс Redshift: Redshift використовує стандартний SQL для запитів, що робить його доступним для багатьох розробників та аналітиків даних.
– Redshift пропонує підтримку Business Intelligence (BI): Redshift легко інтегрується з популярними інструментами BI, такими як Tableau, PowerBI та іншими для візуального аналізу даних.

Amazon Redshift широко використовується організаціями для зберігання та аналізу даних, підтримки ініціатив бізнес-аналітики та виконання складних запитів до великих наборів даних у хмарній інфраструктурі. Таким чином це сховище ідеально підходить для опрацювання великого потоку даних з мінімальними затримками.

Для втілення системи було обрано лямбда-архітектуру Рис. 4.2.

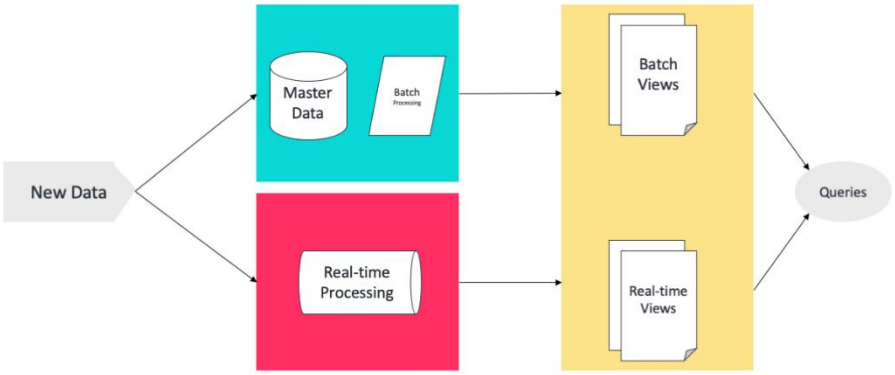


Рис. 4.2. Схема лямбда-архітектури

Така архітектура дозволяє розділити потік даних на швидкий і пакетний рівні, за рахунок чого можна працювати з великими даними практично в режимі реального часу. Швидкий рівень аналізує дані на ходу і зберігає лише короткотривалу інформацію, щоб уникнути дублювання[72].

Пакетний рівень при цьому забезпечує довготривале зберігання даних та надійне їх опрацювання з максимально високою точністю результату. Поєднання таких підходів дозволяє нівелювати недоліки кожного окремого рівня[73].

Для реалізації швидкого шару було обрано кластер сервісів Amazon Redshift, які дозволять аналізувати великі потоки даних паралельно.

Для довготривалого збергіання даних у пакетному шарі було обрано комбінацію Amazon Kinesis Data Firehose та Amazon S3 Datalake.

Amazon Simple Storage Service (S3) – це об'єктно-орієнтована служба зберігання даних, що пропонується Amazon Web Services (AWS), яка поєднує в собі масштабованість, доступність, безпеку та продуктивність в одному пакеті для використання в різних сценаріях хмарного зберігання[74]. Ось деякі з ключових особливостей та переваг S3:

1. Масштабованість: Amazon S3 пропонує неперевершену масштабованість для зберігання та обробки необмежених обсягів даних, що робить його ідеальним для обробки великих обсягів даних.

2. Висока доступність та надійність: S3 забезпечує неперевершену доступність і надійність даних завдяки автоматичній реплікації даних між кількома фізичними центрами обробки даних.

3. Безпека: Amazon S3 пропонує розширені функції безпеки, такі як шифрування даних, контроль доступу на основі ролей і політик, а також інтеграцію з AWS Identity and Access Management (IAM)[75].

4. Гнучкість управління даними: S3 надає можливості управління життєвим циклом даних, які дозволяють безперешкодно мігрувати дані між класами сховищ для оптимізації витрат і продуктивності.

5. Інтеграція з іншими сервісами AWS: S3 безперешкодно працює з декількома сервісами AWS, включаючи Amazon EC2, Glacier для довгострокового архівування та AWS Lambda для виконання коду у відповідь на події в S3[76].

6. Спрощений веб-хостинг: S3 може забезпечити ефективний, економічно вигідний спосіб розміщення статичних веб-сайтів, роблячи доставку веб-контенту простою та ефективною.

7. Економічні переваги Amazon S3: Amazon S3 пропонує недороге сховище з оплатою по мірі використання, що робить його економічним рішенням як для особистих проектів, так і для корпоративних додатків.

Amazon S3 – це ефективне рішення для зберігання великої кількості довговічних даних, яке чудово підійде для розробленої системи.

Amazon Kinesis Firehose – це повністю керований сервіс для збору, перетворення та завантаження потокових даних до AWS. Це один з основних елементів платформи Amazon Kinesis для обробки великих обсягів потокових даних в режимі реального часу[77]. Amazon Kinesis Firehose має чотири ключові особливості:

1. Простота використання: Amazon Kinesis Firehose забезпечує легкий збір даних без необхідності управління ресурсами. Просто створіть і налаштуйте потоки, використовуючи потоки Firehose з конкретними адресами джерела та призначення даних, щоб розпочати збір.

2. Автоматичне масштабування: Firehose автоматично масштабується відповідно до обсягу вхідних даних, дозволяючи обробляти великі потоки без втручання користувача.

3. Інтеграція з сервісами AWS: Amazon Kinesis Firehose може завантажувати дані безпосередньо в Amazon S3, Amazon Redshift, Elasticsearch Service і Splunk, забезпечуючи безперешкодну інтеграцію між потоковими даними та інструментами аналітики і сховищами даних[78].

4. Трансформація даних: Firehose надає можливості трансформації даних у реальному часі, які дозволяють змінювати формат даних, виконувати прості обчислення або фільтрувати інформацію перед завантаженням до місця призначення[79].

Таким чином Amazon Kinesis Firehose є оптимальним рішенням для трансформування та завантаження даних у сховище системи.

Останнє технічне рішення – спосіб отримання даних. Оскільки система збирає дані з різних джерел (публічні набори даних, соціальні мережі, майнінг) то оптимальним рішенням буде набір невеликих скраперів, кожен з яких витягуватиме дані з свого джерела. Для цього оптимальним буде застосування AWS Lambda сервісів, які в свою чергу віддаватимуть ці дані на швидкий та пакетний рівні за допомогою Amazon Kinesis Data Streams[80].

Обчислювальний сервіс AWS Lambda від Amazon Web Services (AWS) дозволяє запускати код без розгортання та керування серверами. Lambda масштабується автоматично, виконуючи код у відповідь на такі події, як HTTP-запити через Amazon API Gateway, доступ до файлів в Amazon S3, оновлення записів у базі даних тощо. Основні переваги AWS Lambda:

1. Безсерверне виконання: Lambda надає можливості безсерверного виконання, щоб ви могли запускати код без керування серверами або кластерами - AWS піклується про виділення достатньої кількості обчислювальних ресурсів для ефективного запуску вашого коду.

2. Керована подіями: Lambda можна налаштувати на виконання коду у відповідь на такі події, як зміни в сховищі даних, оновлення статусу та HTTP-запити - навіть без ручного втручання!

3. Масштабованість: Lambda автоматично масштабується для обробки декількох запитів одночасно, забезпечуючи обчислювальну потужність на основі вимог коду.

4. Безперешкодна робота з іншими сервісами AWS: Lambda легко інтегрується з такими сервісами AWS, як S3, DynamoDB, SNS, SQS, Kinesis CloudWatch та API Gateway для створення масштабованих та гнучких додатків.

Таким чином AWS Lambda є простим та швидким рішенням поставленої задачі зі збору даних. Після чого ці дані відправляться в стореджі за допомогою Amazon Kinesis Data Streams.

Amazon Kinesis Data Streams на AWS - це хмарний сервіс, розроблений спеціально для збору та обробки великих обсягів потокових даних у режимі реального часу, що дозволяє розробникам легко збирати, обробляти, аналізувати та складати звіти про вхідну потокову інформацію, зберігаючи при цьому високу пропускну здатність і масштабованість. Основні особливості та переваги Amazon Kinesis Data Streams:

1. Масштабованість: Kinesis Data Streams може обробляти величезні обсяги потокових даних з тисяч джерел у режимі реального часу, автоматично масштабуючись відповідно до вимог обробки.

2. Низька затримка забезпечує обробку та аналіз даних з низькою затримкою, що важливо для додатків, які вимагають швидкої реакції на потокові потоки даних.

3. Надійність та відмовостійкість: Kinesis Data Streams забезпечує неперевершену доступність та відмовостійкість, зберігаючи дані протягом годин або днів (залежно від конфігурації) та забезпечуючи відновлення даних у разі збою[81].

4. Легка інтеграція: сервіс легко інтегрується з іншими сервісами AWS, такими як Amazon S3, Redshift, Elasticsearch Service та Lambda, що дозволяє швидко та легко створювати комплексні аналітичні рішення.

5. Гнучкість обробки даних: розробники можуть використовувати стандартні бібліотеки та інструменти, такі як Kinesis Client Library (KCL), для обробки потоків даних, а також писати власні додатки з використанням SQL або інших мов програмування для аналітичної обробки.

7. Реалізація в режимі реального часу: ця послуга ідеально підходить для рішень, які вимагають аналізу даних в режимі реального часу, таких як моніторинг додатків, аналіз журналів і обробка фінансових транзакцій.

Таким чином, завершено розробку архітектури для проектованої системи. Результат представлено у вигляді діаграми контейнерів на рис. 4.3.

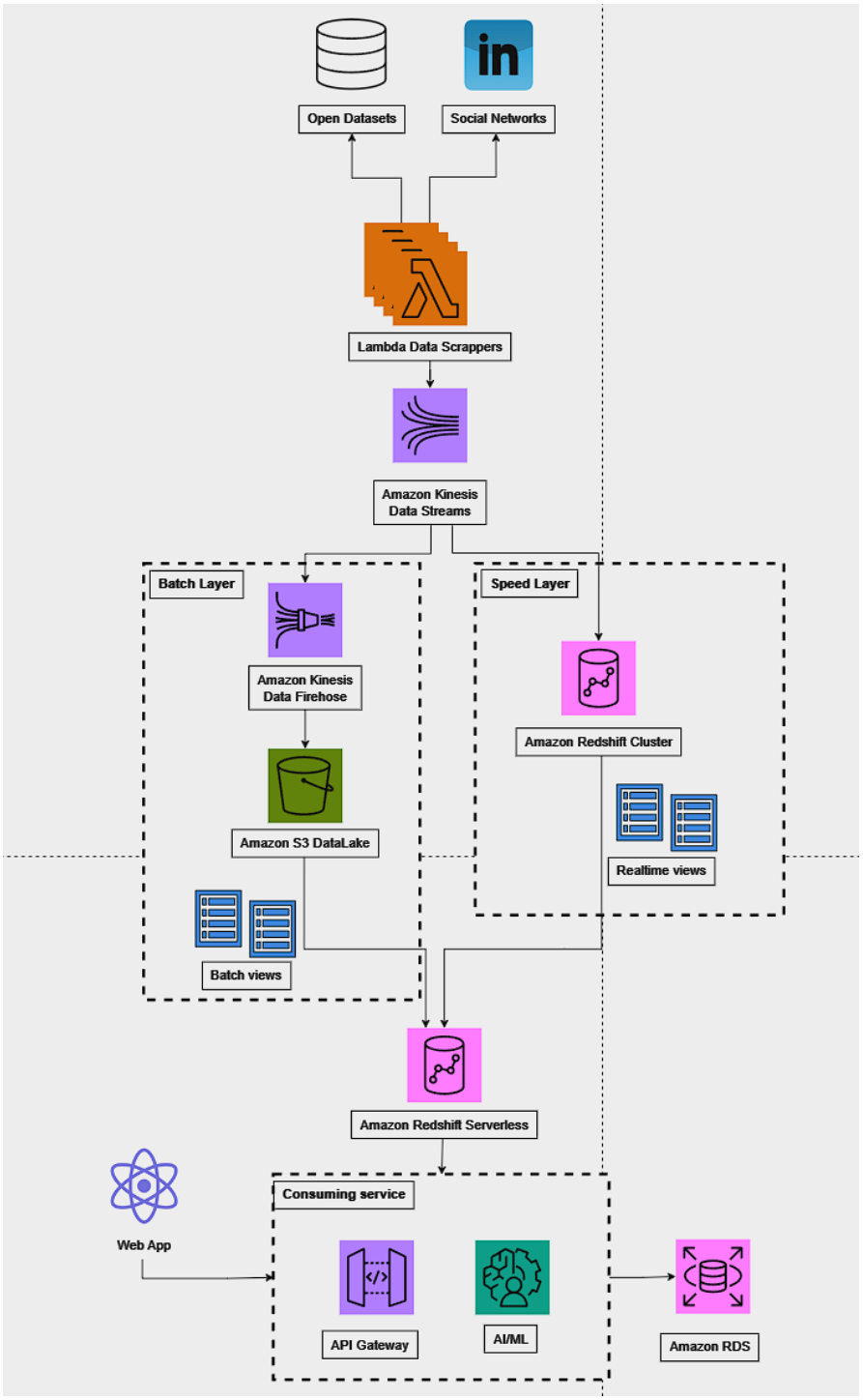


Рис. 4.3. Діаграма контейнерів

4.2. Розробка прикладного інтерфейсу

Для розробки прикладного інтерфейсу було обрано прототипування, яке передбачає створення робочого прототипу на основі намальованих макетів або каркасів і дозволяє відразу візуалізувати як варіанти використання, так і інтерфейс програми.

Прототипи слугують проміжним етапом на шляху до розробки високоякісних інтерфейсів кінцевих продуктів. Розробники повинні використовувати прототипи як спосіб полегшення виконання різних завдань, вивчення того, як користувачі сприймають контент і взаємодіють з інтерфейсом, тестування взаємодії користувачів з інтерфейсом шляхом імітації кінцевого продукту.

Прототипування використовується для імітації кінцевої взаємодії між користувачем та інтерфейсом. Хоча прототипи можуть мати незначні відмінності від кінцевого продукту, вони все одно повинні відображати основну функціональність програми або пунктів меню для користувача. Взаємодії повинні бути точно змодельовані, щоб максимально нагадувати те, що в кінцевому підсумку стане частиною кінцевого функціоналу інтерфейсу – незалежність між фронтендом і бекендом часто не береться до уваги з метою скорочення витрат і прискорення циклів розробки.

Для створення інтерфейсу для розробки мобільних додатків можна обійти етап створення макету, оскільки його дизайн не вимагає складного розміщення кнопок/меню, а наявного функціоналу достатньо для створення прототипу інтерфейсу одразу.

Прототипи дають можливість протестувати майбутній функціонал і користувальницький інтерфейс, а також встановити зв'язки між екранами вашого додатку і кнопками, що відповідають за зміну його стану. Тому важливо побудувати правильний прототип та потенційні варіанти взаємодії користувача з ним (Рис. 4.4.)

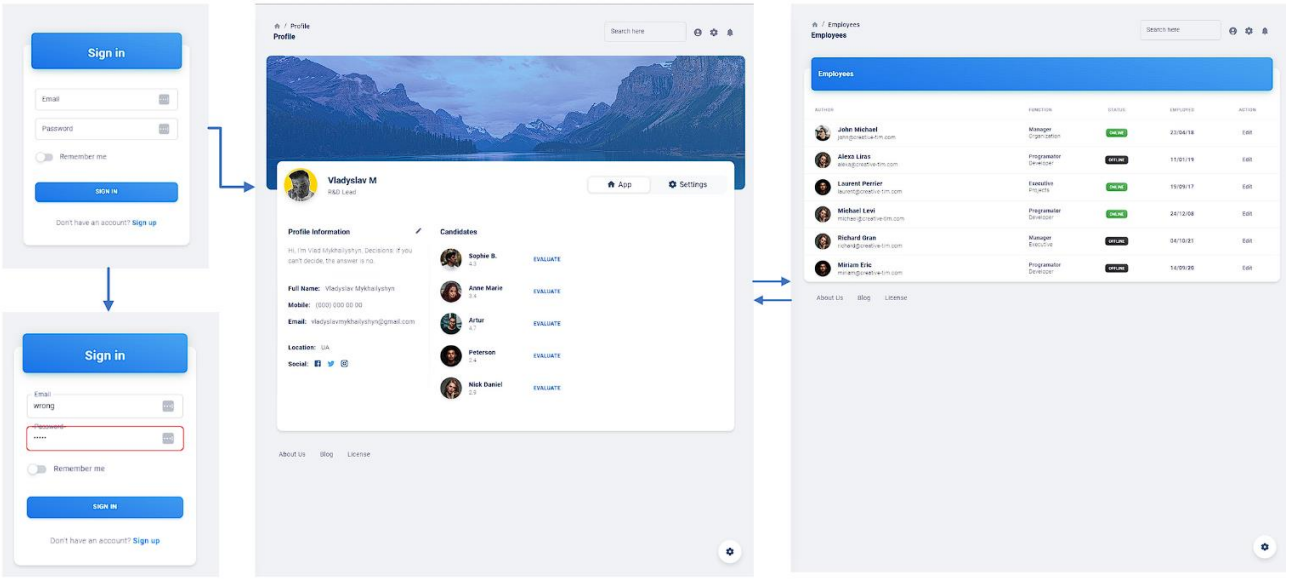


Рис. 4.4. Прототип системи та варіанти переходу між сторінками

На Рис. 4.4. можна побачити дизайн прототипи усіх основних сторінок та елементів інтерфейсу користувача – сторінки входу, портал користувача, сторінка перегляду кандидатів.

4.3. Розробка функціоналу

Отже розроблювана система складається з трьох основних компонент, які потрібно імплементувати:

- веб-додаток для користувача;
- бекенд API системи підтримки прийняття рішень;
- Лямбда сервіси для скрапінгу рекрутингових даних.

Для розробки веб-додатку було обрано технологію React – це фреймворк JavaScript з відкритим вихідним кодом, розроблений Facebook, який широко використовується для створення користувацьких інтерфейсів для односторінкових додатків[54]. Ключові переваги React:

1. Декларативний підхід: React дозволяє розробникам легко створювати інтерактивні інтерфейси, використовуючи декларативний підхід, що робить код більш читабельним і легшим для налагодження[82].

2. Компонентний підхід: React використовує компонентний підхід до розробки, що дозволяє розбивати інтерфейс користувача на незалежні, багаторазові компоненти для полегшення управління кодом і збільшення його повторного використання[83].

3. React унікальний своєю здатністю створювати полегшену копію структури даних DOM у браузерах, відому як віртуальний DOM, що дозволяє React ефективно оновлювати вузли, мінімізуючи при цьому накладні витрати на продуктивність та покращуючи швидкість роботи додатків.

4. Підтримка JSX: JSX – це синтаксис, який дозволяє розробникам писати HTML-структури безпосередньо в JavaScript-коді, роблячи структури компонентів більш зрозумілими та спрощуючи процеси проектування інтерфейсу користувача.

5. Широка спільнота та екосистема: React є одним з найпопулярніших JavaScript-фреймворків і може похвалитися великою спільнотою розробників з постійно зростаючим набором інструментів і бібліотек, які продовжують розвиватися з часом[84].

6. React є адаптивним та сумісним та може бути легко інтегрований у проекти, починаючи від невеликих додатків для мобільних телефонів і закінчуючи масштабними корпоративними програмними системами. React також добре інтегрується з різними бібліотеками та фреймворками, такими як Redux для державного управління[85].

7. Підтримка односторінкових додатків (SPA): React чудово підходить для створення односторінкових додатків (SPA), де інформацією можна маніпулювати та відображати без перезавантаження сторінки.

Усі ці переваги є причинами зробленого вибору стосовно технології для веб-додатку.

Для реалізації лямбда скраперів та API було обрано технологію Python.

Ось ключові переваги, пов'язаних з використанням Python:

1. Легкість у вивченні та читанні: простий синтаксис мови Python робить її доступною для програмістів-початківців, що робить вивчення мови легким і зрозумілим[86].

2) Велика спільнота: Python має одну з найбільших спільнот розробників, з широкою підтримкою, документацією, навчальними посібниками та дискусійними форумами.

3) Широкі бібліотеки та фреймворки: Python має безліч бібліотек, таких як NumPy, Pandas, Scikit-learn TensorFlow PyTorch, які є важливими інструментами для аналізу даних, машинного навчання, глибокого навчання та обробки природної мови[60].

4. Гнучкість Python полягає у підтримці багатьох парадигм програмування, включаючи об'єктно-орієнтоване, функціональне та процедурне програмування.

5. Інтеграція з іншими мовами програмування: гнучкість Python робить її ідеальною для використання в складних системах і може бути легко інтегрована з іншими мовами програмування, що робить її легкою і простою у використанні.

6. Відкритий вихідний код: Python - це мова з відкритим вихідним кодом, що дає розробникам свободу вносити будь-які необхідні модифікації та адаптувати її до своїх індивідуальних потреб.

Python вважається однією з найкращих мов для проектів зі штучного інтелекту та машинного навчання завдяки цим перевагам та підтримує об’єктно-орієнтоване проектування[87].

Об'єктно-орієнтоване проектування (ООП) має численні переваги, які роблять його популярним підходом у розробці програмного забезпечення. Однією з головних переваг є модульність, яка дозволяє розділити систему на окремі об'єкти або класи, кожен з яких відповідає за конкретну частину функціональності. Це

значно спрощує розробку, тестування та підтримку коду, оскільки зміни в одному модулі не впливають на інші.

Інкапсуляція, ще одна ключова концепція ООП, дозволяє приховувати внутрішню реалізацію об'єкта та надавати лише необхідний інтерфейс для взаємодії з ним. Це забезпечує захист даних та зменшує ризик випадкових помилок.

Наслідування сприяє повторному використанню коду, дозволяючи створювати нові класи на основі існуючих. Це спрощує розширення функціональності без необхідності переписування великої кількості коду. Поліморфізм, в свою чергу, дозволяє використовувати один інтерфейс для різних об'єктів, що спрощує взаємодію між компонентами системи[88].

Об'єктно-орієнтоване проектування також сприяє кращій відповідності реальним моделям, оскільки об'єкти можуть безпосередньо відображати реальні сутності та їх взаємодію. Це полегшує розуміння і моделювання складних систем. Усе це робить ООП потужним інструментом для створення масштабованих, гнучких та легко підтримуваних програмних систем[89].

Розглянемо імплементацію лямбда скраперів для збирання даних з різних джерел. Лямбда підхід дозволяє реалізувати ці задачі за допомогою Serverless підходу[90], що дозволяє не витрачати зусилля на підтримування та обслуговування інфраструктури а перекласти ці задачі на клауд провайдера.

Таким чином написаний код запускатиметься та масштабуватиметься згідно налаштувань інфраструктурою провайдера, а не власною.

Оскільки джерел даних у нас кілька то потрібно розробити кілька різних скраперів для них.

Перший – скрапер резюме в текстових форматах (Рис. 4.5.) з різних платформ та відкритих джерел.

Другий – скрапінг тезисних текстових даних з профільних платформ та соціальних мереж (Рис. 4.6., Рис. 4.7.)

Третій – витягнення даних з профільних ресурсів за допомого API в структурованому, JSON, або XML форматі (Рис. 4.8.)

Solution Architect, Competency Manager

About

Contacts

LinkedIn Profile | Telegram IM |

Education

Zhytomyr State Technological University '04–'08.

Industrial Equipment Control System Automatization, incomplete.

Languages

English — C1 | Ukrainian — Native

Summary

As a developer with over 12 years of experience, I have had the opportunity to work on a wide variety of projects, from dashboards to trading tools to job boards. I am always excited to take on new challenges and to continue learning and growing as a developer.

As a public tech speaker and content creator, I am also passionate about mentoring and sharing my knowledge with others. I believe that by working together and learning from each other, we can achieve wonderful things and make a positive impact in the world of web development.

So, whether you are looking for a skilled developer to help bring your project to life or a mentor to guide your team to success, I am here and ready to help!

Skills

Web application design and development; UI components development; Complex data processing services development; Full development cycle experience; New functionality implementation; Project management and collaboration; Stakeholder communication; Technical and non-technical communication.

Technologies

Frameworks & libraries

ReactJS; RecoilJS; Redux; Material UI; Angular 2+; RxJS; NGRX; Angular Material; Antd; Bulma CSS; VueJS; MobX; D3; KnockoutJS; RactiveJS;

Core technologies & concepts

JavaScript (ES5+); Typescript; HTML; CSS; SVG; Browser API; PWA; NodeJS; REST API; Databases;

OOP design; Functional programming; Immutability;

Agile/Scrum;

Tools

VS Code; Git; Jira.

Рис. 4.5. Приклад резюме в текстовому форматі

Рис. 4.5. показує приклад резюме кандидата, яке з однієї сторони є прикладом якісних даних, бо містить, як правило, лише необхідну інформацію, але з іншої сторони довільний формат кожного резюме робить процес його опрацювання досить складним.

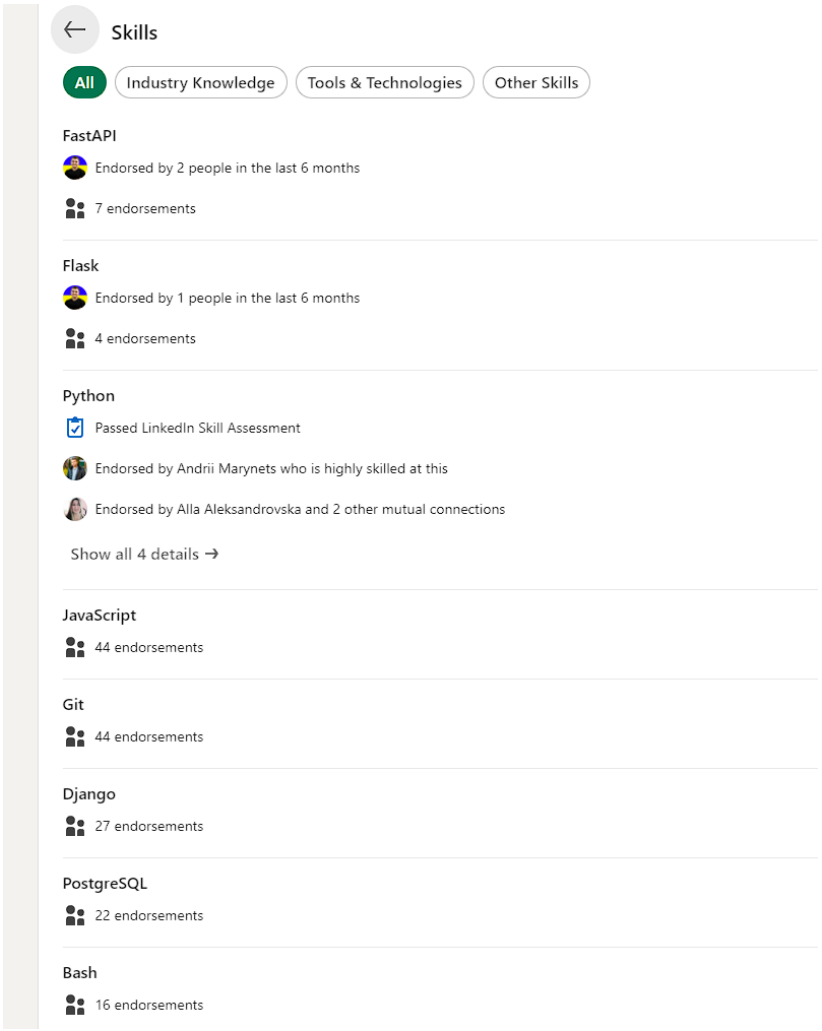


Рис. 4.6. Приклад окремих тезисних даних для скрапінгу

На Рис.4.6. наведено приклад сторінки кандидата в LinkedIn[91], з якої за допомогою скраперів ми можемо отримати тезисну інформацію про технології та інструменти з якими працював кандидат.

MANUAL AND AUTOMATIC JUNIOR QA ENGINEER

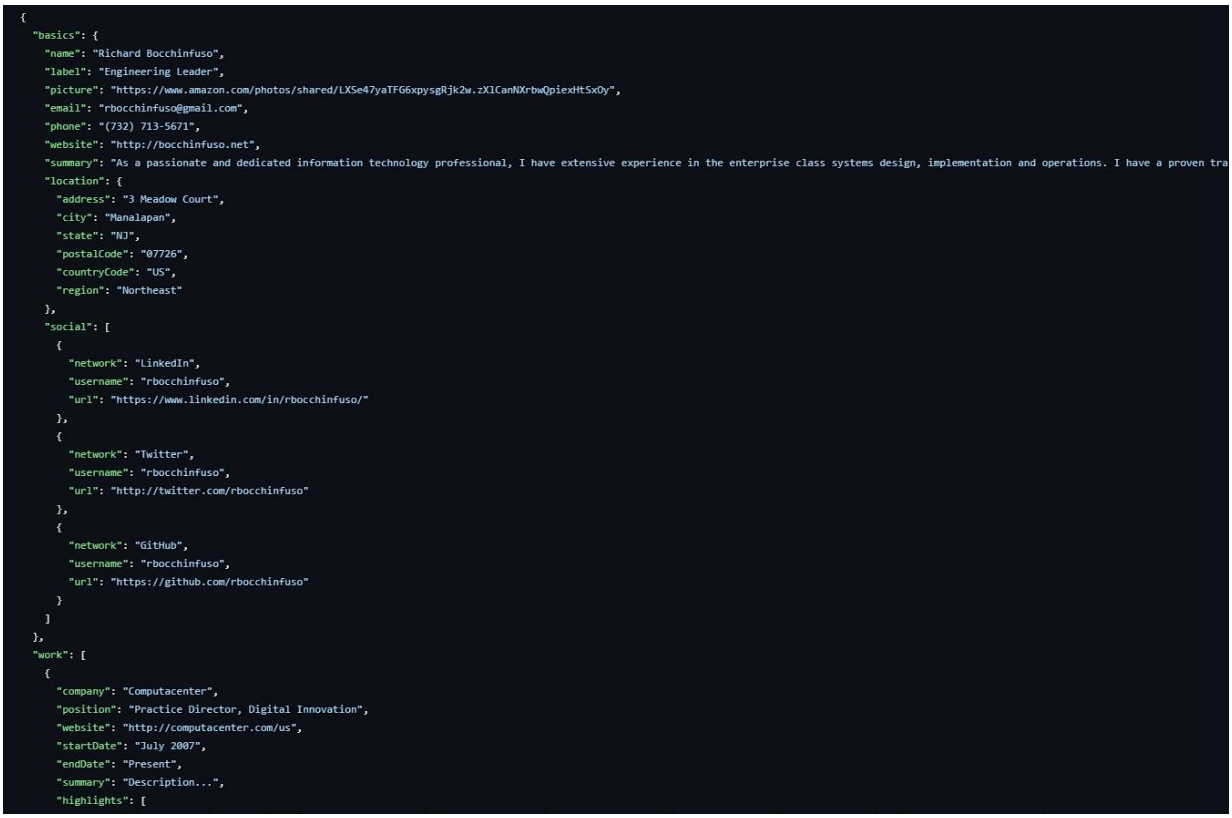
SUMMARY
I am a manual and automatic junior QA engineer. I understand what QA is, how it works, what are the goals of QA and what is the role of QA in IT projects. I know the basics of testing, test documentation, test methods, test tools. My experience in my previous job instilled in me a deep sense of responsibility. I am sure that my knowledge
and skills will be useful for your team and working in your company will be the next step
in developing my skills.

PROJECT EXPERIENCE
CONTACTS
Blynk IOT Postman
Automatic testing
[відкрити контакти] Role - manual QA
Testing API
[відкрити контакти] Hardware testing
Telegram Web console & mobile testing Postman + New Man
Linkedin API testing Automatic testing with CL
Kiyv Testing API
Project Wallet
Cypress and CI technology
TECH SKILLS: Role - Team Lead
Automatic testing
Test planning
Testing the "login - password"
Scrum methodology Testing Login-Password form form
Swagger. Back-end test Cypress and API
Test documentation
Negative testing, Security testing Automatic testing
Fundamentals of Testing
Testing API weatherstack.com.
Types of testing Privat24.ua QA (Manual)
Cypress and Cucumber
Test management Compiling Main Use cases
Automatic testing
Writing Test Cases
Monitoring and control testing BDD approach
Smoke testing
Mobile Testing Negative testing Store testing

Рис. 4.7. Приклад резюме на ресурсі Work.ua

На Рис.4.7. наведено приклад сторінки кандидата на платформі Work.ua, інформація з якої також буде зібрана за допомогою скраперів.

Ще одним варіантом отримання даних з мережі інтернет є формат Json[92] (Рис. 4.8.)



```
{
  "basics": {
    "name": "Richard Bocchinfuso",
    "label": "Engineering Leader",
    "picture": "https://www.amazon.com/photos/shared/LXSe47yaTFG6xpysgRjk2w.zX1CanNXrbwQpiexHtSxOy",
    "email": "rbocchinfuso@gmail.com",
    "phone": "(732) 713-5671",
    "website": "http://bocchinfuso.net",
    "summary": "As a passionate and dedicated information technology professional, I have extensive experience in the enterprise class systems design, implementation and operations. I have a proven tra
    "location": {
      "address": "3 Meadow Court",
      "city": "Manalapan",
      "state": "NJ",
      "postalCode": "07726",
      "countryCode": "US",
      "region": "Northeast"
    },
    "social": [
      {
        "network": "LinkedIn",
        "username": "rbocchinfuso",
        "url": "https://www.linkedin.com/in/rbocchinfuso/"
      },
      {
        "network": "Twitter",
        "username": "rbocchinfuso",
        "url": "http://twitter.com/rbocchinfuso"
      },
      {
        "network": "GitHub",
        "username": "rbocchinfuso",
        "url": "https://github.com/rbocchinfuso"
      }
    ]
  },
  "work": [
    {
      "company": "Computacenter",
      "position": "Practice Director, Digital Innovation",
      "website": "http://computacenter.com/us",
      "startDate": "July 2007",
      "endDate": "Present",
      "summary": "Description...",
      "highlights": [
```


Рис. 4.8. Приклад даних резюме в JSON форматі

Такий формат даних, як на Рис. 4.8. передбачає отримання даних в форматі ключ-значення, де зеленим на зображенні підсвічені ключі, які стануть окремими властивостями в процесі обробки, а синім значення цих ключів, які система використовуватиме для проведення оцінки.

Такі різні засоби для отримання даних дозволяють максимально ефективно наповнювати базу унікальними записами, що дозволить збільшити об’єми аналізованої інформації та підвищити точність роботи системи.

Також, такий підхід до збирання даних, дозволяє не тримати процеси запущені постійно, а виконувати операції збору даних з певною періодичністю. Це дозволяє не використовувати зайві обчислювальні ресурси постійно, а отримувати їх тоді, коли скрапери запущені та працюють (Рис. 4.9)

Рис. 4.9. Схема роботи скраперів

З Рис.4.10 та Рис.4.11 видно, що для ресурсів встановлено обмеження, при досягненні яких кількість скраперів може масштабуватись для вирішення проблем з навантаженням на кожен окремий процес[93].

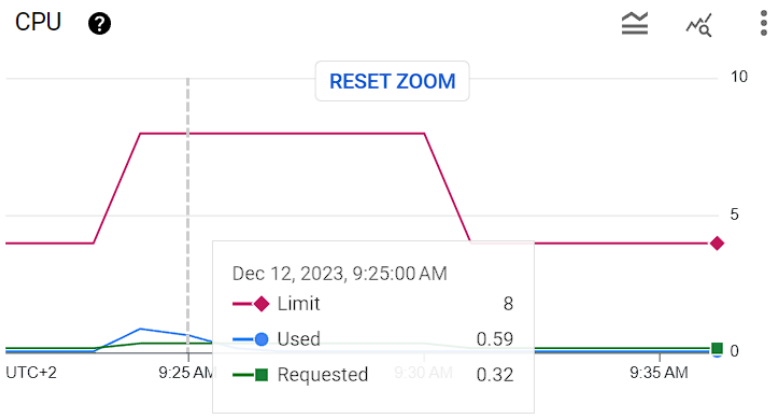


Рис. 4.10. Споживання ресурсів процесора скраперами

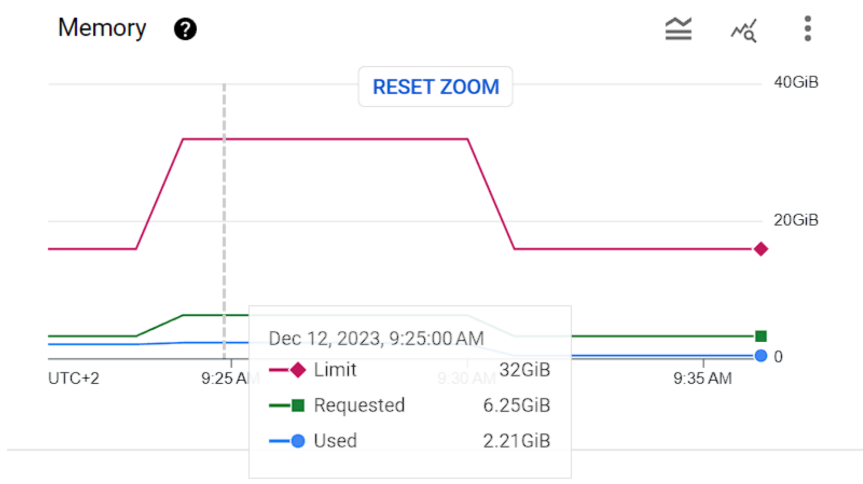


Рис. 4.11. Споживання оперативної пам’яті скраперами

Як показано на Рис.4.10. та Рис.4.11. споживання ресурсів (процесора та оперативної пам’яті) має певні пікові періоди, коли відбувається скрапинг нового джерела інформації, або відбувається отримання аномально великого об’єму даних. Проте ці періоди є стабільними і не перевищують встановлений ліміт споживання пам’яті.

Наступним етапом є розроблення сховища даних для довготривалого зберігання інформації. Оскільки дані неоднорідні, планується подальший розвиток системи. Для цього потрібне сховище, яке підтримує можливість розширення структури даних, таке як Amazon S3[94]. Тепер потрібно визначити модель зберігання даних. Виходячи зі способів використання даних, потрібно зберігати всю інформацію на кореневому рівні документу, що дозволить легко індексувати потрібні поля для пришвидшення роботи з ними (Рис. 4.12.)

```
{
    "_id": "e2bdd63f-f8e9-474a-97f8-f6a49132feb7",
    "score": 4.2,
    "resume": "Data Science,'Skills  Programming Languages: Python (pandas, numpy
    "tokens": [ "data science", "data", "python", "pandas", "numpy", "scipy", "ba
    "tf-idf": [0,0,0,0.02206973,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0.023
    "class": "datascience",
    "cluster_id": "b2efb211-ea72-4bf7-bbdb-860871abbc2d",
}

```

Рис. 4.12. Структура даних в сховищі даних

Останнім етапом є розробка API та системи підтримки та прийняття рішень.

Потрібно реалізувати та натренувати моделі для кластеризації та регресії XGBoost.

Для реалізації було обрано програмні засоби бібліотеки sklearn, вона надає зручний та простий інтерфейс для тренування та застосування моделей[95]. Зокрема Рис.4.13 демонструє використання цієї бібліотеки для тренування моделі кластеризації K-medians[96].

```
# K-medians clustering
1 usage
def k_medians(data, k, max_iter=100):
    # Randomly initialize k centroids from the data points
    centroids = data[np.random.choice(data.shape[0], k, replace=False), :]

    for i in range(max_iter):
        # Compute distances between data points and centroids
        dist_matrix = distance.cdist(data, centroids, metric: 'cityblock')  # Using Manhattan d
        # Assign clusters based on closest centroid
        cluster_labels = np.argmin(dist_matrix, axis=1)

        # Recompute centroids as the median of assigned points
        new_centroids = np.array([np.median(data[cluster_labels == j], axis=0) for j in range
```

Рис. 4.13. Застосування бібліотеки sklearn

Важливо також порівняти розроблену систему з аналогами, які існують на ринку. Це дасть змогу виділити ключові особливості розробленої системи підтримки та прийняття рішень та краще зрозуміти її переваги і недоліки (Таблиця 4.1.)

Таблиця. 4.1. Огляд аналогів

Назва системи	Переваги	Недоліки
Manatal[97]	1. Простота використання: інтуїтивно зрозумілий інтерфейс, що полегшує роботу рекрутерів.	1. Обмежені можливості кастомізації: деякі користувачі зазначають, що

Продовження Таблиці 4.1

Назва системи	Переваги	Недоліки
	2. Штучний інтелект: використання AI для покращення пошуку та відповідності кандидатів. 3. Інтеграції: легка інтеграція з іншими сервісами, такими як LinkedIn, Gmail, і календарі. 4. Ціноутворення: конкурентоспроможні ціни з різними планами, що підходять для різних розмірів компаній.	платформа обмежена в налаштуваннях. 2. Підтримка: відгуки щодо служби підтримки змішані, з деякими повідомленнями про затримки у відповідях.
Celential.ai[98]	1. Штучний інтелект: потужний AI для пошуку та оцінки кандидатів. 2. Автоматизація: автоматизовані процеси, що знижують ручну працю та підвищують ефективність. 3. Якість кандидатів: висока точність у підборі кандидатів, що відповідають вимогам вакансій.	1. Обмежена інтеграція: менше інтеграцій з іншими платформами порівняно з конкурентами. 2. Вартість: високий ціновий діапазон, що може бути непідходящим для малих компаній.
Hiration[99]	1. Резюме та профілі: потужні інструменти для створення резюме та оптимізації профілів кандидатів.	1. Обмежені функції рекрутингу: менше функціоналу для рекрутерів порівняно з іншими ATS системами.

Продовження Таблиці 4.1

Назва системи	Переваги	Недоліки
	2. Персоналізація: можливість створювати персоналізовані рекомендації для кандидатів. 3. Інтуїтивний інтерфейс: зручний у використанні інтерфейс, який допомагає користувачам швидко освоїтись.	2. Інтеграції: обмежені можливості інтеграції з іншими рекрутинговими платформами.
Jobylon[100]	1. Інтуїтивний інтерфейс: Простий у використанні інтерфейс, що полегшує роботу для рекрутерів. 2. Кооперація та комунікація: Інструменти для покращення співпраці в команді та комунікації з кандидатами. 3. Бренд роботодавця: Можливість налаштування сторінок вакансій у відповідності до бренду компанії.	1. Ціноутворення: Висока вартість може бути недоступною для малих компаній. 2. Інтеграції: Менше інтеграцій з іншими платформами порівняно з деякими конкурентами.
Zoho Recruit[101]	1. Повнота функцій: Широкий набір функцій для управління процесом рекрутингу від початку до кінця. 2. Інтеграція: Легка інтеграція з іншими продуктами Zoho, а також зовнішніми сервісами.	1. Крива навчання: Деякі користувачі зазначають, що потрібен час для освоєння всіх функцій платформи.

Продовження Таблиці 4.1

Назва системи	Переваги	Недоліки
	3. Автоматизація: Інструменти для автоматизації багатьох аспектів рекрутингу.	2. Підтримка: Змішані відгуки щодо якості та швидкості підтримки клієнтів.

Проаналізувавши системи схожої тематики та направленості (Таблиця 4.1.), можна зробити висновок про відсутність аналогів, які використовують публічні дані ринку праці та дані конкретних компаній для надання рекомендацій рекрутингової діяльності. Водночас, на ринку присутні системи, які надають рекомендації на основі одного з описаних джерел даних, або з використанням інших підходів. Такі системи є потенційними конкурентами, однак їхнім суттєвим недоліком буде відсутність спеціалізації для конкретної компанії, яка проводить рекрутинг.

4.4. Апробація результатів

Розглянемо готовий веб-додаток прикладного інтерфейсу.

При відкритті веб додатка система попросить користувача авторизуватись ввівши коректні логін і пароль (Рис. 4.14.)

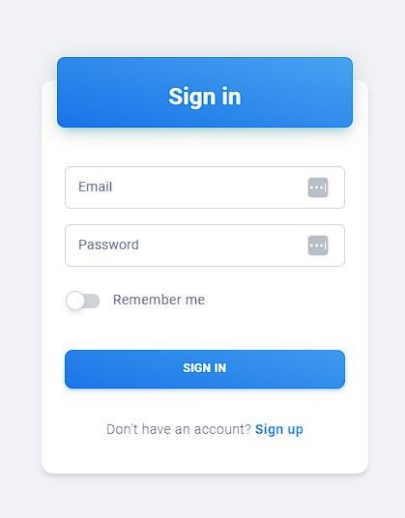


Рис. 4.14. Екран авторизації

При неправильно введеному паролі або логіну користувач не зможе авторизуватись, а застосунок підсвітить поле червоним кольором (Рис. 4.15.)

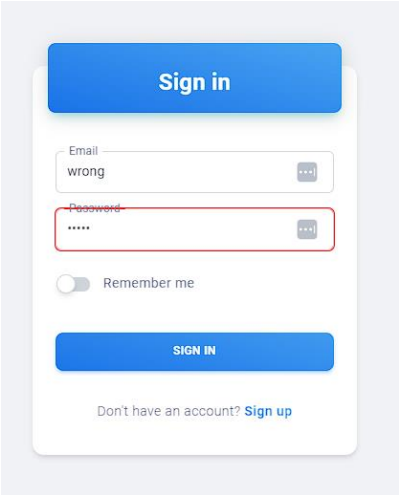


Рис. 4.15. Невірний пароль

При успішній авторизації користувач буде перенаправлений на головну сторінку програми (Рис. 4.16), на якій він зможе переглянути кандидатів, їх оцінку системою. Також є можливість додавати нових кандидатів, завантаживши їх резюме (Рис. 4.16), яке буде автоматично опрацьовуватись.

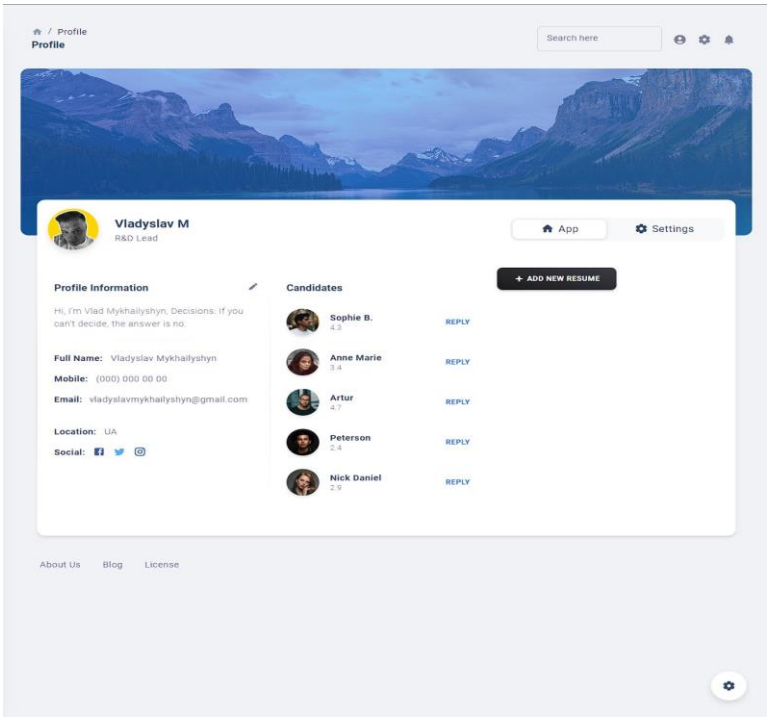


Рис. 4.16. Головна сторінка додатку

Для додавання нового кандидата вручну необхідно натиснути кнопку «Add new resume» після чого з’явиться вікно завантаження резюме (Рис. 4.17.), в якому потрібно обрати необхідний документ у форматі pdf/docx.

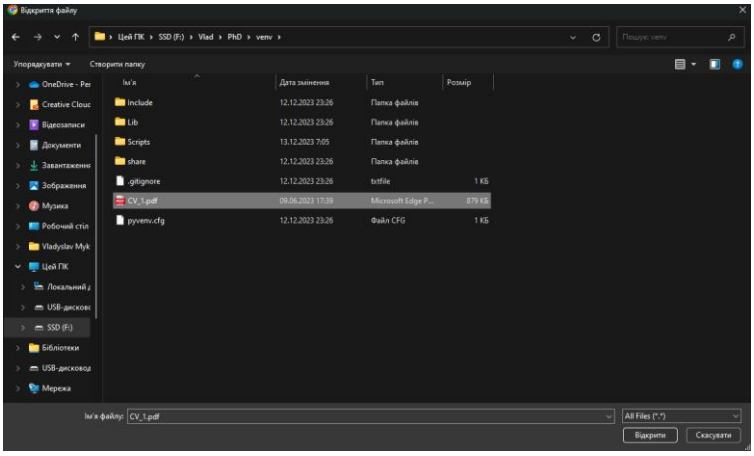
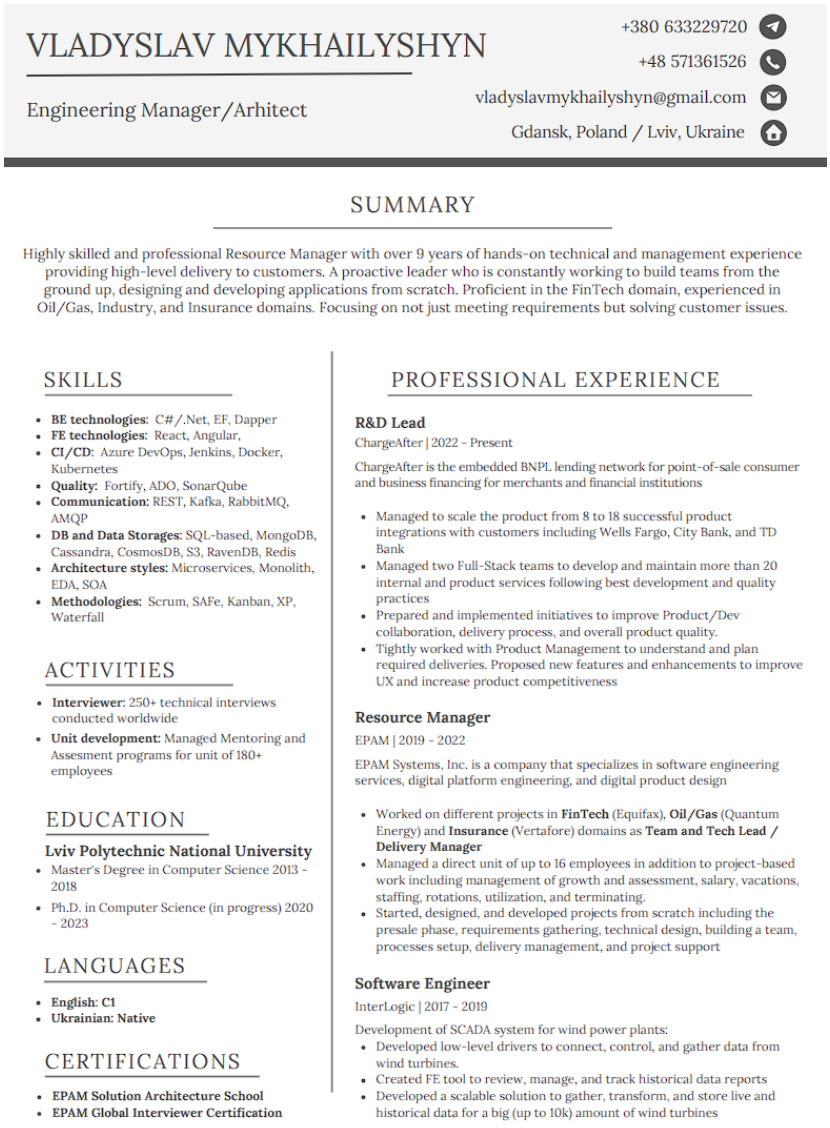


Рис. 4.17. Завантаження резюме

Після вибору файлу, резюме (Рис. 4.18.) буде завантажено в додаток та відправлено на опрацювання (Рис. 4.19.)



VLADYSLAV MYKHAILYSHYN

Engineering Manager/Arhitect

+380 633229720
+48 571361526
vladyslavmykhailyshyn@gmail.com
Gdansk, Poland / Lviv, Ukraine

SUMMARY

Highly skilled and professional Resource Manager with over 9 years of hands-on technical and management experience providing high-level delivery to customers. A proactive leader who is constantly working to build teams from the ground up, designing and developing applications from scratch. Proficient in the FinTech domain, experienced in Oil/Gas, Industry, and Insurance domains. Focusing on not just meeting requirements but solving customer issues.

SKILLS

- **BE technologies:** C#/.Net, EF, Dapper
- **FE technologies:** React, Angular,
- **CI/CD:** Azure DevOps, Jenkins, Docker, Kubernetes
- **Quality:** Fortify, ADO, SonarQube
- **Communication:** REST, Kafka, RabbitMQ, AMQP
- **DB and Data Storages:** SQL-based, MongoDB, Cassandra, CosmosDB, S3, RavenDB, Redis
- **Architecture styles:** Microservices, Monolith, EDA, SOA
- **Methodologies:** Scrum, SAFe, Kanban, XP, Waterfall

ACTIVITIES

- **Interviewer:** 250+ technical interviews conducted worldwide
- **Unit development:** Managed Mentoring and Assesment programs for unit of 180+ employees

EDUCATION

Lviv Polytechnic National University

- Master's Degree in Computer Science 2013 - 2018
- Ph.D. in Computer Science (in progress) 2020 - 2023

LANGUAGES

- **English: C1**
- **Ukrainian: Native**

CERTIFICATIONS

- **EPAM Solution Architecture School**
- **EPAM Global Interviewer Certification**

PROFESSIONAL EXPERIENCE

R&D Lead

ChargeAfter | 2022 - Present

ChargeAfter is the embedded BNPL lending network for point-of-sale consumer and business financing for merchants and financial institutions

- Managed to scale the product from 8 to 18 successful product integrations with customers including Wells Fargo, City Bank, and TD Bank
- Managed two Full-Stack teams to develop and maintain more than 20 internal and product services following best development and quality practices
- Prepared and implemented initiatives to improve Product/Dev collaboration, delivery process, and overall product quality.
- Tightly worked with Product Management to understand and plan required deliveries. Proposed new features and enhancements to improve UX and increase product competitiveness

Resource Manager

EPAM | 2019 - 2022

EPAM Systems, Inc. is a company that specializes in software engineering services, digital platform engineering, and digital product design

- Worked on different projects in **FinTech** (Equifax), **Oil/Gas** (Quantum Energy) and **Insurance** (Vertafore) domains as **Team and Tech Lead / Delivery Manager**
- Managed a direct unit of up to 16 employees in addition to project-based work including management of growth and assessment, salary, vacations, staffing, rotations, utilization, and terminating.
- Started, designed, and developed projects from scratch including the presale phase, requirements gathering, technical design, building a team, processes setup, delivery management, and project support

Software Engineer

InterLogic | 2017 - 2019

Development of SCADA system for wind power plants:

- Developed low-level drivers to connect, control, and gather data from wind turbines.
- Created FE tool to review, manage, and track historical data reports
- Developed a scalable solution to gather, transform, and store live and historical data for a big (up to 10k) amount of wind turbines

Рис. 4.18. Резюме для завантаження

Опрацювання відбуватиметься в кілька етапів, проходячи процеси описані в попередніх розділах (токенізацію, кластеризацію, визначення оцінки СППР). Під час цих етапів відбувається визначення основних даних кандидата, кластеризація напрямків його діяльності та визначення оцінки на основі даних системи про кандидатів, формуючи результуючу оцінку для відображення в інтерфейсі користувача.

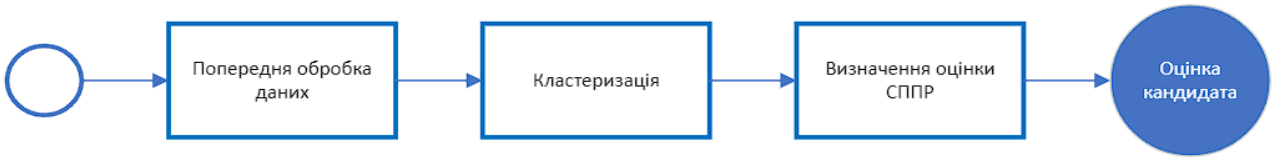


Рис. 4.19. Опрацювання резюме

Швидке обчислення оцінки для нового резюме відбувається практично непомітно користувачеві (~1 секунда) (Рис. 4.20)

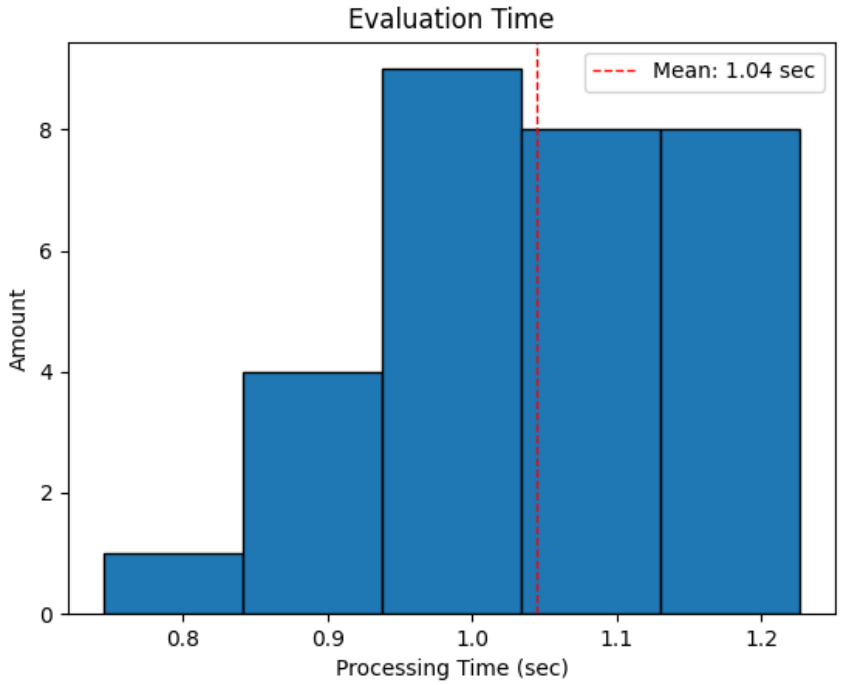


Рис. 4.20. Середня швидкість оцінювання нового кандидата

Після оцінювання нова кандидатура додається до списку кандидатів (Рис. 4.21.)

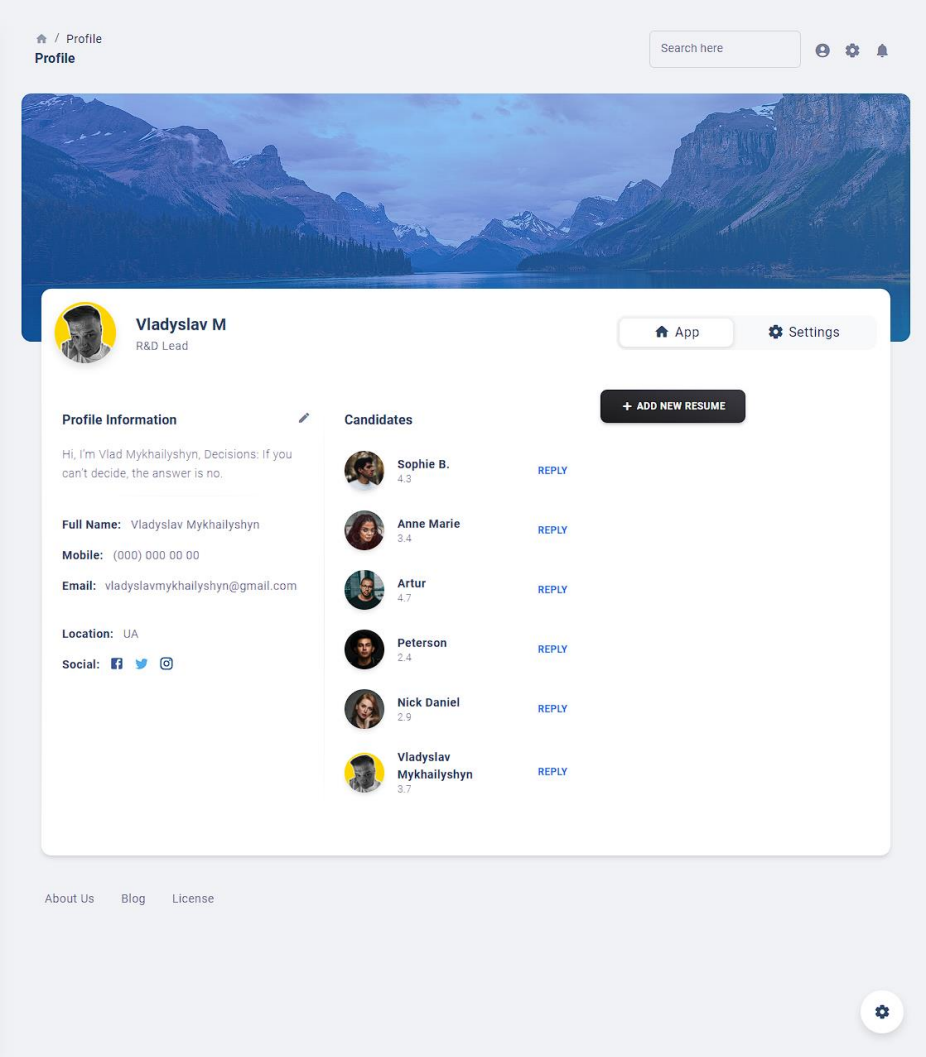


Рис. 4.21. Доданий новий кандидат

Як видно з Рис. 4.21, в результаті роботи системи, новий користувач добавився до системи та отримав свою оцінку. Тепер його дані доступні для перегляду та для порівняння з іншими кандидатурами.

4.5. Висновки

В даному розділі було розроблено архітектуру системи підтримки та прийняття рішень. При виборі архітектури були враховані основні варіанти використання

системи та вимоги до нефункціональних вимог стосовно швидкості опрацювання великих даних, в тому числі за рахунок застосування паралельних обчислень.

Було розроблено прототипи та втілений графічний інтерфейс користувача на їх основі.

Розроблено лямбда архітектуру для оптимізації використання обраних методів та засобів підготовки та опрацювання інформації рекрутингу.

У якості платформи для розгортання обрано AWS, як оптимальну за показниками надійності та співвідношення ціни та продуктивності для розробленої архітектури.

Розроблено основні сервіси системи, включаючи лямбда скрапери, сховище даних, веб-застосунок та API сервіс системи підтримки прийняття рішень.

Також представлено отримані результати роботи, за допомогою скріншотів показано роботу системи, додавання та оцінювання нового кандидата системою підтримки прийняття рішень.

ВИСНОВКИ

У дисертаційній роботі вирішено актуальну наукову задачу розроблення методів та засобів підтримки та прийняття рішень аналізу рекрутингової діяльності для забезпечення якісного процесу прийняття рекрутингових рішень.

1. Проведено аналіз методів опрацювання неоднорідних рекрутингових даних і природних мовних обробників для попередньої підготовки та автоматичної категоризації резюме. Досліджено використання методів опрацювання природньої мови таких як статистичний, лінгвістичний методи, метод умовних випадкових полів, токенізація та прихована марковська модель, TF-IDF.
2. Розроблено алгоритм для обробки даних, що включає попереднє опрацювання вхідної інформації з використанням аналізу природної мови та кластеризації з використанням ансамблю методів K-means, K-medians, DBSCAN, C-means і голосуванням на основі індексів Силуета та Данна.
3. Розроблено систему автоматичної рекомендації кандидатів на основі аналізу неоднорідних структурованих та неструктурованих даних про резюме та профілі кандидатів на основі методу регресії XGBoost.
4. Розроблено систему автоматичного аналізу та прогнозування зворотного зв'язку від кандидатів з метою постійного вдосконалення рекрутингового процесу, створено метод обрахування індексу впливу на оцінку, що дозволяє коригувати її в залежності від отриманого зворотнього зв’язку
5. Розроблено лямбда архітектуру системи і програмні модулі та апробувано їх застосування щодо підтримки прийняття верифікованих рішень на основі великих даних.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Shakhovska, N., Kaminskyy, R., Khudoba, B., Mykhailyshyn, V., Helzhynskyi, I. A Novel Methodology Analyzing the Influence of Micro-Stresses on Human-Centric Environments. Computation, 2023, 11(11), 224. https://doi.org/10.3390/computation11110224 (квартиль Q2 у НМБД Scopus).
2. Бойко Н. І., Михайлишин В. Ю. Алгоритм класифікації текстового контенту соціальних мереж для визначення емоційного тону. Вісник Херсонського національного технічного університету, № 2(85) (2023): с. 133-140. https://doi.org/10.35546/kntu2078-4481.2023.2.18
3. Бойко Н. І., Михайлишин В. Ю. Оцінка ефективності рекурсивного процесу розподілу набору даних з використанням алгоритму CART. Вісник Хмельницького національного університету. Серія: «Технічні науки», №4, 2023, с. 25-35. https://www.doi.org/10.31891/2307-5732-2023-323-4-25-35
4. Boyko N. I., Mykhailyshyn V. Yu. K-NN'S NEAREST NEIGHBORS METHOD FOR CLASSIFYING TEXT DOCUMENTS BY THEIR TOPICS. Радіоелектроніка, інформатика, управління. 2023. № 3. (Radio Electronics, Computer Science, Control. 2023. № 3) pp. 83-97. https://www.doi.org/10.15588/1607-3274-2023-3-9 (WOS)
5. Бойко Н. І., Шаховська Н. Б., Михайлишин В. Ю. Розроблення методу класифікації користувачів за рівнем стресостійкості з використанням модифікованої автоасоціативної нейронної мережі // Вісник Хмельницького національного університету, № 6, 2021 (303), с. 64-68. https://www.doi.org/10.31891/2307-5732-2021-303-6-64-68
6. Boyko N., Mykhailyshyn V. Methods of Searching for Associative Rules for Inhomogeneous Data in Semantic Networks. Proceedings of the 3rd International Workshop on Intelligent Information Technologies & Systems of Information Security Khmelnytskyi, Ukraine, March 23–25, 2022, pp. 54-71.

7. The impact of the COVID-19 pandemic on jobs and incomes in G20 economies. ILO-OECD paper prepared at the request of G20 Leaders. Saudi Arabia's G20 Presidency 2020
8. MITTAL, Vrinda, et al. Methodology for resume parsing and job domain prediction. Journal of Statistics and Management Systems, 2020, 23.7: 1265-1274.
9. WANG, Lidong. Heterogeneous data and big data analytics. Automatic Control and Information Sciences, 2017, 3.1: 8-15.
10. Fronza, I., Janes, A., Sillitti, A., Succi, G., & Trebeschi, S. (2013). Cooperation wordle using pre-attentive processing techniques. International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE), 6(1), 57–64.
11. HUNKENSCHROER, Anna Lena; LUETGE, Christoph. Ethics of AI-enabled recruiting and selection: A review and research agenda. Journal of Business Ethics, 2022, 178.4: 977-1007.
12. ALGHAMDI, Turki Ali; JAVAID, Nadeem. A survey of preprocessing methods used for analysis of big data originated from smart grids. IEEE Access, 2022, 10: 29149-29171.
13. MAHMUD, Mohammad Sultan, et al. A survey of data partitioning and sampling methods to support big data analysis. Big Data Mining and Analytics, 2020, 3.2: 85-101.
14. GARCÍA, Salvador, et al. Big data preprocessing: methods and prospects. Big Data Analytics, 2016, 1.1: 1-22.
15. PRAKASH, Andrea; NAVYA, Narem; NATARAJAN, Jayapandian. Big data preprocessing for modern world: opportunities and challenges. In: International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018. Springer International Publishing, 2019. p. 335-343.
16. CAI-MING, Zhang; HAO-NAN, Chu. Preprocessing method of structured big data in human resource archives database. In: 2020 IEEE International Conference on Industrial Application of Artificial Intelligence (IAAI). IEEE, 2020. p. 379-384.

17. SINGH, Dalwinder; SINGH, Birmohan. Investigating the impact of data normalization on classification performance. Applied Soft Computing, 2020, 97: 105524.
18. RAJU, VN Ganapathi, et al. Study the influence of normalization/transformation process on the accuracy of supervised classification. In: 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT). IEEE, 2020. p. 729-735.
19. SINGH, Dalwinder; SINGH, Birmohan. Investigating the impact of data normalization on classification performance. Applied Soft Computing, 2020, 97: 105524.
20. HUANG, Lei, et al. Normalization techniques in training dnns: Methodology, analysis and application. IEEE transactions on pattern analysis and machine intelligence, 2023, 45.8: 10173-10196.
21. MILLIDGE, Beren; SETH, Anil; BUCKLEY, Christopher L. Predictive coding: a theoretical and experimental review. arXiv preprint arXiv:2107.12979, 2021.
22. LI, Boyi, et al. On feature normalization and data augmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021. p. 12383-12392.
23. DU, Yuxuan; SUN, Fengzhu. HiCBin: binning metagenomic contigs and recovering metagenome-assembled genomes using Hi-C contact maps. Genome biology, 2022, 23.1: 63.
24. TAHTALI, Yalcin. Classification of Raw Milk Composition and Somatic Cell Count in Water Buffaloes with Support Vector Machines. KAFKAS ÜNİVERSİTESİ VETERİNER FAKÜLTESİ DERGİSİ, 2020, 26.4.
25. SINGH, Dalwinder; SINGH, Birmohan. Investigating the impact of data normalization on classification performance. Applied Soft Computing, 2020, 97: 105524.

26. GEWERS, Felipe L., et al. Principal component analysis: A natural approach to data exploration. ACM Computing Surveys (CSUR), 2021, 54.4: 1-34.
27. BOYKO, Nataliya. Models and Algorithms for Multimodal Data Processing. WSEAS Transactions on Information Science and Applications, 2023, 20: 87-97.
28. BOYKO, Nataliya. Tools for Implementing the Models and Algorithms for Processing Multimodal Data. 2023.
29. BOYKO, Nataliya; MYKHAILYSHYN, Vladyslav. Model of Finding Associative Rules in Inhomogeneous Data of Semantic Networks. In: CMIS. 2022. p. 58-67.
30. SHAKHOVSKA, Nataliya, et al. Big data processing technologies in distributed information systems. Procedia Computer Science, 2019, 160: 561-566.
31. RAHMAN, Azizur. Statistics-based data preprocessing methods and machine learning algorithms for big data analysis. International Journal of Artificial Intelligence, 2019, 17.2: 44-65.
32. WEBSTER, Jonathan J.; KIT, Chunyu. Tokenization as the initial phase in NLP. In: COLING 1992 volume 4: The 14th international conference on computational linguistics. 1992.
33. SOLANGI, Yasir Ali, et al. Review on natural language processing (NLP) and its toolkits for opinion mining and sentiment analysis. In: 2018 IEEE 5th International Conference on Engineering Technologies and Applied Sciences (ICETAS). IEEE, 2018. p. 1-4.
34. PARK, Kyubyong, et al. An empirical study of tokenization strategies for various Korean NLP tasks. arXiv preprint arXiv:2010.02534, 2020.
35. VIJAYARANI, S., et al. Text mining: open source tokenization tools-an analysis. Advanced Computational Intelligence: An International Journal (ACII), 2016, 3.1: 37-47.
36. MEE, Alexander, et al. Sentiment analysis using TF–IDF weighting of UK MPs' tweets on Brexit. Knowledge-Based Systems, 2021, 228: 107238.

37. GAN, Guojun; MA, Chaoqun; WU, Jianhong. Data clustering: theory, algorithms, and applications. Society for Industrial and Applied Mathematics, 2020.
38. AHMED, Mohiuddin; SERAJ, Raihan; ISLAM, Syed Mohammed Shamsul. The k-means algorithm: A comprehensive survey and performance evaluation. Electronics, 2020, 9.8: 1295.
39. ROKACH, Lior; MAIMON, Oded. Clustering methods. Data mining and knowledge discovery handbook, 2005, 321-352.
40. GAN, Guojun; MA, Chaoqun; WU, Jianhong. Data clustering: theory, algorithms, and applications. Society for Industrial and Applied Mathematics, 2020.
41. GARCIA-DIAS, Rafael, et al. Clustering analysis. In: machine learning. Academic Press, 2020. p. 227-247.
42. JAIN, Anil K.; MURTY, M. Narasimha; FLYNN, Patrick J. Data clustering: a review. ACM computing surveys (CSUR), 1999, 31.3: 264-323.
43. ZHANG, Tian; RAMAKRISHNAN, Raghu; LIVNY, Miron. BIRCH: an efficient data clustering method for very large databases. ACM sigmod record, 1996, 25.2: 103-114.
44. BOUVEYRON, Charles; GIRARD, Stéphane; SCHMID, Cordelia. High-dimensional data clustering. Computational statistics & data analysis, 2007, 52.1: 502-519.
45. PAL, Riya, et al. Resume classification using various machine learning algorithms. In: ITM Web of Conferences. EDP Sciences, 2022. p. 03011.
46. NCIR, Chiheb-Eddine Ben; HAMZA, Abdallah; BOUAGUEL, Waad. Parallel and scalable Dunn Index for the validation of big data clusters. Parallel Computing, 2021, 102: 102751.
47. DUDEK, Andrzej. Silhouette index as clustering evaluation tool. In: Classification and Data Analysis: Theory and Applications 28. Springer International Publishing, 2020. p. 19-33.

48. KARIM, Md Rezaul, et al. Deep learning-based clustering approaches for bioinformatics. Briefings in bioinformatics, 2021, 22.1: 393-415.
49. NASSER, Shabna; SREEJITH, C.; IRSHAD, M. Convolutional neural network with word embedding based approach for resume classification. In: 2018 International Conference on Emerging Trends and Innovations In Engineering And Technological Research (ICETIETR). IEEE, 2018. p. 1-6.
50. GOLALIPOUR, Keyvan, et al. From clustering to clustering ensemble selection: A review. Engineering Applications of Artificial Intelligence, 2021, 104: 104388.
51. BAI, Liang; LIANG, Jiye; CAO, Fuyuan. A multiple k-means clustering ensemble algorithm to find nonlinearly separable clusters. Information Fusion, 2020, 61: 36-47.
52. ZENG, Kun, et al. An ensemble learning strategy for eligibility criteria text classification for clinical trial recruitment: algorithm development and validation. JMIR Medical Informatics, 2020, 8.7: e17832.
53. Najjar, Arwa & Amro, Belal & Macedo, Mário. (2022). An Intelligent Decision Support System For Recruitment: Resumes Screening And Applicants Ranking. Informatica. 45. 617-623. 10.31449/inf.v45i4.3356.
54. PESSACH, Dana, et al. Employees recruitment: A prescriptive analytics approach via machine learning and mathematical programming. Decision Support Systems, 2020, 134: 113290.
55. BOLLINGER, Jacob; HARDTKE, David; MARTIN, Ben. Using social data for resume job matching. In: Proceedings of the 2012 workshop on Data-driven user behavioral modelling and mining from social media. 2012. p. 27-30.
56. CHEN, Yanming; LI, Xinlong. Salary Prediction Based on the Resumes of the Candidates. In: SHS Web of Conferences. EDP Sciences, 2023. p. 03013.
57. KOR, Korhan; ALTUN, Gursat. Is Support Vector Regression method suitable for predicting rate of penetration?. Journal of Petroleum Science and Engineering, 2020, 194: 107542.

58. LA CAVA, William, et al. Contemporary symbolic regression methods and their relative performance. arXiv preprint arXiv:2107.14351, 2021.
59. ROY, Pradeep Kumar; CHOWDHARY, Sarabjeet Singh; BHATIA, Rocky. A Machine Learning approach for automation of Resume Recommendation system. Procedia Computer Science, 2020, 167: 2318-2327.
60. MAULUD, Dastan; ABDULAZEEZ, Adnan M. A review on linear regression comprehensive in machine learning. Journal of Applied Science and Technology Trends, 2020, 1.2: 140-147.
61. BOYKO, Nataliya; OMELIUKH, Roman; DULIABA, Nataliia. The Random Forest Algorithm as an Element of Statistical Learning for Disease Prediction. interactions, 2022, 4: 13.
62. CHEN, Tianqi; GUESTRIN, Carlos. Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016. p. 785-794.
63. ZHAO, Jinxin, et al. Large scale autonomous driving scenarios clustering with self-supervised feature extraction. In: 2021 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2021. p. 473-480.
64. SAGI, Omer; ROKACH, Lior. Approximating XGBoost with an interpretable decision tree. Information sciences, 2021, 572: 522-542.
65. SHEHADEH, Ali, et al. Machine learning models for predicting the residual value of heavy construction equipment: An evaluation of modified decision tree, LightGBM, and XGBoost regression. Automation in Construction, 2021, 129: 103827.
66. Basili, V. R., & Reiter, R. W. (1979). Evaluating automatable measures of software development. Workshop on Quantitative Software Models for Reliability, Complexity and Cost, 107–116."
67. DEWI, Lusiana Citra, et al. Social media web scraping using social media developers API and regex. Procedia Computer Science, 2019, 157: 444-449.

68. KUSUMASARI, Bevaola; PRABOWO, Nias Phydra Aji. Scraping social media data for disaster communication: how the pattern of Twitter users affects disasters in Asia and the Pacific. Natural Hazards, 2020, 103.3: 3415-3435.
69. MARRES, Noortje; WELTEVREDE, Esther. Scraping the social? Issues in live social research. Journal of cultural economy, 2013, 6.3: 313-335
70. HASANI, Zirije; KON-POPOVSKA, Margita; VELINOV, Goran. Lambda architecture for real time big data analytic. ICT Innovations, 2014, 133-143.
71. MUNSHI, Amr A.; MOHAMED, Yasser Abdel-Rady I. Data lake lambda architecture for smart grids big data analytics. IEEE Access, 2018, 6: 40463-40471.
72. PALANKAR, Mayur R., et al. Amazon S3 for science grids: a viable solution?. In: Proceedings of the 2008 international workshop on Data-aware distributed computing. 2008. p. 55-64.
73. BORNHOLT, James, et al. Using lightweight formal methods to validate a key-value storage node in Amazon S3. In: Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles. 2021. p. 836-850.
74. GULABANI, Sunil. Practical Amazon EC2, SQS, Kinesis, and S3. Apress,, 2017.
75. MISTRY, Atul. Expert AWS Development: Efficiently develop, deploy, and manage your enterprise apps on the Amazon Web Services platform. Packt Publishing Ltd, 2018.
76. SRIVASTAVA, Mayank; YADAV, Pradduman. Build a log analytic solution on aws. In: 2021 5th International Conference on Information Systems and Computer Networks (ISCON). IEEE, 2021. p. 1-5.
77. Hofmann, W., Lang, S., Reichardt, P., & Reggelin, T. (2022). A brief introduction to deploy Amazon Web Services for online discrete-event simulation. Procedia Computer Science, 200, 386-393.
78. KIRAN, Mariam, et al. Lambda architecture for cost-effective batch and speed big data processing. In: 2015 IEEE international conference on big data (big data). IEEE, 2015. p. 2785-2792.

79. Choudhary, G. R., Kumar, S., Kumar, K., Mishra, A., & Catal, C. (2018). Empirical analysis of change metrics for software fault prediction. Computers and Electrical Engineering, 67(1), 15–24.
80. AGGARWAL, Sanchit, et al. Modern web-development using reactjs. International Journal of Recent Research Aspects, 2018, 5.1: 133-137.
81. DANIELSSON, William. React Native application development. Linköpings universitet, Swedia, 2016, 10.4: 10.
82. DANIELSSON, William. React Native application development: A comparison between native Android and React Native. 2016.
83. HANSSON, Niclas; VIDHALL, Tomas. Effects on performance and usability for cross-platform application development using React Native. 2016.
84. BALLAMUDI, V. K. R., et al. Getting Started Modern Web Development with Next. js: An Indispensable React Framework. Digitalization & Sustainability Review, 2021, 1.1: 1-11.
85. FORCIER, Jeff; BISSEX, Paul; CHUN, Wesley J. Python web development with Django. Addison-Wesley Professional, 2008.
86. LEI, Kai; MA, Yining; TAN, Zhi. Performance comparison and evaluation of web development technologies in php, python, and node. js. In: 2014 IEEE 17th international conference on computational science and engineering. IEEE, 2014. p. 661-668.
87. DAUZON, Samuel; BENDORAITIS, Aidas; RAVINDRAN, Arun. Django: web development with Python. Packt Publishing Ltd, 2016.
88. KUČAK, Danijel; BELE, Daniel; PAŠIĆ, Đani. Climbing up the leaderboard: an empirical study of improving student outcome by applying Gamification principles to an object-oriented programming course on a university level. In: 2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO). IEEE, 2021. p. 527-531.

89. Chidamber, S. R., & Kemerer, C. F. (1994). A metrics suite for object oriented design. IEEE Transactions on Software Engineering, 20(6), 476–493.
90. LI, Yongkang, et al. Serverless computing: state-of-the-art, challenges and opportunities. IEEE Transactions on Services Computing, 2022, 16.2: 1522-1539.
91. DAVIS, Joanna, et al. Networking via LinkedIn: An examination of usage and career benefits. Journal of Vocational Behavior, 2020, 118: 103396.
92. BOURHIS, Pierre; REUTTER, Juan L.; VRGOČ, Domagoj. JSON: Data model and query languages. Information Systems, 2020, 89: 101478.
93. KAUSHIK, Prakarsh, et al. Cloud computing and comparison based on service and performance between Amazon AWS, Microsoft Azure, and Google Cloud. In: 2021 International Conference on Technological Advancements and Innovations (ICTAI). IEEE, 2021. p. 268-273.
94. AMAZON, E. C. Amazon ec2. línea]. Available: https://aws. amazon. com/es/ec2, 2022.
95. FEURER, Matthias, et al. Auto-sklearn 2.0: The next generation. arXiv preprint arXiv:2007.04074, 2020, 24: 8.
96. JAYAKUMAR, Nithya, et al. On-Demand Job-Based Recruitment For Organisations Using Artificial Intelligence. In: 2023 International Conference on Networking and Communications (ICNWC). IEEE, 2023. p. 1-6.
97. BOYD, Kristi Shevkun. A Comparison of the Attitudes of Human Resource (HR) Executives and HR Practitioners on the Use of Artificial Intelligence (AI)-Enabled Tools in Recruiting. 2022. PhD Thesis. Duke University.
98. SURANA, Ankit, et al. The internationalization process: A contextual analysis of Indian ibusiness firms. International Business Review, 2024, 102255.
99. ZAMBETAKIS, Benita. Diversity, Equity, and Inclusion in Recruitment practices. 2022.
100. DAWWAS, Mohammed; ALLAYMOUN, Mohammad; ALZGOOL, Mahmoud. Enhancing Green Recruitment Through Implementing Artificial

Intelligence: Zoho Recruitment System. In: Artificial Intelligence (AI) and Finance. Cham: Springer Nature Switzerland, 2023. p. 3-13.

101. TÜRKELI, Ihsan. Artificial Intelligence in Recruitment: the ethical implications of AI-powered recruitment tools.

ДОДАТОК А. АКТИ ВПРОВАДЖЕННЯ

"ЗАТВЕРДЖУЮ"
Проректор з науково-педагогічної роботи
Національного університету
"Львівська політехніка"
_____ **Олег ДАВИДЧАК**
"02" 05 2024 р.

АКТ
впровадження в навчальний процес результатів
дисертаційної роботи
Михайлишина Владислава Юрійовича

Цей акт складено про те, що результати дисертаційної роботи Михайлишина Владислава Юрійовича впроваджено у навчальний процес кафедри "Системи штучного інтелекту" Національного університету "Львівська політехніка".

Впровадження результатів дисертаційної роботи полягає в їхньому використанні при викладанні навчальних дисциплін як окремих розділів лекційних курсів, так і в циклах лабораторних робіт.

Зокрема для викладання дисципліни "Об'єктно-орієнтоване програмування" для студентів освітньо-кваліфікаційного рівня "бакалавр", що навчаються за напрямом 122 "Комп'ютерні науки" використано такі результати:

- створення моделей даних;
- розробка алгоритмів обробки даних;
- веб API.

У лекційному курсі "Проектування інформаційних систем" для студентів кваліфікаційного рівня "бакалавр", що навчаються за напрямом 122 "Комп'ютерні науки", використано такі результати:

- проектування баз даних;
- створення архітектури інформаційних систем.
- застосування лямбда-архітектури

Директор ІКНІ
д.т.н., професор Микола МЕДИКОВСЬКИЙ

завідувач кафедри СШІ
д.т.н., професор Наталія ШАХОВСЬКА

професор кафедри СШІ
д.т.н., доцент Наталія МЕЛЬНИКОВА

“ЗАТВЕРДЖУЮ”
Проректор з наукової роботи
Національного університету
“Львівська політехніка”
_____ **Іван ДЕМИДОВ**
“ ____ ” _____ 2024 р.

АКТ
використання наукових результатів
дисертаційної роботи Михайлишина Владислава Юрійовича
представленої на здобуття наукового ступеня доктора філософії

Комісія в складі: голови комісії – начальника науково-дослідної частини д.т.н.,с.н.с. Небесного Р.В. та членів комісії – завідувача кафедри СШІ Шаховської Н.Б., старшого викладача кафедри СШІ Думин І.Б., доцента кафедри СШІ Хавалка В.М., доцента кафедри СШІ Кривенчука Ю.П. цим актом підтверджують, що результати дисертаційної роботи Михайлишина В.Ю., зокрема:

- метод двоетапної обробки великих неоднорідних наборів даних;
- метод зменшення розмірності вхідних даних;
- метод кластеризації вхідних наборів даних.

використано у науково-дослідних роботах фінансованих Міністерством освіти і науки України, що виконувалась на кафедрі систем штучного інтелекту і включено до звіту: «Технологія опрацювання мультимодальних українськомовних наборів даних для визначення рівня стресу» (№ держ. реєстру 0123U100231).
Отримані автором результати використано:

- при розроблені системи автоматичної рекомендації кандидатів на основі аналізу неоднорідних структурованих та неструктурованих даних;
- при розроблені методів консолідації вихідних результатів системи прийняття рішень;
- при розроблені архітектури системи та програмних модулів.

Голова комісії:

Начальник науково-дослідної частини д.т.н., с.н.с. Роман НЕБЕСНИЙ

Члени комісії:

Завідувач кафедри СШІ Наталія ШАХОВСЬКА

Старший викладач кафедри СШІ Ірина ДУМИН

Доцент кафедри СШІ Віктор ХАВАЛКО

Доцент кафедри СШІ Юрій КРИВЕНЧУК

“ЗАТВЕРДЖУЮ”

Директор ТзОВ «Палетний сервіс»

Петро ХУДОБА

“____” _____ 2024 р.

АКТ

про впровадження результатів дисертаційної роботи
аспіранта кафедри «Системи штучного інтелекту»
Національного університету «Львівська політехніка»
Михайлишина Владислава Юрійовича

Цей акт підтверджує, що результати дисертаційної роботи Михайлишина В.Ю. були використані для розроблення системи підтримки та прийняття рішень у рекрутингу в м. Львів за 2022-2023 рр.

Впровадження дисертаційних досліджень Михайлишина В.Ю. полягає у наступному:

- Розроблено метод кластеризації вакансій, резюме та інших даних ринку праці, з метою допомогти компаніям аналізувати попит та пропозицію на ринку праці, ідентифікувати ключові навички та потенційних кандидатів.
- Розроблено архітектуру інформаційної системи і програмні модулі для автоматичного аналізу та прогнозування рекрутингової діяльності.

Даний акт не є підставою для взаємних фінансових розрахунків.

Директор ТзОВ «Палетний сервіс» Петро ХУДОБА