НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«На правах рукопису»
УДК 004.89

«До захисту допущено»
Науковий керівник кафедри
_____ І.А. Дичка
«___»_____2019 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 121 Інженерія програмного забезпечення

на тему: «Модифікований метод автоматизації прийняття управлінських рішень на основі інтелектуального аналізу даних»

Виконала:
студентка ІІ курсу, групи КП-71мн
Чорна Оксана Вікторівна _____

Керівник:
Старший викладач кафедри ПЗКС, к.т.н.,
Люшенко Леся Анатоліївна _____

Консультант з нормоконтролю:
Доцент кафедри ПЗКС, к.т.н.,
Онай Микола Володимирович _____

Рецензент:
Доцент кафедри ММСА, к.ф.-м.н., доцент,
Шубенкова Ірина Анатоліївна _____

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

Київ – 2019 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

Рівень вищої освіти – другий (магістерський) за освітньо-науковою програмою

Спеціальність – 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Науковий керівник кафедри

_____ І.А. Дичка

«___»_____2017 р.

ЗАВДАННЯ
на магістерську дисертацію студентці

Чорній Оксані Вікторівні

1. Тема дисертації «Модифікований метод автоматизації прийняття управлінських рішень на основі інтелектуального аналізу даних», науковий керівник дисертації Люшенко Леся Анатоліївна, к.т.н., старший викладач, затверджені наказом по університету від «8» квітня 2019 р. №1075-С

2. Термін подання студентом дисертації «17» травня 2019 р.

3. Об'єкт дослідження: автоматизація прийняття рішень при підборі учасників команди проекту на основі текстових даних.

4. Предмет дослідження: методи та алгоритми автоматизації прийняття рішень на основі інтелектуального аналізу текстових даних.

5. Перелік завдань, які потрібно розробити:
   - провести аналіз методів інтелектуального аналізу текстових даних;
   - дослідити існуючі автоматизовані системи підтримки прийняття рішень;
   - розробити та дослідити модифікований метод на основі наївної моделі Байєса для автоматизації прийняття управлінських рішень;
   - обґрунтувати вибір критеріїв оптимізації модифікованого методу;
   - розробити автоматизовану систему підтримки прийняття управлінських рішень на основі модифікованого методу;
   - проаналізувати ефективність модифікованого методу в порівнянні з класичним на основі визначених критеріїв оптимізації.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу:
   - теоретичні аспекти побудови та оптимізації модифікованого методу автоматизації підтримки прийняття управлінських рішень;
   - схема роботи модифікованого методу автоматизації підтримки прийняття рішень;
   - діаграми з результатами щодо обґрунтування критеріїв ефективності методу;
   - схема алгоритму інтелектуального аналізу текстових документів;
   - архітектура автоматизованої системи підтримки прийняття рішень;

- часові характеристики роботи автоматизованої системи підтримки прийняття рішень на основі модифікованого методу.

7. Орієнтовний перелік публікацій:
   - Стаття “Модифікований метод автоматизації прийняття управлінських рішень на основі інтелектуального аналізу даних при створенні команди управління проектами”
   - Тези доповіді “Використання систем підтримки прийняття рішень для ефективного управління проектами в програмній інженерії”

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Онай М.В., доцент кафедри ПЗКС		

9. Дата видачі завдання «11» жовтня 2017 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Ґрунтовне ознайомлення з предметною галуззю	18.12.2017	
2.	Визначення структури магістерської дисертації; вивчення літератури, пошук додаткової літератури, патентний пошук	05.03.2018	
3.	Робота над першим розділом магістерської дисертації; проведення наукового дослідження	15.05.2018	
4.	Проведення наукового дослідження; робота над другим розділом магістерської дисертації; розроблення програмного забезпечення; підготовка матеріалів доповіді на конференції ПМК-2018	16.10.2018	
5.	Проведення наукового дослідження; робота над статтею за результатами наукового дослідження	17.12.2018	
6.	Проведення наукового дослідження; робота над третім розділом магістерської дисертації	11.02.2019	
7.	Завершення роботи над основною частиною магістерської дисертації; підготовка ілюстративного матеріалу	16.04.2019	
8.	Оформлення текстової і графічної частини магістерської дисертації	06.05.2019	

Студент О.В. Чорна

Науковий керівник дисертації Л.А. Люшенко

РЕФЕРАТ

Актуальність теми. Процес розробки програмного забезпечення починається з формування команди спеціалістів, що будуть залучені в проект. В компаніях досить часто нехтують процесом правильного вибору команди, що в подальшому приводить до низької якості програмних продуктів. Процес вибору учасників команди має бути автоматизованим та прийматися на підставі інформації про співробітників. Програмне забезпечення для автоматизації процесу підбору команди проекту будується на обробці та аналізі текстових документів для подальшого прийняття рішення. В даній магістерській дисертації розглядається вирішення задачі автоматизованого підбору учасників команди проектів з використанням текстових даних, що містять інформацію про професійні та особисті якості співробітників компанії, з метою підвищення точності вибору та швидкодії в порівнянні з існуючими методами.

Об'єктом дослідження автоматизація прийняття рішень при підборі учасників команди проекту на основі текстових даних.

Предметом дослідження є методи та алгоритми автоматизації прийняття рішень на основі інтелектуального аналізу текстових даних.

Мета роботи полягає у розробці ефективного методу автоматизації підтримки прийняття рішень з підбору учасників команди проекту на основі наївної моделі Байєса за критерієм точності та швидкодії отримуваних результатів.

Методи дослідження: в роботі використовуються методи теоретичного дослідження: аналіз та синтез. Також застосовувалися емпіричні методи: експеримент, вимірювання та порівняння.

Наукова новизна роботи полягає у розробленні модифікованого методу Байєса для автоматизації підтримки прийняття рішень з підбору учасників команди проекту, який на відміну від класичного методу видає

рішення за критерієм точності на 10-13% вище та в середньому на 20% швидше.

Практична цінність отриманих результатів роботи полягає в тому, що запропонований метод дає змогу підвищити точність в прийнятті рішення з підбору учасників команди проекту.

Також в рамках даного дослідження була розроблена автоматизована система підтримки прийняття рішень з підбору співробітників компанії для участі у нових проектах на основі запропонованого модифікованого методу.

Апробація роботи. Основні положення і результати роботи були представлені та обговорювались на XI науковій конференції магістрантів та аспірантів «Прикладна математика та комп'ютинг» ПМК-2018-2 та опубліковані у збірнику тез доповідей.

За результатами роботи була написана наукова стаття на тему «Модифікований метод автоматизації прийняття управлінських рішень на основі інтелектуального аналізу даних при створенні команди управління проектами» до наукового міжнародного журналу «Керуючі системи та комп'ютери» 2019 №4.

Структура та обсяг роботи. Магістерська дисертація складається з вступу, чотирьох розділів, висновків та додатків.

У вступі надано загальну характеристику роботи, виконано оцінку сучасного стану проблеми, обґрунтовано актуальність напрямку досліджень, сформульовано мету і задачі досліджень.

У першому розділі описана задача підбору учасників команди для виконання проектів з розроблення програмного забезпечення, розглянуті особливості автоматизованих систем підтримки прийняття рішень, оглянуті існуючі методи інтелектуального аналізу даних для вирішення поставленої задачі.

У другому розділі розглянуто принцип роботи наївної моделі Байєса, проаналізовані її переваги та недоліки. Запропонований модифікований метод на основі моделі Байєса для автоматизації підтримки прийняття рішень з підбору учасників команди.

У третьому розділі сформовані основні вимоги до автоматизованої системи підтримки прийняття рішень; обґрунтовано вибір засобів, що використовувались при розробці; описана розроблена система, що реалізує модифікований метод автоматизації прийняття рішень з підбору учасників команди проекту.

У четвертому розділі визначено критерії оцінки ефективності, які застосовуються до розробленого методу; наведена інформація про дані, що використовувались при аналізі ефективності; проведений аналіз ефективності модифікованого та базового методів автоматизації прийняття рішень з підбору учасників команди.

У висновках проаналізовано отримані результати роботи.

Робота виконана на 70 аркушах, містить 2 додатки та посилання на список використаних літературних джерел з 30 найменувань. У роботі наведено 10 рисунків та 2 таблиці.

Ключові слова: автоматизована система підтримки прийняття рішень, модифікований метод автоматизації прийняття рішень, модель Байєса, підбір учасників команди проекту.

ABSTRACT

Actuality. The process of the software development begins with the formation of a team of specialists who will be involved in the project. Companies often overlook the process of choosing the right team, which leads to low quality of software products in the future. The process of selecting team members must be automated and taken on the basis of employee information. The software for automating the project selection process is based on the processing and analysis of text documents for further decision-making. This master's thesis deals with the problem of automated selection of project team members using text data that contains information about the professional and personal qualities of the company's employees in order to improve the accuracy of the choice and performance compared to existing methods.

Object of research is decision-making automation at selection of the project team participants on the basis of text data.

Subjects of research are methods and algorithms of decision-making automation on the basis of text data intellectual analysis.

Goal of the work is to develop an effective method of automating decision support for selection of project team members based on the Bayesian naive model for the criterion of accuracy and performance of the results.

Methods of research include methods of theoretical research: analysis and synthesis. Also there were used empirical methods: experiment, measurement and comparison.

Scientific novelty of the work is to develop a modified Bayesian method to automate decision-making support for selection of project team members, which, in contrast to the classical method, gives solutions by accuracy criterion of 10-13% higher and an average of 20% faster.

Practical value of the received results of work is that the proposed method allows increase the accuracy in decision making of the selection of project team members.

Also, in this research, an automated decision-making support system was developed for selecting company employees for participation in new projects on the basis of the proposed modified method.

Approbation. The main provisions and results of the work were presented and discussed at the XI scientific conference of masters and postgraduates "Applied Mathematics and Computer" PMK-2018-2 and published in the proceedings.

As a result of the work, a scientific article on the topic "Modified method of decision making automation on the basis of intelligent data analysis when creating a team of project management" was written to the international scientific journal "Control systems and computers" 2019 № 4.

Structure and content of the thesis. Master's thesis consists of an introduction, four chapters, conclusions and appendices.

The introduction provides a general description of the work, evaluated the current state of the problem, substantiated the relevance of the research direction, formulated the purpose and objectives of the study.

The first chapter describes the task of selecting team members to implement software development projects, features of automated decision support systems were considered, the existing methods of intelligent data analysis for solving the problem were examined.

The second chapter discusses the principle of the Bayesian naive model, analyzing its advantages and disadvantages. A modified method based on Bayesian model is proposed for automating decision support for selection of team members.

In the third section, the main requirements for an automated decision support system are formed; the choice of the means used during development was substantiated; describes a developed system that implements a modified method of decision making automation in selecting project team members.

The fourth chapter defines the criteria for assessing the effectiveness of the developed method; provides information on the data used in the analysis of efficiency; the analysis of the efficiency for the modified and basic methods of automated decision making on the selection of team members was conducted.

The conclusion contains brief overview of the results obtained in the work.

The work is done on 70 pages, contains 2 appendices and reference list of 30 titles. The work contains 10 pictures and 2 tables.

Keywords: automated decision support system, modified method of decision making automation, Bayes model, selection of project team members.

ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

iOS – мобільна операційна система від компанії Apple;

Android – мобільна операційна система від компанії Google;

Управління проектами – це область діяльності менеджменту, в якій визначаються і досягаються чіткі поставлені цілі при балансуванні між ресурсами (такими як праця, гроші, матеріали, простір, енергія та ін.), обсягом робіт, якістю, часом і ризиком в межах деяких проектів;

IT (Information Technology) – інформаційні технології;

АСППР – автоматизована система підтримки прийняття рішень;

ІАД – інтелектуальний аналіз даних;

Інформаційний пошук – наука про пошук неструктурованої документальної інформації;

Імітаційне моделювання – метод, призначений для побудови моделей процесів, що описують як ці процеси проходили б насправді;

Штучна нейронна мережа – обчислювана система на основі роботи біологічної нейронної мережі;

Data mining – процес аналізу великих баз даних щ метою пошуку корисних фактів;

Рекрутинг – метод підбору персоналу в штат компанії;

Text mining – інтелектуальний аналіз текстових даних;

НБА – «наївний» байєсовський алгоритм;

Logistic regression – статистичний регресійний метод, який застосовують у випадку, коли залежна змінна може набувати тільки двох значень;

ID3 – алгоритм, який використовується для генерації дерева рішень у машинному навчанні з деякого набору даних;

БМ – байєсівська мережа;

ПЗ – програмне забезпечення;

ЕОМ – електронна обчислювальна машина;

ОС – операційна система;

CASE (Computer-Aided Software Engineering) – набір інструментів і методів програмної інженерії для проектування програмного забезпечення;

RAD (Rapid Application Development) – концепція швидкої розробки програмного забезпечення;

БД – база даних;

SQL (Structured Query Language) – декларативна мова програмування для взаємодії користувача з базами даних;

UNIX – відкрита операційна система.

ВСТУП

Процес розробки нового програмного забезпечення починається з формування команди дизайнерів, аналітиків, розробників, тестувальників та інших спеціалістів галузі інформаційних технологій. Можна стверджувати, що вже на етапі створення команди є дуже важливим правильний підбір кожного учасника за його досвідом роботи, професійними та особистими якостями. Проте, в компаніях досить часто нехтують процесом правильного вибору команди. Часто підбір учасників здійснюється лише за критерієм завантаженості та досвідом роботи з конкретною технологією або мовою програмування.

За прийняття рішень з підбору команди в організаціях як правило відповідає рекрутер, керівник проекту або директор. Якщо прийняття рішення виконується людиною без використання спеціальних програмних систем, можливий вплив «людського» фактору та суб'єктивної оцінки.

Процес вибору учасників команди має бути автоматизованим, адже це та дія, яка обов'язково виконується на старті нового проекту та повторюється щоразу в міру того, як компанія отримує нові проекти.

Метою дослідження є проаналізувати сучасні методи, які вирішують задачу автоматизації прийняття рішень з підбору учасників команди проекту, обрати метод, що буде найкраще вирішувати поставлену задачу та модифікувати його для отримання найбільшої точності в прийнятті рішень.

В рамках проведення даного дослідження були проведені наступні роботи:

- огляд та визначення завдань автоматизованих систем підтримки прийняття рішень;
- огляд провідних методів інтелектуального аналізу даних та вибір методу, що підійде для вирішення задачі підбору учасників команди проекту;

- аналіз аналогічного програмного забезпечення, основною функцією якого є підбір учасників для проектів за різними вхідними параметрами;
- постановка задачі для вирішення досліджуваної проблеми;
- створення модифікованого методу, який буде перевершує класичний метод за критеріями точності та швидкодії;
- розроблення автоматизованої системи підтримки прийняття рішень на основі створеного модифікованого методу;
- тестування модифікованого методу;
- дослідження ефективності запропонованого методу, порівняння модифікованого методу з класичним на різних наборах вхідних даних;
- аналіз та оцінка розробленого модифікованого методу.

Запропонований модифікований метод автоматизації підтримки прийняття рішень з підбору учасників команди вирішує поставлену задачу з високою точністю та швидкодією.

1. ОГЛЯД ІСНУЮЧИХ МЕТОДІВ ПРИЙНЯТТЯ УПРАВЛІНСЬКИХ РІШЕНЬ З ПІДБОРУ КОМАНДИ ПРОЕКТУ

1.1. Загальний опис задачі прийняття рішень при підборі команди проекту по розробці програмного забезпечення

Головними факторами успіху управління проектами програмної інженерії є наявність чіткого, заздалегідь визначеного плану виконання проекту, ефективний розподіл ресурсів, мінімізація ризиків і відхилень від плану, а також ефективне управління змінами [1].

Людські ресурси – це один з головних видів ресурсів в управлінні проектами. Цей вид ресурсу значно відрізняється від інших (матеріальних, фінансових, сировинних) тим, що співробітник має повне право на наступні дії:

- відмовитися від тих умов, при яких працівника збираються використовувати;
- вести переговори з керівництвом про більший рівень оплати своєї праці;
- перевчатися на іншу спеціальність;
- приймати участь у страйках;
- звільнятися з займаної посади за своїм власним бажанням;
- приймати рішення, які професії для нього є соціально неприйнятними.

Прийняття рішень в проектах є найбільш важливим видом діяльності, здійснюваної проектними менеджерами, і являє собою одноразовий акт остаточного вибору одного з можливих варіантів дій по досягненню поставлених цілей.

Підбір учасників команди та, власне, створення нової команди для конкретного проекту є досить складною задачею, адже саме від кожного учасника окремо та усієї команди разом залежить весь процес розробки та успішність програмних продуктів.

Створення команди пов’язане з необхідністю враховувати, як професійні фахові компетенції, так і поведінкові компетенції кожного члена команди проекту.

За умови, що в проект необхідно залучити одного спеціаліста, наприклад, програміста, створення команди зводиться до пошуку одного фахівця, який буде відповідати наступним критеріям:

- якнайкраще задовольняє вимогам проекту, які встановляє керівник проекту самостійно, опираючись на висунуті вимоги від замовника, споживачів або від бізнес-аналітика (знання певних технологій та інструментів, розуміння предметної області);
- зайнятість. Фахівець має завершити всі поточні задачі інших проектів на момент прийняття нового проекту або брати та виконувати задачі нового проекту паралельно до наявних задач;
- досвід роботи (на аналогічних проектах, з конкретними технологіями, зі суміжною предметною областю);
- швидкість – різниця між часом закінчення виконання завдання та часом початку (швидкість виконання є важливою на проектах зі стиснутими термінами).

Навіть для пошуку одного фахівця, що підійде для виконання конкретної задачі проекту потрібно витратити чимало зусиль.

Задача з прийняття рішень при підборі команди спеціалістів для проектів розробки програмного забезпечення сьогодні є актуальною, особливо коли мова йде про стратегічні та довготривалі проекти, де планується команда на п’ятдесят або більше учасників, адже фахівці мають бути бездоганно підібрані на всіх етапах реалізації проекту.

В багатьох ІТ компаніях підбір команди або конкретного фахівця може зробити керівник проекту самостійно, проте немає гарантії, що такий вибір буде найкращим та в подальшій роботі не призведе до виникнення

конфліктів в проекті. Саме тому автоматизація прийняття управлінських рішень при підборі команди проекту (людських ресурсів проекту) є актуальною. Остаточне рішення завжди буде залишатися за керівником, проте використання автоматизованих систем є кращою практикою, адже вони створені для спрощення процесу прийняття рішень, пошуку найоптимальнішого рішення та уникнення впливу людського фактору на етапі прийняття рішення.

1.2. Використання систем підтримки прийняття рішень для вирішення задачі підбору команди проекту з розроблення програмного забезпечення

Своєчасна розробка і прийняття правильного рішення – головні завдання роботи менеджерів з управління проектами кожної ІТ організації. Коли потрібно прийняти кілька рішень в умовах невизначеності на допомогу приходить використання авторизованих систем підтримки прийняття рішення (АСППР).

АСППР представляють собою програмне забезпечення вузького профілю, які призначені для аналітиків та менеджерів, що спеціалізуються на даному профілі [2]. Вони характеризуються можливістю реалізації високого рівня формалізації вироблення рекомендацій для прийняття різних рішень, включаючи раціональний розподіл управлінських рішень. При цьому, сильно знижується рівень «людського фактору», який може привести до суб’єктивної оцінки. Одним з основних завдань інформаційної системи підтримки прийняття рішень є вибір серед безлічі альтернатив однієї, яка підходить якнайкраще для досягнення певної мети.

Для аналізу даних в АСППР використовується безліч різних методів. Серед передових можна виділити:

- інтелектуальний аналіз даних;
- інформаційний пошук (аналітичні запити);

- пошук знань в базах даних (пошук прихованої, неочевидної інформації в даних);
- імітаційне моделювання;
- методи міркування на основі прецедентів (аналогічних рішень);
- нейронні мережі.

АСППР, що підтримують роботу методів штучного інтелекту, називають інтелектуальними.

Чіткого визначення АСППР не існує, однак, є деякий набір характеристик для АСППР, яким система повинна відповідати [3]:

- АСППР для роботи використовує дані і моделі;
- АСППР призначається для допомоги в прийнятті рішень слабо-структурованих, неструктурованих або нетривіальних завдань;
- АСППР служить підтримкою для аналітика (менеджера, консультанта) в прийнятті рішень, а не замінює його;
- основною метою АСППР є збільшення показників ефективності рішень.

Процес прийняття рішень – це отримання та вибір найбільш оптимального альтернативного рішення з прорахуванням всіх наслідків. При виборі альтернативних рішень потрібно обирати те, яке найбільш повно та чітко відповідає поставленій цілі, однак при цьому потрібно враховувати велику кількість суперечливих вимог і, отже, оцінювати обраний варіант за багатьма критеріями [4].

Для вирішення поставленої задачі – підбору найоптимальнішого складу команди проекту, необхідно розробити автоматизовану систему підтримки прийняття рішень на основі одного з передових методів, що буде видавати найоптимальніше рішення за короткий проміжок часу.

1.3. Способи інтелектуального аналізу даних для використання у системах підтримки прийняття рішень з підбору команди проекту

Інтенсивне застосування інтелектуального аналізу даних (ІАД) здійснюється завдяки наявності робочих інструментів (пакетів програм), що реалізують різноманітні методи ІАД [5].

Інтелектуальний аналіз даних дозволяє знаходити не тільки статистичні закономірності, тобто закономірності «в середньому», а й моделі (закономірності), які описують, пояснюють і прогнозують, явища, що рідко зустрічаються.

ІАД для прийняття управлінських рішень є актуальним, оскільки існує необхідність доповнити дослідження середніх тенденцій вивченням індивідуальних особливостей, що проявилося у визнанні ефективності аналізу існуючих феноменів одночасно кількісними і якісними методами.

Методологія ІАД поєднує аналіз середніх тенденцій і тенденцій відхилення від середніх (тобто аналіз рідкісних, можливо, навіть унікальних випадків) в рамках тільки кількісного підходу. Аналогічно, якісні дослідження, що проводяться з використанням ІАД, також інтегрують названі аспекти аналізу даних [6].

У загальному випадку процес інтелектуального аналізу (тобто пошуку нового знання) складається з наступних етапів [7]:

1) відбір даних (вибір ознак, які, за припущенням дослідників, є значущими для конкретного дослідження);
2) попередня обробка даних або очищення (усунення неточностей);
3) трансформація (перетворення шкал для застосування обраних методів);
4) власне вилучення знань (Data Mining);
5) інтерпретація результатів в контексті змістовних гіпотез.

Головним елементом цього процесу є методи Data Mining, що дозволяють виявляти нові закономірності (шаблони) і знання, які є наслідком інтерпретації знайдених закономірностей.

Щоб вирішити завдання підбору команди фахівців для виконання проекту дослідити, який з існуючих методів, що застосовуються для створення систем підтримки прийняття рішень, буде швидше аналізувати вхідні дані.

Специфіка вимог до обробки вхідних даних наступна:

- дані мають необмежений обсяг;
- дані є різнорідними (кількісними, якісними, текстовими);
- результати повинні бути конкретні і зрозумілі;
- інструменти для обробки «сирих даних» повинні бути прості і зрозумілі у використанні.

Технологія Data Mining використовується для оброблення великих обсягів даних.

Методи Data Mining допомагають вирішити завдання класифікації, регресії, пошуку асоціативних правил та кластеризації [8].

Завдання класифікації зводиться до визначення класу об’єкта по його характеристикам. В цьому завданні безліч класів, до яких може бути віднесений об’єкт, заздалегідь відомо.

Завдання регресії подібне до задачі класифікації, дозволяє визначити за відомими характеристиками об’єкта значення деякого його параметра. На відміну від завдання класифікації значенням параметра є не кінцева безліч класів, а множина дійсних чисел [9].

При пошуку асоціативних правил метою є знаходження частих залежностей (або асоціацій) між об’єктами або подіями. Знайдені залежності представляються у вигляді правил і можуть бути використані як для кращого розуміння природи аналізованих даних, так і для передбачення появи подій.

Завдання кластеризації полягає в пошуку незалежних груп (кластерів) і їх характеристик у всій безлічі вхідних аналізованих даних.

Вирішення цього завдання допомагає краще зрозуміти дані. Крім того, угрупування однорідних об’єктів дозволяє скоротити їх число, а отже, і полегшити аналіз.

Велика кількість сучасних методів Data Mining має певні недоліки з оброблення інформації описаного типу.

Зі збільшенням кількості оброблюваних даних можливе занадто сильне збільшення обчислень, зокрема надмірна кількість вхідної інформації може ускладнити аналіз та спричинити зайвий підрахунок асоціативних правил [10].

Потрібно описати та розробити модифікацію методу, який в результаті буде достатньо швидко виконувати багатоскладову класифікацію, а також приймати рішення не тільки за допомогою узагальнення попереднього досвіду, а за допомогою створення нових правил та моделей.

1.4. Постановка задачі з прийняття рішень в підборі команди проекту

Оскільки за прийняття рішень в підборі учасників команди проекту відповідає людина, яка скоріше за все не має великого досвіду в програмуванні (рекрутер, менеджер або директор), потрібно розробити застосунок з графічним інтерфейсом для зручного користування та візуалізації даних.

Дані кожного потенційного учасника проекту повинні зберігатися у структурованих текстових документах для однозначної обробки програмним застосунком.

Кожний текстовий вхідний документ має містити наступні поля:

– прізвище, ім’я, по-батькові учасника;

– тип проекту (веб-сайт, веб-додаток, мобільний додаток, прикладна програма);
– стаж роботи вказаний у роках;
– вік учасника;
– кількість виконаних (завершених) проектів в компанії, що розглядається.

Для розробників потрібно враховувати наступні критерії:

– мова програмування;
– середня швидкість набору коду, яка задається цілим числом і відповідає за кількість знаків коду, що розробник набирає за одну хвилину.

Також потрібно врахувати інші компетенції та персональні якості кожного учасника, відповідь на які повинна бути бінарною: так чи ні. За ці характеристики будуть відповідати наступні поля:

– відповідальність;
– дотримання термінів;
– технічні навички;
– розуміння бізнес-процесів;
– якості керівника;
– стресостійкість;
– ініціативність;
– генерація ідей;
– якості виконавця.

Для розроблення автоматизованої системи підтримки прийняття рішень з підбору учасників проекту потрібно проаналізувати та описати модифікацію методу, який буде показувати найкращий показник швидкодії та точності.

Окрім того, модифікований метод повинен аналізувати та оброблювати всі ознаки (характеристики) учасника, враховуючи їх незалежність між собою.

У випадках, коли характеристики учасників будуть співпадати або відрізнятися деякими полями, автоматизована система підтримки прийняття рішень на основі модифікованого методу інтелектуального аналізу даних повинна обирати учасника, який найкраще підійде за усіма параметрами, які будуть розглянуті при підборі.

1.5. Огляд існуючих систем прийняття рішень з підбору персоналу команди проекту

Підбір персоналу – це комплекс направлених дій по залученню на роботу кандидатів (пошук, оцінка та найм людей), що володіють якостями, необхідними для досягнення цілей компанії [11].

Дуже часто поняття підбору персоналу та рекрутингу прирівнюють, хоча рекрутинг є одним з методів підбору персоналу.

На сьогоднішній день використовують наступні методи пошуку персоналу [12]:

- рекрутинг – метод підбору персоналу поширених професій, при якому готується та розміщується опис вакансії на сайті компанії та сайтах пошуку роботу, де потенційні кандидати зможуть побачити вакансію та відгукнутися на неї, відправивши своє резюме [13];
- executive search – підбір персоналу рідкісних та унікальних професій, на відміну від рекрутинга даний метод передбачає активний пошук потенційних кандидатів;
- headhunting – переманювання найкращих спеціалістів, які вже працевлаштовані, з однієї компанії в іншу;

– скрінінг – швидкий відбір кандидатів виключно за формальними ознаками, персональні якості при такому підборі не враховуються.

Весь процес підбору персоналу можна розділити на такі етапи:

1) Визначення потреби організації в нових співробітниках, відкриття відповідних вакансій;
2) аналіз отриманих резюме, первинний відбір претендентів, що відповідають вимогам вакансії;
3) первинна співбесіда по телефону;
4) тестування для визначення рівня професійних знань;
5) аналіз результатів;
6) прийняття рішення щодо працевлаштування;
7) узгодження дати виходу на роботу та підписання договору;
8) адаптація нового співробітника та його супровід впродовж випробовувального терміну.

На сьогоднішній день існує досить багато різних програмних систем для автоматизації процесу підбору персоналу. Однак, більшість з них не надають відкритої інформації щодо програмної реалізації, адже такі системи в своїй більшості є платними.

1.5.1. Огляд системи підбору персоналу Clobbi

Clobbi Рекрутинг – це система, яка дозволяє зручно поповнювати базу кандидатів компанії новими резюме з сайтів пошуку роботи, корпоративного сайта, електронної пошти та соціальних мереж, а також зберігати всю інформацію про кандидатів та результати співбесід з можливістю відслідковувати статистику для кожної вакансії.

Система Clobbi в певній мірі автоматизує процес пошуку нових співробітників, дозволяючи зберігати резюме по кожному кандидату з інформацією про дату, час, результати співбесіди, зарплатні очікування, кваліфікацію та професійні навички.

У випадку коли компанія відкриває багато різних вакансій процес підбору кандидатів та їх прийом на роботу може відрізнятися. Іноді необхідно провести декілька етапів співбесід, іноді потрібно провести додаткове тестування під час співбесіди або дати тестове завдання додому. Також може знадобитися оформлення додаткових документів. З використанням системи Clobbi можна легко керувати всіма процесами за різними вакансіями.

Отримані резюме з сайтів пошуку роботи можна легко додавати в базу резюме системи. Якщо резюме повторюється, система вкаже на це.

За додаткову плату можна налаштувати додаткові функції з управління персоналом.

1.5.2. Огляд системи підбору персоналу Ін-агро

Ін-агро – це комплексна система управління персоналом, яка має багато можливостей та функцій.

Для рекрутингу система надає наступні можливості:

- формування вакансій на вільні позиції штатного розкладу;
- опис вимог для кожного кандидата;
- статистична звітність процесу підбору персоналу;
- визначення способу пошуку кандидатів;
- первинна реєстрація кандидатів та отримання відгуків;
- відображення взаємодії з кандидатом;
- автоматизований процес затвердження кандидатів.

Також система Ін-агро надає можливості для кадрового обліку:

- відбір персоналу за різними категоріями, посадами та спеціальностями;
- вибір робочого графіку для кожного співробітника;
- деталізована інформація про співробітника;
- механізм занесення співробітників до списку небажаних співробітників.

Ін-агро забезпечена функціями створення і проведення тестувань та опитувань, а також можна завантажити детальний індивідуальний звіт за кожним співробітником.

Після найму персоналу на роботу є період адаптації та випробовувального терміну. Такі функції як формування плану робіт співробітника та фіксація виконання робіт на випробовувальному терміні закладені в систему Ін-агро.

Система може бути розширена функціями атестації, навчання персоналу та створенням індивідуальних планів розвитку співробітників.

1.5.3. Аналіз систем підбору персоналу Clobbi та Ін-агро

Огляд двох популярних систем з підбору персоналу Clobbi та Ін-агро показав, що дані системи мають безліч потрібних функцій та механізмів, які значно спрощують та пришвидшують процес пошуку та відбору потенційних кандидатів. Створення та розміщення вакансії з описом необхідних критеріїв кандидата на різних сайтах пошуку роботи, занесення усіх результатів співбесід до системи, аналіз наявних резюме, тестування кандидатів, прийом на роботу, супровід під час випробовувального терміну – всі ці етапи є невід’ємними складовими процесу пошуку та підбору спеціалістів.

Незважаючи на беззаперечні переваги розглянутих систем, вони направлені тільки на процес пошуку нових кандидатів. Жодна система не має підтримки аналізу анкет працевлаштованих спеціалістів компанії для створення нових команд та залучення в нових проектах за їх професійними та особистими якостями.

Для автоматизованої системи підтримки прийняття рішень з процесу підбору учасників команди проектів з розроблення програмного забезпечення запропоновано використовувати один з методів інтелектуального аналізу даних, який буде видавати найоптимальніше рішення за короткий проміжок часу.

1.6. Висновки

У першому розділі проведений загальний опис задачі управління персоналом на проектах з розроблення програмного забезпечення, а саме прийняття рішень в підборі команди проекту, зокрема підбір кожного учасника команди окремо.

На основі аналізу сучасних методів для прийняття рішень з підбору персоналу на основі його фахових та поведінкових компетенцій та дослідженню кількості програмних продуктів, що створюються у світі щоденно, показана важливість та актуальність досліджуваного питання.

Розглянуті системи підтримки прийняття рішень, як один з провідних інструментів в області управління персоналом. Основною метою АСППР є збільшення показників ефективності рішень. Розробка даних систем базується на використанні різних методів, таких як інтелектуальний аналіз даних, пошук знань в базах даних, інформаційний пошук, нейронні мережі та імітаційне моделювання.

Одним з основних завдань автоматичної або автоматизованої системи підтримки прийняття рішень є вибір серед безлічі альтернативних рішень одного, яке є найбільш релевантним та підходить якнайкраще для досягнення певної мети.

Розглянуті популярні системи з пошуку та підбору персоналу, які направлені на пошук нових співробітників компаній, проведення співбесід, адаптацію під час випробовувального терміну. Проте такі системи не мають автоматизованого аналізу анкет вже працевлаштованих співробітників для залучення їх в нові проекти.

Розглянуті способи інтелектуального аналізу даних для прийняття рішень у створенні команди проекту. Визначені основні переваги використання методів інтелектуального аналізу даних, а саме використання Data Mining для вирішення завдань класифікації, регресії, пошуку асоціативних правил і кластеризації.

Поставлена задача розробити автоматизовану систему підтримки прийняття рішень на основі модифікації одного з методів інтелектуального аналізу даних, який за швидкодією та оптимальністю прийняття рішення з підбору учасників команди проектів перевершить класичний метод.

2. МОДИФІКОВАНИЙ МЕТОД ПРИЙНЯТТЯ УПРАВЛІНСЬКИХ РІШЕНЬ З ПІДБОРУ КОМАНДИ НА ОСНОВІ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ

2.1. Використання технології глибинного аналізу тексту Text Mining для оброблення текстових даних профілю кожного потенційного учасника команди

Термін глибинний аналіз даних можна визначити як процес вилучення закономірностей у наборах даних великого об’єму з використанням поєднання методів штучного інтелекту з управлінням базами даних та статистичних методів.

Це глибоке визначення, можливо, виходить за межі потреб більшості людей. Лише деякі працюють зі штучним інтелектом; найчастіше глибинний аналіз даних передбачає поглинання великих наборів даних і пошук в них корисної інформації.

Технологія Text Mining – це сучасний інструментарій, що дозволяє аналізувати великі обсяги необроблених даних в пошуках тенденцій, взаємозв’язків та шаблонів, що допомагають прийняти стратегічні рішення [14].

Окрім того, технологія глибокого аналізу тексту є новим видом пошуку, який на відміну традиційних підходів знаходить списки документів, формально релевантних запитам [15].

Технології глибинного аналізу тексту передувала технологія видобутку даних (Data Mining), методологія та підходи якої широко використовуються і в методах Text Mining.

Для видобутку текстів цілком справедливе визначення, дане для видобутку даних одним з провідних світових експертів Григорієм Пятецьким-Шапіро з GTE Labs: «Процес виявлення в сирих даних раніше невідомих нетривіальних практично корисних і доступних інтерпретації

знань, необхідних для прийняття рішень в різних сферах людської діяльності [16].»

Як і більшість когнітивних технологій, Text Mining – це алгоритмічне виявлення раніше не відомих зав’язків і кореляцій в уже наявних текстових даних [17].

До основних елементів Text Mining можна віднести наступні технології:

- пошук за ключовими словами;
- відповіді на запити;
- кластеризація;
- класифікація;
- виділення понять;
- тематичне індексування;
- сумарізація.

Для класифікації тексту використовуються статистичні кореляції для побудови правил розміщення документів в зумовлені категорії. Кластеризація, що базується на ознаках документів, використовує математичні та лінгвістичні методи без використання визначених категорій. Результат – таксономія або візуальна карта, яка забезпечує ефективне охоплення великих обсягів даних. Семантичні мережі або аналіз зв’язків, які визначають появу дескрипторів (ключових фраз) в документі для забезпечення та навігації. Витяг фактів призначений для отримання деяких фактів з тексту з метою поліпшення класифікації, пошуку і кластеризації.

Так склалося, що завдання, яке найчастіше зустрічається в Text Mining – це класифікація, тобто віднесення об’єктів бази даних до певних категорій [18].

Фактично завдання класифікації – це класична задача розпізнавання, де за навчальною вибіркою система відносить новий об’єкт до тієї чи іншої категорії.

Особливість системи Text Mining полягає в тому, що кількість об’єктів і їх атрибутів може бути дуже великою; тому повинні бути передбачені інтелектуальні механізми оптимізації процесу класифікації.

Друге завдання – це кластеризація, тобто виділення компактних підгруп об’єктів з близькими властивостями.

Система повинна самостійно знайти ознаки і розділити об’єкти по підгрупах. Вона, як правило, передує завданню класифікації, оскільки дозволяє визначити групи об’єктів. Розрізняють два основних типи кластеризації: ієрархічний і бінарний.

Ієрархічна кластеризація полягає в побудові дерева кластерів, в кожному з яких розміщується невелика група документів.

Бінарна кластеризація забезпечує угруповання і перегляд документальних кластерів по посиланнях подібності. В один кластер поміщаються найближчі за своїми властивостями документи. В процесі кластеризації будується базис посилань від документа до документа, заснований на вагах і спільному вживанні ключових слів.

Задача кластеризації сьогодні широко застосовується при реферуванні великих документальних масивів, визначенні взаємопов’язаних груп документів, спрощенні процесу перегляду при пошуку необхідної інформації, знаходженні унікальних документів з колекції, виявленні дублікатів або дуже близьких за змістом документів.

Можна назвати ще кілька завдань технології Text Mining, наприклад, прогнозування, яке полягає в тому, щоб передбачити за значеннями одних ознак об’єкта значення інших [19].

Ще одним з важливих завдань технології Text Mining є знаходження винятків, тобто пошук об’єктів, які своїми характеристиками сильно виділяються із загальної маси [20].

Для пошуку винятків спочатку з’ясовуються середні параметри об’єктів, а потім досліджуються ті об’єкти, параметри яких найбільш сильно відрізняються від середніх значень. Подібний аналіз часто проводиться після класифікації, для того щоб з’ясувати, наскільки точним був результат.

Дещо окремо від завдання кластеризації стоїть завдання пошуку пов’язаних ознак (полів, понять) окремих документів. Від передбачення ця задача відрізняється тим, що заздалегідь не відомо, за якими саме ознаками реалізується взаємозв’язок; мета саме в тому і полягає, щоб знайти зв’язок ознак. Це завдання схоже з кластерізацією, але не по кількості документів, а по безлічі притаманних їм ознак [21].

І нарешті, для обробки й інтерпретації результатів Text Mining велике значення має візуалізація. Візуалізація даних має на увазі обробку структурованих числових даних, однак вона також є ключовою ланкою при поданні схем неструктурованих текстових документів. Зокрема, сучасні системи класу Text Mining можуть здійснювати аналіз великих масивів документів і формувати предметні покажчики понять і тем, висвітлених у цих документах. Візуалізація зазвичай використовується як засіб подання контенту всього масиву документів, а також для реалізації навігаційного механізму, який може застосовуватися при дослідженні документів та їх класів [22].

Таким чином, виходячи з вимог до модифікації методу прийняття рішень з підбору фахової команди проекту, та, враховуючи роботу з текстовим форматом вхідних даних, найбільш прийнятною для виконання завдання є розробка інтелектуальної системи підтримки прийняття рішень, базованої на механізмах інтелектуального аналізу даних з використанням глибинного аналізу даних (підготовка, кластеризація, побудова моделей) та побудови рішення (візуалізація даних).

2.2. Опис методу автоматизації прийняття управлінських рішень з підбору команди проекту на основі наївного байєсовського алгоритму

Для розроблення інтелектуальної автоматизованої системи підтримки прийняття рішень з підбору учасників команди проекту, а саме для етапів підготовки, кластеризації та побудови моделей буде використовуватися один з провідних методів глибинного аналізу даних – наївний байєсовський алгоритм.

Наївний байєсовський алгоритм (НБА) – це алгоритм класифікації, заснований на теоремі Байєса з припущенням про незалежність ознак. Іншими словами, НБА передбачає, що наявність якої-небудь ознаки в класі не пов’язана з наявністю будь-якої іншої ознаки. Навіть якщо ці ознаки залежать одна від одної або від інших ознак, в будь-якому випадку вони вносять незалежний внесок у ймовірність того, що об’єкт, який розглядається належить до класу даних ознак. У зв’язку з таким припущенням алгоритм називається «наївним» [23].

Моделі на основі НБА досить прості і вкрай корисні при роботі з дуже великими наборами даних.

При своїй простоті НБА здатний перевершити навіть деякі складні алгоритми класифікації.

Теорема Байєса використовується для отримання максимально точної оцінки ймовірності події, яка уточняється з отриманням більшої кількості інформації [24].

Принцип вирішення задачі з підбору команди проекту полягає саме в оцінці ймовірності того, який співробітник стане учасником команди. Велика кількість додаткових даних, таких як досвід роботи, кількість виконаних проектів конкретного типу, швидкість виконання задач, знання певних технологій допомагає максимально уточнити оцінку вибору учасника команди.

Теорема Байєса дозволяє розрахувати апостеріорну ймовірність $P(C|X)$ на основі $P(C)$, $P(X)$ і $P(X|C)$ за формулою (2.1):

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)}, \tag{2.1}$$

де $P(C|X)$ – апостеріорна ймовірність даного класу C (тобто даного значення цільової змінної) при даному значенні ознаки X;

$P(C)$ – апріорна ймовірність даного класу.

$P(X|C)$ – правдоподібність, тобто ймовірність даного значення ознаки при даному класі.

$P(X)$ – апріорна ймовірність даного значення ознаки.

Переваги наївного байєсовського алгоритму [25]:

1. Класифікація, в тому числі багатоскладова, виконується легко і швидко.
2. Коли допущення про незалежність виконується, НБА перевершує інші алгоритми, такі як логістична регресія (logistic regression), і при цьому вимагає менший обсяг навчальних даних.
3. НБА краще працює з категорійними ознаками, ніж з безперервними. Для безперервних ознак передбачається нормальний розподіл, що є досить сильним припущенням.

Для візуалізації отриманих даних буде використаний рекурсивний алгоритм побудови дерева рішень ID3 [26].

Вхід: S – навчаюча вибірка, B – множина базових предикатів.

Вихід: коренева верхівка дерева, побудованого по вибірці S.

Опис алгоритму, реалізованого процедурою *LearnID3*(S):

1. Якщо всі об’єкти з вибірки S належать одному класу $c \in Y$, то утворюємо новий лист v, $cv := c$, повертаємо (v);

2. Знаходимо предикат з найбільшою інформативністю за формулою (2.2):

$$\beta := arg^{max}\ \ \beta \epsilon B^{I(\beta,S)} \quad (2.2)$$

3. Розбиваємо вибірку на дві частини $S = S_0\ U\ S_1$ по предикату β за формулами (2.3) та (2.4) відповідно:

$$S_0 := \{x\ \epsilon\ S :\ \beta(x) = 0\} \quad (2.3)$$

$$S_1 := \{x\ \epsilon\ S :\ \beta(x) = 1\} \quad (2.4)$$

4. Якщо $S_0 = \otimes$ або $S_1 = \otimes$, то створюємо новий лист *v*, який належить класу, в якому знаходиться більшість об'єктів вибірки *S*, повертаємо (*v*);
5. В іншому випадку створюємо нову внутрішню верхівку *v*, де $\beta_v = \beta$;
6. Будуємо ліву гілку L_v:=*LearnID3*(S_0), будуємо праву гілку R_v:=*LearnID3*(S_1), повертаємо (*v*).

В результаті роботи алгоритму отримаємо дерево рішень, схематично зображене на рис. 2.1.

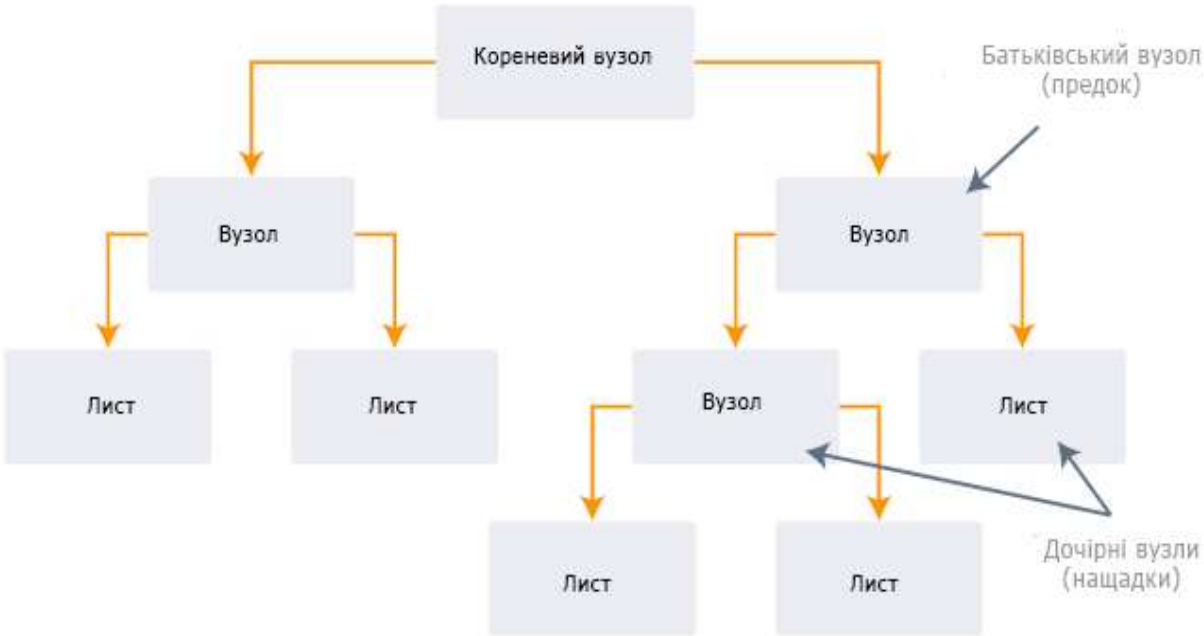


Рис. 2.1. Дерево рішень

Дерево складається з листів, що вказують на клас, і вузлів. Воно може бути використаним для класифікації об’єктів, що не увійшли в навчальну множину.

Пошук починається з кореня, поки не буде виявлений клас, відповідний об’єкту.

Однак, недоліком наївного байєсовського алгоритму при обробці даних для класифікації згідно поставленої задачі є умовна незалежність атрибутів, тобто їх незалежність від класів. Тому для вирішення задачі підбору учасників команди вирішено використовувати модифіковані байєсівські мережі (БМ).

2.3. Опис методу автоматизації прийняття управлінських рішень з підбору команди проекту на основі байєсівських мереж

Байєсівською мережею називається спрямований граф без циклів, що дозволяє представляти спільний розподіл випадкових змінних [27].

Кожен вузол графа являє випадкову змінну, а дуги – прямі залежності між ними.

Більш точно мережа описує наступні висловлювання: кожна змінна залежить тільки від безпосередніх батьків. Таким чином, граф описує обмеження на залежність змінних один від одного, що зменшує кількість параметрів спільного розподілу.

Параметри спільного розподілу кодуються в наборі таблиць для кожної змінної в формі умовних розподілів за умови на значення змінних-батьків.

Структура графа і умовні розподіли вузлів при значеннях їх батьків однозначно описують спільний розподіл всіх змінних, що дозволяє вирішувати задачу класифікації як визначення значення змінної класу, максимізуючи її умовну ймовірність при заданих значеннях вхідних змінних [28].

Байєсівська мережа схематично зображена на рис. 2.2.

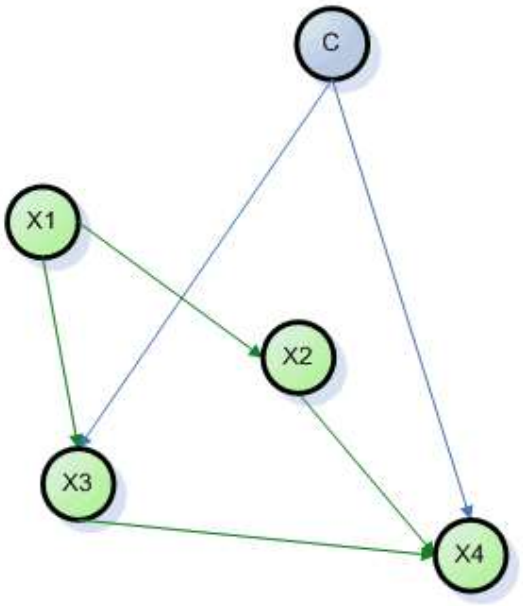


Рис. 2.2. Байєсівська мережа

Очевидно, що «наївний» алгоритм Байєса є окремим випадком байєсівської мережі, де кожна вхідна змінна залежить тільки від змінної класу, яка є єдиним коренем графа (рис. 2.3).

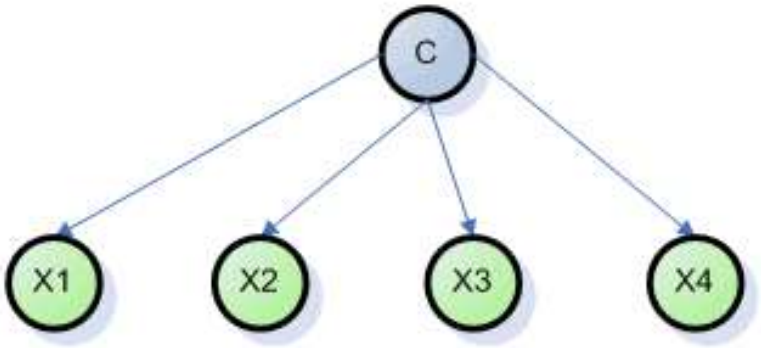


Рис. 2.3. «Наївний» алгоритм Байєса

Навчання байєсівських мереж стало одним з актуальних напрямків обчислювальної математики і до сьогодні є предметом активних досліджень. Однак, до цього часу визначення структури байєсівської мережі в загальному вигляді є складним завданням як з теоретичної, так і з обчислювальної точки зору [29-30].

Підхід до навчання байєсівських мереж в загальному вигляді має наступні недоліки:

1. Обчислювальна складність.
2. При спробі врахувати велику кількість залежностей між змінними, оцінки умовних ймовірностей набувають велику дисперсію, так як їх спільна поява в даних є малоймовірною подією. Таким чином, оцінки параметрів можуть стати недостовірними, що в підсумку може призвести до погіршення якості класифікації навіть у порівнянні з «наївним» алгоритмом Байєса.
3. Через велику кількість параметрів, модель виходить занадто орієнтованою на навчальні дані. Це призводить до дуже гарних результатів класифікації на навчальних даних і незадовільних результатів на тестових даних. Тобто модель описує не загальні закономірності в структурі даних, а скоріше набір окремих випадків в навчальній вибірці.

2.4. Опис методу автоматизації прийняття управлінських рішень з підбору команди проекту з використанням модифікованої моделі Байєса

Для вирішення проблем байєсівських мереж запропоновано використовувати обмеження на структуру графа і розглядати таке розширення «наївної» моделі Байєса, де кожен вузол додатково може мати не більше одного з батьків серед інших вхідних змінних.

Модифікований деревовидний «наївний» алгоритм Байєса схематично зображений на рис. 2.4.

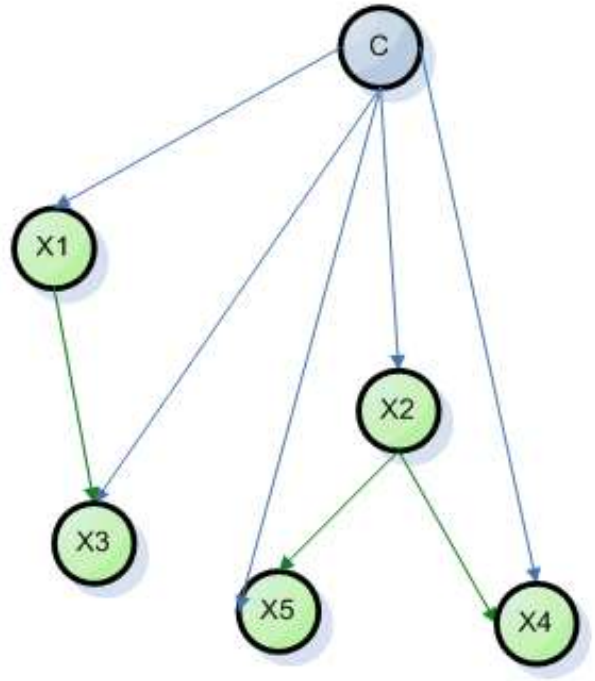


Рис. 2.4. Модифікований алгоритм Байєса

За допомогою такої модифікації буде досягнута більша точність оцінки підбору кожного з учасників команди при врахуванні великої кількості вхідних параметрів, які описують професійні навички та компетенції робітників.

Розширюємо «наївну» модель Байєса додавши взаємозв'язки між вузлами моделі.

Однак, обмежимося розглядом тільки таких графів, де у кожного вузла не може бути більше одного батька за винятком вузла *C*.

Дану модель можна описати за формулою (2.5):

$$C \in \pi(X_i), \#\pi(X_i) \leq 2, \forall i \in \{1, \dots, n\}, \pi(C) = \emptyset. \qquad (2.5)$$

Для вирішення зазначених проблем пропонується використання наступних модифікацій методу:

1. Визначення кореневого вузла, як вузла, що в найбільшій мірі впливає на результат класифікуючої змінної *C*. Як критерій ступеня впливу обрано взаємну інформацію між кожним незалежним вузлом і класифікуючим вузлом, що описано за формулою (2.6):

$$l(X_i,C) = \sum_k \sum_i P(X_i = k, C = l) log \frac{P(X_i = k, C = l)}{P(X_i = k)P(C = l)}. \quad (2.6)$$

2. Введення мінімального порогу на значення взаємної умовної інформації між вузлами. Якщо взаємна інформація між вузлами виявиться менше цього порогу, вважатимемо відповідні вузли незалежними. В якості такого порогу виберемо середнє значення взаємної умовної інформації по всіх вузлах, яке описане формулою (2.7):

$$l_{avg} = \frac{2}{n(n-1)} \sum_i \sum_{j>i} l(X_i, X_j|C). \quad (2.7)$$

Суть модифікованого «наївного» алгоритму Байєса та використання алгоритму побудови дерева рішень ID3 у площині прийняття управлінських рішень з підбору складу команди проекту, полягає у реалізації наступних етапів:

1. Обробка початкових даних проекту. На даному етапі особа, яка приймає рішення про склад команди проекту вводить початкові дані проекту (тематику, терміни, кількість членів команди і т. д.) для створення профілю проекту. По введеним даним виконується обробка початкових текстових даних за наступними критеріями для кожного члена команди: досвід роботи, кількість виконаних проектів, відповідальність, дотримання термінів, розуміння бізнес-процесів; далі отримані дані класифікуються (кластеризуються) і на їх основі формується база знань та будуються моделі.
2. Обробка вхідних даних для прийняття рішення з бази знань. Далі з бази знань робиться вибірка всіх працівників компанії і їх даних (атрибутів), які співпадають з профілем заповненого проекту.

3. Побудова дерева рішень для проекту. На основі отриманої вибірки будується дерево рішень, яке відображає склад команди проекту з розроблення програмного забезпечення.
4. Визначення оптимального складу команди проекту. Перелік учасників команди проекту, отриманий при побудові дерева рішень перевіряється механізмом оцінки якості рішення. У разі низької якості команди, проводиться уточнення початкових даних проекту і виконується побудова дерева рішень. Ці дії виконуються до тих пір, поки не буде прийнято оптимального рішення з підбору складу команди.

2.5. Висновки

У другому розділі розглянутий метод Text Mining, що є методом глибинного аналізу даних, який буде застосовуватись для аналізу вхідних текстових даних поставленої задачі – підбору фахової команди проекту з розроблення програмних продуктів.

В результаті проведеного дослідження було обрано найбільш прийнятний метод для вирішення задачі – байєсівський наївний алгоритм. Детально розглянуті та проаналізовані методи НБА, а також байєсівські мережі.

Запропоновано модифікацію методу, за допомогою якої буде досягнута більша точність оцінки підбору кожного учасника команди при врахуванні великої кількості вхідних даних, які містять в собі інформацію про досвід роботи, професійні навички, особисті якості та компетенції робітників.

Для візуалізації отриманих результатів за допомогою модифікованого алгоритму буде використаний рекурсивний алгоритм побудови дерева рішень ID3.

Описані основні етапи роботи модифікованого «наївного» алгоритму Байєса: обробка початкових даних нового проекту з розробки програмного забезпечення, обробка вхідних даних з бази знань для прийняття рішення, побудова дерева рішень та визначення оптимального складу команди фахівців.

3. ОСОБЛИВОСТІ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ АВТОМАТИЗОВАНОЇ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ З ПІДБОРУ УЧАСНИКІВ КОМАНДИ ПРОЕКТУ НА ОСНОВІ МОДИФІКОВАНОЇ МОДЕЛІ БАЙЄСА

3.1. Основні вимоги до автоматизованої системи підтримки прийняття рішень

На основі поставленої задачі з підбору учасників команди проектів розробки програмного забезпечення в першому розділі та описаному модифікованому методу на основі наївного байєсовського алгоритму в другому розділі можна виділити наступні функціональні можливості, які повинна забезпечувати автоматизована система:

- обробка вхідних текстових документів, що відповідають заданому шаблону та містять інформацію про співробітників компанії;
- графічний інтерфейс для користувача з можливістю задавати потрібні критерії пошуку спеціаліста;
- швидка обробка запитів;
- надання найоптимальнішого варіанту рішення з підбору учасників команди.

Окрім головних функцій автоматизованої системи підтримки прийняття рішень, потрібно врахувати загальні вимоги до розроблення якісних програмних продуктів.

Функціональна повнота.

Даний показник характеризує ступінь задоволення потреб користувача в сенсі можливості вирішення конкретних завдань, що стоять перед ним, тобто це набір атрибутів, що визначає призначення, основні необхідні і достатні функції програмного забезпечення, зазначені в технічному завданні.

Деталізується на такі характеристики:

- придатність для застосування (відповідність призначенню);
- точність;
- здатність взаємодіяти з середовищем;
- відповідність нормам.

Надійність.

Один з найбільш неоднозначних, суперечливих і важливих критеріїв якості програмного забезпечення, під яким розуміють перш за все забезпечення досить низької ймовірності відмови в процесі функціонування в реальному часі. Саме в силу ненадійності програмних продуктів виникають величезні витрати на супровід ПЗ, пов'язаних з розробкою і експлуатацією. Можна виділити два основних аспекти надійності:

– відсутність в готовій програмі помилок проектування і програмування;

– захищеність програми від непередбачених умов експлуатації.

Надійна програма не реагує на натискання недозволених клавіш; ненадійна може завершитися аварійно, поставивши користувача в скрутне становище. Навіть не здійснюючи неправильних дій, користувач, тим не менш, може ввести неприпустиме поєднання вхідних даних, що призводять до аварійного завершення програми. Надійна програма повинна виконувати відповідні перевірки і не допускати обчислень з цими даними, попереджаючи користувача про неможливість виконання його вимог і повертаючись в початковий стан.

Зручність.

З точки зору задоволення споживчого попиту цей показник якості може іноді мати вирішальний вплив на вибір того чи іншого програмного продукту. Цей критерій показує, наскільки зручно користувач-непрофесіонал може застосовувати його в своїй повсякденній роботі.

Зручність використання програмного продукту можна охарактеризувати наступними приватними показниками:

- зручність для користувача інтерфейсу, зручність розташування та подання часто використовуваних елементів екрану, способів введення даних;
- простота освоєння, трудові і тимчасові витрати на освоєння засобів;
- адаптованість до конкретних вимог користувача;
- кількість інформації, що пред'являється системою, яку необхідно переробити користувачеві;
- кількість дій, використаних користувачем при роботі з системою;
- простота використання.

Технічна ефективність.

Цей показник характеризує обсяги потрібних ресурсів обчислювальної системи для роботи програмного забезпечення, такі як, обсяг оперативної пам'яті, обсяг зовнішньої пам'яті, час роботи процесора, залучення компонентів операційної системи. Практично всі ці показники можуть отримати числову оцінку, і природно, можна сказати, що з двох програм з однаковими функціональними властивостями краще та, яка споживає менше ресурсів. Таким чином, ефективність ПЗ слід розглядати за такими двома параметрами:

- швидкодію і час відгуку;
- споживання ресурсів, вимоги до оптимального розміру зовнішньої і оперативної пам'яті, типу і продуктивності процесора, що забезпечують прийнятний рівень продуктивності.

Адаптивність.

Розуміється можливість модифікації ПЗ в разі зміни параметрів вирішуваних завдань, операційного оточення, апаратурного складу або

взагалі типу ЕОМ. При зміні характеру і параметрів вирішуваних завдань, як правило, потрібно змінити кількісні і структурні параметри програми, тобто змінити її вихідний текст. Адаптованість ПЗ характеризують такі приватні показники:

- переносимість;
- сумісність з версіями операційних систем (можливість роботи в середовищі різних версій однієї і тієї ж ОС);
- структурованість;
- замінність;
- легкість інсталяції;
- відповідність нормам переносимості та інсталяції;
- впровадженність.

Основні вимоги, які в подальшому стануть критерієм оцінки ефективності розроблюваної системи:

1. Функціональна повнота. Передбачає реалізацію в повному обсязі поставлених завдань зі зберігання, введення, редагування і надання інформації. Створена система повинна в повній мірі виконувати завдання підтримки прийняття рішення, зберігати потрібну інформацію, і виключати зайву.

2. Зручність. Створений програмний продукт повинен бути ергономічно зручний, володіти зрозумілим, зручним «дружнім» інтерфейсом. Здійснювати максимально-можливий набір дій при мінімальному втручанні користувача.

3. Надійність. Створений програмний продукт повинен мати високу відмовостійкість, бути стабільний в роботі і надавати достовірну інформацію.

4. Технічна ефективність. Досягається шляхом використання найменших показників витрати пам’яті.

5. Адаптивність. Дозволяє з мінімальними витратами проводити модифікацію продукту, в залежності від додаткових потреб.

3.2. Опис обраних засобів розроблення автоматизованої системи підтримки прийняття рішень

Для розроблення автоматизованої системи підтримки прийняття рішень з підбору учасників команди проектів, що виконує глибинний аналіз текстових даних та їх класифікацію, обрано середовище розробки Microsoft Visual Studio.

При виборі середовища реалізації порівнюють програмні продукти і користуються різними засобами розробки додатків. Використання можливостей засобів розробки додатків дозволяє автоматизувати процес розробки. Інструментальні засоби дозволяють:

- створювати інтерфейс, використовуючи стандартні компоненти;
- передавати управління процесам, в залежності від стану системи;
- створювати оболонки для баз даних, як і самі бази даних;
- розробляти більш надійні програми шляхом обробки виняткових ситуацій що виникають при некоректній роботі програми.

Сучасні засоби розробки характеризуються параметрами:

- підтримка об’єктно-орієнтованого стилю програмування;
- можливість використання CASE-технологій, як для проектування розроблюваної системи, так і для розробки моделей реляційних баз даних;
- використання візуальних компонент для наочного проектування інтерфейсу;
- підтримка БД.

Вище перерахованими властивостями володіє мова програмування Visual C#. Це мова і середовище програмування, що відноситься до класу RAD (Rapid Application Development «Засіб швидкої розробки додатків»).

В Visual C# входять локальний SQL-сервер, генератори звітів, бібліотеки візуальних компонентів та інші інструменти, необхідні для того, щоб відчувати себе абсолютно впевненим при професійній розробці програмного забезпечення для Windows-середовища.

Вигоди від проектування застосунків в середовищі Visual Studio за допомогою Visual C#:

1) усувається необхідність в повторному введенні даних;
2) забезпечується узгодженість проекту та його реалізації;
3) збільшується продуктивність розробки і переносимість програм.

Visual C# дозволяє розробляти додатки швидким процесом лише за рахунок засобів візуалізації, бо візуальне програмування дає можливість зображати ці об’єкти на екрані монітора до виконання самої програми. Без візуального програмування процес відображення вимагає написання фрагмента коду, що створює і налаштовує об’єкт «за місцем». Побачити закодовані об’єкти раніше було можливо тільки в ході виконання програми.

При такому підході досягнення того, щоб об’єкти виглядали і поводилися заданим чином, стає втомливим процесом, який вимагає неодноразових виправлень програмного коду з наступною прогонкою програми і спостереження за тим, що в результаті вийшло.

Завдяки засобам візуальної розробки можна працювати з об’єктами, тримаючи їх перед очима й одержуючи результати практично одразу. Здатність бачити об’єкти такими, якими вони з’являються в ході виконання програми, знімає необхідність проведення безлічі операцій вручну, що характерно для роботи в середовищі, що не володіє візуальними засобами, незалежно від того, є воно об’єктно-орієнтованим чи ні.

Після того, як об’єкт поміщений в форму середовища візуального програмування, всі його атрибути відразу відображаються у вигляді коду,

який відповідає об’єкту як одиниці, що виконується під час роботи програми.

Розміщення об’єктів в Visual C# пов’язане з тісним взаємозв’язком між об’єктами і реальним програмним кодом.

Об’єкти поміщаються в форму, при цьому код, який відповідає об’єктам, автоматично записується в вихідний файл.

Код компілюється, забезпечуючи істотно вищу продуктивність, ніж візуальне середовище, яке інтерпретує інформацію лише в ході виконання програми.

Зокрема, Visual C# дозволяє додавати до вікон поля введення, меню, командні кнопки, перемикачі, прапорці, списки, лінійки прокрутки, а також діалогові вікна для вибору файлу або каталогу. Такі компоненти зазвичай називають елементами управління.

Переваги Visual C# в порівнянні з аналогічними програмними продуктами:

– швидкість розробки програми;

– висока продуктивність розробленого додатка;

– низькі вимоги розробленого додатка до ресурсів комп’ютера;

– нарощуванність за рахунок вбудовування нових компонент і інструментів в середовище Visual C#;

– можливість розробки нових компонент і інструментів власними засобами Visual C#.

Для коректної роботи автоматизованої системи необхідно, щоб виконувалися наступні вимоги до апаратного забезпечення та кількості вільної оперативної пам’яті:

1. Платформа Windows: процесор 80386SX або вище (рекомендується 80486), пам’ять 8Мб (рекомендується 12Мб), простір на диску 8Мб + 1-3Мб для однієї моделі.

2. Платформа UNIX: пам’ять 32Мб і більше (16 * число користувачів), простір на диску 30Мб + 20Мб при інсталяції + 1-3Мб для однієї моделі.

3.3. Опис розробленої автоматизованої системи підтримки прийняття рішень з підбору учасників команди проекту

В рамках поставленої задачі розроблене наступне програмне забезпечення:

- автоматизована система підтримки прийняття рішень з підбору учасників команди, що реалізує модифікований наївний метод класифікації Байєса з обмеженням на структуру графа, де кожен вузол додатково може мати не більше одного з батьків серед інших вхідних змінних;
- застосунок із графічним інтерфейсом для прийняття рішення по вибору складу команди проекту.

Відповідно до опису модифікованого методу була спроектована архітектура системи для його реалізації. Архітектура автоматизованої системи підтримки прийняття рішень ASPPR показана на рис. 3.1.

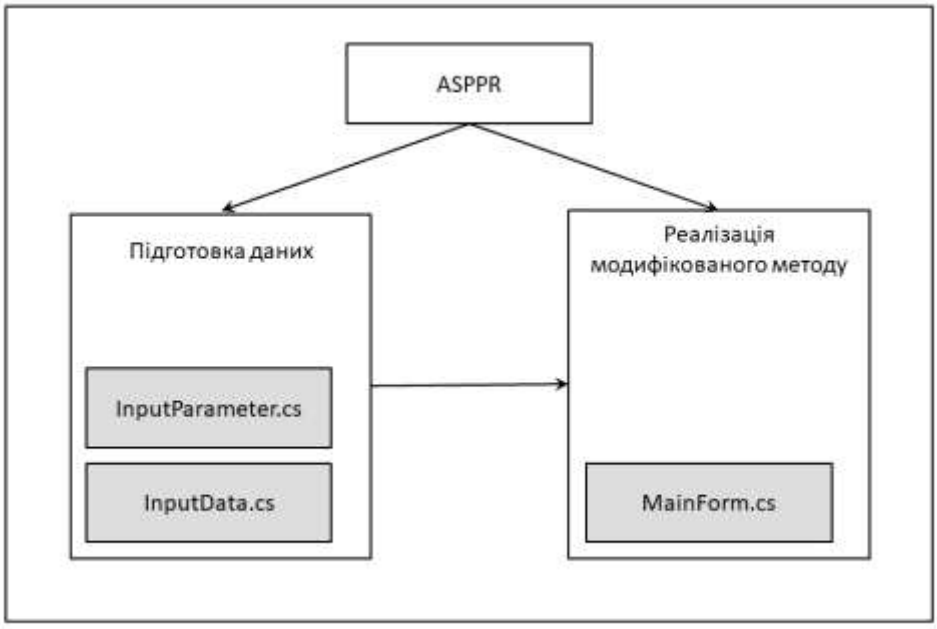


Рис. 3.1. Архітектура системи ASPPR

Архітектура автоматизованої системи складається з наступних модулів:

1. Модуль попередньої підготовки текстових даних, який реалізовано за допомогою класів InputParameter.cs і InputData.cs.
2. Модуль реалізації модифікованого методу, що базується на «наївному» алгоритмі класифікації Байєса та реалізований за допомогою класу MainForm.cs.

Діаграма класів автоматизованої системи підтримки прийняття рішень ASPPR показана на рис. 3.2.

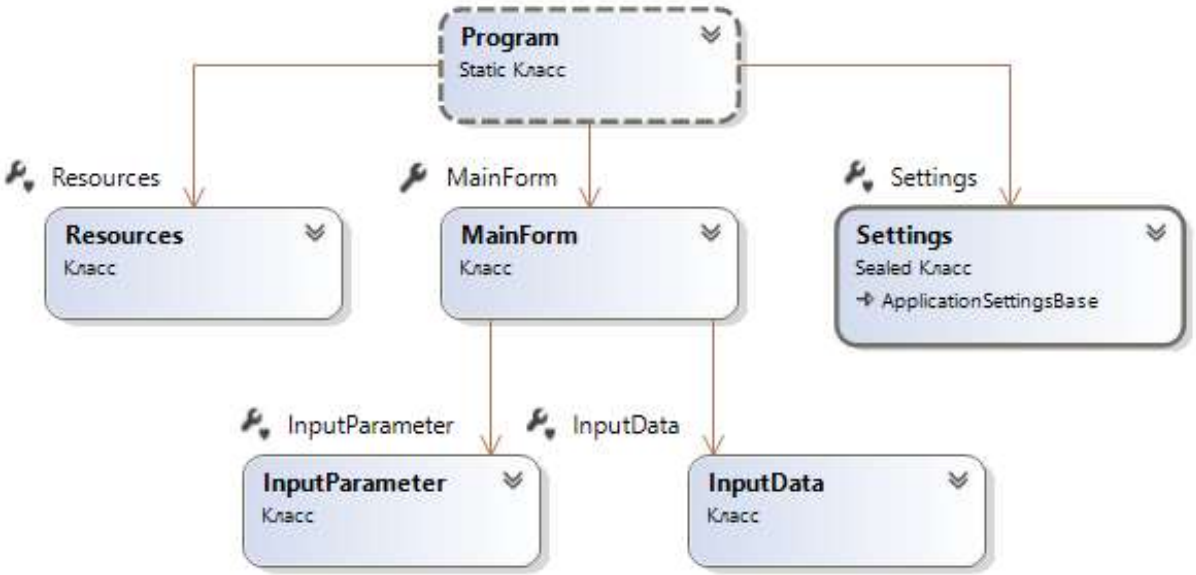


Рис. 3.2. Діаграма класів автоматизованої системи

Модулі InputParameter.cs та InputData.cs., що організують попередню обробку текстових документів та їх класифікацію організовані у ієрархію зі спільним предком у вигляді класів.

Вхідними даними для аналізу професійних та особистих якостей співробітників та вибору найкращого учасника команди за заданими критеріями є анкети співробітників, що зберігаються у текстовому форматі.

Шаблон анкети для співробітника показаний на рис. 3.3.

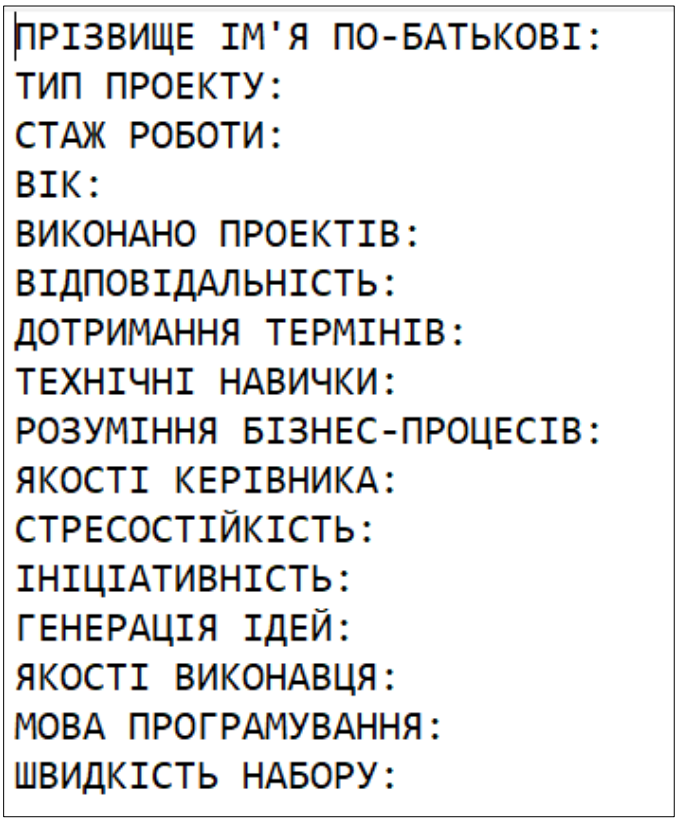


Рис. 3.3. Шаблон анкети для співробітника

Розглянемо програмний код реалізації класу InputParameter, наведений у лістингу 3.1, який виконує класифікацію параметрів для використання у модифікованому методі, що базується на «наївному» алгоритмі класифікації Байєса.

Лістинг 3.1. Програмний код класу InputParameter

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
```

```
namespace ASPPR
{
    class InputParameter
    {
        public static Dictionary<string, List<string>> GetAllSavedParams()
        {
         Dictionary<string, List<string>> output =
            new Dictionary<string, List<string>>();
         using (StreamReader sr = new StreamReader(@"..\..\Parameter.txt",
Encoding.Default))
           {
           string[]_tmpAllData = sr.ReadToEnd().Replace("\r", string.Empty)
             .Replace(".", string.Empty).Split(new char[] { '\n' },
StringSplitOptions.RemoveEmptyEntries);
                  foreach (string _tmpStr in _tmpAllData)
                  {
                      string[] _tmpParamNWords = _tmpStr.Split(':');
                      string _tmpNameParam = _tmpParamNWords[0];
                      output.Add(_tmpNameParam,
_tmpParamNWords[1].Split(',').ToList());
                      }
                  }
                  return output;
              }
          }
      }
```

Результатом використання даного модулю є список параметрів для роботи модифікованого методу, що базується на «наївному» алгоритмі класифікації Байєса.

Розглянемо програмний код методу вводу даних InputData для роботи класифікатора Байєса, що наведений у лістингу 3.2.

Лістинг 3.2. Програмний код класу InputData

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
```

```
namespace ASPPR
{
    class InputData
    {
        public static Dictionary<string, List<string>> GetOneSavedData()
        {    Dictionary<string,List<string>>output_one=newDictionary<string,
List<string>>();
             using (StreamReader sr_one = new StreamReader(@"..\..\Особова
справа №1.txt", Encoding.Default))
              {    string[]_tmpAllData  =  sr_one.ReadToEnd().Replace("\r",
string.Empty)
      .Replace(".",    string.Empty).Split(new    char[]    {    '\n'    },
StringSplitOptions.RemoveEmptyEntries);
                    foreach (string _tmpStr in _tmpAllData)
                    {   string[] _tmpDataWords = _tmpStr.Split(':');
                        string _tmpNameParam = _tmpDataWords[0];
                        output_one.Add(_tmpNameParam,
_tmpDataWords[1].Split(',').ToList());
                    }
                } return output_one;
            }
    public static Dictionary<string, List<string>> GetTwoSavedData()
    { Dictionary<string, List<string>> output_two = new Dictionary<string,
List<string>>();
                using       (StreamReader       sr_two       =       new
StreamReader(@"..\..\Особова справа №2.txt", Encoding.Default))
                { string[] _tmpAllData = sr_two.ReadToEnd().Replace("\r",
string.Empty)
                        .Replace(".",  string.Empty).Split(new  char[]  {
'\n' }, StringSplitOptions.RemoveEmptyEntries);
                    foreach (string _tmpStr in _tmpAllData)
                    {   string[] _tmpDataWords = _tmpStr.Split(':');
                        string _tmpNameParam = _tmpDataWords[0];
                        output_two.Add(_tmpNameParam,
_tmpDataWords[1].Split(',').ToList());
                    }
                }
                return output_two;
            }
      public static Dictionary<string, List<string>> GetTreeSavedData()
            {
      Dictionary<string, List<string>> output_tree = new Dictionary<string,
List<string>>();
```

```
      using (StreamReader sr_tree = new StreamReader(@"..\..\Особова справа
№3.txt", Encoding.Default))
{ string[] _tmpAllData = sr_tree.ReadToEnd().Replace("\r", string.Empty)
                          .Replace(".",  string.Empty).Split(new  char[]  {
'\n' }, StringSplitOptions.RemoveEmptyEntries);
                      foreach (string _tmpStr in _tmpAllData)
                      {
                          string[] _tmpDataWords = _tmpStr.Split(':');
                          string _tmpNameParam = _tmpDataWords[0];
                          output_tree.Add(_tmpNameParam,
_tmpDataWords[1].Split(',').ToList());
                      }
                  }
                  return output_tree;
              }
      public static Dictionary<string, List<string>> GetFourSavedData()
              {   Dictionary<string,   List<string>>   output_four   =   new
Dictionary<string, List<string>>();
      using (StreamReader sr_four = new StreamReader(@"..\..\Особова справа
№4.txt", Encoding.Default))
                  {            string[]            _tmpAllData            =
sr_four.ReadToEnd().Replace("\r", string.Empty)
                          .Replace(".",  string.Empty).Split(new  char[]  {
'\n' }, StringSplitOptions.RemoveEmptyEntries);
                      foreach (string _tmpStr in _tmpAllData)
                      {   string[] _tmpDataWords = _tmpStr.Split(':');
                          string _tmpNameParam = _tmpDataWords[0];
                          output_four.Add(_tmpNameParam,
_tmpDataWords[1].Split(',').ToList());
                      }
                  }
                  return output_four;
              }
          }
      }
```

Результатом використання даного модулю є перелік даних для роботи модифікованого методу, що базується на «наївному» алгоритмі класифікації Байєса.

В якості кінцевого застосунку для автоматизованого прийняття рішення з вибору учасників команди проекту розроблено застосунок з графічним інтерфейсом. Такий варіант реалізації користувацького інтерфейсу дозволяє легко керувати введенням початкових даних та отримувати найоптимальніше рішення.

Структура інтерфейсу застосунку складається з однієї форми, на якій розміщені всі елементи введення вхідних даних нового проекту та вікно з відображенням прийнятого системою рішення (рис. 3.4).

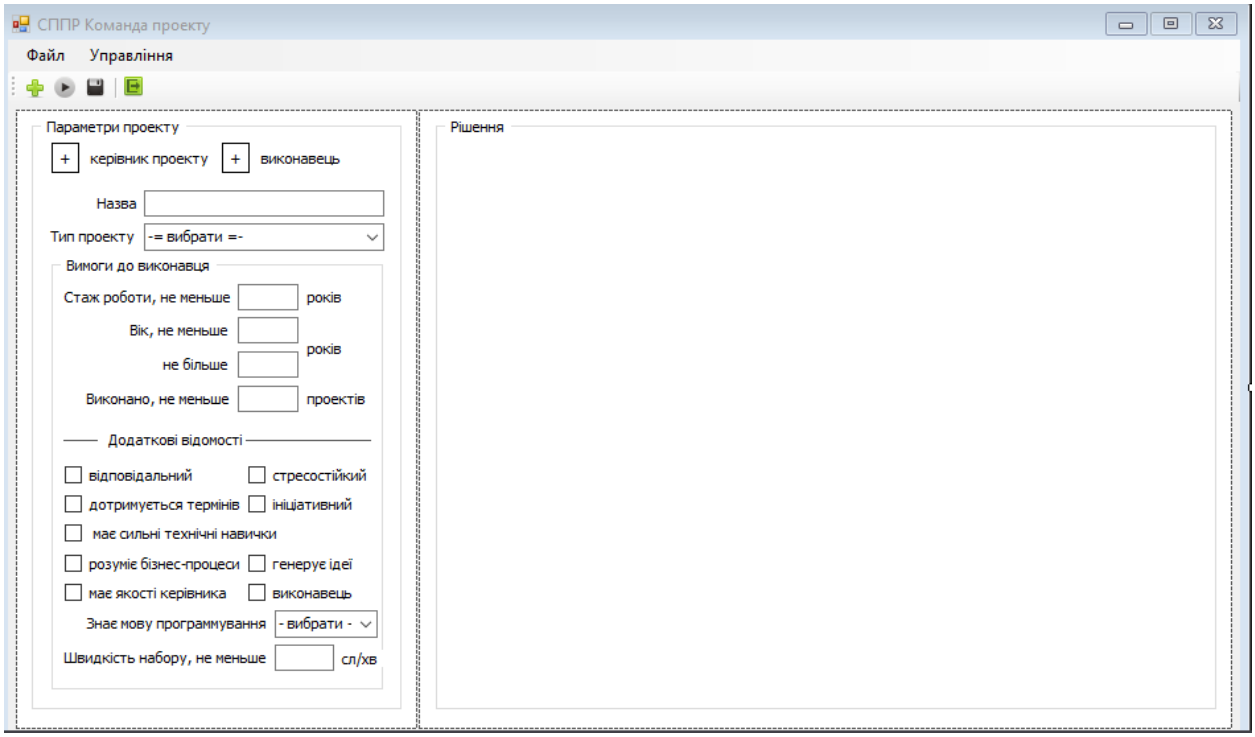


Рис. 3.4. Графічний інтерфейс автоматизованої системи

За допомогою графічного інтерфейсу людина, яка хоче сформувати команду проекту може легко та швидко заповнити всі необхідні вхідні дані та запустити роботу системи. Після аналізу та підбору співробітника, який найкраще задовольняє вимоги проекту, система покаже прізвище та ім’я обраного учасника команди.

Застосунок отримує вхідні данні та відображає результат роботи у діалоговому режимі.

3.4. Висновки

У третьому розділі були описані основні вимоги до розробленої автоматизованої системи підтримки прийняття рішень на основі модифікованого наївного методу Байєса, які були сформовані згідно постановки задачі.

Також були висунуті загальні вимоги до якісної програмної системи, зокрема, функціональна повнота, надійність, зручність, технічна ефективність та адаптованість.

Розглянута мова програмування Visual C# та середовище розробки Microsoft Visual Studio, які використовувались для розроблення системи. Описані основні переваги використання даних засобів створення програмного забезпечення.

В процесі розроблення автоматизованої системи підтримки прийняття рішень з підбору учасників команди проекту були описані наступні елементи:

- архітектура системи, що реалізує попереднє оброблення текстових документів та реалізацію модифікованого методу прийняття рішень;
- основні класи системи, їх реалізація та програмний код;
- структура графічного інтерфейсу системи.

Розроблена система приймає на вхід вимоги до співробітника, аналізує їх та видає найоптимальніший результат.

4. АНАЛІЗ ЕФЕКТИВНОСТІ АВТОМАТИЗОВАНОЇ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ З ПІДБОРУ УЧАСНИКІВ КОМАНДИ ПРОЕКТУ НА ОСНОВІ МОДИФІКОВАНОЇ МОДЕЛІ БАЙЄСА

4.1. Критерії оцінки та аналіз ефективності модифікованого методу

Метою дослідження було дослідити створений модифікований метод автоматизації прийняття рішень з підбору учасників проекту на основі байєсівської моделі, на базі якого будо створено автоматизовану систему підтримки прийняття рішень.

Основними критеріями оцінки ефективності модифікованого методу є точність прийняття рішення, а саме максимальна відповідність в запиті на вакансію в підборі конкретного співробітника на конкретний проект за заданими початковими параметрами.

Другим за значенням важливим критерієм оцінки ефективності методу є його швидкодія. Адже чим швидше буде працювати метод, тим більше учасників команди можна підібрати за одиницю часу.

Зокрема, при аналізі ефективності методу потрібно аналізувати співвідношення критерію точності до зменшення швидкодії роботи методу.

В основі модифікованого байєсівського методу лежить аналіз та класифікація текстової інформації.

Приклад текстових даних заповненої анкети співробітника, які зберігаються у текстових документах та використовуються для аналізу системою:

«ПРІЗВИЩЕ ІМ'Я ПО-БАТЬКОВІ:Іванов Іван Петрович

ТИП ПРОЕКТУ:Веб-сайт

СТАЖ РОБОТИ:2

ВІК:22

ВИКОНАНО ПРОЕКТІВ:2

ВІДПОВІДАЛЬНІСТЬ:Так

ДОТРИМАННЯ ТЕРМІНІВ:Так

ТЕХНІЧНІ НАВИЧКИ:Так

РОЗУМІННЯ БІЗНЕС-ПРОЦЕСІВ:Ні

ЯКОСТІ КЕРІВНИКА:Ні

СТРЕСОСТІЙКІСТЬ:Так

ІНІЦІАТИВНІСТЬ:Так

ГЕНЕРАЦІЯ ІДЕЙ:Ні

ЯКОСТІ ВИКОНАВЦЯ:Так

МОВА ПРОГРАМУВАННЯ:Java

ШВИДКІСТЬ НАБОРУ:120».

Для порівняння критеріїв точності та швидкодії модифікованого методу та класичної моделі Байєса використовувалася розроблена автоматизована система на основі модифікованого методу та аналогічна автоматизована система, на основі класичної моделі. В класичній моделі Байєса вузли графів можуть мати більше одного батька, не враховуючи кореневий вузол *C*.

Для оцінювання методів була взята різна кількість анкет (особових справ) співробітників, які займаються розробкою програмного забезпечення.

Модель Байєса розраховує ймовірність належності ознаки до певного класу. В контексті дослідженої задачі, шукається ймовірність того, що конкретний співробітник якнайкраще підходить для участі у новому проекті.

Числове значення ймовірності лежить у діапазоні від 0 до 1 або від 0% до 100% відповідно.

Для порівняння точності визначення ймовірності того, що співробітник відповідає заданим вимогам розглядаємо значення від 0 до 1. Чим ближче отримане значення до одиниці – тим більше виконавець задовольняє критерії проекту.

Для моніторингу швидкодії розробленої автоматизованої системи підтримки прийняття рішень на основі модифікованого методу та окремо на основі класичної моделі Байєса був використаний стандартний клас мови C# Stopwatch Class, який надає набір методів та властивостей, які призначені для точного вимірювання витраченого часу.

Результати проведення тестування методу з метою визначення критеріїв точності та швидкодії показані у табл. 4.1.

Таблиця 4.1

Результати аналізу ефективності модифікованого методу

Кількість анкет співробітників	Автоматизована система на основі модифікованого методу		Автоматизована система на основі класичної моделі	
	Точність визначення	Час, секунди	Точність визначення	Час, секунди
10	0,83	0,10	0,80	0,21
20	0,82	0,20	0,81	0,33
30	0,85	0,25	0,80	0,35
40	0,85	0,30	0,83	0,38
50	0,91	0,31	0,80	0,40
100	0,90	0,51	0,80	0,64
500	0,94	1,20	0,82	2,20

Відповідно до наведених даних тестування методу можна зробити висновок, що розроблена автоматизована система підтримки прийняття рішень з підбору учасників проекту на основі модифікованого байєсівського методу показує кращі результати за критерієм точності та швидкодії, ніж АСППР, створена з використанням класичної моделі Байєса.

Для більш детального аналізу показника точності визначення учасника команди побудовано графік порівняння для обох методів.

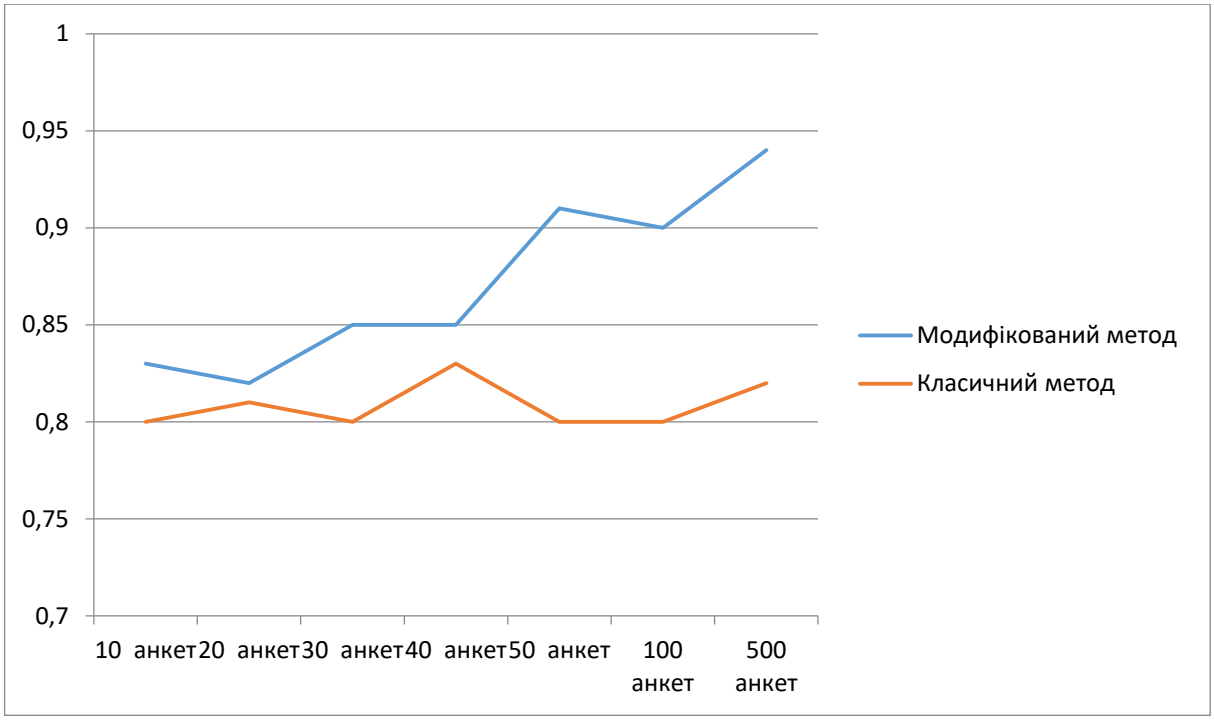


Рис. 4.1. Графік точності визначення для методів

Проаналізувавши графік точності, можна побачити, що автоматизована система підтримки прийняття рішень на основі модифікованого байєсовського методу в порівнянні з програмною системою на основі класичної моделі Байєса на різних обсягах вхідних даних обрала учасника команди краще:

- при аналізі 10 анкет показник точності в 1,04 рази вище;
- при аналізі 20 анкет показник точності в 1,01 рази вище;
- при аналізі 30 анкет показник точності в 1,06 рази вище;
- при аналізі 40 анкет показник точності в 1,02 рази вище;
- при аналізі 50 анкет показник точності в 1,14 рази вище;
- при аналізі 100 анкет показник точності в 1,13 рази вище;
- при аналізі 500 анкет показник точності в 1,15 рази вище.

Можна зробити висновок, що модифікований метод показує вищі показники точності підбору учасників, ніж класичний метод. Особливо це помітно на великих обсягах вхідних текстових даних.

Другим важливим критерієм оцінки модифікованого методу є швидкодія його роботи. Для більш детального аналізу швидкодії визначення учасника команди побудовано графік порівняння для обох методів.

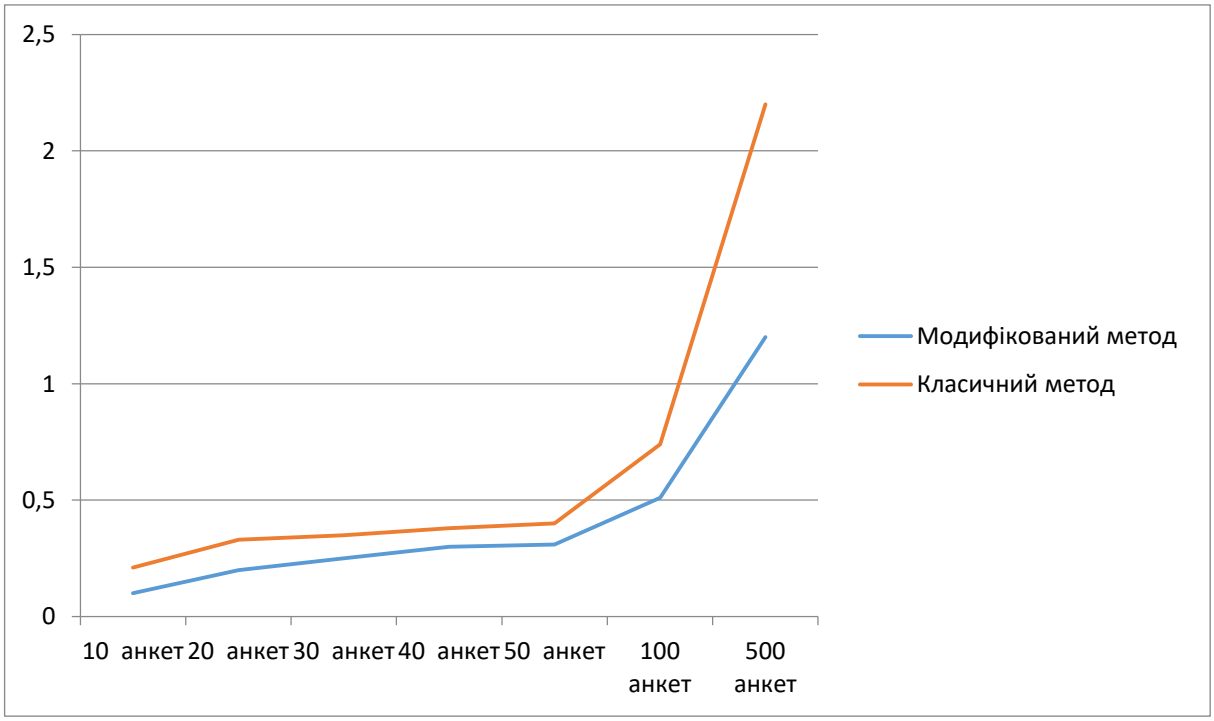


Рис. 4.2. Графік швидкодії роботи методів

Проаналізувавши графік швидкодії роботи, можна побачити, що автоматизована система підтримки прийняття рішень на основі модифікованого байєсовського методу в порівнянні з програмною системою на основі класичної моделі Байєса на різних обсягах вхідних даних потребує меншого часу роботи:

- при аналізі 10 анкет швидкість роботи в 0,48 рази менша;
- при аналізі 20 анкет швидкість роботи в 0,60 рази менша;
- при аналізі 30 анкет швидкість роботи в 0,71 рази менша;
- при аналізі 40 анкет швидкість роботи в 0,79 рази менша;
- при аналізі 50 анкет швидкість роботи в 0,78 рази менша;
- при аналізі 100 анкет швидкість роботи в 0,80 рази менша;
- при аналізі 500 анкет швидкість роботи в 0,55 рази менша.

Модифікований метод показав кращий показник швидкодії.

4.2. Тестування та аналіз розробленої автоматизованої системи

Для тестування автоматизованої системи підтримки прийняття рішень використовується функціональне тестування, яке є тестуванням на підставі виконання сценаріїв використання системи.

В ході виконання функціонального тестування вирішуються наступні завдання:

- розробка стратегії тестування;
- розробка або доопрацювання тестових сценаріїв;
- виконання тестів (в тому числі «димне тестування» – перевірка первинної працездатності системи і «тестування критичного шляху» – гарантія виконання основного сценарію за будь-яких умов).

Для виконання тестування автоматизованої системи вирішено використовувати методику «швидкого тестування», яке проводиться з метою переконання в тому, що основні функції програми працюють правильно. Такий метод не включає глибокого тестування кожної функції окремо.

Вимоги, які є критеріями оцінки ефективності розробленої автоматизованої системи:

1. Функціональна повнота системи.

Основним з критеріїв оцінки розробленої системи є її функціональна повнота, тобто відповідність всім поставленим вимогам, які були зазначені в постановці задачі.

Створена система в повній мірі виконує задачу підтримки прийняття рішення з підбору учасників команди проектів та надає результати високої точності за короткий проміжок часу.

Можна зробити висновок, що критерій «Функціональна повнота» для системи відповідає оцінці 10 балів за 10-ти бальною шкалою.

2. Оцінка зручності та стійкості системи.

Тестування на зручність застосування здійснюється на різних етапах розробки програмного продукту (зразок дизайну на папері, програмні прототипи і кінцевий продукт), щоб забезпечити зворотний зв’язок з користувачами. Це допомагає вдосконалювати весь проект в цілому, скорочує кількість помилок, проводить порівняльний аналіз продуктів і версій, а також підтверджує відповідність продукту пропонованим йому вимогам.

У всіх режимах розроблена система працює коректно, без зауважень. Графічний інтерфейс визнаний досить зручним і дружнім та включає всі необхідні елементи для роботи та розуміння функціональних можливостей системи. Кількість дій користувача, а також витрачений час при роботі з програмою зведено до необхідного мінімуму.

Можна зробити висновок, що критерій «Зручність» для системи відповідає оцінці 9 балів за 10-ти бальною шкалою.

3. Надійність розробленої системи.

В результаті тестування виявлено, що система виключає помилкове введення даних, так як користувач ці дані не вносить вручну, а обирає в графічному інтерфейсі.

Можна зробити висновок, що критерій «Надійність» для системи реалізований на рівні оцінки 10 балів за 10-ти бальною шкалою.

4. Технічна ефективність.

Цей показник характеризує обсяги потрібних ресурсів обчислювальної системи для роботи програмного забезпечення. Технічну ефективність розробленої автоматизованої системи можна оцінити за параметрами швидкодії та споживання ресурсів.

Аналіз модифікованого методу на різних наборах вхідних даних показав високі показники швидкодії, тобто час прийняття рішення системою є досить малим та лежить в межах однієї секунди.

Для зберігання розробленої програми потрібно всього 5,08 Мб вільного простору на диску, а отже програма не займає багато пам’яті, що є гарним показником споживання ресурсів комп’ютера.

Можна зробити висновок, що критерій «Технічна ефективність» для системи реалізований на рівні оцінки 10 балів за 10-ти бальною шкалою.

Таким чином, всі характеристики вимог програмної системи виконані, а отже вимоги функціональної повноти, зручності експлуатації, надійності та технічної ефективності реалізовані.

Зведені оцінки реалізованих вимог показників по 10 бальній шкалі відображені у табл. 4.2.

Таблиця 4.2

Оцінка розробленої автоматизованої системи

Характеристика показника	Оцінка	Перенормована оцінка	Ваговий коефіцієнт
Функціональна повнота	10	1,0	0,3
Зручність	9	0,9	0,1
Надійність	10	1,0	0,3
Технічна ефективність	10	1,0	0,3

Розрахуємо нормований інтегральний критерій розробленої програмної системи за вказаними вище значеннями параметрів за формулою (4.1) і порівняємо його з планованим.

$$\eta = 0{,}3 * 1{,}0 + 0{,}1 * 0{,}9 + 0{,}3 * 1{,}0 + 0{,}3 * 1{,}0 = 0{,}99 \qquad (4.1)$$

Розроблена автоматизована система підтримки прийняття рішень за нормованим критерієм має оцінку 0,99, що є більше, ніж 0,95, а отже всі вимоги, що були висунуті до системи виконуються в повній мірі.

4.3. Вдосконалення автоматизованої системи підтримки прийняття рішень з підбору учасників команди проекту

В даному дослідженні було визначено один з методів інтелектуального аналізу даних, який є найбільш оптимальним для розв'язання задачі підбору учасників команди проектів на основі аналізу текстових анкет кожного претендента.

Класична модель Байєса була модифікована за алгоритмом, описаним у другому розділі та використана для створення автоматизованої системи підтримки прийняття рішень, що показала кращі результати за критеріями точності та швидкості.

Автоматизовану систему на основі модифікованого методу можна використовувати вже зараз за прямим призначенням, але також можна виділити шляхи подальшого дослідження та вдосконалення системи і методу.

4.3.1. Використання неструктурованих текстових даних

Розроблений метод був протестований та проаналізований на основі структурованих текстових документів одного формату, що мають спільний шаблон.

Не завжди в компаніях дані про співробітників можуть зберігатися в однакових за змістом документах. Особливо, якщо в компанії працює досить велика кількість працівників різного профілю та дуже часто проходить набір нових співробітників.

В подальшому вдосконаленні можна розглядати анкети співробітників у вигляді неструктурованих текстових даних. Потрібно налаштувати метод таким чином, щоб обробка текстових даних та пошук потрібної інформації відбувався коректно, зокрема, щоб показник швидкодії не набував помітно більшого значення.

Окрім коректного аналізу неструктурованих текстових даних можна також додати функцію перетворення таких документів за одним шаблоном для зручності в подальшому використанні рекрутерами і менеджерами та безпосередньо системою.

4.3.2. Аналіз завантаженості працівника

Розроблена автоматизована система приймає найоптимальніше рішення з вибору учасника команди за його професійними та особистими якостями. Проте, якщо обраний системою співробітник вже є залученим на іншому проекті, такий вибір автоматично стане неактуальним.

У випадку, коли не можна підібрати іншого співробітника, а обраний працює на іншому проекті, можна розглянути три варіанти та обрати найоптимальніший:

- дочекатися, коли працівник закінчить роботу над поточним проектом;
- розглянути можливість паралельного виконання двох проектів, якщо терміни дозволяють витрачати на виконання задач кожного проекту меншу кількості годин на день;
- зняти працівника з поточного проекту та замінити іншим, якщо таке рішення буде оптимальним для обох проектів.

Багато провідних організацій з розробки програмного забезпечення користуються тайм-трекерами – програмами, в яких відображається завантаженість працівника на проектах та його прогрес по кожній задачі.

При інтеграції даних з такої програми до автоматизованої системи підтримки прийняття рішень з підбору учасників команди можна отримати найбільш достовірні результати, адже окрім професійних компетенцій, інформації про досвід та володіння певними технологіями, система буде аналізувати завантаженість працівників.

4.3.3. Підбір учасників, які працювали разом

Дуже частими є випадки командної роботи, коли учасники працюють доволі довго на одному проекті та звикають один до одного, підлаштовуються до стилю роботи, краще розуміють один одного, можуть більш ефективно оцінювати задачі, проводити командні статуси та розподіляти обов'язки.

Якщо досвід показує, що учасники таких команд працюють краще та продуктивніше і з психологічної точки зору у них не виникає непорозумінь та конфліктів між собою, то варто розглядати такий самий або схожий склад команди для подальшого залучення у нових проектах організації.

Для підбору таких команд потрібно буде включати додаткові вхідні дані учасників, які працювали між собою та модифікувати метод таким чином, щоб він коректно обробляв нові дані та видавав найоптимальніше рішення.

В такому рішенні з вибору учасника має бути зазначена інформація, з якими колегами він працював раніше, зокрема наскільки успішними були такі проекти.

Якщо спостерігається зворотна ситуація, коли деякі учасники команди не можуть знайти спільну мову та від цього страждає якість виконання задач проекту, потрібно в майбутньому мінімізувати можливість створення команд з таким самим складом.

З цією метою потрібно також зберігати, аналізувати та обробляти інформацію про склад команди проекту, на якому виникали конфліктні ситуації.

Подальше вдосконалення модифікованого методу та автоматизованої системи підтримки прийняття рішень може значно покращити показник точності вибору учасника команди, так як буде врахована додаткова інформація про кожного співробітника.

4.4. Висновки

У четвертому розділі були описані критерії оцінки ефективності модифікованого методу:

- точність визначення учасника команди – максимальна відповідність запиту на вакансію;
- швидкодія роботи.

Було проведене тестування за цими критеріями для розробленої автоматизованої системи підтримки прийняття рішень на основі модифікованого байєсівського методу та розробленим програмним забезпеченням на основі класичної моделі Байєса.

Тестування проводилося для різної кількості анкет співробітників, від 10 до 500 вхідних текстових документів.

На основі отриманих результатів можна зробити висновок, що модифікований метод показав кращі результати за критеріями точності та швидкодії.

Точність визначення учасника модифікованим методом в середньому в 1,08 рази вища за класичний метод.

Швидкодія роботи модифікованого методу в середньому в 0,67 разів менша за роботу класичного методу.

Проведене тестування та аналіз автоматизованої системи за наступними критеріями:

- функціональна повнота;
- зручність;
- надійність;
- технічна ефективність.

Розрахований нормований інтегральний критерій розробленої автоматизованої системи має оцінку 0,99, що є більше, ніж допустиме значення, отже всі вимоги, що були висунуті до системи виконуються в повній мірі.

Були розглянуті шляхи подальшого вдосконалення методу та системи, а саме:

- використання неструктурованих текстових даних як вхідних даних, що містять інформацію про учасників;
- врахування поточної завантаженості кожного співробітника при аналізі для залучення його на новий проект;
- можливість підбору учасників, які вже працювали разом в команді.

ВИСНОВКИ

В першому розділі роботи описана загальна задача прийняття рішень при підборі учасників команди проекту по розробці програмного забезпечення.

Розглянуті автоматизовані системи підтримки прийняття рішень, які характеризуються можливістю реалізації вироблення рекомендацій для прийняття різних рішень, при цьому, сильно знижується рівень «людського фактору», який дуже часто приводить до суб'єктивної оцінки.

Для розв'язання досліджуваної проблеми вирішено розробити автоматизовану систему підтримки прийняття рішень.

Були розглянуті провідні методи інтелектуального аналізу даних, на основі яких розроблюються АСППР.

Сформульована постановка задачі та визначений формат вхідних даних – текстові документи, створені за спільним шаблоном, що містять інформацію про досвід роботи, професійні та особисті якості співробітників.

Проаналізовані схожі програмні системи з підбору персоналу Clobbi та Ін-агро. Визначено їх основні недоліки для розв'язання поставленої задачі.

У другому розділі роботи розглянута технологія глибинного аналізу тексту Text Mining для оброблення текстових даних профілю кожного потенційного учасника команди.

Для модифікації обрана модель Байєса. Запропоновано використовувати обмеження на структуру графа і розглядати таке розширення моделі Байєса, де кожен вузол додатково може мати не більше одного з батьків серед інших вхідних змінних для досягнення більшої точності підбору учасника проекту.

В третьому розділі роботи висунуті основні вимоги до автоматизованої системи підтримки прийняття рішень:

- обробка вхідних текстових документів, що відповідають заданому шаблону та містять інформацію про співробітників компанії;
- графічний інтерфейс для користувача з можливістю задавати потрібні критерії пошуку спеціаліста;
- швидка обробка запитів;
- надання найоптимальнішого варіанту рішення з підбору учасників команди.

Описані обрані засоби розроблення АСППР та безпосередньо описана реалізація системи: спроектована архітектура, описані програмні класи, визначений шаблон вхідних даних.

В якості кінцевого застосунку для автоматизованого прийняття рішення з вибору учасників команди проекту розроблено застосунок з графічним інтерфейсом.

В четвертому розділі роботи описані критерії оцінки для аналізу ефективності методу: точність вибору рішення та швидкодія роботи методу.

Було проведено тестування розробленої автоматизованої системи підтримки прийняття рішень на основі модифікованого методу та створеного програмного забезпечення на основі класичного методу на різних обсягах вхідних даних. Результати тестування були проаналізовані та зроблено висновок, що модифікований метод показав кращі результати, а саме:

- точність визначення учасника модифікованим методом в середньому в 1,08 рази вища за класичний метод;
- швидкодія роботи модифікованого методу в середньому в 0,67 разів менша за роботу класичного методу.

Також було проведене тестування всієї системи за вимогами, визначеними в третьому розділі: функціональна повнота системи, зручність та стійкість системи; надійність; технічна ефективність. Результати тестування системи показали високі показники та підтвердили, що система відповідає всім висунутим вимогам та вирішує поставлену задачу в повній мірі.

Визначені шляхи подальшого вдосконалення методу та системи:

- можливість використання неструктурованих текстових даних;
- врахування поточної завантаженості кожного співробітника для залучення його на новий проект;
- можливість підбору учасників, які раніше працювали разом в команді.

За результатами роботи розроблено модифікований метод для автоматизації прийняття рішень при підборі учасників команди проектів, який показав найкращий результат за критерієм точності прийняття рішення.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Шапиро В.Д. Управление проектами [Текст] // [В. Д. Шапиро, Н. И. Ильин и др.]. / – СПб.: Газпром, 1996. – 610 с
2. Keen, Peter. 1980. Decision support systems: a research perspective. Cambridge, Massachusetts: Center for Information Systems Research, Alfred P. Sloan School of Management
3. Turban, E. Decision support and expert systems: management support systems. — Englewood Cliffs, N.J.: Prentice Hall, 1995.
4. Robert Clemen. Making Hard Decisions: An Introduction to Decision Analysis, 2nd edition. Belmont CA: Duxbury Press, 1996
5. John Wiley & Sons – Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data. – 2014-12-19
6. Ian H. Witten, Eibe Frank and Mark A. Hall. Data Mining: Practical Machine Learning Tools and Techniques. — 3rd Edition. — Morgan Kaufmann, 2011. — P. 664
7. Nisbet, Robert; Elder, John; Miner, Gary (2009); Handbook of Statistical Analysis & Data Mining Applications, Academic Press/Elsevier
8. Poncelet, Pascal; Masseglia, Florent; and Teisseire, Maguelonne (editors) (October 2007); "Data Mining Patterns: New Methods and Applications", Information Science Reference
9. Theodoridis, Sergios; and Koutroumbas, Konstantinos (2009); Pattern Recognition, 4th Edition, Academic Press
10. Witten, Ian H.; Frank, Eibe; Hall, Mark A. (30 January 2011). Data Mining: Practical Machine Learning Tools and Techniques (3 ed.). Elsevier
11. Менеджмент персоналу: навч. посіб. / З. О. Коваль ; М-во освіти і науки України, Нац. ун-т «Львів. політехніка». — Львів: Вид-во Львів. політехніки, 2014. — 452 с.

12. Johnason, P. (2009). HRM in changing organizational contexts. In D. G. Collings & G. Wood (Eds.), Human resource management: A critical approach (pp. 19–37)
13. Paauwe, J., & Boon, C. (2009). Strategic HRM: A critical review. In D. G. Collings, G. Wood (Eds.) & M.A. Reid, Human resource management: A critical approach (pp. 38-54)
14. Survey of Text Mining I: Clustering, Classification, and Retrieval / Ed. by M. W. Berry. — 2004. — Springer, 2003. — 261 с.
15. Do Prado H. A. Emerging Technologies of Text Mining: Techniques and Applications / Ed. by H. A. Do Prado, E. Ferneda. — Idea Group Reference. — Springer, 2007. — 358 с.
16. G. Piatetsky-Shapiro, Knowledge Stream Partners
17. Aggarwal C. C., Zhai C. Mining Text Data. — Springer, 2012. — 527 с.
18. Feldman, R., and Sanger, J. (2006). The Text Mining Handbook. New York: Cambridge University Press
19. Indurkhya, N., and Damerau, F. (2010). Handbook Of Natural Language Processing, 2nd Edition. Boca Raton, FL: CRC Press
20. Miner, G., Elder, J., Hill. T, Nisbet, R., Delen, D. and Fast, A. (2012). Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications. Elsevier Academic Press
21. McKnight, W. (2005). "Building business intelligence: Text data mining in business intelligence". DM Review, 21-22
22. Srivastava, A., and Sahami. M. (2009). Text Mining: Classification, Clustering, and Applications. Boca Raton, FL: CRC Press
23. Maron, M. E – Automatic Indexing: An Experimental Inquiry. Maron, M. E. – 1961
24. Р. М. Онищук, І. М. Терещенко – Опис алгоритму для побудови наївного байєсовського класифікатора. – 2015

25. Gelman, Andrew; Carlin, John B.; Stern, Hal S.; Dunson, David B.; Vehtari, Aki; Rubin, Donald B. (2013). Bayesian Data Analysis (вид. III). CRC Press
26. Mitchell, Tom Michael (1997). Machine Learning. New York, NY: McGraw-Hill. с. 55–58
27. Pearl, Judea. 2000. Causality: Models, Reasoning, and Inference. Cambridge University Press. – ISBN 978-0-521-77362-1
28. Згуровський М. З., Бідюк П. І., Терентьєв О. М., Просянкіна-Жарова Т. І. Байєсівські мережі в системах підтримки прийняття рішень — Київ : ТОВ «Видавниче Підприємство «Едельвейс», 2015. — 300 с.
29. Rish, Irina. (2001). «An empirical study of the naive Bayes classifier». IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence
30. Mozina M, Demsar J, Kattan M, & Zupan B. (2004). «Nomograms for Visualization of Naive Bayesian Classifier». In Proc. of PKDD-2004, pages 337—348

ДОДАТКИ

Додаток 1

Копія презентації

Модифікований метод автоматизації прийняття управлінських рішень на основі інтелектуального аналізу даних

Виконала студентка 6 курсу, групи КП-71мн,
Чорна Оксана Вікторівна

Керівник старший викладач кафедри ПЗКС, к.т.н.,
Люшенко Леся Анатоліївна

Мета роботи

Розроблення ефективного методу автоматизації підтримки прийняття рішень з підбору учасників команди проекту на основі наївної моделі Байєса за критерієм точності та швидкодії отримуваних результатів.

Актуальність теми

Обробка та аналіз текстових документів з метою подальшого прийняття рішення є розповсюдженою задачею у програмній інженерії.

Розглядається вирішення задачі автоматизованого підбору учасників команди проектів з розробки програмного забезпечення.

Об’єкт та предмет дослідження

Об’єктом дослідження є автоматизація прийняття рішень при підборі учасників команди проекту на основі текстових даних.

Предметом дослідження є методи та алгоритми автоматизації прийняття рішень на основі інтелектуального аналізу текстових даних.

Задачі

- Провести аналіз методів інтелектуального аналізу;
- Обґрунтувати вибір критеріїв оптимізації модифікованого методу;
- Розробити та дослідити модифікований метод на основі наївної моделі Байєса для автоматизації прийняття управлінських рішень;
- Розробити автоматизовану систему підтримки прийняття управлінських рішень на основі модифікованого методу;
- Проаналізувати ефективність модифікованого методу в порівнянні з класичним на основі визначених критеріїв оптимізації.

Визначення

Автоматизована система підтримки прийняття рішень – програмне забезпечення, метою якого є прийняття рішень в слабо-структурованих, неструктурованих або нетривіальних завданнях.

Методи аналізу даних в АСППР

- інтелектуальний аналіз даних;
- інформаційний пошук ;
- імітаційне моделювання;
- методи міркування на основі прецедентів;
- нейронні мережі.

Методи інтелектуального аналізу даних

Методологія ІАД поєднує аналіз середніх тенденцій і тенденцій відхилення від середніх (тобто аналіз рідкісних, можливо, навіть унікальних випадків) в рамках тільки кількісного підходу.

Аналіз існуючих систем підбору персоналу

Огляд двох популярних систем з підбору персоналу Clobbi та Ін-агро показав, що дані системи мають безліч потрібних функцій та механізмів, які значно спрощують та пришвидшують процес пошуку та відбору потенційних кандидатів.
Незважаючи на беззаперечні переваги розглянутих систем, вони направлені тільки на процес пошуку нових кандидатів.

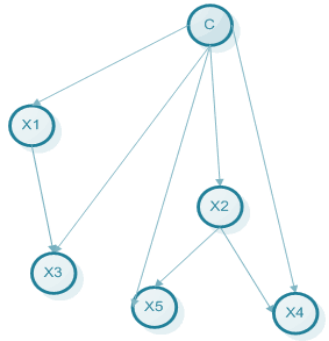
Модифікований метод автоматизації підтримки прийняття управлінських рішень

Наївний байєсовський алгоритм (НБА) – це алгоритм класифікації, заснований на теоремі Байєса з припущенням про незалежність ознак.

Байєсівською мережею називається спрямований граф без циклів, що дозволяє представляти спільний розподіл випадкових змінних.

Модифікований метод автоматизації підтримки прийняття управлінських рішень

Для вирішення проблем байєсівських мереж запропоновано використовувати обмеження на структуру графа і розглядати таке розширення «наївної» моделі Байєса, де кожен вузол додатково може мати не більше одного з батьків серед інших вхідних змінних.



Модифікований метод автоматизації підтримки прийняття управлінських рішень

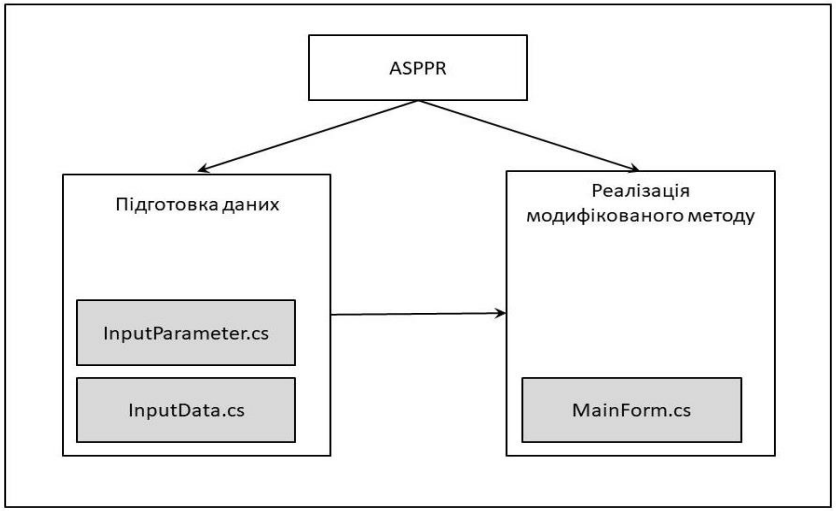
1) Обробка початкових даних проекту;
2) Обробка вхідних даних для прийняття рішення з бази знань;
3) Вибір оптимального складу учасників команди.

Вимоги до АСППР на основі модифікованого методу

- обробка вхідних текстових документів;
- графічний інтерфейс для користувача з можливістю задавати потрібні критерії пошуку спеціаліста;
- швидка обробка запитів;
- надання оптимального варіанту рішення з підбору учасників команди.

Реалізація АСППР на основі модифікованого методу

- Visual Studio 2017
- C#



Реалізація АСППР на основі модифікованого методу

ПРІЗВИЩЕ ІМ'Я ПО-БАТЬКОВІ:
ТИП ПРОЕКТУ:
СТАЖ РОБОТИ:
ВІК:
ВИКОНАНО ПРОЕКТІВ:
ВІДПОВІДАЛЬНІСТЬ:
ДОТРИМАННЯ ТЕРМІНІВ:
ТЕХНІЧНІ НАВИЧКИ:
РОЗУМІННЯ БІЗНЕС-ПРОЦЕСІВ:
ЯКОСТІ КЕРІВНИКА:
СТРЕСОСТІЙКІСТЬ:
ІНІЦІАТИВНІСТЬ:
ГЕНЕРАЦІЯ ІДЕЙ:
ЯКОСТІ ВИКОНАВЦЯ:
МОВА ПРОГРАМУВАННЯ:
ШВИДКІСТЬ НАБОРУ:

Аналіз ефективності АСППР з підбору учасників команди на основі модифікованого методу

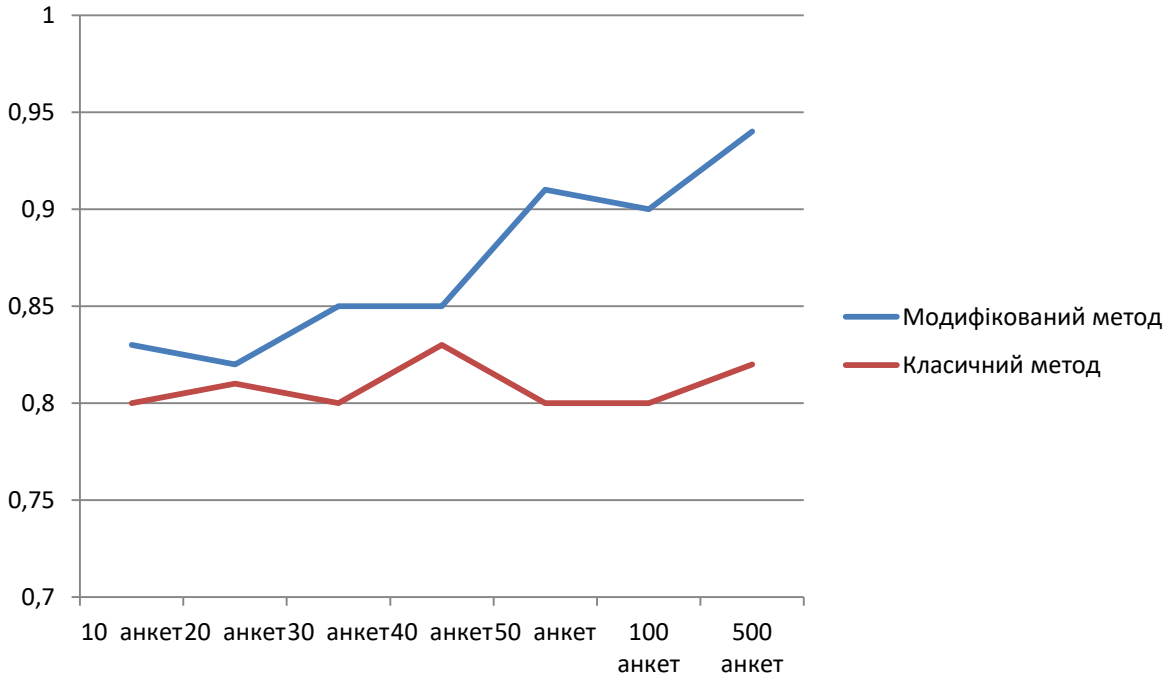
Критерії:

- Точність – максимальна відповідність в запиті на вакансію в підборі конкретного співробітника;
- Швидкодія роботи методу.

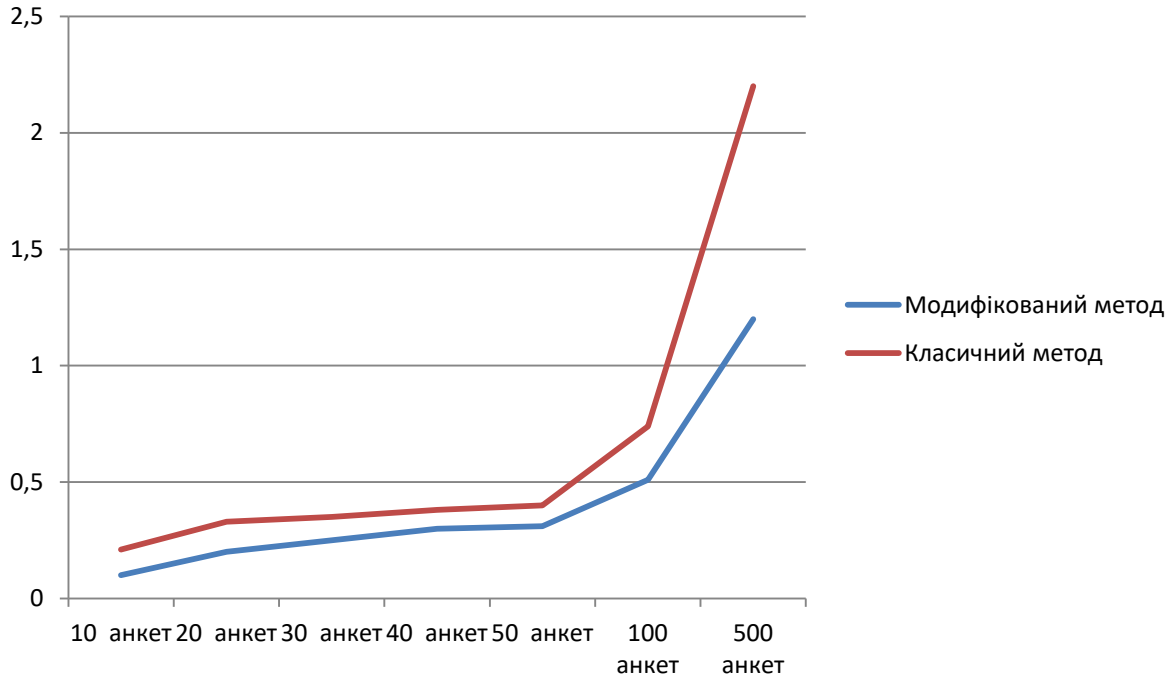
Результати аналізу ефективності методу

Кількість анкет	Автоматизована система на основі модифікованого методу		Автоматизована система на основі класичної моделі	
	Точність визначення	Час, секунди	Точність визначення	Час, секунди
10	0,83	0,10	0,80	0,21
20	0,82	0,20	0,81	0,33
30	0,85	0,25	0,80	0,35
40	0,85	0,30	0,83	0,38
50	0,91	0,31	0,80	0,40
100	0,90	0,51	0,80	0,64
500	0,94	1,20	0,82	2,20

Результати аналізу точності методу



Результати аналізу швидкодії методу



Шляхи подальшого дослідження

- Використання неструктурованих текстових даних;
- Аналіз завантаженості працівника;
- Підбір учасників, які працювали разом;

- Подальша модифікація методу для підтримки прийняття інших управлінських рішень.

Наукова новизна

Розроблений модифікований метод Байєса для автоматизації підтримки прийняття управлінських рішень з підбору учасників команди проекту, який на відміну від класичного методу видає результат

- за критерієм точності на 10-13% вище
- в середньому на 20% швидше

Апробація роботи

- «Використання систем підтримки прийняття рішень для ефективного управління проектами в програмній інженерії» на XI науковій конференції магістрантів та аспірантів «Прикладна математика та комп’ютинг» (ПМК-2018-2)

- «Модифікований метод автоматизації прийняття управлінських рішень на основі інтелектуального аналізу даних при створенні команди управління проектами» в міжнародному науковому журналі «Керуючі системи та комп’ютери» — 2019. — №4

Висновки

- Поставлена задача для вирішення проблеми автоматизації прийняття рішень з підбору учасників команди проектів програмної інженерії;
- Розглянуті методи інтелектуального аналізу даних, які лежать в основі розроблення АСППР;
- Проаналізовані схожі системи підбору персоналу та визначені їх недоліки;
- Визначені критерії ефективності для аналізу методів;

Висновки

- Запропонований модифікований метод для вирішення поставленої задачі;
- Розроблено автоматизовану систему підтримки прийняття на основі модифікованого методу Байєса;
- Проведене тестування та аналіз модифікованого та класичного методу;
- Запропоновано шляхи подальшого дослідження.

Дякую за увагу!

Додаток 2

Лістинг тексту програми

Лістинг 1. Фрагмент тексту класу MainForm, що реалізує модифікований метод

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;

namespace SPPRResours
{
    public partial class MainForm : Form
    {

        // вхідні параметри
        string A01 = string.Empty;      // піб виконавця
        string A02 = string.Empty;
        string A03 = string.Empty;
        string A04 = string.Empty;
        string A1 = string.Empty;       // тип проекту
        int A2 = 0;                     // стаж
        int A3min = 0; int A3max = 0;   // вік від - до
        int A4 = 0;                     // виконано проектів
        int A5 = 0;                     // відповідальність
        int A6 = 0;                     // дотримання термінів
        int A7 = 0;                     // технічні навички
        int A8 = 0;                     // розуміння бп
        int A9 = 0;                     // якості керівника
        int A10 = 0;                    // стресостійкість
        int A11 = 0;                    // ініціативність
        int A12 = 0;                    // генерація ідей
        int A13 = 0;                    // якості виконавця
        string A14 = string.Empty;      // мова програмування
        int A15 = 0;                    // швидкість набору

        //string _document_1 = string.Empty;

        Dictionary<string, List<string>> _dictParameter;
        Dictionary<string, List<string>> _dictDataOne;
        Dictionary<string, List<string>> _dictDataTwo;
        Dictionary<string, List<string>> _dictDataTree;
        Dictionary<string, List<string>> _dictDataFour;

        float R1 = 0.0F; float R2 = 0.0F; float R3 = 0.0F; float R4 = 0.0F;

        public MainForm()
        {
            InitializeComponent();
            _dictParameter = InputParameter.GetAllSavedParams();
            List<string> _ParamsWords = _dictParameter["KEYWORDS"];
        }

        private void завершитиРоботуToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            this.Close();
```

```
        }

        private void toolStripButton6_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void MainForm_Load(object sender, EventArgs e)
        {

        }

        private void EmptyParametr()
        {

            checkBox1.Checked = false; checkBox2.Checked = false;
checkBox3.Checked = false;
            checkBox4.Checked = false; checkBox5.Checked = false;
checkBox6.Checked = false;
            checkBox7.Checked = false; checkBox8.Checked = false;
checkBox9.Checked = false;
        }

        private void button2_Click(object sender, EventArgs e)
        {
            EmptyParametr();
            checkBox6.Checked = true;
            checkBox1.Checked = true;
            checkBox5.Checked = true;
            checkBox8.Checked = true;
            checkBox7.Checked = true;
            checkBox3.Checked = true;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            EmptyParametr();
            textBox4.Text = "3";
            checkBox2.Checked = true;
            checkBox1.Checked = true;
            checkBox4.Checked = true;
            checkBox5.Checked = true;
            checkBox9.Checked = true;
            textBox5.Text = "90";
        }

        private void toolStripButton2_Click(object sender, EventArgs e)
        {
            EmptyParametr();
            textBoxName.Text = string.Empty;
            comboBox1.Text = "-= вибрати =-";
            comboBox2.Text = "- вибрати -";
            textBox6.Clear();
        }

        private void новийПроектToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            EmptyParametr();
            textBoxName.Text = string.Empty;
            comboBox1.Text = "-= вибрати =-";
            comboBox2.Text = "- вибрати -";
            textBox6.Clear();
        }
```

```
        private void comboBox1_SelectedIndexChanged(object sender,
EventArgs e)
        {
            textBox6.Text = "Прийняття рішення по складу команди
виконавців";
            textBox6.Text += System.Environment.NewLine + "Для проекту: " +
textBoxName.Text;
            textBox6.Text += System.Environment.NewLine + "Тип проекту: " +
comboBox1.Text;
            textBox6.Text += System.Environment.NewLine + string.Empty;
            textBox6.Text += System.Environment.NewLine + "Оберіть
керівника проекту";
        }

        private void toolStripButton3_Click(object sender, EventArgs e)
        {

            A1 = comboBox1.Text;
            if (textBox1.Text != "") { A2 = Convert.ToInt16(textBox1.Text);
} else { A2 = 0; };
            if (textBox2.Text != "") { A3min =
Convert.ToInt16(textBox2.Text); } else { A3min = 0; };
            if (textBox3.Text != "") { A3max =
Convert.ToInt16(textBox3.Text); } else { A3max = 0; };
            if (textBox4.Text != "") { A4 = Convert.ToInt16(textBox4.Text);
} else { A4 = 0; };
            if (checkBox1.Checked) { A5 = 1; } else { A5 = 0; };
            if (checkBox2.Checked) { A6 = 1; } else { A6 = 0; };
            if (checkBox4.Checked) { A7 = 1; } else { A7 = 0; };
            if (checkBox5.Checked) { A8 = 1; } else { A8 = 0; };
            if (checkBox6.Checked) { A9 = 1; } else { A9 = 0; };
            if (checkBox3.Checked) { A10 = 1; } else { A10 = 0; };
            if (checkBox7.Checked) { A11 = 1; } else { A11 = 0; };
            if (checkBox8.Checked) { A12 = 1; } else { A12 = 0; };
            if (checkBox9.Checked) { A13 = 1; } else { A13 = 0; };
            A14 = comboBox2.Text;
            if (textBox5.Text != "") { A15 =
Convert.ToInt16(textBox5.Text); } else { A15 = 0; };

            R1 = 0.0F; R2 = 0.0F; R3 = 0.0F; R4 = 0.0F;
            // вибір керівника проекту
            if (A9 == 1)
            {
                _dictDataOne = InputData.GetOneSavedData();
                List<string> _Params19 = _dictDataOne["ЯКОСТІ КЕРІВНИКА"];
                if (_Params19[0] == "Так")
                {
                    List<string> _Params10 = _dictDataOne["ПРІЗВИЩЕ ІМ'Я
ПО-БАТЬКОВІ"];
                    A01 = _Params10[0];
                    if (A1 != "-= вибрати =-")
                    {
                        List<string> _Params11 = _dictDataOne["ТИП
ПРОЕКТУ"];
                        if (A1 == _Params11[0])
                        {
                            if (A5 == 1)
                            {
                                List<string> _Params15 =
_dictDataOne["ВІДПОВІДАЛЬНІСТЬ"];
                                if (_Params15[0] == "Так") { R1 += w1_6; R1
+= w9_6; }
                            }
```

```
                            if (A6 == 1)
                            {
                                List<string> _Params16 =
_dictDataOne["ДОТРИМАННЯ ТЕРМІНІВ"];
                                if (_Params16[0] == "Так") { R1 += w1_5; R1
+= w0_5; }
                            }
                            if (A8 == 1)
                            {
                                List<string> _Params18 =
_dictDataOne["РОЗУМІННЯ БІЗНЕС-ПРОЦЕСІВ"];
                                if (_Params18[0] == "Так") { R1 += w1_8; R1
+= w0_8; R1 += w9_8; }
                            }
                            if (A4 > 0)
                            {
                                List<string> _Params14 =
_dictDataOne["ВИКОНАНО ПРОЕКТІВ"];
                                if (Convert.ToInt16(_Params14[0]) > A4) {
R1 += w1_4; R1 += w0_4; }
                            }
                            if (A14 != "- вибрати -")
                            {
                                List<string> _Params114 =
_dictDataOne["МОВА ПРОГРАМУВАННЯ"];
                                if (A14 == _Params114[0]) { R1 += w1_14; R1
+= w0_14; }
                            }
                            if (A15 > 0)
                            {
                                List<string> _Params115 =
_dictDataOne["ШВИДКІСТЬ НАБОРУ"];
                                if (Convert.ToInt16(_Params115[0]) > A15) {
R1 += w1_15; R1 += w0_15; }
                            }

                        }
                        if (A2 > 0)
                        {
                            List<string> _Params12 = _dictDataOne["СТАЖ
РОБОТИ"];
                            if (Convert.ToInt16(_Params12[0]) > A12) { R1
+= w0_2; }
                        }
                        if (A3min > 0)
                        {
                            List<string> _Params13 = _dictDataOne["ВІК"];
                            if ((Convert.ToInt16(_Params13[0]) > A3min) &
(Convert.ToInt16(_Params13[0]) < A3max)) { R1 += w0_3; }
                        }
                        if (A7 == 1)
                        {
                            List<string> _Params17 = _dictDataOne["ТЕХНІЧНІ
НАВИЧКИ"];
                            if (_Params17[0] == "Так") { R1 += w0_7; }
                        }
                        if (A10 == 1)
                        {
                            List<string> _Params110 =
_dictDataOne["СТРЕСОСТІЙКІСТЬ"];
                            if (_Params110[0] == "Так") { R1 += w0_10; R1
+= w9_10; }
                        }
                        if (A11 == 1)
```

```
                        {
                            List<string> _Params111 =
_dictDataOne["ІНІЦІАТИВНІСТЬ"];
                            if (_Params111[0] == "Так") { R1 += w0_11; R1
+= w9_11; }
                        }
                        if (A12 == 1)
                        {
                            List<string> _Params112 =
_dictDataOne["ГЕНЕРАЦІЯ ІДЕЙ"];
                            if (_Params112[0] == "Так") { R1 += w0_12; R1
+= w9_12; }
                        }

                    }
                }
                 else
                 {
                     R1 = 0.0F;
                 }

                 _dictDataTwo = InputData.GetTwoSavedData();
                 List<string> _Params29 = _dictDataTwo["ЯКОСТІ КЕРІВНИКА"];
                 if (_Params29[0] == "Так")
                 {
                     List<string> _Params20 = _dictDataTwo["ПРІЗВИЩЕ ІМ'Я
ПО-БАТЬКОВІ"];
                     A02 = _Params20[0];
                     if (A1 != "-= вибрати =-")
                     {
                         List<string> _Params21 = _dictDataTwo["ТИП
ПРОЕКТУ"];
                         if (A1 == _Params21[0])
                         {
                             if (A5 == 1)
                             {
                                 List<string> _Params25 =
_dictDataTwo["ВІДПОВІДАЛЬНІСТЬ"];
                                 if (_Params25[0] == "Так") { R2 += w1_6; R2
+= w9_6; }
                             }
                             if (A6 == 1)
                             {
                                 List<string> _Params26 =
_dictDataTwo["ДОТРИМАННЯ ТЕРМІНІВ"];
                                 if (_Params26[0] == "Так") { R2 += w1_5; R2
+= w0_5; }
                             }
                             if (A8 == 1)
                             {
                                 List<string> _Params28 =
_dictDataTwo["РОЗУМІННЯ БІЗНЕС-ПРОЦЕСІВ"];
                                 if (_Params28[0] == "Так") { R2 += w1_8; R2
+= w0_8; R2 += w9_8; }
                             }
                             if (A4 > 0)
                             {
                                 List<string> _Params24 =
_dictDataTwo["ВИКОНАНО ПРОЕКТІВ"];
                                 if (Convert.ToInt16(_Params24[0]) > A4) {
R2 += w1_4; R2 += w0_4; }
                             }
                             if (A14 != "- вибрати -")
                             {
```

```
                                List<string> _Params214 =
_dictDataTwo["МОВА ПРОГРАМУВАННЯ"];
                                if (A14 == _Params214[0]) { R2 += w1_14; R2
+= w0_14; }
                            }
                            if (A15 > 0)
                            {
                                List<string> _Params215 =
_dictDataTwo["ШВИДКІСТЬ НАБОРУ"];
                                if (Convert.ToInt16(_Params215[0]) > A15) {
R2 += w1_15; R2 += w0_15; }
                            }

                        }
                        if (A2 > 0)
                        {
                            List<string> _Params22 = _dictDataTwo["СТАЖ
РОБОТИ"];
                            if (Convert.ToInt16(_Params22[0]) > A12) { R2
+= w0_2; }
                        }
                        if (A3min > 0)
                        {
                            List<string> _Params23 = _dictDataTwo["ВІК"];
                            if ((Convert.ToInt16(_Params23[0]) > A3min) &
(Convert.ToInt16(_Params23[0]) < A3max)) { R2 += w0_3; }
                        }
                        if (A7 == 1)
                        {
                            List<string> _Params27 = _dictDataTwo["ТЕХНІЧНІ
НАВИЧКИ"];
                            if (_Params27[0] == "Так") { R2 += w0_7; }
                        }
                        if (A10 == 1)
                        {
                            List<string> _Params210 =
_dictDataTwo["СТРЕСОСТІЙКІСТЬ"];
                            if (_Params210[0] == "Так") { R2 += w0_10; R2
+= w9_10; }
                        }
                        if (A11 == 1)
                        {
                            List<string> _Params211 =
_dictDataTwo["ІНІЦІАТИВНІСТЬ"];
                            if (_Params211[0] == "Так") { R2 += w0_11; R2
+= w9_11; }
                        }
                        if (A12 == 1)
                        {
                            List<string> _Params212 =
_dictDataTwo["ГЕНЕРАЦІЯ ІДЕЙ"];
                            if (_Params212[0] == "Так") { R2 += w0_12; R2
+= w9_12; }
                        }

                    }
                }
                else
                {
                    R2 = 0.0F;
                }

                _dictDataTree = InputData.GetTreeSavedData();
                List<string> _Params39 = _dictDataTree["ЯКОСТІ КЕРІВНИКА"];
```

```
                if (_Params39[0] == "Так")
                {
                    List<string> _Params30 = _dictDataTree["ПРІЗВИЩЕ ІМ'Я
ПО-БАТЬКОВІ"];
                    A03 = _Params30[0];
                    if (A1 != "-= вибрати =-")
                    {
                        List<string> _Params31 = _dictDataTree["ТИП
ПРОЕКТУ"];
                        if (A1 == _Params31[0])
                        {
                            if (A5 == 1)
                            {
                                List<string> _Params35 =
_dictDataTree["ВІДПОВІДАЛЬНІСТЬ"];
                                if (_Params35[0] == "Так") { R3 += w1_6; R3
+= w9_6; }
                            }
                            if (A6 == 1)
                            {
                                List<string> _Params36 =
_dictDataTree["ДОТРИМАННЯ ТЕРМІНІВ"];
                                if (_Params36[0] == "Так") { R3 += w1_5; R3
+= w0_5; }
                            }
                            if (A8 == 1)
                            {
                                List<string> _Params38 =
_dictDataTree["РОЗУМІННЯ БІЗНЕС-ПРОЦЕСІВ"];
                                if (_Params38[0] == "Так") { R3 += w1_8; R3
+= w0_8; R3 += w9_8; }
                            }
                            if (A4 > 0)
                            {
                                List<string> _Params34 =
_dictDataTree["ВИКОНАНО ПРОЕКТІВ"];
                                if (Convert.ToInt16(_Params34[0]) > A4) {
R3 += w1_4; R3 += w0_4; }
                            }
                            if (A14 != "- вибрати -")
                            {
                                List<string> _Params314 =
_dictDataTree["МОВА ПРОГРАМУВАННЯ"];
                                if (A14 == _Params314[0]) { R3 += w1_14; R3
+= w0_14; }
                            }
                            if (A15 > 0)
                            {
                                List<string> _Params315 =
_dictDataTree["ШВИДКІСТЬ НАБОРУ"];
                                if (Convert.ToInt16(_Params315[0]) > A15) {
R3 += w1_15; R3 += w0_15; }
                            }

                        }
                        if (A2 > 0)
                        {
                            List<string> _Params32 = _dictDataTree["СТАЖ
РОБОТИ"];
                            if (Convert.ToInt16(_Params32[0]) > A12) { R3
+= w0_2; }
                        }
                        if (A3min > 0)
                        {
```

```
                            List<string> _Params33 = _dictDataTree["ВІК"];
                            if ((Convert.ToInt16(_Params33[0]) > A3min) &
(Convert.ToInt16(_Params33[0]) < A3max)) { R3 += w0_3; }
                        }
                        if (A7 == 1)
                        {
                            List<string> _Params37 =
_dictDataTree["ТЕХНІЧНІ НАВИЧКИ"];
                            if (_Params37[0] == "Так") { R3 += w0_7; }
                        }
                        if (A10 == 1)
                        {
                            List<string> _Params310 =
_dictDataTree["СТРЕСОСТІЙКІСТЬ"];
                            if (_Params310[0] == "Так") { R3 += w0_10; R3
+= w9_10; }
                        }
                        if (A11 == 1)
                        {
                            List<string> _Params311 =
_dictDataTree["ІНІЦІАТИВНІСТЬ"];
                            if (_Params311[0] == "Так") { R3 += w0_11; R3
+= w9_11; }
                        }
                        if (A12 == 1)
                        {
                            List<string> _Params312 =
_dictDataTree["ГЕНЕРАЦІЯ ІДЕЙ"];
                            if (_Params312[0] == "Так") { R3 += w0_12; R3
+= w9_12; }
                        }

                    }
                }
                else
                {
                    R3 = 0.0F;
                }

                _dictDataFour = InputData.GetFourSavedData();
                List<string> _Params49 = _dictDataFour["ЯКОСТІ КЕРІВНИКА"];
                if (_Params49[0] == "Так")
                {
                    List<string> _Params40 = _dictDataFour["ПРІЗВИЩЕ ІМ'Я
ПО-БАТЬКОВІ"];
                    A04 = _Params40[0];
                    if (A1 != "-= вибрати =-")
                    {
                        List<string> _Params41 = _dictDataFour["ТИП
ПРОЕКТУ"];
                        if (A1 == _Params41[0])
                        {
                            if (A5 == 1)
                            {
                                List<string> _Params45 =
_dictDataFour["ВІДПОВІДАЛЬНІСТЬ"];
                                if (_Params45[0] == "Так") { R4 += w1_6; R4
+= w9_6; }
                            }
                            if (A6 == 1)
                            {
                                List<string> _Params46 =
_dictDataFour["ДОТРИМАННЯ ТЕРМІНІВ"];
```

```
                                if (_Params46[0] == "Так") { R4 += w1_5; R4
+= w0_5; }
                            }
                            if (A8 == 1)
                            {
                                List<string> _Params48 =
_dictDataFour["РОЗУМІННЯ БІЗНЕС-ПРОЦЕСІВ"];
                                if (_Params48[0] == "Так") { R4 += w1_8; R4
+= w0_8; R4 += w9_8; }
                            }
                            if (A4 > 0)
                            {
                                List<string> _Params44 =
_dictDataFour["ВИКОНАНО ПРОЕКТІВ"];
                                if (Convert.ToInt16(_Params44[0]) > A4) {
R4 += w1_4; R4 += w0_4; }
                            }
                            if (A14 != "- вибрати -")
                            {
                                List<string> _Params414 =
_dictDataFour["МОВА ПРОГРАМУВАННЯ"];
                                if (A14 == _Params414[0]) { R4 += w1_14; R4
+= w0_14; }
                            }
                            if (A15 > 0)
                            {
                                List<string> _Params415 =
_dictDataFour["ШВИДКІСТЬ НАБОРУ"];
                                if (Convert.ToInt16(_Params415[0]) > A15) {
R4 += w1_15; R4 += w0_15; }
                            }

                        }
                        if (A2 > 0)
                        {
                            List<string> _Params42 = _dictDataFour["СТАЖ
РОБОТИ"];
                            if (Convert.ToInt16(_Params42[0]) > A12) { R4
+= w0_2; }
                        }
                        if (A3min > 0)
                        {
                            List<string> _Params43 = _dictDataFour["ВІК"];
                            if ((Convert.ToInt16(_Params43[0]) > A3min) &
(Convert.ToInt16(_Params43[0]) < A3max)) { R4 += w0_3; }
                        }
                        if (A7 == 1)
                        {
                            List<string> _Params47 =
_dictDataFour["ТЕХНІЧНІ НАВИЧКИ"];
                            if (_Params47[0] == "Так") { R4 += w0_7; }
                        }
                        if (A10 == 1)
                        {
                            List<string> _Params410 =
_dictDataFour["СТРЕСОСТІЙКІСТЬ"];
                            if (_Params410[0] == "Так") { R4 += w0_10; R4
+= w9_10; }
                        }
                        if (A11 == 1)
                        {
                            List<string> _Params411 =
_dictDataFour["ІНІЦІАТИВНІСТЬ"];
```

```
                            if (_Params411[0] == "Так") { R4 += w0_11; R4
+= w9_11; }
                        }
                        if (A12 == 1)
                        {
                            List<string> _Params412 =
_dictDataFour["ГЕНЕРАЦІЯ ІДЕЙ"];
                            if (_Params412[0] == "Так") { R4 += w0_12; R4
+= w9_12; }                         }
                    }
                }
                else
                {
                    R4 = 0.0F;
                }
                if ((R1 > R2) & (R1 > R3) & (R1 > R4))
                {
                    textBox6.Text += System.Environment.NewLine + "Керівник
проекту: " + A01;
                }
                if ((R2 > R1) & (R2 > R3) & (R2 > R4))
                {
                    textBox6.Text += System.Environment.NewLine + "Керівник
проекту: " + A02;
                }
                if ((R3 > R1) & (R3 > R2) & (R3 > R4))
                {
                    textBox6.Text += System.Environment.NewLine + "Керівник
проекту: " + A03;
                }

                if ((R1 == R2) & (R1 == R3) & (R1 == R4))
                {
                    textBox6.Text += System.Environment.NewLine +
"Кандидатур немає";
                }

                textBox6.Text += System.Environment.NewLine + string.Empty;
                textBox6.Text += System.Environment.NewLine + "Оберіть
виконавців";
            }
        private void прийнятиРішенняToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            toolStripButton3_Click(sender,e);
        }
        private void toolStripButton4_Click(object sender, EventArgs e)
        {
            string fname = textBoxName.Text + ".txt";
            TextWriter tw = new StreamWriter(fname);
            tw.WriteLine(textBox6.Text, Encoding.Default);
            tw.Close();
            MessageBox.Show("Збережено " + fname, "Збереження",
MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
        private void зберегтиРішенняToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            toolStripButton4_Click(sender, e);
        }
    }
}
```