Вінницький національний технічний університет
Факультет інформаційних технологій та комп’ютерної інженерії
Кафедра комп’ютерних наук

Пояснювальна записка

до магістерської кваліфікаційної роботи

на тему «Інформаційна технологія розв’язання задачі скорингу на основі нейронної мережі»

Виконала: студентка 2 курсу,
групи 1КН-18м
спеціальності 122 «Комп'ютерні науки»
Семенова Л.М.
Керівник: к.т.н., доц. Колесницький О.К.
Рецензент: к.т.н., доц. Войтко В.В.

Вінниця - 2019 року

ЗАТВЕРДЖУЮ
Завідувач кафедри____КН____
д.т.н., проф.. Яровий А.А.

_____
(підпис)
“____” _____2019 року

ЗАВДАННЯ

на магістерську кваліфікаційну роботу на здобуття кваліфікації магістра зі спеціальності: 122 – «Комп'ютерні науки»

08-22.МКР.020.18.094.ПЗ

Магістранта групи 1КН-18м Семенової Людмили Миколаївни

Тема магістерської кваліфікаційної роботи: «Інформаційна технологія розв’язання задачі скорингу на основі нейронної мережі»

Вхідні дані: кількість параметрів для аналізу – не менше 10, вихідна інформація – рекомендація і скоринговий бал, кількість нейронів – не менше 10, кількість шарів – не менше 3, мова програмування – об’єктно-орієнтована.

Короткий зміст частин магістерської кваліфікаційної роботи:

1. Графічна: Схема алгоритму роботи програми визначення скорингу, структура інформаційної технології розв’язання задачі скорингу на основі нейронної мережі; структура нейронної мережі радіально-базисних функцій, UML-діаграма класів інформаційної технології розв’язання задачі скорингу на основі нейронної мережі, початкове вікно програми розв’язання задачі скорингу, робочі вікна програми розв’язання задачі скорингу.
2. Текстова (пояснювальна записка): вступ, аналіз предметної області розв’язання задачі скорингу, розробка інформаційної технології розв’язання задачі скорингу на основі нейронної мережі, програмна реалізація інформаційної технології розв’язання задачі скорингу на основі нейронної мережі, економічна частина, висновки, перелік використаних джерел, додатки.

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ МКР

№ етапу	Назва етапу	Термін виконання		Очікувані результати
		початок	кінець	
1	Аналіз сучасного рівня розвитку інформаційної технології розв'язання задачі скорингу			Аналітичний огляд літературних джерел, задачі досліджень, розділ 1 ПЗ
2	Розробка методу та інформаційної технології розв'язання задачі скорингу на основі нейронної мережі			Метод, інформаційна технологія, розділ 2
3	Програмна реалізація розробленої інформаційної технології, тестування та оцінка параметрів			Програмне забезпечення, розділ 3
4	Підготовка економічної частини			Розділ 4
5	Апробація та/або впровадження результатів дослідження			Тези доповідей/акт впровадження
6	Оформлення пояснювальної записки, графічного матеріалу та презентації			Пояснювальна записка, графічний матеріал, презентація

Консультанти з окремих розділів магістерської кваліфікаційної роботи

1. Науковий керівник _____ канд. техн. наук, доц., доц. кафедри КН
(підпис) наук. ступінь, вчене звання (посада)

"____" _____20___ р. _____О. К. Колесницький_____
ініціали та прізвище

2. Економічна частина _____ канд. екон. наук, доц. кафедри ЕПВМ
(підпис) наук. ступінь, вчене звання (посада)

_____М. В. Бальзан _____
ініціали та прізвище

"____" _____20___ р.

Дата попереднього захисту роботи "____" _____20___ р.

Рецензент _____ канд. техн. наук, доц., доц. кафедри ПЗ
(підпис) наук. ступінь, вчене звання (посада)

_____ В. В. Войтко_____
ініціали та прізвище

Завдання видав
науковий керівник _____ канд. техн. наук, доц., доц. кафедри КН
(підпис) наук. ступінь, вчене звання (посада)

_____О. К. Колесницький_____
ініціали та прізвище

"____" _____20___ р.

Завдання отримав магістрант _____ _____ Л. М. Семенова_____
(підпис) ініціали та прізвище

"____" _____20___ р.

РЕФЕРАТ

Магістерська кваліфікаційна робота присвячена розробці інформаційної технології розв’язання задачі скорингу на основі нейронної мережі. Технологія забезпечує визначення скорингової оцінки клієнта на основі його анкетування, а також дає рекомендацію – видавати клієнту кредит чи ні. У роботі досліджено предметну область задачі скорингу, методи її реалізації. Було проаналізовано нейронні мережі, що реалізують класифікацію об’єктів та апроксимацію функцій. На основі об’єктивних переваг як нейронну мережу для реалізації технології було обрано мережу радіально-базисних функцій. Розроблено архітектуру проектованої нейронної мережі, удосконалено модель її навчання за рахунок введення процедури підбору початкових значень вагових коефіцієнтів вихідного шару, що дозволило суттєво підвищити швидкість навчання нейронної мережі. Було побудовано структуру інформаційної технлогії та розроблено алгоритм роботи програмного забезпечення на її основі. Як мову програмування для реалізації технології було обрано Java. Технологія орієнтована на банківські, фінансові та інші установи, де виникає потреба визначення скорингової оцінки клієнта.

ABSTRACT

The master's qualification work is dedicated to the development of information technology for solving scoring problem based on the neural network. The technology provides a rating for a customer based on his / her questionnaire, and also advises whether or not to give a customer a loan. The subject area of the scoring problem, methods of its realization are investigated. Neural networks for object classification and function approximation were analyzed. On the basis of objective advantages, a network of radial basis functions was chosen as a neural network for the implementation of technology. The architecture of the designed neural network was developed, and the model of its training was improved by introducing a procedure for selecting the initial values of the weight coefficients of the source layer, which significantly reduced the learning speed of the neural network. The structure of information technology was built and the algorithm of software on its basis was developed. Java was chosen as the programming language for implementing the technology. The technology is focused on banking, financial and other institutions where there is a need to determine the scoring of the client.

ЗМІСТ

ВСТУП

Актуальність теми дослідження. Одним з актуальних напрямків у галузі кібернетики є побудова програмних додатків для класифікації та кластеризації даних. Ці методи широко застосовуються до бізнес-задач для покращення ефективності роботи фірми та при аналізі характеристик клієнтів. Найбільш ефективним методом для такого типу завдань є застосування нейронних мереж.

Відомі бізнес-компанії, банківські установи щодня обслуговують величезну кількість клієнтів. Для покращення якості та швидкості обслуговування, продуктивності праці працівників, і, як наслідок, збільшення прибутку компанії необхідно оптимізувати та автоматизувати процес взаємодії з клієнтом. Крім того, це дозволить зекономити час працівника та збільшить кількість клієнтів, яку він може обслуговувати. Величезна кількість часу та ресурсів витрачається на опитування користувачів. Оптимізацією цього процесу займається скоринг.

Скоринг передбачає аналіз даних про клієнта з метою визначення, чи купить він товар, чи можна давати йому кредит, а також визначення ступеня надійності такого клієнта. Автоматизація цього процесу має місце уже більше 60 років, проте у вітчизняних компаніях вона майже не застосовується. Тому є потреба у розробці програмних засобів, що реалізують процес скорингу і значно підвищать продуктивність праці на підприємствах.

Класична математична модель процесу скорингу є лінійною. Через це вона, з одного боку, дуже проста, а з іншого боку, не враховує багатьох нелінійних залежностей між відповідями на питання потенційних клієнтів та їх кредитоспроможністю. Тому для підвищення точності моделі скорингу в даній роботі пропонується використовувати штучні нейронні мережі. Загальновідомим є той факт, що нейронні мережі є найкращими апроксиматорами багатозначних функцій від кількох змінних. В задачі скорингу набір відповідей на питання є вхідними змінними, а скоринговий прогноз є вихідним параметром. Тобто, результат скорингу є функцією від

багатьох змінних. А апроксимація за допомогою нейронних мереж дозволить підвищити точність самого скорингу.

Отже, задача визначення скорингу носить важливий прикладний характер, і для її розв’язання відсутні достатньо ефективні класичні методи, що спричинює можливість застосування засобів штучного інтелекту для досягнення кращих показників достовірності визначення скорингу.

Тема магістерської кваліфікаційної роботи є актуальною, оскільки завжди існує потреба у точному визначенні скорингу для ефективнішої роботи з клієнтами, а використання при цьому нейронної мережі дозволить врахувати нелінійні залежності між відповідями клієнтів та їх кредитоспроможністю. Дана технологія буде корисною для державних та приватних банківських та маркетингових систем, особливо в умовах її незначної вартості придбання, розробки та обслуговування.

Використання методів штучного інтелекту для розв'язання поставленої задачі є актуальним через те, що надає великі можливості для модифікацій і налаштування різних методів під дану задачу, а отже дозволяє таким чином досягати кращий у порівнянні з традиційними існуючими методами результат.

Таким чином, застосування для розв’язання задачі сучасних методів штучного інтелекту забезпечує подальше впровадження інформаційних технологій у банківську та маркетингову галузь, а також є актуальною темою дослідження нових методів використання штучного інтелекту.

Зв'язок роботи з науковими програмами, планами, темами. Магістерська робота виконана відповідно до напрямку наукових досліджень кафедри комп’ютерних наук Вінницького національного технічного університету 22 К1 «Моделі, методи, технології та пристрої інтелектуальних інформаційних систем управління, економіки, навчання та комунікацій» та плану наукової та навчально-методичної роботи кафедри.

Мета та завдання дослідження. Метою дослідження магістерської кваліфікаційної роботи є підвищення достовірності розв’язання задачі скорингу програмними засобами за рахунок використання штучних нейронних мереж.

Для досягнення поставленої мети необхідно розв’язати такі наступні завдання:

− провести аналіз проблеми розв’язання задачі скорингу;

− розглянути існуючі методи вирішення задачі скорингу та обрати й обґрунтувати вибір методу, який задовольняє мету даної магістерської кваліфікаційної роботи;

− розробити математичну модель інформаційної системи розв’язання задачі скорингу та удосконалити її згідно з метою роботи;

− сформулювати стадії інформаційної технології та на їх основі розробити структуру та алгоритм роботи програмного засобу;

− виконати програмну реалізацію запропонованої інформаційної технології розв’язання задачі скорингу;

− провести тестування програмного продукту та виконати аналіз отриманих результатів.

Об’єкт дослідження – це процес розв’язання задачі скорингу комп’ютерними засобами.

Предмет дослідження – це інформаційна технологія та програмні засоби розв’язання задачі скорингу з використанням штучних нейронних мереж та достовірність їх роботи.

Методи дослідження. У роботі використані наступні методи наукових досліджень: системного аналізу для аналізу структури інформаційної системи, теорія нейронних мереж для реалізації інформаційної технології розв’язання задачі скорингу, методи математичної статистики для розробки процесу класифікації та обрахунків результатів експериментів над програмним засобом, об’єктно-орієнтованого програмування для автоматизації розрахунків.

Наукова новизна одержаних результатів полягає в наступному:

− знайшла подальшого розвитку інформаційна технологія розв’язання задачі скорингу, яка використовує нейронну мережу радіально-базисних функцій, що дозволяє підвищити достовірність розв’язання задачі скорингу;

− удосконалено модель навчання нейронної мережі РБФ за рахунок введення процедури підбору початкових значень вагових коефіцієнтів вихідного шару, що дозволило суттєво підвищити швидкість навчання нейронної мережі.

Практичне значення одержаних результатів полягає у наступному:

1. Розроблено алгоритм функціонування нейронної мережі радіально-базисних функцій для підвищення достовірності кластеризації об’єктів.

2. Розроблено UML-діаграму для проектування програмного засобу для інформаційної технології розв’язання задачі скорингу.

3. Розроблено програмний засіб для розв’язання задачі скорингу на основі нейронної мережі радіально-базисних функцій.

Розроблені алгоритми можуть бути впроваджені в начальний процес у якості лекції на тему «Нейромережевий метод розв’язання задачі скорингу з використанням нейронної мережі радіально-базисних функцій» дисципліни «Нейромережеві методи обчислювального інтелекту», дослідження методу розв’язання задачі скорингу та роботи нейронної мережі радіально-базисних функцій.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується строгістю постановки задач, коректним застосуванням математичних методів під час доведення наукових положень, строгим виведенням аналітичних співвідношень, порівнянням результатів з відомими, та збіжністю результатів математичного моделювання з результатами, що отримані під час впровадження розроблених програмних засобів.

Особистий внесок магістранта. Усі результати, наведені у магістерській кваліфікаційній роботі, отримані самостійно. У роботах, опублікованих у співавторстві, автору належать такі результати: [1] – інформаційна технологія розв’язання задачі скорингу на основі нейронної мережі.

Апробація результатів роботи. Результати роботи були апробовані на Всеукраїнській науково-практичній інтернет-конференції [1] та плануються до впровадження у виробництво (додаток А).

Публікації. За результатами магістерської кваліфікаційної роботи опубліковано 3 тези довоповідей конференцій [1, 2, 3].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ РОЗВ'ЯЗАННЯ ЗАДАЧІ СКОРИНГУ

1.1 Постановка проблеми

Скоринг – це методика визначення кредитного ризику, яка дозволяє, оцінивши набір ознак, що характеризують клієнта, визначити, чи варто надавати йому кредит, чи продавати товар і який [4].

Суть скорингу полягає в тому, що кожному параметру, що характеризує позичальника, надається реальна оцінка в балах. У спрощеному вигляді скорингова модель представляє собою зважену суму визначених характеристик клієнта. Така методика є знеособленою і може застосовуватись як для фізичних, так і для юридичних осіб. За допомогою скорингу на основі кредитної історії попередніх клієнтів банк може визначити, наскільки велика ймовірність того, що конкретний потенційний позичальник поверне кредит у визначений термін. Якщо ж це бізнес-компанія, скоринг може допомогти працівникам краще аналізувати покупки клієнтів, розбивати останніх на кластери та визначати, які пропозиції та акції їм надавати.

Основні типи скорингу, що застосовуються у банківській практиці [5]:

– application-скоринг - оцінка кредитоспроможності клієнта для отримання кредиту;

– fraud-скоринг - оцінка ймовірності шахрайства потенційного клієнта;

– collection-скоринг - механізм роботи з простроченою заборгованістю.

Application-scoring це найпоширеніший і відомий клієнтам вид скорингу. В його основі лежать первинний збір анкетних даних позичальника, їх обробка комп'ютером і висновок результату: надавати позику чи ні.

Collection-scoring – система скорингу на стадії роботи з неповерненими позиками. Визначає пріоритетні дії співробітників банку для повернення «поганих» кредитів. Фактично програма дозволяє зробити ряд кроків по роботі з неповерненими боргами, наприклад, від первинного попередження до передачі справи колекторському агентству. Вважається, що в процесі такої

обробки близько 40% клієнтів посилаються на забудькуватість і повертають кредит.

Виділяють також behavioral-scoring, «скоринг поведінки» – це оцінка найбільш ймовірних фінансових дій позичальника. Така система дає можливість прогнозувати зміну платоспроможності позичальника, коригувати встановлені для нього ліміти. Основою аналізу можуть служити дії клієнта за певний період, наприклад операції по кредитній карті.

Fraud-scoring являє собою статистичну оцінку ймовірності шахрайських дій з боку потенційного позичальника. Такий скоринг, як правило, використовується спільно з іншими видами дослідження клієнтів. При цьому вважається, що до 10% неповернень по кредитах пов'язані з відвертим шахрайством і цей показник зростає [6].

В основу кредитного скорингу покладено вивчення кредитних історій позичальників, які отримували позики в минулому, з метою їх класифікації та визначення характерних ознак надійних та безнадійних клієнтів щодо погашення кредитної заборгованості. Для бізнес-задач таким джерелом є покупки клієнтів, а також результати їх опитування.

Скоринг у галузі маркетингу дозволяє збільшувати обсяги продажів та покупців за рахунок правильного визначення стратегій продажів [4]. Наприклад, задля збереження постійних клієнтів супермаркет проводить акції, рекламні компанії, розсилку новин тощо. Проте супермаркет не може зробити знижки одразу на всі товари, аби задовільнити усіх покупців відразу. А розсилка новин може бути цікава не кожному покупцю, потрібно враховувати їх інтереси та потреби в покупках. Саме тому потрібно розділити усіх покупців на окремі групи, щоб врахувати їх інтереси та запропонувати саме ті новини та знижки, які їм будуть цікаві та зможуть більше привернути їхню увагу.

Уперше скоринг-система для оцінки кредитного ризику позичальника була застосована Д. Дюраном у 1941 р. У ній враховувалися такі

характеристики клієнта: вік, стать, строк проживання в даній місцевості, професія, трудовий стаж, наявність банківських рахунків, володіння нерухомістю, наявність полісу страхування життя [6]. У сучасній зарубіжній банківській практиці при побудові скоринг-систем найчастіше враховуються такі характеристики клієнта: кількість дітей, сімейний стан, дохід, наявність телефону, строк співробітництва з банком. В останні роки скоринг-системи набули поширення і в діяльності вітчизняних банків.

При здійсненні оцінки фінансового стану позичальника фізичної особи враховуються:

• соціальна стабільність клієнта, тобто наявність власної нерухомості, цінних паперів тощо, робота, сімейний стан і, як наслідок:

- наявність реальної застави;
- вік та здоров'я клієнта;
- загальний матеріальний стан клієнта, його прибутки та витрати.

• користування банківськими позиками в минулому та своєчасність погашення їх та відсотків за ними, а також користування іншими банківськими послугами;

• зв'язки клієнта з діловим світом тощо.

На Заході банки вже давно не вважають заробітну плату основним чинником кредитоспроможності. Крім доходу на неї впливають стать і сімейний стан (розлучені жінки дисциплінованіше, ніж неодружені чоловіки), а також вік (банки не довіряють дуже молодим). Для розробки доброї скорингової карти передусім необхідно вибрати ряд факторів, що найбільше впливають на поведінку позичальника в майбутньому [7].

Також важливим є визначення критичного значення результату скорингової моделі, що розділить позичальників на «поганих» і «добрих», тобто рівень, при якому доходи від «добрих» позичальників є достатніми для покриття збитків за потенційно «поганими» позичальниками. Для цього можна звернутися до комплексного аналізу і співвідношення доходності

кредитного портфеля і рівня списання боргів, віднесених до безнадійних та інших витрат. Припустимо, що в середньому збитки за одним «поганим» кредитним рахунком покриваються доходом за десятьма «добрими». У даному випадку таким рівнем буде значення скорингової карти, відповідне такому співвідношенню: 10/1. Саме таке значення і буде свого роду точкою беззбитковості кредитних операцій банку. Такий аналіз також повинен виявити ступінь кореляції і важливості даних з якістю кредитного рахунку і внаслідок цього їх важливість у скоринговій моделі.

Успішність скорингової моделі пояснюють деякі ключові фактори [7]:

- неупередженість оцінки (скоринг відмежовує суб'єктивність оцінок, традиційно пов'язану з кредитними рішеннями);

- стандартизація кредитних оцінок;

- можливість автоматизації;

- контроль (завдяки стандартизації кредитних операцій банкам не важко контролювати і відстежувати ефективність кредитних рішень);

- зростання доходності (автоматизація знижує витрати на ручну обробку заявок на кредит до мінімуму).

Результатом реалізації методики скорингу в кожному окремому випадку є інтегральний показник, який порівнюється з певним числовим порогом, або лінією поділу, яка є лінією беззбитковості. Клієнтам з інтегральним показником вище цієї лінії видається кредит, клієнтам з інтегральним показником нижче лінії беззбитковості – відмовляють у наданні позички [4].

Важливим також є те, що в центрі аналітичної роботи, пов'язаної зі скорингом, знаходиться систематична перевірка ефективності діючої моделі з метою коригування шкали оцінок. Підсумком перевірки може бути рішення про зміщення акцентів з одного оціночного показника на інший, який, на думку аналітиків фірми, є більш вагомим для визначення купівельної спроможності клієнтів. І навпаки: окремі параметри можуть бути знижені в

балах чи виключені з діючої моделі зовсім. Можливо, з’явиться потреба оновити і внутрішню градацію балів за одним чи рядом показників [4].

Скорингові системи дозволяють знизити витрати і мінімізувати операційний ризик за рахунок автоматизації прийняття рішення, скорочують час обробки заявок на надання кредиту, дають можливість банкам проводити свою кредитну політику централізовано, забезпечують додатковий захист фінансових організацій від шахрайства. У той же час скоринг має і ряд недоліків: часто рішення системи засновано на аналізі даних, наданих виключно самим позичальником. Крім того, скорингові системи необхідно постійно допрацьовувати і підтримувати, так як вони враховують тільки минулий досвід і реагують на зміни соціально-економічної ситуації із запізненням [8].

Технології скорингу мають постійні тенденції до розвитку та вдосконалення, що дозволяє розробляти нові алгоритми, які, у свою чергу, дозволяють підвищувати достовірність та швидкодію обчислення скорингу.

Завдання визначення скорингу ставиться таким чином. Є певний набір відповідей клієнта на питання анкети, що належать заздалегідь відомій кінцевій кількості класів $C = \{C_1, ..., C_q\}$. Є деяка кінцева кількість об'єктів (навчальна вибірка), про кожен з яких відомо до якого класу він належить. Потрібно розробити програму, яка для будь-якого вхідного набору відповідей, який не обов'язково має належати навчальній множині, вирішувала би, до якого класу цей об'єкт належить і знаходила його скоринговий бал.

1.2 Аналітичний огляд відомих методів реалізації задачі скорингу та обґрунтування вибору нейромережевого методу

Створення скорингової моделі – досить складна задача. Крім організаційної роботи, компанія повинна мати кваліфікованих спеціалістів в області сучасних технологій обробки і аналізу даних. Технології скорингу мають постійні тенденції до розвитку та вдосконалення, що дозволяє

розробляти нові алгоритми, які, у свою чергу, дозволяють підвищувати ефективність та швидкодію обчислення скорингу.

Для побудови скорингових моделей застосовуються різні класифікаційні методи, зокрема [9]:

- дискримінаційний аналіз;
- класифікаційне дерево (рекурсивне розбиття);
- наївний класифікатор Байєса;
- нейронні мережі;
- генетичний алгоритм;
- метод найближчих сусідів.

Дискримінантний аналіз - один з методів багатовимірного аналізу, метою якого є класифікація об'єктів, тобто віднесення об'єкта до однієї з відомих груп деяким оптимальним способом (наприклад, розбиття сукупності підприємств на кілька однорідних груп за значеннями будь-яких показників виробничо-господарської діяльності).

Відмінною властивістю дискримінантного аналізу як методу класифікації є те, що досліднику заздалегідь відомі число груп, на які потрібно розбити розглянуту сукупність об'єктів, і їх властивості. Відомо також, що об'єкт свідомо належить до однієї з певних груп (але до якої саме - невідомо).

При дискримінаційному аналізі виникають завдання двох типів:

- опису відмінностей між класами;
- класифікації об'єктів, що не входили до початкової навчальної вибірки.

Для вирішення першого завдання (опису відмінностей між класами) будуються канонічні дискримінантні функції, які дозволяють з максимальною ефективністю розділити класи. Для того щоб виділити p класів, потрібно не більше p - 1 канонічних дискримінантних функцій. Наприклад, для поділу двох класів достатньо однієї функції, для поділу трьох класів - двох функцій і так далі (рисунок 1.1).

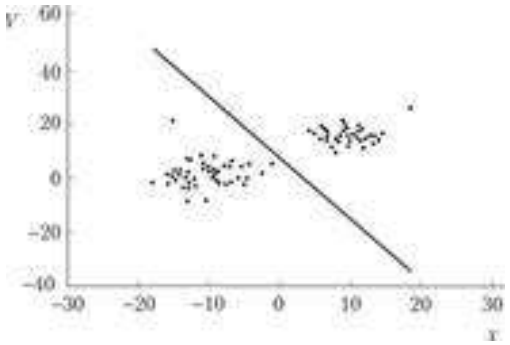


Рисунок 1.1 – Поділ сукупності на два класи за допомогою однієї дискримінантної функції

Канонічні дискримінантні функції можна розглядати як аналог регресійній моделі, побудованої з метою класифікації об'єктів. В цьому випадку дискримінантні змінні є незалежними змінними. Для вимірювання абсолютного і відносного вкладів дискримінантних змінних при поділі класів використовуються нестандартизовані і стандартизовані коефіцієнти канонічних функцій. Чим більше значення коефіцієнта, тим більший внесок в дискримінацію вносить змінна [10].

Найкращим показником інформативності відібраних дискримінантних змінних і корисності застосування дискримінантної функції для інтерпретації міжгрупових відмінностей є відсоток правильно розпізнаних об'єктів з використанням отриманих дискримінантних функцій. Число правильно розпізнаних нових об'єктів (як в цілому, так і по окремим групам) свідчить про відповідність дискримінантної моделі емпіричним даним.

Для вирішення другого завдання (класифікації об'єктів, що не входили до початкової навчальної вибірки) обчислюються відстані від кожного нового об'єкта, що підлягає класифікації, до геометричного центру (центру ваги) кожного класу.

Дискримінантний аналіз висуває суворі вимоги до вихідних даних: в моделі повинно бути не менше двох класів, в кожному класі – не менше двох об'єктів з навчальної вибірки, число дискримінантних змінних не повинно перевищувати обсяг навчальної вибірки, дискримінантні змінні повинні бути

кількісними і лінійно незалежними. Для кожного класу потрібні приблизна рівність коваріаційних матриць, а також багатовимірна нормальність розподілу [10].

Бінарне розбиття простору – це метод рекурсивного розбиття евклідового простору на опуклі множини за допомогою гіперплощин. Це розбиття призводить до подання об'єктів у просторі за допомогою деревоподібної структури даних, відомої як BSP-дерево (рисунок 1.2).

BSP-дерево розроблялось для використання у тривимірній комп'ютерній графіці, оскільки структура BSP-дерева дозволяє виокремити інформацію, щодо об'єктів сцени, яка дозволяє при рендерингу ефективно виконувати такі операції, як сортування візуальних об'єктів в порядку віддалення від спостерігача та виявлення зіткнень. Також BSP-дерево використовується в додатках для виконання операцій над формами (конструктивна блокова геометрія) в САПР, виявлення зіткнень у робототехніці, трасуванні променів та в інших додатках, які виконують обробку складних просторових сцен. Бінарне розбиття простору часто використовується для 3D-відеоігор, зокрема у шутерах від першої особи [11].

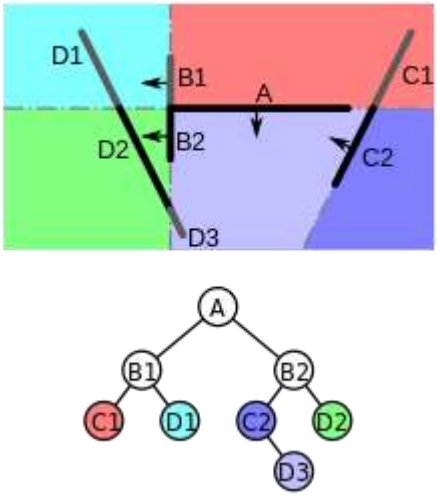


Рисунок 1.2 – Бінарне розбиття простору і відповідне BSP-дерево

Конкретний вибір розбиття площини і критерій для завершення процесу поділу варіюється в залежності від призначення BSP-дерева. У BSP-дереві кожен вузол пов'язаний з прямою, яка розбиває або площиною в 2-вимірному або 3-вимірному просторі відповідно. При цьому всі об'єкти, що лежать з фронтальної сторони площини, відносяться до фронтального піддерева, а всі об'єкти, що лежать зі зворотного боку площини, відносяться до оборотного піддерева. Для визначення належності об'єкта до фронтальної або зворотної сторони прямій або площині, що розбиває, необхідно досліджувати стан кожної його точки.

Якщо для всіх точок об'єкта скалярний добуток більше або дорівнює 0, то він відноситься до фронтального піддерева. Якщо для всіх точок об'єкта скалярний добуток менше або дорівнює 0, то він відноситься до оборотного піддерева. Якщо скалярні добутки для точок об'єкта мають різний знак, то його розтинають площиною, яка розбиває так, щоб отримані об’єкти лежали тільки з фронтальної або тільки зі зворотної сторони. Для кожного підвузла BSP-дерева справедливо вищенаведене твердження, з тим винятком, що для розгляду підлягають тільки ті об'єкти, які належать до фронтальної або зворотної сторони батьківського вузла площини, яка розбиває [11].

Наївний баєсів класифікатор – ймовірнісний класифікатор, що використовує теорему Баєса для визначення ймовірності приналежності спостереження (елемента вибірки) до одного з класів при припущенні (наївному) незалежності змінних (рисунок 1.3).

Якщо на основі значень змінних можна однозначно визначити, до якого класу належить спостереження, байєсів класифікатор повідомить ймовірність приналежності до цього класу. У проміжних же випадках, коли спостереження може з різною ймовірністю належати до різних класів, результатом роботи класифікатора буде вектор, компоненти якого є ймовірностями приналежності до того чи іншого класу.

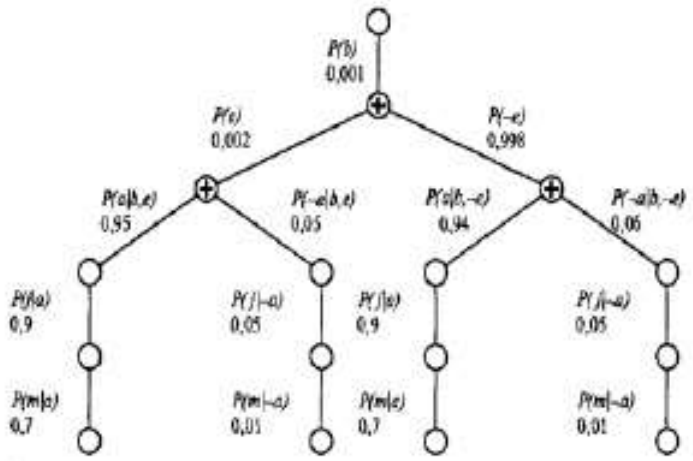


Рисунок 1.3 – Наївний баєсів класифікатор

Ідеальний байєсів класифікатор в якомусь сенсі є оптимальним. Його результат не може бути поліпшений, тому що у всіх випадках, коли можлива однозначна відповідь, він його дасть, а в тих випадках, коли відповідь неоднозначна, результат кількісно характеризує міру цієї неоднозначності.

Разом з тим, в оптимальності криється і основний недолік ідеального байєсового класифікатора: для його побудови потрібна вибірка, що містить всі можливі комбінації змінних — а розмір такої вибірки експоненціально зростає із зростанням числа змінних. Для подолання описаної вище проблеми на практиці використовують наївний байєсів класифікатор — класифікатор, побудований на основі припущення про незалежність змінних, тобто припущення про те, що використання цього припущення дозволяє не вивчати взаємодію всіх можливих поєднань змінних, обмежившись лише впливом кожної змінної окремо на приналежність образу до одного з класів.

Перевагою цього підходу є те, що вимоги до розміру вибірки скорочуються від експоненційних до лінійних. Недоліком є те, що модель є точною лише у випадку, коли виконується припущення про незалежність. В іншому випадку, строго кажучи, обчислені ймовірності вже не є точними (і навіть більше того, їх сума може не дорівнювати одиниці, через що потрібно нормувати результат).

Однак на практиці незначні відхилення від незалежності призводять лише до незначного зниження точності, і навіть у разі істотної залежності між змінними результат роботи класифікатора продовжує корелювати з істинною приналежністю образу до класів. При цьому переваги класифікатора (висока швидкість роботи, простота і масштабованість, помірні вимоги до пам'яті) часто переважують недоліки [12].

Генетичний алгоритм – це еволюційний алгоритм пошуку, що використовується для вирішення задач оптимізації і моделювання шляхом послідовного підбору, комбінування і варіації шуканих параметрів з використанням механізмів, що нагадують біологічну еволюцію (рисунок 1.4).

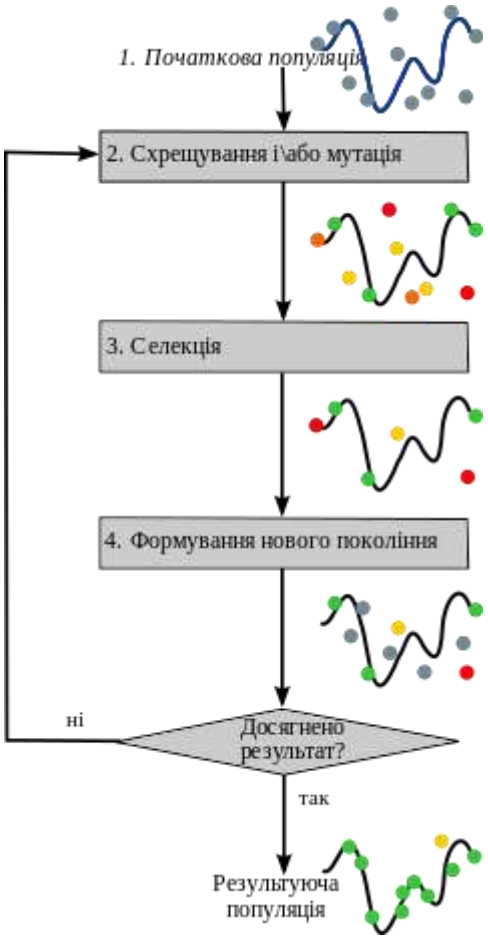


Рисунок 1.4 – Приклад генетичного алгоритму

Особливістю генетичного алгоритму є акцент на використанні оператора «схрещення», який виконує операцію рекомбінації рішень-кандидатів, роль якої аналогічна ролі схрещення в живій природі.

Задача кодується таким чином, щоб її вирішення могло бути представлено в вигляді масиву, подібного до інформації складу хромосоми. Випадковим чином в масиві створюється деяка кількість початкових елементів «осіб», або початкова популяція. Особи оцінюються з використанням функції пристосування, в результаті якої кожній особі присвоюється певне значення пристосованості, яке визначає можливість виживання особи. Після цього з використанням отриманих значень пристосованості вибираються особи, допущені до схрещення (селекція). До осіб застосовується «генетичні оператори» (в більшості випадків це оператор схрещення (crossover) і оператор мутації (mutation)), створюючи таким чином наступне покоління осіб.

Особи наступного покоління також оцінюються застосуванням генетичних операторів і виконується селекція і мутація. Так моделюється еволюційний процес, що продовжується декілька життєвих циклів (поколінь), поки не буде виконано критерій зупинки алгоритму. Таким критерієм може бути:

- знаходження глобального, або надоптимального вирішення;
- вичерпання числа поколінь, що відпущені на еволюцію;
- вичерпання часу, відпущеного на еволюцію.

Генетичні алгоритми можуть бути використані для пошуку рішень в дуже великих і важких просторах пошуку [13].

Метод k-найближчих сусідів – простий непараметричний класифікаційний метод, де для класифікації об'єктів у рамках простору властивостей використовуються відстані (зазвичай евклідові), пораховані до усіх інших об'єктів. Вибираються об'єкти, до яких відстань найменша, і вони виділяються в окремий клас (рисунок 1.5).

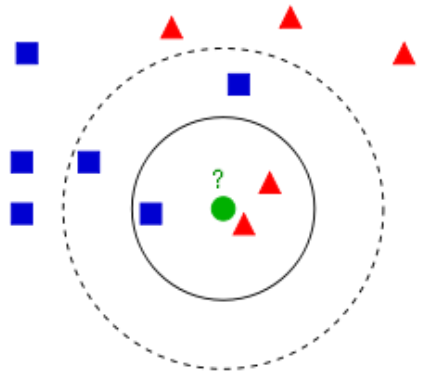


Рисунок 1.5 – Метод k-найближчих сусідів

Метод k-найближчих сусідів являє собою метричний алгоритм для автоматичної класифікації об'єктів. Основним принципом методу найближчих сусідів є те, що об'єкт присвоюється тому класу, який є найбільш поширеним серед сусідів даного елемента. Сусіди беруться, виходячи з множини об'єктів, класи яких уже відомі, і, виходячи з ключового для даного методу значення k, вираховується, який клас є найчисленнішим серед них. Кожен об'єкт має кінцеву кількість атрибутів (розмірностей). Передбачається, що існує певний набір об'єктів з уже наявною класифікацією [14]. Ефективність застосування різних методів розв’язання задачі скорингу представлена на рисунку 1.6 [15].

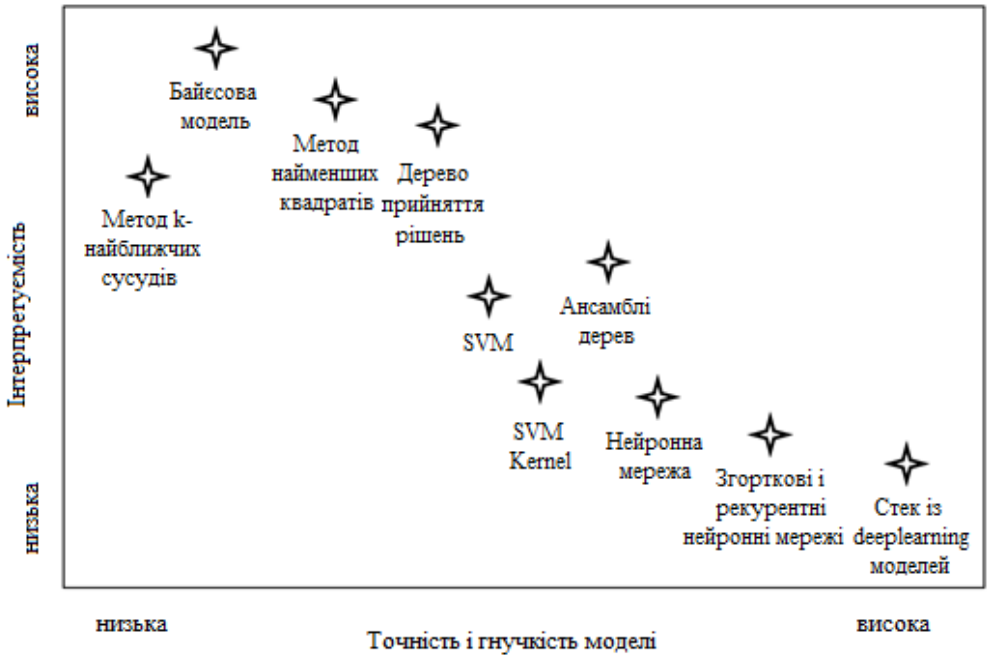


Рисунок 1.6 – Ефективність методів розв’язання скорингу

Класичним лінійним методом визначення скорингу є проведення тестування та підрахунок балів за його результатами. На основі отриманого балу можна визначити, наскільки купівельноспроможним є клієнт. Головним недоліком цього методу є низька швидкодія і ризики виникнення похибок при обчисленнях.

На основі проведеного аналізу було виділено основні переваги і недоліки різних методів, що застосовуються для розв’язання поставленої задачі (таблиця 1.1).

Таблиця 1.1 – Порівняння основних методів розв’язання задачі скорингу

	Переваги	Недоліки
Дискримінаційний аналіз	- використання декількох незалежних змінних для побудови цілісної моделі класифікації об'єктів; - дуже чітке розмежування груп за рахунок врахування одразу багатьох параметрів.	- високий рівень чутливості до розподілення вихідних даних.
Класифікаційне дерево (рекурсивне розбиття)	- наочність; - зрозумілість; - швидкість роботи.	- низька швидкість роботи при великій кількості вхідних параметрів; - неможливість знаходити найбільш повні і точні правила в даних;

Продовження таблиці 1.1

Наївний класифікатор Байєса	- висока швидкість роботи; - простота; - масштабованість; - помірні вимоги до пам'яті.	- для побудови потрібна вибірка, що містить всі можливі комбінації змінних; - модель є точною лише у випадку, коли виконується припущення про незалежність.
Нейронні мережі	- швидкість; - точність; - можливість виявляти нелінійні зв’язки між даними.	- необхідність мати велику навчальну вибірку; - «чорний ящик».
Генетичний алгоритм	- здатність до легкого розпаралелювання.	- використання евристичних алгоритмів, що не завжди дає точний результат.
Метод найближчих сусідів	- стійкість до аномальних викидів; - інтерпретуємість.	- набір даних має бути репрезентативним; - невідокремленість від даних.

Серед усіх відомих методів реалізації скорингової задачі для дослідження було обрано нейронну мережу, так як вона має досить високий рівень інтерпритуємості та точності моделі. Це підвищить швидкодію процесу скорингу, підвищить точність обрахунків. Після правильного навчання нейромережі вона зможе відразу давати відповідь на основі аналізу вхідних

параметрів, що враховує різні особливості, що не могли бути враховані при обрахунку скорингу іншими методами.

Крім того, модель з використанням нейронних мереж є нелінійною, на відміну від більшості інших методів реалізації скорингу. Лінійна модель не може врахувати всі особливості досліджуваної предметної області, а також може неточно ділити на класи такі області, де лінійний поділ неможливий. Нейронні мережі враховують усі ці особливості, що дозволяє розширити множину об’єктів, які модель може з набагато вищою точністю поділити на класи.

Саме швидкість обрахунків, нелінійність моделі і, внаслідок чого, висока точність класифікації зумовила вибір нейронних мереж для реалізації проектованого програмного забезпечення.

1.3 Аналітичний огляд програм-аналогів

Сьогодні існує досить багато сервісів, які автоматизовують процес скорингу. У сфері маркетингу це такі сервіси, як Dripmail, Битрикс24.CRM, AmoCRM, GetResponse, TimeDigital CRM. У сфері кредитного скорингу добре відомі такі програми, як SAS Credit Scoring, EGAR Scoring, Plug & Score Modeler, Clementine (SPSS).

Більшість з цих програм є платними та пропонують широкий спектр послуг, зокрема управління основними процесами банку, візуалізацію даних, побудову скорингових карт тощо. Усі вони використовують різні методи, як статичні, так і методи Data Mining та штучного інтелекту.

SAS Credit Scoring – система оцінки кредитних ризиків, кредитоспроможності клієнтів. Вона використовує чисельні та статистичні методи анализу факторів. Система аналізує дані потенційного позичальника і видає готові рекомендації з видачі кредиту з урахуванням можливих ризиків.

Система оцінки кредитних ризиків включає набір методик та інструментів, які дозволяють передбачити поведінкову модель клієнта,

визначити ймовірність невиплати кредиту, виходу позичальника в дефолт [16]. Робоче вікно програми зображено на рисунку 1.7.

Рисунок 1.7 – Робоче вікно програми SAS Credit Scoring

SAS Credit Scoring надає готові рекомендації по кредитуванню клієнтів, засновані на ретельному аналізі даних позичальників. Система істотно спрощує роботу кредитних менеджерів, скорочує час обробки даних і мінімізує втрати по кредитах. SAS Credit Scoring дозволяє проводити оцінку кредитоспроможності і поведінки практично для всіх кредитних продуктів, включаючи комерційні кредити, кредитні карти, розстрочку та іпотечні кредити.

Система оцінки кредитних ризиків забезпечує збір, перетворення, стандартизацію та очистку всіх релевантних даних для створення повного профілю клієнта з використанням внутрішніх даних компанії, а також з можливістю залучення зовнішньої інформації (наприклад, дані податкових служб) [17].

Система надає наступні можливості:

- Оцінка ймовірності втрати клієнтів в дефолт.
- Прогнозування кредитного ліміту для кожного клієнта.
- Розробка рекомендацій та превентивних заходів для попередження випадків заборгованості.
- Аналіз і виявлення складних особливостей і закономірностей в поведінці клієнтів.

EGAR Scoring – програмний продукт компанії EGAR Technology. EGAR Technology пропонує високотехнологічне рішення EGAR E4 Banking (фізичні особи) по наскрізний автоматизації операцій банку в області кредитування фізичних осіб і індивідуальних підприємців.

Функціонально система повністю покриває операції фронт і бек-офісів кредитної організації, включаючи модулі з обробки кредитної заявки, скорингу, обліку кредитних договорів, управління резервом, управлінського обліку для продуктів, резервів і груп ризиків.

Системою формуються чіткі критерії визначення кредитоспроможності позичальників з використанням сучасних скорингових і макроекономічних підходів. До числа підтримуваних типів продуктів відносяться: цільовий кредит, кредит на невідкладні потреби, кредити під заставу майна, що купується (авто- та іпотека) і поновлювані кредитні лінії (кредитні картки, овердрафт).

Технологічна платформа EGAR E4 Banking реалізована в стандартах J2EE с використанням переваг SOA-архітектури. Рішення побудоване на концепції тонкого клієнта і централізованого сховища даних і володіє широкими інтеграційними можливостями. Масштабування системи досягається простим нарощуванням централізованих серверних ресурсів без зміни складу ПО в точках розгортання за мінімальної участі IT-фахівців [18].

Plug & Score Modeler – програмне забезпечення фірми Plug&Score. Дозволяє створювати скорингові карти клієнтів, порівнювати їх, візуалізувати тощо. Робоче вікно програми зображено на рисунку 1.8.

Основні можливості даної програми [19]:

- підготовка і аналіз даних;

- автоматичне створення скорингових карт;

- моніторинг і перевірка оціночних карт з використанням набору попередньо визначених звітів;

- висновок по відмові за рахунок використання автоматичних і ручних методів виведення;

- автоматичне збереження;

- оцінка коефіцієнта кореляції для кожної пари змінних набору даних;

- робота з декількома перевірочними наборами даних.

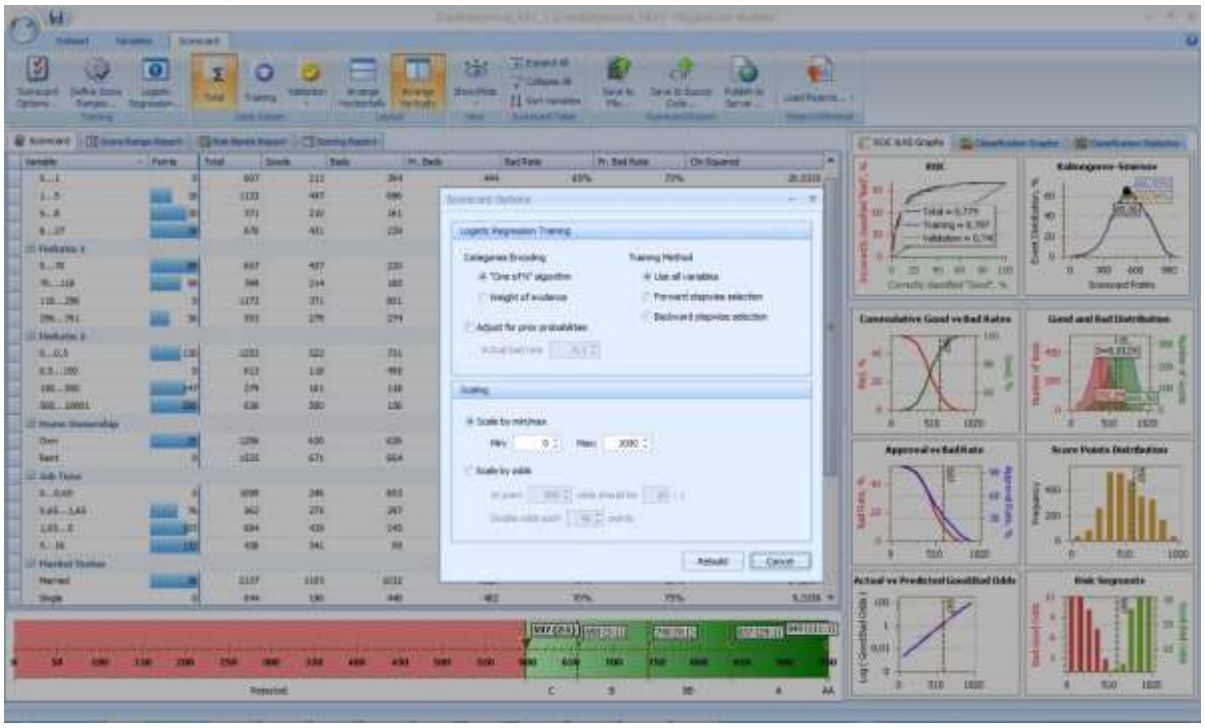


Рисунок 1.8 – Робоче вікно програми Plug & Score Modeler

Clementine (IBM SPSS Modeler) – програмний продукт, який використовує сучасні аналітичні інструменти Data Mining поряд з унікальним

візуальним інтерфейсом, що дозволяє швидко виявляти взаємозв'язки і тренди в даних.

Система дозволяє виявляти неочевидні закономірності в даних, будувати надійні моделі і оперативно впроваджувати отримані результати в процеси прийняття рішень [20]. Робоче вікно програми зображено на рисунку 1.9.

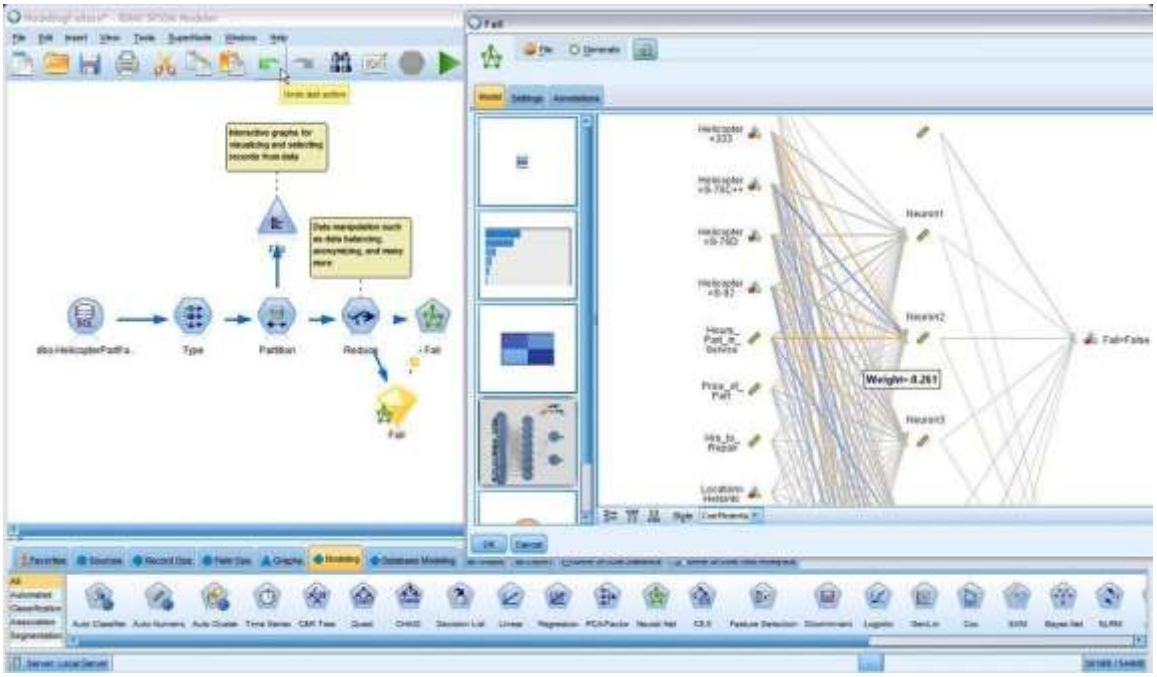


Рисунок 1.9 – Робоче вікно програми Clementine (IBM SPSS Modeler)

Завдання, які вирішуються за допомогою IBM SPSS Modeler:

- підвищення ефективності політики утримання клієнтів;
- стимуляція крос-продажів і повторних покупок;
- сегментація клієнтів;
- мінімізація кредитних ризиків;
- виявлення та запобігання шахрайства.

IBM SPSS Modeler використовують організації з глибоким розумінням і активним використанням даних про поведінку клієнтів і про свою діяльність. Користувачами IBM SPSS Modeler є організації фінансової, страхової,

телекомунікаційної галузей, підприємства роздрібного бізнесу та інших сфер діяльності.

Процес моделювання полягає у виявленні в даних стійких закономірностей, які можуть бути використані для прийняття рішень і управління взаємовідносинами з клієнтами. У IBM SPSS Modeler реалізована трирівнева архітектура обробки даних. Перший рівень складають завдання, які не потребують великих обсягів обчислень і доступу до великих масивів даних, і можуть виконуватися в IBM SPSS Modeler на локальних робочих станціях.

В міру ускладнення задач і збільшення часу очікування результатів стає за доцільне перенесення обчислень на сервер, де користувачі отримують значну перевагу в швидкості обчислень за рахунок застосування більш потужної апаратної частини. Це є другий рівень архітектури програми. Оптимальним є розташування сервера IBM SPSS Modeler і сервера бази даних на єдиній апаратній платформі. Клієнтська частина IBM SPSS Modeler при цьому використовується для підготовки стрімів, запуску завдань на сервері, перегляду та аналізу отриманих результатів.

Наступний рівень продуктивності забезпечується організацією основних обчислень безпосередньо в базі даних за рахунок застосування технології SQL Pushback. Це особливість третього рівеня архітектури програми. При цьому швидкість обчислень дозволяє здійснювати аналітичну обробку дуже великих інформаційних масивів [20].

Більшість автоматизованих сервісів, що пропонують обчислення скорингу, не розповсюджують інформацію про методи, які вони використовують у своїх скорингових моделях. Більше того, використання певної моделі визначається самим підприємством чи банком в залежності від того, які цілі йому необхідно досягти та які завдання виконати. Тобто вид скорингової моделі визначається індивідуально, адже кожна з них має свої переваги та може найкраще підходити лише для певного типу задач.

На основі проведеного аналізу відомих систем-аналогів, було зібрано їх основні переваги та недоліки (таблиця 1.2). Усі якісні програми для визначення скорингової оцінки є платними. Деякі з них використовують статистичні методи обробки, які є неефективними та неточними в умовах реальних даних, які можуть мати нелінійні зв’язки. Більшість програм використовують нестатичні методи обробки, зокрема методи Data Mining, серед яких не всі можуть ефективно визначати правила в даних, що може приводити до неточності у прийнятих рішеннях.

Таблиця 1.2 – Порівняння програм-аналогів

	Переваги	Недоліки
SAS Credit Scoring	- мінімізація операційних ризиків; - скорочення часу обробки заявок на надання кредиту; - самонавчання; - додатковий захист від шахрайства.	- платна ліцензія; - використання статистичних методів, які є незавжди ефективними і не виявляють нелінійних зв’язків між вхідними параметрами.
EGAR Scoring	- масштабованість бізнесу; - продуктивність обробки; - мінімальний час для прийняття рішення.	- платна ліцензія; - використання недостатньо ефективних методів обрахунку скорингу.
Plug & Score Modeler	- багатовимірний аналіз; - простота у використанні; - швидкість роботи; - автоматизація багатьох задач.	- платна ліцензія; - дозволяє лише аналізувати та візуалізувати уже наявні дані про користувачів.

Продовження таблиці 1.2

Clementine (SPSS)	- простота доступу до даних, їх обробки і зміни структури; - швидка побудова та оцінка якості моделей; - швидке отримання віддачі від інвестицій за рахунок високої продуктивності, інтегрованості і масштабованості.	- платна ліцензія; - використовувані методи (зокрема, дерева рішень) не можуть знаходити найбільш повні і точні правила в даних.

1.4 Висновок

У даному розділі було розглянуто предметну область задачі скорингу, основні методи для її розв’язання. Було проведено аналіз переваг та недоліків цих методів. На його основі було обрано нейромережевий метод, який має досить високу швидкодію, а також може знаходити нелінійні зв’язки між вхідними даними, що дозволяє отримати найбільш достовірні результати. Також було проведено аналіз існуючих програм-аналогів, які забезпечують визначення скорингу клієнтів, проаналізовано їх переваги і недоліки. В якості програми-аналога для подальшого дослідження було обрано програмний продукт Clementine (IBM SPSS Modeler). Оскільки існуючі сервіси платні і мають ряд недоліків, є сенс покращувати існуючі методи та моделі підрахунку скорингу. У даному дослідженні пропонується вдосконалити достовірність моделі скорингу шляхом використання нейронних мереж.

2 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ РОЗВ’ЯЗАННЯ ЗАДАЧІ СКОРИНГУ НА ОСНОВІ НЕЙРОННОЇ МЕРЕЖІ

2.1 Обґрунтування вибору виду нейронної мережі для інформаційної технології розв’язання задачі скорингу

Серед відомих методів реалізації скорингової задачі для дослідження було обрано нейронну мережу, так як вона має досить високий рівень інтерпритуємості моделі. Крім того, це підвищить швидкодію процесу скорингу, підвищить достовірність обрахунків. Після правильного навчання нейромережі вона зможе відразу давати відповідь на основі аналізу вхідних параметрів, що враховує різні особливості, що не могли бути враховані при обрахунку скорингу іншими методами.

Для вирішення задачі скорингу необхідно обрати нейронну мережу, що вирішує задачу класифікації об’єктів (для визначення кредитоспроможності клієнта) та апроксимації функцій, оскільки результат скорингу є функцією від багатьох змінних. Такими нейронними мережами є:

- Перцептрон;
- Багатошаровий перцептрон;
- РБФ-мережі;
- Імовірнісна нейронна мережа.

Перцептрон – математична або комп'ютерна модель сприйняття інформації мозком (кібернетична модель мозку), запропонована Френком Розенблатом в 1957 році й реалізована у вигляді електронної машини «Марк-1» у 1960 році. Перцептрон став однією з перших моделей нейромереж, а «Марк-1» – першим у світі нейрокомп'ютером. Незважаючи на свою простоту, перцептрон здатен навчатися і розв'язувати досить складні завдання. Основна математична задача, з якою він здатний впоратися – це лінійне розділення довільних нелінійних множин, так зване забезпечення лінійної сепарабельності.

Перцептрон складається з трьох типів елементів, а саме: сигнали, що надходять від давачів, передаються до асоціативних елементів, а відтак до реагуючих. Таким чином, перцептрони дозволяють створити набір «асоціацій» між вхідними стимулами та необхідною реакцією на виході. В біологічному плані це відповідає перетворенню, наприклад, зорової інформації у фізіологічну відповідь рухових нейронів. Відповідно до сучасної термінології, перцептрони можна класифікувати як штучні нейронні мережі:

1. з одним прихованим шаром;
2. з пороговою передавальною функцією;
3. з прямим розповсюдженням сигналу.

Елементарний перцептрон складається з елементів трьох типів: S-елементів, A-елементів та одного R-елементу (рисунок 2.1). S-елементи являють собою шар сенсорів, або рецепторів. У фізичному втіленні вони відповідають, наприклад, світлочутливим клітинам сітківки ока або фоторезисторам матриці камери. Кожен рецептор може перебувати в одному з двох станів: спокою або збудження; і лише в останньому випадку він передає одиничний сигнал до наступний шару, асоціативним елементам [21].

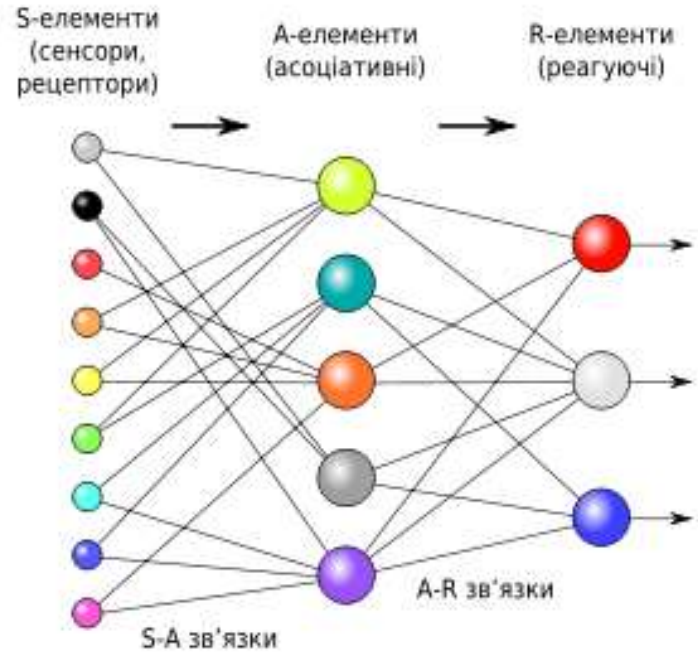


Рисунок 2.1 – Структура перцептрону

A-елементи називаються асоціативними, тому що кожному такому елементові, як правило, відповідає цілий набір (асоціація) S-елементів. A-елемент активізується, щойно кількість сигналів від S-елементів на його вході перевищує певну величину θ.

Сигнали від збуджених A-елементів, своєю чергою, передаються до суматора R, причому сигнал від і-го асоціативного елемента передається з коефіцієнтом. Цей коефіцієнт називається вагою A-R зв'язку.

Так само як і A-елементи, R-елемент підраховує суму значень вхідних сигналів, помножених на ваги (лінійну форму). R-елемент, а разом з ним і елементарний перцептрон, видає «1», якщо лінійна форма перевищує поріг θ, інакше на виході буде «-1». Математично, функцію, що реалізує R-елемент, можна записати так:

$$f(x) = sign\left(\sum_{i=1}^{n} w_i x_i - \theta\right). \quad (2.1)$$

Навчання елементарного перцептрона полягає у зміні вагових коефіцієнтів зв'язків A-R. Ваги зв'язків S-A (які можуть приймати значення (-1; 0; 1)) і значення порогів A-елементів вибираються випадковим чином на самому початку і потім не змінюються.

Після навчання перцептрон готовий працювати в режимі розпізнавання або узагальнення. У цьому режимі перцептрону пред'являються раніше невідомі йому об'єкти, а він повинен встановити, до якого класу вони належать. Робота перцептрона полягає в наступному: при пред'явленні об'єкта, збуджені A-елементи передають сигнал R-елементу, що дорівнює сумі відповідних коефіцієнтів. Якщо ця сума позитивна, то ухвалюється рішення, що даний об'єкт належить до першого класу, а якщо вона негативна – то до другого [21].

Багатошаровий перцептрон Розенблатта – перцептрон з додатковими шарами А-елементів, розташованими між S і R елементами. Перцептрон Розенблатта відрізняється від багатошарового перцептрону Румельхарта, і є загальнішим випадком по відношенню до нього. Оскільки елементарний перцептрон вже володів двома шарами зв'язків та трьома шарами елементів (нейронів), то такий перцептрон не вважався багатошаровим. У багатошаровому перцептроні Розенблатта не обов'язково всі зв'язки можна навчати, частина з них може бути випадково обрана і зафіксована [22]. Загальна структура багатошарового перцептрону зображена на рисунку 2.2.

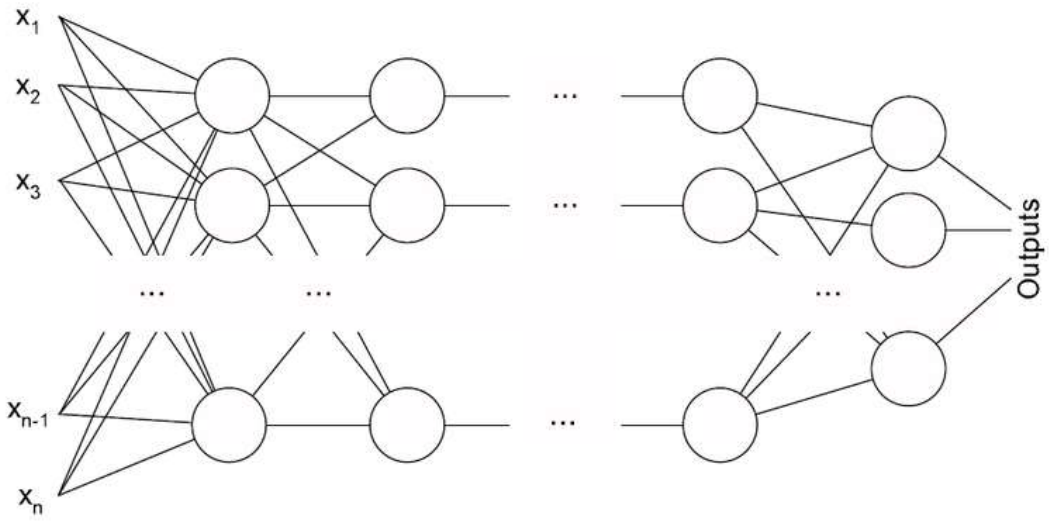


Рисунок 2.2 – Загальна структура багатошарового перцептрону

Багатошаровий перцептрон Румельхарта – окремий випадок перцептрона Розенблатта, в якому один алгоритм зворотного поширення помилки навчає всі шари.

Особливістю є наявність більш ніж одного учня шару (як правило – два чи три, для застосування більшої кількості наразі немає обґрунтування – втрачається швидкість без придбання якості). Необхідність у великій кількості шарів-учнів відпадає, оскільки теоретично єдиного прихованого шару достатньо, щоб перекодувати вхідний сигнал таким чином, щоб отримати лінійну карту для вихідного сигналу. Але є припущення, що, використовуючи

більше число шарів, можна зменшити число елементів у них, тобто сумарне число елементів у шарах буде менше, ніж при використанні одного прихованого шару [23].

Відмінності багатошарового перцептрона від перцептрону Розенблатта:

- використання нелінійної функції активації, як правило сигмоїдної;
- число шарів, які навчають, більше одного, найчастіше використовується не більше трьох;
- сигнали, що надходять на вхід, та одержувані з виходу не бінарні, а можуть кодуватися десятковими числами, які потрібно нормалізувати, так щоб значення були на відрізку була від 0 до 1 (нормалізація необхідна як мінімум для вихідних даних, згідно з функцією активації — сигмоїдою);
- допускається довільна архітектура зв'язків (у тому числі, і повнозв'язані мережі);
- помилка мережі обчислюється не як число неправильних образів після ітерації навчання, а як деяка статистична міра нев'язаності між потрібним і одержаним значенням;
- навчання проводиться не до відсутності помилок після навчання, а до стабілізації вагових коефіцієнтів при навчанні або переривається раніше, щоб уникнути перенавчання.

Багатошаровий перцептрон буде володіти функціональними перевагами в порівнянні з перцептроном Розенблатта лише в тому випадку, якщо у відповідь на стимули не просто буде виконана якась реакція (оскільки вже в перцептроні може бути отримана реакція кожного типу), а виразиться у підвищенні ефективності вироблення таких реакцій. Наприклад, покращиться здатність до узагальнення, тобто до правильних реакцій на стимули, яким перцептрон не навчався. Але зараз таких узагальнюючих теорем немає, існує лише маса досліджень різних стандартизованих тестів, на яких порівнюються різні архітектури [23].

Перцептрон успішно виконує задачу класифікації, проте його реалізація та навчання досить складні, що потребує часу. Те ж саме стосується

багатошарового перцептрону. Він більш ефективний, проте і більш складніший. Крім того, для задачі скорингу передбачається велика кількість входів мережі, що зумовлює велику кількість шарів для перцептрона. Модель перцептрона є лінійною, а для складних областей, які передбачаються задачею скорингу, це вимагає побудови великої кількості лінійних залежностей, аби повністю описати необхідну область. Це вимагає набагато більшої кількості обчислень.

Мережа радіально базисних функцій у математичному моделюванні — це штучна нейронна мережа, яка використовує радіальні базисні функції у якості функції активації. Виходом мережі є лінійна комбінація радіальних базисних функцій входу та параметрів нейрона. РБФ-мережа – це двошарова мережа без зворотних зв'язків, що містить прихований шар радіально-симетричних нейронів (шаблонний шар) [24]. Розташування центрів має відповідати кластерам, що присутні у вхідних даних.

Для того, щоб шаблонний шар був радіально-симетричним, необхідне виконання умов [25]:

- наявність центра, поданого у вигляді вектора у вхідному просторі;

- наявність способу вимірювання відстані між вхідним вектором і центром;

- наявність спеціальної функції одного аргумента, що визначає вихідний сигнал нейрона шляхом відображення функції відстані (використовується функція Гауса).

Мережі радіальних базисних функцій має багато застосувань, зокрема, такі як апроксімацію функції, прогнозування часових рядів, задачі класифікації та керування системою [26].

Мережі радіально базисних функцій (RBF) зазвичай мають три шари (рисунок 2.3): вхідний шар, прихований шар з нелінійною RBF функцією активації та лінійний вихідний рівень. Вхід можна моделювати як вектор дійсних чисел . Вихід мережі є скалярною функцією вхідного вектора.

Враховуючи певні м'які умови за формою функції активації, RBF мережі є універсальними апроксиматорами на компактному просторі. Це означає, що мережа RBF з достатньою кількістю прихованих нейронів може апроксимувати будь-яку неперервну функцію на замкнутому обмеженому наборі з довільною точністю.

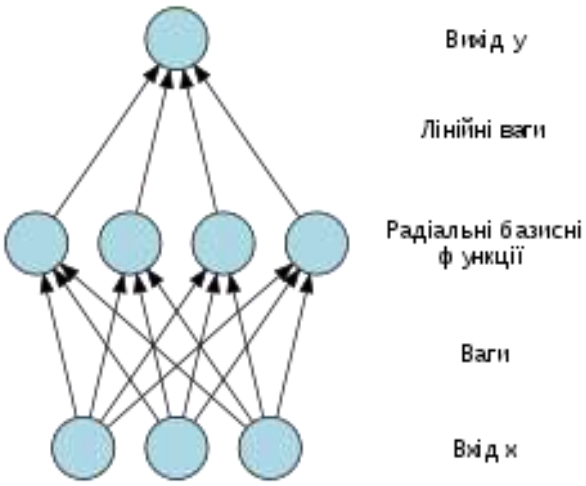


Рисунок 2.3 – Спрощена структура РБФ-мережі

Функції, які залежать лише від відстані від центру вектора, є радіально симетричними щодо цього вектора, і називаються радіально базисною функцією. У базовій формі всі входи пов'язані з кожним прихованим нейроном. Норма, як правило, вважається Евклідовою відстанню (хоча відстань Махаланобіса, загалом, виглядає краще), а радіальна базисна функція зазвичай вважається розподілом Ґауса [27].

Мережі RBF, як правило, тренуються з пар вхідних і цільових значень, за двоетапним алгоритмом. На першому етапі обирається центр вектору RBF функції у прихованому шарі. Цей етап виконується кількома способами; центри можуть бути випадково відібрані з деякого набору прикладів, або їх можна визначити за допомогою кластеризації методом к–середніх. Цей крок не контролюється. Третій етап, метод зворотного поширення помилки, може бути виконаний для точного настроювання всіх параметрів мережі RBF.

Другий крок відповідає лінійнії моделі з коефіцієнтами до виходів прихованого шару з відношенням до деякої цільової функції. Загальна цільова функція, принаймні для регресії/оцінки функції, є функцією найменших квадратів. Тут є явне включення залежності від ваг. Мінімізація цільової функції найменших квадратів за оптимального вибору ваг оптимізує точність підгонки [28].

Є випадки, коли потрібно оптимізувати багато цілей, таких як гладкість, а також точність. У цьому випадку корисно оптимізувати регуляризовану цільову функцію [26].

Ймовірнісна нейронна мережа – вид штучних нейронних мереж, який використовує баєсову статистику для виконання певних завдань. Ймовірнісна нейронна мережа була розроблена Дональдом Спехтом. Загальна структура ймовірнісної нейронної мережі зображена на рисунку 2.4.

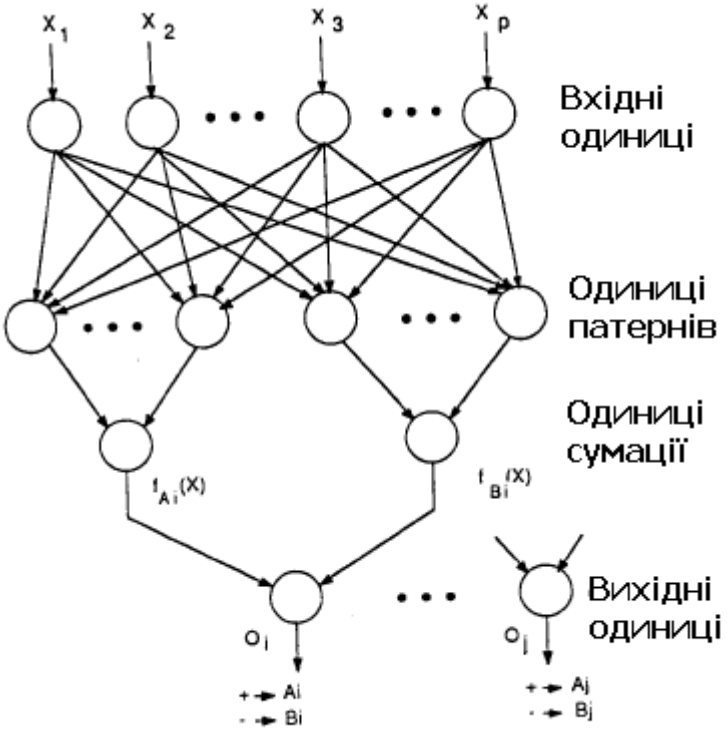


Рисунок 2.4 – Загальна структура ймовірнісної нейронної мережі

Виходи мережі можна інтерпретувати як оцінки ймовірності належності елементу певному класу. Ймовірнісна мережа вчиться оцінювати функцію густини ймовірності, її вихід розглядається як очікуване значення моделі в даній точці простору входів. Це значення пов'язане з густиною ймовірності спільного розподілу вхідних і вихідних даних.

Задача оцінки густини ймовірності відноситься до області баєсової статистики. Звичайна статистика по заданій моделі показує, яка ймовірність того або іншого виходу. Баєсова статистика інтерпретує по-іншому: правильність моделі оцінюється по наявних достовірних даних, тобто надає можливість оцінювати густину ймовірності розподілу параметрів моделі по наявних даних.

При розв'язанні задач класифікації можна оцінити густину ймовірності для кожного класу, порівняти між собою ймовірності приналежності до різних класів і обрати модель з параметрами, при яких густина ймовірності буде найбільшою.

Оцінка густини ймовірності в мережі заснована на ядерних оцінках. Якщо приклад розташований в даній точці простору, тоді в цій точці є певна густина ймовірності. Кластери з близько розташованих точок свідчать, що в цьому місці густина ймовірності велика. Поблизу спостереження є більша довіра до рівня густини, а по мірі віддалення від нього довіра зменшується і прямує до нуля. В методі ядерних оцінок в точку, що відповідає кожному прикладу, поміщається деяка проста функція, потім вони всі додаються і в результаті утворюється оцінка для загальної густини ймовірності. Найчастіше як ядерні функції беруть дзвоноподібні функції (гаусові). Якщо є достатня кількість навчальних прикладів, такий метод дає добрі наближення до істинної густини ймовірності [29].

Ймовірнісна мережа має три шари: вхідний, радіальний та вихідний. Радіальні елементи беруться по одному на кожний приклад. Кожний з них містить гаусову функцію з центром в цьому прикладі. Кожному класу відповідає один вихідний елемент. Вихідний елемент з'єднаний лише з

радіальними елементами, що відносяться до його класу і підсумовує виходи всіх елементів, що належать до його класу. Значення вихідних сигналів утворюються пропорційно ядерним оцінкам ймовірності приналежності відповідним класам.

Навчання ймовірнісної нейронної мережі є набагато простішим, ніж метод зворотного поширення помилки. Істотним недоліком мережі є її розмір, оскільки вона фактично вміщує в собі всі навчальні дані, потребує багато пам'яті і може повільно працювати [29].

РБФ-мережі та імовірнісні мережі схожі за своєю будовою: вони містять лише один прихований шар, ваги якого вираховуються за окремою функцією та нескладні у реалізації та навчанні. РБФ-мережі використовують радіально-базисні функції як функцію активації, а імовірнісні мережі – імовірнісні функції. Проте великим недоліком імовірнісних нейронних мереж є їх об’ємність (вони містять у собі усі навчальні вибірки), а отже і повільність. Вони ефективні лише для пробних тестів, де швидкість та об’єм пам’яті не є важливими.

Вибір нейронної мережі залежить від поставленої задачі. У деяких випадках оптимальніше обрати перцептрон, у інших – імовірнісну нейронну мережу тощо. Для поставленої задачі передбачається велика кількість вхідних параметрів та нелінійність досліджуваних областей. Найкраще для цієї задачі підходить радіально-базисна нейронна мережа. Вона добре апроксимує функції, ефективна для класифікації даних, а також її модель нелінійна. Вона передбачає побудову еліпсів, параметри яких можуть варіюватися, що дозволяє знаходити також феномени у даних вибірки. Якщо аргументів велика кількість, то область рішень будується уже не в площині, а у багатовимірному просторі. Тут РБФ-мережі також мають перевагу, адже, на відміну від перцептрона, будуються нелінійні залежності уже у вигляді еліпсоїдів обертання, що також значно зменшує обсяг та складність обчислень [28]. Тому на основі проведеного аналізу для вирішення даної задачі було обрано РБФ-мережі.

2.2 Розробка архітектури нейронної мережі радіально-базисних функцій для задачі скорингу

В загальному випадку, мережі радіально базисних функцій мають три шари: вхідний шар, прихований шар з нелінійною RBF-функцією активації та лінійний вихідний рівень (рисунок 2.5). Вхідні параметри передаються з вхідного шару на прихований шар, де обробляються фукцією активації з відповідними вагами, а результат передається на вихідний рівень.

Для проектуємої нейронної мережі ці 3 шари будуть оптимальною кількістю. Кількість нейронів на вхідному шарі визначається кількістю вхідних параметрів. Для задачі скорингу це кількість тестових запитань для клієнта. Оскільки їх буде 20, то і кількість нейронів складатиме 20.

На вихідному рівні буде два нейрони, оскільки мережа має розділяти клієнтів на два класи: клас, яким можна видавати кредит, і клас, у яких скорингова оцінка недостатньо висока для видачі кредиту. Виходи нейромережі є цифровими, оскільки вони показують скорингову оцінку клієнта. Діапазон значень складає 0..1. Чим ближче значення до 1, тим вища скорингова оцінка.

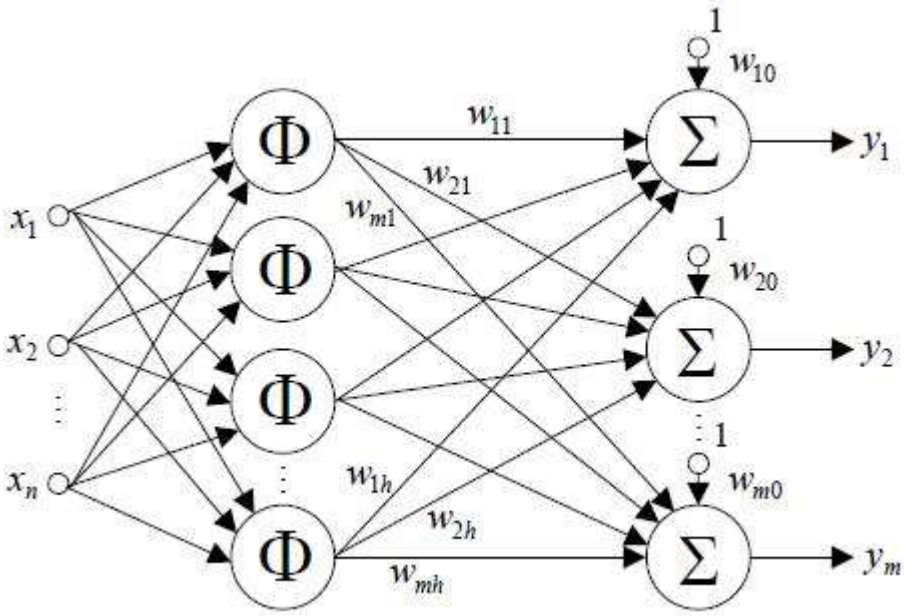


Рисунок 2.5 – Структура радіально-базисної нейронної мережі

У прихованому шарі кількість нейронів обирається на основі практичних міркувань: з одного боку більша кількість нейронів забезпечить кращу точність і роздільну здатність, а з іншого боку, призведе до великого часу навчання і більшої кількості обчислень (тобто низької швидкодії) при функціонуванні мережі. Тому треба знайти оптимальну кількість нейронів, коли вона буде достатня для досягнення потрібної точності і не дуже велика, щоб не витрачати великої кількості ресурсів. Для проектованої мережі оптимальною кількістю буде 50 нейронів у прихованому шарі.

Навчання нейронної мережі буде проводитись за трьома етапами, стандартними для нейронних мереж. Третій етап – метод зворотного поширення помилки.

Вхідними даними для нейронної мережі будуть попередньо підготовлені відповіді клієнта на тестові запитання. Підготовка буде полягати у тому, щоб надати відповідям значення, придатні для подальшої обробки нейронною мережею. Відповіді на запитання можуть бути як текстовими, так і числовими. Тому, якщо відповідь на питання буде «Так-Ні», відповіді буде присвоєно значення 1 або 0 відповідно. Якщо числове значення може бути

дуже великим, порівняно з іншими числовими значеннями, то воно буде нормалізовано, аби у вихідній функції активації воно не мало більшого значення, ніж цього потрібно для скорингової оцінки. Таким чином усі дані будуть підготовлені для подальшої обробки.

Вихідними даними будуть клас, до якого мережа віднесла клієнта, а також його скорингова оцінка. Далі ці результати можуть бути інтерпретовані у потрібному для компанії вигляді і збережені у їхній базі даних. На основі вихідних даних можна формувати подальші рекомандації по роботі з клієнтами.

2.3 Удосконалення математичної моделі нейронної мережі радіально-базисних функцій

Математична модель РБФ-мережі являє собою функцію від багатьох змінних. Мережа дозволяє провести апроксимацію цієї функції для підвищення точності, а отже і достовірності знаходження скорингової оцінки. На основі цієї оцінки проводиться класифікація клієнта за можливістю надання йому кредиту.

Вхід мережі можна моделювати як вектор дійсних чисел $x \in \mathbb{R}^n$ [30]. Вихід мережі є скалярною функцією вхідного вектора $\varphi : \mathbb{R}^n \to \mathbb{R}$, і має вигляд:

$$\varphi(x) = \sum_{i=1}^{N} a_i \rho(\|x - c_i\|), \tag{2.2}$$

де N — кількість нейронів у прихованому шарі, c_i є центральним вектором для нейрона i, та a_i — це вага нейрона i в лінійному виході нейронів. Функції, які залежать лише від відстані від центру вектора, є радіально

симетричними щодо цього вектора, отже, називаються радіальною базисною функцією.

У базовій формі всі входи пов'язані з кожним прихованим нейроном. За норму, як правило, обирається Евклідова відстань. Радіальна базисна функція зазвичай вважається розподілом Ґауса (формула 2.3).

$$\rho(\|x - c_i\|) = \exp[-\beta\|x - c_i\|^2] \tag{2.3}$$

Гаусові базисні функції близькі до центрального вектора в тому сенсі, що:

$$\lim_{\|x\|\to\infty} \rho(\|x - c_i\|) = 0. \tag{2.4}$$

Тобто зміна параметрів одного нейрона має лише невеликий ефект для вхідних значень, що знаходяться далеко від центру цього нейрона.

Завдяки гнучким умовам на форму функції активації, RBF мережі є універсальними апроксиматорами на компактному просторі $\mathbb{R}^n$. Це означає, що мережа RBF з достатньою кількістю прихованих нейронів може апроксимувати будь-яку неперервну функцію на замкненій обмеженій множині з довільною точністю. Параметри a_i, c_i, та β_i визначаються так, щоб оптимізувати відповідність між φ і даними [30].

Крім загальної архітектури, використовують також нормалізовану архітектуру РБФ-мережі. Замість звичайної радіально-базисної функцій активації вона використовує «нормовану радіально-базисну функцію» (формула 2.5).

$$u(\|x - c_i\|) \stackrel{\text{def}}{=} \frac{\rho(\|x - c_i\|)}{\sum_{j=1}^{N} \rho(\|x - c_j\|)} \tag{2.5}$$

Тоді функція активації набуде вигляду:

$$\varphi(x) \stackrel{\text{def}}{=} \frac{\sum_{i=1}^{N} a_i \rho(\|x - c_i\|)}{\sum_{i=1}^{N} \rho(\|x - c_i\|)} = \sum_{i=1}^{N} a_i u(\|x - c_i\|). \qquad (2.6)$$

Для порівняння ненормалізованої та нормалізованої функцій активації було здійснено моделювання відповідних функцій з центрами в $c_1 = 0.75$ та $c_2 = 3.25$ (рисунки 2.6, 2.7) [30].

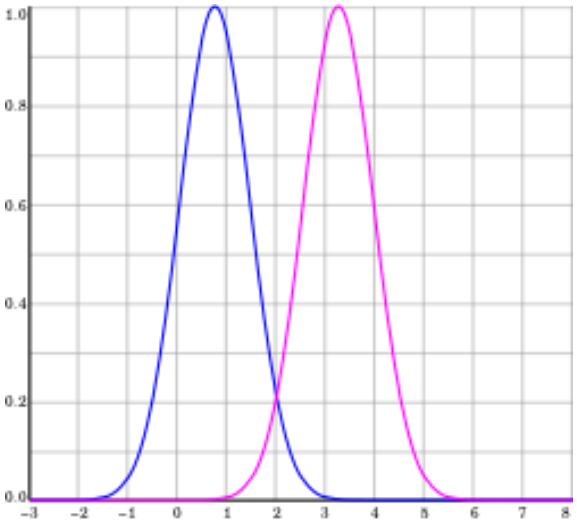


Рисунок 2.6 – Дві ненормовані радіальні базисні функції

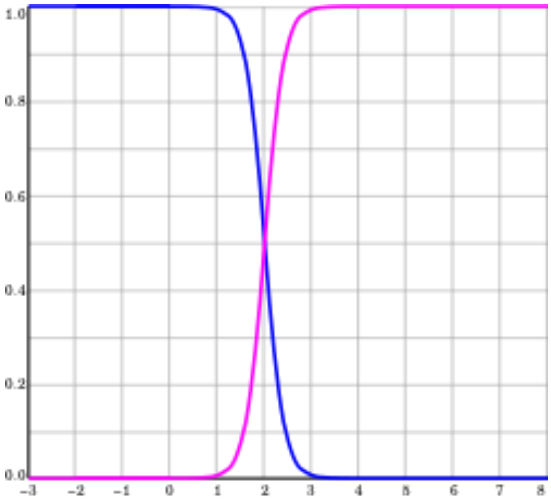


Рисунок 2.7 – Дві нормовані радіальні базисні функції

Іноді архітектуру нейронної мережі розширюють, щоб включити локальні лінійні моделі. У цьому випадку архітектури зводяться до першого порядку, відповідно для ненормалізованої (формула 2.7) та нормалізованої форми (формула 2.8) [30].

$$\varphi(x) = \sum_{i=1}^{N} \big(a_i + b_i(x - c_i)\big)\rho(\|x - c_i\|), \tag{2.7}$$

$$\varphi(x) = \sum_{i=1}^{N} \big(a_i + b_i(x - c_i)\big)u(\|x - c_i\|), \tag{2.8}$$

Тут додатково визначаються ваги b_i. Можливі також вирази більш високого порядку від лінійних термів, які використовують дельта-функцію Кронекера.

Мережі RBF, як правило, тренуються з пар вхідних і цільових значень $x(t), y(t), t = 1, \dots, T,$ за двохетапним алгоритмом. На першому етапі обирається центр вектору c_i RBF функції у прихованому шарі. Цей етап виконується кількома способами: центри можуть бути випадково відібрані з деякого набору прикладів, або їх можна визначити за допомогою кластеризації методом к–середніх. Цей крок не контролюється.

Другий крок просто відповідає лінійній моделі з коефіцієнтами w_i до виходів прихованого шару з відношенням до деякої цільової функції. Загальна цільова функція, принаймні для регресії/оцінки функції, є функцією найменших квадратів [30]:

$$K(w) \stackrel{\text{def}}{=} \sum_{t=1}^{T} K_t(w), \tag{2.9}$$

де

$$K_t(w) \stackrel{\text{def}}{=} [y(t) - \varphi(x(t), w)]^2. \tag{2.10}$$

Третій, необов'язковий етап зворотного поширення помилки, може бути виконаний для точного настроювання всіх параметрів мережі RBF [31].

Загальна математична модель РБФ-мережі передбачає, що мережа сама буде знаходити усі ваги під час навчання на тестовій вибірці. Щоб підібрати оптимальні ваги для поставленої задачі, знадобиться великий обсяг навчальних даних, оскільки на вхід буде надходити велика кількість параметрів.

Крім того, при визначенні скорингової оцінки клієнта деякі параметри можуть мати більш суттєве значення, ніж інші. Наприклад, вік, рівень заробітної плати, наявність кредитів та заборгованостей по ним впливають на скорингову оцінку більше, аніж сімейний стан людини чи якийсь інший параметр. Крім того, рівень важливості кожного з показників може варіюватися в залежності від банку, особливих умов надання кредитів, економічної ситуації в країні тощо.

Тому пропонується удосконалити модель навчання нейронної мережі радіально-базисних функцій за рахунок введення процедури підбору початкових значень вагових коефіцієнтів вихідного шару, що дозволить суттєво підвищити швидкість навчання та достовірність визначення скорингової оцінки. При процесі навчання буде додатково врахований вектор початкових значень вагових коефіцієнтів вихідного шару w_i.

2.4 Підготовка вхідних даних для нейронної мережі визначення скорингу

Для обрахунку скорингового балу необхідно виділити необхідні характеристики, які будуть визначатися шляхом анкетування клієнтів. Відомі наукові розробки, що пропонують різну кількість та різні концепції запитань, у тому числі моделі Альтмана, Фулмера, Чессера, Олсона тощо [32]. Вони

орієнтовані на визначення ризику банкрутства, оцінюють, наскільки надійним може бути клієнт.

В основі формули Альтмана лежить комбінація 4-5 ключових фінансових коефіцієнтів, що характеризують фінансовий стан і результати діяльності підприємства. Спочатку формула була запропонована Альтманом в 60-х роках минулого століття. Пізніше, автор запропонував варіації цієї формули з урахуванням галузевих особливостей організацій.

4-х факторна Z-модель Альтмана використовується для невиробничих підприємств (акції яких не котируються на біржі). Формула чотирьохфакторної моделі виглядає наступним чином [33]:

$$\text{Z оцінка} = 6{,}56T_1 + 3{,}26T_2 + 6{,}72T_3 + 1{,}05T_4, \quad (2.11)$$

де T_1 = Робочий капітал / Активи,

T_2 = Нерозподілений прибуток / Активи,

T_3 = EBIT / Активи,

T_4 = Власний капітал / Зобов'язання.

Модель Чессера дозволяє передбачити можливу фінансову неспроможність потенційного позичальника. Причому модель прогнозує не тільки ризики неповернення кредиту, але і будь-які інші відхилення, що роблять позику менш вигідною для кредитора, ніж було передбачено спочатку. Модель Чессера має вигляд [34]:

$$Y = -2.0434 - 5.24X_1 + 0.0053X_2 - 6.6507X_3 + 4.4009X_4 - 0.0791X_5 - 0.1220X_6, \quad (2.12)$$

де X_1 - (Грошові кошти + Швидко реалізованих цінних паперів) / Сукупні активи,

X_2 - Нетто-продажу / (Грошові кошти + Швидко реалізованих цінних паперів),

X_3 - Брутто-доходи / Сукупні активи,

X_4 - Сукупна заборгованість / Сукупні активи,

X_5 - Основний капітал / Чисті активи,

X_6 - Оборотний капітал / Нетто-продажу.

При цьому має виконуватись умова:

$$Z = \frac{1}{[1 + e^{-Y}]}, \tag{2.13}$$

де e = 2,71828 (число Ейлера - основа натуральних логарифмів).

У разі якщо $Z \geq 0{,}50$, то клієнта слід віднести до групи, яка не виконає умов договору.

В сучасних моделях скорингу і програмних засобах на їх основі враховують набагато більшу кількість параметрів. Такий вид скорингу заснований на анкетуванні клієнтів. При цьому параметрів для аналізу може бути зібрано більше 10.

Для перевірки мети роботи було сформовано запитання для опитування потенційних позичальників на основі відомих анкетувань та відповідних програмних засобів. Було розроблено 20 запитань, які дозволять визначити скоринговий бал клієнта, а також чи можна видавати йому кредит.

Список запитань, які будуть використані для визначення скорингового балу:

1) Вкажіть Ваш вік?
2) Ваша стать?
3) Який Ваш соціальний статус?
4) Який Ваш сімейний стан?
5) Скільки у Вас дітей?
6) Який рівень освіти Ви маєте?

7) Вкажіть Ваш місячний прибуток?
8) В якій сфері діяльності Ви працюєте?
9) Чи є у Вас авто?
10) У Вас власне житло?
11) У Вас є інша нерухомість?
12) Як довго Ви проживаєте у цьому місті?
13) У Вас є поліс страхування житла?
14) Який Ваш стаж на останньому місці роботи?
15) У Вас є заборгованості по штрафам?
16) У Вас є заборгованості по комунальним платежам?
17) У Вас є інші заборгованості?
18) У Вас є банківські рахунки?
19) Який рівень Вашої заробітної плати?
20) Який Ваш загальний трудовий стаж?

Наведені вище питання є питаннями закритого типу з декількома варіантами відповідей. В залежності від обраних варіантів, програма буде обраховувати скоринговий бал. Кожне питання буде мати різну вагу по відношенню до інших, що буде враховано при визначенні початкових ваг нейронної мережі.

2.5 Структура інформаційної технології розв’язання задачі скорингу на основі нейронної мережі

Інформаційна технологія призначена для знаходження скорингової оцінки клієнта на основі відповідей анкетування, та видачі рекомендації щодо надання кредиту клієнтові. Технологія пропонує підвищення достовірності визначення скорингової оцінки за рахунок використання нейронної мережі з удосконаленою математичною моделлю.

Структура інформаційної технології складається з трьох основних частин: вхідні дані, інформаційна технологія, програмне забезпечення інформаційної технології.

Вхідними даними є характеристики клієнта, визначені відповідями на питання анкети. Вихідними даними є скорингова оцінка і можливість видачі кредиту клієнтові.

Проектуєма інформаційна технологія передбачає виконання наступних процесів:

1. Процес перетворення характеристик у формат, придатний для подання на вхід нейронної мережі.
2. Процес формування і навчання нейронної мережі.
3. Процес подачі характеристик на вхід нейронної мережі.
4. Процес обробки вхідних даних нейронною мережею.
5. Процес формування скорингового балу та рекомендації.
6. Процес виведення результату на екран.

Структура інформаційної технології розв’язання задачі скорингу на основі нейронної мережі зображена на рисунку 2.8.

Рисунок 2.8 – Структура інформаційної технології розв'язання задачі скорингу на основі нейронної мережі

2.6 Розробка алгоритму роботи програмного забезпечення розв’язання задачі скорингу на основі нейронної мережі

На основі структури інформаційної технології було розроблено алгоритм її роботи. Вхідні дані спочатку проходять процедуру перетворення, аби при обробці нейронною мережею усі характеристики були враховані відповідно до ступеня їх важливості. Адже дані можуть мати різний формат (текстовий і числовий), а також різний діапазон значень. Тому їх потрібно спершу нормалізувати.

Після цього потрібно сформувати нейронну мережу і провести її навчання. Для цього загальна вибірка ділиться на навчальну і тестову. Чим більша навчальна вибірка, тим точніше будуть визначені вагові коефіцієнти і тим достовірнішими будуть результати.

Сформована і навчена нейронна мережа приймає на вхід перетворені вхідні дані, обробляє їх, класифікує до відповідного класу та видає скорингову оцінку на їх основі. Ці дані виводяться на екран. Далі їх можна інтерпретувати і розробити на їх основі рекомендації.

Весь процес визначення скорингу з навченою нейронною мережею можна поділити на такі частини:

- анкетування;
- підготовка даних до подачі на вхід нейронної мережі;
- визначення скорингу;
- відображення результатів.

Частина анкетування представлятиме собою перелік питань різного типу та перелік відповідей до них. Передбачається також кнопка завершення тестування, після натиснення якої отриманий масив відповідей передасться для визначення скорингу. Крім цього, створимо кнопку для повернення до основного меню. При підготовці даних до подачі на вхід нейронної мережі усі відповіді користувача нормалізуються, текстові дані інтерпретуються у

числові. Це необхідно, щоб усі характеристики були враховані з потрібним рівнем важливості.

Частину визначення скорингу побудуємо на основі нейронної мережі. Ця частина не матиме інтерфейсу з користувачем. Після частини анкетування користувач одразу переходитиме до частини відображення результатів.

При відображенні результатів має відображатись результуючий скоринговий бал клієнта та до якого класу нейронна мережа його віднесла. Також створимо кнопку, яка повертає користувача на початок для можливості проведення наступного анкетування.

Вхідними даними для скорингу є послідовність відповідей користувача на питання анкети. Послідовність має заздалегідь визначену довжину, так як відома кількість питань анкети. В залежності від того, питання якого типу використовуються, відповідь може буде подана текстом, цифровим значенням або булевим значенням, якщо воно передбачало відповідь «Так-Ні». Тому для вихідного вектору використаємо масив, у якому для кожного запитання буде записано числовий відповідник обраного варіанту. Вихідними даними буде результат: скоринговий бал та можливість надання кредиту клієнтові.

Структура процесу визначення скорингу зображена на рисунку 2.9.

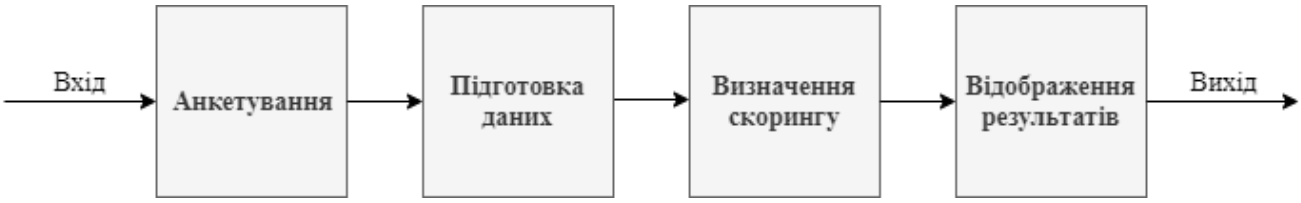


Рисунок 2.9 – Структура процесу визначення скорингу

Алгоритм роботи програми визначення скорингу можна подати у вигляді схеми алгоритму, поданому на рисунку 2.10.

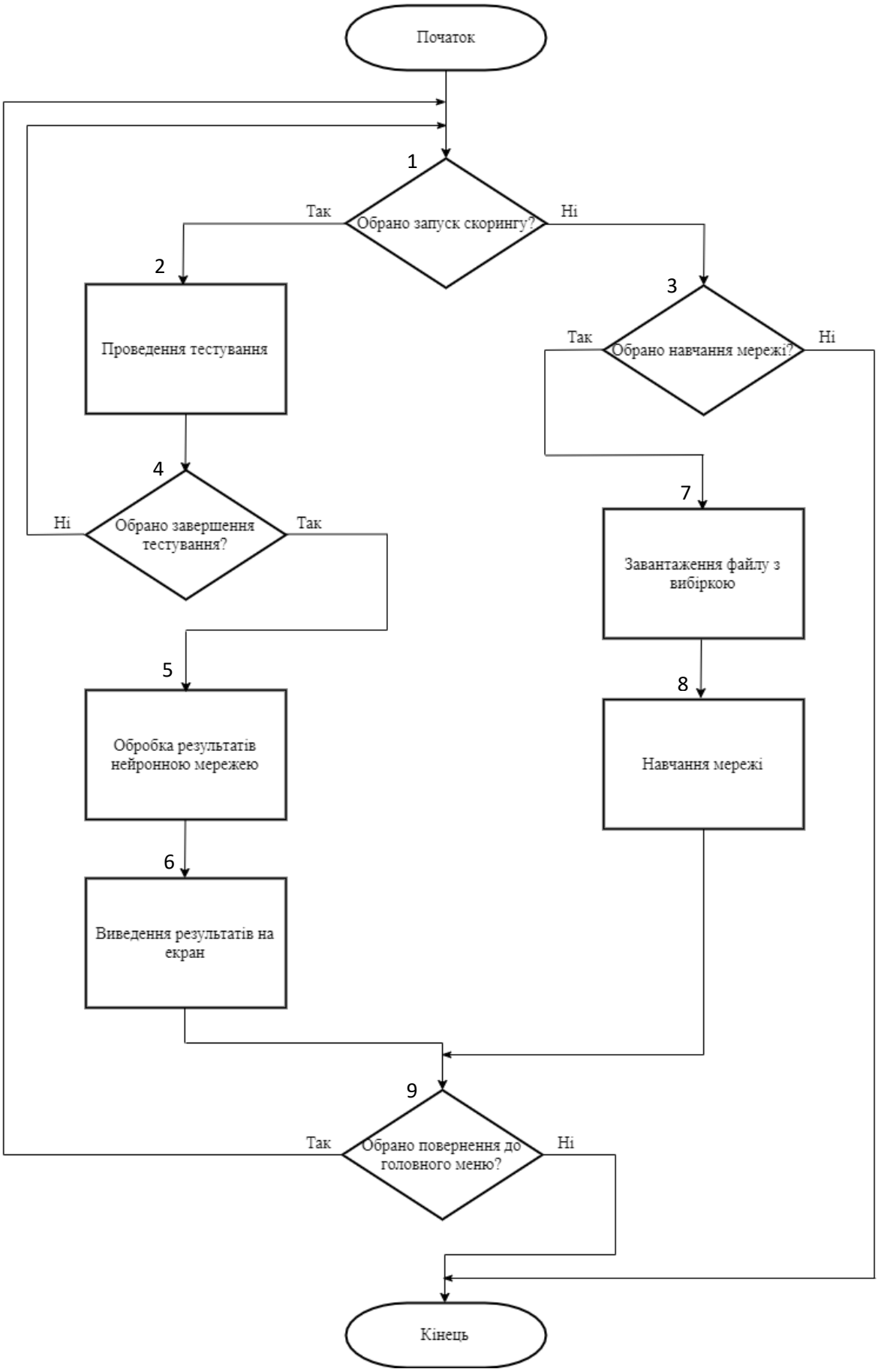


Рисунок 2.10 – Схема алгоритму роботи програми визначення скорингу

Блок 1 зі схеми алгоритму дозволяє перейти до запуску скорингу, проведення навчання мережі або вийти з програми. Він стає активним після першого запуску програми, або після виходу з режиму тестування чи навчання.

Якщо у блоці 1 було обрано запуск скорингу, відбувається перехід до блоку 2. Він відповідає за відображення тестів для користувача та на основі введених відповідей формує вектори результатів проходження тесту, які потім записуються у файл та передаються для визначення скорингу.

Якщо ж у блоці 1 було обрано проведення навчання, то відбувається перехід до блоку 3. Блок 3 визначає, що було обрано користувачем: перехід до навчання мережі чи вихід із програми. Після завершення активності блоку 2 відбувається перехід до блоку 4. Блок 4 визначає, що обрав користувач: завершення тестування чи вихід до головного меню програми.

Якщо під час активності блоку 4 користувач обрав завершення тестування, тоді він переходить до блоку 5, який є невидимим для користувача. Він проводить обробку вхідного вектору відповідей, подаючи на вихід результат процесу скорингу. Одночасно вихід із цього блоку є вхідними даними до блоку 6. Результати обробки вхідного вектору також записуються до файлу.

Програма обробляє дані у наступному порядку:

1) прийняття на вхід попередньо обробленого вектору відповідей користувача;

2) визначення скорингу для конкретного користувача;

3) запис результатів скорингу у файл;

4) передача результатів роботи нейронної мережі до наступного блоку.

Блок 6 схеми алгоритму відповідає за відображення результатів. Тут відображається результат скорингу: скоринговий бал та можливість надання кредиту. Результат скорингу може варіюватися в межах від 0 до 1. Чим

ближче результат до 1, тим більшою є ймовірність того, що клієнт отримає кредит. Після завершення активності блоку 6 відбувається перехід до блоку 9.

Блок 7 передбачає, що для навчання мережі буде завантажено файл з навчальною вибіркою. Після того, як файл було завантажено, активізується блок 8, під час активності якого нейронну мережу буде навчено за даними вибірки. Далі відбувається перехід до блоку 9.

Блок 9 схеми алгоритму відповідає за перехід до головного меню або вихід із програми, в залежності від того, що обере користувач.

Представлений алгоритм відображає роботу програмного забезпечення інформаційної технології. Тут враховано можливі переходи із кожного блоку, в залежності від вибору користувача.

2.7 Розробка UML-діаграми класів

Зв’язки класів відобразимо на UML-діаграмі класів розроблюваної програми (рисунок 2.11). Також відобразимо усі класи та їх ієрархію зв’язків. Разом вони утворюють програму вирішення задачі скорингу на основі нейронної мережі.

Подана діаграма класів відображає статичне представлення структури моделі. Вона відображає декларативні елементи, такі як: класи, їх атрибути, типи даних, їх зміст та відношення.

Найважливішим класом у моделі є клас Unit. Він має найбільшу кількість полів і методів, адже відповідає за побудову нейронної мережі. За роботу нейронної мережі, а також за її навчання та тестування, відповідає клас Net.

Також на діаграмі класів можна побачити, що дані надходять до входів мережі за допомогою класу Input, з використанням класів Link та Neuron.

Головним класом у моделі є клас Main, що разом з класами TestQ1 та Result відповідає за графічний інтерфейс.

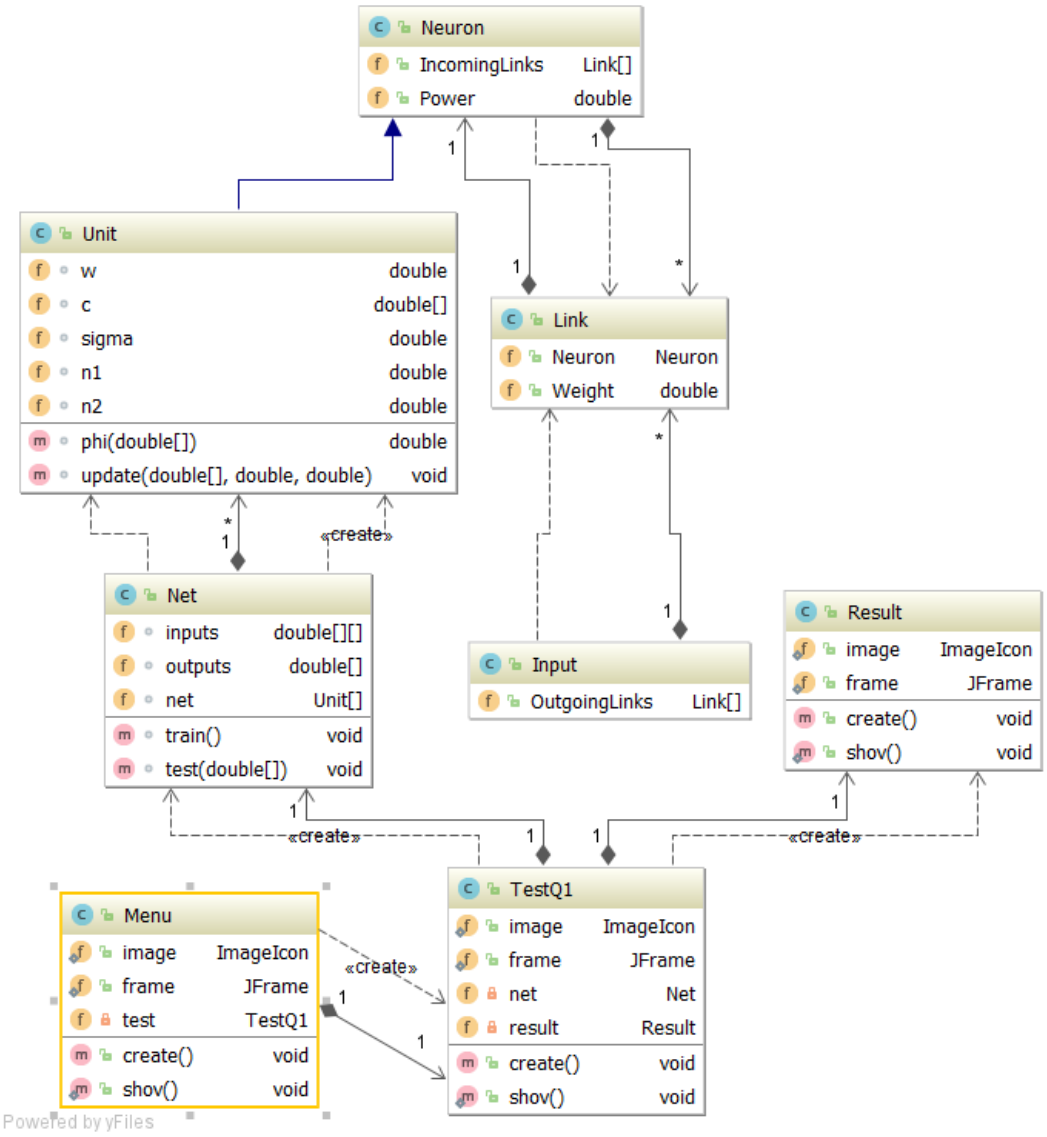


Рисунок 2.11 – UML-діаграма класів програми

2.8 Висновок

У другому розділі було розглянуто основні типи нейронних мереж, які використовуються для апроксимації функцій та класифікації об’єктів, було проведено їх аналіз та порівняння, в результаті чого для реалізації інформаційної технології було обрано мережу радіально-базисних функцій. Було проведено аналіз математичної моделі РБФ-мережі та її навчання і запропоновано удосконалення моделі навчання за рахунок введення процедури підбору початкових значень вагових коефіцієнтів вихідного шару,

що дозволить суттєво підвищити швидкість навчання та достовірність визначення скорингової оцінки. На основі наявних скорингових моделей та програмних засобів було сформовано 20 тестових запитань, які будуть характеризувати клієнта, та описано попередню обробку відповідей перед подачею на вхід нейронної мережі. Було побудовано структуру інформаційної технології розв’язання задачі скорингу та розроблено алгоритм її роботи. На основі алгоритму було розроблено UML-діаграму класів програми, що реалізує інформаційну технологію.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ РОЗВ’ЯЗАННЯ ЗАДАЧІ СКОРИНГУ НА ОСНОВІ НЕЙРОННОЇ МЕРЕЖІ

3.1 Обґрунтування вибору мови та середовища програмування інформаційної технології розв’язання задачі скорингу на основі нейронної мережі

Для реалізації інформаційної технології необхідна мова програмування, яка підтримує об’єктно-орієнтований підхід та придатна для застосування інтелектуальних методів обробки даних. Такими мовами є Java, C++, C# тощо. Для реалізації програми вибір стояв між мовами C++ та Java. На основі аналізу літератури [35] було сформовано порівняльну характеристику цих мов програмування (таблиця 3.1).

Таблиця 3.1 – Порівняння мов програмування

	C++	Java
Статична типізація	+	+
Динамічна типізація	-	-
Багатопотокове компілювання	+	+
Ручне управління пам’яттю	+	-
Покажчики	+	-
Збір сміття	+/-	+
Інтерфейси	+	+
Використання шаблонів	+	+
Перезагрузка функцій	+	+

C++ – мова програмування високого рівня з підтримкою кількох парадигм програмування: об'єктно-орієнтованої, узагальненої та процедурної [36]. Стандартна бібліотека C++ включає стандартну бібліотеку C з

невеликими змінами, які роблять її відповіднішою для мови C++. Інша велика частина бібліотеки C++ заснована на Стандартній Бібліотеці Шаблонів (STL). Вона надає такі важливі інструменти, як контейнери (наприклад, вектори і списки) і ітератори (узагальнені вказівники), що надають доступ до цих контейнерів як до масивів. Крім того, STL дозволяє схожим чином працювати і з іншими типами контейнерів, наприклад, асоціативними списками, стеками, чергами.

Використовуючи шаблони, можна писати узагальнені алгоритми, здатні працювати з будь-якими контейнерами або послідовностями, доступ до членів яких забезпечують ітератори. Так само, як і в C, можливості бібліотек активізуються використанням директиви #include для включення стандартних файлів. Всього в стандарті C++ визначено 50 таких файлів.

Мова C++ багато в чому є надмножиною C. Нові можливості C++ включають оголошення у вигляді виразів, перетворення типів у вигляді функцій, оператори new і delete, тип bool, посилання, розширене поняття константності та змінності, функції, що підставляються, аргументи за замовчанням, перевизначення, простори імен, класи (включаючи і всі пов'язані з класами можливості, такі як успадкування, функції-члени (методи), віртуальні функції, абстрактні класи і конструктори), перевизначення операторів, шаблони, оператор ::, обробку винятків, динамічну ідентифікацію і багато що інше. Сі++ є також мовою строгого типування і накладає більше вимагань щодо дотримання типів, порівняно з Сі. У C++ з'явилися коментарі у вигляді подвійної косої риски («//»), які були в попереднику C — мові BCPL.

Деякі особливості C++ пізніше були перенесені в C, наприклад ключові слова const і inline, оголошення в циклах for і коментарі в стилі C++ («//»). У пізніших реалізаціях C також були представлені можливості, яких немає в C++, наприклад макроси vararg і покращена робота з масивами-параметрами. C++ додає до C об'єктно-орієнтовані можливості. Він вводить класи, які забезпечують три найважливіші властивості ООП: інкапсуляцію, успадкування і поліморфізм [36].

Java – об'єктно-орієнтована мова програмування, випущена 1995 року компанією «Sun Microsystems» як основний компонент платформи Java. В офіційній реалізації Java-програми компілюються у байт-код, який при виконанні інтерпретується віртуальною машиною для конкретної платформи [37]. Мова значно запозичила синтаксис із C і C++. Зокрема, взято за основу об'єктну модель C++, проте її модифіковано. Усунуто можливість появи деяких конфліктних ситуацій, що могли виникнути через помилки програміста та полегшено сам процес розробки об'єктно-орієнтованих програм. Ряд дій, які в C/C++ повинні здійснювати програмісти, доручено віртуальній машині. Передусім Java розроблялась як платформо-незалежна мова, тому вона має менше низькорівневих можливостей для роботи з апаратним забезпеченням, що в порівнянні, наприклад, з C++ зменшує швидкість роботи програм. За необхідності таких дій Java дозволяє викликати підпрограми, написані іншими мовами програмування.

На противагу C++, Java об'єктно-орієнтованіша. Всі дані і дії групуються в класи об'єктів. Виключенням з повної об'єктності (як скажімо в Smalltalk) є примітивні типи (int, float тощо). Це було свідомим рішенням проектувальників мови задля збільшення швидкості. Через це Java не вважається повністю об'єктно-орієнтовною мовою.

У Java всі об'єкти є похідними від головного об'єкта (він називається просто Object), з якого вони успадковують базову поведінку і властивості. Хоча у C++ вперше стало доступне множинне успадкування, але у Java можливе тільки одинарне успадкування, завдяки чому виключається можливість конфліктів між членами класу (методи і змінні), які успадковуються від базових класів.

Java використовує автоматичний збирач сміття для керування пам'яттю під час життєвого циклу об'єкта. Програміст вирішує, коли створювати об'єкти, а віртуальна машина відповідальна за звільнення пам'яті після того, як об'єкт стає непотрібним. Коли до певного об'єкта вже не залишається посилань, збирач сміття може автоматично прибирати його із пам'яті. Проте,

витік пам'яті все ж може статися, якщо код, написаний програмістом, має посилання на вже непотрібні об'єкти, наприклад на об'єкти, що зберігаються у діючих контейнерах.

Збирання сміття дозволене у будь-який час. В ідеалі воно відбувається під час бездіяльності програми. Збірка сміття автоматично форсується при нестачі вільної пам'яті в купі для розміщення нового об'єкта, що може призводити до кількасекундного зависання. Тому існують реалізації віртуальної машини Java з прибиральником сміття, спеціально створеним для програмування систем реального часу. Java не має підтримки вказівників у стилі C/C++. Це зроблено задля безпеки й надійності, аби дозволити збирачу сміття переміщувати вказівникові об'єкти.

Програми на Java утворені з визначень класів та інтерфейсів. Класи містять змінні та константи, які утримують дані, методи, які виконують дії, та конструктори, які створюють екземпляри класів — об'єкти. Дані можуть мати простий тип (наприклад байт, ціле число, символ) або бути посиланням на об'єкт. Мова Java є статично типізованою.

Java є суворо типізованою мовою, кожна змінна та вираз має тип, відомий на етапі компіляції. Типи даних Java належать до двох категорій: прості (primitive) та вказівникові (reference). До простих типів належить булевий(логічний) тип, числові типи та символьний тип. Числові типи складаються із цілих типів byte, short, int, long та дійсних типів float, double. Символьний тип представлений типом char. Вказівникові типи складаються із класів, інтерфейсів, масивів. Значенням вказівникового типу є вказівник на об'єкт — екземпляр класу чи масиву. Рядки є об'єктами класу String [37].

Мова Java має такі особливості, яких немає в мові C++ [36]:

- Java є типобезпечною мовою.
- Java-код компілюється спочатку не в машинний код, а в певний проміжний код, який надалі інтерпретується або компілюється, тоді як багато C++ компіляторів орієнтовані на компіляцію в машинний код заданої платформи.

- У мові Java є чіткі певні стандарти на введення-виведення, графіку, геометрію, діалог, доступ до баз даних і інших типових застосувань. Завдяки цим особливостям, застосунки на Java мають значно кращу кросплатформність, ніж C++, і часто, будучи написані для певного комп'ютера і операційної системи, працюють під іншими системами без змін. Програмісти, що пишуть на мові Java, не залежать від пакунків, нав'язаних розробниками компіляторів на дане конкретне середовище, що різко спрощує портування програм.

- У мові Java реалізовано повноцінне збирання сміття, якого немає в C++. Немає в C++ і засобів перевірки правильності вказівників.

- Мова Java є чисто об'єктно-орієнтованою, тоді як C++ підтримує як об'єктно-орієнтоване, так і процедурне програмування.

- В C++ відсутня повноцінна інформація про типи під час виконання RTTI. Цю можливість можна було б реалізувати в C++, маючи повну інформацію про типи під час компіляції CTTI.

- У C++ є можливість введення призначеного для користувача синтаксису за допомогою #define, що може привести до того, що модулі у великих пакетах програм стають сильно пов'язані один з одним. Це різко знижує надійність пакетів і можливість організації розділених модулів. З іншого боку, C++ надає достатньо засобів (константи, шаблони, вбудовані функції) для того, щоб практично повністю виключити використання #define.

Для створення інтерфейсів у Java передбачено різні фреймворки, які полегшують процес розробки. У курсовому проекті буде використано Swing Framework. Swing бібліотека має велику кількість елементів інтерфейсу за допомогою яких можна створити прийнятний дизайн з достатньою інтуїтивністю та продуктивністю.

В результаті порівняння було обрано мову програмування Java. Для поставленої задачі можна використати будь-яку із представлених мов, проте Java є зручнішою для написання програм і для створення інтерфейсів.

Для створення інтерфейсів у Java передбачено різні фреймворки, які полегшують процес розробки. У курсовому проекті буде використано Swing Framework. Swing бібліотека має велику кількість елементів інтерфейсу за допомогою яких можна створити прийнятний дизайн з достатньою інтуїтивністю та продуктивністю.

Як середовище програмування можна використати IntelliJ IDEA та Eclipse. Обидві платформи є дуже потужними, і підходять для задач будь-якої складності.

IntelliJ IDEA — комерційне інтегроване середовище розробки для різних мов програмування (Java, Python, Scala, PHP та ін.) від компанії JetBrains. IntelliJ IDEA надає інструменти для продуктивної роботи і ідеально підходить для створення комерційних, мобільних і веб-додатків.

Community версія середовища IntelliJ IDEA підтримує інструменти (у вигляді плагінів) для проведення тестування TestNG і JUnit, системи контролю версій CVS, Subversion, Mercurial і Git, засоби складання Maven, Ant, Gradle, мови програмування Java, Scala, Clojure, Groovy і Dart. Підтримується розробка застосунків для мобільної платформи Android. До складу входить модуль візуального проектування GUI-інтерфейсу Swing UI Designer, XML-редактор, редактор регулярних виразів, система перевірки коректності коду, система контролю за виконанням завдань і доповнення для імпорту та експорту проектів з Eclipse. Доступні засоби інтеграції з системами відстеження помилок JIRA, Trac, Redmine, Pivotal Tracker, GitHub, YouTrack, Lighthouse [38].

Можливості платформи передбачають наступне [39]:

- Розумне автодоповнення, інструменти для аналізу якості коду, зручна навігація, розширені рефакторингом і форматування для Java, Groovy, Scala, HTML, CSS, JavaScript, CoffeeScript, ActionScript, LESS, XML і багатьох інших мов.

- Підтримка всіх популярних фреймворків і платформ, включаючи Java EE, Spring Framework, Grails, Play Framework, GWT, Struts, Node.js, AngularJS, Android, Flex, AIR Mobile і багатьох інших.
- Інтеграція з серверами додатків, включаючи Tomcat, TomEE, GlassFish, JBoss, WebLogic, WebSphere, Geronimo, Resin, Jetty і Virgo.
- Інструменти для роботи з базами даних і SQL файлами, включаючи зручний клієнт і редактор для схеми бази даних.
- Інтеграція з комерційними системами управління версіями Perforce, Team Foundation Server, ClearCase, Visual SourceSafe.
- Інструменти для запуску тестів і аналізу покриття коду, включаючи підтримку всіх популярних фреймворків для тестування.

Eclipse — вільне модульне інтегроване середовище розробки програмного забезпечення. Розробляється і підтримується Eclipse Foundation і включає проекти, такі як платформа Eclipse, набір інструментів для програмістів на мові Java, системи контролю версій, конструктори GUI тощо. Написаний в основному на Java, може бути використаний для розробки застосунків на Java і, за допомогою різних плагінів, на інших мовах програмування, включаючи Ada, C, C++, COBOL, Fortran, Perl, PHP, Python, R, Ruby (включно з каркасом Ruby on Rails), Scala, Clojure та Scheme. Середовища розробки зокрема включають Eclipse ADT (Ada Development Toolkit) для Ada, Eclipse CDT для C/C++, Eclipse JDT для Java, Eclipse PDT для PHP [40].

Випущена на умовах Eclipse Public License, Eclipse є вільним програмним забезпеченням. Він став одним з перших IDE під GNU Classpath і без проблем працює під IcedTea.

Eclipse це фреймворк для розробки модульних платформонезалежних застосунків із низкою особливостей:

- можливість розробки ПЗ на багатьох мовах програмування (рідною є Java);
- платформонезалежна;

- модульна, призначена для подальшого розширення незалежним розробниками;

- з відкритим сирцевим кодом;

- розробляється і підтримується фондом Eclipse, куди входять такі постачальники ПЗ, як IBM, Oracle, Borland.

Спочатку проект розроблявся в IBM як корпоративний стандарт IDE для розробки на багатьох мовах під платформи IBM. Потім проект було перейменовано на Eclipse і надано для подальшого розвитку спільноті.

Eclipse насамперед повноцінна Java IDE, націлена на групову розробку, має засоби роботи з системами контролю версій (підтримка CVS входить у поставку Eclipse, активно розвиваються кілька варіантів SVN модулів, існує підтримка VSS та інших). З огляду на безкоштовність, у багатьох організаціях Eclipse — корпоративний стандарт для розробки ПЗ на Java.

Друге призначення Eclipse — служити платформою для нових розширень. Такими стали C/C++ Development Tools (CDT), розроблювані інженерами QNX разом із IBM, засоби для підтримки інших мов різних розробників. Безліч розширень доповнює Eclipse менеджерами для роботи з базами даних, серверами застосунків та інших.

З версії 3.0 Eclipse став не монолітною IDE, яка підтримує розширення, а набором розширень. У основі лежать фреймворки OSGi, і SWT/JFace, на основі яких розроблений наступний шар — платформа і засоби розробки повноцінних клієнтських застосунків RCP (Rich Client Platform). Платформа RCP є базою для розробки різних RCP програм як торент-клієнт Azareus чи File Arranger. Наступний шар — платформа Eclipse, що є набором розширень RCP — редактори, панелі, перспективи, модуль CVS і модуль Java Development Tools (JDT).

Eclipse написана на Java, тому є платформонезалежним продуктом, крім бібліотеки графічного інтерфейсу SWT, яка розробляється окремо для більшості поширених платформ. Бібліотека SWT використовує графічні

засоби платформи (ОС), що забезпечує швидкість і звичний зовнішній вигляд інтерфейсу користувача [40].

Оскільки платформа IntelliJ IDEA має більш потужні можливості, ніж платформа Eclipse, має зручніший інтерфейс та має ряд переваг, її було обрано для розробки програмного забезпечення для інформаційної технології.

3.2 Програмна реалізація інформаційної технології розв'язання задачі скорингу

Для реалізації програмного забезпечення було спроектовано нейронну мережу з використанням наступних класів та методів:

- клас Input – передбачає створення входів нейронної мережі та зв'язків між нейронами:

```
public class Input
{
   public Link[] OutgoingLinks;
};
```

- клас Link – створює нейрон Neuron та зв'язки Weight:

```
public class Link
{
   public Neuron Neuron;
   public double Weight;
};
```

- клас Neuron – визначає входи нейрона IncomingLinks та накопичений ним заряд Power:

```
public class Neuron
{
   public Link[] IncomingLinks;
   public double Power { get; set; }
};
```

- клас RbfNetwork – визначає РБФ-мережу, її входи та нейрони:

```
public class RbfNetwork
{
    private readonly Input[] _inputs;
    private readonly Neuron[] _neurons;
};
```

- метод Handle – отримує вхідний вектор та обробляє його;

- метод Train – навчання мережі:

```
void train(){
      for(int i=0;i<inputs.length;i++){
         double output=outputs[i];
         double predictedoutput=0;
         for(int j=0;j<inputs[i].length;j++){
            predictedoutput+=net[j].phi(inputs[i])*net[j].w;
         }
         for(int j=0;j<inputs[i].length;j++){
            net[j].update(inputs[i], output, predictedoutput);
         }
      }
   }
```

- метод Test – тестування мережі:

```
void test(double[] inputs){
      double predictedOutput=0;
      for(int i=0;i<inputs.length;i++){
         predictedOutput+=net[i].phi(inputs)*net[i].w;
      }
   }.
```

Ці класи моделюють нейронну мережу, дозволяють навчити її та протестувати. Також було використано бібліотеку Swing для створення інтерфейсу програми. Для цього було створено такі класи:

- клас Menu – створює головне меню програми та визначає розташування всіх його елементів, використовуючи методи create(), shov(), listener();

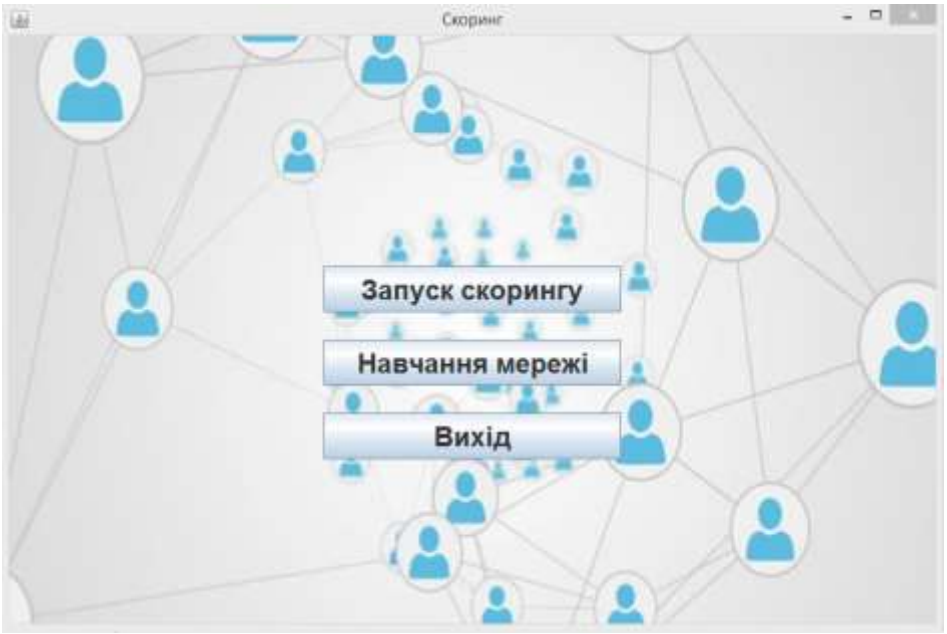
- клас TestQ1 – відповідає за відображення тестових запитань, використовуючи методи create(), shov() та listener();

- клас Result – відповідає за відображення результатів тестування, використовує методи create(), shov() та listener().

Усі наведені вище класи та методи використовують стандартні бібліотеки, запропоновані середовищем Android Studio. Вони дозволяють у повній мірі реалізувати поставлені в технічному завданні задачі.

3.3 Тестування та аналіз результатів роботи програми

Здійснимо тестування розробленого програмного продукту. Процес тестування відбувається у такому порядку. Запускаємо програму, внаслідок чого на екрані монітора з’являється головне меню програми (рисунок 3.1).



Рисусок 3.1 – Головне меню програми

Натисненням на кнопку Тестування відкриваємо вікно Тестування (рисунок 3.2). При завершення програми можна натиснути на кнопку Вихід.

У вікні тестування розміщенні тести з варіантами відповіді. Також тут є дві кнопки – повернення до головного меню та перехід до наступних тестів.

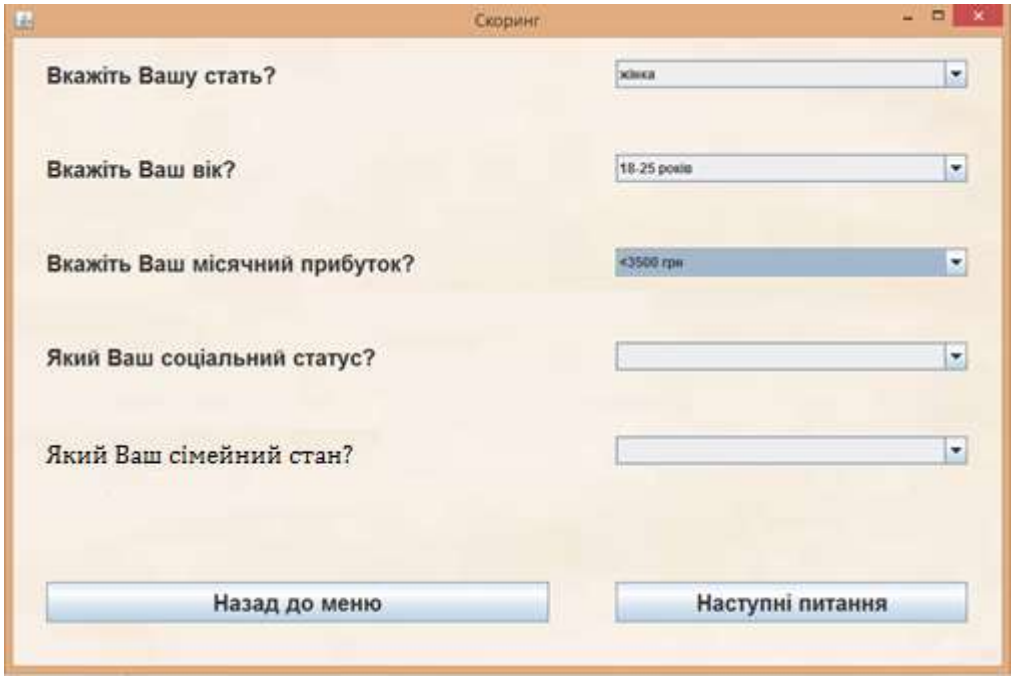


Рисунок 3.2 – Вікно тестування

Після останнього тесту з’являється кнопка завершення тестування. Завершивши тестування, користувач переходить на екран відображення результатів роботи програми (рисунок 3.3). Також тут є кнопка повернення до головного меню.

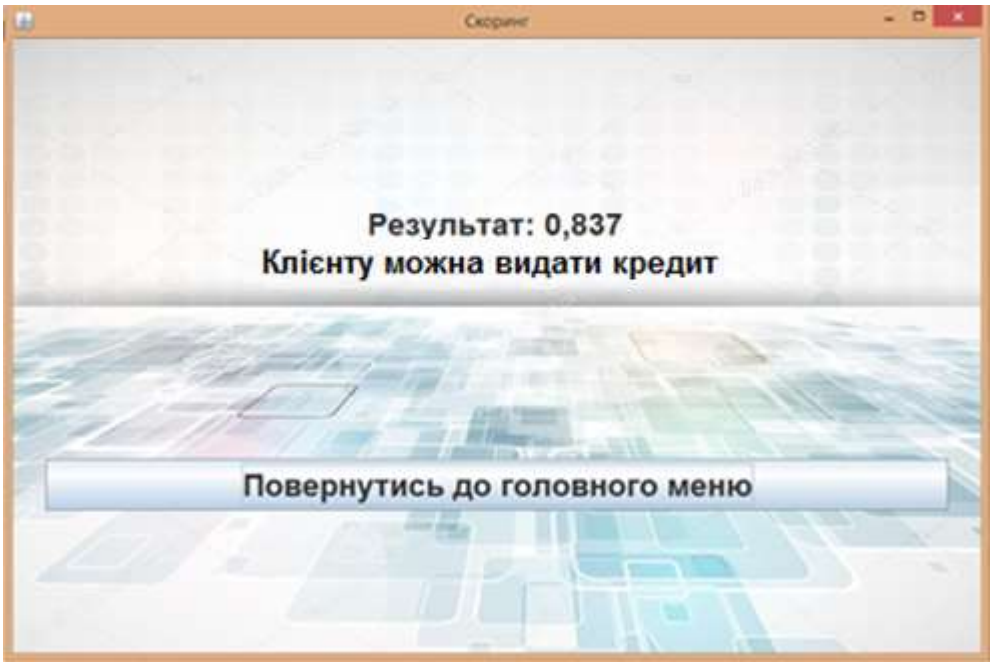


Рисунок 3.3 – Результат роботи програми

Доведемо поставлену мету роботи шляхом тестування. Для цього перевіримо достовірність розробленої програми. Достовірність будемо перевіряти на одних і тих самих даних для нашої програми та програми-аналогу. В якості програми-аналогу було взято програму Clementine (IBM SPSS Modeler). Для перевірки достовірності була використана вибірка із 600 клієнтів. В якості вибірки було використано анонімні дані клієнтів банку, надані на безкоштовній основі з перспективою подальшого використання програми в роботі банку. На практиці застосовують наступний підхід: ділять вибірку на навчальну та тестову, причому навчальна має бути більше тестової у декілька разів. Тому на навчальну вибірку було виділено 500 клієнтів, а на тестову – 100. Результати тестування подані у таблиці 3.2.

Таблиця 3.2 – Результати тестування програмного забезпечення

Параметр	Програма-аналог	Розроблена програма
Розмір тестової вибірки	100	100
Кількість правильно визначених прогнозів	85	92
Кількість неправильно	15	8

визначених прогнозів		
Достовірність	85%	92%

Із тестової вибірки наша програма правильно визначила скоринг для 92 клієнтів, а для 8 визначила неправильно. Отже, достовірність розробленого програмного продукту становить 92%.

На цій же вибірці перевіримо роботу програми-аналога. Із 100 наборів програма правильно визначила скоринг для 85 наборів, а для 15 – неправильно. Тому достовірність цієї програми становить 85%.

Достовірність роботи також визначається показником accuracy [41]. Він визначається наступним чином:

$$Accuracy = \frac{tp + tn}{tp + fp + fn + tn}.$$

Тут використовуються наступні характеристики:

- TP (True Positive) — істиннопозитивний. Класифікатор вирішив, що клієнту можна видати кредит, і кредит було видано.
- FP (False Positive) — хибнопозитивний. Класифікатор вирішив, що клієнту можна видати кредит, але кредит не було видано. Таке явище називається помилкою першого роду.
- FN (False Negative) — хибнонегативний. Класифікатор вирішив, що клієнту не можна видати кредит, але кредит було видано. Це помилка другого роду.
- TN (True Negative) — істиннонегативний. Класифікатор вирішив, що клієнту не можна видати кредит, і кредит не було видано.

Під час тестування програми було визначено, що характеристика tp для нього склала 52 випадків, fp склала 5 випадків, fn – 3 випадків, tn – 40 випадків. Тоді параметр accuracy для розробленого програмного забезпечення становить 92%.

Таким чином, результати тестування підтвердили високий ступінь достовірності визначення скорингу клієнтів. Розроблений програмний продукт дозволив підвищити вказаний параметр на 7%. Отже, мету роботи було досягнуто.

3.4 Висновок

У даному розділі було спроектовано програмне забезпечення для реалізації інформаційної технології розв'язання задачі скорингу. Було проаналізовано відомі мови програмування та середовища розробки, які можуть бути використані для розробки програми. На основі аналізу було обрано мову програмування Java та середовище розробки Intellij IDEA. Розроблену програму було навчено на навчальній вибірці та протестовано на тестовій. У результаті порівняння роботи розробленої програми та програми-аналогу було з’ясновано, що розроблена програма має на 7% вище достовірність, ніж програма-аналог. Отже, мета роботи була досягнута.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Оцінювання комерційного потенціалу розробки

Метою проведення технологічного аудиту є оцінювання комерційного потенціалу розробки. Для проведення технологічного аудиту було залучено 2-х незалежних експертів. Такими експертами будуть Колесницький Олег Костянтинович та Арсенюк Ігор Ростиславович.

Здійснюємо оцінювання комерційного потенціалу розробки за 12-ма критеріями за 5-ти бальною шкалою.

Результати оцінювання комерційного потенціалу розробки наведено в таблиці 4.1.

Таблиця 4.1 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта	
	1. Експерт 1	2. Експерт 2
	Бали, виставлені експертами:	
1	4	4
2	3	3
3	4	4
4	4	3
5	3	4
6	4	4
7	3	3
8	4	4
9	4	3
10	4	4
11	3	4
12	3	4
Сума балів	$СБ_1 = 43$	$СБ_2 = 44$
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_{1}^{3} СБ_i}{2} = 44$	

Отже, з отриманих даних таблиці 4.1 видно, що нова розробка має високий рівень комерційного потенціалу.

4.2 Прогнозування витрат на виконання науково-дослідної роботи та конструкторсько–технологічної роботи.

Для розробки нового програмного продукту необхідні такі витрати.

Основна заробітна плата для розробників визначається за формулою (4.1):

$$З_о = \frac{М}{Т_р} \cdot t, \qquad (4.1)$$

де М- місячний посадовий оклад конкретного розробника;

$Т_р$ - кількість робочих днів у місяці, $Т_р$ = 22 дні;

t - число днів роботи розробника, t = 50 днів.

Розрахунки заробітних плат для керівника і програміста наведені в таблиці 4.2.

Таблиця 4.2 – Розрахунки основної заробітної плати

Працівник	Оклад М, грн.	Оплата за робочий день, грн.	Число днів роботи, t	Витрати на оплату праці, грн.
Науковий керівник	6500	295,45	5	1477,25
Інженер-програміст	4000	181,81	50	9090,5
Всього:				10567,75

Розрахуємо додаткову заробітну плату:

$$З_{дод} = 0,1 \cdot 10567,75 = 1056,775 \text{ (грн.)}$$

Нарахування на заробітну плату операторів НЗП розраховується як 37,5…40% від суми їхньої основної та додаткової заробітної плати:

$$Н_{зп}= (З_о + З_р)\cdot \frac{\beta}{100}, \quad (4.2)$$

$$Н_{зп}= (10567,75+1056,775)\cdot \frac{36,3}{100} = 4219,7 \text{ (грн.)}.$$

Розрахунок амортизаційних витрат для програмного забезпечення виконується за такою формулою:

$$А=\frac{Ц\cdot Н_а}{100}\cdot\frac{Т}{12}, \quad (4.3)$$

де Ц – балансова вартість обладнання, грн;

$Н_а$ – річна норма амортизаційних відрахувань % (для програмного забезпечення 25%);

Т – Термін використання (Т=3 міс.).

Таблиця 4.3 – Розрахунок амортизаційних відрахувань

Найменування програмного забезпечення	Балансова вартість, грн.	Норма амортизації, %	Термін використання, міс.	Величина амортизаційних відрахувань,грн
Персональний комп’ютер	9500	25	3	593,75
Всього:				593,75

Розрахуємо витрати на комплектуючі. Витрати на комплектуючі розрахуємо за формулою:

$$K = \sum_1^n Н_і \cdot Ц_і \cdot К_і, \quad (4.4)$$

де n – кількість комплектуючих;

Ні - кількість комплектуючих і-го виду;

Ці – покупна ціна комплектуючих і-го виду, грн;

Кі – коефіцієнт транспортних витрат (приймемо Кі = 1,1).

Таблиця 4.4 - Витрати на комплектуючі, що були використані для розробки ПЗ.

Найменування матеріалу	Одиниці виміру	Ціна, грн.	Витрачено	Вартість витрачених матеріалів, грн.
Флешка	шт.	190	1	190
Пачка паперу	уп.	130	1	130
Ручка	шт.	10	1	10
Всього з урахуванням транспортних витрат				362

Витрати на силову електроенергію розраховуються за формулою:

$$В_е=В\cdot П\cdot Ф\cdot К_п, \quad (4.5)$$

де В – вартість 1кВт-години електроенергії (В=1,7 грн/кВт);

П – установлена потужність комп'ютера (П=0,6кВт);

Ф – фактична кількість годин роботи комп'ютера (Ф=195 год.);

$К_п$ – коефіцієнт використання потужності ($К_п< 1$, $К_п = 0,7$).

$$В_е= 1,7 \cdot 0,6 \cdot 195 \cdot 0,7 = 139,23 \text{ (грн.)}$$

Розрахуємо інші витрати $В_{ін}$.

Інші витрати $І_в$ можна прийняти як (100…300)% від суми основної заробітної плати розробників та робітників, які виконували дану роботу, тобто:

$$В_{ін} = (1..3) \cdot (З_о + З_р). \quad (4.6)$$

Отже, розрахуємо інші витрати:

$$В_{ін} = 1 * (10567,75+1056,775) = 11624,525 \text{ (грн).}$$

Сума всіх попередніх статей витрат дає витрати на виконання даної частини роботи:

$$В = З_о + З_д + Н_{зп} + А + К + В_е + І_в$$

$$В = 10567,75+1056,775+4219,7+593,75+362+139,23+11624,525= 28563,73 \text{ (грн.)}$$

Розрахуємо загальну вартість наукової роботи $B_{заг}$ за формулою:

$$В_{заг} = \frac{В_{ін}}{\alpha} \quad (4.7)$$

де α – частка витрат, які безпосередньо здійснює виконавець даного етапу роботи, у відн. одиницях = 1.

$$В_{заг} = \frac{28563,73}{1} = 28563,73$$

Прогнозування загальних витрат ЗВ на виконання та впровадження результатів виконаної наукової роботи здійснюється за формулою:

$$ЗВ = \frac{В_{заг}}{\beta} \quad (4.8)$$

де β – коефіцієнт, який характеризує етап (стадію) виконання даної роботи.

Отже, розрахуємо загальні витрати:

$$ЗВ = \frac{28563,73}{0,9} = 31737,48 \text{ (грн.)}$$

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки.

Спрогнозуємо отримання прибутку від реалізації результатів нашої розробки. Зростання чистого прибутку можна оцінити у теперішній вартості грошей. Це забезпечить підприємству (організації) надходження додаткових коштів, які дозволять покращити фінансові результати діяльності .

Оцінка зростання чистого прибутку підприємства від впровадження результатів наукової розробки. У цьому випадку збільшення чистого прибутку підприємства $\Delta П_і$ для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, розраховується за формулою:

$$\Delta П_i = \sum_1^n (\Delta П_я \cdot N + П_я \Delta N)_i \qquad (4.9)$$

де $\Delta П_я$ – покращення основного якісного показника від впровадження результатів розробки у даному році;

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки;

$П_я$ – основний якісний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки.

В результаті впровадження результатів наукової розробки витрати на виготовлення інформаційної технології зменшаться на 25 грн (що автоматично спричинить збільшення чистого прибутку підприємства на 25 грн), а кількість користувачів, які будуть користуватись збільшиться: протягом першого року – на 150 користувачів, протягом другого року – на 125 користувачів, протягом третього року – 100 користувачів. Реалізація інформаційної технології до впровадження результатів наукової розробки складала 500 користувачів, а прибуток, що отримував розробник до впровадження результатів наукової розробки – 400 грн.

Спрогнозуємо збільшення чистого прибутку від впровадження результатів наукової розробки у кожному році відносно базового.

Отже, збільшення чистого продукту $\Delta П_1$ протягом першого року складатиме:

$$\Delta П_1 = 25 \cdot 600 + (300 + 25) \cdot 180 = 73500 \text{ грн.}$$

Протягом другого року:

$$\Delta П_2 = 25 \cdot 600 + (300 + 25) \cdot (180 + 125) = 114125 \text{ грн.}$$

Протягом третього року:

$$\Delta П_3 = 25 \cdot 600 + (300 + 25) \cdot (180 + 125 + 100) = 146625 \text{ грн.}$$

4.4 Розрахунок ефективності вкладених інвестицій та період їх окупності

Визначимо абсолютну і відносну ефективність вкладених інвестором інвестицій та розрахуємо термін окупності.

Абсолютна ефективність $E_{абс}$вкладених інвестицій розраховується за формулою:

$$E_{абс} = (ПП - PV), \quad (4.10)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн;

т – період часу, протягом якого виявляються результати впровадженої НДДКР, 3 роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,1;

t – період часу (в роках) від моменту отримання чистого прибутку до точки 2, 3,4.

Рисунок, що характеризує рух платежів (інвестицій та додаткових прибутків) буде мати вигляд, рисунок 4.1.

Розрахуємо вартість чистих прибутків за формулою:

$$ПП = \sum_1^m \frac{\Delta П_i}{(1+\tau)^t} \quad (4.11)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн;

т – період часу, протягом якого виявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,1;

t – період часу (в роках) від моменту отримання чистого прибутку до точки.

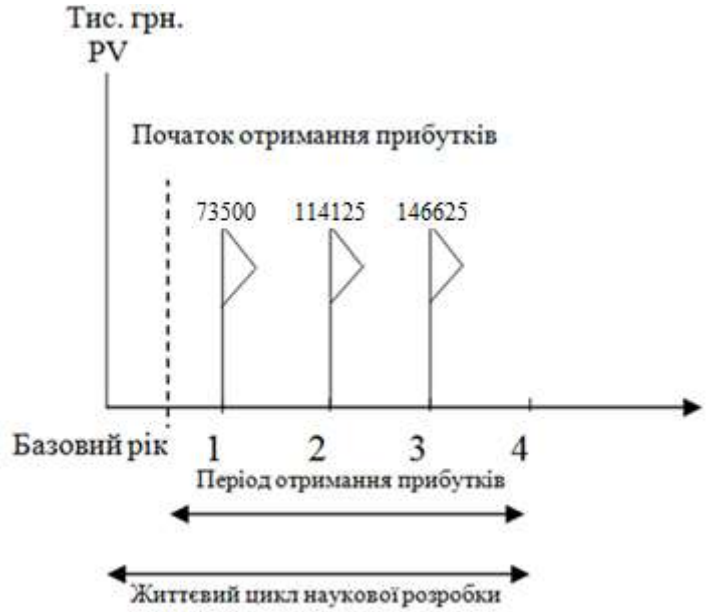


Рисунок 4.1 – Вісь часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів НДДКР

Отже, розрахуємо вартість чистого прибутку:

$$ПП = \frac{31737,48}{(1+0,1)^0} + \frac{73500}{(1+0,1)^2} + \frac{114125}{(1+0,1)^3} + \frac{146625}{(1+0,1)^4} = 278371,93 \text{ (грн.)}$$

Тоді розрахуємо $E_{абс}$:

$$E_{абс} = 278371,93 - 31737,48 = 246634,45 \text{ грн.}$$

Оскільки $E_{абс} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР буде доцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$ за формулою:

$$E_в = \sqrt[T]{1 + \frac{E_{абс}}{PV}} - 1 \qquad (4.12)$$

де $E_{абс}$ – абсолютна ефективність вкладених інвестицій, грн;

PV – теперішня вартість інвестицій PV = ЗВ, грн;

$T_ж$ – життєвий цикл наукової розробки, роки.

Тоді будемо мати:

$$E_в = \sqrt[3]{1 + \frac{246634,45}{31737,48}} - 1 = 2 \text{ або } 200\ \%$$

Далі, розраховану величина $E_в$ порівнюємо з мінімальною (бар'єрною) ставкою дисконтування $\tau_{мін}$, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть. У загальному вигляді мінімальна (бар'єрна) ставка дисконтування $\tau_{мін}$ визначається за формулою:

$$\tau = d + f,$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2019 році в Україні d = 0,2;

f – показник, що характеризує ризикованість вкладень, величина f = 0,1.

$$\tau = 0,2 + 0,1 = 0,3$$

Оскільки $E_в$ = 200% > $\tau_{мін}$ = 0,3 = 30%, то інвестор буде зацікавлений вкладати гроші в дану наукову розробку.

Термін окупності вкладених у реалізацію наукового проекту інвестицій. Термін окупності вкладених у реалізацію наукового проекту інвестицій $T_{ок}$ розраховується за формулою:

$$T_{ок} = \frac{1}{E_в}$$

$$T_{ок} = \frac{1}{2} = 0{,}5 \text{ року}$$

Обрахувавши термін окупності даної наукової розробки, можна зробити висновок, що фінансування даної наукової розробки буде доцільним.

4.5 Висновок

У даному розділі було проведено оцінювання комерційного потенціалу розробки, в результаті чого було встановлено, що нова розробка має високий рівень комерційного потенціалу. Було спрогнозовано витрати на виконання науково-дослідної роботи та конструкторсько-технологічної роботи, що будуть складати 31737,48 грн. Також було проведено прогнозування комерційних ефектів від реалізації результатів розробки. В результаті прогнозування було обраховано збільшення чистого продукту за перший, другий і третій роки впровадження розробки. Було розраховано ефективність вкладених інвестицій та період їх окупності. Вони склали 200 % та 0,5 року відповідно. Отже, розробка має високий комерціний потенціал розробки, що підтверджується розрахунками та терміном окупності.

ВИСНОВКИ

1. В ході виконання магістерської кваліфікаційної роботи було розроблено інформаційну технологію розв’язання задачі скорингу з використанням нейронної мережі. Одним з актуальних напрямків у галузі кібернетики є побудова програмних додатків для класифікації та кластеризації даних, що має місце у процесі визначення скорингу, адже відбувається поділ клієнтів на класи: можна видавати клієнту кредит чи ні.

2. При аналізі предметної області задачі скорингу було зв’ясовано, що більшість методів, які застосовуються до даної задачі, не є ефективними. На основі аналізу переваг та недоліків цих методів було обрано нейромережевий метод для реалізації інформаційної технології. Він має досить високу точність та швидкодію, а також може знаходити нелінійні зв’язки між вхідними даними, що дозволяє отримати найбільш достовірні результати. Було проведено аналіз існуючих програм-аналогів, які забезпечують визначення скорингу клієнтів. В якості програми-аналога для подальшого дослідження було обрано програмний продукт Clementine (IBM SPSS Modeler).

3. У другому розділі була проведена розробка інформаційної технології. Було розглянуто основні типи нейронних мереж, які використовуються для апроксимації функцій та класифікації об’єктів. На основі їх аналізу було обрано мережу радіально-базисних функцій. Було проведено аналіз математичної моделі РБФ-мережі та її навчання і запропоновано удосконалення моделі навчання за рахунок введення процедури підбору початкових значень вагових коефіцієнтів вихідного шару, що дозволить суттєво підвищити швидкість навчання та підвищити достовірність визначення скорингової оцінки.

4. Відповідно до поставлених задач було побудовано структуру інформаційної технології розв’язання задачі скорингу та розроблено алгоритм її роботи. На основі алгоритму було розроблено UML-діаграму класів програми, що реалізує інформаційну технологію.

5. У третьому розділі було спроектовано програмне забезпечення для реалізації інформаційної технології розв'язання задачі скорингу. Було розглянуто відомі мови програмування та середовища розробки, які можуть бути використані для розробки програми. Було обгрунтовано вибір мови програмування для реалізації технології та вибір середовища розробки.

6. Було проведено тестування розробленої програми. Було створено вибірку із 600 клієнтів. Вибірку було поділено на навчальну та тестову. Навчальна вибірка становила 500 клієнтів, тестова 100. За результатами тестування достовірність роботи програмного забезпечення становила 92%, що на 7% вище, ніж у програми-аналога. Отже, мету роботи було досягнуто. Помилка першого роду розробленої програми склала 5 випадків, а помилка другого роду – 3 випадки.

7. В ході економічного обгрунтування розробки було проведено оцінювання комерційного потенціалу розробки, спрогнозовано витрати на виконання науково-дослідної, дослідно-конструкторської та конструкторсь-котехнологічної роботи, спрогнозовано комерційні ефекти від реалізації результатів розробки, розраховано ефективність вкладених інвестицій та періоду їх окупності. В результаті оцінки було встановлено, що розробка має високий комерційний потенціал, адже ефективність вкладений інвестицій становитиме 200%, а період окупності складе 0,5 року.

8. Результати роботи були апробовані на Всеукраїнській науково-практичній інтернет-конференції [1] та плануються до впровадження у виробництво (додаток А).

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Семенова Л.М. Інформаційна технологія розв’язання задачі скорингу на основі нейронної мережі / Семенова Л. М., Колесницький О.К. – Вінниця: МН ВНТУ. Факультет інформаційних технологій та комп'ютерної інженерії, 2020. [Електронний ресурс] – https://conferences.vntu.edu.ua/index.php/mn/mn 2020/paper/view/8488.

2. Семенова Л. М. Програмний модуль вирішення задачі скорингу з використанням нейронної мережі / Семенова Л. М., Колесницький О.К. – Вінниця: НТКП ВНТУ. Факультет інформаційних технологій та комп'ютерної інженерії, 2018.

3. Semenova L. Artificial neural networks and their current level of development: article / Одеса: «Наукові записки Міжнародного гуманітарного університету», № 28, 2018.

4. Волик Н.Г. Скоринг як експертний метод оцінювання кредитного ризику комерційного банку при споживчому кредитуванні: стаття / Запоріжжя: «Вісник Запорізького національного університету», 2008.

5. Пещанская И.В. Краткосрочный кредит: теория и практика. / М.: Видавництво «Экзамен», 2011. 320 с.

6. Скоринг. Види. [Електронний ресурс] – режим доступа: https://www.banki.ru/wikibank/skoring/.

7. Куссуль Н. М. Інтелектуальні обчислення: навчальний посібник / Куссуль Н. М., Шелестов А. Ю., Лавренюк А. Н. – К. : «Наукова думка», 2006. – 186 с.

8. Скоринг. Застосування скорингу у маркетингу. [Електронний ресурс] – режим доступа: http://statsoft.ru/solutions/tasks/scoring/.

9. Любунь З. М. Основи теорії нейромереж: текст лекцій /Любунь З. М. – Львів : Видавничий центр ЛНУ ім. Івана Франка, 2006.

10. Класифікація з навчанням. Дискримінантний аналіз . [Електронний ресурс] – режим доступа: https://stud.com.ua/93361/statistika/klasifikatsiya_ navchannyam_diskriminantniy_analiz.

11. Бінарне розбиття простору. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/Бінарне_розбиття_простору.

12. Наївний баєсів класифікатор. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/Наївний_баєсів_класифікатор.

13. Генетичний алгоритм. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/Генетичний_алгоритм.

14. Метод k-найближчих сусідів. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/Метод_k-найближчих_сусідів.

15. Краткий курс машинного обучения. [Електронний ресурс] – режим доступа: https://habr.com/post/340792/.

16. SAS CREDIT SCORING. Офіційний сайт. [Електронний ресурс] – режим доступа: https://www.sas.com/ru_ua/software/credit-scoring.html

17. Можливості SAS CREDIT SCORING. [Електронний ресурс] – режим доступа: https://korusconsulting.ru/platforms/advanced-analytics/sas/sas-credit-scoring/

18. EGAR Technology. Кредитование физических лиц. [Електронний ресурс] – режим доступа: http://www.egartech.ru/fields/consumerlending/.

19. Plug&Score Modeler. Product Details . [Електронний ресурс] – режим доступа: https://www.capterra.com/p/133040/Plug-Score-Modeler/.

20. Predictive solutions. IBM SPSS Modeler. [Електронний ресурс] – режим доступа: http://www.predictivesolutions.ru/software/modeler.htm.

21. Перцептрон. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/Перцептрон.

22. Багатошаровий перцептрон Розенблатта. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/Багатошаровий_перцептрон_ Розенблатта.

23. Перцептрон. [Електронний ресурс] – режим доступа: https://www.wikiwand.com/uk/Перцептрон.

24. Внедряем скоринг лидов. [Електронний ресурс] – режим доступа: https://timedigitalcrm.com/blog/marketing-automation/lead-scoring-vybor-servisa/.

25. STATISTICA Automated Neural Networks. [Електронний ресурс] –режим доступа: http://statsoft.ru/products/STATISTICA_Neural_Networks/.

26. Мережа радіальних базисних функцій. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/Мережа_радіальних_базисних_функцій.

27. Основи нейронних мереж. Теорія та практика. / Кондратенко Н.Р., Куземко С.М.: посібник – Вінниця: ВНТУ, 2006. – 104 с.

28. Медведев В. С. Нейронные сети / В. С. Медведев, В. Г. Потемкин/ - М.: "Диалог-МИФИ", 2002 г, 496 с.

29. Ймовірнісна нейронна мережа. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/Ймовірнісна_нейронна_мережа.

30. Мережа радіальних базисних функцій. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/Мережа_радіальних_базисних_функцій.

31. Schwenker, Friedhelm; Kestler, Hans A.; Palm, Günther (2001). Three learning phases for radial-basis-function networks. Neural Networks 14: 439–458. doi:10.1016/s0893-6080(01)00027-2.

32. Э.М. Меликов. Использование кредитного скоринга и аппарата искусственных нейронных сетей для оценки кредитоспособности юридических лиц: стаття / Э.М. Меликов. СГУТИ, 2014 г.

33. Z-модель Альтмана. [Електронний ресурс] – режим доступа: https://www.audit-it.ru/finanaliz/terms/analysis/altman_z_model.html.

34. Аналіз фінансового стану підприємства. Модель Чессера. [Електронний ресурс] – режим доступа: http://afdanalyse.ru/publ/finansovyj_analiz/1/model_chessera/16-1-0-142.

35. Сравнение языков программирования [Електронний ресурс] – режим доступа: https://ru.wikipedia.org/wiki/Сравнение_языков_программирования.

36. Мова програмування C++. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/C%2B%2B.

37. Мова програмування Java. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/Java.

38. IntelliJ IDEA. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/IntelliJ_IDEA.

39. IntelliJ IDEA. Офіційний сайт. [Електронний ресурс] – режим доступа: https://jetbrains.ru/products/idea/.

40. Eclipse. [Електронний ресурс] – режим доступа: https://uk.wikipedia.org/wiki/Eclipse.

41. Метрики в задачах машинного обучения [Електронний ресурс] – режим доступа: https://habr.com/ru/company/ods/blog/328372/.