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

НН Інститут прикладного системного аналізу

Кафедра математичних методів системного аналізу

До захисту допущено:

Завідувач кафедри

_____ Оксана ТИМОЩУК

«___»_____2023 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системний аналіз і управління»

спеціальності 124 «Системний аналіз»

на тему: «Система оцінювання кредитоспроможності позичальників з використанням методів інтелектуального аналізу даних»

Виконав:
Студент IV курсу, групи КА-95
Загуменний Олексій Олексійович _____

Керівник:
професор, д.т.н., професор кафедри ММСА
Бідюк Петро Іванович _____

Консультант з економічного розділу:
канд.економ.наук, доцент Рощина Надія Василівна _____

Консультант з нормоконтролю:
науковий співробітник кафедри ММСА,
к.ф.-м.н. Статкевич Віталій Михайлович _____

Рецензент:
д.т.н., проф., НТУУ «КПІ ім. І. Сікорського»,
Сергій Федорович Теленик _____

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших авторів
без відповідних посилань.
Студент _____

Київ – 2023 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
НН Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 124 «Системний аналіз»

Освітня програма «Системний аналіз і управління»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Оксана ТИМОЩУК
«__» травня 2023 р.

ЗАВДАННЯ

на дипломну роботу студенту
Загуменному Олексію Олексійовичу

1. Тема роботи «Система оцінювання кредитоспроможності позичальників з використанням методів інтелектуального аналізу даних», керівник роботи Бідюк Петро Іванович, д.т.н., професор, затверджені наказом по університету від «__» травня 2023 р. № _____

2. Термін подання студентом роботи 12.06.2023.

3. Вихідні дані до роботи

Датасет з інформацією про позичальників банку.

4. Зміст роботи

Теоретичний аналіз методів оцінювання кредитоспроможності позичальників, аналіз використання методів інтелектуального аналізу в кредитуванні, побудова системи оцінювання кредитоспроможності з використанням методів інтелектуального аналізу даних, функціонально-вартісний аналіз продукту.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): обробка даних, вибір моделей, тренування моделей.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Н. В., доцент		

7. Дата видачі завдання лютий 2023р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Формулювання тематики (напрямку) дослідження.	15.02.2023 – 31.02.2023	виконано
2	Огляд літератури за темою дослідження.	01.03.2023 – 07.03.2023	виконано
3	Формулювання задач дослідження, розробка плану роботи.	07.03.2023 – 21.03.2023	виконано
4	Аналіз відомих результатів стосовно тематики дослідження.	22.03.2023 – 06.04.2023	виконано
5	Обробка та аналіз даних.	07.03.2023 – 13.03.2023	виконано
6	Затвердження теми дипломної роботи.	14.03.2023 – 15.03.2023	виконано
7	Розробка програмного продукту.	16.03.2023 – 05.05.2023	виконано
8	Виконання обчислювальних експериментів, аналіз та оформлення результатів.	06.05.2023 – 13.05.2023	виконано
9	Оформлення пояснювальної записки.	14.05.2023 – 16.05.2023	виконано

Студент Олексій ЗАГУМЕННИЙ

Керівник Петро БІДЮК

РЕФЕРАТ

Дипломна робота: 140 с., 47 рис., 9 табл., 29 джерел, 2 додатки.

ПРОГНОЗУВАННЯ, СТРАХУВАННЯ, АНСАМБЛЕВІ МОДЕЛІ.

Тема: Система оцінювання кредитоспроможності позичальників з використанням методів інтелектуального аналізу даних

У роботі розглянуті різні моделі оцінювання кредитоспроможності позичальників із використанням машинного навчання.

Об’єкт дослідження: застосування методів інтелектуального аналізу для визначення кредитоспроможності.

Предмет дослідження: методи інтелектуального аналізу та засоби їх реалізації

Мета роботи: розробка програмного забезпечення для оцінки кредитоспроможності клієнта

Створено програмне забезпечення у вигляді моделі для оцінки кредитоспроможності клієнта. Для розробки моделі був використаний набір даних із сайту Kaggle.

ABSTRACT

Diploma Thesis: 140 p., 47 fig., 9 tabl., 29 sources, 2 appendices.

FORECASTING, INSURANCE, ENSEMBLE MODELS.

Theme: The creditworthiness evaluation system of borrowers using methods of intelligent data analysis

The paper considers various models for assessing borrowers' creditworthiness using machine learning.

Object of research: application of intellectual analysis methods for determining creditworthiness.

Subject of research: methods of intellectual analysis and means of their implementation

Purpose: to develop software for assessing the client's creditworthiness

Software in the form of a model for assessing a client's creditworthiness was created. To develop the model, a dataset from the Kaggle website was used.

ЗМІСТ

ВСТУП 9

Обґрунтування актуальності теми 9

Мета та завдання дослідження 10

Об’єкт та предмет дослідження 11

1. ТЕОРЕТИЧНИЙ АНАЛІЗ МЕТОДІВ ОЦІНЮВАННЯ КРЕДИТОСПРОМОЖНОСТІ ПОЗИЧАЛЬНИКІВ 13

1.1. Основні підходи та методи оцінки кредитоспроможності 13

1.2. Моделі скорингу та їх застосування 15

1.3. Методи інтелектуального аналізу даних в оцінюванні кредитоспроможності 19

1.4. Аналіз існуючих систем оцінювання кредитоспроможності в Україні та за кордоном 21

1.5. Висновки до розділу та постановка задачі 25

2. АНАЛІЗ ВИКОРИСТАННЯ МЕТОДІВ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ В КРЕДИТУВАННІ 27

2.1. Загальний аналіз алгоритмів машинного навчання в банківському кредитуванні 27

2.2. Аналіз використання подібних моделей у кредитному скорингу 28

2.3. Оцінка потенційного економічного ефекту від використання системи 30

2.4. Вибір моделей для кредитного скорингу 33

2.4.1. Логістична регресія 34

2.4.2. Лінійний дискримінантний аналіз 35

2.4.3. К-найближчих сусідів 37

2.4.4. Дерева рішень 38

2.4.5. Наївний байєсівський класифікатор 39

2.4.6. Метод опорних векторів 39

2.4.7. Градієнтний бустинг 40

2.4.8. Адаптивний бустинг 42

2.5. Висновки до розділу 43

3. ПОБУДОВА СИСТЕМИ ОЦІНЮВАННЯ КРЕДИТОСПРОМОЖНОСТІ З ВИКОРИСТАННЯМ МЕТОДІВ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ 44

3.1. Створення комп'ютерної системи для аналізу кредитоспроможності фізичних осіб 44

3.2. Створення бази даних та її обробка 45

3.2.1. Опис обраної бази даних 45

3.2.2. Аналіз наявних даних 47

3.2.3. Аналіз розподілу класів та відношень між даними 53

3.2.4. Аналіз даних на кореляцію 60

3.2.5. Обробка даних 61

3.2.6. Збалансування даних, створення декількох наборів для порівняння 67

3.3. Валідація та тестування розробленої системи 69

3.4. Висновки до розділу 83

4. ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОДУКТУ 85

4.1. Постановка задачі 86

4.2. Функції програмного продукту 87

4.3. Вибір параметрів для продукту 93

4.4. Аналіз параметрів від експертів 94

4.5. Аналіз якості реалізації варіантів функцій 98

4.6. Економічний аналіз розробки 100

4.7. Вибір кращого варіанту ПП техніко-економічного рівня 105

4.8. Висновки до розділу 106

ВИСНОВКИ 107

Висновки за результатами дослідження 107

Рекомендації щодо використання розробленої системи оцінювання кредитоспроможності 108

Перспективи подальшого дослідження теми 108

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 110

ДОДАТОК А ЛІСТИНГ 115

ДОДАТОК Б ПРЕЗЕНТАЦІЯ 134

ВСТУП

Обґрунтування актуальності теми.

Тема "Система оцінювання кредитоспроможності позичальників з використанням методів інтелектуального аналізу даних" має високу актуальність у сучасних умовах фінансової індустрії. Оцінка кредитоспроможності є однією з найважливіших процедур, що здійснюється банками та іншими фінансовими установами перед наданням кредитів та інших видів фінансових послуг.

Однак, зрозуміло, що традиційні методи оцінки кредитоспроможності, такі як рейтингові системи та оцінка досвідченими працівниками, мають свої обмеження та можуть бути недостатньо ефективними, особливо при обробці великих обсягів даних. У таких умовах, використання методів інтелектуального аналізу даних може забезпечити значно більш точну та ефективну оцінку кредитоспроможності позичальників.

Також, враховуючи зростаючу конкуренцію у фінансовій індустрії, банки та інші фінансові установи потребують інноваційних технологій та підходів, щоб забезпечити свою конкурентну позицію на ринку та збільшити свої прибутки. Використання систем оцінки кредитоспроможності, заснованих на інтелектуальному аналізі даних, може бути одним з ключових факторів у покращенні ефективності та конкурентоспроможності фінансових установ.

Основними перевагами таких систем є те, що подібні системи можуть використовувати більш широкий спектр даних для оцінки

кредитоспроможності, включаючи не тільки фінансові, але й соціальні та демографічні дані. Це дозволяє банкам та іншим фінансовим установам отримувати більш повну та точну картину про позичальників, їхній фінансовий стан та здатність повернути кредит. Також використання подібних систем дозволяє автоматизувати процес оцінки та покращити ефективність банкових установ.

Отже, розробка системи оцінювання кредитоспроможності позичальників з використанням методів інтелектуального аналізу даних є актуальною і важливою темою для досліджень в наш час та може принести багато користі банковій сфері.

Мета та завдання дослідження.

Метою дослідження є розробка та впровадження системи оцінювання кредитоспроможності позичальників з використанням методів інтелектуального аналізу даних. Для досягнення цієї мети передбачається вирішення наступних завдань:

1. Аналіз існуючих методів та систем оцінювання кредитоспроможності позичальників.
2. Вивчення методів інтелектуального аналізу даних та їх застосування для оцінювання кредитоспроможності.
3. Розробка та побудова системи оцінювання кредитоспроможності на основі методів інтелектуального аналізу даних.
4. Валідація та перевірка ефективності розробленої системи на тестових даних.

5. Оцінка практичної значущості та переваг використання розробленої системи в порівнянні з традиційними методами оцінювання кредитоспроможності.

Дослідження має на меті покращити ефективність та точність оцінювання кредитоспроможності позичальників, що може призвести до зниження ризику неповернення кредитів та збільшення прибутку банків та інших фінансових установ. Результати дослідження можуть бути корисними для фінансових установ, що займаються кредитуванням позичальників, а також для наукових досліджень в галузі інтелектуального аналізу даних та його застосування в фінансових сферах.

Об’єкт та предмет дослідження.

Об’єктом дослідження є процес оцінювання кредитоспроможності позичальників, що є важливим елементом діяльності фінансових установ.

Предметом дослідження є система оцінювання кредитоспроможності позичальників з використанням методів інтелектуального аналізу даних. Основні складові системи це:

1. Вхідні дані - інформація про позичальника, його доходи, заборгованості, кредитну історію та інші фінансові показники.
2. Методи інтелектуального аналізу даних - алгоритми та моделі, які дозволяють виконувати аналіз та прогнозування на основі вхідних даних.
3. Вихідні результати - оцінка кредитоспроможності позичальника та прийняття рішення про виділення або відмову у кредитуванні.

В першому розділі роботи буде проведено теоретичний аналіз наявних методів оцінювання кредитоспроможності позичальників різних фінансових установ.

В другому розділі буде розглянуто методи та приклади оцінювання кредитоспроможності клієнтів із використанням методів інтелектуального аналізу даних. Також буде проведено оцінку потенційного економічного ефекту від розробки за темою роботи та буде обрано алгоритми для визначення кредитоспроможності клієнтів.

В третьому розділі буде проведено аналіз даних для оцінки кредитоспроможності клієнтів, її обробка, побудова моделей для проведення аналізу та оцінка роботи моделей.

В останньому розділі буде проведено функціонально-вартісний аналіз, що дозволяє оцінити вартість та шляхи впровадження та розробки даної системи.

1 ТЕОРЕТИЧНИЙ АНАЛІЗ МЕТОДІВ ОЦІНЮВАННЯ КРЕДИТОСПРОМОЖНОСТІ ПОЗИЧАЛЬНИКІВ

1.1 Основні підходи та методи оцінки кредитоспроможності.

Оцінка кредитоспроможності є важливим етапом у процесі кредитування, що допомагає кредиторам оцінити ризик дефолту позичальника.

Існує декілька підходів та методів, що використовуються для оцінки кредитоспроможності, деякі з них розглядаються нижче:

1. За даними з джерела [1], оцінка кредитоспроможності повинна відповідати двом взаємопов'язаним основним цілям економічної політики: зменшенню фінансової нестабільності та розвитку надійних методів управління кредитними ризиками відповідно до вимог відповідального банкінгу. Для досягнення цих цілей оцінка кредитоспроможності повинна включати кредитний ризик (для кредитора) та ризик доступності (для позичальника). Кредитний ризик - це ймовірність того, що позичальник не виконує свої зобов'язання, тоді як ризик доступності - це здатність позичальника погасити кредит без фінансових труднощів.
2. Інше джерело [2] свідчить, що при використанні автоматизованих моделей для оцінки кредитоспроможності та прийняття кредитних рішень установи повинні розуміти моделі, що використовуються, їх методологію, вхідні дані, припущення, обмеження та результати. Установи також повинні мати політику і процедури

для виявлення і запобігання упередженості та забезпечення якості вхідних даних.

3. У статті [3] зазначається, що перед укладенням кредитного договору кредитодавець повинен оцінити кредитоспроможність споживача на підставі достатньої інформації, отриманої від споживача, та, за необхідності, на підставі консультацій з бюро кредитних історій. Оцінка повинна враховувати доходи, витрати та інші фінансові зобов'язання споживача. Кредитор також повинен враховувати характер і тривалість кредитного договору, період погашення та суму запитуваного кредиту.
4. В іншій статті [4] було розроблено методику z-скорингу для оцінки кредитоспроможності, врегульовано кредитні ризики, з'явилися провайдери рейтингів, які забезпечують певний рівень незалежності. Z-скоринг - це статистичний метод, який використовує фінансові коефіцієнти для прогнозування ймовірності банкрутства компанії. Кредитні ризики регулюються шляхом встановлення лімітів на співвідношення кредиту до вартості, співвідношення кредиту до доходу та співвідношення боргу до доходу. Рейтингові агентства оцінюють кредитоспроможність позичальників на основі їхньої фінансової стійкості, доходів та кредитної історії.

Отже, існує декілька підходів та методів, які можуть бути використані для оцінки кредитоспроможності. Вибір методу залежить від типу кредитної угоди, фінансового стану позичальника та схильності кредитора до ризику. Кредиторам важливо розуміти моделі та методології, що використовуються для оцінки кредитоспроможності, а також мати політику та процедури, що забезпечують якість вхідних даних та запобігають упередженості.

1.2 Моделі скорингу та їх застосування.

Існує кілька скорингових моделей, що використовуються для оцінки кредитоспроможності, і кожна з них має свої сильні та слабкі сторони.

Традиційні методи кредитного скорингу використовуються в кредитній індустрії протягом десятиліть. Ці методи зазвичай використовують набір заздалегідь визначених правил і критеріїв для оцінки кредитоспроможності заявника [5]. Вони базуються на статистичних моделях, які використовують таку інформацію, як дохід, історія працевлаштування та кредитна історія для розрахунку кредитного рейтингу. Хоча ці методи були ефективними в минулому, вони мають певні обмеження. Наприклад, вони можуть не враховувати певні фактори, важливі для оцінки кредитоспроможності, такі як активність у соціальних мережах або інші нетрадиційні джерела даних.

З іншого боку, методи інтелектуального аналізу даних можуть допомогти усунути деякі з цих обмежень. Вони призначені для швидкого і точного аналізу великих обсягів даних і можуть виявити закономірності та взаємозв'язки, які можуть бути очевидними при використанні традиційних методів кредитного скорингу. Наприклад, одне дослідження показало, що штучні нейронні мережі дають більш точну оцінку ймовірності дефолту, ніж інші досліджувані методи інтелектуального аналізу даних [6]. В іншому дослідженні для кредитного скорингу використовувалася гібридна модель інтелектуального аналізу даних з алгоритмів відбору ознак і класифікаторів ансамблевого навчання, яка перевершила інші моделі за точністю і прогностичною здатністю [7].

Хоча методи інтелектуального аналізу даних можуть бути більш точними, ніж традиційні методи кредитного скорингу, вони також мають певні обмеження. Наприклад, вони можуть бути більш складними і важкими для розуміння, ніж традиційні методи, і можуть вимагати більше досвіду і ресурсів

для впровадження. Крім того, методи інтелектуального аналізу даних можуть вимагати більше даних, ніж традиційні методи, що може бути проблемою для невеликих кредиторів або тих, хто має обмежені джерела даних [8].

Також методи інтелектуального аналізу даних можна використовувати разом із традиційними методами кредитного скорингу. Насправді, багато досліджень вивчали використання гібридних моделей, які поєднують як традиційні методи, так і методи інтелектуального аналізу даних. Наприклад, в одному дослідженні для покращення результатів кредитного скорингу використовували двоетапну процедуру гібридного моделювання зі штучними нейронними мережами та багатовимірними адаптивними регресійними сплайнами [9]. В іншому дослідженні було запропоновано гібридну ансамблеву модель кредитного скорингу, яка поєднувала два методи попередньої обробки даних з різними алгоритмами класифікації на етапі ансамблевого моделювання. Було показано, що модель покращує ефективність прогнозування порівняно з усіма базовими класифікаторами та традиційними комбінованими методами з точки зору середньої точності, площі під кривою (AUC, або Area Under the Curve) та оцінки Brier Score [9].

Інші дослідження порівнювали точність різних методів інтелектуального аналізу даних з традиційними методами кредитного скорингу. Наприклад, в одному дослідженні порівнювалася точність логістичної регресії, нейронних мереж і дерев рішень у кредитному скорингу для кредитування малого бізнесу. Результати показали, що ймовірнісна нейромережева модель досягла найвищої точності. Аналогічно, в іншому дослідженні було виявлено, що штучні нейронні мережі забезпечують більш точну оцінку ймовірності дефолту, ніж інші досліджувані методи інтелектуального аналізу даних [10].

Отже, методи інтелектуального аналізу даних можна використовувати в поєднанні з традиційними методами кредитного скорингу для підвищення точності та прогностичної здатності моделей кредитного скорингу. Особливо

ефективними виявилися гібридні моделі, які поєднують як традиційні методи, так і методи інтелектуального аналізу даних. Однак кредиторам важливо ретельно оцінити сильні та слабкі сторони різних методів і вибрати той, який найкраще відповідає їхнім потребам.

Деякі з найпоширеніших скорингових моделей включають:

1. Показник FICO (Fair Isaac Corporation Score) - це тризначне число, яке підсумовує ваш кредитний ризик на основі вашої кредитної справи в одному з трьох основних бюро кредитних історій і використовується кредиторами для визначення вашої здатності погашати позики. Показники FICO зазвичай знаходяться в діапазоні 300-850 балів, при цьому хороший кредитний рейтинг зазвичай вважається між 670 та 739 балами в межах цього діапазону.
2. VantageScore - це ще одна система кредитного рейтингу, яка була спільно розроблена трьома найбільшими кредитними бюро (Experian, TransUnion та Equifax), щоб прогнозувати ймовірність повернення кредиту. Вона теж надає оцінку в діапазони 300-850 балів для позичальників, як і попередня.
3. Дерево кредитного скорингу - це ще один загальновживаний метод у сфері скорингу. Даний метод використовує дерева рішень для Дерева рішень набули популярності в кредитному скорингу за останні роки і зараз широко використовуються для підбору даних і прогнозування дефолту. Алгоритми в деревах рішень дотримуються низхідного підходу, де на кожному кроці вибирається змінна, яка найкраще розбиває набір даних.
4. Логістична регресія - це широко використовуваний метод статистичного моделювання для прогнозування кредитного ризику. Логістична регресія пов'язує оцінку та ймовірність дефолту за допомогою регресії та сигмоїдної функції активації. У кредитному скорингу логістична

регресія є стандартною моделлю для розробки інструментів оцінки клієнтів.

5. Кредитний скоринг із використанням штучних нейромереж - це ще один метод для оцінки кредитоспроможності позичальників, в якому використовують штучні нейронні мережі для аналізу кредитних даних і прогнозування кредитного ризику. Було виявлено, що нейромережі дозволяють досягти покращення точності кредитного скорингу, проте вони вимагають певних навичок моделювання для розробки топології мережі та досконаліших методів навчання.

Кожна скорингова модель має свої переваги та недоліки, і вибір моделі буде залежати від схильності кредитора до ризику, типу кредиту та фінансового стану позичальника. Деякі моделі можуть бути більш придатними для оцінки кредитоспроможності великих корпорацій, тоді як інші можуть бути більш придатними для оцінки кредитоспроможності фізичних осіб або малого бізнесу.

Кредитори зазвичай обирають, яку скорингову модель використовувати, виходячи з типу кредиту та фінансового стану позичальника. Наприклад, при оцінці іпотечної заявки кредитори можуть використовувати іншу скорингову модель, ніж при оцінці заявки на кредитну картку. За даними джерела [11], кредитори, як правило, можуть вибирати, який показник FICO використовувати при перевірці кредитоспроможності, але зазвичай послуговуються повними версіями в залежності від виду кредиту, на який ви подаєте заявку. Наприклад, при оцінці заявки на іпотечний кредит кредитори можуть використовувати показники FICO, пропоновані трьома основними кредитними бюро, які відрізняються від показників, що використовуються для оцінки автокредитів або кредитних карток.

Кредитори також можуть використовувати власні методики підрахунку балів, які використовують схожі, але не ідентичні фактори при визначенні права заявника на отримання кредиту. За даними статті [12], кредитори

можуть враховувати такі фактори, як розмір доходу позичальника, кількість грошей на рахунку в банку та тривалість його трудового стажу, на додаток до його кредитного рейтингу.

У деяких випадках кредитори можуть використовувати декілька скорингових моделей для оцінки кредитоспроможності позичальника. Наприклад, вони можуть використовувати як оцінку FICO, так і VantageScore, щоб отримати більш повну картину кредитної історії позичальника та його платіжної поведінки.

Отже, кредитори вирішують, яку скорингову модель використовувати, виходячи з декількох факторів, включаючи тип кредиту, фінансовий стан позичальника та власні скорингові методології. Хоча для оцінки одного й того ж позичальника можуть використовуватися різні скорингові моделі, кожна з них має свої сильні та слабкі сторони і призначена для оцінки кредитоспроможності під різним кутом зору.

1.3 Методи інтелектуального аналізу даних в оцінюванні кредитоспроможності.

Разом з традиційними методами, з'явилися і нові підходи до оцінювання кредитної ризикованості. Методи інтелектуального аналізу даних, такі як логістична регресія, штучні нейронні мережі, дерева рішень, машини опорних векторів (SVM) та Байєсівські мережі, надають нові можливості для прогнозування ймовірності дефолту та встановлення категорій ризику.

Ці методи базуються на статистичних алгоритмах та машинному навчанні, які дозволяють аналізувати великі обсяги даних та виявляти складні

залежності між різними факторами. Вони допомагають кредиторам отримувати більш об'єктивну та точну інформацію про кредитоспроможність позичальників, знижуючи ризики та покращуючи прогнози моделей.

Логістична регресія є статистичним методом, який застосовується для прогнозування ймовірності бінарних результатів, таких як дефолт або не дефолт, заснованої на кредитній історії позичальника, історії платежів та інших факторах.

Штучні нейронні мережі є типом алгоритму машинного навчання, що застосовується для прогнозування ймовірності дефолту, враховуючи кредитну історію позичальника, історію платежів та інші фактори. Нейронні мережі продемонстрували свою ефективність у кредитному скорингу, хоча їх інтерпретація ускладнюється через комплекність моделі та велику кількість нелінійних зв’язків, що утворюються.

Дерева рішень є методом інтелектуального аналізу даних, який дозволяє виявити важливі фактори, впливаючи на кредитоспроможність. Вони особливо корисні для виявлення складних залежностей між змінними і можуть підтримати кредиторів у прийнятті обґрунтованих рішень.

Машини опорних векторів використовуються для класифікації позичальників залежно від різних категорій ризику, враховуючи їхню кредитну історію та інші фактори. SVM особливо ефективні у виявленні нелінійних залежностей між змінними.

Байєсівські мережі є імовірнісною графічною моделлю, що використовується для моделювання ймовірностей різних результатів на основі кредитної історії позичальника та інших факторів. Вони корисні у виявленні складних залежностей між змінними й допомагають кредиторам приймати обґрунтовані рішення.

Отже, існує декілька методів інтелектуального аналізу даних, які зазвичай використовуються для оцінки кредитоспроможності. Ці методи можуть допомогти кредиторам визначити найважливіші фактори, що

впливають на кредитоспроможність, і прийняти більш обґрунтовані кредитні рішення. Кожна методика має свої сильні та слабкі сторони, вибір методу буде залежати від схильності кредитора до ризику, типу кредиту та фінансового стану позичальника.

1.4 Аналіз існуючих систем оцінювання кредитоспроможності в Україні та за кордоном.

Системи оцінки кредитоспроможності є важливим аспектом банківської діяльності, оскільки вони дозволяють банкам визначати кредитоспроможність потенційних позичальників і мінімізувати ризик дефолту. Існують різні методи оцінки кредитоспроможності, які відрізняються залежно від країни та фінансової установи. За даними дослідження [13], одним із підходів є встановлення набору критеріїв оцінки кредитоспроможності позичальника та їх аналіз за допомогою дискримінантного аналізу, методів Хельвіга та кластерного аналізу. Дискримінантний аналіз - це статистичний метод, який використовує набір змінних для прогнозування класифікації категоріальної змінної. Метод Хельвіга - це статистичний підхід, який використовує набір змінних для прогнозування класифікації категоріальної змінної, а кластерний аналіз - це статистичний метод, який групує точки даних на основі їхньої схожості.

Інший підхід до оцінки кредитоспроможності базується на кредитній історії позичальника, і саме цей підхід використовується скоринговую системою, розробленою найбільшою базою даних бюро кредитних історій в Україні, як описано в [14]. Скорингова система використовує набір "предикторів" або факторів, які гіпотетично впливають на кредитну поведінку

позичальника, таких як наявність поточних і минулих прострочених платежів, кількість непогашених і погашених кредитів, їх суми і тривалість, а також частота запитів на отримання кредитної історії. Потім ці предиктори тестуються на великих обсягах даних для визначення фактичного впливу та значущості кожного предиктора, і на основі цих результатів будується скорингова модель. Результатом скорингової моделі є кредитний рейтинг - числовий показник, що оцінює ймовірність дефолту позичальника на основі даних його кредитної історії.

Однією з переваг використання скорингової системи на основі кредитної історії є те, що вона спрощує та прискорює процес прийняття кредитного рішення, а також забезпечує максимальну ефективність прогнозування ризиків. Вона також дозволяє в режимі реального часу отримувати інформацію про кількість запитів по суб'єкту, коментарі від суб'єкта, повідомлення про втрату паспорта, оскарження кредитного договору або активовану функцію FREEZE. Такий підхід також дозволяє прискорити розгляд кредитних заявок, підвищує ефективність бізнесу та лояльність клієнтів, дозволяє своєчасно виявляти позичальників з високою ймовірністю дефолту, що допомагає мінімізувати портфель проблемних кредитів. Однак недоліком цього підходу є те, що він може неточно прогнозувати кредитоспроможність позичальників, які мають обмежену кредитну історію або взагалі не мають її.

За даними статті [15], існують різні методологічні підходи до визначення сутності кредитного рейтингу банку, його аналізу та оцінки. Сучасні рейтингові оцінки мають бути адаптовані для українських банків з метою забезпечення їх релевантності та ефективності. Одним із підходів до аналізу кредитного рейтингу є використання адаптованої версії моделі Z-score Альтмана, яка являє собою формулу, що використовує комбінацію фінансових коефіцієнтів для прогнозування ймовірності банкрутства. Ця модель успішно

використовується в багатьох країнах, в тому числі і в Україні, для оцінки кредитоспроможності компаній і банків.

Інший підхід до оцінки кредитоспроможності полягає у використанні алгоритмів машинного навчання для аналізу даних про позичальника та прогнозування його кредитоспроможності. За даними дослідження [18], алгоритми машинного навчання можуть допомогти підвищити точність кредитних оцінок, аналізуючи великі масиви даних і виявляючи закономірності, які традиційні методи оцінки кредитоспроможності можуть пропустити. Наприклад, алгоритми машинного навчання можуть виявити кореляцію між активністю позичальника в соціальних мережах та його кредитоспроможністю. Однак недоліком цього підходу є те, що для навчання алгоритму потрібна велика кількість даних, які можуть бути доступні не для всіх позичальників.

За даними [14], найбільша база даних бюро кредитних історій в Україні використовує набір "предикторів" або факторів, які гіпотетично впливають на кредитну поведінку позичальника. Ці предиктори включають наявність поточних і минулих прострочених платежів, кількість непогашених і погашених кредитів, їх суми і тривалість, а також частоту запитів на отримання кредитної історії. Потім ці предиктори тестуються на великих обсягах даних, щоб визначити фактичний вплив і значущість кожного з них. Результатом тестування є скорингова модель, яка присвоює числовий показник для оцінки ймовірності дефолту позичальника на основі даних його кредитної історії.

У статті [18] надається більш детальна інформація про ключові предиктори, що використовуються в скоринговій системі. Джерело стверджує, що значущими змінними в скоринговій системі є кредитний рейтинг, кількість кредитних карток, кредитна історія в минулому, сума кредиту, отримання боргових консультацій, тривалість кредиту, залишок на рахунку та мета кредиту. Ці змінні були протестовані за допомогою статистичного методу,

який називається дискримінантний аналіз, що використовує набір змінних для прогнозування класифікації категоріальної змінної.

В одному з досліджень було використано нечітку теорію нечітких множин (fuzzy rough-set theory) та нечітку кластеризацію за методом С-середніх (fuzzy C-means) для оцінки та дослідження складних взаємозв'язків між характеристиками фермерів, конкурентними факторами навколишнього середовища та рівнем кредитування фермерів. Методологія була використана для прогнозування кредитоспроможності фермерів та для допомоги у прийнятті рішень щодо надання сільськогосподарських кредитів. Точність прогнозування за допомогою бази правил склала 81,16%. Основний висновок полягає в тому, що характеристики, пов'язані з освітою та навичками, є важливими для визначення кредитоспроможності фермерів [19].

В іншому дослідженні порівнювалася точність логістичної регресії, нейронних мереж і дерев рішень на основі класифікаційно-регресійного дерева (CART, або Classification and Regression Trees) у кредитному скорингу для кредитування малого бізнесу. Результати показали, що ймовірнісна нейромережева модель досягла найкращої точності. Ще одне дослідження досліджувало методи інтелектуального аналізу даних у спробі знайти найточніші та найпрогнозованіші методи для визначення ймовірності дефолту. Вони виявили, що штучні нейронні мережі забезпечують найбільш точну оцінку ймовірності дефолту серед шести досліджуваних методів інтелектуального аналізу даних [20].

Важливо зазначити, що конкретний набір предикторів, які використовуються в скоринговій системі, може відрізнятися залежно від кредитного бюро та країни. Однак загальний підхід використання даних кредитної історії для оцінки кредитоспроможності та присвоєння числового балу широко застосовується в багатьох країнах, у тому числі в США, як описано в джерелі [21].

1.5 Висновки до розділу та постановка задачі.

Визначення кредитоспроможності позичальника - це складний процес, що вимагає використання різних методів та моделей, а також врахування багатьох факторів, які можуть вплинути на ризик невиплати кредиту. Відомості про кредитну історію та фінансовий стан позичальника є ключовими для оцінки його кредитоспроможності, але також важливо враховувати й інші фактори, такі як рівень доходів, зайнятість, вік та інші. Кредитори повинні мати чітку політику та процедури щодо оцінки кредитоспроможності, щоб забезпечити якість вхідних даних та запобігти упередженості. Вибір конкретного методу оцінювання кредитоспроможності залежить від типу кредиту, фінансового стану позичальника та схильності кредитора до ризику, і кожен метод має свої сильні та слабкі сторони.

Наступне дослідження має включати в себе такі етапи:

1. Розробка моделей передбачення кредитоспроможності позичальників на основі інтелектуального аналізу даних.
2. Аналіз різних методів інтелектуального аналізу даних, таких як машинне навчання.
3. Порівняння розроблених моделей передбачення та визначення найбільш ефективної моделі.
4. Оцінка точності та надійності розробленої системи оцінювання кредитоспроможності позичальників.
5. Проведення економічного аналізу системи оцінювання кредитоспроможності клієнтів фінансових установ.

Отже, основною метою дослідження є розробка системи передбачення кредитоспроможності позичальників з використанням методів

інтелектуального аналізу даних та визначення найбільш ефективної моделі, що може допомогти фінансовим установам приймати обґрунтовані рішення з надання кредитів.

2 АНАЛІЗ ВИКОРИСТАННЯ МЕТОДІВ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ В КРЕДИТУВАННІ

2.1 Загальний аналіз алгоритмів машинного навчання в банківському кредитуванні.

Існує кілька досліджень, присвячених банківському кредитуванню з використанням машинного навчання. Одне дослідження [22] описує використання історичних даних для прогнозування помилок банківських кредитів та управління кредитним ризиком за допомогою алгоритмів машинного навчання. Дослідження показало, що використання алгоритмів машинного навчання (ML, або Machine Learning) може забезпечити точний спосіб виявлення неплатників на основі даних про транзакції. Модель XGBoost-LSTM (eXtreme Gradient Boosting - Long Short-Term Memory), яка є комбінацією алгоритмів XGBoost (eXtreme Gradient Boosting) та довгострокової і короткострокової пам'яті, виявилася ефективною для прогнозування ризиків дефолту кредитних карток. У дослідженні використано понад 10 мільйонів записів Банку Тайваню для аналізу логістичної регресії, що відображає зв'язок між змінною класу та набором незалежних змінних.

В іншому дослідженні [23] запропонований метод застосовується для прогнозування кредитного ризику великої вибірки малих і середніх підприємств. У дослідженні використовується набір даних, що складається з офіційної фінансової інформації (балансові змінні) про 15 045 малих і середніх підприємств, переважно розташованих у Південній Європі, за 2015 рік. Також надається інформація про статус (0 = активний, 1 = дефолт) кожної компанії

через рік (2016). Частка компаній-банкрутів у цьому наборі даних становить 10,9%. У дослідженні використовується алгоритм дерева XGBoost та моделі мереж подібності для покращення прогностичних характеристик моделі.

Крім того, в іншому дослідженні [24] обговорюється аналітична платформа для обробки великих обсягів даних та впровадження різних алгоритмів машинного навчання для виявлення шахрайства в банківській сфері. У дослідженні спостерігається ефективність різних алгоритмів машинного навчання на еталонних наборах даних для виявлення шахрайства в режимі реального часу, що забезпечує низький ризик і високу задоволеність клієнтів. У дослідженні використовується інтерфейсний аналітичний фреймворк з Hadoop для ефективного зчитування даних і передачі їх на аналітичний сервер для прогнозування шахрайства.

2.2 Аналіз використання подібних моделей у кредитному скорингу.

Кредитний скоринг є важливим завданням для фінансових установ для прийняття обґрунтованих кредитних рішень. З розвитком технологій штучного інтелекту та великих даних процедури кредитного аналізу стали більш ефективними та продуктивними, зменшивши навантаження на статистиків і дозволивши підвищити рівень автоматизації процесу видачі кредитів.

В дослідженні [25] автори починають з огляду традиційних методів кредитного скорингу, таких як логістична регресія, дискримінантний аналіз та дерева рішень. Вони зазначають, що ці методи мають обмеження, наприклад, вони менш ефективні в оцінці кредитного ризику для позичальників з обмеженою кредитною історією. Вони також підкреслюють важливість

кредитного скорингу в процесі кредитування і зазначають, що точний кредитний скоринг може призвести до кращих кредитних рішень, підвищення прибутковості та зниження кредитного ризику.

Далі автори надають огляд методів машинного навчання, таких як машини опорних векторів, випадкові ліси та нейронні мережі, і як вони можуть бути застосовані в кредитному скорингу. Вони зазначають, що методи ML можуть бути використані для підвищення точності моделей кредитного скорингу, особливо для позичальників з обмеженою кредитною історією. Однак вони також звертають увагу на проблеми, пов'язані з впровадженням методів ML у кредитний скоринг, такі як потреба у великих обсягах даних, потенційна можливість перенастроювання моделі та недостатня інтерпретованість деяких ML-моделей.

Далі автори розглядають існуючу літературу щодо використання методів ML у кредитному скорингу, висвітлюючи різні підходи та методи, що застосовуються в різних дослідженнях. Вони зазначають, що більшість досліджень зосереджені на порівнянні традиційних методів кредитного скорингу з методами ML і що результати є неоднозначними: одні дослідження повідомляють про підвищення точності за допомогою методів ML, а інші - про відсутність значного покращення. Вони також підкреслюють важливість відбору ознак у моделях кредитного скорингу і зазначають, що методи ML можуть бути використані для визначення найбільш важливих ознак для кредитного скорингу.

В результаті тестування моделей частина з них показала результати близькі до 70% на наборі даних для валідації. Ансамблеві моделі на основі дерев рішень показали найкращі результати, близькі до 80%, що є підвищенням в точності порівняно із моделлю, розробленою в ході виконання даної роботи. Однак, слід зазначити, що набір даних, використаний у дослідженні був більший порівняно із тим, що використовувався у цій роботі та для збалансування даних були використані методи збільшення вибірки, що

спотворюють дані, або ж збільшують кількість повторюваних записів, що не може відображати в повній мірі стан справ у реальномі світі. Враховуючи те, що менші класи були збільшені щонайменше десятикратно, результати дослідження в джерелі слід сприймати під призмою скепсису та очікувати гірших результатів у роботі в реальному світі.

2.3 Оцінка потенційного економічного ефекту від використання системи.

Алгоритми машинного навчання можуть суттєво вплинути на кредитний скоринг у банківській сфері. Традиційні моделі, такі як логістична регресія та дерева рішень, все ще широко використовуються в кредитному скорингу завдяки своїй простоті та інтерпретованості, але сучасні алгоритми машинного навчання продемонстрували чудову прогностичну силу. Однак ці алгоритми часто називають "чорними скриньками" через їхню складність, що ускладнює розуміння та інтерпретацію моделей фахівцями з кредитних ризиків [26].

Доведено, що моделі на основі дерев дають найкращі результати в прогнозуванні дефолту клієнта, але їх інтерпретацію можна покращити, використовуючи методи, які дозволять їм бути більш доступними для фахівців з кредитних ризиків. Однією з таких методик є пояснюваний штучний інтелект (Explainable Artificial Intelligence, XAI), який дозволяє інтерпретувати складні моделі машинного навчання. Методи XAI спрямовані на надання чітких і зрозумілих пояснень того, як модель прийшла до своїх рішень. Це особливо важливо для моделей кредитного скорингу, оскільки менеджери повинні

ефективно управляти кредитним ризиком, а регулятори повинні бути впевнені в якості моделі [27].

Ансамблеві або агреговані методи - це ще один спосіб покращити прогностичну ефективність алгоритмів машинного навчання. Ці методи об'єднують прогнози багатьох варіантів даного алгоритму за допомогою лінійної комбінації, наприклад, усереднення або більшості голосів, замість того, щоб покладатися на один прогноз [28]. Поєднуючи сильні сторони різних моделей, ансамблеві методи можуть підвищити точність і стабільність моделей кредитного скорингу [29].

Хоча алгоритми машинного навчання можуть підвищити точність і прогностичну силу моделей кредитного скорингу, є також потенційні недоліки, які слід враховувати. Однією з проблем є потенційна можливість упередженості даних, що використовуються для навчання моделей. Якщо дані, що використовуються для навчання алгоритму, є упередженими, модель також буде упередженою, що потенційно може призвести до дискримінаційних практик. Ще одне занепокоєння викликає характер "чорної скриньки" деяких алгоритмів машинного навчання, що ускладнює розуміння того, як модель прийшла до своїх рішень. Така непрозорість може бути проблематичною для регуляторів, які повинні забезпечити справедливість і точність моделей.

Незважаючи на ці занепокоєння, алгоритми машинного навчання продемонстрували багатообіцяючі можливості для покращення моделей кредитного скорингу. Використовуючи методи XAI та ансамблеві методи, фахівці-практики можуть покращити інтерпретованість та прогностичну силу моделей машинного навчання. Однак важливо ретельно розглянути потенційні упередження і відсутність прозорості, пов'язані з цими моделями, і вжити заходів для пом'якшення цих проблем.

В майбутньому використання моделей для скорингу буде потребувати прозорості та змоги пояснити їх роботу, щоб не викликати етичних та юридичних питань зі сторони клієнтів банків. Використання подібних моделей

може значним чином покращити оцінку надійності клієнтів та знизити ризики для банків.

У випадку дефолту кредитора банк може втрачати до 100% зайнятої суми, але такі випадки трапляються рідше. В даній роботі дослідження відбувалось на даних із доволі малими сумами кредитів, отже і втрати від неповернення були б не такими значними. В оригінальному наборі даних кількість випадків неповернення сягає 25%. Моделі, розроблені в даній роботі можуть зменшити кількість випадків неповернення до 14-24% від початкового числа, що становитиме від 4% до 6.5% клієнтів, що не змогли повернути кредит, замість 25%. Також слід зазначити, що використання даних моделей буде зменшувати кількість клієнтів, яким було одобрено кредит. Так, для отримання кількості клієнтів, що не повернули кредит у 4% доведеться відхилити близько 45% надійних клієнтів, а для отримання результату у 6,5% неповернених кредитів доведеться відхилити близько 25% надійних клієнтів.

Отже, під час вибору моделі для оцінки кредитного скорингу слід оцінити суму втрати від одного клієнта, суму надходження від одного клієнта та ціну отримання одного клієнта. Ці показники є ключовими для вибору моделі із запропонованих в цій роботі, оскільки вони разом із даними про результати роботи моделей дозволяють максимізувати прибутки банків.

2.4 Вибір моделей для кредитного скорингу.

Для створення системи оцінювання кредитоспроможності позичальників буде використано декілька моделей машинного навчання, що включають в себе логістичну регресію, лінійний дискримінантний аналіз, метод K-найближчих сусідів, дерево рішень, наївний Байєсівський класифікатор, метод опорних векторів, випадковий ліс, градієнтний бустинг та адаптивний бустинг.

Вибір цих моделей обґрунтовується їх високою точністю та ефективністю в розв'язанні подібних завдань, що підтверджено результатами проведених наукових досліджень та відгуками відповідних фахівців.

В подальшому, в ході роботи буде проведений детальний аналіз та порівняння обраних моделей, їх реалізація та тестування на відповідних даних. Також будуть враховані можливі обмеження та переваги кожної з моделей, щоб забезпечити максимально ефективне та точне вирішення поставленої задачі із передбачення.

Під час розробки скорингової моделі слід завжди розглядати якомога більше алгоритмів щоб збільшити точність. Для цієї роботи ми будемо використовувати 9 різних моделей класифікації та оцінимо роботу кожної з них в майбутньому для вибору найбільш успішної.

2.4.1 Логістична регресія.

LogisticRegression (Логістична Регресія) - це модель, яка оцінює ймовірність належності даних до певного класу, використовуючи логістичну функцію (формула (2.1)).

$$p(x) = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}} \quad (2.1),$$

де $\beta_0 = -\mu/s$ і називається перехопленням (це вертикальне перехоплення або y-перехоплення прямої y = $\beta_0 + \beta_1 x$), та $\beta_1 = 1/s$ (параметр оберненого масштабу або параметр швидкості): це інтервал у та нахил логарифма як функції від x. І навпаки, $\mu = \beta_0/\beta_1$ та $s = 1/\beta_1$.

Приклад роботи логістичної регресії наведено на рисунку 2.1:

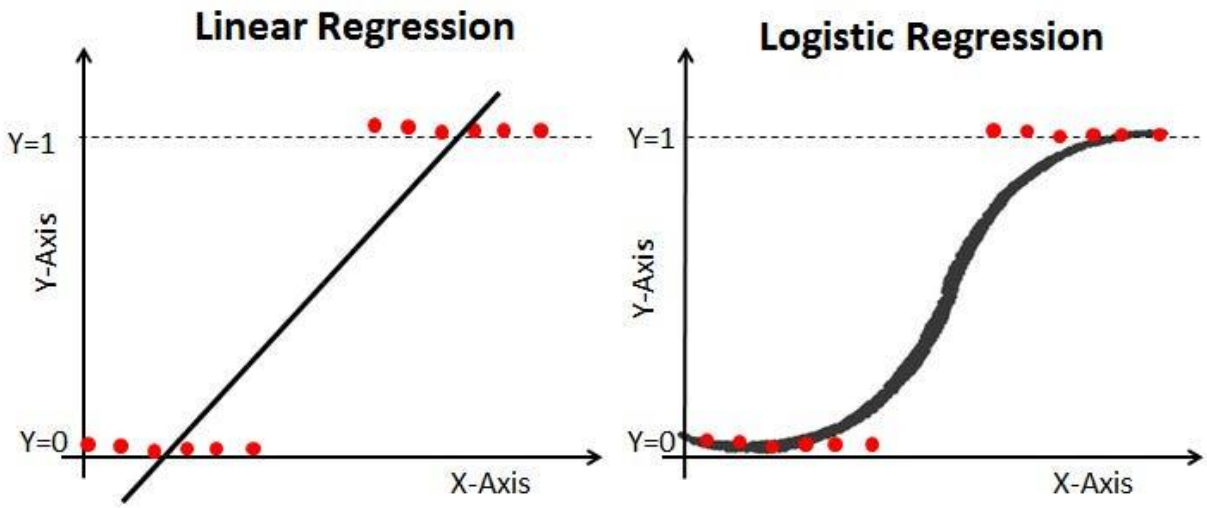


Рисунок 2.1 - Результати роботи логістисної регресії

2.4.2 Лінійний дискримінантний аналіз.

LinearDiscriminantAnalysis (LDA) - це метод, який використовується для знаходження лінійної комбінації ознак, яка найкраще розділяє класи. Основна ідея полягає у тому, щоб знайти напрямок, в якому відстань між середніми значеннями класів буде максимальною.

Ось пояснення роботи LDA з відповідними формулами:

1. LDA обчислює дві матриці розсіювання на основі вхідних даних:
    a. Матриця розсіювання всередині класу (S_w) - вимірює розкид даних у межах кожного класу і обчислюється як сума коваріаційних матриць для кожного класу.
    b. Матриця міжкласового розсіювання (S_b) - кількісно вимірює поділ між класами і обчислюється як сума зовнішнього добутку різниці між середніми значеннями класів та загальним середнім значенням.
2. Обчислення середнього значення:
    a. Загальне середнє (m) - середній вектор усіх точок даних у всіх класах.
    b. Середні для класів (m_i) - середні вектори для кожного класу.
3. Максимізація критерію Фішера - LDA має на меті максимізувати критерій Фішера, який визначається як відношення міжкласового розсіювання до внутрішньокласового розсіювання (формула (2.2)).

$$J(W) = (W^T * S_b * W) / (W^T * S_b * W) \quad (2.2),$$

де W являє собою проекційну матрицю, яка перетворює вхідні дані в простір меншої розмірності.

4. Знаходження проекційної матриці - матриця W знаходиться шляхом розв'язання узагальненої задачі на власні значення (формула (2.3)).

$$S_w^{-1} * S_w * w = \lambda * w \quad (2.3),$$

lде w - власний вектор,

а λ - відповідне власне значення.

5. Вибір верхніх власних векторів - власні вектори, що відповідають найбільшим власним значенням, обираються як напрямки проекцій, що містять найбільше дискримінативної інформації.
6. Перетворення даних - вхідні дані проектуються на вибрані власні вектори для отримання представлення в нижній розмірності.
7. Класифікація - після перетворення даних до зменшеного простору даних можна застосувати алгоритм класифікації, наприклад, логістичну регресію або класифікатор найближчого сусіда, для вирішення завдань класифікації.

Максимізуючи критерій Фішера, LDA знаходить проекцію, яка оптимально розділяє класи, що робить його ефективним для зменшення розмірності та класифікації, особливо у випадках, коли класи добре розділені.

Наведене вище пояснення стосується бінарної задачі класифікації, але LDA може бути розширений для роботи з багатокласовими задачами.

2.4.3 K-найближчих сусідів.

KNeighborsClassifier (K-найближчих сусідів) - це модель, яка використовує метод найближчих сусідів для класифікації даних. Ідея полягає в тому, щоб знайти K найближчих точок до точки, яку потрібно класифікувати, та визначити, які класи зустрічаються найчастіше серед цих сусідів.

Приклад роботи алгоритму наведено на рисунку 2.2.:

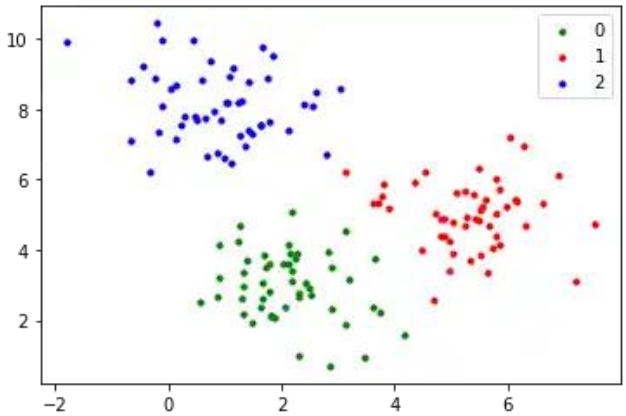


Рисунок 2.2 - Демонстрація роботи KNN Classifier

Для метрики відстані в цьому алгоритмі використовувалась дистанція Міхновського. Ця міра відстані є узагальненою формою евклідової та манхеттенської метрик відстані. Параметр p у формулі нижче дозволяє створювати інші метрики відстані. Евклідова відстань представлена цією формулою, коли p дорівнює двом, а Манхеттенська відстань позначається з p, що дорівнює одиниці. Формула відстані наведена нижче (формула (2.4)):

$$Minkowski\ Distance\ =\ (\sum_{i=1}^{n}\quad |x_i - y_i|)^{\frac{1}{p}} \quad (2.4).$$

2.4.4 Дерева рішень.

Decision Tree Classifier (CART) - модель, яка будує дерево прийняття рішень з урахуванням різних параметрів, щоб розділити дані на більш гомогенні групи. Кожна вершина дерева представляє певний вибірковий критерій, який використовується для розділення даних на дві частини, а наступні рівні дерева продовжують цей процес рекурсивно до тих пір, поки всі дані не будуть розділені на окремі категорії.

Візуалізація роботи алгоритму наведена на рисунку 2.3:

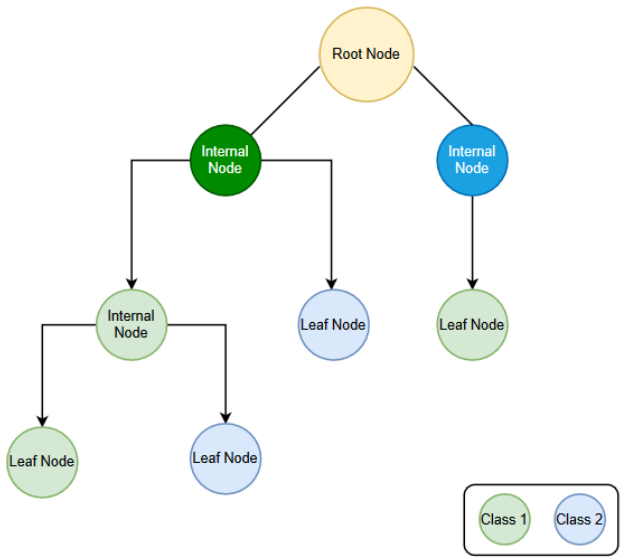


Рисунок 2.3 - Візуалізація роботи алгоритму дерева рішень

Як видно з рисунку 2.3, дерево рішень певним чином нагадує бінарне дерево пошуку, але в даному випадку кожна головна нода (Internal Node) означає певну умову, а кінцева нода (Leaf Node) відповідає певному класу.

2.4.5 Наївний байєсівський класифікатор.

Naive Bayes (NB) - модель, яка використовує теорему Байєса для підрахунку ймовірності належності даних до певного класу на основі вхідних факторів. Вона припускає, що вхідні фактори незалежні між собою, що може бути неправдивим у реальних даних, але модель зазвичай працює добре на практиці.

Проста форма розрахунку за теоремою Байеса виглядає наступним чином (формула (2.5)):

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \text{ (2.5)},$$

де ймовірність, яка нас цікавить при обчисленні P(A|B), називається апостеріорною ймовірністю. P(B) та P(A) це ймовірність виникнення події B та A відповідно, а P(B|A) є ймовірністю B за умови істинності A.

2.4.6 Метод опорних векторів.

Support Vector Machine (SVM) - модель, яка використовує геометричний підхід для розділення даних на різні класи. Вона будує гіперплощину в просторі високої розмірності, яка розділяє дані на класи, що наближається до оптимальної гіперплощини якомога більше. Якщо дані не розділяються лінійно, то використовуються ядерні функції, щоб збільшити просторову розмірність та розділити дані. Додатково, SVM може

розраховувати ймовірність належності даних до певного класу за допомогою параметру probability=True.

Приклад роботи алгоритму наведено на рисунку 2.4:

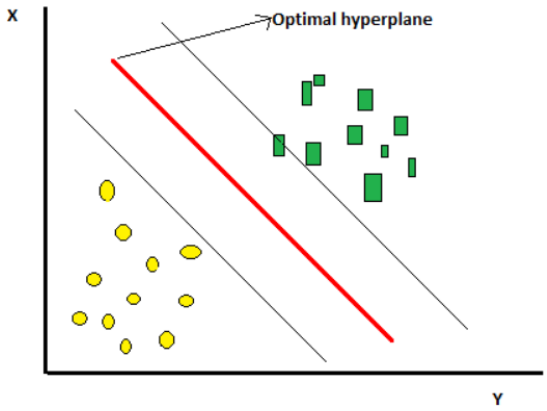


Рисунок 2.4 - Результати роботи SVM

Як видно з рисунку 2.4, модель SVM проводить класифікацію на основі відстані до гіперплощини. В данову випадку використовується лінійний підхід.

2.4.7 Градієнтний бустинг.

GradientBoostingClassifier (Класифікатор Градієнтного Бустингу) - модель, яка базується на створенні послідовності дерев рішень, при цьому кожне наступне дерево коригує помилки попереднього дерева. Таким чином, модель намагається покращити результати попереднього дерева на кожному кроці. Кожне дерево побудоване на основі випадкової підмножини даних та випадкової підмножини ознак, що допомагає уникнути перенавчання.

Візуалізацію роботи алгоритму наведено на рисунку 2.5:

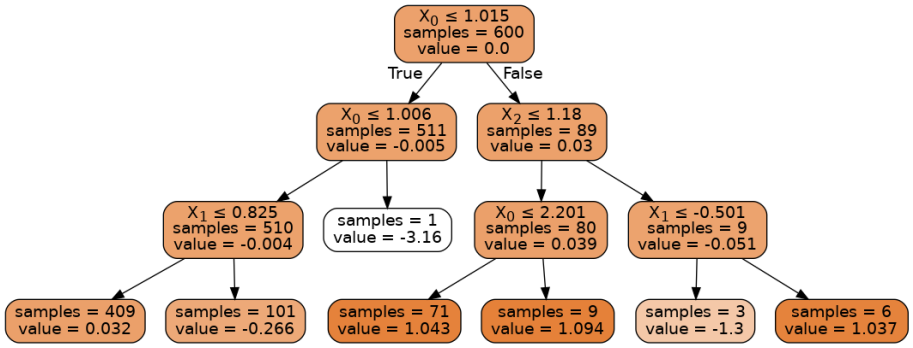


Рисунок 2.5 - Візуалізація роботи GradientBoostingClassifier

Як видно з рисунку 2.5, зображена структура індивідуальних дерев рішень, які використовуються в алгоритмі градієнтного бустингу. Кожен вузол (нода) представляє розгалуження або листок дерева і має певні властивості.

X0, X1, X2 та інші представляють вхідні ознаки або змінні, які використовуються для прийняття рішень у даному вузлі. Кожен вузол обробляє значення певної ознаки і визначає, яким шляхом продовжувати подальше розгалуження дерева.

Samples вказує кількість прикладів, які проходять через даний вузол. Кожен вузол має свою власну кількість прикладів, які досягли його, і це може змінюватися по мірі спуску по дереву. Приклади представляють собою рядки у навчальному наборі даних, де кожен рядок містить значення ознак і відповідну мітку або значення цільової змінної.

У градієнтному бустингу, коли будується дерево рішень, приклади розділяються на різні вузли дерева залежно від значень їхніх ознак. Кожен вузол має свою власну кількість прикладів, які досягли його, що відображає розподіл даних у вузлах дерева. Кількість прикладів у вузлі може змінюватися по мірі спуску по дереву, оскільки приклади розгалужуються на різні гілки дерева залежно від умов розгалуження.

Value представляє прогнозовану величину (або клас) для прикладів, які досягли даного вузла. В листках дерева (коли не можливе подальше

розгалуження) значення Value вказує на прогноз для цих прикладів. В нашому випадку модель регерсивна, тому значення не приймають якісь класи.

2.4.8 Адаптивний бустинг.

AdaBoostClassifier (Класифікатор Адаптивного Бустингу) - модель, яка також працює з послідовністю дерев рішень, але замість того, щоб коригувати помилки попереднього дерева, вона звертає увагу на ті зразки, які були неправильно класифіковані на попередньому кроці. Таким чином, модель "навчається" на тих зразках, де вона робить помилку, щоб покращити результати наступних кроків. Кожне дерево побудоване на основі зменшеного набору даних, які стають важчими для класифікації з кожним новим деревом.

Візуалізацію роботи алгоритму наведено на рисунку 2.6:

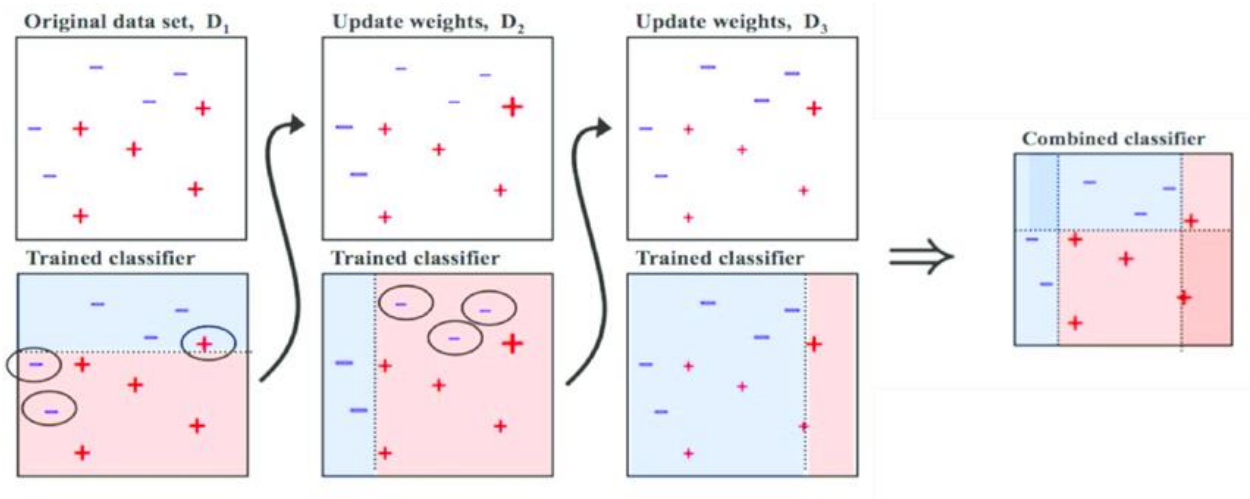


Рисунок 2.6 - Результати роботи класифікатора AdaBoost

Як видно з рисунку 2.6РИС. 3.18, модель ADABoost оновлює свої ваги таким чином, щоб максимально точно виявляти всі ознаки даних,

використовуючи комбінацію декількох більш слабких моделей, що були навчені на даних, що були невірно класифіковані.

2.5 Висновки до розділу.

У цьому розділі було проведено аналіз використання методів інтелектуального аналізу в кредитуванні. Були розглянуті переваги застосування алгоритмів машинного навчання в банківському кредитуванні. Виявлено, що ці алгоритми дозволяють автоматизувати процес прийняття рішень щодо кредитування та забезпечують більш точні результати порівняно з традиційними методами.

Досліджено використання подібних моделей у кредитному скорингу. Встановлено, що ці моделі використовуються для прогнозування кредитного ризику шляхом аналізу великої кількості даних. Вони дозволяють банкам оцінювати ймовірність повернення кредиту клієнтами і зменшують ризик недоречного призначення кредиту.

Оцінено потенційний економічний ефект від використання системи інтелектуального аналізу в кредитуванні. Виявлено, що така система може призвести до зниження витрат на внутрішні операції, поліпшення точності прийняття рішень та збільшення прибутковості кредитного портфеля.

Також був проведений вибір моделей для кредитного скорингу, були проаналізовані різні алгоритми, такі як логістична регресія, лінійний дискримінантний аналіз, к-найближчих сусідів, дерева рішень, наївний байєсівський класифікатор, метод опорних векторів, градієнтний бустинг та адаптивний бустинг. Всі моделі будуть використані в наступному розділі для тестування та визначення найбільш оптимального варіанта.

3 ПОБУДОВА СИСТЕМИ ОЦІНЮВАННЯ КРЕДИТОСПРОМОЖНОСТІ З ВИКОРИСТАННЯМ МЕТОДІВ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ

3.1 Створення комп'ютерної системи для аналізу кредитоспроможності фізичних осіб.

Для розробки системи буде використовуватись популярна мова програмування Python та деякі бібліотеки із машинного навчання для неї. Python - це популярна мова програмування, яка часто використовується для розробки моделей машинного навчання. Вона має велику кількість бібліотек та інструментів, що дозволяють зробити розробку моделей більш ефективною.

Pandas - це бібліотека Python, яка надає зручний та ефективний інтерфейс для роботи з даними. Pandas дозволяє зчитувати, зберігати та обробляти дані в різних форматах, включаючи CSV, Excel, SQL та багато інших.

Scikit-learn (sklearn) - це одна з найпопулярніших бібліотек Python для машинного навчання. Scikit-learn містить багато алгоритмів машинного навчання та інструментів для побудови, навчання та оцінки моделей.

Imbalanced-learn (imblearn) - це бібліотека Python для роботи з незбалансованими даними, що часто зустрічаються в задачах машинного навчання. Imbalanced-learn містить різні методи для обробки незбалансованих даних, включаючи збільшення/зменшення вибірки, генерацію синтетичних даних та інші.

Також для розробки була використана платформа Kaggle, що має можливість інтерактивно виконувати код в середовищі розробки

JypiterNotebook та має велику кількість готових датасетів для задач машинного навчання.

Основними функціями системи має бути оцінка кредитоспроможності позичальників на основі даних про них та присвоєння статусу “надійний” або “ненадійний” для кожної окремої особи.

3.2 Створення бази даних та її обробка.

3.2.1 Опис обраної бази даних.

Даний розділ містить детальний опис бази даних, яку ми обрали для нашого дослідження. В цьому розділі ми розглянемо основні характеристики обраної бази даних, її структуру та особливості використання.

Обрана база даних для тренування моделі містить дані із українських банків. Опис колонок наведено нижче на рисунку 3.1:

	Назва колонки	Опис даних
0	SROK	Термін погашення кредиту
1	PLATEG	Сума платежу
2	POL	Стать (чоловік/жінка)
3	VOZRAST	Вік клієнта
4	OBL_REGISTRACII	Область реєстрації клієнта
5	OBL_PROGIVANIA	Область проживання клієнта
6	CREDIT	Сума кредиту
7	TIP_PROGIV	Тип проживання клієнта
8	SROK_PROGIV	Термін проживання на поточному місці
9	OBRAZOVANIE	Рівень освіти
10	SEM_POL	Сімейний стан клієнта
11	KOL_DETEI	Кількість дітей
12	SOBSTVENNOST	Тип нерухомості, яку володіє клієнт
13	DOLGNOST	Посада клієнта
14	SROK_RABOTI	Стаж роботи
15	KOL_RABOTNIKOV	Кількість співробітників компанії
16	DOHOD	Місячний дохід клієнта
17	NEVOZVRAT_CREDITA	Чи була невдача у погашенні попереднього кредиту

Рисунок 3.1 - Опис всіх атрибутів набору даних

Обрана база даних має 18 атрибутів: 11 категорійних та 7 числових, із яких 1 цільовий (Рис 3.2).

```
SROK                 int64
PLATEG             float64
POL                 object
VOZRAST              int64
OBL_REGISTRACII     object
OBL_PROGIVANIA      object
CREDIT               int64
TIP_PROGIV          object
SROK_PROGIV         object
OBRAZOVANIE         object
SEM_POL             object
KOL_DETEI            int64
SOBSTVENNOST        object
DOLGNOST            object
SROK_RABOTI         object
KOL_RABOTNIKOV      object
DOHOD              float64
NEVOZVRAT_CREDITA    int64
dtype: object
```

Рисунок 3.2 - Розподіл типів даних по атрибутах бази даних

База даних не містить пропущених значень, тому в ході обробки набору не доведеться мати справу із цією проблемою (Рис. 3.3).

SROK	0
PLATEG	0
POL	0
VOZRAST	0
OBL_REGISTRACII	0
OBL_PROGIVANIA	0
CREDIT	0
TIP_PROGIV	0
SROK_PROGIV	0
OBRAZOVANIE	0
SEM_POL	0
KOL_DETEI	0
SOBSTVENNOST	0
DOLGNOST	0
SROK_RABOTI	0
KOL_RABOTNIKOV	0
DOHOD	0
NEVOZVRAT_CREDITA	0

Рисунок 3.3 - Кількість пропущених даних в кожному атрибуті бази даних

3.2.2 Аналіз наявних даних.

У цьому розділі ми зосередимося на вивченні даних, що маємо в нашому наборі, з метою зрозуміти, які інформаційну значущість мають різні атрибути і як вони можуть вплинути на результати нашого дослідження. Ми розглянемо кількісні та категоріальні змінні в наборі даних, проведемо

візуальний аналіз даних за допомогою графіків, щоб зрозуміти, як розподіл в кожному атрибуті та чи присутні аномальні значення в даних.

Для розподілу даних по категоріях ми розділимо всі атрибути на ті, що мають більше 10 унікальних значень та інші, і виведемо barplot для кожного атрибуту, що задовольняє вимогу (Рис. 3.4).

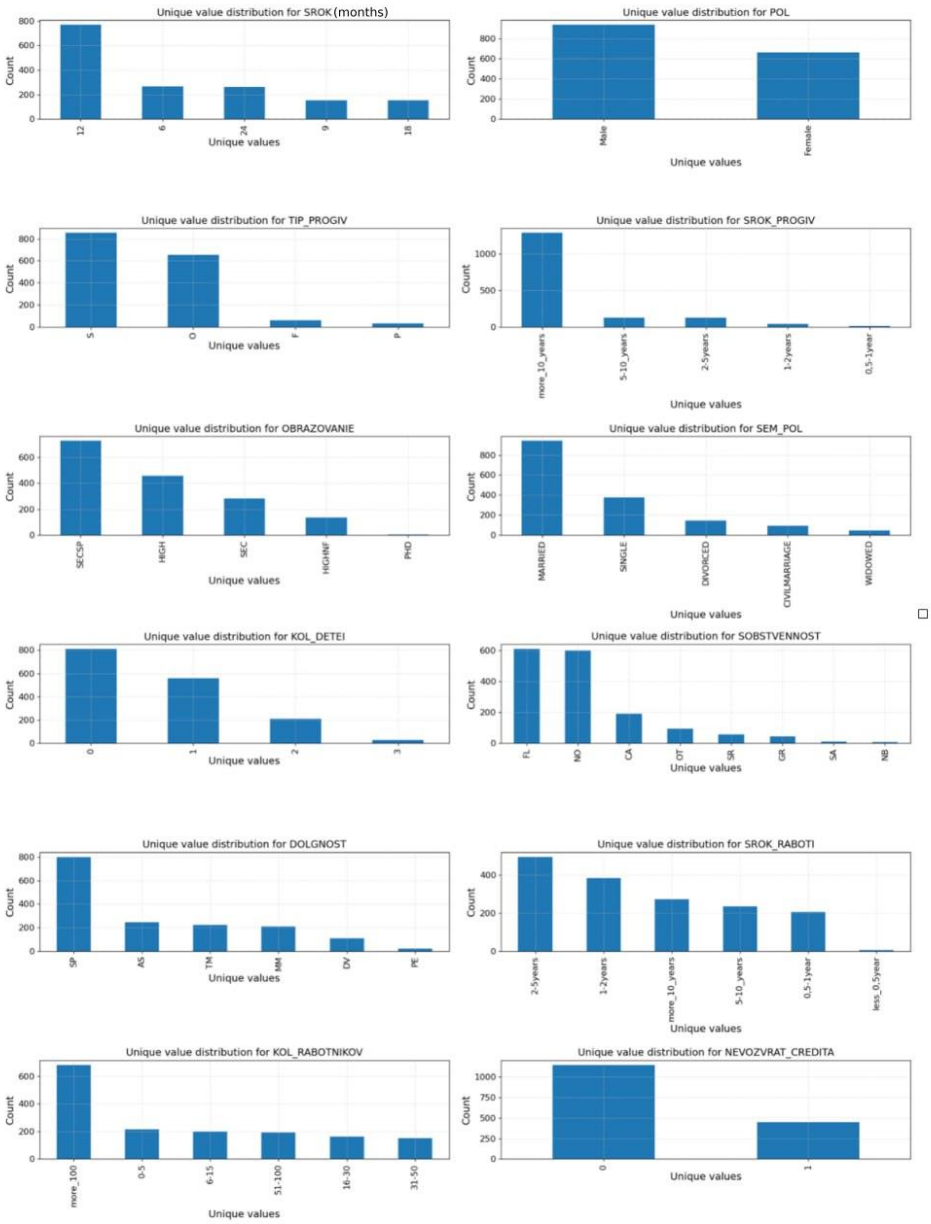


Рисунок 3.4 - Розподіл даних по категоріях серед атрибутів

Як видно з рисунку наведеного вище, більшість клієнтів має кредит на 1 рік. Серед клієнтів переважають жінки, майже всі клієнти живуть на поточному місці проживання більше 10 років. Також більшість клієнтів є одруженими. Майже відсутні клієнти які мають більше 1 дитини. Найбільш частими є випадки коли клієнт працює на поточній роботі від 2 до 5 років. В більшості випадків кількість працівників у компанії більше 100 осіб, інші ж випадки розподілені майже однаково.

Із останнього графіку на рисунку 3.4 можна побачити, що кількість клієнтів, що повернули кредит та кількість клієнтів, що не повернули, є різними. Такий розподіл може спричинити проблеми в навчанні моделі в майбутньому, оскільки модель буде більш схильною до передбачення в сторону більшого класу.

Далі ми проведемо аналіз розподілу всіх атрибутів, що мають кількість унікальних значень більше 10, окрім областей проживання (Рис. 3.5).

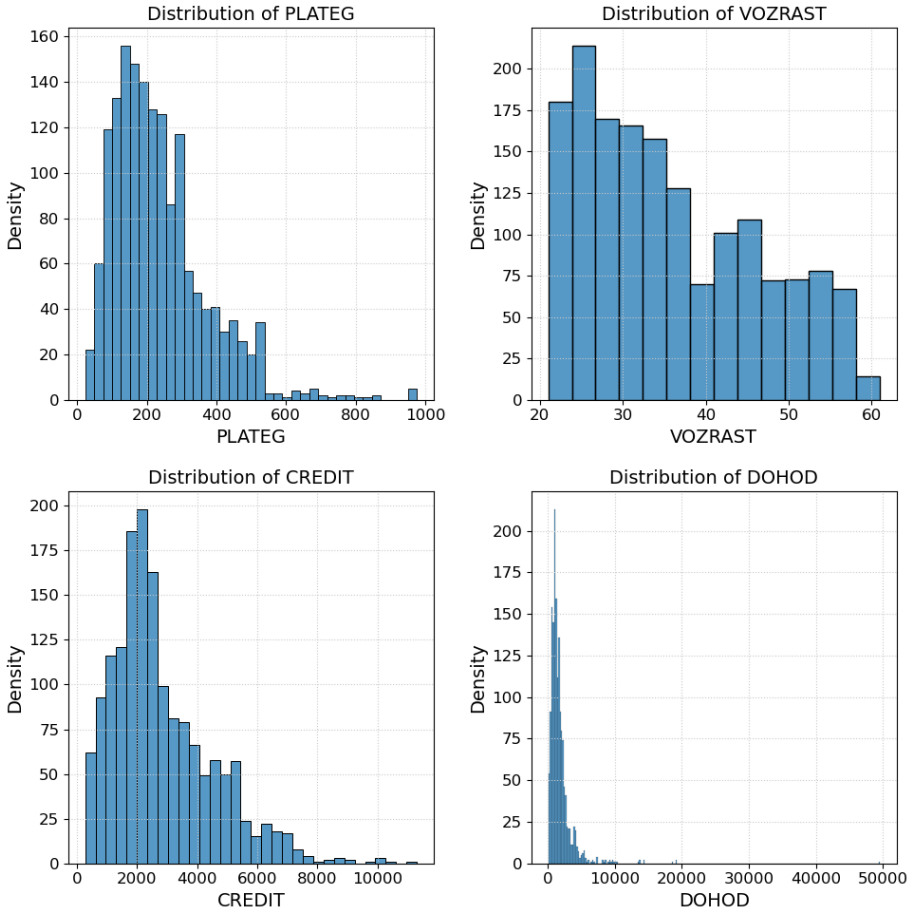


Рисунок 3.5 - Розподіл числових атрибутів, що мають більше 10 унікальних значень

Як видно з рисунку наведеного вище, ми маємо зміщені вліво графіки для атрибутів, що відповідають за розмір кредиту, розмір місячного платежу, вік клієнта та його доходи. Маємо велику кількість викидів в атрибуті, що відповідає за доходи клієнтів, що може стати проблемою в майбутньому для моделі класифікації.

Далі ми проведемо аналіз розподілу даних про реєстрацію та місце проживання клієнтів (Рис. 3.6).

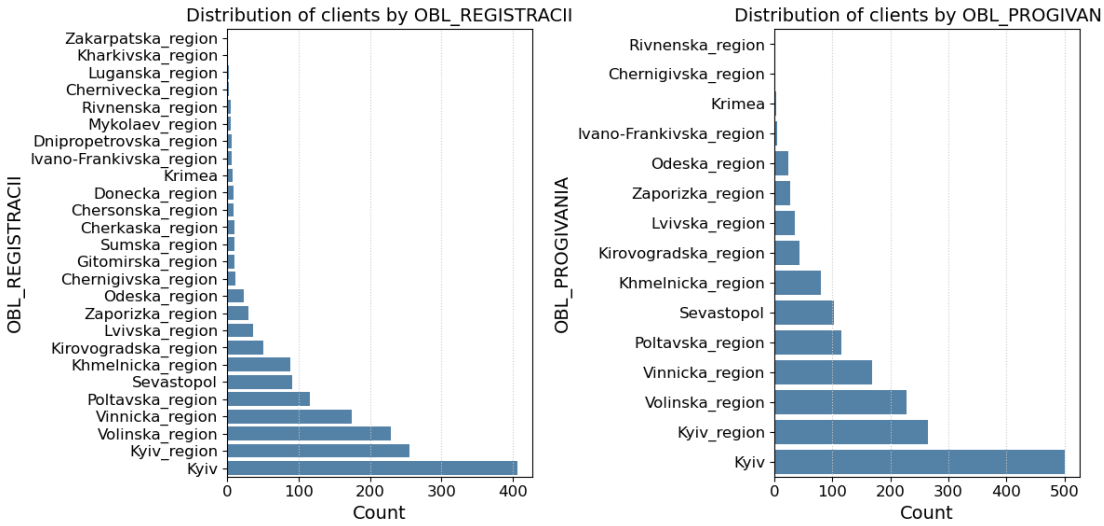


Рисунок 3.6 - Розподіл клієнтів за областями реєстрації та проживання

Як видно із наведеного рисунку вище, більшість клієнтів живуть в Києві та київській області. Проте місце реєстрації має набагато більшу кількість категорій. Це свідчить про те, що відділення банків присутні тільки в деяких областях. Дані про реєстрацію та проживання клієнтів можуть надати нам інформацію про те чи є приїзжим клієнт чи ні. Також маємо певну кількість областей, що має незначну кількість клієнтів та може збивати модель під час тренування, тому ми виключимо їх із набору.

Для більшої наглядності дані про клієнтів за областями можна вивести на карту та візуалізувати використовуючи теплову карту (Рис. 3.7-3.8).

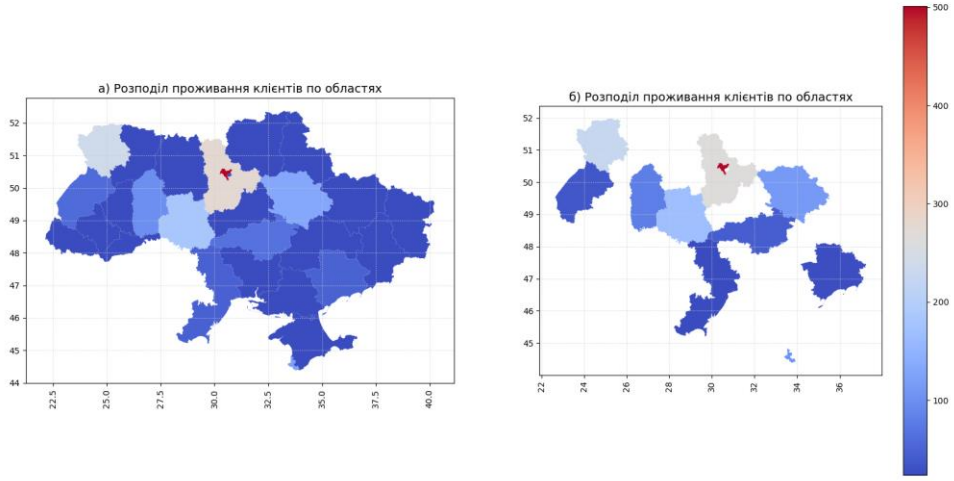


Рисунок 3.7 - Розподіл проживання клієнтів по областях: а) із областями, які відсутні б) без відсутніх областей

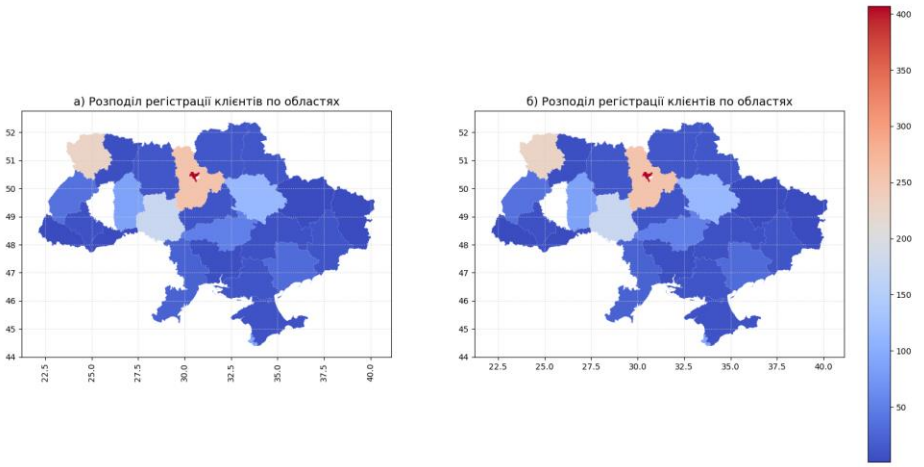


Рисунок 3.8 - Розподіл реєстрації клієнтів по областях: а) із областями, які відсутні б) без відсутніх областей

Як видно із першого рисунку, робота банку є доволі обмеженою за областями. Другий рисунок демострує те, що місця реєстрацій клієнтів розповсюджені по всій країні, а не тільки по областях роботи банків. Також ми бачимо, що є області, які мають велику кількість клієнтів за місцем реєстрації,

проте філіали банків там не присутні. Прикладом такого випадку є Тернопільська область, де кількість клієнтів із місцем реєстрації в цій області є відносно великою, враховуючи те, що банк там не має філіалу.

3.2.3 Аналіз розподілу класів та відношень між даними.

У цьому розділі ми проведемо аналіз доступних даних на розподіл між класами та проведемо більш комплексний аналіз співвідношень між ними. Розгляд взаємозв'язку між даними та їхнім розподілом є важливим кроком на етапі аналізу даних. Цей метод допомагає зрозуміти структуру та залежності між даними, виявити можливі аномалії та визначити оптимальний підхід для подальшої обробки даних.

Спершу ми проведемо аналіз розподілу всіх категорійних даних за атрибутом статі (Рис. 3.9).

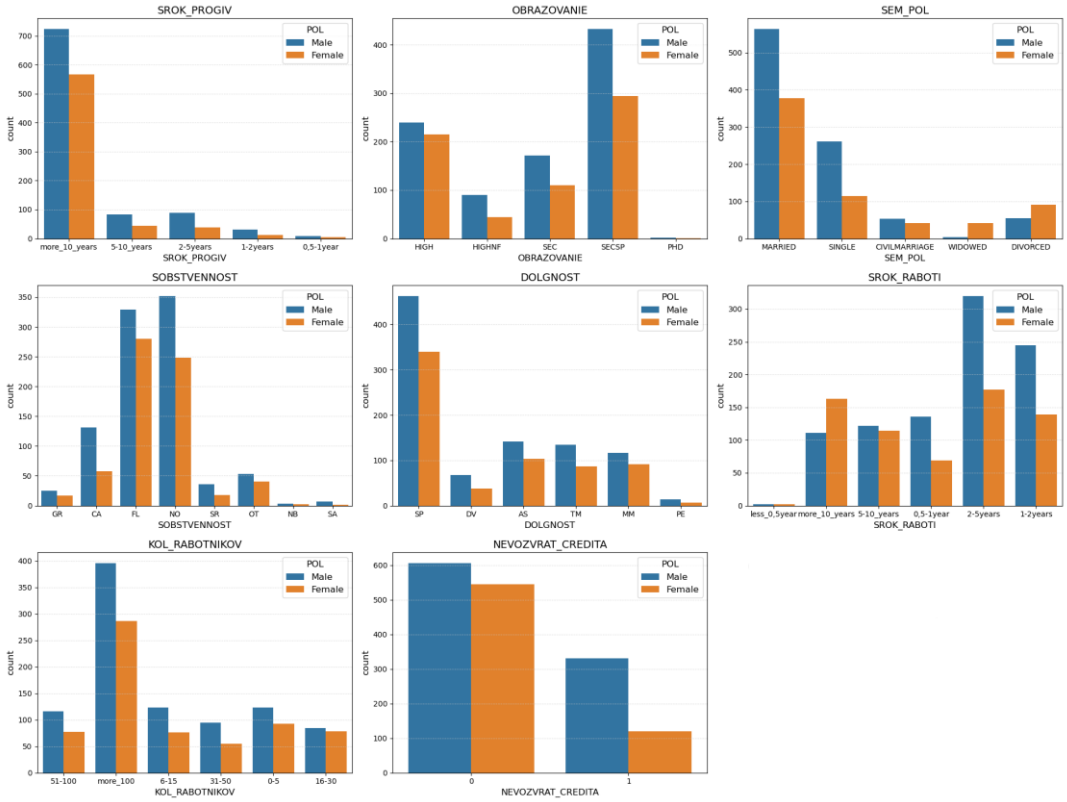


Рисунок 3.9 - Розподіл категорійних даних за статтю клієнта

Як видно із наведеного рисунку, відсоток чоловіків самотніх клієнтів більше, ніж серед одружених. Серед працівників, що на роботі більше 10 років більша частина це жінки, а серед людей, що не повернули кредит відсоток чоловіків значно більше, ніж серед людей, що кредит повернули.

Далі ми проведемо аналіз аналогічний попередньому, але на основі цільового атрибуту ‘NEVOZVRAT_CREDITA’ (Рис. 3.10).

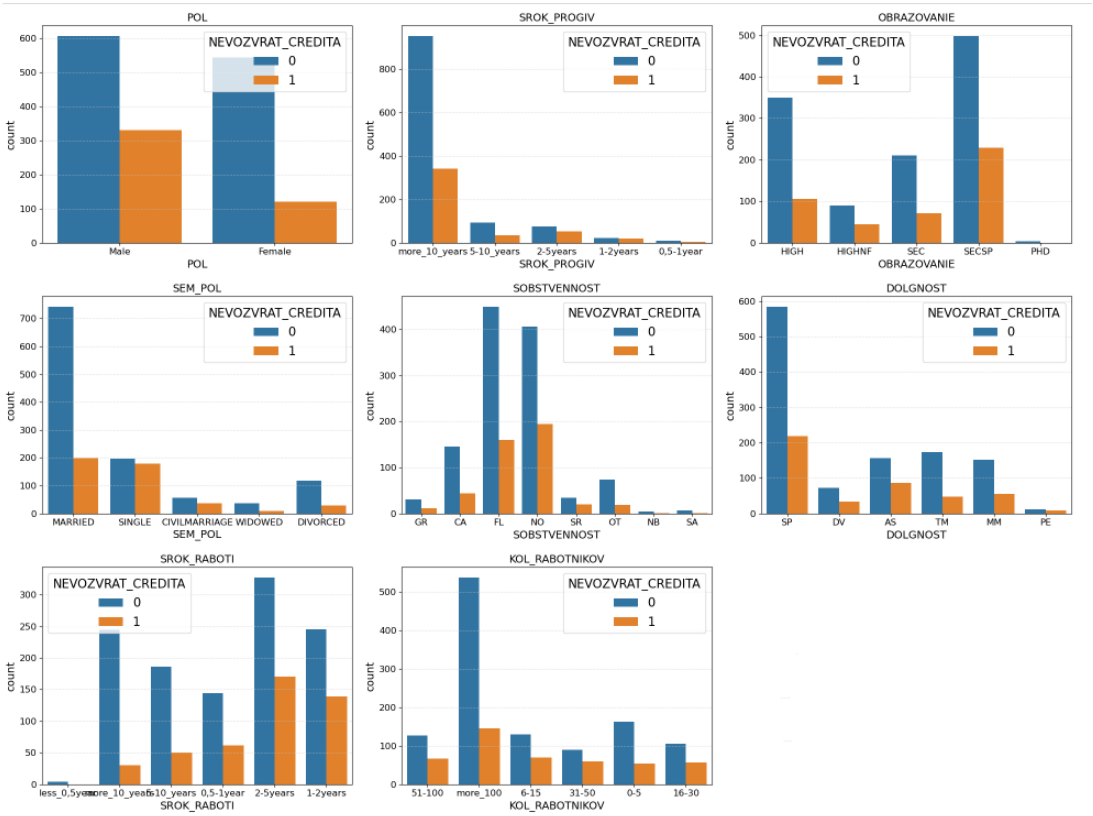


Рисунок 3.10 - Розподіл категорійних даних за поверненням та неповерненням кредита

Як видно із наведеного рисунку, серед людей, що живуть на поточному місці більше 10 років вірогідність неповернення кредиту найменша, серед людей що на поточному місці 1-2 роки вірогідність неповернення дорівнює вірогідності повернення, що робить цю категорію занадто ризикованою. Також прослідковується тенденція до збільшення вірогідності неповернення кредиту зі зменшенням строку роботи на поточній позиції.

Далі ми проведемо аналіз всіх областей за середніми числовими значеннями для поточної області (Рис 3.11-3.17).

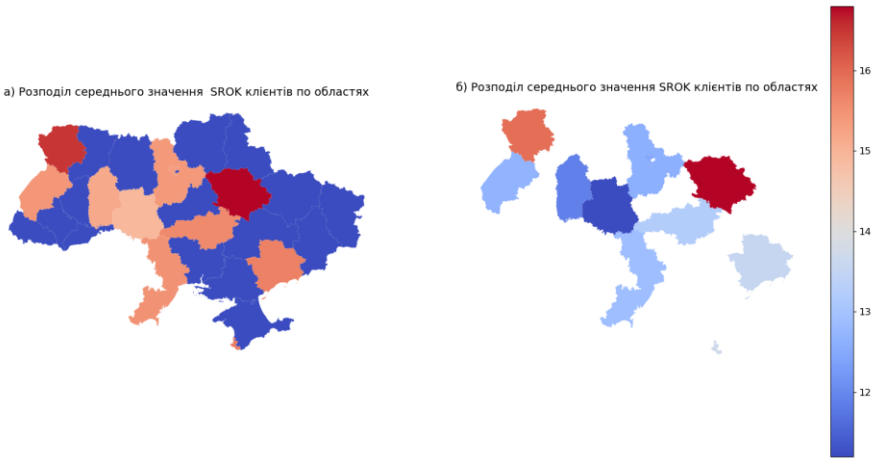


Рисунок 3.11 - Розподіл середнього строку кредиту по областях в місяцях

Як видно з рисунку, наведеного вище, в Полтавський та Волинській області в середньому найбільші строки кредитів.

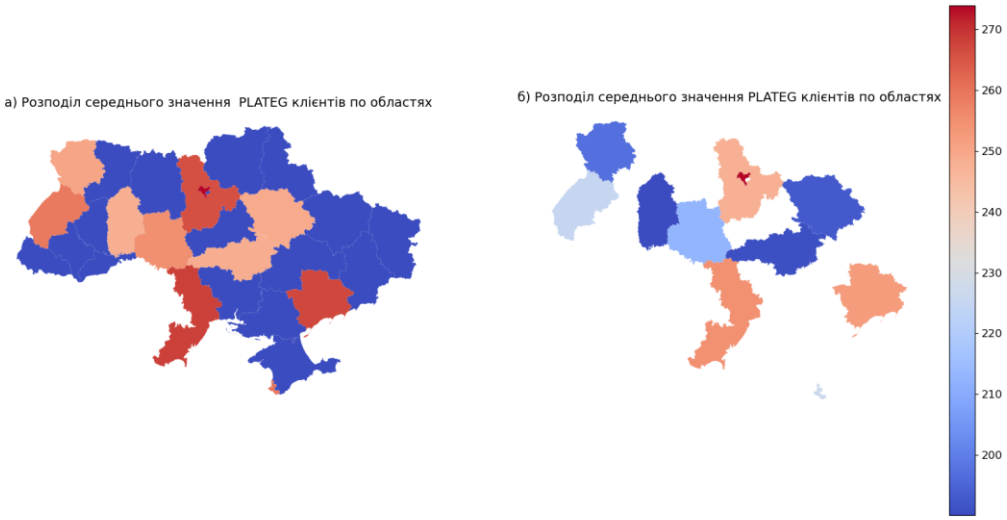


Рисунок 3.12 - Розподіл середнього строку по виплаті кредиту на місяць по областях

Як видно з рисунку, наведеного вище, найбільші платежі на місяць в Києві, Київській області, Одеській області та Запорізькій області.

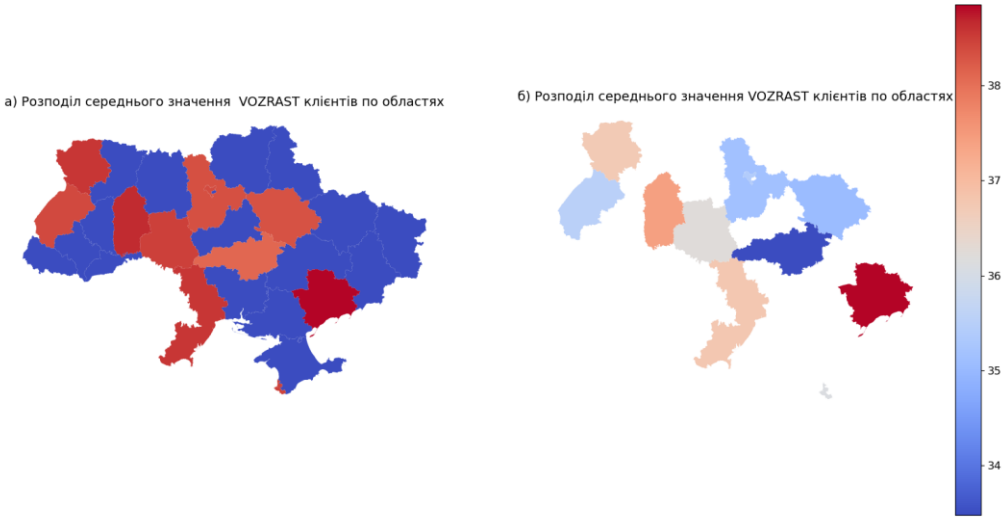


Рисунок 3.13 - Розподіл середнього віку клієнта по областях

Як видно з рисунку, наведеного вище, найстарші клієнти у Запорізькій області.

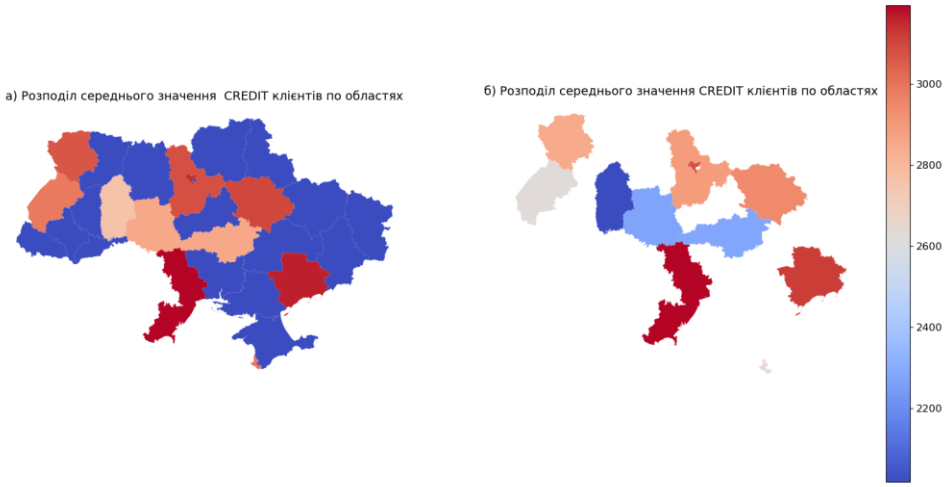


Рисунок 3.14 - Розподіл середнього розміру кредиту по областях

Як видно з рисунку, наведеного вище, найбільші кредити в Одеській та Запорізькій областях, а також в Києві.

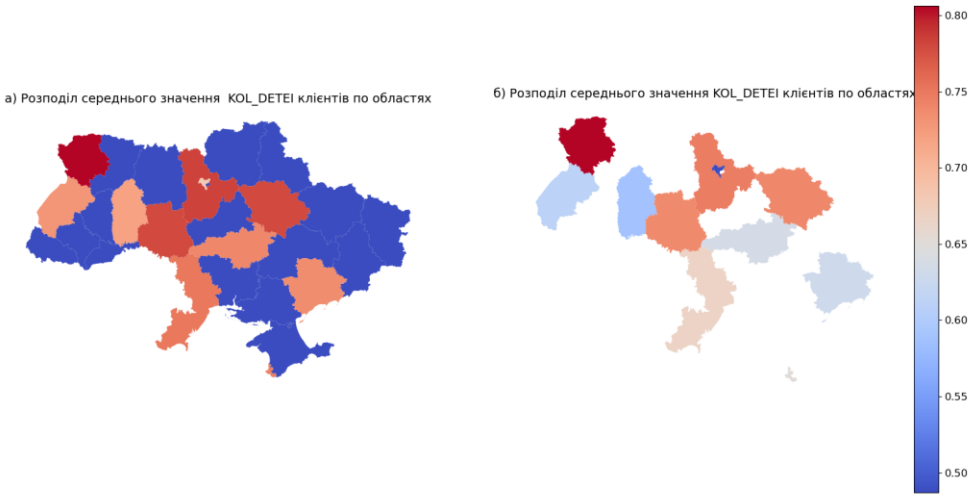


Рисунок 3.15 - Розподіл середньої кількості дітей клієнтів по областях

Як видно з рисунку, наведеного вище, в середньому найбільша кількість дітей у клієнтів із Волинської області.

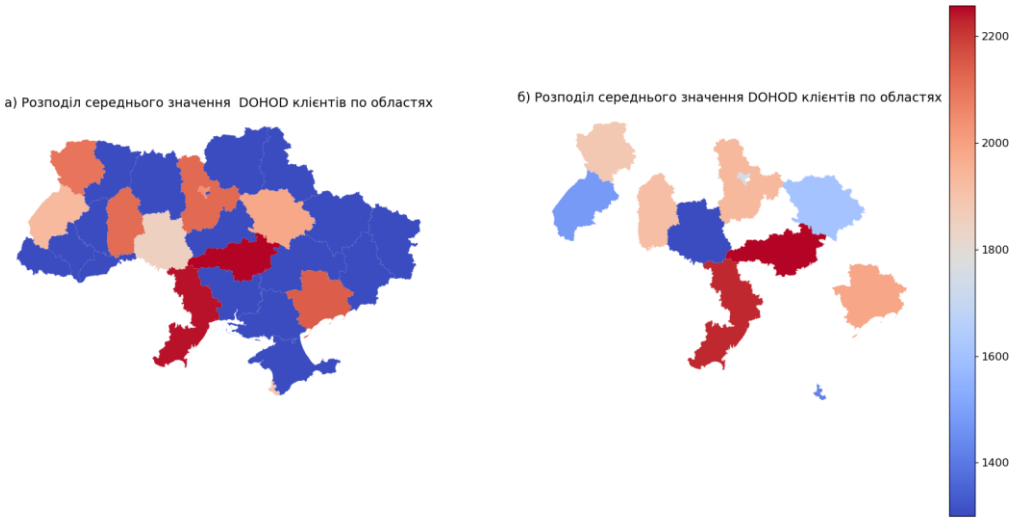


Рисунок 3.16 - Розподіл середнього розміру доходу клієнтів по областях

Як видно з рисунку, наведеного вище, найбільший середній дохід у клієнтів із Кіровоградської, Миколаївської та Одеської областей.

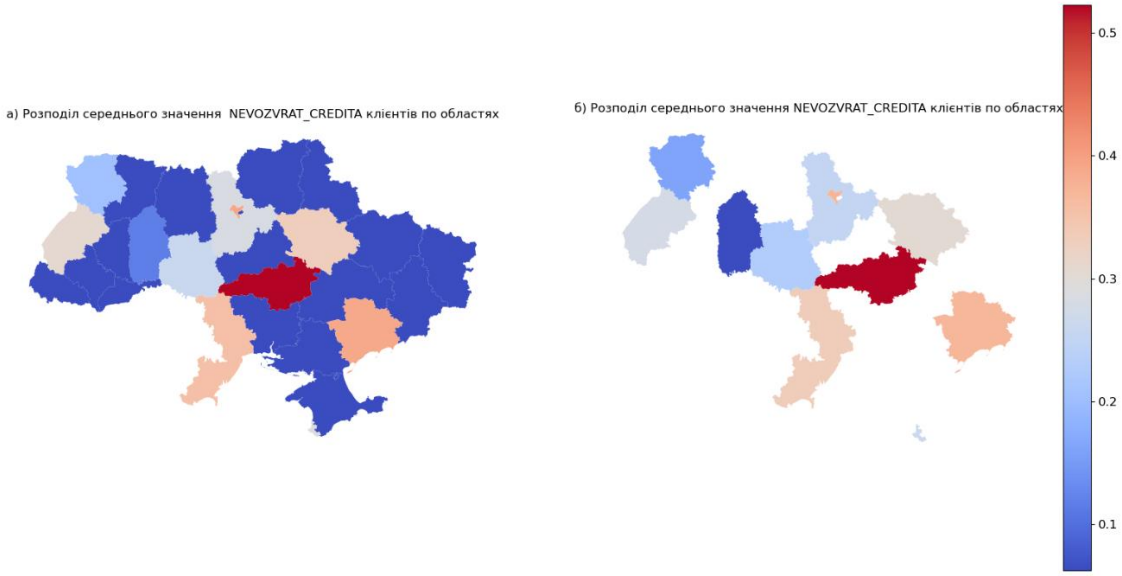


Рисунок 3.17 - Розподіл випадків неповернення кредиту по областях

Як видно з рисунку, наведеного вище, найбільш ризикованими є клієнти із Кіровоградської області, а найбільш надійними клієнти із Хмельницької та Волинської областей.

Отже, в ході аналізу розподілення клієнтів по областях, можна дійти до висновку, що присутня певна кореляція між областю та результатами кредитування клієнтів. Також присутній зв’язок між вірогідністю повернення кредиту та розміром та сроком виплати кредита.

3.2.4 Аналіз даних на кореляцію.

Аналіз даних на кореляцію є важливою частиною багатьох досліджень та аналітичних проектів. Кореляція визначає ступінь зв'язку між двома змінними і може бути використана для виявлення закономірностей та прогнозування майбутніх значень.

В цій частині буде наведена таблиця кореляції вже після обробки даних для того, щоб була можливість в повній мірі оцінити результати кореляції в тому числі і між категорійними атрибутами (Рис. 3.18).

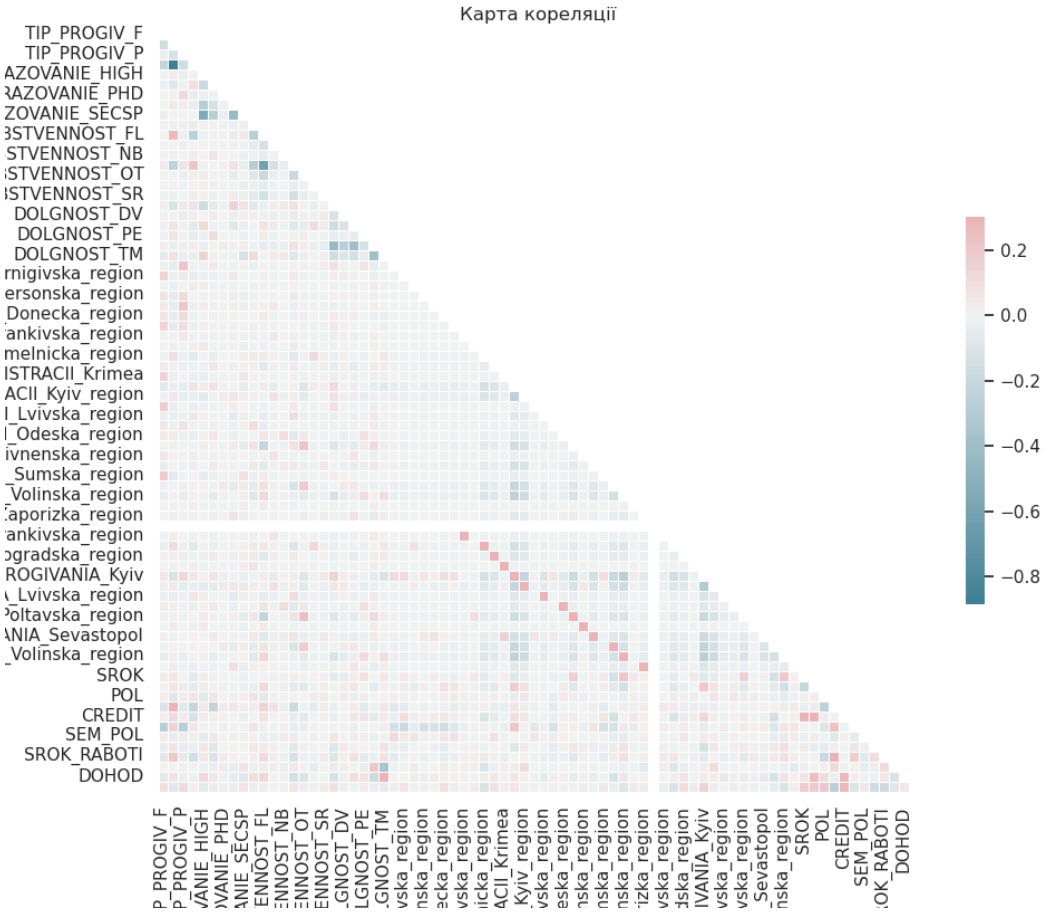


Рисунок 3.18 - Кореляційна таблиця для оброблених даних

Як видно з наведеної кореляційної таблиці, маємо лінію, паралельну основній. Ця лінія відображає кореляцію між місцем проживання та місцем реєстрації. Ці показники навряд можна назвати завеликими, оскільки вони становлять всього 20%. Також в лівому верхньому куті знаходиться відображення кореляції між значенням вищої освіти та типом проживання P. Для уникнення проблем під час класифікації колонку TIP_PROGIV_P буде видалено.

3.2.5 Обробка даних.

Після аналізу даних на кореляцію можуть з'явитися необхідність у проведенні додаткової обробки даних, щоб підготувати їх для подальшого використання в аналітиці або машинному навчанні. Обробка даних цього датасету включає в себе такі етапи:

1. Декодування даних

Декодування даних - це процес перетворення категоріальних змінних в числові значення. Якщо набір даних містить категоріальні змінні, такі як стать, місце проживання тощо, їх необхідно закодувати числовими значеннями, щоб можна було використовувати їх у подальшому аналітичному або машинному навчанні. Існують різні методи кодування категоріальних змінних, такі як one-hot encoding (кожна категорія представлена бінарним стовпчиком), label encoding (унікальний числовий код для кожної категорії) та ordinal encoding (унікальний числовий код для кожної категорії в певному порядку). В цій роботі будуть використані всі зі згаданих методів.

2. Видалення викидів

Видалення викидів - це процес видалення аномальних значень, які виходять за межі діапазону очікуваних значень. Викиди можуть виникати через помилки вимірювання, аномальні події або проблеми з обладнанням. Вони можуть значно вплинути на результати аналізу даних та спотворити їх інтерпретацію. Для виявлення викидів можна використовувати статистичні методи, такі як межі розмаху та Z-score. В цій роботі буде виконано метод Z-score.

Z-score, також відомий як стандартизоване значення, використовується для вимірювання того, наскільки далеко окреме спостереження розташоване від середнього значення в популяції, виражаючи його відхилення у відношенні до стандартного відхилення; позитивне значення Z-score вказує на значення, яке вище середнього, тоді як негативне значення вказує на значення, яке нижче середнього.

3. Нормалізація

Нормалізація - це процес масштабування даних так, щоб вони мали однаковий масштаб. Це дозволяє зробити дані порівняними та менш чутливими до впливу великих значень та в деяких випадках покращити результати передбачень. Нормалізацію можна проводити за допомогою різних методів, таких як мінімаксна нормалізація, середньо-квадратична нормалізація та нормалізація з допомогою Z-score. В даній роботі буде використана мінімаксна нормалізація, оскільки вона дозволяє максимально зберегти відношення між змінними.

Декодування на цьому наборі даних відбувався в кілька етапів. Спочатку були декодовані всі категорійні колонки що не мають послідовного зв’язку між значеннями із використанням методу OneHot методу (Рис. 3.19-3.20).

TIP_PROGIV
O
S
S
O
O
...
S
O
S
O
S

Рисунок 3.19 - Вигляд категорійної колонки до One Hot трансформації

	TIP_PROGIV_F	TIP_PROGIV_O	TIP_PROGIV_P	TIP_PROGIV_S
0	0.0	1.0	0.0	0.0
1	0.0	0.0	0.0	1.0
2	0.0	0.0	0.0	1.0
3	0.0	1.0	0.0	0.0
4	0.0	1.0	0.0	0.0
...	...	...	...	...
1595	0.0	0.0	0.0	1.0
1596	0.0	1.0	0.0	0.0
1597	0.0	0.0	0.0	1.0
1598	0.0	1.0	0.0	0.0
1599	0.0	0.0	0.0	1.0

Рисунок 3.20 - Вигляд категорійної колонки після One Hot трансформації

Далі були декодовані категорійні колонки, що містили послідовний зв’язок між своїми значеннями. Для цього кожному значенню в залежності від порядкового номера призначався відповідне число, починаючи з 0 (Рис. 3.21-3.22).

SROK_PROGIV
more_10_years
5-10_years
more_10_years
2-5years
more_10_years
...
5-10_years
more_10_years
2-5years
more_10_years
more_10_years

Рисунок 3.21 - Вигляд колонки до послідовного декодування

SROK_PROGIV
4.0
3.0
4.0
2.0
4.0
...
3.0
4.0
2.0
4.0
4.0

Рисунок 3.22 - Вигляд колонки після послідовного декодування

Після цього була проведена бінаризація всіх категорійних значень, що мають два різних значення. Для цього одному зі значень призначався 0 за відповідне число, іншому 1 (Рисунок 3.23-3.24).

POL	SEM_POL
Male	MARRIED
Female	SINGLE
Female	SINGLE
Male	MARRIED
Female	MARRIED
...	...
Female	MARRIED
Male	MARRIED
Female	MARRIED
Male	MARRIED
Female	MARRIED

Рисунок 3.23 - Вигляд двох колонок до бінаризації

POL	SEM_POL
1	0
0	0
0	0
1	0
0	0
...	...
0	0
1	0
0	0
1	0
0	0

Рисунок 3.24 - Вигляд колонок після бінаризації

Наступним кроком було видалення всіх викидів із набору даних. Для цього був використаний метод Z-score (Рис. 3.25).

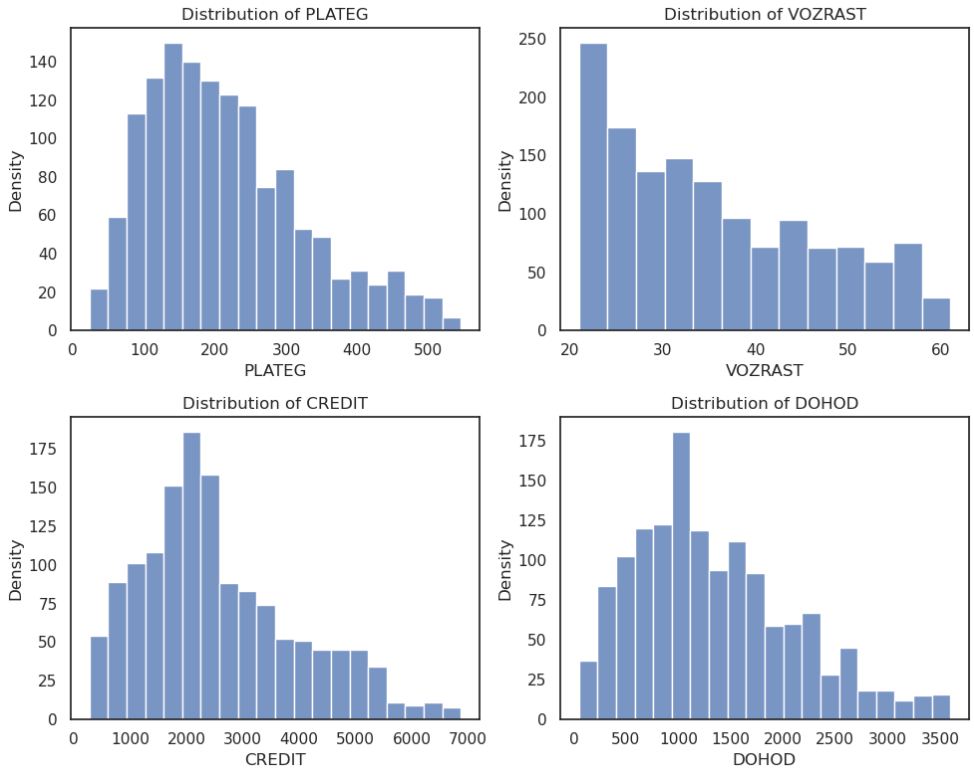


Рисунок 3.25 - Розподіл даних в числових колонках після видалення викидів методом Z-score

Останнім кроком в обробці даних було нормалізування. Для цього був використаний мінімаксний метод. Мінімаксна нормалізація - це метод масштабування даних в діапазон від 0 до 1. Для цього спочатку знаходять максимальне і мінімальне значення в наборі даних, а потім використовують формулу для перетворення кожного значення так, щоб воно лежало в діапазоні від 0 до 1 (Рис. 3.26).

SROK	PLATEG	POL	VOZRAST	CREDIT
0.333333	0.229433	1.0	0.125	0.168991
0.000000	0.463588	0.0	0.075	0.215975
0.166667	0.145024	0.0	0.025	0.070476
0.333333	0.042105	1.0	0.025	0.031070
0.333333	0.427189	0.0	0.825	0.321309
...	...	...	...	...
0.000000	0.395109	1.0	0.150	0.181267
1.000000	0.228071	1.0	0.300	0.336466
0.333333	0.263671	0.0	0.050	0.195362
0.333333	0.506911	1.0	0.850	0.425887
1.000000	0.346422	0.0	0.750	0.564565

Рисунок 3.26 - Вигляд деяких колонок після мінімаксного перетворення

3.2.6 Збалансування даних, створення декількох наборів для порівняння.

Як було вказано в розділі 3.2.2, цільові дані в датасеті незбалансовані. Це спричиняє значні проблеми під час моделювання роблячи моделі предвзятими до більшого класу. Для вирішення цієї проблеми ми будемо використовувати методи зменшення вибірки. Існує декілька різних методів зменшення вибірки, для даної роботи ми використаємо декілька та порівняємо результати під час оцінки моделей. Використані методи в себе включають:

1. Метод Tomek Links - цей метод видаляє точки даних з класу меншості, які є близькими до точок даних у класі більшості. Таким чином отримується більш збалансований набір даних.

2. Метод InstanceHardnessThreshold - цей метод використовує модель класифікації для оцінки "твердості" або достовірності кожного прикладу в наборі даних. Приклади з низькою достовірністю видаляються з набору даних, оскільки вони можуть бути зашумленими. У цьому випадку для оцінки "твердості" використовується модель логістичної регресії.

3. Метод RandomUnderSampling - цей метод випадковим чином видаляє приклади з класу більшості, поки набір даних не стане більш збалансованим.

В результаті використання цих методів було повністю збалансовано набір даних (Рис 3.27-3.28).

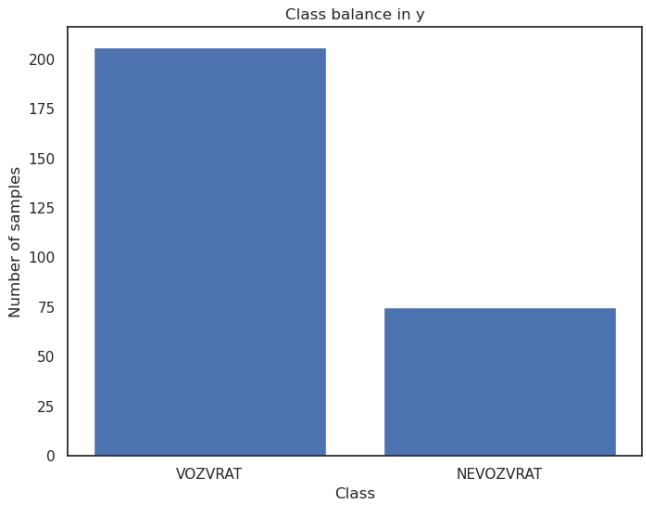


Рисунок 3.27 - Розподіл класів до використання методів зменшення вибірки

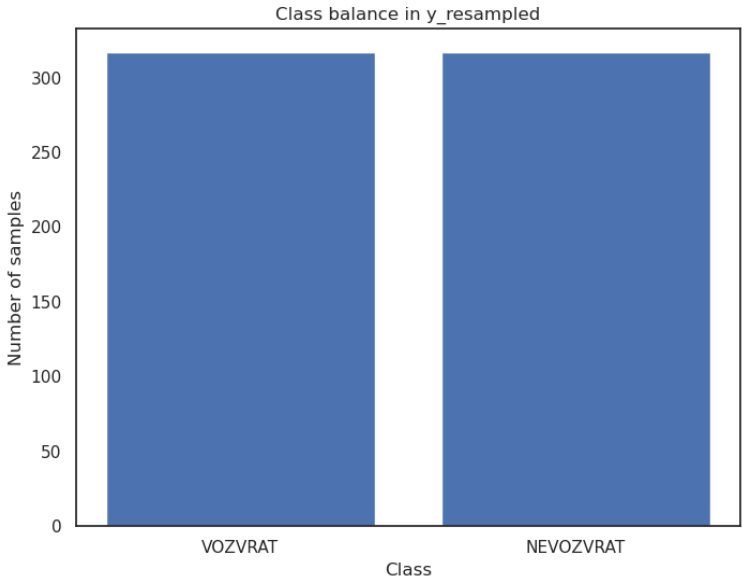


Рисунок 3.28 - Розподіл класів після використання методів зменшення вибірки

3.3 Валідація та тестування розробленої системи.

В даному розділі розглянуто процес валідації та тестування розробленої системи скорингу. Для розробки були використані моделі із розділу 2. Для валідації були використані різні метрики, такі як Accuracy, F1-Score та ROC AUC.

Accuracy - це метрика, яка вимірює точність класифікаційної моделі шляхом обчислення відношення правильно класифікованих прикладів до загальної кількості прикладів.

ROC AUC (Area Under the Curve) - це метрика, що оцінює якість моделі класифікації, вимірюючи площу під кривою ROC (Receiver Operating Characteristic). Вона враховує як чутливість (True Positive Rate) до правильно

класифікованих позитивних прикладів, так і специфічність (True Negative Rate) до правильно класифікованих негативних прикладів.

F1 - це гармонічне середнє між точністю (precision) та повнотою (recall) класифікаційної моделі. Вона враховує і точність, і повноту, що робить її корисною для оцінки моделей в умовах незбалансованих класів. F1 ближче до 1 вказує на кращу здатність моделі в точності та повноті.

Всі моделі було протестовано на декількох наборах даних, щоб впевнитись у якості результатів. Для валідації був використаний метод K-Fold, який дозволяє розділити навчальний набір на K піднаборів і виконувати K разів процес навчання та тестування на різних комбінаціях піднаборів. Це дозволяє отримати більш стійкі та надійні результати.

Для аналізу результатів використовувалися матриці помилок (confusion matrices), які відображають кількість правильно та неправильно класифікованих записів в кожному з класів.

В першу чергу моделі були натреновані та перевірені на збалансованому наборі даних, для якого був використаний попередній метод TomeLinks із доведенням набору до збалансованого із методом RandomUnderSampling (Рис. 3.29).

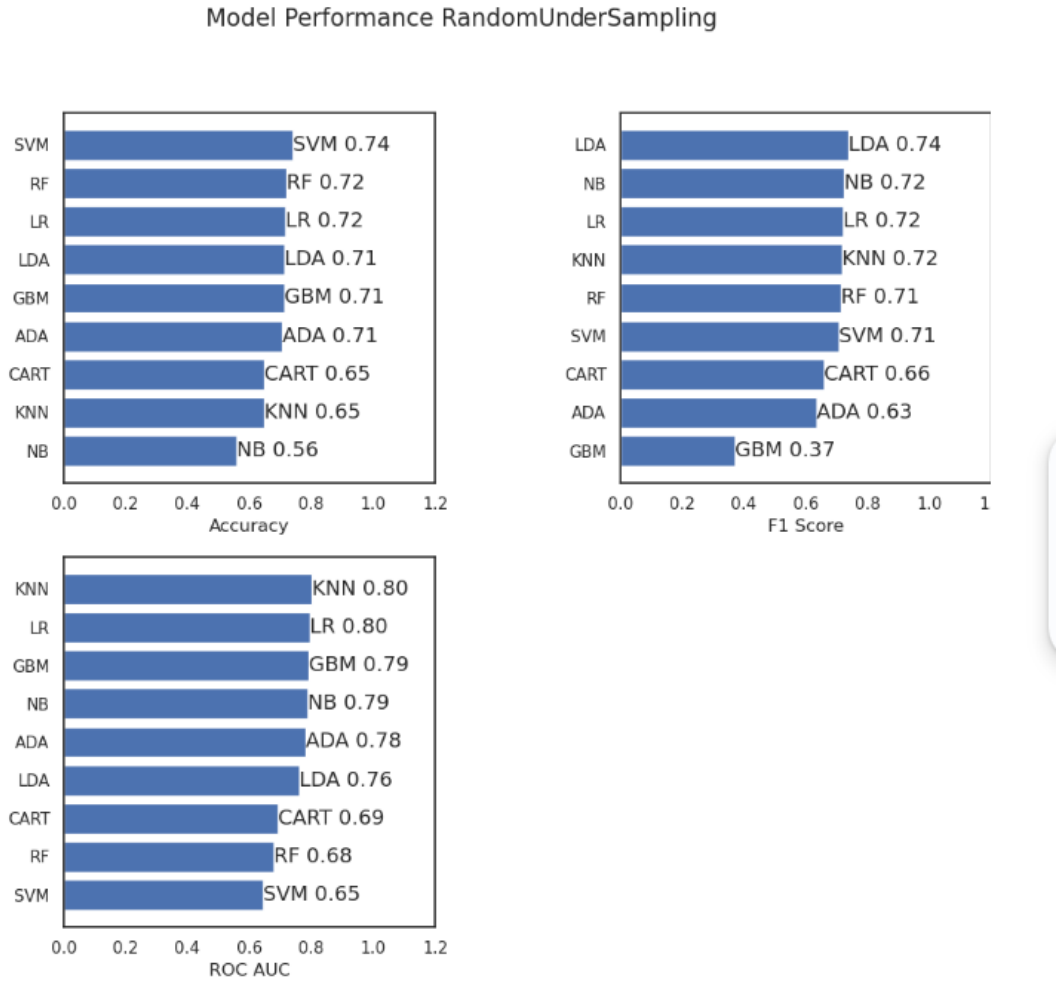


Рисунок 3.29 - Результати передбачень моделей із використанням методу K-folds(10 частин) для метрик Accuracy, F1-Score та ROC AUC на датсеті із використанням методу RandomUnderSampling

Як видно з рисунку наведеного вище, моделі показують себе доволі вдало на тестових даних. Слід врахувати, що для оцінки їх роботи було використано метод K-fold із розбиттям на 10 частин, що дозволяє дійти високої точності в процесі оцінки роботи моделей. Також слід зауважити, що оцінка роботи вказаних моделей відбулася на збалансованому датасеті та не відповідає реальним умовам, де кількість класів для передбачень є зміщеною.

Далі буде наведено результати роботи моделей із RandomUnderSampling датасетом (Табл. 3.1):

Таблиця 3.1 - Результати роботи моделей із RandomUnderSampling датасетом

	Model Name	Accuracy	F1 Score	ROC AUC
0	LR	0.734679	0.731968	0.801021
1	LDA	0.732406	0.728364	0.795057
2	KNN	0.652547	0.653063	0.687935
3	CART	0.625353	0.628801	0.642336
4	NB	0.535397	0.224235	0.682906
5	SVM	0.734613	0.724087	0.806592
6	RF	0.723145	0.730901	0.796977
7	GBM	0.731230	0.725517	0.788552
8	ADA	0.722153	0.718704	0.761910

Далі в ході роботи було натреновано ще одну групу тих самих моделей, але вже на іншому наборі даних із використанням попереднього методу TomeLinks та доведення набору даних до збалансованого методом InstanceHardnessThreshold. Оцінка роботи даних моделей відбувалась аналогічно попереднім (Рис. 3.30).

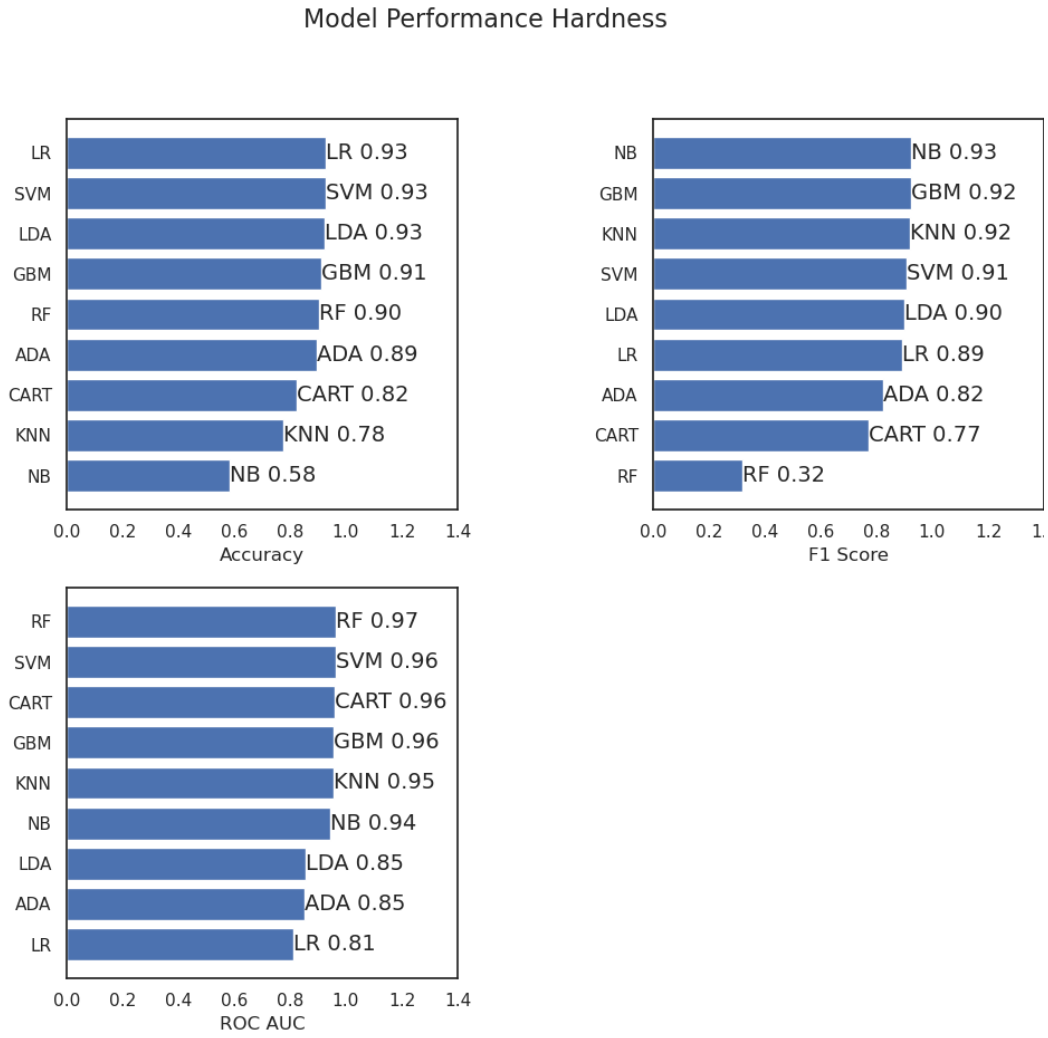


Рисунок 3.30 - Результати передбачень моделей із використанням методу K-folds(10) для метрик Accuracy, F1-Score та ROC AUC на датасеті із використанням методу IstanceHardnessThreshold

Як видно з рисунку наведеного вище, моделі справились із завданням на надзвичайно високому рівні. Такі результати є наслідком використання методу InstanceHardnessThreshold, де всі випадки, де модель не є впевненою в результаті відкидаються. Такий метод зменшення вибірки може бути корисним та використовуватись як ще один інструмент для боротьби із викидами, тільки на вищому рівні, але для даних із великою різницею в

кількості класів він може скласти негативний вплив на передбачення в майбутньому.

Далі буде наведено результати роботи моделей із датасетом Hardness (Табл. 3.2):

Таблиця 3.2 - Результати роботи моделей із Hardness датасетом

	Model Name	Accuracy	F1 Score	ROC AUC
0	LR	0.929336	0.924832	0.965814
1	LDA	0.928226	0.922595	0.962542
2	KNN	0.768809	0.762542	0.853261
3	CART	0.806243	0.799965	0.799419
4	NB	0.582145	0.298176	0.851921
5	SVM	0.927116	0.921358	0.961755
6	RF	0.905381	0.883033	0.949006
7	GBM	0.902025	0.901586	0.951115
8	ADA	0.873602	0.871730	0.937101

Далі для порівняння ті самі моделі було натреновано на незбалансованому наборі даних для порівняння результатів із двома попередніми (Рис. 3.31).

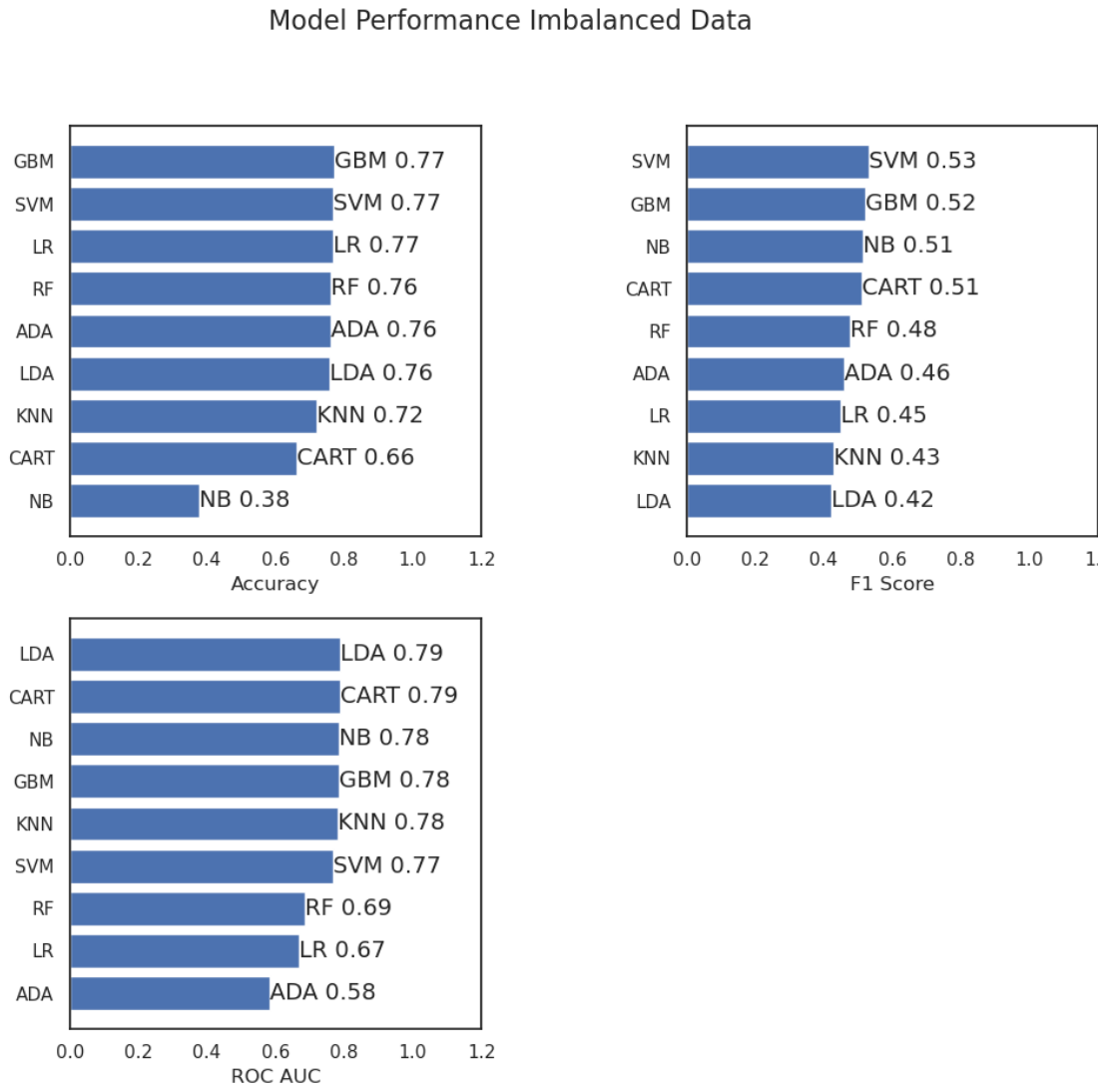


Рисунок 3.31 - Результати передбачень моделей із використанням методу K-folds(10) для метрик Accuracy, F1-Score та ROC AUC на незбалансованому датасеті

Як видно з рисунку наведеного вище, результати роботи моделей нижче ніж попередні. Це є наслідком незбалансованості в даних, що спричиняє упередженість моделей під час роботи із реальними даними. Для запобігання упередженості слід завжди використовувати збалансовані набори даних та перевіряти результати класифікації в матрицях помилок.

Далі буде наведено результати роботи моделей із незбалансованим датасетом (Табл. 3.3):

Таблиця 3.3 - Результати роботи моделей із незбалансованим датасетом

	Model Name	Accuracy	F1 Score	ROC AUC
0	LR	0.766396	0.514278	0.788401
1	LDA	0.758510	0.509512	0.781956
2	KNN	0.719174	0.429615	0.668209
3	CART	0.673302	0.415196	0.596217
4	NB	0.376028	0.457822	0.685696
5	SVM	0.767049	0.477782	0.786923
6	RF	0.763093	0.454195	0.788398
7	GBM	0.770975	0.533410	0.784664
8	ADA	0.760483	0.522728	0.767943

Після попередніх досліджень були проведено тестування всіх моделей на матрицях суперечностей. Кожна з моделей була навчена із використанням методу K-fold із розбиттям на 10 класів.

На першому етапі було використано збалансований набір даних із використанням методу RandomUnderSampling та TomeLinks. Набір даних розбивався на 10 частин, 9 з яких використовувались для тренування моделі а 1 для перевірки. Такий процес повторювався 10 разів із поступовим зміщенням частини для тестування та в результаті була обрахована середня точність моделей. Для 9 частин для тренування був використаний метод зменшення вибірки, а частина для тестування лишилась максимально наближеною до реальних умов із незбалансованим набором даних. Результати роботи кожної моделі було виведено в матрицю суперечностей (Рис. 3.32).

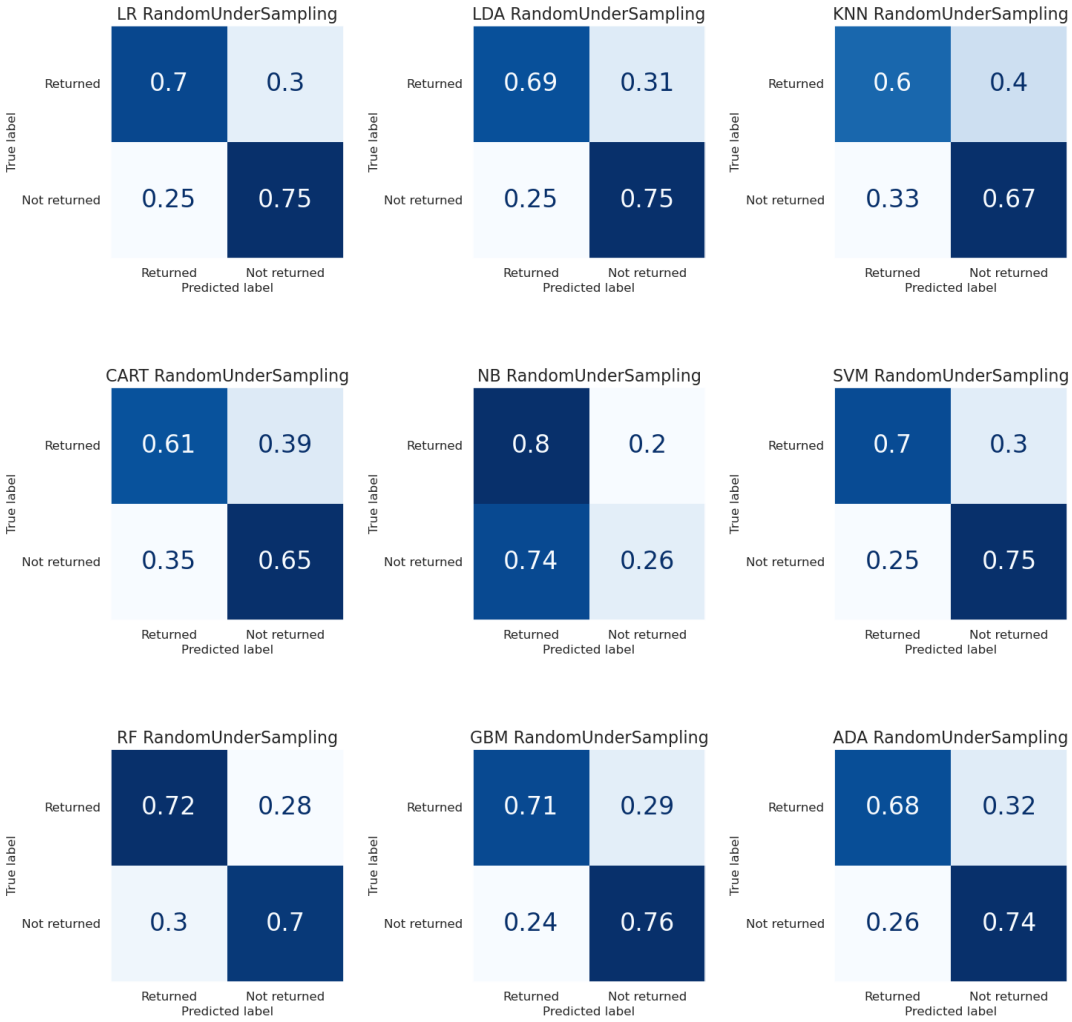


Рисунок 3.32 - Результати роботи всіх моделей із використанням методу K-folds(10) та збалансуванням тренувального набору із використанням методу RandomUnderSampling

Як видно з рисунку наведеного вище, точність моделей в такому випадку є меншою, ніж у випадку, де дані для тестування були також збалансовані. Але для логістичної регресії та випадкових лісів результати майже повністю співпадають із попередніми.

Далі було використано тестування таким же методом, тільки із використанням методу InstanceHardnessThreshold (Рис. 3.33).

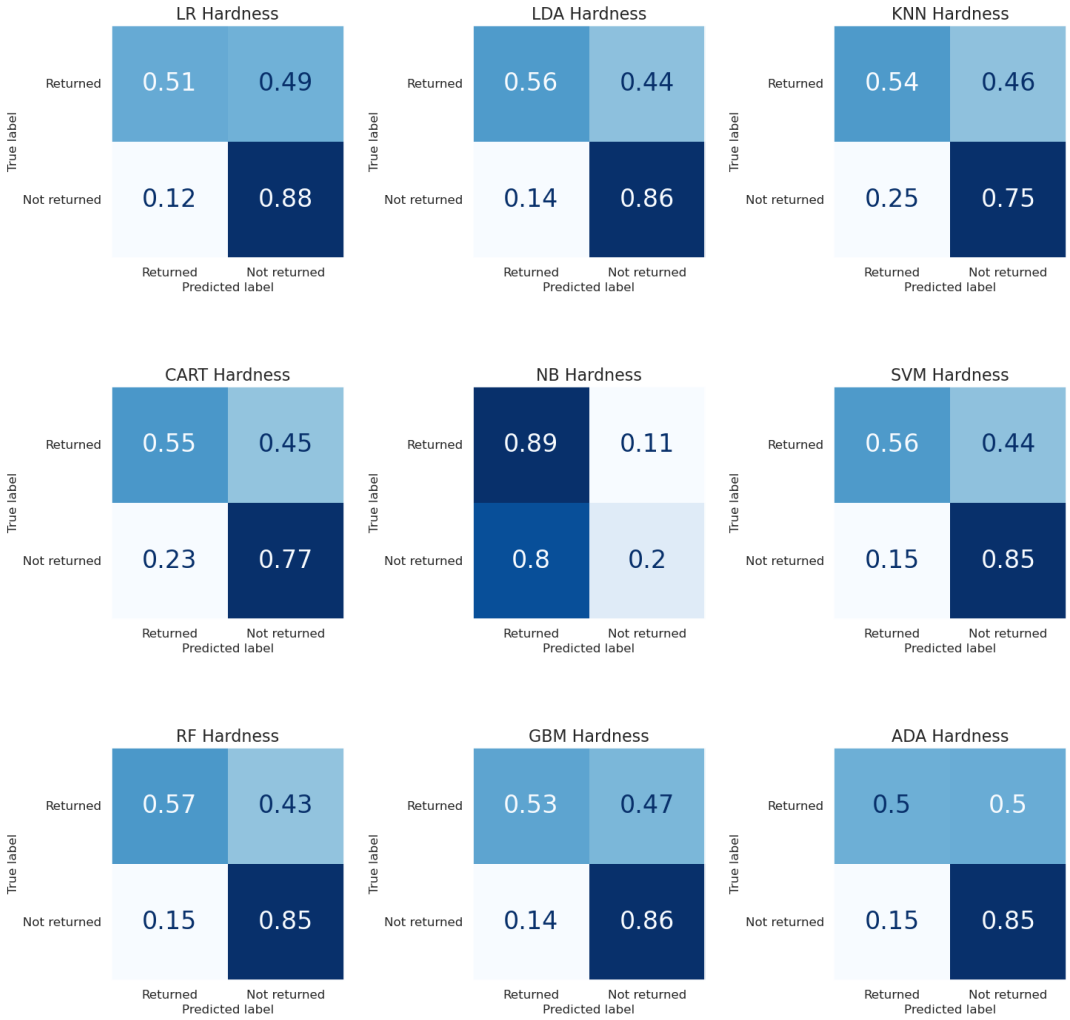


Рисунок 3.33 - Результати роботи всіх моделей із використанням методу K-folds(10) та збалансуванням тренувального набору із використанням методу InstanceHardnessThreshold

Як видно із результатів тестування, моделі мають значне упередження в сторону класу з неповерненням кредиту. Це є наслідком методу Hardness, що прибирає із тренувального датасету випадки, де модель виражає невпевненість, доки не буде досягнуто балансу даних. В деяких випадках моделі досягли точності у 87% для другого класу, що є надзвичайно хорошим результатом в таких умовах.

В середньому найкраще себе продемонструвала модель LDA, яка набрала 86% точність у класі із неповерненням та 56% в другому класі. Серед

всіх варіантів цей буде найбільш оптимальним для кредиторів, оскільки він дозволяє із великою точністю уникати клієнтів, що приносять збитки банку, при цьому не втрачати більше половини добросовісних клієнтів.

Далі для порівняння роботи між усіма випадками було проведено тестування на третьому наборі даних, де моделі тренувались і оцінювались на одному й тому ж незбалансованому наборі даних (Рис. 3.34).

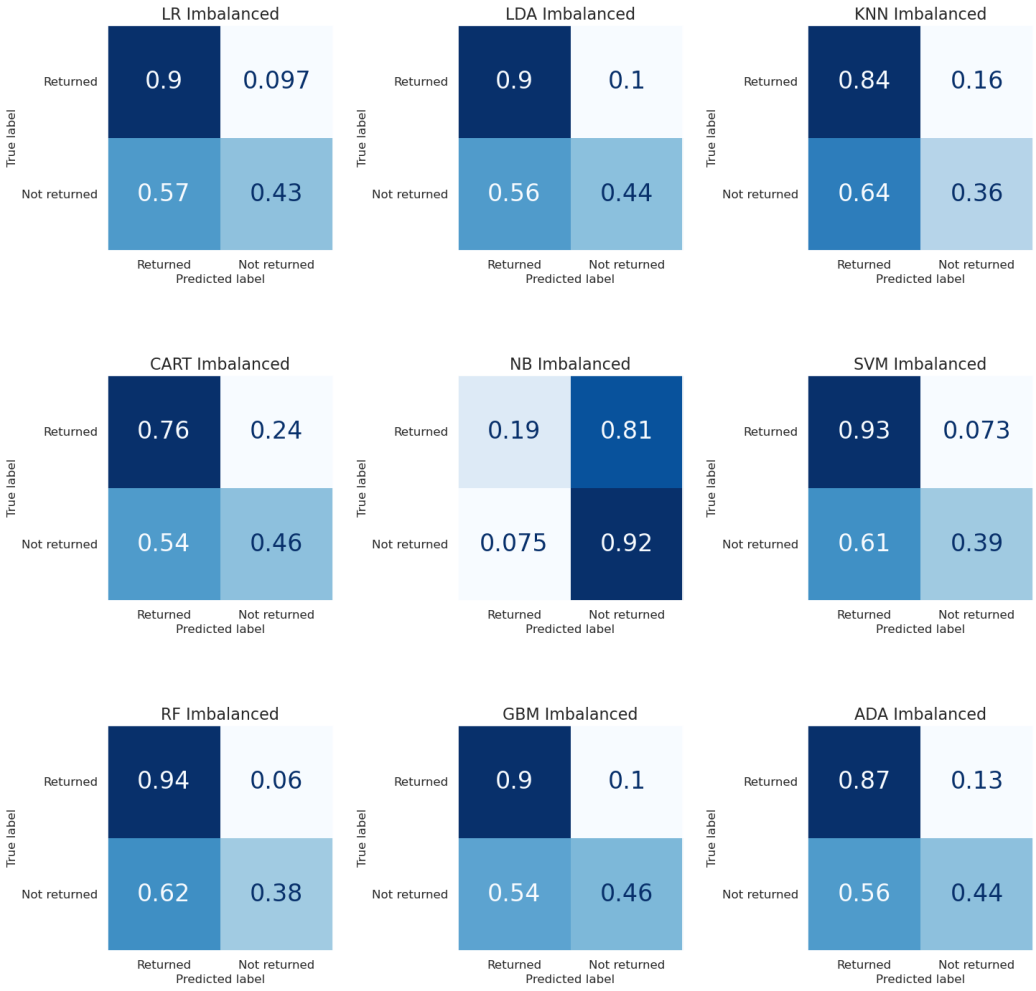


Рисунок 3.34 - Результати роботи всіх моделей із використанням методу K-folds(10) та незбалансованим тренувальним набором даних

Як видно з графіка наведеного вище, всі моделі сильним чином упереджені в сторону більшого набору даних. Хоча в деяких випадках

результати і сягають 94% точності для більшого класу, менший клас в 66% випадків передбачається невірно.

Ця частина експерименту є важливою тому що ми можемо зробити припущення, що у випадку зі збільшенням кількості даних для меншого класу, ми зможемо досягти точності для нього близької до 94%, як наведено на рисунку 3.34.

Далі для кращого розуміння важливості факторів, що були використані при побудові моделі було зібрано показники із деяких моделей із використанням Sklearn, що мали можливість виділити важливості атрибутів (Рис. 3.35-3.38).

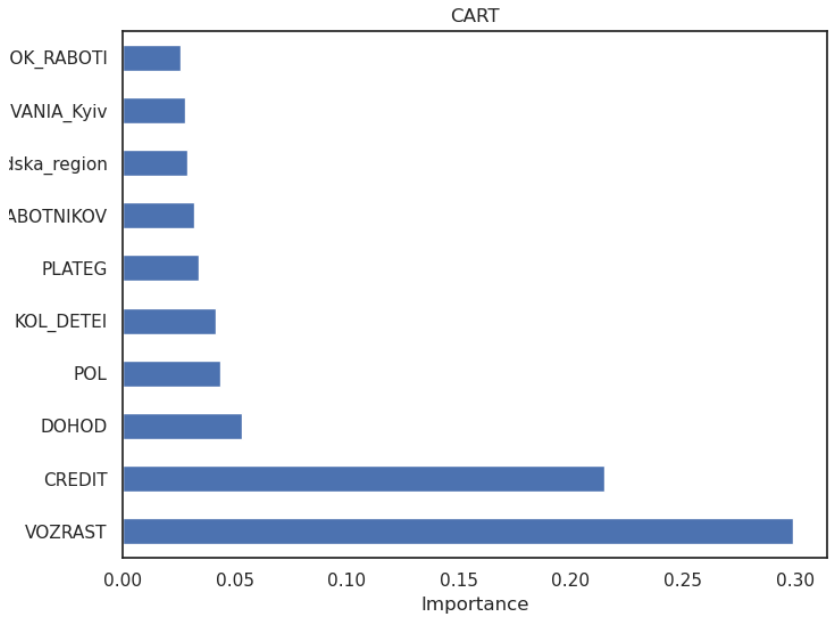


Рисунок 3.35 - Графік важливості атрибутів для класифікатора на основі дерева рішень

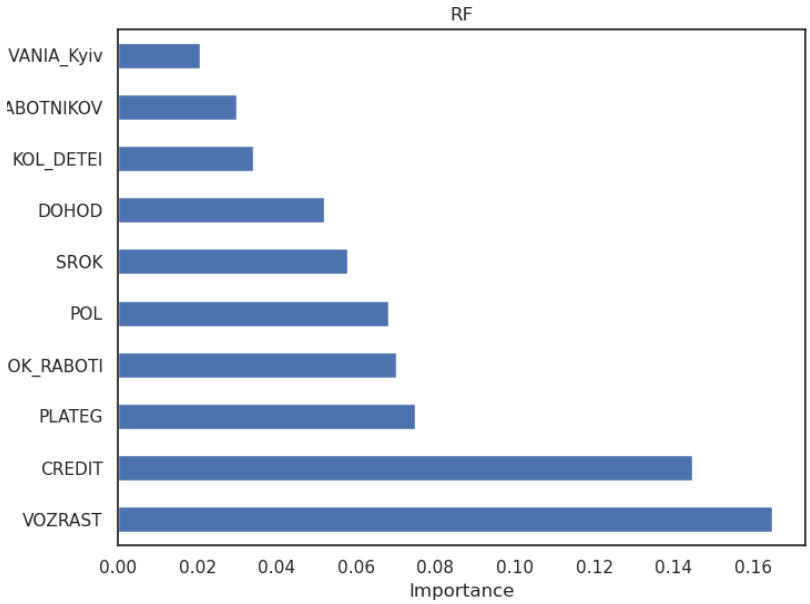


Рисунок 3.36 - Графік важливості атрибутів для класифікатора на основі випадкового лісу

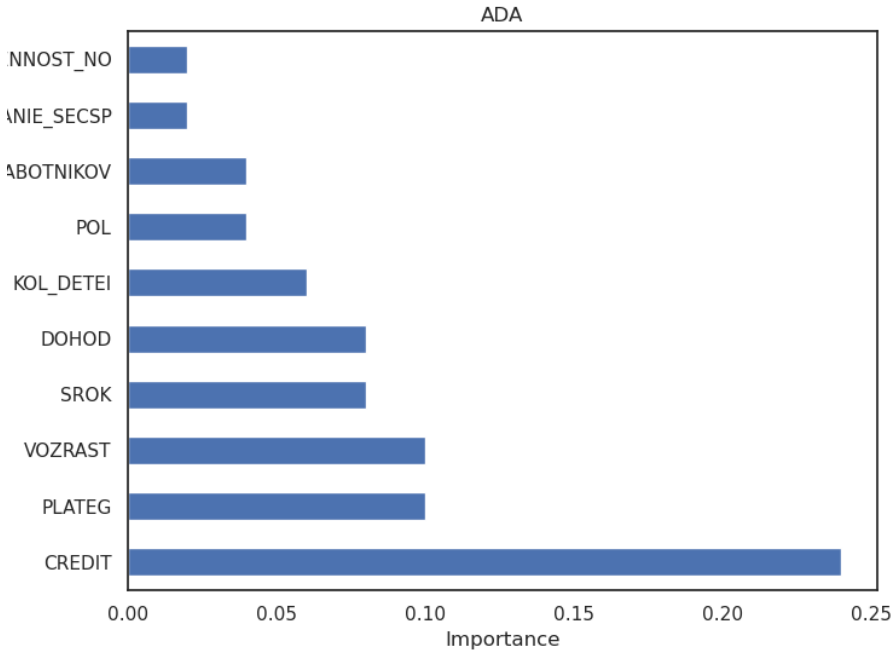


Рисунок 3.37 - Графік важливості атрибутів для класифікатора на основі алгоритму ADABoost

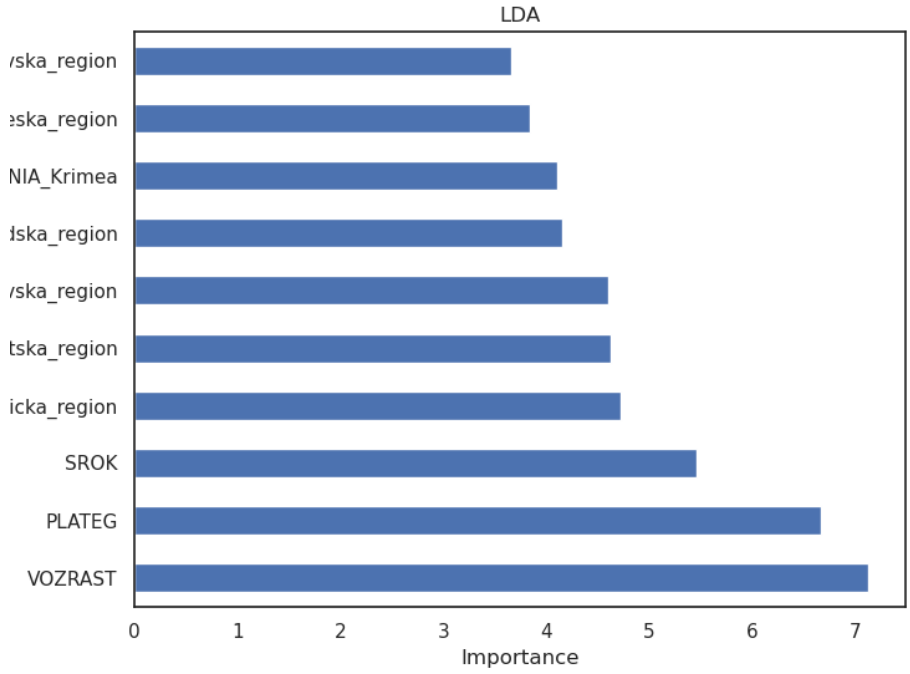


Рисунок 3.38 - Графк важливості атрибутів для класифікатора на основі LDA

Отже, в результаті тренування та оцінки моделей було розглянуто 9 моделей різних моделей, що були натреновані на трьох різних датасетах із використанням методу K-fold. Найбільш точними вийшли моделі логістичної регресії та випадкових лісів натреновані на збалансованому наборі даних із використанням методу TomeLinks та RandomUnderSampling.

Найбільш успішними з точки зору принесення прибутку для банків виявились моделі SVM та випадкові ліси. Вони показали точність передбачень для неповернення кредиту в 86% та точність у передбаченні надійних клієнтів у 57%, що може значним чином збільшити результати банкінгу в Україні.

Аналіз важливості атрибутів показав, що для декількох моделей вік та розмір кредиту є найважливішими показниками для оцінки кредитоспроможності позичальників.

3.4 Висновки до розділу.

В ході роботи над розділом був проведений аналіз, обробка та збалансування вхідних даних. Також було розроблено декілька моделей машинного навчання та оцінено їх роботу на декількох наборах даних.

В результаті дослідження, що описано в розділі можна зробити деякі висновки.

Перш за все, облік місця реєстрації клієнтів може бути корисним для банків, щоб підвищити ефективність своєї діяльності. Це допоможе зрозуміти, які регіони потребують більшого фінансового покриття та розвитку мережі філіалів.

Друге, з огляду на кореляцію між місцем проживання та місцем реєстрації, можна зробити висновок, що розподіл населення може вплинути на результати кредитування клієнтів. Це може допомогти банкам зрозуміти, як краще вибирати своїх клієнтів та де розміщувати філіали.

Третє, результати аналізу моделей показали, що моделі логістичної регресії та випадкових лісів є найбільш точними для передбачення неповернення кредиту, а моделі SVM та випадкові ліси є найбільш успішними з точки зору прибутковості для банків. Це може допомогти банкам зрозуміти, які моделі можуть бути корисними для їх діяльності та як їх вдосконалювати.

Четверте, результати аналізу декількох моделей на важливість атрибутів при проведенні класифікації показали, що для більшості з них найважливішими показниками є розмір кредиту та вік позичальника.

Узагальнюючи, можна сказати, що аналіз даних та моделей може значно допомогти банкам у покращенні їх діяльності, зрозуміти потреби своїх клієнтів, розвивати філіальну мережу та використовувати ефективні моделі для передбачення неповернення кредитів та підвищення прибутковості. І

задача оцінки кредитоспроможності позичальників є цілком реальною та має всі шанси на реалізацію.

4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОДУКТУ

В цьому розділі буде проведено докладний аналіз функцій та вартості розробленого продукту. Також будуть розглянуті основні функціональні можливості системи оцінювання кредитоспроможності позичальників, а також їх значення для різних зацікавлених сторін, таких як фінансові установи, кредитні агентства та інші організації, що займаються кредитуванням.

Передбачається, що розроблена система забезпечує аналіз великого обсягу даних про позичальників з використанням методів інтелектуального аналізу даних. Це дозволяє автоматизувати процес оцінки кредитоспроможності та забезпечує більш точні та об'єктивні результати. Крім того, система надає можливість створення персоналізованих моделей оцінювання, враховуючи специфічні потреби кожної фінансової установи.

У розділі буде проведено аналіз вартості розробленої системи, включаючи оцінку витрат на розробку, впровадження та підтримку продукту. Будуть розглянуті економічні фактори для покращення ефективності процесу кредитування та збільшення прибутковості для фінансових установ.

Для проведення функціонально-вартісного аналізу будуть використані відповідні методи та інструменти. Це дозволить виявити можливі резерви зниження витрат, покращення якості послуг та забезпечення оптимальної стратегії вибору для розробленого продукту.

Розділ детально розгляне функціональні можливості системи оцінювання кредитоспроможності позичальників та оцінить її вартість для різних зацікавлених сторін. Дане дослідження сприятиме покращенню процесу кредитування та прийняттю обґрунтованих кредитних рішень з використанням методів інтелектуального аналізу даних.

4.1 Постановка задачі.

У даній роботі проводиться застосування методу функціонально-вартісного аналізу (ФВА) для техніко-економічного аналізу розробки системи прогнозу стійкості фінансових показників. Оскільки рішення, пов'язані з проектуванням та реалізацією компонентів системи, мають вплив на її загальну ефективність, кожна окрема підсистема повинна відповідати вимогам та функціональним потребам. Таким чином, проведений аналіз включає оцінку функцій програмного продукту, призначеного для збору, обробки та аналізу даних компанії.

Технічні вимоги до розробленого програмного продукту включають наступні аспекти:

1. Функціонування на персональних комп'ютерах із стандартним набором компонентів - забезпечення сумісності з розповсюдженими платформами та операційними системами, що дозволить користувачам зручно використовувати систему на своїх пристроях
2. Зручність та зрозумілість для користувача - створення інтуїтивно зрозумілого та зручного інтерфейсу, що дозволить користувачам ефективно взаємодіяти з системою та виконувати необхідні функції без складнощів.
3. Швидкість обробки даних та доступ до інформації в реальному часі - забезпечення ефективного обробки та аналізу великого обсягу даних, що дозволить користувачам отримувати швидкий та актуальний доступ до необхідної інформації.
4. Можливість зручного масштабування та обслуговування - розробка системи, яка легко масштабується та розширюється для врахування зростаючих потреб користувачів. Крім того, система

повинна бути легко обслуговуватися та підтримуватися без значних зусиль.

5. Мінімальні витрати на впровадження програмного продукту - врахування економічних факторів та забезпечення оптимального співвідношення між вартістю продукту та його користувальницькою цінністю, що дозволить знизити витрати на впровадження та максимізувати вигоди для користувачів.

В розділі будуть розглянуті згадані вище технічні вимоги до програмного продукту з використанням методу функціонально-вартісного аналізу. Такий підхід дозволить зрозуміти, наскільки продукт відповідає поставленим цілям та ефективно вирішує завдання з прогнозування стійкості фінансових показників.

4.2 Функції програмного продукту.

У даній роботі головною функцією, позначеною як F0, є розробка потенційного програмного продукту, який дозволяє аналізувати різні характеристики, що безпосередньо впливають на стійкість підприємства. З метою досягнення цієї цілі, в рамках даного розділу будуть розглянуті такі функції:

1. F1 - вибір мови програмування для розробки;
2. F2 - аналіз даних та розробка моделі;
3. F3 - графічні показники.

Функція F1, пов'язана з вибором програмного забезпечення для розробки, має два варіанти реалізації:

а) R

б) Python

Функція F2 стосується якісного аналізу даних і також має два варіанти реалізації:

а) застосування готових бібліотек та функцій;

б) створення своїх класифікаторів з нуля.

Це дозволить гнучко налаштувати процес аналізу відповідно до специфічних потреб та вимог досліджуваної сфери.

Функція F3, пов'язана з графічними показниками, також має два варіанти реалізації:

а) використання готових бібліотек для графіків;

б) створення своїх інструментів візуалізації.

В розділі буде проведено детальний аналіз кожної з цих функцій та їх варіантів реалізації. Варіанти реалізації всіх функцій описані в морфологічній картці нижче на рисунку 4.1:

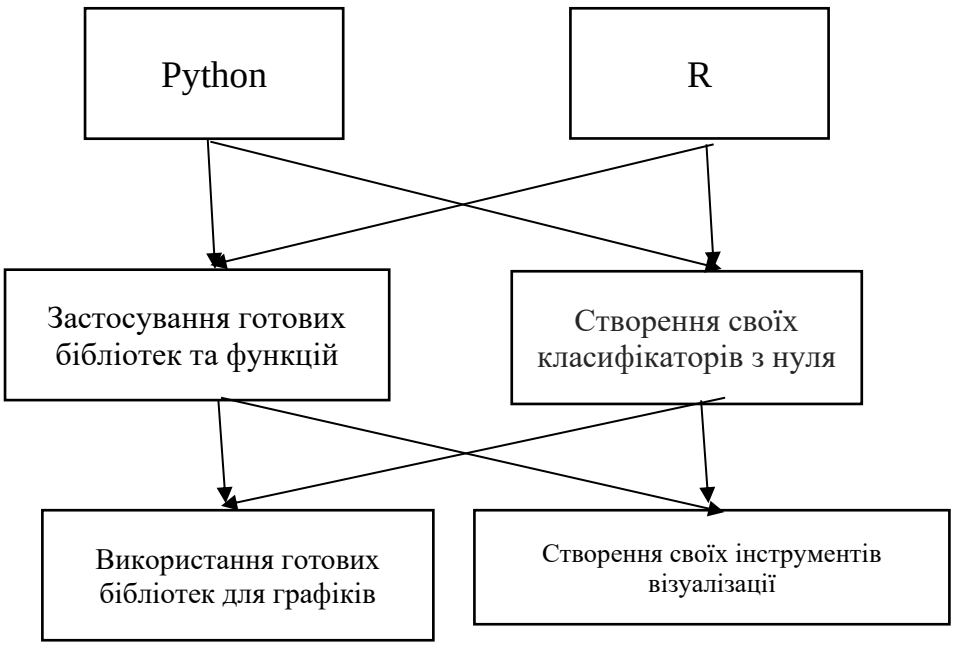


Рисунок 4.1 – Морфологічна карта системи

Наведена карта відображає всі можливі варіанти реалізації основних функцій.

Далі була створена таблиця 4.1 для визначення всіх переваг та недоліків для всіх варіантів реалізації.

Таблиця 4.1 - Позитивно-негативна таблиця для функцій та варіантів реалізації

Функція	Варіанти реалізації	Переваги	Недоліки
F1	A	Широкий спектр статистичних та аналітичних функцій. Велика та активна спільнота користувачів. Гнучкість та можливість налаштування алгоритмів обробки даних.	Вимагає певного рівня програмування та знань мови R для оптимального використання. Можливі проблеми з продуктивністю при обробці великого обсягу даних.

Продовження таблиці 4.1

F2	А	Швидкість реалізації. Можливість використання перевірених та оптимізованих рішень. Зменшення часу та зусиль, потрібних для розробки та тестування власних алгоритмів.	Обмежена гнучкість та можливості налаштування під конкретні потреби дослідження. Відсутність універсального рішення для всіх типів аналізу.
	Б	Повний контроль над процесом розробки та вибору алгоритмів. Гнучкість та можливість налаштування алгоритмів. Можливість розробки унікальних рішень	Вимагає значних зусиль і часу для розробки Можлива складність у досягненні оптимальних результатів

Продовження таблиці 4.1

	А	Швидкість реалізації та зручність використання Розширені можливості для візуалізації Наявність широкого спектру стилів та параметрів для налаштування	Обмежена гнучкість та можливість налаштування Можливі проблеми із сумісністю
F3	Б	Повний контроль над виглядом та формою графічних показників. Гнучкість та можливість налаштування під конкретні потреби Можливість розробки унікальних та інноваційних рішень	Вимагає значних зусиль та знань для розробки Можлива складність у досягненні високої якості графічного представлення

Порівнюючи варіанти реалізації для кожної функції, що наведені в таблиці вище, можна зробити висновки, що варіанти Б та 2 та 3 (Python для F1, створення власних класифікаторів для F2, створення своїх інструментів візуалізації для F3) підходять для великих проектів, оскільки вони надають повний контроль та гнучкість, але вимагають значних зусиль та знань для розробки та впровадження. З іншого боку, варіант А (R для F1, застосування

готових бібліотек та функцій для F2, використання готових бібліотек для графіків для F3) є швидким та економним варіантом, який може бути використаний для швидкого аналізу та візуалізації даних. Python - простий у використанні та має широкі можливості для інтеграції з різними системами, що робить його зручним вибором для розробки та забезпечення взаємодії між різними компонентами проекту. З іншого боку, R є потужним інструментом з багатим набором статистичних функцій і пакетів, що дозволяє здійснювати складний аналіз даних та проводити статистичні дослідження.

Таким чином, вибір конкретного варіанту залежатиме від масштабу проєкту, доступних ресурсів та вимог до гнучкості та контролю над розробкою. У нашому випадку деякі варіанти реалізації доведеться відкинути:

F1:

Система може бути розроблена як і з використанням варіанту А, так і Б. Вибір між двома варіантами в даному випадку не є принциповим.

F2:

Обираємо швидкість розробки та економію часу та ресурсів, оскільки кредитний скоринг є доволі тривіальним завданням для машинного навчання, стандартні бібліотеки мають справитись. Обираємо варіант А.

F3:

Обираємо простоту розробки та економію ресурсів, оскільки система не вимагає візуалізації для клієнтів, а тільки виведення результатів передбачень. Візуалізація буде присутня у звітності для аналітичного відділу, тому вона не відіграє такої значної ролі. Обираємо варіант А.

Тому в результаті оцінки буде розглянуто такі варіанти реалізації ПЗ:

1) *F1(A) - F2(A) - F3(A)*
2) *F1(B) - F2(A) - F3(A)*

4.3 Вибір параметрів для продукту.

На підставі аналізу, який був проведений у попередньому пункті, визначаються основні параметри вибору, які будуть використовуватися для обчислення коефіцієнта технічного рівня. У ході оцінки програмного продукту, будуть враховуватися наступні параметри:

X1 - швидкодія мови програмування;

X2 - обсяг пам'яті для обчислень та збереження даних;

X3 - час, необхідний для навчання моделі;

X4 - очікуваний обсяг програмного коду.

Значення цих параметрів (погано, посередньо та добре) визначаються з урахуванням вимог замовника та умов експлуатації програмного продукту, як показано у таблиці 4.2.

Таблиця 4.2 - Параметри продукту

Параметр	Позначення	Одиниці виміру	Значення		
			Погано	Посередньо	Добре
Швидкодія	X1	Операція/мс	40	80	200
Обсяг пам’яті	X2	Мб	1 024	500	300
Час для навчання	X3	Год	10	5	3
Очікуваний обсяг продукту	X4	Кількість рядків коду	1500	900	400

На основі таблиці вище, було побудовано графіки для кожного параметра. (Рис. 4.2)

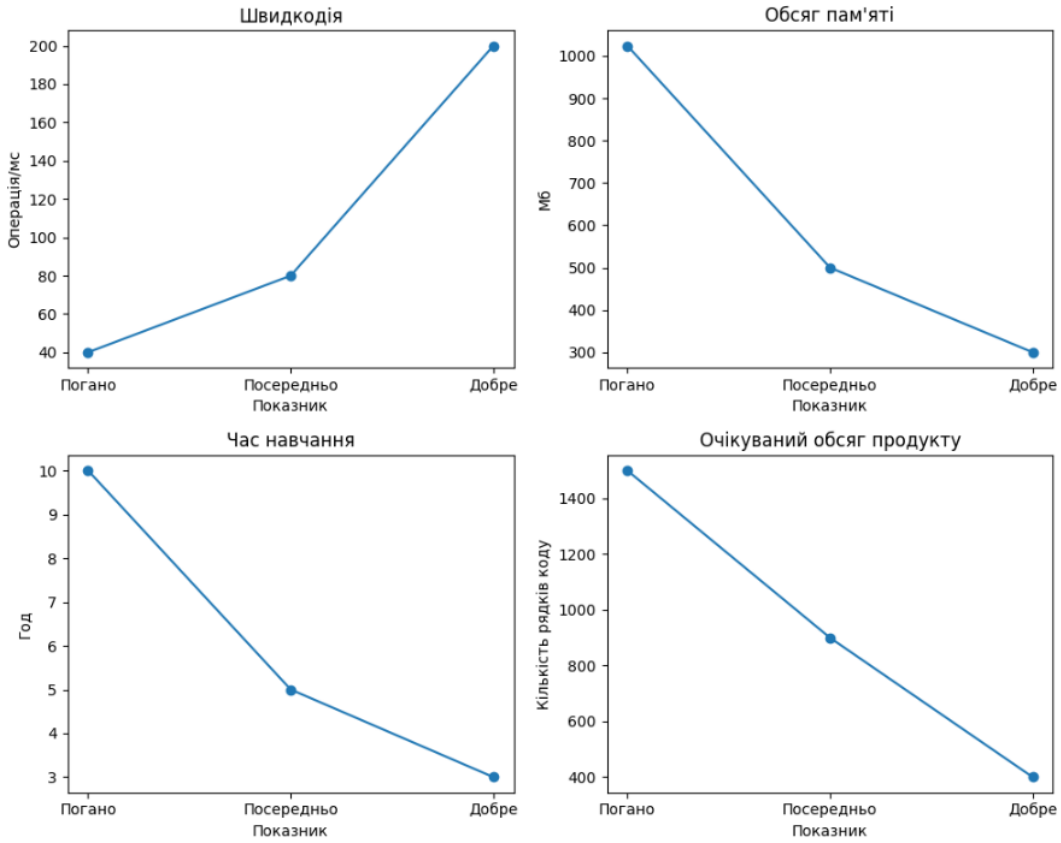


Рисунок 4.2 - Візуалізація для кожного доступного параметра

4.4 Аналіз параметрів від експертів.

Після детального обговорення та аналізу кожен експерт оцінює ступінь важливості кожного параметра для розробки програмного продукту, спрямованого на досягнення найточніших результатів у моделюванні адаптивного прогнозування та обчисленні прогнозних значень. Визначення значимості кожного параметра здійснюється за допомогою методу попарного

порівняння. Оцінку проводить експертна комісія, що складається з 7 осіб. Процес визначення коефіцієнтів значущості включає:

1. Встановлення рівня значимості параметра шляхом призначення різних рангів.
2. Перевірку придатності експертних оцінок для подальшого використання.
3. Визначення оцінки попарного пріоритету параметрів.
4. Обробку отриманих результатів та визначення коефіцієнта значимості.

Цей підхід дозволяє систематично оцінити та врахувати важливість кожного параметра у розробці програмного продукту, забезпечуючи наочну інформацію для подальшого прийняття рішень та налаштування моделей адаптивного прогнозування. Результати оцінки параметрів від експертів записані в таблицю 4.3.

Таблиця 4.3 - Результати оцінки параметрів від експертів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія	Операція/мс	1	1	2	2	1	1	2	10	-7,5	56,25
X2	Обсяг пам’яті	Мб	3	4	3	3	4	3	4	24	6,5	42,25
X3	Час для навчання	Год	4	3	4	4	3	4	2	24	6,5	42,25
X4	Очікуваний обсяг продукту	Кількість рядків коду	3	2	1	1	2	2	1	12	-5,5	30,25
	Сума		10	10	10	10	10	10	10	70	0	171

Для перевірки достовірності експертних оцінок визначимо наступні параметри:

1. Обчислимо суму рангів кожного з параметрів та загальну суму рангів за формулою (4.1):

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70, (4.1)$$

де N - кількість експертів,

n - кількість параметрів.

2. Обчислимо середню суму рангів за формулою (4.2):

$$T = \frac{1}{n} R_{ij} = 17{,}5 \ (4.2)$$

3. Обчислимо відхилення суми рангів кожного параметра від середньої суми рангів за формулою (4.3):

$$\Delta_i = R_i - T \ (4.3)$$

4. Обчислимо загальну суму квадратів відхилень за формулою (4.4):

$$S = \sum_{i=1}^{N} \Delta_i^2 = 171. (4.4)$$

Порахуємо коефіцієнт узгодженості за формулою (4.5):

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 171}{7^2(4^3 - 4)} = 0{,}698 > W_k = 0{,}67. \#(4.5)$$

Знайдений коефіцієнт узгодженості перевищує нормативний значення 0.67, тому ранжування можна вважати достовірним. З використанням результатів ранжування проведемо попарне порівняння всіх параметрів, результати заносимо у таблицю 4.4.

Таблиця 4.4 - Попарне порівняння параметрів.

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	<	<	<	<	<	<	<	<	0,5
X1 і X3	<	>	>	<	<	<	=	<	0,5
X1 і X4	<	<	>	>	<	<	>	>	1,5
X2 і X3	<	>	<	<	>	<	>	<	0,5
X2 і X4	=	>	>	>	>	>	>	>	1,5
X3 и X4	>	>	>	>	>	>	>	>	1,5

Для визначення ступеня переваги одного параметра над іншим, числове значення аіj визначається згідно з формулою (4.6):

$$a_{ij} = \{1.5 \text{ при } X_i > X_j \; 1.0 \text{ при } X_i = X_j \; 0.5 \text{ при } X_i < X_j \; . (4.6)$$

З отриманих числових оцінок переваги складається матриця А=$\| a_{ij} \|$

Для кожного параметра розраховується вагомість Кві згідно з наступними формулами:

$$K_{\text{ві}} = \frac{b_i}{\sum_{i=1}^{n} b_i} \quad (4.7)$$

$$b_i = \sum_{i=1}^{N} a_{ij} \; (4.8)$$

Відносні оцінки розраховуються декілька разів, поки наступні значення не відрізняються від попередніх менше ніж на 2%. На наступних кроках відносні оцінки розраховуються згідно з наступними формулами:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}, \qquad (4.9)$$

$$b_i' = \sum_{i=1}^{N} a_{ij} b_j \ (4.10)$$

З таблиці 4.5 видно, що різниця між значеннями коефіцієнтів вагомості не перевищує 2%, тому додаткові ітерації не потрібні.

Таблиця 4.5 - Розрахунок вагомості параметрів

Параметри x_i	Параметри x_j				Перша ітер.		Друга ітер.		Третя ітер		Четверта ітер	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$	b_i^3	$K_{\text{ві}}^3$
X1	1	0,5	0,5	1,5	3,5	0,21	12,25	0,2	42,875	0,18	150,0625	0.20
X2	1,5	1	0,5	1,5	4,5	0,28	20,25	0,33	70,875	0,30	318,9375	0.31
X3	1,5	1,5	1	1,5	5,5	0,34	30,25	0,51	105,875	0,45	915,0625	0.45
X4	0,5	0,5	0,5	1	2,5	0,15	6,25	0,11	15,625	0,06	39,0625	0.04
Всього:					16	1	60	1	235,5	1	1514,25	1

4.5 Аналіз якості реалізації варіантів функцій.

Для оцінки якості виконання основних функцій програмного продукту, ми розглядаємо абсолютні значення параметрів, таких як об'єм пам'яті (X2), час для навчання (X3) та потенційний об'єм програмного коду (X4), що

відповідають вимогам технічних умов його функціонування. Додатково, оцінюється параметр швидкості роботи мови програмування (X1), яке має прийнятне значення.

Для розрахунку показників рівня якості кожного варіанта реалізації програмного продукту, застосовується коефіцієнт технічного рівня (ККj), який враховує вагомість кожного параметра.

$$K_K(j) = \sum_{i=1}^{n} K_{\text{в}i,j} B_{i,j}, \#(4.11)$$

де n – кількість параметрів;

$K_{\text{в}i}$– коефіцієнт вагомості i–го параметра;

B_i – оцінка i–го параметра в балах.

Згідно з формулою (4.11), кожний показник рівня якості обчислюється шляхом множення вагомості параметра (Кві) на його оцінку (Ві) в балах. Ці розрахунки представлені в таблиці 4.6, яка надає оцінку якості варіантів реалізації основних функцій програмного продукту.

Таблиця 4.6 - Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	А	X1	87	10	0,2	2
	Б	X2	100	24	0,31	7,44
F2	А	X3	25	24	0.45	10,8
F4	А	X4	25	12	0,04	0,48

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}], \#(4.12)$$

визначаємо рівень якості кожного з варіантів:

$$K_{K1} = 2 + 2 + 0{,}48 = 4{,}48 ;$$

$$K_{K2} = 2 + 7{,}44 + 0{,}48 = 9{,}92 .$$

Отже, кращим в даному випадку є варіант 1, що має вищий рівень якості.

4.6 Економічний аналіз розробки.

Для оцінки вартості розробки програмного продукту, спочатку здійснюється розрахунок трудомісткості для кожного варіанту виконання основних завдань, які включають розробку проекту програмного продукту і розробку програмної оболонки.

Функція 1 відрізняється від функції 1 тільки рівнем мови програмування. Обрахування загальної трудомісткості відбувалось із використанням формули (4.13).

$$\text{Т}_\text{О} = \text{Т}_\text{Р} \cdot \text{К}_\text{П} \cdot \text{К}_\text{СК} \cdot \text{К}_\text{М} \cdot \text{К}_\text{СТ} \cdot \text{К}_\text{СТ.М}, (4.13) ,$$

де ТР – трудомісткість розробки ПП;

КП – поправочний коефіцієнт;

КСК – коефіцієнт на складність вхідної інформації;

КМ – коефіцієнт рівня мови програмування;

КСТ – коефіцієнт використання стандартних модулів і прикладних програм;

КСТ.М – коефіцієнт стандартного математичного забезпечення

Загальна трудомісткість обчислюється шляхом сумування трудомісткості всіх завдань.

Для першого завдання, що відноситься до категорії розрахункового характеру та має ступінь новизни А, була обчислена трудомісткість, що складає 30 людино-днів. Застосувавши поправочний коефіцієнт КП = 1.6, який враховує вид нормативно-довідкової інформації, та коефіцієнт КСК = 1.2, що враховує складність контролю вхідної та вихідної інформації, отримали загальну трудомісткість програмування першого завдання. Врахувавши використання стандартних модулів за допомогою коефіцієнта КСТ = 0.9, та КМ=1 загальна трудомісткість першого завдання становить:

$$T_1 = 30 \cdot 1.8 \cdot 0.9 \cdot 1 = 48{,}6 \text{ людино-днів, або } 388{,}8 \text{ людино-годин}$$

Для другого завдання було проведено аналогічні обрахунки із використанням КМ=1.3 та КСТ 0.8:

$$T_2 = 30 \cdot 1.8 \cdot 0.8 \cdot 1.3 = 56{,}16 \text{ людино-днів, або } 449{,}28 \text{ людино-годин}$$

Як видно із обрахунків, найбільшу трудомісткість має функція під номером 2.

В розробці буде брати участь один програміст з окладом 40 000 та один спеціаліст із даних із окладом 50 000.

Середня зарплата буде обрахована за такою формулою:

$$C_ч = \frac{M}{T_m \cdot t} \text{ грн.}, (4.14)$$

де М – місячний оклад працівників;

Tm – кількість робочих днів місяць;

t – кількість робочих годин в день.

$$С_ч = \frac{30000 + 40000}{2 \cdot 21 \cdot 8} = 267{,}8 \text{ грн.} \quad (4.15)$$

Далі обраховується середня зарплата за формулою (4.16)

$$СЗП = С_ч \cdot Т_i \cdot КД\,, \quad (4.16)$$

де $С_ч$ – величина погодинної оплати праці програміста;

$Т_i$ – трудомісткість відповідного завдання;

$К_Д$ – норматив, який враховує додаткову заробітну плату (20%).

Зарплата робітників із соц. внеском, що становить 22% дорівнює:

1. $С_{ЗП} = 267{,}8 \cdot 388{,}8 \cdot 1.2 = 1\ 524\ 651$ грн.
2. $С_{ЗП} = 267{,}8 \cdot 449{,}28 \cdot 1.2 = 1\ 761\ 819$ грн.

Далі буде обраховано ціну використання ЕОМ. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 40 000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 40\ 000 \cdot 0{,}2 = 96000 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 40800 \cdot (1 + 0.2) = 115200 \text{ грн.}$$

Відрахування на соціальний внесок:

1. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 1\ 524\ 651 \cdot 0,22 = 335423,22$ грн.
2. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 1\ 761\ 819 \cdot 0,22 = 387600,18$ грн.

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 40 000 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.2 \cdot 0.25 \cdot 40\ 000 = 12000 \text{ грн.},$$

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача;

$К_А$– річна норма амортизації;

$Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.2 \cdot 40000 \cdot 0.08 = 3840 \text{ грн.},$$

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 12 – 16) \cdot 8 \cdot 0.35 =$$

$$= 627,2 \text{ години},$$

де $Д_К$ – календарна кількість днів у році;

$Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів;

$Д_Р$ – кількість днів планових ремонтів устаткування;

t –кількість робочих годин в день;

$К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 627,2 \cdot 0,3 \cdot 0,2 \cdot 4,86889 = 183,226 \text{ грн.},$$

де $N_С$ – середньо-споживана потужність приладу;

$К_З$– коефіцієнтом зайнятості приладу;

$Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$С_Н = Ц_{ПР} \cdot 0.67 = 40000 \cdot 0,67 = 32160 \text{ грн.}$$

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_А + С_Р + С_{ЕЛ} + С_Н, (4.17)$$

$С_{ЕКС} = 12000 + 3840 + 183,226 + 32160 = 48\ 183,226$ грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 48\ 183,226 / 627,2 = 76,8$ грн/год.

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$С_М = С_{М-Г} \cdot T, (4.18)$$

1. $С_М = 76,8 \cdot 388,8 = 29\ 859,84$ грн.
2. $С_М = 76,8 \cdot 449,28 = 34\ 504,704$ грн.

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0,67, \#(4.19)$$

1. $C_Н$ = 1 524 651 · 0,67 = 1 02 1516,17 грн.
2. $C_Н$ = 1 761 819 · 0,67 = 1 180 418,73 грн.

Отже, вартість розробки ПП за варіантами становить:

$$C_{ПП} = C_{ЗП} + C_{ВІД} + C_М + C_Н, \#(4.20)$$

1. $C_{ПП}$ = 1 524 651 + 335 423,22 + 29 859,84 + 1 021 516,17 = 2 911 450,23 грн.
2. $C_{ПП}$ = 1 761 819 + 387 600,18 + 34 504,704 + 1 180 418,73 = 3 364 342,614 грн.

4.7 Вибір кращого варіанту ПП техніко-економічного рівня.

В цьому розділі буде відбуватись розрахунок і вибір найкращого варіанту для розробки програмного продукту за формулою (4.21)

$$К_{ТЕР_j} = К_{К_j} / C_{Ф_j}, (4.21)$$

$$К_{ТЕР1} = 4,48 / 2\,911\,450,23 = 1,539 \cdot 10^{-6},$$
$$К_{ТЕР2} = 9,92 / 3\,364\,342,614 = 2,949 \cdot 10^{-6}.$$

Отже, як видно із обрахунків вище, кращим варіантом є другий, що має коефіцієнт розміром $2,949 \cdot 10^{-6}$.

Після проведеного функціонально-вартісного аналізу, можна дійти до висновків що серед двох варіантів, що залишились для виконання роботи

найкращим є другий варіант. У даного варіанту виявився найкращий показник коефіцієнту якості.

Обраний варіант має такі параметри:

- Використання мови програмування R
- Використання готових бібліотек та функцій для моделі
- Використання готових бібліотек для візуалізації

Цей варіант виконання забезпечує швидку та надійну розробку системи, та не буде вимагати зайвого використання ресурсів для компанії.

4.8 Висновки до розділу.

В даній частині було проведено повний функціонально-вартісний аналіз програмного продукту з метою визначення та оцінки його основних функцій. Результати аналізу дозволили знайти параметри, що характеризують програмний продукт. Крім того, на основі проведеного аналізу був обраний варіант реалізації програмного продукту. Цей етап дозволив отримати необхідну інформацію для подальшого розроблення та визначення вартості програмного комплексу.

ВИСНОВКИ

Висновки за результатами дослідження.

В результаті аналізу предметної області було досліджено попередній досвід використання алгоритмів машинного навчання для інтелектуального аналізу даних в банківській сфері. У ході розробки системи мовою Python було створено декілька моделей оцінювання кредитоспроможності позичальників, що здатні забезпечувати середню точність результатів в 74-76%, а також точність у 86% для більш важливого класу.

Дослідження показало, що найбільш ефективними методами для досягнення збалансованих даних є Random Under Sampling для гарного результату по всіх класах та Instance Hardness Threshold для покращення результатів для більш важливого класу. Було навчено дев'ять різних моделей машинного навчання, а саме: Decision Tree, Random Forest, AdaBoost, Gradient Boosting, XGBoost, LightGBM, Logistic Regression, K-Nearest Neighbors та Support Vector Machines. Також було проведене функціонально-вартісне дослідження з метою виявлення найкращого способу та шляху реалізації моделі.

На основі отриманих результатів можна зробити висновок, що розроблена система оцінювання кредитоспроможності позичальників може стати ефективним інструментом для оцінки ризику кредитування та може бути використана у банківській сфері та інших фінансових установах. Однак, доцільним є подальше покращення точності та розширення функціональності системи для більш точного та повного аналізу кредитоспроможності позичальників.

Рекомендації щодо використання розробленої системи оцінювання кредитоспроможності.

При виборі моделі для оцінки кредитного скорингу, необхідно враховувати кілька важливих показників, які можуть вплинути на прибутковість банку. Наприклад, при виборі моделі варто оцінити суму втрати від одного клієнта, що може бути спричинена невідшкодованим кредитом або збитками в разі порушення договору кредиту. Також варто враховувати суму надходження від одного клієнта, яка може бути забезпечена процентами за кредитом, а також ціну отримання одного клієнта, яка може бути пов'язана з рекламними витратами та витратами на обслуговування клієнтів.

Оскільки ці показники можуть бути ключовими для вибору найбільш ефективної моделі для банку, їх важливо враховувати разом з даними про результати роботи моделей. Це допоможе максимізувати прибутки банку та зменшити ризики, пов'язані з кредитуванням. Тому, під час вибору моделі для кредитного скорингу, необхідно звернути увагу на всі згадані показники та провести аналіз їх впливу на прибутковість банку.

Перспективи подальшого дослідження теми.

Під час дослідження було виявлено деякі обмеження та можливості для подальшого дослідження теми роботи.

Одним з можливих напрямків є розширення кількості даних, які використовуються для навчання моделей, щоб збільшити точність прогнозів. Також можна дослідити можливість використання додаткових факторів, таких

як інформація про поточне місце роботи та юридичне минуле клієнта, які можуть покращити точність кредитного скорингу.

Крім того, можна дослідити можливість застосування глибинного навчання (deep learning) для створення більш складних моделей, які можуть дати кращі результати для прогнозування кредитної спроможності клієнтів.

Також важливим напрямком може бути розробка системи автоматизованого моніторингу кредитної спроможності клієнтів, яка буде оперативно реагувати на зміни в поведінці клієнтів та економічній ситуації в Україні.

Такі дослідження можуть бути корисними для банків та фінансових установ, що прагнуть зменшити ризик неповернення кредитів та покращити прибутковість своєї діяльності.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Principles On Creditworthiness Assessments. A perspective from the credit reference industry. *Association of Consumer Information Suppliers*. 6 November 2018. URL: https://accis.eu/wp-content/uploads/2018/11/Final-Proofread-Principles-on-Creditworthiness-Assessments-and-the-Role-of-CRAs.pdf (Дата звернення 19.04.2023)

2. Creditworthiness Assessment Model. *Openriskmanual*. URL: https://www.openriskmanual.org/wiki/Creditworthiness_Assessment_Model (Дата звернення 19.04.2023)

3. Christina K. Livada. Assessment of consumers' creditworthiness. 04 September 2019. URL: https://link.springer.com/article/10.1007/s12027-019-00574-w (Дата звернення 19.04.2023)

4. Răzvan Rareş Sorin Şerbu. Creditworthiness Assessment for Credit Institutions and for the Risk Associated with Excessive Leverage toward Sustainable Performance. October 2021. URL: https://www.researchgate.net/publication/355572938_Creditworthiness_Assessment_for_Credit_Institutions_and_for_the_Risk_Associated_with_Excessive_Leverage_toward_Sustainable_Performance (Дата звернення 19.04.2023)

5. Huseyin Ince & Bora Aktan. A comparison of data mining techniques for credit scoring in banking: A managerial perspective. 14 Oct 2010. URL: https://www.tandfonline.com/doi/pdf/10.3846/1611-1699.2009.10.233-240 (Дата звернення 19.04.2023)

6. Alireza Hooman, Govindan Marthandan, Wan Fadzilah Wan Yusoff, Mohana Omid and Sasan Karamizadeh. STATISTICAL AND DATA MINING METHODS IN CREDIT SCORING. Nov 2015. URL: https://www.jstor.org/stable/26415601(Дата звернення 19.04.2023)

7. Eiman Kambal; Izzeldin Osman; Methag Taha; Noon Mohammed; Sara Mohammed. Credit scoring using data mining techniques with particular reference to Sudanese banks. 2016. URL: https://ieeexplore.ieee.org/document/6633966 (Дата звернення 19.04.2023)

8. Huseyin Ince, Bora Aktan. A comparison of data mining techniques for credit scoring in banking: A managerial perspective. September 2009. URL: https://www.researchgate.net/publication/247904757_A_comparison_of_data_mining_techniques_for_credit_scoring_in_banking_A_managerial_perspective(Дата звернення 19.04.2023)

9. Cheng-Lung Huang, Mu-Chen Chen, Chieh-Jen Wang. Credit scoring with a data mining approach based on support vector machines. November 2007. URL: https://www.sciencedirect.com/science/article/abs/pii/S095741740600217X (Дата звернення 19.04.2023)

10. Jasmina Nalić, Goran Martinovic. Building a Credit Scoring Model Based on Data Mining Approaches. February 2020. URL: https://www.researchgate.net/publication/340098236_Building_a_Credit_Scoring_Model_Based_on_Data_Mining_Approaches (Дата звернення 19.04.2023)

11. Tom Barkley. Which FICO Scores Do Lenders Use? March 20, 2023. URL: https://www.investopedia.com/which-fico-scores-do-lenders-use-5116055 (Дата звернення 22.04.2023)

12. James McWhinney. What Lenders Look at on Your Credit Report. June 29, 2021. URL: https://www.investopedia.com/articles/personal-finance/072514/what-lenders-look-your-credit-report.asp (Дата звернення 22.04.2023)

13. Lesia Dmytryshyn, Ivan Blahun. A model for achieving the allocative efficiency of credit resources in Ukraine's banking system. October 2016. URL: https://www.researchgate.net/publication/309144591_A_model_for_achieving_the_allocative_efficiency_of_credit_resources_in_Ukraine%27s_banking_system (Дата звернення 19.04.2023)

14. How does credit scoring work? *Ukrainian Bureu of Credit Histories*. URL: https://www.ubki.ua/en/credit-score (Дата звернення 19.04.2023)

15. Elmira Kahramanian, Svitlana Ganziuk. Theoretical and practical aspects of determining the credit rating of banks in Ukraine. January 2018. URL: https://www.researchgate.net/publication/336598846_THEORETICAL_AND_PRACTICAL_ASPECTS_OF_DETERMINING_THE_CREDIT_RATING_OF_BANKS_IN_UKRAINE (Дата звернення 19.04.2023)

16. Olesia Solodovnik, Kateryna Dokunina. Assessment of the borrower's credit capacity: analysis of main approaches. January 2021. URL: https://www.researchgate.net/publication/352074728_ASSESSMENT_OF_THE_BORROWER%27S_CREDIT_CAPACITY_ANALYSIS_OF_MAIN_APPROACHES (Дата звернення 19.04.2023)

17. Credit scoring Case study in data analytics. *Deloitte.* 18 April 2016. URL: https://www2.deloitte.com/content/dam/Deloitte/global/Documents/Financial-Services/gx-be-aers-fsi-credit-scoring.pdf (Дата звернення 19.04.2023)

18. Büşra Alma Çallı, Erman Coşkun. A Longitudinal Systematic Review of Credit Risk Assessment and Credit Default Predictor. November 23, 2021. URL: https://journals.sagepub.com/doi/full/10.1177/21582440211061333 (Дата звернення 19.04.2023)

19. Bee Wah Yap, Seng Huat Ong, Nor Huselina Mohamed Husain. Using data mining to improve assessment of credit worthiness via credit scoring models. 15 September 2011. URL: https://www.sciencedirect.com/science/article/abs/pii/S0957417411006749 (Дата звернення 19.04.2023)

20. Saba Moradi, Farimah Mokhatab Rafie. A dynamic credit risk assessment model with data mining techniques: evidence from Iranian banks. 20 March 2019. URL: https://jfin-swufe.springeropen.com/articles/10.1186/s40854-019-0121-9 (Дата звернення 19.04.2023)

21. Credit Score: Definition, Factors, and Improving It. *Investopedia*. September 18, 2022. URL: https://www.investopedia.com/terms/c/credit_score.asp (Дата звернення 19.04.2023)

23. Saurabh Arora, Sushant Bindra, Survesh Singh, Vinay Kumar Nassa. Prediction of credit card defaults through data analysis and machine learning techniques. URL: https://www.sciencedirect.com/science/article/abs/pii/S2214785321035148 (Дата звернення 03.04.2023)

24. Niklas Bussmann, Paolo Giudici, Dimitri Marinelli & Jochen Papenbrock. Explainable Machine Learning in Credit Risk Management. URL: https://link.springer.com/article/10.1007/s10614-020-10042-0 (Дата звернення 03.04.2023)

25. Suraj Patil, Varsha Nemade, Piyush Kumar Soni. Predictive Modelling For Credit Card Fraud Detection Using Data Analytics. URL: https://www.sciencedirect.com/science/article/pii/S1877050918309347 (Дата звернення 03.04.2023)

26. Michael Bücker, Gero Szepannek, Alicja Gosiewska, Przemyslaw Biecek. Transparency, auditability, and explainability of machine learning models in credit scoring. 21 Jun 2021 URL: https://www.tandfonline.com/doi/abs/10.1080/01605682.2021.1922098?journalCode=tjor20 (Дата звернення 03.04.2023)

27. Przemysław Biecek, Marcin Chlebus, Janusz Gajda, Alicja Gosiewska, Anna Kozak, Dominik Ogonowski, Jakub Sztachelski, Piotr Wojewnik. Enabling Machine Learning Algorithms for Credit Scoring -- Explainable Artificial Intelligence (XAI) methods for clear understanding complex predictive models. 14 Apr 2021. URL: https://arxiv.org/abs/2104.06735 (Дата звернення 03.04.2023)

28. Elena Dumitrescu, Sullivan Hué, Christophe Hurlin, Sessi Tokpavi. Machine learning for credit scoring: Improving logistic regression with non-linear decision-tree effects. 16 March 2022 URL:

https://www.sciencedirect.com/science/article/abs/pii/S0377221721005695 (Дата звернення 03.04.2023)

29. Evander E T Nyoni, Ntandoyenkosi Matshisela. Credit scoring using machine learning algorithms. November 2018. URL: https://www.researchgate.net/publication/336240486_Credit_scoring_using_machine_learning_algorithims (Дата звернення 03.04.2023)

ДОДАТОК А ЛІСТИНГ

```
        # Виводимо розподіл колонок по класах, що мають менше 10 унікальних
значень
        cols_to_print = []
        for col in df.columns:
            if df[col].nunique() < 10:
                cols_to_print.append(col)

        n_cols = 2
        n_rows = (len(cols_to_print) + 1) // n_cols
        fig, axes = plt.subplots(n_rows, n_cols, figsize=(18, 24))
        for i, col in enumerate(cols_to_print):
            row_idx = i // n_cols
            col_idx = i % n_cols
            df[col].value_counts().plot(kind='bar', ax=axes[row_idx, col_idx])
            axes[row_idx,  col_idx].set_title(f'Unique  value  distribution  for
{col}')
            axes[row_idx, col_idx].set_xlabel('Unique values')
            axes[row_idx, col_idx].set_ylabel('Count')
        plt.tight_layout()
        plt.savefig('cls_dist_plt.png')
        plt.show()


        # In[35]:


        # Виводимо розподіл колонок, що мають більше 10 унікальних значень

        cols_to_plot = []
        for col in df.columns:
            if col in ['OBL_REGISTRACII', 'OBL_PROGIVANIA']:
                continue
            if df[col].nunique() > 10:
                cols_to_plot.append(col)
```

```
        fig, axes = plt.subplots(2, 2, figsize=(10, 10))
        for i, col in enumerate(cols_to_plot):
            row_idx = i // 2
            col_idx = i % 2
            sns.histplot(df[col], ax=axes[row_idx, col_idx])
            axes[row_idx, col_idx].set_title(f'Distribution of {col}')
            axes[row_idx, col_idx].set_xlabel(col)
            axes[row_idx, col_idx].set_ylabel('Density')
        plt.tight_layout()
        plt.savefig('continuous_dist.png')
        plt.show()


        # In[36]:


        df['OBL_REGISTRACII'].sort_index()


        # In[37]:


        # Виводимо розподіл по областях

        fig, axes = plt.subplots(1, 2, figsize=(12, 6))

        for i, col in enumerate(['OBL_REGISTRACII', 'OBL_PROGIVANIA']):
            category_counts = df[col].value_counts().sort_values()

            sns.barplot(x=category_counts.values,      y=category_counts.index,
ax=axes[i], color='steelblue')
            axes[i].set_title(f'Distribution of clients by {col}')
            axes[i].set_xlabel('Count')
            axes[i].set_ylabel(col)

        plt.tight_layout()
        plt.savefig('region_dist.png')
        plt.show()
```

```
# In[38]:


df = pd.read_excel('/kaggle/input/scoring-data/NK_ScoringBuilding4_1600.xls')

# Видаляємо замалі регіони що є шумом

regions_to_drop = ['Chernigivska_region', 'Rivnenska_region', 'Krimea', 'Ivano-Frankivska_region']
mask = df['OBL_PROGIVANIA'].isin(regions_to_drop)
df = df[~mask]

# Читаємо карту України

ukraine_map = gpd.read_file('/kaggle/input/ukraine-map/gadm41_UKR_1.shp')
ukraine_map_raw = gpd.read_file('/kaggle/input/ukraine-map/gadm41_UKR_1.shp')
column_names = {'NAME_1': 'OBL_PROGIVANIA'}

# Перейменовуємо назви областей щоб вони співпадали із датасетом
ukraine_map = ukraine_map.rename(columns=column_names)

mapping = {
    'Kyiv': 'Kiev City',
    'Kyiv_region': 'Kiev',
    'Vinnicka_region': 'Vinnytsya',
    'Poltavska_region': 'Poltava',
    'Lvivska_region': "L'viv",
    'Sevastopol': "Sevastopol'",
    'Krimea': 'Crimea',
    'Volinska_region': 'Volyn',
    'Rivnenska_region': 'Rivne',
    'Odeska_region': 'Odessa',
    'Khmelnicka_region': "Khmel'nyts'kyy",
    'Zaporizka_region': 'Zaporizhia',
    'Kirovogradska_region': 'Kirovohrad',
    'Chernigivska_region': 'Chernihiv',
    'Ivano-Frankivska_region': 'Ivano-Frankivs\'k',
    'Mykolaev_region': 'Mykolayiv'
```

```
    }

    df['OBL_PROGIVANIA'] = df['OBL_PROGIVANIA'].map(mapping)
    df['OBL_PROGIVANIA']

    merged = pd.merge(ukraine_map, df, on='OBL_PROGIVANIA')

    res = pd.DataFrame(merged['OBL_PROGIVANIA'].value_counts())
    res = res.rename(columns=({'OBL_PROGIVANIA': 'COUNT'}))
    res['OBL_PROGIVANIA'] = res.index

    # Виводимо різні зображення карти

    ukraine_map_raw = gpd.read_file('/kaggle/input/ukraine-
map/gadm41_UKR_1.shp')

    ukraine_map_raw = ukraine_map_raw.rename(columns={'NAME_1':
'OBL_PROGIVANIA'})
    ukraine_map_raw = pd.merge(ukraine_map_raw, res, on='OBL_PROGIVANIA',
how='left')
    ukraine_map_raw = ukraine_map_raw.fillna(value=0)

    fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(20, 10),
gridspec_kw={'width_ratios': [1, 1]})

    ukraine_map_raw.plot(column='COUNT', cmap='coolwarm', legend=False,
ax=ax1)
    ax1.set_title('а) Розподіл проживання клієнтів по областях')
    ax1.set_xticklabels(ax1.get_xticklabels(), rotation=90)

    ukraine_map_raw = gpd.read_file('/kaggle/input/ukraine-
map/gadm41_UKR_1.shp')

    ukraine_map_raw = ukraine_map_raw.rename(columns={'NAME_1':
'OBL_PROGIVANIA'})
    ukraine_map_raw = pd.merge(ukraine_map_raw, res, on='OBL_PROGIVANIA',
how='left')
    ukraine_map_raw.plot(column='COUNT', cmap='coolwarm', legend=True,
ax=ax2)
    ax2.set_title('б) Розподіл проживання клієнтів по областях')
    ax2.set_xticklabels(ax2.get_xticklabels(), rotation=90)
```

```
plt.savefig('client_region_dist.png')

plt.show()


# In[39]:


# Робимо те саме тільки для області регістрації

df = pd.read_excel('/kaggle/input/scoring-data/NK_ScoringBuilding4_1600.xls')

regions_to_drop = ['Zakarpattia', 'Kharkiv', 'Chernivtsi', 'Luhansk']
mask = df['OBL_REGISTRACII'].isin(regions_to_drop)


ukraine_map = gpd.read_file('/kaggle/input/ukraine-map/gadm41_UKR_1.shp')
ukraine_map_raw = gpd.read_file('/kaggle/input/ukraine-map/gadm41_UKR_1.shp')

column_names = {'NAME_1': 'OBL_REGISTRACII'}

ukraine_map = ukraine_map.rename(columns=column_names)

mapping = {
    '?': None,
    'Cherkaska_region': 'Cherkasy',
    'Chernigivska_region': 'Chernihiv',
    'Chernivecka_region': 'Chernivtsi',
    'Krimea': 'Crimea',
    'Dnipropetrovska_region': "Dnipropetrovs'k",
    'Donecka_region': "Donets'k",
    'Ivano-Frankivska_region': "Ivano-Frankivs'k",
    'Kharkivska_region': 'Kharkiv',
    'Khmelnicka_region': "Khmel'nyts'kyy",
    'Kyiv_region': 'Kiev',
    'Kyiv': 'Kiev City',
    'Chersonska_region': 'Kherson',
```

```
            'Kirovogradska_region': 'Kirovohrad',
            'Lvivska_region': "L'viv",
            'Luganska_region': "Luhans'k",
            'Mykolaev_region': 'Mykolayiv',
            'Odeska_region': 'Odessa',
            'Poltavska_region': 'Poltava',
            'Rivnenska_region': 'Rivne',
            'Sevastopol': "Sevastopol'",
            'Sumska_region': 'Sumy',
            'Vinnicka_region': 'Vinnytsya',
            'Volinska_region': 'Volyn',
            'Zakarpatska_region': 'Zakarpattia',
            'Zaporizka_region': 'Zaporizhia',
            'Gitomirska_region': 'Zhytomyr'
        }

        print(df['OBL_REGISTRACII'].nunique())

        df['OBL_REGISTRACII'] = df['OBL_REGISTRACII'].map(mapping)
        print(df['OBL_REGISTRACII'].nunique())
        merged = pd.merge(ukraine_map, df, on='OBL_REGISTRACII')

        res = pd.DataFrame(merged['OBL_REGISTRACII'].value_counts())
        res = res.rename(columns=({'OBL_REGISTRACII': 'COUNT'}))
        res['OBL_REGISTRACII'] = res.index

        fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(20, 10),
gridspec_kw={'width_ratios': [1, 1.25]})

        ukraine_map_raw = gpd.read_file('/kaggle/input/ukraine-
map/gadm41_UKR_1.shp')
        ukraine_map_raw = ukraine_map_raw.rename(columns={'NAME_1':
'OBL_REGISTRACII'})
        ukraine_map_raw = pd.merge(ukraine_map_raw, res, on='OBL_REGISTRACII')
        ukraine_map_raw = ukraine_map_raw.fillna(value=0)
        ukraine_map_raw.plot(column='COUNT', cmap='coolwarm', ax=ax1,
legend=False)
        ax1.set_title('a) Розподіл реєстрації клієнтів по областях')
        ax1.tick_params(axis='x', labelrotation=90)
```

```
        ukraine_map_raw         =         gpd.read_file('/kaggle/input/ukraine-
map/gadm41_UKR_1.shp')
        ukraine_map_raw       =       ukraine_map_raw.rename(columns={'NAME_1':
'OBL_REGISTRACII'})
        ukraine_map_raw = pd.merge(ukraine_map_raw, res, on='OBL_REGISTRACII',
how='left')
        ukraine_map_raw.plot(column='COUNT',      cmap='coolwarm',      ax=ax2,
legend=True)
        ax2.set_title('б) Розподіл регістрації клієнтів по областях')

        plt.savefig('client_region_reg_dist_ab.png')
        plt.show()


        # In[40]:


        # Створюємо функцію для виведення карти в залежності від атрибута

        def make_map(arg):
            df              =              pd.read_excel('/kaggle/input/scoring-
data/NK_ScoringBuilding4_1600.xls')

            regions_to_drop   =   ['Chernigivska_region',   'Rivnenska_region',
'Krimea', 'Ivano-Frankivska_region']
            mask = df['OBL_PROGIVANIA'].isin(regions_to_drop)
            df = df[~mask]

            ukraine_map         =         gpd.read_file('/kaggle/input/ukraine-
map/gadm41_UKR_1.shp')
            ukraine_map_raw       =       gpd.read_file('/kaggle/input/ukraine-
map/gadm41_UKR_1.shp')

            column_names = {'NAME_1': 'OBL_PROGIVANIA'}

            ukraine_map = ukraine_map.rename(columns=column_names)

            mapping = {
                'Kyiv': 'Kiev City',
                'Kyiv_region': 'Kiev',
                'Vinnicka_region': 'Vinnytsya',
```

```
            'Poltavska_region': 'Poltava',
            'Lvivska_region': "L'viv",
            'Sevastopol': "Sevastopol'",
            'Krimea': 'Crimea',
            'Volinska_region': 'Volyn',
            'Rivnenska_region': 'Rivne',
            'Odeska_region': 'Odessa',
            'Khmelnicka_region': "Khmel'nyts'kyy",
            'Zaporizka_region': 'Zaporizhia',
            'Kirovogradska_region': 'Kirovohrad',
            'Chernigivska_region': 'Chernihiv',
            'Ivano-Frankivska_region': 'Ivano-Frankivs\'k',
            'Mykolaev_region': 'Mykolayiv'
        }

        df['OBL_PROGIVANIA'] = df['OBL_PROGIVANIA'].map(mapping)

        merged = pd.merge(ukraine_map, df, on='OBL_PROGIVANIA')

        res = pd.DataFrame(merged.groupby('OBL_PROGIVANIA')[arg].mean())
        res = res.rename(columns=({arg: 'COUNT'}))
        res['OBL_PROGIVANIA'] = res.index
        res = res.reset_index(drop=True)

        fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(20, 10),
gridspec_kw={'width_ratios': [1, 1.1]})
        ukraine_map_raw = gpd.read_file('/kaggle/input/ukraine-
map/gadm41_UKR_1.shp')
        ukraine_map_raw = ukraine_map_raw.rename(columns={'NAME_1':
'OBL_PROGIVANIA'})
        ukraine_map_raw = pd.merge(ukraine_map_raw, res,
on='OBL_PROGIVANIA', how='left')
        ukraine_map_raw = ukraine_map_raw.fillna(value=0)

        ukraine_map_raw.plot(column='COUNT', cmap='coolwarm', ax=ax1,
legend=False)

        ax1.set_title(f'a) Розподіл середнього значення {arg} клієнтів по
областях')
        ax1.set_axis_off()
```

```
            ukraine_map_raw       =       gpd.read_file('/kaggle/input/ukraine-
map/gadm41_UKR_1.shp')
            ukraine_map_raw     =     ukraine_map_raw.rename(columns={'NAME_1':
'OBL_PROGIVANIA'})
            ukraine_map_raw       =       pd.merge(ukraine_map_raw,       res,
on='OBL_PROGIVANIA', how='left')
            ukraine_map_raw.plot(column='COUNT',    cmap='coolwarm',    ax=ax2,
legend=True)
            ax2.set_title(f'б) Розподіл середнього значення {arg} клієнтів по
областях')
            ax2.set_axis_off()

            plt.savefig(f'client_region_{arg}_dist.png')
            plt.show()


        # In[41]:


        # Пробігаємось по атрибутах та виводимо для них карти
        for col in ['SROK', 'PLATEG', 'VOZRAST', 'CREDIT', 'KOL_DETEI', 'DOHOD',
'NEVOZVRAT_CREDITA']:
            make_map(col)


        # In[42]:

        (Пропущено)



        # Розділяємо колонки на категорійні, впорядковані та бінарні

        categorical_cols   =   ['TIP_PROGIV',   'OBRAZOVANIE',   'SOBSTVENNOST',
'DOLGNOST', 'OBL_REGISTRACII', 'OBL_PROGIVANIA']
        ordinal_cols = ['SROK_PROGIV', 'SROK_RABOTI', 'KOL_RABOTNIKOV']
        binary_cols = ['POL', 'SEM_POL']
        numerical_cols  =  ['SROK',  'PLATEG',  'VOZRAST',  'CREDIT',  'DOHOD',
'KOL_DETEI']

        # Впорядковуємо значення в колонках
```

```
ordinal_cols_srok_progiv = [['0,5-1year', '1-2years', '2-5years', '5-10_years', 'more_10_years']]
ordinal_cols_tip_progiv = [['less_0,5year', '0,5-1year', '1-2years', '2-5years', '5-10_years', 'more_10_years']]
ordinal_cols_kol_rabotnikov = [['0-5', '16-30', '31-50', '51-100', '6-15', 'more_100']]


# In[51]:


# Переводимо дані в числові

from sklearn.preprocessing import OneHotEncoder, OrdinalEncoder, LabelBinarizer
from scipy.sparse import hstack

df = pd.read_excel('/kaggle/input/scoring-data/NK_ScoringBuilding4_1600.xls')
display(df)

ohe = OneHotEncoder()
df_cat = ohe.fit_transform(df[categorical_cols])
df_cat = pd.DataFrame(df_cat.toarray(), columns=ohe.get_feature_names_out(categorical_cols))

display(df_cat)

oe_srok_progiv = OrdinalEncoder(categories=ordinal_cols_srok_progiv)
df['SROK_PROGIV'] = oe_srok_progiv.fit_transform(df['SROK_PROGIV'].values.reshape(-1, 1))

oe_srok_raboti = OrdinalEncoder()
df['SROK_RABOTI'] = oe_srok_raboti.fit_transform(df['SROK_RABOTI'].values.reshape(-1, 1))

oe_kol_rabotnikov = OrdinalEncoder(categories=ordinal_cols_kol_rabotnikov)
df['KOL_RABOTNIKOV'] = oe_kol_rabotnikov.fit_transform(df['KOL_RABOTNIKOV'].values.reshape(-1, 1))
```

```
display(df)

lb = LabelBinarizer()
for col in binary_cols:
    df[col] = lb.fit_transform(df[col])

df = pd.concat([df_cat, df.drop(columns=categorical_cols)], axis=1)

display(df)


# In[52]:


# Видаляємо викиди

import scipy

z_scores = np.abs(scipy.stats.zscore(df[numerical_cols]))

threshold = 3

df = df[(z_scores < threshold).all(axis=1)]


cols_to_plot = []
for col in df.columns:
    if col in ['OBL_REGISTRACII', 'OBL_PROGIVANIA']:
        continue
    if df[col].nunique() > 10:
        cols_to_plot.append(col)

fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(10, 8))

for i, col in enumerate(cols_to_plot):
    row = i // 2
    col_ = i % 2
    sns.histplot(df[col], ax=axs[row][col_])
    axs[row][col_].set_title(f'Distribution of {col}')
    axs[row][col_].set_xlabel(col)
```

```
            axs[row][col_].set_ylabel('Density')

        plt.tight_layout()
        plt.savefig('numerical_dist_after_z_score.png')
        plt.show()


        # In[53]:


        # Проводимо нормалізацію

        from sklearn.preprocessing import MinMaxScaler

        scaler = MinMaxScaler()

        df_norm = scaler.fit_transform(df)

        df = pd.DataFrame(df_norm, columns=df.columns)

        df


        # In[54]:


        # Видаляємо один атрибут щоб позбутись мультіколінеарності

        df = df.drop(columns='TIP_PROGIV_P')


        # In[61]:


        # Розбиваємо дані на цільові та тренувальні, проводимо розбивку на тест
та тренування

        from sklearn.model_selection import train_test_split

        X = df.drop('NEVOZVRAT_CREDITA', axis=1)
        y = df['NEVOZVRAT_CREDITA']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)


# In[62]:


# Свторюємо функцію для збалансування даних

def apply_resampling(X, y, type_='undersampled'):
    tl = TomekLinks()

    X_resampled, y_resampled = tl.fit_resample(X, y.astype(int))

    if type_ == 'hardness':
        smp = InstanceHardnessThreshold(estimator=LogisticRegression(),
cv=5)
    elif type_ == 'undersampled':
        smp = RandomUnderSampler()

    X_train, y_train = smp.fit_resample(X_resampled, y_resampled)
    return X_train, y_train

X_train, y_train = apply_resampling(X_train, y_train)


# In[63]:

(Пропущено)



# Визначаємо моделі

models = [
    ('LR', LogisticRegression()),
    ('LDA', LinearDiscriminantAnalysis()),
    ('KNN', KNeighborsClassifier()),
    ('CART', DecisionTreeClassifier()),
    ('NB', GaussianNB()),
```

```
            ('SVM', SVC(probability=True)),
            ('RF', RandomForestClassifier()),
            ('GBM', GradientBoostingClassifier()),
            ('ADA', AdaBoostClassifier()),
        ]

        results = []
        names = []

        # Збалансовуємо дані

        X = df.drop('NEVOZVRAT_CREDITA', axis=1)
        y = df['NEVOZVRAT_CREDITA']

        X_train, y_train = apply_resampling(X, y, type_='undersampled')

        # Проводимо тренування та оцінку роботи методом K-folds

        for name, model in models:
            kfold_accuracy = cross_val_score(model, X_train, y_train, cv=10,
scoring='accuracy').mean()
            kfold_f1 = cross_val_score(model, X_train, y_train, cv=10,
scoring='f1').mean()
            kfold_roc = cross_val_score(model, X_train, y_train, cv=10,
scoring='roc_auc').mean()
            results.append([kfold_accuracy, kfold_f1, kfold_roc])
            names.append(name)


        # In[65]:


        # Виводимо результати

        df_results = pd.DataFrame(results, columns=['Accuracy', 'F1 Score', 'ROC
AUC'])
        df_results['Model Name'] = names
        df_results = df_results[['Model Name', 'Accuracy', 'F1 Score', 'ROC
AUC']]
        df_results
```

```
# In[66]:


# Виводимо графік результатів

metrics = ['Accuracy', 'F1 Score', 'ROC AUC']

fig, ax = plt.subplots(2, 2, figsize=(10, 10))

axes = ax.flatten()

for i, metric in enumerate(metrics):
    data = [result[i] for result in results]

    data, names = zip(*sorted(zip(data, names)))

    x = np.arange(len(models))
    y = np.array(data)

    # Plot the bars
    axes[i].barh(x, y)
    axes[i].set_xlim([0,1.2])

    for j in range(len(models)):
        axes[i].text(y[j], j, f'{names[j]} {data[j]:.2f}', ha='left',
va='center', fontsize=14)

    axes[i].set_yticks(x)
    axes[i].set_yticklabels(names)

    axes[i].set_xlabel(metric)

fig.delaxes(axes[3])

plt.suptitle('Model Performance RandomUnserSampling', fontsize=16)

plt.subplots_adjust(wspace=0.5, right=1)

plt.savefig('models_perfomance_balanced_dataset_undersample.png')
```

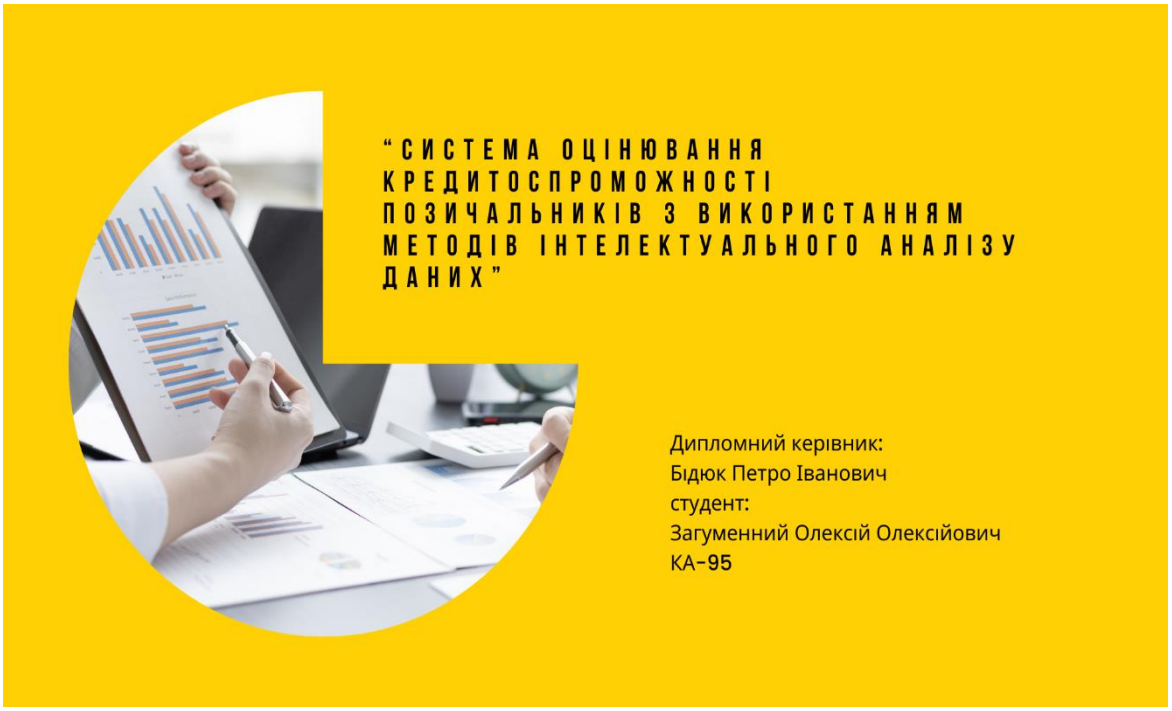
```
        plt.show()


        # In[ ]:

        (Пропущено)



        # In[ ]:


        # Виводимо матриці суперечностей для всіх моделей

        labels = ['Returned', 'Not returned']

        X = df.drop('NEVOZVRAT_CREDITA', axis=1)
        y = df['NEVOZVRAT_CREDITA']

        kf = KFold(n_splits=5, shuffle=True, random_state=42)

        fig, axs = plt.subplots(3, 3, figsize=(15, 15))

        for i, (name, model) in enumerate(models):
            print(f"Evaluating {name}...")
            cm_total = np.zeros((2, 2))
            for train_index, test_index in kf.split(X):
                X_train, X_test = X.iloc[train_index], X.iloc[test_index]
                y_train, y_test = y[train_index], y[test_index]

                # Збалансовуємо тренувальні дані
                X_train,    y_train    =    apply_resampling(X_train,    y_train,
type_='undersampled')

                model.fit(X_train, y_train)

                cm = confusion_matrix(y_test, model.predict(X_test), labels=[0,
1])
                percentages = cm / cm.sum(axis=1)[:, np.newaxis]
                cm_total += percentages
```

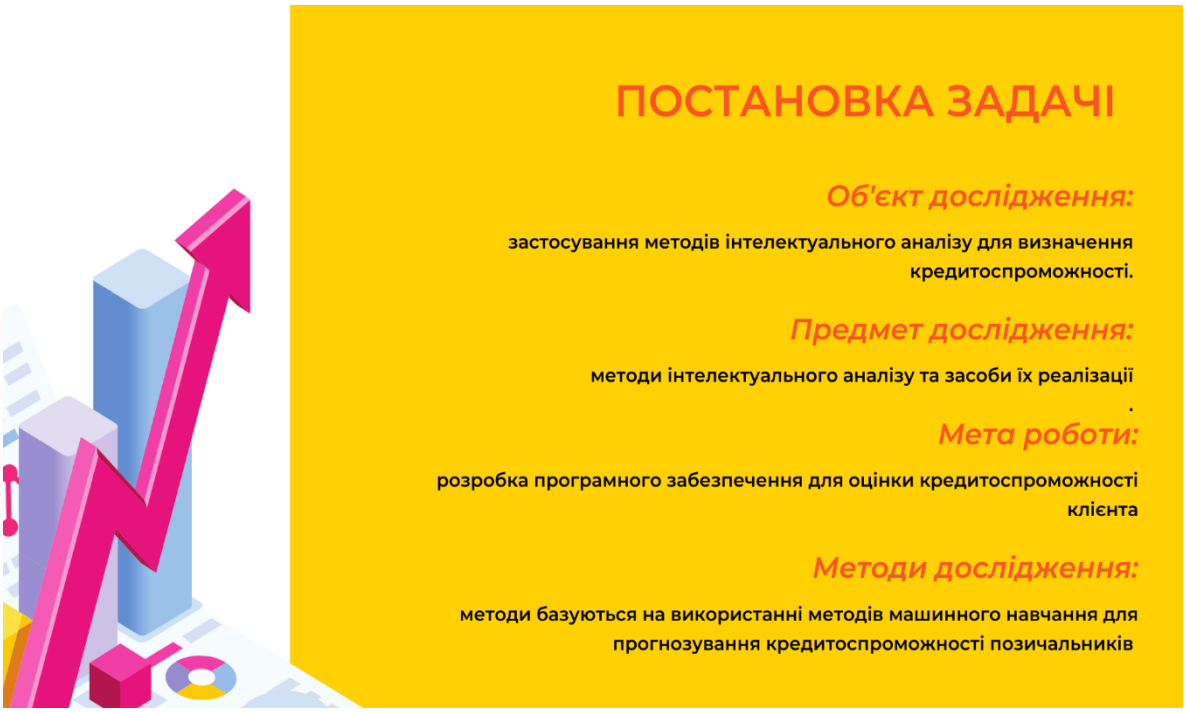
```
            cm_avg = cm_total / kf.get_n_splits()
            disp       =       ConfusionMatrixDisplay(confusion_matrix=cm_avg,
display_labels=labels)

            row = i // 3
            col = i % 3
            disp.plot(cmap=plt.cm.Blues, ax=axs[row, col], colorbar=False)
            axs[row, col].set_title(f'{name} RandomUnderSampling', fontsize=16)
            axs[row, col].set_xlabel('Predicted label', fontsize=12)
            axs[row, col].set_ylabel('True label', fontsize=12)
            axs[row, col].tick_params(axis='both', which='major', labelsize=12)

        plt.tight_layout()
        plt.savefig('confusion_matrices_all_undersampled.png')
        plt.show()


        # In[ ]:


        labels = ['Returned', 'Not returned']

        X = df.drop('NEVOZVRAT_CREDITA', axis=1)
        y = df['NEVOZVRAT_CREDITA']

        kf = KFold(n_splits=5, shuffle=True, random_state=42)

        fig, axs = plt.subplots(3, 3, figsize=(15, 15))
        sns.set(font_scale=2.0) # Adjust to fit

        for i, (name, model) in enumerate(models):
            print(f"Evaluating {name}...")
            cm_total = np.zeros((2, 2))
            for train_index, test_index in kf.split(X):
                X_train, X_test = X.iloc[train_index], X.iloc[test_index]
                y_train, y_test = y[train_index], y[test_index]

                # Збалансовуємо тренувальні дані
                X_train,    y_train    =    apply_resampling(X_train,    y_train,
type_='hardness')
```

```
                model.fit(X_train, y_train)

                cm = confusion_matrix(y_test, model.predict(X_test), labels=[0,
1])
                percentages = cm / cm.sum(axis=1)[:, np.newaxis]
                cm_total += percentages

            cm_avg = cm_total / kf.get_n_splits()
            disp       =       ConfusionMatrixDisplay(confusion_matrix=cm_avg,
display_labels=labels)

            row = i // 3
            col = i % 3
            disp.plot(cmap=plt.cm.Blues, ax=axs[row, col], colorbar=False)
            axs[row, col].set_title(f'{name} Hardness', fontsize=16)
            axs[row, col].set_xlabel('Predicted label', fontsize=12)
            axs[row, col].set_ylabel('True label', fontsize=12)
            axs[row, col].tick_params(axis='both', which='major', labelsize=12)

        plt.tight_layout()
        plt.savefig('confusion_matrices_all_hardness.png')
        plt.show()


        # In[ ]:


        labels = ['Returned', 'Not returned']

        X = df.drop('NEVOZVRAT_CREDITA', axis=1)
        y = df['NEVOZVRAT_CREDITA']

        kf = KFold(n_splits=5, shuffle=True, random_state=42)

        fig, axs = plt.subplots(3, 3, figsize=(15, 15))

        for i, (name, model) in enumerate(models):
            print(f"Evaluating {name}...")
            cm_total = np.zeros((2, 2))
            for train_index, test_index in kf.split(X):
                X_train, X_test = X.iloc[train_index], X.iloc[test_index]
```

```
                y_train, y_test = y[train_index], y[test_index]

                # Збалансування не проводимо

                model.fit(X_train, y_train)

                cm = confusion_matrix(y_test, model.predict(X_test), labels=[0,
1])
                percentages = cm / cm.sum(axis=1)[:, np.newaxis]
                cm_total += percentages

            cm_avg = cm_total / kf.get_n_splits()
            disp       =       ConfusionMatrixDisplay(confusion_matrix=cm_avg,
display_labels=labels)

            row = i // 3
            col = i % 3
            disp.plot(cmap=plt.cm.Blues, ax=axs[row, col], colorbar=False)
            axs[row, col].set_title(f'{name} Imbalanced', fontsize=16)
            axs[row, col].set_xlabel('Predicted label', fontsize=12)
            axs[row, col].set_ylabel('True label', fontsize=12)
            axs[row, col].tick_params(axis='both', which='major', labelsize=12)

        plt.tight_layout()
        plt.savefig('confusion_matrices_all_imbalanced.png')
        plt.show()
```

ДОДАТОК Б ПРЕЗЕНТАЦІЯ



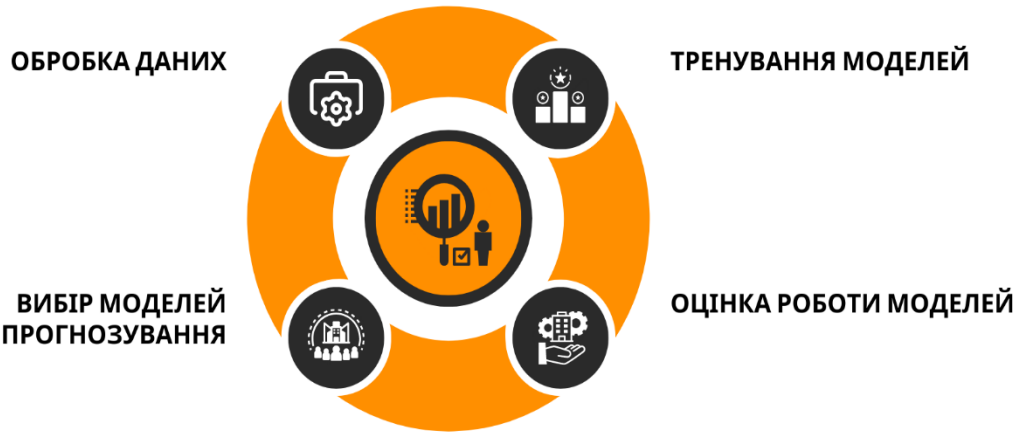


АКУАЛЬНІСТЬ ТЕМИ ДОСЛІДЖЕННЯ

Тема роботи має високу актуальність у сучасних умовах фінансової індустрії. Оцінка кредитоспроможності є однією з найважливіших процедур, що здійснюється банками та іншими фінансовими установами перед наданням кредитів та інших видів фінансових послуг.

Зрозуміло, що традиційні методи оцінки кредитоспроможності, такі як рейтингові системи та оцінка досвідченими працівниками, мають свої обмеження та можуть бути недостатньо ефективними, особливо при обробці великих обсягів даних. У таких умовах, використання методів інтелектуального аналізу даних може забезпечити значно більш точну та ефективну оцінку кредитоспроможності позичальників.

Кроки розробки моделей прогнозування



ОБРОБКА ДАНИХ: ЕТАПИ

Декодування даних

Для використання категоріальних змінних у машинному навчанні та аналітиці.

Видалення викидів

Для уникнення аномальних значень, які виходять за межі діапазону очікуваних значень.

Нормалізація

Для досягнення порівняності між різними ознаками та забезпечення стабільності та ефективності алгоритмів машинного навчання.

Збалансування

Для урівноваження пропорцій класів у наборі даних для покращення точності та об'єктивності моделі машинного навчання.

Обробка даних: збалансування – створені датасети

Random Under Sampling: датасет зі зменшенною кількістью зразків більш представленого класу з метою збалансування даних та покращення ефективності моделі. Датасет отримали видаляючи випадковим чином зразки з класу більшості.

Instance Hardness Threshold: датасет в якому використовувалося пороговоге значення складності екземплярів для видалення менш важливих або шумових зразків з набору даних з метою покращення якості моделі.

Imbalanced Dataset: оригінальний датасет без застосування методів збалансування.

Вибір моделей

Вибір моделей для дослідження обґрунтовується їх високою точністю та ефективністю в розв'язанні подібних завдань, що підтверджено результатами проведених наукових досліджень та відгуками відповідних фахівців.
Список моделей:

1. Логістична регресія
2. Лінійний дискримінантний аналіз
3. К-найближчих сусідів
4. Дерева рішень
5. Наївний байєсівський класифікатор
6. Метод опорних векторів
7. Випадковий ліс
8. Градієнтний бустинг
9. Адаптивний бустинг

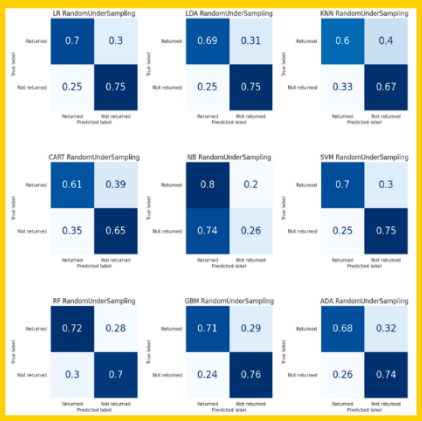
Тренування моделей: метрики

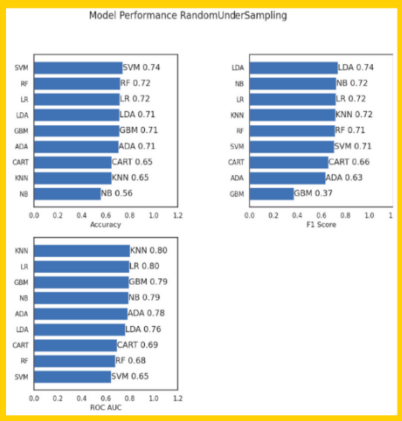
1. **F1-score**: Метрика для оцінювання балансу між точністю і повнотою для досягнення надійної та збалансованої оцінки моделі в умовах незбалансованих класів.

2. **Accuracy**: Метрика для вимірювання точності класифікаційної моделі шляхом обчислення відношення правильно класифікованих прикладів до загальної кількості прикладів.

3. **ROC-AUC**: Метрика, що оцінює якість моделі класифікації, вимірюючи площу під кривою ROC

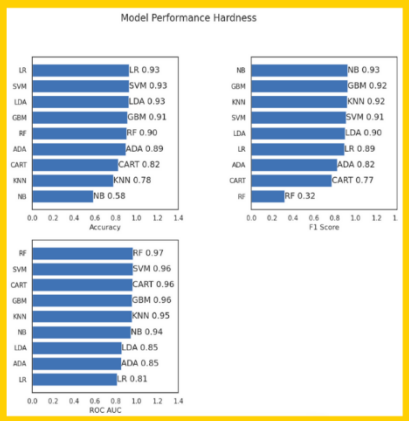
Тренування моделей: результати Random Under Sampling





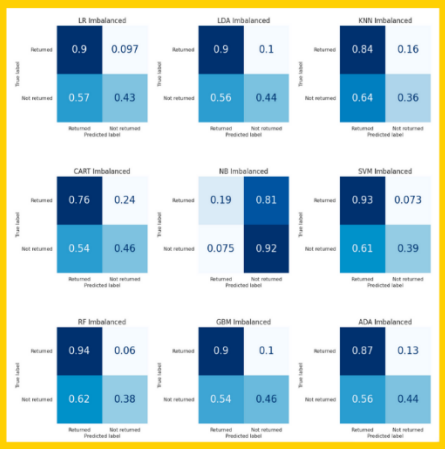
Моделі показують себе доволі вдало на тестових даних. Найкращі результати вийшли у логістичної регресії, випадкових лісів, машин підтримки векторів та класифікатора к-найближчих сусідів.

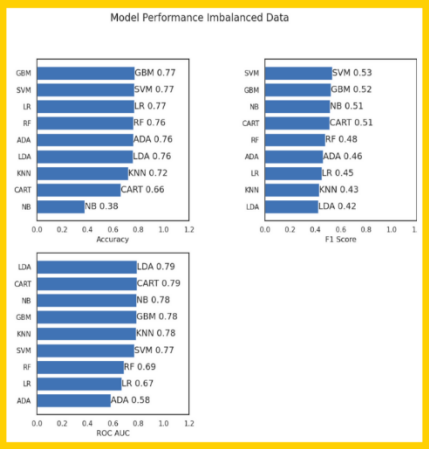
Тренування моделей: результати Instance Hardness Threshold



Результати метрик в даному випадку вийшли значно краще, ніж в минулому випадку, проте моделі вийшли значним чином упередженими в сторону одного з класів.

Тренування моделей: результати Imbalanced Data





На цьому датасеті моделі показали результати вище, ніж при Random Under Sampling, проте результати вийшли значно упередженими в сторону більшого класу.

Рекомендації щодо застосування розроблених систем

При виборі моделі для оцінки кредитного скорингу, необхідно враховувати кілька важливих показників, які можуть вплинути на прибутковість банку. Наприклад, при виборі моделі варто оцінити суму втрати від одного клієнта, що може бути спричинена невідшкодованим кредитом або збитками в разі порушення договору кредиту.

Також варто враховувати суму надходження від одного клієнта, яка може бути забезпечена процентами за кредитом, а також ціну отримання одного клієнта, яка може бути пов'язана з рекламними витратами та витратами на обслуговування клієнтів.

Висновки

В результаті тренування та оцінки моделей було розглянуто 9 різних моделей, що були натреновані на трьох різних датасетах із використанням методу валідації K-fold. Найбільш точними вийшли моделі логістичної регресії та випадкових лісів натреновані на збалансованому наборі даних із використанням методу TomeLinks та RandomUnderSampling.

Найбільш успішними з точки зору принесення прибутку для банків виявились моделі SVM та випадкові ліси. Вони показали точність передбачень для неповернення кредиту в 86% та точність у передбаченні надійних клієнтів у 57%, що може значним чином збільшити результати банкінгу в Україні.

ДЯКУЮ
ЗА УВАГУ!