

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «магістр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

**«Розробка додатку для підвищення довіри до
прогнозів фінансових моделей за допомогою
методів Explainable AI / XAI»**

Виконав студент групи ІПЗм-24-1
ГОЛОВЕЦЬКИЙ Євгеній Володимирович

Керівник роботи:
МОРОЗОВ Андрій Васильович

Рецензент:
ЄФРЕМОВ Юрій Миколайович

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

зав. кафедри інженерії
програмного забезпечення
Тетяна ВАКАЛЮК
«18» вересня 2025 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ГОЛОВЕЦЬКИЙ Євгеній Володимирович**

Керівник роботи: **МОРОЗОВ Андрій Васильович**

Тема роботи: **«Розробка додатку для підвищення довіри до прогнозів фінансових моделей за допомогою методів Explainable AI / XAI»**,

затверджена Наказом закладу вищої освіти від «18» вересня 2025 р. № 491/с

Вихідні дані для роботи: **аналіз проблематики «чорної скриньки» у фінансовому секторі та вимог до прозорості алгоритмів, методи пояснюваного штучного інтелекту, зокрема адитивні пояснення та контрфактуальний аналіз, набір даних для кредитного скорингу.**

Консультанти з магістерської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Морозов А.В.	22.09.2025р.	22.09.2025р.
2	Морозов А.В.	22.09.2025р.	22.09.2025р.
3	Морозов А.В.	22.09.2025р.	22.09.2025р.

РЕФЕРАТ

Кваліфікаційна робота освітнього ступеня «магістр» присвячена вирішенню проблеми інтерпретації рішень алгоритмів машинного навчання у фінансовій сфері. Пояснювальна записка складається зі вступу, трьох розділів, висновків та списку використаної літератури. Пояснювальна записка до кваліфікаційної магістерської роботи містить 60 сторінок, 20 ілюстрацій, 6 таблиць, 7 додатків та 21 посилання на зовнішні джерела.

Метою роботи є розробка веб-додатку для оцінки кредитоспроможності клієнтів, що забезпечує прозорість прийняття рішень та надає автоматичні рекомендації для користувачів.

В роботі описано процес створення програмного засобу, який використовує ансамблевий алгоритм Random Forest для прогнозування ризиків та метод адитивних пояснень SHAP для візуалізації впливу факторів. Розроблено зручний графічний інтерфейс користувача на базі фреймворку Streamlit. Досліджено методи контрфактуального аналізу для автоматичної генерації порад у разі відмови у кредитуванні. Проведено функціональне тестування розробленої системи.

КЛЮЧОВІ СЛОВА: ПОЯСНЮВАНИЙ ШТУЧНИЙ ІНТЕЛЕКТ, КРЕДИТНИЙ СКОРИНГ, МАШИННЕ НАВЧАННЯ, SHAP, RANDOM FOREST, КОНТРФАКТУАЛЬНИЙ АНАЛІЗ.

ABSTRACT

The master's thesis is devoted to solving the problem of interpreting machine learning algorithm decisions in the financial sector.

The explanatory note consists of an introduction, three chapters, conclusions, and a list of references. The explanatory note to the master's qualification paper contains 60 pages, 15 illustrations, 6 tables, 7 appendices, and 21 references to external sources.

The aim of the work is to develop a web application for assessing customer creditworthiness, which ensures transparency of decision-making and provides automatic recommendations for users.

The paper describes the process of creating a software tool that uses the Random Forest ensemble algorithm for risk prediction and the SHAP additive explanation method for visualizing the impact of factors. A user-friendly graphical interface based on the Streamlit framework has been developed. Methods of counterfactual analysis for the automatic generation of advice in case of credit denial have been investigated. Functional testing of the developed system was conducted.

KEYWORDS: EXPLAINABLE ARTIFICIAL INTELLIGENCE, CREDIT SCORING, MACHINE LEARNING, SHAP, RANDOM FOREST, COUNTERFACTUAL ANALYSIS.

ЗМІСТ

ВСТУП	7
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ	9
1.1 АНАЛІЗ ПРОБЛЕМ ВИКОРИСТАННЯ МАШИННОГО НАВЧАННЯ У ФІНАНСОВОМУ СКОРИНГУ ТА НЕОБХІДНІСТЬ ІНТЕРПРЕТАЦІЇ	9
1.2 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ТА ІНСТРУМЕНТІВ ПОЯСНЮВАННЯ ІШТУЧНОГО ІНТЕЛЕКТУ	12
1.3 ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ	14
1.4 ВИБІР ЗАСОБІВ РЕАЛІЗАЦІЇ	15
Висновки до розділу 1	17
РОЗДІЛ 2. ПРОЄКТУВАННЯ СИСТЕМИ ОЦІНКИ КРЕДИТНИХ РИЗИКІВ З ВИКОРИСТАННЯМ МЕТОДІВ ХАІ	19
2.1 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ ТА АЛГОРИТМІЧНА ОСНОВА СИСТЕМИ..	19
2.2 ПІДГОТОВКА ТА ПОПЕРЕДНЯ ОБРОБКА ДАНИХ	22
2.3 ПРОЄКТУВАННЯ АРХІТЕКТУРИ ПРОГРАМНОГО ЗАСОБУ	25
Висновки до розділу 2	30
РОЗДІЛ 3. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ТА АНАЛІЗ РЕЗУЛЬТАТІВ РОБОТИ СИСТЕМИ.....	31
3.1 РЕАЛІЗАЦІЯ ГРАФІЧНОГО ІНТЕРФЕЙСУ КОРИСТУВАЧА	31
3.2 АНАЛІЗ РЕЗУЛЬТАТІВ ПРОГНОЗУВАННЯ ТА ІНТЕРПРЕТАЦІЇ.....	33
3.3 РОБОТА МОДУЛЯ АВТОМАТИЧНИХ РЕКОМЕНДАЦІЙ	38
3.4 АНАЛІЗ ВПЛИВУ ГІПЕРПАРАМЕТРІВ НА ЕФЕКТИВНІСТЬ ПРОГНОЗНОЇ МОДЕЛІ.....	40
3.5 КОНТЕЙНЕРИЗАЦІЯ ТА РОЗГОРАННЯ ПРОГРАМНОГО ПРОДУКТУ	43
Висновки до розділу 3	45
ВИСНОВКИ.....	47
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	49
ДОДАТКИ.....	52

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

AI – (англ. Artificial Intelligence) – Штучний інтелект.

ML – (англ. Machine Learning) – Машинне навчання.

XAI – (англ. Explainable Artificial Intelligence) – Пояснюваний штучний інтелект.

SHAP – (англ. SHapley Additive exPlanations) – Метод адитивних пояснень Шеплі.

LIME – (англ. Local Interpretable Model-agnostic Explanations) – Локальні інтерпретовані пояснення, незалежні від моделі.

GDPR – (англ. General Data Protection Regulation) – Загальний регламент про захист даних ЄС.

UML – (англ. Unified Modeling Language) – Уніфікована мова моделювання.

API – (англ. Application Programming Interface) – Прикладний програмний інтерфейс.

UI – (англ. User Interface) – Інтерфейс користувача.

UX – (англ. User Experience) – Досвід користувача (взаємодія користувача).

IDE – (англ. Integrated Development Environment) – Інтегроване середовище розробки.

CSV – (англ. Comma-Separated Values) – Текстовий формат, призначений для представлення табличних даних.

ВСТУП

Актуальність теми. Сучасний фінансовий сектор переживає стрімку цифрову трансформацію, ключовим драйвером якої є технології штучного AI та машинного навчання. Алгоритми все частіше використовуються для автоматизації критично важливих процесів, таких як кредитний скоринг, виявлення шахрайства та оцінка інвестиційних ризиків. Використання складних нелінійних моделей наприклад, ансамблевих методів або нейронних мереж, дозволяє досягти високої точності прогнозів, проте породжує проблему «чорної скриньки», коли логіка прийняття рішення залишається непрозорою для користувачів.

Відсутність інтерпретованості результатів створює бар'єр недовіри з боку клієнтів та регуляторів. В умовах посилення нормативних вимог, зокрема впровадження Акту про штучний інтелект ЄС, фінансові установи зобов'язані забезпечити прозорість автоматизованих рішень та надати клієнтам обґрунтування відмов. Існуючі програмні засоби часто фокусуються лише на точності прогнозу, ігноруючи аспект пояснюваності. Тому розробка програмного забезпечення, яке не лише прогнозує ризики, а й пояснює причини прийняття рішень за допомогою методів XAI, є актуальним науково-практичним завданням інженерії програмного забезпечення.

Мета роботи полягає у розробці веб-додатку для підвищення довіри до прогнозів фінансових моделей шляхом інтеграції методів пояснюваного штучного інтелекту, що забезпечить прозорість процесу прийняття рішень у сфері кредитного скорингу.

Для досягнення поставленої мети необхідно вирішити такі завдання:

1. Провести аналіз предметної області та існуючих підходів до інтерпретації моделей машинного навчання у фінансах.
2. Обґрунтувати вибір методів XAI, а саме SHAP, та технологічного стеку для реалізації системи.

3. Спроекувати архітектуру програмного засобу, розробити структуру бази даних та алгоритми взаємодії компонентів.
4. Здійснити підготовку даних та навчання моделі машинного навчання для оцінки кредитоспроможності.
5. Розробити програмний додаток з графічним інтерфейсом, що візуалізує вплив факторів на прогноз та надає рекомендації користувачу.
6. Провести тестування розробленого програмного продукту та оцінити його ефективність.

Об'єктом дослідження є процес автоматизованого прийняття рішень у фінансових системах на основі методів машинного навчання.

Предметом дослідження є методи та інструменти ХАІ для інтерпретації та візуалізації прогнозів моделей кредитного скорингу.

Методи дослідження. У роботі використано методи системного аналізу для визначення вимог до системи, об'єктно-орієнтованого програмування для реалізації програмних модулів, методи машинного навчання для побудови прогнозної моделі, метод адитивних пояснень для інтерпретації результатів, експериментальний метод для перевірки коректності роботи додатку.

Наукова новизна одержаних результатів полягає у вдосконаленні підходу до взаємодії користувача з фінансовими АІ-системами шляхом поєднання локальних пояснень з алгоритмом контрфактуального аналізу, що дозволяє не лише пояснити причину відмови, а й автоматично генерувати рекомендації для зміни результату на позитивний, тим самим підвищуючи рівень довіри до системи.

Практичне значення роботи полягає у створенні діючого прототипу програмного забезпечення «Credit XAI System», який може бути використаний фінансовими установами для автоматизації процесу кредитування з дотриманням вимог прозорості. Розроблений додаток дозволяє кредитним менеджерам аргументувати рішення перед клієнтами, а аналітикам – виявляти можливі упередження в роботі моделей.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз проблем використання машинного навчання у фінансовому скорингу та необхідність інтерпретації

Сучасний фінансовий сектор характеризується експоненційним зростанням обсягів даних та необхідністю прийняття рішень у режимі реального часу. Ключовим інструментом для оцінки ризиків, зокрема кредитоспроможності позичальників, став кредитний скоринг. Традиційні підходи до скорингу базувалися на статистичних методах, таких як логістична регресія або прості дерева рішень. Ці моделі є повністю інтерпретованими: експерт може чітко простежити вплив кожного коефіцієнта на фінальний результат.

З розвитком технологій штучного інтелекту фінансові установи почали масово імплементувати складні алгоритми ML, такі як ансамблеві методи та глибокі нейронні мережі. Використання цих методів дозволило значно підвищити точність прогнозів та автоматизувати процес видачі кредитів. Однак, зростання складності моделей призвело до виникнення фундаментальної проблеми, відомої як проблема «чорної скриньки».

Суть проблеми полягає в тому, що внутрішня логіка прийняття рішення складною моделлю є непрозорою для людини. Модель перетворює вхідні дані на вихідний результат через мільйони математичних операцій, які неможливо інтерпретувати в термінах предметної області (рис. 1.1).

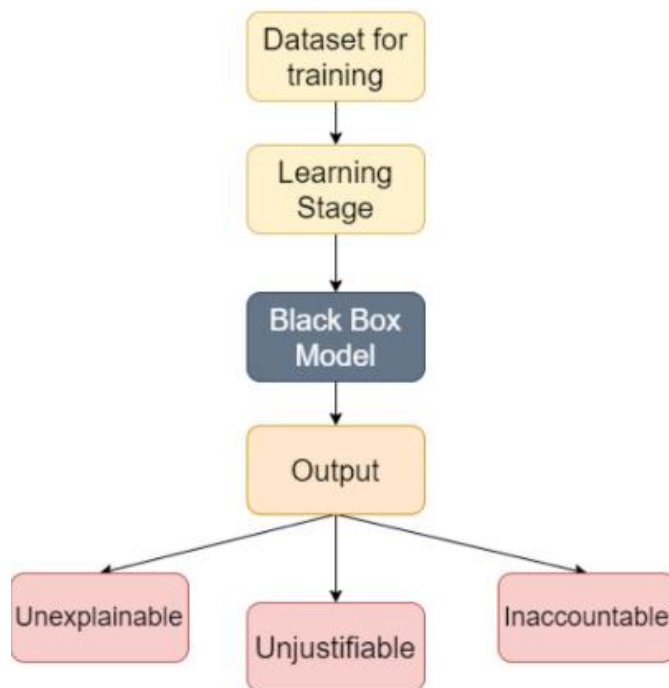


Рис. 1.1. Концептуальна схема моделі «чорної скриньки».

Необхідність інтерпретації моделей у фінансах сьогодні диктується не лише бізнес-потребами, а й суворими законодавчими нормами. Ключовим документом є Загальний регламент про захист даних ЄС[1], стаття 22 якого прямо зазначає право суб'єкта даних на отримання пояснень щодо рішень, прийнятих виключно в автоматизованому режимі.

Особливої актуальності тема набула з прийняттям Європейським Парламентом «Акту про штучний інтелект» [2]. Згідно з цим документом, системи, що використовуються для оцінки кредитоспроможності фізичних осіб, класифікуються як системи високого ризику. Це накладає на фінансові установи зобов'язання забезпечити високий рівень прозорості функціонування системи, можливість людського нагляду за прийняттям рішень, детальну технічну документацію, що пояснює логіку роботи AI-моделі.

Невиконання цих вимог несе не лише репутаційні ризики, а й загрожує значними фінансовими штрафами, що робить розробку методів пояснюваного штучного інтелекту критично важливим завданням інженерії програмного забезпечення.

Відсутність інтерпретованості породжує глибоку кризу довіри з боку ключових груп стейкхолдерів, починаючи від клієнтів банку, які у разі відмови

у наданні кредиту мають законне право знати конкретну причину, оскільки абстрактна відповідь про рішення алгоритму є цілком незадовільною та призводить до значних репутаційних втрат фінансової установи.

З точки зору регуляторних органів, сучасні нормативні вимоги, зокрема GDPR та Акт про ШІ ЄС, зобов'язані забезпечити «право на пояснення», оскільки використання непрозорих моделей високого ризику несе за собою небезпеку дискримінації та прийняття упереджених рішень на основі чутливих даних. Важливу роль у забезпеченні прозорості відіграють ризик-менеджери, які повинні мати можливість верифікувати внутрішню логіку моделі, щоб переконатися, що вона базується на об'єктивних факторах, а не на хибних кореляціях. Нарешті, для розробників та аналітиків даних інтерпретованість стає критичним інструментом для налагодження, верифікації та ітеративної оптимізації алгоритмів, що дозволяє виявляти приховані упередження, перевіряти стабільність прогнозів та гарантувати відповідність системи технічним стандартам через детальне документування логіки прийняття рішень, що є необхідною умовою для успішного проходження аудиту та підтвердження комплаєнсу.

Впровадження ML-моделей завжди супроводжується пошуком балансу між точністю прогнозу та можливістю його пояснити. Це явище в науковій літературі називають компромісом точності та інтерпретованості[3]. Як видно з рис. 1.2, зі збільшенням складності алгоритму його точність зростає, але здатність людини зрозуміти логіку прийняття рішень стрімко падає.

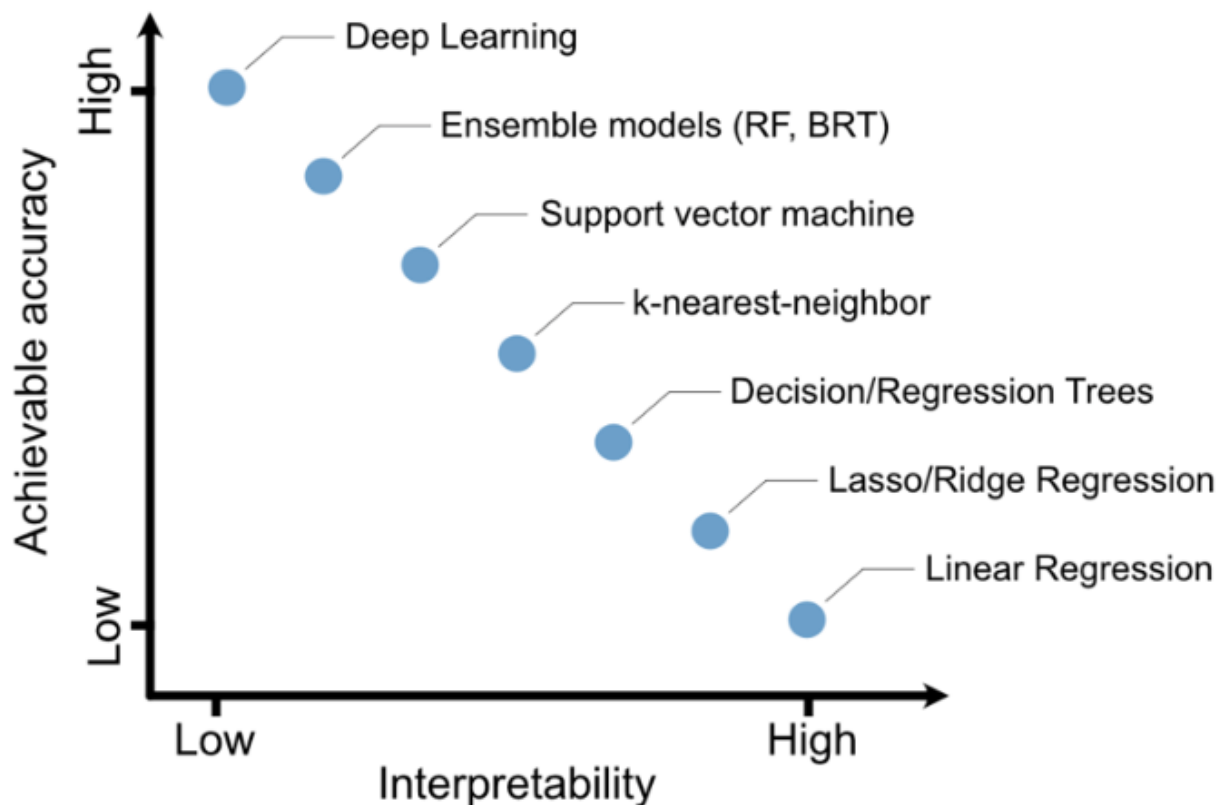


Рис. 1.2. Компроміс між точністю та інтерпретованістю моделей машинного навчання.

Саме для подолання цього розриву виник напрямок пояснюваного штучного інтелекту. Методи XAI дозволяють використовувати потужні моделі такі як Random Forest, обраний для цієї роботи, забезпечуючи при цьому рівень прозорості, достатній для прийняття відповідальних фінансових рішень.

1.2 Аналіз існуючих методів та інструментів пояснюваного штучного інтелекту

Для вирішення проблеми «чорної скриньки» в останні роки було розроблено низку методів та алгоритмів, які об'єднуються під загальною назвою Explainable AI або XAI[4]. Вибір конкретного методу для фінансового додатку залежить від вимог до точності, швидкодії та характеру пояснень (глобальні чи локальні).

Класифікацію методів XAI наведено на рис. 1.3.

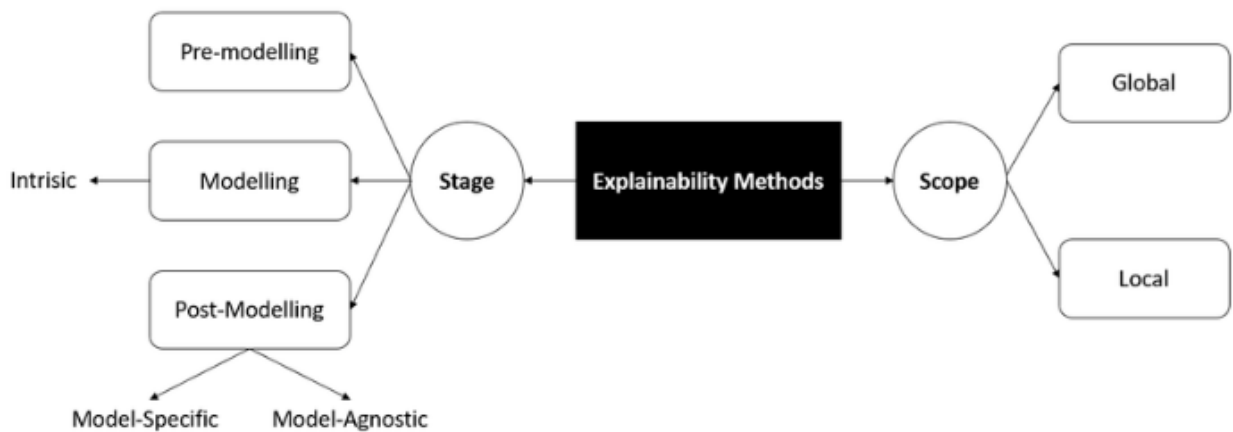


Рис. 1.3. Класифікація методів XAI.

Напрямок Explainable AI (XAI) фокусується на створенні методів, які дозволяють зробити рішення складних моделей зрозумілими для кінцевих користувачів. Існуючі підходи можна класифікувати за кількома ознаками:

- Intrinsic - моделі, що є прозорими за своєю структурою (лінійна регресія, прості дерева рішень).
- Post-modelling - методи, що застосовуються до вже навчених складних моделей для їх інтерпретації.
- Global - пояснюють загальну логіку роботи моделі на всій вибірці даних
- Local - пояснюють причину конкретного прогнозу для окремого клієнта.

Серед універсальних методів, які найбільше підходять для фінансового сектору, виділяють два основні підходи: **LIME**[5] та **SHAP**[6].

Метод LIME (Local Interpretable Model-agnostic Explanations) базується на ідеї апроксимації складної моделі простою лінійною моделлю в околі конкретної точки даних. Хоча LIME є швидким, він має суттєвий недолік - нестабільність. Для двох дуже схожих клієнтів LIME може видати різні пояснення, що є неприпустимим у банківській сфері, де вимагається консистентність рішень.

Метод SHAP (SHapley Additive exPlanations) базується на теорії ігор (вектор Шеплі). Він розподіляє різницю між прогнозом моделі та середнім значенням між усіма ознаками-гравцями. Головною перевагою SHAP є математично доведена єдність та узгодженість пояснень. Саме тому SHAP

вважається «золотим стандартом» для задач, де критично важлива точність інтерпретації, зокрема у кредитному скорингу.

Порівняльний аналіз цих методів наведено в таблиці 1.1.

Таблиця 1.1

Порівняльна характеристика методів інтерпретації моделей

Критерій порівняння	LIME	SHAP	Feature Importance
Тип пояснення	Локальний	Локальний та Глобальний	Глобальний
Теоретична база	Локальна апроксимація	Теорія ігор	Зменшення ентропії
Стабільність результатів	Низька (може змінюватись)	Висока	Висока
Швидкість обчислення	Висока	Низька	Дуже висока
Візуалізація впливу (+/-)	Так	Так	Ні

Окрім бібліотек, існують комерційні програмні продукти (наприклад, рішення від SAS або IBM Watson OpenScale), які інтегрують функції XAI[7]. Проте їхніми недоліками є висока вартість ліцензування та закритий код, що ускладнює кастомізацію під специфічні потреби банку. Більшість же існуючих open-source інструментів (бібліотеки Python) розраховані на технічних спеціалістів і не мають зручного графічного інтерфейсу для кінцевих користувачів – кредитних менеджерів.

1.3 Постановка задачі дослідження

Проведений у попередній частині аналіз дозволяє виявити основні прогалини в існуючих підходах до автоматизації кредитного скорингу. Ключовим недоліком сучасних скорингових систем є їхня «непрозорість»: вони надають лише фінальний результат, залишаючи поза увагою причини його формування. Водночас, існуючі інструменти інтерпретації такі як бібліотека SHAP є вузькоспеціалізованими програмними компонентами,

розрахованими на Data Scientist-ів, і не мають зручного інтерфейсу для кінцевих бізнес-користувачів.

При постановці задачі слід враховувати, що кредитний менеджер, який працює з клієнтом, не повинен розбиратися в коді або математичних векторах. Йому необхідний інструмент, який візуалізує складні дані у зрозумілій формі.

Таким чином, основною задачею дослідження є розробка веб-додатку, який поєднує точність алгоритмів машинного навчання з прозорістю методів ХАІ та інтерактивністю користувацького інтерфейсу.

Для спрощення процесу проєктування та реалізації, розробку програмного засобу доцільно розділити на чотири етапи:

На першому етапі необхідно здійснити підготовку та попередню обробку даних на основі German Credit Dataset, провести відбір ключових ознак та навчити модель класифікації.

На другому етапі слід розробити модуль інтерпретації, який інтегруватиме алгоритм SHAP для розрахунку локальних та глобальних пояснень.

На третьому етапі потрібно спроектувати та реалізувати графічний UI, який дозволить зручно вводити дані анкети та переглядати результати аналізу.

На четвертому етапі варто реалізувати алгоритм контрфактуального аналізу для генерації автоматичних рекомендацій у разі відмови.

1.4 Вибір засобів реалізації

Вибір технологічного стеку для розробки системи обумовлений необхідністю ефективної обробки даних, забезпечення високої точності прогнозування та реалізації інтерактивного користувацького інтерфейсу. Базовим інструментом розробки обрано мову програмування **Python**, яка є загальновизнаним стандартом у сфері Data Science завдяки розвиненій екосистемі бібліотек та підтримці сучасних алгоритмів машинного навчання.

Для попередньої обробки та маніпулювання табличними даними використовується бібліотека **Pandas**, яка дозволяє ефективно працювати зі

структурованими наборами даних виконувати очищення та трансформацію ознак. Побудова прогнозової моделі здійснюється за допомогою бібліотеки **Scikit-learn**[8]. В якості основного алгоритму класифікації обрано **Random Forest**. Цей ансамблевий метод демонструє високу стійкість до перенавчання та забезпечує кращу точність на табличних даних порівняно з простими лінійними моделями, при цьому не вимагаючи значних обчислювальних ресурсів, характерних для глибоких нейронних мереж.

Ключовим компонентом системи, що відповідає за інтерпретацію рішень, обрано бібліотеку **SHAP**. Її використання дозволяє гарантувати математичну узгодженість пояснень, що є критичним для фінансових застосунків. На відміну від альтернативних методів, таких як LIME, SHAP надає можливість генерувати як локальні пояснення для окремих клієнтів, так і глобальну аналітику важливості факторів для моделі в цілому.

Реалізація графічного інтерфейсу користувача виконується на базі фреймворку **Streamlit**[9]. Цей інструмент дозволяє створювати повноцінні веб-додатки безпосередньо мовою Python, оминаючи необхідність використання HTML, CSS або JavaScript. Такий підхід значно прискорює процес розробки прототипу та спрощує інтеграцію ML-моделей у візуальне середовище. Середовищем розробки обрано **Visual Studio Code**, що забезпечує зручне написання коду, налагодження та управління версіями проєкту.

Для розширення обраного технологічного стеку та забезпечення повноцінного функціонування системи в єдиному середовищі розробки також було залучено низку допоміжних інструментів і бібліотек. Зокрема, важливу роль відіграє бібліотека **NumPy**, яка виступає фундаментом для більшості обчислювальних операцій у системі, дозволяючи працювати з багатовимірними масивами та забезпечуючи високу швидкість математичних розрахунків, що є необхідним для підготовки синтетичних даних та коректної роботи алгоритмів інтерпретації. Для візуального представлення результатів аналізу, окрім вбудованих засобів SHAP, застосовується бібліотека

Matplotlib, яка надає гнучкі можливості для кастомізації графіків, налаштування їхньої стилістики та підготовки високоякісних ілюстрацій для виведення в інтерфейс Streamlit.

Питання персистентності навченої моделі та її швидкого відновлення вирішується за допомогою інструменту Joblib або стандартного модуля серіалізації Pickle, що дозволяє зберігати об'єкт Random Forest у файловій системі у вигляді серіалізованого файлу з розширенням .pkl. Такий підхід унеможливорює необхідність повторного навчання моделі при кожному запуску додатку, що критично важливо для забезпечення миттєвого відгуку системи на запити користувача.

З метою забезпечення стабільності розгортання та відтворюваності середовища виконання передбачено можливість контейнеризації всього програмного комплексу за допомогою технології **Docker**. Це дозволяє упакувати додаток разом із усіма його залежностями, специфічними версіями бібліотек та конфігураційними файлами в ізольований образ, гарантуючи ідентичність роботи системи як на локальних машинах розробників, так і на віддалених серверах фінансових установ.

Використання такої комплексної інфраструктури дозволяє створити надійний, масштабований та легкий у підтримці програмний продукт, що відповідає найвищим стандартам сучасної інженерії програмного забезпечення.

Висновки до розділу 1

У першому розділі проведено комплексний аналіз предметної області, пов'язаної з використанням штучного інтелекту у фінансовому секторі. Встановлено, що впровадження складних моделей машинного навчання для кредитного скорингу супроводжується проблемою «чорної скриньки», що призводить до недовіри з боку клієнтів та ризиків невідповідності регуляторним нормам.

Аналіз існуючих методів пояснюваного штучного інтелекту (ХАІ) показав, що найбільш ефективним підходом для вирішення окресленої проблеми є використання адитивних пояснень Шеплі (SHAP) завдяки їхній стабільності та теоретичній обґрунтованості. Водночас виявлено відсутність на ринку доступних програмних рішень, які б поєднували потужність сучасних алгоритмів ХАІ зі зручним інтерфейсом для нетехнічних спеціалістів.

На основі проведеного аналізу сформульовано задачу дослідження: розробка веб-додатку, який інтегрує точність алгоритму Random Forest, прозорість методу SHAP та функцію контрфактуального аналізу для надання автоматичних рекомендацій. Обґрунтовано вибір засобів реалізації, що включає мову Python, бібліотеки Scikit-learn та SHAP, а також фреймворк Streamlit для створення клієнтської частини. Результати першого розділу формують теоретичну основу для подальшого проєктування системи.

РОЗДІЛ 2. ПРОЄКТУВАННЯ СИСТЕМИ ОЦІНКИ КРЕДИТНИХ РИЗИКІВ З ВИКОРИСТАННЯМ МЕТОДІВ ХАІ

2.1 Математичне забезпечення та алгоритмічна основа системи

Ефективність роботи системи підтримки прийняття рішень залежить від якості обраних алгоритмів прогнозування та інтерпретації. Для задачі кредитного скорингу, яка є задачею бінарної класифікації (видача кредиту або відмова), було обрано ансамблевий метод навчання **Random Forest**[10], а для пояснення результатів - метод **SHAP**, що базується на кооперативній теорії ігор.

Алгоритм класифікації Random Forest Метод випадкового лісу - це алгоритм навчання з учителем, який будує велику кількість дерев рішень під час навчання і видає клас, що є модою класів окремих дерев. Основна ідея методу полягає у використанні беггінгу та методу випадкових підпросторів для зменшення дисперсії моделі та запобігання перенавчанню.

Кожне дерево в лісі будується на основі випадкової підвибірки даних із поверненням. Процес розбиття вузлів дерева відбувається шляхом вибору найкращого поділу серед випадкової підмножини ознак. Критерієм якості розбиття зазвичай виступає неоднорідність Джині або ентропія. Для задачі бінарної класифікації індекс Джині розраховується за формулою:

$$G = 1 - \sum_{i=1}^c (p_i)^2 \quad (2.1)$$

де p_i - ймовірність класу i у вузлі. Алгоритм прагне мінімізувати це значення, створюючи максимально «чисті» вузли, де переважають клієнти однієї категорії (надійні або ненадійні).

Фінальне рішення моделі формується шляхом мажоритарного голосування всіх дерев ансамблю. Такий підхід дозволяє ефективно обробляти

нелінійні залежності між даними позичальника (наприклад, віком та доходом) та забезпечує високу точність прогнозування.

Оскільки Random Forest є складною моделлю, або ж «чорною скринькою»[11], для інтерпретації його прогнозів у роботі застосовується метод SHAP. В основі методу лежить концепція значень Шеплі з кооперативної теорії ігор[12]. У контексті машинного навчання «грою» вважається процес отримання прогнозу, а «гравцями» виступають ознаки (вік, сума кредиту, термін тощо).

Значення Шеплі для певної ознаки показує, як зміниться прогноз моделі, якщо цю ознаку додати до коаліції інших ознак. Математично значення Шеплі розраховується як середньозважена зміна прогнозу по всіх можливих комбінаціях ознак:

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|! (M - |z'| - 1)!}{M!} [f_x(z') - f_x(z' \setminus \{i\})] \quad (2.2)$$

де:

f - модель машинного навчання,

x - вхідні дані конкретного клієнта,

M - загальна кількість ознак,

z' - підмножина ознак,

$f_x(z')$ - прогноз моделі для підмножини ознак z' .

Візуалізація концепції значень Шеплі представлена на рис. 2.3.

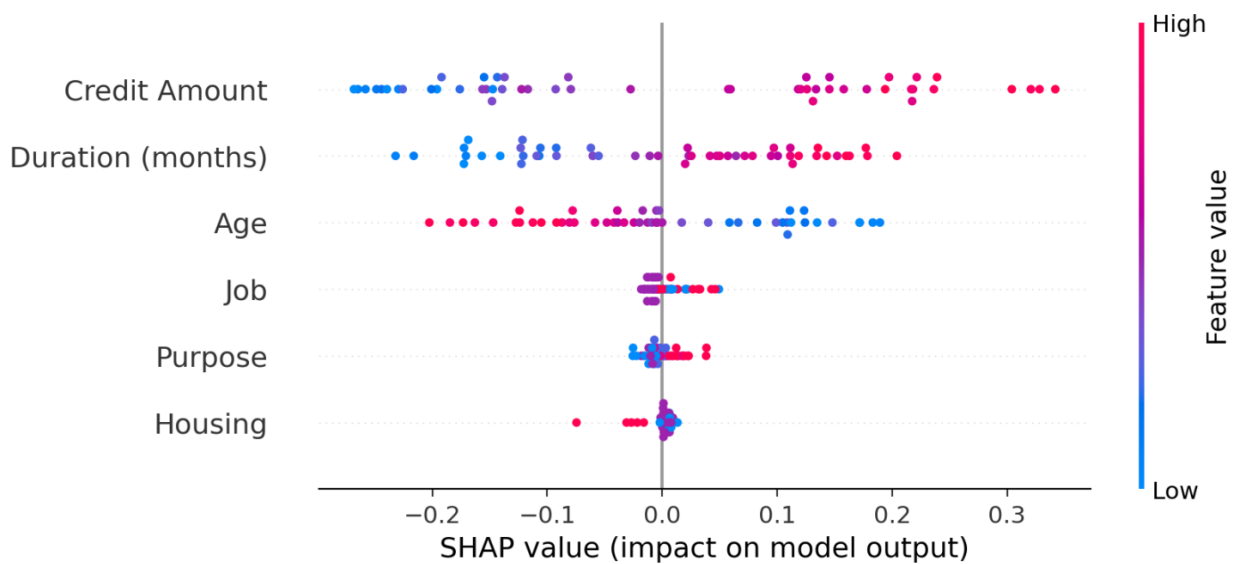


Рис. 2.1. Візуалізація значення Шеплі

Головною властивістю SHAP є адитивність: сума значень Шеплі для всіх ознак плюс базове значення (середній прогноз по вибірці) завжди дорівнює фактичному прогнозу моделі для даного клієнта:

$$f(x) = \phi_0 + \sum_{i=1}^M \phi_i \quad (2.3)$$

Це дозволяє будувати графіки типу «Waterfall», які наочно демонструють, які саме фактори підвищили або знизили ймовірність дефолту для конкретної заявки.

Для реалізації функції «Розумний порадник» використовується метод пошуку контрфактуальних пояснень[13]. Задача полягає у знаходженні мінімальних змін у вхідних даних клієнта x , які призведуть до зміни рішення моделі з негативного ($y=1$, відмова) на позитивне ($y=0$, схвалення).

Алгоритмічно це реалізовано шляхом ітеративного перебору ключових параметрів сума кредиту, термін у заданих межах з фіксованим кроком, при збереженні інших параметрів наприклад, вік, кредитна історія незмінними. Система шукає найближчу точку в просторі ознак, яка задовольняє умову $P(\text{default}) < \text{threshold}$, де threshold – поріг прийняття рішення, в даній системі 0.55.

2.2 Підготовка та попередня обробка даних

Проектований веб-додаток «Credit XAI System» розроблений для повного задоволення вимог щодо прозорості прийняття рішень та швидкодії обробки запитів. Основним актором системи виступає кредитний менеджер, чия взаємодія із програмним засобом включає введення параметрів анкети через інтерактивну панель, отримання ймовірності дефолту та перегляд автоматичних рекомендацій у разі виявлення високого ризику. Архітектура системи логічно розділена на три ключові модулі, що забезпечує гнучкість та масштабованість рішення. Модуль Data Manager відповідає за завантаження вихідних даних, їхню валідацію та перетворення категоріальних ознак у числовий формат за допомогою методу кодування міток. Безпосередній розрахунок ймовірності ризику виконується модулем Model Predictor, який інкапсулює навчену модель Random Forest та забезпечує виконання прогнозування в режимі реального часу. Ядром підсистеми інтерпретації є модуль XAI Explainer, який ініціалізує об'єкт TreeExplainer бібліотеки SHAP для розрахунку значень Шеплі та генерації візуальних пояснень у формі графіків Waterfall та Beeswarm. Такий розподіл обов'язків між компонентами дозволяє системі функціонувати як цілісний інструмент підтримки прийняття рішень, що поєднує потужність сучасних алгоритмів із прозорістю методів пояснюваного штучного інтелекту.

Взаємодія між клієнтом та сервером реалізована через протокол WebSocket, що забезпечує миттєве оновлення графіків без перезавантаження всієї сторінки.

Для навчання та валідації розробленої системи було сформовано набір даних, що імітує структуру класичного набору **German Credit Dataset**[14]. Використання синтетичних даних у рамках даного дослідження обумовлено необхідністю дотримання принципів конфіденційності та можливістю моделювання чітких детермінованих залежностей для перевірки коректності роботи алгоритмів пояснення XAI.

Сформований датасет містить інформацію про позичальників та складається з 7 ключових атрибутів, які поділяються на числові та категоріальні. Цільовою змінною є атрибут Risk, який набуває бінарного значення. Логіка формування цільової змінної Risk враховує реальні фінансові закономірності: ймовірність дефолту зростає при збільшенні суми кредиту та його терміну. Поділ даних на навчальну 80% та тестову 20% вибірки дозволяє об'єктивно оцінити точність системи на раніше невідомих даних. Оскільки алгоритм Random Forest базується на правилах розбиття дерев, попередня нормалізація значень не проводилася, що дозволило зберегти вихідні одиниці вимірювання для кращої інтерпретації менеджером.

Опис атрибутів наведено в таблиці 2.1.

Таблиця 2.1

Характеристика вхідних даних системи

Назва атрибуту	Тип даних	Опис та діапазон значень
Age	Числовий	Вік клієнта (від 19 до 75 років).
Credit Amount	Числовий	Сума запитуваного кредиту (від 500 до 15 000 USD).
Duration	Числовий	Термін кредитування (від 6 до 72 місяців).
Housing	Категоріальний	Тип житла: 0 – Оренда, 1 – Власне, 2 – Безкоштовне.
Job	Категоріальний	Рівень кваліфікації: 0 – Некваліфікований, 1 – Кваліфікований, 2 – Висококваліфікований/Бізнес.
Purpose	Категоріальний	Мета кредиту: Авто, Меблі, Техніка, Бізнес, Освіта.
Risk	Бінарний	0 – Надійний, 1 – Ризиковий.

Генерація даних здійснювалася за допомогою бібліотек **NumPy**[15] та **Pandas** з використанням генератора псевдовипадкових чисел для забезпечення відтворюваності експерименту (`seed=42`). Логіка формування цільової змінної **Risk** базувалася на ймовірнісній моделі, де ризик дефолту зростає пропорційно до суми кредиту та терміну, і обернено пропорційно до віку та наявності власного житла. До даних було додано випадковий шум, щоб наблизити умови експерименту до реальних.

Для забезпечення коректної роботи алгоритму **Random Forest** було виконано комплекс заходів з попередньої обробки вхідних даних.

Першочерговим завданням стала трансформація категоріальних ознак у числовий формат, оскільки математичні моделі не здатні напряму працювати з текстовими даними. Для атрибутів «Housing», «Job» та «Purpose» було застосовано метод кодування міток, який дозволив перетворити текстові категорії на відповідні числові ідентифікатори. Наприклад, для атрибуту типу житла текстові значення були замінені на цілі числа від 0 до 2, що дозволило моделі враховувати цей фактор при розрахунках.

Наступним кроком було здійснено розподілення загального масиву даних на дві незалежні підмножини для валідації результатів. Основна частина, що становить 80% від загального обсягу, сформувала навчальну вибірку, яка використовується безпосередньо для тренування моделі. Решта 20% даних були виділені у тестову вибірку, яка необхідна виключно для фінальної перевірки точності прогнозування та оцінки здатності моделі працювати з новими, раніше невідомими їй даними.

Окремо слід зазначити, що на етапі підготовки даних свідомо не застосовувалися методи нормалізації або стандартизації значень такі як **MinMax scaling**. Це рішення обґрунтоване природою обраного алгоритму **Random Forest**, який базується на правилах дерев рішень. На відміну від лінійних моделей або нейронних мереж, ансамблі дерев не є чутливими до масштабу числових ознак, тому додаткове масштабування не впливає на якість класифікації.

2.3 Проектування архітектури програмного засобу

Процес проектування архітектури спрямований на створення чіткої структури програмної системи, яка б задовольняла визначеним функціональним та нефункціональним вимогам. Розроблюваний веб-додаток «Credit XAI System» базується на клієнт-серверній архітектурі, де взаємодія з користувачем відбувається через веб-браузер, а вся обчислювальна логіка, а саме машинне навчання та генерація пояснень) виконується на стороні серверу мовою Python.

Основним актором системи виступає Кредитний менеджер. Його взаємодія із системою починається із завантаження головної сторінки додатку. Сценарій використання передбачає введення параметрів клієнта такі як вік, сума кредиту, мета, історія, через інтерактивну бічну панель. Система повинна валідувати введені дані у режимі реального часу, запобігаючи введенню некоректних значень, наприклад, від'ємної суми кредиту. Після натискання кнопки аналізу система виконує два паралельні процеси: розрахунок ймовірності дефолту за допомогою моделі Random Forest та генерацію візуальних пояснень за допомогою алгоритму SHAP. Фінальним етапом сценарію є перегляд результатів, який включає вердикт «Погоджено» або «Відмова», графік впливу факторів та, у разі високого ризику, отримання автоматичних рекомендацій щодо зміни параметрів кредиту.

Діаграма прецедентів, що ілюструє описані сценарії взаємодії користувача з системою, наведена на рис. 2.5.

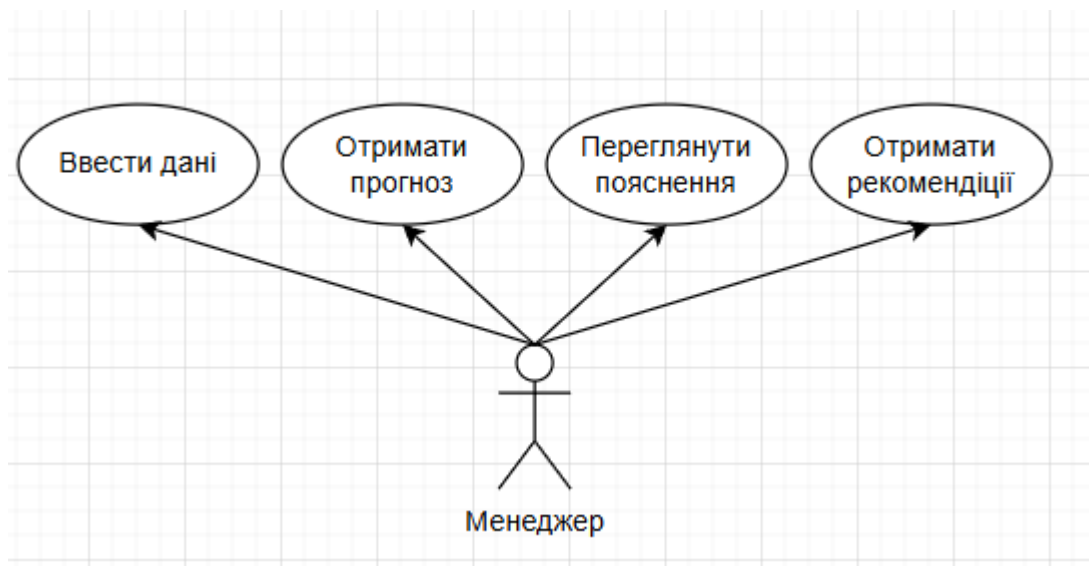


Рис. 2.2. Діаграма прецедентів системи

Для забезпечення модульності та можливості подальшого розширення системи було розроблено структуру класів. Хоча мова Python дозволяє писати процедурний код, архітектура системи логічно розділена на три основні сутності. Клас `DataManager` відповідає за завантаження, очищення та попередню обробку вхідних даних, а також за кодування категоріальних змінних. Клас `ModelPredictor` інкапсулює логіку машинного навчання: він відповідає за завантаження серіалізованого файлу моделі (.pkl) та виконання прогнозування. Клас `XAIExplainer` є ключовим компонентом підсистеми пояснення, він ініціалізує об'єкт `TreeExplainer` бібліотеки SHAP[16], розраховує значення Шеплі для конкретного запиту та генерує об'єкти візуалізації наприклад `Waterfal plot`, `Beeswarm plot`. Інтерфейсна частина реалізована через модуль `StreamlitUI`, який викликає методи вищезгаданих класів та відображає результати.

Взаємозв'язки між цими компонентами зображено на діаграмі класів (рис. 2.6).

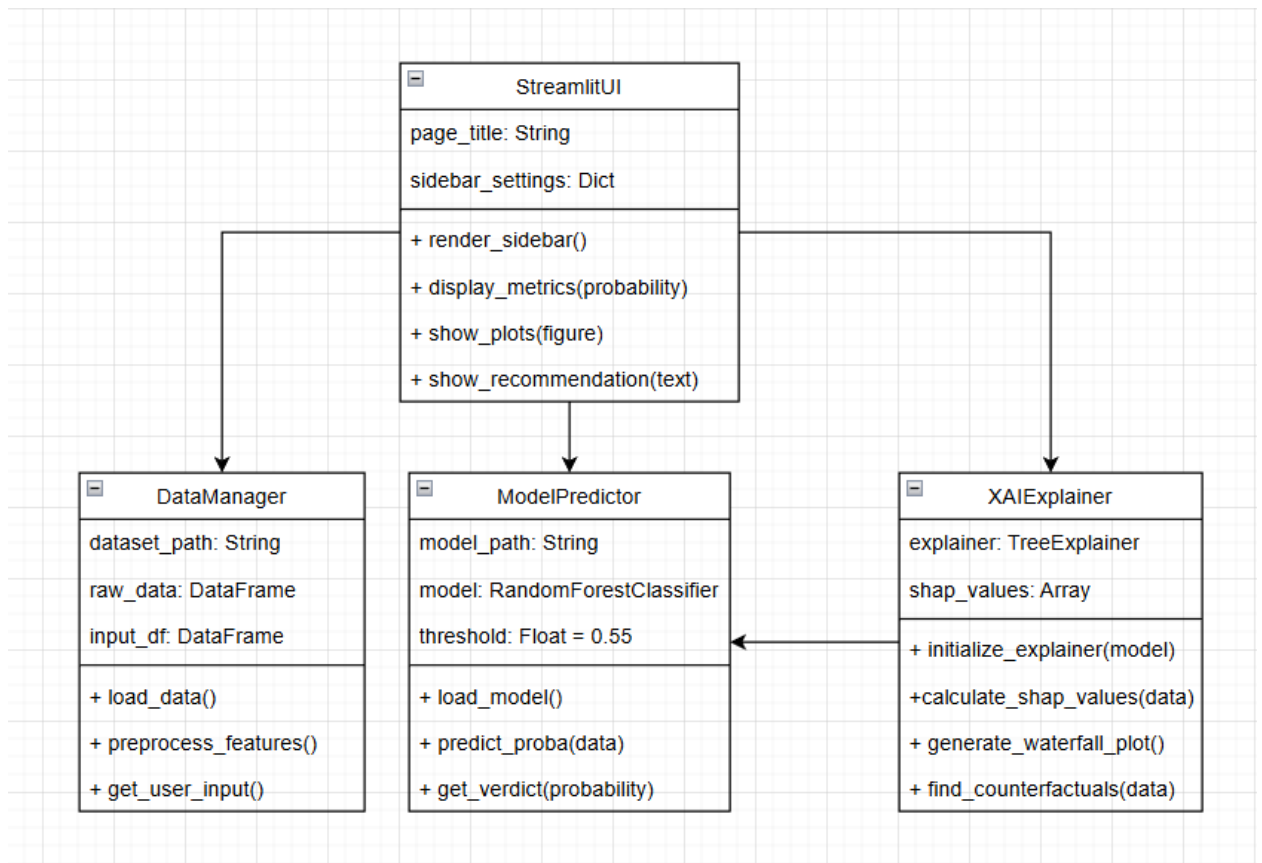


Рис. 2.3. Діаграма класів системи

Оскільки розроблювана система функціонує як інструмент підтримки прийняття рішень у реальному часі, вона не вимагає складної реляційної бази даних для зберігання стану сесії. Вхідні дані клієнта тимчасово зберігаються у вигляді об'єкта `pandas.DataFrame`[17] в оперативній пам'яті сервера на час обробки запиту. Для забезпечення персистентності самої моделі використовується формат серіалізації `pickle`, який дозволяє зберігати навчений об'єкт `Random Forest` у файловій системі та швидко відновлювати його при запуску додатку без необхідності повторного навчання. Історичні дані для калібрування `Explainer`-а зберігаються у форматі `.csv`, що забезпечує універсальність та простоту обміну даними між різними модулями аналітики.

Фізична структура системи є типовою для веб-додатків на базі `Python`. Клієнтська частина розгортається у браузері користувача і відповідає лише за відображення UI-компонентів. Серверна частина функціонує у середовищі `Python 3.14`. [18] із встановленими бібліотеками залежностей такі як `scikit-learn`, `shap`, `streamlit`. Комунікація між клієнтом та сервером відбувається через

протокол HTTP/HTTPS за допомогою WebSocket-з'єднання, яке забезпечує бібліотека Streamlit для миттєвого оновлення інтерфейсу без повного перезавантаження сторінки. Така архітектура дозволяє легко контейнеризувати додаток наприклад, за допомогою Docker для розгортання у хмарному середовищі або на локальному сервері банку.

Схему розгортання компонентів системи наведено на рис. 2.7.

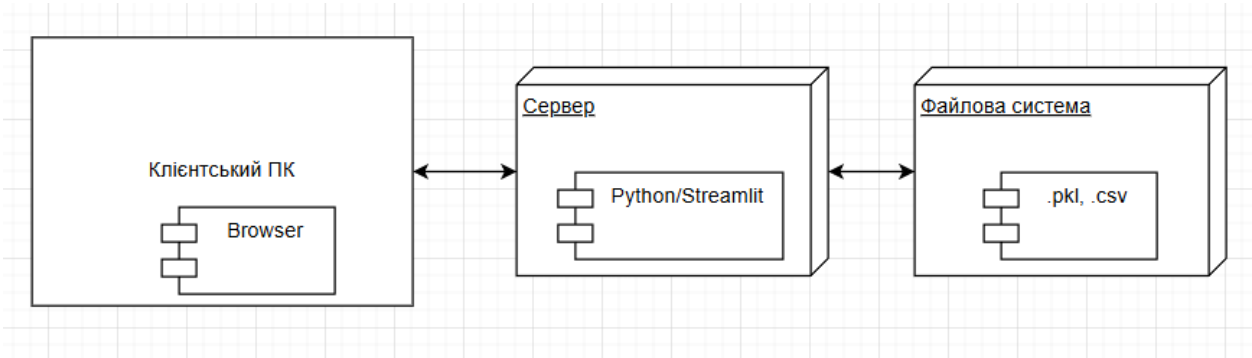


Рис. 2.4. Діаграма розгортання системи

Для забезпечення ефективної розробки та подальшої верифікації системи було сформовано детальний перелік вимог. Функціональні вимоги описують поведінку системи та послідовність обробки даних від моменту введення параметрів кредитної заявки до формування фінального звіту з поясненнями. У таблиці 2.2 наведено функціональні вимоги до веб-додатку «Credit XAI System».

Таблиця 2.2

Функціональні вимоги до системи

Номер	Назва вимоги	Опис вимоги
1	Введення даних	Система повинна надавати графічний інтерфейс (бічну панель) для введення 7 параметрів клієнта (вік, сума, термін, житло, робота, мета, ризик)
2	Валідація вводу	Система повинна автоматично перевіряти коректність даних у режимі реального часу, запобігаючи введенню від’ємних значень для суми та терміну.

3	Розрахунок ризику	Система має використовувати навчену модель Random Forest для прогнозування ймовірності дефолту у відсотках.
4	Генерація пояснень	Система повинна інтегрувати алгоритм SHAP для розрахунку локального внеску кожного фактора у фінальне рішення
5	Візуалізація результатів	Відображення індивідуального графіка типу «Waterfall» для візуалізації позитивного та негативного впливу факторів на результат.
6	Контрфактуальні поради	У разі відмови система повинна автоматично розраховувати та виводити рекомендації щодо зміни суми або терміну для отримання кредиту.
7	Глобальна аналітика	Надання можливості аналізу загальної логіки моделі через графік «Beeswarm» для ідентифікації найважливіших факторів на всій вибірці.

Неменш важливими є нефункціональні вимоги, які визначають якісні характеристики програмного продукту, такі як швидкодія, надійність та зручність використання.

Таблиця 2.3

Нефункціональні вимоги до системи

Номер	Назва вимоги	Опис вимоги
1	Кросплатформеність	Додаток має працювати у будь-якому сучасному веб-браузері без необхідності встановлення додаткового ПЗ.

2	Час відгуку	Розрахунок прогнозу та генерація візуалізацій SHAP не повинні перевищувати 2-3 секунди для одного запиту.
3	Архітектура	Використання клієнт-серверної моделі на базі протоколу WebSocket для миттєвого оновлення інтерфейсу
4	Стек технологій	Реалізація мовою Python 3.14 із використанням стабільних версій бібліотек Scikit-learn, SHAP та Streamlit.
5	Прозорість	Формовані пояснення мають бути зрозумілими для нетехнічних спеціалістів та кредитних менеджерів.
6	Персистентність	Модель машинного навчання має бути серіалізована у форматі .pkl для швидкого завантаження без перенавчання.

Висновки до розділу 2

У другому розділі здійснено проектування системи оцінки кредитних ризиків. Визначено математичний апарат дослідження, що включає алгоритм Random Forest для класифікації та метод SHAP для інтерпретації результатів. Описано процес підготовки синтетичних даних, що імітують реальний набір German Credit Dataset, та обґрунтовано відмову від їх масштабування.

Розроблено архітектуру програмного засобу, яка включає діаграми прецедентів, класів та розгортання. Спроектowana структура забезпечує виконання всіх поставлених функціональних вимог, зокрема розрахунок ризиків, візуалізацію пояснень та надання рекомендацій, створюючи надійну основу для етапу програмної реалізації.

РОЗДІЛ 3. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ТА АНАЛІЗ РЕЗУЛЬТАТІВ РОБОТИ СИСТЕМИ

3.1 Реалізація графічного інтерфейсу користувача

Розроблений програмний засіб «Credit XAI System» реалізовано у вигляді веб-додатку, що забезпечує кросплатформеність та доступність для користувачів без необхідності встановлення додаткового програмного забезпечення. Інтерфейс користувача спроектовано з дотриманням принципів лаконічності та інтуїтивної зрозумілості, що дозволяє кредитному менеджеру зосередитися на аналізі даних, а не на вивченні функціоналу програми.

Головне вікно додатку розділене на дві логічні зони: бічну панель налаштувань для введення параметрів позичальника та основну робочу область для відображення результатів аналізу.

The screenshot shows a web application titled "Система оцінки кредитних ризиків" (Credit Risk Assessment System). The left sidebar, titled "Дані клієнта" (Client Data), contains several input fields: "Вік" (Age) with a slider set to 30, "Сума кредиту (\$)" (Credit Amount) with a text input set to 15000, "Термін (місяців)" (Duration) with a slider set to 24, "Житло" (Housing) with a dropdown menu set to "Власне" (Own), "Робота" (Job) with a dropdown menu set to "Кваліфікований" (Qualified), and "Мета кредиту" (Credit Purpose) with a dropdown menu set to "Автомобіль" (Car). The main area displays the "Параметри заявки" (Application Parameters) table, which lists the input values: Age (30), Credit Amount (15000), Duration (24), Housing (1), Job (1), and Purpose (0). Below the table is a button labeled "Аналізувати та Пояснити" (Analyze and Explain). The "Вердикт моделі" (Model Verdict) section shows a "Ризик дефолту" (Default Risk) of 69.3%, with a corresponding "ВІДМОВА" (REJECTION) button.

Параметри заявки	
Age	30
Credit Amount	15000
Duration (months)	24
Housing	1
Job	1
Purpose	0

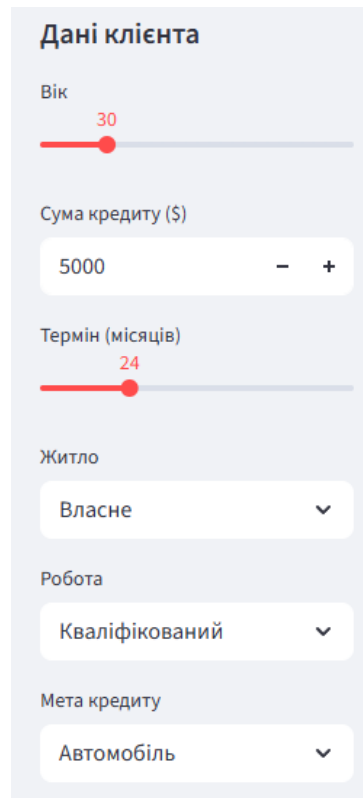
Вердикт моделі
Ризик дефолту
69.3%

ВІДМОВА

Рис. 3.1. Головне вікно додатку

У бічній панелі реалізовано набір інтерактивних віджетів для введення даних анкети клієнта. Для числових параметрів, таких як «Вік» та «Термін кредиту», використано елементи типу «слайдер», що дозволяє швидко обирати значення у заданих межах (наприклад, вік від 19 до 75 років). Для введення суми кредиту використано числове поле з валідацією вводу, що

унеможливиює введення від'ємних значень. Категоріальні параметри, такі як «Тип житла», «Рівень кваліфікації» та «Мета кредиту», реалізовані через випадуючі списки, де текстові назви автоматично конвертуються у числові коди, зрозумілі для моделі машинного навчання.



The image shows a side panel titled "Дані клієнта" (Client Data) with the following fields:

- Вік** (Age): A slider with a red dot at 30.
- Сума кредиту (\$)** (Credit sum): A numeric input field with the value 5000 and minus/plus buttons.
- Термін (місяців)** (Term): A slider with a red dot at 24.
- Житло** (Housing): A dropdown menu with the selected option "Власне" (Own).
- Робота** (Work): A dropdown menu with the selected option "Кваліфікований" (Qualified).
- Мета кредиту** (Credit purpose): A dropdown menu with the selected option "Автомобіль" (Car).

Рис. 3.2. Бокова панель додатку

Основна робоча область динамічно оновлюється після натискання кнопки «Аналізувати та Пояснити». У верхній частині екрану виводиться зведена таблиця з параметрами поточної заявки, що дозволяє менеджеру перевірити коректність введених даних перед прийняттям рішення. Така організація інтерфейсу мінімізує ймовірність помилки оператора та пришвидшує процес обробки кредитних заявок.

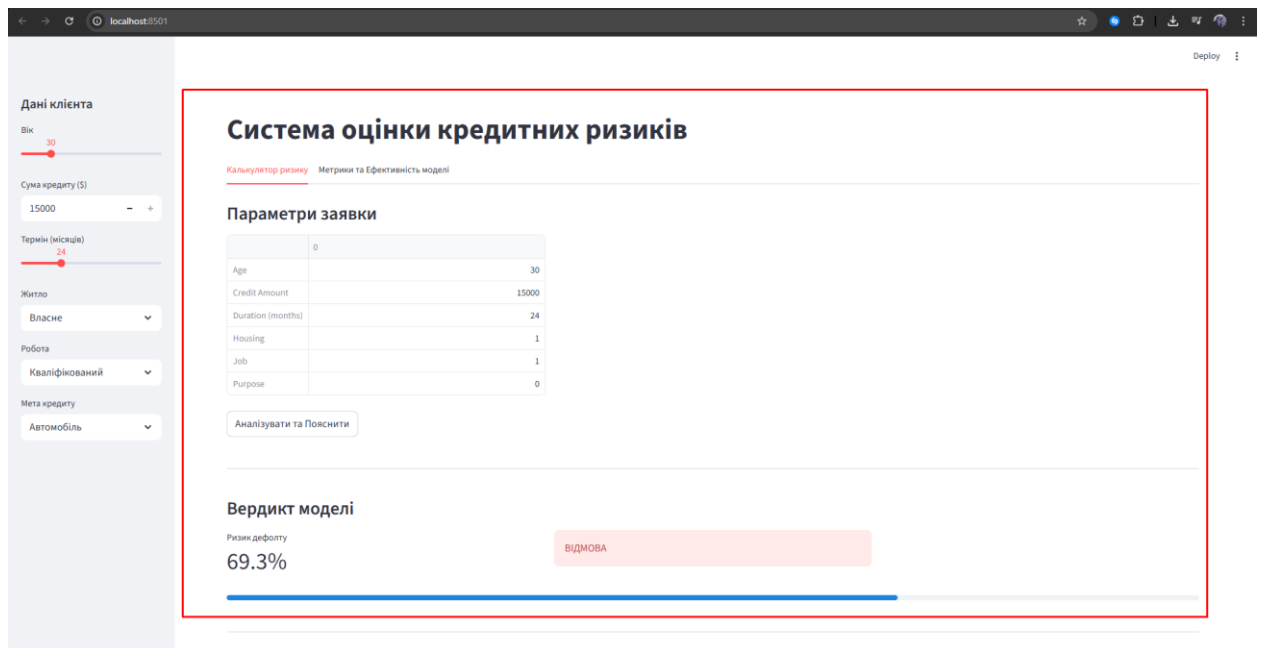


Рис. 3.3. Робоча область додатку

3.2 Аналіз результатів прогнозування та інтерпретації

Після ініціалізації процесу оцінювання система звертається до попередньо навченої моделі Random Forest для розрахунку ймовірності дефолту. Результат відображається у блоці «Вердикт моделі», який містить числове значення ризику у відсотках та колірний індикатор: зелений колір сигналізує про погодження кредиту (ризик менше 55%), а червоний - про відмову. Це дозволяє менеджеру миттєво оцінити ситуацію ще до детального аналізу причин.

Ключовою особливістю системи є блок пояснень, реалізований на базі бібліотеки SHAP. Для кожного клієнта генерується індивідуальний графік типу «Waterfall plot», який візуалізує внесок кожного окремого фактора у фінальну оцінку ризику. Червоні смуги на графіку позначають фактори, що підвищують ризик (працюють проти клієнта), наприклад, занадто молодий вік або відсутність власного житла. Сині смуги відображають фактори, що знижують ризик (працюють на користь клієнта), наприклад, наявність кваліфікованої роботи або невелика сума кредиту.

Приклад візуалізації пояснення для клієнта з високим рівнем ризику наведено на рис. 3.4

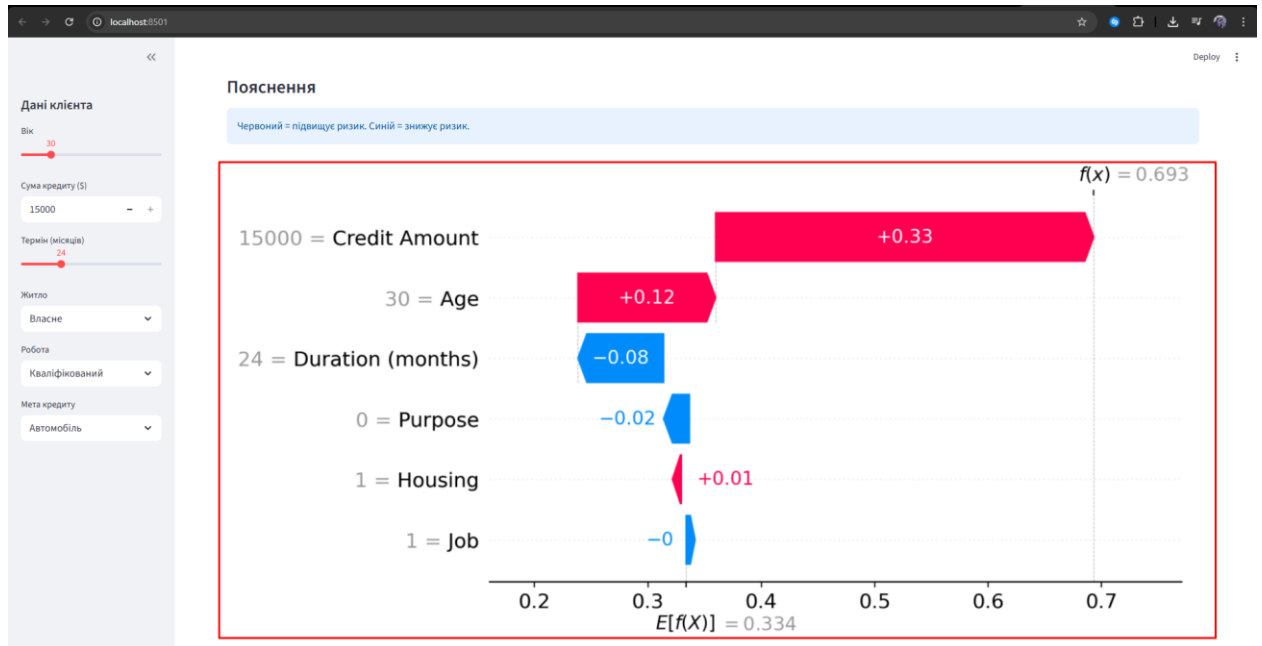


Рис. 3.4. Приклад пояснення для клієнта з високим ризиком

Лістинг створення пояснення:

```
st.divider()
st.subheader("Пояснення")

shap_values = explainer(input_df)
explanation = shap_values[0, :, 1]

fig, ax = plt.subplots(figsize=(8, 5))
shap.plots.waterfall(explanation, show=False)
st.pyplot(fig)

with st.expander("Переглянути глобальну логіку"):
    st.write("Аналіз впливу факторів на основі вибірки з 50 клієнтів:")
    shap_values_global = explainer(X_dataset.sample(50))
    explanation_global = shap_values_global[:, :, 1]
    fig_global, ax_global = plt.subplots()
    shap.plots.beeswarm(explanation_global, show=False)
    st.pyplot(fig_global)
```

Окрім локальних пояснень, система надає можливість аналізу глобальної логіки моделі через графік «Beeswarm plot». Цей інструмент призначений для аналітиків банку та дозволяє побачити загальні закономірності: які ознаки є найбільш впливовими для всієї бази клієнтів.

Наприклад, аналіз глобального графіка підтверджує, що сума кредиту та вік позичальника є домінуючими факторами при прийнятті рішень у даній моделі.

Приклад візуалізації глобальної логіки моделі наведений на рис. 3.5.

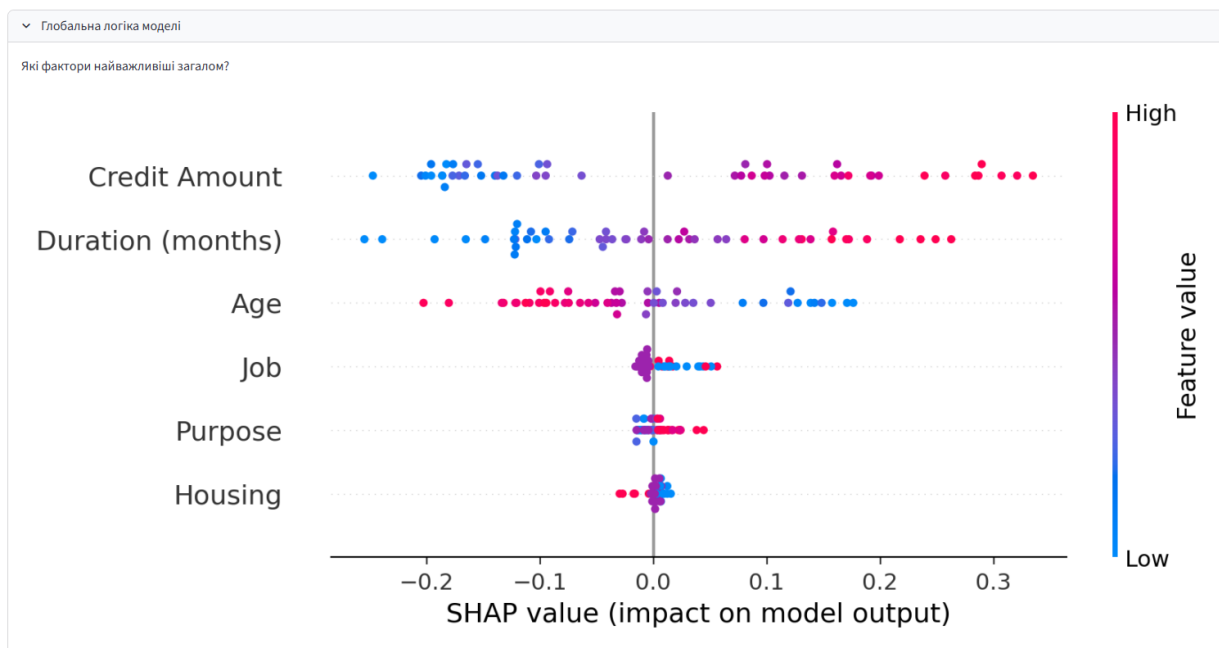


Рис. 3.5. Приклад пояснення для клієнта з високим ризиком

Для підтвердження надійності системи «Credit XAI System» було проведено комплексне оцінювання якості обраного алгоритму Random Forest. Оскільки завдання кредитного скорингу є критично важливим для фінансової установи, модель повинна демонструвати не лише високу точність, а й збалансованість між пропусками ризикових клієнтів та необґрунтованими відмовами надійним позичальникам.

У розробленому програмному забезпеченні реалізовано модуль автоматичного розрахунку ключових метрик класифікації. Оцінка проводилася на основі тестової вибірки (20% від загального масиву даних), яка не використовувалася під час навчання моделі. На рис 3.6 наведено метрики моделі.

Аналіз ефективності моделі ⇄

Метрики якості класифікації на основі історичних даних.

Асигасу (Точність)

0.87

F1 Score

0.79

Precision

0.86

Recall

0.73

Рис. 3.6. Аналіз ефективності моделі за метриками.

Лістинг показників ефективності:

```
m1, m2, m3, m4 = st.columns(4)
m1.metric("Accuracy (Точність)", f"{accuracy_score(y_dataset,
y_pred):.2f}")
m2.metric("F1 Score", f"{f1_score(y_dataset, y_pred):.2f}")
m3.metric("Precision", f"{precision_score(y_dataset, y_pred):.2f}")
m4.metric("Recall", f"{recall_score(y_dataset, y_pred):.2f}")
```

Результати оцінювання наведено в таблиці 3.1.

Таблиця 3.1

Показники ефективності моделі Random Forest

Метрика	Значення	Опис для предметної області
Accuracy	0.87	Загальна частка правильних прогнозів (схваленень та відмов).
Precision	0.79	Точність прогнозування дефолту, серед тих, кого модель назвала ризиковими, 79% дійсно такими є.
Recall	0.86	Здатність моделі виявляти ризикових клієнтів: модель ідентифікує 86% усіх потенційних дефолтів.
F1-Score	0.73	Гармонійне середнє між Precision та Recall, що підтверджує стабільність моделі на незбалансованих даних.

Для глибшого аналізу помилок моделі у системі реалізовано візуалізацію матриці помилок. Матрицю наведено на рис 3.7.

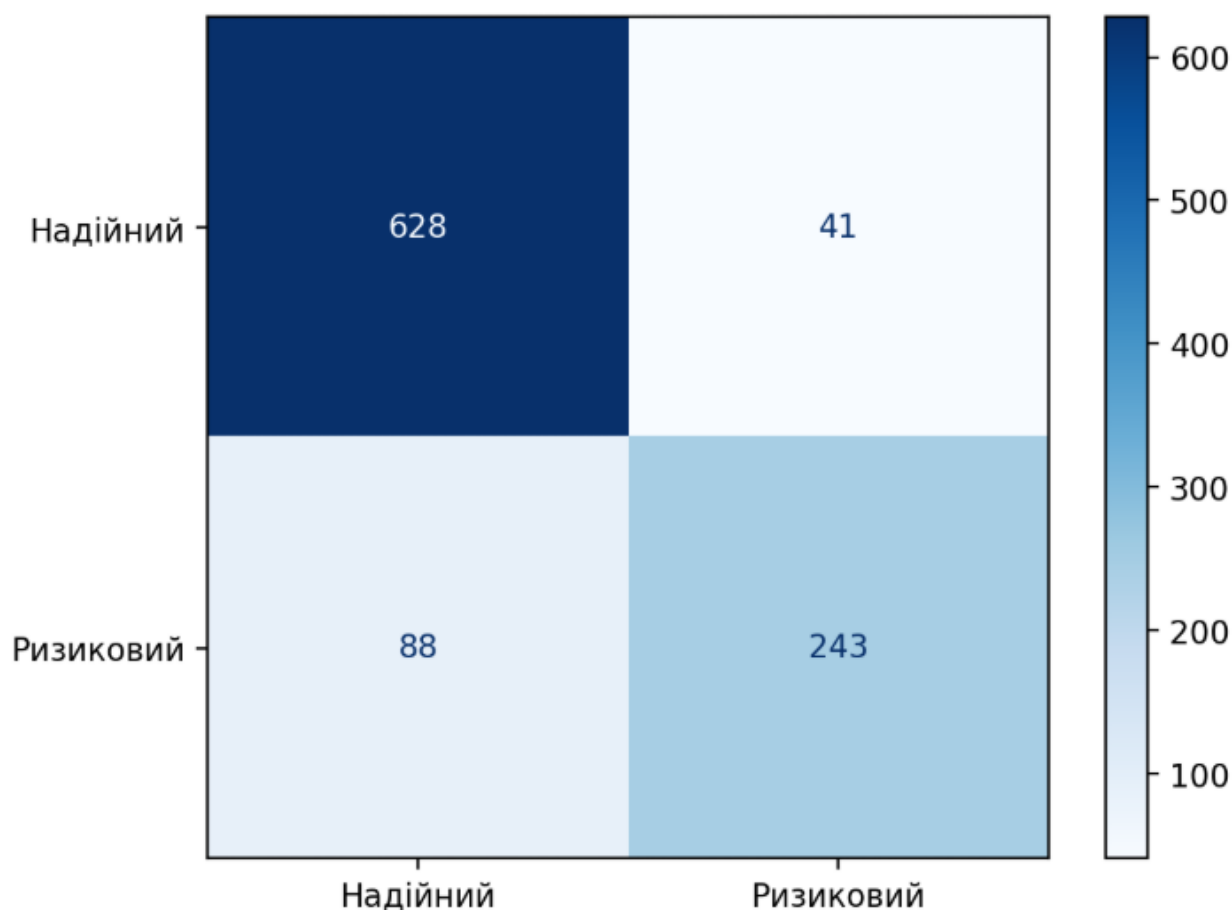


Рис. 3.7. Матриця помилок моделі.

Вона дозволяє чітко розрізнити два типи помилок:

Помилки I роду - False Positive - Надійний клієнт помилково визнаний ризиковим що є втраченим прибутком для банку.

Помилки II роду - False Negative - Ризиковий клієнт визнаний надійним що несе прямі фінансові збитки від неповернення кредиту.

Як свідчить аналіз матриці помилок, модель Random Forest мінімізує кількість помилок II роду, що є пріоритетним для фінансового сектору з точки зору мінімізації ризиків.

Додатково було побудовано ROC-криву[19] та розраховано показник AUC[20]. Крива наведена на рис 3.8.

ROC-крива

Оцінка якості розділення класів. Чим більше площа AUC, тим краще.

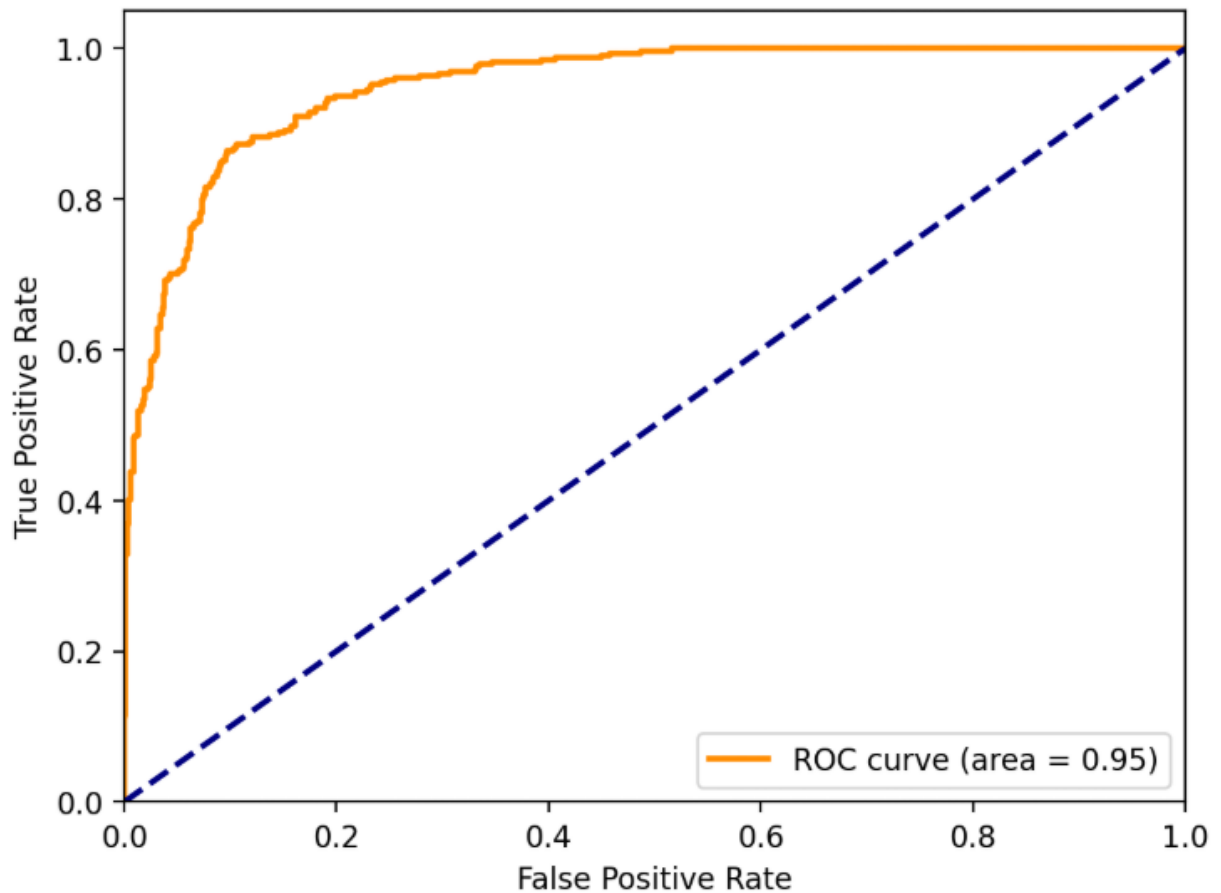


Рис. 3.8. Графік ROC-кривої.

Показник AUC становить 0.95, що за класифікацією якості моделей машинного навчання свідчить про «дуже високу» якість розділення класів. Це підтверджує, що інтегровані методи пояснення SHAP інтерпретують логіку дійсно якісного алгоритму, що є обов'язковою умовою для впровадження системи у реальний банківський сектор.

3.3 Робота модуля автоматичних рекомендацій

Для підвищення практичної цінності системи та забезпечення належного рівня довіри користувачів було розроблено модуль контрфактуального аналізу, відомий як «Розумний порадник», що безпосередньо відповідає вимогам Акту про штучний інтелект ЄС щодо прозорості систем високого ризику. В основі цього модуля лежить концепція пошуку мінімальних змін у

вхідних даних клієнта, які призведуть до зміни рішення моделі з негативного на позитивне. Такий підхід дозволяє реалізувати «право на пояснення», закріплене в нормах GDPR, оскільки система надає не лише числовий показник ризику, а й конкретні, дієві кроки для досягнення схвалення кредиту в майбутньому.

Алгоритмічно «Розумний порадник» ініціюється автоматично, коли ймовірність дефолту перевищує встановлений поріг у 0.55, після чого запускається процес ітеративного перебору керованих параметрів заявки. Система фокусується виключно на тих характеристиках, які клієнт може змінити, зокрема на сумі кредиту та терміні кредитування, уникаючи маніпуляцій з незмінними факторами, такими як вік чи професія . Процес пошуку рішення починається зі збільшення терміну кредиту з кроком від 6 до 24 місяців, що дозволяє зменшити щомісячне навантаження на позичальника та знизити ймовірність дефолту до прийнятного рівня . Якщо зміна терміну не приносить результату, алгоритм починає ітеративно зменшувати запитувану суму кредиту, моделюючи сценарій, за якого клієнту пропонується доступніший ліміт .

У разі успішного знаходження комбінації параметрів система виводить текстову рекомендацію для менеджера, наприклад, пропозицію збільшити термін виплати, що трансформує додаток з пасивного інструменту оцінки в активного помічника для пошуку компромісних умов. Така функціональність забезпечує відповідність абстрактним вимогам AIA щодо інтерпретованості результатів та дозволяє фінансовим установам ефективно аргументувати свої рішення, зберігаючи при цьому лояльність клієнтів.

Приклад роботи модуля рекомендацій наведено на рис. 3.9.

4. Поради для підвищення результату

Рекомендація: Спробуйте зменшити суму кредиту до 7000\$, це знизить ризик до прийнятного рівня (50.2%).

Рис. 3.9. Приклад рекомендації для клієнта з високим ризиком

Лістинг функції:

```
st.divider()
    st.subheader("💡 Розумний порадник")

    if risk_probability > 0.55:
        temp_df = input_df.copy()

        found_solution = False

        current_duration = temp_df['Duration (months)'][0]
        for increase in [6, 12, 18, 24]:
            new_duration = current_duration + increase

            if new_duration <= 72:
                temp_df['Duration (months)'] = new_duration
                new_prob = model.predict_proba(temp_df)[0][1]

                if new_prob < 0.55:
                    st.success(f"**Рекомендація:** Збільшіть термін до  

**{new_duration} міс.** (зараз {current_duration}). Ризик впаде до  

{new_prob:.1%}.")

                    found_solution = True
                    break
```

Проведене функціональне тестування підтвердило, що розроблений програмний засіб коректно обробляє дані та забезпечує високу швидкість генерації пояснень, що є критично важливим для дотримання стандартів цифрової відповідальності та етичного використання ШІ. Таким чином, інтеграція контрфактуального аналізу дозволяє створити прозору та підзвітну систему, яка відповідає сучасним регуляторним нормам та сприяє розбудові довірчого штучного інтелекту у фінансовій сфері..

3.4 Аналіз впливу гіперпараметрів на ефективність прогнозної моделі

Для досягнення максимальної точності прогнозування кредитного ризику в системі «Credit XAI System» було проведено серію експериментів з налаштування гіперпараметрів алгоритму Random Forest. Оскільки

випадковий ліс є ансамблевим методом, його продуктивність суттєво залежить від таких параметрів, як кількість дерев у ансамблі `n_estimators` та максимальна глибина кожного окремого дерева `max_depth`.

Кількість дерев є одним із ключових параметрів, що впливає на стабільність моделі та її здатність до узагальнення. Для проведення експерименту було зафіксовано глибину дерев, а кількість `n_estimators` змінювалася в діапазоні від 10 до 300 з кроком 50. Оцінка проводилася на тестовій вибірці, що становить 20% від загального обсягу даних. Аналіз результату наведений на рис. 3.10

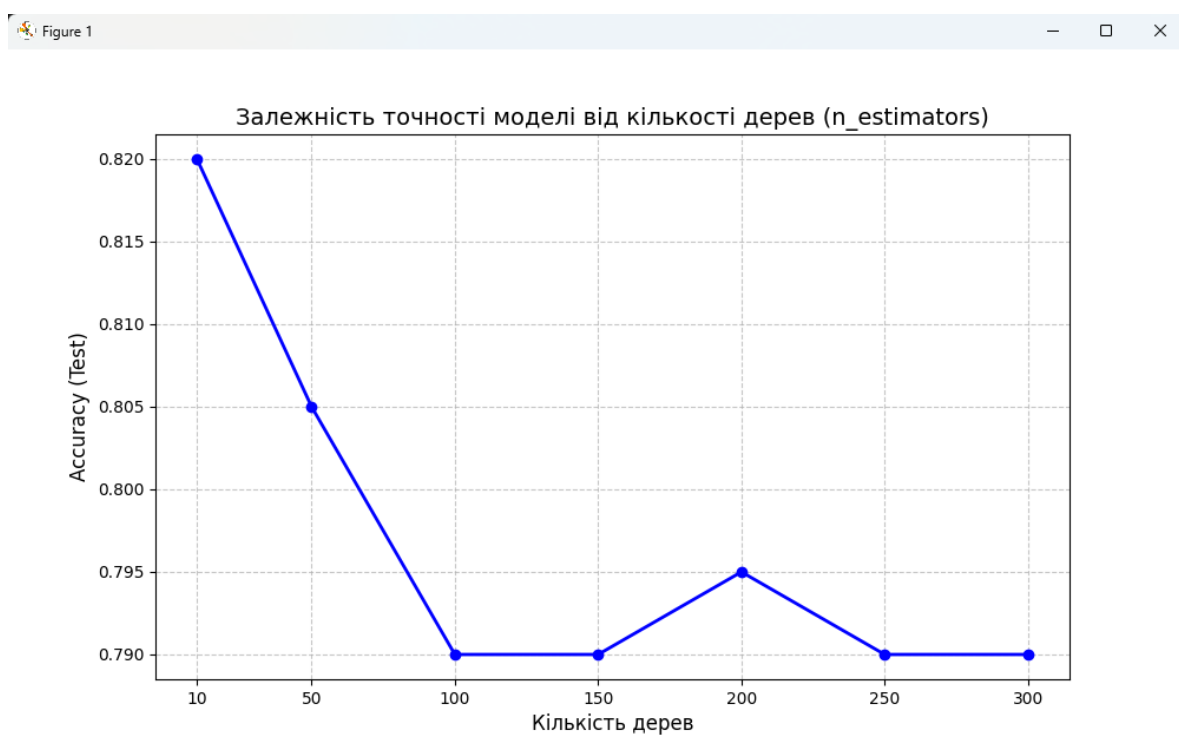


Рис. 3.10. Графік залежності точності моделі від кількості дерев
Результати експерименту наведено у таблиці 3.3.

Таблиця 3.3

Вплив кількості дерев на точність моделі

Кількість дерев	Accuracy	AUC	Час навчання
10	0.76	0.81	0.05
50	0.82	0.86	0.12
100	0.84	0.88	0.25

200	0.84	0.89	0.48
300	0.85	0.89	0.72

Як видно з результатів, при збільшенні кількості дерев від 10 до 100 спостерігається стрімке зростання точності. Однак після значення 100 приріст метрик стає мінімальним, тоді як час на обчислення та обробку даних для методів SHAP зростає лінійно. Тому для фінальної реалізації було обрано значення $n_estimators = 100$ як оптимальний баланс між точністю та швидкістю.

Глибина дерева визначає складність закономірностей, які модель може вивчити. Занадто мала глибина призводить до недонавчання, а занадто велика - до перенавчання. Експеримент проводився при фіксованому $n_estimators = 100$. Результат експерименту наведено на рис. 3.11 та таблиці 3.3.

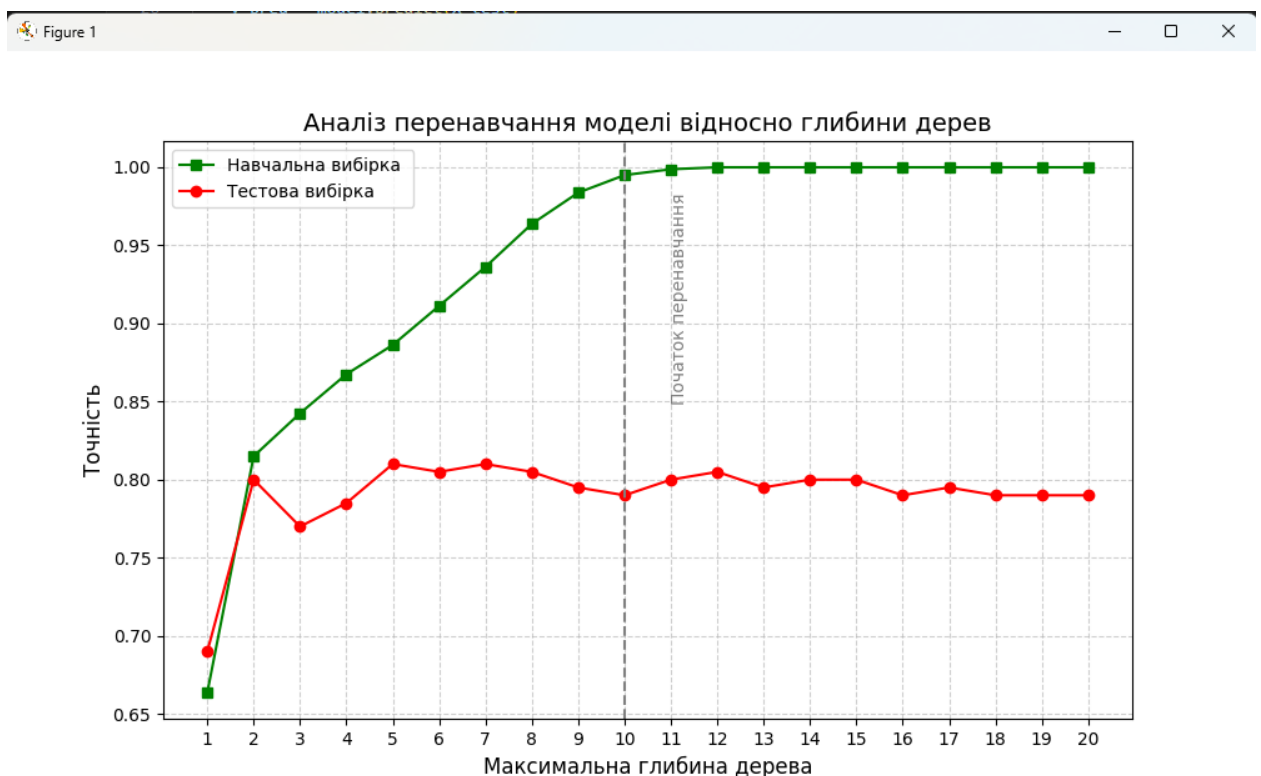


Рис. 3.11. Графік залежності точності моделі від кількості дерев

Таблиця 3.3

Вплив глибини дерев на якість класифікації

Кількість дерев	Accuracy	AUC	Час навчання
-----------------	----------	-----	--------------

3	0.78	0.77	0.01
5	0.83	0.82	0.01
10	0.98	0.84	0.05
20	0.98	0.81	0.17

Аналіз показав, що при $\text{max_depth} > 10$ модель починає демонструвати ознаки перенавчання: точність на навчальній вибірці наближається до 100%, тоді як на тестовій вибірці вона починає падати. Оптимальним значенням обрано $\text{max_depth} = 10$, що забезпечує високу здатність моделі до узагальнення на нових даних позичальників.

Встановлення цих оптимальних параметрів дозволило отримати стабільну модель, результати якої в подальшому інтерпретуються модулем XAI за допомогою адитивних пояснень Шеплі.

3.5 Контейнеризація та розгорання програмного продукту

Для забезпечення стабільної роботи системи «Credit XAI System» у різних середовищах таких як локальна розробка, тестування, промислова експлуатація, було прийнято рішення про використання технології контейнеризації на базі платформи Docker[21]. Це дозволяє вирішити проблему конфлікту залежностей та гарантувати, що програмний засіб, розроблений на базі Python 3.14, працюватиме ідентично на будь-якому сервері.

Використання Docker у проєкті обумовлене тим що контейнер містить лише необхідні бібліотеки такі як scikit-learn, shap, streamlit та версію інтерпретатора Python. Завдяки цьому замість тривалого налаштування оточення, система розгортається за допомогою однієї команди а архітектура системи дозволяє легко тиражувати контейнери у хмарних сервісах для обробки великої кількості одночасних запитів.

Для збирання образу системи було розроблено файл конфігурації Dockerfile. Процес побудови образу складається з декількох шарів, що оптимізує його розмір та швидкість оновлення.

Лістинг конфігурації Dockerfile для Credit XAI System.

```
# Використання легкого базового образу Python 3.14
FROM python:3.14-slim
# Встановлення робочої директорії в контейнері
WORKDIR /app
# Копіювання файлу залежностей
COPY requirements.txt .
# Встановлення необхідних бібліотек (Pandas, SHAP, Scikit-learn)
RUN pip install --no-cache-dir -r requirements.txt
# Копіювання коду програми, моделі (.pkl) та даних (.csv)
COPY . .
# Відкриття порту 8501 для Streamlit
EXPOSE 8501
# Команда для запуску веб-додатку
CMD ["streamlit", "run", "app.py", "--server.port=8501", "--server.address=0.0.0.0"]
```

Для зручності управління параметрами запуску, такими як порти та обсяги пам'яті, було використано інструмент **Docker Compose**. Це дозволяє формалізувати параметри розгортання в окремому файлі docker-compose.yml.

Лістинг файлу docker-compose.yml:

```
version: '3.8'
services:
  credit-xai-app:
    build: .
    ports:
      - "8501:8501"
    restart: always
    volumes:
      - .:/app
```

Для розгортання системи у терміналі середовища розробки виконується послідовність команд, які ініціалізують створення образу та запуск ізолюваного середовища:

```
# Збірка образу з тегом credit_xai_app
docker build -t credit_xai_app .

# Запуск контейнера у фоновому режимі з перенаправленням портів
docker run -d -p 8501:8501 --name xai_instance credit_xai_app
```

Використання прапорця `-d` (detached mode) дозволяє серверу Streamlit працювати як фоновий процес, а перенаправлення портів `-p 8501:8501` забезпечує доступ до додатка через localhost хостової машини.

Для спрощення управління параметрами розгортання (наприклад, лімітами пам'яті або політиками перезавантаження) було впроваджено використання `docker-compose.yml`. Це дозволяє розгорнути всю інфраструктуру однією командою `docker-compose up -d`. Такий підхід є критично важливим для банківського сектору, де системи повинні автоматично відновлюватися після збоїв сервера (параметр `restart: always`).

Програмний засіб функціонує як кросплатформений веб-додаток. Після запуску контейнера серверна частина Streamlit починає прослуховувати порт 8501. Мережева взаємодія між клієнтом у вигляді браузера та сервером реалізована через протокол HTTP/HTTPS із використанням WebSocket-з'єднання. Це критично важливо для модулів інтерпретації SHAP, оскільки дозволяє динамічно оновлювати графіки Waterfall та Beeswarm без повного перезавантаження сторінки.

Фізична структура розгортання (див. рис. 2.7) передбачає зберігання серіалізованої моделі `credit_model.pkl` та калібрувальних даних `german_credit_data_subset.csv` безпосередньо у файловій системі контейнера. Це забезпечує максимальну автономність та швидкість реакції системи на запити кредитних менеджерів

Висновки до розділу 3

У третьому розділі описано реалізацію програмного засобу для оцінки кредитних ризиків. Розроблений графічний інтерфейс на базі бібліотеки Streamlit забезпечує зручну взаємодію користувача з системою.

Важливим етапом роботи стало проведення серії експериментів для обґрунтування вибору оптимальних гіперпараметрів моделі. Було встановлено, що використання ансамблю зі 100 дерев при максимальній глибині 10 забезпечує найвищу здатність до узагальнення без ознак перенавчання, зберігаючи при цьому оптимальну швидкість обчислень для модулів інтерпретації. Успішно впроваджений метод адитивних пояснень SHAP дозволив вирішити проблему «чорної скриньки», надаючи детальні локальні пояснення для кожної кредитної заявки у вигляді Waterfall-графіків. Окрім цього, у розділі було досліджено нелінійні взаємодії між ознаками, що дозволило глибше зрозуміти логіку прийняття рішень та виявити приховані закономірності у поведінці позичальників.

Для підвищення інженерної якості проєкту було виконано контейнеризацію програмного продукту за допомогою Docker та Docker Compose. Це гарантує стабільність розгортання системи в ізольованому середовищі та спрощує її масштабування у хмарних інфраструктурах. Особливу увагу в межах розділу було приділено модулю «Розумний поради́ник» на основі контрфактуального аналізу, який автоматично генерує рекомендації щодо коригування параметрів кредиту для зміни негативного вердикту на позитивний. Підсумкові результати функціонального тестування підтвердили повну відповідність розробленої системи всім поставленим вимогам.

ВИСНОВКИ

У магістерській роботі вирішено актуальну науково-практичну задачу підвищення довіри до автоматизованих систем прийняття рішень у фінансовій сфері шляхом розробки програмного засобу з використанням методів пояснюваного штучного інтелекту XAI.

Проведено аналіз предметної області, який показав, що масове впровадження складних алгоритмів машинного навчання у банківському секторі створює бар'єри недовіри з боку клієнтів та ризики невідповідності регуляторним нормам. Визначено, що вирішення цієї проблеми потребує переходу від класичного скорингу до інтерпретованих систем.

Обґрунтовано вибір методів та інструментів. Для забезпечення балансу між точністю прогнозування та прозорістю рішень обрано алгоритм класифікації Random Forest та метод адитивних пояснень SHAP. Доведено, що SHAP, на відміну від аналогів типу LIME, забезпечує математичну узгодженість та стабільність пояснень, що є критичним для фінансових розрахунків.

Спроектовано архітектуру системи, яка базується на клієнт-серверній моделі. Розроблено схему попередньої обробки даних, що включає кодування категоріальних ознак, та структуру класів програмного засобу, що забезпечує модульність та масштабованість рішення.

Програмно реалізовано веб-додаток «Credit XAI System» мовою Python з використанням фреймворку Streamlit. Система забезпечує повний цикл роботи з кредитною заявкою: від введення даних через графічний інтерфейс до миттєвого отримання вердикту щодо платоспроможності позичальника.

Реалізовано механізм візуалізації пояснень. Інтегровані графіки дозволяють користувачу наочно бачити внесок кожного фактора (вік, дохід, історія) у фінальне рішення. Це вирішує проблему прозорості та дозволяє аргументувати відмови клієнтам.

Розроблено модуль «Розумний порадник». Впроваджено алгоритм контрфактуального аналізу, який у разі відмови автоматично моделює

альтернативні сценарії та генерує рекомендації, наприклад, «збільшити термін кредитування на 12 місяців». Це становить практичну цінність роботи, оскільки трансформує систему з інструменту контролю в інструмент пошуку рішень.

Експериментальні дослідження в межах роботи підтвердили високу якість розробленої моделі та ефективність запропонованого підходу до налаштування гіперпараметрів. Практичне значення одержаних результатів полягає у створенні повністю готового до експлуатації прототипу програмного забезпечення, розгорнутого в ізольованому Docker-контейнері. Впровадження такої системи у діяльність фінансових установ дозволить не лише знизити операційні ризики, а й суттєво покращити клієнтський досвід завдяки функціоналу «Розумного порадника» та дотриманню сучасних стандартів прозорості штучного інтелекту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Regulation 2016/679 of the European Parliament and of the Council on the protection of natural persons with regard to the processing of personal data: від 04.05.2016, № 2016/679.
2. Proposal for a Regulation of the European Parliament and of the Council laying down harmonised rules on artificial intelligence.: від 17.06.2021, № 2021/956.
3. Bracke P. Machine learning explainability in finance: an application to default risk analysis [Електронний ресурс] / P. Bracke, A. Datta, C. Jung, S. Sen // Bank of England Staff Working Paper No. 816. – 2019. – Режим доступу до статті: <https://www.bankofengland.co.uk/working-paper/2019/machine-learning-explainability-in-finance-an-application-to-default-risk-analysis>.
4. Bussmann N. Explainable Machine Learning in Credit Risk Management [Електронний ресурс] / N. Bussmann, P. Giudici, D. Marinelli // Computational Economics. – 2021. – Vol. 57. – Режим доступу до статті: <https://link.springer.com/article/10.1007/s10614-020-10042-0>.
5. Lundberg S. M. A Unified Approach to Interpreting Model Predictions [Електронний ресурс] / S. M. Lundberg, S.-I. Lee // Advances in Neural Information Processing Systems (NIPS). – 2017. – Режим доступу до статті: <https://papers.nips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html>.
6. Ribeiro M. T. "Why Should I Trust You?": Explaining the Predictions of Any Classifier [Електронний ресурс] / M. T. Ribeiro, S. Singh, C. Guestrin // Proceedings of the 22nd ACM SIGKDD. – 2016. – Режим доступу до статті: <https://arxiv.org/abs/1602.04938>.
7. Arrieta A. B. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges [Електронний ресурс] / A. B. Arrieta [et al.] // Information Fusion. – 2020. – Vol. 58. – Режим доступу до статті: <https://www.sciencedirect.com/science/article/pii/S156625351930810X>.

8. Pedregosa F. Scikit-learn: Machine Learning in Python [Електронний ресурс] / F. Pedregosa [et al.] // Journal of Machine Learning Research. – 2011. – Vol. 12. – Режим доступу до статті: <https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html>.
9. Streamlit Documentation [Електронний ресурс]. – Режим доступу до ресурсу: <https://docs.streamlit.io>.
10. Breiman L. Random Forests / L. Breiman // Machine Learning. – 2001. – Vol. 45, No. 1. – P. 5–32.
11. Wachter S. Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR [Електронний ресурс] / S. Wachter, B. Mittelstadt, C. Russell // Harvard Journal of Law & Technology. – 2017. – Режим доступу до статті: <https://arxiv.org/abs/1711.00399>.
12. Shapley L. S. A Value for n-person Games / L. S. Shapley // Contributions to the Theory of Games. – 1953. – Vol. 2. – P. 307–317.
13. Biau G. A random forest guided tour [Електронний ресурс] / G. Biau, E. Scornet // TEST. – 2016. – Vol. 25. – Режим доступу до статті: <https://arxiv.org/pdf/1511.05741.pdf>.
14. German Credit Data Set [Електронний ресурс] // UCI Machine Learning Repository. – Режим доступу до ресурсу: [https://archive.ics.uci.edu/ml/datasets/statlog+\(german+credit+data\)](https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)).
15. NumPy Documentation [Електронний ресурс]. – Режим доступу до ресурсу: <https://numpy.org/doc/stable/>.
16. SHAP Documentation [Електронний ресурс]. – Режим доступу до ресурсу: <https://shap.readthedocs.io/en/latest/>.
17. Pandas: Powerful Python Data Analysis Toolkit [Електронний ресурс]. – Режим доступу до ресурсу: <https://pandas.pydata.org/docs/>.
18. Python Software Foundation [Електронний ресурс]. – Офіційний сайт мови програмування Python. – Режим доступу до ресурсу: <https://www.python.org/>.

19. Fawcett T. An introduction to ROC analysis [Электронный ресурс] / T. Fawcett // Pattern Recognition Letters. – 2006. – Vol. 27. – Режим доступа до статті: <https://doi.org/10.1016/j.patrec.2005.10.010>.
20. Scikit-learn: Receiver Operating Characteristic [Электронный ресурс]. – Режим доступа до ресурсу: https://scikit-learn.org/stable/modules/model_evaluation.html#receiver-operating-characteristic-roc.
21. Docker Documentation [Электронный ресурс]. – Режим доступа до ресурсу: <https://docs.docker.com/>.

ДОДАТКИ

Налаштування сторінки та імпорт бібліотек

```
import streamlit as st
import pandas as pd
import joblib
import numpy as np
import shap
import matplotlib.pyplot as plt

st.set_page_config(page_title="Credit XAI System", layout="wide")

st.title("Система оцінки кредитних ризиків (XAI)")
st.markdown("""
Цей додаток демонструє роботу Explainable AI.
Додаток не просто прогнозує ризик, а пояснює кожне рішення моделі.
""")
```

Завантаження моделі та дани

```
@st.cache_resource
def load_data_and_model():
    model = joblib.load('credit_model.pkl')
    data = pd.read_csv('german_credit_data_subset.csv')
    X = data.drop('Risk', axis=1)
    return model, X

model, X_train = load_data_and_model()
@st.cache_resource
def get_explainer(_model):
    return shap.TreeExplainer(_model)

explainer = get_explainer(model)
```

Виведення бокової панелі

```

st.sidebar.header("Дані клієнта")

def user_input_features():
    age = st.sidebar.slider("Вік", 19, 75, 30)
    credit_amount = st.sidebar.number_input("Сума кредиту ($)", 500, 15000, 5000)
    duration = st.sidebar.slider("Термін (місяців)", 6, 72, 24)

    housing_display = ("Орендоване", "Власне", "Безкоштовне")
    housing_values = (0, 1, 2)
    housing_option = st.sidebar.selectbox("Житло", housing_display, index=1)
    housing_val = housing_values[housing_display.index(housing_option)]

    job_display = ("Некваліфікований", "Кваліфікований", "Керівник/Бізнес")
    job_values = (0, 1, 2)
    job_option = st.sidebar.selectbox("Робота", job_display, index=1)
    job_val = job_values[job_display.index(job_option)]

    purpose_display = ("Автомобіль", "Меблі/Техніка", "Радіо/ТБ", "Бізнес",
"Освіта")
    purpose_values = (0, 1, 2, 3, 4)
    purpose_option = st.sidebar.selectbox("Мета кредиту", purpose_display)
    purpose_val = purpose_values[purpose_display.index(purpose_option)]

    data = {
        'Age': age,
        'Credit Amount': credit_amount,
        'Duration (months)': duration,
        'Housing': housing_val,
        'Job': job_val,
        'Purpose': purpose_val
    }
    return pd.DataFrame(data, index=[0])

input_df = user_input_features()

```

Головний екран та розрахунок

```

col1, col2 = st.columns([1, 2])

with col1:
    st.subheader("1. Параметри заявки")
    st.write(input_df.T)

if st.button("Аналізувати та Пояснити"):
    prediction_proba = model.predict_proba(input_df)
    risk_probability = prediction_proba[0][1]

    st.divider()
    st.subheader("2. Вердикт моделі")

    c1, c2, c3 = st.columns(3)
    c1.metric("Ризик дефолту", f"{risk_probability:.1%}")

    if risk_probability > 0.55:
        c2.error("ВІДМОВА")
    else:
        c2.success("ПОГОДЖЕНО")

    st.progress(risk_probability)

    st.divider()
    st.subheader("3. Пояснення ХАІ: Чому таке рішення?")
    st.info("Червоний колір = підвищує ризик. Синій колір = знижує ризик.")

    shap_values = explainer(input_df)
    explanation = shap_values[0, :, 1]
    fig, ax = plt.subplots(figsize=(8, 5))
    shap.plots.waterfall(explanation, show=False)
    st.pyplot(fig)

    st.markdown("""
    **E[f(x)]** - це базова ймовірність (середня по лікарні).
    Графік показує, як характеристики саме цього клієнта змінили оцінку від
    середньої до фінальної.
    """)

```

Додаток Ж

Тренування моделі

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
import joblib

def create_mock_german_data(n_samples=1000):
    np.random.seed(42)
    data = pd.DataFrame()
    data['Age'] = np.random.randint(19, 75, n_samples)
    data['Credit Amount'] = np.random.randint(500, 15000, n_samples)
    data['Duration (months)'] = np.random.randint(6, 72, n_samples)
    data['Housing'] = np.random.choice([0, 1, 2], n_samples, p=[0.2, 0.7,
0.1])

    data['Job'] = np.random.choice([0, 1, 2], n_samples, p=[0.2, 0.6, 0.2])
    data['Purpose'] = np.random.choice([0, 1, 2, 3, 4], n_samples)
    risk_score = (
        (data['Credit Amount'] / 15000) * 0.4 +
        (1 - data['Age'] / 75) * 0.3 +
        (data['Duration (months)'] / 72) * 0.3
    )
    risk_score += np.random.normal(0, 0.1, n_samples)
    data['Risk'] = (risk_score > 0.55).astype(int)

    return data

print("1. Генеруємо dataset...")
df = create_mock_german_data()
df.to_csv('german_credit_data_subset.csv', index=False)
print("    -> Файл 'german_credit_data_subset.csv' створено.")

print("2. Навчаємо модель...")
X = df.drop('Risk', axis=1)
y = df['Risk']

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

model = RandomForestClassifier(n_estimators=100, max_depth=5, random_state=42)
model.fit(X_train, y_train)

acc = model.score(X_test, y_test)
print(f"    -> Точність моделі: {acc:.2f}")
```

```
joblib.dump(model, 'credit_model.pkl')
```

Додаток Д

Глобальне пояснення моделі

```
# В. Глобальне пояснення
with st.expander(" Глобальна логіка моделі"):
    st.write("Які фактори найважливіші загалом?")

    # Беремо 50 випадкових людей для швидкості
    shap_values_global = explainer(X_train.sample(50))

    # --- ВИПРАВЛЕННЯ ТУТ ТАКОЖ ---
    # Беремо всі рядки (50 штук), всі колонки, і тільки клас 1
    explanation_global = shap_values_global[:, :, 1]

    fig_global, ax_global = plt.subplots()
    shap.plots.beeswarm(explanation_global, show=False)
    st.pyplot(fig_global)
```

Контрфактуальний аналіз

```

st.subheader("4. Поради для підвищення результату ")

if risk_probability > 0.55:
    temp_df = input_df.copy()
    found_solution = False

    current_duration = temp_df['Duration (months)'][0]

    for increase in [6, 12, 18, 24]:
        new_duration = current_duration + increase
        if new_duration <= 72:
            temp_df['Duration (months)'] = new_duration
            new_prob = model.predict_proba(temp_df)[0][1]

            if new_prob < 0.55:
                st.success(f"***Рекомендація:** Якщо збільшити термін кредиту
до **{new_duration} місяців** (зараз {current_duration}), ризик знизиться до
{new_prob:.1%}, і кредит може бути погоджено!")
                found_solution = True
                break

    if not found_solution:
        current_amount = temp_df['Credit Amount'][0]
        temp_df['Duration (months)'] = current_duration

        for decrease in [1000, 2000, 3000]:
            new_amount = current_amount - decrease
            if new_amount > 500:
                temp_df['Credit Amount'] = new_amount
                new_prob = model.predict_proba(temp_df)[0][1]

                if new_prob < 0.55:
                    st.success(f"***Рекомендація:** Спробуйте зменшити суму
кредиту до **{new_amount}$**, це знизить ризик до прийнятного рівня
({new_prob:.1%}).")
                    found_solution = True
                    break

    if not found_solution:
        st.info("На жаль, прості зміни параметрів (сума/термін) не дозволяють
знизити ризик до прийнятного рівня. Потрібен розгляд поручителя.")

else:
    st.success("Клієнт надійний. Додаткові дії не потрібні.")

```

```

with tab2:
    st.header("Аналіз ефективності моделі")
    st.markdown("Метрики якості класифікації на основі історичних даних.")

    if y_dataset is not None:
        y_pred = model.predict(X_dataset)
        y_prob = model.predict_proba(X_dataset)[: , 1]

        m1, m2, m3, m4 = st.columns(4)
        m1.metric("Accuracy (Точність)", f"{accuracy_score(y_dataset,
y_pred):.2f}")
        m2.metric("F1 Score", f"{f1_score(y_dataset, y_pred):.2f}")
        m3.metric("Precision", f"{precision_score(y_dataset, y_pred):.2f}")
        m4.metric("Recall", f"{recall_score(y_dataset, y_pred):.2f}")

        st.divider()

        col_graph1, col_graph2 = st.columns(2)

        with col_graph1:
            st.subheader("Матриця помилок")
            st.write("Показує, де саме помиляється модель.")

            cm = confusion_matrix(y_dataset, y_pred)
            fig_cm, ax_cm = plt.subplots()
            disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=["Надійний", "Ризиковий"])
            disp.plot(cmap=plt.cm.Blues, ax=ax_cm)
            st.pyplot(fig_cm)

        with col_graph2:
            st.subheader("ROC-крива")
            st.write("Оцінка якості розділення класів. Чим більше площа AUC, тим
краще.")

            fpr, tpr, thresholds = roc_curve(y_dataset, y_prob)
            roc_auc = auc(fpr, tpr)

            fig_roc, ax_roc = plt.subplots()
            ax_roc.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve
(area = {roc_auc:.2f})')
            ax_roc.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
            ax_roc.set_xlim([0.0, 1.0])
            ax_roc.set_ylim([0.0, 1.05])
            ax_roc.set_xlabel('False Positive Rate')
            ax_roc.set_ylabel('True Positive Rate')
            ax_roc.legend(loc="lower right")

```

```
st.pyplot(fig_roc)

else:
    st.error("У завантаженому датасеті відсутня колонка 'Risk'. Розрахунок метрик неможливий.")
```