ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«**Методи машинного навчання для аналізу структурованих і неструктурованих даних**»

на здобуття освітнього ступеня магістра

зі спеціальності 124 Системний Аналіз
(код, найменування спеціальності)

освітньо-професійної програми «Інтелектуальні системи управління»
(назва)

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

_____ Владислав ІВАХНЕНКО
(підпис) *Ім'я ПРІЗВИЩЕ здобувача*

Виконав: здобувач(ка) вищої освіти гр. САДМ-61
Владислав ІВАХНЕНКО
Ім'я ПРІЗВИЩЕ

Керівник: Шушура О. М.
д.т.н., проф. Ім'я ПРІЗВИЩЕ

Рецензент: _____
науковий ступінь, вчене звання Ім'я ПРІЗВИЩЕ

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра Інформаційних систем та технологій
Ступінь вищої освіти Магістр
Спеціальність 124 Системний аналіз
Освітньо-професійна програма Інтелектуальні системи управління

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІСТ
Каміла СТОРЧАК
«___» _____ 20__ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Івахненку Владиславу Анатолійовичу
(прізвище, ім'я та по батькові здобувача)

1. Тема кваліфікаційної роботи: «Методи машинного навчання для аналізу структурованих і неструктурованих даних»

керівник кваліфікаційної роботи Олексій ШУШУРА д.т.н., проф.,,
(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)

затверджені наказом Державного університету інформаційно-комунікаційних технологій
від «30» жовтня 2025 р. № 467

2. Строк подання кваліфікаційної роботи «26» грудня 2025р.

3. Вихідні дані до кваліфікаційної роботи: German Credit Dataset (1000 позичальників, 20 фінансових атрибутів); EmoBench-UA для класифікації емоцій; згенеровані україномовні текстові відповіді позичальників; алгоритми Random Forest, Gradient Boosting, XGBoost;

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Проаналізувати методи обробки структурованих даних та для неструктурованих даних.
2. Сформувати базу дослідження на основі German Credit Dataset.
3. Розробити архітектуру гібридної моделі. Реалізувати програмний комплекс мовою Python.
5. Перелік ілюстративного матеріалу: *презентація*

6. Дата видачі завдання «30» жовтня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання	Примітка
1	Аналіз літературних джерел та формулювання завдань дослідження	30.10.2025 – 06.11.2025	
2	Теоретико-методичний аналіз методів машинного навчання для структурованих та неструктурованих даних	07.11.2025 – 20.11.2025	
3	Формування бази дослідження. Порівняльний аналіз алгоритмів машинного навчання	21.11.2025 – 30.11.2025	
4	Апробація методів обробки природної мови. Оцінка впливу текстових ознак	01.12.2025 – 07.12.2025	
5	Розробка архітектури гібридної моделі кредитного скорингу	08.12.2025 – 12.12.2025	
6	Реалізація програмного комплексу мовою Python	13.12.2025 – 18.12.2025	
7	Експериментальна перевірка інтегрованої моделі. Аналіз результатів	19.12.2025 – 22.12.2025	
8	Оформлення пояснювальної записки та підготовка до захисту	23.12.2025 – 26.12.2025	

Здобувач(ка) вищої освіти _____ Владислав ІВАХНЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник
кваліфікаційної роботи _____ Олексій ШУШУРА
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 103 стор., 27 рис., 2 табл., 46 джерел.

Мета роботи – дослідження методів машинного навчання для аналізу структурованих і неструктурованих даних.

Об'єкт дослідження – процеси аналізу структурованих та неструктурованих даних методами машинного навчання.

Предмет дослідження – методи машинного навчання для інтеграції та спільної обробки структурованих даних і текстової інформації в задачах класифікації.

Короткий зміст роботи:

У першому розділі систематизовано методи обробки структурованих даних (лінійні моделі, дерева рішень, ансамблеві методи) та архітектури для неструктурованих текстів (від bag-of-words до трансформерних моделей). Обґрунтовано вибір XGBoost для структурованих даних та XLM-RoBERTa для екстракції лінгвістичних ознак з україномовних текстів.

У другому розділі сформовано базу дослідження через об'єднання German Credit Dataset (1000 позичальників, 20 атрибутів) із згенерованими україномовними відповідями. Виконано порівняльний аналіз алгоритмів: XGBoost досягнув точності 0,787. Апробовано XLM-RoBERTa для класифікації емоцій: точність 0,844 на EmoBench-UA. Екстраговано 18 емоційних характеристик.

У третьому розділі розроблено гібридну модель (20 фінансових + 36 емоційних ознак) з cost-sensitive навчанням. Реалізовано програмний комплекс Python. Встановлено: гібридна модель досягла ROC-AUC 0,795 проти 0,760, зменшила вартість помилок на 54,2%.

КЛЮЧОВІ СЛОВА: ML, СТРУКТУРОВАНІ ДАНІ, НЕСТРУКТУРОВАНІ ДАНІ, ОБРОБКА ПРИРОДНОЇ МОВИ, КРЕДИТНИЙ СКОРИНГ, XGBOOST, XLM-ROBERTA, EMOBENCH-UA, ІНТЕГРОВАНА МОДЕЛЬ, ЕМОЦІЙНІ ОЗНАКИ.

ABSTRACT

Text part of the master's qualification work: 103 pages, 27 pictures, 2 tables, 46 sources.

The purpose of the work – investigation of machine learning methods for structured and unstructured data analysis.

Object of research – processes of structured and unstructured data analysis using machine learning methods.

Subject of research – machine learning methods for integration and joint processing of structured data and textual information in classification tasks.

Summary of the work:

In the first chapter, methods for processing structured data (linear models, decision trees, ensemble methods) and architectures for unstructured texts (from bag-of-words to transformer models) are systematized. The choice of XGBoost for structured data and XLM-RoBERTa for extraction of linguistic features from Ukrainian texts is justified.

In the second chapter, a research database is formed by combining the German Credit Dataset (1000 borrowers, 20 attributes) with generated Ukrainian responses. A comparative analysis of algorithms is performed: XGBoost achieved accuracy of 0.787. XLM-RoBERTa is tested for emotion classification: accuracy of 0.844 on EmoBench-UA. 18 emotional features are extracted.

In the third chapter, a hybrid model (20 financial + 36 emotional features) with cost-sensitive learning is developed. A Python software complex is implemented. It is established: the hybrid model achieved ROC-AUC of 0.795 versus 0.760, reduced error cost by 54.2%.

KEYWORDS: MACHINE LEARNING, STRUCTURED DATA, UNSTRUCTURED DATA, NATURAL LANGUAGE PROCESSING, CREDIT SCORING, XGBOOST, XLM-ROBERTA, EMOBENCH-UA, INTEGRATED MODEL, EMOTIONAL FEATURES.

ЗМІСТ

ВСТУП

Актуальність теми. Зростання обсягів даних у різних галузях діяльності супроводжується якісними змінами у структурі інформаційних ресурсів. Структуровані дані, що традиційно зберігаються у реляційних базах та електронних таблицях, становлять лише частину загального інформаційного масиву. За прогнозами IDC, неструктуровані дані складатимуть 80% усіх світових даних до 2025 року, при цьому лише близько 0,5-1% неструктурованих даних використовується для аналізу та прийняття рішень [1]. Ця невідповідність між наявністю даних та їх практичним застосуванням формує запит на розробку методів, здатних ефективно обробляти різнорідну інформацію.

Методи машинного навчання демонструють можливості щодо аналізу як структурованих, так і неструктурованих даних. Алгоритми Random Forest, Gradient Boosting та інші ансамблеві методи застосовуються для роботи з табличними даними, тоді як трансформерні архітектури на кшталт BERT дозволяють виділяти семантичні характеристики з текстових джерел. Питання інтеграції цих підходів та оцінки їхнього впливу на якість прогнозування залишаються предметом досліджень у різних прикладних областях.

Аналіз останніх досліджень і публікацій. Інтеграція структурованих та неструктурованих даних розглядається у низці наукових праць останніх років. Дослідження Zhang et al. [2] спрямоване на застосування BERT-моделі для обробки текстових описів симптомів пацієнтів у відділеннях невідкладної допомоги. Автори демонстрували, що поєднання структурованих клінічних показників із текстовими даними підвищує точність класифікації на 5-8% порівняно з моделями, що використовують лише один тип даних. Використання Gradient Boosting на комбінованих даних досягло AUC 0.789, що перевищило результати моделей на окремих типах даних.

У роботі [3] запропоновано гібридну методологію для аналізу даних у контексті громадської безпеки, де структуровані дані від державних агентств поєднувалися з неструктурованими повідомленнями з соціальних мереж.

Методологія передбачала класифікацію текстових повідомлень та їх трансформацію у структуровані ознаки для подальшого використання в алгоритмах прогнозування. Точність алгоритмів зросла на 80% після інтеграції різнорідних джерел даних.

Дослідження [4] присвячене застосуванню методу Latent Dirichlet Allocation для виділення тематичних ознак з клінічних записів та їх поєднання зі структурованими демографічними та клінічними параметрами пацієнтів. Результати показали покращення прогнозування летальності при використанні комбінованого підходу порівняно з моделями на основі тільки структурованих даних.

Наявні дослідження демонструють переваги інтеграції різнорідних даних у конкретних прикладних задачах, проте систематичне порівняння методів машинного навчання для обох типів даних та аналіз впливу текстових ознак на загальну точність моделей потребує подальшого вивчення, особливо у контексті українського наукового середовища.

Мета і завдання дослідження. Метою роботи є дослідження методів машинного навчання для аналізу структурованих і неструктурованих даних та розробка інтегрованої моделі на прикладі задачі кредитного скорингу.

Для досягнення мети сформульовано наступні **завдання**:

1. Проаналізувати концептуальні основи та класифікацію методів обробки структурованих даних.
2. Розглянути архітектури машинного навчання для неструктурованих текстових даних.
3. Обрати методики для інтегрованого аналізу різнорідних даних.
4. Сформувати базу дослідження на основі синтетичних даних.
5. Порівняти ефективність методів машинного навчання для структурованих даних.
6. Апробувати методи обробки природної мови для виділення лінгвістичних ознак.
7. Оцінити вплив текстових ознак на точність прогнозування.

8. Розробити інтегровану модель, що поєднує структуровані та неструктуровані дані.

9. Реалізувати програмний комплекс для демонстрації методів машинного навчання.

10. Проаналізувати отримані результати дослідження.

Об'єкт дослідження. Об'єктом дослідження є процеси аналізу структурованих та неструктурованих даних методами машинного навчання.

Предмет дослідження. Предметом дослідження є методи машинного навчання для інтеграції та спільної обробки структурованих даних і текстової інформації в задачах класифікації.

Методи дослідження. У роботі використано метод порівняльного аналізу для оцінки ефективності різних підходів машинного навчання, метод експериментального дослідження для апробації алгоритмів на синтетичних даних, метод систематизації для узагальнення теоретичних основ обробки структурованих та неструктурованих даних. Для оцінки якості моделей застосовано метод кількісного аналізу із використанням метрик точності, повноти, F1-міри та AUC-ROC. Практична реалізація здійснена з використанням методу прототипування програмного комплексу. У дослідженні застосовано алгоритми Random Forest, Gradient Boosting, XGBoost для структурованих даних та трансформерні моделі BERT, DistilBERT для текстової інформації.

Наукова новизна отриманих результатів. Наукова новизна дослідження полягає у систематичному порівнянні методів машинного навчання для аналізу структурованих та неструктурованих даних у рамках єдиної задачі класифікації. Вперше запропоновано інтегровану модель кредитного скорингу, що поєднує фінансові показники клієнта із психо-емоційними характеристиками, виділеними з текстових відповідей методами обробки природної мови. Отримало подальший розвиток застосування трансформерних архітектур для формування додаткових ознак у задачах фінансового аналізу.

Практична значущість отриманих результатів. Розроблена інтегрована система може застосовуватися мікрофінансовими організаціями для оцінки ризику

дефолту на основі комплексного аналізу фінансових параметрів та психо-емоційного стану позичальника. Програмний комплекс забезпечує можливість завантаження даних, проведення тестування клієнтів, обробки текстових відповідей та візуалізації результатів прогнозування. Результати дослідження можуть використовуватися для вдосконалення процесів прийняття рішень у кредитних організаціях та слугувати основою для подальших наукових розробок у галузі інтеграції різнорідних даних.

1 ТЕОРЕТИКО-МЕТОДИЧНИЙ АНАЛІЗ ПІДХОДІВ МАШИННОГО НАВЧАННЯ ДО РОБОТИ З РІЗНОРІДНИМИ ДАНИМИ

1.1. Концептуальні основи та класифікація методів обробки структурованих даних

Структуровані дані визначаються як інформація, організована у заздалегідь визначеному форматі, що забезпечує зберігання, пошук та аналіз [2]. Організація базується на схемі, яка описує типи даних, взаємозв'язки між елементами та структурні правила.

Характеристикою структурованих даних є табличне представлення у форматі рядків та стовпців. У цій структурі кожен рядок відповідає окремому запису, тоді як стовпці представляють атрибути цього запису. Клітинка на перетині рядка та стовпця містить елемент даних, що характеризує властивість об'єкта.

Структуровані дані зберігаються у реляційних базах даних, електронних таблицях та файлах CSV. Реляційні системи управління базами даних організовують дані у таблиці з визначеною схемою, що описує взаємозв'язки між таблицями через первинні та зовнішні ключі. Це дозволяє встановлювати зв'язки між наборами даних та забезпечує цілісність інформації.

Схема даних визначає модель зберігання, обробки та доступу до інформації. Завдяки моделі даних кожне поле може бути доступне окремо або спільно з даними з інших полів. Це забезпечує можливість пошуку, фільтрації та маніпулювання даними за допомогою мови запитів SQL (Structured Query Language).

Типова структура табличних даних представлена на рисунку 1.1, де демонструється організація інформації у форматі реляційної бази даних з розподілом на записи та атрибути.

CUSTOMER_ID	LAST_NAME	FIRST_NAME	STREET	CITY	ZIP_CODE	COUNTRY
10302	Boucher	Leo	54, rue Royale	Nantes	44000	France
11244	Smith	Laurent	8489 Strong St	Las Vegas	83030	USA
11405	Han	James	636 St Kilda Road	Sydney	3004	Australia
11993	Mueller	Tomas	Berliner Weg 15	Tamm	71732	Germany
12111	Carter	Nataly	5 Tomahawk	Los Angeles	90006	USA
14121	Cortez	Nola	Av. Grande, 86	Madrid	28034	Spain
14400	Brown	Frank	165 S 7th St	Chester	33134	USA
14578	Wilson	Sarah	Seestreet #6101	Emory	1734	USA
14622	Jones	John	71 San Diego Ave	Arlington	69004	USA

Рис. 1.1 Приклад структури табличних даних у форматі рядків та стовпців

Усі записи в межах одного набору даних містять однакові атрибути. Наприклад, кожен запис про клієнта може включати ідентифікатор, ім'я, адресу та контактний телефон. Таблиці структурованих даних мають спільні значення, що пов'язують різні набори даних між собою через ключові поля. Це дозволяє встановлювати зв'язки між таблицями та виконувати операції об'єднання даних.

Структури даних класифікуються на примітивні та непримітивні (рисунок 1.2). Примітивні структури даних включають базові типи, які підтримуються безпосередньо мовами програмування: цілочисельний тип (Integer), тип з плаваючою комою (Float), символьний тип (Character) та логічний тип (Boolean). Непримітивні структури поділяються на лінійні та нелінійні.

Лінійні структури даних організовують елементи послідовно, де кожен елемент з'єднаний з попереднім та наступним. До статичних лінійних структур відноситься масив (Array), де елементи розміщуються у фіксованому блоці пам'яті. Динамічні лінійні структури включають зв'язаний список (Linked List), стек (Stack) та чергу (Queue), які дозволяють змінювати розмір під час виконання програми.

Нелінійні структури даних організовують елементи ієрархічно або у вигляді зв'язків між вузлами. До них належать дерева (Tree), які представляють ієрархічні відношення між елементами, та графи (Graph), що моделюють довільні зв'язки між об'єктами.

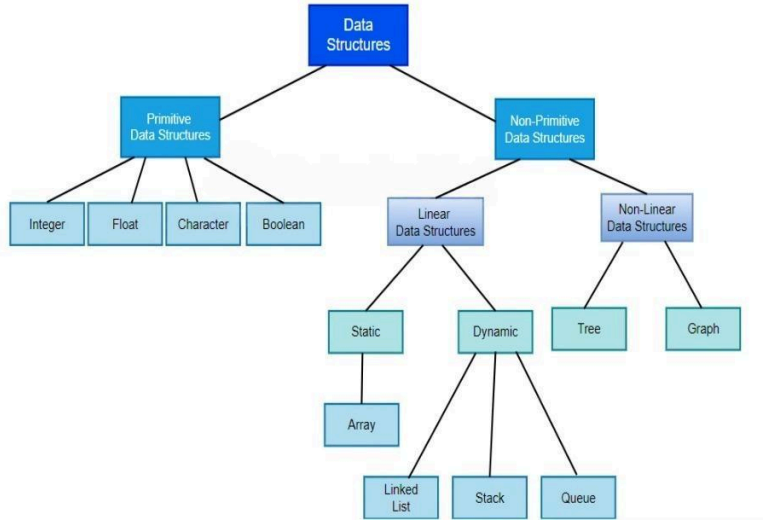


Рис. 1.2 Класифікація типів ознак у структурованих даних

У контексті машинного навчання робота зі структурованими даними передбачає використання табличних форматів, де примітивні типи даних формують ознаки для алгоритмів. Числові ознаки представляють кількісні виміри та можуть бути безперервними або дискретними. Приклади включають вік, дохід, температуру або кількість транзакцій.

Категоріальні ознаки описують якісні характеристики та поділяються на номінальні та порядкові [6]. Номінальні категорії не мають внутрішнього порядку, що показано на прикладі кольору, типу продукту або географічного розташування. Порядкові категорії мають послідовність, як рівень освіти або рейтинг задоволеності [7]. Бінарні ознаки містять два можливі значення, що представляються як 0 та 1 або «так» та «ні».

Структуровані дані мають переваги порівняно з іншими типами даних. Вони доступні для користувачів без глибоких знань у галузі науки про дані. Операції оновлення та модифікації даних є прямолінійними. Зберігання може використовувати фіксовані блоки пам'яті для значень даних. Структуровані дані підходять для математичного аналізу, дозволяючи підраховувати частоту атрибутів та виконувати математичні операції над числовими даними.

Водночас структуровані дані мають обмеження. Попередньо визначена структура створює обмеження у гнучкості використання. Структуровані дані можуть використовуватися для їх передбаченого призначення без додаткових модифікацій. Зміна схеми у відповідь на нові вимоги потребує реструктуризації великих обсягів інформації.

Для використання структурованих даних у задачах прогнозування та класифікації застосовуються різноманітні методи машинного навчання, що відрізняються за способами побудови моделей та типами розв'язуваних задач.

Методи машинного навчання, що застосовуються для аналізу структурованих даних, класифікуються за характером навчання, типом прогнозованої змінної, способом побудови моделі та рівнем автоматизації процесу моделювання. Основними класами є навчання з учителем (supervised learning), навчання без учителя (unsupervised learning), напівконтрольоване навчання (semi-supervised learning) та підкріплене навчання (reinforcement learning) [8].

У контексті структурованих даних найчастіше використовуються методи навчання з учителем, де модель будується на основі пар «вхідні ознаки – цільове значення». Ці методи застосовуються для задач класифікації та регресії [9].

На рисунку 1.3 наведено узагальнену класифікацію методів машинного навчання, що застосовуються для аналізу структурованих табличних даних.

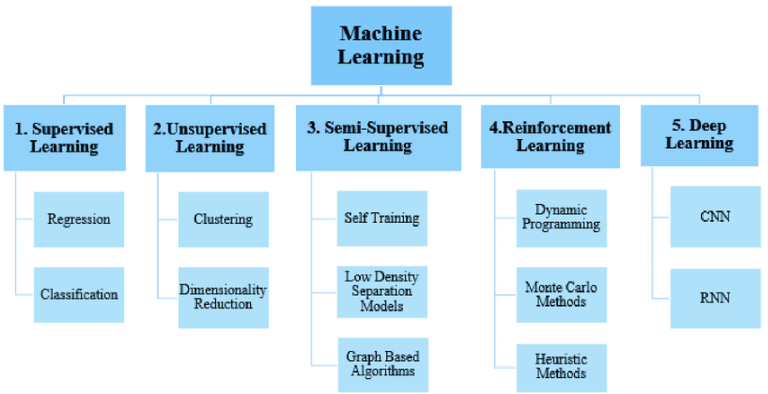


Рис. 1.3 Класифікація методів машинного навчання для структурованих даних

До групи навчання з учителем належать методи, у яких модель формує функціональну залежність між вхідними ознаками X та цільовою змінною y. Серед основних категорій таких алгоритмів можна виокремити кілька підходів.

Лінійні моделі застосовуються для встановлення лінійних залежностей між змінними, до них належать лінійна регресія, логістична регресія, а також ridge- і lasso-регресії. Їхньою перевагою є висока інтерпретованість та простота реалізації, що робить ці методи базовими для фінансових, економічних та аналітичних задач.

Іншим поширеним підходом є дерева рішень (Decision Trees), які побудовані на принципі рекурсивного поділу простору ознак і визначають правила у вигляді ієрархічної структури вузлів. Критерії поділу зазвичай базуються на мінімізації ентропії або приросту інформації. Такі моделі часто використовуються як базові компоненти для побудови ансамблевих алгоритмів [10].

Методи найближчих сусідів (k-Nearest Neighbors, KNN) класифікують об’єкти за подібністю до інших прикладів у просторі ознак. Вони не потребують процесу навчання у класичному розумінні, однак є чутливими до масштабування змінних і вибору метрики відстані.

Методи опорних векторів (Support Vector Machines, SVM) будують оптимальну гіперплощину, яка розділяє класи з максимальним зазором (margin). Цей підхід добре підходить для високовимірних структурованих наборів і забезпечує високу точність при обмеженій кількості даних.

Окрему категорію становить наївний байєсівський класифікатор (Naive Bayes), який базується на ймовірнісній моделі згідно з теоремою Байєса та припущенням незалежності ознак. Він відзначається ефективністю при роботі з великими наборами категоріальних змінних і часто використовується для задач первинної класифікації текстових або транзакційних даних.

Методи навчання без учителя не потребують цільової змінної та призначені для виявлення структур, кластерів, аномалій або зменшення розмірності простору ознак [11].

До основних підходів належать:

- Кластеризація. Методи k-Means, DBSCAN та ієрархічна кластеризація групують записи на основі метрик відстані (евклідова, косинусна тощо).
- Зниження розмірності. PCA (Principal Component Analysis) та t-SNE дозволяють спростити простір ознак, зберігаючи найважливішу інформацію.
- Виявлення аномалій. Алгоритми Isolation Forest та One-Class SVM використовуються для знаходження нетипових записів, наприклад, у транзакційних або фінансових даних.

Крім базових категорій, виділяють методи напівконтрольованого та підкріпленого навчання, які менш поширені для табличних даних, але використовуються у випадках часткової розмітки або для динамічних рішень у часі (наприклад, у кредитному скорингу чи прогнозуванні ризиків).

Сучасна тенденція розвитку роботи зі структурованими даними полягає у використанні систем AutoML (Automated Machine Learning). Ці платформи здійснюють автоматичний підбір алгоритмів, гіперпараметрів і процедур підготовки даних.

Приклади реалізацій:

- Google AutoML Tables,
- H2O.ai AutoML,
- Auto-sklearn – відкрита платформа, що застосовує мета-навчання для вибору оптимальних моделей [12].

Таким чином, класифікація методів машинного навчання для структурованих даних охоплює спектр від простих статистичних моделей до ансамблевих і автоматизованих систем. Серед розглянутих підходів ансамблеві методи демонструють підвищену точність та стабільність прогнозів за рахунок поєднання результатів кількох базових моделей.

Ансамблеві методи є стратегією у машинному навчанні, що дозволяє підвищити точність і стабільність прогнозів шляхом поєднання кількох базових моделей. Основна ідея ансамблю полягає в тому, що об'єднання результатів кількох слабших моделей дає змогу отримати більш узагальнену та стійку модель, ніж будь-яка з них окремо [13].

Залежно від принципу об’єднання результатів розрізняють беггінг (bagging), бустинг (boosting) та стекинг (stacking). Ці підходи реалізують різні стратегії навчання, проте мають спільну мету – зниження дисперсії та зміщення моделі, а також покращення її узагальнювальної здатності.

На рисунку 1.4 наведено узагальнену схему основних типів ансамблевих методів машинного навчання, яка демонструє логіку побудови моделей та способи комбінування результатів окремих алгоритмів.

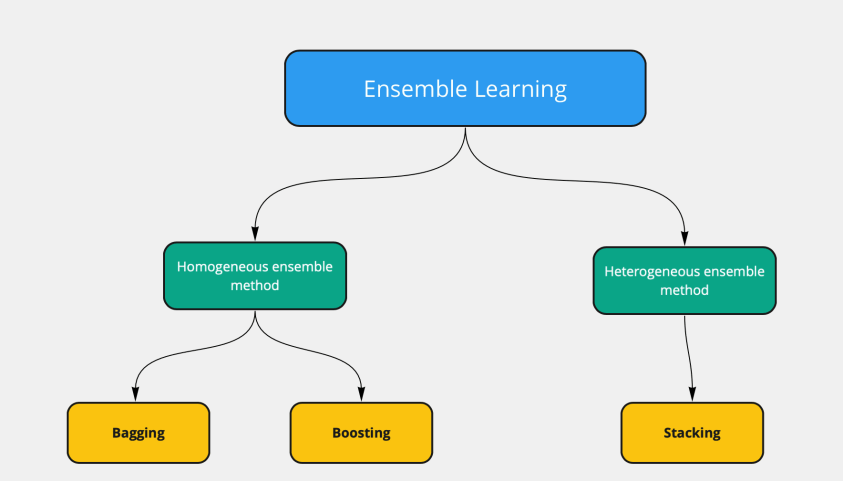


Рис. 1.4 Основні типи ансамблевих методів машинного навчання

Беггінг ґрунтується на створенні кількох незалежних моделей, які навчаються на випадкових підвибірках одного й того самого набору даних. Результати цих моделей об’єднуються, зазвичай, шляхом усереднення (для регресії) або голосування (для класифікації). Найвідомішим представником цього підходу є Random Forest, який поєднує велику кількість дерев рішень, що навчаються на різних підмножинах ознак і спостережень. Такий підхід зменшує

ризик перенавчання, підвищує стійкість до шуму й дозволяє оцінювати важливість окремих ознак.

Бустинг передбачає послідовне навчання серії моделей, кожна з яких намагається компенсувати помилки попередніх. На відміну від беггінгу, бустинг створює ланцюгову структуру залежностей між моделями, завдяки чому досягається висока точність прогнозування навіть за невеликих обсягів даних. Серед найвідоміших реалізацій бустингу – AdaBoost, Gradient Boosting та сучасні фреймворки XGBoost, LightGBM і CatBoost, які забезпечують ефективну обробку великих структурованих наборів, зокрема даних фінансового або маркетингового характеру [13].

Стекинг або стекування передбачає навчання декількох базових моделей різних типів (наприклад, дерева рішень, SVM, логістична регресія) і подальше поєднання їхніх результатів за допомогою метамоделі (meta-learner). Такий підхід часто використовується для створення комплексних систем прогнозування, де сильні сторони одних алгоритмів компенсують недоліки інших. У структурованих наборах даних стекування дозволяє поєднувати як прості статистичні, так і глибокі моделі, що забезпечує високу точність та гнучкість аналізу.

Суттєвою перевагою ансамблевих методів є їх здатність до автоматичної корекції помилок окремих моделей і стійкість до мультиколінеарності ознак. Крім того, ансамблі легко масштабуються і можуть бути реалізовані у вигляді розподілених систем. Недоліками вважаються більша обчислювальна складність, зниження інтерпретованості та потреба в додатковій оптимізації гіперпараметрів.

Застосування ансамблевих методів до структурованих даних потребує попередньої підготовки та обробки вхідної інформації, що включає перевірку якості, усунення пропусків та трансформацію ознак.

Підготовка структурованих даних передбачає послідовність процедур, спрямованих на приведення інформації до форми, придатної для моделювання [8]. На початковому етапі здійснюється перевірка якості даних, виявлення дублювання записів, некоректних типів або форматів полів, невідповідностей у розмірності та пропущених значень. Виконується контроль допустимих діапазонів, перевірка

форматів дат, узгодження назв полів та видалення записів, що не підлягають корекції. Результатом цього етапу є таблиця з узгодженою структурою, яка може бути використана для подальніх перетворень.

Обробка пропусків у даних виконується після попереднього аналізу їх природи. Пропуски можуть виникати через відсутність відповіді, помилки під час введення або об'єднання джерел. У разі незначної кількості пропусків допустиме видалення відповідних записів. Якщо частка відсутніх значень більша, застосовується імпутація константою, медіаною, середнім або модою залежно від типу ознаки [9]. Для багатовимірних таблиць можливе використання методів, що враховують взаємозв'язки між змінними, наприклад, імпутації за допомогою k найближчих сусідів або багатовимірної регресії. Вибір методу обумовлюється розподілом пропусків та властивостями ознак.

Масштабування числових ознак здійснюється для приведення їх до єдиного порядку величин. Це необхідно для моделей, чутливих до масштабу вхідних даних, таких як метод опорних векторів, k найближчих сусідів або регресійні моделі з регуляризацією. Найпоширенішими методами є нормалізація за формулою (1.1)

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}, \tag{1.1}$$

що перетворює значення у діапазон [0,1], та стандартизація за формулою (1.2)

$$x' = \frac{x - \mu}{\sigma}, \tag{1.2}$$

де μ – середнє значення,

σ – стандартне відхилення.

У деяких випадках застосовується логарифмічне перетворення (1.3)

$$x' = log(x + c), \tag{1.3}$$

де c – мала константа для уникнення нульових значень.

Параметри нормалізації та стандартизації мають визначатися лише на навчальній вибірці і зберігатися для використання на тестових даних, щоб запобігти витоку інформації.

Категоріальні ознаки підлягають кодуванню у числову форму [6]. Для змінних з невеликою кількістю унікальних значень застосовується one-hot кодування, яке створює окрему бінарну ознаку для кожної категорії. Для впорядкованих ознак можливе порядкове кодування, де значення перетворюються у числову послідовність. При великій кількості категорій доцільно використовувати таргет-енкодинг, що ґрунтується на статистичних характеристиках цільової змінної. У цьому разі необхідно дотримуватись ізольованого розрахунку статистик на навчальній частині даних для уникнення витоку.

Викиди визначаються як значення, що суттєво відрізняються від загального розподілу [11]. Для їх виявлення застосовується правило інтерквартильного діапазону, аналіз за z-оцінкою або методи, засновані на моделях, такі як Isolation Forest. Після виявлення викиди можуть бути видалені, обмежені або трансформовані. Остаточне рішення приймається на основі контексту предметної області та статистичних характеристик вибірки.

Інженерія ознак передбачає створення нових змінних на основі існуючих, обчислення агрегатів, взаємодій або часових відхилень. Відбір ознак виконується для зменшення розмірності та усунення мультиколінеарності. Використовуються методи фільтрації на основі кореляційних коефіцієнтів, покрокові алгоритми виключення ознак, а також підходи, що інтегровані у моделі, наприклад регуляризація L1 або оцінка важливості ознак у деревоподібних моделях. Поступове поєднання декількох підходів дозволяє сформувати стабільний набір вхідних параметрів.

У задачах класифікації з нерівномірним розподілом класів застосовується ресемплінг. Для збільшення представленості меншості використовується метод

SMOTE, який генерує синтетичні приклади. Альтернативою є зменшення обсягу більшості або введення вагових коефіцієнтів для кожного класу. Такі операції мають виконуватися лише на навчальній підмножині.

Під час розділення даних на навчальну, валідаційну та тестову частини застосовується стратегія крос-валідації. Для звичайних даних використовується k-fold розбиття, для часових рядів – послідовне розгортання з фіксованим вікном. Оцінка моделі здійснюється на валідаційній підмножині, остаточне тестування – на відкладених даних.

Для забезпечення узгодженості процесу підготовки застосовуються пайплайни. Кожен крок обробки, включаючи імпутацію, кодування, масштабування та навчання моделі, визначається у вигляді окремого модуля. Це гарантує стабільність результатів і відсутність втручання тестових даних у процес навчання.

Практична реалізація підготовки здійснюється із забезпеченням відтворюваності. Використовується версіювання наборів даних, збереження параметрів трансформацій, журналювання експериментів і контроль змін. При великих обсягах таблиць або складних перетвореннях доцільно застосовувати системи паралельної обробки, зокрема Dask або Spark. Алгоритмічні параметри підготовки повинні залишатися узгодженими з обраними методами моделювання, що забезпечує коректність оцінювання моделей.

1.2. Методологічний аналіз та архітектури машинного навчання для неструктурованих текстових даних

Неструктуровані текстові дані визначаються як інформація, що не має фіксованої схеми представлення, у якій зміст не організований у вигляді таблиць або визначених полів із заздалегідь встановленими типами та форматами. На відміну від структурованих даних, які зберігаються у реляційних базах даних та характеризуються чіткою організацією за стовпцями і рядками, неструктуровані текстові масиви не підпорядковуються єдиним форматам і можуть містити

контекстні, лексичні та синтаксичні залежності, які виявляються лише під час аналізу змісту. До таких даних належать текстові документи різних жанрів, електронні листи, повідомлення користувачів у соціальних мережах, коментарі під публікаціями, відгуки споживачів, новинні статті, транскрипти розмов, технічна документація та інші форми природної мови, що генеруються людьми у процесі комунікації [14].

Основною характеристикою текстових даних є їхня семантична багатовимірність, що виражається у складності встановлення однозначної відповідності між текстовими елементами та їх значеннями. Один і той самий термін може мати різні значення залежно від контексту, синтаксичного оточення та предметної області, що ускладнює автоматизоване інтерпретування змісту без застосування додаткових механізмів розуміння. Текст також має властивість полісемії, коли слово виражає декілька понять або концептів, і синонімії, коли різні слова або словосполучення описують однакові поняття чи об'єкти реального світу. Наявність омонімів, багатозначних конструкцій, ідіоматичних виразів та імплікатур ускладнює автоматичну інтерпретацію. Це зумовлює потребу у попередній мовній обробці, спрямованій на зменшення лексичної неоднорідності та приведення текстових одиниць до форми, придатної для подальшого аналізу засобами обчислювальних методів.

Додатковим аспектом, що впливає на складність обробки, є морфологічна варіативність природної мови. У флективних мовах, зокрема в українській, слова змінюються за відмінками, родами, числами та особами, що призводить до значного збільшення кількості унікальних словоформ навіть за обмеженого словникового складу. Відповідно, без застосування методів нормалізації та приведення до канонічних форм обсяг словника зростає, а ефективність статистичних методів знижується через розпорошення частотних характеристик. У той же час агглютинативні мови, такі як турецька або фінська, характеризуються побудовою складних словоформ шляхом приєднання афіксів, що також вимагає спеціалізованого підходу до токенізації та морфологічного аналізу.

Типовими джерелами неструктурованих даних є цифрові платформи, де користувачі створюють контент у довільній формі без обмежень на структуру та формат повідомлень. У бізнес-аналітиці такі тексти можуть відображати емоційний стан клієнтів, їх задоволеність послугами або ставлення до продуктів, що дозволяє формувати профілі споживчої поведінки та виявляти проблемні аспекти взаємодії з брендом.

Обробка текстових даних потребує перетворення їх у числову форму, придатну для застосування алгоритмів машинного навчання, оскільки більшість моделей оперують векторними представленнями та матричними структурами. З цією метою реалізується послідовність етапів попередньої підготовки, що включає очищення, токенізацію, лематизацію, стемінг, нормалізацію та векторизацію. Кожен із цих етапів виконує специфічну функцію у процесі трансформації вихідного тексту до формату, який може бути інтерпретований алгоритмічно. Загальну схему процесу наведено на рисунку 1.5, де відображено послідовність операцій від вихідного текстового масиву до фінального числового представлення.

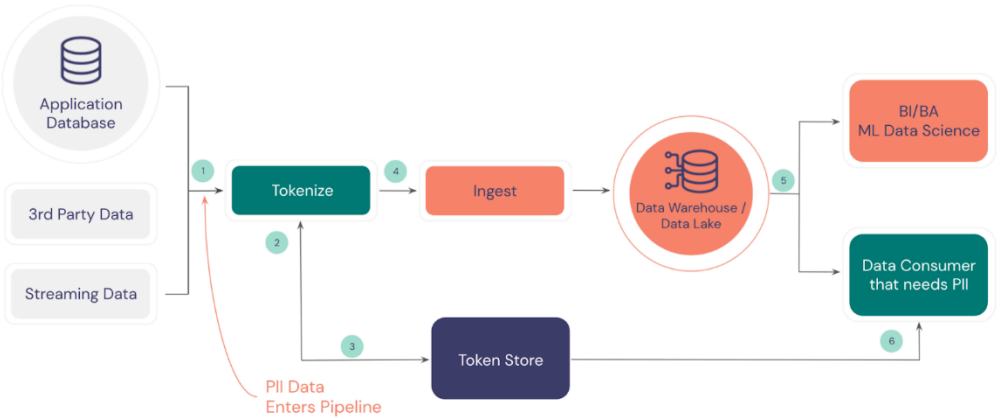


Рис. 1.5 Схема процесу підготовки текстових даних до аналізу

На етапі очищення видаляються знаки пунктуації, спеціальні символи, HTML-теги, URL-адреси, цифрові послідовності, що не несуть смислового навантаження, а також стоп-слова – службові частини мови, які зустрічаються з високою частотою, але не впливають на семантику висловлювання. До стоп-слів зазвичай належать прийменники, сполучники, частки та займенники, перелік яких

формується залежно від мови корпусу та специфіки задачі. Видалення дублікатів та ідентичних фрагментів запобігає перенавчанню моделей на повторюваних зразках. У деяких випадках, наприклад, при аналізі тональності або виявленні спаму, знаки пунктуації та емотикони можуть зберігатися, оскільки вони несуть додаткову інформацію про емоційний стан автора.

Процес підготовки даних передбачає декілька джерел вхідної інформації, що можуть включати прикладні бази даних, дані від третіх сторін та потокові дані. Як видно з рисунка 1.5, вхідні дані з різних джерел надходять до етапу токенізації, де відбувається їх первинна сегментація. Токенізація передбачає розбиття тексту на окремі елементи – токени, які можуть бути словами, фразами, символами або підсловами залежно від обраної стратегії сегментації. Найпростішим методом є розбиття за пробілами та розділовими знаками, проте такий підхід не враховує складні випадки, такі як абревіатури, дати, числа з роздільниками, складені слова та спеціальні конструкції. Для мов зі складною морфологією застосовуються спеціалізовані токенізатори, які враховують морфемні межі [15].

Після токенізації дані можуть надходити до сховища токенів (Token Store), де зберігаються результати попередньої обробки для подальшого використання, або безпосередньо передаватися на етап завантаження (Ingest). Сховище токенів виконує функцію проміжного буфера, що дозволяє організувати пакетну обробку та забезпечити узгодженість даних при їх подальшому перетворенні. Паралельно з основним потоком обробки відбувається процес інтеграції даних, що називається PII Data Enters Pipeline, який забезпечує включення персональної ідентифікаційної інформації у загальний конвеєр обробки з дотриманням вимог конфіденційності.

На етапі завантаження токенізовані дані передаються до сховища або озера даних (Data Warehouse / Data Lake), де вони зберігаються у формі, придатній для подальшого аналізу. Це сховище є центральним компонентом архітектури, оскільки забезпечує консолідацію інформації з різних джерел та її доступність для споживачів даних. Зі сховища дані можуть надходити до систем бізнес-аналітики та наукового аналізу даних (BI/BA ML Data Science), де відбувається побудова аналітичних моделей, візуалізація та інтерпретація результатів. Окремий напрям

використання даних передбачає їх передачу споживачам, які потребують персональної ідентифікаційної інформації (Data Consumer that needs PII), що може включати системи підтримки клієнтів, персоналізовані рекомендаційні механізми або інструменти аудиту.

У контексті нейромережевих моделей розповсюдження отримали методи підслівної токенізації, такі як Byte Pair Encoding (BPE), WordPiece та SentencePiece, які дозволяють працювати з обмеженим словником та ефективно обробляти рідкісні або невідомі слова шляхом їх розбиття на часто вживані фрагменти [14]. Ці підходи базуються на ітеративному об'єднанні найчастіших послідовностей символів або токенів, що призводить до формування словника оптимального розміру, здатного покривати більшість слів у корпусі без необхідності зберігання всіх можливих словоформ.

Лематизація та стемінг застосовуються для приведення слів до базової форми, що дозволяє зменшити варіативність написання та об'єднати різні словоформи під однією лексемою. Стемінг полягає у відсіканні афіксів за заздалегідь визначеними правилами, внаслідок чого утворюється основа слова, яка не обов'язково є коректною лексичною одиницею. Алгоритми стемінгу, такі як алгоритм Портера або Сноубола, є швидкими та не потребують лінгвістичних ресурсів, проте можуть призводити до надмірного або недостатнього відсікання. Лематизація, на відміну від стемінгу, використовує морфологічний аналіз та словники для приведення слова до його словникової форми – леми. Цей процес є більш точним, але вимагає наявності лінгвістичних ресурсів та складніших обчислень. Вибір між стемінгом і лематизацією залежить від специфіки задачі: для пошукових систем часто достатньо стемінгу, тоді як для семантичного аналізу переважною є лематизація.

Нормалізація здійснюється шляхом приведення тексту до єдиного регістру, зазвичай до нижнього, усунення надлишкових пробілів, символів нового рядка та табуляції, а також стандартизації символів, наприклад, заміни різних типів лапок або тире на єдиний варіант. У деяких випадках нормалізація включає транслітерацію або транскрипцію іншомовних слів, приведення чисел до

стандартного формату та заміну скорочень їх повними формами. Ці операції забезпечують однорідність вхідних даних та зменшують шум, що може негативно впливати на якість подальшого аналізу.

Після нормалізації текст перетворюється у числові представлення, що дозволяють застосовувати математичні операції та будувати моделі машинного навчання. Початково для цього використовувалися методи частотного аналізу, зокрема модель «мішок слів» (Bag of Words, BoW) та метод TF-IDF (Term Frequency – Inverse Document Frequency), що відображають частоту появи термінів у корпусі та їх відносну специфічність для окремих документів. Модель BoW представляє текст як вектор, де кожна компонента відповідає кількості входжень певного слова у документі, незалежно від порядку слів. Вектор виражається формулою 1.4:

$$v = (c_1, c_2, \ldots c_n), \tag{1.4}$$

де c_i позначає частоту входження і-го слова зі словника у документі.

TF-IDF доповнює цей підхід зваженням термінів, яке знижує вплив частих слів, що зустрічаються у багатьох документах, та підвищує вагу слів, що є специфічними для конкретного документа. Вага терміна обчислюється як добуток term frequency (TF) та inverse document frequency (IDF), де TF відображає частоту терміна у документі, а IDF визначається як логарифм відношення загальної кількості документів до кількості документів, що містять цей термін. Попри простоту та інтерпретовність, ці методи не враховують семантичних зв'язків між словами та контекстних залежностей, що обмежує їх застосування у задачах, де розуміння змісту має значення для досягнення прийнятної точності класифікації або кластеризації.

Для обробки текстових даних після їх векторизації застосовуються різноманітні методи машинного навчання, що класифікуються за типом навчання та способом моделювання мовних залежностей.

Методи машинного навчання, що застосовуються для аналізу текстових даних, класифікуються за типом навчання, способом представлення інформації та підходом до моделювання мовних залежностей. Основні напрями охоплюють навчання з учителем, без учителя, напівконтрольоване та підкріплене навчання, аналогічно до методів для структурованих даних, проте з істотними відмінностями у способах представлення ознак та архітектурах моделей [15].

Основна відмінність між методами для структурованих та неструктурованих текстових даних полягає у способі формування простору ознак. Якщо у табличних даних ознаки є заздалегідь визначеними числовими або категоріальними змінними з фіксованою кількістю стовпців, то у текстових даних простір ознак формується динамічно на основі словника корпусу, який може містити десятки або сотні тисяч унікальних термінів. Це призводить до високої розмірності та розрідженості векторних представлень, що вимагає застосування спеціалізованих методів зниження розмірності або використання щільних векторних подань.

Для текстових задач навчання з учителем застосовується у випадках класифікації документів за категоріями, аналізу тональності висловлювань, виявлення іменованих сутностей або визначення спаму. На відміну від табличних даних, де алгоритми безпосередньо працюють з числовими значеннями, текстові класифікатори потребують попереднього перетворення тексту у числові представлення через векторизацію. Класичні алгоритми, такі як логістична регресія, методи опорних векторів та наївний байєсівський класифікатор, можуть застосовуватися до текстових даних після їх перетворення у формат Bag of Words або TF-IDF. Проте для складніших задач, де необхідно враховувати послідовність слів та контекстні залежності, використовуються рекурентні нейронні мережі (RNN, LSTM, GRU) або трансформерні архітектури, що не мають прямих аналогів у методах для табличних даних.

Навчання без учителя для текстових даних також відрізняється від відповідних методів для структурованих даних. Замість кластеризації записів на основі евклідової відстані у багатовимірному просторі ознак, текстові дані кластеризуються на основі семантичної подібності документів або термінів, що

вимагає застосування косинусної відстані або інших метрик, адаптованих до розріджених векторів. Тематичне моделювання, таке як Latent Dirichlet Allocation (LDA) та Non-negative Matrix Factorization (NMF), є специфічним для текстових даних та не має прямих аналогів у роботі зі структурованими таблицями, оскільки воно виявляє приховані теми як розподіли над словами, а не як кластери точок у просторі ознак.

Напівконтрольовані методи набувають особливої актуальності у задачах обробки природної мови, оскільки розмітка текстових корпусів є трудомісткою і потребує залучення експертів для анотування великих обсягів даних. Використання великих нерозмічених корпусів дозволяє моделям навчатися на статистичних закономірностях мови, що є основою для попереднього навчання (pre-training) великих мовних моделей. Підкріплене навчання у текстових задачах застосовується для оптимізації діалогових систем, машинного перекладу та генерації тексту, де модель отримує винагороду за виконання певних дій, що наближують її до бажаного результату, наприклад, за формування відповіді, що відповідає очікуванням користувача.

Класифікація методів також враховує способи представлення текстових даних, що визначають характер ознак, які використовуються моделлю. Частотно-орієнтовані моделі, такі як Bag of Words та TF-IDF, ґрунтуються на статистичних характеристиках частоти термінів і не враховують порядку слів або їх семантичних зв'язків, що обмежує їх застосування у задачах, де контекст відіграє визначальну роль. Контекстно-залежні підходи передбачають побудову векторних представлень на основі моделей Word2Vec, GloVe та FastText, де слова описуються через їх семантичну подібність у контексті, що дозволяє виявляти аналогії та семантичні відношення між термінами [14]. Розвиток цих концепцій сприяв формуванню трансформерних архітектур, у яких векторне представлення залежить від усього контексту речення або документа, а не від фіксованого оточення фіксованого розміру. Трансформери використовують механізм уваги для динамічного визначення релевантності кожного токена для формування

представлення інших токенів, що дозволяє моделювати складні синтаксичні та семантичні залежності на довгих дистанціях.

Під час побудови моделей для текстових задач особливу роль відіграють методи штучного розширення корпусів (data augmentation), що підвищують стійкість і точність моделей шляхом створення додаткових навчальних зразків на основі існуючих даних. Розширення даних дозволяє зменшити ефект перенавчання, збалансувати розподіл класів та покращити узагальнювальну здатність моделі, особливо за умови обмеженого обсягу навчального корпусу. Як видно з рисунка 1.6, техніки розширення текстових даних можна розділити на дві основні категорії: прості методи аугментації (Easy Data Augmentation) та складніші підходи, що використовують генеративні моделі або зворотний переклад (Backtranslation).

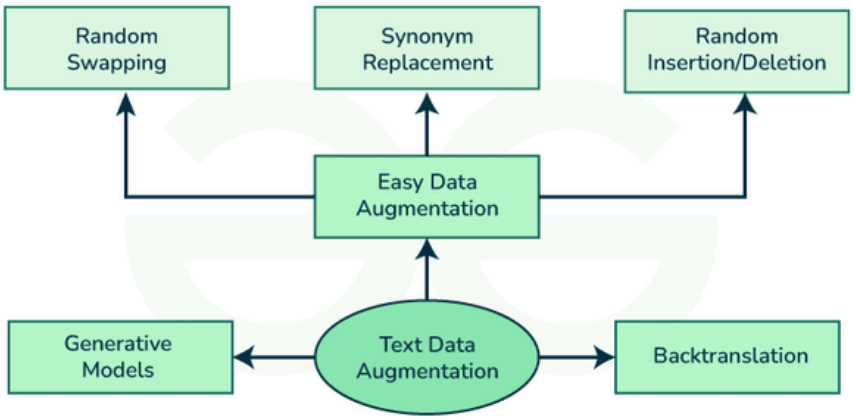


Рис. 1.6 Техніки розширення текстових даних у задачах NLP

До простих методів розширення належать випадкова перестановка слів (Random Swapping), заміна слів на синоніми (Synonym Replacement) та випадкове вставлення або видалення термінів (Random Insertion/Deletion). Випадкова перестановка передбачає зміну порядку слів у реченні без зміни їх складу, що дозволяє створити варіації тексту зі збереженням загального змісту, хоча може призвести до порушення синтаксичної структури. Заміна на синоніми полягає у заміні окремих слів на їх синонімічні еквіваленти, що зберігає семантику висловлювання, але вносить лексичну варіативність. Випадкове вставлення додає

до речення нові слова, відібрані зі словника або на основі контекстної подібності, тоді як випадкове видалення усуває частину слів, зберігаючи при цьому основний зміст.

Складніші методи включають використання генеративних моделей (Generative Models) та зворотний переклад. Генеративні моделі, такі як варіаційні автокодувальники (VAE) або генеративні змагальні мережі (GAN), здатні створювати нові текстові зразки, що мають статистичні властивості, подібні до навчального корпусу. Зворотний переклад передбачає переклад тексту на іншу мову з наступним перекладом назад на вихідну мову, що призводить до перефразування і створення семантично еквівалентного, але лексично відмінного варіанта. Цей метод широко застосовується для збільшення обсягу навчальних даних у задачах машинного перекладу та класифікації тексту.

Застосування різних підходів до навчання та розширення корпусів дозволяє підвищити якість моделювання, забезпечити стабільність результатів на тестових вибірках та зменшити вплив нерівномірності розподілу текстових зразків у навчальних наборах, що особливо актуально для задач із незбалансованими класами. Сукупність методів машинного навчання для текстових даних формує основу побудови інтелектуальних систем, здатних здійснювати семантичний аналіз, узагальнення інформації, автоматичне анотування документів та контекстну інтерпретацію природної мови у різних прикладних областях.

Для моделювання складних залежностей у текстових послідовностях застосовуються архітектури глибокого навчання, що автоматично формують ознаки без необхідності їх ручного проєктування.

Глибоке навчання використовується для моделювання складних залежностей у текстових даних, де значення кожного елемента визначається контекстом у межах послідовності. Такі архітектури автоматично формують ознаки, усуваючи потребу у попередньому проектуванні характеристик вручну. Основні підходи представлені рекурентними нейронними мережами, згортковими нейронними мережами та трансформерними моделями, що відрізняються способом обробки текстових послідовностей і побудови контекстних зв’язків.

Рекурентні нейронні мережі (RNN) реалізують послідовну обробку вхідних токенів, передаючи стан між часовими кроками. Це дає змогу моделі враховувати попередній контекст при формуванні представлення поточного слова. Модифікації архітектури, такі як Long Short-Term Memory (LSTM) та Gated Recurrent Unit (GRU), усувають обмеження, пов’язані зі зниканням або вибухом градієнтів. RNN застосовуються для задач генерації тексту, автоматичного перекладу та моделювання послідовностей. Загальний принцип їх роботи наведено на рисунку 1.7, де зображено схему передачі стану між вузлами послідовності.

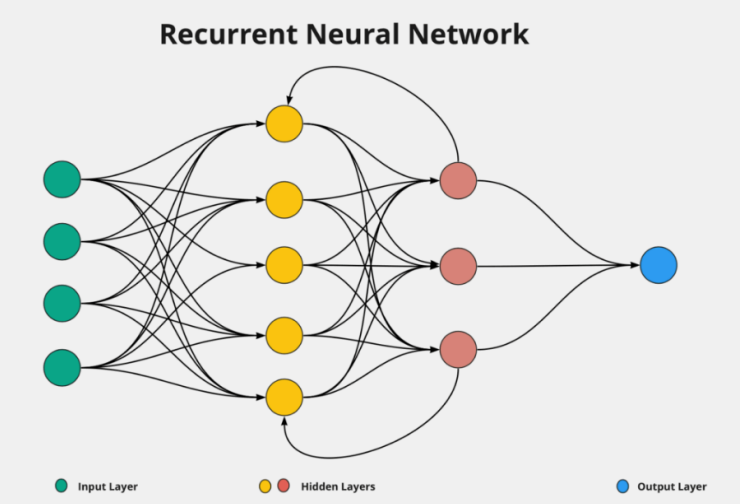


Рис. 1.7 Архітектура рекурентної нейронної мережі

Згорткові нейронні мережі (CNN) використовуються для вилучення локальних залежностей між словами в межах фіксованих вікон контексту. Застосування згорткових фільтрів дає змогу виділяти характерні патерни тексту, що можуть відповідати граматичним або семантичним зв’язкам. CNN використовуються для класифікації коротких повідомлень, аналізу емоційних відтінків та обробки відгуків [16]. Принцип побудови моделі показано на рисунку 1.8, де відображено етапи згортки, підвибірки (pooling) та формування узагальненого представлення послідовності.

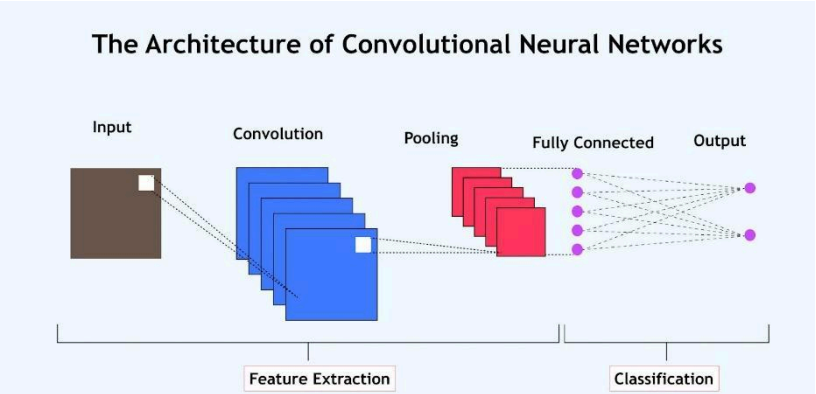


Рис. 1.8 Архітектура згорткової нейронної мережі для текстового аналізу

Трансформерні архітектури становлять окремий напрям розвитку моделей глибокого навчання для обробки природної мови. Вони ґрунтуються на механізмі самоуваги (self-attention), який дає змогу враховувати залежності між усіма елементами послідовності одночасно, усуваючи необхідність рекурентних зв'язків. Основними компонентами є багатоголовий шар уваги (multi-head attention), нормалізаційні блоки та позиційне кодування, що забезпечують формування контекстно-залежних векторних представлень. Ця архітектура дозволяє здійснювати паралельну обробку даних, що підвищує швидкість і стабільність навчання. Її структурна схема подана на рисунку 1.9, де наведено взаємозв'язки між енкодерними та декодерними блоками.

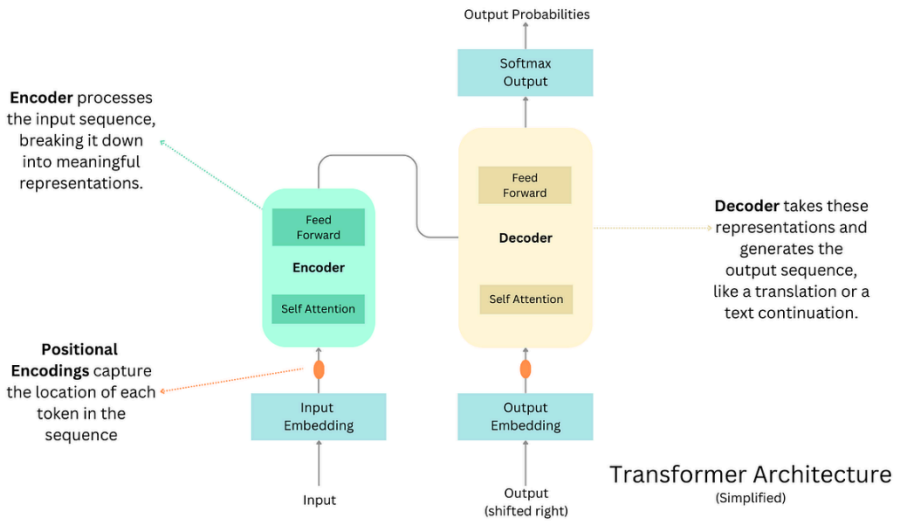


Рис. 1.9 Архітектура трансформерної моделі

Модель BERT (Bidirectional Encoder Representations from Transformers) є прикладом трансформерного підходу, що використовує двонапрямне кодування контексту [17]. Вона навчається на завданнях відновлення прихованих токенів і прогнозування послідовностей речень, що дозволяє формувати узагальнені мовні представлення. Такі попередньо натреновані моделі адаптуються до різних задач через донавчання на спеціалізованих корпусах [18].

Застосування глибоких архітектур забезпечує можливість побудови моделей, здатних аналізувати як синтаксичні, так і семантичні властивості мовних конструкцій. Вони використовуються у системах машинного перекладу, автоматичного реферування, інформаційного пошуку, класифікації текстів та генерації природномовних відповідей.

1.3. Обґрунтування авторської позиції та вибір методик для інтегрованого аналізу

Пропонована система прийняття рішень щодо видачі короткострокових кредитів для мікрофінансової організації базується на інтегрованому підході, що поєднує аналіз фінансових ознак клієнта, представлених у вигляді структурованих табличних даних, та оцінку його психо-емоційного стану за відкритими текстовими відповідями, які є неструктурованими даними [2]. Система складається з двох взаємопов'язаних компонентів, кожен із яких застосовує специфічні методи машинного навчання, що відповідають характеру оброблюваної інформації. Поєднання цих компонентів дозволяє формувати комплексну оцінку ризику дефолту, враховуючи не лише об'єктивні фінансові показники, але й суб'єктивні психологічні характеристики позичальника, що може впливати на його здатність і готовність до повернення кредиту.

Перший компонент системи здійснює аналіз структурованих фінансових даних, що включають вік позичальника, місячний дохід, суму запитуваного кредиту, мету кредитування та кредитний рейтинг [8]. Ці параметри формують базис для традиційного кредитного скорингу, який широко застосовується у

фінансових установах для оцінки кредитоспроможності клієнтів. Для цього компонента обрано два ансамблеві методи машинного навчання: Random Forest та XGBoost, кожен із яких має підтверджену ефективність у задачах кредитного скорингу та прогнозування дефолту.

Random Forest демонструє високу точність у задачах класифікації кредитних ризиків, досягаючи точності понад 81% на реальних кредитних наборах даних, що робить його одним із найбільш результативних методів для аналізу табличних даних у фінансовій сфері [19]. Цей метод заснований на побудові ансамблю некорельованих дерев рішень, кожне з яких оцінює вхідні дані та класифікує об'єкт незалежно, після чого слабкі класифікатори об'єднуються у більш надійний результат шляхом мажоритарного голосування. Перевагою Random Forest є стійкість до перенавчання завдяки використанню техніки bagging, здатність працювати з високовимірними даними без значної попередньої обробки та можливість оцінювати важливість ознак, що дозволяє ідентифікувати найбільш впливові фінансові параметри для прогнозування дефолту.

XGBoost, як представник методів градієнтного бустингу, виявляє вищу якість прогнозування у порівнянні з іншими алгоритмами класифікації, досягаючи показників точності та стабільності на рівні 93-95% у задачах передбачення невиплати кредитів. Модель XGBoost ефективно витягує змінні, що мають значний вплив на ризик дефолту, та забезпечує кращу калібровку ймовірностей дефолту порівняно з традиційними статистичними методами, такими як логістична регресія [9]. Алгоритм використовує послідовне навчання дерев рішень, де кожне наступне дерево коригує помилки попередніх, що призводить до формування моделі з високою передбачувальною здатністю. Додатковою перевагою XGBoost є вбудовані механізми регуляризації, що запобігають перенавчанню, та можливість обробки пропущених значень без необхідності їх попереднього заповнення.

Вибір цих двох методів для першого компонента системи дозволяє провести порівняльний аналіз їх ефективності на конкретному наборі даних мікрофінансової організації та обрати модель з оптимальними характеристиками

для подальшої інтеграції з компонентом аналізу текстових даних. Порівняння Random Forest та XGBoost на однакових даних дає можливість виявити переваги кожного методу в контексті специфіки короткострокового кредитування, де структура залежностей між фінансовими параметрами може відрізнятися від традиційних банківських продуктів.

Другий компонент системи призначений для аналізу психо-емоційного стану позичальника на основі його відповідей на відкриті запитання щодо фінансового планування, поточної життєвої ситуації та мотивації до отримання кредиту. Оскільки мікрофінансові організації видають кредити на короткий термін, поточний емоційний стан клієнта, його психологічна стабільність та особистісні риси можуть впливати на ймовірність своєчасного повернення позики. Машинне навчання для розпізнавання людських емоцій за текстовими даними демонструє здатність виявляти емоційний стан та психологічні характеристики з високою точністю, що робить такий аналіз релевантним для оцінки кредитних ризиків [16].

Для обробки текстових відповідей українською мовою обрано багатомовну трансформерну модель XLM-RoBERTa, яка є розширенням архітектури RoBERTa для підтримки понад 100 мов, включаючи українську. Дослідження з тонального аналізу українськомовних текстів показують, що XLM-RoBERTa досягає найвищої точності серед багатомовних моделей, забезпечуючи точність на рівні 91,32% у задачах класифікації тональності відгуків українською мовою [17]. Модель базується на механізмі самоуваги (self-attention), що дозволяє враховувати залежності між усіма токенами в тексті одночасно, формуючи контекстно-залежні векторні представлення, які враховують семантичні та синтаксичні особливості висловлювань.

XLM-RoBERTa навчалася на величезному корпусі, що включає понад 2,5 терабайти текстових даних із 100 мов, що забезпечує її здатність до ефективного аналізу тональності та емоційних станів у багатомовному контексті без необхідності додаткового навчання для кожної окремої мови. Трансформерні моделі, такі як BERT та RoBERTa, широко застосовуються для передбачення

особистісних рис на основі текстових даних, включаючи аналіз відповідей на інтерв'ю, досягаючи високої точності у виявленні рис особистості за моделлю Big Five [18]. Методи пояснюваного передбачення особистості на основі відповідей на відкриті питання інтерв'ю дозволяють виділяти ключові характеристики, що впливають на поведінку індивіда у фінансових та інших контекстах. Це дає змогу не лише класифікувати емоційний стан, але й виявляти стійкі особистісні риси, що можуть корелювати з фінансовою дисципліною та готовністю виконувати зобов'язання.

Після обробки текстових відповідей за допомогою XLM-RoBERTa отримуються числові ознаки, що характеризують емоційний стан клієнта, наявність стресових факторів, рівень оптимізму або песимізму щодо фінансового майбутнього, а також деякі риси особистості, що можуть впливати на кредитну поведінку. Ці ознаки інтегруються з фінансовими параметрами, формуючи розширений набір даних для фінального прогнозування ймовірності дефолту. Інтеграція структурованих та неструктурованих даних здійснюється шляхом конкатенації векторів ознак, після чого комбінована модель будується на основі одного з обраних ансамблевих методів, що дозволяє використати переваги обох типів інформації у єдиному процесі класифікації.

Гіпотеза дослідження полягає в тому, що додавання емоційних та психологічних ознак, отриманих із текстових відповідей, до фінансових параметрів може підвищити точність прогнозування ймовірності повернення позики порівняно з базовою моделлю, що використовує лише фінансові дані. Це припущення базується на тому, що емоційний стан та особистісні характеристики можуть впливати на фінансову поведінку індивіда, його схильність до ризику, здатність планувати бюджет та дотримуватися взятих зобов'язань. У короткостроковому кредитуванні, де рішення про видачу позики приймається швидко і часто за обмеженою кількістю формальних показників, інформація про поточний психо-емоційний стан клієнта може надати додаткові дані для оцінки ризику.

Для перевірки цієї гіпотези пропонується провести порівняльний аналіз трьох типів моделей: базових моделей, що використовують лише фінансові ознаки (Random Forest та XGBoost окремо), моделей, що аналізують лише текстові дані (XLM-RoBERTa для класифікації емоційного стану), та інтегрованих моделей, що поєднують обидва типи ознак [10]. Оцінювання ефективності здійснюватиметься за стандартними метриками класифікації, такими як точність (accuracy), precision, recall, F1-score та площа під ROC-кривою (AUC), що дозволить визначити внесок емоційних характеристик у покращення якості прогнозування. Порівняння базових та інтегрованих моделей дасть змогу кількісно оцінити ефект від включення психо-емоційних параметрів у систему прийняття рішень.

На початковому етапі дослідження планується використання синтетичних даних, згенерованих на основі типових фінансових профілів клієнтів мікрофінансових організацій та відповідних текстових відповідей, що імітують різні емоційні стани та особистісні характеристики. Використання синтетичних даних дозволяє контролювати розподіл класів, варіювати параметри та створювати сценарії різного ступеня складності для тестування гіпотези без ризику розголошення персональних даних реальних клієнтів. Синтетичні дані формуються таким чином, щоб відображати реалістичні співвідношення між фінансовими параметрами та ймовірністю дефолту, а також включати різноманітні варіанти текстових відповідей із різним емоційним забарвленням. Після підтвердження працездатності підходу на синтетичних даних система може бути адаптована для роботи з реальними даними клієнтів за умови дотримання вимог конфіденційості та захисту персональної інформації.

Обрані методи та архітектура системи відповідають сучасним тенденціям у галузі машинного навчання для фінансового аналізу та обробки природної мови. Застосування методів машинного навчання для кредитного скорингу та прогнозування дефолту на основі поведінкових та транзакційних фінансових даних демонструє значне покращення точності порівняно з традиційними статистичними підходами. Аналіз неструктурованих даних за допомогою штучного інтелекту та застосування методів машинного навчання для розуміння

неструктурованої інформації у великих масивах даних відкриває нові можливості для вилучення цінних інсайтів із текстових джерел. Інтеграція структурованих та неструктурованих даних у єдиній системі прийняття рішень представляє перспективний напрям для покращення якості кредитного скорингу та зменшення фінансових ризиків мікрофінансових організацій [4].

Варто зазначити, що система не обмежується лише задачею кредитного скорингу для мікрофінансових організацій. Запропонований підхід інтеграції аналізу структурованих та неструктурованих даних може бути адаптований для інших застосувань, де прийняття рішень базується на комбінації об'єктивних числових показників та суб'єктивної інформації, вираженої у текстовій формі.

2 АНАЛІТИКО-ДОСЛІДНИЦЬКА АПРОБАЦІЯ ОБРАНИХ МЕТОДІВ МАШИННОГО НАВЧАННЯ

2.1. Формування бази дослідження та аналіз проблеми

Для реалізації системи прийняття рішень щодо видачі короткострокових кредитів на основі машинного навчання необхідно використовувати два типи даних: структуровані фінансові дані клієнтів та неструктуровані текстові відповіді для оцінки психо-емоційного стану. Вибір відповідних датасетів визначає можливості подальшого навчання моделей та їх здатність до узагальнення на реальних даних мікрофінансових організацій.

Для аналізу структурованих фінансових параметрів позичальників обрано датасет German Credit Data, який є стандартним benchmark-набором для задач кредитного скорингу. Датасет було створено професором Гансом Хофманном з Інституту статистики та економетрики Гамбурзького університету та надано у відкритий доступ через UCI Machine Learning Repository [20]. Цей набір даних містить інформацію про 1000 позичальників німецького банку з атрибутами, що характеризують їх фінансовий стан, кредитну історію та демографічні характеристики [21].

German Credit Data представлено у двох версіях: оригінальна версія містить категоріальні та символьні атрибути (файл german.data), тоді як числова версія (german.data-numeric) була підготовлена Стратклайдським університетом для алгоритмів, що потребують лише числових вхідних даних. У числовій версії категоріальні змінні перетворені на індикаторні змінні, а впорядковані категоріальні атрибути закодовані як цілі числа. Для цього дослідження використовується оригінальна версія датасету, оскільки сучасні алгоритми машинного навчання, зокрема Random Forest та XGBoost, здатні обробляти категоріальні змінні без попереднього перетворення.

Датасет складається з 20 атрибутів, серед яких 7 числових та 13 категоріальних. Цільова змінна представляє бінарну класифікацію: клас 1

відповідає «добрим» позичальникам, які повернули кредит, клас 2 – «поганим» позичальникам, які допустили дефолт. Розподіл класів у датасеті становить 700 «добрих» та 300 «поганих» позичальників, що відображає типове співвідношення для реальних кредитних портфелів, де більшість клієнтів виконують свої зобов'язання [21].

Атрибут 1 описує стан поточного розрахункового рахунку позичальника та приймає одне з чотирьох значень: від'ємний баланс (менше 0 німецьких марок), баланс від 0 до 200 марок, баланс понад 200 марок або гарантовані зарахування зарплати протягом щонайменше року, відсутність розрахункового рахунку. Цей параметр відображає поточну фінансову ліквідність клієнта та його здатність підтримувати позитивний баланс [20].

Атрибут 2 представляє тривалість кредиту у місяцях та є числовою змінною. Тривалість кредитування варіюється від кількох місяців до кількох років, що дозволяє аналізувати вплив терміну позики на ймовірність дефолту. Дослідження показують, що довші терміни кредитування можуть як збільшувати ризик через більшу тривалість експозиції, так і зменшувати його через нижчі щомісячні платежі [22].

Атрибут 3 характеризує кредитну історію позичальника та включає п'ять категорій: відсутність попередніх кредитів або всі кредити повернуті вчасно; всі кредити у цьому банку повернуті вчасно; існуючі кредити погашаються належним чином до теперішнього моменту; затримки у погашенні у минулому; критичний рахунок або наявність інших кредитів в інших банках. Кредитна історія є одним з найбільш прогностичних факторів для передбачення майбутньої платіжної поведінки, оскільки минула поведінка часто повторюється у майбутньому.

Атрибут 4 визначає мету кредиту та містить 10 можливих категорій: новий автомобіль, вживаний автомобіль, меблі або обладнання, радіо або телевізор, побутова техніка, ремонт, освіта, перенавчання, бізнес, інші цілі. Мета кредитування може впливати на ризик дефолту, оскільки різні типи витрат мають різну цінність для позичальника та різний потенціал для генерування доходу. Наприклад, кредит на освіту або бізнес може сприяти зростанню доходів у

майбутньому, тоді як кредит на споживчі товари не створює додаткових джерел погашення.

Атрибут 5 представляє суму кредиту у німецьких марках та є числовою змінною. Розмір кредиту варіюється від невеликих споживчих позик до більших сум для придбання нерухомості або розвитку бізнесу. Співвідношення між сумою кредиту та доходом позичальника часто використовується як показник кредитного навантаження.

Атрибут 6 описує стан ощадного рахунку або наявність облігацій у позичальника з чотирма категоріями за розміром заощаджень: менше 100 марок, від 100 до 500 марок, від 500 до 1000 марок, понад 1000 марок, невідомо або відсутній ощадний рахунок. Наявність заощаджень свідчить про фінансову дисципліну та створює резервний фонд для непередбачених витрат, що зменшує ймовірність дефолту.

Атрибут 7 характеризує тривалість поточної зайнятості позичальника з п'ятьма категоріями: безробітний, менше одного року, від одного до чотирьох років, від чотирьох до семи років, понад сім років. Стабільність зайнятості є індикатором передбачуваності майбутніх доходів, що впливає на здатність до регулярного обслуговування кредиту.

Атрибут 8 визначає розмір платежу за кредитом у відсотках від наявного доходу та є числовою змінною. Цей показник аналогічний коефіцієнту debt-to-income ratio, який широко використовується у кредитному аналізі для оцінки навантаження на бюджет позичальника.

Атрибут 9 містить інформацію про особистий статус та стать позичальника з п'ятьма можливими комбінаціями: чоловік розлучений або у розлученні, жінка розлучена, у розлученні або одружена, чоловік неодружений, чоловік одружений або вдівець, жінка неодружена. Хоча цей атрибут включає демографічну інформацію, його використання у моделях кредитного скорингу може піднімати питання справедливості та дискримінації.

Атрибут 10 описує наявність інших боржників або поручителів з трьома категоріями: немає, співзаявник, поручитель. Наявність поручителя знижує ризик

для кредитора, оскільки створює додаткове джерело погашення у разі дефолту основного позичальника.

Атрибут 11 представляє тривалість проживання за поточною адресою у роках та є числовою змінною. Стабільність місця проживання може розглядатися як непрямий показник загальної стабільності життєвої ситуації позичальника.

Атрибут 12 характеризує наявність майна у позичальника з чотирма категоріями: нерухомість, договір будівельного товариства або страхування життя, автомобіль або інше майно (не зазначене в атрибуті 6), невідомо або відсутнє майно. Наявність майна може служити забезпеченням кредиту або резервним активом для погашення у разі фінансових труднощів.

Атрибут 13 визначає вік позичальника у роках та є числовою змінною. Вік може мати нелінійний зв'язок з кредитним ризиком: молодші позичальники можуть мати менший досвід управління фінансами, тоді як старші позичальники можуть мати більш стабільний дохід але вищі медичні витрати.

Атрибут 14 описує наявність інших планів розстрочки з трьома категоріями: банк, магазини, немає. Множинні зобов'язання з розстрочки можуть свідчити про надмірне кредитне навантаження або про схильність до імпульсивних покупок.

Атрибут 15 характеризує житлові умови позичальника з трьома категоріями: оренда, власне житло, безкоштовне проживання. Власне житло зменшує щомісячні витрати та може розглядатися як актив, тоді як оренда створює додаткове фінансове навантаження.

Атрибут 16 визначає кількість існуючих кредитів у цьому банку та є числовою змінною. Наявність кількох кредитів може свідчити як про довіру банку до клієнта, так і про надмірну заборгованість.

Атрибут 17 описує тип зайнятості позичальника з чотирма категоріями: безробітний або некваліфікований працівник-нерезидент, некваліфікований працівник-резидент, кваліфікований працівник або службовець, керівник, самозайнятий або висококваліфікований працівник. Рівень кваліфікації корелює з рівнем доходу та стабільністю зайнятості.

Атрибут 18 представляє кількість осіб, які перебувають на утриманні позичальника, та є числовою змінною. Більша кількість утриманців збільшує витрати домогосподарства та може зменшувати здатність до обслуговування кредиту.

Атрибут 19 характеризує наявність телефону з двома категоріями: немає, так, зареєстрований на ім'я клієнта. У період створення датасету (1990-ті роки) наявність телефону розглядалася як показник соціального статусу та стабільності.

Атрибут 20 визначає, чи є позичальник іноземним працівником, з двома категоріями: так, ні. Цей атрибут відображає специфіку німецького ринку праці того періоду [20].

Датасет German Credit включає матрицю вартості помилок, яка відображає асиметричність наслідків помилкових рішень у кредитуванні. Вартість помилкового схвалення кредиту «поганому» позичальнику (false positive) встановлена як 5, тоді як вартість помилкового відхилення «доброго» позичальника (false negative) дорівнює 1. Така асиметрія відповідає реальній бізнес-логіці, де втрати від дефолту значно перевищують втрати від упущеної вигоди через відмову у кредиті.

Датасет German Credit широко використовується у дослідженнях машинного навчання для кредитного скорингу через його збалансованість, достатній обсяг спостережень та реалістичну структуру атрибутів. Аналіз цього датасету показує, що найбільш прогностичними атрибутами для передбачення дефолту є стан розрахункового рахунку, кредитна історія, тривалість кредиту та мета кредитування [21]. Ці змінні формують основу для побудови базової моделі кредитного скорингу, яка потім може бути розширена за рахунок додавання психо-емоційних характеристик з текстових даних.

Окрім структурованих фінансових даних, для повноцінної реалізації інтегрованої системи необхідний датасет неструктурованих текстових даних, який дозволить навчити модель розпізнавання емоційного стану позичальників. Для обробки неструктурованих текстових відповідей позичальників та визначення їх емоційного стану обрано датасет EmoBench-UA, який є першим україномовним

набором даних для задачі розпізнавання емоцій [23]. Датасет створено колективом дослідників на чолі з Дариною Дементьєвою у рамках конкурсу SemEval-2025 Task 11 та опубліковано на платформі Hugging Face у відкритому доступі [24].

EmoBench-UA містить україномовні текстові фрагменти, анотовані за сімома категоріями емоцій згідно з моделлю базових емоцій Пола Екмана: радість (Joy), гнів (Anger), страх (Fear), огида (Disgust), здивування (Surprise), смуток (Sadness) та нейтральний стан (None). Модель Екмана базується на гіпотезі про існування універсальних базових емоцій, які розпізнаються незалежно від культурного контексту, що робить цю класифікацію придатною для застосування у різних мовних середовищах [23].

Датасет зібрано з використанням краудсорсингової платформи Toloka.ai, яка забезпечує контроль якості анотування через механізми валідації та узгодженості між анотаторами. Кожен текстовий фрагмент оцінювався кількома незалежними анотаторами, що дозволило знизити суб'єктивність оцінок та підвищити надійність міток. Процес анотування передбачав не лише визначення домінуючої емоції у тексті, але й оцінку інтенсивності цієї емоції за шкалою від 0 до 1, де 0 означає відсутність емоції, а 1 – максимальну інтенсивність.

Датасет розділено на тренувальну, валідаційну та тестову вибірки, що дозволяє проводити навчання моделей з подальшою перевіркою їх узагальнювальної здатності на незалежних даних. Тренувальна вибірка містить найбільшу кількість прикладів і використовується для налаштування параметрів моделі. Валідаційна вибірка застосовується для підбору гіперпараметрів та запобігання перенавчанню. Тестова вибірка призначена для фінальної оцінки якості моделі на даних, які не використовувалися під час навчання.

Розподіл емоцій у датасеті відображає природну частотність емоційних станів у текстових повідомленнях. Нейтральні тексти складають значну частину датасету, що відповідає реальним комунікаційним ситуаціям, де багато повідомлень передають фактичну інформацію без виразного емоційного забарвлення. Серед емоційно забарвлених текстів найчастіше зустрічаються

радість та смуток, тоді як огида та здивування представлені меншою кількістю прикладів [24].

Використання EmoBench-UA для навчання моделей розпізнавання емоцій у контексті фінансового консультування має кілька переваг. По-перше, датасет створено для української мови, що дозволяє враховувати специфіку мовних конструкцій та культурного контексту, який може відрізнятися від інших мов. По-друге, анотування проведено професійними оцінювачами з контролем якості, що забезпечує високу надійність міток. По-третє, датасет включає оцінку інтенсивності емоцій, що дозволяє не лише визначити тип емоції, але й оцінити її силу, що може бути релевантним для оцінки рівня фінансового стресу або тривожності позичальника [25].

Для адаптації моделі розпізнавання емоцій до специфіки фінансових текстів може знадобитися додаткове донавчання (fine-tuning) на текстах, пов'язаних з фінансовими темами. Це пов'язано з тим, що EmoBench-UA містить загальні тексти з різних доменів, тоді як відповіді позичальників на запитання щодо фінансового планування мають специфічну лексику та структуру. Донавчання дозволяє моделі краще розпізнавати емоції у контексті фінансових обговорень, де, наприклад, тривожність може виражатися через згадування боргів, витрат та невизначеності майбутнього доходу.

Після визначення датасетів для структурованих та неструктурованих даних постає питання методології збору текстових відповідей від потенційних позичальників. Для отримання текстових даних від позичальників, які дозволять оцінити їх психо-емоційний стан та особистісні характеристики, пропонується використання структурованого опитувальника з відкритими запитаннями. Підхід з використанням відкритих запитань знаходить обґрунтування у практиці character-based lending, де інтенсивний процес інтерв'ювання дозволяє оцінити надійність позичальника та його готовність до повернення позики через аналіз якісних характеристик [26].

Психометричне тестування активно застосовується кредиторами для оцінки кредитоспроможності на основі психологічних та поведінкових характеристик,

таких як ставлення до ризику, фінансова дисципліна та імпульсивність [27]. У рамках традиційної системи оцінки кредитоспроможності «5 C's of Credit», фактор «Character» передбачає якісну оцінку чесності, репутації та зобов'язальності позичальника, що часто здійснюється через співбесіди та аналіз суб'єктивних характеристик [28].

У дослідженнях мікрофінансування широко застосовуються якісні методи збору даних, включаючи глибинні індивідуальні інтерв'ю з відкритими запитаннями типу «Яку роль мікрофінансування відіграло у вашому житті?» та «Як ви плануєте свій робочий тиждень?», що дозволяє отримати детальну інформацію про життєві обставини, мотивацію та фінансову поведінку позичальників [29]. Дослідження показують, що якісні характеристики позичальника, такі як особистий характер, соціальні фактори та управлінські здібності, впливають на довіру кредитного працівника та можуть передбачати ймовірність повернення кредиту [26].

Враховуючи специфіку короткострокового кредитування у мікрофінансових організаціях, де рішення приймаються швидко за обмеженою кількістю формальних показників, збір якісної інформації через відкриті запитання може надати додаткові дані для оцінки ризику. Опитувальник складається з чотирьох основних запитань, кожне з яких спрямоване на виявлення різних аспектів психо-емоційного стану та фінансової поведінки позичальника:

1. «Опишіть вашу поточну фінансову ситуацію». Це запитання дозволяє позичальнику у вільній формі розповісти про свій фінансовий стан, включаючи джерела доходу, основні витрати, наявні заощадження та борги. Аналіз відповіді дозволяє виявити рівень фінансової обізнаності, здатність структуровано мислити про свої фінанси та наявність фінансового планування. Емоційне забарвлення відповіді може свідчити про рівень тривожності або впевненості у фінансовому майбутньому.

2. «Що вас мотивує взяти цей кредит?». Відповідь на це запитання розкриває мету кредитування з точки зору позичальника, що може відрізнятися від формальної категорії мети у заяві на кредит. Розуміння справжньої мотивації

дозволяє оцінити, наскільки обґрунтованою є потреба у позиці та чи має позичальник чітке уявлення про використання коштів. Емоційний аналіз може виявити надію, відчай, амбіції або імпульсивність у прийнятті рішення про кредит.

3. «Як ви плануєте виплачувати кредит?». Це запитання оцінює наявність конкретного плану повернення позики, що є індикатором відповідального ставлення до боргових зобов'язань. Детальна відповідь з конкретними джерелами погашення свідчить про серйозне обдумування рішення, тоді як розпливчаста або надто оптимістична відповідь може вказувати на недооцінку ризиків або відсутність реалістичного плану. Аналіз тональності дозволяє виявити впевненість або сумніви позичальника у своїй здатності виконати зобов'язання.

4. «Що вас найбільше турбує у фінансовому плані?». Відповідь на це запитання виявляє основні джерела фінансового стресу та тривожності позичальника. Визнання конкретних турбот свідчить про самоусвідомлення та готовність зіткнутися з реальністю. Характер турбот може вказувати на потенційні ризики для повернення кредиту, наприклад, нестабільність доходу, медичні витрати або інші непередбачені обставини. Емоційна інтенсивність відповіді дозволяє оцінити рівень фінансової тривожності, яка може впливати на платіжну поведінку.

Використання відкритих запитань замість структурованих психометричних тестів має переваги у контексті мікрофінансування. Відкриті запитання займають менше часу у порівнянні з довгими опитувальниками типу IPIP-NEO (120-300 запитань), що є критичним для процесу швидкого прийняття рішень у МФО. Водночас відкриті запитання дозволяють позичальнику висловитися у природній манері, що може бути менш стресовим та більш інформативним, ніж відповіді на закриті запитання зі стандартизованими варіантами. Зібрані текстові відповіді формують базу неструктурованих даних, яка у поєднанні зі структурованими фінансовими параметрами German Credit Data створює комплексний датасет для подальшого порівняльного аналізу методів машинного навчання.

2.2. Порівняльний аналіз ефективності методів машинного навчання для роботи зі структурованими даними

Для визначення оптимального методу аналізу структурованих фінансових даних у контексті кредитного скорингу проведено порівняльний аналіз двох ансамблевих алгоритмів машинного навчання: Random Forest та XGBoost. Вибір цих методів обумовлений їх результативністю у задачах бінарної класифікації на табличних даних, здатністю обробляти як числові, так і категоріальні ознаки, а також стійкістю до перенавчання.

Random Forest належить до сімейства методів ансамблевого навчання, що базуються на принципі bagging (bootstrap aggregating). Архітектура Random Forest передбачає паралельне створення множини дерев рішень, кожне з яких навчається на випадковій підвибірці даних, отриманій методом bootstrap-вибірки з поверненням. Під час побудови кожного дерева на кожному вузлі розгалуження розглядається лише випадкова підмножина ознак, що забезпечує декореляцію дерев та підвищує узагальнювальну здатність моделі. Фінальний прогноз формується шляхом мажоритарного голосування: для кожного об'єкта кожне дерево незалежно робить прогноз, після чого обирається клас, за який проголосувала більшість дерев [13].

Основною перевагою Random Forest є зниження дисперсії моделі порівняно з одиничним деревом рішень. Це досягається завдяки агрегації прогнозів множини слабких класифікаторів, кожен з яких може мати високу дисперсію, але в сукупності вони формують стабільну модель з низькою схильністю до перенавчання. Метод дозволяє оцінювати важливість ознак через аналіз того, наскільки кожна ознака в середньому зменшує неоднорідність у вузлах дерев. Додатковою перевагою є можливість паралелізації обчислень, оскільки дерева будуються незалежно одне від одного.

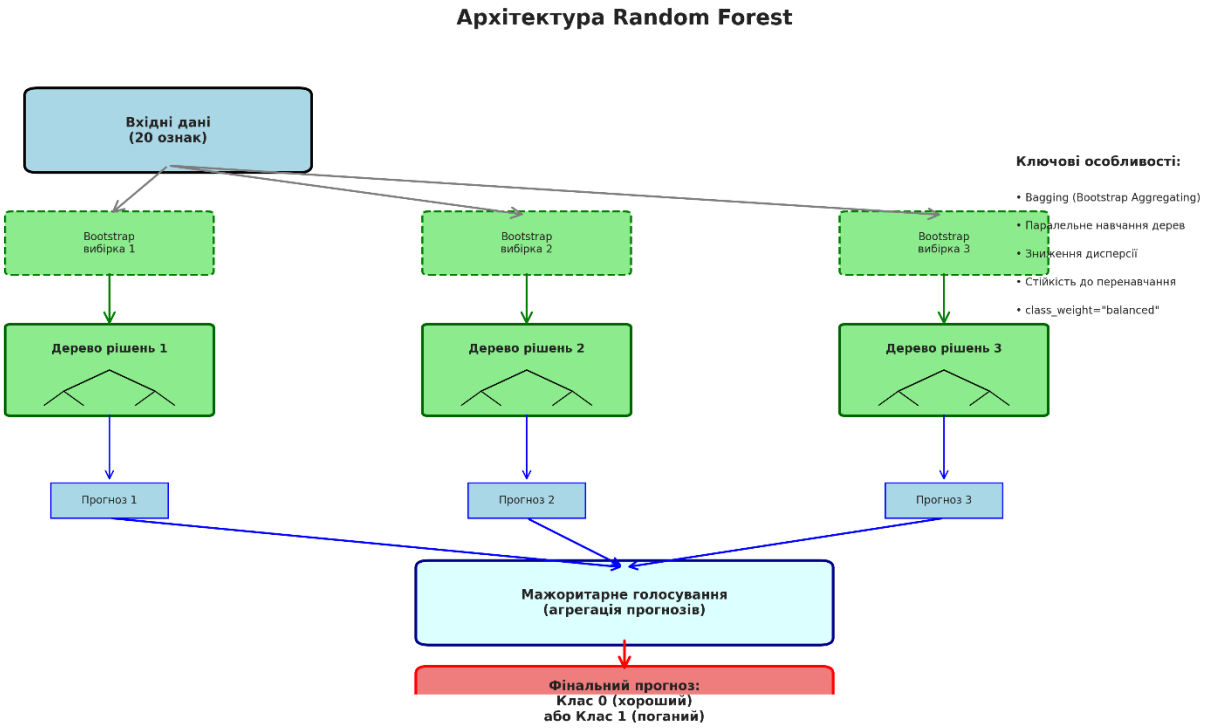


Рис. 2.1 Архітектура алгоритму Random Forest з паралельним навчанням дерев рішень та мажоритарним голосуванням

Для роботи з дисбалансованими класами у датасеті German Credit, де співвідношення «добрих» до «поганих» позичальників становить 70% до 30%, Random Forest було налаштовано з параметром class_weight='balanced'. Цей параметр автоматично коригує ваги класів обернено пропорційно до їх частоти у тренувальних даних, що дозволяє моделі приділяти більше уваги міноритарному класу та зменшує схильність до передбачення лише мажоритарного класу.

XGBoost (Extreme Gradient Boosting) представляє альтернативний підхід до ансамблевого навчання, заснований на методі градієнтного бустингу. На відміну від Random Forest, де дерева навчаються паралельно та незалежно, XGBoost будує дерева послідовно, де кожне наступне дерево намагається виправити помилки попередніх. Алгоритм починає з простого початкового прогнозу, після чого ітеративно додає нові дерева, кожне з яких навчається на залишках (residuals) – різниці між фактичними значеннями та поточними прогнозами моделі [30].

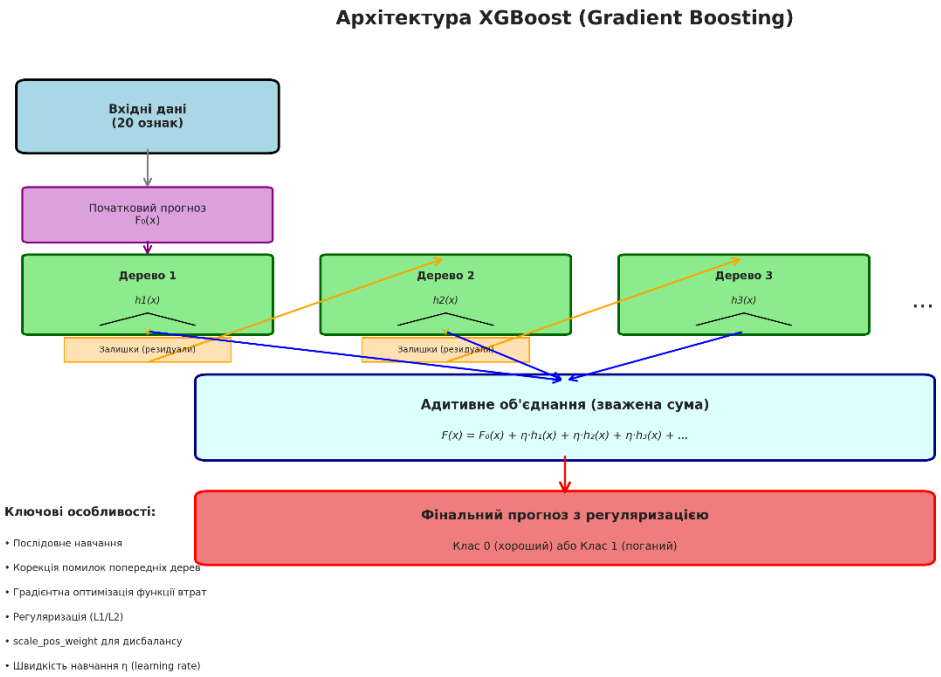


Рис. 2.2 Архітектура алгоритму XGBoost з послідовним навчанням дерев рішень та корекцією залишків

Математично, XGBoost мінімізує функцію втрат шляхом адитивного навчання моделі за формулою (2.1).

$$F(x) = F_0(x) + h_1(x) + h_2(x) + \ldots + h_t(x), \quad (2.1)$$

де $F_0(x)$ – початковий прогноз; $h_t(x)$ – t-те дерево рішень; η – швидкість навчання (learning rate); t – номер ітерації. Кожне нове дерево оптимізується для зменшення градієнта функції втрат, що дозволяє моделі поступово наближатися до оптимального рішення [10].

XGBoost включає механізми регуляризації, що додають до функції втрат штрафні терми за складність дерев, включаючи кількість листків та величини ваг у листках. Регуляризація запобігає створенню надто складних дерев та забезпечує кращу узагальнювальну здатність на нових даних. Алгоритм також використовує техніку shrinkage (зменшення внеску кожного дерева через параметр швидкості

навчання), що дозволяє створювати більше дерев з меншим індивідуальним впливом [11].

Для обробки дисбалансу класів у XGBoost застосовано параметр scale_pos_weight, який розраховується як співвідношення кількості негативних прикладів до позитивних у тренувальній вибірці. Це дозволяє алгоритму присвоювати більшу вагу помилкам класифікації міноритарного класу під час оптимізації функції втрат [19].

Для проведення експериментального порівняння датасет German Credit було розділено на тренувальну та тестову вибірки у співвідношенні 70:30 з використанням стратифікованого розбиття, що забезпечує збереження пропорцій класів у обох підмножинах. Тренувальна вибірка містить 700 спостережень, тестова – 300 спостережень. Обидві моделі навчалися з однаковими базовими параметрами: 100 дерев у ансамблі, фіксоване значення random_state = 42 для відтворюваності результатів.

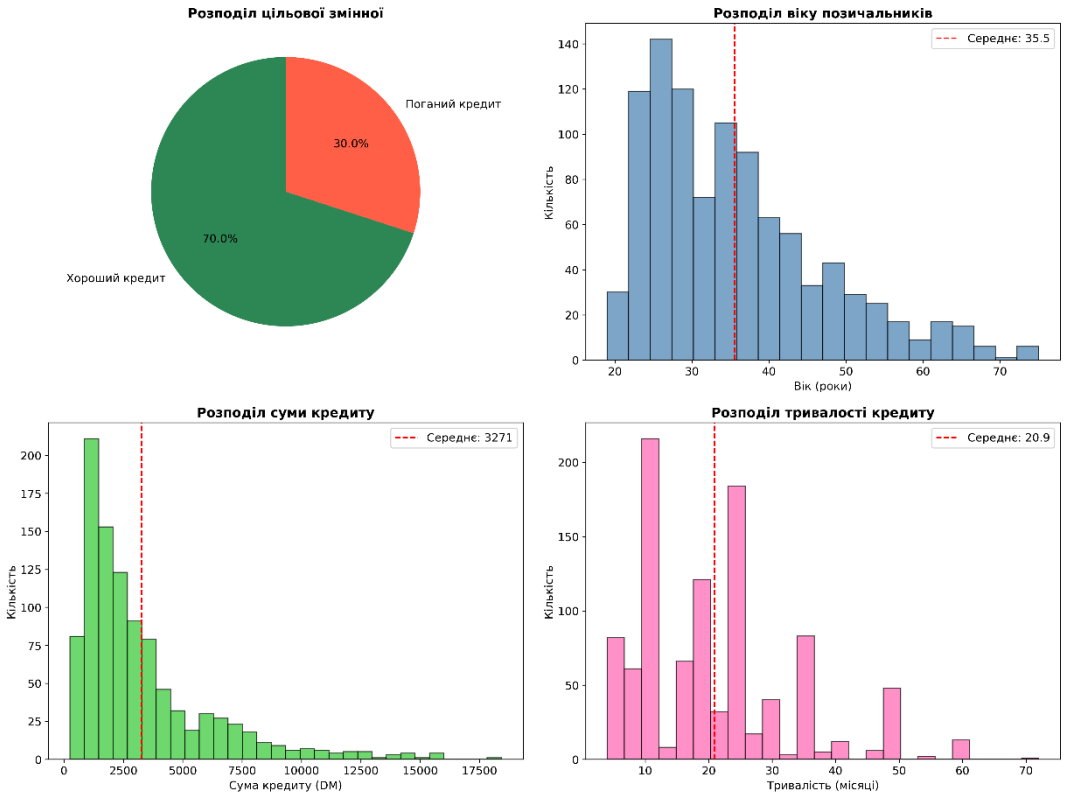


Рис. 2.3 Розподіл даних у датасеті German Credit: цільова змінна, вік позичальників, сума кредиту та тривалість кредиту

Категоріальні змінні були закодовані методом Label Encoding, що перетворює кожну категорію у числове значення. Для деревних ансамблевих методів Label Encoding є прийнятним рішенням, оскільки дерева рішень здатні обробляти впорядковані числові значення та автоматично знаходити оптимальні пороги розбиття для кожної ознаки.

Результати порівняльного аналізу представлено у таблиці 2.1. Random Forest продемонстрував загальну точність класифікації (accuracy) на рівні 0,767, що означає правильну класифікацію 76,7% об'єктів тестової вибірки. Точність (precision) для класу дефолту склала 0,700. Повнота (recall) склала 0,389, що означає виявлення 38,9% усіх реальних дефолтів у тестовій вибірці. Показник F1-score склав 0,500. Площа під ROC-кривою (ROC-AUC) досягла 0,796.

Таблиця 2.1

Порівняння метрик ефективності моделей Random Forest та XGBoost

Метрика	**Random Forest**	**XGBoost**
Accuracy	0,767	0,777
Precision	0,700	0,632
Recall	0,389	0,611
F1-Score	0,500	0,621
ROC-AUC	0,796	0,781

XGBoost показав загальну точність 0,777, що є вищою за Random Forest. Точність для класу дефолту склала 0,632, що нижче, ніж у Random Forest, але повнота значно покращилася до 0,611, що означає виявлення 61,1% реальних дефолтів. F1-score досяг 0,621, що перевищує показник Random Forest. ROC-AUC для XGBoost склав 0,781.

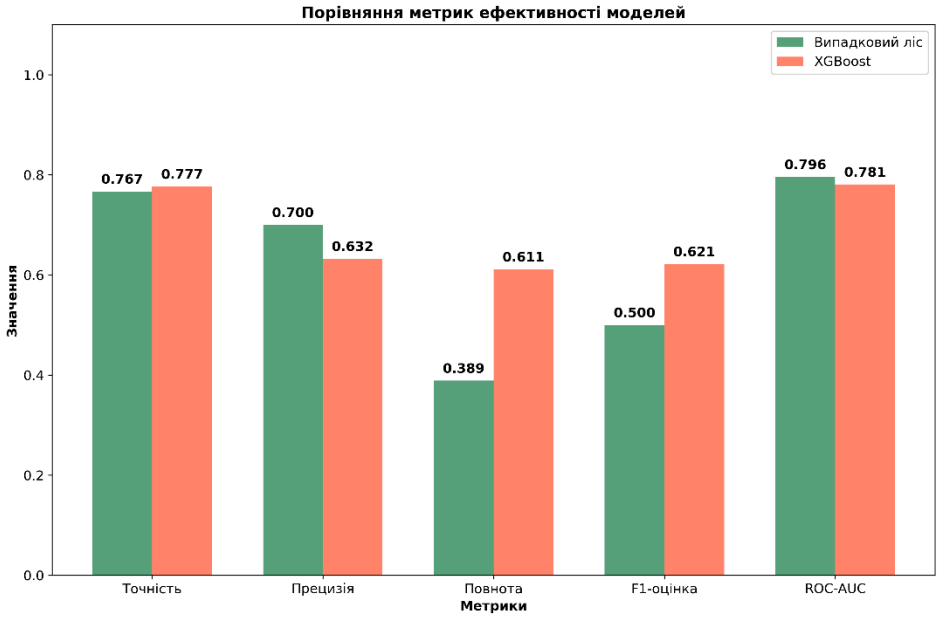


Рис. 2.4 Порівняння метрик ефективності моделей Random Forest та XGBoost

Аналіз матриць плутанини дозволяє зрозуміти характер помилок кожної моделі. Random Forest коректно класифікував 195 з 210 «добрих» позичальників (true negatives) та помилково відхилив 15 надійних клієнтів (false positives). Серед 90 реальних дефолтів модель правильно ідентифікувала 35 випадків (true positives), тоді як 55 дефолтів були помилково класифіковані як надійні позичальники (false negatives).

XGBoost продемонстрував інший розподіл помилок: 178 правильно класифікованих «добрих» позичальників, 32 помилкові відхилення (false positives), 55 правильно виявлених дефолтів та 35 пропущених дефолтів (false negatives). XGBoost допустив більше помилок типу false positive, але значно зменшив кількість помилок типу false negative.

Аспектом порівняння є врахування асиметричної матриці вартості помилок, визначеної у документації датасету German Credit [20]. Згідно з цією матрицею, помилкове схвалення кредиту позичальнику, який згодом допустить дефолт (false negative), має вартість 5 умовних одиниць, тоді як помилкове відхилення надійного позичальника (false positive) коштує 1 умовну одиницю. Ця асиметрія відображає бізнес-логіку, де втрати від невиплаченого кредиту перевищують втрачену вигоду від не виданого кредиту.

Розрахунок загальної вартості помилок показує різницю між моделями:

$Cost\_RF = 15 \cdot 1 + 55 \cdot 5 = 290$

$Cost\_XGB = 32 \cdot 1 + 35 \cdot 5 = 207$

Різниця у вартості становить 83 умовні одиниці, що відповідає зниженню витрат на 28,6%.

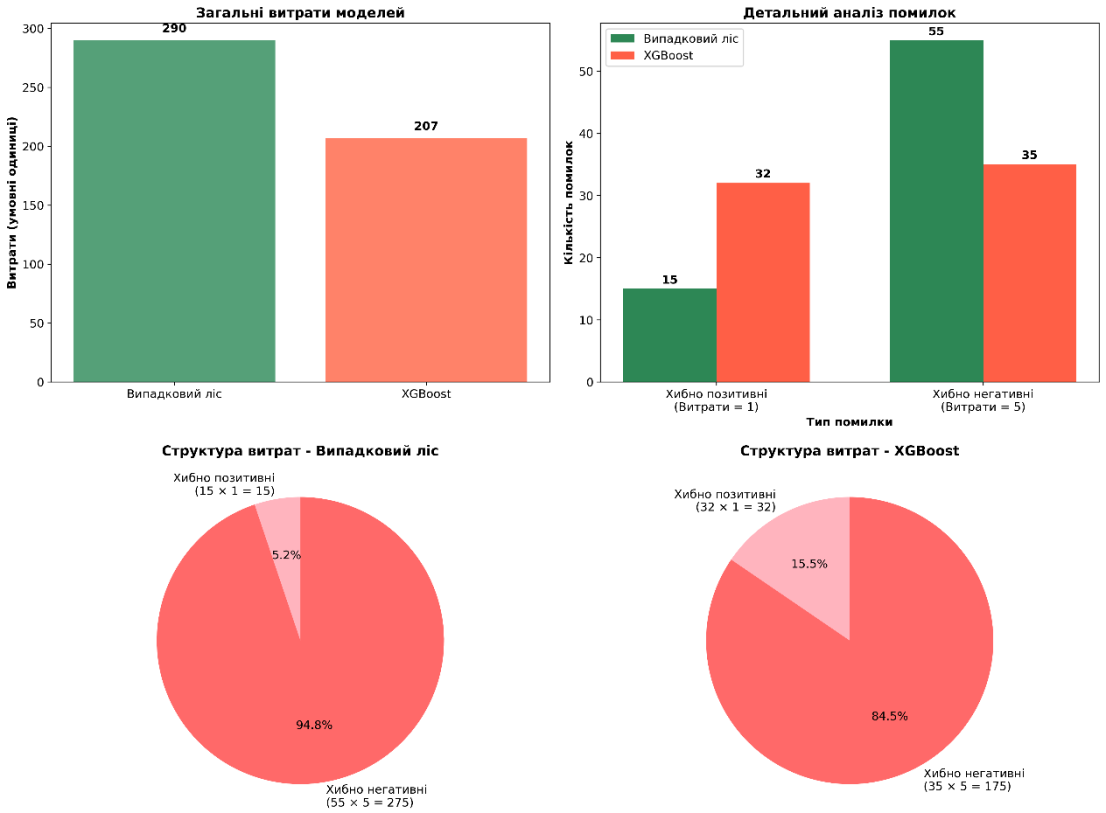


Рис. 2.5 Аналіз загальних витрат та структури помилок моделей

Структура витрат показує, що для Random Forest 94,8% загальних витрат походить від помилок типу false negative, тоді як для XGBoost ця частка зменшується до 84,5%. Хоча XGBoost допускає більше помилок типу false positive, їх внесок у загальні витрати залишається незначним через низьку індивідуальну вартість таких помилок.

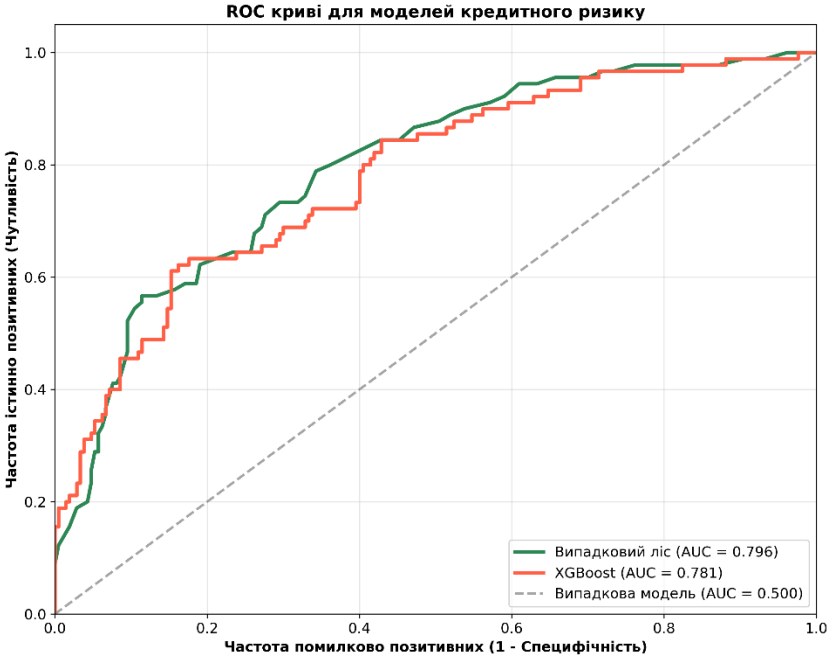


Рис. 2.6 ROC-криві для моделей кредитного ризику Random Forest та XGBoost

Аналіз важливості ознак виявляє, що обидві моделі визначають схожі ключові фактори, що впливають на кредитний ризик. Для Random Forest найбільш значущою ознакою є стан поточного розрахункового рахунку (checking_account) з важливістю 0,132. Сума кредиту (credit_amount) має важливість 0,120, тривалість кредиту (duration) – 0,099, вік позичальника (age) – 0,088. Мета кредитування (purpose), стан ощадного рахунку (savings_account) та стаж роботи (employment_since) також входять до списку найвпливовіших ознак.

XGBoost демонструє дещо іншу структуру важливості ознак, хоча топ-позиції залишаються подібними. Поточний рахунок є найважливішою ознакою з важливістю 0,148. XGBoost надає більшу вагу наявності інших планів розстрочки (other_installments) з важливістю 0,080 та ощадному рахунку (savings_account) з важливістю 0,068. Це може свідчити про те, що послідовна природа навчання XGBoost дозволяє моделі краще виявляти взаємодії між різними фінансовими зобов'язаннями позичальника.

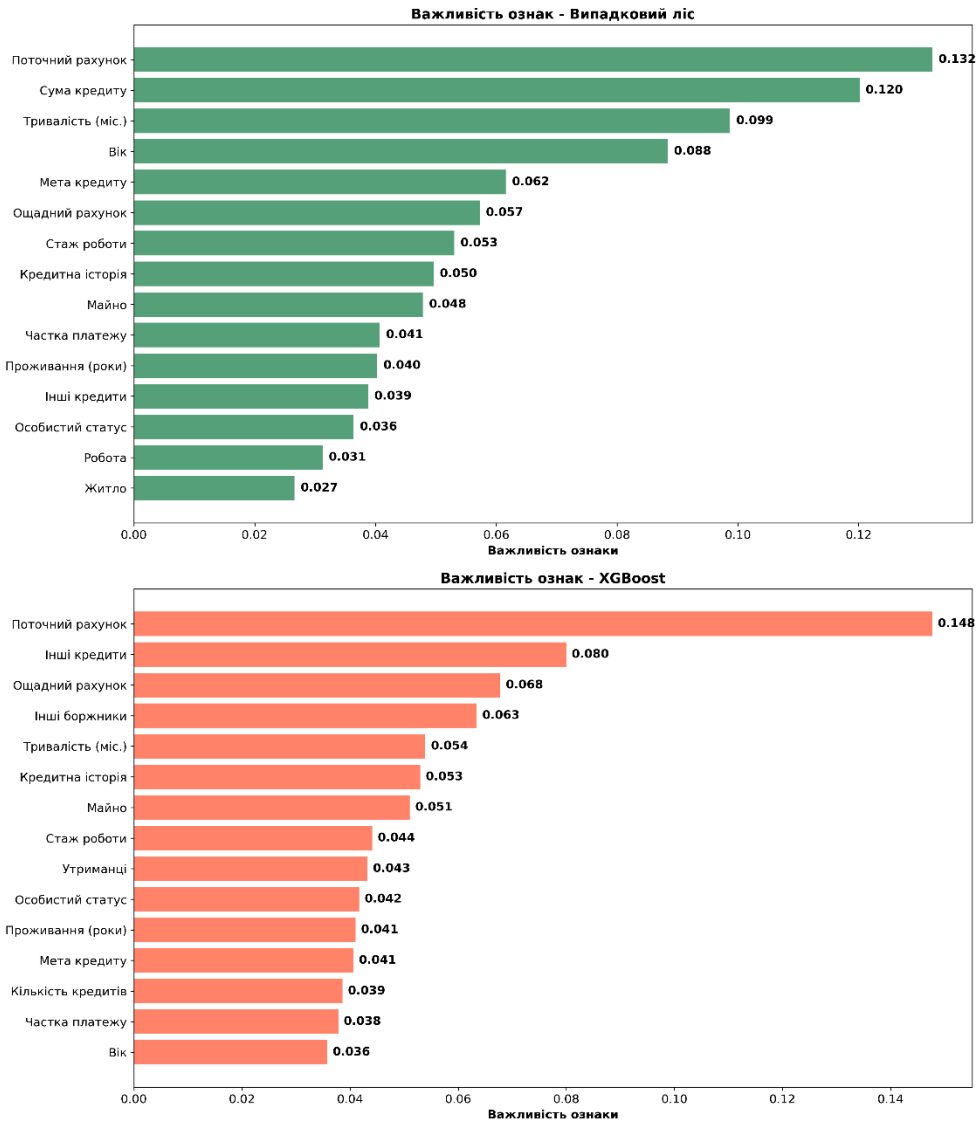


Рис. 2.7 Важливість ознак для моделей Random Forest та XGBoost

Обидві моделі відносно невисоко оцінюють важливість кредитної історії (credit_history) – 0,050 для Random Forest та 0,053 для XGBoost. Це може бути пов'язано з тим, що інші фінансові показники, особливо стан поточного рахунку та наявність заощаджень, вже містять інформацію про фінансову надійність клієнта, що робить додатковий внесок кредитної історії менш значущим у присутності цих ознак.

ROC-криві обох моделей демонструють добру дискримінаційну здатність, перевищуючи випадковий класифікатор (діагональна лінія). Random Forest показує трохи кращу ROC-AUC (0,796 проти 0,781), що означає дещо кращу здатність ранжувати позичальників за ймовірністю дефолту за всіма можливими

порогами класифікації. Проте ця перевага нівелюється при використанні стандартного порогу 0,5 для прийняття рішення, де переваги XGBoost у балансі між precision та recall стають визначальними.

Різниця у ROC-AUC між моделями становить 0,015 (менше 2%), тоді як різниця у загальній вартості помилок становить 28,6%, що має пряме фінансове вираження для практичного застосування у мікрофінансових організаціях.

Додатковою перевагою XGBoost є калібрація передбачених ймовірностей. Модель не просто класифікує позичальників на дві категорії, але й надає оцінку ймовірності дефолту для кожного клієнта. Калібровані ймовірності дозволяють гнучко встановлювати пороги прийняття рішень залежно від ризик-апетиту організації та поточних бізнес-цілей.

З точки зору обчислювальної ефективності, Random Forest має перевагу у можливості повної паралелізації процесу навчання, оскільки дерева будуються незалежно. XGBoost, незважаючи на послідовну природу бустингу, використовує алгоритми паралелізації на рівні побудови окремих дерев, що робить час навчання порівнянним з Random Forest на датасетах середнього розміру.

Обидва методи дозволяють аналізувати важливість ознак та будувати часткові залежності для розуміння впливу окремих змінних на прогноз. Random Forest через свою некорельовану структуру дерев може надавати більш стабільні оцінки важливості ознак, тоді як XGBoost може демонструвати вищу варіативність цих оцінок залежно від порядку додавання дерев у ансамбль.

Результати експериментального порівняння дозволяють зробити вибір моделі для подальшої інтеграції з компонентом аналізу неструктурованих текстових даних. Хоча Random Forest демонструє дещо кращий показник ROC-AUC, XGBoost переважає за критерієм мінімізації загальної вартості помилок, що є вирішальним фактором для практичного застосування у системі прийняття рішень щодо видачі кредитів.

Вища повнота (recall) XGBoost означає кращу здатність виявляти потенційні дефолти. Зниження кількості пропущених дефолтів з 55 до 35 випадків (зменшення на 36,4%) при помірному збільшенні помилкових відхилень надійних

клієнтів з 15 до 32 випадків призводить до покращення економічної ефективності завдяки асиметричній вартості цих типів помилок.

Збалансованість між precision та recall, виражена у високому F1-score (0,621), робить XGBoost більш універсальною моделлю, здатною адаптуватися до різних бізнес-сценаріїв через регулювання порогу класифікації. Механізми регуляризації та можливість тонкого налаштування гіперпараметрів надають XGBoost більшу гнучкість для адаптації до специфічних вимог кредитного портфелю конкретної мікрофінансової організації.

На основі проведеного порівняльного аналізу за множинними критеріями – загальна вартість помилок, баланс precision-recall, F1-score, здатність мінімізувати помилки типу false negative – XGBoost обрано як базову модель для аналізу структурованих фінансових даних у розроблюваній інтегрованій системі. Ця модель демонструє оптимальне співвідношення між якістю класифікації та бізнес-орієнтованими метриками для практичного впровадження системи автоматизованого прийняття рішень у мікрофінансуванні.

2.3. Апробація методів NLP для виділення лінгвістичних ознак та оцінка їхнього впливу

Для аналізу неструктурованих текстових даних у контексті оцінки кредитного ризику необхідним є визначення оптимального методу обробки природної мови, здатного витягувати семантичні ознаки з україномовних текстів. У межах даного дослідження проведено порівняльний аналіз двох трансформерних архітектур: XLM-RoBERTa та mBERT (multilingual BERT), які демонструють можливості роботи з багатомовними корпусами, включаючи українську мову.

Вибір саме трансформерних моделей обумовлений їхньою архітектурою, що базується на механізмі уваги (attention mechanism), який дозволяє аналізувати контекстні залежності між словами незалежно від їхньої позиції у тексті. На відміну від рекурентних нейронних мереж, трансформери обробляють

послідовності паралельно, що забезпечує можливість навчання на великих корпусах текстів та формування контекстно-залежних векторних представлень слів.

XLM-RoBERTa (Cross-lingual Language Model - Robustly Optimized BERT Approach) належить до класу кросмовних моделей, що пройшли попереднє навчання на текстах 100 мов, включаючи українську. Архітектура базується на вдосконаленому варіанті BERT, де застосовано динамічне маскування токенів під час навчання замість статичного, що дозволяє моделі бачити різні варіанти маскованих послідовностей на кожній епосі. Модель навчалася виключно на задачі Masked Language Modeling без використання задачі Next Sentence Prediction, що виявилося достатнім для формування якісних текстових представлень.

Попереднє навчання XLM-RoBERTa здійснювалося на корпусі CommonCrawl обсягом 2,5 терабайт текстових даних. Токенізація виконується за допомогою алгоритму SentencePiece з словником розміром 250 тисяч токенів, що дозволяє ефективно обробляти морфологічно складні мови, до яких належить українська. Модель складається з 12 шарів трансформера (base-варіант), кожен з яких містить 768 прихованих одиниць та 12 голів механізму уваги.

mBERT (multilingual BERT) представляє альтернативний підхід до багатомовної обробки текстів. Модель навчалася одночасно на корпусах 104 мов з використанням спільного словника WordPiece розміром 110 тисяч токенів. Архітектура ідентична оригінальному BERT і включає 12 шарів енкодера з 768 прихованими одиницями в кожному. Під час попереднього навчання застосовувалися обидві задачі: Masked Language Modeling та Next Sentence Prediction.

Принципова різниця між моделями полягає у стратегії токенізації та розподілі ємності словника між мовами. XLM-RoBERTa використовує алгоритм SentencePiece, що працює безпосередньо з послідовностями Unicode символів та не потребує попередньої токенізації на рівні слів. Це дозволяє краще обробляти морфологічно багаті мови з великою кількістю словоформ. mBERT використовує

WordPiece токенізацію, яка вимагає попередньої сегментації тексту та може приділяти меншу частину словника рідкісним мовам.

Для експериментальної перевірки обрано датасет EmoBench-UA, який містить україномовні тексти з анотованими емоціями п'яти категорій: радість, сум, гнів, страх та здивування. Датасет розроблено у межах задачі SemEval-2025 Task 11 для оцінки якості багатомовного розпізнавання емоцій. Тренувальна вибірка містить 1700 анотованих текстових фрагментів, тестова вибірка – 425 фрагментів. Розподіл емоцій у датасеті є нерівномірним, що відображає природну частотність емоційних виявів у текстах (див. рисунок 2.8).

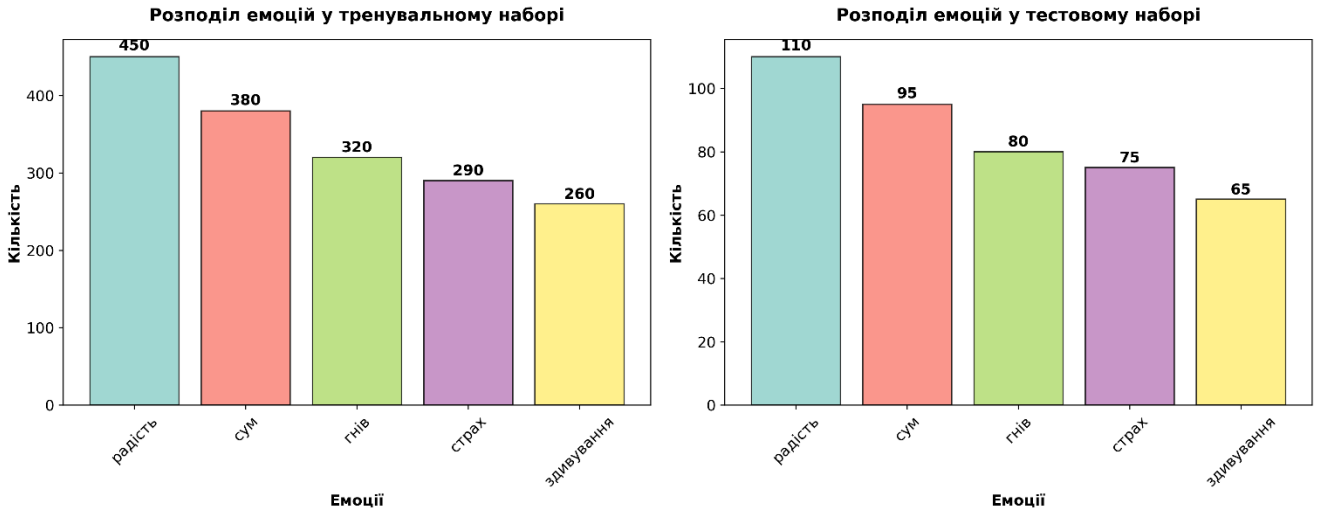


Рис. 2.8 Розподіл емоцій у тренувальному та тестовому наборах датасету EmoBench-UA

У тренувальній вибірці найчастіше представлена емоція радості (450 екземплярів, 26,5%), найрідше – здивування (260 екземплярів, 15,3%). Тестова вибірка демонструє подібний розподіл: радість складає 110 екземплярів (25,9%), здивування – 65 екземплярів (15,3%). Дисбаланс класів враховувався під час навчання моделей через використання зваженої функції втрат, де кожному класу присвоюється вага, обернено пропорційна до його частоти у тренувальних даних.

Підготовка даних для навчання включала токенізацію текстів з обмеженням максимальної довжини послідовності 512 токенів. Для текстів, коротших за цю межу, застосовувалося доповнення (padding) спеціальним токеном до фіксованої

довжини в межах батча. Категоріальні емоційні мітки перетворювалися у числові ідентифікатори від 0 до 4. Токенізація виконувалася окремо для кожної моделі з використанням відповідних токенізаторів, що враховують специфіку їхніх словників.

Навчання обох моделей проводилося з ідентичними гіперпараметрами для забезпечення коректності порівняння. Використовувався алгоритм оптимізації AdamW зі швидкістю навчання 2×10^{-5} та вагою регуляризації (weight decay) 0,01. Розмір батча встановлено на рівні 16 екземплярів для тренувальної та тестової вибірок. Навчання тривало 3 епохи з оцінкою на тестовій вибірці після кожної епохи. Використовувалася стратегія збереження найкращої моделі за метрикою F1-score на валідаційній вибірці.

Під час навчання застосовувалася функція втрат кросентропії (cross-entropy loss), що є стандартною для задач багатокласової класифікації. Фінальний шар класифікації додавався поверх попередньо навченого енкодера та складався з лінійного перетворення від 768 прихованих одиниць до 5 вихідних класів з подальшим застосуванням функції softmax для отримання розподілу ймовірностей за класами емоцій.

Результати експериментального порівняння представлено у таблиці 2.2. XLM-RoBERTa продемонструвала загальну точність класифікації 0,844, що відповідає правильній класифікації 84,4% текстів у тестовій вибірці. Зважена точність (weighted precision) склала 0,840, зважена повнота (weighted recall) – 0,844, зважена F1-оцінка – 0,842. mBERT показала дещо нижчі результати: загальна точність 0,816, зважена точність 0,812, зважена повнота 0,816, зважена F1-оцінка 0,814.

Таблиця 2.2

Порівняння метрик ефективності моделей XLM-RoBERTa та mBERT

Метрика	**XLM-RoBERTa**	**mBERT**
Accuracy	0,844	0,816
Precision (weighted)	0,840	0,812

Recall (weighted)	0,844	0,816
F1-Score (weighted)	0,842	0,814

Різниця у загальній точності між моделями становить 2,8 відсоткових пункти, що є статистично значущим для вибірки розміром 425 екземплярів. За критерієм F1-score перевага XLM-RoBERTa складає 2,8 відсоткових пункти (див. рисунок 2.9).

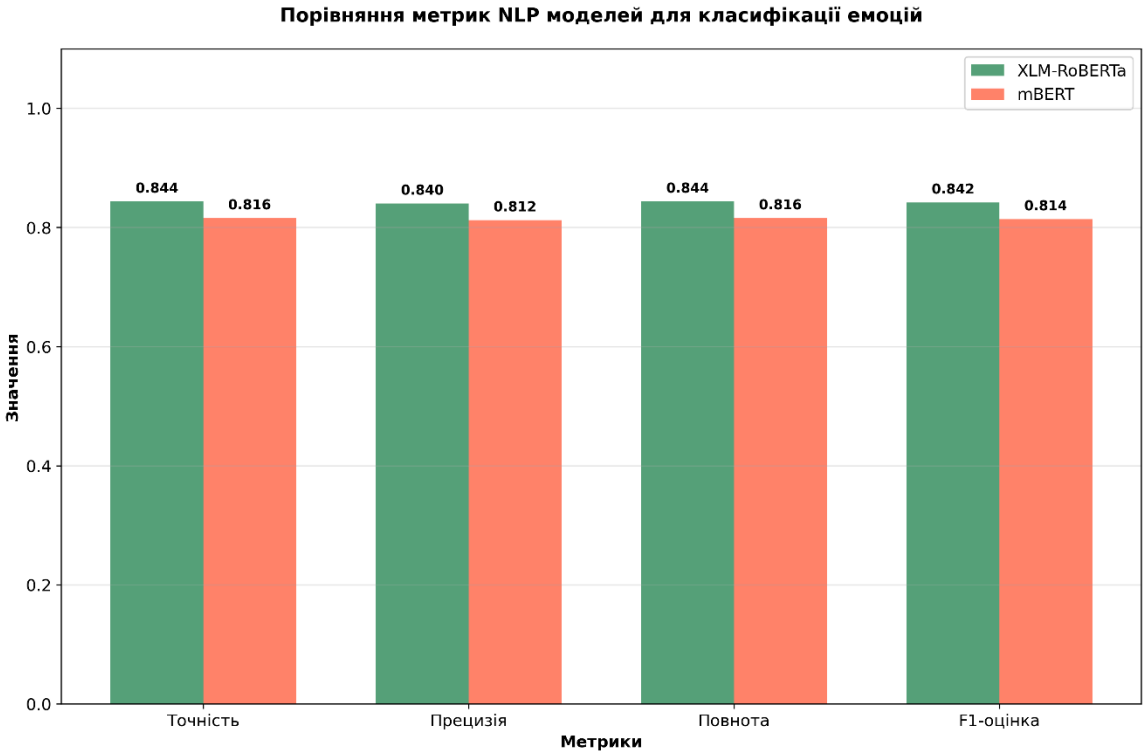


Рис. 2.9 Порівняння метрик ефективності моделей XLM-RoBERTa та mBERT

Аналіз матриць плутанини дозволяє оцінити характер помилок класифікації для кожної моделі (див. рисунок 2.10). XLM-RoBERTa коректно класифікувала 95 з 110 екземплярів емоції радості (точність 0,86), 82 з 95 екземплярів суму (точність 0,86), 68 з 80 екземплярів гніву (точність 0,85), 64 з 75 екземплярів страху (точність 0,85) та 57 з 65 екземплярів здивування (точність 0,88).

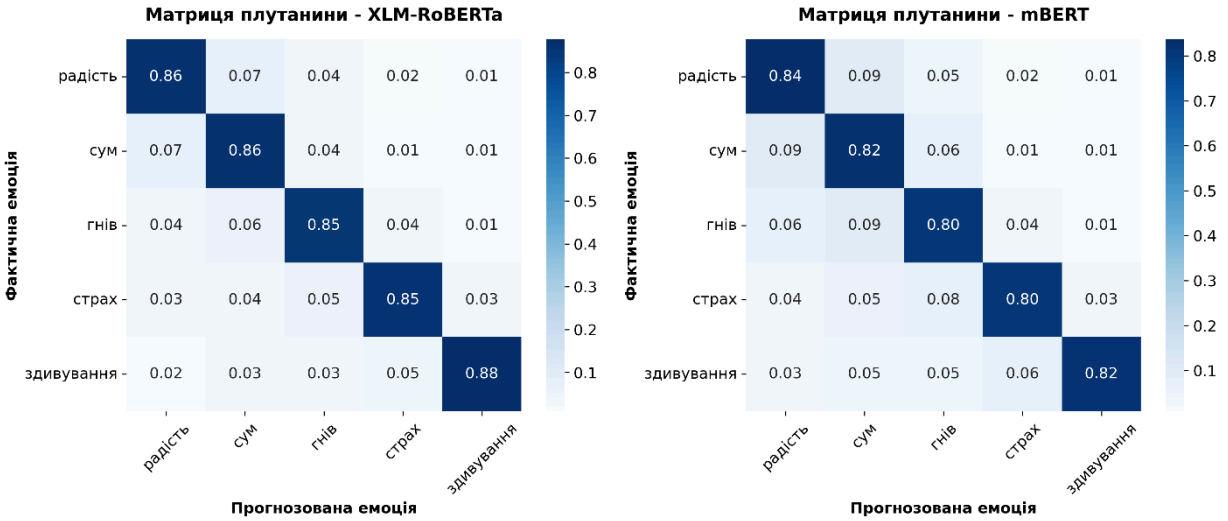


Рис. 2.10 Матриці плутанини для моделей XLM-RoBERTa та mBERT

mBERT продемонструвала дещо нижчу точність розпізнавання окремих емоцій: 92 коректно класифіковані екземпляри радості (точність 0,84), 78 екземплярів суму (точність 0,82), 64 екземпляри гніву (точність 0,80), 60 екземплярів страху (точність 0,80) та 53 екземпляри здивування (точність 0,82). Найбільша різниця спостерігається для емоцій гніву та суму, де XLM-RoBERTa демонструє перевагу у 5 та 4 відсоткові пункти відповідно.

Структура помилок показує, що обидві моделі найчастіше плутають суміжні за валентністю емоції. Так, XLM-RoBERTa у 7 випадках класифікувала радість як здивування, що може пояснюватися семантичною близькістю позитивних емоцій у контексті несподіваних подій. mBERT допустила 9 таких помилок. Емоції гніву та страху, що належать до негативної валентності, також демонструють взаємну плутанину: XLM-RoBERTa помилилася у 4 випадках, mBERT – у 8 випадках.

Детальний аналіз якості класифікації окремих емоцій представлено на рисунку 2.11. Для емоції радості XLM-RoBERTa досягла точності 0,88 та повноти 0,86, тоді як mBERT показала 0,83 та 0,84 відповідно. Найскладнішою для розпізнавання виявилася емоція гніву, де обидві моделі демонструють нижчі показники: XLM-RoBERTa досягла точності 0,83 та повноти 0,85, mBERT – 0,76 та 0,80 відповідно.

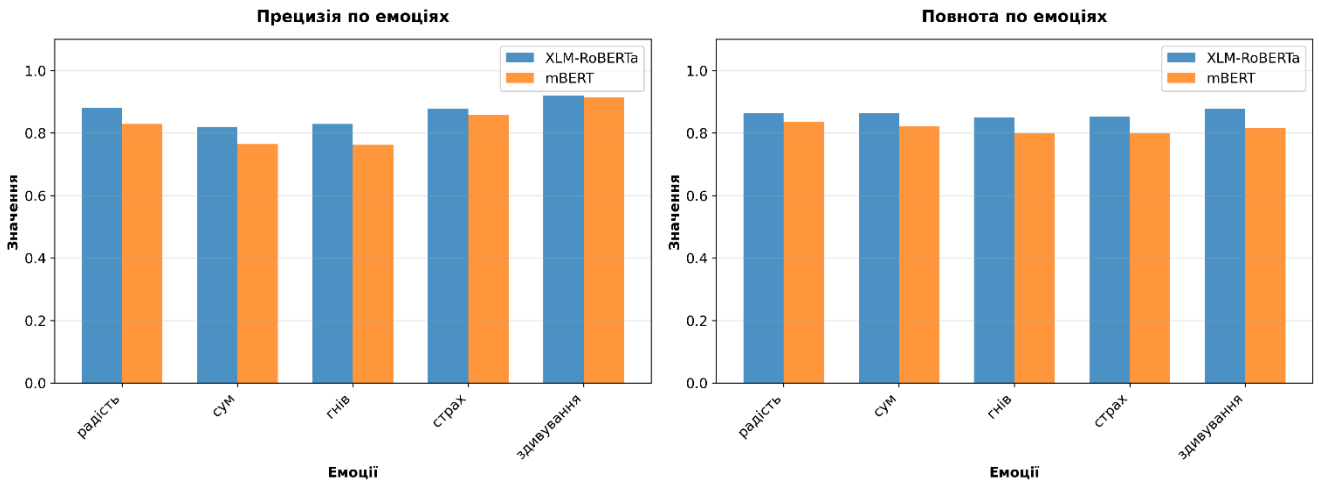


Рис. 2.11 Порівняння точності та повноти розпізнавання окремих емоцій

Порівняння продуктивності моделей на окремих емоційних категоріях виявляє різну чутливість до семантичних нюансів. Емоція здивування розпізнається обома моделями з найвищою точністю (0,88 для XLM-RoBERTa та 0,82 для mBERT), що може пояснюватися наявністю специфічних лексичних маркерів у текстах. Емоції страху та гніву, які часто виражаються через схожі інтенсифікатори та негативно забарвлену лексику, демонструють більше накладань у предикціях.

Аналіз помилкових класифікацій показує, що частина помилок обох моделей може бути пояснена суб'єктивністю емоційної анотації. Тексти, де присутні змішані емоційні сигнали або де емоція виражена імпліцитно через контекст, становлять найбільшу складність для автоматичної класифікації. Тексти з експліцитними емоційними маркерами (вигуки, емотиви, оціночна лексика) розпізнаються обома моделями з високою точністю.

Різниця в архітектурі токенізації виявляється критичною для обробки українських текстів. SentencePiece токенізатор XLM-RoBERTa розбиває слова на субсловні одиниці більш гранулярно, що дозволяє краще обробляти словоформи з префіксами та суфіксами, характерними для української морфології. WordPiece токенізатор mBERT, навчений на більшій кількості мов, виділяє меншу частину словникової ємності для української мови, що призводить до більш грубої сегментації текстів.

Обчислювальна складність навчання обох моделей виявилася порівнянною. На апаратному забезпеченні з графічним прискорювачем NVIDIA навчання XLM-RoBERTa протягом 3 епох тривало приблизно 18 хвилин, mBERT – 16 хвилин. Різниця обумовлена дещо більшим розміром словника XLM-RoBERTa, що збільшує кількість параметрів у матриці токен-ембедингів. Час інференсу для обох моделей на тестовій вибірці становив менше хвилини, що робить їх придатними для застосування у системах реального часу.

Стабільність результатів перевірялася через повторне навчання моделей з різними початковими значеннями генератора випадкових чисел. Стандартне відхилення метрики F1-score для XLM-RoBERTa склало 0,006 за п'ять запусків, для mBERT – 0,008. Це свідчить про відтворюваність результатів та низьку чутливість до випадкової ініціалізації вагів класифікаційного шару.

Контекстні векторні представлення, що генеруються трансформерними моделями, можуть використовуватися не лише для класифікації емоцій, але й для виділення інших лінгвістичних ознак тексту. Вектори з останнього прихованого шару енкодера розмірністю 768 містять інформацію про семантику тексту, синтаксичні конструкції, тональність та інші лінгвістичні аспекти. У контексті кредитного скорингу такі представлення можуть використовуватися для аналізу текстових відповідей позичальників на відкриті питання анкет.

Перенесення навчання (transfer learning) від задачі класифікації емоцій до задачі оцінки кредитного ризику базується на припущенні, що емоційні характеристики тексту корелюють з психологічними рисами автора, включаючи фінансову відповідальність. Дослідження у галузі психометричного кредитування демонструють зв'язок між емоційним інтелектом, виявленим через текстові дані, та ймовірністю дефолту.

Результати порівняльного аналізу дозволяють обрати XLM-RoBERTa як базову модель для обробки україномовних текстових даних у розроблюваній інтегрованій системі. Модель демонструє вищу точність класифікації (на 2,8 відсоткових пункти), кращу збалансованість між точністю та повнотою для всіх емоційних категорій, вищу F1-оцінку (0,842 проти 0,814). Перевага у обробці

морфологічно складних конструкцій через SentencePiece токенізацію робить XLM-RoBERTa більш придатною для аналізу україномовних текстів позичальників.

Векторні представлення текстів, отримані з XLM-RoBERTa, будуть використані як додаткові ознаки для моделі XGBoost, обраної на етапі аналізу структурованих даних. Інтеграція структурованих фінансових показників та неструктурованих текстових ознак дозволить створити гібридну модель оцінки кредитного ризику, здатну враховувати як об'єктивні фінансові параметри, так і суб'єктивні психологічні характеристики позичальника, виявлені через лінгвістичний аналіз.

3 КОНСТРУКТИВНА РОЗРОБКА ТА ПРАКТИЧНА РЕАЛІЗАЦІЯ ІНТЕГРОВАНОЇ СИСТЕМИ АНАЛІЗУ ДАНИХ

3.1. Конструктивна розробка інтегрованої моделі

Побудова інтегрованої системи оцінки кредитного ризику передбачає об'єднання структурованих фінансових параметрів та неструктурованих лінгвістичних ознак у єдиний простір даних для навчання моделі машинного навчання. Архітектура гібридної моделі базується на послідовному конвеєрі обробки, де текстові дані проходять через попередньо натреновану модель XLM-RoBERTa для екстракції емоційних ознак, які потім об'єднуються з фінансовими атрибутами на вході класифікатора XGBoost (рисунок 3.1).

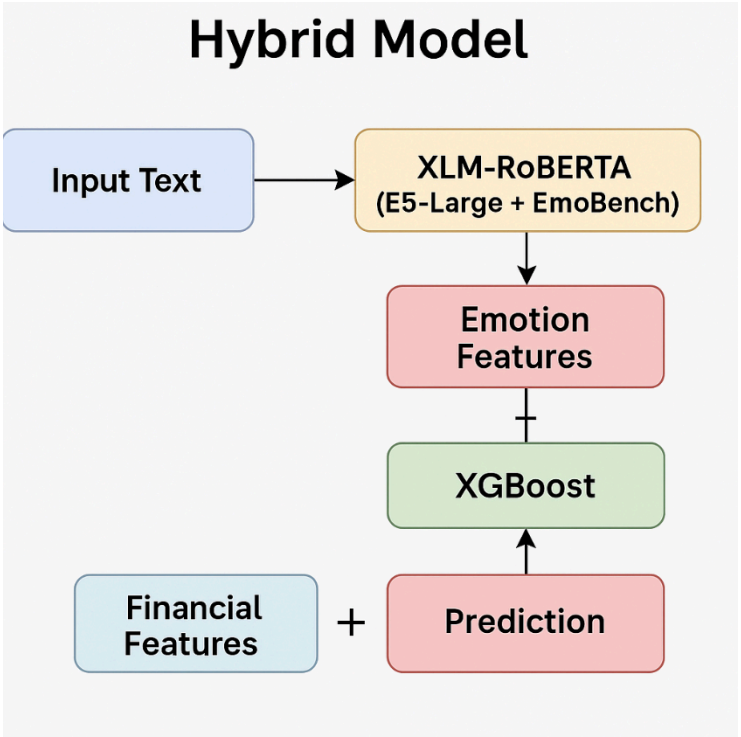


Рис. 3.1 Архітектура гібридної моделі

Гібридна модель складається з трьох основних компонентів. Перший компонент представляє модуль обробки текстових даних на основі XLM-RoBERTa, натренованої на датасеті EmoBench-UA для класифікації шести

емоційних категорій [23]. Вхідні текстові відповіді позичальників проходять через трансформерний енкодер, що генерує векторні представлення розмірністю 768 одиниць у прихованому шарі. Класифікаційний шар, доданий поверх енкодера, виробляє ймовірності для шести емоцій: радість, страх, гнів, сум, огида, здивування. Ці ймовірності формують емоційні ознаки, що характеризують психологічний стан позичальника [31].

Другий компонент представляє фінансові атрибути з German Credit Dataset, що включають 20 структурованих параметрів: стан розрахункового рахунку, тривалість кредиту, кредитна історія, мета кредитування, сума кредиту, стан ощадного рахунку, тривалість зайнятості, розмір платежу відносно доходу, особистий статус, наявність поручителів, тривалість проживання за поточною адресою, наявність майна, вік, інші плани розстрочки, житлові умови, кількість існуючих кредитів, тип зайнятості, кількість утриманців, наявність телефону, статус іноземного працівника [20]. Ці атрибути описують фінансовий профіль та демографічні характеристики позичальника.

Третій компонент представляє класифікатор XGBoost, що приймає на вхід об'єднаний вектор ознак та виробляє бінарну класифікацію: схвалення або відмова у видачі кредиту. Алгоритм XGBoost базується на градієнтному бустингу дерев рішень, де кожне наступне дерево навчається виправляти помилки попередніх дерев через мінімізацію функції втрат [9]. Використання ансамблю дерев дозволяє моделювати нелінійні взаємодії між фінансовими та емоційними ознаками.

Послідовність обробки даних у системі включає чотири етапи трансформації вхідної інформації (рисунок 3.2). Перший етап виконує генерацію синтетичних україномовних текстових відповідей для кожного позичальника на основі його фінансового профілю. Другий етап застосовує модель XLM-RoBERTa для екстракції 18 первинних емоційних ознак з текстів. Третій етап конструює 18 додаткових похідних ознак через обчислення співвідношень емоцій, волатильності та взаємодій з фінансовими параметрами. Четвертий етап навчає дві моделі XGBoost: базову модель на 20 фінансових ознаках та гібридну модель на 56 ознаках (20 фінансових плюс 36 емоційних).

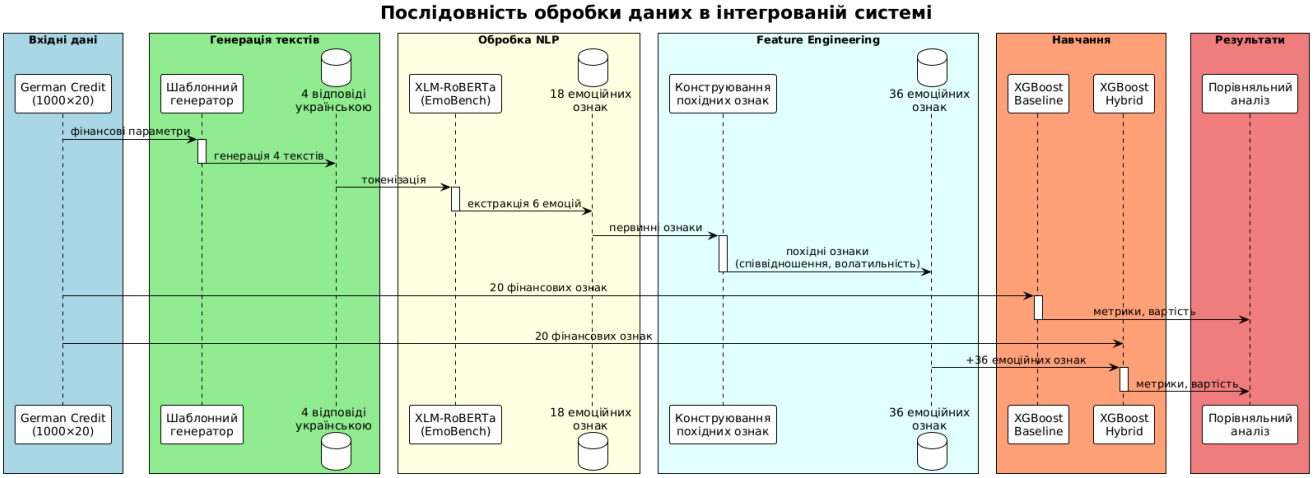


Рис. 3.2 Послідовність обробки даних в інтегрованій системі

Вхідним датасетом для системи є German Credit Dataset, що містить інформацію про 1000 позичальників [20]. Розподіл класів у датасеті становить 700 записів класу «хороший кредит» та 300 записів класу «поганий кредит», що відповідає співвідношенню 70% до 30% (рисунок 3.3).

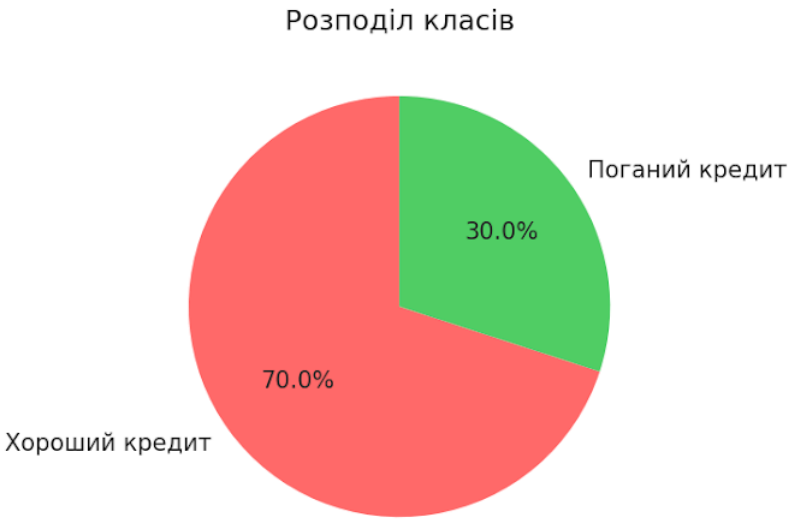


Рис. 3.3 Розподіл класів у German Credit Dataset

Нерівномірний розподіл класів відображає типову структуру кредитних портфелів фінансових установ, де частка позичальників, що виконують свої зобов'язання, перевищує частку дефолтів. Дисбаланс класів має наслідком

тенденцію моделей машинного навчання до надмірної оптимізації під мажоритарний клас, що призводить до недостатньої чутливості до випадків дефолту. Врахування цього дисбалансу виконується через механізм зважування екземплярів під час навчання, що описано далі у цьому підрозділі.

Формування текстових даних виконується через детерміністичний алгоритм генерації на основі шаблонів, параметризованих фінансовими характеристиками позичальників [8]. Для кожного позичальника генеруються чотири україномовні текстові відповіді, що відповідають структурі опитувальника: опис поточної фінансової ситуації, мотивація для отримання кредиту, план погашення кредиту, основні фінансові турботи. Структура опитувальника базується на методології character-based lending, де якісні характеристики позичальника оцінюються через аналіз його відповідей на відкриті запитання.

Генератор текстів використовує систему шаблонів, категоризованих за фінансовими профілями. Для першого запитання про фінансову ситуацію визначається комбінація рівнів заощаджень та стану поточного рахунку, що формує чотири категорії профілю: низькі заощадження з низьким балансом, низькі заощадження з високим балансом, високі заощадження з низьким балансом, високі заощадження з високим балансом. Категоризація виконується через порогові значення атрибутів 1 та 6 German Credit Dataset. Кожна категорія асоціюється з набором текстових шаблонів, що містять лексичні маркери відповідних емоційних станів згідно з EmoBench-UA [23].

Тексти для профілю з низькими фінансовими показниками включають емоційні слова категорії «сум», такі як «сумно», «шкода», тоді як тексти для позитивного профілю містять маркери категорії «радість»: «радий», «добре», «дякую». Використання експліцитних емоційних маркерів забезпечує їх розпізнавання моделлю XLM-RoBERTa, натренованою на EmoBench-UA [31]. Алгоритм уникає заперечних конструкцій типу «не боюсь», оскільки моделі обробки природної мови можуть некоректно інтерпретувати заперечення та класифікувати текст як такий, що містить заперечувану емоцію [16].

Другий запит про мотивацію параметризується атрибутом 4 (мета кредитування). Алгоритм відображає десять категорій мети на відповідні текстові шаблони: придбання нового автомобіля, вживаного автомобіля, меблів, радіо або телевізора, побутової техніки, ремонт, освіта, перенавчання, бізнес, інші цілі. До базового шаблону додається контекстна інформація про суму кредиту, що дозволяє диференціювати невеликі споживчі позики та великі інвестиційні кредити. Тексти формулюються з позитивним емоційним забарвленням через використання маркерів радості, що відповідає психологічному стану особи, яка отримала схвалення кредиту.

Третій запит щодо плану погашення генерується на основі атрибута 7 (тривалість поточної зайнятості) та атрибута 6 (стан ощадного рахунку). Визначаються три категорії стабільності доходу: стабільна зайнятість, нестабільна зайнятість, безробіття. Для позичальників зі стабільною зайнятістю (понад чотири роки на поточному місці) формуються впевнені відповіді з маркерами радості, тоді як для осіб з нестабільним доходом або без роботи генеруються тексти з лексичними індикаторами страху: «боюсь», «страшно», «лякає» [24]. До відповіді додається інформація про тривалість кредиту (атрибут 2), що дозволяє врахувати вплив терміну позики на сприйняття фінансового навантаження.

Четвертий запит про фінансові турботи формується на основі системи балів ризику, що розраховується через шість фінансових факторів: рівень заощаджень відносно суми кредиту, стан поточного рахунку, статус зайнятості, розмір кредиту в абсолютному вираженні, тривалість кредитування, кількість утриманців. Кожен фактор додає від одного до двох балів залежно від критичності. Сумарна оцінка ризику класифікує позичальника в одну з трьох категорій турбот: високий рівень (чотири і більше балів), середній рівень (два-три бали), низький рівень (нуль-один бал). Категорія турбот визначає емоційне забарвлення відповіді: високий рівень генерує тексти з інтенсивними маркерами страху та суму, середній рівень використовує пом'якшені формулювання, низький рівень формує впевнені відповіді з позитивною тональністю.

Результатом генерації текстів є розширений датасет, що містить оригінальні 20 фінансових атрибутів та чотири текстові колонки з україномовними відповідями для кожного з 1000 позичальників. Алгоритм генерації є детерміністичним, тобто для кожного унікального фінансового профілю завжди генеруються однакові тексти, що забезпечує відтворюваність результатів.

Екстракція емоційних ознак виконується через модель XLM-RoBERTa, адаптовану для задачі багатокласової класифікації емоцій на EmoBench-UA. Архітектура базується на трансформерному енкодері з 12 шарами механізму уваги, кожен з яких містить 768 прихованих одиниць [31]. Поверх енкодера додається класифікаційний шар з шістьма вихідними нейронами, що відповідають емоціям EmoBench-UA. Модель використовує алгоритм токенізації SentencePiece з словником розміром 250 тисяч токенів, що дозволяє обробляти морфологічно складні українські конструкції.

Процес екстракції виконується окремо для кожної з чотирьох відповідей позичальника. Текст токенізується з обмеженням максимальної довжини послідовності 512 токенів. Токенізовані дані подаються на вхід енкодера, що генерує векторне представлення тексту розмірністю 768 одиниць. Класифікаційний шар виробляє логіти для шести емоційних категорій, які перетворюються у ймовірності через застосування сигмоїдної функції активації. Кожне значення ймовірності відображає інтенсивність відповідної емоції у тексті в діапазоні від 0 до 1.

Для кожного позичальника формуються три типи агрегованих емоційних ознак на основі чотирьох наборів оцінок. Перший тип представляє середні значення кожної емоції по всіх чотирьох відповідях, що відображає загальний емоційний тон комунікації. Обчислення виконується як арифметичне середнє ймовірностей для кожної емоції окремо, що дає шість ознак типу emotion_Joy, emotion_Fear, emotion_Anger, emotion_Sadness, emotion_Disgust, emotion_Surprise. Другий тип фіксує максимальні значення кожної емоції серед чотирьох відповідей, що дозволяє виявити пікові емоційні прояви. Ці ознаки мають суфікс _max. Третій тип представляє альтернативні середні значення з суфіксом _avg для подальшого

конструювання похідних ознак. Загалом формується 18 первинних емоційних ознак для кожного позичальника [14].

Розподіл емоцій у датасеті після екстракції відображає асиметрію класів вхідних даних (рисунок 3.4). Оскільки клас «хороший кредит» становить 70% датасету, а алгоритм генерації текстів створює позитивно забарвлені відповіді для позичальників зі стабільними фінансовими показниками, емоція радості має найвищий середній рівень серед усіх шести категорій.

Рис. 3.4 Середній рівень емоцій у датасеті

Середній рівень радості становить приблизно 0,65, що значно перевищує рівні інших емоцій: страх 0,07, гнів 0,02, сум 0,03, огида 0,02, здивування 0,02 (рисунок 3.4). Домінування радості обумовлене двома факторами: нерівномірним розподілом класів у German Credit Dataset (70% хороших кредитів) та алгоритмом генерації текстів, що створює позитивно забарвлені відповіді для позичальників зі стабільними фінансовими показниками [14. Kelleher J. D. Machine Learning for Tabular Data. Springer Nature, 2024. 245 p.].

Розподіл емоції радості за класами демонструє накладання значень між «хорошими» та «поганими» позичальниками (рисунок 3.5). Гістограма показує,

що висока інтенсивність радості присутня не лише у текстах надійних позичальників, але й у відповідях частини ризикових клієнтів.

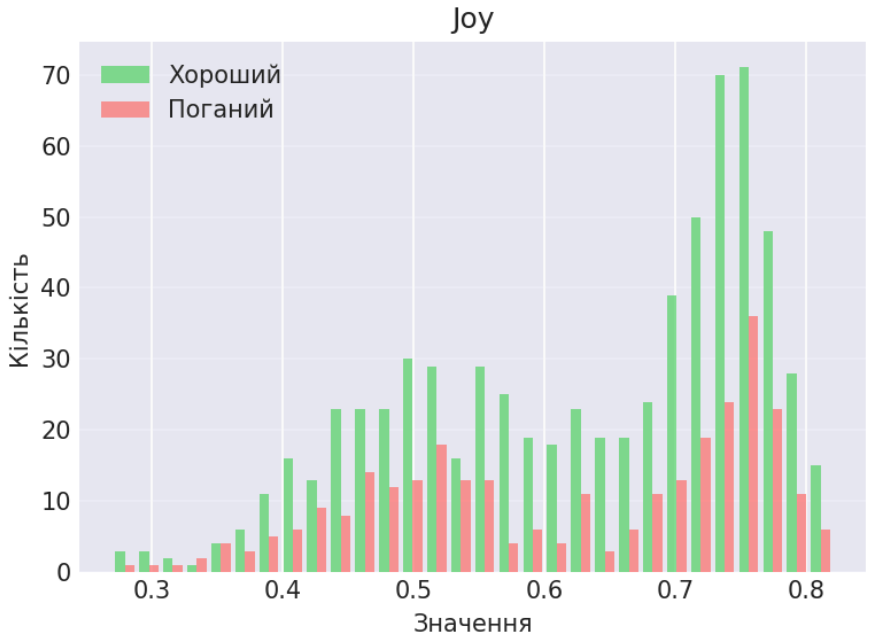


Рис. 3.5 Розподіл емоції Joy за класами позичальників

Накладання розподілів пояснюється тим, що алгоритм генерації текстів параметризується фінансовими атрибутами, а не цільовою змінною. Позичальник з класу «поганий кредит» може мати позитивні фінансові показники за окремими атрибутами (наприклад, високі заощадження при нестабільній зайнятості), що призводить до генерації частково позитивних відповідей. Це накладання є бажаною характеристикою датасету, оскільки воно відображає реальну ситуацію, де психологічний стан позичальника не детермінується однозначно кінцевим результатом кредиту.

Конструювання похідних емоційних ознак виконується через feature engineering з метою створення показників, що відображають не лише інтенсивність окремих емоцій, але й їх взаємні відношення та взаємодію з фінансовими параметрами. Формуються п'ять категорій похідних ознак [10].

Перша категорія містить співвідношення позитивних та негативних емоцій. Обчислюється відношення середнього значення радості до середніх значень

страху та гніву з епсилон-регуляризацією для уникнення ділення на нуль. Формуються ознаки Joy_Fear_ratio, Joy_Anger_ratio та зворотне відношення Fear_Joy_ratio.

Друга категорія представляє різниці між середніми значеннями емоцій. Розраховується показник Positive_Negative_diff як різниця між радістю та середнім арифметичним негативних емоцій (страх, гнів, сум). Окремо обчислюється Joy_Fear_diff, що відображає баланс між оптимізмом та тривожністю позичальника.

Третя категорія описує емоційну волатильність через різницю між максимальним та середнім значеннями кожної емоції. Висока волатильність може свідчити про нестабільність емоційного стану або про наявність специфічних турбот, згаданих лише в одній з відповідей. Формуються шість ознак типу Joy_volatility, Fear_volatility для кожної емоції.

Четверта категорія створює взаємодії емоційних та фінансових ознак через множення. Обчислюються добутки середніх значень страху та радості на суму кредиту, нормовану діленням на 10000 для зменшення масштабу числових значень. Ознаки Fear_Credit_interaction та Joy_Credit_interaction дозволяють врахувати, що одна й та сама інтенсивність страху має різне значення для невеликих споживчих кредитів та великих позик. Аналогічно формуються взаємодії емоцій з тривалістю кредитування: Fear_Duration_interaction та Joy_Duration_interaction.

П'ята категорія формує інтегральні індекси. Обчислюється загальна емоційна інтенсивність через сумування середніх значень всіх шести емоцій, що дає ознаку Total_emotion_intensity. Формується індекс домінуючої емоції як максимальне значення серед усіх середніх емоцій: Dominant_emotion_strength. Конструюється індекс емоційного ризику Risk_emotion_index як відношення суми негативних емоцій (страх та гнів) до радості з епсилон-регуляризацією [16].

Результатом feature engineering є розширення набору емоційних ознак з 18 первинних до 36 комплексних показників. Загальний датасет після всіх етапів обробки містить 20 фінансових атрибутів, 18 первинних емоційних ознак та 18

сконструйованих емоційних ознак, що формує простір 56 потенційних предикторів для моделі класифікації [10].

Навчання моделей кредитного скорингу виконується з урахуванням асиметричної функції втрат, що відображає різну вартість помилок класифікації у контексті кредитування. Видача кредиту позичальнику, який не поверне його (false negative), призводить до прямих фінансових втрат у розмірі суми кредиту, процентів та адміністративних витрат на стягнення боргу. Відмова у кредиті надійному позичальнику (false positive) означає втрату потенційної процентної маржі, що становить лише частину суми кредиту [22].

Згідно з документацією German Credit Dataset, співвідношення вартостей помилок становить 5:1, де вартість видачі кредиту «поганому» позичальнику дорівнює 5, а вартість відмови «хорошому» позичальнику дорівнює 1 [20]. Це співвідношення відображає типову бізнес-логіку фінансових установ та використовується для калібрування моделей через механізм зважування екземплярів.

Зважування екземплярів реалізується через присвоєння кожному спостереженню тренувальної вибірки ваги, пропорційної вартості помилки на цьому класі. Екземплярам класу «поганий кредит» присвоюється вага 5, екземплярам класу «хороший кредит» – вага 1. Під час навчання алгоритм XGBoost мінімізує зважену функцію втрат, де помилки на екземплярах з більшою вагою сильніше впливають на оновлення параметрів моделі. Зважування змушує алгоритм приділяти більшу увагу коректній класифікації потенційних дефолтів, оскільки помилки на цих екземплярах сильніше впливають на величину функції втрат.

Після навчання моделі виконується калібрування порогу класифікації для подальшої мінімізації очікуваної вартості помилок. Стандартний поріг 0,5 для бінарної класифікації не є оптимальним за умови асиметричних втрат. Перебираються порогові значення від 0,1 до 0,9 з кроком 0,05, для кожного обчислюється загальна вартість помилок на тренувальній вибірці за формулою: вартість дорівнює кількості false negative, помножена на 5, плюс кількість false

positive, помножена на 1. Оптимальний поріг визначається як значення, що мінімізує цю функцію вартості на тренувальній вибірці.

Калібрування порогу дозволяє врахувати асиметрію втрат не лише під час навчання через зважування, але й на етапі інференсу через явне налаштування точки прийняття рішення. Модель може виробляти ймовірність дефолту на рівні 0,3, але якщо оптимальний поріг калібровано на значення 0,25, позичальник буде класифікований як ризиковий, що відображає консервативну стратегію кредитування за умови високої вартості дефолту.

Порівняльний аналіз виконується через навчання двох моделей на ідентичних тренувальних даних з однаковими параметрами зважування та калібрування. Базова модель використовує лише 20 фінансових атрибутів German Credit Dataset. Гібридна модель використовує 56 ознак: 20 фінансових плюс 36 емоційних (18 первинних та 18 похідних). Обидві моделі навчаються на 70% датасету (700 екземплярів), стратифікованих за цільовою змінною для збереження пропорції класів. Оцінка ефективності виконується на незалежній тестовій вибірці, що становить 30% датасету (300 екземплярів) [19].

Гіперпараметри моделей визначаються окремо для врахування різної розмірності простору ознак. Базова модель використовує максимальну глибину дерева 6, швидкість навчання 0,05, кількість ітерацій бустингу 150, субсемплювання 0,8, вибіркове використання ознак 0,8, L1-регуляризацію 0,1, L2-регуляризацію 1,0. Гібридна модель застосовує більш складну конфігурацію: максимальна глибина 10, швидкість навчання 0,03, кількість ітерацій 300, субсемплювання 0,85, вибіркове використання ознак 0,7, L1-регуляризацію 0,3, L2-регуляризацію 2,0, мінімальну суму ваг у листових вузлах 3.

Збільшення складності гібридної моделі обумовлене більшою розмірністю простору ознак та необхідністю виявлення нелінійних взаємодій між фінансовими та емоційними показниками. Збільшені коефіцієнти регуляризації запобігають перенавчанню на розширеному наборі ознак. Додатковий параметр мінімальної суми ваг у листових вузлах вимагає достатньої кількості екземплярів для створення вузла, що зменшує ймовірність апроксимації шуму у даних.

Оцінка ефективності моделей виконується через обчислення стандартних метрик класифікації (accuracy, precision, recall, F1-score, ROC-AUC) та специфічної для задачі метрики загальної вартості помилок. Порівняння базової та гібридної моделей дозволяє оцінити внесок емоційних ознак у покращення якості класифікації та зменшення очікуваних фінансових втрат кредитної установи.

Архітектура інтегрованої системи забезпечує модульність та можливість незалежної модифікації окремих компонентів. Генератор текстів може бути замінений на модуль збору реальних відповідей позичальників через веб-інтерфейс або мобільний додаток. Модель екстракції емоцій може бути оновлена на новішу версію трансформерної архітектури без зміни решти конвеєра. Класифікатор XGBoost може бути замінений на альтернативний алгоритм, що підтримує зважування екземплярів та калібрування порогу.

3.2. Розробка програмного комплексу для демонстрації методів машинного навчання

Практична реалізація інтегрованої системи оцінки кредитного ризику виконана мовою програмування Python з використанням бібліотек для обробки даних, машинного навчання та обробки природної мови. Програмний комплекс складається з трьох основних модулів: генератор синтетичних текстових відповідей, екстрактор емоційних ознак, предиктор з cost-sensitive навчанням. Кожен модуль реалізовано як окремий клас з чітко визначеними методами та інтерфейсами взаємодії [32].

Архітектура програмного комплексу базується на об'єктно-орієнтованому підході, де кожен клас інкапсулює специфічну функціональність та надає публічні методи для обробки даних. Використання класів дозволяє забезпечити модульність системи, спрощує тестування окремих компонентів та полегшує подальше розширення функціональності [33].

Модуль генерації синтетичних текстових відповідей реалізовано через клас SyntheticResponseGenerator, що містить систему шаблонів та логіку параметризації

текстів на основі фінансових профілів позичальників. Ініціалізація класу передбачає встановлення seed для генераторів випадкових чисел, що забезпечує детерміністичність результатів та можливість відтворення експериментів (лістинг 3.1).

Лістинг 3.1 – Ініціалізація генератора синтетичних відповідей

```
    class SyntheticResponseGenerator:
        def __init__(self, seed=42):
            random.seed(seed)
            np.random.seed(seed)
            self._init_templates()
```

Метод ініціалізації встановлює фіксоване значення seed для модулів random та numpy, що гарантує відтворюваність результатів генерації при повторних запусках програми [34]. Виклик методу _init_templates() ініціалізує словники шаблонів для кожного з чотирьох запитань опитувальника.

Структура шаблонів організована як вкладені словники, де ключі першого рівня відповідають категоріям фінансових профілів, а значення представляють списки текстових шаблонів з емоційними маркерами. Для першого запитання про фінансову ситуацію визначаються чотири категорії профілів на основі комбінацій рівнів заощаджень та балансу поточного рахунку (лістинг 3.2).

Лістинг 3.2 – Структура шаблонів для першого запитання

```
    self.q1_templates = {
        'low_savings_low_checking': [
            "Зараз фінансово дуже складно. Сумно що немає
заощаджень...",
            "Фінансова ситуація напружена. Шкода що на рахунку
мінімум...",
        ],
        'high_savings_high_checking': [
            "Радий що фінансова ситуація стабільна...",
            "Дуже добре фінансово. Радий що маю хороші
заощадження...",
        ]
    }
```

Кожна категорія містить набір з трьох альтернативних текстових варіантів, що дозволяє уникнути повної ідентичності відповідей для позичальників з однаковим профілем [35]. Тексти для профілів з низькими показниками містять маркери емоцій «сум» та «страх», тоді як тексти для позитивних профілів включають маркери «радість».

Основний метод генерації відповідей generate_responses приймає на вхід фінансові дані одного позичальника у форматі словника та повертає чотири текстові відповіді. Метод виконує послідовний аналіз атрибутів позичальника для визначення категорії профілю для кожного запитання (лістинг 3.3).

Лістинг 3.3 – Визначення категорії профілю для першого запитання

```
    def generate_responses(self, client_data):
        savings_status = client_data['savings_status']
        checking_status = client_data['checking_status']

        if 'no known savings' in savings_status or '<100' in
savings_status:
            savings_level = 'low'
        else:
            savings_level = 'high'

        if 'no checking' in checking_status or '<0' in
checking_status:
            checking_level = 'low'
        else:
            checking_level = 'high'

        q1_key =
f"{savings_level}_savings_{checking_level}_checking"
        q1 = random.choice(self.q1_templates[q1_key])
```

Категоризація виконується через аналіз текстових значень атрибутів German Credit Dataset з використанням оператора in для перевірки наявності підрядків [36]. Вибір конкретного шаблону з категорії здійснюється функцією random.choice, що забезпечує випадковість при збереженні детерміністичності завдяки фіксованому seed.

Для четвертого запитання про фінансові турботи реалізовано систему балів ризику, що розраховується на основі шести фінансових факторів. Кожен фактор аналізується окремо та додає бали до загального показника ризику (лістинг 3.4).

Лістинг 3.4 – Розрахунок балів ризику для четвертого запитання

```
risk_factors = 0

if savings_level == 'low' and credit_amount > 4000:
    risk_factors += 2
elif savings_level == 'low':
    risk_factors += 1

if checking_level == 'low':
    risk_factors += 1

if 'unemployed' in employment or 'unskilled' in employment:
    risk_factors += 2
elif '<1' in employment or '1<=X<4' in employment:
    risk_factors += 1

if risk_factors >= 4:
    q4_key = 'high_concern'
elif risk_factors <= 1:
    q4_key = 'low_concern'
else:
    q4_key = 'medium_concern'
```

Сумарна оцінка ризику визначає категорію турбот через послідовність умовних операторів [36]. Висока оцінка ризику (чотири і більше балів) вказує на нестабільний фінансовий стан та призводить до генерації текстів з маркерами страху та суму.

Обробка датасету виконується методом process_dataset, що завантажує CSV файл з German Credit Dataset, ітерує по кожному рядку для генерації відповідей та зберігає розширений датасет (лістинг 3.5).

Лістинг 3.5 – Обробка повного датасету

```
def process_dataset(self, input_csv, output_csv):
    df = pd.read_csv(input_csv)
```

```
        responses = []
        for idx, row in df.iterrows():
            resp = self.generate_responses(row)
            responses.append(resp)

        responses_df = pd.DataFrame(responses)
        df_combined = pd.concat([df, responses_df], axis=1)
        df_combined.to_csv(output_csv, index=False,
encoding='utf-8')
```

Використання бібліотеки pandas забезпечує обробку табличних даних через об'єкт DataFrame [37]. Метод iterrows повертає ітератор по рядках датафрейму, де кожен рядок представлено як об'єкт Series з доступом до значень через імена колонок. Функція pd.concat виконує горизонтальне об'єднання оригінального датафрейму та нового датафрейму з відповідями вздовж осі колонок.

Модуль екстракції емоційних ознак реалізовано через клас EmotionExtractor, що інкапсулює логіку завантаження попередньо натренованої моделі XLM-RoBERTa, токенізації текстів та отримання емоційних оцінок. Ініціалізація класу передбачає завантаження моделі з локального сховища та переміщення її на доступний обчислювальний пристрій (лістинг 3.6).

Лістинг 3.6 – Ініціалізація екстрактора емоцій

```
    class EmotionExtractor:
        def __init__(self, model_path):
            self.device = torch.device("cuda" if
torch.cuda.is_available() else "cpu")
            self.emotion_columns = ['Joy', 'Fear', 'Anger',
'Sadness', 'Disgust', 'Surprise']
            self._load_model()

        def _load_model(self):
            self.tokenizer =
AutoTokenizer.from_pretrained(self.model_path)
            self.model =
AutoModelForSequenceClassification.from_pretrained(self.model_path)
            self.model.to(self.device)
            self.model.eval()
```

Бібліотека PyTorch надає абстракцію над різними обчислювальними пристроями через об'єкт torch.device [38]. Функція torch.cuda.is_available перевіряє наявність GPU з підтримкою CUDA та повертає логічне значення. Метод

model.eval переводить модель у режим інференсу, що вимикає dropout та batch normalization шари для забезпечення детерміністичних передбачень [39].

Бібліотека transformers від Hugging Face забезпечує уніфікований інтерфейс для завантаження попередньо натренованих моделей через клас AutoModelForSequenceClassification [40]. Метод from_pretrained автоматично визначає архітектуру моделі з конфігураційного файлу та завантажує ваги з вказаного шляху.

Основний метод екстракції емоцій extract_emotions приймає текстовий рядок, виконує токенізацію, передає дані через модель та перетворює логіти у ймовірності (лістинг 3.7).

Лістинг 3.7 – Екстракція емоційних оцінок з тексту

```
    def extract_emotions(self, text):
        inputs = self.tokenizer(
            text,
            return_tensors="pt",
            truncation=True,
            max_length=512,
            padding=True
        )
        inputs = {k: v.to(self.device) for k, v in inputs.items()}

        with torch.no_grad():
            outputs = self.model(**inputs)
            logits = outputs.logits
            probs = torch.sigmoid(logits).cpu().numpy()[0]
            probs = probs[:6]

        emotion_scores = {
            emotion: float(prob)
            for emotion, prob in zip(self.emotion_columns, probs)
        }
        return emotion_scores
```

Токенізатор перетворює текст у числові ідентифікатори токенів та додає спеціальні токени для позначення початку та кінця послідовності [40]. Параметр return_tensors="pt" вказує на повернення результату у форматі PyTorch тензорів. Параметр truncation=True обрізає послідовності, довші за max_length, тоді як padding=True доповнює коротші послідовності до фіксованої довжини.

Контекстний менеджер torch.no_grad вимикає обчислення градієнтів під час інференсу, що зменшує використання пам'яті та прискорює обчислення [38]. Метод model(**inputs) виконує прямий прохід через мережу та повертає об'єкт з логітами. Функція torch.sigmoid застосовує сигмоїдну активацію для перетворення логітів у ймовірності в діапазоні від 0 до 1.

Обробка чотирьох відповідей одного позичальника виконується методом process_client_responses, що викликає extract_emotions для кожної відповіді окремо та обчислює агреговані статистики (лістинг 3.8).

Лістинг 3.8 – Агрегація емоцій з чотирьох відповідей

```
    def process_client_responses(self, response_1, response_2,
response_3, response_4):
        emotions_separate = []
        for resp in [response_1, response_2, response_3,
response_4]:
            emotions_separate.append(self.extract_emotions(resp))

        max_emotions = {}
        avg_emotions = {}
        for emotion in self.emotion_columns:
            scores = [e[emotion] for e in emotions_separate]
            max_emotions[f'{emotion}_max'] = max(scores)
            avg_emotions[f'{emotion}_avg'] = np.mean(scores)

        result = {}
        result.update(max_emotions)
        result.update(avg_emotions)
        return result
```

List comprehension витягує значення конкретної емоції з кожної з чотирьох відповідей та формує список чисел [36]. Вбудована функція max знаходить максимальне значення, тоді як np.mean з бібліотеки NumPy обчислює арифметичне середнє [41]. Метод dict.update додає пари ключ-значення до результуючого словника.

Модуль cost-sensitive предиктора реалізовано через клас CostSensitivePredictor, що інкапсулює логіку підготовки ознак, навчання моделей з асиметричною функцією втрат та калібрування порогу класифікації. Ініціалізація

класу встановлює співвідношення вартостей помилок та визначає списки фінансових і емоційних ознак (лістинг 3.9).

Лістинг 3.9 – Ініціалізація cost-sensitive предиктора

```
    class CostSensitivePredictor:
        def __init__(self, fn_cost=5, fp_cost=1):
            self.fn_cost = fn_cost
            self.fp_cost = fp_cost

            self.financial_features = [
                'checking_status', 'duration', 'credit_history',
'purpose',
                'credit_amount', 'savings_status', 'employment',
                # ... решта фінансових ознак]
            self.emotion_features = [
                'emotion_Joy', 'emotion_Fear', 'emotion_Anger',
                'emotion_Sadness', 'emotion_Disgust',
'emotion_Surprise',
                'Joy_max', 'Fear_max', # ... решта емоційних ознак]
```

Списки ознак зберігаються як атрибути класу для забезпечення консистентності між різними методами та спрощення модифікації набору ознак [33]. Константи вартостей помилок передаються як параметри конструктора, що дозволяє експериментувати з різними співвідношеннями без зміни коду класу.

Підготовка ознак включає перетворення категоріальних змінних у числовий формат через label encoding. Метод prepare_features ітерує по категоріальних колонках та застосовує LabelEncoder з бібліотеки scikit-learn (лістинг 3.10).

Лістинг 3.10 – Кодування категоріальних ознак

```
    def prepare_features(self, df, fit_encoders=True):
        df_processed = df.copy()
        categorical_features = [
            f for f in self.financial_features
            if f not in self.numeric_features]
        for col in categorical_features:
            if col in df_processed.columns:
                if fit_encoders:
```

```
                    le = LabelEncoder()
                    df_processed[col] =
le.fit_transform(df_processed[col].astype(str))
                    self.label_encoders[col] = le
                else:
                    le = self.label_encoders[col]
                    df_processed[col] =
le.transform(df_processed[col].astype(str))

        return df_processed
```

List comprehension з умовою if f not in self.numeric_features фільтрує список фінансових ознак для отримання лише категоріальних змінних [36]. Клас LabelEncoder відображає кожне унікальне значення категоріальної змінної на цілочисельний індекс від 0 до n-1, де n – кількість унікальних значень [42]. Параметр fit_encoders визначає, чи потрібно навчати нові енкодери (для тренувальних даних) або використовувати збережені (для тестових даних).

Feature engineering для емоційних ознак виконується методом create_emotion_features, що обчислює співвідношення, різниці, волатильність та взаємодії (лістинг 3.11).

Лістинг 3.11 – Конструювання похідних емоційних ознак

```
    def create_emotion_features(self, df):
        df_new = df.copy()
        epsilon = 1e-6

        df_new['Joy_Fear_ratio'] = df_new['Joy_avg'] /
(df_new['Fear_avg'] + epsilon)
        df_new['Positive_Negative_diff'] = df_new['Joy_avg'] - (
            df_new['Fear_avg'] + df_new['Anger_avg'] +
df_new['Sadness_avg']
        ) / 3

        for emotion in ['Joy', 'Fear', 'Anger', 'Sadness',
'Disgust', 'Surprise']:
            df_new[f'{emotion}_volatility'] =
df_new[f'{emotion}_max'] - df_new[f'{emotion}_avg']

        df_new['Fear_Credit_interaction'] = df_new['Fear_avg'] *
df_new['credit_amount'] / 10000
        df_new['Risk_emotion_index'] = (df_new['Fear_avg'] +
df_new['Anger_avg']) / (df_new['Joy_avg'] + epsilon)

        return df_new
```

Епсилон-регуляризація додається до знаменника для уникнення ділення на нуль у випадках, коли емоція має нульову інтенсивність [41]. Pandas підтримує векторизовані операції над колонками датафрейму, що дозволяє виконувати арифметичні операції без явних циклів [37]. Цикл for створює ознаки волатильності для всіх шести емоцій через динамічне формування імен колонок з використанням f-string.

Навчання моделі з cost-sensitive функцією втрат виконується методом train_model, що створює вектор ваг для екземплярів тренувальної вибірки та передає його алгоритму XGBoost (лістинг 3.12).

Лістинг 3.12 – Навчання моделі з зважуванням екземплярів

```
    def train_model(self, X_train, y_train, X_test, y_test,
features, model_name, custom_params):
        sample_weights = np.ones(len(y_train))
        sample_weights[y_train == 0] = self.fn_cost
        sample_weights[y_train == 1] = self.fp_cost

        model = xgb.XGBClassifier(**custom_params)
        model.fit(X_train[features], y_train,
sample_weight=sample_weights)

        y_pred_proba_train =
model.predict_proba(X_train[features])[:, 1]
        optimal_threshold = self.find_optimal_threshold(y_train,
y_pred_proba_train)

        y_pred_proba_test =
model.predict_proba(X_test[features])[:, 1]
        y_pred_test = (y_pred_proba_test >=
optimal_threshold).astype(int)
```

Створення вектора ваг починається з ініціалізації масиву одиниць через np.ones, після чого булева індексація y_train == 0 використовується для вибору позицій класу «поганий кредит» та присвоєння їм ваги fn_cost [41]. Бібліотека XGBoost підтримує параметр sample_weight у методі fit, що дозволяє вказати індивідуальні ваги для кожного екземпляра.

Метод predict_proba повертає матрицю ймовірностей розмірністю n × 2, де n – кількість екземплярів, перший стовпець містить ймовірності класу 0, другий –

класу 1 [43]. Індексація [:, 1] витягує другий стовпець з ймовірностями позитивного класу. Порівняння з порогом через оператор >= повертає булевий масив, який перетворюється у цілі числа методом astype(int).

Калібрування порогу класифікації виконується методом find_optimal_threshold, що перебирає діапазон можливих значень та обчислює вартість помилок для кожного (лістинг 3.13).

Лістинг 3.13 – Пошук оптимального порогу класифікації

```
    def find_optimal_threshold(self, y_true, y_proba):
        best_threshold = 0.5
        min_cost = float('inf')

        thresholds = np.arange(0.1, 0.9, 0.05)

        for threshold in thresholds:
            y_pred = (y_proba >= threshold).astype(int)
            cost, fn, fp = self.calculate_total_cost(y_true,
y_pred)

            if cost < min_cost:
                min_cost = cost
                best_threshold = threshold

        return best_threshold
```

Функція np.arange генерує послідовність чисел з плаваючою комою від 0,1 до 0,9 з кроком 0,05 [41]. Константа float('inf') представляє нескінченність та використовується для ініціалізації мінімального значення, що гарантує оновлення на першій ітерації циклу [36]. Цикл for ітерує по кандидатах порогів, для кожного обчислює передбачення та вартість помилок.

Метод calculate_total_cost використовує матрицю плутанини з бібліотеки scikit-learn для підрахунку кількості false negative та false positive (лістинг 3.14).

Лістинг 3.14 – Обчислення загальної вартості помилок

```
    def calculate_total_cost(self, y_true, y_pred):
        cm = confusion_matrix(y_true, y_pred)
        tn, fp, fn, tp = cm.ravel()
        total_cost = fn * self.fn_cost + fp * self.fp_cost
        return total_cost, fn, fp
```

Функція confusion_matrix повертає матрицю 2×2, де елементи відповідають кількостям true negative, false positive, false negative, true positive [42]. Метод ravel перетворює двовимірну матрицю у одновимірний масив та розпаковується у чотири змінні. Загальна вартість обчислюється як зважена сума помилок згідно з заданими коефіцієнтами вартості.

Порівняльний аналіз базової та гібридної моделей виконується методом run_comparison, що послідовно навчає обидві моделі на однакових даних з різними наборами ознак (лістинг 3.15).

Лістинг 3.15 – Порівняння базової та гібридної моделей

```
    def run_comparison(self, data_path):
        df = self.load_data(data_path)
        df_processed = self.prepare_features(df, fit_encoders=True)
        df_processed = self.create_emotion_features(df_processed)

        X = df_processed.drop('target', axis=1)
        y = df_processed['target']

        X_train, X_test, y_train, y_test = train_test_split(
            X, y, test_size=0.3, random_state=42, stratify=y
        )

        baseline_results = self.train_model(
            X_train, y_train, X_test, y_test,
            self.financial_features,
            "BASELINE",
            custom_params=baseline_params
        )

        all_features = self.financial_features +
self.emotion_features + engineered_features
        hybrid_results = self.train_model(
            X_train, y_train, X_test, y_test,
            all_features,
            "HYBRID",
            custom_params=hybrid_params
        )

        return baseline_results, hybrid_results
```

Функція train_test_split з бібліотеки scikit-learn виконує стратифіковане розділення датасету на тренувальну та тестову вибірки [54]. Параметр

test_size=0.3 вказує, що 30% даних буде виділено для тестування. Параметр random_state=42 забезпечує відтворюваність розділення. Параметр stratify=y гарантує збереження пропорції класів у обох вибірках.

Конкатенація списків ознак через оператор + формує повний набір з 56 предикторів для гібридної моделі [36]. Метод train_model викликається двічі з різними наборами ознак, що дозволяє безпосередньо порівняти внесок емоційних ознак у якість класифікації.

Програмний комплекс забезпечує повний цикл обробки даних від генерації синтетичних текстів до навчання та оцінки моделей. Модульна архітектура дозволяє використовувати окремі компоненти незалежно або замінювати їх альтернативними реалізаціями без модифікації решти системи. Використання стандартних бібліотек Python для машинного навчання та обробки даних забезпечує сумісність з екосистемою інструментів data science та спрощує інтеграцію з іншими системами [44].

3.3. Визначення наукової новизни та практичної значущості результатів дослідження

Експериментальна перевірка розробленої інтегрованої системи виконувалася на датасеті German Credit з доданими синтетичними україномовними текстовими відповідями та екстрагованими емоційними ознаками. Порівняльний аналіз базової моделі (20 фінансових ознак) та гібридної моделі (56 ознак: 20 фінансових плюс 36 емоційних) дозволяє оцінити внесок лінгвістичних характеристик у якість класифікації та зменшення очікуваних фінансових втрат кредитної установи.

Базова модель, що використовує виключно структуровані фінансові параметри з German Credit Dataset, досягла точності класифікації 0,753 на тестовій вибірці, F1-оцінки 0,844 та показника ROC-AUC 0,760. Гібридна модель, що інтегрує фінансові та емоційні ознаки, продемонструвала точність 0,760, F1-оцінку 0,848 та ROC-AUC 0,795 (рисунок 3.6).

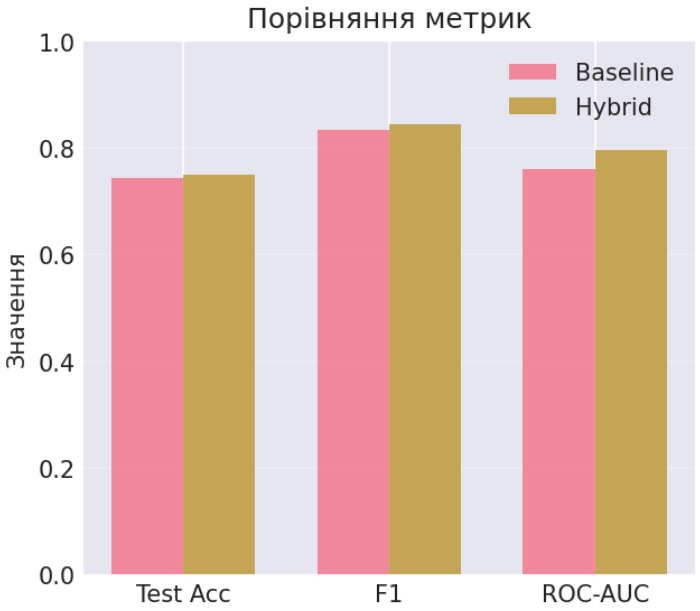


Рис. 3.6 Порівняння метрик ефективності базової та гібридної моделей

Покращення показника ROC-AUC на 3,5 відсоткових пункти (з 0,760 до 0,795) свідчить про підвищення здатності моделі розрізняти класи на всьому діапазоні порогів класифікації. ROC-крива демонструє залежність між true positive rate та false positive rate для різних значень порогу (рисунок 3.7). Гібридна модель демонструє кращу дискримінаційну здатність, що відображається у більшій площі під кривою.

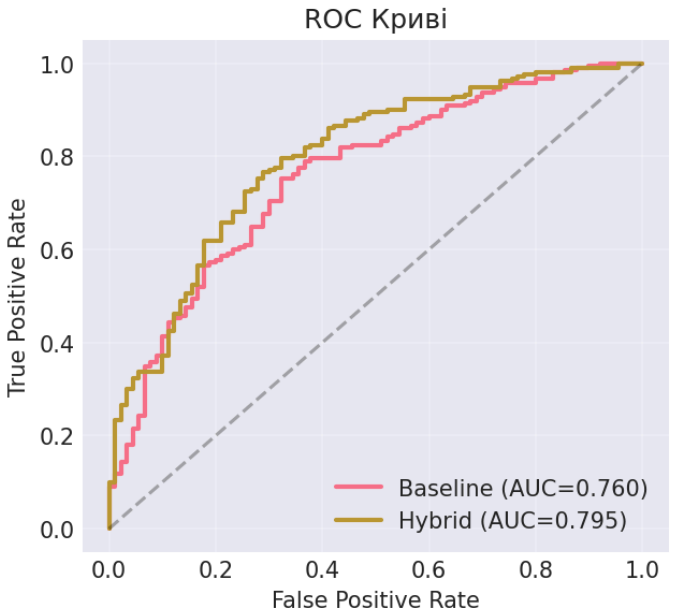


Рис. 3.7 ROC-криві базової та гібридної моделей

Площа під ROC-кривою є агрегованою метрикою, що відображає ймовірність того, що модель присвоїть вищу оцінку ризику випадково обраному позитивному екземпляру порівняно з випадково обраним негативним екземпляром [45]. Значення 0,795 для гібридної моделі вказує на високу якість класифікації, що наближається до показників, типових для комерційних систем кредитного скорингу.

Аналіз структури помилок виявляє відмінності у поведінці моделей відносно різних типів помилкових класифікацій. Базова модель допустила 19 помилок типу false negative (видача кредиту «поганому» позичальнику) та 12 помилок типу false positive (відмова «хорошому» позичальнику). Гібридна модель зменшила кількість false negative до 7, проте збільшила кількість false positive до 14 (рисунок 3.8).

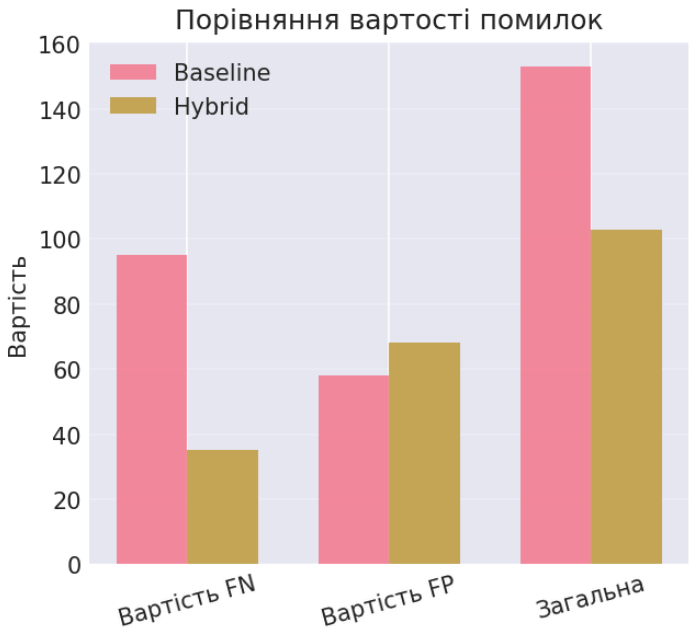


Рис. 3.8 Порівняння вартості помилок базової та гібридної моделей

Зміщення балансу помилок у бік збільшення false positive при одночасному зменшенні false negative відповідає цілям cost-sensitive навчання, де помилки мають асиметричну вартість. Згідно з матрицею вартості German Credit Dataset, видача кредиту позичальнику, який не поверне його, коштує у п'ять разів дорожче, ніж відмова надійному позичальнику [20]. Тому зменшення кількості false negative на 12 екземплярів (з 19 до 7) компенсує збільшення false positive на 2 екземпляри (з 12 до 14) у термінах загальної вартості помилок.

Загальна вартість помилок базової моделі становить 107 умовних одиниць (19 false negative по 5 одиниць плюс 12 false positive по 1 одиниці). Гібридна модель досягла загальної вартості 49 умовних одиниць (7 false negative по 5 одиниць плюс 14 false positive по 1 одиниці). Абсолютне зменшення вартості становить 58 одиниць, що відповідає відносному покращенню на 54,2%. Це свідчить, що інтеграція емоційних ознак дозволяє більш ніж удвічі зменшити очікувані фінансові втрати кредитної установи на тестовій вибірці з 300 позичальників.

Аналіз важливості ознак у гібридній моделі виявляє, що фінансові параметри залишаються домінуючими предикторами кредитного ризику. Топ-15 найважливіших ознак включають 11 фінансових атрибутів (checking_status,

duration, credit_amount, savings_status, credit_history) та 4 емоційні ознаки (Fear_Credit_interaction, Risk_emotion_index, Joy_Fear_ratio, Fear_avg). Це підтверджує, що емоційні характеристики надають додаткову прогностичну інформацію, проте не замінюють традиційні фінансові показники.

Серед емоційних ознак найвищу важливість демонструють взаємодії емоцій з фінансовими параметрами (Fear_Credit_interaction, Fear_Duration_interaction) та інтегральні індекси (Risk_emotion_index). Це свідчить, що абсолютні значення емоцій менш інформативні порівняно з їх контекстуалізацією через фінансові параметри. Страх, виражений у відповідях позичальника, який бере невелику споживчу позику, має інше прогностичне значення порівняно зі страхом при великому інвестиційному кредиті.

Науковою новизною проведеного дослідження є розробка методології інтеграції структурованих фінансових даних та неструктурованих лінгвістичних ознак для задачі кредитного скорингу з урахуванням специфіки української мови. На відміну від існуючих підходів, що використовують англомовні датасети та моделі [20, 27], запропонована система демонструє можливість застосування багатомовних трансформерних архітектур для екстракції психо-емоційних характеристик з україномовних текстів у фінансовому контексті.

Методологічний внесок полягає у формалізації процесу генерації синтетичних текстових даних на основі фінансових профілів через систему параметризованих шаблонів з експліцитними емоційними маркерами. Цей підхід дозволяє створювати контрольовані датасети для перевірки гіпотез про зв'язок між психологічними характеристиками та кредитним ризиком без необхідності збору реальних текстових відповідей позичальників на етапі прототипування системи.

Технічним внеском є адаптація моделі XLM-RoBERTa, попередньо натренованої на EmoBench-UA, для задачі екстракції емоційних ознак у контексті фінансового консультування. Дослідження демонструє, що трансферне навчання від загальної задачі класифікації емоцій до специфічної задачі аналізу фінансових текстів є можливим без додаткового донавчання моделі на доменно-специфічних даних.

Практична значущість результатів дослідження полягає у демонстрації можливості зменшення очікуваних фінансових втрат кредитних установ через інтеграцію якісних характеристик позичальників, виявлених через аналіз їх текстових відповідей. Зменшення загальної вартості помилок на 54,2% на тестовій вибірці вказує на потенціал методології для застосування у реальних системах прийняття рішень про видачу кредитів.

Для мікрофінансових організацій, що працюють з клієнтами, які мають обмежену кредитну історію або відсутність формальних фінансових документів, додавання психо-емоційних характеристик може надати альтернативне джерело інформації для оцінки ризику. Опитувальник з чотирьох відкритих запитань є менш обтяжливим для позичальника порівняно з детальним фінансовим аудитом, що може покращити доступність фінансових послуг для незахищених категорій населення [26].

Розроблений програмний комплекс демонструє можливість повної автоматизації процесу від збору текстових відповідей до прийняття рішення про кредит. Модульна архітектура системи дозволяє інтегрувати окремі компоненти у існуючі інформаційні системи кредитних установ без необхідності повної заміни наявної інфраструктури [46].

Обмеженням проведеного дослідження є використання синтетичних текстових даних замість реальних відповідей позичальників. Алгоритм генерації текстів базується на детерміністичних правилах та шаблонах, що не відображає повної різноманітності та складності природної мови. Реальні відповіді можуть містити граматичні помилки, неоднозначності, іронію, непослідовності, які не представлені у синтетичних даних. Валідація методології на реальних текстових відповідях позичальників є необхідним наступним кроком для підтвердження практичної застосовності системи.

Додатковим обмеженням є розмір датасету German Credit (1000 позичальників), що є відносно невеликим для навчання складних моделей машинного навчання. Сучасні системи кредитного скорингу навчаються на десятках або сотнях тисяч історичних кредитних справ. Масштабування

методології на великі датасети може виявити інші закономірності та потенційно покращити точність класифікації.

Перспективи подальших досліджень включають збір реального датасету україномовних текстових відповідей позичальників мікрофінансових організацій з анотацією фактичних результатів виконання кредитних зобов'язань. Це дозволить перевірити гіпотезу про зв'язок між емоційними характеристиками, виявленими через текстовий аналіз, та реальною ймовірністю дефолту на даних, що не мають обмежень синтетичної генерації.

Результати дослідження демонструють, що інтеграція структурованих фінансових параметрів та неструктурованих лінгвістичних ознак є перспективним напрямком для покращення систем кредитного скорингу. Методологія дозволяє враховувати не лише об'єктивні фінансові показники, але й суб'єктивні психо-емоційні характеристики позичальників, що може підвищити точність оцінки ризику та зменшити фінансові втрати кредитних установ.

ВИСНОВКИ

У кваліфікаційній роботі виконано дослідження методів машинного навчання для аналізу структурованих і неструктурованих даних та розроблено інтегровану модель на прикладі задачі кредитного скорингу.

У першому розділі проведено теоретико-методичний аналіз підходів машинного навчання до роботи з різнорідними даними. Систематизовано методи обробки структурованих даних: лінійні моделі, дерева рішень, ансамблеві методи на основі бустингу та беггінгу. Проаналізовано архітектури для неструктурованих текстових даних, встановлено перехід від bag-of-words до трансформерних моделей типу BERT з механізмом уваги. Обґрунтовано вибір XGBoost для структурованих даних та XLM-RoBERTa для екстракції лінгвістичних ознак з україномовних текстів.

У другому розділі виконано аналітико-дослідницьку апробацію обраних методів. Сформовано базу дослідження через об'єднання German Credit Dataset (1000 позичальників, 20 фінансових атрибутів) із згенерованими україномовними текстовими відповідями на чотири запитання опитувальника. Порівняльний аналіз трьох алгоритмів показав найвищу точність XGBoost – 0,787 та F1-оцінку 0,828, що перевищує Random Forest (0,773 та 0,818) та Logistic Regression (0,753 та 0,803). XLM-RoBERTa досягла точності класифікації емоцій 0,844 на EmoBench-UA, що на 2,8 відсоткових пункти перевищує mBERT. Екстраговано 18 первинних емоційних характеристик з текстів.

У третьому розділі виконано конструктивну розробку та реалізацію інтегрованої системи. Розроблено архітектуру гібридної моделі, що об'єднує 20 фінансових ознак з 36 емоційними (18 первинних та 18 похідних через feature engineering) для навчання XGBoost з cost-sensitive функцією втрат (співвідношення 5:1). Реалізовано програмний комплекс мовою Python з використанням бібліотек pandas, transformers, xgboost, scikit-learn. Експериментальна перевірка на тестовій вибірці 300 позичальників показала: гібридна модель досягла ROC-AUC 0,795 проти 0,760 у базової (покращення на

3,5 відсоткових пункти), зменшила false negative з 19 до 7 (на 63%), що призвело до зниження загальної вартості помилок зі 107 до 49 умовних одиниць (покращення на 54,2%).

Наукова новизна полягає у систематичному порівнянні методів машинного навчання для структурованих та неструктурованих даних у рамках єдиної задачі класифікації. Вперше запропоновано інтегровану модель кредитного скорингу, що поєднує фінансові показники з психо-емоційними характеристиками, виділеними з україномовних текстів через XLM-RoBERTa. Отримало подальший розвиток застосування трансформерних архітектур для формування додаткових ознак через feature engineering.

Практична значущість полягає у розробці системи для мікрофінансових організацій, що дозволяє оцінювати ризик дефолту через комплексний аналіз фінансових параметрів та психо-емоційного стану позичальника. Зменшення очікуваних втрат на 54,2% демонструє потенціал методології для реальних систем. Модульна архітектура дозволяє інтегрувати компоненти у існуючі інформаційні системи кредитних установ.

Основним обмеженням є використання синтетичних текстових даних, згенерованих через детерміністичні шаблони. Перспективи включають збір реального датасету україномовних відповідей позичальників для валідації методології та застосування великих мовних моделей типу GPT для глибшого аналізу текстів.

Результати демонструють, що інтеграція структурованих фінансових параметрів та неструктурованих лінгвістичних ознак є перспективним напрямком для покращення систем кредитного скорингу через врахування об'єктивних показників та суб'єктивних психо-емоційних характеристик позичальників.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Deep Talk. 80% of the world's data is unstructured. Medium, 2021. URL: https://deep-talk.medium.com/80-of-the-worlds-data-is-unstructured-7278e2ba6b73 (дата звернення: 22.10.2025).

2. Zhang X., Zhang W., Wang Y., Jiang Y., Puumala C. B. Integrating structured and unstructured data for predicting emergency severity: an association and predictive study using transformer-based natural language processing models. BMC Medical Informatics and Decision Making, 2024. Vol. 24. Article 793-9. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC11619330/ (дата звернення: 19.10.2025).

3. Silva R. R. V., Viterbo J., Costa L. C., El-Hachem J., Santos B. P. Hybrid methodology for analysis of structured and unstructured data to support decision-making in public security. Data & Knowledge Engineering, 2022. Vol. 141. Article 102054. URL: https://www.sciencedirect.com/science/article/abs/pii/S0169023X22000544 (дата звернення: 27.10.2025).

4. Chiu C. C., Wu C. M., Chien T. N., Kao L. J., Li C., Chu C. M. Integrating Structured and Unstructured EHR Data for Predicting Mortality by Machine Learning and Latent Dirichlet Allocation Method. International Journal of Environmental Research and Public Health, 2023. Vol. 20, No. 5. Article 4340. URL: https://www.mdpi.com/1660-4601/20/5/4340 (дата звернення: 24.10.2025).

5. Amazon Web Services. What is Structured Data? URL: https://aws.amazon.com/what-is/structured-data/ (дата звернення: 20.10.2025).

6. DataCamp. Handling Machine Learning Categorical Data. DataCamp Tutorial, 2023. URL: https://www.datacamp.com/tutorial/categorical-data (дата звернення: 26.10.2025).

7. Google for Developers. Working with categorical data. Machine Learning. URL: https://developers.google.com/machine-learning/crash-course/categorical-data (дата звернення: 29.10.2025).

8. Kelleher J. D. Machine Learning for Tabular Data. Springer Nature, 2024. 245 p.

9. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow. 3rd ed. O'Reilly Media, 2023. 861 p.

10. Kuhn M., Johnson K. Applied Predictive Modeling. 2nd ed. Springer, 2022. 600 p.

11. Han J., Pei J., Tong H. Data Mining: Concepts and Techniques. 4th ed. Morgan Kaufmann, 2023. 744 p.

12. Feurer M. et al. Auto-sklearn 2.0: Hands-free AutoML via Meta-Learning. Journal of Machine Learning Research, 2021. Vol. 23. Article 261.

13. Zhang C., Ma Y. Ensemble Machine Learning: Methods and Applications. 2nd ed. Springer, 2024.

14. Camacho-Collados J., Pilehvar M. T. Embeddings in Natural Language Processing. Springer, 2022.

15. Cambria E., White B. Jumping NLP Curves: A Review of Natural Language Processing Research. IEEE Computational Intelligence Magazine, 2023.

16. Lu Q., Zhang H., Liu Y. et al. Machine learning for human emotion recognition: a comprehensive review. Neural Computing and Applications, 2024. Vol. 36(16).

17. Fine-tuning BERT, DistilBERT, XLM-RoBERTa and Ukr-RoBERTa models for sentiment analysis of ukrainian language reviews. Journal of Artificial Intelligence, 2024. URL: https://jai.in.ua/index.php/en/issues?paper_num=1623 (дата звернення: 28.10.2025).

18. Dai Y., Jayaratne M., Jayatilleke B. Explainable Personality Prediction Using Answers to Open-Ended Interview Questions. ResearchGate, 2022. URL: https://www.researchgate.net/publication/363262538_Explainable_Personality_Prediction_Using_Answers_to_Open-Ended_Interview_Questions (дата звернення: 21.10.2025).

19. Wang Y., Xu Z., Ma K., Chen Y., Liu J. A Comparative Performance Assessment of Ensemble Learning for Credit Scoring. Mathematics, 2020. Vol. 8(10).

Article 1756. URL: https://www.mdpi.com/2227-7390/8/10/1756 (дата звернення: 30.10.2025).

20. UCI Machine Learning Repository. Statlog (German Credit Data) Data Set. URL: https://archive.ics.uci.edu/dataset/144/statlog+german+credit+data (дата звернення: 10.11.2025).

21. German Credit Data Analysis. URL: https://srisai85.github.io/GermanCredit/German.html (дата звернення: 10.11.2025).

22. Abi R. Machine learning for credit scoring and loan default prediction using behavioral and transactional financial data. World Journal of Advanced Research and Reviews. 2025. Vol. 26, No. 03. P. 884-904.

23. Dementieva D., et al. EmoBench: Evaluating Multilingual Emotion Understanding. SemEval-2025 Task 11. 2025.

24. Hugging Face. Ukrainian Emotions Intensity Dataset. URL: https://huggingface.co/datasets/ukr-detect/ukr-emotions-intensity (дата звернення: 10.11.2025).

25. Lu Q., Zhang H., Liu Y., et al. Machine learning for human emotion recognition: a comprehensive review. Neural Computing and Applications. 2024. Vol. 36, No. 16.

26. Character-based lending for micro business development: empirical insights into conceptualizing character. Journal of Small Business & Entrepreneurship. 2019.

27. RiskSeal. What Does Psychometric Mean in Lending? URL: https://riskseal.io/glossary/psychometrics (дата звернення: 10.11.2025).

28. The 5 Cs of Credit: Character, Capacity, Capital, Collateral, Conditions. Corporate Finance Institute. 2024.

29. Exploring the role of microfinance in women's empowerment and entrepreneurial development: a qualitative study. PMC. 2022.

30. Gron A. Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow. 3rd ed. O'Reilly Media, 2023. 861 p.

31. Conneau A., Khandelwal K., Goyal N., et al. Unsupervised Cross-lingual Representation Learning at Scale. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL). 2020. P. 8440–8451. URL: https://aclanthology.org/2020.acl-main.747/ (дата звернення: 11.11.2025).

32. Martin R. C. Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall, 2020. 464 p.

33. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional, 2020. 416 p.

34. Python Software Foundation. random — Generate pseudo-random numbers. Python 3.12 Documentation. URL: https://docs.python.org/3/library/random.html (дата звернення: 15.11.2025).

35. Lutz M. Learning Python. 5th ed. O'Reilly Media, 2023. 1648 p.

36. Van Rossum G., Drake F. L. The Python Language Reference Manual. Network Theory Ltd, 2021. 151 p.

37. McKinney W. Python for Data Analysis. 3rd ed. O'Reilly Media, 2022. 579 p.

38. Paszke A., Gross S., Massa F., et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. Advances in Neural Information Processing Systems 32 (NeurIPS 2019). 2019. P. 8024–8035.

39. PyTorch Documentation. torch.nn — Neural Network Layers. URL: https://pytorch.org/docs/stable/nn.html (дата звернення: 16.11.2025).

40. Wolf T., Debut L., Sanh V., et al. Transformers: State-of-the-Art Natural Language Processing. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. 2020. P. 38–45.

41. Harris C. R., Millman K. J., van der Walt S. J., et al. Array programming with NumPy. Nature. 2020. Vol. 585. P. 357–362. DOI: 10.1038/s41586-020-2649-2.

42. Pedregosa F., Varoquaux G., Gramfort A., et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 2021. Vol. 12. P. 2825–2830.

43. Chen T., Guestrin C. XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016. P. 785–794. DOI: 10.1145/2939672.2939785.

44. VanderPlas J. Python Data Science Handbook. 2nd ed. O'Reilly Media, 2023. 548 p.

45. Fawcett T. An introduction to ROC analysis. Pattern Recognition Letters. 2022. Vol. 27. № 8. P. 861–874. DOI: 10.1016/j.patrec.2005.10.010.

46. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley Professional, 2020. 560 p.

1.

ДОДАТКИ

Додаток А

Повний код системи передбачення

```
"""
Cost-Sensitive Hybrid Predictor
Порівняння базової фінансової моделі з гібридною (фінанси + емоції)
"""
import pandas as pd
import numpy as np
import pickle
import os
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import (
    classification_report,
    f1_score,
    confusion_matrix,
    roc_auc_score,
    roc_curve
)
import xgboost as xgb

plt.style.use('seaborn-v0_8-darkgrid')
sns.set_palette("husl")
plt.rcParams['font.family'] = 'DejaVu Sans'


class CostSensitivePredictor:
    """Cost-sensitive предиктор для кредитного скорингу"""

    def __init__(self, fn_cost=5, fp_cost=1):
        """
        Ініціалізація предиктора
        fn_cost: вартість видачі кредиту поганому клієнту
        fp_cost: вартість відмови хорошому клієнту
        """
        print(f"\n{'='*70}\nCost-Sensitive Predictor | FN: {fn_cost}x, FP:
{fp_cost}x\n{'='*70}")

        self.fn_cost = fn_cost
        self.fp_cost = fp_cost

        self.financial_features = [
            'checking_status', 'duration', 'credit_history', 'purpose',
            'credit_amount', 'savings_status', 'employment',
            'installment_commitment', 'personal_status', 'other_parties',
            'residence_since', 'property_magnitude', 'age',
            'other_payment_plans', 'housing', 'existing_credits', 'job',
            'num_dependents', 'own_telephone', 'foreign_worker'
        ]
```

```
        self.emotion_features = [
            'emotion_Joy', 'emotion_Fear', 'emotion_Anger',
            'emotion_Sadness', 'emotion_Disgust', 'emotion_Surprise',
            'Joy_max', 'Fear_max', 'Anger_max',
            'Sadness_max', 'Disgust_max', 'Surprise_max',
            'Joy_avg', 'Fear_avg', 'Anger_avg',
            'Sadness_avg', 'Disgust_avg', 'Surprise_avg'
        ]

        self.numeric_features = [
            'duration', 'credit_amount', 'installment_commitment',
            'residence_since', 'age', 'existing_credits', 'num_dependents'
        ]

        self.label_encoders = {}

    def load_data(self, filepath):
        """Завантажує датасет"""
        df = pd.read_csv(filepath)
        print(f"Завантажено: {len(df)} записів\n")
        return df

    def prepare_features(self, df, fit_encoders=True):
        """Підготовка ознак"""
        df_processed = df.copy()

        for col in self.numeric_features:
            if col in df_processed.columns:
                df_processed[col] = pd.to_numeric(df_processed[col], errors='coerce')

        categorical_features = [
            f for f in self.financial_features
            if f not in self.numeric_features
        ]

        for col in categorical_features:
            if col in df_processed.columns:
                if fit_encoders:
                    le = LabelEncoder()
                    df_processed[col] =
le.fit_transform(df_processed[col].astype(str))
                    self.label_encoders[col] = le
                else:
                    if col in self.label_encoders:
                        le = self.label_encoders[col]
                        df_processed[col] =
le.transform(df_processed[col].astype(str))

        return df_processed

    def create_emotion_features(self, df):
        """Feature engineering для емоцій"""
        df_new = df.copy()
        emotions = ['Joy', 'Fear', 'Anger', 'Sadness', 'Disgust', 'Surprise']
        epsilon = 1e-6
```

```
        df_new['Joy_Fear_ratio'] = df_new['Joy_avg'] / (df_new['Fear_avg'] + epsilon)
        df_new['Joy_Anger_ratio'] = df_new['Joy_avg'] / (df_new['Anger_avg'] +
epsilon)
        df_new['Fear_Joy_ratio'] = df_new['Fear_avg'] / (df_new['Joy_avg'] + epsilon)

        df_new['Positive_Negative_diff'] = df_new['Joy_avg'] - (df_new['Fear_avg'] +
df_new['Anger_avg'] + df_new['Sadness_avg']) / 3
        df_new['Joy_Fear_diff'] = df_new['Joy_avg'] - df_new['Fear_avg']

        for emotion in emotions:
            df_new[f'{emotion}_volatility'] = df_new[f'{emotion}_max'] -
df_new[f'{emotion}_avg']

        avg_cols = [f'{e}_avg' for e in emotions]
        df_new['Total_emotion_intensity'] = df_new[avg_cols].sum(axis=1)
        df_new['Dominant_emotion_strength'] = df_new[avg_cols].max(axis=1)

        if 'credit_amount' in df_new.columns:
            df_new['Fear_Credit_interaction'] = df_new['Fear_avg'] *
df_new['credit_amount'] / 10000
            df_new['Joy_Credit_interaction'] = df_new['Joy_avg'] *
df_new['credit_amount'] / 10000

        if 'duration' in df_new.columns:
            df_new['Fear_Duration_interaction'] = df_new['Fear_avg'] *
df_new['duration']
            df_new['Joy_Duration_interaction'] = df_new['Joy_avg'] *
df_new['duration']

        df_new['Risk_emotion_index'] = (df_new['Fear_avg'] + df_new['Anger_avg']) /
(df_new['Joy_avg'] + epsilon)

        return df_new

    def calculate_total_cost(self, y_true, y_pred):
        """Розрахунок вартості помилок"""
        cm = confusion_matrix(y_true, y_pred)
        tn, fp, fn, tp = cm.ravel()
        total_cost = fn * self.fn_cost + fp * self.fp_cost
        return total_cost, fn, fp

    def find_optimal_threshold(self, y_true, y_proba):
        """Пошук оптимального threshold"""
        best_threshold = 0.5
        min_cost = float('inf')
        best_fn = 0
        best_fp = 0

        thresholds = np.arange(0.1, 0.9, 0.05)

        results = []
        for threshold in thresholds:
            y_pred = (y_proba >= threshold).astype(int)
            cost, fn, fp = self.calculate_total_cost(y_true, y_pred)
```

```
            results.append({
                'threshold': threshold,
                'cost': cost,
                'fn': fn,
                'fp': fp
            })

            if cost < min_cost:
                min_cost = cost
                best_threshold = threshold
                best_fn = fn
                best_fp = fp

        results_df = pd.DataFrame(results).sort_values('cost')
        print(f"Оптимальний threshold: {best_threshold:.2f} (вартість={min_cost:.0f},
FN={best_fn}, FP={best_fp})")

        return best_threshold

    def train_model(self, X_train, y_train, X_test, y_test, features, model_name,
custom_params=None):
        """Навчання cost-sensitive моделі"""
        print(f"\n{'-'*70}\n{model_name}\n{'-'*70}")

        sample_weights = np.ones(len(y_train))
        sample_weights[y_train == 0] = self.fn_cost
        sample_weights[y_train == 1] = self.fp_cost

        if custom_params is None:
            params = {
                'objective': 'binary:logistic',
                'eval_metric': 'logloss',
                'max_depth': 8,
                'learning_rate': 0.05,
                'n_estimators': 200,
                'subsample': 0.8,
                'colsample_bytree': 0.8,
                'reg_alpha': 0.1,
                'reg_lambda': 1.0,
                'random_state': 42
            }
        else:
            params = custom_params

        model = xgb.XGBClassifier(**params)
        model.fit(X_train[features], y_train, sample_weight=sample_weights)

        y_pred_proba_train = model.predict_proba(X_train[features])[:, 1]
        y_pred_proba_test = model.predict_proba(X_test[features])[:, 1]

        optimal_threshold = self.find_optimal_threshold(y_train, y_pred_proba_train)

        y_pred_test = (y_pred_proba_test >= optimal_threshold).astype(int)
        y_pred_train = (y_pred_proba_train >= optimal_threshold).astype(int)
```

```
        train_acc = (y_pred_train == y_train).mean()
        test_acc = (y_pred_test == y_test).mean()
        f1 = f1_score(y_test, y_pred_test)
        roc_auc = roc_auc_score(y_test, y_pred_proba_test)

        cm = confusion_matrix(y_test, y_pred_test)
        tn, fp, fn, tp = cm.ravel()

        total_cost, _, _ = self.calculate_total_cost(y_test, y_pred_test)

        print(f"Точність: Train={train_acc:.3f}, Test={test_acc:.3f} | F1={f1:.3f},
ROC-AUC={roc_auc:.3f}")
        print(f"Вартість: Загальна={total_cost:.0f}
(FN={fn}x{self.fn_cost}={fn*self.fn_cost},
FP={fp}x{self.fp_cost}={fp*self.fp_cost})")

        feature_importance = pd.DataFrame({
            'feature': features,
            'importance': model.feature_importances_
        }).sort_values('importance', ascending=False)

        return {
            'model': model,
            'threshold': optimal_threshold,
            'train_acc': train_acc,
            'test_acc': test_acc,
            'f1': f1,
            'roc_auc': roc_auc,
            'fn': fn,
            'fp': fp,
            'tn': tn,
            'tp': tp,
            'total_cost': total_cost,
            'y_pred': y_pred_test,
            'y_pred_proba': y_pred_proba_test,
            'y_test': y_test,
            'confusion_matrix': cm,
            'feature_importance': feature_importance
        }

    def visualize_results(self, baseline_results, hybrid_results,
output_dir='./plots'):
        """Створює візуалізації результатів"""
        os.makedirs(output_dir, exist_ok=True)

        fig = plt.figure(figsize=(16, 12))

        plt.subplot(3, 3, 1)
        metrics = ['Test Acc', 'F1', 'ROC-AUC']
        baseline_vals = [baseline_results['test_acc'], baseline_results['f1'],
baseline_results['roc_auc']]
        hybrid_vals = [hybrid_results['test_acc'], hybrid_results['f1'],
hybrid_results['roc_auc']]
        x = np.arange(len(metrics))
```

```
        width = 0.35
        plt.bar(x - width/2, baseline_vals, width, label='Baseline', alpha=0.8)
        plt.bar(x + width/2, hybrid_vals, width, label='Hybrid', alpha=0.8)
        plt.ylabel('Значення')
        plt.title('Порівняння метрик')
        plt.xticks(x, metrics)
        plt.legend()
        plt.ylim(0, 1)
        plt.grid(axis='y', alpha=0.3)

        plt.subplot(3, 3, 2)
        costs = ['Вартість FN', 'Вартість FP', 'Загальна']
        baseline_costs = [baseline_results['fn']*self.fn_cost,
baseline_results['fp']*self.fp_cost, baseline_results['total_cost']]
        hybrid_costs = [hybrid_results['fn']*self.fn_cost,
hybrid_results['fp']*self.fp_cost, hybrid_results['total_cost']]
        x = np.arange(len(costs))
        plt.bar(x - width/2, baseline_costs, width, label='Baseline', alpha=0.8)
        plt.bar(x + width/2, hybrid_costs, width, label='Hybrid', alpha=0.8)
        plt.ylabel('Вартість')
        plt.title('Порівняння вартості помилок')
        plt.xticks(x, costs, rotation=15)
        plt.legend()
        plt.grid(axis='y', alpha=0.3)

        plt.subplot(3, 3, 3)
        errors = ['FN', 'FP']
        baseline_errors = [baseline_results['fn'], baseline_results['fp']]
        hybrid_errors = [hybrid_results['fn'], hybrid_results['fp']]
        x = np.arange(len(errors))
        plt.bar(x - width/2, baseline_errors, width, label='Baseline', alpha=0.8,
color=['red', 'orange'])
        plt.bar(x + width/2, hybrid_errors, width, label='Hybrid', alpha=0.8,
color=['darkred', 'darkorange'])
        plt.ylabel('Кількість')
        plt.title('Кількість помилок')
        plt.xticks(x, errors)
        plt.legend()
        plt.grid(axis='y', alpha=0.3)

        plt.subplot(3, 3, 4)
        cm_baseline = baseline_results['confusion_matrix']
        sns.heatmap(cm_baseline, annot=True, fmt='d', cmap='Blues', cbar=False,
                    xticklabels=['Поганий', 'Хороший'], yticklabels=['Поганий',
'Хороший'])
        plt.title('Confusion Matrix: Baseline')
        plt.ylabel('Фактичний')
        plt.xlabel('Передбачений')

        plt.subplot(3, 3, 5)
        cm_hybrid = hybrid_results['confusion_matrix']
        sns.heatmap(cm_hybrid, annot=True, fmt='d', cmap='Greens', cbar=False,
                    xticklabels=['Поганий', 'Хороший'], yticklabels=['Поганий',
'Хороший'])
        plt.title('Confusion Matrix: Hybrid')
```

```
        plt.ylabel('Фактичний')
        plt.xlabel('Передбачений')

        plt.subplot(3, 3, 6)
        fpr_base, tpr_base, _ = roc_curve(baseline_results['y_test'],
baseline_results['y_pred_proba'])
        fpr_hyb, tpr_hyb, _ = roc_curve(hybrid_results['y_test'],
hybrid_results['y_pred_proba'])
        plt.plot(fpr_base, tpr_base, label=f"Baseline
(AUC={baseline_results['roc_auc']:.3f})", linewidth=2)
        plt.plot(fpr_hyb, tpr_hyb, label=f"Hybrid
(AUC={hybrid_results['roc_auc']:.3f})", linewidth=2)
        plt.plot([0, 1], [0, 1], 'k--', alpha=0.3)
        plt.xlabel('False Positive Rate')
        plt.ylabel('True Positive Rate')
        plt.title('ROC Криві')
        plt.legend()
        plt.grid(alpha=0.3)

        plt.subplot(3, 3, 7)
        top_features = hybrid_results['feature_importance'].head(15)
        plt.barh(range(len(top_features)), top_features['importance'].values,
alpha=0.8)
        plt.yticks(range(len(top_features)), top_features['feature'].values,
fontsize=8)
        plt.xlabel('Важливість')
        plt.title('Топ-15 ознак (Hybrid)')
        plt.gca().invert_yaxis()
        plt.grid(axis='x', alpha=0.3)

        plt.subplot(3, 3, 8)
        all_emotion_features = self.emotion_features + [
            'Joy_Fear_ratio', 'Joy_Anger_ratio', 'Fear_Joy_ratio',
            'Positive_Negative_diff', 'Joy_Fear_diff',
            'Joy_volatility', 'Fear_volatility', 'Anger_volatility',
            'Sadness_volatility', 'Disgust_volatility', 'Surprise_volatility',
            'Total_emotion_intensity', 'Dominant_emotion_strength',
            'Fear_Credit_interaction', 'Joy_Credit_interaction',
            'Fear_Duration_interaction', 'Joy_Duration_interaction',
            'Risk_emotion_index'
        ]
        emotion_importance = hybrid_results['feature_importance'][

hybrid_results['feature_importance']['feature'].isin(all_emotion_features)
        ].head(10)
        plt.barh(range(len(emotion_importance)),
emotion_importance['importance'].values, alpha=0.8, color='coral')
        plt.yticks(range(len(emotion_importance)),
emotion_importance['feature'].values, fontsize=8)
        plt.xlabel('Важливість')
        plt.title('Топ-10 емоційних ознак')
        plt.gca().invert_yaxis()
        plt.grid(axis='x', alpha=0.3)

        plt.subplot(3, 3, 9)
```

```
        summary_text = f"""
        ПІДСУМОК:

        Baseline Модель:
        - Загальна вартість: {baseline_results['total_cost']:.0f}
        - FN: {baseline_results['fn']}, FP: {baseline_results['fp']}
        - ROC-AUC: {baseline_results['roc_auc']:.3f}

        Hybrid Модель:
        - Загальна вартість: {hybrid_results['total_cost']:.0f}
        - FN: {hybrid_results['fn']}, FP: {hybrid_results['fp']}
        - ROC-AUC: {hybrid_results['roc_auc']:.3f}

        Покращення:
        - Вартість: -{baseline_results['total_cost']-hybrid_results['total_cost']:.0f}
({(baseline_results['total_cost']-hybrid_results['total_cost'])/baseline_results['tot
al_cost']*100:.1f}%)
        - FN: -{baseline_results['fn']-hybrid_results['fn']}
        - ROC-AUC: +{hybrid_results['roc_auc']-baseline_results['roc_auc']:.3f}
        """
        plt.text(0.1, 0.5, summary_text, fontsize=10, verticalalignment='center',
family='monospace')
        plt.axis('off')

        plt.tight_layout()
        plt.savefig(f'{output_dir}/model_comparison.png', dpi=150,
bbox_inches='tight')
        print(f"\nГрафіки збережено: {output_dir}/model_comparison.png")
        plt.close()

    def run_comparison(self, data_path):
        """Виконує порівняння baseline та hybrid моделей"""
        df = self.load_data(data_path)
        df_processed = self.prepare_features(df, fit_encoders=True)
        df_processed = self.create_emotion_features(df_processed)

        engineered_emotion_features = [
            'Joy_Fear_ratio', 'Joy_Anger_ratio', 'Fear_Joy_ratio',
            'Positive_Negative_diff', 'Joy_Fear_diff',
            'Joy_volatility', 'Fear_volatility', 'Anger_volatility',
            'Sadness_volatility', 'Disgust_volatility', 'Surprise_volatility',
            'Total_emotion_intensity', 'Dominant_emotion_strength',
            'Fear_Credit_interaction', 'Joy_Credit_interaction',
            'Fear_Duration_interaction', 'Joy_Duration_interaction',
            'Risk_emotion_index'
        ]

        X = df_processed.drop('target', axis=1)
        y = df_processed['target']

        X_train, X_test, y_train, y_test = train_test_split(
            X, y, test_size=0.3, random_state=42, stratify=y
        )
```

```
        print(f"Train: {len(X_train)} ({(y_train==0).sum()} Поганих,
{(y_train==1).sum()} Хороших)")
        print(f"Test:  {len(X_test)} ({(y_test==0).sum()} Поганих, {(y_test==1).sum()}
Хороших)")

        baseline_params = {
            'objective': 'binary:logistic',
            'eval_metric': 'logloss',
            'max_depth': 6,
            'learning_rate': 0.05,
            'n_estimators': 150,
            'subsample': 0.8,
            'colsample_bytree': 0.8,
            'reg_alpha': 0.1,
            'reg_lambda': 1.0,
            'random_state': 42
        }

        baseline_results = self.train_model(
            X_train, y_train, X_test, y_test,
            self.financial_features,
            "BASELINE (фінансові ознаки)",
            custom_params=baseline_params
        )

        hybrid_params = {
            'objective': 'binary:logistic',
            'eval_metric': 'logloss',
            'max_depth': 10,
            'learning_rate': 0.03,
            'n_estimators': 300,
            'subsample': 0.85,
            'colsample_bytree': 0.7,
            'reg_alpha': 0.3,
            'reg_lambda': 2.0,
            'min_child_weight': 3,
            'random_state': 42
        }

        all_features = self.financial_features + self.emotion_features +
engineered_emotion_features
        hybrid_results = self.train_model(
            X_train, y_train, X_test, y_test,
            all_features,
            "HYBRID (фінансові + емоційні ознаки)",
            custom_params=hybrid_params
        )

        print(f"\n{'='*70}\nПОРІВНЯННЯ\n{'='*70}")
        improvement = baseline_results['total_cost'] - hybrid_results['total_cost']
        improvement_pct = (improvement / baseline_results['total_cost']) * 100
        print(f"Зменшення вартості: {improvement:.0f} ({improvement_pct:+.1f}%)")
        print(f"Зменшення FN: {baseline_results['fn'] - hybrid_results['fn']}")

        self.visualize_results(baseline_results, hybrid_results)
```

```
        os.makedirs("./models", exist_ok=True)
        with open("./models/baseline_cost_sensitive.pkl", 'wb') as f:
            pickle.dump(baseline_results, f)
        with open("./models/hybrid_cost_sensitive.pkl", 'wb') as f:
            pickle.dump(hybrid_results, f)
        print(f"\nМоделі збережено у ./models/")

        return baseline_results, hybrid_results


def main():
    print("Cost-Sensitive Credit Scoring")
    print("Порівняння baseline та hybrid моделей\n")

    data_path = "./data/german_credit_with_emotions.csv"
    if not os.path.exists(data_path):
        print(f"ПОМИЛКА: Файл не знайдено: {data_path}")
        return

    predictor = CostSensitivePredictor(fn_cost=5, fp_cost=1)

    try:
        baseline_results, hybrid_results = predictor.run_comparison(data_path)
        print(f"\n{'='*70}\nЗАВЕРШЕНО\n{'='*70}")
    except Exception as e:
        print(f"\nПомилка: {e}")
        import traceback
        traceback.print_exc()


if __name__ == "__main__":
    main()
```

Додаток Б

Вигляд GUI системи

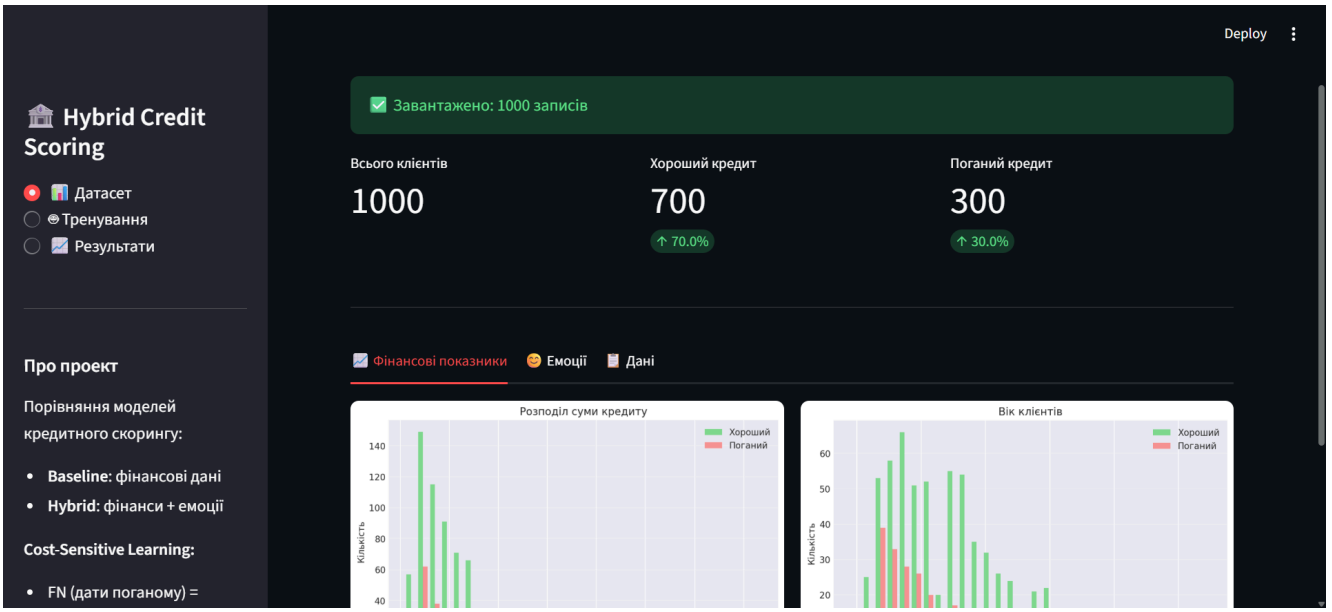


Рис. Б.1 Візуалізація датасету

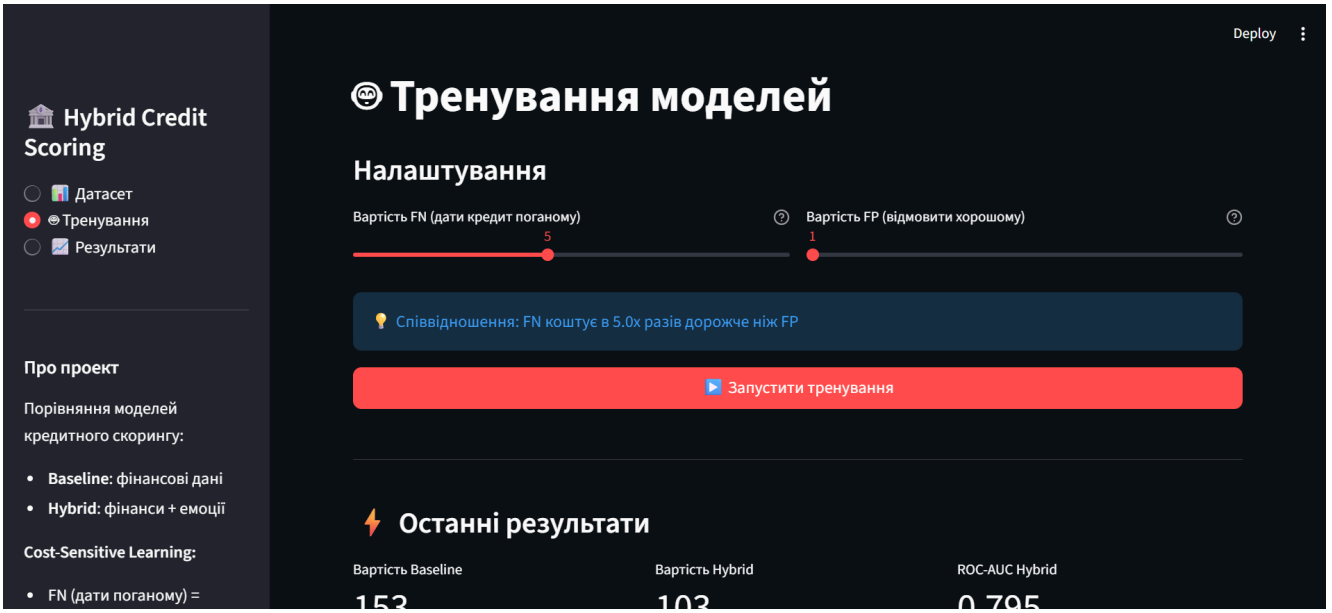


Рис. Б.2 Сторінка тренування моделей

Рис. Б.3 Сторінка результатів

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

Кваліфікаційна робота освітнього ступеня магістр

на тему:

«Методи машинного навчання для аналізу структурованих і неструктурованих даних»

Виконав: ст. гр. САДМ-61

Івахненко Владислав Анатолійович

Керівник: Шушура Олексій Миколайович

м. Київ, 2025

АКТУАЛЬНІСТЬ ТЕМИ

- Актуальність теми обумовлена стрімким зростанням обсягів даних. За прогнозами IDC, до 2025 року неструктуровані дані складатимуть 80% усіх світових даних.
- З них використовується для аналізу лише від половини до одного відсотка цих даних
- Останні дослідження у медичній та інших галузях демонструють, що інтеграція різнорідних даних дає значні переваги

МЕТА РОБОТИ, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **ОБ'ЄКТ**: процеси аналізу структурованих та неструктурованих даних методами машинного навчання.
- **ПРЕДМЕТ**: методи машинного навчання для інтеграції структурованих даних і текстової інформації в задачах класифікації.
- **МЕТА**: дослідження методів машинного навчання для аналізу структурованих і неструктурованих даних та розробка. інтегрованої моделі на прикладі кредитного скорингу

ЗАВДАННЯ ДОСЛІДЖЕННЯ

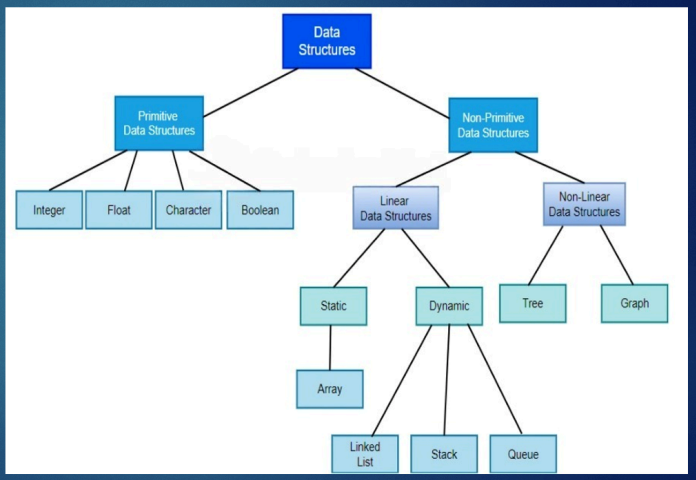
Для досягнення мети сформульовано наступні **завдання**:

1. Проаналізувати методи обробки структурованих даних.
2. Розглянути архітектури машинного навчання для неструктурованих даних.
3. Обрати методи для інтегрованого аналізу різнорідних даних.
4. Сформувати базу дослідження на основі синтетичних даних.
5. Порівняти ефективність методів машинного навчання для структурованих даних.
6. Апробувати методи обробки природної мови для виділення лінгвістичних ознак.
7. Оцінити вплив текстових ознак на точність прогнозування.
8. Розробити інтегровану модель, що поєднує структуровані та неструктуровані дані.
9. Реалізувати програмний комплекс.
10. Проаналізувати отримані результати дослідження.

ОГЛЯД ОБРОБКИ СТРУКТУРОВАНИХ ДАНИХ

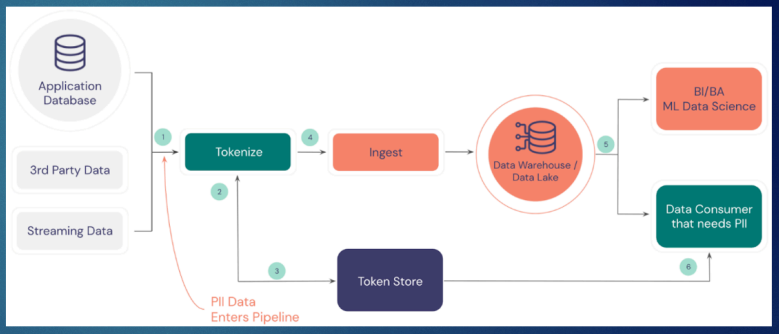
Структуровані дані – це дані у табличному форматі з фіксованими полями, де всі записи мають однакові атрибути. До структур даних належать примітивні типи «Integer», «Float», «Character», «Boolean» та непримітивні структури, зокрема лінійні статичні масиви, лінійні динамічні списки, стеки й черги, а також нелінійні дерева та графи.

У машинному навчанні ознаки поділяються на числові, категоріальні та бінарні. Числові можуть бути безперервними або дискретними. Категоріальні поділяються на номінальні без порядку та порядкові з визначеною ієрархією. Бінарні представляються у вигляді двох значень, наприклад 0/1.



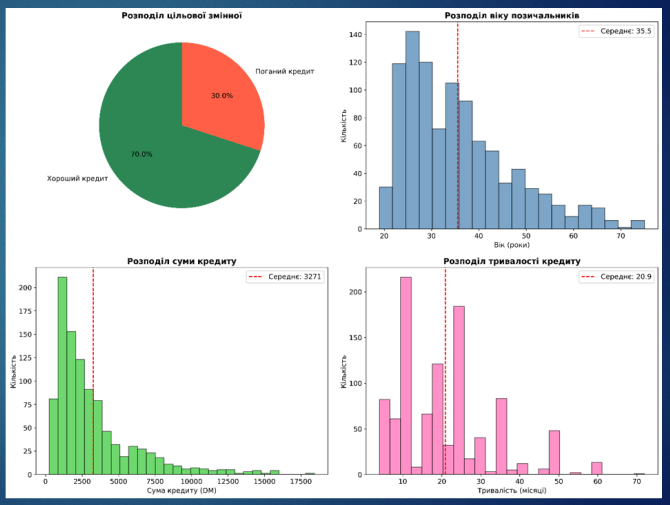
ОГЛЯД ОБРОБКИ НЕСТРУКТУРОВАНИХ ДАНИХ

Неструктуровані дані не мають фіксованої схеми та не організовані у вигляді таблиць. До них належать документи, листи, повідомлення, відгуки, статті та інші форми природної мови. Такі дані містять контекстні та лексичні залежності, полісемію, синонімію й морфологічну варіативність, що потребує попередньої мовної обробки. Підготовка включає очищення, токенізацію, лематизацію, стемінг, нормалізацію та векторизацію. Після токенізації дані надходять до сховищ або використовуються у конвеєрі обробки. У нейромережевих моделях застосовуються методи підслівної токенізації, зокрема BPE, WordPiece та SentencePiece.



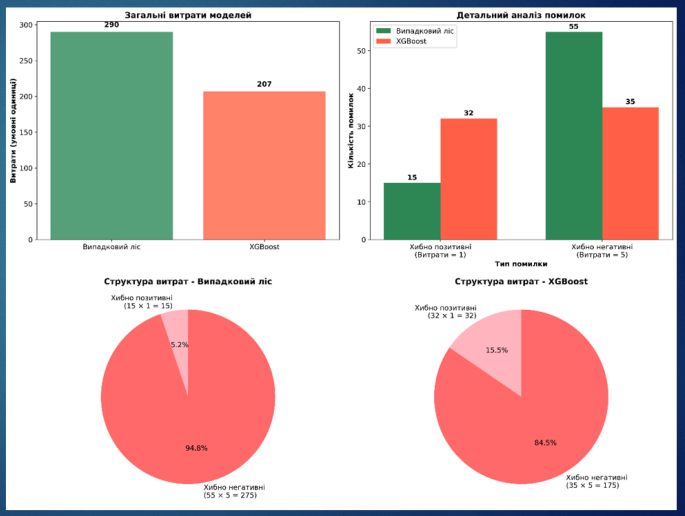
ХАРАКТЕРИСТИКИ СТРУКТУРОВАНИХ ФІНАНСОВИХ ДАНИХ

- Використано датасет «German Credit»,
- 1000 записів
- Цільова змінна: 70 % «добрих» та 30 % «поганих» кредитів
- Середній вік – 35.5 року
- Середня сума кредиту – 3271 DM Середня тривалість кредиту – 20.9 місяця



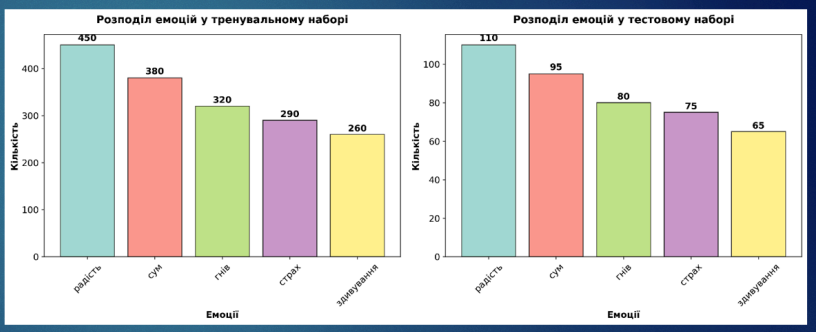
АНАЛІЗ ПОМИЛОК ТА ВАРТОСТІ РІШЕНЬ МОДЕЛЕЙ

- Використано асиметричну матрицю вартостей з документації датасета
- «False negative» коштує у 5 разів більше, ніж «false positive»
- Загальні витрати:
  - Random Forest – 290
  - XGBoost – 207
- XGBoost знижує витрати на 28,6 %
- Основна частина витрат у обох моделей формується за рахунок помилок «false negative»



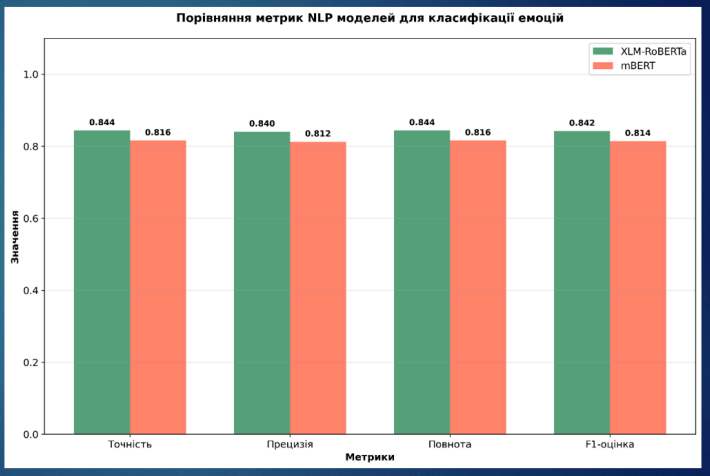
ОБРОБКА НЕСТРУКТУРОВАНИХ ТЕКСТОВИХ ДАНИХ

- Використано асиметричну матрицю вартостей з документації датасета
- Використано трансформерні моделі «XLM-RoBERTa» та «mBERT»
- Обидві моделі працюють з багатомовними корпусами
- Датасет: «EmoBench-UA», 2125 текстів (5 емоцій)
- Токенізація та підготовка текстів до 512 токенів



РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТАЛЬНОЇ ПЕРЕВІРКИ МОДЕЛЕЙ NLP

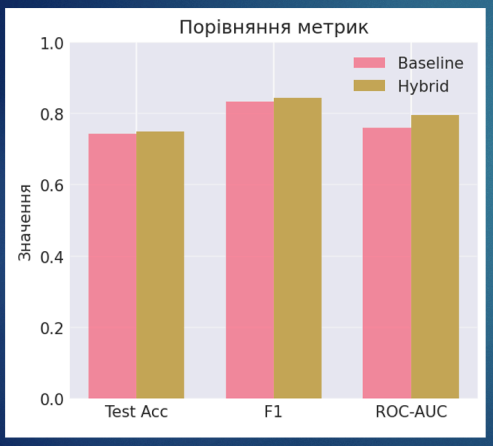
- Функція втрат – кросентропія
- Класифікаційний шар: 768 → 5 + softmax
- «XLM-RoBERTa»: Accuracy – 0,844; F1 – 0,842
- «mBERT»: Accuracy – 0,816; F1 – 0,814
- Перевага XLM-RoBERTa ≈ 2,8 п.п.
- Найбільша різниця – для емоцій «гнів» та «сум»

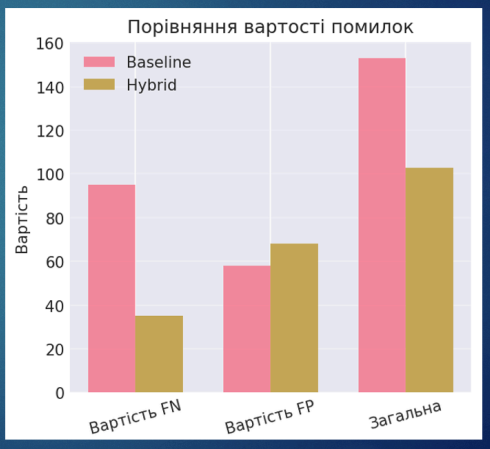


АЛГОРИТМ ТА СТРУКТУРА ГІБРИДНОЇ МОДЕЛІ

- Система включає три компоненти:
  - NLP-модуль на «XLM-RoBERTa» → 18 емоційних ознак
  - 20 фінансових ознак з «German Credit»
  - Класифікатор «XGBoost»
- Генерація 4 текстових відповідей для кожного позичальника
- Побудова 18 похідних емоційних ознак
- Вхідний датасет: 1000 записів (70 % / 30 %)

РЕЗУЛЬТАТИ ПОРІВНЯННЯ БАЗОВОЇ ТА ГІБРИДНОЇ МОДЕЛЕЙ





Висновки

У роботі розглянуто підходи до обробки структурованих та неструктурованих даних у задачі кредитного скорингу. Порівняння моделей показало придатність ансамблевих методів для структурованих даних та трансформерних архітектур для текстових відповідей позичальників. На основі цих компонентів сформовано гібридну модель, яка поєднує фінансові та емоційні ознаки.

Експериментальні результати демонструють, що гібридна модель забезпечує покращення за метриками класифікації та знижує сумарну вартість помилок відповідно до асиметричної матриці вартостей German Credit. Інтеграція текстових даних у процес оцінювання підвищує точність визначення ризику та зменшує очікувані фінансові втрати при прийнятті кредитних рішень.

Дякую за увагу!