ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Методика виявлення фінансового шахрайства в банківській сфері на основі машинного навчання»

на здобуття освітнього ступеня магістра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

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

_____ Олексій СОЛОМОНИК
(підпис)

Виконав: здобувач вищої освіти групи ПДМ-63
Олексій СОЛОМОНИК

Керівник: Сергій КОМПАН
к.ф -м.н.

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

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Магістр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
«_____» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Соломонику Олексію Павловичу

1. Тема кваліфікаційної роботи: «Методика виявлення фінансового шахрайства в банківській сфері на основі машинного навчання»

керівник кваліфікаційної роботи Сергій КОМПАН, к.ф.-м.н., доцент кафедри ІТ,

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «15» жовтня 2024 р. № 320.

2. Строк подання кваліфікаційної роботи «17» грудня 2024 р.

3. Вихідні дані до кваліфікаційної роботи: Науково-технічна література щодо виявлення фінансового шахрайства, методи машинного навчання, характеристики та функціональні можливості платформи Corezoid, демонстраційні дані для навчання та тестування ML-моделі.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Дослідження сучасних методів виявлення фінансового шахрайства.
2. Огляд можливостей платформи Corezoid у реалізації процесів обробки транзакцій.

3. Розробка та оцінка ML-моделі для виявлення шахрайських операцій.
4. Експериментальне дослідження роботи розробленого процесу та аналіз результатів.

5. Перелік ілюстративного матеріалу: *презентація*
   1. Методи виявлення фінансового шахрайства.
   2. Аналіз алгоритмів машинного навчання.
   3. Представлення архітектури ML-моделі для виявлення шахрайства.
   4. Демонстрація роботи процесу з використанням ML-моделі.
   5. Результати оцінки ML-моделі: метрики, графіки, порівняння.
   6. Візуалізація інтеграції ML-моделі у бізнес-процеси.

6. Дата видачі завдання «16» жовтня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	16.10-23.10.2024	
2	Вивчення матеріалів для аналізу технологій машинного навчання	24.10-30.10.2024	
3	Дослідження алгоритмів машинного навчання	31.10-06.11.2024	
4	Аналіз можливостей платформи Corezoid для побудови процесу перевірки транзакцій	07.11-13.11.2024	
5	Розробка ML-моделі для виявлення шахрайства: підготовка даних, навчання, оцінка якості	14.11-22.11.2024	
6	Реалізація процесу перевірки транзакцій на базі Corezoid	23.11-24.11.2024	
7	Оформлення роботи: вступ, висновки, реферат	24.11-25.11.2024	
8	Розробка демонстраційних матеріалів	26.11-28.11.2024	
9	Попередній захист роботи	29.11.2024	

Здобувач вищої освіти _____ Олексій СОЛОМОНИК
(підпис)

Керівник
кваліфікаційної роботи _____ Сергій КОМПАН
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 86 стор., 2 табл., 21 рис., 21 джерело.

Мета роботи – підвищення ефективності виявлення фінансового шахрайства у банківській сфері за допомогою машинного навчання та автоматизації процесів на базі платформи Corezoid.

Об'єкт дослідження – процес виявлення шахрайських транзакцій у банківських системах.

Предмет дослідження – методи машинного навчання для виявлення фінансового шахрайства та їх впровадження у фінансові процеси.

У роботі використано алгоритм Balanced Random Forest Classifier, реалізований у бібліотеці Scikit-learn. Для оцінки якості моделі застосовано інструменти: confusion_matrix, log_loss, roc_auc_score, balanced_accuracy_score. Проведено візуалізацію результатів за допомогою бібліотеки Seaborn, зокрема створено графіки важливості ознак (feature importance), графіки часткової залежності та теплові карти для аналізу метрик. Здійснено автоматизацію обробки транзакцій за допомогою платформи Corezoid.

Проведено аналіз сучасних методів машинного навчання для задач детектування фінансового шахрайства.

Розроблено та оптимізовано модель для покращення боротьби з шахрайством. Виконано підготовку даних, фіче-інжиніринг, навчання моделі та аналіз її метрик якості. Проведено оцінку та аналіз гіперпараметрів, визначено важливість ознак, виконано інтерпретацію ймовірностей та побудовано графіки часткової залежності (PDP plots). Реалізовано аналіз результатів моделі за допомогою TreeExplainer.

Проведено експериментальну оцінку моделі, зокрема аналіз метрик ROC-AUC, balanced_accuracy та log_loss. Результати дослідження візуалізовано за допомогою графіків на основі бібліотеки Seaborn.

КЛЮЧОВІ СЛОВА: ФІНАНСОВЕ ШАХРАЙСТВО, МАШИННЕ НАВЧАННЯ, ЗБАЛАНСОВАНИЙ ВИПАДКОВИЙ ЛІС, ОЦІНКА МЕТРИК, ВАЖЛИВІСТЬ ОЗНАК, ГРАФІКИ ЧАСТКОВОЇ ЗАЛЕЖНОСТІ, ПОЯСНЕННЯ МОДЕЛІ, АВТОМАТИЗАЦІЯ ПРОЦЕСІВ, АНТИФРОД-СИСТЕМИ.

ABSTRACT

Text part of the master's qualification work: 86 pages, 21 pictures, 2 table, 21 sources.

The purpose of the work is to improve the efficiency of financial fraud detection in the banking sector using machine learning and automation of processes based on the Corezoid platform.

Object of research – the process of detecting fraudulent transactions in banking systems.

Subject of research – machine learning methods for detecting financial fraud and their implementation on the Corezoid platform.

Summary of the work: The Balanced Random Forest Classifier algorithm, implemented in the Scikit-learn library, was used in the study. Tools such as confusion_matrix, log_loss, roc_auc_score, and balanced_accuracy_score were applied for model evaluation. Data visualization was conducted using the Seaborn library, including feature importance plots, partial dependency plots (PDP), and heatmaps for metric analysis. The automation of transaction processing was implemented through the Corezoid platform.

A comprehensive analysis of modern machine learning methods for detecting financial fraud was performed. A model was developed and optimized to enhance fraud detection. Data preparation, feature engineering, model training, and evaluation of its metrics were conducted. The analysis included hyperparameter tuning, probability interpretation, and graphical representation of feature dependencies using PDP plots. TreeExplainer was utilized for detailed model interpretation.

The experimental evaluation of the model was performed, including the analysis of metrics such as ROC-AUC, balanced_accuracy, and log_loss. The results were visualized using Seaborn-based graphs.

KEYWORDS: INANCIAL FRAUD, MACHINE LEARNING, BALANCED RANDOM FOREST, METRIC EVALUATION, FEATURE IMPORTANCE, PARTIAL DEPENDENCY PLOTS, MODEL EXPLANATION, PROCESS AUTOMATION, ANTI-FRAUD SYSTEMS.

ЗМІСТ

ВСТУП

Фінансове шахрайство є однією з найбільших загроз для сучасної банківської сфери. З розвитком технологій та збільшенням обсягу цифрових транзакцій масштаби шахрайських операцій стрімко зростають, що створює значні виклики для фінансових установ. Банки та інші фінансові організації стикаються з необхідністю забезпечення безпеки своїх клієнтів, зберігання їхніх активів та мінімізації збитків від шахрайських дій. У таких умовах виникає гостра потреба у впровадженні ефективних методів виявлення фінансового шахрайства, здатних протистояти новим загрозам.

Традиційні методи боротьби з фінансовим шахрайством, такі як ручний аналіз транзакцій або базові правила для їх перевірки, стають недостатньо ефективними. Вони не можуть своєчасно реагувати на зростаючі обсяги даних і складність шахрайських схем. Сучасні підходи дедалі більше орієнтовані на використання алгоритмів машинного навчання, які дозволяють автоматизувати процес аналізу даних, виявляти приховані закономірності та швидко адаптуватися до нових типів загроз.

Машинне навчання відкриває можливості для побудови високоточних моделей, які аналізують транзакції в реальному часі, враховуючи десятки та сотні факторів. Такі моделі здатні виявляти нетипову поведінку, ідентифікувати шахрайські дії та допомагати мінімізувати фінансові втрати. Особливу увагу привертають підходи, що дозволяють інтегрувати моделі в існуючі бізнес-процеси, забезпечуючи безперервний моніторинг та оперативну реакцію на підозрілі дії.

Мета даної магістерської роботи – підвищення ефективності виявлення фінансового шахрайства у банківській сфері за рахунок застосування методів машинного навчання та автоматизації процесів на базі платформи Corezoid.

Об'єктом дослідження є процес виявлення шахрайських транзакцій у банківських системах.

Предмет дослідження – методи машинного навчання для виявлення фінансового шахрайства та їх впровадження у фінансові процеси.

Для досягнення поставленої мети необхідно вирішити такі завдання:

1. Провести аналіз природи фінансового шахрайства, його видів, масштабів і тенденцій розвитку.
2. Вивчити сучасні методи та інструменти боротьби з шахрайством, включаючи підходи, засновані на аналітиці даних.
3. Дослідити можливості машинного навчання для виявлення фінансового шахрайства, зокрема популярні алгоритми, такі як дерева рішень, Random Forest та методи бустингу.
4. Розробити та протестувати модель машинного навчання для аналізу банківських транзакцій і виявлення шахрайських операцій.
5. Інтегрувати модель у бізнес-логіку платформи Corezoid для автоматизації процесу перевірки транзакцій.
6. Провести експериментальне дослідження ефективності моделі, включаючи оцінку її точності та швидкості роботи.

Актуальність роботи полягає у зростаючій кількості шахрайських дій у фінансовій сфері, що вимагає впровадження сучасних автоматизованих рішень для їх виявлення. Розробка та впровадження методів машинного навчання є одним із найперспективніших напрямків у боротьбі з цією проблемою, що має важливе значення для забезпечення стійкості банківських систем у сучасному цифровому середовищі.

1 ТЕОРЕТИЧНІ ОСНОВИ ВИЯВЛЕННЯ ФІНАНСОВОГО ШАХРАЙСТВА

1.1 Природа фінансового шахрайства: поняття, види, масштаби

Фінансове шахрайство (рис. 1.1) – це умисні дії, спрямовані на отримання неправомірної вигоди шляхом обману, маніпуляції або зловживання довірою, що завдають фінансових збитків окремим особам, організаціям чи державі. Фінансове шахрайство є складною соціально-економічною проблемою, яка охоплює широкий спектр протиправних дій, включаючи, але не обмежуючись, банківські махінації, підробки документів, крадіжку даних та інтернет-шахрайство.

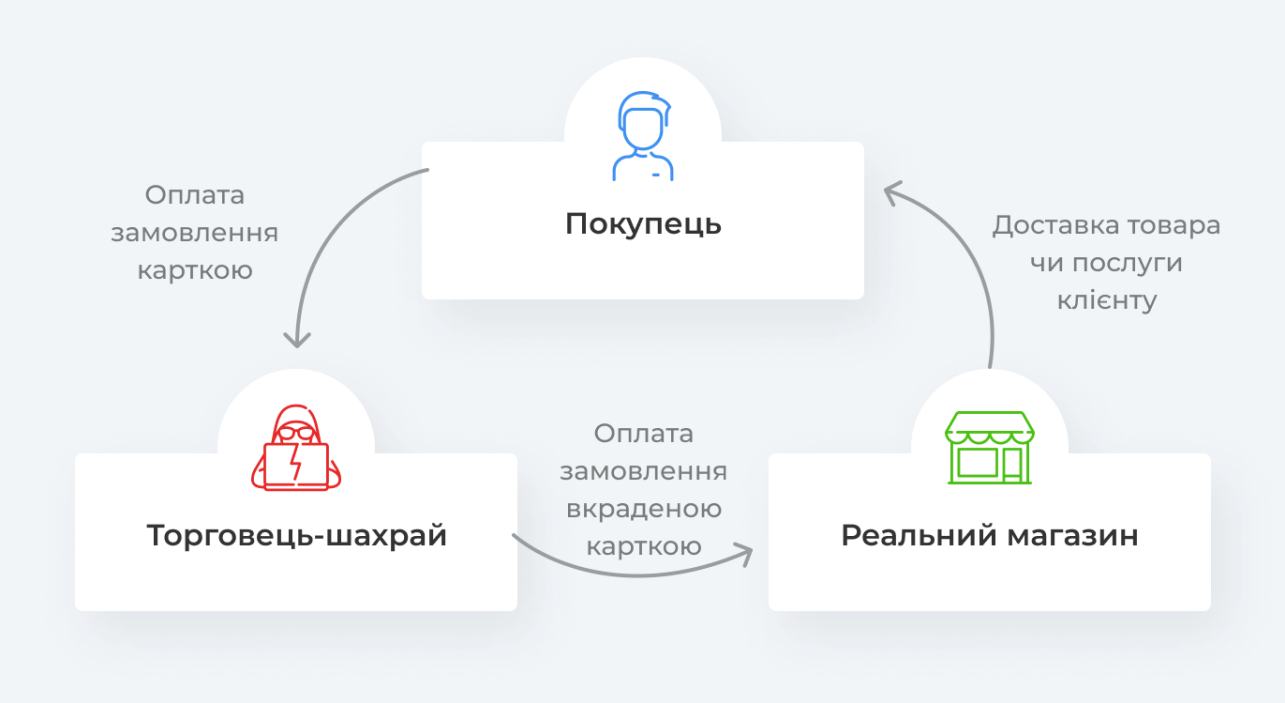


Рис. 1.1. Типовий приклад процесу фінансового шахрайства в електронній комерції

Фінансове шахрайство можна класифікувати за кількома критеріями:

1. За сферою застосування:
    - Шахрайство з банківськими картками: фальсифікація карток, крадіжка даних, фішинг.
    - Махінації з кредитами: створення фальшивих кредитних заявок, підробка документів.
    - Зловживання у страхуванні: фальшиві заявки на виплату.
    - Шахрайство у електронній комерції: створення фіктивних магазинів, обман покупців.
2. За технологіями:
    - Використання шкідливого програмного забезпечення (віруси, трояни).
    - Соціальна інженерія (фішинг, вішинг).
    - Злом систем безпеки та крадіжка даних.
3. За мотивами:
    - Особиста вигода (крадіжка коштів).
    - Корпоративна вигода (відмивання грошей, маніпуляція звітністю).
    - Політичні мотиви (фінансування тероризму).

Масштаби фінансового шахрайства зростають у геометричній прогресії разом із розширенням цифрових фінансових послуг. За даними різних джерел, фінансові втрати від шахрайства в глобальному масштабі вимірюються сотнями мільярдів доларів щороку. Наприклад, лише у сфері банківських карток у 2023 році збитки досягли $32 млрд.

Базові аспекти протидії шахрайству включають:

- Попередження: впровадження технологій для виявлення потенційно підозрілих транзакцій.
- Детекція: використання методів аналізу даних для оперативного виявлення шахрайства.

- Мітрагація: розробка систем, що дозволяють зменшити збитки після виявлення шахрайської діяльності.

Методи боротьби з фінансовим шахрайством (рис. 1.2) включають як традиційні підходи, такі як аналіз вручну, так і автоматизовані рішення, які базуються на сучасних технологіях, включаючи машинне навчання, штучний інтелект і блокчейн. Важливим фактором є також постійна адаптація до нових типів загроз, які еволюціонують разом із розвитком технологій.

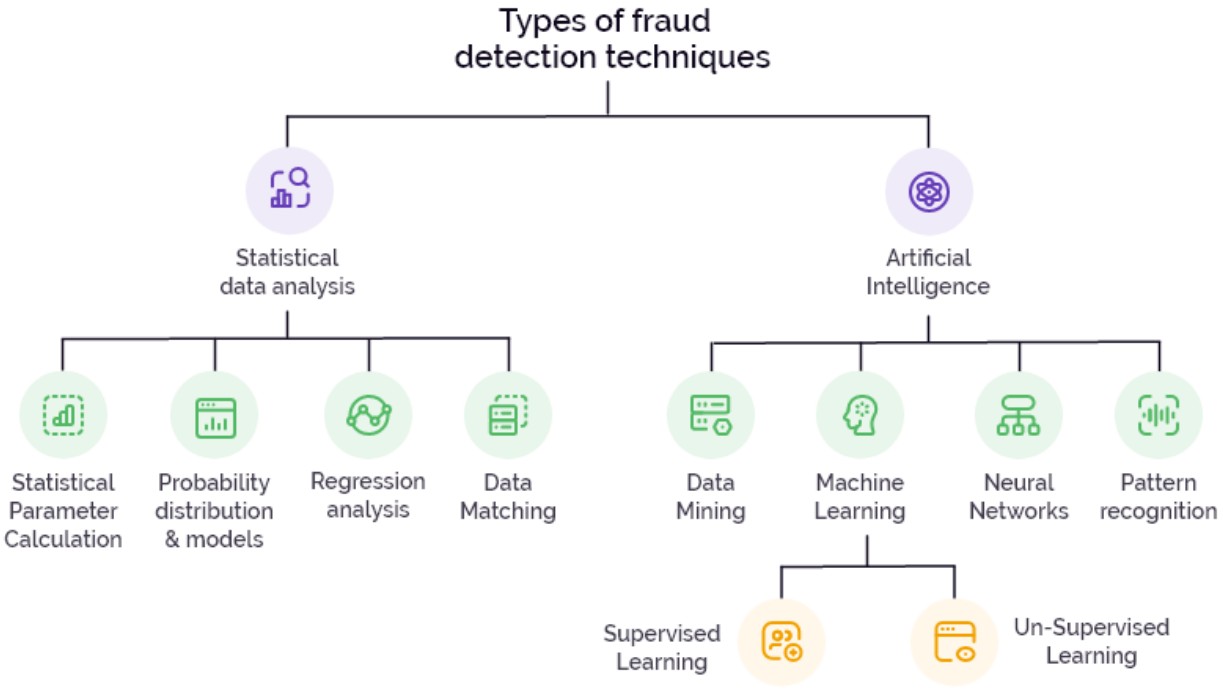


Рис. 1.2. Інструменти для виявлення та протидії фінансовому шахрайству

1.2 Методи та інструменти боротьби з фінансовим шахрайством

Методи та інструменти боротьби з фінансовим шахрайством (рис. 1.3) включають широкий спектр підходів, що дозволяють виявляти шахрайські дії та мінімізувати ризики. Основна мета цих інструментів – забезпечення безпеки фінансових операцій, зменшення збитків та підвищення довіри клієнтів до банківських і фінансових установ.

Основні методи боротьби з шахрайством:

1. Правила та ліміти:

- Використання попередньо встановлених правил, таких як ліміти на суми транзакцій, блокування підозрілих дій.
- Налаштування індивідуальних порогів ризику для клієнтів.

2. Методи статистичного аналізу:
    - Виявлення аномалій у поведінці користувачів.
    - Створення профілів клієнтів для ідентифікації нетипових транзакцій.
3. Методи машинного навчання:
    - Використання кластерного аналізу для групування транзакцій за подібними ознаками.
    - Розробка моделей для прогнозування ймовірності шахрайства.
    - Застосування алгоритмів бустингу, дерев рішень та нейронних мереж.
4. Блокчейн-технології:
    - Використання децентралізованих мереж для підвищення прозорості та унеможливлення підробок транзакцій.
5. Мультимодальні системи верифікації:
    - Поєднання кількох рівнів аутентифікації: біометричні дані, SMS верифікація, двофакторна аутентифікація.
6. Соціальна інженерія та освіта клієнтів:
    - Підвищення обізнаності користувачів про методи шахраїв, такі як фішинг чи шкідливі програми.
    - Проведення кампаній з кібербезпеки.

Інструменти боротьби з шахрайством:

1. Системи моніторингу транзакцій:
    - Платформи, які аналізують транзакції в режимі реального часу для виявлення підозрілої активності.
    - Наприклад, система Corezoid, яка автоматизує перевірку транзакцій та інтегрує результати з бізнес-процесами.

2. Інструменти аналітики даних:
    - Використання інструментів для роботи з великими обсягами даних (Big Data) для виявлення прихованих закономірностей у поведінці клієнтів.
3. Модулі кібербезпеки:
    - Захист систем від хакерських атак, крадіжки даних та шкідливого програмного забезпечення.
    - Використання антивірусних систем і брандмауерів.
4. Хмарні рішення:
    - Платформи, що дозволяють масштабувати ресурси для обробки великих обсягів фінансових даних і аналізу транзакцій у реальному часі.

Розвиток технологій та збільшення обсягів транзакцій вимагає постійного вдосконалення методів боротьби з шахрайством. Сучасні інструменти забезпечують швидкість і точність виявлення підозрілих дій, але потребують регулярного оновлення, адаптації до нових типів атак та інтеграції з існуючими процесами.

Рис. 1.3. Основні методи та інструменти боротьби з фінансовим шахрайством

1.3 Використання аналітики даних та алгоритмів для виявлення шахрайства

Аналітика даних стала невід'ємною частиною боротьби з фінансовим шахрайством, оскільки сучасні технології дозволяють ефективно аналізувати величезні обсяги даних у реальному часі. Застосування алгоритмів аналітики забезпечує можливість оперативно виявляти підозрілі дії, прогнозувати ризики та адаптувати системи до нових методів шахрайства.

Основні аспекти аналітики даних у виявленні шахрайства:

1. Аналіз транзакцій у реальному часі. Аналітичні системи дозволяють моніторити транзакції у режимі реального часу, виявляючи аномальні патерни поведінки. Для цього використовуються алгоритми машинного навчання, що виявляють нетипові операції, наприклад, надто часті платежі, нестандартні суми або транзакції з нових геолокацій.
2. Використання великих даних (Big Data). Технології Big Data забезпечують обробку великих обсягів інформації з різних джерел, таких як:
    - історія транзакцій;
    - геолокаційні дані;
    - інформація про пристрої та IP-адреси;
    - соціальні мережі та публічні дані. Big Data дозволяє створювати багатовимірні моделі поведінки клієнтів і виявляти приховані залежності.
3. Візуалізація даних. Застосування графічних інструментів допомагає аналітикам швидко інтерпретувати дані, виявляючи тренди та потенційно шахрайські дії. Наприклад, графічні моделі можуть демонструвати зв'язки між транзакціями, що вказують на можливу організовану діяльність.

Аналітичні алгоритми можна умовно поділити на кілька категорій:

1. Методи класифікації:
    - Логістична регресія: використовується для оцінки ймовірності шахрайства, базуючись на історичних даних.

- SVM (підтримуючі вектори): алгоритм, що будує гіперплощини для розділення нормальних і шахрайських транзакцій.

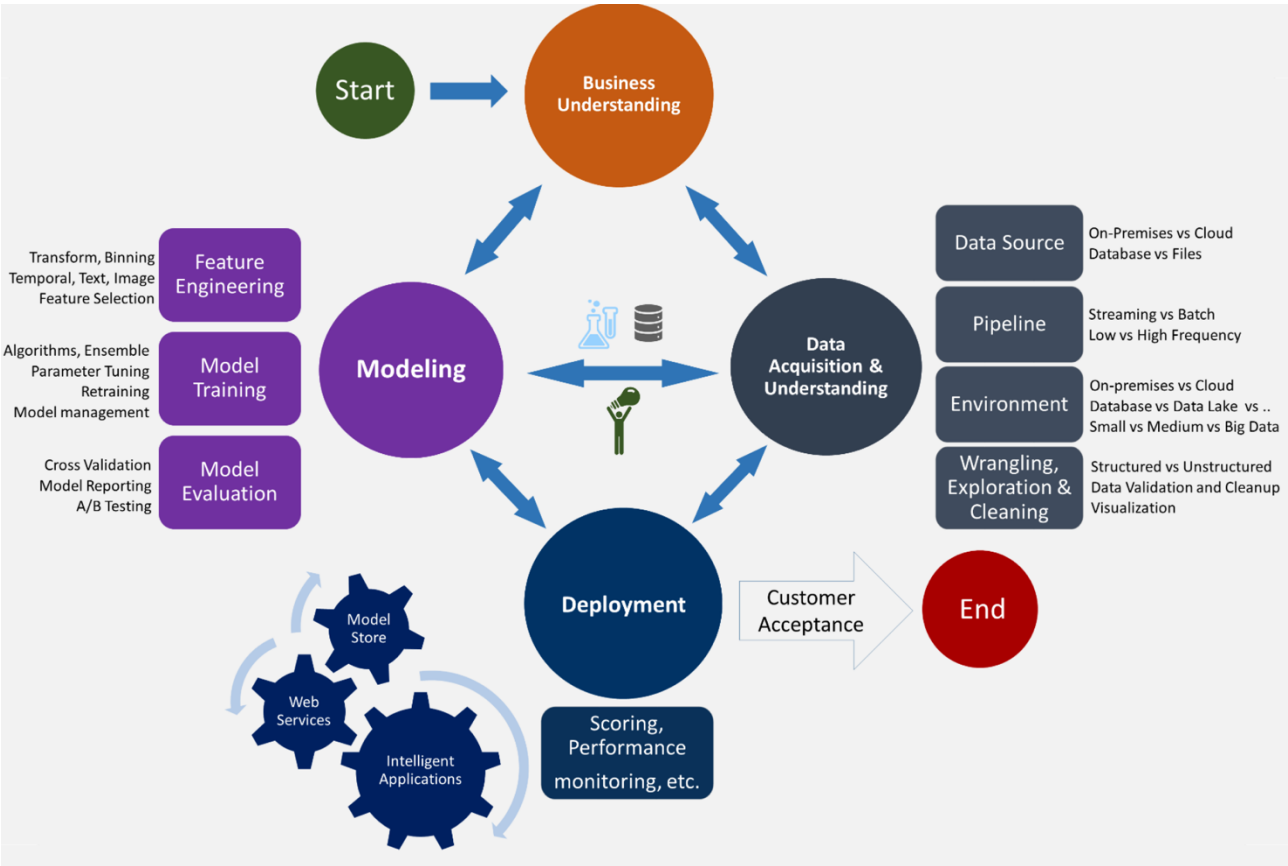
2. Методи кластеризації
    - K-Means: групує дані за схожими ознаками, що дозволяє виділити аномальні кластери.
    - DBSCAN: виявляє аномалії шляхом аналізу щільності точок у багатовимірному просторі.
3. Аномалії
    - Алгоритми на основі правил (rule-based systems): використовують заздалегідь встановлені правила для виявлення підозрілих дій, наприклад, великі транзакції у незвичних геолокаціях.
    - Гаусівські моделі для пошуку відхилень у нормальному розподілі даних.
4. Гібридні підходи Комбінація кількох алгоритмів для досягнення високої точності. Наприклад, використання кластеризації для попереднього відбору даних, а потім застосування моделей класифікації для точного визначення шахрайських транзакцій.

Практичні приклади алгоритмів:

1. Дерева рішень Простий та зрозумілий метод для моделювання правил виявлення шахрайства. Дерева рішень ідеально підходять для невеликих наборів даних, оскільки вони легко інтерпретуються.
2. Random Forest Використання ансамблевого підходу, що підвищує точність і зменшує ризик переобучення.
3. Boosting Алгоритми, такі як XGBoost, CatBoost, LightGBM, дозволяють працювати з великими наборами даних, зберігаючи швидкість і точність.

Алгоритми машинного навчання використовуються у сучасних платформах, таких як Corezoid, для:

- оцінки ризику в реальному часі;
- автоматизації перевірки підозрілих транзакцій;
- формування рекомендацій для операторів безпеки.



На рис. 1.4 представлено загальну схему використання аналітики даних для виявлення шахрайства.

Аналітика даних продовжує розвиватися, забезпечуючи фінансові установи потужними інструментами для протидії шахрайству. Ефективність таких рішень залежить від якості даних, правильного вибору алгоритмів та інтеграції систем моніторингу з існуючими процесами.

2 ПЛАТФОРМА COREZOID: ФУНКЦІЇ ТА ЗАСТОСУВАННЯ У ФІНАНСОВИХ ПРОЦЕСАХ

2.1 Основні функції та можливості платформи Corezoid

Corezoid є інноваційною платформою для управління бізнес-процесами, яка змінює підхід до автоматизації фінансових операцій і моніторингу. Вона забезпечує не лише швидкість та ефективність, але й надає можливість адаптувати процеси відповідно до потреб конкретного бізнесу. Завдяки використанню Corezoid, фінансові установи можуть досягти вищого рівня автоматизації, зменшити операційні витрати, покращити безпеку та прискорити взаємодію з клієнтами.

Платформа Corezoid забезпечує комплексний підхід до автоматизації процесів, пропонуючи набір інструментів, які можна налаштовувати відповідно до потреб користувача. Це дозволяє фінансовим організаціям гнучко реагувати на виклики, пов’язані з управлінням операціями, обробкою великих обсягів даних і забезпеченням прозорості у процесах.

1. Моделювання бізнес-процесів

Однією з найважливіших функцій Corezoid є можливість моделювання бізнес-процесів. Інтуїтивний інтерфейс дозволяє створювати та налаштовувати складні процеси за допомогою блоків і вузлів. Цей підхід значно зменшує залежність від програмістів і скорочує час впровадження нових процесів.

Наприклад, створення процесу перевірки транзакцій може включати такі етапи:

- отримання даних із зовнішніх джерел (API або баз даних);
- застосування бізнес-правил для оцінки ризиків транзакції;
- автоматичне ухвалення рішення (підтвердження чи блокування);

- зберігання результатів у системах аналітики.

Інтуїтивний підхід до налаштування процесів на Corezoid дозволяє значно зменшити час на розробку нових функціональних модулів.

2. Обробка даних у реальному часі

Corezoid забезпечує обробку великих обсягів даних у режимі реального часу. Ця функція є особливо важливою для фінансових установ, які щодня обробляють мільйони транзакцій.

Платформа підтримує інтеграцію з різними джерелами даних, такими як:

- банківські системи, що надають дані про транзакції;
- CRM-системи, які містять інформацію про клієнтів;
- системи управління ризиками для перевірки підозрілих операцій.

Обробка даних у реальному часі дозволяє виявляти потенційно шахрайські дії на ранніх етапах і зменшувати ризики втрат для організації. Наприклад, якщо транзакція з великою сумою проводиться з незвичної геолокації, Corezoid може автоматично блокувати її та ініціювати перевірку.

3. Інтеграція з існуючими системами

Corezoid підтримує багаторівневу інтеграцію з іншими системами, такими як ERP, CRM та банківські платформи. Це забезпечує швидку взаємодію між різними модулями компанії та дозволяє об’єднувати дані з різних джерел.

Платформа також підтримує різні методи інтеграції:

- API-запити: отримання та відправлення даних у зовнішні системи;
- Вебхуки: автоматичні повідомлення про події;
- Файлові формати: обмін даними через файли (CSV, XML).

Завдяки цій інтеграції, компанії можуть створювати єдину екосистему для управління своїми бізнес-процесами. Наприклад, Corezoid може об’єднувати дані з мобільних додатків банку, транзакційних систем і аналітичних інструментів у єдиному процесі.

4. Автоматизація рутинних процесів

Фінансові установи щодня стикаються з великою кількістю рутинних завдань, таких як перевірка документів, аналіз транзакцій та створення звітів. Corezoid дозволяє автоматизувати ці завдання, зменшуючи потребу у ручній праці та мінімізуючи людські помилки.

Наприклад, автоматизація обробки кредитних заявок може включати:

- збір даних про клієнта із зовнішніх джерел;
- перевірку кредитної історії;
- оцінку ризиків на основі машинного навчання;
- автоматичне прийняття рішення про видачу кредиту.

Це значно скорочує час, необхідний для обробки заявки, та покращує клієнтський досвід.

5. Моніторинг і звітність

Corezoid пропонує розвинуті інструменти для моніторингу виконання процесів у реальному часі. На кожному етапі виконання процесу можна отримати деталізовану інформацію про статуси задач, вузли, які обробляють дані, та час виконання.

Для аналітиків та керівників платформа дозволяє створювати звіти, які включають ключові показники ефективності (KPI). Наприклад, звіт про виконання транзакцій може містити:

- кількість успішно виконаних операцій;
- кількість заблокованих транзакцій;
- середній час обробки.

Крім того, платформа підтримує візуалізацію даних у вигляді графіків, діаграм і таблиць, що спрощує аналіз великих обсягів інформації.

Роль Corezoid у фінансових процесах

Corezoid став важливим інструментом у фінансовій галузі завдяки своїй здатності адаптуватися до змін та забезпечувати автоматизацію складних процесів. Платформа значно підвищує ефективність управління ризиками, мінімізує людський фактор і сприяє покращенню клієнтського обслуговування.

На рис. 2.1 зображено типовий процес автоматизації перевірки транзакцій у Corezoid, який включає інтеграцію з банківськими системами, автоматичну оцінку ризиків та формування звітів.

Завдяки Corezoid, фінансові установи можуть значно зменшити витрати на обробку даних, мінімізувати ризики шахрайства та покращити взаємодію з клієнтами. Платформа залишається лідером у галузі завдяки постійному вдосконаленню функціоналу та інтеграції з новими технологіями.

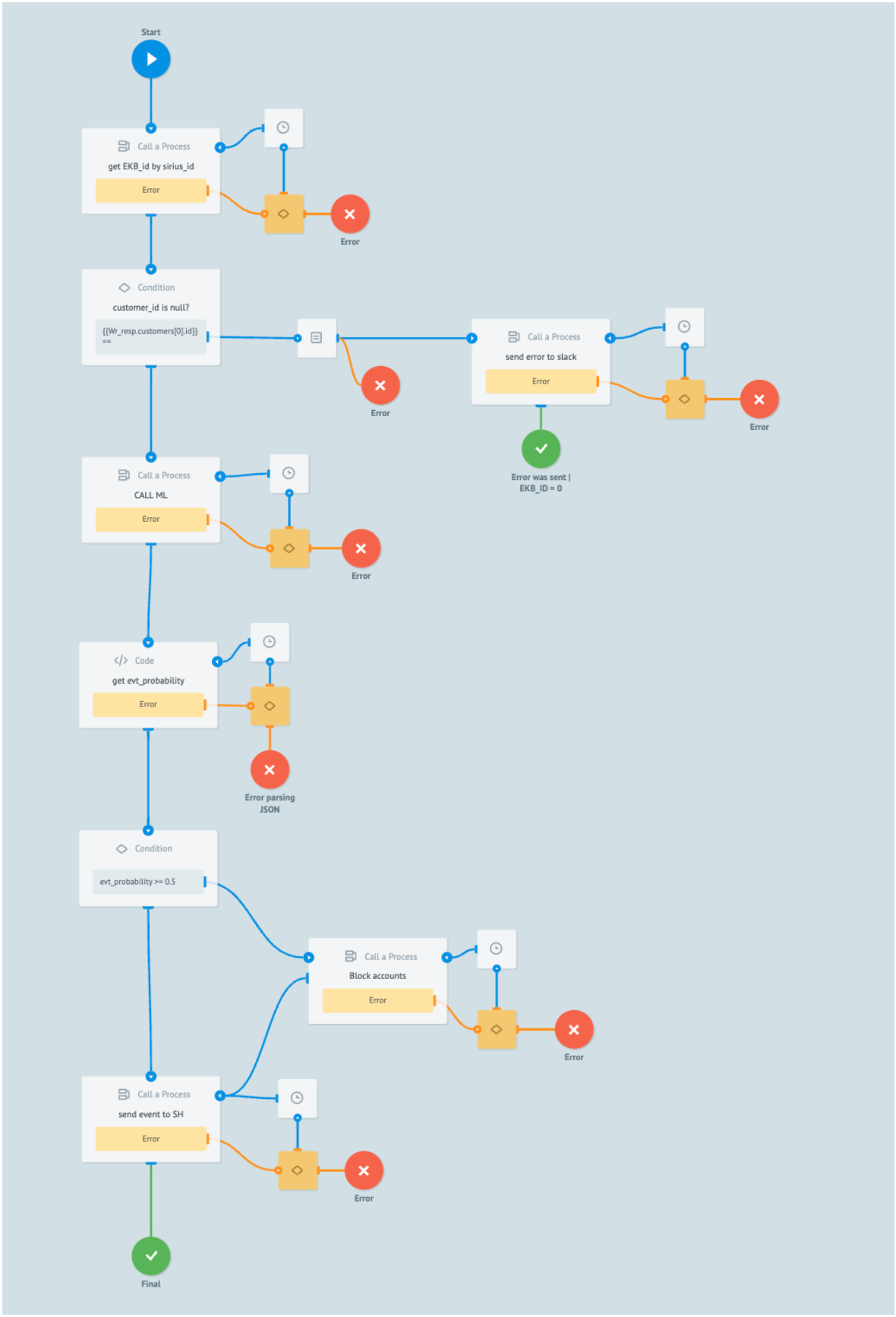


Рис. 2.1. Автоматизація перевірки транзакцій у Corezoid

2.2 Архітектура процесів на платформі Corezoid

Corezoid пропонує набір базових вузлів (нод), які забезпечують будівництво складних бізнес-процесів без необхідності написання коду. Кожен вузол виконує конкретне завдання, і їх комбінація дозволяє створювати динамічні та інтегровані рішення. Детальний опис основних вузлів платформи дає змогу зрозуміти їхню роль і можливості.

Головні вузли платформи Corezoid:

1. API Node (Вузол для роботи з API)
    - Опис: Цей вузол дозволяє здійснювати інтеграцію з іншими системами через API-запити. Він може виконувати як вхідні, так і вихідні запити до зовнішніх систем.
    - Основні функції:
        - Отримання даних із зовнішніх джерел.
        - Відправка запитів до API для оновлення або запису даних.
        - Підтримка REST API та SOAP-протоколів.
    - Приклад використання: Здійснення запиту до зовнішньої платформи для перевірки банківського рахунку клієнта перед здійсненням транзакції.
2. Timer Node (Таймер)
    - Опис: Вузол використовується для затримки виконання процесу або періодичного виконання завдань.
    - Основні функції:
        - Затримка між виконанням дій (наприклад, 5 хвилин очікування перед відправкою нагадування клієнту).
        - Планування завдань у визначений час або з певною періодичністю.

- Приклад використання: Надсилання щомісячного звіту клієнту через заданий інтервал часу.

3. State Node (Стан)
    - Опис: Цей вузол використовується для визначення поточного стану процесу. Він дозволяє реалізовувати умовні логіки (наприклад, "якщо транзакція перевищує ліміт, то...").
    - Основні функції:
        - Розгалуження процесу залежно від умов.
        - Збереження поточного стану процесу для подальшого відновлення.
    - Приклад використання: Визначення, чи є клієнт у списку VIP для подальшого вибору персоналізованого сценарію обслуговування.
4. Math Node (Математичний вузол)
    - Опис: Вузол для виконання математичних обчислень.
    - Основні функції:
        - Підрахунок суми, відсотків, середнього значення та інших числових параметрів.
        - Обробка даних для використання в бізнес-правилах.
    - Приклад використання: Розрахунок комісії за транзакцію залежно від суми операції.
5. Condition Node (Умовний вузол)
    - Опис: Вузол, який дозволяє задавати умови для виконання або перенаправлення процесу.
    - Основні функції:
        - Виконання дій на основі умов (наприклад, "якщо сума транзакції більше 10 000 — відправити на перевірку").
        - Розгалуження процесів залежно від параметрів.
    - Приклад використання: Перевірка ліміту транзакцій клієнта перед її виконанням.
6. Data Storage Node (Зберігання даних)

- Опис: Вузол використовується для запису та зберігання даних під час виконання процесу.
- Основні функції:
    - Тимчасове збереження даних між етапами процесу.
    - Збереження результатів для подальшої аналітики.
- Приклад використання: Запис деталей транзакції (сума, дата, клієнт) для звітності.

7. Notification Node (Вузол сповіщень)
    - Опис: Забезпечує надсилання повідомлень через різні канали (SMS, email, push-сповіщення).
    - Основні функції:
        - Генерація повідомлень на основі параметрів процесу.
        - Надсилання інформації клієнту, оператору чи іншій системі.
    - Приклад використання: Повідомлення клієнта про успішне виконання транзакції або блокування підозрілої дії.
8. Webhook Node (Вебхук)
    - Опис: Вузол використовується для прийому даних із зовнішніх систем через вебхуки.
    - Основні функції:
        - Обробка подій, які надходять із зовнішніх джерел.
        - Ініціювання процесів у Corezoid у відповідь на зовнішні події.
    - Приклад використання: Запуск процесу перевірки транзакції при отриманні вебхука від платіжної системи.
9. Loop Node (Цикл)
    - Опис: Вузол дозволяє організовувати цикли у процесах.
    - Основні функції:
        - Повторення певних дій до виконання умови.
        - Обробка масивів даних.
    - Приклад використання: Перевірка статусу транзакції через регулярні інтервали часу до отримання підтвердження.

10. Log Node (Логування)
    - Опис: Використовується для запису інформації про виконання процесів.
    - Основні функції:
        - Збереження даних про статус вузлів.
        - Формування логів для аналізу виконання.
    - Приклад використання: Логування даних про успішність інтеграційних запитів для подальшої діагностики.

Переваги вузлів Corezoid

Завдяки доступним вузлам, Corezoid дозволяє створювати складні бізнес-логіки без потреби в складному кодуванні. Основні переваги включають:

- Гнучкість: можливість комбінації вузлів для реалізації різних сценаріїв.
- Масштабованість: вузли працюють ефективно навіть при великому навантаженні.
- Швидкість: створення бізнес-логіки у Corezoid займає мінімум часу завдяки інтуїтивному інтерфейсу.

Таким чином, набір вузлів Corezoid є основою, яка забезпечує можливість реалізації майже будь-якого процесу, включаючи фінансові операції, моніторинг транзакцій та автоматизацію бізнес-завдань.

2.3 Роль Corezoid у автоматизації перевірки транзакцій

Corezoid займає центральне місце в автоматизації перевірки транзакцій у фінансовій сфері. Завдяки гнучкій архітектурі та інтеграційним можливостям, платформа дозволяє автоматизувати аналіз великих обсягів даних, зменшити час обробки операцій та підвищити точність виявлення шахрайства. Це робить

Corezoid важливим інструментом для банків, платіжних систем і фінансових організацій.

Основні аспекти ролі Corezoid у перевірці транзакцій

1. Автоматизація правил перевірки Corezoid дозволяє реалізовувати бізнес-правила для перевірки транзакцій без написання коду. Наприклад:
    - Ліміти на суми транзакцій;
    - Геолокаційні обмеження (транзакції з незвичних регіонів);
    - Перевірка типів карток або платіжних засобів.

Завдяки вбудованим умовам (Condition Nodes), платформа забезпечує динамічне виконання перевірок залежно від вхідних даних.

2. Інтеграція з зовнішніми системами Платформа підтримує інтеграцію з:
    - CRM для отримання інформації про клієнта;
    - Платіжними шлюзами для обробки даних про транзакції;
    - Системами моніторингу ризиків (Risk Management Systems) для оцінки шахрайства.

Це дозволяє створити єдину екосистему для аналізу транзакцій у реальному часі.

3. Обробка великих обсягів даних Завдяки масштабованості Corezoid може обробляти сотні тисяч транзакцій одночасно. Кожна транзакція проходить через визначені етапи обробки:
    - Первинна перевірка відповідності правил.
    - Застосування алгоритмів оцінки ризиків.
    - Формування рішення: схвалення, відхилення чи відправлення на додаткову перевірку.
4. Швидкість і точність Платформа обробляє транзакції у реальному часі, що критично важливо для фінансових операцій. Використання умовних вузлів (State Nodes) та інтеграції з API забезпечує точність перевірок.

Основні етапи автоматизації перевірки транзакцій у Corezoid

1. Збір даних Інформація про транзакцію надходить до Corezoid через API-запит або вебхук. Сюди можуть входити:
    - Ідентифікатор клієнта;
    - Сума транзакції;
    - Геолокація;
    - Тип платіжного засобу.
2. Перевірка бізнес-правил Транзакція перевіряється на відповідність попередньо визначеним умовам:
    - Чи перевищує сума ліміт?
    - Чи здійснюється операція у дозволеній геолокації?
    - Чи є рахунок клієнта активним?
3. Оцінка ризиків На цьому етапі можуть застосовуватись алгоритми машинного навчання для оцінки ймовірності шахрайства. Наприклад:
    - Використання кластеризації для визначення нетипової поведінки.
    - Прогнозування ризиків на основі історичних даних.
4. Формування рішення На основі результатів перевірки платформа приймає рішення:
    - Схвалити транзакцію;
    - Заблокувати транзакцію;
    - Передати операцію на ручну перевірку.
5. Запис результатів Результати обробки зберігаються у системі для подальшої аналітики та звітності.

Для кращого розуміння основних етапів перевірки транзакцій та відповідних вузлів платформи Corezoid зображено таблицю 2.1.

Таблиця 2.1

Приклад процесу перевірки транзакції на платформі Corezoid

Етап	Опис	Приклад вузлів
Збір даних	Отримання даних про транзакцію із зовнішніх систем.	API Node, Webhook Node
Перевірка бізнес-правил	Застосування правил для перевірки лімітів, геолокації та статусу рахунку.	Condition Node, State Node
Оцінка ризиків	Використання алгоритмів оцінки ризиків на основі даних клієнта та транзакцій.	Math Node, API Node
Формування рішення	Схвалення, блокування чи передача на ручну перевірку залежно від результатів оцінки.	State Node
Збереження результатів	Занесення результатів перевірки до бази даних для подальшого аналізу.	Data Storage Node
Сповіщення	Надсилання повідомлення клієнту про статус транзакції (успішна, заблокована).	Notification Node

Переваги використання Corezoid у перевірці транзакцій:

1. Ефективність Завдяки автоматизації перевірки знижується час обробки транзакцій, що дозволяє обслуговувати більшу кількість операцій у короткий термін.
2. Масштабованість Платформа може одночасно обробляти тисячі транзакцій, що забезпечує її актуальність для великих фінансових установ.

3. Точність Використання бізнес-правил і алгоритмів машинного навчання дозволяє знижувати кількість помилкових блокувань або пропущеного шахрайства.
4. Гнучкість Corezoid дозволяє змінювати та доповнювати логіку перевірки залежно від змін у регуляторних вимогах або бізнес-процесах.

Corezoid забезпечує потужну платформу для автоматизації перевірки транзакцій. Завдяки модульності, інтеграції та аналітичним можливостям, платформа дозволяє фінансовим установам досягати нових рівнів ефективності та безпеки, відповідаючи сучасним викликам у сфері управління транзакціями.

3 ОГЛЯД МЕТОДІВ МАШИННОГО НАВЧАННЯ ДЛЯ ВИЯВЛЕННЯ ШАХРАЙСТВА

3.1 Популярні алгоритми машинного навчання: теорія та приклади

Машинне навчання є одним із найбільш ефективних підходів до вирішення задач виявлення шахрайства. Завдяки аналізу великих масивів даних і побудові моделей на основі статистичних і математичних закономірностей, алгоритми машинного навчання дозволяють виявляти як очевидні, так і приховані шахрайські дії. У цьому розділі розглянемо найпоширеніші алгоритми, їхні теоретичні основи та практичне застосування.

Методи машинного навчання, які використовуються для виявлення шахрайства, поділяються на кілька основних категорій:

- Алгоритми класифікації, які дозволяють розділяти дані на категорії ("шахрайська" чи "звичайна" транзакція);
- Ансамблеві методи, що поєднують результати декількох алгоритмів для підвищення точності;
- Методи кластеризації, що групують транзакції за схожими характеристиками;
- Методи виявлення аномалій, які шукають операції, що відхиляються від загальної поведінки.

3.1.1 Дерева рішень: переваги та недоліки

Дерева рішень є базовим і простим методом класифікації, який часто використовується як основа для складніших ансамблевих методів.

- Теорія: Дерева рішень розділяють дані на основі логічних умов. Кожен вузол дерева відповідає за перевірку однієї умови, а листя — за кінцевий результат (наприклад, "шахрайська" чи "нормальна" транзакція). Алгоритм будує дерево шляхом пошуку атрибутів, які найбільше зменшують ентропію або збільшують приріст інформації.
- Принцип роботи:
    - Визначається набір правил (наприклад, "якщо сума транзакції > 10 000 і IP-адреса нова, то...").
    - Дані проходять через дерево зверху вниз, перевіряючи кожне правило.
    - Кінцеве рішення приймається на рівні листя.
- Приклад використання: Дерево рішень може бути використане для оцінки ризиків транзакції на основі таких характеристик, як сума операції, час проведення, тип пристрою клієнта тощо.
- Переваги:
    - Простота реалізації та візуалізації.
    - Інтуїтивно зрозуміле пояснення результатів.
    - Можливість роботи з якісними й кількісними даними.
- Недоліки:
    - Схильність до переобучення, особливо на невеликих наборах даних.
    - Менш точний результат у порівнянні з ансамблевими методами.

На рисунку 3.1. показано структуру дерева рішень, яке складається з вузлів, що виконують перевірку умов (Decision Node), і листя (Leaf Node), які містять кінцеві рішення. Кореневий вузол (Root Node) є початковою точкою, з якої дані проходять через гілки (Branch) на основі відповідності умовам, аж до

досягнення листя. Цей процес ілюструє, як дерево розподіляє дані, виконуючи поступові логічні перевірки.

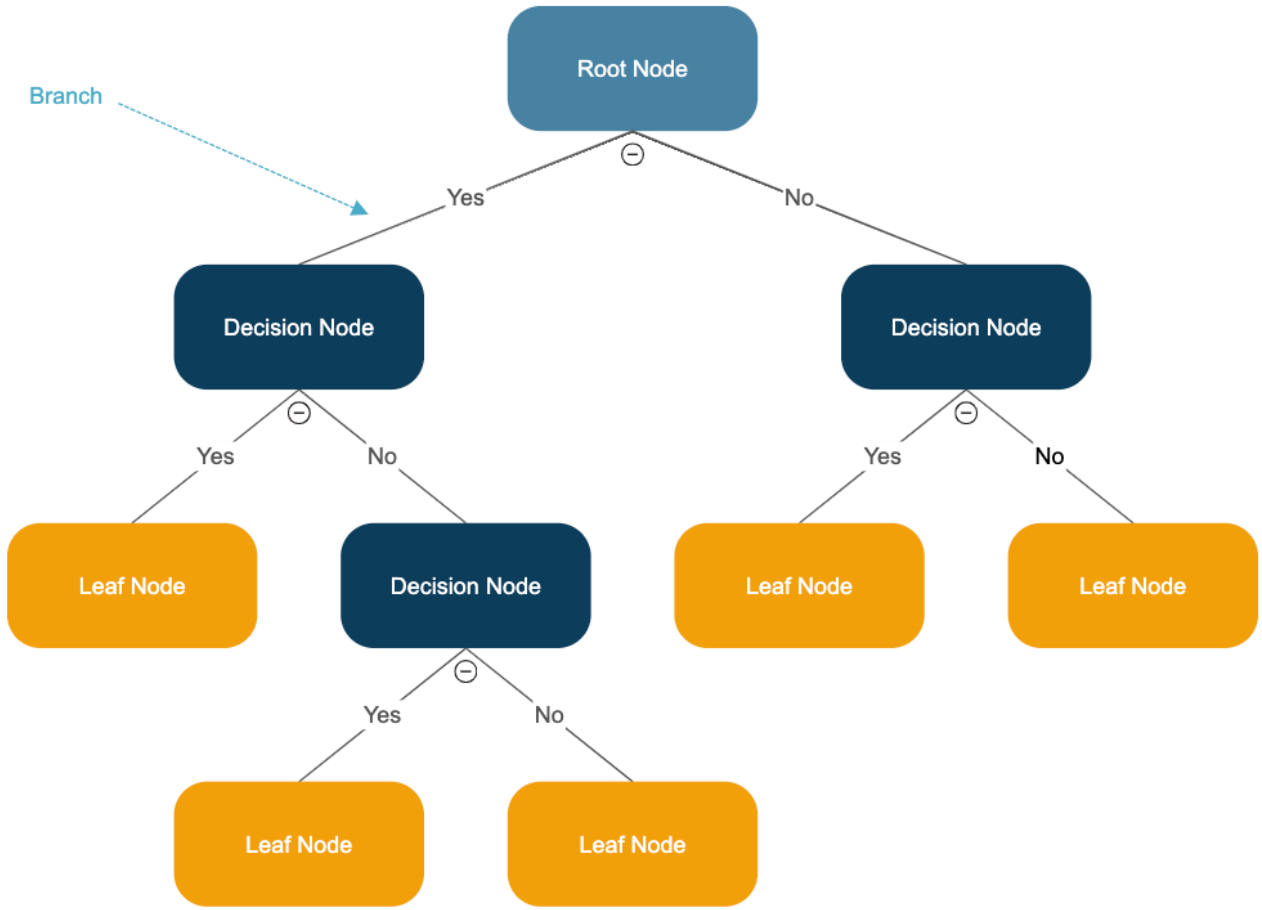


Рис. 3.1. Структура дерева рішень

3.1.2 Random Forest: принцип роботи та використання у задачах фроду

- Random Forest є вдосконаленою версією дерев рішень, що забезпечує більш високу точність і стійкість до переобучення.
- Теорія: Random Forest працює за принципом створення ансамблю дерев рішень. Кожне дерево генерується на випадковій підмножині даних і вибирає випадкові атрибути для поділу на кожному вузлі. Результат визначається голосуванням усіх дерев.
- Принцип роботи:

1. Генеруються N дерев, кожне з яких працює з випадковою вибіркою даних.
2. Кожне дерево приймає рішення ("шахрайська" чи "нормальна" транзакція).
3. Підсумкове рішення приймається більшістю голосів.

- Приклад використання: Random Forest використовується для виявлення шахрайства в банківських транзакціях, де враховуються такі фактори, як геолокація, тип картки, історія клієнта, IP-адреса.
- Переваги:
  1. Висока точність і стійкість до переобучення.
  2. Підходить для великих і складних наборів даних.
  3. Менше схильний до впливу шуму в даних.
- Недоліки:
  1. Великі обчислювальні витрати, особливо для великих ансамблів.
  2. Складність інтерпретації результатів.

На рисунку 3.2. представлено принцип роботи Random Forest, який використовує ансамбль дерев рішень. Кожне дерево будується на випадковій підмножині даних (Dataset) і генерує окремий результат (Result-1, Result-2, ..., Result-N). Остаточне рішення визначається методом голосування більшістю (Majority Voting) для класифікації або середнім значенням (Averaging) для регресії. Цей підхід забезпечує високу точність та стійкість до переобучення завдяки використанню кількох незалежних дерев.

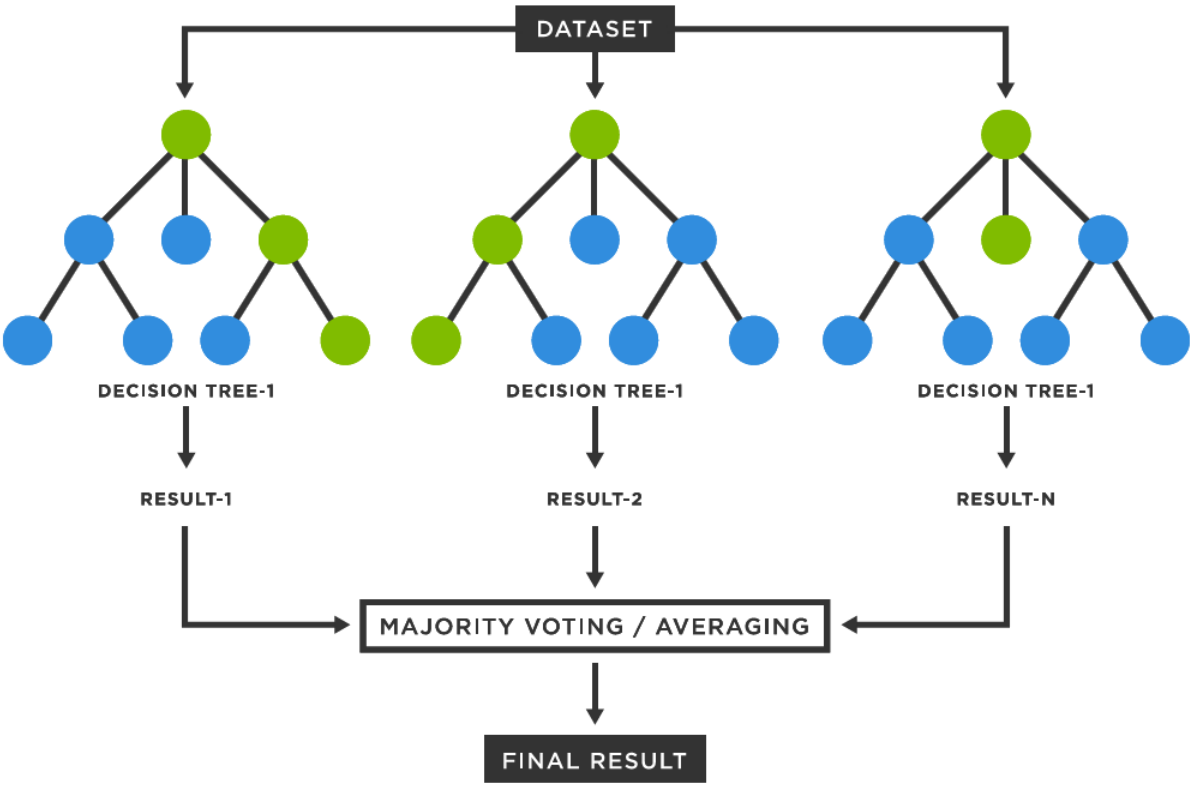


Рис. 3.2. Принцип роботи алгоритму Random Forest

3.1.3 Boosting: поняття, популярні реалізації (XGBoost, CatBoost, LightGBM)

Boosting є ансамблевим методом, що концентрується на поліпшенні точності шляхом послідовного навчання моделей. Кожна наступна модель зосереджується на помилках попередньої.

1. Теорія: Boosting будує кілька слабких моделей (зазвичай дерев рішень), кожна з яких зменшує помилки попередньої. Популярні реалізації boosting включають XGBoost, CatBoost і LightGBM.
2. Принцип роботи:
    - Навчається базова модель на початковому наборі даних.
    - Результати моделі порівнюються з реальними значеннями, а помилки додаються до нового набору даних.

- Наступна модель фокусується на цих помилках, покращуючи загальний результат.

3. Популярні реалізації:
   - XGBoost: Підтримує багатопотокові обчислення, висока точність.
   - CatBoost: Ефективний для роботи з категорійними даними.
   - LightGBM: Оптимізований для швидкої обробки великих наборів даних.
4. Приклад використання: Boosting використовується для складних задач, таких як прогнозування ризику кредитних дефолтів чи аналіз поведінки клієнтів у реальному часі.
5. Переваги:
   - Висока продуктивність і точність.
   - Підходить для великих наборів даних із багатьма змінними.
6. Недоліки:
   - Потребує точного налаштування параметрів.
   - Висока обчислювальна складність.

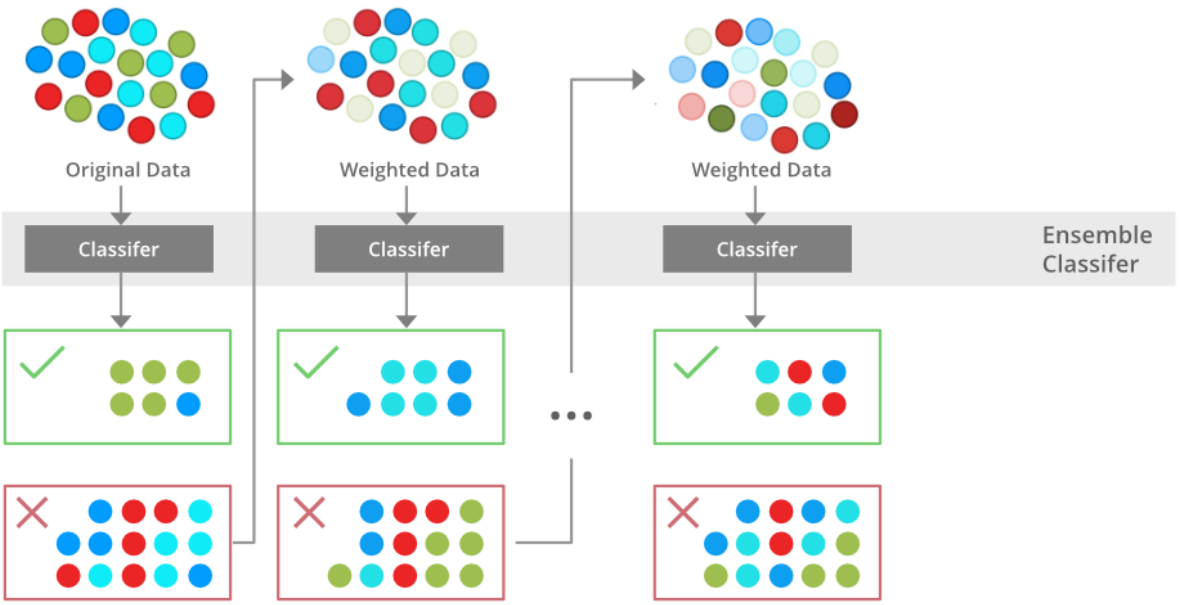


Рис. 3.3. Принцип роботи Boosting-алгоритму

На рисунку 3.3. зображено процес роботи Boosting-алгоритму, який послідовно навчає кілька слабких класифікаторів (Classifier). На кожному етапі ваги даних оновлюються таким чином, щоб наступний класифікатор фокусувався на помилках попереднього. У результаті формується ансамблевий класифікатор (Ensemble Classifier), що поєднує результати всіх слабких моделей, забезпечуючи високу точність та ефективність при класифікації складних даних.

3.2 Особливості використання машинного навчання у банківській сфері

Машинне навчання (ML) стало невід'ємною частиною банківської діяльності завдяки своїй здатності аналізувати великі обсяги даних, знаходити приховані закономірності та приймати високоточні рішення у реальному часі. У банківській сфері ML застосовується для різних завдань, таких як виявлення шахрайства, прогнозування ризиків, покращення клієнтського досвіду, управління активами та автоматизація процесів.

Основні напрями використання машинного навчання в банках:

1. Виявлення шахрайства Машинне навчання дозволяє аналізувати транзакції у реальному часі, виявляючи підозрілі операції. Алгоритми класифікації та виявлення аномалій є основними інструментами у цьому напрямі. Наприклад, вони допомагають визначити транзакції з нестандартних геолокацій, великі суми операцій чи поведінкові зміни клієнтів.
2. Кредитний скоринг ML дозволяє банкам оцінювати кредитоспроможність клієнтів з використанням історичних даних, таких як доходи, поведінка щодо погашення кредитів, демографічна інформація тощо. Ансамблеві методи, такі як Random Forest і Boosting, забезпечують високу точність прогнозування.

3. Сегментація клієнтів Кластеризація клієнтів із застосуванням ML допомагає банкам пропонувати персоналізовані продукти та послуги. Наприклад, клієнтів можна розділити на групи залежно від їхніх фінансових звичок, уподобань і потреб.
4. Оптимізація процесів і автоматизація Машинне навчання використовується для автоматизації рутинних завдань, таких як обробка заявок, управління документообігом і управління ризиками. Це знижує витрати та підвищує продуктивність.
5. Прогнозування ризиків Моделі машинного навчання допомагають передбачати ризики, пов’язані з ринковими змінами, ліквідністю, дефолтами клієнтів або змінами кредитних рейтингів.

Особливості впровадження машинного навчання в банківській сфері:

1. Робота з великими обсягами даних Банки володіють великими масивами даних, які постійно оновлюються. Це дозволяє ML-моделям працювати з великою кількістю ознак для побудови точних прогнозів. Наприклад, при виявленні шахрайства аналізу піддаються тисячі параметрів, таких як IP-адреса, час транзакції, тип картки, історія покупок.
2. Реальний час У банківській сфері рішення часто потрібно приймати у реальному часі, наприклад, під час авторизації транзакції або розгляду кредитної заявки. Машинне навчання дозволяє оперативно аналізувати потік даних і надавати результат за секунди.
3. Дотримання регуляторних вимог Однією з ключових вимог у банківській сфері є прозорість та інтерпретованість моделей. Банки повинні пояснювати свої рішення регуляторам і клієнтам, тому моделі машинного навчання повинні бути зрозумілими та перевіреними.
4. Безпека та конфіденційність Банківські дані є чутливими, тому моделі ML повинні працювати в умовах високих вимог до безпеки.

Усі дані мають бути надійно захищені, а алгоритми — адаптовані для роботи в умовах анонімності.

Таблиця 3.1

Основні переваги та виклики використання ML у банківській сфері

Аспект	Переваги	Виклики
Виявлення шахрайства	Висока точність, виявлення аномалій у реальному часі	Складність побудови моделей для змінних патернів шахрайства
Кредитний скоринг	Поліпшення прогнозування ризиків, автоматизація ухвалення рішень	Інтерпретованість результатів для регуляторів
Аспект	Переваги	Виклики
Сегментація клієнтів	Персоналізація продуктів і послуг, підвищення лояльності	Складність збирання та аналізу неконсолідованих даних
Оптимізація процесів	Зменшення операційних витрат, підвищення продуктивності	Інтеграція ML-моделей у наявну IT-інфраструктуру
Прогнозування ризиків	Своєчасне виявлення потенційних проблем, підвищення стійкості до ринкових коливань	Залежність від якості даних та їхнього постійного оновлення

Таблиця 3.1. Основні переваги та виклики використання ML у банківській сфері відображає ключові аспекти застосування машинного навчання у фінансових установах. У ній порівнюються основні переваги, які ML приносить

до різних напрямків банківської діяльності, із викликами, які можуть виникати при його впровадженні.

Наприклад, для виявлення шахрайства машинне навчання забезпечує високу точність аналізу та оперативне виявлення аномалій у реальному часі. Однак викликом є динамічна природа шахрайських схем, які постійно змінюються, і необхідність регулярно оновлювати моделі. У кредитному скорингу перевагою є автоматизація ухвалення рішень і покращення прогнозування ризиків, проте це супроводжується труднощами з інтерпретованістю моделей для регуляторів.

Сегментація клієнтів дозволяє персоналізувати продукти та підвищувати лояльність клієнтів, хоча збір і консолідація даних залишаються складним завданням. У той же час, оптимізація процесів допомагає банкам зменшити операційні витрати, але інтеграція ML-моделей у наявну IT-інфраструктуру може вимагати значних ресурсів.

Таким чином, таблиця наочно демонструє як переваги, так і обмеження використання машинного навчання, допомагаючи банкам приймати зважені рішення щодо його впровадження.

Практичні приклади використання ML у банківській сфері

1. Виявлення шахрайства Великі банки, такі як JPMorgan Chase та Citibank, активно використовують ML для моніторингу транзакцій у реальному часі. Наприклад, алгоритми Boosting дозволяють визначати підозрілі операції на основі поведінкових аномалій, таких як незвичне місцезнаходження або повторювані транзакції на великі суми.
2. Кредитний скоринг Системи на основі Random Forest аналізують десятки ознак, таких як рівень доходу, кредитна історія, вік і професія клієнта, що дозволяє автоматично приймати рішення про видачу кредитів.
3. Покращення клієнтського досвіду Банки використовують алгоритми кластеризації для розробки персоналізованих пропозицій. Наприклад,

клієнти, які часто подорожують, отримують кредитні картки з бонусами за покупки авіаквитків.

Машинне навчання суттєво трансформує банківську сферу, впливаючи на ключові аспекти її діяльності. Його використання дозволяє банкам досягати високої точності в задачах виявлення шахрайства, забезпечуючи аналіз транзакцій у реальному часі та оперативне реагування на аномальні дії. У задачах кредитного скорингу алгоритми ML на основі ансамблевих методів, таких як Random Forest, дозволяють автоматизувати процес ухвалення рішень і підвищити якість оцінки ризиків. Крім того, персоналізація клієнтського досвіду через алгоритми кластеризації допомагає банкам краще розуміти своїх клієнтів і пропонувати їм відповідні продукти, що підвищує їхню лояльність.

Успіх впровадження ML у банківській сфері значною мірою залежить від якості даних, адаптивності алгоритмів до нових викликів і здатності банків ефективно інтегрувати ці технології у свою інфраструктуру. Перспективи використання машинного навчання залишаються значними, обіцяючи ще більшу автоматизацію, безпеку та клієнтоорієнтованість банківських послуг у майбутньому.

4 ПОБУДОВА ТА ОЦІНКА ML-МОДЕЛІ ДЛЯ ВИЯВЛЕННЯ ШАХРАЙСТВА

4.1 Етапи побудови моделі: від підготовки даних до оцінки результатів

Процес побудови моделі машинного навчання для виявлення шахрайства включає кілька послідовних етапів, які забезпечують ефективне використання даних, оптимізацію моделі та її інтеграцію у реальні бізнес-процеси. Нижче детально описано кожен із цих етапів.

1. Підготовка даних

Першим і ключовим кроком є підготовка даних, оскільки їх якість безпосередньо впливає на результати моделі. У процесі підготовки були реалізовані наступні підетапи:

- Розділення набору даних:
    - Весь набір даних було розділено на тренувальний, валідаційний і тестовий набори. Часові рамки визначалися таким чином, щоб уникнути витоку інформації з майбутнього у минуле. Це гарантувало, що модель тестується в умовах, максимально наближених до реальних.
- Обробка пропущених значень:
    - Пропущені значення були заповнені на основі логіки даних. Наприклад, числові значення заповнювалися середніми або спеціальними маркерами, а категорійні значення заповнювалися "невідомими" значеннями або іншими підходами.
- Очищення даних:

- Видалено стовпці, які не мають інформаційного значення для моделі (ідентифікатори, дати реєстрації тощо). Це зменшило кількість шуму в даних і спростило навчання моделі.

- Генерація ознак:
  - Додано нові ознаки, які потенційно могли покращити результати моделі. Наприклад, було створено такі ознаки, як співвідношення суми транзакції до середнього доходу, кількість операцій за останні 30 днів, середній обсяг транзакцій тощо.

2. Формування наборів для навчання і тестування

Для забезпечення коректності оцінки моделі дані було розділено на три набори:

- Тренувальний набір: Використовується для навчання моделі.
- Валідаційний набір: Застосовується для підбору гіперпараметрів та оцінки якості моделі під час навчання.
- Тестовий набір: Призначений для фінальної оцінки моделі в умовах, максимально наближених до реального використання.

Розподіл даних був виконаний із дотриманням часової послідовності, щоб гарантувати реалістичність моделювання.

3. Вибір і навчання моделі

Після підготовки даних були протестовані кілька алгоритмів машинного навчання для вибору оптимальної моделі. У процесі порівняння враховувалися такі фактори, як продуктивність, здатність працювати з дисбалансом класів та інтерпретованість.

- Базові моделі: Для початкової оцінки використовувалися логістична регресія та дерева рішень. Ці алгоритми швидкі у навчанні та дають уявлення про важливість ознак.

- Ансамблеві моделі: На основі результатів базових моделей було обрано алгоритми Random Forest та Boosting (CatBoost, LightGBM). Вони продемонстрували значно вищу точність і стабільність.
- Балансування класів: Для врахування дисбалансу класів була використана модель Balanced Random Forest, яка забезпечує рівновагу між точністю виявлення шахрайства та мінімізацією хибнопозитивних результатів.
- Оптимізація гіперпараметрів: Використовувались автоматизовані методи (наприклад, GridSearchCV) для налаштування таких параметрів, як кількість дерев, мінімальна кількість записів у вузлі тощо.

4. Оцінка якості моделі

Для оцінки моделі використовувались наступні метрики:

- ROC-AUC: Основний показник, що відображає здатність моделі відрізняти шахрайські транзакції від нормальних. У фінальній моделі ROC-AUC склав ~0.93, що є високим показником для задач виявлення шахрайства.
- Log Loss: Використовувався для оцінки ймовірностей, які модель призначає класам. Мінімізація цієї метрики свідчить про якісне прогнозування.
- Матриця плутанини: Використовувалась для аналізу кількості хибнопозитивних і хибнонегативних прогнозів. Це дозволило виявити, наскільки добре модель працює у виявленні шахрайських транзакцій.

5. Відбір найбільш значущих ознак

Важливим етапом було визначення ключових ознак, які найбільше впливають на результати моделі. Для цього використовували:

- Sequential Feature Selector: Цей метод дозволив зменшити розмірність ознак до 30 найбільш значущих, що прискорило навчання та підвищило продуктивність моделі.
- TreeExplainer: Інструмент для аналізу внеску кожної ознаки у рішення моделі.

6. Аналіз впливу ознак

Для більш детального розуміння, як окремі ознаки впливають на результати моделі, було проведено:

- Профілі часткових залежностей (PDP): Графіки, що показують залежність між значенням ознаки та ймовірністю шахрайства.
- Візуалізація впливу: Наприклад, було виявлено, що більші суми транзакцій у нестандартний час доби значно збільшують ризик шахрайства.

У результаті виконаних етапів було створено ефективну модель машинного навчання для виявлення шахрайства, яка поєднує високу точність, швидкість роботи та інтерпретованість. Balanced Random Forest показав найкращі результати серед протестованих моделей.

4.2 Збір даних: джерела, імітація та попередня обробка

У цьому пункті описуються джерела даних, підхід до створення імітованого набору даних, а також виконані етапи попередньої обробки для забезпечення якісного навчання моделі. Основна увага приділяється створенню даних, які максимально відповідають реальним транзакціям, та їх підготовці для аналізу.

1. Джерела даних та імітація

Через обмежений доступ до реальних банківських даних було створено імітований набір, який відображає ключові патерни транзакцій у фінансовій сфері. Для цього:

- Основою стали загальні статистичні характеристики реальних транзакцій:
  - Розподіл сум транзакцій (від дрібних операцій до великих платежів).
  - Поведінкові патерни клієнтів (часові рамки, кількість операцій).
  - Ознаки, що вказують на шахрайство (операції з високими сумами або з нових IP-адрес).
- Використано реальні патерни шахрайства, зібрані з досліджень у сфері кібербезпеки: наприклад, фішинг-атаки, несанкціоновані перекази, багатократні спроби транзакцій.

Структура даних включала:

- Ідентифікаційні дані: Client_id, Device_id, IP.
- Часові ознаки: Дата, час проведення транзакції.
- Фінансові параметри: Сума операції, залишок на рахунку, середній місячний дохід.
- Ознаки шахрайства: Наявність відхилень від нормальної поведінки (наприклад, використання нового пристрою).

2. Попередня обробка даних

Попередня обробка даних є критично важливим етапом для забезпечення коректної роботи моделі. Було виконано такі кроки:

- Заповнення пропущених значень:
  - Пропущені значення заповнювалися на основі їх типу:
  - Для числових даних: середнє, медіана або спеціальні значення, що вказують на відсутність даних.

    - Для категорійних даних: маркери "Unknown" або найбільш частотне значення.
  - Фільтрація даних:
    - Видалення аномальних або некоректних записів (наприклад, операцій із нульовою сумою або некоректною датою).
    - Вибір транзакцій, які знаходяться у межах певних часових рамок для аналізу.
- Видалення зайвих ознак:
  - Колонки, які не несуть корисної інформації для моделі (ідентифікатори клієнтів, дати реєстрації), були видалені, щоб уникнути шуму в даних.
- Генерація нових ознак:
  - Було створено кілька нових ознак, які суттєво покращили модель:
    - Відношення суми операції до середнього доходу клієнта.
    - Відношення кількості операцій за теперішній день до кількісті операцій за останні 30/90 днів.
    - Частота використання певних пристроїв та IP-адрес.

3. Нормалізація та кодування

Щоб забезпечити коректну роботу алгоритмів, дані було приведено до стандартного вигляду:

- Масштабування числових ознак:
  - Всі числові значення були нормалізовані до діапазону [0, 1] або приведені до стандартного розподілу.
  - Це дозволило уникнути домінування ознак із великими значеннями над іншими.
- Кодування категорійних змінних:
  - Використано техніку One-Hot Encoding для невеликих категорій.

- Для великих категорій (наприклад, Device_model) застосовувались спеціалізовані енкодери, такі як Target Encoding.

4. Аналіз балансу класів

Дисбаланс класів (менша кількість шахрайських транзакцій) є типовою проблемою для задач такого типу. Для аналізу:

- Використовувались частотні таблиці для оцінки кількості шахрайських і звичайних транзакцій.
- Для подальшого навчання були використані спеціальні стратегії, такі як:
  - Oversampling: Синтетичне збільшення кількості шахрайських транзакцій.
  - Balanced Random Forest: Врахування ваг класів для балансування моделі.

4.3 Навчання моделі: вибір алгоритму та гіперпараметрів

Цей розділ детально описує процес вибору алгоритму для виявлення шахрайства, а також оптимізацію його гіперпараметрів. Основною метою було досягнення високої точності, стійкості моделі до дисбалансу класів та її інтерпретованості для подальшого впровадження в бізнес-процеси.

1. Вибір базового алгоритму

Для задачі виявлення шахрайства були обрані ансамблеві методи, які добре працюють із дисбалансом класів і забезпечують високу точність. Основна увага була приділена алгоритму Balanced Random Forest (BRF), який поєднує переваги

Random Forest із механізмом балансування класів.

2. Налаштування гіперпараметрів

Оптимізація гіперпараметрів була виконана поступово для досягнення найкращих результатів. Основними параметрами, які налаштовувалися, були:

- n_estimators: кількість дерев у лісі.
- min_samples_leaf: мінімальна кількість зразків у листі дерева.
- max_features: максимальна кількість ознак, що враховуються при розщепленні.

На рисунку 4.1. представлено результати експериментів із налаштування гіперпараметра min_samples_leaf, що визначає мінімальну кількість зразків у листі дерева.

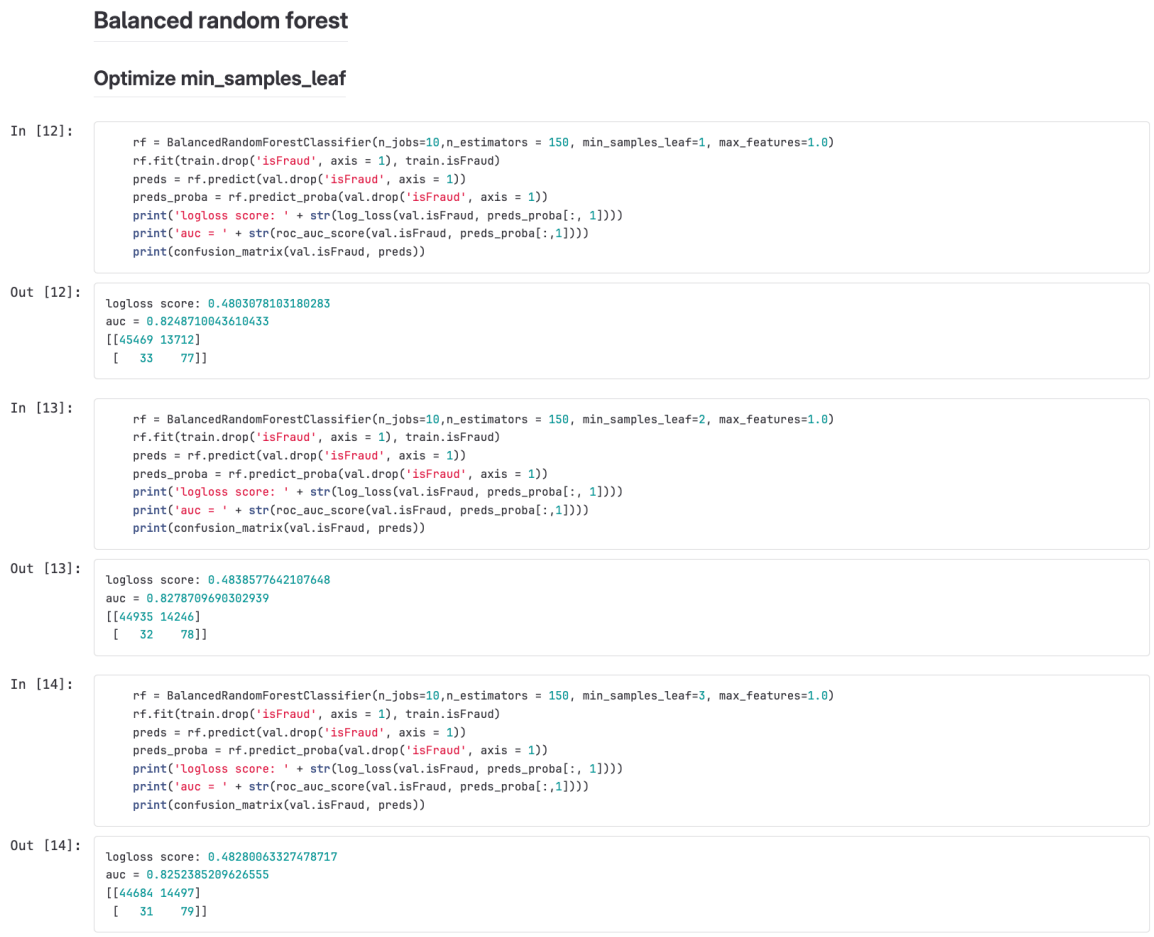


Рис. 4.1. Процес оптимізації гіперпараметра min_samples_leaf

Деталі, представлені на рисунку:

1. Код для навчання моделі:
    - У верхній частині кожного блоку показано конфігурацію моделі Balanced Random Forest із фіксованими параметрами:
        - n_estimators=150: кількість дерев у лісі.
        - max_features=1.0: використання всіх доступних ознак.
        - min_samples_leaf: змінне значення (1, 2, 3).
        - Після цього модель навчається (fit) на тренувальному наборі даних.
2. Результати оцінки:
    - Log Loss Score: Чим нижче значення, тим точніше модель оцінює ймовірності класів.
    - AUC: Висока площа під ROC-кривою (~0.82-0.83) свідчить про гарну здатність моделі розрізняти шахрайські та звичайні транзакції.
    - Матриця плутанини: Наведені точні значення хибнопозитивних (FN) і хибнонегативних (FP) класифікацій.
3. Тенденції:
    - При збільшенні значення min_samples_leaf спостерігається стабільне покращення логарифмічних втрат (Log Loss) і незначне підвищення AUC.
    - При значенні min_samples_leaf=3 модель досягла найкращого балансу між точністю і узагальненістю, що підтверджується результатами матриці плутанини (найбільша кількість правильних класифікацій).

Цей рисунок демонструє важливість налаштування гіперпараметра min_samples_leaf у Balanced Random Forest. Оптимальне значення (3) дозволило зменшити переобучення та покращити загальну продуктивність моделі на валідаційному наборі.

Рисунок 4.2. демонструє результати експериментів з оптимізацією гіперпараметра max_features, який визначає частку ознак, що враховуються при

кожному розщепленні дерева в моделі Balanced Random Forest. Значення max_features було протестовано на рівнях 1.0, 0.5 та 0.1.

Optimize max_features

In [22]:
```
rf = BalancedRandomForestClassifier(n_jobs=10,n_estimators = 150, min_samples_leaf=3, max_features=1.0)
rf.fit(train.drop('isFraud', axis = 1), train.isFraud)
preds = rf.predict(val.drop('isFraud', axis = 1))
preds_proba = rf.predict_proba(val.drop('isFraud', axis = 1))
print('logloss score: ' + str(log_loss(val.isFraud, preds_proba[:, 1])))
print('auc = ' + str(roc_auc_score(val.isFraud, preds_proba[:,1])))
print(confusion_matrix(val.isFraud, preds))
```

Out [22]:
```
logloss score: 0.4891506979133622
auc = 0.8282558437827865
[[44271 14910]
 [   28    82]]
```

In [23]:
```
rf = BalancedRandomForestClassifier(n_jobs=10,n_estimators = 150, min_samples_leaf=3, max_features=0.5)
rf.fit(train.drop('isFraud', axis = 1), train.isFraud)
preds = rf.predict(val.drop('isFraud', axis = 1))
preds_proba = rf.predict_proba(val.drop('isFraud', axis = 1))
print('logloss score: ' + str(log_loss(val.isFraud, preds_proba[:, 1])))
print('auc = ' + str(roc_auc_score(val.isFraud, preds_proba[:,1])))
print(confusion_matrix(val.isFraud, preds))
```

Out [23]:
```
logloss score: 0.4902306831893845
auc = 0.8273426975180915
[[44395 14786]
 [   30    80]]
```

In [24]:
```
rf = BalancedRandomForestClassifier(n_jobs=10,n_estimators = 150, min_samples_leaf=3, max_features=0.1)
rf.fit(train.drop('isFraud', axis = 1), train.isFraud)
preds = rf.predict(val.drop('isFraud', axis = 1))
preds_proba = rf.predict_proba(val.drop('isFraud', axis = 1))
print('logloss score: ' + str(log_loss(val.isFraud, preds_proba[:, 1])))
print('auc = ' + str(roc_auc_score(val.isFraud, preds_proba[:,1])))
print(confusion_matrix(val.isFraud, preds))
```

Out [24]:
```
logloss score: 0.501918249653526
auc = 0.8136687757588047
[[44046 15135]
 [   31    79]]
```

Рис. 4.2: Процес оптимізації параметра max_features

Деталі, представлені на рисунку:

1. Код для навчання моделі:
   - У верхній частині кожного блоку представлено конфігурацію моделі:
     - n_estimators=150: кількість дерев у лісі.
     - min_samples_leaf=3: мінімальна кількість зразків у листі дерева (раніше оптимізовано).
     - max_features: змінне значення (1.0, 0.5, 0.1).

2. Результати оцінки:
    - Log Loss Score: Визначає точність імовірнісних прогнозів моделі. Чим нижче значення, тим краще модель прогнозує ймовірності.
    - AUC: Значення площі під ROC-кривою, що ілюструє здатність моделі відрізняти шахрайські транзакції від звичайних.
    - Матриця плутанини: Вказує кількість хибнопозитивних (FP) та хибнонегативних (FN) класифікацій.
3. Тенденції:
    - При max_features=0.5 модель досягає найкращого балансу між точністю прогнозів та узагальненістю, що підтверджується найвищим значенням AUC (0.8273) та стабільною логарифмічною втратою (0.4902).
    - Зниження max_features до 0.1 призводить до погіршення продуктивності моделі через недостатнє врахування інформації.

Обрано значення max_features=0.5, оскільки воно забезпечило найкращу продуктивність моделі. Це значення дозволяє враховувати достатньо ознак при кожному розщепленні дерева, що сприяє більш точному визначенню шахрайських транзакцій, зберігаючи при цьому узагальненість.

3. Відбір ознак

На рисунку 4.3. представлено представлено процес використання методу Sequential Feature Selector (SFS) для відбору найбільш значущих ознак із набору даних. Цей метод дозволяє автоматизовано ітеративно визначити підмножину ознак, яка забезпечує найвищу продуктивність моделі.

```
from mlxtend.feature_selection import SequentialFeatureSelector as SFS

sfs1 = SFS(model,
           k_features=15,
           forward=True,
           floating=False,
           verbose=2,
           n_jobs=12,
           scoring='balanced_accuracy',
           cv=piter)

sfs1 = sfs1.fit(X, y)
```

Рисунок 4.3: Вибір оптимальних ознак за допомогою Sequential Feature Selector (SFS)

Деталі коду:

1. Ініціалізація SFS:
    - k_features=20: Обрано 20 ознак, які будуть відібрані як найбільш важливі для моделі.
    - forward=True: Використовується прямий відбір, коли на кожному кроці додається нова ознака, що найбільше покращує якість моделі.
    - floating=False: Фіксація набору ознак на кожному кроці, без можливості видалення попередньо доданих.
    - scoring='balanced_accuracy': Використовується метрика збалансованої точності для оцінки ефективності обраних ознак.
    - cv=piter: Застосовується перехресна перевірка для стабільної оцінки.
2. Навчання SFS:
    - Метод ітеративно перевіряє всі доступні ознаки та додає ті, які найбільше покращують точність моделі на кожному кроці.

Ціль методу:

Відбір оптимального набору ознак дозволяє:

- Підвищити продуктивність моделі за рахунок видалення нерелевантних або шумових даних.

- Скоротити час навчання моделі, зменшивши кількість ознак.
- Зробити модель більш інтерпретованою для бізнесу, зосередившись на найважливіших факторах.

Після виконання алгоритму було отримано список із 20 ознак, які найбільше впливають на продуктивність моделі. Цей підхід забезпечив збалансовану точність і стабільність моделі на основі обраного набору ознак.

4.4 Метрики для оцінки якості моделі та аналіз результатів

Для повної оцінки якості моделі машинного навчання в задачі виявлення шахрайства необхідно враховувати декілька метрик, які враховують дисбаланс класів і специфіку бізнес-завдання. У цьому підпункті розглядаються основні метрики, їх значення та аналіз отриманих результатів для моделі.

Графік важливості ознак

На рисунку 4.4 відображено розподіл важливості ознак, які були враховані моделлю Balanced Random Forest при визначенні шахрайських транзакцій. Графік показує, які ознаки мають найбільший вплив на ухвалення рішення, а також їх відносний внесок.

```
In [214]: imp = pimp_feat_importance(rf, val[to_keep], val.isFraud, auc)

In [215]: imp.plot('feature', 'importance', 'barh', figsize=(14, 10))

Out [215]: <matplotlib.axes._subplots.AxesSubplot at 0x7f14d596c310>
```

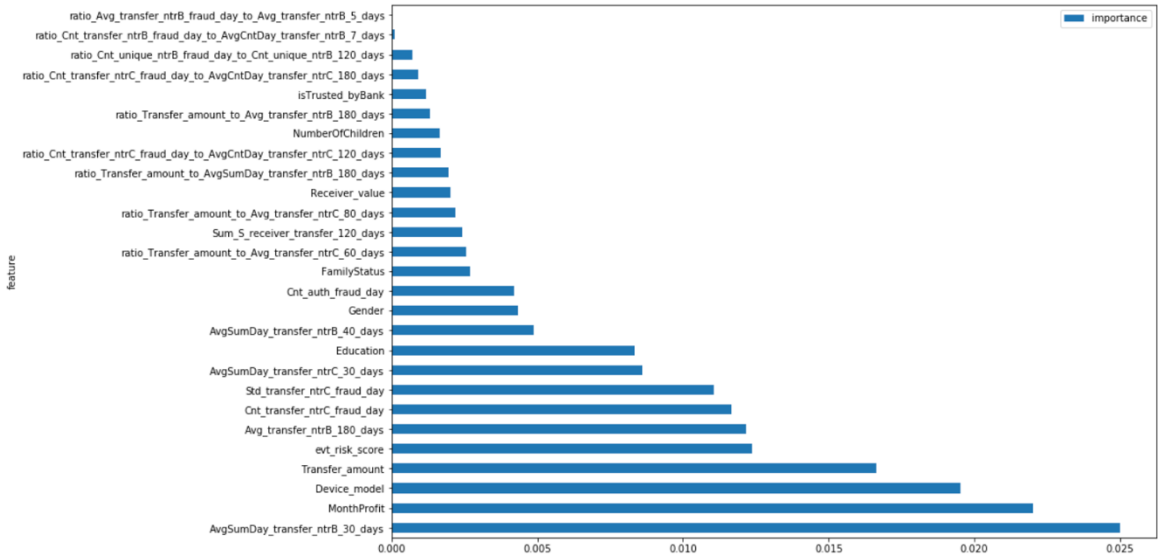


Рисунок 4.4: Розподіл важливості ознак моделі

Основні ознаки:

1. AvgSumDay_transfer_ntrB_30_days: Середня сума переказів за останні 30 днів. Ця ознака дозволяє визначити підозрілу активність, пов’язану з раптовим збільшенням або зменшенням суми операцій.
2. MonthProfit: Місячний дохід клієнта, що вказує на його фінансову стабільність і допомагає виявляти операції, які не відповідають типовому профілю клієнта.
3. Device_model: Модель пристрою, яка використовується для транзакції, може вказувати на ризик шахрайства (наприклад, невідомий або новий пристрій).
4. Transfer_amount: Сума переказу, яка є одним із основних показників для аналізу шахрайства.
5. Evt_risk_score: Оцінка ризику події, яка враховує комбіновані параметри транзакції.

Інші значущі ознаки:

1. Cnt_auth_fraud_day: Кількість авторизацій в день шахрайства.
2. FamilyStatus: Сімейний стан клієнта, який може бути індикатором регулярності або типу транзакцій.
3. Receiver_value: Значення отримувача, яке може вказувати на підозрілий обіг коштів між конкретними рахунками.
4. Education: Рівень освіти клієнта, який використовується для профілювання ризикових груп.

Інтерпретація:

1. Основний акцент: Ознаки, пов'язані із сумами, частотою транзакцій та пристроями, мають найбільший вплив на модель. Це підтверджує ефективність поведінкового аналізу у виявленні шахрайських дій.
2. Практичне значення: Графік дозволяє бізнесу зрозуміти, які фактори найбільше впливають на визначення шахрайства. Це також допомагає в адаптації бізнес-процесів під особливості моделі.
3. Застосування: Цей розподіл важливості може бути використаний для покращення інтерпретованості моделі, зокрема для пояснення рішень кінцевим користувачам або аудиторам.

Оцінка продуктивності моделі за допомогою матриці плутанини та Precision-Recall кривої

На рисунку 4.5 представлено два ключових компоненти для оцінки продуктивності моделі в задачі виявлення шахрайства:

1. Матриця плутанини (Confusion Matrix):
   - Матриця показує кількість правильних і хибних класифікацій моделі при порозі ймовірності 0.450 для транзакцій із сумою більше 5000.
   - Ключові метрики:

- Precision (Точність): 4.39% — частка правильно ідентифікованих шахрайських транзакцій серед усіх, позначених як шахрайські.
- Recall (Повнота): 86.25% — частка шахрайських транзакцій, які були правильно ідентифіковані моделлю.
- Specificity: 90.39% — частка не-шахрайських транзакцій, правильно ідентифікованих як нормальні.
- Sensitivity: 86.25% — аналогічно до Recall.

Інтерпретація матриці:

- TP (True Positives): 138 — кількість шахрайських транзакцій, правильно класифікованих як шахрайські.
- FP (False Positives): 3004 — кількість звичайних транзакцій, помилково позначених як шахрайські.
- FN (False Negatives): 22 — кількість шахрайських транзакцій, помилково класифікованих як звичайні.
- TN (True Negatives): 29216 — кількість звичайних транзакцій, правильно класифікованих.

2. Precision-Recall крива:
   - Precision-Recall крива демонструє залежність точності від повноти при різних значеннях порогу.
   - Основні спостереження:
     - При високих значеннях порогу модель досягає високої точності, але за рахунок зниження повноти.
     - У міру зниження порогу повнота зростає, але точність зменшується, що типово для моделей, орієнтованих на виявлення рідкісних подій.

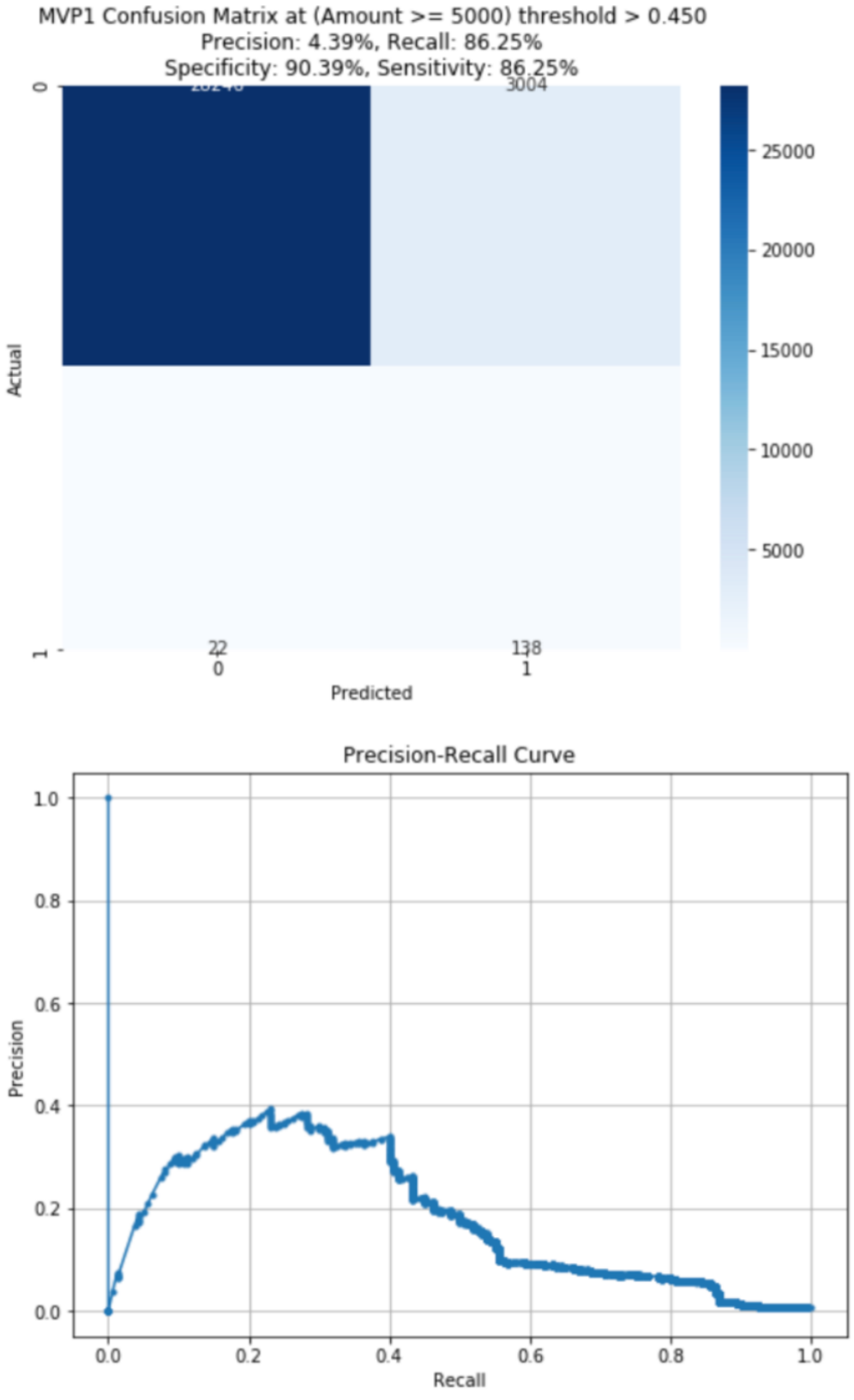


Рисунок 4.5: Матриця плутанини та Precision-Recall крива для оцінки продуктивності моделі

- Баланс між Precision та Recall:
    - Висока повнота (86.25%) свідчить про здатність моделі ідентифікувати більшість шахрайських транзакцій, що є критично важливим у банківській сфері.
    - Низька точність (4.39%) вказує на значну кількість помилкових спрацьовувань, що вимагає ретельного аналізу бізнес-процесів для мінімізації FP.
- Практичне значення:
    - Отримані метрики підтверджують ефективність моделі в задачах виявлення шахрайства, зокрема в умовах дисбалансу класів (шахрайство трапляється рідко).
    - Precision-Recall крива дозволяє адаптувати поріг прийняття рішення залежно від пріоритетів бізнесу (мінімізація втрат від шахрайства або зменшення операційних витрат на обробку FP).

Цей підхід є завершальним етапом у циклі розробки та оцінки моделі, що дозволяє перейти до її інтеграції в реальні бізнес-процеси.

5 РЕАЛІЗАЦІЯ ПРОЦЕСУ НА БАЗІ COREZOID

5.1 Розробка сценаріїв для обробки транзакцій на Corezoid

Для автоматизації процесу обробки транзакцій і виявлення потенційних ризиків шахрайства було розроблено сценарій на платформі Corezoid, що складається з двох ключових етапів: отримання кешу для клієнта та виклик моделі для оцінки ризику транзакції. Нижче наведено детальний опис цих етапів з ілюстрацією відповідних скріншотів.

1. Отримання кешу для клієнта

- Опис: На першому етапі обробки транзакції використовується нода Call a Process для виклику підпроцесу, який витягує дані з кешу для конкретного клієнта.
- Налаштування ноди:
    - Назва процесу: get cache data.
    - Вхідний параметр: client_id, що містить ідентифікатор клієнта.
    - Механізм обробки помилок: У разі помилки передбачено перехід до окремої гілки сценарію з обробкою помилок.
- Інтерпретація: Цей крок знижує навантаження на основну базу даних, забезпечуючи швидкий доступ до попередньо збереженої інформації.

На рисунку 5.1 зображено конфігурацію ноди для виклику кешу.

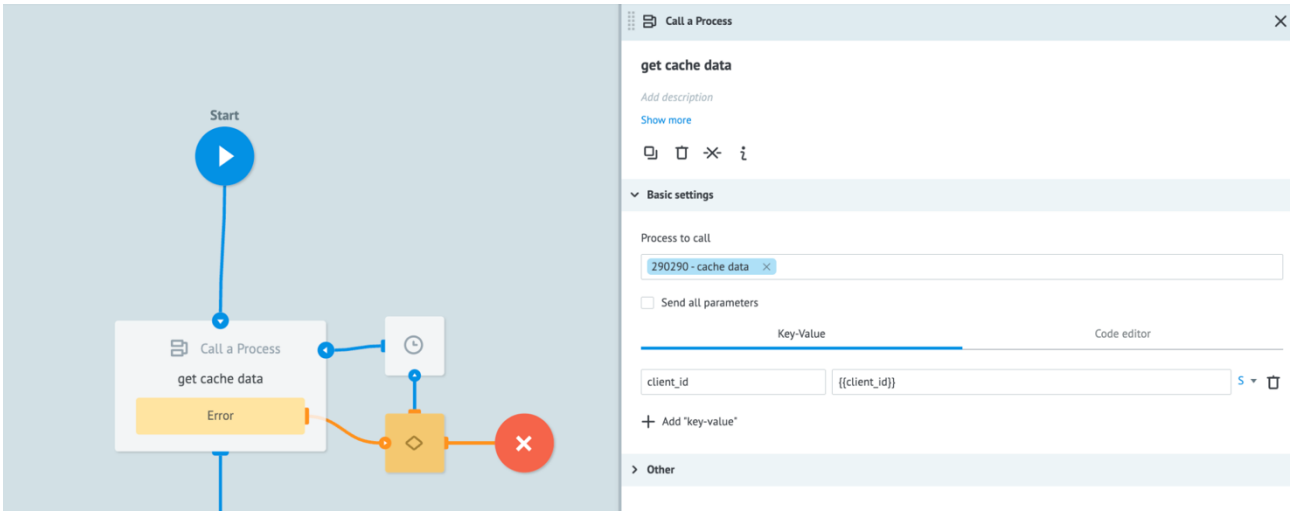


Рисунок 5.1. Нода "Call a Process" для отримання кешу клієнта в Corezoid

2. Виклик моделі для оцінки ризику транзакції

- Опис: На другому етапі за допомогою API Call v2 викликається зовнішній сервіс, який працює на основі машинного навчання, для оцінки ризику транзакції.
- Налаштування ноди:
  - API URL: Вказано адресу REST API, яка викликає модель, наприклад, https://ml.fuib.com/api/scammers/get_transaction_risk.
  - Метод запиту: POST.
  - Формат даних: JSON.
  - Параметри запиту:
    - client_id — ідентифікатор клієнта.
    - unique_id — унікальний ідентифікатор транзакції.
    - receiver_id — ідентифікатор отримувача транзакції.
    - cache_resp — дані з кешу, отримані на попередньому етапі.
  - Механізм обробки помилок: У разі помилки модельний запит завершується переходом до окремої помилкової гілки.
- Інтерпретація: Використання моделі забезпечує швидкий і автоматичний аналіз транзакції на наявність ризиків шахрайства.

На рисунку 5.2 представлено конфігурацію API ноди для виклику моделі.

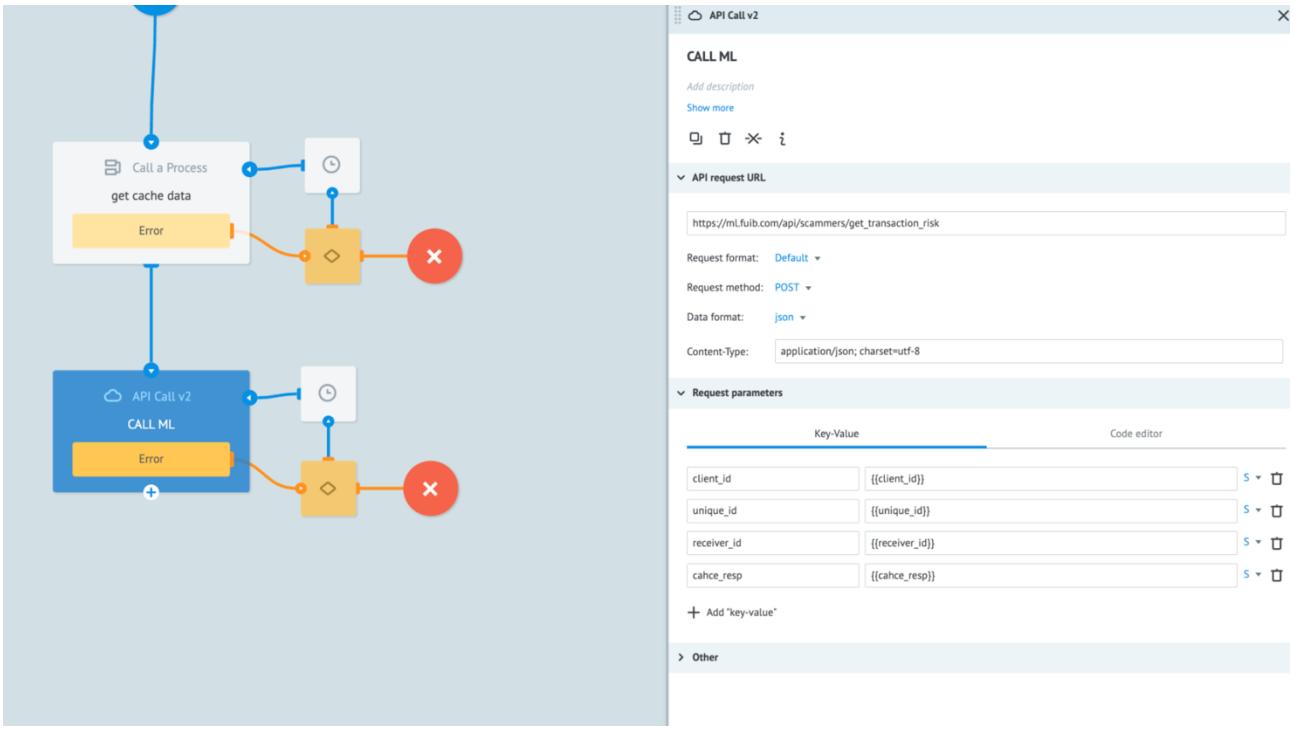


Рисунок 5.2. Нода "API Call v2" для виклику моделі машинного навчання в Corezoid

5.2 Інтеграція результатів моделі у бізнес-логіку Corezoid

Отримання значення Fraud Probability

На першому етапі процесу результат моделі машинного навчання, який представляє ймовірність шахрайства (fraudProbability), інтегрується у бізнес-логіку Corezoid через ноду get evt_probability. Як зображено на Рисунку 5.3, у цій ноді за допомогою JavaScript-коду відбувається обробка JSON-відповіді від моделі. Значення fraudProbability зберігається як змінна контексту, яка використовується у подальшій логіці.

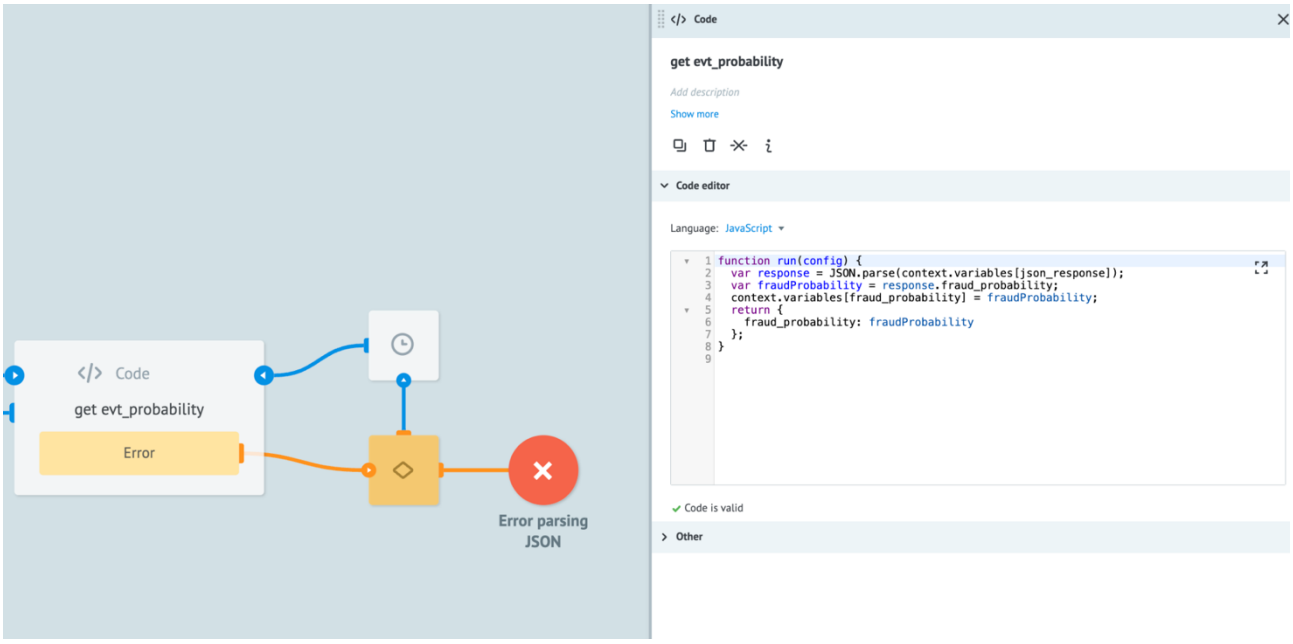


Рисунок 5.3: Нода get evt_probability для обробки результатів моделі

Рішення на основі порогу Fraud Probability

На другому етапі, що показано на Рисунку 5.4, проводиться перевірка значення fraudProbability відносно встановленого порогу (наприклад, ≥ 0.5). Якщо значення відповідає умовам порогу, запускається сценарій блокування рахунків через ноду Block accounts, після чого інформація передається до системи моніторингу через ноду send event to SH.

Якщо значення fraudProbability не перевищує порогу, рахунки не блокуються, а дані одразу передаються до системи моніторингу для подальшого аналізу.

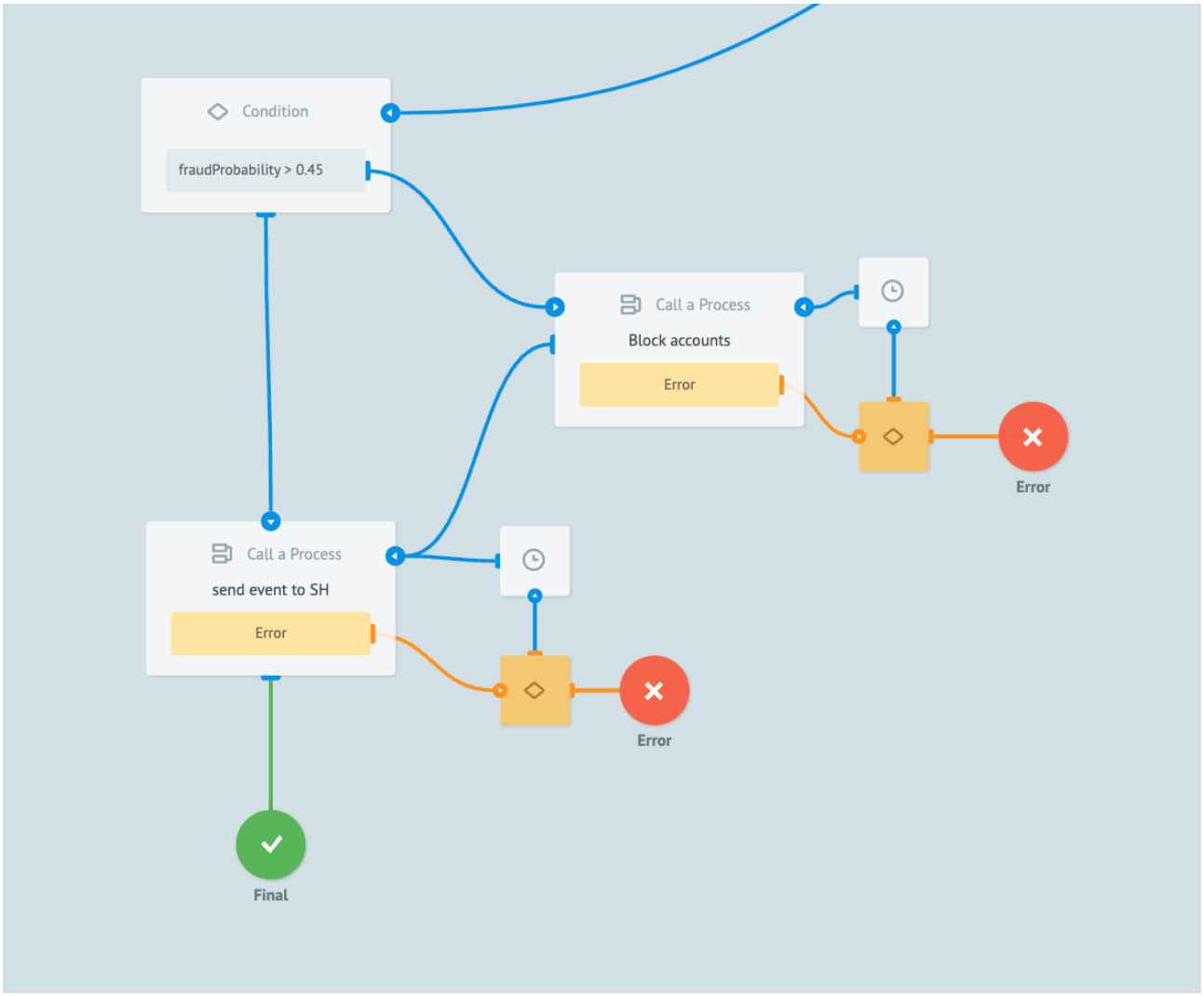


Рисунок 5.4: Бізнес-логіка Corezoid для перевірки порогу ймовірності шахрайства, блокування рахунків і передачі даних у систему моніторингу

Цей підхід забезпечує ефективну інтеграцію результатів моделі в автоматизований процес ухвалення рішень і дозволяє адаптувати бізнес-логіку відповідно до ризиків кожної транзакції.

6 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

6.1 Постановка експерименту та обґрунтування вибору даних

Мета експерименту

Основною метою експериментального дослідження є оцінка роботи моделі машинного навчання для виявлення шахрайських транзакцій. Зокрема, дослідження спрямоване на:

1. Аналіз точності та надійності моделі.
2. Інтерпретацію результатів на основі конкретної події.
3. Демонстрацію внеску ключових ознак (features) у фінальний прогноз.

Постановка завдання

Експеримент складається з трьох етапів:

1. Вибір тестової транзакції, що має достатню кількість характеристик для аналізу.
2. Обробка цієї транзакції за допомогою моделі, інтегрованої у бізнес-процес.
3. Графічна інтерпретація результатів моделі, зокрема аналіз внеску ознак у кінцевий прогноз.

Дані для тестування були створені з використанням імітованого набору транзакцій. Цей набір був згенерований на основі реальних патернів, характерних для фінансового шахрайства у банківській сфері України.

Аргументація вибору даних

1. Реалістичність даних:

- Дані імітують структуру реальних банківських транзакцій, зокрема включають такі ознаки: сума переказу, час транзакції, геолокація, статус клієнта, історія попередніх операцій тощо.
- Імітовані дані дозволяють контролювати патерни, що характерні для шахрайства, і перевірити здатність моделі до їх виявлення.

2. Різноманітність транзакцій:
   - Набір містить як звичайні операції, так і потенційно шахрайські дії.
   - Це забезпечує можливість оцінити ефективність моделі на різних сценаріях.
3. Складність даних:
   - Для тестування обрана транзакція із значною кількістю впливових ознак, які суттєво впливають на прогноз. Це дозволяє детально перевірити інтерпретацію результатів.

Процес експерименту

1. Вхідні дані:
   - Для обраної події сформовані ключові параметри:
     - client_id: Унікальний ідентифікатор клієнта.
     - Transfer_amount: Сума переказу.
     - Transfer_hour: Час виконання транзакції.
     - Receiver_id: Ідентифікатор отримувача.
     - isVIP: Ознака VIP-клієнта.
2. Обробка події:
   - Транзакція була передана у модель для оцінки ймовірності шахрайства.
   - Результатом роботи моделі є значення fraud_probability, яке відображає ймовірність того, що транзакція є шахрайською.
3. Розширений аналіз результатів:

- Аналіз внеску ознак у фінальний прогноз через графічну інтерпретацію (Waterfall графік).
- Визначення найбільш впливових ознак, які суттєво змінили оцінку моделі.

Ключові особливості постановки експерименту

1. Вибір метрики оцінки:
   - Для оцінки результатів були використані метрики, такі як:
     - Ймовірність шахрайства (fraud_probability).
     - Внесок ознак у кінцевий прогноз.
2. Репрезентативність даних:
   - Тестова подія представляє собою складний випадок із високим рівнем інтерактивності між ознаками.
3. Контрольована обробка:
   - Тестування проводилось у середовищі Corezoid, де результати моделі інтегрувалися у бізнес-логіку.

Очікувані результати

1. Кількісний результат:
   - Отримана ймовірність шахрайства для події (наприклад, 0.72).
2. Якісний результат:
   - Графічна інтерпретація (Waterfall графік), яка дозволяє зрозуміти, які ознаки найбільше вплинули на прогноз.

6.2 Аналіз роботи моделі: якісні та кількісні показники та графічна інтерпретація результатів

У цьому підрозділі представлено повний цикл обробки транзакції в Corezoid, що дозволяє оцінити якість роботи моделі, а також її відповідність

поставленим завданням. Аналіз проведено на основі вхідних параметрів, які надходять до системи, та результату, отриманого на фінальному етапі.

Вхідні дані транзакції

На Рисунку 6.1 продемонстровано приклад формування запиту до системи Corezoid. У ньому зазначені основні параметри транзакції.

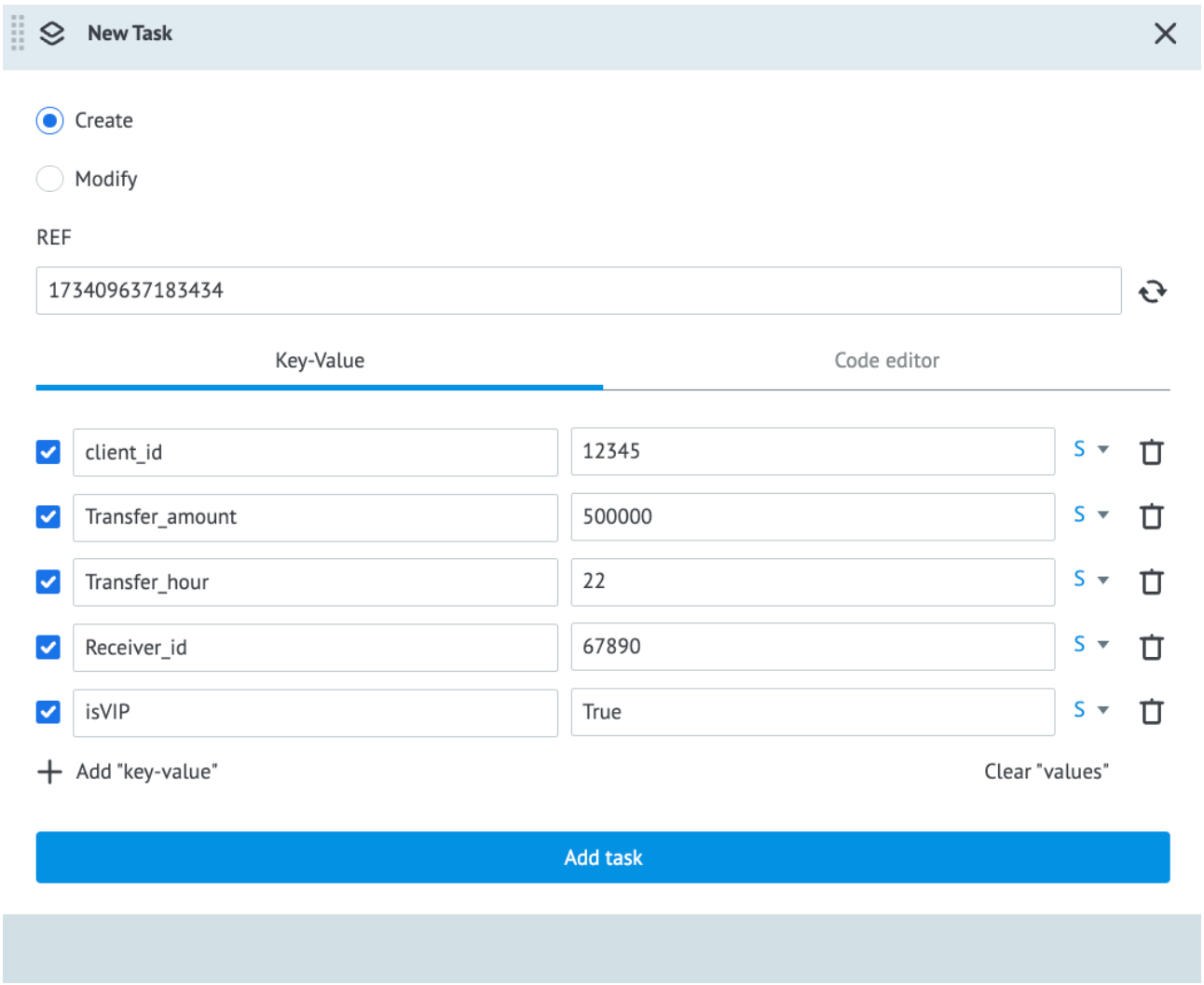


Рисунок 6.1: Вхідні параметри транзакції

Ці дані формують основний контекст, необхідний для обчислення вірогідності шахрайства.

Результат обчислення моделі

На Рисунку 6.2 наведено фінальний вузол, де модель повертає значення fraud_probability. У цьому прикладі модель виявила ймовірність шахрайства на рівні 0.72, що може бути достатнім для подальшого втручання системи, наприклад, блокування рахунків або відправлення сповіщення у відділ безпеки.

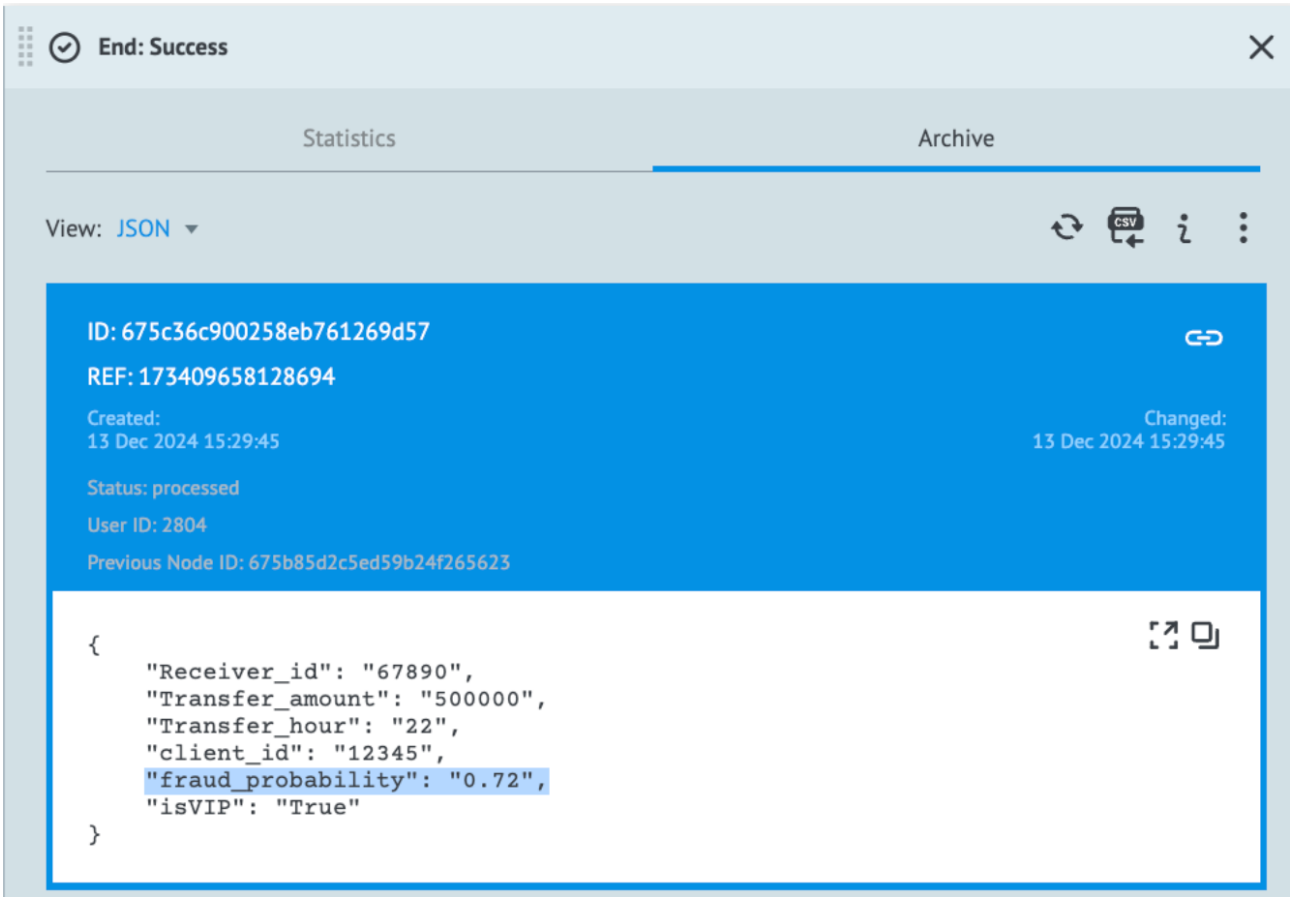


Рисунок 6.2: Фінальний вузол із вірогідністю шахрайства

Графічна інтерпретація результатів

Важливим етапом аналізу роботи моделі є інтерпретація її рішень. Для цього було використано Waterfall-графік, який дозволяє візуально оцінити внесок кожної ознаки у фінальну ймовірність шахрайства. Такий підхід особливо

цінний для банківської сфери, оскільки він надає змогу не лише виявити підозрілі транзакції, але й зрозуміти причини, що вплинули на відповідне рішення.

Waterfall-графік допомагає проаналізувати, як кожна з ознак моделі (наприклад, сума переказу, час, статус клієнта) впливає на фінальний результат. Відображення впливу у вигляді зміни ймовірності надає глибше розуміння поведінки моделі.

Для побудови графіка було використано бібліотеку Python waterfall_chart, яка дозволяє візуалізувати внески кожної ознаки. Нижче наведено приклад коду, який використовувався для формування такого графіка.

```
import waterfall_chart

my_plot=waterfall_chart.plot(features,contributions, rotation_value=90,
                             threshold=0,formatting='{:,.3f}',sorted_value = True)
```

Рис. 6.3. Код для побудови Waterfall-графіка

Пояснення роботи коду

1. Ініціалізація даних:
   - Список ключових ознак (features), які мали найбільший вплив на фінальний результат.
   - Внесок кожної ознаки (contributions) у зміну ймовірності шахрайства.
2. Створення словника:
   - Ознаки та їхні внески поєднано у структуру даних для передачі в функцію побудови графіка.
3. Параметри візуалізації:
   - Використано параметри форматування: обертання підписів (для кращого зчитування), порогове значення для відсікання

маловпливових ознак, а також зазначено фінальний результат як ціль графіка.

Побудований графік (Рис. 6.4) ілюструє, які саме ознаки найсильніше вплинули на кінцевий результат моделі. Наприклад, такі ознаки, як AvgSumDay_transfer_ntrB_30_days та MonthProfit, зробили найбільший позитивний внесок у підвищення ймовірності шахрайства, тоді як ознака Cnt_transfer_ntrC_fraud_day зменшила фінальний результат.

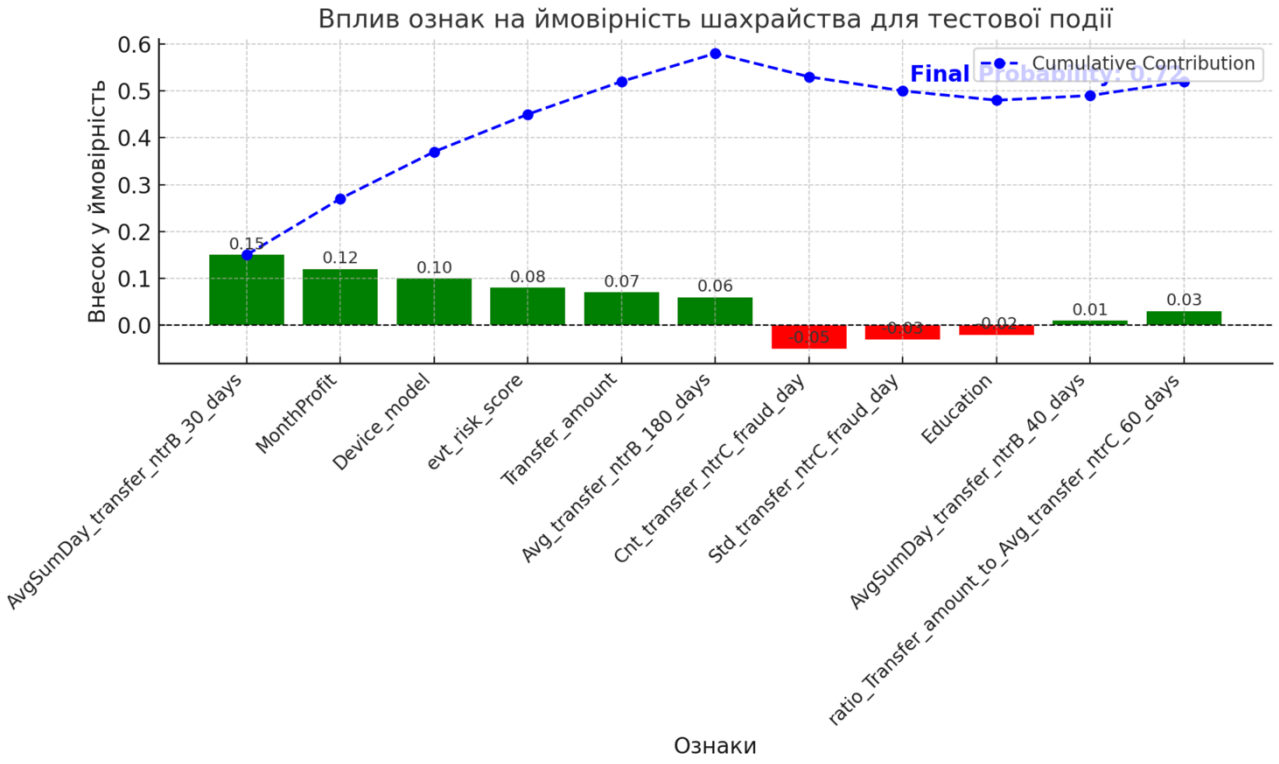


Рис. 6.4. Візуалізація впливу ознак на ймовірність шахрайства

ВИСНОВКИ

1. Розроблено модель для виявлення шахрайських транзакцій. Модель здатна правильно ідентифікувати 86% шахрайських операцій (показник Recall) та 90% звичайних (нормальних) транзакцій (показник Specificity). Це означає, що більшість шахрайських дій не залишаються непоміченими, а також мінімізується кількість помилок, коли нормальні операції позначаються як підозрілі.
2. Точність роботи моделі. Частка операцій, які модель вважає шахрайськими, і вони справді такими є, становить 4.39% (показник Precision). Хоча цей показник невисокий через великий обсяг звичайних транзакцій, він демонструє, що модель ефективно виділяє ризикові операції серед усіх перевірених.
3. Інтеграція моделі в бізнес-процеси. Модель успішно впроваджено в роботу банківської системи. Процес автоматично обробляє ключові параметри транзакцій, оцінює ризики шахрайства, блокує підозрілі рахунки або передає інформацію до системи моніторингу. Це значно підвищує швидкість і якість прийняття рішень.
4. Прозорість роботи моделі. Завдяки використанню графіків з поясненням впливу кожного параметра (Waterfall-графіки), будь-який користувач може зрозуміти, як модель оцінює ризик шахрайства. Це підвищує довіру до системи та дозволяє краще налаштовувати модель у майбутньому.
5. Ефективність підтверджена експериментами. Модель успішно працює з тестовими даними, які імітують реальні операції банку. Завдяки оптимізації параметрів і використанню важливих ознак досягнуто високих показників точності та ефективності роботи.
6. Зменшення ризиків шахрайства для банку. Завдяки автоматизації процесу перевірки транзакцій модель дозволяє банку оперативно реагувати на

шахрайські дії, знижуючи ризики фінансових втрат і підвищуючи захищеність клієнтів.

7. Модель готова до масштабування. Система розроблена з урахуванням можливості роботи з великим обсягом даних і адаптації до нових сценаріїв шахрайства. Це робить її корисним інструментом для довгострокового використання в банківській сфері.

ПЕРЕЛІК ПОСИЛАНЬ

1. Lundberg, S. M., & Lee, S.-I. A unified approach to interpreting model predictions. Advances in Neural Information Processing Systems, 2017, 30, pp. 4765-4774. URL: https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf (дата звернення: 11.11.2024).
2. Breiman, L. Random Forests. Machine Learning Journal, 2001, Vol. 45, pp. 5-32. DOI: https://doi.org/10.1023/A:1010933404324 (дата звернення: 12.11.2024).
3. López, V., Fernández, A., García, S., Palade, V., & Herrera, F. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. Information Sciences, 2013, 250, pp. 113-141. DOI: https://doi.org/10.1016/j.ins.2013.07.007 (дата звернення: 13.11.2024).
4. Закон України «Про захист інформації в інформаційно-телекомунікаційних системах»: Закон України від 5 липня 1994 року № 80/94-ВР. URL: https://zakon.rada.gov.ua/laws/show/80/94-%D0%B2%D1%80 (дата звернення: 14.11.2024).
5. Lundberg, S. M., Erion, G. G., Chen, H., et al. Explainable AI for Trees: From Local Explanations to Global Understanding. Nature Machine Intelligence, 2020, 2, pp. 56-67. DOI: https://doi.org/10.1038/s42256-019-0138-9 (дата звернення: 15.11.2024).
6. Khurana, U., Turaga, D., & Samulowitz, H. Feature engineering for predictive modeling using reinforcement learning. Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM), 2018, pp. 2043-2046. DOI: https://doi.org/10.1145/3269206.3269254 (дата звернення: 16.11.2024).

7. API Documentation for Corezoid. Corezoid Documentation Platform. URL: https://doc.corezoid.com/ (дата звернення: 17.11.2024).
8. Pedregosa, F., Varoquaux, G., Gramfort, A., et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 2011, 12, pp. 2825-2830. URL: https://jmlr.org/papers/v12/pedregosa11a.html (дата звернення: 18.11.2024).
9. Kohavi, R., & Longbotham, R. Online Controlled Experiments and A/B Testing. Encyclopedia of Machine Learning and Data Mining. Springer, 2017. DOI: https://doi.org/10.1007/978-1-4899-7687-1_101 (дата звернення: 19.11.2024).
10. Fawcett, T. An introduction to ROC analysis. Pattern Recognition Letters, 2006, 27(8), pp. 861-874. DOI: https://doi.org/10.1016/j.patrec.2005.10.010 (дата звернення: 20.11.2024).
11. Kuhn, M., & Johnson, K. Applied predictive modeling. Springer, 2013. 600 p. DOI: https://doi.org/10.1007/978-1-4614-6849-3 (дата звернення: 21.11.2024).
12. Парент, Р. Основи програмування в Corezoid: Посібник з API-інтеграції. Київ: IT-платформа, 2023. 240 с. (дата звернення: 22.11.2024).
13. Thomas, P., & Sutton, C. Feature selection for classification: A review. Data Mining and Knowledge Discovery, 2020, 34(3), pp. 422-439. DOI: https://doi.org/10.1007/s10618-019-00617-y (дата звернення: 23.11.2024).
14. Офіційна документація Scikit-learn. URL: https://scikit-learn.org/stable/documentation.html (дата звернення: 24.11.2024).
15. Shrikumar, A., Greenside, P., & Kundaje, A. Learning important features through propagating activation differences. Proceedings of the 34th International Conference on Machine Learning (ICML), 2017, Vol. 70, pp. 3145-3153. DOI: https://proceedings.mlr.press/v70/shrikumar17a.html (дата звернення: 25.11.2024).
16. Patel D., Patel F., Chauhan U. Recommendation Systems: Types, Applications, and Challenges. International Journal of Computing and

Digital Systems, 2023, Vol. 13, no. 1, pp. 851-868. DOI: https://doi.org/10.12785/ijcds/130168 (дата звернення: 26.11.2024).

17. Zhang Q., Lu J., Zhang G. Recommender Systems in E-learning. Journal of Smart Environments and Green Computing, 2022, No.1(2), pp. 76-89 (дата звернення: 27.11.2024).
18. Cherednichenko O.Y., Ivashchenko O.V., Gontar Y.M., Vorona B.M. Інтелектуальний аналіз пропозицій товарів на основі контекстних рекомендацій. Вісник НТУ «ХПІ». Серія: Системний аналіз, управління та інформаційні технології, 2021, № 1320 (44), с. 57-66. DOI: https://doi.org/10.20998/2079-0023.2018.44.10 (дата звернення: 28.11.2024).
19. Documentation of XGBoost. URL: https://xgboost.readthedocs.io (дата звернення: 29.11.2024).
20. Documentation of LightGBM. URL: https://lightgbm.readthedocs.io (дата звернення: 30.11.2024).
21. Documentation of CatBoost. URL: https://catboost.ai/en/docs (дата звернення: 01.12.2024).

ДОДАТОК А ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

МАГІСТЕРСЬКА РОБОТА

МЕТОДИКА ВИЯВЛЕННЯ ШАХРАЙСТВА В БАНКІВСЬКІЙ СФЕРІ НА ОСНОВІ МАШИННОГО НАВЧАННЯ

Виконав: студент групи ПДМ-63 Соломоник Олексій Павлович

Керівник: к.ф-м.н., доцент кафедри ІТ Компан Сергій Володимирович

Київ - 2025

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: Підвищення ефективності виявлення фінансового шахрайства у банківській сфері за допомогою машинного навчання та автоматизації процесів на базі платформи Corezoid.

Об'єкт дослідження: Процес виявлення шахрайських транзакцій у банківських системах.

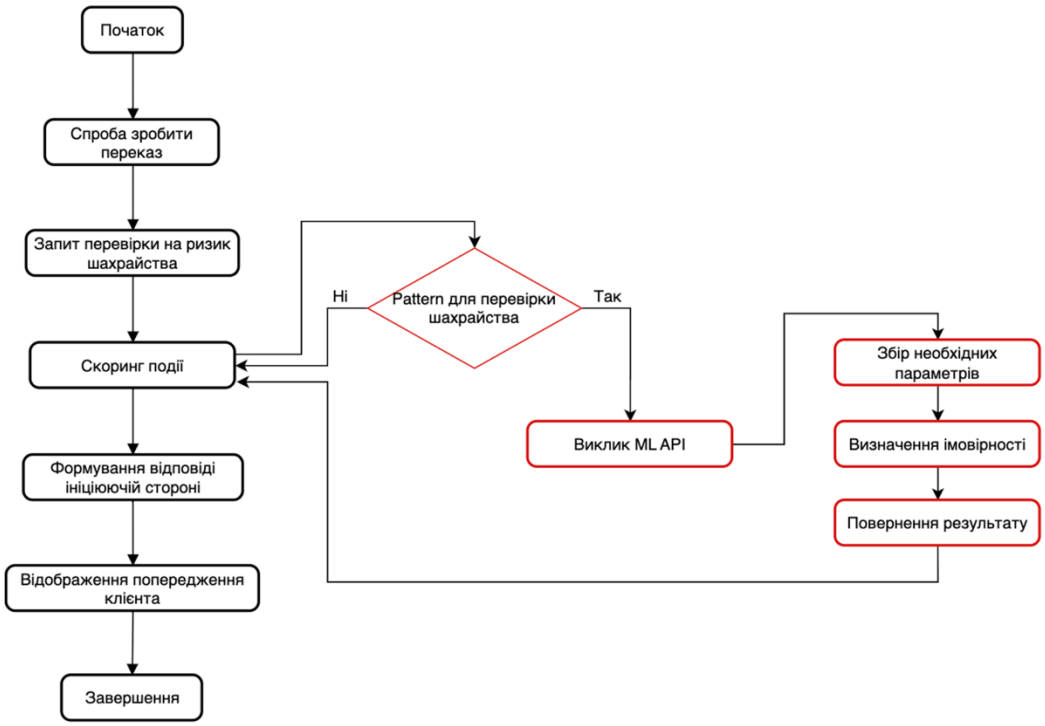
Предмет дослідження: Методи машинного навчання для виявлення фінансового шахрайства та їх впровадження у фінансові процеси.

ТРАДИЦІЙНА МОДЕЛЬ БОРОТЬБИ З ШАХРАЙСТВОМ У БАНКІВСЬКІЙ СФЕРІ

Головний недолік:

• Відсутність механізму взаємодії з клієнтом у випадках "скаму" (коли клієнт самостійно здійснює транзакцію).

МОДИФІКОВАНА СХЕМА ПЕРЕВІРКИ ТРАНЗАКЦІЇ НА ПРЕДМЕТ ШАХРАЙСТВА



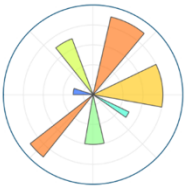
ВИКОРИСТАНІ ІНСТРУМЕНТИ

1.Програмне забезпечення для моделювання :

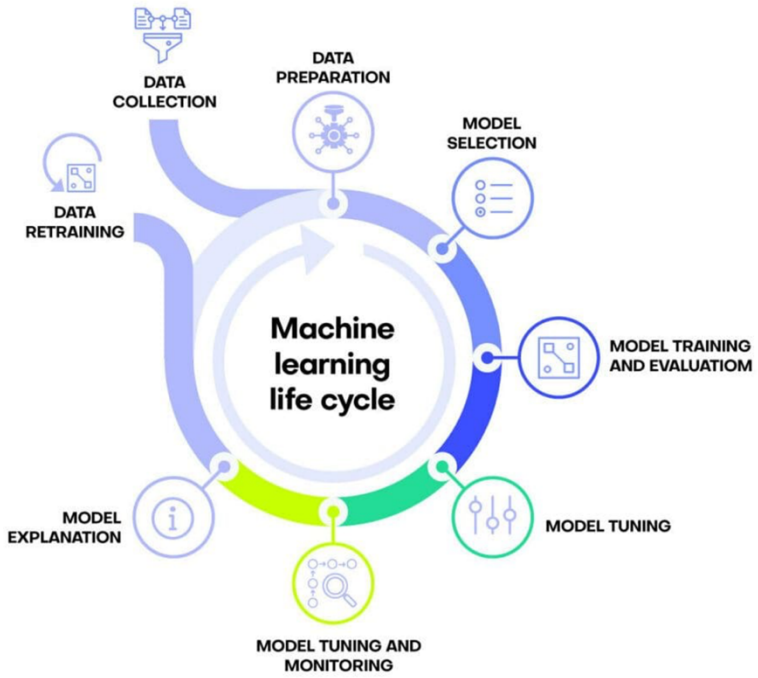
1. **Python**
2. **Бібліотеки**:
    1. Scikit-learn
    2. Seaborn та Matplotlib
    3. Pandas та NumPy

2.Програмне забезпечення для інтеграції:

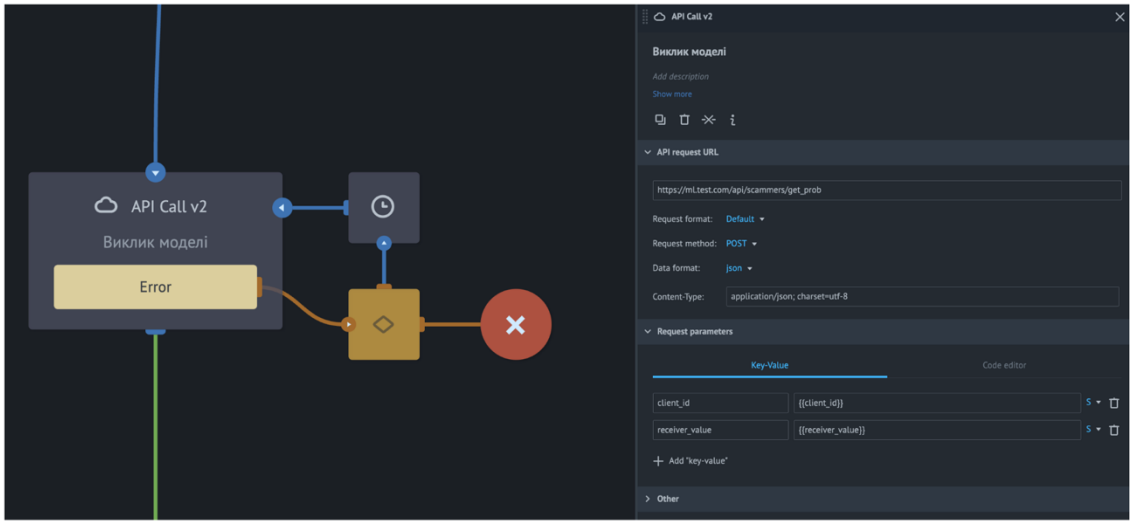
1. **Corezoid**:
    1. Створено процес обробки транзакцій у реальному часі.
    2. Інтегровано виклик ML-моделі



ЕТАПИ СТВОРЕННЯ МОДЕЛІ МАШИННОГО НАВЧАННЯ



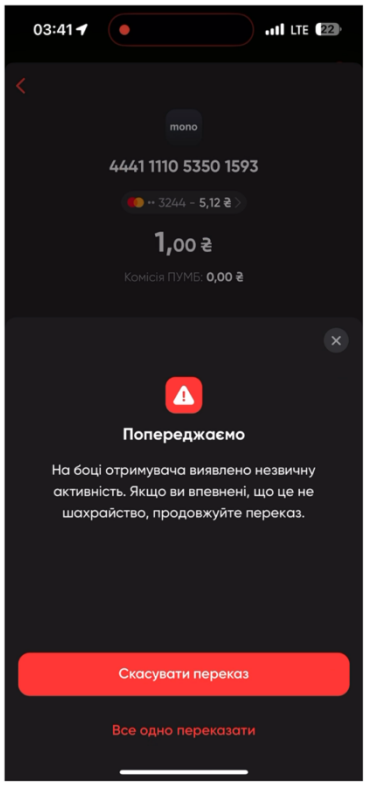
ОБРОБКА ТРАНЗАКЦІЙ У РЕАЛЬНОМУ ЧАСІ



Реальний час як ключова особливість:

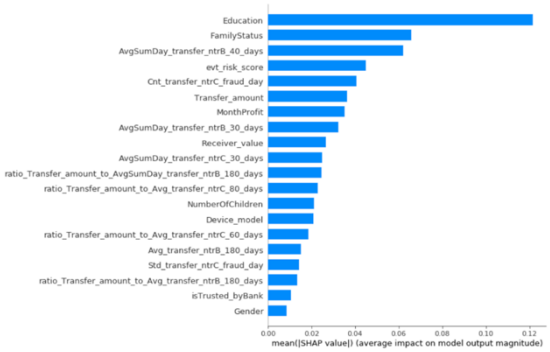
- Модель прогнозує ймовірність шахрайства для транзакцій в онлайні.
- Процес інтегрований у платформу Corezoid, забезпечуючи оперативність.

ПРАКТИЧНИЙ РЕЗУЛЬТАТ

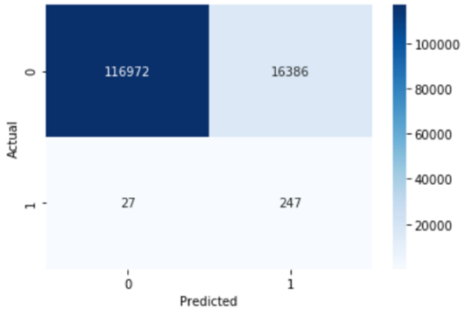


ЕКРАННІ ФОРМИ

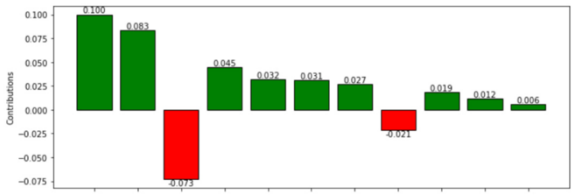
Вплив ознак на прогноз моделі
(SHAP-аналіз)



Матриця плутанини (Confusion Matrix):
Оцінка точності класифікації

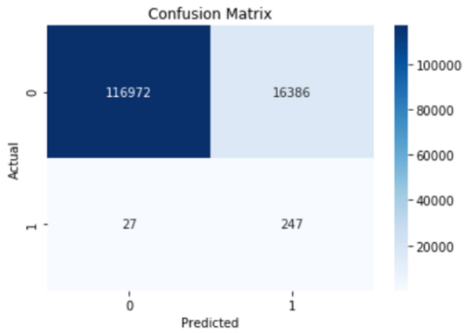


Вплив параметрів на рішення моделі
(Waterfall-графік)



РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ: АНАЛІЗ ЕФЕКТИВНОСТІ МОДЕЛІ

```
Logloss score: 0.2881217096293793
AUC score: 0.9501676131521508
Confusion matrix:
 [[116972  16386]
 [    27    247]]
Balanced accuracy score: 0.8892937926921476
```



	precision	recall	f1-score	support
0	1.00	0.88	0.93	133358
1	0.01	0.90	0.03	274
accuracy			0.88	133632
macro avg	0.51	0.89	0.48	133632
weighted avg	1.00	0.88	0.93	133632

ВИСНОВКИ

1. Розроблено модель, яка виявляє 86% шахрайських операцій (Recall) і правильно класифікує 90% звичайних транзакцій (Specificity).
2. Модель інтегровано в банківську систему, автоматизовано оцінку ризиків і обробку транзакцій.
3. Завдяки прозорим графікам впливу (Waterfall) забезпечено інтерпретованість і довіру до моделі.
4. Ефективність моделі підтверджено на тестових даних, досягнуто високої точності.

ПУБЛІКАЦІЇ ТА АПРОБАЦІЯ РОБОТИ

Тези доповідей:

1. Соломоник О. П. Розробка системи машинного навчання для оптимізації банківських операцій // Всеукраїнська науково-технічна конференція «Виклики та рішення в програмній інженерії». Збірник тез. – К.: ДУІКТ, 26 листопада 2024 року, С. 129-131

2. Соломоник О. П. Інтеграція потокової та історичної обробки даних для інтелектуального аналізу // Всеукраїнська науково-технічна конференція «Виклики та рішення в програмній інженерії». Збірник тез. – К.: ДУІКТ, 26 листопада 2024 року, С. 39-42