Київський національний університет імені Тараса Шевченка

Економічний факультет

Кафедра економічної кібернетики

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА
на тему «КЛАСИФІКАЦІЯ КЛІЄНТІВ НА РАННІХ ЕТАПАХ ВИКОРИСТАННЯ МОБІЛЬНОГО ДОДАТКУ»

студентки 4 курсу
спеціальності 051 «Економіка»
ОП «Економічна кібернетика»
денної форми навчання
Могильної Руслани Русланівни

Науковий керівний:
кандидат фізико-математичних наук, доцент
Кравець Тетяна Вікторівна

Засвідчую, що в цій роботі немає запозичень із праць інших авторів без відповідних посилань

Студент_____
(підпис)

Роботу допущено до захисту перед ЕК
рішенням кафедри економічної кібернетики
від 12 червня 2023 р., протокол № 17

Завідувач кафедри:
Доктор економічних наук, професор
Ляшенко Олена Ігорівна _____
(підпис)

КИЇВ – 2023

РЕФЕРАТ

Кваліфікаційна робота бакалавра містить: 64 ст., 36 рис., 3 табл., 47 джерел, додатки

Ключові слова: трансакційна модель монетизації, кити в бізнесі, задача класифікації, методи машинного навчання, CatBoost.

Об'єкт дослідження: кити в трансакційній бізнес моделі.

Мета дослідження: класифікація користувачів на ранніх етапах користування мобільним додатком.

Методи дослідження: спостереження, порівняння, узагальнення, статистичний аналіз, когортний аналіз, машинне навчання з використанням різних алгоритмів, зокрема CatBoost.

Наукова новизна, теоретична значимість дослідження: застосовано методи машинного навчання для класифікації користувачів мобільного додатку.

Практична цінність: результати дослідження можуть бути використані для покращення маркетингових та продуктових стратегій мобільного додатку для підвищення задоволеності та лояльності найцінніших клієнтів.

RESUME

Taras Shevchenko National University of Kyiv,

Faculty of Economics, Department of Economic Cybernetics

Key words: transactional monetization model, whales in business, classification task, machine learning methods, CatBoost.

The graduation research describes the features of classification mobile application users into whales and regular users, by applying machine learning methods (CatBoost). The resulting model can also be applied to new customers for their classification.

The work is interesting for those who are engaged in improving the experience of high-profit mobile application users.

Pages 64, pictures 36, tables 3, bibliog. 47, append.

ЗМІСТ

ВСТУП

Ніша розробки мобільних додатків стає все більш популярною і привабливою для ведення бізнесу. Вона має значний об’єм потенційних клієнтів та є перспективною з точки зору отримання високого рівня доходу. Для збереження конкурентоспроможності потрібно мати глибоке розуміння того які саме клієнти найбільше впливають на фінансові показники та активно працювати над їх залученням, покращенням досвіду користування та підвищенням задоволеності. Далі високодохідних користувачів, які генерують значну долю виручки компанії будемо називати китами.

Виявлення китів на ранніх етапах використання мобільного додатку дозволяє забезпечити максимальну ефективність роботи всіх відділів та спрямувати ресурси на залучення та утримання саме цільової аудиторії, яка має великий потенціал для генерації виручки. Це може включати розробку персоналізованих стратегій маркетингу та продажу, персоналізованих пропозицій, програм лояльності. Зосередження зусиль на високодохідних користувачах сприяє збільшенню середнього чеку, частоти покупок та загальної виручки компанії. Виявлення китів на ранніх етапах дозволяє виявити ознаки, які можуть вказувати на зростання ризику втрати цих користувачів. Це надає можливість вжити необхідних заходів для збереження їхньої лояльності та задоволеності, що сприяє збереженню прибутковості та фінансової стабільності бізнесу.

Вищенаведені фактори обумовлюють **актуальність** досліджень які спрямовані на класифікацію користувачів на високодохідних та звичайних за допомогою методів машинного навчання. Це дасть знання про ознаки які ще на початку використання певного мобільного додатку свідчать про перспективність клієнта і необхідність сфокусованої роботи над його активацією та утриманням.

Метою роботи є класифікація користувачів на ранніх етапах користування мобільним додатком. Для її досягнення необхідно виконати наступні **завдання**:

1. Проаналізувати наукову літературу та узагальнити теоретичні матеріали що стосуються особливостей трансакційної моделі монетизації та важливості китів для бізнесу.
2. Дослідити методи машинного навчання для задачі класифікації та метрики оцінки якості моделей.
3. Провести навчання моделей, вибрати оптимальний алгоритм, визначити ознаки що мають найбільший вплив на китовість клієнта та провести їх класифікацію.

Об'єктом дослідження є кити в трансакційній бізнес моделі.

Предметом дослідження – методи класифікації користувачів на китів та не китів на ранніх етапах використання мобільного додатку.

Для виконання поставлених завдань було використано наступні загальнонаукові **методи**: метод спостереження, порівняння, узагальнення, статистичний аналіз, когортний аналіз. До спеціальних методів даного дослідження відноситься машинне навчання з використанням різних алгоритмів, таких як CatBoost, логістична регресія, kNN, випадковий ліс, SVM, нейронна мережа.

Практична цінність роботи полягає в тому що результати класифікації можуть бути використані для покращення маркетингових та продуктових стратегій мобільного додатку для більш сфокусованої роботи над перспективними користувачами. Це дозволить покращити задоволеність найцінніших клієнтів, їх лояльність, що як наслідок підвищить фінансові показники компанії.

Структура роботи. Робота складається зі вступу, трьох розділів, висновків, списку використаних джерел із 47 позицій та додатків. Загальний обсяг становить 80 сторінок, основний текст складає 64 сторінки.

РОЗДІЛ 1. ТРАНСАКЦІЙНА МОДЕЛЬ МОНЕТИЗАЦІЇ ТА РОЛЬ КИТІВ У БІЗНЕСІ

1.1. Трансакційна модель, як вид монетизації бізнесу

Моделі монетизації бізнесу включають різні стратегії заробітку коштів на продуктах або послугах. Дві загально поширені моделі монетизації - це трансакційна та підписочна. Кожна з них має свої особливості та може більше чи менше підходити для конкретних бізнесів зі своєю специфікою, цілями, клієнтською базою. Деякі компанії комбінують ці дві моделі або використовують альтернативні моделі, такі як рекламна чи фріміум-модель, залежно від своїх потреб та ринкових умов. Розглянемо детальніше особливості, переваги та недоліки трансакційної та моделі з підпискою.

Підписочна модель передбачає отримання прибутку від регулярного платежу за доступ до продукту або послуги. Основними особливостями є регулярність платежів з певною періодичністю в залежності від тривалості підписки (щотижнево, щомісячно, щорічно), наявність в користувачів постійного доступу до контенту чи функціоналу. Ці особливості і обумовлюють такі переваги як стабільний потік доходу, кращу прогнозованість. Також у бізнесів з такою моделлю є можливість встановлювати різні рівні підписок для певних типів клієнтів. До недоліків можна віднести важкість залучення нових підписників та збереження існуючих, постійна необхідність надання цінного контенту або покращення продукту, потреба у відповідному інфраструктурному та підтримуючому середовищі [1].

Трансакційна модель це бізнес стратегія, що передбачає отримання прибутку з окремих трансакцій або продажу конкретних товарів або послуг. Особливостями даного виду монетизації є пряма оплата за конкретний продукт або послугу та здійснення покупки при кожній трансакції. Його часто використовують онлайн бізнеси, такі як мобільні додатки, проте він також доступний і для офлайн магазинів [2].

Перевагами трансакційної моделі є швидке отримання прибутку, масштабованість та відсутність обмежень щодо фізичної можливості здійснення

великої кількість покупок за обмежений проміжок часу. Дана модель приваблива для компаній, оскільки дає можливість потенційно заробляти більше коштів з кожної окремої трансакції і відповідно з кожного клієнта. Це дозволяє отримувати більший прибуток ніж при моделі з підпискою. Також стратегія дає інструмент у вигляді індивідуального контролю над кожною операцією та здатності установити ціну за окремі товари чи послуги. Даний тип монетизації також є привабливим і для іншої сторони бізнесу – клієнтів, оскільки вони можуть не купувати увесь пакет послуг у вигляді підписки, а отримувати необхідні товари та послуги точково [3]. Ця гнучкість дозволяє залучати різних клієнтів з різними потребами: ті хто готовий платити мало – буде платити мало, бо йому не обов'язково купувати повний пакет доступу до інформації чи послуг, а ті, хто готові платити багато і хочуть отримати більше послуг, ті можуть платити багато і ця виручка не буде обмежуватися ціною підписки.

До *недоліків* трансакційної моделі монетизації відносяться нестабільність прибутку через залежність від обсягів продажу товарів чи надання послуг, залежність від постійного привернення нових клієнтів та збереження існуючих. Оскільки клієнти можуть купувати окремі товари чи послуги не системно, а в разі виникнення потреби, їх досить важко утримувати та створювати для них різні програми лояльності. Крім того, для того щоб клієнти мали потребу в продукті чи послузі що надає бізнес, йому потрібно створювати дійсно якісні продукти щоб надавати реальну цінність і закривати потреби. Варто також зауважити, що модель монетизації на основі трансакцій потребує хорошої платіжної інфраструктури для забезпечення якісного процесингу платежів. Компанія повинна забезпечити зручний та надійний процес оплати, що захищає дані клієнтів [2].

Як ми вже відзначили особливістю моделі є нестабільний дохід та покупки за потреби. Вони спричинюють ще один недолік трансакційної моделі у вигляді складності прогнозування. Розглянемо трішки детальніше причини [4]:

- Нестабільність платежів. У трансакційній моделі, де клієнти оплачують за окремі товари або послуги, виручка може коливатися в залежності від кількості

і ціни куплених товарів, частоти покупок та інших факторів. Це може призводити до нестабільності виручки, що робить прогнозування складним завданням;

- Складність у відстеженні патернів поведінки користувачів. Кожен клієнт має власні унікальні патерни поведінки, які впливають на його витрати. Відстеження і аналіз цих патернів може бути складним завданням, особливо якщо у бізнесу велика кількість клієнтів і всі вони мають різні стилі поведінки. Нерегулярність у покупках та зміна уподобань користувачів можуть ускладнювати прогнозування виручки.
- Нестабільність ринку. Ринок, в якому працює бізнес, може піддаватися змінам, швидким технологічним зрушенням, змінам у споживацьких тенденціях тощо. Ці зміни можуть впливати на попит, конкуренцію та виручку. Це зобов’язує бізнес постійно покращувати свій продукт чи послуги, оптимізувати витрати щоб зберігати конкурентоспроможність на ринку та цінність для клієнтів.

Отже, трансакційна модель монетизації є цікавою та перспективною з точки зору виручки, оскільки дозволяє отримувати прибуток гнучко з кожного окремого користувача. Однак, важливо зазначити, що вона має свої складнощі, особливо з точки зору прогнозування виручки [5].

Нестабільність платежів, складність у відстеженні патернів поведінки користувачів, непередбачуваність на ринку є факторами, які ускладнюють прогнозування виручки в рамках трансакційної моделі [6]. Бізнесам необхідно активно вивчати та аналізувати дані, враховувати зміни в споживацьких звичках та ринкових умовах, а також застосовувати аналітичні інструменти для ефективного прогнозування виручки.

Незважаючи на складнощі, трансакційна модель залишається привабливою для багатьох компаній, особливо в сфері інформаційних технологій. Вона дозволяє максимізувати дохід від кожного клієнта та стимулює покупки на індивідуальній основі. Проте, успіх використання трансакційної моделі вимагає досить великих зусиль та сильної команди аналітики щоб мати змогу розуміти та прогнозувати поведінку користувачів.

1.2. Кити в трансакційній моделі та їх значення для бізнесу

Однією з особливостей трансакційної моделі монетизації є наявність групи користувачів, які складають досить малий відсоток від всієї клієнтської бази, які генерують значну долю виручки компанії. Саме їх і називають китами.

Отже, кит у контексті бізнесу - особа, що зробила значний внесок у виручку компанії або здійснила значну кількість трансакцій. Кити можуть бути окремими користувачами або групами користувачів, які є основними джерелами прибутку для бізнесу. Китовість виникає через нерівномірний розподіл виручки, коли частина клієнтів платить багато, в той час як інша більшість роблять менший внесок або взагалі не здійснюють трансакції [7].

Це може мати як позитивний, так і негативний вплив на бізнес. З одного боку, кити можуть бути головним джерелом прибутку, допомагаючи компанії забезпечити фінансову стабільність. Вони можуть стати цільовою аудиторією для маркетингових стратегій та пропозицій, спрямованих на збільшення їх віддачі. З іншого боку, залежність від китів може бути ризикованою [8]. Якщо компанія занадто сильно покладається на китів, вона стає вразливою до втрати прибутку у випадку, якщо кити змінять свої споживчі звички або перестануть взаємодіяти з бізнесом і, наприклад, перейдуть до конкурентів. Крім того, нерівномірний розподіл виручки може призвести до незадоволення інших користувачів, які відчувають, що компанія не надає їм достатньо уваги або переваг. Управління китами вимагає уважного аналізу та стратегій. Компанія повинна розуміти, хто є її китами і які чинники сприяють їхньому успіху. Важливо забезпечити належну підтримку та обслуговування китів, а також залучати і стимулювати інших користувачів, щоб розширити базу прибутку і зменшити ризики, пов'язані залежністю від окремих клієнтів або груп клієнтів.

В цілому, китовість є важливою особливістю трансакційної моделі монетизації, яка вимагає уваги та балансу з боку бізнесу. Це може бути сприятливою стратегією, але водночас вимагає постійного аналізу, адаптації та управління, щоб забезпечити стабільність та розвиток компанії.

Одним з аналітичних інструментів є китова крива на рис. 1.1. Вона візуалізує клієнтів за трьома категоріями – найбільш прибуткові для бізнесу, беззбиткові та збиткові [9]. По осі ординат відображається відсоток сукупного прибутку, а по осі абсцис – відсоток клієнтів, упорядкованих за прибутковістю (від найбільшої до найменшої). Тоді відсоток користувачів на яких крива швидко зростає є найбільш цінними для бізнесу та відповідно приносять найбільше виручки, після піку будуть користувачі, які є беззбитковими, та витрачають відносно мало, покриваючи витрати на їх залучення та утримання, а потім, коли крива починає спадати на хвості кита це свідчить, що там знаходяться користувачі, які є збитковими для компанії.

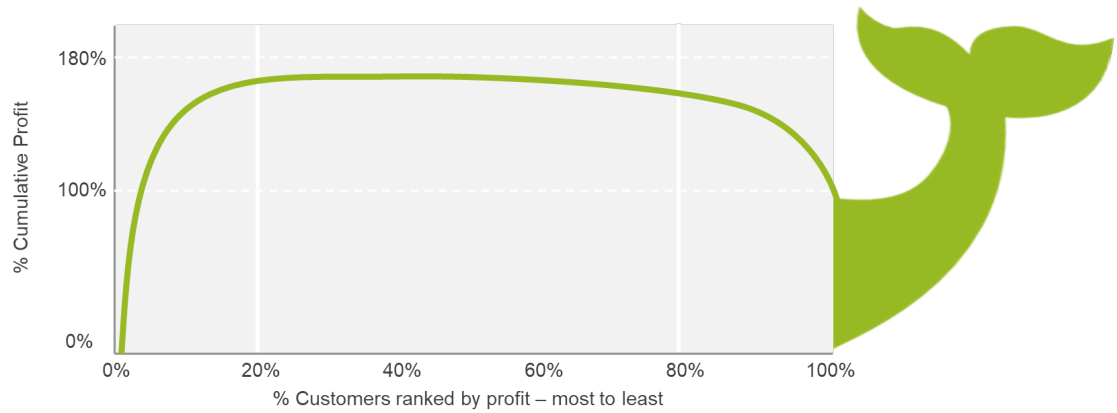

Рис. 1.1. Китова крива

Джерело: [9]

Розглянемо детальніше типи ринків та їхню китовість. Для цього введемо такий показник, як Whale index для 80 перцентиля (WI80). Він показує у скільки разів більше витрачає користувач з верхніх топ 20%, ніж користувач з найнижчих 80% позицій [10].

На рис. 1.2 відображений відсоток кумулятивної виручки по осі ординат та кумулятивної кількості користувачів по осі абсцис. Це досить схоже на китову криву, але без вирахування витрат. Криві показують частку продажів, створених кожним персентилем клієнтів. Чим більш випукла лінія, тим більше в ніші користувачів, які можна вважати китами. Якщо крива наближається до діагоналі, це свідчить про те, що частка продажів, створена кожним користувачем є приблизно однаковою.

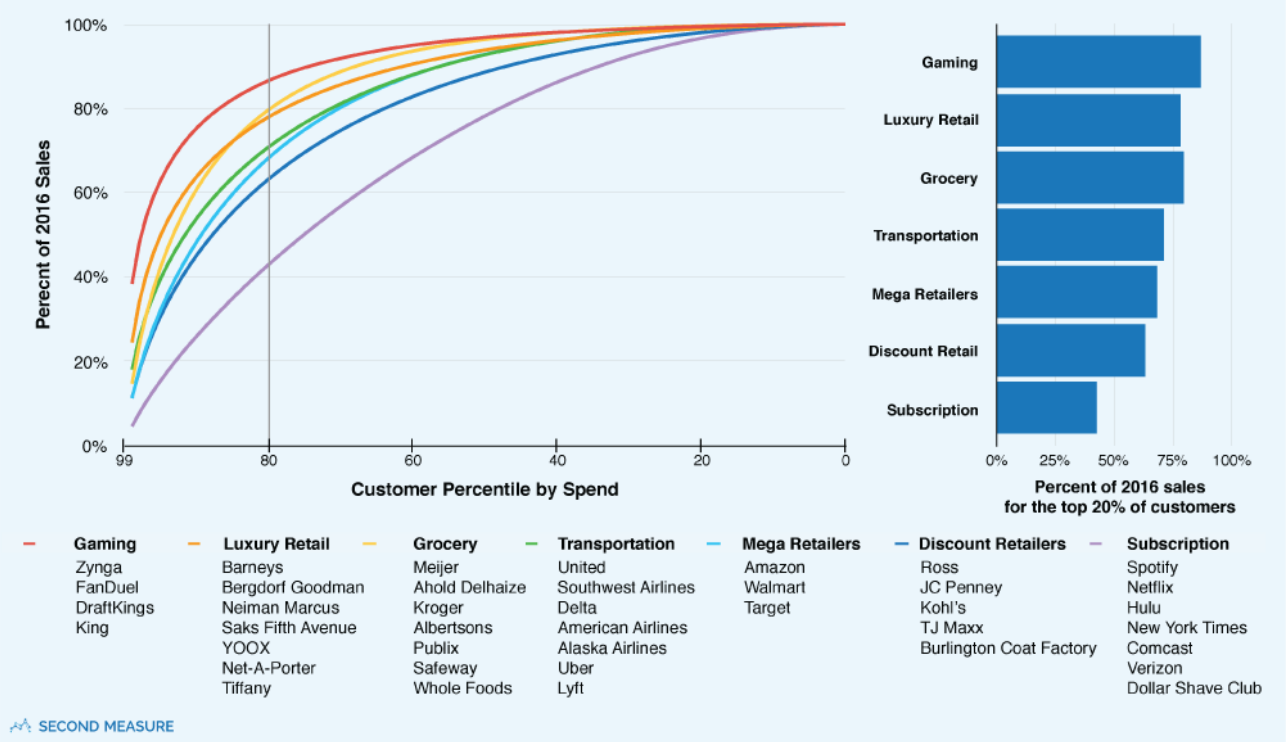

Рис. 1.2. Відношення відсотку кумулятивної виручки в США до кумулятивної кількості користувачів

Джерело: [10]

Бачимо, що найбільшою індустрією з китовою моделлю є ігрова. Серед компаній які можна назвати це Zynga, King - американські компанії, що спеціалізуються на розробці та виданні мобільних та онлайн-ігор [11]. Поряд з ними також FanDuel, DraftKings – компанії що займається ставками на спорт [12]. Для цієї ніші 20% користувачів приносять більше 85% доходу.

Наступною за китовістю є Luxury retail, куди відносяться такі компанії як Barneys, Bergdorf Goodman, Neiman Marcus та інші. Тут предмети розкоші купують лише невеликий відсоток клієнтів. Наприклад компанія Tiffany має показник WI80 на рівні 14, що означає що середній клієнт з топ 20% генерує в 14 разів більше доходу ніж середній клієнт з інших 80% [13].

Бачимо, що найнижчий рівень WI80 спостерігається в компаній з моделями підписки, таких як Netflix, Spotify чи Hulu. Там виручка між клієнтською базою розподілена більш рівномірно.

Отже, китова модель монетизації є популярною та ефективною в багатьох нішах, зокрема в ігровій індустрії та предметів розкоші. Вони мають незначну

кількість користувачів, які вносять значний обсяг платежів і є ключовими джерелами прибутку. Саме тому кити є надзвичайно важливими для бізнесу і потребують особливого підходу та персоналізованих стратегій. Важливо вміти працювати з високодохідними користувачами, адаптувати продукти і послуги до їхніх потреб та очікувань. Усвідомлення значення китів і зосередження зусиль на їхньому задоволенні може принести значний ріст виручки та позитивно вплинути на бізнес.

1.3. Важливість визначення високодохідних користувачів на ранніх етапах використання мобільного додатку

З попередніх розділів було визначено що кити є важливими факторами впливу на успішність бізнесу. Саме тому дуже важливо правильно їх ідентифікувати на ранніх етапах використання мобільного додатку та будувати персоналізовані стратегії залучення на продукт, активації для того щоб показати цінність, розвитку, повернення та підтримки у разі виникнення якихось проблем чи труднощів.

Розглянемо детальніше важливість раннього виявлення китів для кожного етапу життя користувача в мобільному додатку та можливості використання цього знання для побудови кращих стратегій і надання найбільш цінного досвіду для кожного клієнта.

Цифровий маркетинг. Важливість раннього виявлення китів полягає в тому, що це дозволяє ефективно використовувати рекламні ресурси і маркетингові зусилля для максимізації виручки та результативності кампаній. Високодохідні користувачі, зазвичай, становлять невелику частку від загальної кількості користувачів, але приносять значну частину виручки. Виявлення їх на ранній стадії дозволяє спрямовувати рекламні витрати саме на цю цільову аудиторію, забезпечуючи кращу ефективність витрат і досягнення високих показників конверсії. Виявлення китів дозволяє зрозуміти їхні потреби, рівень задоволеності та звички [14]. Ця інформація може бути використана для розробки персоналізованих маркетингових стратегій, які відповідають унікальним

потребам та очікуванням, що сприяє підвищенню рівня задоволеності, лояльності та виручки.

Активація користувача та його утримання в додатку. Раннє виявлення китів є надзвичайно важливим для продуктової команди з багатьох причин. Першою з них є можливість застосовувати персоналізовані активаційні механіки. Розуміння поведінкових патернів та інтересів цільових користувачів дозволяє продуктовій команді розробляти персоналізовані активаційні механіки. Це означає, що команда може створити унікальний досвід взаємодії з продуктом, який найкраще відповідає потребам та очікуванням китів. Це може включати індивідуальні пропозиції, рекомендації або спеціальні функції, які залучають та зацікавлюють цільову аудиторію. Також важливим є доведення користувача до «aha» моменту, коли користувач розуміє цінність продукту та «habit» моменту, коли в клієнта створюється звичка користуватися продуктом [15]. Виявлення китів у ранніх стадіях дозволяє продуктовій команді сконцентруватись на наданні користувачам позитивного досвіду, який сприяє досягненню цих моментів. Це може включати надання особливої підтримки, додаткових ресурсів або зручних інструментів, які допомагають користувачам отримувати максимальну цінність від продукту.

Додатково, знання того що користувач в майбутньому стане китом допоможе слідкувати за його задоволеністю та поверненням. Вони можуть бути джерелом цінного фідбеку щодо продукту та допомогти виявити потенційні проблеми або можливості для поліпшення. Підтримка та задоволеність китів важлива для підтримання їх у довгостроковому плані та забезпечення стабільної виручки.

Продуктовий маркетинг. Знання патернів поведінки китів та імовірності того що певна група в майбутньому ними стане дає можливість краще налаштовувати сегментований продуктовий маркетинг і забезпечувати ефективну взаємодію з клієнтами. Команда може розробляти персоналізовані стратегії повернення на продукт, шляхом налаштування спеціальних пуш-повідомлень, відправки поштових листів та інших механіки комунікації, які

відповідають інтересам, потребам та поведінковим звичкам [16]. Це допомагає створити персоналізований досвід, який залучає китів та стимулює їх повернення на продукт. Також можна відправляти спеціальні пропозиції та повідомлення, які мають більший вплив та є релевантними для цієї групи користувачів. Вони будуть більш схильні реагувати на такі комунікації і виконувати вказані дії, такі як повернення на продукт, покупка або залучення інших користувачів [17].

Служба підтримки. Раннє виявлення китів може бути використане для покращення роботи команди підтримки користувачів. Перш за все, знання про китів на ранніх етапах дозволяє визначити, які користувачі потребують найбільшої уваги та негайної підтримки. За допомогою аналізу їхньої активності, покупок, поведінки та взаємодії з продуктом можна встановити пріоритети і зосередити зусилля на задоволенні їхніх потреб. Також важливо перспективній частині користувачів надати персоналізовану підтримку та відповідати на їхні конкретні потреби та запитання. За допомогою інформації про їхній контекст та історію взаємодії з продуктом, команда підтримки може швидше та ефективніше вирішувати їхні проблеми та надавати рекомендації [16].

Організація відділу підтримки користувачів, який буде забезпечувати проактивну комунікацію, яка дозволить спрогнозувати їхні потреби та проблеми заздалегідь. Надалі можна використати цю інформацію, щоб ініціювати контакт з китами, надсилати персоналізовані повідомлення, сповіщення та поради щодо продукту. Це допоможе підтримати високий рівень зацікавленості та залученості китів, а також позитивно вплине на їхню взаємодію з продуктом.

Додатково можна збирати цінну інформацію для вдосконалення продукту. Команда підтримки може передавати зібрані дані та фідбек розробникам, щоб покращити функціонал, виправити помилки або додати нові можливості, які краще задовольнять потреби. Це допоможе підвищити задоволеність та залученість цільових користувачів, а також збільшити їхню лояльність до продукту.

РОЗДІЛ 2. МЕТОДИ ТА ОЦІНКА МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ ДЛЯ ЗАДАЧІ КЛАСИФІКАЦІЇ

2.1. Етапи моделювання для виконання задачі класифікації методами машинного навчання

У цьому підрозділі розглянемо сутність задачі класифікації методами машинного навчання та її основні етапи. Класифікація є однією з фундаментальних задач у сфері машинного навчання, де модель навчається розподіляти об'єкти на певні класи на основі тренувальних даних.

Отже, *задача класифікації* – це задача розбиття множини об'єктів або спостережень на апріорно задані групи, називані класами, всередині кожної з яких вони вважаються схожими один на одного, та мають приблизно однакові властивості й ознаки. При цьому рішення здійснюється на основі аналізу значень атрибутів (ознак) [18]. Цей процес включає в себе важливі етапи, які розглянемо далі.

Збір та підготовка даних. Перший етап моделювання для задачі класифікації полягає у зборі та підготовці даних. Він включає в себе збір потрібних даних, які відображають характеристики об'єктів, що підлягають класифікації, а також мітки класів, до яких вони належать. Залежно від конкретної задачі та домену, в якому виконується моделювання вхідні дані можуть подаватися у різному форматі, наприклад у текстовому, у вигляді зображень, проте найбільш поширеним є табличний.

Для даного дослідження джерелом є база даних Vertica, тому для збору інформації потрібно опрацювати наявні таблиці, їх структуру, взаємозв’язки [19]. Наступним кроком є написання SQL запиту, який обирає потрібні стовпці, що містять дані для побудови моделі. Важливо використовувати потрібні фільтри та обмеження для того щоб отримати релевантний набір інформації [20].

Після збору даних важливим кроком є їх підготовка. Це включає в себе обробку даних, таку як видалення аномальних значень, заповнення пропущених даних, нормалізацію або стандартизацію функцій [21].

Вибір моделі. Другий етап полягає у виборі моделі для класифікації даних. Існує багато різних алгоритмів і моделей машинного навчання, які можуть бути використані для класифікації, таких як логістична регресія, дерева рішень, випадковий ліс, метод опорних векторів, нейронні мережі та інші [22]. Вибір моделі залежить від властивостей даних, обсягу даних, кількості класів, складності задачі та багатьох інших факторів.

Навчання моделі. Після вибору моделі наступний етапом є її навчання. Цей процес містить в собі поділ даних на тренувальний і тестовий набори, встановлення параметрів і виконання процесу навчання для оптимальної класифікації даних. Під час процесу навчання модель адаптується до даних шляхом оновлення внутрішніх характеристик. Навчання може бути ітераційним для підбору оптимальних гіперпараметрів та досягнення оптимальної якості [21].

Оцінка та підбір моделі. Після навчання моделі необхідно оцінити її ефективність і здатність до класифікації нових даних. Для цього використовується тестова вибірка, яка не брала участь в тренуванні. Далі обраховуються метрики оцінки якості моделі, такі як точність, F1-Score, матриця помилок, ROC-крива, Precision-Recall Curve, які детальніше описані в наступному підрозділі [23]. Залежно від результатів оцінки в модель можуть бути внесені зміни або взагалі обрано іншу для отримання кращих результатів.

Використання моделі для класифікації нових даних. Останній етап полягає у використанні навченої моделі для нових, раніше невідомих даних. Це може бути виконано шляхом виклику методу прогнозування моделі для визначення категорії нового об’єкта з відомими вхідними характеристиками. В рамках даного дослідження новими об’єктами будуть всі користувачі, які нещодавно потрапили в додаток і яких потрібно класифікувати як майбутній кит або звичайний користувач в залежності від характеристик та поведінки.

Отже, у цьому розділі ми детально розглянули основні етапи моделювання для виконання задачі класифікації методами машинного навчання. Від збору та підготовки даних до вибору моделі, навчання, оцінки та використання моделі для

класифікації нових даних, кожен етап відіграє важливу роль у досягненні точності та надійності класифікації.

2.2. Вибір моделей машинного навчання

Модель К-найближчих сусідів (K-Nearest Neighbors, KNN) є одним із найпростіших алгоритмів машинного навчання для задач класифікації та регресії. У моделі KNN, для класифікації нового прикладу даних, алгоритм шукає K найближчих прикладів даних з навчального набору, що мають найбільш близькі значення. Класифікація проводиться шляхом голосування більшості класів серед цих K найближчих сусідів. У випадку задачі регресії, KNN використовує середнє значення вихідних змінних з K найближчих сусідів як прогнозоване значення для нового прикладу.

Для обрахунку відстані між точками використовується формула Евклідової відстані $d(x, x') = \sqrt{(x_1 - x_1')^2 + \cdots + (x_n - x_n')^2}$ [24].

Головна ідея KNN полягає в тому, що подібні приклади даних зазвичай мають схожі вихідні значення. Алгоритм не вимагає попередньої побудови моделі навчання, але він зберігає весь навчальний набір даних, що може бути вимогливим для обробки великих наборів даних.

При використанні моделі KNN важливими параметрами є кількість сусідів (k), метрика відстані для визначення близькості між прикладами даних та можливість нормалізації ознак для уникнення нерелевантного впливу.

Оптимізація параметра k (кількості сусідів) для моделі kNN може бути здійснена шляхом використання методу перехресної перевірки (cross-validation) для оцінки якості моделі при різних значеннях k.

Основні кроки для підбору оптимального значення k у моделі kNN [25]:

- Розбити навчальний набір даних на навчальний та валідаційний піднабори;
- Визначити діапазон значень для параметра k, які потрібно перевірити. Часто він вибирається на підставі досвіду чи знань про розв'язувану задачу класифікації. Бажано, щоб параметр k був непарним, щоб зменшити імовірність «нічиєї». Найчастіше вибираються значення k = 3 і k = 5, але використовуються і великі значення, між 50 і 100. Як альтернативу параметр k можна вибрати так,

щоб він гарантував найкращі результати тренувальній множині. Можна також приписати ваги “голосам” k найближчих сусідів, використовуючи їх косинусну міру подібності;

- Для кожного значення k в діапазоні потрібно навчити модель з використанням тренувального піднабору даних, оцінити її якість на валідаційному піднаборі (наприклад метрикою точності, F1-Score) та зберегти результати для кожного значення.
- Вибрати значення k, що дає найкращий результат точності;
- Перетренувати модель з використанням обраного оптимального значення k на повному навчальному наборі даних;
- Оцінити якість остаточної моделі на тестовому наборі даних для оцінки її загальної точності.

До *переваг* моделі можна віднести відносну простоту у реалізації через невелику кількість гіперпараметрів, стійкість до викидів за рахунок того, що імовірність їх потрапляння до k сусідів незначна. Також при досить великих значеннях кількості сусідів вага голосу викидів ще більше зменшується. Ще одним плюсом є легкість в інтерпретації спеціалістами з різних сфер [26].

Основним *недоліком* даного алгоритму є необхідність великого об’єму пам’яті, особливо в порівнянні з іншими класифікаторами, оскільки для його використання потрібно зберігати всі тренувальні дані. Це може бути невигідним з точки зору бізнесу та фінансових витрат на процесинг.

Логістична регресія – ще один алгоритм, що використовується для прогнозування ймовірності виникнення бінарної події на основі вхідних змінних. Вона використовує логістичну функцію (також відому як сигмоїда) для моделювання ймовірності в межах від 0 до 1. Математичне представлення має вигляд:

$$E(Y_i|X_i) = p_i = \frac{e^{(\beta_0+\beta_1 X_i)}}{1 + e^{(\beta_0+\beta_1 X_i)}}$$

Логістична регресія заснована на припущенні, що залежність між вхідними змінними і ймовірністю належить до логістичної функції. Вона використовує метод максимальної правдоподібності для оцінки параметрів моделі на основі навчальних даних [27].

Однією з основних *переваг* логістичної регресії є те, що вона надає інтерпретовані результати, оскільки можна оцінити вплив кожної вхідної змінної на ймовірність віднесення до певного класу. Крім того, логістична регресія добре працює з наборами даних різних розмірів і може бути ефективною навіть при великій кількості змінних.

Для оптимізації логістичної регресії можна налаштовувати різні гіперпараметри, зокрема [28]:

- penalty, який може отримувати значення L1 та/або L2 – параметри, що відносяться до регуляризації, яка використовується для контролю розмірів коефіцієнтів моделі логістичної регресії
- C – обернена сила регуляризації

L1 (Lasso) регуляризація штрафує модель за наявність ненульових коефіцієнтів, тобто вона сприяє розрідженості коефіцієнтів. Це може бути корисно для вибору найбільш важливих ознак і виключення менш важливих.

L2 (Ridge) регуляризація штрафує модель за великі значення коефіцієнтів, але не сприяє розрідженості. Вона намагається зменшити величину коефіцієнтів, але не обов'язково робить їх нульовими.

Параметр C визначає обернену силу регуляризації. Величина параметра C контролює компроміс між використанням малих коефіцієнтів і точністю моделі. Чим менше значення параметра C, тим сильніше регуляризація, і модель буде вибирати меншу кількість функцій. Проте якщо воно буде занадто малим, то модель може мати більш загальну природу і ігнорувати окремі навчальні зразки або шум, що призведе до недооцінки важливості окремих функцій. З іншого боку, велике значення параметра C дозволяє моделі більше адаптуватися до навчальних даних, що може призвести до перенавчання. Зазвичай для підбору параметра тестують значення в діапазоні від 0.1 до 10.

Випадковий ліс - це ансамбль алгоритмів машинного навчання, який використовується для задач класифікації, регресії та інших завдань. В основі випадкового лісу лежить ідея комбінування декількох дерев рішень, що вирішують проблему. Кожне дерево рішень в лісі побудоване на випадково підібраних підмножинах даних та ознак [29]. Приклад структури випадкового лісу наведено на рис. 2.1.

Основні етапи побудови включають в себе [30]:

- Вибір випадкового підмножини даних з навчального набору замість використання всього набору даних;
- Вибір випадкової підмножини ознак;
- Побудова дерев рішень для кожної підмножини даних і ознак;
- Комбінування прогнозів кожного дерева, наприклад, за допомогою голосування для задач класифікації або середнього значення для задач регресії.

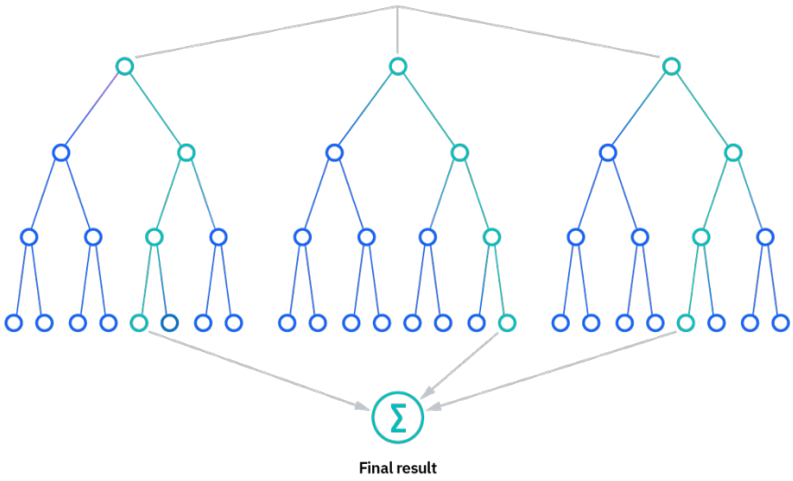

Рис. 2.1. Приклад випадкового лісу

Джерело: [30]

Перевагами алгоритму є висока точність, здатність до обробки великих наборів даних, стійкість до перенавчання і відповідно хороша застосовуваність на нових даних. Також важливою характеристикою є здатність надавати інформацію про важливість ознак, що спрощує інтерпретацію.

До *недоліків* можна віднести певну обчислювальну складність та тривалість навчання у випадках коли модель містить багато дерев рішень. Також якість

випадкового лісу сильно залежить від налаштувань параметрів, що може призвести до неефективності при неправильному виборі.

Отже, для оптимізації Random Forest необхідно налаштувати наступні такі параметри як кількість дерев, глибина дерев, кількість ознак та інші. Розглянемо їх детальніше [31].

Кількість дерев. Визначення оптимальної кількості дерев у лісі є важливим параметром. Зазвичай, більша кількість покращує точність моделі, але при цьому збільшується час навчання. Для оптимізації можна використовувати метод перехресної перевірки для визначення оптимальної кількості дерев.

Глибина дерев. Впливає на роздільну здатність моделі. Глибше дерево може вивчати складніші залежності в даних, але при цьому може виникнути перенавчання. Необхідно експериментувати з різними значеннями глибини дерева і вибрати оптимальне значення, яке забезпечує баланс між точністю та уникненням перенавчання.

Кількість ознак для розгляду при розділенні вузлів. Параметр, що визначає кількість ознак, які випадково вибираються для розгляду при розділенні вузлів дерева, впливає на різноманітність моделі. Зазвичай, розглядається квадратний корінь від загальної кількості ознак. При занадто малій кількості ознак модель може бути недосяжною, а занадто велика кількість може призводити до зайвого розгалуження.

Функція, що використовується для вимірювання якості розбиття на вузлах дерева (criterion). Цей параметр визначає, яка метрика буде використовуватись для визначення оптимального розбиття. Це можуть бути критерій Джині, що максимізує роздільну здатність, ентропійний критерій, що шукає максимальну неоднорідність, логарифмічна втрата.

Вибір критерію залежить від конкретної задачі та набору даних. Зазвичай Джині та ентропійний є популярними, оскільки вони добре працюють в багатьох випадках класифікації. Логарифмічна втрата частіше використовується в задачах, де необхідно прогнозувати ймовірності класів.

Також для оптимізації алгоритму можна використовувати або не використовувати Bootstrap. Це статистичний метод, у якому замість використання усього набору даних, вибірка для кожного дерева створюється шляхом випадкового вибору спостережень з повторенням з оригінального набору даних. Тобто кожен зразок може бути вибраний більше одного разу або може бути пропущений взагалі.

Цей підхід дозволяє кожному дереву у Random Forest отримувати унікальний піднабір даних, що сприяє різноманітності моделей і зменшує кореляцію між ними. Це допомагає уникнути перенавчання і забезпечує більш стабільні та узагальнюючі прогнози.

Support Vector Machine – ще один з алгоритмів машинного навчання, який використовується для задач класифікації і регресії. Головна ідея полягає у пошуку гіперплощини в просторі яка максимізуватиме відстань між двома класами даних. Ця гіперплощина служить роздільною межею для класифікації нових зразків. Алгоритм використовує припущення про те, що чим більша відстань між паралельними гіперплощинами – тим краще буде працювати класифікатор [32].

Гіперплощина визначається за допомогою підмножини тренувальних прикладів, які називаються опорними векторами. Ці вектори знаходяться на найближчій відстані до границі розділення [33].

SVM може використовувати різні ядерні функції, такі як лінійні, поліноміальні, радіальні базисні функції (RBF) тощо, що дозволяє моделі бути гнучкою у використанні різних форм розділяючих поверхонь. Крім того, SVM володіє хорошою здатністю до управління перенавчанням та підтримує розріджені вектори, що дозволяє ефективно обробляти набори даних з великою кількістю ознак.

SVM є потужним і широко використовуваним алгоритмом, особливо в задачах бінарної класифікації, де дані можуть бути лінійно або нелінійно роздільними. Він також може бути застосований до задач регресії та відомий своєю здатністю до обробки даних високої розмірності [34].

Support Vector Machine (SVM) має свої переваги і недоліки, які можуть впливати на його ефективність і придатність для конкретних завдань. До *плюсів* можна віднести ефективність у високорозмірних просторах зі значною кількістю ознак, розпізнаваність класів за умови правильно підібраних ядер, незначна кількість параметрів для налаштування що полегшує використання. *Недоліками* є чутливість до стандартизації даних та шумів, чутливість до налаштувань параметрів. При використанні на великих об’ємах даних може виникати обчислювальна складність та необхідність великої кількості пам’яті, оскільки модель вимагає збереження всієї тренувальної множини.

Нейронна мережа – математична модель, що імітує принципи функціонування біологічних нейронних мереж для вирішення складних завдань обробки інформації.

Нейронна мережа складається зі штучних нейронів, які взаємодіють між собою шляхом передачі сигналів. Кожен нейрон приймає вхідні дані, обчислює їх зважену суму, застосовує нелінійну функцію активації і видає вихідний сигнал. Ваги, які зв'язують нейрони, є параметрами моделі, які підлаштовуються під час навчання.

Нейронні мережі можуть мати різну архітектуру та складність, включаючи прості одношарові мережі та складні багатошарові архітектури, такі як згорткові нейронні мережі (Convolutional Neural Networks) та рекурентні нейронні мережі (Recurrent Neural Networks). Вони використовуються для розпізнавання образів, класифікації, прогнозування, обробки природної мови та багатьох інших завдань машинного навчання [35].

Важливою властивістю нейронних мереж є їх вміння навчатись за рахунок даних навколишнього середовища та в процесі підвищувати точність. Це відбувається за допомогою коригування синаптичних ваг і порогів. Синаптичні ваги це числове значення, яке відображає силу зв’язку між нейронами, тобто наскільки важливим є вхідний сигнал для вихідного сигналу нейрона. Великі значення ваг означають сильну впливовість, тоді як невеликі значення ваг вказують на слабку впливовість. Під час навчання нейронної мережі синаптичні

ваги підлаштовуються таким чином, щоб досягти бажаного виходу. Синаптичні пороги визначають наскільки потужним має бути вхідний сигнал, щоб активувати нейрон і спричинити вихідний сигнал. Якщо сума зважених вхідних сигналів перевищує поріг, нейрон стає активним і генерує вихідний сигнал. Якщо сума сигналів не досягає порогового значення, нейрон залишається неактивним.

Навчання нейронної мережі - це процес налаштування вільних параметрів, за допомогою моделювання середовища, в яке ця мережа вбудована. Розрізняють два види навчання – з учителем та без. Перший вид припускає, що для кожного вхідного вектора існує цільовий вихідний вектор і ваги параметрів змінюються доки не буде отримано прийнятний рівень наближеності виходу до цільового вектора. Навчання без вчителя складається тільки з вхідних векторів і алгоритм намагається підлаштувати ваги так, щоб отримати схожий вихід від схожих вхідних векторів [36].

До *переваг* нейронних мереж можна віднести здатність до вивчення складних залежностей. Вони здатні розпізнавати різні шаблони та здійснювати складні обчислення, що робить їх потужним інструментом для розв'язання різноманітних завдань. Наступним плюсом є їх універсальність та можливість адаптуватися до різних видів завдань таких як класифікація, регресія, кластеризація. Також нейронні мережі можуть ефективно працювати з неструктурованими даними.

Основним *недоліком* нейронних мереж є необхідність мати великий набір даних для ефективного навчання. Без достатньої кількості інформації можуть виникнути проблеми з перенавчанням або недостатньою узагальненою здатністю моделі. Нейромережі є вимогливими до обчислювальних ресурсів через високу складність та важкість. Ще одним мінусом є погана інтерпретованість, через що іноді їх називають «чорними скриньками».

CatBoost (Categorical Boosting) - це алгоритм машинного навчання, який використовує градієнтний бустінг для розв'язання задач класифікації та регресії. Основна особливість CatBoost полягає в тому, що він ефективно працює з

категоріальними ознаками без необхідності їх попереднього перетворення на числові значення. Він автоматично обробляє категоріальні ознаки, використовуючи методи, такі як усереднення на листі, статистику Байєса та категоріальне кодування.

CatBoost використовує градієнтний бустінг на деревах рішень для покращення якості прогнозів. Градієнтний бустінг - це ітеративний алгоритм, який починає зі створення простої базової моделі (дерева рішень) і поступово додає нові моделі, які коригують помилки попередніх моделей. Цей процес продовжується до досягнення оптимальної прогностичної точності [37].

Переваги CatBoost:

- Ефективність у роботі з великими наборами даних і високою швидкістю навчання;
- Зменшення ризику перенавчання за рахунок вбудованої регуляризації і використання стохастичного градієнтного спуску;
- Можливість працювати з різними типами даних, включаючи числові, категоріальні і текстові ознаки;
- Підтримка паралельної обробки і багатопотокового навчання.

До *недоліків* відносяться:

- Високе споживання пам'яті, особливо при роботі з великими наборами даних;
- Важкість інтерпретації результатів. Ваги ознак та важливість ознак можуть бути складні для розуміння через використання багатьох дерев та методів оптимізації.

2.3. Показники оцінки якості моделей

Одним з важливих етапів роботи з моделями перед початком їх практичного застосування є оцінка їх якості. Просте вимірювання точності є недостатнім, оскільки воно не враховує імовірність перенавчання, коли модель запам’ятовує тренувальні дані, але при цьому вона не здатна точно обробити нову інформацію, що в кінці кінців призводить до неточного прогнозу. Отже, для оцінки точності моделей машинного навчання в даному дослідженні ми будемо використовувати

ряд показників, які забезпечать об'єктивну оцінку та допоможуть зрозуміти на скільки точні прогнози зможемо отримати на реальних даних, які не використовувались у навчальній виборці [38].

Матриця невідповідностей (Confusion matrix) – таблиця що використовується з метою оцінки ефективності класифікатора (моделі класифікації) на тестових даних для яких відомі значення залежної змінної. Приклад наведений в табл. 2.1.

Таблиця 2.1

Структура матриці невідповідностей

		Прогнозований клас	
		Negative	*Positive*
Реальний клас	*Negative*	TN (True Negative)	FP (False Positive)
	Positive	FN (False Negative)	TP (True Positive)

Джерело: [39]

Прогноз для кожного випадку може потрапити в одне з чотирьох полів матриці в залежності від того, чи збігається він з реальність, чи ні. В рядках зазначається реальний клас, а в стовпцях – прогнозований. Отже матриця складається з чотирьох елементів [39]:

- True Positive (TP) – істинно позитивні випадки. В межах даного дослідження це випадок, коли класифікатор передбачив що користувач буде китом, і він дійсно став китом.
- True Negative (TN) – істинно негативні випадки. Класифікатор передбачив що користувач не буде китом і він в реальності ним не став.
- False Positive (FP) – хибно позитивні випадки. Класифікатор передбачив що користувач буде китом, але в реальності він ним не став. Також відома як помилка першого роду.
- False Negative (FN) – хибно негативні випадки. Класифікатор передбачив що користувач не буде китом, але він ним став. Також відома як помилка другого роду.

Загалом матриця невідповідностей дає можливість зрозуміти типи помилок, які робить модель при класифікації. В подальшому дані з неї можна використовувати для обчислення інших оцінок, наприклад точність, чутливість, специфічність та інші.

Оцінка точності – показує загальну точність моделі та вимірюється як для тренувальної, так і для тестової вибірки. Обраховується як відношення кількості правильно класифікованих випадків до їх загальної кількості, за формулою:

$$Accuracy = \frac{TP + TN}{total}$$

Проте точність є загальною оцінкою, яка не є оптимальною метрикою для роботи з розбалансованими моделями класифікації. Це важлива умова, оскільки в даному дослідженні також спостерігається значний дисбаланс класів. Тому при використанні оцінки потрібно обережно інтерпретувати результати та додатково користуватись іншими показниками [40].

ROC крива (Receiver Operating Characteristics) – графік оцінки бінарної класифікації шляхом візуалізації відношення True Positive Rate (чутливість) по осі Y до False Positive Rate (специфічність) по осі X при різних порогових значеннях.

$True\ Positive\ Rate = \frac{TP}{TP+FN}$ – показує відношення правильно класифікованих позитивних об’єктів до загальної кількості позитивних об’єктів.

$False\ Positive\ Rate = \frac{FP}{FP+TN}$ – показує відношення помилково класифікованих негативних об’єктів до загальної кількості негативних об’єктів.

Порогове значення – значення яке використовується для перетворення імовірності належності об’єкта до певного класу, що є виходом моделі на власне бінарний розподіл. Тобто, це межа, яку потрібно перетнути, щоб класифікувати об’єкт до позитивного класу. Часто оптимальне порогове значення для незбалансованої моделі знаходиться шляхом максимізації F1-score, що відповідає за досягнення потрібного балансу між точністю та повнотою.

Чим ближче ROC крива знаходиться до лівого верхнього кута – тим кращою є модель класифікації.

AUC – площа під ROC кривою, використовується як числова метрика оцінки класифікатора. Чим більша площа, тим краща ефективність. Максимальна площа під кривою дорівнює 1 – значення, до якого ми прагнемо. Якщо значення менше за 0.5, це свідчить про те що класифікатор не є придатним і працює гірше ніж вгадування.

Проте ROC крива також не є кращим методом оцінки для розбалансованої моделі, особливо, коли потрібно точно прогнозувати менший клас, що релевантно для даного дослідження.

Precision-Recall Curve – крива оцінки бінарної класифікації, що відображає залежність між точністю (precision) та повнотою (recall) за умови різних порогових значень. Візуально показує на скільки зменшується точність при збільшенні повноти. Даний метод оцінки є особливо точним при оцінці моделей з розбалансованими класами [40].

$Precision = \frac{TP}{TP+FP}$ – показує відношення правильно класифікованих позитивних об’єктів до усіх позитивно класифікованих об’єктів.

$Recall = TPR = \frac{TP}{TP+FN}$ – показує відношення правильно класифікованих позитивних об’єктів до усіх фактично позитивних об’єктів. Тобто, на скільки добре модель знаходить позитивні екземпляри і не пропускає їх.

Для числової оцінки кривої можна обраховувати площу під нею – *AUC для PRC*. Вона вимірює середнє значення площі під PRC кривою, яка показує, наскільки добре модель здатна зберігати високу точність при високій повноті. Значення може бути від нуля до одиниці, де наближення до одиниці говорить про кращу модель. При цьому значення менше ніж 0.5 вказує на незадовільну модель, яка прогнозує гірше ніж випадковий класифікатор.

F1-Score - метрика, що також використовується для вимірювання балансу між точністю та повнотою в задачі класифікації. Вона об'єднує ці дві метрики в одне число, яке відображає загальну ефективність класифікатора за формулою:

$$F1\ Score = \frac{2 * (precision * recall)}{precision + recall}$$

Величина F1-Score також може бути трактована як середнє гармонічне значення між точністю і повнотою. Вона дає усереднену оцінку ефективності моделі, зважаючи на обидві метрики.

F1-Score особливо корисний у випадках, коли класи нерівномірно представлені або коли важно досягти балансу між точністю і повнотою. Він дозволяє знаходити компроміс між цими двома метриками, що є важливим у багатьох задачах класифікації [40].

РОЗДІЛ 3. ЗАСТОСУВАННЯ МАШИННОГО НАВЧАННЯ ДЛЯ КЛАСИФІКАЦІЇ КОРИСТУВАЧІВ ДОДАТКУ

3.1. Визначення критерію класифікації користувача як кита

Дані, використані в цьому дослідженні надані українською IT компанією OBRIO, флагманським продуктом якої є мобільний додаток що надає послуги онлайн спілкування з астрологами та іншими експертами [41]. Важливо зазначити, що ці дані піддалися попередній обробці та шифруванню і не відповідають реальним показникам діяльності компанії, але вони використані в рамках дослідження з метою аналізу та отримання уявлення про можливі взаємозв'язки та результати застосування моделей машинного навчання.

Перш за все потрібно визначити хто такий кит за допомогою числової характеристики. Це може залежати від контексту та конкретної задачі, а також від доступних даних та обраної моделі класифікації. Зазвичай визначення критерію базується на певних характеристиках чи поведінці користувача, які вказують на його активність, значну кількість трансакцій, часті запити до системи або інші фактори, що свідчать про його значущість або вплив.

Для задачі класифікації користувачів трансакційної моделі, можна використовувати такі ознаки як загальна кількість трансакцій, сума грошових операцій, частота використання сервісу, активність в певному періоді часу тощо. За допомогою певних порогових значень цих ознак можна визначити, чи відноситься користувач до категорії кита. Варто також зауважити що визначення параметру значно залежить від самого продукту та ключових послуг, які він надає.

У процесі вибору цільової метрики для даного дослідження були розглянуті наступні показники, які можуть свідчити про активність та значущість користувача в додатку:

Кількість часу проведеного в додатку. Цей показник вказує на те, скільки часу людина витрачає на користування продуктом. Проте, сама кількість проведеного часу не завжди є надійним показником цінності, оскільки клієнт

може лише переглядати контент без здійснення покупок або взаємодії з іншими функціями додатку, які є основними і містять в собі найбільшу цінність.

Кількість покупок. Цей показник відображає кількість трансакцій, зроблених користувачем в додатку. Проте, важливо враховувати не лише кількість, а й вартість, оскільки невеликі покупки можуть не вказувати на значну цінність користувача для бізнесу.

Кількість чатів із спеціалістами. Цей показник вказує на активність користувача в чатах зі спеціалістами. Проте, важливо враховувати якість цих взаємодій та їх вплив на задоволеність користувача.

Виручка, яку приніс користувач. Цей показник є прямим і конкретним показником вартості користувача для бізнесу. Виручка відображає фінансовий внесок користувача через здійснені покупки або інші дії в додатку. Використання цієї метрики дозволяє зорієнтуватися на безпосередній вплив користувача на дохід компанії.

З урахуванням можливих спекуляцій та обґрунтованості, вибір для цього дослідження був зроблений на користь метрики виручки. Ця метрика є простою, зрозумілою та точно відображає внесок користувача в бізнес. Її використання допомагає встановити чіткі цілі та спрямувати зусилля на досягнення фінансових показників у контексті розуміння користувачів та їх впливу на продукт.

Також цією метрикою досить важко спекулювати, хоча теж можливо, за рахунок рефандів чи чарджбеків. Але вона є найбільш робастою з перелічених, тому і була обрана як цільова, яка описує користувачів за допомогою чисел.

Варто взяти до уваги, що кожен користувач прийшов в додаток у різний період часу, тому вони знаходяться в різних умовах. Наприклад користувач який прийшов рік тому мав цілий рік на те щоб користуватись додатком і генерувати виручку для компанії, на відміну від користувача який прийшов місяць тому. Отже, для того щоб створити однакові умови для всіх клієнтів було обрано вікно в 3 місяці, протягом яких будемо рахувати виручку з користувача. Це означає що для кожного клієнта в дослідженні будемо брати тільки перші 3 місяці його

життя в додатку і рахувати виручку тільки за цей час [42]. Таким чином не залежно від того, коли прийшов користувач – умови будуть однакові.

Для того щоб зрозуміти, якою є межа метрики виручки за 3 місяці для китів та звичайних користувачів було побудовано графік на якому по осі х знаходяться групи користувачів за величиною принесеної виручки. По осі у відкладені відсотки. Блакитна площа означає відсоток кумулятивно згенерованої виручки від усієї певною групою клієнтів. А помаранчева лінія означає власне кумулятивну кількість клієнтів.

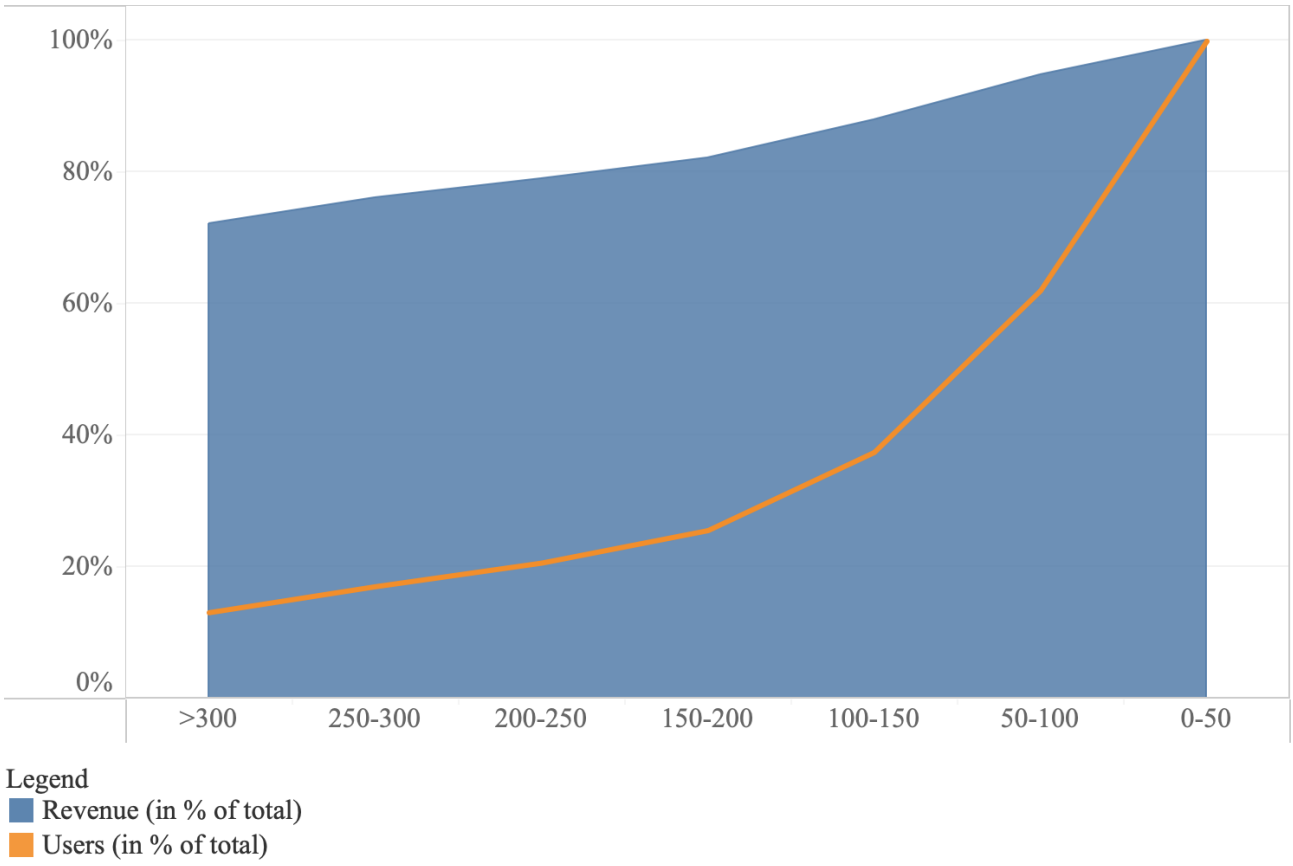

Рис. 3.1. Криві кумулятивної виручки та кількості користувачів

Джерело: створено автором

З рис. 3.1, що відображає залежність між кумулятивною виручкою і кумулятивною кількістю користувачів, можна зробити наступні спостереження. Користувачі, у яких величина виручки перевищує 150 доларів, складають лише 26% від усіх клієнтів, проте вони приносять 82% від загального доходу компанії. Це свідчить про велику цінність цієї групи користувачів для бізнесу.

На основі цих результатів можна зробити висновок, що виручка більше 150 доларів є важливим показником, який відділяє високодохідних користувачів від інших. Оскільки цей поріг виручки є значимим з точки зору впливу на загальний дохід компанії, його можна використовувати для трансформації грошової змінної на бінарну. Така бінарна змінна буде слугувати залежною змінною для подальшого моделювання та класифікації користувачів на китів і не китів на ранніх етапах життя продукту.

3.2. Огляд структури даних

Набір даних складається з 34 колонки, 8 з них типу object та 26 float/int. Розглянемо ці характеристики користувача з розподілом по типу.

Технічні характеристики:

- user_id – унікальний ідентифікатор користувача;
- media_source – маркетинговий канал з якого був залучений користувач;
- platform – платформа користувача (iOS, Android);
- pushes_enabled – бінарна величина, що відповідає на запитання, чи дав користувач доступ на відправку йому пуш повідомлень;
- language – мова, яка використовується в додатку;
- paying – бінарна величина, що відповідає на питання, чи є користувач платником в інших сервісах в додатку, що не стосуються досліджуваних онлайн консультацій.

Демографічні характеристики:

- country – країна користувача;
- age – вік;
- gender – стать;
- relationship – статус відносин, в якому знаходиться користувач.

Поведінкові характеристики.

Тут знаходяться дані про поведінку користувача протягом перших 2 тижнів його життя в додатку. Це було зроблено для того, щоб всіх користувачів поставити в однакові умови, а також, якщо в майбутньому ці показники будуть слугувати вхідними даними для моделі, то для їх збору потрібно було відносно

небагато часу. Серед них теж можна виділити підгрупи: характеристики що стосуються поповнень користувача, та ті, які стосуються чатів користувача.

- revenue_3m – виручка, що приніс користувач за перші три місяці, в подальшому буде використана як база для створення залежної змінної моделей;
- first_chat_day – кількість днів що пройшла він встановлення додатку до першого чату;
- first_chat_type – тип першого чату (онлайн / офлайн / месенджер);
- first_chat_duration – довжина першого чату в секундах;
- first_3chats_duration – довжина перших трьох чатів в секундах;
- chats_duration_7d – довжина чатів протягом перших 7 днів в додатку;
- chats_duration_2w – загальна тривалість чатів за перші 2 тижні після встановлення додатку;
- chats_number_2w – кількість чатів користувача за перші 2 тижні після встановлення додатку;
- specialists_count_2w – кількість спеціалістів, з якими спілкувався користувач протягом перших 2 тижнів;
- avg_review_rate_2w – середня оцінка користувачем його чатів зі спеціалістами протягом перших 2 тижнів;
- first_purchase_day – кількість днів що пройшла від встановлення додатку до першої покупки;
- first_purchase_revenue – виручка з першої покупки користувача;
- third_purchase_day – кількість днів, що пройшла з встановлення додатку до третьої покупки;
- first_3purchases_revenue – виручка з перших трьох покупок;
- revenue_1d – виручка, яку приніс користувач в свій перший день;
- revenue_3d – виручка, яку приніс користувач в перші три дні;
- revenue_7d – виручка за перші 7 днів;
- revenue_2w – виручка за перші 2 тижні;
- purchases_number_2w – кількість покупок користувача в перші 2 тижні;

- paypal_purchases_2w – чи були в користувача покупки методом PayPal;
- google_pay_purchases_2w – чи були покупки методом Google Pay;
- card_purchases_2w – чи були покупки які оплачувались картою;
- recurrent_purchases_2w – чи були в користувача рекурентні платежі (повторні платежі певним видом оплати);
- applepay_purchases_2w – чи були покупки методом Apple Pay.

Дані були зібрані і опрацьовані через базу даних Vertica. *Vertica Database* — це стовпчаста база даних, призначена для обробки великих обсягів даних і забезпечення швидкої роботи. База даних покращує продуктивність запитів порівняно з традиційними системами реляційних баз, забезпечує високу доступність і значну масштабованість на стандартних корпоративних серверах [19].

Безпосередньо з Vertica потрібний датасет був отриманий за допомогою SQL запиту, наведеного на в додатку А.

Розглянемо детальніше структуру даних. На рис. 3.2 зображений відсоток китів серед усіх користувачів по країнам. В дужках виведена абсолютна кількість китів. Бачимо що найбільший відсоток для країн Індонезія, Аргентина, Пакистан. Проте якщо дивитись в абсолютних величинах, то найбільше перспективних платників в США та Канаді.

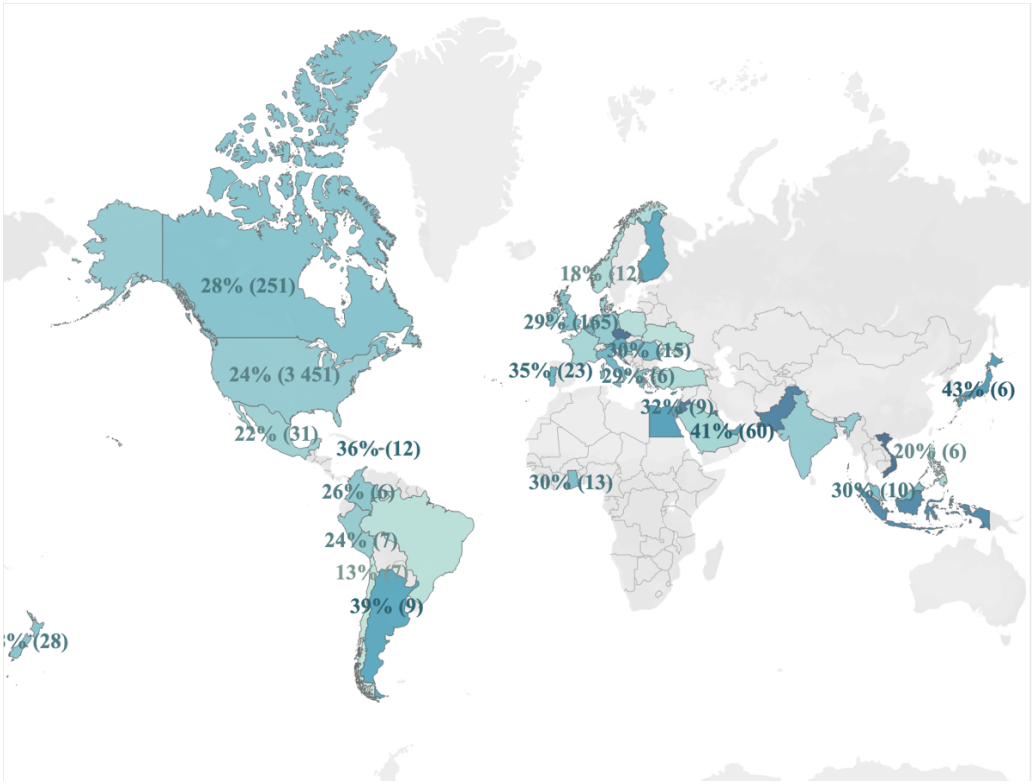

Рис. 3.2. Відсоток китів серед усіх користувачів за країнами

Джерело: створено автором

На рис. 3.3 бачимо відсоток топ платників з розбивкою по джерелу залучення трафіку. Серед користувачів, що прийшли органічно відсоток високоприбуткових користувачів вищий ніж для тих, хто залучений на платній основі. Це досить логічно, оскільки органічні юзери є більш зацікавлені в продукті, бо шукали його власноруч.

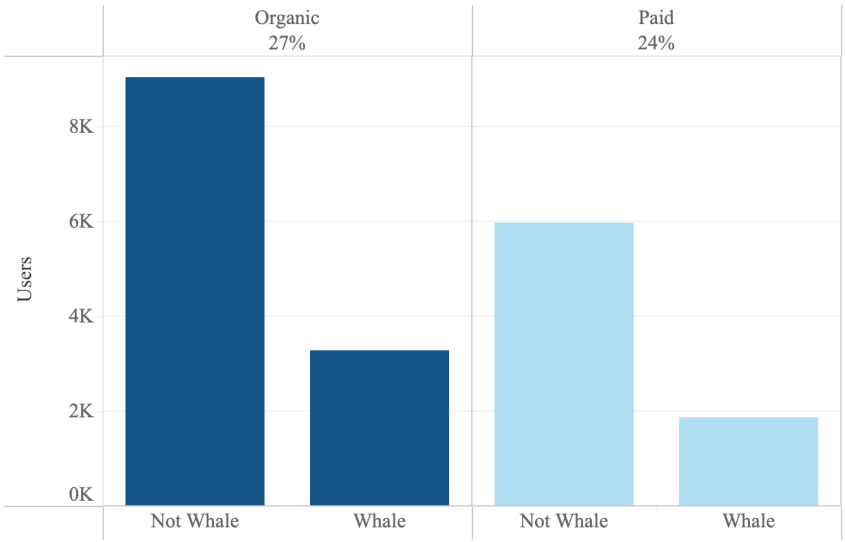

Рис. 3.3. Відсоток китів за джерелом залучення

Джерело: створено автором

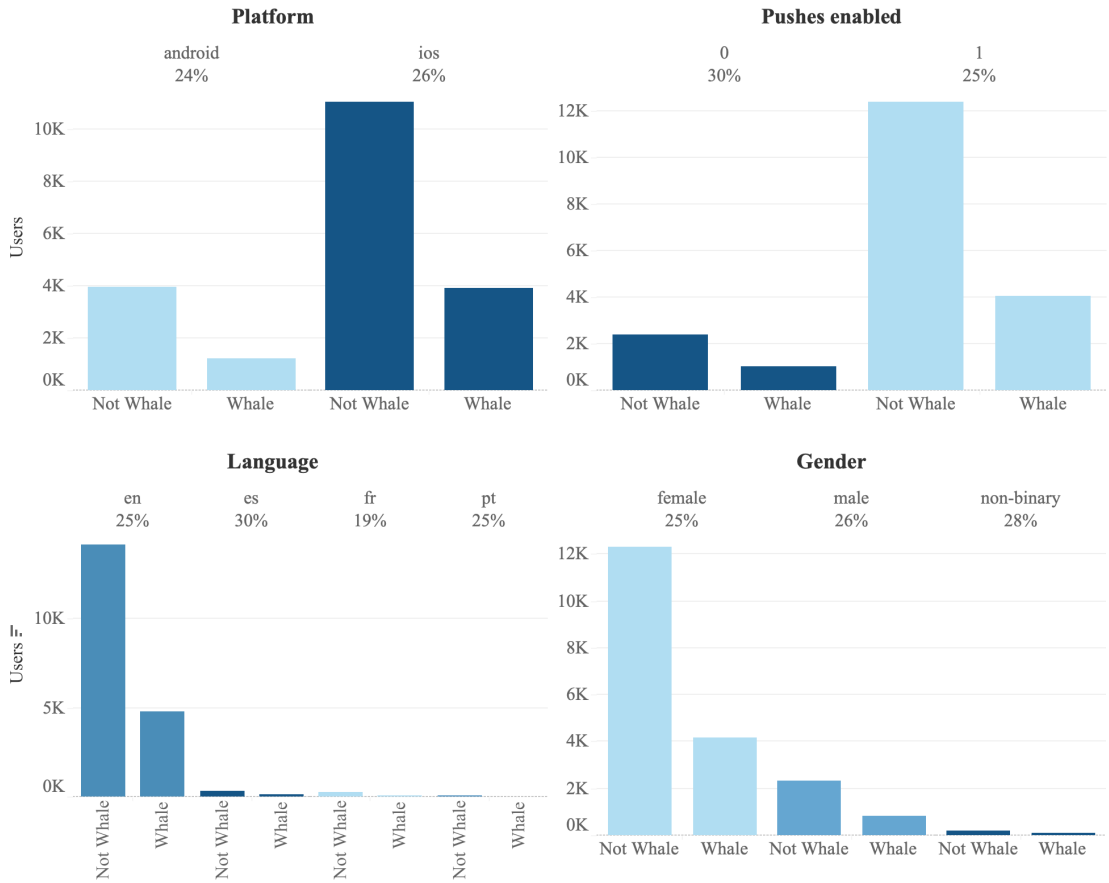

Рис. 3.4. Відсоток китів за платформою, дозволом на пуші, мовою, статтю

Джерело: створено автором

З рис. 3.4 бачимо що значно більший відсоток китів як і в абсолютній кількості на платформі iOS. Також можна помітити, що серед користувачів які не дозволили відправляти собі пуш-повідомлення більший відсоток китів. Проте абсолютне значення вище для групи яка дала дозвіл на пуш повідомлення. Наступною ознакою є мова. Найбільший відсоток китів для іспанської локалі, при чому власне користувачів звідти не багато. Дивлячись на стать, то не можна сказати, що в якійсь із груп відсоток китів вищий. Проте абсолютної кількості користувачів більше жіночої статі.

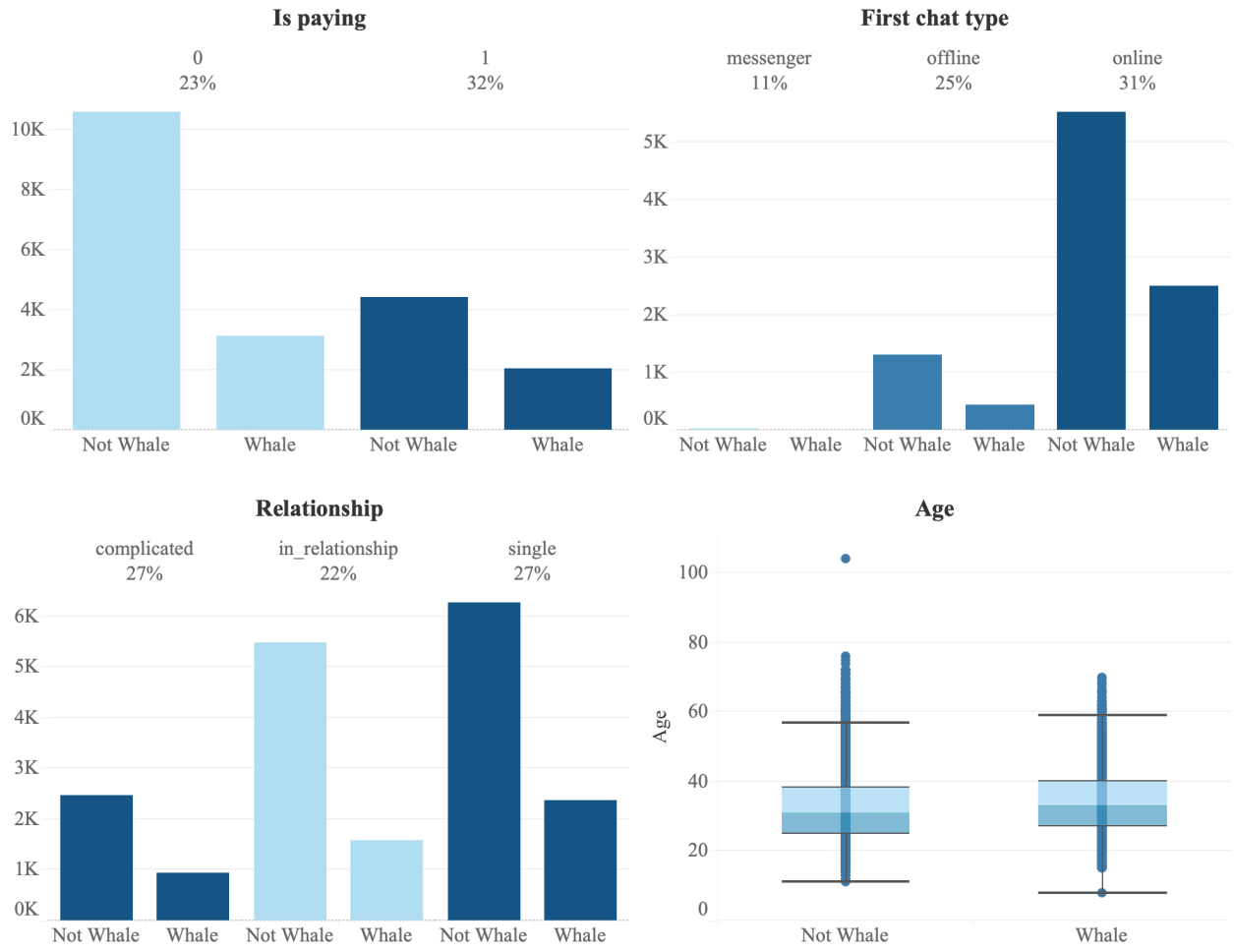

Рис. 3.5. Відсоток китів за оплатами на інших частинах продукту, типом першого чату, сімейному статусу, віку

Джерело: створено автором

Продукт, користувачі якого досліджуються є комплексним і містить не лише консультації, а й інші частини, за які також можна платити. Рис. 3.5 показує, що саме серед платників відсоток китів більший 32% проти 23% для не платників. Поясненням цьому є те, що користувачі, які вже заплатили мають більшу зацікавленість продуктом та є більш платоспроможними. Якщо дивитись на тип першого чату, то найбільш ефективним з точки зору активації є онлайн чат, коли спеціаліст і клієнт спілкуються в реальному часі. Сімейний статус говорить про те, що більш перспективними є групи без пари та ті хто вважає, що в нього складнощі у відносинах. Різниці у віку не помітно.

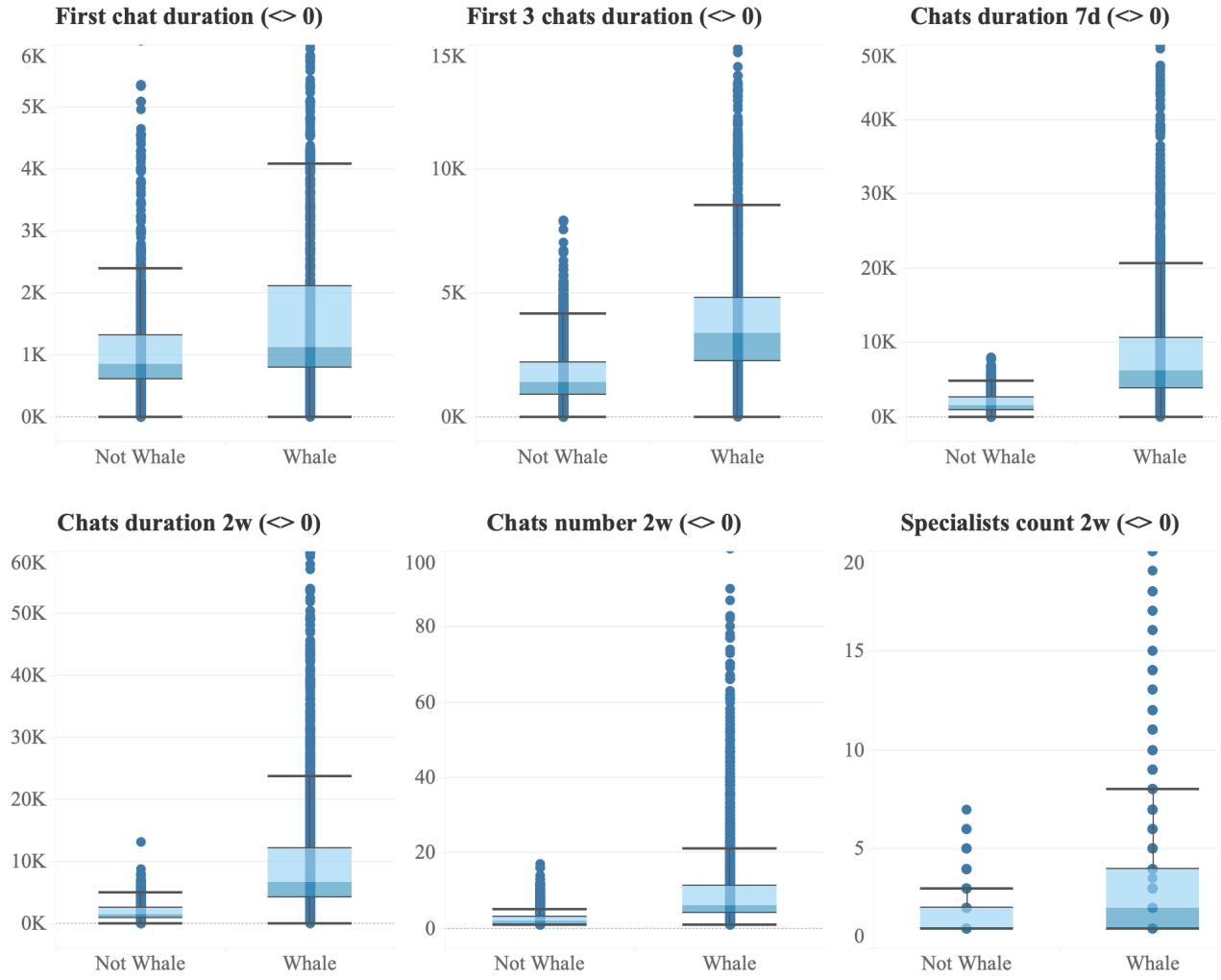

Рис. 3.6. Коробкові графіки для показників пов'язаних з тривалістю чатів протягом різних періодів та кількістю спеціалістів, з якими були чати

Джерело: створено автором

Рис. 3.6 говорить про значну різницю в тривалості першого чату, перших трьох чатів, тривалості чатів за перші 7 та 14 днів після встановлення додатку для групи китів та звичайних користувачів. Як медіанні значення, так і 1-3 квартилі мають значно більші показники. Те ж саме можна сказати і про кількість чатів за перші два тижні. Також графік кількості спеціалістів показує, що медіана і третій квартиль вищі для групи китів, але мінімальне значення таке ж саме, що означає що є високодохідні користувачі, які мали чат лише з одним спеціалістом.

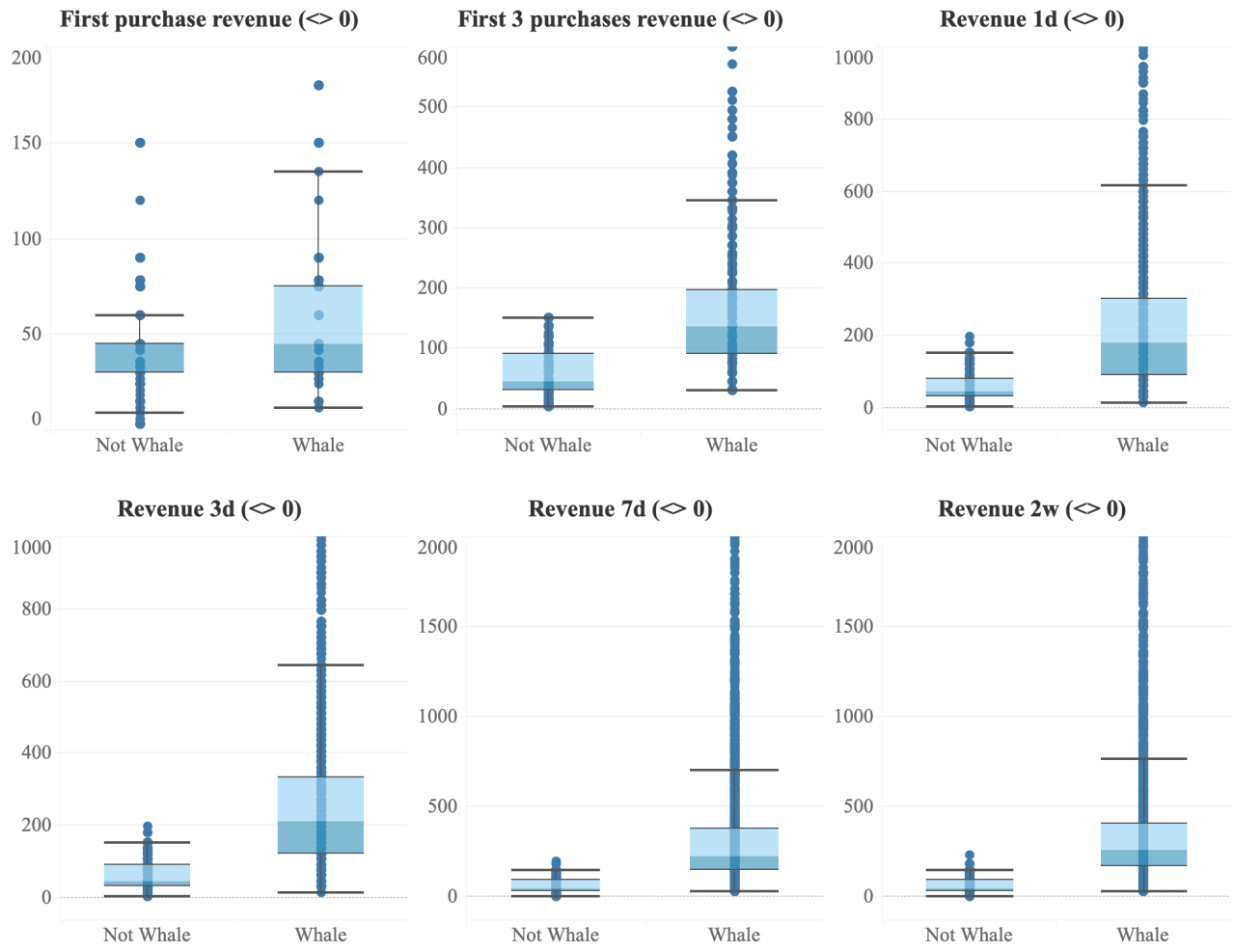

Рис. 3.7. Коробкові графіки для показників пов’язаних з виручкою

Джерело: створено автором

Перейдемо до показників, пов’язаних з виручкою. З рис. 3.7 бачимо що як і у випадку в чатами, кількість виручки за різні періоди виглядає досить схоже: для групи китів коробкові графіки є ширшими та знаходяться вище. Імовірніше за все вони досить сильно корелюють між собою, тому в подальшому потрібно буде видалити частину з них.

Наступний рис. 3.8 показує різницю в кількості покупом між високодохідними і звичайними користувачами. Бачимо, що коробкова діаграма для китів знаходиться вище і є значно ширшою, тобто має більше медіанне значення та більший розмах.

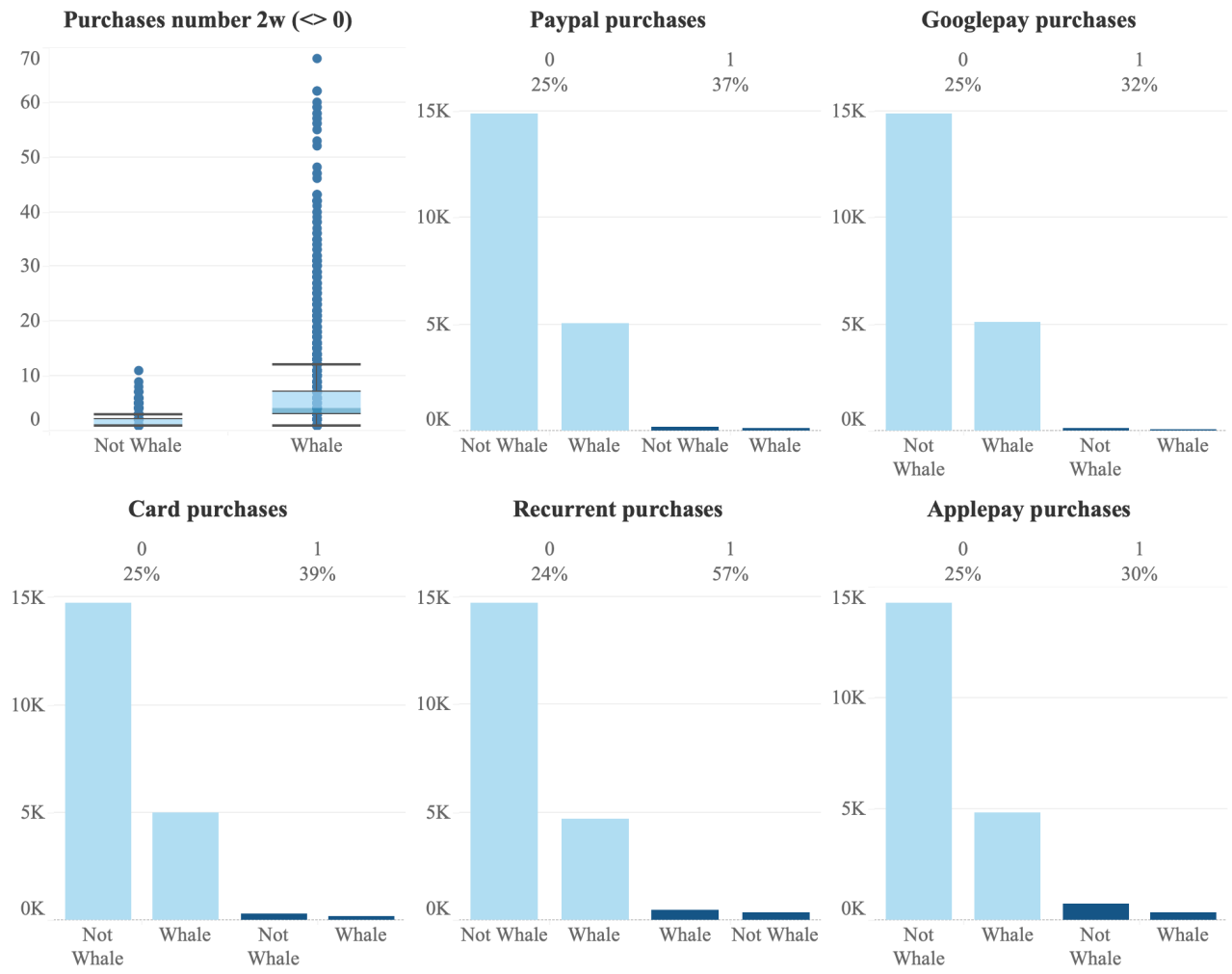

Рис. 3.8. Розподіл користувачів за покупками та різними типами оплати

Джерело: створено автором

Також на рис. 3.8 зображені розбивки по типу користувачів для різних типів оплати: PayPal, card, Apple Pay, Google Pay. Закономірно, що для користувачів, які мали хоч одну покупку будь-яким з цих методів оплати імовірність стати китом буде вищою. Проте цікавим показником є наявність рекурентних платежів. Для нього різниця відсотків китів дуже відрізняється: 57% китів у групі, що мала повторні платежі та 24% китів у групі, що не мала. Отже наявність рекурентних покупок може бути хорошим показником для моделі, як і метрики виручки та тривалості чатів.

3.3. Попередня обробка даних

На цьому етапі було проведено попередню обробку даних, їх очищення він викидів, заповнення або видалення рядків/стовпців з відсутніми значеннями.

Python код наведений в додатку Б. Для того щоб визначити колонки, в яких потрібно робити заміни було виведено їх описову статистику в табл. 3.1.

Таблиця 3.1

Описова статистика числових характеристик

Column	Count	Mean	Std	Min	Q1	Median	Q3	Max
pushes_enabled	19915	0.83	0.38	0.00	1.00	1.00	1.00	1.00
paying	20217	0.32	0.47	0.00	0.00	0.00	1.00	1.00
age	20217	32.70	9.49	8.00	25.00	32.00	38.00	104.00
revenue_3m	20217	275.58	1698.75	0.98	44.98	74.99	164.91	108114.72
first_chat_day	9785	2.28	3.33	0.00	0.00	0.00	4.00	12.00
first_chat_duration	20217	457.02	853.51	0.00	0.00	0.00	853.60	14344.00
first_3chats_duration	20217	1032.73	1828.62	0.00	0.00	0.00	1562.00	28177.60
chats_duration_7d	20217	1604.30	6192.29	0.00	0.00	0.00	1324.40	336903.60
chats_duration_2w	20217	2231.10	9070.19	0.00	0.00	0.00	1716.00	419452.00
chats_number_2w	20217	2.26	7.20	0.00	0.00	0.00	2.00	440.00
specialists_count_2w	20217	0.98	1.98	0.00	0.00	0.00	1.00	58.00
avg_review_rate_2w	1424	4.84	0.56	1.00	5.00	5.00	5.00	5.00
first_purchase_day	12288	2.32	3.36	0.00	0.00	1.00	4.00	12.00
first_purchase_revenue	20217	30.14	39.54	-0.01	0.00	29.99	44.99	449.99
third_purchase_day	3558	2.80	3.54	0.00	0.00	1.00	5.00	12.00
first_3purchases_revenue	20217	56.87	83.31	0.00	0.00	29.99	89.97	1349.97
revenue_1d	20217	34.44	111.13	0.00	0.00	0.00	29.99	5399.82
revenue_3d	20217	47.43	153.80	0.00	0.00	0.00	44.99	9899.67
revenue_7d	20217	71.95	235.78	0.00	0.00	0.00	74.99	12599.58
revenue_2w	20217	98.73	331.24	0.00	0.00	29.99	89.98	12599.58
purchases_number_2w	20217	1.88	5.19	0.00	0.00	1.00	2.00	262.00
paypal_purchases_2w	20217	0.01	0.11	0.00	0.00	0.00	0.00	1.00
googlepay_purchases_2w	20217	0.01	0.10	0.00	0.00	0.00	0.00	1.00
card_purchases_2w	20217	0.02	0.15	0.00	0.00	0.00	0.00	1.00
recurrent_purchases_2w	20217	0.04	0.19	0.00	0.00	0.00	0.00	1.00
applepay_purchases_2w	20217	0.05	0.22	0.00	0.00	0.00	0.00	1.00

Джерело: створено автором

В ході підготовки набору даних були видалені наступні нерелевантні колонки:

- user_id – унікальний ідентифікатор користувача, який не дає жодної інформації про його характеристики чи поведінку;

- first_chat_day – має близько половини значень null. Це означає, що ці користувачі не мали чату протягом перших 2 тижнів на продукті. Щоб не видаляти всі рядки де користувачі не мали першого чату змінимо значення колонки з first_chat_day на first_chat_day_exists і всі null значення стануть нулями, а інші – одиницями. Проте в таблиці також є стовпець first_chat_duration, який має значення нуль у випадку, якщо першого чату не було, тому колонка first_chat_day_exists буде дублювати інформацію. Отже, колонку first_chat_day було видалено;
- first_purchase_day – як і first_chat_day видалено через велику кількість пропущених значень.

Деякі категоріальні змінні були згруповані для зручності подальшої стандартизації:

- relationship – complicated, in_relationship, single
- country - United_States, Canada, Other, Australia, United_Kingdom
- media_source – всі значення колонки були віднесені до одної з груп: Organic, Facebook, Apple Search Ads, Others, Google, TikTok.

Заповнення пропущених значень:

- media_source заповнені значенням ‘Unknown’ оскільки це користувачі з не відслідкованим джерелом, що може виникати з певних причин, наприклад заборона на збір даних;
- country, language, gender, relationship – порожні значення були заповнені модою;
- pushes_enabled – порожні значення заповнені медіаною.

Також виходячи з описової статистики числових характеристик було оброблено такі колонки:

- Бачимо, що мінімальним віком в колонці age є 8 років, проте в додатку заборонено реєструватись людям молодше 18. Імовірніше за все відбулась помилка при записі даних, тому значення молодше 18 років замінимо на медіану – 32 роки.

Для подальшої класифікації було створено нову змінну revenue_3m_binary на основі revenue_3m з розбиттям на рівні 150 доларів. Подивимось на розподіл залежної змінної. З рис. 3.9 видно що існує різниця між величиною класів оскільки китів значно менше ніж звичайних користувачів. Це може призвести до певних обмежень у використанні методів оцінки моделей.

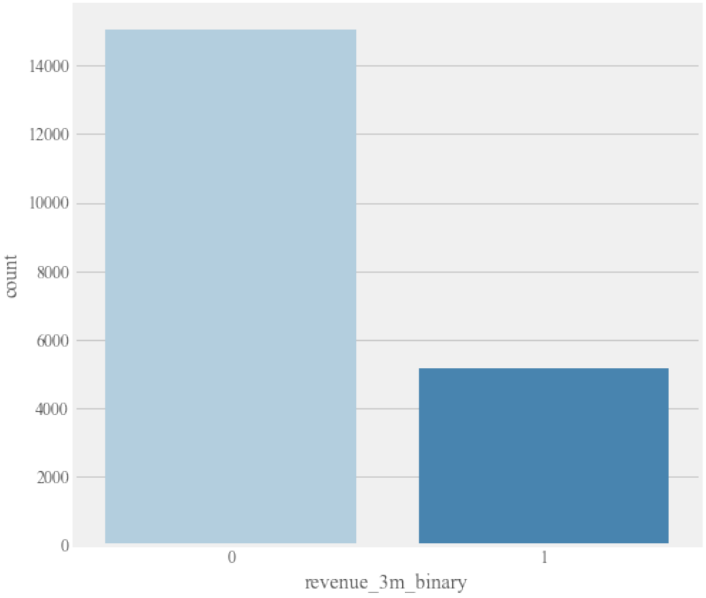

Рис. 3.9. Розподіл класів

Джерело: створено автором

Цього дисбалансу можна позбутися різними шляхами, наприклад методом повторної вибірки штучно збільшити кількість користувачів у класі китів шляхом випадкового дублювання записів для вирівнювання кількості екземплярів в обох класах. Також можна випадково відрізати частину негативного класу. Якщо використовувати другий варіант і зменшувати більший клас, то даних лишиться мало для подальшої роботи з ними. Якщо ж навпаки збільшувати менший клас, то через великий дисбаланс це може призвести до викривлення показників оцінок моделі оскільки один і той же приклад буде зустрічатись багато разів [43]. Отже було прийнято рішення не збалансовувати модель і враховувати це при оцінці моделі.

Для більшого розуміння даних подивимось на кореляцію змінних між собою на рис. 3.10.

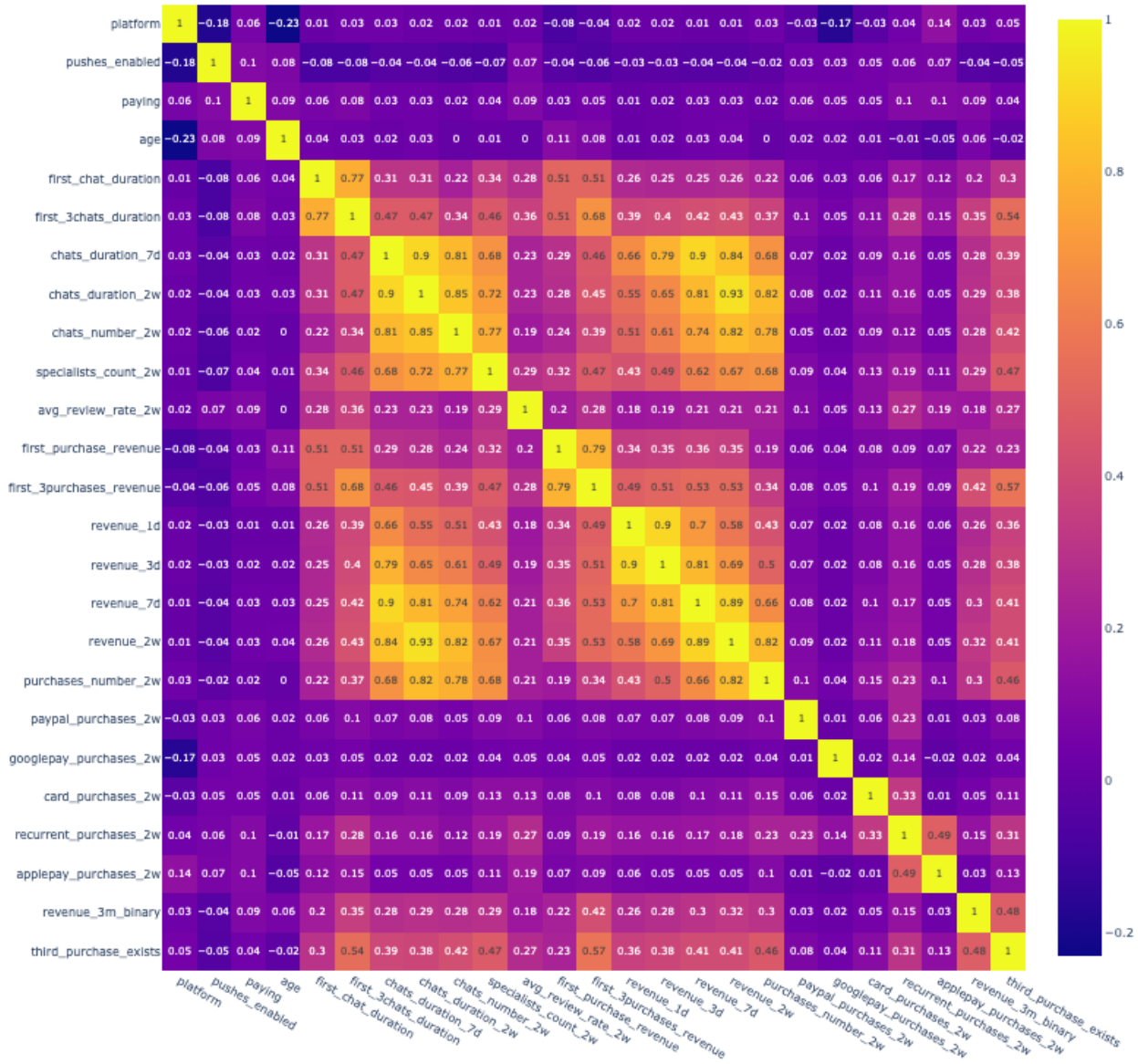

Рис. 3.10. Кореляція незалежних змінних між собою

Джерело: створено автором

Бачимо, що є метрики, що пов’язані з виручкою та чатами, які досить сильно корелюють між собою. Це може негативно впливати на модель, тому для оптимізації видалимо частину з них:

- first_chat_duration (має кореляцію з first_3chats_duration)
- chats_duration_2w (має кореляцію з chats_duration_7d, при цьому довжину чату за перші 7 днів можна визначити значно швидше, тому залишимо змінну, яка є більш вигідною з точки зору швидкості оцінки користувача на потенційного кита)

- revenue_7d (корелює з chats_duration_7d, при цьому довжина чату є більш робастою метрикою та буде більш стабільно поводитись при зміні розцінок на чати)
- revenue_1d (корелює з revenue_3d)
- chats_number_2w, purchases_number_2w, revenue_2w (корелюють з chats_duration_7d)

На рис. 3.11 бачимо що найбільше з залежною змінною корелюють наявність третьої покупки, виручка з трьох перших покупок, довжина 3 перших чатів.

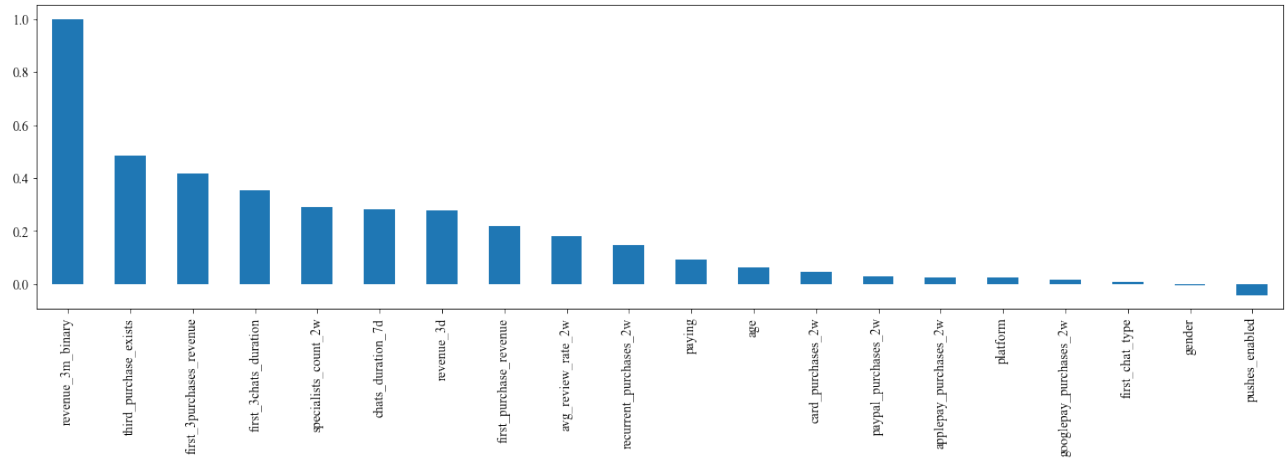

Рис. 3.11. Кореляція залежної змінної з незалежними

Джерело: створено автором

Наступним кроком є нормалізація – техніка, яка використовується для підготовки даних з метою зведення до єдиної уніфікованої шкали без втрати інформації чи спотворення значень.

Наступні колонки мають лише по два значення, тому були закодовані в бінарні цілі числа: platform, pushes_enabled.

Категоріальні дані, що приймають більше двох значень розділені на нові змінні, кожна з яких відповідає окремому значенню початкової колонки: language, gender, first_chat_type, media_source_group, country_group, relationship_group. Для цього було використано метод One-Hot Encoding [44].

Однак, при використанні цього методу, існує ризик виникнення проблеми мультиколінеарності, коли одна з змінних може бути лінійно залежна від інших

змінних. Це може стати проблемою при використанні деяких моделей машинного навчання, таких як лінійна регресія. Для уникнення цієї проблеми було видалено одну з бінарних змінних, щоб уникнути лінійної залежності між ними. При цьому ми все ще зберігаємо інформацію про всі категорії, але уникаємо проблеми мультиколінеарності.

До числових даних було застосовано Min-Max Scaling, тобто переведені в діапазон від 0 до 1. Для цього була використана формула $x\_scaled = \frac{x - min(x)}{max(x) - min(x)}$ [45].

Останнім підготовчим етапом є розділення на тестову і тренувальну вибірки за допомогою функції train_test_split з пакету sklearn.model_selection. Для тестування моделей було залишено 30%.

3.4. Навчання та оцінка моделей. Аналіз результатів класифікації користувачів

У цьому підрозділі представлено опис результатів застосування моделей та їх оцінка в контексті задачі класифікації користувачів трансакційної моделі на китів та звичайних користувачів на ранніх етапах життя на продукті.

K найближчих сусідів. З рис. 3.12 бачимо що результати застосування моделі показують прийнятну точність на рівні 0.79. Значення AUC (ROC) для даної моделі становить 0.75, що свідчить про відносно хорошу здатність розрізняти класи. Проте згадаємо про те що маємо справу з розбалансованими класами, тому детальніше звернемо увагу на криву Precision-Recall на рис. 3.13, що використовується для вимірювання точності та повноти.

Значення AUC (PRC) дорівнює 0.59, що може вказувати на меншу здатність моделі до визначення позитивних екземплярів.

F1 Score, який говорить про співвідношення точності та повноти, має значення 0.51, що є досить низьким показником. Враховуючи ці результати, модель kNN можна спробувати покращити за допомогою налаштування гіперпараметра k.

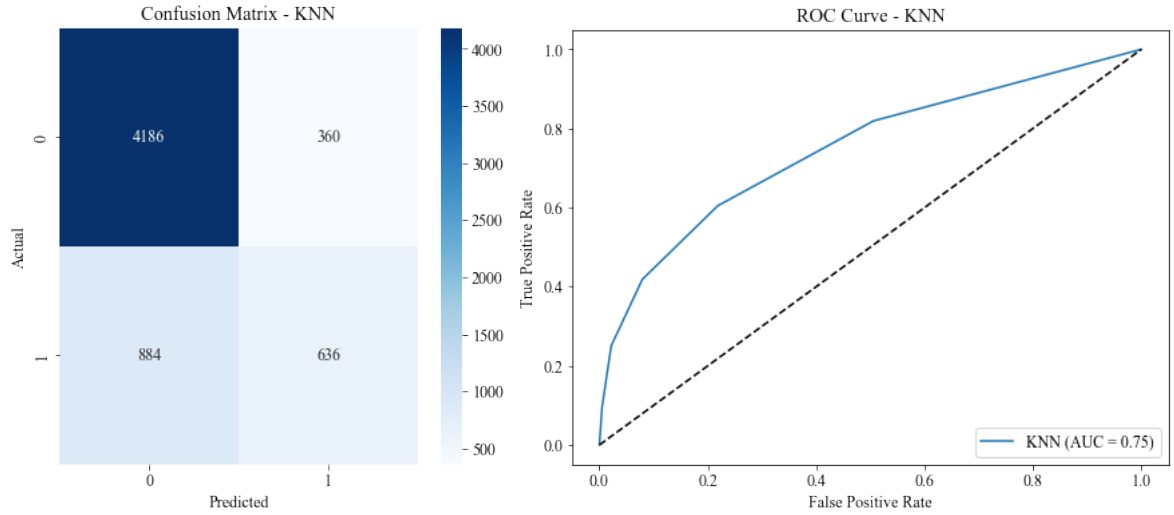

Рис. 3.12. Матриця невідповідностей та ROC крива моделі kNN

Джерело: створено автором

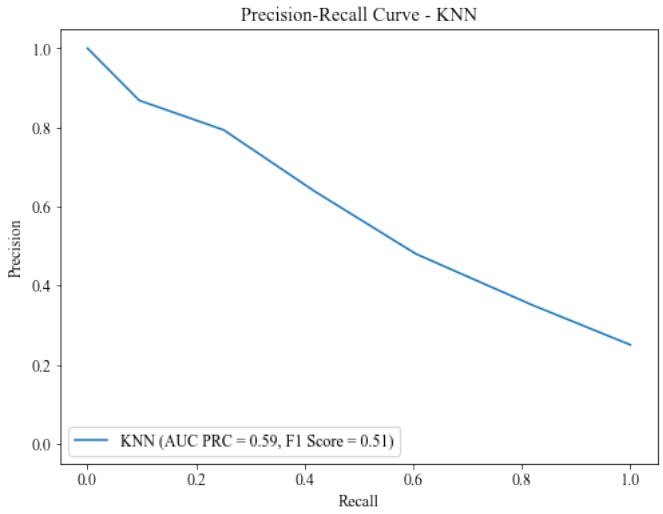

Рис. 3.13. Крива Precision-Recall моделі kNN

Джерело: створено автором

Для оптимізації алгоритму було використано GridSearchCV - метод визначення оптимальних гіперпараметрів моделі шляхом перебору всіх можливих комбінацій заздалегідь заданого простору [46]. Використовуючи крос-валідацію він знаходить комбінацію що дасть найкращу точність моделі.

Для покращення даного алгоритму були перебрані кількості найближчих сусідів у межах від 3 до 30 з кроком 1.

В результаті отримали оптимальне значення параметра 17. За його використання модель kNN досягає точності 0.81, AUC (ROC) 0.78, що бачимо на рис. 3.14. Проте якщо дивитись на показник AUC (PRC) на рис. 3.15, то він

покращився всього лише до рівня 0.62, а F1-Score взагалі не змінився, що говорить про недостатню точність моделі.

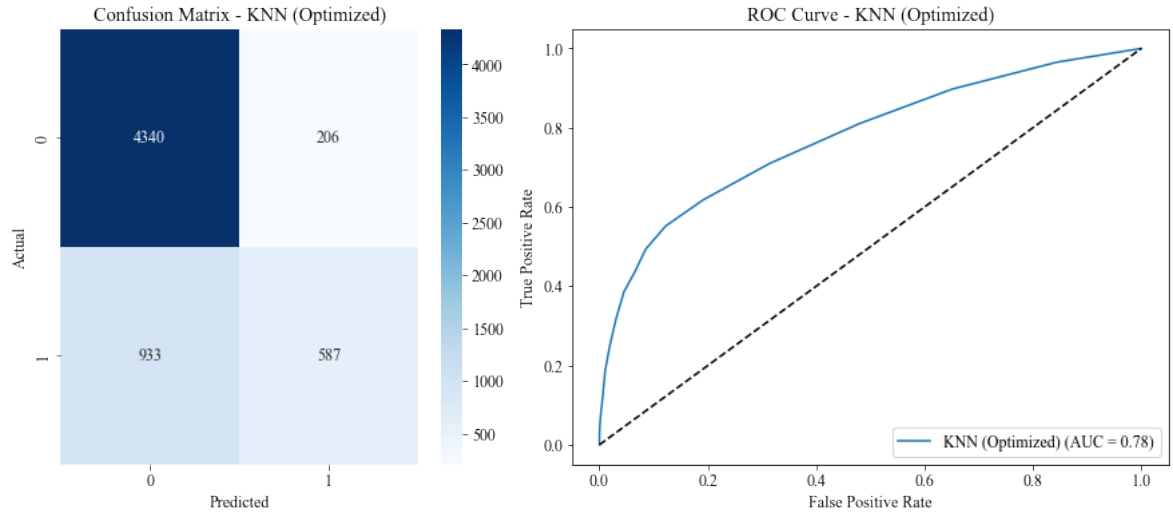

Рис. 3.14. Матриця невідповідностей та ROC крива оптимізованої моделі kNN

Джерело: створено автором

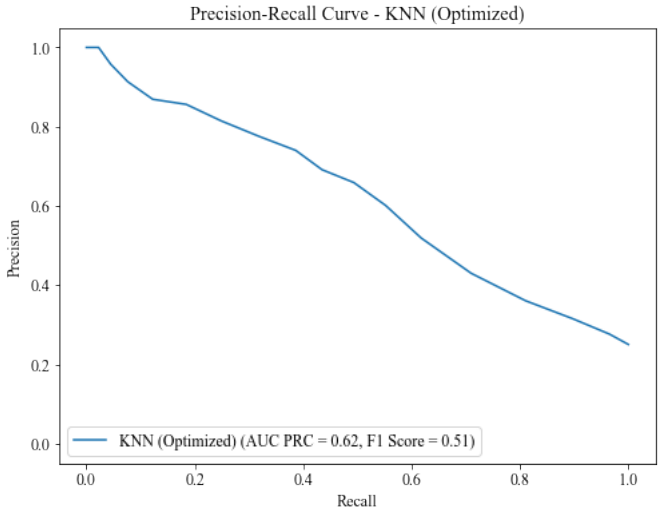

Рис. 3.15. Крива Precision-Recall оптимізованої моделі kNN

Джерело: створено автором

Логістична регресія правильно класифікує 83% прикладів. При цьому на рис. 3.16 бачимо, що AUC рівний 0.76, що вказує на те, що модель має добру роздільну здатність. Значення AUC (PRC) на рис. 3.17 рівне 0.66 вказує на помірну здатність моделі зберігати точність при визначенні позитивних класів. F1-Score = 0.60, що означає помірний баланс між точністю та повнотою моделі. Проте якщо порівнювати з попереднім алгоритмом, то значення гармонічного

середнього вже є значно кращим, беручи до уваги також те що позитивний клас є значно меншим і його досить важко класифікувати.

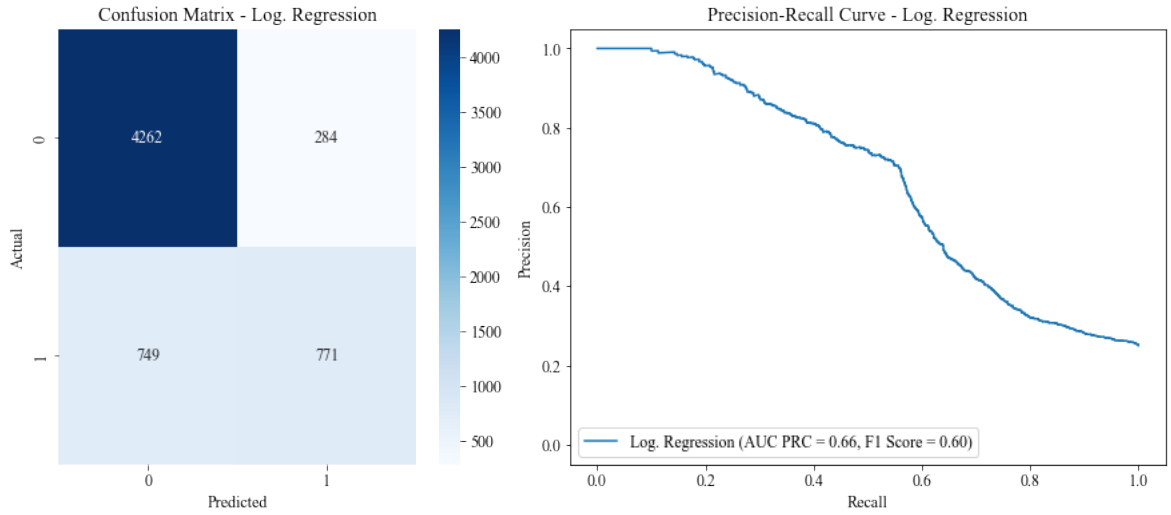

Рис. 3.16. Матриця невідповідностей та ROC крива моделі логістична регресія

Джерело: створено автором

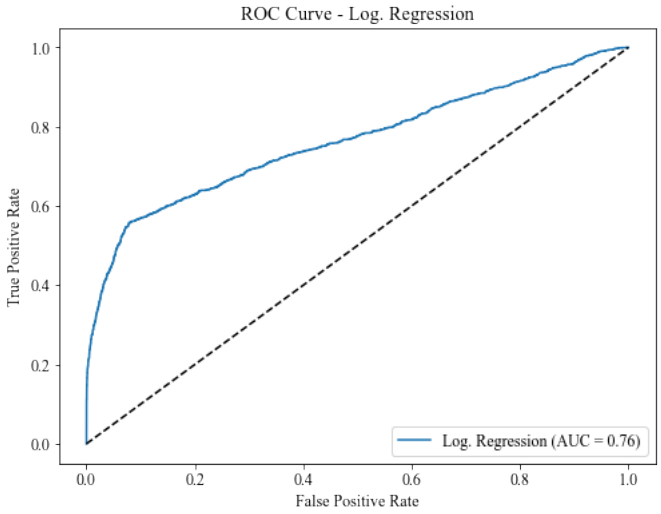

Рис. 3.17. Крива Precision-Recall моделі логістична регресія

Джерело: створено автором

Перевагою логістичної регресії є її простота в інтерпретації, тож подивимось, яким ознакам вона надала найбільшу вагу. З рис. 3.18 бачимо що найбільше на фінальний результат впливають виручка перших трьох покупок, тривалість чатів за перші 7 днів та виручка за три дні. Також вагомими ознаками є кількість спеціалістів, з якими користувач мав чати, наявність третьої покупка та вік. Це свідчить про те, що китовість користувача проявляється вже на першому тижні перебування на продукті.

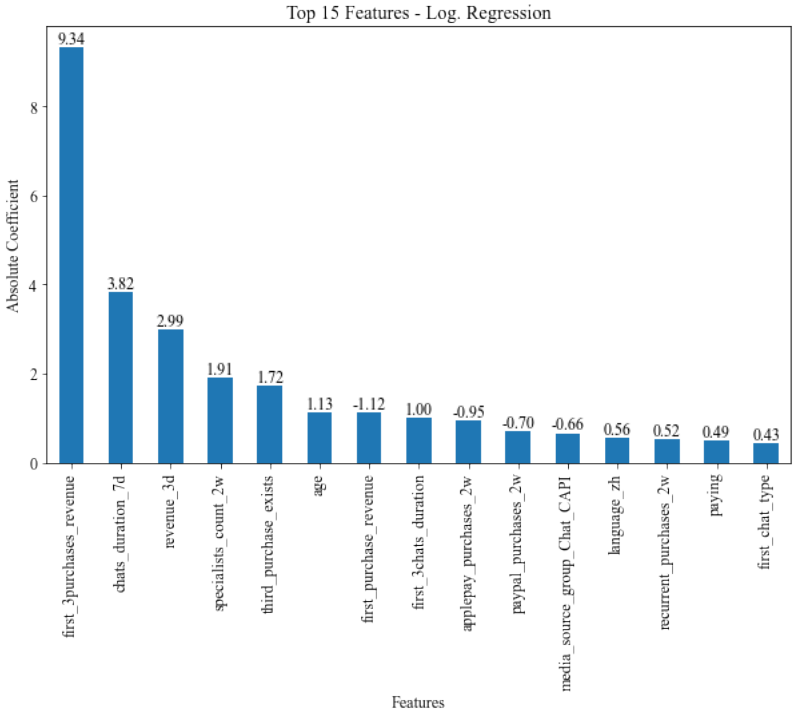

Рис. 3.18. Найбільш вагомі ознаки моделі логістична регресія

Джерело: створено автором

Спробуємо оптимізувати модель шляхом підбору гіперпараметрів. Для цього знову використаємо GridSearchCV та спробуємо підібрати потрібні penalty, що будуть штрафувати модель за дуже низькі або високі коефіцієнти перед змінними та параметр C, який є оберненою силою регуляризації. Використаємо алгоритм оптимізації saga, оскільки він підтримує регуляризацію L1 та L2 і є більш швидким та ефективним на великих наборах даних зі значною кількістю параметрів. Оптимальним виявилось використовувати регулятор L1 з C = 1.9. Це дало наступні результати моделі.

Загальна точність не покращилась, залишившись на рівні 0.83. Проте з рис. 3.19 бачимо, що AUC (ROC) зріс до 0.77. При цьому якщо говорити про криву Precision-Recall на рис. 3.20, то AUC (PRC) покращився до 0.68, а F1 Score навпаки зменшився з 0.60 до 0.58 після підбору гіперпараметрів. Оскільки ми маємо розбалансовані дані, то показник F1 є досить важливим, що не дає змоги використовувати дану модель для прогнозування.

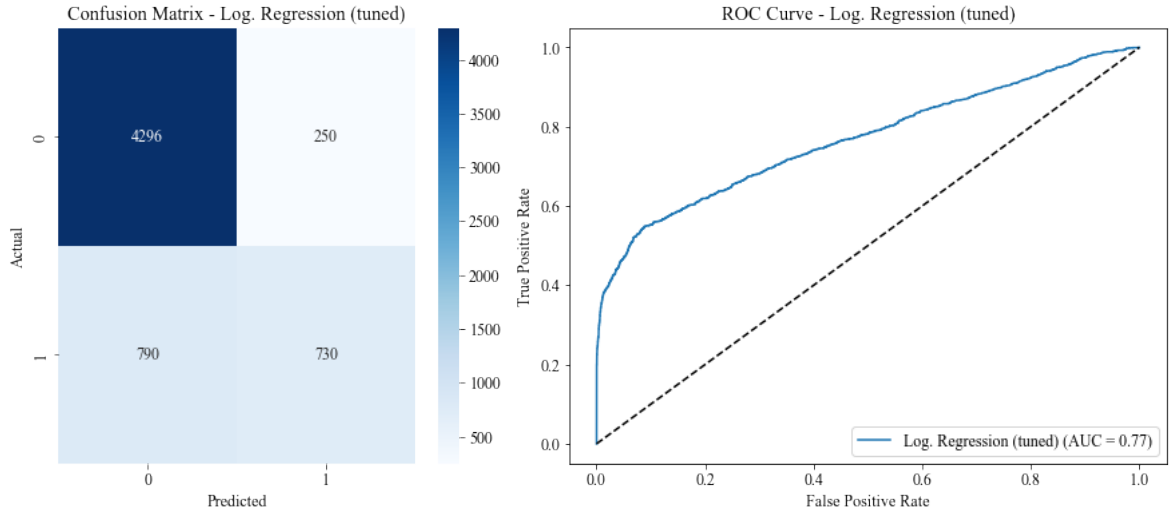

Рис. 3.19. Матриця невідповідностей та ROC крива оптимізованої моделі логістична регресія

Джерело: створено автором

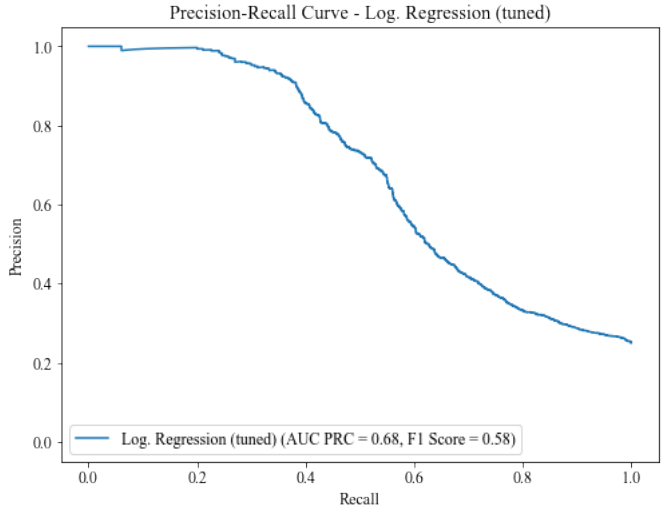

Рис. 3.20. Крива Precision-Recall оптимізованої моделі логістична регресія

Джерело: створено автором

Модель ***Випадковий ліс*** на рис. 3.21 показує високу точність на рівні 0.84 та здатність розділити класи, про що свідчить AUC (ROC) на рівні 0.82. Також, як видно з рис. 3.22, має добру здатністю зберігати точність при виявленні позитивних класів (AUC (PRC) = 0.75). F1-Score 0.65 також вказує на відносно хороший баланс між точністю та повнотою моделі. Враховуючи ці показники, можна стверджувати, що модель випадкового лісу показує хорошу ефективність при класифікації.

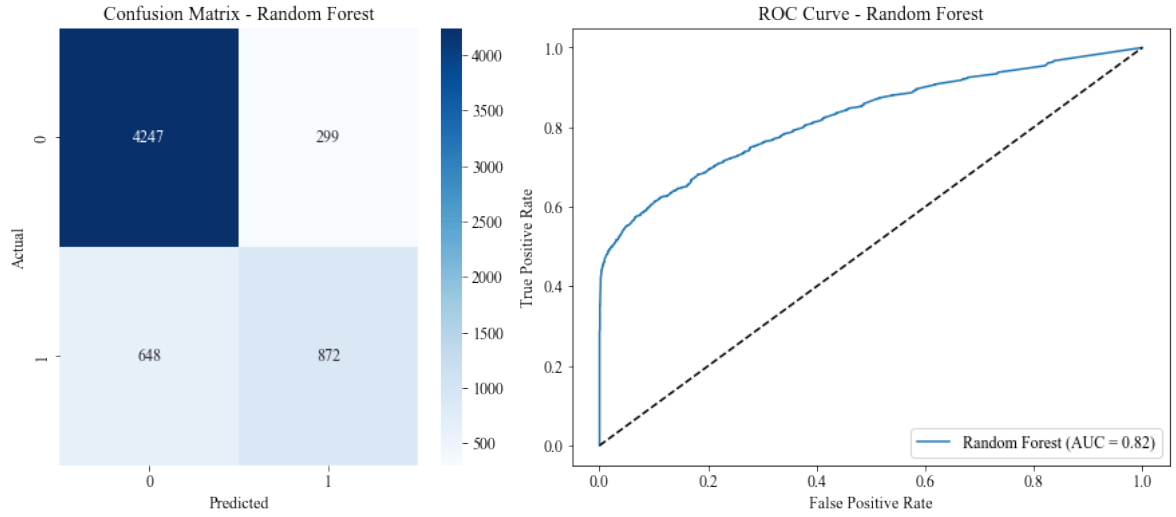

Рис. 3.21. Матриця невідповідностей та ROC крива моделі випадковий ліс

Джерело: створено автором

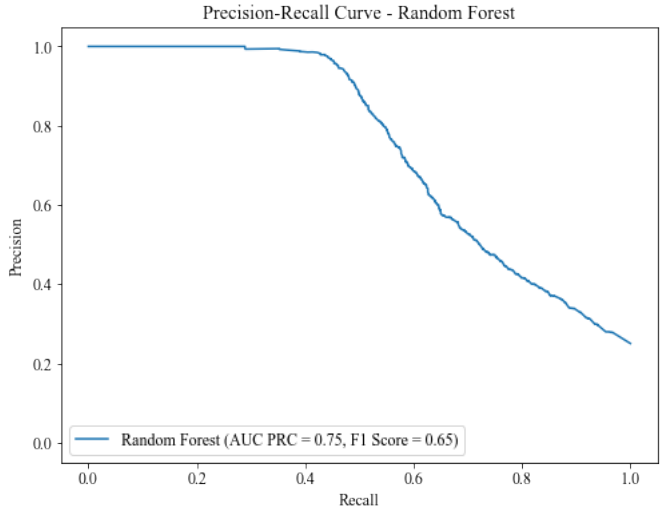

Рис. 3.22. Крива Precision-Recall моделі випадковий ліс

Джерело: створено автором

Для того щоб оптимізувати Random Forest переберемо гіперпараметри, які використовуються в моделі за допомогою RandomizedSearchCV, який обирає випадкові набори параметрів і шукає найкращі [47]. Він працює швидше ніж GridSearchCV, оскільки не перевіряє кожен набір, проте одночасно з цим є менш точним. Для підбору візьмемо такі варіанти гіперпараметрів: кількість дерев від 10 до 1000 з кроком 10, кількість ознак sqrt та log2, максимальна глибина від 10 до 200 з кроком 10, метрика оптимальності розбиття gini, entropy та наявність бутстрапінгу.

В результаті отримано такий оптимальний набір параметрів: 590 дерев, кількість ознак що рівна квадратному кореню від загальної кількості ознак, глибина дерев на рівні 10, метрика оптимальності розбиття entropy та використання бутстрапінгу. На рис. 3.23 бачимо, що ці показники дали результати точності моделі 0.87, площі під кривою ROC 0.86, що є досить високим показником. AUC (PRC), як видно з рис. 3.24 має невелике покращення до 0.78, проте F1-Score лишився на тому ж рівні 0.65.

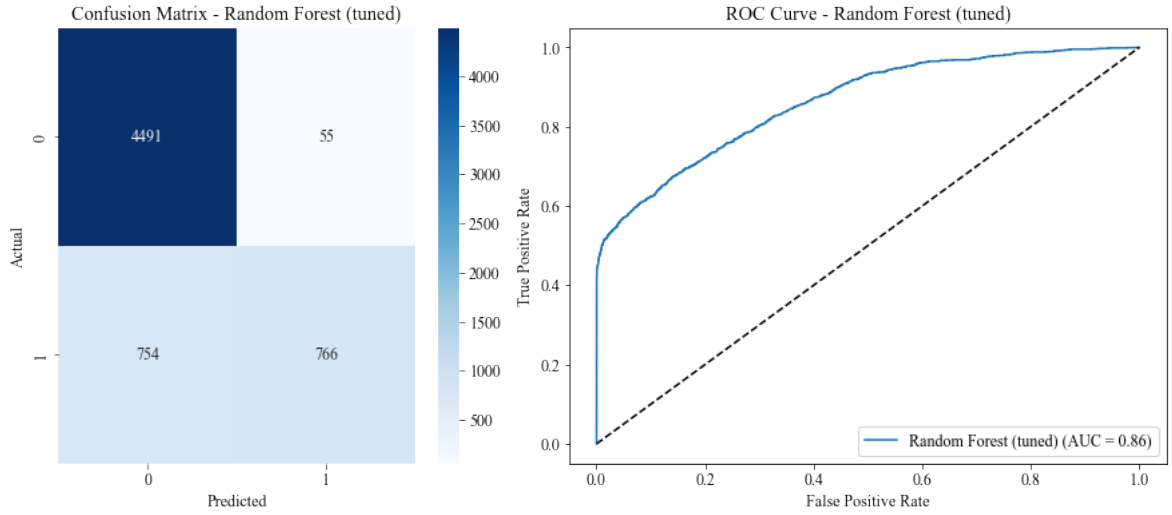

Рис. 3.23. Матриця невідповідностей та ROC крива оптимізованої моделі випадковий ліс

Джерело: створено автором

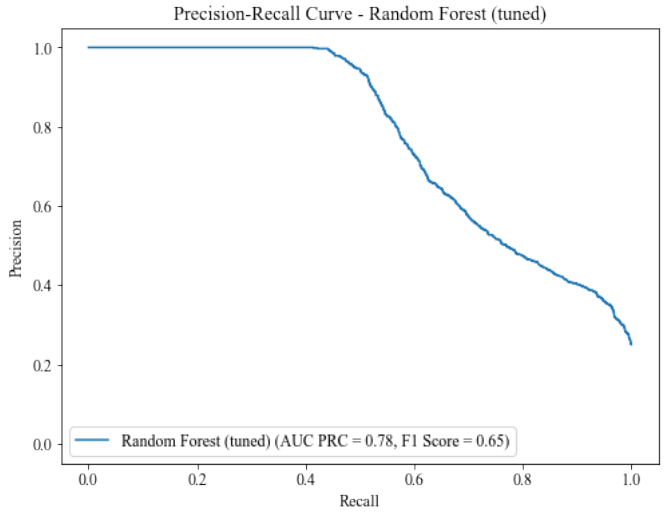

Рис. 3.24. Крива Precision-Recall оптимізованої моделі випадковий ліс

Джерело: створено автором

Support Vector Machine на рис. 3.25 показує добру точність на рівні 0.82 та здатність розділити класи AUC (ROC) 0.79, але має помірну здатність зберігати точність при виявленні позитивних класів (на рис. 3.26 бачимо, що AUC (PRC) = 0.67). F1-Score також вказує на помірний баланс між точністю та повнотою моделі і рівний 0.59. Ці показники є досить низькими в порівнянні з іншими моделями, тому ми не будемо використовувати SVM для класифікації китів та звичайних користувачів.

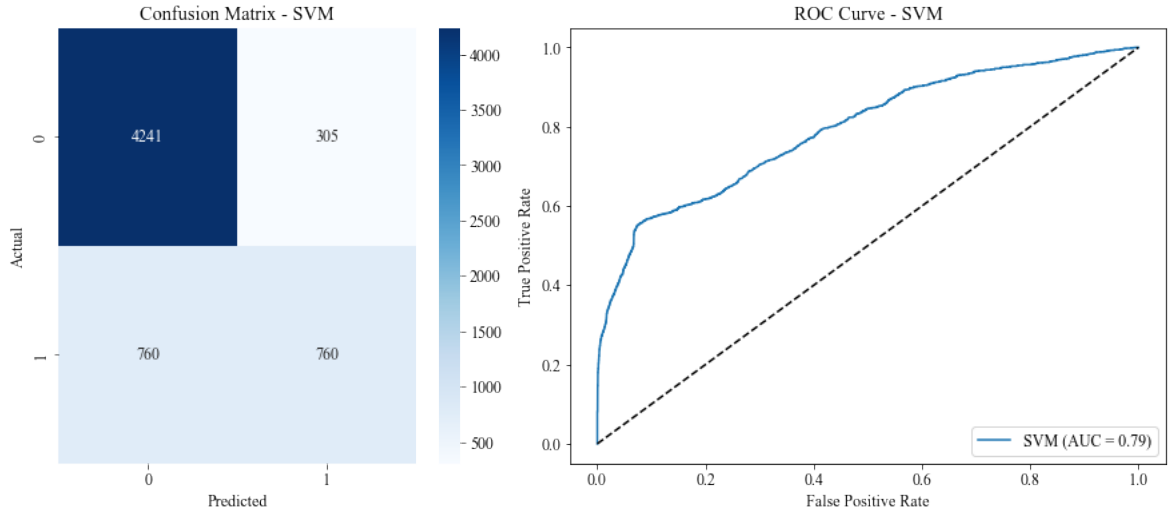

Рис. 3.25. Матриця невідповідностей та ROC крива моделі SVM

Джерело: створено автором

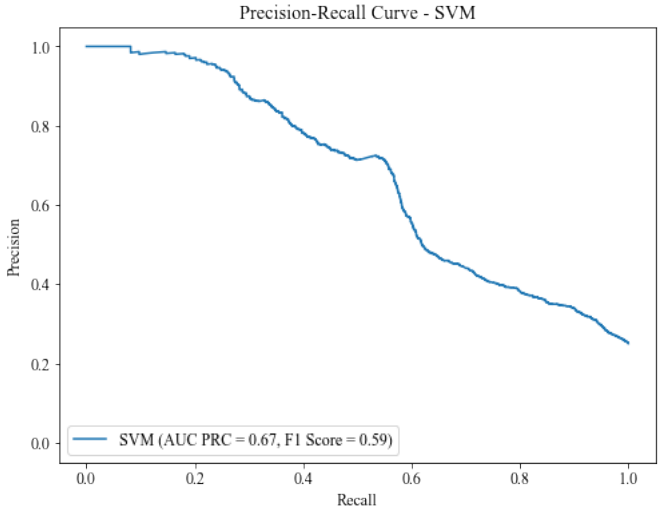

Рис. 3.26. Крива Precision-Recall моделі SVM

Джерело: створено автором

Перейдемо до моделі ***Нейронної мережі***. Загалом, виходячи з матриці невідповідностей на рис. 3.27 вона показує високу точність і правильно

класифікує 83% прикладів. Якщо говорити про її здатність розділити класи, то вона теж досить хороша і показує AUC (ROC), що рівний 0.84. Разом з цим вона має добру здатність зберігати точність при виявленні позитивних класів (AUC (PRC) = 0.73). F1-Score рівний 0.63 також вказує на помірний баланс між точністю та повнотою моделі.

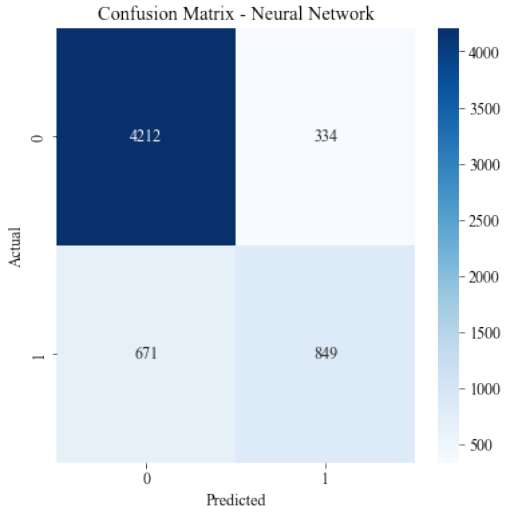

Рис. 3.27. Матриця невідповідностей нейронної мережі

Джерело: створено автором

На рис. 3.28 відображений графік змін значень функції втрати (loss) та точності (accuracy) протягом процесу тренування моделі. Він дає змогу візуально оцінити якість тренування моделі та її здатність навчатися з кожною ітерацією.

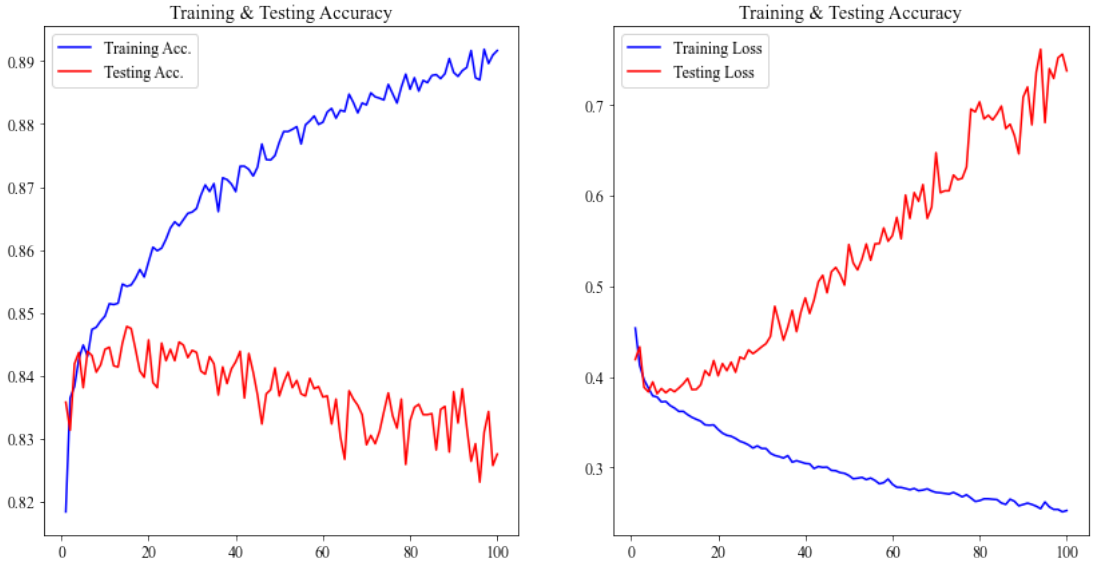

Рис. 3.28. Графік змін значень функції втрати та точності нейронної мережі

Джерело: створено автором

Графік history of loss показує, як змінюється значення функції втрати протягом тренування. Метою моделі є мінімізація цієї функції, тому бачимо, що значення loss зменшується з кожною ітерацією. Графік history of accuracy показує, як змінюється точність моделі. Бачимо що вона збільшується до рівня 0.89 на тренувальній вибірці та до рівня 0.83 на тестовій.

Незважаючи на досить гарні показники, вони є гіршими ніж наприклад у моделі випадкового лісу. Імовірніше за все нейронна мережа показала гірші результати через недостатньо велику кількість даних.

Наступна модель – ***CatBoost.*** Якщо говорити про загальну точність, то вона правильно класифікувала 86% прикладів, що є досить високим показником. При цьому на рис. 3.29 ми бачимо ROC криву, що має AUC на рівні 0.85. Це свідчить про хорошу здатність відокремлювати позитивні та негативні приклади.

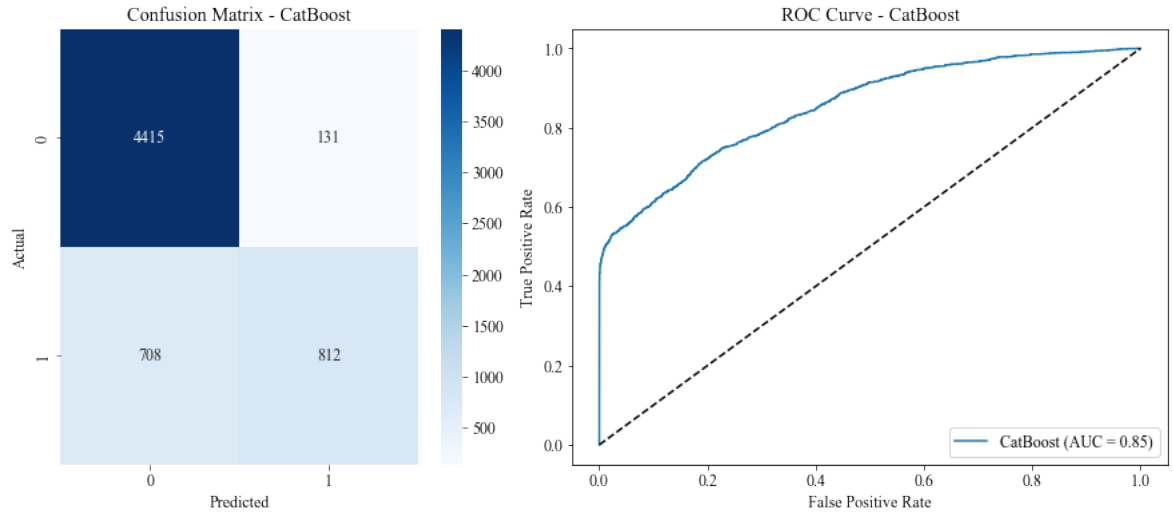

Рис. 3.29. Матриця невідповідностей та ROC крива моделі CatBoost

Джерело: створено автором

Перейдемо до Precision-Recall на рис. 3.30. Крива говорить про хорошу здатність моделі зберігати високу точність при виявленні позитивних прикладів, AUC (PRC) = 0.77. F1-Score, що рівний 0.66 також вказує на хороший баланс між точністю та повнотою моделі.

Спробуємо покращити точність моделі завдяки налаштуванням гіперпараметрів. Перший з них це швидкість навчання. Встановлення меншого значення може допомогти покращити збіжність моделі, але може зайняти більше часу для тренування.

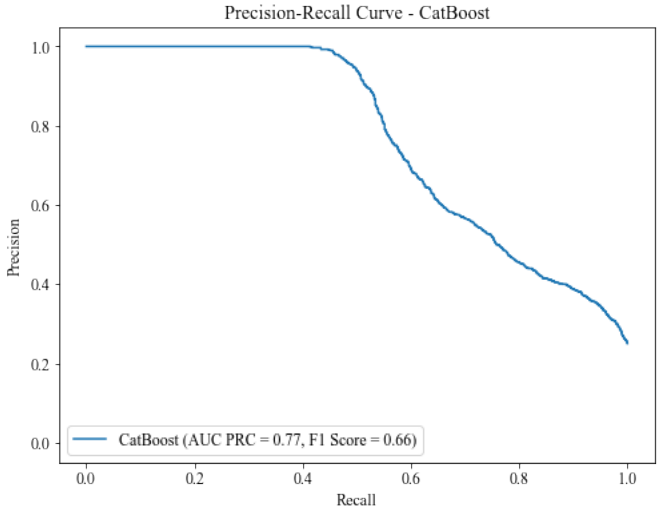

Рис. 3.30. Крива Precision-Recall моделі CatBoost

Джерело: створено автором

Для підбору використаємо значення 0.01, 0.1 та 0.5, що відповідають повільному, середньому та швидкому навчанню. Також переберемо глибину дерев в діапазоні від 4 до 9. Збільшення глибини дерева може дозволити моделі вчитися більш складним залежностям у даних, але при цьому може збільшитися ймовірність перенавчання.

В результаті отримали значення глибини дерев на рівні 6 та швидкості навчання 0.01, тобто повільному.

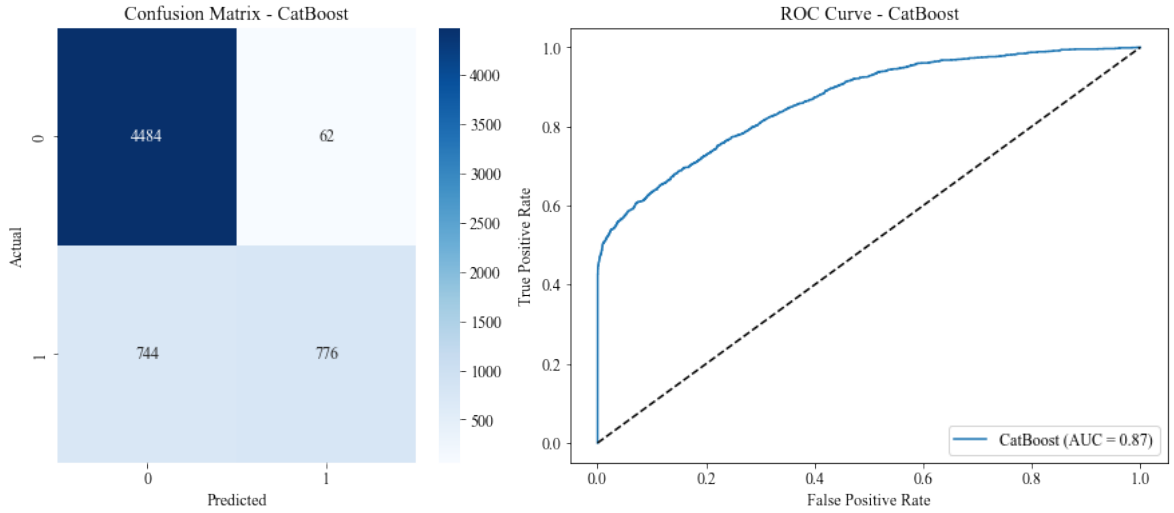

Рис. 3.31. Матриця невідповідностей та ROC крива оптимізованої моделі CatBoost

Джерело: створено автором

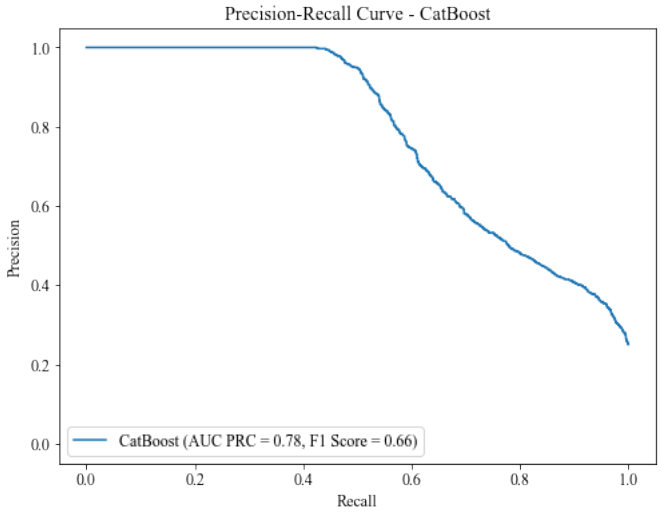

Рис. 3.32. Крива Precision-Recall оптимізованої моделі CatBoost

Джерело: створено автором

Застосування моделі з такими гіперпараметрами дає змогу правильно класифікувати 87% прикладів, що на 1% більше ніж до оптимізації. Також з рис. 3.31 бачимо збільшення показника AUC (ROC) до рівня 0.87. При цьому трішки підвищилась здатність точно класифікувати позитивні приклади, оскільки AUC (PRC) на рис. 3.32 рівний 0.78. F1-Score на тому ж рівні 0.66 говорить про гарний баланс між точністю і повнотою моделі.

На рис. 3.33 виведені топ-15 ознак, які мають найбільший вплив. Сюди увійшли виручка перших трьох покупок користувача, виручка за перші 3 дні на продукті, тривалість чатів за перші 7 днів, величина першої покупки, кількість спеціалістів, з якими спілкувався користувач протягом перших двох тижнів. Це свідчить про те, що користувач уже при перших діях своєю поведінкою показує що він є платоспроможним і готовий витрачати багато. Якщо говорити про демографічні показники, то вони знаходяться значно нижче по списку та мають менший вплив на результат моделі. Це говорить про те, що ці ознаки значно менше визначають імовірність стати китом. Проте вони все ж входять в топ 15, зокрема вік користувача, країна, статус відносин, джерело залучення.

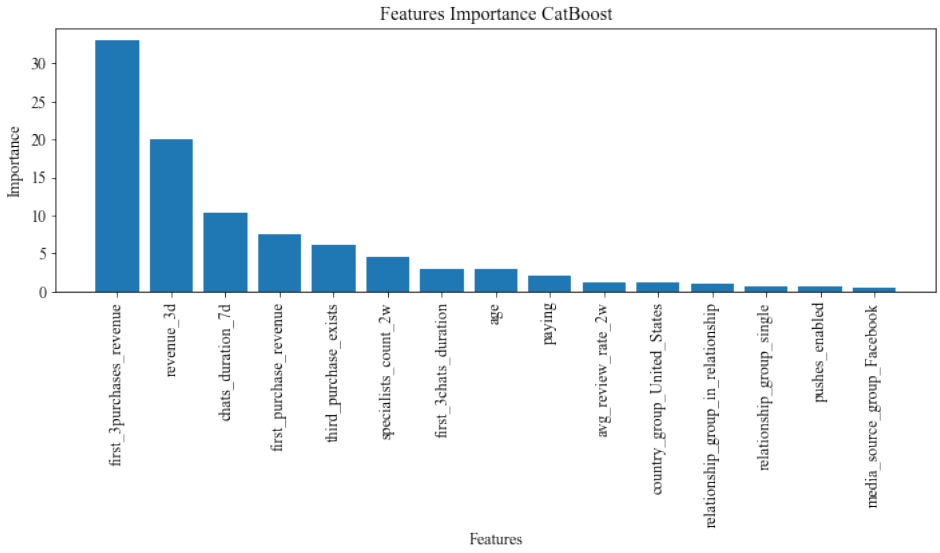

Рис. 3.33. Найбільш вагомі ознаки оптимізованої моделі CatBoost

Джерело: створено автором

Для вибору оптимальної моделі порівняємо показники точності для кожної з них в табл. 3.2.

Таблиця 3.2

Порівняння показників якості моделей

Модель	*Accuracy*	*AUC (ROC)*	*AUC (PRC)*	*F1 Score*
K найближчих сусідів	0.79	0.75	0.59	0.51
KNN (Оптимізована)	0.81	0.78	0.62	0.51
Логістична регресія	0.83	0.76	0.66	0.60
Логістична регресія (Оптимізована)	0.83	0.77	0.68	0.58
Випадковий ліс	0.84	0.82	0.75	0.65
Випадковий ліс (Оптимізована)	0.84	0.85	0.76	0.64
Support Vector Machine	0.82	0.79	0.67	0.59
Нейронна мережа	0.83	0.84	0.73	0.63
CatBoost	0.86	0.85	0.77	0.66
CatBoost (Оптимізована)	0.87	0.87	0.78	0.66

Джерело: створено автором

Бачимо що за всіма параметрами оцінок, найкращі показники показав алгоритм CatBoost, який і буде використовуватись для класифікації користувачів на китів та не китів.

Надані результати є важливим внеском у розуміння поведінки користувачів на ранніх етапах життя на продукті. Результати аналізу впливу ознак на модель надають цінну інформацію для команди маркетингу, що займається залученням користувачів, продуктовій команді з точки зору стратегій активації та утримання та службі підтримки користувачів. Давайте розглянемо ці результати детальніше.

Маркетинг. Результати показують, що виручка перших трьох покупок користувача, виручка за перші 3 дні на продукті та величина першої покупки мають найбільший вплив на ймовірність стати китом. Це означає, що з точки зору маркетингової закупки варто звернути увагу на конверсію саме в користувачів, які потенційно можуть стати китами. Це допоможе краще оцінити ефективність маркетингових кампаній, ще на ранніх етапах після залучення. Важливість величини першої покупки говорить про те, що потрібно передавати це значення разом з активаційними івентами в джерело залучення (наприклад Facebook) для налаштування кращої оптимізації на більш цільових користувачів.

Продукт. Результати показують, що тривалість чатів за перші 7 днів та кількість спеціалістів, з якими спілкувався користувач, також мають значний вплив. Ці знання можна використати для формування активаційних стратегій. Наприклад давати користувачу провести більше часу в чатах з різними спеціалістами, щоб він зміг отримати розуміння якості послуг та отримати цінність. Також вплив кількості спеціалістів говорить про те, що кити часто спілкуються не з кимось одним, а з різними людьми. Тому після спілкування з одним спеціалістом можна пропонувати відразу інших. Важливо також надавати достатньо інформації про продукт протягом перших днів використання. Забезпечення позитивного досвіду взаємодії з продуктом та ефективного вирішення питань користувачів може сприяти їх задоволенню та подальшій активності. Також продуктовій команді важливо зосередитись на привабливих

акціях та пропозиціях для нових користувачів, щоб залучити їх до перших покупок. Слід розглянути можливості стимулювання великих перших покупок, так як це може бути ознакою високої платоспроможності та готовності витрачати багато.

Служба підтримки. Перш за все, наявність сегментації за тим чи є користувач потенційним китом надасть працівникам служби підтримки необхідне знання що допоможе пріоритетно, а отже і більш швидко та якісно оброблювати запити, вибудовувати хороші відносини з користувачами та розвивати довіру до продукту та команди, що над ним працює. Результати показують, що демографічні ознаки, такі як вік користувача, країна, статус відносин, хоч і мають менший вплив на ймовірність стати китом, все ж можуть бути важливими для підходу служби підтримки користувачів. Наприклад, знання країни та мови користувача може допомогти забезпечити більш персоналізовану підтримку, а урахування статусу відносин може допомогти виявити та вирішити можливі проблеми або незадоволення.

В цілому, розуміння впливу окремих ознак на модель може допомогти в розробці стратегій та плануванні дій в різних аспектах бізнесу. Це дозволяє зосередитися на ключових факторах, що впливають на важливі метрики успіху та підвищують ефективність маркетингових та продуктових дій.

Комбінування цих результатів з іншими дослідженнями та даними може сприяти покращенню ефективності бізнес-процесів та досягненню більшого задоволення клієнтів.

ВИСНОВКИ

Визначення китів на ранніх етапах використання мобільного додатку є важливою складовою стратегії бізнесу та розвитку продукту з трансакційною моделлю монетизації. Це процес ідентифікації найцінніших, високодохідних та лояльних клієнтів, які стануть джерелом значної частки прибутку для компанії. Правильне виявлення китів та сфокусована робота над такими користувачами на самому початку їхнього досвіду користування продуктом є запорукою успішності бізнесу та рентабельність проєкту загалом.

У *першому розділі* дослідження було розглянуто дві основні моделі монетизації – підписочна, що надає стабільний, легко прогнозований дохід та трансакційна, що не має обмежень щодо кількості виручки з користувача і є менш стабільною. Особливістю трансакційної моделі є наявність китів, виявлення та утримання яких має велике значення для бізнесу. Особливо гостро дане питання постає для ігрової ніші та роздрібної торгівлі класу люкс, де середня виручка з верхніх 20% клієнтів більш ніж в 14 разів більша за аналогічний показники в нижніх 80%. Ідентифікація китів на ранніх етапах користування продуктом має значення для різних команд, таких як цифровий маркетинг, продуктовий менеджмент, продуктовий маркетинг та служба підтримки.

У *другому розділі* роботи описано етапи використання машинного навчання для вирішення задачі класифікації. Вони включають в себе підготовку даних, розбиття їх на навчальну та тестову вибірки, вибір моделей машинного навчання, їх навчання, а також оцінку їх якості. Для даного дослідження обрано такі моделі як k найближчих сусідів, логістична регресія, випадковий ліс, Support Vector Machine, нейронна мережа, CatBoost. Кожна з них має свої особливості, переваги та недоліки, можливості оптимізації шляхом підбору оптимальних гіперпараметрів.

Для оцінки якості моделей були використані показники, такі як матриця невідповідностей, точність, ROC крива та площа під нею, крива Precision-Recall

та AUC для неї, F1-Score з врахуванням їх пристосованості до до розбалансованих наборів даних.

У *третьому розділі* наведено вибір метрики, що влучно описує китовість користувача з математичної точки зору. Нею стала виручка за перші 3 місяці використання додатку. Вона є робастою, порівнюваною для клієнтів з різних когорт та добре описує фінансову цінність для бізнесу. Пороговим значенням для розділення користувачів на китів та не китів було обрано 150 доларів шляхом наближення до правила Парето, коли 80% виручки генерується 20% клієнтів.

Було проведено огляд структури даних та їх підготовка до моделювання. Вона включає в себе обробку відсутніх значень, викидів, видалення певних колонок, нормалізація числових даних, кодування категоріальних стовпців з допомогою One-Hot Encoder-a, розділення на тренувальні і тестову вибірки.

Наступним кроком було навчання моделей з використанням python бібліотек sklearn та keras. Також більшість моделей було оптимізовано шляхом підбору оптимальних гіперпараметрів методами GridSearchCV або RandomizedSearchCV. Після порівняння різних моделей машинного навчання оптимальною виявилась CatBoost, яка правильно класифікувала 87% об’єктів. AUC (PRC) на рівні 0.78 та F1-Score 0.66 говорять про хорошу здатність класифікувати позитивні класи та достатній баланс між точністю і повнотою. До найбільш вагомих однак моделі увійшли виручка перших трьох покупок, виручка за перші 3 дні, тривалість чатів за перші 7 днів, величина першої покупки, кількість спеціалістів протягом перших двох тижнів.

Отже, виявлення китів на ранніх етапах використання мобільного додатку є важливим для отримання високих фінансових показників компанії. Дана наукова робота застосовує методи машинного навчання, які широко поширені та детально розглянуті серед українських авторів, для моделювання показника, який раніше не отримував достатньої уваги у вітчизняному просторі. Вона може послужити основою для подальших досліджень та розвитку стратегій в сфері маркетингу, продуктового управління та підтримки клієнтів українських IT компаній.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Subscription business models in 2023: Definition, benefits, tips & metrics to track. Paddle. URL: https://www.paddle.com/resources/subscription-business-model.

2. Transaction-Based Revenue Model: Explained. Causal: Modern Business Planning. URL: https://www.causal.app/define/transaction-based-revenue-model.

3. Subscription Model Vs Transaction Model. Oboloo. URL: https://oboloo.com/glossary/subscription-model-vs-transaction-model/.

4. Predictive Modeling Using Transactional Data. Capgemini. URL: https://www.capgemini.com/wp-content/uploads/2017/07/Predictive_Modeling_Using_Transactional_Data.pdf.

5. Business Intelligence: Predictive Revenue Modeling. Arcalea | Award-winning Marketing Consultancy. URL: https://www.arcalea.com/blog/business-intelligence-predictive-revenue-modeling.

6. Revenue Model Types in Software Business: Examples and Model Choice. AltexSoft. URL: https://www.altexsoft.com/blog/revenue-model-types/#:~:text=A%20transaction-based%20model%20is,the%20production%20costs%20and%20margin.

7. Dieffenbacher S. F. Revenue Streams in Business Model Canvas. Digital Leadership. URL: https://digitalleadership.com/blog/revenue-streams/.

8. Зоріна К., Романко О. Building segment based revenue prediction for CLV model. Український Католицький Університет. URL: https://er.ucu.edu.ua/bitstream/handle/1/1339/Zorina_%20Building%20segment%20based.pdf?sequence=1&isAllowed=y.

9. Visualizing customer profitability with the whale curve. Baker Tilly. URL: https://www.bakertilly.com/insights/visualizing-customer-profitability-with-the-whale-curve.

10. Liverence B. Whale Watching: Many companies earn a huge portion of sales from a few customers - Bloomberg Second Measure. Bloomberg Second Measure. URL: https://secondmeasure.com/datapoints/whales/.

11. Grguric M. What Are Mobile Game Whales & How To Find Them. Udonis Mobile Marketing Agency. URL: https://www.blog.udonis.co/mobile-marketing/mobile-games/mobile-games-whales#:~:text=A%20mobile%20game%20whale%20is,purchases,%20whales%20spend%20a%20lot.

12. What is a Whale in Gambling? - Sports Betting Terms. realonlinegambling. URL: https://www.realonlinegambling.com/terms/whale/.

13. Tiffany (TIF) Income Statement. Investing.com. URL: https://www.investing.com/equities/tiffany---co-income-statement.

14. A Beginner's Guide to Performance Marketing: Everything You Need to Know. Bigcommerce. URL: https://www.bigcommerce.com/articles/ecommerce/performance-marketing/.

15. Avigo E. Activation playbook. June - Product analytics for B2B SaaS. URL: https://www.june.so/blog/activation-playbook.

16. Katie. Email Marketing for your Ecommerce Whales. Keetrax. URL: https://keetrax.com/email-marketing-for-your-ecommerce-whales/.

17. How Whales Can Make You More Successful. The Mighty Marketer. URL: https://themightymarketer.com/whales/.

18. Інформаційні системи та технології в управлінні. Методичні вказівки, теоретичні відомості і завдання до лабораторних робіт для студентів та магістрів денної форми навчання спеціальності 7.803060101 Менеджмент організацій і адміністрування. Частина 3. Класифікація в бізнес-аналітиці. / Укл.: Біла Н.І. – Запоріжжя: ЗНТУ, 2014. – с. 50.

19. Vertica. Powering the World's Data-Driven Leaders. Vertica. URL: https://www.vertica.com/about/.

20. H. Mannila, Data mining: machine learning, statistics, and databases, Proceedings of 8th International Conference on Scientific and Statistical Data Base Management, Stockholm, Sweden, 1996, pp. 2-9, doi: 10.1109/SSDM.1996.505910.

21. A.D. Joseph, B. Nelson, B.I.P. Rubinstein, J.D. Tygar: Adversarial Machine Learning – Cambridge University Press, 2019.

22. E. Alpaydin: Introduction to Machine Learning, 3rd ed. – The MIT Press, 2014.

23. R. Fernandes de Mello, M.A. Ponti: Machine Learning: A Practical Approach on the Statistical Learning Theory – Springer, 2018.

24. S. Zhang, X. Li, M. Zong, X. Zhu and R. Wang, "Efficient kNN Classification With Different Numbers of Nearest Neighbors," in IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 5, pp. 1774-1785, May 2018, doi: 10.1109/TNNLS.2017.2673241

25. W. Cherif, Optimization of K-NN algorithm by clustering and reliability coefficients: application to breast-cancer diagnosis, Procedia Computer Science, vol. 127, pp. 293-299, 2018, ISSN 1877-0509. URL https://www.sciencedirect.com/science/article/pii/S1877050918301376.

26. Ставицький А. В. Метричні методи класифікації та регресії. Алгоритм к найближчих сусідів. URL: http://www.andriystav.cc.ua/Downloads/MITER/Lecture_06.pdf.

27. LaValley M. P. Logistic Regression. Lippincott Williams & Wilkins. 2008. No. 18. P. 2395–2399. URL: https://www.ahajournals.org/doi/10.1161/CIRCULATIONAHA.106.682658.

28. Zisserman A. Logistic Regression. Information Engineering. URL: https://www.robots.ox.ac.uk/~az/lectures/ml/2011/lect4.pdf.

29. Hastie T. The elements of statistical learning: Data mining, inference, and prediction. New York : Springer, 2001. 533 p. URL: https://hastie.su.domains/Papers/ESLII.pdf.

30. What is Random Forest?. IBM. URL: https://www.ibm.com/topics/random-forest.

31. Ramadhan, Muhammad & Sitanggang, Imas & NASUTION, Fahrendi & GHIFARI, Abdullah. (2017). Parameter Tuning in Random Forest Based on Grid Search Method for Gender Classification Based on Voice Frequency. DEStech Transactions on Computer Science and Engineering. 10.12783/dtcse/cece2017/14611.

32. Лифшиц Ю. Метод опорных векторов Лекция № 7 курса «Алгоритмы для Интернета». URL: https://logic.pdmi.ras.ru/~yura/internet/07ianote.pdf.

33. Guyon, I., Weston, J., Barnhill, S. et al. Gene Selection for Cancer Classification using Support Vector Machines. Machine Learning 46, 389–422 (2002). https://doi.org/10.1023/A:1012487302797. URL: https://link.springer.com/article/10.1023/A:1012487302797.

34. Kowalczyk A. Support Vector Machines Succinctly. Morrisville : Syncfusion, 2017. 114 p., 2501 Aerial Center Parkway, Suite 200, NC 27560. URL: https://s3.amazonaws.com/ebooks.syncfusion.com/downloads/support_vector_machines_succinctly/support_vector_machines_succinctly.pdf?AWSAccessKeyId=AKIAWH6GYCX3YH6S4NTZ&Expires=1686487504&Signature=VyOAkCuu4Wk7OS%2Bw7oanaSNtnRU%3D.

35. Kriegeskorte N., Golan T. Neural network models and deep learning. Current Biology. 2019. Vol. 29, no. 7. P. R231–R236. URL: https://doi.org/10.1016/j.cub.2019.02.034.

36. Hinton G. Lecture 13: Learning without a teacher: Autoencoders and Principal Components Analysis. Department of Computer Science, University of Toronto. URL: https://www.cs.toronto.edu/~hinton/csc321/notes/lec13.pdf.

37. Kononenko V., Krasnoshlyk N. Using boosting methods for machine learning problems. Cherkasy univercity bulletin: applied mathematics. informatics. 2022. No. 1. P. 58–68. URL: https://doi.org/10.31651/2076-5886-2021-1-58-68.

38. Model Evaluation Metrics in Machine Learning, KDnuggets. URL: https://www.kdnuggets.com/2020/05/model-evaluation-metrics-machine-learning.html.

39. Gad A. F. Accuracy, Precision, and Recall in Deep Learning. Paperspace Blog. URL: https://blog.paperspace.com/deep-learning-metrics-precision-recall-accuracy/#:~:text=Accuracy%20is%20a%20metric%20that,the%20total%20number%20of%20predictions.

40. Sokolova, M., Japkowicz, N., Szpakowicz, S. (2006). Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation. In: Sattar, A., Kang, Bh. (eds) AI 2006: Advances in Artificial Intelligence. AI 2006. Lecture Notes in Computer Science(), vol 4304. Springer, Berlin, Heidelberg. URL: https://doi.org/10.1007/11941439_114.

41. Продукти. Obrio. URL: https://obrio.co/products.

42. Що таке когортний аналіз, чому він важливий для маркетингу і як його провести. Міжнародна Маркетингова Група. URL: https://www.marketing-ua.com/article/shho-take-kogortnij-analiz-chomu-vin-vazhlivij-dlya-marketingu-i-yak-jogo-provesti/.

43. Introduction to Balanced and Imbalanced Datasets in Machine Learning. Encord | Data-Centric AI Development Software for Computer Vision. URL: https://encord.com/blog/an-introduction-to-balanced-and-imbalanced-datasets-in-machine-learning/.

44. sklearn.preprocessing.OneHotEncoder. scikit-learn. URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html.

45. Комбінаторні конфігурації та їхні застосування: Матеріали XXII Міжнародного науково-практичного семінару імені А.Я. Петренюка (Запоріжжя - Кропивницький, 15-16 травня 2020 року) / за ред. Г.П. Донця – Кропивницький: ПП «Ексклюзив-Систем», 2020. – 187с. URL: https://dspace.sfa.org.ua/bitstream/123456789/488/1/Materiali_22_seminaru.pdf#page=187.

46. sklearn.model_selection.GridSearchCV. scikit-learn. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

47. sklearn.model_selection.RandomizedSearchCV. scikit-learn. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html.

ДОДАТКИ

Додаток А

SQL запит для отримання та первинної обробки даних з БД Vertica

```
with aggregarion as(
select t.user_id,
       install_time,
       media_source,
       country,
       t.platform,
       u.gender,
       u.relationship,
       u.send_push,
       u.local,
       u.paying,
       usd_gross,
       datediff('year', birth_day, date(sysdate)) as age,
       datediff('day', install_time, date(sysdate)) as age_on_product,
       datediff('day', install_time, t.event_time) as event_day,
       event_name,
       t.event_time,
       event_row_number,
       finished_at,
       transaction_type,
       session_time,
       type_session,
       specialist_id,
       session_id,
       transaction_id,
       rating,
       product_id,
       min(case when event_name = 'purchase' and
                transaction_type = 'income' and
                datediff('day', install_time, t.event_time) < 13 and
                event_row_number = 1 then datediff('day', install_time,
t.event_time) end)
           over(partition by t.user_id, install_time, media_source, country,
t.platform) as first_purchase_day,
       min(case when event_name = 'purchase' and
                transaction_type = 'income' and
                datediff('day', install_time, t.event_time) < 13 and
                event_row_number = 3 then datediff('day', install_time,
t.event_time) end)
           over(partition by t.user_id, install_time, media_source, country,
t.platform) as third_purchase_day,
       min(case when event_name = 'chat' and
                datediff('day', install_time, t.event_time) < 13 and
                event_row_number = 1 then datediff('day', install_time,
t.event_time) end)
           over(partition by t.user_id, install_time, media_source, country,
t.platform) as first_chat_day,
       min(case when event_name = 'chat' and
                datediff('day', install_time, t.event_time) < 13 and
                event_row_number = 1 then type_session end)
           over(partition by t.user_id, install_time, media_source, country,
t.platform) as first_chat_type
from chats_events t
left join users_data u on u.user_id = t.user_id
where install_time is not null and
      t.user_id is not null and
      t.event_time >= install_time)
```

```
select user_id
       media_source,
       platform,
       send_push as pushes_enabled,
       local as language,
       paying,
       country,
       age,
       gender,
       relationship,
       sum(case when event_day <= 90 and age_on_product>=90 then usd_gross else
0 end) as revenue_3m,
       first_chat_day,
       first_chat_type,
       sum(case when event_name = 'chat' and
                     event_day < 13 and
                     event_row_number = 1 then session_time else 0 end) as
first_chat_duration,
       sum(case when event_name = 'chat' and
                     event_day < 13 and
                     event_row_number between 1 and 3 then session_time else 0
end) as first_3chats_duration,
       sum(case when event_name = 'chat' and
                     event_day < 6 then session_time else 0 end) as
chats_duration_7d,
       sum(case when event_name = 'chat' and
                     event_day < 13 then session_time else 0 end) as
chats_duration_2w,
       count(distinct case when event_day < 13 then session_id end) as
chats_number_2w,
       count(distinct case when event_day < 13 then specialist_id end) as
specialists_count_2w,
       avg(case when event_day < 13 then rating end) as avg_review_rate_2w,
       first_purchase_day,
       sum(case when event_name = 'purchase' and
                 transaction_type = 'income' and
                 event_day < 13 and
                 event_row_number = 1 then usd_gross else 0 end) as
first_purchase_revenue,
       third_purchase_day,
       sum(case when event_name = 'purchase' and
                 transaction_type = 'income' and
                 event_day < 13 and
                 event_row_number between 1 and 3 then usd_gross else 0 end) as
first_3purchases_revenue,
       sum(case when event_name = 'purchase' and
                 transaction_type = 'income' and
                 event_day = 0 then usd_gross else 0 end) as revenue_1d,
       sum(case when event_name = 'purchase' and
                 transaction_type = 'income' and
                 event_day < 2 then usd_gross else 0 end) as revenue_3d,
       sum(case when event_name = 'purchase' and
                 transaction_type = 'income' and
                 event_day < 6 then usd_gross else 0 end) as revenue_7d,
       sum(case when event_name = 'purchase' and
                     transaction_type = 'income' and
                     event_day < 13 then usd_gross else 0 end) as revenue_2w,
       count(distinct case when transaction_type = 'income' and
                                event_day < 13 then transaction_id end) as
purchases_number_2w,
       max(case when product_id = 'chat_balance_paypal' and
```

Продовження додатку А

```
                    event_day < 13 then 1 else 0 end) as paypal_purchases_2w,
       max(case when product_id = 'chat_balance_googlepay' and
                    event_day < 13 then 1 else 0 end) as
googlepay_purchases_2w,
       max(case when product_id = 'chat_balance_card' and
                    event_day < 13 then 1 else 0 end) as card_purchases_2w,
       max(case when product_id = 'chat_balance_recurrent' and
                    event_day < 13 then 1 else 0 end) as
recurrent_purchases_2w,
       max(case when product_id = 'chat_balance_applepay' and
                    event_day < 13 then 1 else 0 end) as applepay_purchases_2w
from aggregarion
where age_on_product>=90
group by 1,2,3,4,5,6,7,8,9,10,12,13,21,23
having sum(case when event_day <= 90 and age_on_product>=90 then usd_gross end)
> 0
order by 11 desc
```

Додаток Б

Python код для обробки даних, побудови та оцінки моделей

```
# Імпорт необхідних бібліотек
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
import matplotlib
import plotly.express as px
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split, GridSearchCV,
RandomizedSearchCV
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report, roc_curve, roc_auc_score,
from sklearn.metrics import precision_recall_curve, auc, f1_score,
plot_confusion_matrix, precision_score, recall_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from keras.models import Sequential, load_model
from keras import layers
from keras.layers.core import Dropout
from keras.callbacks import ModelCheckpoint
from catboost import CatBoostClassifier
from scipy.stats import randint as sp_randint

# Огляд структури даних
df = pd.read_csv('dataset.csv')
df.head(5)
df.info()
df.describe()

# Обробка даних
df.loc[df['age'] < 18, 'age'] = df['age'].median()
df.loc[df['age'] > 90, 'age'] = df['age'].median()
df.loc[df['first_purchase_revenue'] < 0, 'first_purchase_revenue'] = 0

df['revenue_3m_binary'] = df['revenue_3m'].apply(lambda x: 1 if x >= 150 else 0)
matplotlib.rc('font', **{'family': 'Times New Roman', 'size': 12})
plt.figure(figsize=(7, 7))
ax = sns.countplot(x=df['revenue_3m_binary'], palette="Blues", linewidth=1)
plt.show()

for i in df.columns:
    print(f"Unique {i}'s count: {df[i].nunique()}")
    print(f"{df[i].unique()}\n")
df.isna().sum()

df = df.drop(columns=['user_id', 'first_chat_day', 'first_purchase_day',
'revenue_3m'])
df['media_source'].fillna('Unknown', inplace=True)

media_source_groups = {
    'Social_facebook': 'Facebook',
    'Facebook Ads': 'Facebook',
    'Facebook page': 'Facebook',
    'googleadwords_int': 'Google',
    'tiktok': 'TikTok',
    'TikTok_ads': 'TikTok',
    'TikTok': 'TikTok',
    'organic': 'Organic',
```

Продовження додатку Б

```
    'Apple Search Ads': 'Apple Search Ads',
    'Unknown': 'Unknown'
}
df['media_source_group'] =
df['media_source'].map(media_source_groups).fillna('Other')
df = df.drop(columns='media_source')
print(df['media_source_group'].value_counts())

df['country'].fillna(df['country'].mode()[0], inplace=True)
country_groups = {
    'US': 'United_States',
    'CA': 'Canada',
    'AU': 'Australia',
    'UK': 'United_Kingdom'}
df['country_group'] = df['country'].map(country_groups).fillna('Other')
df = df.drop(columns='country')
print(df['country_group'].value_counts())

df['pushes_enabled'].fillna(df['pushes_enabled'].median(), inplace=True)
df['language'].fillna(df['language'].mode()[0], inplace=True)
df['gender'].fillna(df['gender'].mode()[0], inplace=True)

relationship_groups = {
    'single': 'single',
    'in_relationship': 'in_relationship',
    'complicated': 'complicated',
    'married': 'in_relationship',
    'divorced': 'single',
    'engaged': 'in_relationship',
    'separated': 'single',
    'cohabitants': 'in_relationship',
    'widowed': 'single',
}
df['relationship_group'] = df['relationship'].map(relationship_groups)
df = df.drop(columns='relationship')
df['relationship_group'].fillna(df['relationship_group'].mode()[0],
inplace=True)
print(df['relationship_group'].value_counts())

df['third_purchase_exists'] = df['third_purchase_day'].notnull().astype(int)
df = df.drop(columns='third_purchase_day')
df['avg_review_rate_2w'].fillna(0.00, inplace=True)
df['platform'] = df['platform'].map({'ios': 1, 'android': 0})
df['gender'] = df['gender'].map({'female': 1, 'male': 0})
df['gender'].fillna(df['gender'].mode()[0], inplace=True)
df['first_chat_type'].fillna(df['first_chat_type'].mode()[0], inplace=True)
df['first_chat_type'] = df['first_chat_type'].map({'online': 1, 'offline': 0,
'messenger': 0})
df['pushes_enabled'] = df['pushes_enabled'].map({1.0: 1, 0.0: 0})

df = df.drop(columns=['first_chat_duration', 'chats_duration_2w', 'revenue_7d',
                      'revenue_1d', 'chats_number_2w', 'purchases_number_2w',
'revenue_2w'])

# Побудова графіків кореляцій
plt.figure(figsize=(16, 10))
df.corr()['revenue_3m_binary'].sort_values(ascending=False).plot(kind='bar',
figsize=(20, 5))

fig = px.imshow(df.corr().round(2), text_auto=True, aspect="auto", width=800,
height=800)
```

Продовження додатку Б

```
fig.update_layout(font=dict(size=8))
fig.show()

# One-Hot-Encoding
features_ohe = ['media_source_group', 'country_group', 'relationship_group',
'language']
df = pd.get_dummies(df, columns=features_ohe)
df = df.drop(columns=['media_source_group_Other', 'country_group_Other',
                      'relationship_group_complicated', 'language_pt'])

# Min-Max-Scaling
features_mms = ['age', 'first_3chats_duration', 'chats_duration_7d',
'specialists_count_2w',
                'avg_review_rate_2w', 'first_purchase_revenue',
'first_3purchases_revenue',
                'revenue_3d']
df_features_mms = pd.DataFrame(df, columns=features_mms)
df_remaining_features = df.drop(columns=features_mms)
mms = MinMaxScaler()
rescaled_features = mms.fit_transform(df_features_mms)
df_rescaled_features = pd.DataFrame(rescaled_features,
                                    columns=features_mms,
index=df_remaining_features.index)
df = pd.concat([df_remaining_features, df_rescaled_features], axis=1)

# Розділення на тренувальну та тестову вибірки
X1 = df.drop('revenue_3m_binary', axis=1)
Y1 = df['revenue_3m_binary']
X = X1.values
y = Y1.values
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)


# Створення функцій для оцінки якості моделей
def plot_top_features(X_df, classifier, classifier_name):
    feature_weights = pd.Series(np.abs(classifier.coef_[0]), index=X_df.columns)
    top_features = feature_weights.sort_values(ascending=False)[:15]

    plt.figure(figsize=(10, 6))
    top_features.plot(kind='bar')
    plt.title(f'Top 15 Features - {classifier_name}')
    plt.xlabel('Features')
    plt.ylabel('Absolute Coefficient')

    for i, feature in enumerate(top_features.index):
        plt.text(i, top_features[i],
f'{classifier.coef_[0][X_df.columns.get_loc(feature)]:.2f}', ha='center',
                 va='bottom', color='black')
    plt.show()


def confusion_matrix_and_accuracy(X_train, y_train, X_test, y_test, classifier,
y_pred, classifier_name):
    classifier.fit(X_train, y_train)
    y_pred = classifier.predict(X_test)
    cm = confusion_matrix(y_test, y_pred)
    accuracy = accuracy_score(y_test, y_pred)

    plt.figure(figsize=(6, 6))
    ax = plt.subplot()
```

```
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", ax=ax)
    ax.set_xlabel('Predicted')
    ax.set_ylabel('Actual')
    ax.set_title(f'Confusion Matrix - {classifier_name}')
    plt.show()
    print(f'Accuracy: {accuracy:.2f}')


def plot_roc_curve_and_auc(X_test, y_test, y_pred_probabilities,
classifier_name):
    if y_pred_probabilities.ndim == 2:
        y_pred_prob = y_pred_probabilities[:, 1]
    else:
        y_pred_prob = y_pred_probabilities
    fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)
    roc_auc = auc(fpr, tpr)

    plt.figure(figsize=(8, 6))
    plt.plot(fpr, tpr, label=f'{classifier_name} (AUC = {roc_auc:.2f})')
    plt.plot([0, 1], [0, 1], 'k--')
    plt.xlim([-0.05, 1.05])
    plt.ylim([-0.05, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title(f'ROC Curve - {classifier_name}')
    legend = plt.legend(loc='lower right')
    legend.get_texts()[0].set_color('black')
    plt.show()


def plot_prc_curve_and_auc(X_test, y_test, y_pred, y_pred_probabilities,
classifier_name):
    if y_pred_probabilities.ndim == 2:
        y_pred_prob = y_pred_probabilities[:, 1]
    else:
        y_pred_prob = y_pred_probabilities
    precision, recall, thresholds = precision_recall_curve(y_test, y_pred_prob)
    prc_auc = auc(recall, precision)
    f1 = f1_score(y_test, y_pred)

    plt.figure(figsize=(8, 6))
    plt.plot(recall, precision, label=f'{classifier_name} (AUC PRC =
{prc_auc:.2f}, F1 Score = {f1:.2f})')
    plt.xlim([-0.05, 1.05])
    plt.ylim([-0.05, 1.05])
    plt.xlabel('Recall')
    plt.ylabel('Precision')
    plt.title(f'Precision-Recall Curve - {classifier_name}')
    legend = plt.legend(loc='lower left')
    legend.get_texts()[0].set_color('black')
    plt.show()
    print(f'F1 Score: {f1:.2f}')
    print(f'AUC Score (PRC): {prc_auc:.2f}')


def plot_history_acc_loss(fit_keras):
    acc = fit_keras.history['accuracy']
    val_acc = fit_keras.history['val_accuracy']
    loss = fit_keras.history['loss']
    val_loss = fit_keras.history['val_loss']
    x = range(1, len(acc) + 1)
```

```
    plt.figure(figsize=(14, 7))
    plt.subplot(1, 2, 1)
    plt.plot(x, acc, 'b', label='Training Acc.')
    plt.plot(x, val_acc, 'r', label='Testing Acc.')
    plt.title('Training & Testing Accuracy')
    plt.legend()
    plt.subplot(1, 2, 2)
    plt.plot(x, loss, 'b', label='Training Loss')
    plt.plot(x, val_loss, 'r', label='Testing Loss')
    plt.title('Training & Testing Accuracy')
    plt.legend()


# Навчання та оцінка моделей
# k найближчих сусідів
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)
y_pred_knn = knn.predict(X_test)
y_pred_knn_prob = knn.predict_proba(X_test)
confusion_matrix_and_accuracy(X_train, y_train, X_test, y_test, knn, y_pred_knn,
'KNN')
plot_roc_curve_and_auc(X_test, y_test, y_pred_knn_prob, 'KNN')
plot_prc_curve_and_auc(X_test, y_test, y_pred_knn, y_pred_knn_prob, 'KNN')

param_grid = {'n_neighbors': np.arange(3, 30)}
knn = KNeighborsClassifier()
knn_cv = GridSearchCV(knn, param_grid, cv=5)
knn_cv.fit(X_train, y_train)
y_pred_knn_tuned = knn_cv.predict(X_test)
y_pred_knn_tuned_prob = knn_cv.predict_proba(X_test)
print('Best parameters:', knn_cv.best_params_, '\n')
confusion_matrix_and_accuracy(X_train, y_train, X_test, y_test, knn_cv,
y_pred_knn_tuned, 'KNN (Optimized)')
plot_roc_curve_and_auc(X_test, y_test, y_pred_knn_tuned_prob, 'KNN (Optimized)')
plot_prc_curve_and_auc(X_test, y_test, y_pred_knn_tuned, y_pred_knn_tuned_prob,
'KNN (Optimized)')

# Логістична регресія
logreg = LogisticRegression(max_iter=1000)
logreg.fit(X_train, y_train)
y_pred_logreg = logreg.predict(X_test)
y_pred_logreg_prob = logreg.predict_proba(X_test)
plot_top_features(X1, logreg, 'Log. Regression')
confusion_matrix_and_accuracy(X_train, y_train, X_test, y_test, logreg,
y_pred_logreg, 'Log. Regression')
plot_roc_curve_and_auc(X_test, y_test, y_pred_logreg_prob, 'Log. Regression')
plot_prc_curve_and_auc(X_test, y_test, y_pred_logreg, y_pred_logreg_prob, 'Log.
Regression')

param_grid_L1 = {'penalty': ['l1', 'l2'], 'C': np.arange(0.1, 5, 0.1)}
logreg_tuned = LogisticRegression(solver='saga', max_iter=1000)
logreg_tuned_gs = GridSearchCV(logreg_tuned, param_grid_L1, cv=5)
logreg_tuned_gs.fit(X_train, y_train)
y_pred_logreg_tuned = logreg_tuned_gs.predict(X_test)
y_pred_logreg_tuned_prob = logreg_tuned_gs.predict_proba(X_test)
print('Best parameters: ', logreg_tuned_gs.best_params_)
confusion_matrix_and_accuracy(X_train, y_train, X_test, y_test, logreg_tuned_gs,
y_pred_logreg_tuned,
                              'Log. Regression (tuned)')
plot_roc_curve_and_auc(X_test, y_test, y_pred_logreg_tuned_prob, 'Log.
```

Продовження додатку Б

```
 Regression (tuned)')
plot_prc_curve_and_auc(X_test, y_test, y_pred_logreg_tuned,
y_pred_logreg_tuned_prob, 'Log. Regression (tuned)')

# Випадковий ліс
rf = RandomForestClassifier()
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)
y_pred_rf_prob = rf.predict_proba(X_test)
confusion_matrix_and_accuracy(X_train, y_train, X_test, y_test, rf, y_pred_rf,
'Random Forest')
plot_roc_curve_and_auc(X_test, y_test, y_pred_rf_prob, 'Random Forest')
plot_prc_curve_and_auc(X_test, y_test, y_pred_rf, y_pred_rf_prob, 'Random
Forest')

param_grid_rf = {'n_estimators': np.arange(10, 1000, 10),
                 'max_features': ['log2', 'sqrt'],
                 'max_depth': np.arange(1, 20, 1),
                 'criterion': ['gini', 'entropy'],
                 'bootstrap': [True, False]}
rf = RandomForestClassifier()
rf_random_grid = RandomizedSearchCV(estimator=rf,
param_distributions=param_grid_rf, cv=5, verbose=0)
rf_random_grid.fit(X_train, y_train)
y_pred_rf_tuned = rf_random_grid.predict(X_test)
y_pred_rf_tuned_prob = rf_random_grid.predict_proba(X_test)
print('Best parameters: ', rf_random_grid.best_params_)
confusion_matrix_and_accuracy(X_train, y_train, X_test, y_test, rf_random_grid,
y_pred_rf_tuned,
                              'Random Forest (tuned)')
plot_roc_curve_and_auc(X_test, y_test, y_pred_rf_tuned_prob, 'Random Forest
(tuned)')
plot_prc_curve_and_auc(X_test, y_test, y_pred_rf_tuned, y_pred_rf_tuned_prob,
'Random Forest (tuned)')

# Support Vector Machine
support_vector_m = SVC(kernel='rbf', probability=True)
support_vector_m.fit(X_train, y_train)
y_pred_svm = support_vector_m.predict(X_test)
y_pred_svm_prob = support_vector_m.predict_proba(X_test)
confusion_matrix_and_accuracy(X_train, y_train, X_test, y_test,
support_vector_m, y_pred_svm, 'SVM')
plot_roc_curve_and_auc(X_test, y_test, y_pred_svm_prob, 'SVM')
plot_prc_curve_and_auc(X_test, y_test, y_pred_svm, y_pred_svm_prob, 'SVM')

# Нейронна мережа
nn = Sequential()
Input_Shape = X_train.shape[1]
nn.add(layers.Dense(1024, input_shape=(Input_Shape,), activation='relu'))
nn.add(Dropout(0.2))
nn.add(layers.Dense(1024, activation='relu'))
nn.add(Dropout(0.2))
nn.add(layers.Dense(1, activation='sigmoid'))
nn.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
nn.summary()
mc = ModelCheckpoint('best_model.h5', monitor='val_accuracy', mode='max',
verbose=0, save_best_only=True)
fit_nn = nn.fit(X_train, y_train,
                epochs=100,
                verbose=False,
                validation_data=(X_test, y_test),
                batch_size=30, callbacks=[mc])
```

Продовження додатку Б

```
best_nn = load_model('best_model.h5')
plot_history_acc_loss(fit_nn)

y_pred_nn_prob = best_nn.predict(X_test)
y_pred_nn_prob_train = best_nn.predict(X_train)
precision, recall, thresholds = precision_recall_curve(y_train,
y_pred_nn_prob_train)
fscore = np.where(precision + recall != 0,
                  (2 * precision * recall) / np.where(precision + recall == 0,
1, precision + recall), 0)
ix = np.argmax(fscore)
print('Best Threshold = ', round(thresholds[ix], 3))
y_pred_nn_classes = (y_pred_nn_prob > thresholds[ix]).astype("int32")
y_pred_nn_prob_1 = y_pred_nn_prob[:, 0]
y_pred_nn_classes_1 = y_pred_nn_classes[:, 0]

cm = confusion_matrix(y_test, y_pred_nn_classes_1)
accuracy = accuracy_score(y_test, y_pred_nn_classes_1)
plt.figure(figsize=(6, 6))
ax = plt.subplot()
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", ax=ax)
ax.set_xlabel('Predicted')
ax.set_ylabel('Actual')
ax.set_title(f'Confusion Matrix - Neural Network')
plt.show()
print(f'Accuracy: {accuracy:.2f}')
nn_auc = roc_auc_score(y_test, y_pred_nn_prob_1)
print('ROC AUC: %f' % nn_auc)
nn_precision = precision_score(y_test, y_pred_nn_classes_1)
print('Precision: %f' % nn_precision)
nn_recall = recall_score(y_test, y_pred_nn_classes_1)
print('Recall: %f' % nn_recall)
nn_f1 = f1_score(y_test, y_pred_nn_classes_1)
print('F1 score: %f' % nn_f1)
precision, recall, _ = precision_recall_curve(y_test, y_pred_nn_prob_1)
nn_prc_auc = auc(recall, precision)
print('PRC AUC: %f' % nn_prc_auc)

# CatBoostClassifier
catboost_model = CatBoostClassifier(iterations=1000, depth=6, learning_rate=0.1,
loss_function='Logloss')
catboost_model.fit(X_train, y_train, verbose=False)
y_pred_catboost = catboost_model.predict(X_test)
y_pred_catboost_prob = catboost_model.predict_proba(X_test)[:, 1]
confusion_matrix_and_accuracy(X_train, y_train, X_test, y_test, catboost_model,
y_pred_catboost, 'CatBoost')
plot_roc_curve_and_auc(X_test, y_test, y_pred_catboost_prob, 'CatBoost')
plot_prc_curve_and_auc(X_test, y_test, y_pred_catboost, y_pred_catboost_prob,
'CatBoost')

# Define the parameter grid
param_grid = {
    'iterations': sp_randint(500, 2000),
    'depth': sp_randint(4, 10),
    'learning_rate': [0.01, 0.1, 0.5],
    'loss_function': ['Logloss', 'CrossEntropy']
}

catboost_model = CatBoostClassifier()
random_search = RandomizedSearchCV(estimator=catboost_model,
param_distributions=param_grid, n_iter=10, cv=5)
```

```
random_search.fit(X_train, y_train)
best_catboost_model = random_search.best_estimator_
best_params = random_search.best_params_
best_catboost_model.fit(X_train, y_train)
y_pred_catboost = best_catboost_model.predict(X_test)
y_pred_catboost_prob = best_catboost_model.predict_proba(X_test)[:, 1]
print("Best parameters: ", best_params)
confusion_matrix_and_accuracy(X_train, y_train, X_test, y_test,
best_catboost_model, y_pred_catboost,
                              'CatBoost (tuned)')
plot_roc_curve_and_auc(X_test, y_test, y_pred_catboost_prob, 'CatBoost (tuned)')
plot_prc_curve_and_auc(X_test, y_test, y_pred_catboost, y_pred_catboost_prob,
'CatBoost (tuned)')

feature_importance = best_catboost_model.get_feature_importance()
feature_names = X1.columns
sorted_indices = np.argsort(feature_importance)[::-1]
sorted_feature_names = feature_names[sorted_indices]
sorted_feature_importance = feature_importance[sorted_indices]
top_feature_names = sorted_feature_names[:15]
top_feature_importance = sorted_feature_importance[:15]
plt.figure(figsize=(10, 6))
plt.bar(range(len(top_feature_names)), top_feature_importance)
plt.xticks(range(len(top_feature_names)), top_feature_names, rotation=90)
plt.xlabel('Features')
plt.ylabel('Importance')
plt.title('Features Importance CatBoost')
plt.tight_layout()
plt.show()
```