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

Навчально-науковий інститут прикладного системного аналізу

Кафедра штучного інтелекту

До захисту допущено:

Завідувач кафедри

_____ Ірина Джигирей

«___»_____20__ р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системи і методи штучного інтелекту»

спеціальності 122 «Комп'ютерні науки»

на тему: «Порівняльний аналіз методів машинного навчання для прогнозу відтоку користувачів на основі поведінкових даних»

Виконав:
студент IV курсу, групи КІ-21,
Масловський Дмитро Володимирович _____

Керівник:
доцентка кафедри штучного інтелекту, д-р. філос., доцент
Гуськова Віра Геннадіївна _____

Консультант з нормоконтролю:
фахівець І категорії кафедри штучного інтелекту,
к.т.н., доцент, Комариста Богдана Миколаївна _____

Рецензент:
професорка кафедри математичних методів системного аналізу,
д.т.н., професор, Дмитрієва Ольга Анатоліївна _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2026 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий інститут прикладного системного аналізу
Кафедра штучного інтелекту

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 «Комп'ютерні науки»

Освітньо-професійна програма «Системи і методи штучного інтелекту»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Ірина Джигирей
«15» січня 2026 р.

ЗАВДАННЯ
на дипломний проєкт студенту
Масловському Дмитру Володимировичу

1. Тема проєкту «Порівняльний аналіз методів машинного навчання для прогнозу відтоку користувачів на основі поведінкових даних», керівник проєкту Гуськова Віра Геннадіївна, д.ф. з к.н., доцент, затверджені наказом по університету від «27» травня 2026 р. №1944-с.

2. Термін подання студентом проєкту «05» червня 2026 року.

3. Вихідні дані до проєкту: історичні агреговані фінансові, демографічні та геопросторові дані абонентів телекомунікаційної компанії, а також історія споживання сервісів, отримані з відкритого набору даних на платформі Kaggle.

4. Зміст пояснювальної записки: Аналіз предметної області відтоку користувачів та датасету, дослідження та вибір оптимальних методів попередньої обробки даних та машинного навчання, розробка алгоритму для прогнозування відтоку користувачів, оцінка ефективності та порівняння отриманих результатів із реальними даними.

5. Перелік графічного матеріалу: електронна презентація.

6. Дата видачі завдання «02» лютого 2026 року.

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1	Дослідження предметної області	27.04.2026	Виконано
2	Написання першого розділу роботи	04.05.2026	Виконано
3	Написання другого розділу роботи	11.05.2026	Виконано
4	Розроблення програмного продукту	18.05.2026	Виконано
5	Написання третього розділу роботи	25.05.2026	Виконано
6	Оформлення записки згідно з положеннями нормоконтролю	01.06.2026	Виконано
7	Підготовка презентації та доповіді	05.06.2026	Виконано

Студент Дмитро МАСЛОВСЬКИЙ

Керівник Віра ГУСЬКОВА

РЕФЕРАТ

Дипломна робота: 118 с., 53 рис., 12 табл., 27 посилань, додаток.

МАШИННЕ НАВЧАННЯ, БІНАРНА КЛАСИФІКАЦІЯ, ПРОГНОЗУВАННЯ ВІДТОКУ, PYTHON, SCIKIT-LEARN, ПОВЕДІНКОВІ ОЗНАКИ.

Об'єктом дослідження є процес прогнозування відтоку користувачів на основі поведінкових даних із застосуванням методів машинного навчання.

Предметом дослідження є методи машинного навчання для бінарної класифікації відтоку користувачів.

Метою роботи є проведення порівняльного аналізу методів машинного навчання для прогнозування відтоку користувачів та визначення найбільш ефективного підходу.

Актуальність дослідження зумовлена зростанням економічного значення задачі утримання клієнтів. У роботі проаналізовано явище відтоку користувачів, його вплив на фінансові показники бізнесу та сучасні підходи до прогнозування відтоку в різних галузях. Розглянуто алгоритми машинного навчання та методи попередньої обробки даних. Розроблено програму мовою Python, яка реалізує навчання та порівняння шести класичних та ансамблевих методів машинного навчання на відкритому наборі даних Telco Customer Churn. Встановлено, що найкращі результати продемонструвала LightGBM із застосуванням методу синтетичного збільшення міноритарного класу (SMOTE). Також показано недоцільність використання методу головних компонент (PCA) для цього набору даних та позитивний вплив SMOTE на якість класифікації. Аналіз важливості ознак підтвердив ефективність застосованого підходу до формування ознак.

Отримані результати підтверджують практичну цінність запропонованого підходу та можливість його адаптації для інших задач прогнозування відтоку користувачів.

ABSTRACT

Thesis: 118 p., 53 figures, 12 tables, 27 references, appendix.

MACHINE LEARNING, BINARY CLASSIFICATION, CHURN PREDICTION, PYTHON, SCIKIT-LEARN, BEHAVIORAL FEATURES.

The object of the study is the process of customer churn prediction based on behavioral data using machine learning methods.

The subject of the study is machine learning methods for the binary classification of customer churn.

The purpose of the study is to conduct a comparative analysis of machine learning methods for customer churn prediction and to determine the most effective approach.

The relevance of the study is driven by the growing economic significance of customer retention. The paper analyzes the phenomenon of customer churn, its impact on business performance indicators, and modern approaches to churn prediction in various industries. Machine learning algorithms and data preprocessing methods are reviewed. A Python program was developed to implement the training and comparison of six classical and ensemble machine learning methods on the open Telco Customer Churn dataset. It was established that LightGBM with the application of the Synthetic Minority Over-sampling Technique (SMOTE) demonstrated the best results. The study also demonstrated the impracticality of using Principal Component Analysis (PCA) for this dataset and the positive impact of SMOTE on classification performance. Feature importance analysis confirmed the effectiveness of the applied feature engineering approach.

The obtained results confirm the practical value of the proposed approach and demonstrate the possibility of its adaptation to other customer churn prediction tasks.

ЗМІСТ

ВСТУП

У сучасному бізнес-середовищі існує значна кількість метрик, які використовуються для оцінювання ефективності продуктів та ухвалення обґрунтованих управлінських рішень. Однією з ключових таких метрик є показник відтоку користувачів, який відображає частку користувачів, що припинили використання продукту або послуги протягом певного періоду часу.

Прогнозування відтоку користувачів є важливою задачею для компаній, оскільки дозволяє завчасно виявляти потенційні ризики втрати клієнтів та ухвалювати проактивні рішення, спрямовані на їх утримання. Зменшення рівня відтоку безпосередньо впливає на фінансові показники бізнесу, зокрема на довгострокову прибутковість та ефективність стратегії залучення клієнтів.

Задачі прогнозування є одним із найпоширеніших напрямів застосування методів машинного навчання у бізнесі. Сучасні компанії активно використовують предиктивні моделі для аналізу історичних даних та прогнозування поведінки користувачів, що дозволяє ухвалювати швидші та обґрунтованіші рішення.

Для розв’язання задачі прогнозування відтоку застосовується широкий спектр методів машинного навчання, які активно використовуються в різних галузях, зокрема в SaaS-сервісах, банківському секторі та e-commerce. При цьому вибір конкретного методу суттєво залежить від специфіки бізнес-контексту та характеристик даних, що використовуються для навчання моделей.

З розвитком методів штучного інтелекту та збільшенням обсягів доступних даних можливості побудови предиктивних моделей значно розширюються. Сучасні алгоритми машинного навчання демонструють високу ефективність у задачах класифікації та прогнозування поведінки користувачів, що робить їх важливим інструментом аналітики даних.

У межах цієї бакалаврської роботи виконується порівняльний аналіз методів машинного навчання для прогнозування відтоку користувачів на основі поведінкових даних з метою визначення найбільш ефективного підходу для розв’язання цієї задачі.[1]

[1] Тут і нижче використано такий інструмент штучного інтелекту як чат-бот з генеративним штучним інтелектом ChatGPT виключно для корегування та редагування тексту, створеного автором цієї дипломної роботи, на основі автоматизованої перевірки граматики, структури та стилю, що відповідає Політиці використання штучного інтелекту для академічної діяльності в КПІ ім. Ігоря Сікорського (протокол №11 Вченої ради КПІ ім. Ігоря Сікорського від 11 грудня 2023 р.).

РОЗДІЛ 1 ПОСТАНОВКА ЗАДАЧІ ПРОГНОЗУВАННЯ ВІДТОКУ КОРИСТУВАЧІВ ТА ЇЇ АКТУАЛЬНІСТЬ

1.1 Поняття відтоку та його значення

У сучасних бізнес-продуктах утримання клієнтів є одним із головних показників ефективності бізнесу. Одним з основних понять у цьому контексті є поняття відтоку користувачів, яке характеризує явище повного припинення користувачем взаємодії з продуктом/послугою або зниження рівня залученості клієнта протягом певного періоду часу.

Під відтоком користувачів розуміють процес, за якого клієнт свідомо або несвідомо відмовляється від подальшого використання сервісу – скасовує підписку, припиняє здійснювати покупки або переходить до конкурента [1, 11]. Визначення моменту відтоку залежить від специфіки галузі: у телекомунікаційній сфері це може бути розірвання контракту, у SaaS (Software as a Service) – скасування підписки, у банківській галузі – закриття рахунку або припинення активного користування продуктами банку.

Наразі виділяють дві основні маркетингові стратегії, що спрямовані на зростання клієнтської бази: залучення нових клієнтів (customer acquisition) та утримання існуючих (customer retention). Вартість залучення нового клієнта може коштувати набагато більше, ніж вартість утримання клієнта, тому інвестування в роботу з клієнтами, які мають ризик відтоку, є доцільним та ефективним. До того ж, якщо ефективно залучати клієнтів і витрачати на це багато коштів, але погано утримувати їх, то виникає питання щодо доцільності витрачати стільки грошей на залучення клієнтів.

Залежно від причин виникнення відтоку можна виділити кілька його типів.

1. Добровільний відтік – ситуація, коли користувач самостійно вирішує припинити користуватися послугами компанії. Це може виникати

через незадоволення продуктом або відчуття, що продукт не приносить очікуваної цінності.

2. Вимушений відтік – виникає через зовнішні фактори, які не залежать від користувача: проблеми з оплатою, мережеві збої або закінчення терміну дії платіжних даних.
3. Випадковий відтік – спричинений зміною місця проживання або фінансового стану користувача.
4. Усвідомлений відтік – спричинений бажанням клієнта змінити постачальника послуг через такі фактори, як низька якість сервісу, цінова політика компанії або невідповідність очікуванням користувача.
5. Прихований відтік – ситуація, коли клієнт ще залишається активним, але його взаємодія з продуктом знижується. Тож фактично він вже несе меншу цінність для бізнесу [3, 11].

У бізнесі для кількісної оцінки відтоку використовується метрика частки відтоку. Вона визначає відсоток клієнтів, яких компанія втрачає за певний період часу, та дозволяє оцінювати динаміку клієнтської бази. Зазвичай ця метрика розраховується як відношення кількості користувачів, що припинили використання продукту протягом певного періоду, до загальної кількості користувачів на початку цього періоду [11].

Однак цю метрику потрібно розглядати у рамках конкретного бізнес-контексту, бо для певного бізнесу її рівень, а отже і підходи для її аналізу та прогнозування будуть відрізнятися. Рівень річної частки відтоку залежно від сектору зображений на рисунку 1.1.

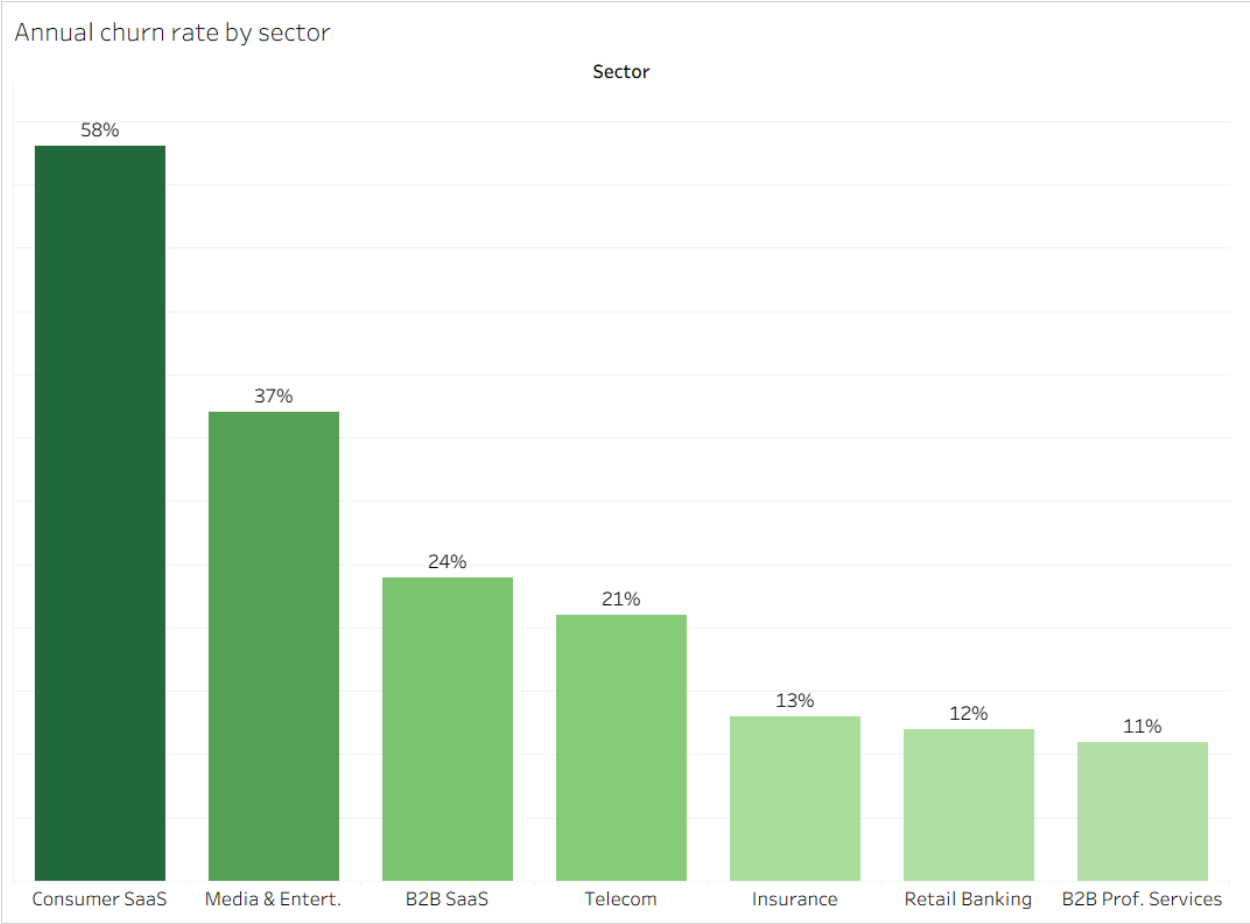


Рисунок 1.1 – Річна частка відтоку за секторами (на основі [17])

На рівень відтоку клієнтів впливає низка факторів, пов’язаних як із якістю сервісу, так і з поведінкою користувачів. Однією з основних причин відтоку є незадоволеність клієнтів продуктом або послугою, що призводить до переходу до конкурентів у разі невідповідності очікуванням. Іншим важливим індикатором ризику відтоку є рівень використання сервісу, що характеризується частотою та інтенсивністю взаємодії користувача з продуктом. На рішення щодо припинення користування також можуть впливати конкурентне середовище та загальна якість сервісу [13].

Відтік є задачею бінарної класифікації: кожен користувач у кінці певного періоду позначається як такий, що покинув продукт або залишився. Відповідно позначається як 1 або 0 [4].

Прогнозування відтоку клієнтів є важливим явищем у бізнесі, яке допомагає виявляти ранні сигнали потенційного відтоку та ідентифікувати користувачів із підвищеним ризиком його виникнення. Це дозволяє компаніям ефективно використовувати наявні дані для формування стратегій утримування клієнтів та підвищення рівня їхньої лояльності [3].

1.2 Бізнес-значення відтоку користувачів

Проблема відтоку користувачів є однією з найзначніших загроз для стабільного розвитку та масштабування багатьох компаній. За результатами сучасних аналітичних та маркетологічних досліджень вартість залучення користувача може коштувати від 5 до 25 разів дорожче, ніж утримання існуючого залежно від галузі (рис. 1.2) [16, 17]. Тож рівень відтоку користувачів напряму впливає на вартість залучення клієнта (CAC, Customer Acquisition Cost). Цей показник відображає скільки в середньому компанія витрачає грошей на залучення одного користувача. Тому зменшення відтоку дозволяє оптимізувати витрати на маркетинг і підвищити загальну прибутковість бізнесу [18].

Ключовою фінансовою метрикою, пов'язаною із відтоком, є цінність користувача за весь час користування продуктом (Customer Lifetime Value, CLTV). CLTV відображає загальний прибуток, який компанія отримує від клієнта за весь час взаємодії з ним. Зменшення відтоку безпосередньо збільшує середнє значення CLTV, оскільки подовжує тривалість відносин із клієнтом. Динаміку зростання CLTV залежно від тривалості утримання клієнта показано на рисунку 1.3. Він демонструє, що лояльні клієнти приносять набагато більше доходу, ніж середньостатистичні користувачі [6, 20]. Для SaaS-компаній CLTV є одним із головних показників оцінки бізнесу, тому що монетизація відбувається через щомісячні або щорічні передплати.

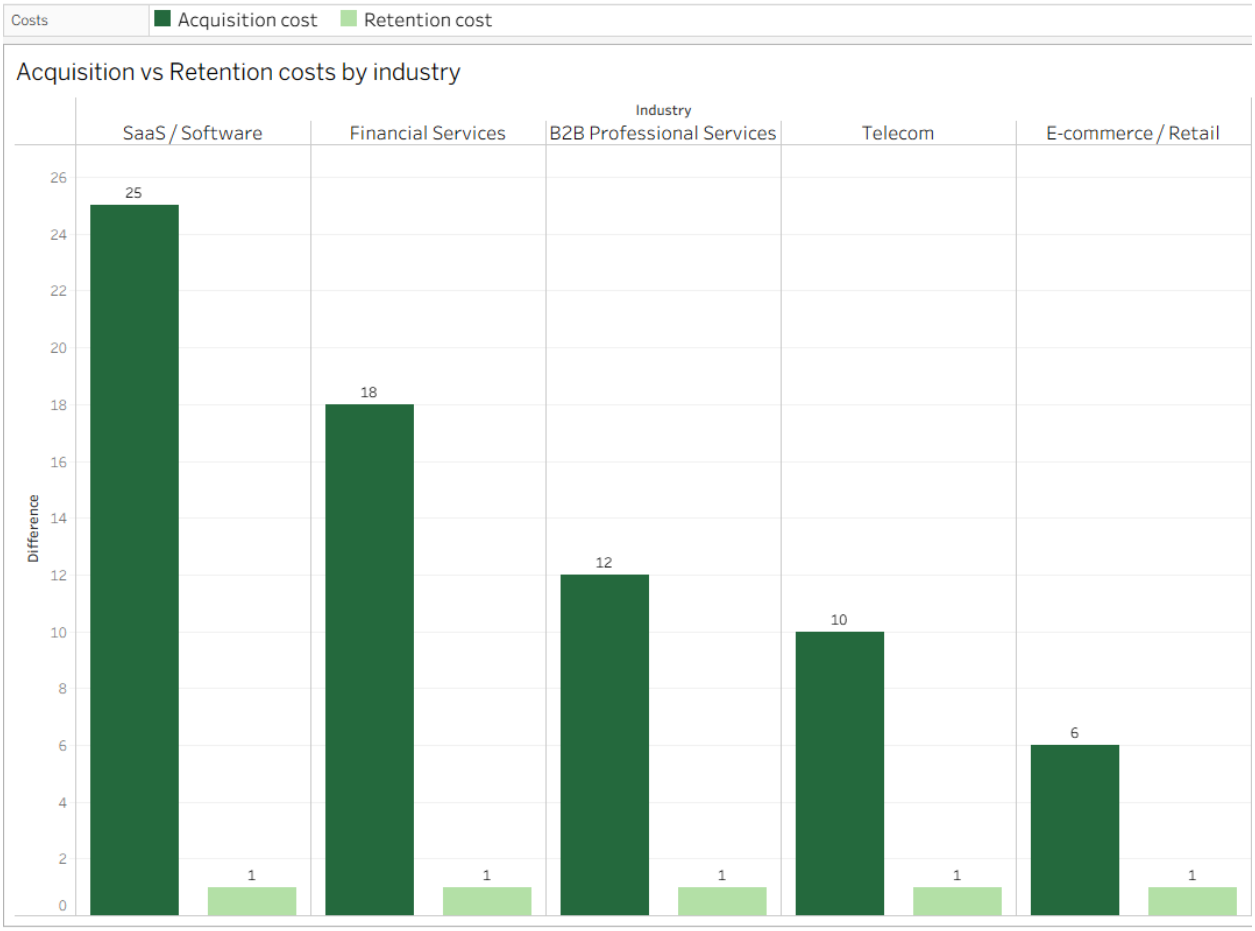


Рисунок 1.2 – Порівняння вартості залучення та утримання для різних сфер (на основі [17])

Вплив відтоку на доходи компанії також проявляється через показник передбачуваного місячного регулярного доходу (MRR, Monthly Recurring Revenue). Відтік клієнтів призводить до втрат доходу, який був внесений у MRR, і у результаті цей показник падає. Тому навіть незначне зниження рівня відтоку здатне суттєво вплинути на фінансову дохідність компанії: навіть утримуючи лише на 5% більше клієнтів, компанії можуть збільшити свої прибутки у рази [5, 7, 18].

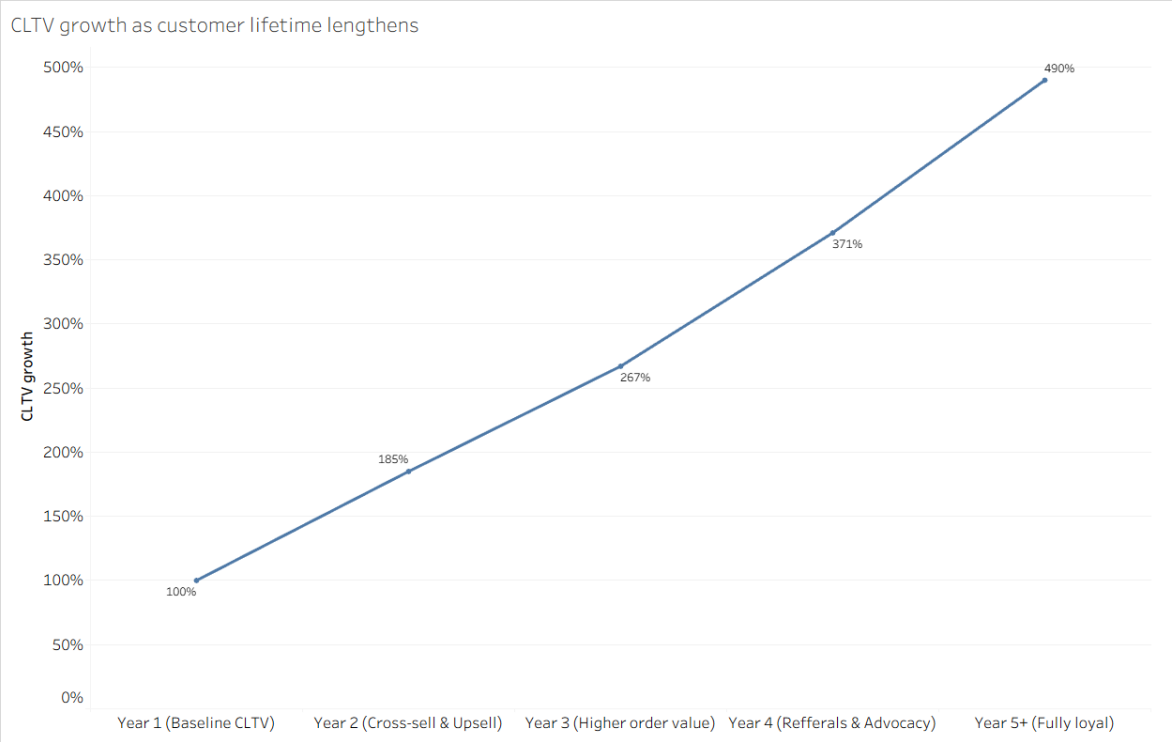


Рисунок 1.3 – Зміна CLTV впродовж збільшення часу користування продуктом (на основі [17])

Відтік клієнтів, окрім безпосередньої втрати користувачів, має також додаткові наслідки для компанії, зокрема вплив на її репутацію та бренд. Незадоволені клієнти можуть залишати негативні відгуки, що потенційно знижує рівень довіри з боку нових користувачів і зменшує ймовірність їх залучення. Водночас лояльні клієнти сприяють органічному зростанню компанії через рекомендації та позитивний досвід використання продукту. Наприклад, у практиці однієї з провідних компаній у сфері житлового будівництва США встановлено, що понад 60% продажів формуються саме завдяки рекомендаціям існуючих клієнтів. Це підкреслює, що ефективне управління відтоком має не лише утримувальний ефект, але й безпосередньо впливає на здатність компанії залучати нових користувачів [5].

Управління відтоком клієнтів можна здійснювати двома способами: реактивним та проактивним. При реактивному підході компанія очікує запиту на скасування, отриманого від клієнта, після чого надає йому різні

пропозиції для утримання. При проактивному підході прогнозується ймовірність відтоку і компанія надає пропозиції для утримання ще до того, як він вирішить завершити користування продуктом. Моделі прогнозування відтоку дозволяють бізнесу перейти від реактивного до проактивного управління аудиторією. Завдяки завчасному виявленню клієнтів із підвищеним ризиком відтоку, компанії можуть ефективно застосовувати персоналізовані пропозиції та програми лояльності [3].

Задача прогнозування відтоку має пряму фінансову цінність для бізнесу і є одним із найбільш практично значущих застосувань методів машинного навчання у сфері аналізу клієнтської поведінки.

1.3 Сфери застосування та сучасні підходи до прогнозування відтоку користувачів

Явище відтоку користувачів зустрічається у компаній майже з будь-якої сфери, тому вони мають вміти з ним працювати. Прогнозуванням відтоку займаються компанії з таких галузей, як телекомунікації, роздрібна торгівля, банкінг, страхування, охорона здоров’я, освіта та інші. Проте ефективність підходів до вирішення задачі суттєво відрізняється залежно від сфери застосування [1, 12, 13].

Серед галузей, де прогнозування відтоку набуло найбільшого розвитку, виділяють 4 основні: телекомунікації, e-commerce, банкінг та SaaS. Кожна з них має власні особливості, що визначає вигляд даних, доступних для прогнозування, та підходів до моделювання [2]. Один з оглядів, що охоплює 61 дослідження за 2020–2024 роки, підтверджує, що ансамблеві методи (XGBoost, LightGBM) переважають у більшості галузей, тоді як методи глибокого навчання частіше застосовуються для роботи зі складними послідовними даними [12].

1.3.1 Особливості та підходи у SaaS та телекомунікаціях

SaaS (Software as a Service) – це модель надання програмного забезпечення на вимогу, за якої користувачі отримують доступ до програмного продукту без локального завантаження. SaaS-бізнес активно потребує аналізу відтоку, оскільки він впливає на стабільність доходу та конкурентоспроможність компаній. У SaaS відтоком зазвичай вважається скасування підписки, що напряму впливає на MRR [13].

Особливістю SaaS є висока деталізація поведінкових даних користувачів (логіни, сесії, використання функцій), що формує багатовимірний простір ознак. Це зумовлює ефективність застосування різних моделей машинного навчання. У дослідженнях показано, що логістична регресія та XGBoost часто демонструють кращі результати, тоді як Random Forest, метод головних компонент (PCA), метод опорних векторів (SVM) та метод k-найближчих сусідів (k-NN) можуть поступатися на даних високої розмірності [14, 15]. Водночас результати різних робіт є несистемними та залежать від конкретного набору даних і постановки задачі [24].

Телекомунікаційна галузь є однією з перших сфер, де задача прогнозування відтоку набула системного розвитку, що пов’язано з високою конкуренцією та доступністю поведінкових даних [1].

У галузі виділяють постоплатні та передоплатні мережі. У постоплатних мережах відтік є явним і відповідає розірванню контракту, тоді як у передоплатних він визначається тривалою пасивністю клієнта. Типові ознаки включають характеристики дзвінків, SMS, тарифного плану, звернення до підтримки та інші поведінкові показники.

Дослідження показують, що телекомунікаційні дані характеризуються значним дисбалансом та великою кількістю ознак, що зумовлює використання як класичних, так і ансамблевих методів машинного навчання. У різних роботах встановлено, що XGBoost, AdaBoost, нейронні мережі та

дерева рішень зазвичай демонструють високу ефективність, тоді як логістична регресія та k-NN можуть поступатися складнішим моделям [1–3, 15]. Наприклад, у дослідженні передоплатної мережі з 10 млн абонентів зафіксовано природний рівень відтоку близько 25% кожні два місяці, а найкраща модель досягла точності 89,4% [15]. В іншій роботі AdaBoost та XGBoost показали найвищий AUC (площа під кривою) на рівні 84%, перевершуючи логістичну регресію, SVM та Naive Bayes [3]. Також зазначається, що k-NN демонструє гірші результати через чутливість до масштабу ознак і обчислювальну складність на великих вибірках [15]. Водночас порівняння результатів між дослідженнями ускладнюється через відмінності у визначеннях відтоку, наборах ознак та часових горизонтах аналізу [12].

1.3.2 Особливості та підходи у банківській сфері та e-commerce

Довгострокові контракти та різноманіття продуктів (рахунки, кредити, депозити, картки) формують особливий контекст, що принципово відрізняє задачу у банківській сфері від інших сфер [2]. На відміну від SaaS, у банківській сфері відтік часто є прихованим та поступовим. Це ускладнює розмітку навчальних даних: мітка відтік/не відтік не повністю відображає реальну динаміку [2, 4].

В одному з досліджень було встановлено, що нейронні мережі та дерева рішень перевершують логістичну регресію за точністю класифікації, проте поступаються їй за прозорістю рішень [2]. Регуляторне середовище GDPR (загальний регламент про захист даних) у Європі зобов'язують фінансові установи пояснювати автоматизовані рішення. Як наслідок, у практиці банків поширені моделі з вбудованою інтерпретованістю або пост-хок пояснення за

допомогою методів адитивних пояснень Шеплі (SHAP) і локально-інтерпретованих модель-агностичних пояснень (LIME) [12].

Результати окремих досліджень демонструють, що у банківській сфері градієнтний бустинг стабільно дає найвищі значення AUC – у діапазоні 0,82–0,91 залежно від набору ознак та рівня дисбалансу класів.

На відміну від SaaS або банківського сектору, відтік у e-commerce визначається відсутністю повторних покупок протягом певного періоду часу. Відсутність чіткого моменту завершення взаємодії є одним із головних викликів. Для його вирішення використовується поняття поведінкового мовчання: якщо клієнт не здійснює покупок протягом визначеного періоду, він розглядається як потенційний кандидат на відтік [3].

Поведінкові дані в e-commerce включають перегляди, пошукові запити, взаємодію із системою, відкриття email-розсилок, участь у програмах лояльності та реакцію на промоакції. Дослідження показують, що використання мобільних поведінкових ознак підвищує якість прогнозування порівняно з моделями, які базуються лише на транзакційних даних [3].

Для прогнозування відтоку застосовуються градієнтний бустинг (XGBoost, LightGBM), рекурентні нейронні мережі (RNN), довга короткочасна пам'ять (LSTM), вентильний рекурентний вузол (GRU) та згорткові нейронні мережі (CNN). LSTM демонструє високу ефективність у випадках, коли важливими є порядок та часові інтервали між подіями, тоді як ансамблеві методи відзначаються швидкістю навчання та можливістю інтерпретації результатів [10, 12].

1.4 Характеристика формування ознак

1.4.1 Види ознак та їхня роль у прогнозуванні відтоку

Побудова ефективної моделі прогнозування відтоку неможлива без якісного формування вхідних ознак. Процес формування ознак – це перетворення сирих даних про поведінку та характеристики клієнтів у числові або категоріальні атрибути, придатні для навчання алгоритмів машинного навчання. Дослідження підтверджують, що навіть відносно простий алгоритм на якісно сформованому наборі ознак перевершує складну модель на слабкому наборі [1, 2, 15].

Ознаки у задачах прогнозування відтоку поділяють на декілька видів, кожна з яких відображає різний аспект взаємодії клієнта з продуктом або сервісом.

1. Демографічні та профільні ознаки характеризують клієнта: вік, стать, географічне розташування, тип підписки, дата реєстрації, канал залучення. Ці ознаки є переважно статичними або змінюються рідко. Вони є важливими для сегментації: різні демографічні групи можуть показувати різну схильність до відтоку, тому демографічні ознаки підвищують якість моделі у поєднанні з іншими категоріями [2].
2. Поведінкові ознаки є найбільш інформативною категорією для задачі прогнозування відтоку. Вони відображають реальну активність користувача в системі: частоту входів, тривалість сесій, кількість і тип використаних функцій, глибину взаємодії з продуктом. Поведінкові ознаки є динамічними – вони змінюються з часом і здатні відображати поступову деградацію залученості клієнта задовго до фактичного відтоку. У дослідженнях саме поведінкові ознаки несуть велику цінність для прогнозування для моделей в SaaS та мобільних застосунках [3, 15].

3. RFM-ознаки (Recency, Frequency, Monetary або Давність, Частота, Грошова цінність) є підкатегорією, що виникла в маркетинговому аналізі та описує транзакційну поведінку клієнта через три виміри: давність останньої взаємодії, частоту та фінансовий обсяг транзакцій. Незважаючи на простоту, RFM стабільно демонструє високу цінність і широко використовуються в e-commerce та банківській сфері. Наукові роботи показуть, що на основі лише цих трьох вимірів можна будувати статистичні моделі з достатньо прийнятною точністю [2, 8].
4. Часові ознаки відображають динаміку змін у поведінці клієнта в часі: тренди активності, волатильність, серії неактивних днів, ковзні середні. Часові ознаки фіксують не поточний стан, а напрямок зміни, що є важливим для раннього виявлення ризику відтоку. Клієнт із поступово спадаючою активністю є більш ризикованим, ніж клієнт зі стабільно низькою активністю [6].
5. Ознаки взаємодії з підтримкою та зворотного зв'язку включають кількість звернень до служби підтримки, теми звернень, оцінки задоволеності (індекс лояльності клієнтів (NPS) та індекс задоволеності клієнтів (CSAT)) та відгуки. Вони є важливими індикаторами невдоволення клієнта і можуть попередити про відтік, що настане у найближчому майбутньому [2].

Дослідження демонструють, що систематичне формування тисяч ознак із сирих даних з наступним відбором найбільш значущих дозволило досягти точності 88–89%, яку важко отримати на стандартному наборі з кількох десятків атрибутів. Загальний принцип, підтверджений у більшості досліджень, полягає у тому, що комбінація кількох категорій ознак ефективніша за будь-яку окрему категорію [12, 15].

Класифікація ознак зображена на таблиці 1.1.

Таблиця 1.1 – Класифікація ознак для прогнозування відтоку

Категорія ознак	*Приклади*	*Тип даних*	*Предиктивна цінність*	*Застосування*
Демографічні	Вік, регіон, тариф, канал залучення	Категоріальні / числові	Низька – середня	Всі сфери
Поведінкові	Частота входів, тривалість сесій, залучені функції	Числові (агрегати)	Висока	SaaS, мобільні застосунки
RFM	Давність, частота, сума транзакцій	Числові	Висока	E-commerce, банківська сфера
Часові	Тренди активності, серії пасивності, ковзні середні	Числові / часові ряди	Дуже висока	Всі сфери
Підтримка	Кількість звернень, NPS, CSAT, теми скарг	Числові / категоріальні	Середня – висока	Всі сфери

1.4.2 Характеристика поведінкових, часових та RFM-ознак

Поведінкові ознаки є головною складовою сучасних систем прогнозування відтоку. Ці ознаки забезпечують найвищу цінність, оскільки показують рівень залученості клієнта у продукті [3].

Головними поведінковими ознаками є: кількість сесій за одиницю часу, середня тривалість сесії, кількість унікальних функцій, якими скористався клієнт, частка ключових функцій продукту, що були активовані, та глибина використання продукту. На основі даних будують сотні похідних

поведінкових ознак – від загальної кількості дзвінків до пропорції вхідних і вихідних викликів у різний час доби, і доводять, що деякі неочевидні комбінації є потужнішими для прогнозу, ніж традиційні прості метрики [15].

Технічно поведінкові ознаки отримуються з систем збору подій: Mixpanel, Amplitude, Segment або на базі Apache Kafka. Ці системи фіксують кожну подію взаємодії з продуктом у вигляді потоку подій із мітками часу, далі виконується формування ознак через SQL-агрегації (SQL – мова структурованих запитів) або потокову обробку даних [3].

На відміну від статичних агрегатів, часові ознаки фіксують динаміку змін у поведінці клієнта, дозволяючи моделі виявляти тенденції спаду залученості на ранніх стадіях ще до того, як відтік стає неминучим [6].

Тренд активності є базовою часовою ознакою і відображає напрямок зміни рівня залученості. Негативний тренд є сигналом можливого відтоку. Різкі перепади між активним та пасивним використанням може свідчити про нестабільне використання продукту, що корелює з підвищеним ризиком відтоку [3, 6].

Ковзне середнє та експоненційно зважене ковзне середнє (EWMA) є поширеними інструментами згладжування та виявлення трендів. Одна з робіт показує, що рекурентні нейронні мережі можуть знаходити довгострокові залежності недоступні класичним методам [6, 10].

RFM-аналіз є класичним підходом до характеристики поведінки клієнта, що виник у прямому маркетингу наприкінці 1990-х років і набув широкого застосування для прогнозування відтоку [8].

Компонент Recency відображає час, що минув з моменту останньої взаємодії клієнта – покупки, входу в систему або транзакції. Малі значення Recency свідчать про нещодавню активність і вказують на низький шанс відтоку [8].

Компонент Frequency вимірює кількість певних подій важливих для бізнесу за певний період. Висока частота свідчить про звикання користувача до продукту, що зменшує ризик його відтоку [8].

Компонент Monetary показує фінансові витрати клієнта за певний період. Високе значення цього параметра не лише демонструє залученість, але й показує, що утримання клієнтів із вищою CLTV є економічно доцільнішим [6].

Попри свою простоту, RFM має ряд обмежень. Вона фіксує стан лише в певний момент без урахування динаміки. Поширенішим є підхід, за якого RFM-ознаки використовуються не ізольовано, а як компонент розширеного вектора ознак поряд із поведінковими та часовими показниками [2, 8].

1.4.3 Характеристика демографічних ознак, ознак підтримки та зворотного зв'язку

Демографічні ознаки є однією з базових категорій у задачах прогнозування відтоку. До них належать статичні або рідко змінювані характеристики клієнта: вік, стать, географічне розташування, тип підписки або тарифного плану, дата реєстрації та канал залучення [2, 3].

Цінність демографічних ознак для прогнозування відтоку є нижчою порівняно з поведінковими або часовими ознаками, тому у багатьох дослідженнях їх зазвичай включають як допоміжний компонент разом з іншими групами ознак [1, 2]. Можна зробити висновок, що демографічні ознаки найбільш корисні не самостійно, а у поєднанні з поведінковими даними [2, 3, 13].

Ознаки взаємодії з підтримкою та зворотного зв'язку відображають досвід клієнта у вирішенні проблем і його загальне ставлення до продукту. До цієї категорії належать: кількість звернень до служби підтримки за певний період, теми та типи звернень, час очікування відповіді й вирішення проблеми, а також оцінки задоволеності – NPS (Net Promoter Score), CSAT (Customer Satisfaction Score) та відгуки у відкритій формі [2].

Зростання кількості звернень до підтримки без їх успішного вирішення спричиняє ризик відтоку [2, 3]. Однак більшість клієнтів не звертаються до підтримки навіть при наявності невдоволення, тому відсутність звернень не є свідченням задоволеності [3, 13]. Комбінування цих ознак з поведінковими даними дозволяє підвищити точність прогнозу [2]. Перелік ознак, які використовуються для прогнозування відтоку, наведені в таблиці 1.2.

Таблиця 1.2 – Ознаки, які використовуються для прогнозування відтоку

Група ознак	*Предиктивна цінність*	*Складність формування*	*Вимоги до даних*	*Пріоритет включення*
Демографічні	Низька – середня	Низька	Профільні дані	Допоміжний
Поведінкові	Висока	Середня	Event logs	Обов'язковий
RFM	Висока	Низька	Транзакційні дані	Обов'язковий
Часові (тренди, EWMA)	Дуже висока	Середня – висока	Часові ряди подій	Рекомендований
Часові ряди (LSTM/GRU)	Дуже висока	Висока	Великий обсяг логів	За наявності ресурсів
Підтримка	Середня – висока	Середня	CRM, тікети	Рекомендований

1.5 Сценарії залучення поведінкових характеристик

Поведінкові дані є важливим джерелом інформації для задачі прогнозування відтоку користувачів, однак їх зміст суттєво залежить від галузі застосування та типу продукту. Перед побудовою моделей необхідно визначити, які саме дії користувачів фіксуються інформаційною системою та які з них можуть бути інформативними для прогнозування відтоку. У

загальному випадку можна виділити кілька основних сценаріїв використання поведінкових характеристик [3, 16].

Сценарій на основі журналів подій (event logs) характерний для SaaS-продуктів та мобільних застосунків. У цьому випадку дії користувача фіксуються як послідовність подій із часовими мітками, наприклад, вхід у систему, використання функцій або завершення сесії. Такі дані дозволяють оцінювати рівень активності та залученості користувача. На їх основі формуються агреговані характеристики, зокрема кількість сесій, тривалість взаємодії та частота використання функцій [3, 16].

Сценарій на основі транзакційних даних використовується переважно у банківській сфері та e-commerce. Поведінка користувача у цьому випадку визначається через фінансові операції, зокрема частоту, обсяг та структуру платежів або покупок [5, 8].

Сценарій на основі CDR-даних (Call Detail Records – деталізовані записи про виклик) є специфічним для телекомунікаційної галузі. Такі дані містять інформацію про дзвінки та SMS, включаючи їх тривалість, напрямок та час здійснення. CDR-дані дозволяють аналізувати активність користувачів та формувати значну кількість похідних характеристик для прогнозування відтоку [16].

Сценарій на основі активності користувача використовується для оцінки рівня залученості до продукту або сервісу. До відповідних характеристик належать різні форми взаємодії користувача з функціональністю системи, наприклад, частота використання окремих функцій, тривалість активності, інтенсивність взаємодії та різноманітність використаних функцій [3, 7].

Сценарій на основі даних служби підтримки передбачає використання інформації про взаємодію клієнта зі службою підтримки: кількість звернень, тематика запитів, час очікування відповіді та рівень задоволеності користувача [2, 3].

Сценарій на основі геопросторових даних використовує інформацію про географічне розташування користувача під час взаємодії із сервісом. До таких даних належать регіон активності, зміна географічного положення та місце використання послуг [3, 7].

Сценарій на основі технічних даних та телеметрії характерний для мобільних застосунків та програмних систем із клієнтською частиною. У цьому випадку фіксуються технічні параметри взаємодії користувача із системою: тип пристрою, версія застосунку, частота виникнення помилок, швидкість завантаження та інші технічні характеристики. Такі дані дозволяють аналізувати вплив технічних проблем на ризик відтоку користувачів [3, 13].

Сценарій на основі соціальних та мережевих даних передбачає аналіз взаємозв'язків між користувачами. У телекомунікаційній сфері CDR-дані дозволяють будувати граф комунікацій між абонентами. Якщо значна частина контактів користувача вже змінила оператора, ризик його власного відтоку також зростає [1, 16].

Описані вище сценарії залучення поведінкових характеристик та їх сфери застосування записані в таблиці 1.3.

Таблиця 1.3 – Сценарії залучення поведінкових характеристик

№	*Сценарій залучення поведінкових характеристик*	*Сфера застосування*
1	На основі журналів подій	SaaS, e-commerce
2	На основі транзакційних даних	Банкінг, фінтех, e-commerce, страхування, retail
3	На основі CDR-даних	Телекомунікації, IoT (інтернет речей)
4	На основі активності користувача	SaaS, стрімінгові сервіси, телекомунікації, e-commerce, EdTech (застосунки у сфері освітніх технологій)
5	На основі даних служби підтримки	SaaS, телекомунікації, банкінг, e-commerce

Кінець табл. 1.3

№	*Сценарій залучення поведінкових характеристик*	*Сфера застосування*
6	На основі геопросторових даних	Логістика, телекомунікації, маркетинг
7	На основі технічних даних та телеметрії	IoT, кібербезпека, хмарні сервіси
8	На основі соціальних та мережевих даних	Соцмережі, маркетплейси, платформи з реферальними механізмами

У цій роботі розглядається телекомунікаційна сфера, тому для цієї задачі актуальними є чотири сценарії залучення поведінкових характеристик. Сценарій на основі агрегованих фінансових/транзакційних даних, на основі аналізу користування сервісами, на основі соціальних та мережевих взаємодій, а також на основі геопросторових даних. Залучення транзакційно-фінансових характеристик дозволяє сформувати метрики економічної поведінки клієнта такі як: тип контракту, регулярні платежі та сукупний обсяг витрат за період обслуговування. Сценарій на основі аналізу використання сервісів відображає ступінь залученості абонентів, що визначається кількістю та типом підключених послуг. Сценарій на основі соціальних та мережевих взаємодій дозволяє оцінити реферальну активність, яка виступає важливим маркером лояльності та наявності соціальних зв’язків користувача всередині продукту. Геопросторовий сценарій передбачає врахування регіональних ознак та координат місця проживання клієнтів для виявлення територіальних закономірностей, що можуть впливати на схильність до відтоку.

1.6 Актуальність та постановка задачі прогнозування відтоку

Тенденція переходу бізнесу до моделей регулярного доходу підвищила важливість вирішення проблеми утримання клієнтів. Також зростає потреба у методах аналізу та прогнозування відтоку користувачів [6].

Раніше більшість підходів базувалася на статистичних методах та логістичній регресії. Однак з розвитком методів машинного навчання з'явились значно потужніші підходи: від ансамблевих методів до рекурентних нейронних мереж, які можуть знаходити певні закономірності в поведінці користувачів. Таке зростання кількості методів створює також необхідність порівняльного аналізу методів. Залишається відкритим питання про вибір оптимального методу для певного бізнес-контексту та даних [9, 10].

Актуальність теми підкреслює й значна кількість досліджень, яка зросла протягом останнього десятиліття. Це підкреслює не тільки практичну затребуваність, а й наукову зацікавленість [1 – 3, 14, 15].

Задача прогнозування відтоку формулюється як задача бінарної класифікації. Нехай X – матриця ознак розміром n × m, де n – кількість користувачів, m – кількість ознак, що характеризують поведінку користувача. Відповідний вектор міток $y \in \{0,1\}^n$, де $y_i = 1$ означає, що і-ий користувач здійснив відтік протягом певного часового періоду. Відповідно, $y_i = 0$ означає, що і-ий користувач продовжив користуватися продуктом. Мета побудови моделі полягає у знаходженні функції $f: x \to [0,1]$, яка повертає оцінку ймовірності відтоку $P(churn \mid x)$. На основі цієї оцінки та визначеного порогу класифікації τ приймається рішення: якщо $P > \tau$, користувач відноситься до групи ризику і потребує заходів утримання [9].

Модель оцінює умовну ймовірність відтоку клієнта $P(y = 1 \mid x)$, де x – вектор ознак певного користувача. Відповідно до підходу, описаного в [9],

задача навчання з учителем зводиться до мінімізації функції втрат, що вимірює різницю між прогнозними та реальними значеннями. Для бінарної класифікації використовується логістична функція втрат:

$$L = -\frac{1}{n} * \sum_{i=1} [y_i * \log(\widehat{y_i}) + (1 - y_i) * \log(1 - \widehat{y_i})],$$

де n – кількість спостережень, y_i – реальне значення і-ого клієнта, $\widehat{y_i} = f(x_i)$ – спрогнозована моделлю ймовірність відтоку. Мінімізація цього виразу спонукає модель повертати значення, близькі до 1 для клієнтів з відтоком і близькі до 0 для лояльних клієнтів [9].

Для побудови моделей прогнозування відтоку використовуються історичні дані про поведінку користувачів. Поведінкові дані можуть включати різні характеристики такі, як: частота використання сервісу, тривалість сесій, рівень оплати користувача, взаємодія з функціоналом продукту та інші поведінкові ознаки. Такі дані дозволяють описати характерні особливості поведінки користувачів та ідентифікувати тих, хто залишається, та тих, хто припиняє використання продукту.

Головна особливість прогнозування відтоку – це дисбаланс класів у навчальній вибірці. У більшості реальних даних частка клієнтів, що здійснили відтік, є значно меншою від частки клієнтів, що залишились. Це означає, що навіть модель, яка завжди передбачає відсутність відтоку, матиме високу точність прогнозу, але при цьому низьку ефективність для бізнесу. Тому для цієї задачі критично важливим є коректний вибір метрик оцінювання – precision, recall, F1-score, ROC-AUC (площа під кривою робочої характеристики приймача) та PR-AUC (площа під кривою точності та повноти), які враховують якість класифікації для обох класів [4].

Постановка задачі складається з таких етапів:

1) вибір сфери застосування;

2) визначення поняття відтоку користувачів на основі попереднього етапу;
3) збір та обробка ознак;
4) вибір методів, які будуть вирішувати задачу бінарної класифікації;
5) обрання метрик для оцінювання моделей;
6) побудова моделі машинного навчання;
7) оцінка моделей за допомогою обраних метрик;
8) формування висновків на основі отриманих метрик.

Блок-схема цього алгоритму зображена на рисунку 1.4.

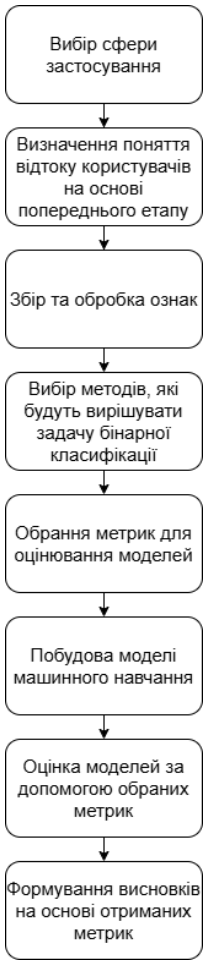


Рисунок 1.4 – Алгоритм постановки задачі прогнозування відтоку користувачів

1.7 Сучасні технічні інструменти та технології для вирішення задачі прогнозування відтоку

Для розробки системи прогнозування відтоку використовується широкий стек технічних засобів, що включає мови програмування, бібліотеки попередньої обробки даних та машинного навчання та інструменти візуалізації. Для написання програмного коду використовується мова програмування Python, для роботи з даними – Pandas та NumPy, а для побудови, навчання та оцінювання моделей – Scikit-learn. Методи градієнтного бустингу реалізуються за допомогою бібліотек XGBoost та LightGBM. Додатково для роботи з незбалансованими даними використовується imbalanced-learn. Візуалізація результатів виконується за допомогою Matplotlib та Seaborn. Середовище розробки – Visual Studio Code, що забезпечує зручну роботу та інтеграцію необхідних технічних інструментів.

Таблиця 1.4 – Технічні інструменти, що застосовуються для прогнозування відтоку користувачів

№	*Назва інструменту*	*Призначення*
1	Python	Мова програмування, яка застосовується для аналізу та попередньої обробки даних, побудови моделей машинного навчання та їх оцінки
2	Pandas	Забезпечує завантаження, фільтрацію, агрегацію та трансформацію табличних даних
3	NumPy	Реалізує обчислення та роботу з числовими масивами
4	Scikit-learn	Бібліотека, яка використовується для попередньої обробки даних, навчання деяких моделей та їх оцінки
5	XGBoost та LightGBM	Бібліотеки для реалізації градієнтного бустингу

Кінець таблиці 1.4

№	*Назва інструменту*	*Призначення*
6	Matplotlib та Seaborn	Використовується для побудови візуалізацій, таких як матриці плутанини, ROC-криві, PR-криві та графіки важливості ознак
7	imbalanced-learn	Бібліотека для роботи з незбалансованими даними
8	Visual Studio Code	Середовище для розробки та запуску програмного коду

Висновки до розділу 1

Розділ 1 присвячено аналізу предметної області прогнозування відтоку користувачів, характеристиці існуючих підходів у різних галузях, опису видів ознак, що використовуються для побудови моделей, та формулюванню постановки задачі дослідження.

Встановлено, що відтік є багатофакторним явищем, яке проявляється по-різному залежно від специфіки бізнесу: від явного скасування підписки у SaaS до поступового зменшення активності у банківській сфері. Виділено п'ять основних типів відтоку – добровільний, вимушений, випадковий, усвідомлений та прихований. Кожен з них має власну природу і вимагає відповідних стратегій виявлення. З фінансової точки зору відтік безпосередньо впливає на ключові бізнес-метрики: CAC, CLTV та MRR. Проактивне виявлення клієнтів із підвищеним ризиком відтоку дозволяє переходити від реактивних до превентивних стратегій утримання, що є економічно більш ефективним підходом.

Аналіз сучасних підходів у SaaS, телекомунікаціях, банківській сфері та e-commerce показав, що попри спільну математичну природу задачі, вибір методів і тип задіяних даних суттєво відрізняються між секторами. Ансамблеві методи (XGBoost, LightGBM, Random Forest) стабільно

демонструють найвищі результати на табличних даних, тоді як нейромережеві підходи (LSTM, GRU, RNN, CNN) є доцільними при роботі з послідовними та часовими даними.

Характеристика формування ознак підтвердила, що якість вхідних даних і їх інформативність мають не менше значення, ніж вибір алгоритму. Серед основних категорій ознак виділено демографічні, поведінкові, RFM, часові та ознаки підтримки. Було визначено, що поєднання ознак, а не їх окреме використання, дає найкращі результати. Найвища предиктивна цінність притаманна часовим і поведінковим ознакам, тоді як демографічні є корисними лише як допоміжний компонент. Для цієї роботи обрано сценарій залучення агрегованих фінансових, геопросторових, соціально-мережевих даних, а також даних активності користувача, що відповідають специфіці телекомунікаційного набору даних.

Постановка задачі сформульована як задача бінарної класифікації з дисбалансом класів. Визначено алгоритм із восьми послідовних етапів, що охоплює вибір сфери, визначення поняття відтоку, збір та обробку ознак, вибір методів, обрання метрик, побудову моделі та формування висновків. Для технічної реалізації обрано стек на основі Python, Scikit-learn, XGBoost, LightGBM, Matplotlib, Seaborn та imbalanced-learn.

РОЗДІЛ 2 ВИБІР МЕТОДІВ МАШИННОГО НАВЧАННЯ ДЛЯ ПРОГНОЗУВАННЯ ВІДТОКУ КОРИСТУВАЧІВ

2.1 Загальна класифікація методів для розв’язання задачі прогнозування відтоку

Методи машинного навчання, які використовуються у роботі для розв’язання задачі прогнозування відтоку користувачів, можна розділити на три групи залежно від їх ролі у цій задачі: методи попереднього аналізу та підготовки даних, методи машинного навчання для побудови прогностичної моделі та методи оцінювання моделей. Роль кожної групи методів описано у таблиці 2.1.

Таблиця 2.1 – Класифікація методів для прогнозу відтоку користувачів

№	*Група*	*Роль*
1	Методи попереднього аналізу та підготовки даних	Вони застосовуються до навчання моделі і забезпечують якість та придатність вхідних даних: усувають пропуски, приводять ознаки до порівнянного масштабу, перетворюють категоріальні змінні у числове представлення та формують нові інформативні ознаки.
2	Методи машинного навчання для побудови прогностичної моделі	Це власне алгоритми, що вирішують задачу бінарної класифікації і навчаються на підготовлених даних і будують функцію відображення вхідних ознак у ймовірність відтоку.
3	Методи оцінювання моделей	Використовуються, щоб кількісно порівняти якість різних моделей і вибрати найкращу відповідно до обраних метрик.

2.2 Методи попереднього аналізу даних

До методів попереднього аналізу даних можна віднести такі методи: обробка пропущених значень, кодування категоріальних змінних, масштабування ознак, формування ознак (feature engineering), Principal Component Analysis (для зменшення розмірності ознак), Synthetic Minority Oversampling Technique (SMOTE) – метод синтетичного збільшення міноритарного класу (для роботи з дисбалансом класів).

2.2.1 Обробка пропущених значень та кодування категоріальних змінних

Реальні набори даних майже завжди містять пропущені значення, які виникають через технічні збої у системах збору подій, нові акаунти з короткою історією або відсутність певних взаємодій у частини користувачів. Для числових ознак найпоширенішими підходами є заміна пропущених значень середнім, медіаною або модою:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i,$$

де $\bar{x}$ – обчислене значення, x_i – наявне значення ознаки, n – їх кількість.

Медіана є стійкішою до викидів і перевагу їй надають при асиметричних розподілах. Альтернативою є множинне імпутування, де пропущене значення замінюється передбаченням моделі, навченої на наявних ознаках. Для поведінкових даних нульове значення часто є змістовним (клієнт не

здійснював жодної дії), тому важливо розрізняти «справжній нуль» і «відсутнє значення» ще на етапі проектування ознак.

Більшість алгоритмів машинного навчання приймають лише числові вхідні дані, тому категоріальні змінні (тип підписки, канал залучення, регіон) потребують перетворення. One-Hot Encoding перетворює категорію з K унікальних значень на K бінарних ознак:

$$x_j = 1, \text{ якщо категорія = j, інакше } x_j = 0,\ j \in \{1, \dots, K\}.$$

Цей підхід підходить для номінальних змінних із невеликою кількістю категорій. При великій кількості категорій застосовується Target Encoding, де категорія замінюється середнім значенням цільової змінної по цій категорії:

$$enc(c_j) = \frac{1}{|S_j|} \sum_{i \in S_j} y_i,$$

де S_j – множина спостережень із категорією c_j, y_i – значення цільової змінної.

Однак цей підхід потребує обережності: при малих вибірках по категорії виникає перенавчання, яке усувається згладжуванням або крос-валідацією при обчисленні.

2.2.2 Масштабування ознак

Алгоритми, що засновані на відстанях або градієнтному спуску (k-NN, SVM, логістична регресія, нейронні мережі) є чутливими до масштабу ознак. Тому перед навчанням моделей часто виконується масштабування даних.

Для масштабування ознак виділяють 3 методи: Standard Scaler, MinMax Scaler та Robust Scaler.

StandardScaler (Z-score normalization) приводить кожну ознаку до нульового середнього та одиничного стандартного відхилення:

$$z = \frac{x - \mu}{\sigma},$$

де μ – середнє значення ознаки по навчальній вибірці, σ – стандартне відхилення.

Цей метод добре працює з алгоритмами, що припускають, що дані центровані, такими як лінійна регресія, логістична регресія, SVM. Він залежить від викидів менше, ніж MinMax Scaler. Його недоліком є неоптимальні результати при даних, що не близькі до нормального розподілу.

Min-Max нормалізація (MinMaxScaler) масштабує ознаку до діапазону [0,1]:

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

де x_{min} і x_{max} – мінімальне та максимальне значення ознаки у навчальній вибірці.

Цей метод може давати хороший результат навіть якщо дані не відповідають нормальному розподілу. Він добре працює з алгоритмами, де важливий абсолютний масштаб, наприклад, k-NN або нейронні мережі. Але він на відміну від Standard Scaler є більш чутливим до викидів: навіть одне екстремальне значення може спотворити масштабування. Також він є чутливим до діапазонів навчальної та тестової вибірки: якщо їх діапазони відрізняються, то результат може бути невідповідним.

Також важливо вказати, що параметри масштабування ($\mu, \sigma, x_{min}, x_{max}$, медіана та квартилі) обчислюються виключно на навчальній вибірці та потім застосовуються до тестової, щоб уникнути витоку інформації.

RobustScaler використовується у випадках, коли дані містять викиди. Масштабування виконується на основі медіани та міжквартильного розмаху:

$$x' = \frac{x - median(x)}{IQR} = \frac{x - median(x)}{Q_3 - Q_1},$$

де IQR – міжквартильний розмах, Q_1 та Q_3 – перший і третій квартилі відповідно.

Цей метод є стійким до викидів, оскільки використовує медіану та міжквартильний розмах замість середнього значення та стандартного відхилення. Його доцільно застосовувати у випадках, коли розподіл даних є асиметричним або містить значну кількість викидів. Недоліком методу є те, що він не приводить дані до фіксованого діапазону, тому для деяких алгоритмів масштаб ознак все ще може залишатися неоднорідним. Також при відсутності викидів Robust Scaler часто не дає помітних переваг порівняно зі Standard Scaler [27].

2.2.3 Формування ознак

Для поведінкових даних сирі агрегати часто є недостатніми, корисніші ознаки формуються шляхом перетворень. Логарифмічне перетворення застосовується для ознак із сильно асиметричним розподілом:

$$x' = \log(1 + x),$$

де додавання 1 забезпечує визначеність при $x = 0$.

Трендові ознаки обчислюються як різниця або відношення між значеннями за суміжні вікна:

$$trend = x_{30} - x_7$$

або

$$trend\ ratio = \frac{x_{30}}{x_{90} + \varepsilon},$$

де x_7, x_{30}, x_{90} – значення ознаки за останні 7, 30 та 90 днів відповідно, ε – мала константа для уникнення ділення на нуль.

Ковзне середнє (EWMA) з коефіцієнтом згасання $\alpha \in (0, 1)$:

$$EWMA_t = \alpha x_t + (1 - \alpha) EWMA_{t-1},$$

де x_t – значення ознаки у момент t, α визначає, наскільки швидко зменшується вплив старих спостережень.

RFM-ознаки використовуються для узагальнення поведінки користувача на основі його історичної активності та є одним із найпоширеніших підходів до побудови агрегованих поведінкових характеристик у задачах прогнозування відтоку.

Recency (R) визначає час, що минув від останньої активності користувача:

$$R = t_{now} - t_{last},$$

де t_{now} – поточний момент часу, t_{last} – час останньої взаємодії користувача з системою.

Frequency (F) відображає кількість активностей користувача за фіксований період часу:

$$F = \sum_{i=1}^{N} a_i,$$

де a_i – індикатор окремої події активності користувача, N – загальна кількість подій у заданому часовому вікні.

Monetary (M) характеризує сумарну економічну цінність користувача:

$$M = \sum_{i=1}^{N} v_i,$$

де v_i – вартість і-ої транзакції або платної взаємодії.

RFM-підхід дозволяє трансформувати сирі поведінкові дані у компактний набір інтерпретованих ознак, які ефективно використовуються у моделях бінарної класифікації, зокрема для задач прогнозування відтоку користувачів.

2.2.4 Principal component analysis ma Synthetic Minority Over-sampling Technique

При великій кількості ознак застосовується метод головних компонент для зниження розмірності та усунення мультиколінеарності перед навчанням класифікатора, але знижує інтерпретованість моделі. Нові ознаки є ортогональними лінійними комбінаціями вихідник ознак, що максимізують дисперсію даних у просторі ознак:

$$Z = XW, W = [w_1, w_2, w_3, \dots, w_n],$$

де $X \in R^{n*m}$ – центрована матриця ознак, $W \in R^{m*k}$ – матриця перших k власних векторів матриці коваріацій $C = \frac{1}{n} * X^T X$, $k < m$ – кількість компонент, що зберігаються.

Вибір k визначається за часткою поясненої дисерсії:

$$EVR(k) = \frac{\sum_{j=1}^{k} \lambda_j}{\sum_{j=1}^{m} \lambda_j},$$

де λ_j – j-те власне значення матриці коваріацій у порядку спадання.

Зазвичай обирають k таким, щоб $EVR(k) \geq 0{,}95$, тобто зберігалося не менше 95% загальної дисперсії [9, 22, 23].

Оскільки частка відтоку у більшості наборів даних становить від 2% до 30%, то навчальна вибірка зазвичай містить дисбаланс класів. Метод Synthetic Minority Oversampling Technique або SMOTE дозволяє ефективніше працювати з дисбалансом класів: він генерує синтетичні приклади меншого класу шляхом інтерполяції між наявними:

$$x_{new} = x_i + \lambda(x_j - x_i),$$

де x_i і x_j – два сусідніх приклади меншого класу, $\lambda \in [0, 1]$ – випадковий коефіцієнт.

Альтернативою є зважування класів при навчанні: алгоритм отримує параметр:

$$class_{weight} = \frac{n}{2n_k},$$

де n – загальна кількість спостережень, n_k – кількість спостережень класу k, що компенсує дисбаланс без зміни розміру вибірки.

2.3 Методи машинного навчання для побудови прогностичної моделі

В цьому підрозділі будуть розглянуті всі методи машинного навчання, які були згадані у розділі 1.3, де описуються алгоритми, які розглядаються у сучасних дослідженнях.

2.3.1 Лінійні, ймовірнісні та метричні алгоритми

Логістична регресія є базовим методом бінарної класифікації і часто використовується як baseline-модель. Незважаючи на назву, це класифікатор, а не регресор: він моделює ймовірність належності до класу через сигмоїдну функцію:

$$P(y = 1 \mid x) = \sigma(w^T x + b) = \frac{1}{1 + e^{-(w^T x + b)}},$$

де $x \in R^m$ – вектор ознак клієнта, $w \in R^m$ – вектор вагових коефіцієнтів, b – вільний член зміщення (bias), σ – сигмоїдна функція, що відображає довільне дійсне число у діапазоні від 0 до 1.

Навчання виконується шляхом мінімізації бінарної крос-ентропії методом градієнтного спуску або його варіантами (SGD – стохастичний градієнтний спуск, Adam). Ключова перевага методу – інтерпретованість:

знак коефіцієнта w_j вказує напрямок впливу j-ої ознаки, а його величина – силу впливу. Регуляризація L1 (Lasso) або L2 (Ridge) додається до функції втрат для запобігання перенавчанню: при L1 частина коефіцієнтів обнуляється, що виконує неявний відбір ознак. Основне обмеження – припущення про лінійну роздільність класів у просторі ознак, яке рідко виконується на реальних поведінкових даних [9, 23].

Naive Bayes застосовує теорему Баєса з припущенням про умовну незалежність ознак при заданому класі:

$$P(y \mid x) = \frac{P(y) * P(x|y)}{P(x)},$$

що еквівалентно такому запису:

$$P(y|x) \propto P(y) \prod_{j=1}^{m} P(x_j|y),$$

де P(y) – апріорна ймовірність класу y, $P(x_j|y)$ – умовна ймовірність j-ої ознаки при заданому класі, m – кількість ознак.

Клас визначається як:

$$\hat{y} = argmax_y P(y) * \prod_{j=1}^{m} P(x_j|y).$$

Для числових ознак використовується Gaussian Naive Bayes, де $P(x_j|y)$ моделюється нормальним розподілом. Перевага методу – надзвичайна швидкість навчання та передбачення, стійкість до нерелевантних ознак при великій кількості спостережень. Головне обмеження – нереалістичне припущення про незалежність ознак, яке майже ніколи не виконується на

поведінкових даних (наприклад, кількість сесій і тривалість сесій очевидно корелюють) [3, 9, 20, 23].

SVM знаходить гіперплощину максимального відступу між класами. Для нелінійно роздільних даних застосовується ядровий метод, що неявно проектує ознаки у простір вищої розмірності. Задача оптимізації для м'якого відступу:

$$\min(w, b, \xi) = \frac{1}{2}\|w\|^2 + C * \sum_{i=1} \xi_i,$$

за умови

$$\begin{cases} y_i(w^T x_i + b) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{cases},$$

для всіх i, де w – нормаль до роздільної гіперплощини, b – зміщення, $C > 0$ – параметр регуляризації, що балансує між максимізацією відступу та мінімізацією помилок класифікації, $\xi_i \geq 0$ – змінні, що допускають порушення відступу.

Вибір ядра (RBF, polynomial, linear) суттєво впливає на результат і потребує крос-валідації. Основне обмеження SVM у задачах прогнозування відтоку це квадратична складність навчання $O(n^2)$, що робить його непрактичним на датасетах із мільйонами клієнтів. На менших вибірках SVM може бути конкурентоспроможним, особливо при правильному підборі ядра та масштабуванні ознак. Тож у результаті цей метод поступається методу градієнтного бустингу у швидкості навчання [9, 23].

Метод k-NN відносить клієнта до класу, що переважає серед k найближчих сусідів у просторі ознак. Відстань між клієнтами вимірюється за евклідовою відстанню:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{m} (x_{ik} - x_{jk})^2},$$

де x_{ik} і x_{jk} – значення k-ої ознаки для клієнтів і та j, m – кількість ознак.

Прогноз для нового клієнта x:

$$\hat{P}(y = 1 \mid x) = \frac{1}{k} * \sum_{i \in N_k(x)} y_i,$$

де $N_k(x)$ – множина k найближчих сусідів.

Метод є непараметричним, тобто він не будує явної моделі, а зберігає весь навчальний набір даних. Це є одночасно перевагою (немає припущень про форму розподілу) і недоліком: швидкість передбачення лінійно залежить від розміру навчальної вибірки O(n), що є критичним для великих наборів даних. Метод k-NN також чутливий до масштабу ознак і нерелевантних ознак, що знижує якість на поведінкових даних із великою розмірністю [9, 23].

2.3.2 Метод дерева рішень та ансамблеві методи на його основі

Дерево рішень виконує рекурентне розбиття простору ознак за критерієм зменшення нечистоти вузла. Найпоширенішим критерієм є індекс Джині [9, 20]:

$$Gini(t) = 1 - \sum_{k=1} p(k|t)^2,$$

де p(k|t) – частка об’єктів класу k у вузлі t, підсумовування ведеться по всіх класах.

Альтернативним критерієм є ентропія:

$$H(t) = -\sum_{k} p(k|t) * log_2 p(k|t).$$

Розбиття у кожному вузлі вибирається таким чином, щоб максимально зменшити зважену суму нечистоти дочірніх вузлів.

Дерева рішень є повністю інтерпретованою моделлю, оскільки кожне рішення можна відобразити у вигляді набору логічних правил (if-then). Проте окреме дерево схильне до перенавчання при великій глибині. Саме тому дерева рішень рідко використовуються самостійно у задачах прогнозування відтоку. Натомість вони є базовими елементами ансамблевих методів, таких як Random Forest та градієнтний бустинг [9, 23].

Random Forest є ансамблевим методом, що будує множину незалежних дерев рішень і усереднює їх прогнози:

$$\hat{P}RF(y = 1 \mid x) = \frac{1}{T} * \sum_{t=1}^{T} \widehat{P_t}(y = 1 \mid x),$$

де T – кількість дерев, $\widehat{P_t}(y = 1 \mid x)$ – ймовірність відтоку, передбачена t-им деревом.

Кожне дерево навчається на бутстреп-вибірці (випадкова вибірка з поверненням розміром n), а в кожному вузлі розглядається випадкова підмножина ознак розміром $\sqrt{m}$, де m – загальна кількість ознак. Така процедура вводить стохастичність як за об’єктами, так і за ознаками, що зменшує кореляцію між деревами та дозволяє знизити дисперсію ансамблю порівняно з окремими деревами рішень. На відміну від одного дерева рішень,

Random Forest є значно стійкішим до перенавчання і добре працює без тонкого налаштування гіперпараметрів. Додатково Random Forest дозволяє оцінювати важливість ознак на основі середнього зниження неоднорідності, що робить метод придатним також для аналізу факторів відтоку користувачів [10, 23].

XGBoost та LightGBM реалізують метод градієнтного бустингу, де ансамбль дерев рішень будується послідовно: кожне нове дерево $h_t(x)$ навчається апроксимувати негативний градієнт функції втрат поточного ансамблю:

$$F_t(x) = F_{t-1}(x) + \eta * h_t(x),$$

де $F_t(x)$ – прогноз на кроці t, $F_{t-1}(x)$ – прогноз попереднього ансамблю, η – швидкість навчання (learning rate), $h_t(x)$ – нове дерево рішень, навчене на основі градієнтів функцій втрат.

XGBoost формулює задачу навчання як мінімізацію регуляризованої функції втрат:

$$L = \sum_i l(y_i, \hat{y_i}) + \sum_t \Omega(f_t),$$

де $l(y_i, \hat{y_i})$ – функція втрат, $\Omega(f_t)$ – функція регуляризації складності дерева, f_t – t-те дерево в ансамблі.

Оптимізація виконується за схемою градієнтного бустингу, де на кожній ітерації функція втрат апроксимується розкладом Тейлора другого порядку:

$$L^{(t)} \approx \sum_i \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t),$$

де g_i – перша похідна функції втрат (градієнт), h_i – друга похідна (Гессіан).

LightGBM розв’язує ту ж задачу оптимізації, але використовує процедури зменшення обчислювальної складності, зокрема підвибірку об’єктів із великими значеннями градієнтів та групування взаємовиключних ознак для зменшення розмірності простору ознак. Обидва методи застосовують L1- та L2-регуляризацію для контролю складності моделі, а також враховують дисбаланс класів через ваги спостережень. У задачах прогнозування відтоку на табличних даних ці методи демонструють високу здатність до узагальнення порівняно з класичними моделями машинного навчання [9, 10, 23].

AdaBoost будує ансамбль слабких класифікаторів послідовно, збільшуючи вагу помилково класифікованих прикладів після кожної ітерації:

$$F(x) = sign\left(\sum_{t=1}^{T} \alpha_t * h_t(x)\right),$$

де $h_t(x)$ – t-ий слабкий класифікатор, α_t – його вага у ансамблі:

$$\alpha_t = \frac{1}{2} * \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right),$$

де ε_t – зважена помилка класифікації t-ого класифікатора на навченій вибірці.

Чим нижча помилка, тим більший вплив класифікатора. Після кожної ітерації ваги помилково класифікованих прикладів збільшуються в e^{α_t} разів. Механізм ваг змушує кожен наступний класифікатор зосереджуватись на найважчих прикладах. AdaBoost чутливий до викидів і зашумлених міток, оскільки важкі приклади отримують надмірно великі ваги, що може призводити до перенавчання [9, 23].

2.3.3 Методи глибокого навчання

RNN – рекурентна нейронна мережа, яка обробляє послідовності подій, передаючи прихований стан між кроками. На кожному кроці t:

$$h^{(t)} = \tanh\left(Wh^{(t-1)} + Ux^{(t)} + b\right),$$

де $x^{(t)}$ – вхідний вектор у момент часу t, $h^{(t-1)}$ – стан прихованого шару на попередньому кроці, $h^{(t)}$ – стан прихованого шару, W – матриця ваг для рекурентних зв'язків, U – матриця ваг для вхідних даних, b – зміщення, tanh – функція активації (гіперболічний тангенс).

RNN теоретично здатна враховувати довільно довгі залежності у послідовності, проте на практиці стикається з проблемою затухання градієнтів. При навчанні через алгоритм зворотного поширення помилки в часі градієнти експоненційно зменшуються для ранніх кроків послідовності, що унеможливлює навчання довгострокових залежностей. Саме ця проблема мотивувала розробку LSTM [9, 25].

LSTM розширює RNN і вирішує проблему затухання градієнтів через систему воріт і окрему клітинку пам'яті c_t, що передає інформацію через довгі послідовності без значного затухання:

$$f_i^{(t)} = \sigma\left(b_i^f + \sum_j U_{i,j}^f x_j^{(t)} + \sum_j W_{i,j}^f h_j^{(t-1)}\right),$$

$$g_i^{(t)} = \sigma\left(b_i^g + \sum_j U_{i,j}^g x_j^{(t)} + \sum_j W_{i,j}^g h_j^{(t-1)}\right),$$

$$q_i^{(t)} = \sigma\left(b_i^o + \sum_j U_{i,j}^o x_j^{(t)} + \sum_j W_{i,j}^o h_j^{(t-1)}\right),$$

$$s_i^{(t)} = f_i^{(t)} s_i^{(t-1)} + g_i^{(t)} \sigma\left(b_i + \sum_j U_{i,j} x_j^{(t)} + \sum_j W_{i,j} h_j^{(t-1)}\right),$$

$$h_i^{(t)} = \tanh\left(s_i^{(t)}\right) q_i^{(t)},$$

де $x^{(t)}$ – вхідні дані на поточному кроці, $h^{(t)}$– прихований стан на поточному кроці, $s^{(t)}$– внутрішній стан комірки, W – матриця ваг для попереднього прихованого стану, U – матриця ваг для поточного входу, $f^{(t)}, g^{(t)}, q^{(t)}$ – значення воріт, b – вектор зсуву, tanh – гіперболічний тангенс, σ – сигмоїдна функція.

Ці 5 формул відповідно називаються так: ворота забування, ворота входу, ворота виходу, оновлення стану комірки, прихований стан. Ворота забування контролюють, яка частина попереднього стану зберігається. Ворота вводу визначають, яка нова інформація додається до стану комірки. Ворота виводу визначають, що передається як прихований стан на наступний крок і як вихід мережі. Завдяки клітинці пам'яті LSTM здатна запам'ятовувати події, що відбулись десятки або сотні кроків тому, що є критичним для виявлення поступової деградації активності клієнта [19, 25].

GRU – це спрощена версія LSTM. Цей алгоритм об'єднує ворота забування та вводу в одні ворота оновлення і не має окремої клітинки пам'яті. Тому цей метод має лише двоє воріт замість трьох, що зменшує кількість параметрів і прискорює навчання:

$$u_i^{(t)} = \sigma\left(b_i^u + \sum_j U_{i,j}^u x_j^{(t)} + \sum_j W_{i,j}^u h_j^{(t)}\right),$$

$$r_i^{(t)} = \sigma\left(b_i^r + \sum_j U_{i,j}^r x_j^{(t)} + \sum_j W_{i,j}^r h_j^{(t)}\right),$$

$$h_i^{(t)} = u_i^{(t-1)} h_i^{(t-1)} + (1 - u_i^{(t-1)})\sigma\left(b_i + \sum_j U_{i,j} x_j^{(t-1)} + \sum_j W_{i,j} r_j^{(t-1)} h_j^{(t-1)}\right),$$

де $x^{(t)}$ – вхідний веткор ознак, $h^{(t)}$– прихований стан попереднього кроку, u – значення воріт оновлення, r – значення воріт скидання, W – матриця ваг для рекурентних зв'язків, U – матриця ваг для поточного входу, σ – сигмоїдна функція, tanh – гіперболічний тангенс, b – вектор зміщення.

Формули для методу GRU мають такі відповідні назви: ворота оновлення, ворота скидання та оновлений прихований стан. Ворота оновлення визначають, наскільки попередній стан зберігається і наскільки замінюється новим кандидатом. Ворота скидання контролюють, яка частина використовується при обчисленні кандидата. GRU дає точність близьку до LSTM для більшості задач, але при цьому має меншу обчислювальну вартість [25].

CNN у застосуванні до часових рядів використовує операцію згортки, яка в практичних реалізаціях нейронних мереж відповідає крос-кореляції:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n) K(m+n),$$

де $S(i,j)$ – значення карти ознак у позиції (i, j), I – вхідна матриця ознак, K – ядро згортки, m, n – індекси елементів ядра.

Отримана карта ознак далі обробляється функцією активації ReLU, що застосовується поелементно:

$$ReLU(z) = \max(0, z).$$

CNN дає змогу ефективно виявляти короткострокові закономірності, але поступається GRU та LSTM у моделюванні довгострокових залежностей. На практиці CNN часто поєднують із LSTM у гібридних архітектурах CNN-LSTM, де CNN виявляє локальні патерни, а LSTM – довгострокові залежності між ними [9, 12, 25].

2.4 Методи оцінювання моделей

Методами оцінки моделей є розрахунок певних обраних статистичних метрик. Вибір правильних метрик є важливою частиною побудови моделей прогнозування відтоку. Зазвичай однією з найголовніших метрик у задачах прогнозування виступає метрика точності (Accuracy), однак для цієї задачі вона є непридатною через дисбаланс класів [4].

Матриця плутанини (confusion matrix) є базовим інструментом для аналізу якості бінарної класифікації. Вона дозволяє виокремити чотири типи результатів: істинно позитивні (TP, true positive – випадок, коли коректно ідентифікований відтік), хибно позитивні (FP, false positive – випадок, коли помилково було визначено відтік), хибно негативні (FN, false negative – випадок, коли помилково визначено як не відтік) та істинно негативні (TN, true negative – випадок, коли коректно ідентифіковано відсутність відтоку) [4, 9].

2.4.1 Accuracy, Precision, Recall та F1-score

Accuracy розраховується як:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}.$$

Однак використання лише метрики Accuracy для оцінки прогнозу може бути небезпечним. Якщо модель буде завжди передбачати відсутність відтоку, то точність може бути достатньо високою, бо реальний рівень відтоку невеликий. У зв’язку з цим, для оцінки моделей використовують інші метрики, такі як: Precision, Recall, F1-score, ROC-AUC, PR-AUC.

Метрики Precision та Recall є взаємопов'язаними і характеризують якість класифікації для позитивного класу. Precision вимірює частку правильно передбачених відтоків серед усіх позначених моделлю як відтік. Recall визначає, яку частку реальних відтоків модель виявила. Для задачі утримання клієнтів пріоритетним є Recall, оскільки пропуск клієнта з ризиком відтоку є більш критичним збитком. Ці метрики розраховуються таким чином:

$$Precision = \frac{TP}{TP + FP},$$

$$Recall = \frac{TP}{TP + FN}.$$

Між Precision та Recall є певний зв’язок: підвищення порогу класифікації τ збільшує Precision, але зменшує Recall, і навпаки. Оптимальний баланс залежить від вартості різних типів помилок у кожному конкретному випадку [3, 9, 20].

F1-score є чимось середнім між Precision та Recall і використовується як метрика якості класифікатора для незбалансованих класів. F-beta score дозволяє регулювати відносну важливість Precision та Recall залежно від пріоритетів. F1-score розраховується як:

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} = \frac{2TP}{2TP + FN + FP}.$$

Fβ-score надає вагу β^2 параметру Recall відносно Precision:

$$F\beta = \frac{(1 + \beta^2) * Precision * Recall}{\beta^2 * Precision + Recall},$$

де β (β > 0) – параметр балансу.

При $\beta = 1$ буде отримано F1-score, при $\beta = 2$ Recall має вдвічі більшу вагу, ніж Precision. Це є важливим, коли пропуск відтоку є вдвічі дорожчим за хибну ідентифікацію [3, 4, 9, 20].

2.4.2 ROC-AUC та PR-AUC

ROC-AUC є однією з найбільш поширених і згадуваних метрик у задачах прогнозування відтоку. Вона характеризує здатність моделі розрізняти користувачів із ризиком відтоку та користувачів без такого ризику незалежно від порогу класифікації. Значення ROC-AUC = 0,5 відповідає випадковій класифікації, тоді як значення близьке до 1 свідчить про високу розрізняльну здатність моделі [1, 4, 9].

ROC-крива будується шляхом зміни порогу класифікації τ від 0 до 1 і нанесення відповідних пар FPR і TPR (False Positive Rate і True Positive Rate):

$$TPR(\tau) = \frac{TP(\tau)}{TP(\tau) + FN(\tau)},$$

$$FPR(\tau) = \frac{FP(\tau)}{FP(\tau) + TN(\tau)},$$

де TP(τ), FP(τ), FN(τ), TN(τ) – елементи матриці плутанини при порозі τ.

ROC-AUC обчислюється як площа під ROC-кривою:

$$ROC\text{-}AUC = \int_0^1 TPR(FPR)d(FPR),$$

де TPR(FPR) – значення True Positive Rate при відповідному рівні False Positive Rate, що змінюється зі зміною порогу τ [9, 26].

PR-AUC вважається більш інформативною за умов сильного дисбалансу класів, тому PR-AUC є більш репрезентативною метрикою для умов, де частка відтоку становить менше 10% [9, 12, 26].

PR-AUC обчислюється за такою формулою, як площа під кривою Precision-Recall:

$$PR\text{-}AUC = \int_0^1 Precision(Recall)d(Recall),$$

де Precision(Recall) – значення Precision при задному рівні Recall, що змінюється зі зміною порогу τ. Базовою лінією для PR-AUC є частка позитивного класу у вибірці:

$$\pi = \frac{|\{i: y_i = 1\}|}{n},$$

на відміну від 0,5 для ROC-AUC.

При сильному дисбалансі класів PR-AUC точніше відображає реальну якість моделі щодо меншого класу [9, 20, 26].

Всі метрики, їх формули, переваги, обмеження та використання наведені в таблиці 2.2.

Таблиця 2.2 – Порівняння метрик оцінювання моделей прогнозування відтоку

Метрика	*Формула*	*Перевага*	*Обмеження*	*Використання*
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$	Проста інтерпретація	Непридатна при дисбалансі	Лише як базовий орієнтир
Precision	$\frac{TP}{TP + FP}$	Оцінює хибні тривоги	Ігнорує пропущений відтік	Обмежені ресурси утримання
Recall	$\frac{TP}{TP + FN}$	Виявляє всі відтоки клієнтів	Збільшує хибні ідентифікації	Для продукту з дорогими/важливими клієнтами
F1-score	$\frac{2TP}{2TP + FN + FP}$	Баланс між Precision та Recall	Рівна вага Precision і Recall	Загальна оцінка якості
ROC-AUC	$\int_0^1 TPR(FPR)d(FPR)$	Незалежна від порогу класифікації	Має меншу ефективність при сильному дисбалансі класів	Порівняння моделей
PR-AUC	$\int_0^1 Precision(Recall)d($	Чутлива до дисбалансу	Складніша інтерпретація	За наявності рідкісних класів

Висновки до розділу 2

Розділ 2 присвячено математичному опису методів машинного навчання, що застосовуються для вирішення задачі прогнозування відтоку користувачів. Методи систематизовано за трьома групами: методи попереднього аналізу та підготовки даних, методи побудови прогностичної моделі та методи оцінювання моделей.

До методів попереднього аналізу даних віднесено обробку пропущених значень (заміна середнім, медіаною або модою), кодування категоріальних змінних (One-Hot Encoding та Target Encoding), три підходи до масштабування (StandardScaler, MinMaxScaler, RobustScaler), формування ознак (логарифмічне перетворення, трендові ознаки, EWMA, RFM-ознаки), а також PCA для зниження розмірності та SMOTE для роботи з дисбалансом класів. Для кожного методу наведено математичне формулювання та обґрунтовано умови застосування.

Серед методів машинного навчання детально описано дванадцять алгоритмів – логістична регресія, градієнтний бустинг, Random Forest, дерево рішень, k-NN, SVM, Naive Bayes, AdaBoost, RNN, LSTM, GRU та CNN. Для кожного методу наведено відповідну математичну формулу, опис механізму навчання. Встановлено, що ансамблеві методи є найефективнішими на табличних даних, тоді як LSTM і GRU кращі для послідовних даних.

У розділі методів оцінювання моделей детально розглянуто шість метрик: Accuracy, Precision, Recall, F1-score, ROC-AUC та PR-AUC. Для кожної наведено формули розрахунку, пояснено математичний зміст та визначено умови, за яких використання конкретної метрики є обґрунтованим. Встановлено, що Accuracy є непридатною для задач із дисбалансом класів, а ROC-AUC і PR-AUC є найбільш інформативними для порівняльного аналізу моделей у досліджуваній задачі.

РОЗДІЛ 3 РОЗРОБКА ТА ОЦІНКА МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ ДЛЯ ПРОГНОЗУ ВІДТОКУ КОРИСТУВАЧІВ

3.1 Постановка експерименту

Суть експерименту полягає у розробці та оцінці моделей машинного навчання для прогнозу відтоку користувачів. Для цього обирається набір даних за певною сферою, виконується його обробка та підготовка для використання моделями. Підготовка даних включає використання методів препроцесингу, результати цих методів будуть зображені на відповідних рисунках. Після цього набір даних розділяється на тренувальні та тестові дані. Тренувальні будуть використовуватись для навчання обраних моделей, а тестовий – для перевірки якості навчання. У результаті буде отримано значення метрик для обраних моделей, на основі яких можна буде оцінити, яка модель дала найкращий результат, яка є неефективною та знайти додаткові цікаві закономірності та інсайти. Крім цього буде проведено порівняльний аналіз, щоб показати як змінюються результати залежно від використання додаткових методів препроцесингу. Також будуть зображені всі візуалізації, побудовані під час виконання програми.

3.2 Аналіз та обробка датасету

Для цієї роботи буде використовуватись відкритий набір даних Telco customer churn (IBM Cognos Analytics sample data sets), який представляє табличні агреговані дані про користувачів. Він складається з окремих 5 таблиць: Telco_customer_churn_demographics, Telco_customer_churn_location, Telco_customer_churn_population, Telco_customer_churn_services,

Telco_customer_churn_status. Розмірність кожної з таблиць зображена на рисунку 3.1, а структура кожної з таблиць, описана у таблицях 3.1 – 3.5.

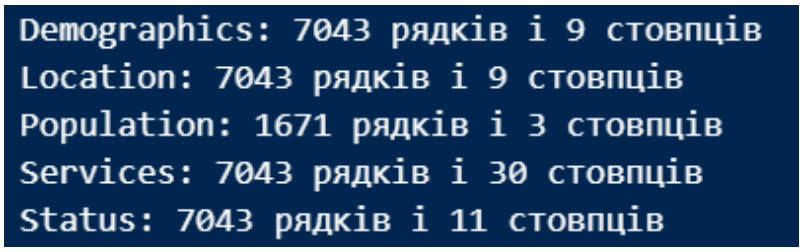


Рисунок 3.1 – Розмірність кожної з таблиць

Таблиця 3.1 – Структура таблиці Telco_customer_churn_demographics

№	*Назва змінної*	*Опис змінної*
1	Customer ID	Унікальний ідентифікатор, який ідентифікує кожного клієнта
2	Count	Значення, яке використовується для підсумовування кількості клієнтів
3	Gender	Стать клієнта: чоловік або жінка
4	Age	Поточний вік клієнта в роках
5	Under 30	Вказує, чи клієнт молодше 30 років: так або ні
6	Senior Citizen	Вказує, чи клієнту 65 років або більше: так або ні
7	Married	Вказує, чи клієнт одружений/одружена: Так, Ні
8	Dependents	Вказує, чи проживає клієнт з утриманцями: так або ні. Утриманцями є діти, батьки, бабусі й дідусі тощо
9	Number of Dependents	Вказує кількість утриманців, які проживають з клієнтом

Таблиця 3.2 – Структура таблиці Telco_customer_churn_location

№	*Назва змінної*	*Опис змінної*
1	Customer ID	Унікальний ідентифікатор клієнта
2	Count	Значення, яке використовується для підсумовування кількості клієнтів
3	Country	Країна основного місця проживання клієнта

Кінець таблиці 3.2

№	*Назва змінної*	*Опис змінної*
4	State	Штат основного місця проживання клієнта
5	City	Місто основного місця проживання клієнта
6	Zip Code	Поштовий індекс основного місця проживання клієнта
7	Lat Long	Поєднання широти та довготи основного місця проживання клієнта
8	Latitude	Широта основного місця проживання клієнта
9	Longitude	Довгота основного місця проживання клієнта

Таблиця 3.3 – Структура таблиці Telco_customer_churn_population

№	*Назва змінної*	*Опис змінної*
1	ID	Унікальний ідентифікатор, який визначає кожен рядок
2	Zip Code	Поштовий індекс основного місця проживання клієнта
3	Population	Поточна оцінка чисельності населення для всієї області поштового індексу

Таблиця 3.4 – Структура таблиці Telco_customer_churn_services

№	*Назва змінної*	*Опис змінної*
1	Customer ID	Унікальний ідентифікатор клієнта
2	Count	Значення, яке використовується для підсумовування кількості клієнтів
3	Quarter	Фінансовий квартал, з якого отримано дані
4	Referred a Friend	Вказує, чи клієнт коли-небудь рекомендував другу або члену родини цю компанію: так або ні
5	Number of Referrals	Вказує кількість рекомендацій, зроблених клієнтом
6	Tenure in Months	Вказує загальну кількість місяців, протягом яких клієнт був у компанії до кінця зазначеного вище кварталу

Продовження таблиці 3.4

№	*Назва змінної*	*Опис змінної*
7	Offer	Визначає останню маркетингову пропозицію, яку клієнт прийняв, якщо це можливо. Значення включають: Немає, Пропозиція A, Пропозиція B, Пропозиція C, Пропозиція D та Пропозиція E
8	Phone Service	Вказує, чи клієнт підписаний на домашнє телефонне обслуговування компанії: так або ні
9	Avg Monthly Long Distance Charges	Вказує середню плату клієнта за міжміські дзвінки, розраховану до кінця зазначеного вище кварталу
10	Multiple Lines	Вказує, чи клієнт підписаний на кілька телефонних ліній у компанії: так або ні
11	Internet Service	Вказує, чи клієнт підписаний на інтернет-послуги у компанії
12	Internet Type	Вказує, на яку інтернет-послугу підписаний клієнт: Ні, DSL, оптоволоконний зв'язок, кабельний зв'язок.
13	Avg Monthly GB Download	Вказує середній обсяг завантажень клієнта в гігабайтах, розрахований на кінець кварталу, зазначеного вище.
14	Online Security	Вказує, чи клієнт підписаний на послугу онлайн-безпеки: так або ні
15	Online Backup	Вказує, чи клієнт підписаний на послугу онлайн-резервного копіювання: так або ні
16	Device Protection Plan	Вказує, чи клієнт підписаний на план захисту пристрою для свого інтернет-обладнання: так або ні
17	Premium Tech Support	Вказує, чи клієнт підписаний на план технічної підтримки зі скороченим часом очікування: так або ні
18	Streaming TV	Вказує, чи клієнт використовує свій інтернет-сервіс для потокового перегляду телевізійних програм від стороннього постачальника: так або ні.
19	Streaming Movies	Вказує, чи використовує клієнт свій інтернет-сервіс для потокової передачі фільмів від стороннього постачальника: так або ні
20	Streaming Music	Вказує, чи використовує клієнт свій інтернет-сервіс для потокової передачі музики від стороннього постачальника: так або ні

Кінець таблиці 3.4

№	*Назва змінної*	*Опис змінної*
21	Unlimited Data	Вказує, чи сплатив клієнт додаткову щомісячну плату за необмежене завантаження/вивантаження даних: так або ні
22	Contract	Вказує поточний тип контракту клієнта: щомісячний, річний або дворічний
23	Paperless Billing	Вказує, чи обрав клієнт безпаперове виставлення рахунків: так або ні
24	Payment Method	Вказує, як клієнт оплачує рахунок: зняття коштів з банківського рахунку, кредитна картка або поштова картка
25	Monthly Charge	Вказує поточну загальну щомісячну плату клієнта за всі його послуги від компанії
26	Total Charges	Вказує загальну суму витрат клієнта, розраховану на кінець кварталу
27	Total Refunds	Позначає загальну суму відшкодувань клієнта, розраховану на кінець кварталу
28	Total Extra Data Charges	Позначає загальну суму витрат клієнта за додаткові завантаження даних понад зазначені в його тарифному плані, на кінець кварталу
29	Total Long Distance Charges	Позначає загальну суму витрат клієнта на міжміські дзвінки понад зазначені в його тарифному плані, на кінець кварталу
30	Total Revenue	Позначає суму всіх витрат клієнта

Таблиця 3.5 – Структура таблиці Telco_customer_churn_status

№	*Назва змінної*	*Опис змінної*
1	Customer ID	Унікальний ідентифікатор клієнта
2	Count	Значення, яке використовується для підсумовування кількості клієнтів
3	Quarter	Фінансовий квартал, з якого отримано дані
4	Satisfaction Score	Загальна оцінка задоволеності клієнта компанією від 1 (дуже незадоволений) до 5 (дуже задоволений)

Кінець таблиці 3.5

№	*Назва змінної*	*Опис змінної*
5	Customer Status	Вказує статус клієнта на кінець кварталу: покинув, залишився або приєднався
6	Churn Label	Так – клієнт залишив компанію, ні – клієнт залишився в компанії. Безпосередньо пов'язано зі значенням відтоку
7	Churn Value	1 – клієнт залишив компанію, 0 – клієнт залишився в компанії. Безпосередньо пов'язано з міткою відтоку.
8	Churn Score	Значення від 0 до 100, яке розраховується за допомогою прогностичного інструменту IBM SPSS Modeler. Модель враховує кілька факторів, які, як відомо, викликають відтік. Чим вищий показник, тим більша ймовірність того, що клієнт відійде.
9	CLTV	Цінність клієнта протягом усього терміну служби (CLTV). Прогнозована CLTV розраховується за допомогою корпоративних формул та існуючих даних. Чим вище значення, тим цінніший клієнт. Клієнтів з високою цінністю слід контролювати на предмет відтоку.
10	Churn Category	Категорія високого рівня для причини відтоку клієнта: ставлення, конкурент, невдоволення, ціна, інше. Коли вони залишають компанію, усіх клієнтів запитують про причини їхнього відходу. Безпосередньо пов'язано з причиною відтоку.
11	Churn Reason	Конкретна причина відтоку клієнта

Для зручності роботи з даними, потрібно звести їх в одну таблицю. У результаті буде отримано таблицю, яка містить 53 стовпці, оскільки було видалено повторювані стовпці. Отримані стовпці та їх типи даних показані на рисунках 3.2–3.3.

```
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   Customer ID                       7043 non-null   object
 1   Count                             7043 non-null   int64
 2   Gender                            7043 non-null   object
 3   Age                               7043 non-null   int64
 4   Under 30                          7043 non-null   object
 5   Senior Citizen                    7043 non-null   object
 6   Married                           7043 non-null   object
 7   Dependents                        7043 non-null   object
 8   Number of Dependents              7043 non-null   int64
 9   Country                           7043 non-null   object
 10  State                             7043 non-null   object
 11  City                              7043 non-null   object
 12  Zip Code                          7043 non-null   int64
 13  Lat Long                          7043 non-null   object
 14  Latitude                          7043 non-null   float64
 15  Longitude                         7043 non-null   float64
 16  Population                        7043 non-null   int64
 17  Quarter                           7043 non-null   object
 18  Referred a Friend                 7043 non-null   object
 19  Number of Referrals               7043 non-null   int64
 20  Tenure in Months                  7043 non-null   int64
 21  Offer                             3166 non-null   object
 22  Phone Service                     7043 non-null   object
 23  Avg Monthly Long Distance Charges 7043 non-null   float64
 24  Multiple Lines                    7043 non-null   object
 25  Internet Service                  7043 non-null   object
 26  Internet Type                     5517 non-null   object
```

Рисунок 3.2 – Стовпці 1–27 та їх типи даних об’єднаної таблиці

```
 27  Avg Monthly GB Download           7043 non-null   int64
 28  Online Security                   7043 non-null   object
 29  Online Backup                     7043 non-null   object
 30  Device Protection Plan            7043 non-null   object
 31  Premium Tech Support              7043 non-null   object
 32  Streaming TV                      7043 non-null   object
 33  Streaming Movies                  7043 non-null   object
 34  Streaming Music                   7043 non-null   object
 35  Unlimited Data                    7043 non-null   object
 36  Contract                          7043 non-null   object
 37  Paperless Billing                 7043 non-null   object
 38  Payment Method                    7043 non-null   object
 39  Monthly Charge                    7043 non-null   float64
 40  Total Charges                     7043 non-null   float64
 41  Total Refunds                     7043 non-null   float64
 42  Total Extra Data Charges          7043 non-null   int64
 43  Total Long Distance Charges       7043 non-null   float64
 44  Total Revenue                     7043 non-null   float64
 45  Satisfaction Score                7043 non-null   int64
 46  Customer Status                   7043 non-null   object
 47  Churn Label                       7043 non-null   object
 48  Churn Value                       7043 non-null   int64
 49  Churn Score                       7043 non-null   int64
 50  CLTV                              7043 non-null   int64
 51  Churn Category                    1869 non-null   object
 52  Churn Reason                      1869 non-null   object
dtypes: float64(8), int64(13), object(32)
```

Рисунок 3.3 – Стовпці 28–53 та їх типи даних об’єднаної таблиці

Тепер можна проаналізувати отримані дані та додатково видалити стовпці, які є зайвими (неінформативними). Стовпці, які будуть видалені показані на рисунку 3.4.

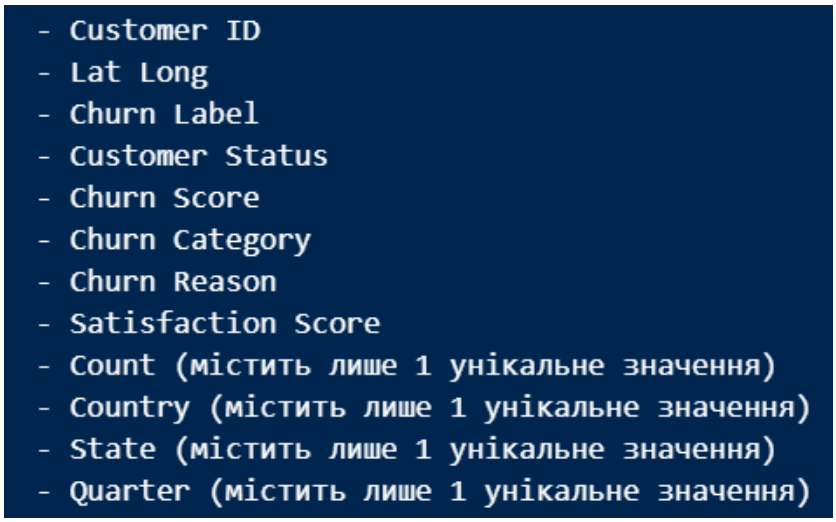
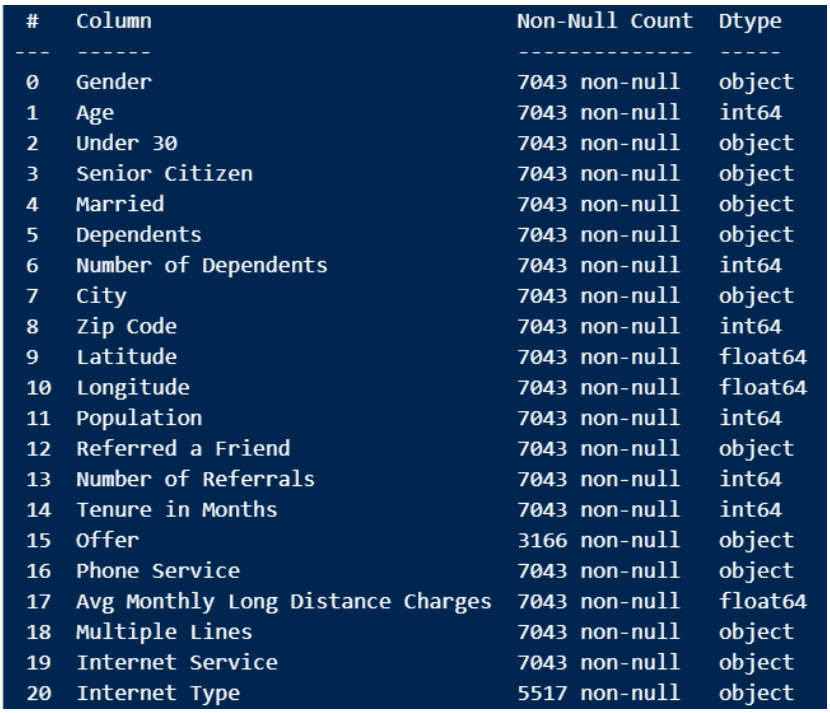


Рисунок 3.4 – Назви стовпців, які будуть видалені з об’єднаної таблиці

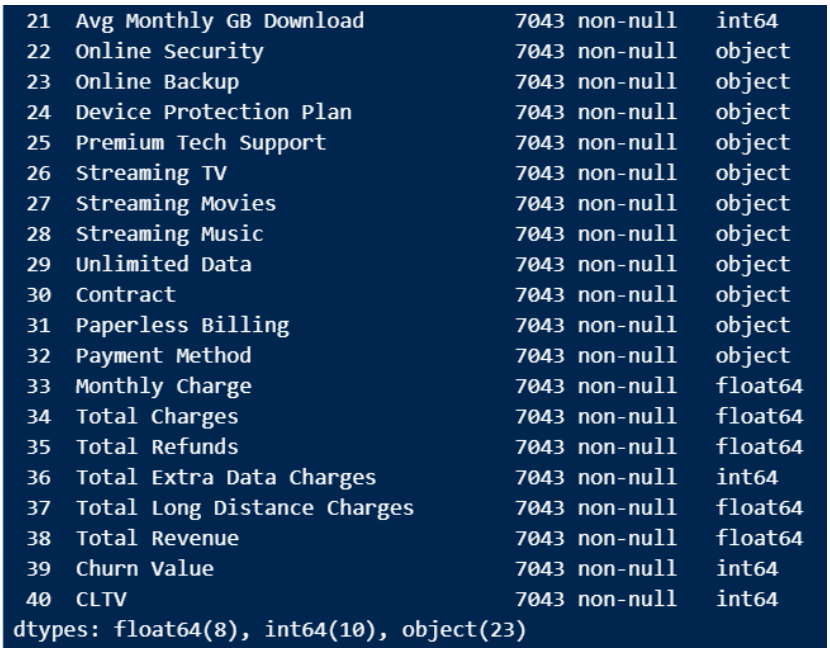
Customer ID є просто ідентифікатором клієнта, він буде унікальним для кожного користувача і не буде нести жодної предиктивної цінності. Оскільки таблиця вже містить окремі стовпці Latitude і Longitude, то стовпець Lat Long є надлишковим, бо він буде дублювати інформацію цих двох стовпців. Стовпці Churn Label та Customer Status фактично дублюють Churn Value і просто представляють його у текстовому вигляді, тому ці 2 стовпці теж можна видалити. Churn Score – це предиктивний бал ймовірності відтоку, який інженери IBM розраховували своїми моделями, тому використовувати цей стовпчик як ознаку є недоречним для побудови прогностичної моделі. Використання стовпців Churn Category і Churn Reason є помилковим, оскільки значення цих стовпців будуть заповнені тільки коли клієнт розриває контракт, тому модель швидко запам’ятовує: є текст у полі – клієнт пішов, немає тексту – залишився, що завищує точність прогнозу моделі. Satisfaction Score є не фактором відтоку, а скоріше його наслідком, тому цей стовпець

теж варто видалити. Стовпці Count, Country, State, Quarter не несуть предиктивної цінності, бо містять лише одне унікальне значення. Після видалення зайвих стовпців буде отримано таблицю, стовпці якої показані на рисунках 3.5–3.6, а статистичний аналіз – на рисунку 3.7.



#	Column	Non-Null Count	Dtype
0	Gender	7043 non-null	object
1	Age	7043 non-null	int64
2	Under 30	7043 non-null	object
3	Senior Citizen	7043 non-null	object
4	Married	7043 non-null	object
5	Dependents	7043 non-null	object
6	Number of Dependents	7043 non-null	int64
7	City	7043 non-null	object
8	Zip Code	7043 non-null	int64
9	Latitude	7043 non-null	float64
10	Longitude	7043 non-null	float64
11	Population	7043 non-null	int64
12	Referred a Friend	7043 non-null	object
13	Number of Referrals	7043 non-null	int64
14	Tenure in Months	7043 non-null	int64
15	Offer	3166 non-null	object
16	Phone Service	7043 non-null	object
17	Avg Monthly Long Distance Charges	7043 non-null	float64
18	Multiple Lines	7043 non-null	object
19	Internet Service	7043 non-null	object
20	Internet Type	5517 non-null	object

Рисунок 3.5 – Стовпці 1–21 та їх типи даних об’єднаної таблиці



21	Avg Monthly GB Download	7043 non-null	int64
22	Online Security	7043 non-null	object
23	Online Backup	7043 non-null	object
24	Device Protection Plan	7043 non-null	object
25	Premium Tech Support	7043 non-null	object
26	Streaming TV	7043 non-null	object
27	Streaming Movies	7043 non-null	object
28	Streaming Music	7043 non-null	object
29	Unlimited Data	7043 non-null	object
30	Contract	7043 non-null	object
31	Paperless Billing	7043 non-null	object
32	Payment Method	7043 non-null	object
33	Monthly Charge	7043 non-null	float64
34	Total Charges	7043 non-null	float64
35	Total Refunds	7043 non-null	float64
36	Total Extra Data Charges	7043 non-null	int64
37	Total Long Distance Charges	7043 non-null	float64
38	Total Revenue	7043 non-null	float64
39	Churn Value	7043 non-null	int64
40	CLTV	7043 non-null	int64

dtypes: float64(8), int64(10), object(23)

Рисунок 3.6 – Стовпці 22–41 та їх типи даних об’єднаної таблиці

	count	mean	std	min	25%	50%	75%	max
Age	7043.0	46.51	16.75	19.00	32.00	46.00	60.00	80.00
Number of Dependents	7043.0	0.47	0.96	0.00	0.00	0.00	0.00	9.00
Latitude	7043.0	36.20	2.47	32.56	33.99	36.21	38.16	41.96
Longitude	7043.0	-119.76	2.15	-124.30	-121.79	-119.60	-117.97	-114.19
Population	7043.0	22139.60	21152.39	11.00	2344.00	17554.00	36125.00	105285.00
Number of Referrals	7043.0	1.95	3.00	0.00	0.00	0.00	3.00	11.00
Tenure in Months	7043.0	32.39	24.54	1.00	9.00	29.00	55.00	72.00
Avg Monthly Long Distance Charges	7043.0	22.96	15.45	0.00	9.21	22.89	36.39	49.99
Avg Monthly GB Download	7043.0	20.52	20.42	0.00	3.00	17.00	27.00	85.00
Monthly Charge	7043.0	64.76	30.09	18.25	35.50	70.35	89.85	118.75
Total Charges	7043.0	2280.38	2266.22	18.80	400.15	1394.55	3786.60	8684.80
Total Refunds	7043.0	1.96	7.90	0.00	0.00	0.00	0.00	49.79
Total Extra Data Charges	7043.0	6.86	25.10	0.00	0.00	0.00	0.00	150.00
Total Long Distance Charges	7043.0	749.10	846.66	0.00	70.54	401.44	1191.10	3564.72
Total Revenue	7043.0	3034.38	2865.20	21.36	605.61	2108.64	4801.15	11979.34
Churn Value	7043.0	0.27	0.44	0.00	0.00	0.00	1.00	1.00
CLTV	7043.0	4400.30	1183.06	2003.00	3469.00	4527.00	5380.50	6500.00

Рисунок 3.7 – Статистичний аналіз для числових стовпців

Також варто перевірити наявність пропущених значень у стовпцях та переглянути співвідношення значень цільової змінної, щоб визначити, чи потрібно виконувати обробку пропущених значень та чи доцільно застосовувати метод SMOTE на етапі препоцесингу. Результати перевірок показані на рисунках 3.8–3.9.

```
Стовпець 'Offer' містить 3877 пропущених значень
Стовпець 'Internet Type' містить 1526 пропущених значень

Фрагменти стовпців з пропущеними значеннями:
   Offer
0    NaN
16   NaN
23   NaN
24   NaN
25   NaN
30   NaN
32   NaN
33   NaN
34   NaN
39   NaN

    Internet Type
25            NaN
41            NaN
199           NaN
373           NaN
375           NaN
490           NaN
526           NaN
550           NaN
554           NaN
561           NaN
```

Рисунок 3.8 – Перевірка наявності пропущених значень

```
Стовпець 'Churn Value' містить 5174 записи зі значенням 0 і 1869 записів зі значенням 1,
що відповідно становить 73.46% і 26.54% від загальної кількості записів.
```

Рисунок 3.9 – Перевірка наявності дисбалансу класів

Можна побачити, що у стовпців Offer та Internet Type є пропущені значення, тому під час препроцесингу буде замінено їх модою, оскільки це категоріальна змінна. Менший клас складає 26,54% від загальної кількості записів, що вказує на помірний дисбаланс класів, тому доцільно застосувати метод SMOTE.

3.3 Підготовка набору даних для використання моделями

3.3.1 Основні методи підготовки даних для використання моделями

Першим кроком потрібно провести формування ознак (feature engineering), для цього буде виконано логарифмічне перетворення для деяких стовпців, будуть побудовані відносні ознаки та створені спеціальні ознаки – чи є новим клієнт та кількість сервісів, якими користується клієнт. Результати формування ознак кожним методом продемонстровані на рисунках 3.10–3.13.

```
1. Формування ознак (feature engineering)

1.1 Логарифмічне перетворення
До логарифмічного перетворення:
      Monthly Charge  Total Charges  Total Long Distance Charges  Total Revenue  Population
4626           55.30         875.35                        92.64         967.99         816
4192          105.30        1275.65                       203.52        1479.17       40137
5457           49.00          49.00                        49.19          98.19        3979
4717           68.40        3972.25                      1565.42        5537.67       69850
4673           80.00         241.30                        57.54         298.84       58198
3840           24.70          24.70                         0.00          94.70       43141
3823           49.90        1410.25                      1159.48        2569.73       66838
617           105.40        6998.95                      1337.91        8336.86        5654
6112           55.25        1924.10                       713.65        2637.75       39458
3535           45.05        1790.60                         0.00        1790.60        2156

Після логарифмічного перетворення
      Log_Monthly Charge  Log_Total Charges  Log_Total Long Distance Charges  Log_Total Revenue  Log_Population
4626              4.0307             6.7758                           4.5395             6.8763          6.7056
4192              4.6663             7.1520                           5.3207             7.2999         10.6001
5457              3.9120             3.9120                           3.9158             4.5970          8.2890
4717              4.2399             8.2873                           7.3565             8.6195         11.1541
4673              4.3944             5.4902                           4.0697             5.7032         10.9716
3840              3.2465             3.2465                           0.0000             4.5612         10.6723
3823              3.9299             7.2522                           7.0566             7.8519         11.1100
617               4.6672             8.8537                           7.1996             9.0286          8.6403
6112              4.0298             7.5627                           6.5718             7.8781         10.5830
3535              3.8297             7.4909                           0.0000             7.4909          7.6765
```

Рисунок 3.10 – Результат застосування логарифмічного перетворення

```
1.2 Відносні ознаки
До створення відносних ознак:
      Monthly Charge  Total Charges  Tenure in Months  Total Extra Data Charges  Avg Monthly GB Download  Number of Referrals
4626           55.30         875.35                16                         0                        4                    0
4192          105.30        1275.65                12                         0                       18                    0
5457           49.00          49.00                 1                         0                       17                    0
4717           68.40        3972.25                58                         0                       53                    8
4673           80.00         241.30                 3                         0                       22                    2
3840           24.70          24.70                 1                        70                       48                    0
3823           49.90        1410.25                28                         0                       30                    6
617           105.40        6998.95                69                         0                       13                    2
6112           55.25        1924.10                35                         0                       21                    0
3535           45.05        1790.60                39                         0                       13                    0

Після створення відносних ознак:
      Charges_Per_Tenure  Monthly_to_Total_Ratio  Extra_Charges_Ratio  GB_Per_Dollar  Referrals_Per_Tenure
4626             54.7094                  0.0632                0.000         0.0723                0.0000
4192            106.3042                  0.0825                0.000         0.1709                0.0000
5457             49.0000                  1.0000                0.000         0.3469                0.0000
4717             68.4871                  0.0172                0.000         0.7749                0.1379
4673             80.4333                  0.3315                0.000         0.2750                0.6667
3840             24.7000                  1.0000                2.834         1.9433                0.0000
3823             50.3661                  0.0354                0.000         0.6012                0.2143
617             101.4341                  0.0151                0.000         0.1233                0.0290
6112             54.9743                  0.0287                0.000         0.3801                0.0000
3535             45.9128                  0.0252                0.000         0.2886                0.0000
```

Рисунок 3.11 – Результат створення відносних ознак

```
1.3 Сегментація клієнтів
До сегментації клієнтів:
      Tenure in Months
4626                16
4192                12
5457                 1
4717                58
4673                 3
3840                 1
3823                28
617                 69
6112                35
3535                39

Після сегментації клієнтів ('Новий клієнт' <= 12 місяців):
      Is_New_Customer
4626                0
4192                1
5457                1
4717                0
4673                1
3840                1
3823                0
617                 0
6112                0
3535                0
```

Рисунок 3.12 – Результат створення ознаки чи є новим клієнт

```
1.4 Кількість сервісів, якими користується клієнт
Статуси окремих сервісів:
     Phone Service Internet Service Online Security Streaming TV
4626           Yes              Yes              No           No
4192           Yes              Yes              No          Yes
5457           Yes              Yes              No           No
4717           Yes              Yes             Yes           No
4673           Yes              Yes              No           No
3840            No              Yes              No           No
3823           Yes              Yes              No           No
617            Yes              Yes              No          Yes
6112           Yes              Yes             Yes           No
3535            No              Yes             Yes          Yes

Агрегована кількість активних сервісів:
      NumberOfServicesUsed
4626                     5
4192                     9
5457                     4
4717                     8
4673                     5
3840                     1
3823                     4
617                      9
6112                     5
3535                     5
```

Рисунок 3.13 – Результат створення ознаки кількості сервісів, якими користується клієнт

Наступним кроком підготовки даних є заповнення пропусків. Для цього набору даних пропущені значення є лише у категоріальних змінних, тому заповнення пропусків буде виконано за допомогою моди вибірки. Також для того щоб моделі могли обробляти категоріальні ознаки, потрібно перетворити їх на числові ознаки за допомогою кодування.

Результат заповнення пропусків показаний на рисунках 3.14–3.15, а результат кодування категоріальних ознак показаний на рисунку 3.16.

```
2. Заповнення пропусків

Стовпець: Offer
Кількість пропусків у тренувальній вибірці до заповнення: 3106
До заповнення пропусків:
     Offer
4626   NaN
4192   NaN
5457   NaN
4717   NaN
4673   NaN
617    NaN
6112   NaN
3822   NaN
3911   NaN
4166   NaN

Після заповнення пропусків:
        Offer
4626  Offer B
4192  Offer B
5457  Offer B
4717  Offer B
4673  Offer B
617   Offer B
6112  Offer B
3822  Offer B
3911  Offer B
4166  Offer B
```

Рисунок 3.14 – Результат заповнення пропусків для стовпця Offer

```
Стовпець: Internet Type
Кількість пропусків у тренувальній вибірці до заповнення: 1217
До заповнення пропусків:
     Internet Type
4769           NaN
3911           NaN
4077           NaN
6454           NaN
4273           NaN
5113           NaN
5808           NaN
3197           NaN
6182           NaN
3708           NaN

Після заповнення пропусків:
     Internet Type
4769   Fiber Optic
3911   Fiber Optic
4077   Fiber Optic
6454   Fiber Optic
4273   Fiber Optic
5113   Fiber Optic
5808   Fiber Optic
3197   Fiber Optic
6182   Fiber Optic
3708   Fiber Optic

Загальна кількість пропусків у всьому датасеті після відпрацювання Pipeline: 0
```

Рисунок 3.15 – Результат заповнення пропусків для стовпця Internet Type

```
3. Кодування категоріальних ознак
До кодування:
            Contract
4626  Month-to-Month
4192  Month-to-Month
5457  Month-to-Month
4717        One Year
4673  Month-to-Month
3840  Month-to-Month
3823  Month-to-Month
617         Two Year
6112        Two Year
3535        One Year

Після кодування:
      Contract_Month-to-Month  Contract_One Year  Contract_Two Year
4626                      1.0                0.0                0.0
4192                      1.0                0.0                0.0
5457                      1.0                0.0                0.0
4717                      0.0                1.0                0.0
4673                      1.0                0.0                0.0
3840                      1.0                0.0                0.0
3823                      1.0                0.0                0.0
617                       0.0                0.0                1.0
6112                      0.0                0.0                1.0
3535                      0.0                1.0                0.0
```

Рисунок 3.16 – Результат кодування категоріальних ознак

Останнім етапом препроцесингу є масштабування. У роботі використовується Robust Scaler, оскільки саме цей метод найменш чутливий до викидів. Результат застосування RobustScaler показаний на рисунку 3.17.

```
4. Масштабування
До масштабування:
      Total Charges
4626         875.35
4192        1275.65
5457          49.00
4717        3972.25
4673         241.30
3840          24.70
3823        1410.25
617         6998.95
6112        1924.10
3535        1790.60

Після масштабування:
      Total Charges
4626        -0.1545
4192        -0.0366
5457        -0.3979
4717         0.7575
4673        -0.3412
3840        -0.4050
3823         0.0030
617          1.6489
6112         0.1544
3535         0.1150
```

Рисунок 3.17 – Результат застосування масштабування

3.3.2 Додаткові методи підготовки даних для використання моделями

У роботі застосовано два додаткові методи підготовки даних. Метод головних компонент (PCA) дозволяє математично стиснути розширений простір ознак, зберігши при цьому понад 95% початкової дисперсії даних. Алгоритм SMOTE вирішує проблему дисбалансу класів. Шляхом генерації синтетичних спостережень меншого класу, SMOTE запобігає зміщенню моделей у бік мажоритарного класу. Результати застосування цих методів показані на рисунках 3.18–3.20.

```
PCA Analysis
Кількість PCA-компонент: 3
Пояснений коефіцієнт дисперсії: [0.8227 0.0823 0.055 ]

[Головна Компонента 1] - Пояснює 82.27% дисперсії:
  Total Extra Data Charges            :   0.9997
  Monthly_to_Total_Ratio              :  -0.0107
  Referrals_Per_Tenure                :  -0.0092
  Unlimited Data_Yes                  :  -0.0074
  Unlimited Data_No                   :   0.0074

[Головна Компонента 2] - Пояснює 8.23% дисперсії:
  Total Refunds                       :   0.9975
  Referrals_Per_Tenure                :  -0.0483
  Monthly_to_Total_Ratio              :  -0.0459
  Log_Total Charges                   :   0.0098
  Log_Total Revenue                   :   0.0097

[Головна Компонента 3] - Пояснює 5.50% дисперсії:
  Referrals_Per_Tenure                :   0.9876
  Monthly_to_Total_Ratio              :   0.1171
  Total Refunds                       :   0.0544
  Number of Referrals                 :   0.0490
  Log_Total Revenue                   :  -0.0272
```

Рисунок 3.18 – Результат застосування PCA

Рисунок 3.19 – Графік поясненої дисперсії залежно від кількості компонент

```
SMOTE Analysis
До застосування SMOTE:
Стовпець 'Churn Value' містить 4139 записи зі значенням 0 і 1495 записів зі значенням 1.

Після застосування SMOTE:
Стовпець 'Churn Value' містить 4139 записи зі значенням 0 і 4139 записів зі значенням 1.
```

Рисунок 3.20 – Результати застосування SMOTE

3.4 Побудова та оцінка моделей

У роботі будуть порівнюватися 6 моделей: Logistic Regression, XGboost, LightGBM, Random Forest, SVM, k-NN. Для кожної з моделей буде проведене навчання, розраховано відповідні метрики, побудовано порівняльну таблицю моделей та для кожної з моделей побудовано візуалізації (матриця плутанини, PR та ROC криві). Також буде побудовано візуалізацію для показу топ-10 важливих ознак для кожної моделі. Після цього буде проведено порівняльний аналіз результатів з застосуванням методів PCA,

SMOTE і без них, щоб перевірити наскільки вони є ефективними для цих моделей та даних. Результати навчання моделей показані на рисунках 3.21–3.44.

```
Logistic Regression

Accuracy: 0.848
ROC-AUC: 0.910
PR-AUC: 0.784
              precision    recall  f1-score   support

           0       0.88      0.92      0.90      1035
           1       0.75      0.64      0.69       374

    accuracy                           0.85      1409
   macro avg       0.81      0.78      0.80      1409
weighted avg       0.84      0.85      0.84      1409
```

Рисунок 3.21 – Значення метрик для Logistic Regression

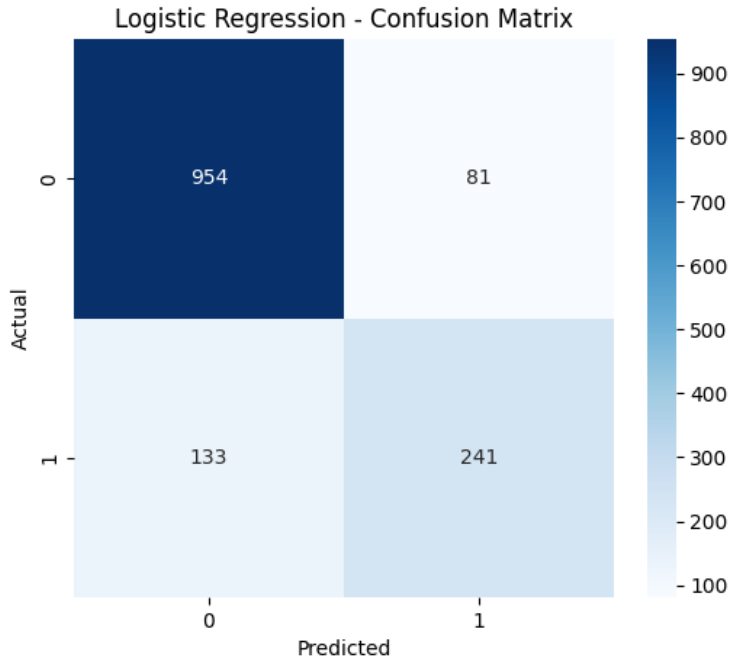


Рисунок 3.22 – Матриця плутанини для Logistic Regression

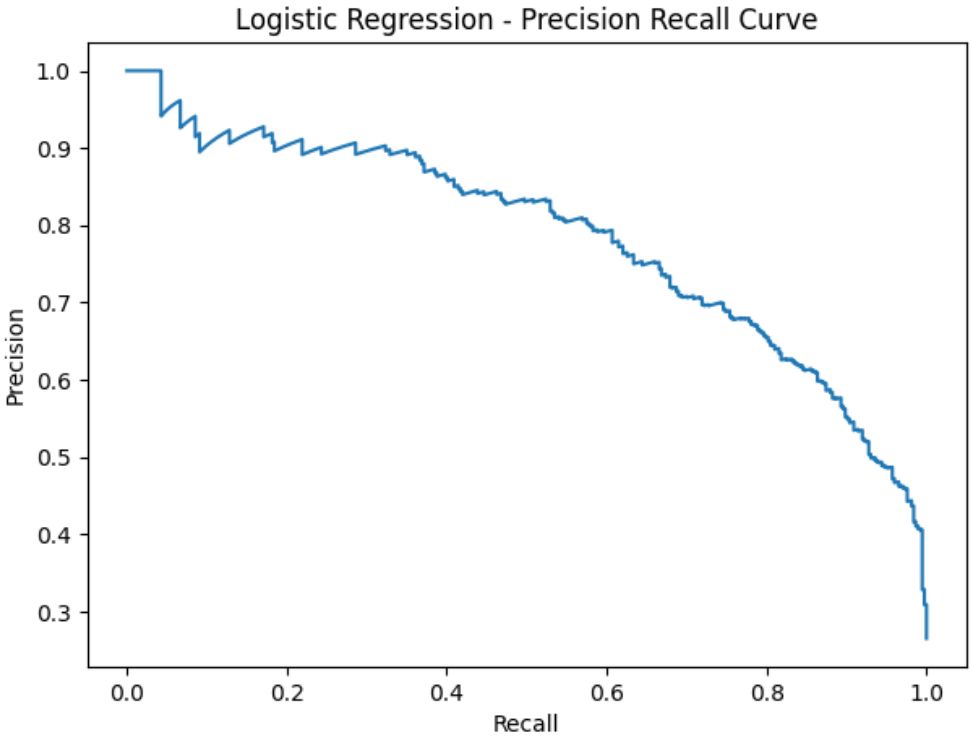


Рисунок 3.23 – PR-крива для Logistic Regression

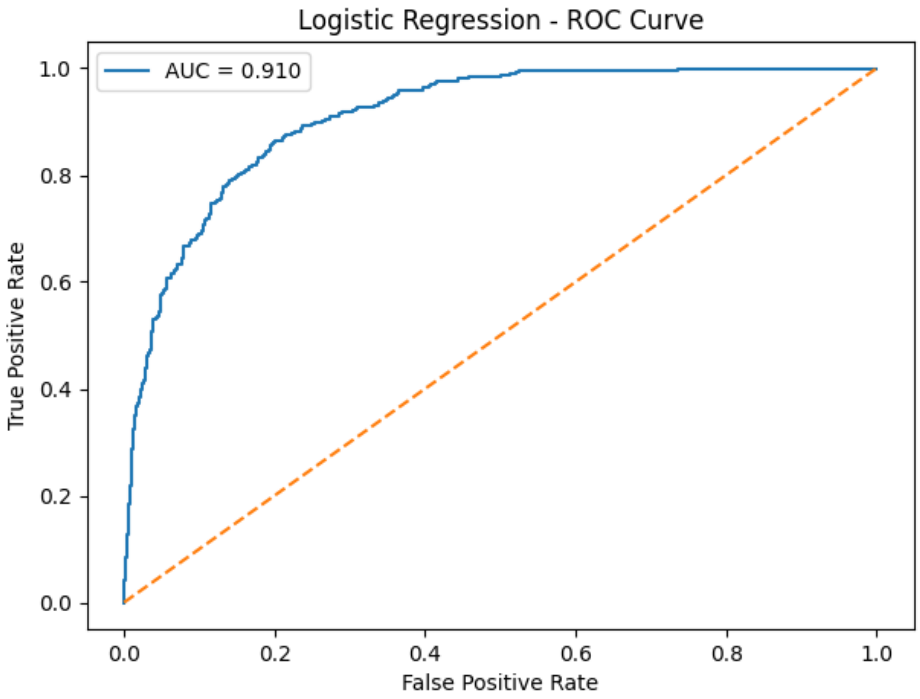


Рисунок 3.24 – ROC-крива для Logistic Regression

```
Random Forest

Accuracy: 0.844
ROC-AUC: 0.895
PR-AUC: 0.755
              precision    recall  f1-score   support

           0       0.87      0.93      0.90      1035
           1       0.76      0.60      0.67       374

    accuracy                           0.84      1409
   macro avg       0.81      0.77      0.78      1409
weighted avg       0.84      0.84      0.84      1409
```

Рисунок 3.25 – Значення метрик для Random Forest

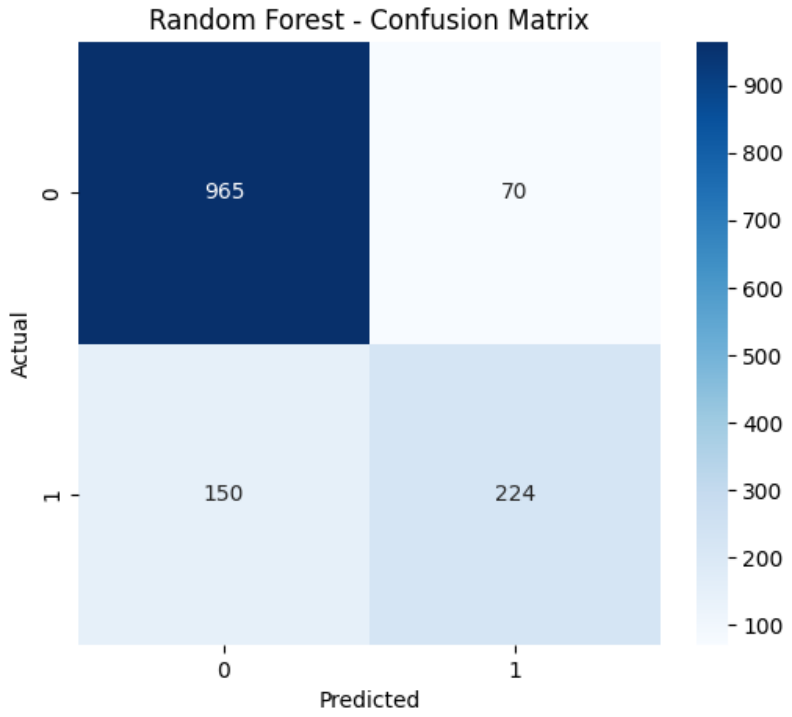


Рисунок 3.26 – Матриця плутанини для Random Forest

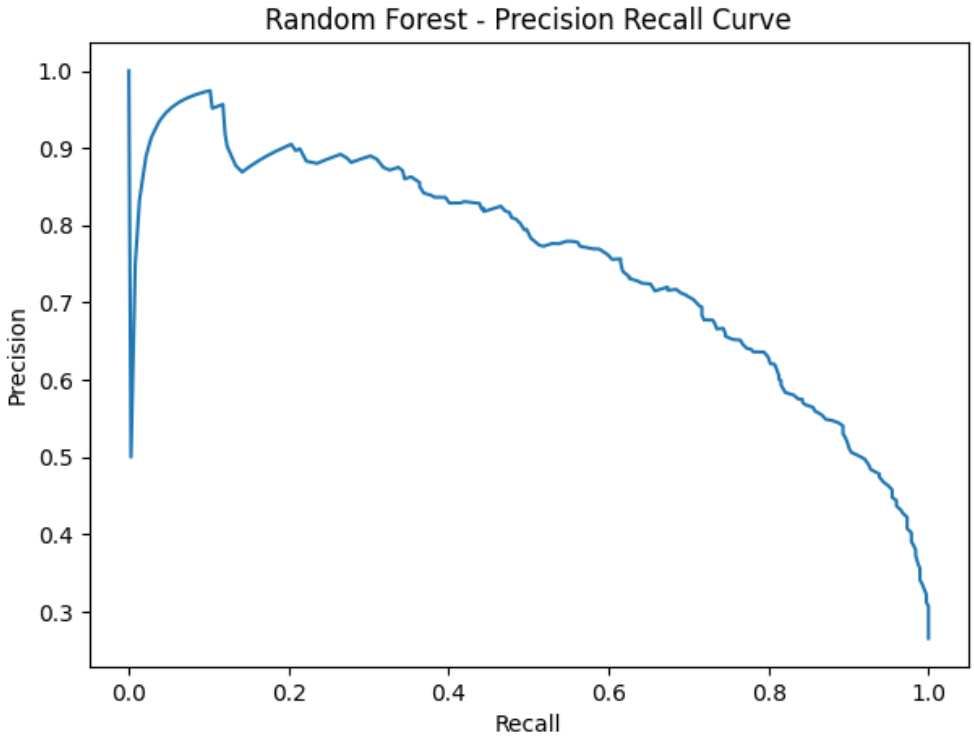


Рисунок 3.27 – PR-крива для Random Forest

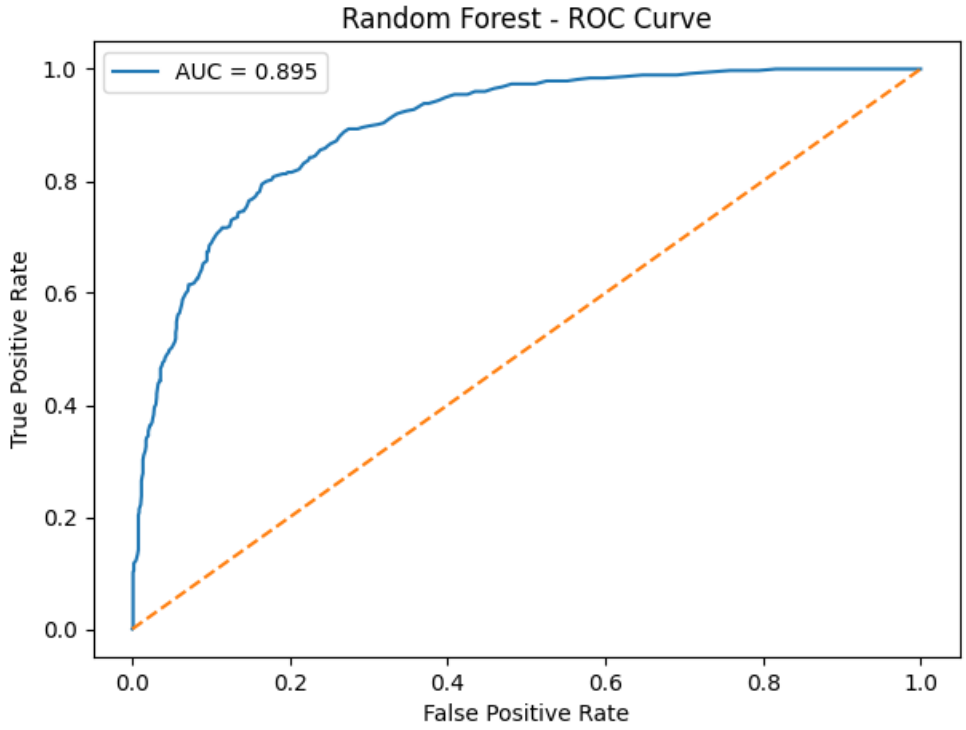


Рисунок 3.28 – ROC-крива для Random Forest

```
SVM

Accuracy: 0.831
ROC-AUC: 0.887
PR-AUC: 0.739
              precision    recall  f1-score   support

           0       0.84      0.95      0.89      1035
           1       0.80      0.49      0.61       374

    accuracy                           0.83      1409
   macro avg       0.82      0.72      0.75      1409
weighted avg       0.83      0.83      0.82      1409
```

Рисунок 3.29 – Значення метрик для SVM

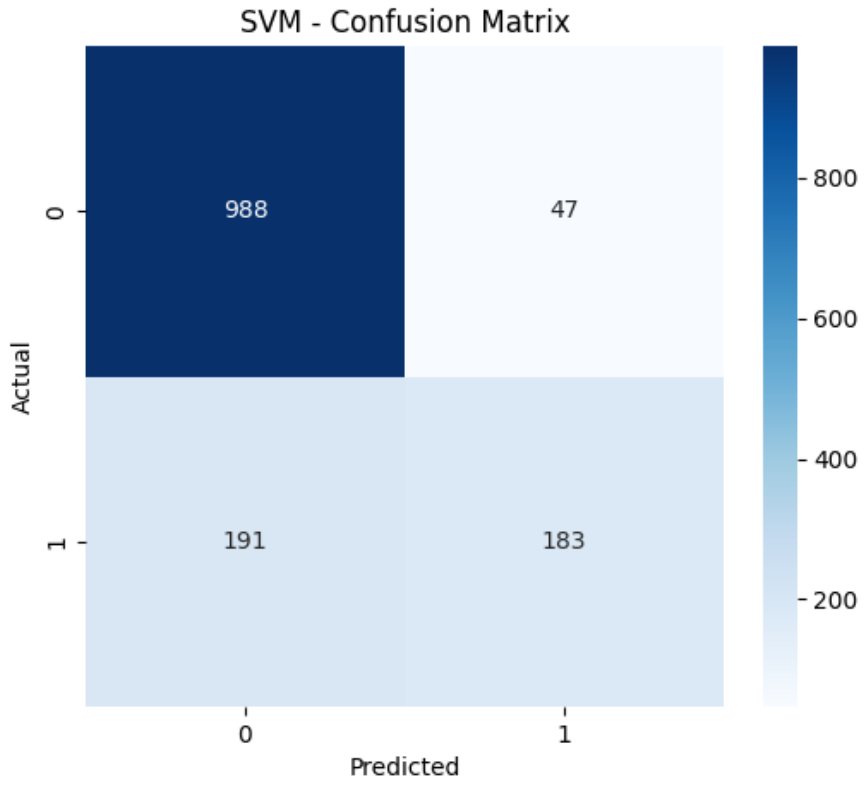


Рисунок 3.30 – Матриця плутанини для SVM

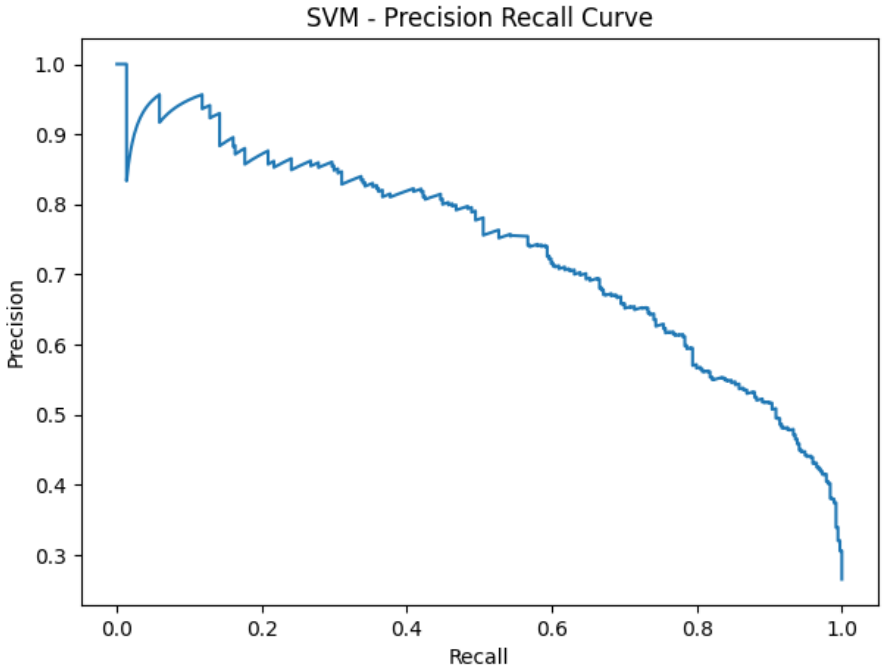


Рисунок 3.31 – PR-крива для SVM

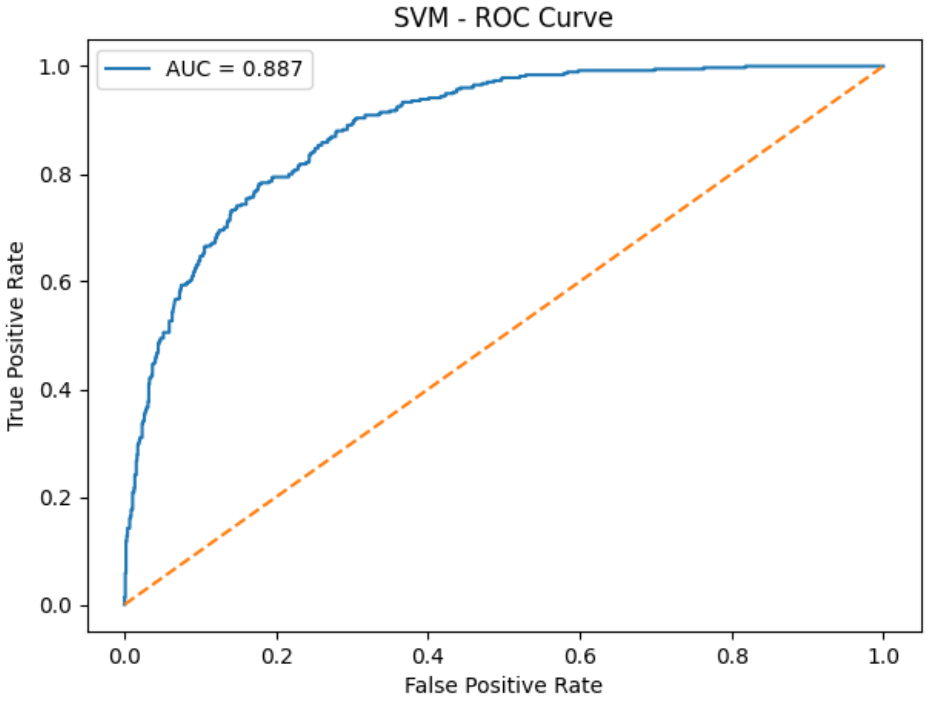


Рисунок 3.32 – ROC-крива для SVM

```
kNN

Accuracy: 0.792
ROC-AUC: 0.822
PR-AUC: 0.589
              precision    recall  f1-score   support

           0       0.85      0.88      0.86      1035
           1       0.62      0.56      0.59       374

    accuracy                           0.79      1409
   macro avg       0.73      0.72      0.72      1409
weighted avg       0.79      0.79      0.79      1409
```

Рисунок 3.33 – Значення метрик для k-NN

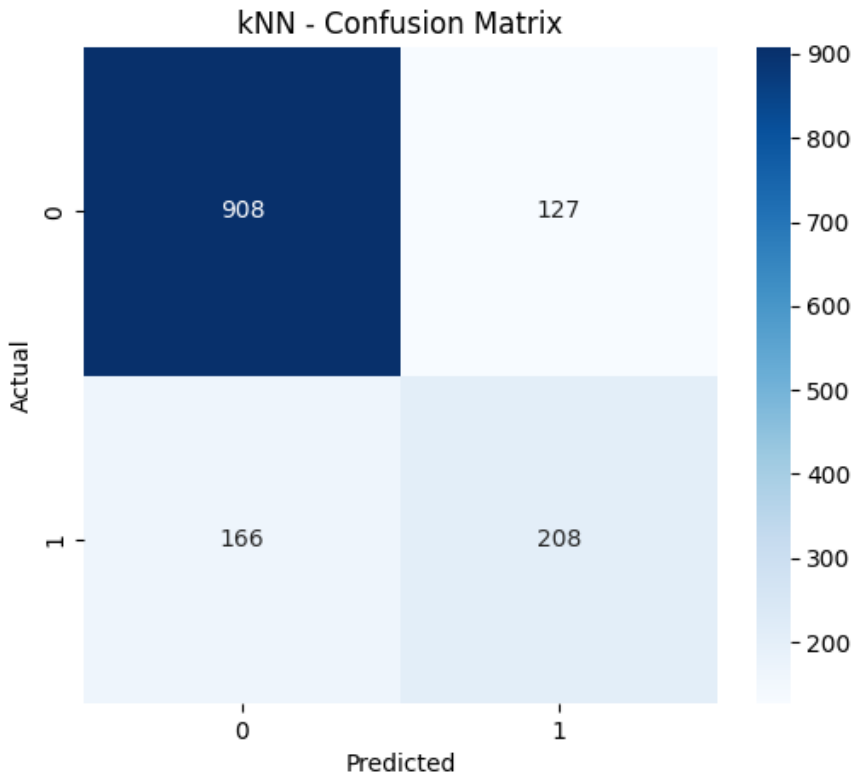


Рисунок 3.34 – Матриця плутанини для k-NN

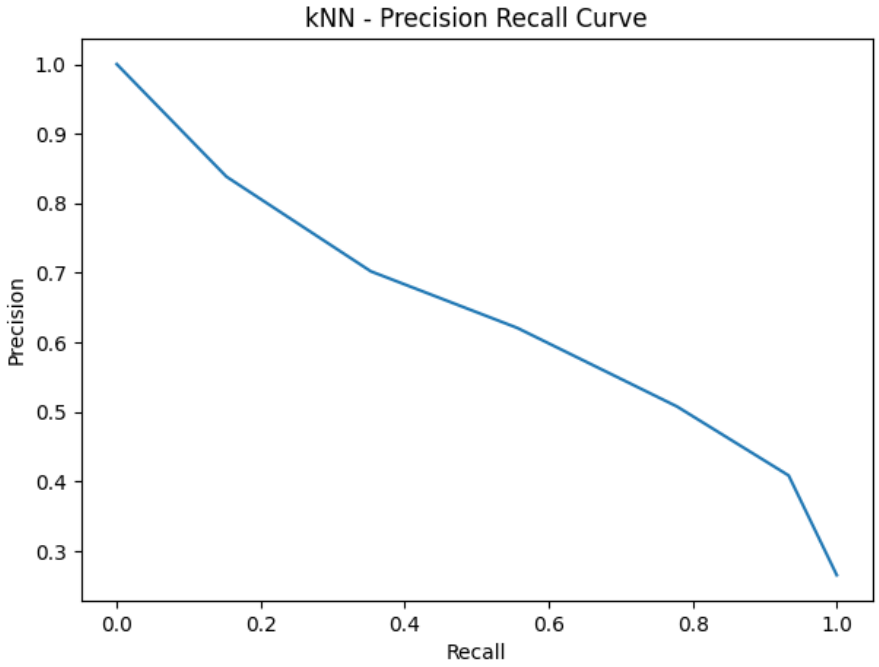


Рисунок 3.35 – PR-крива для k-NN

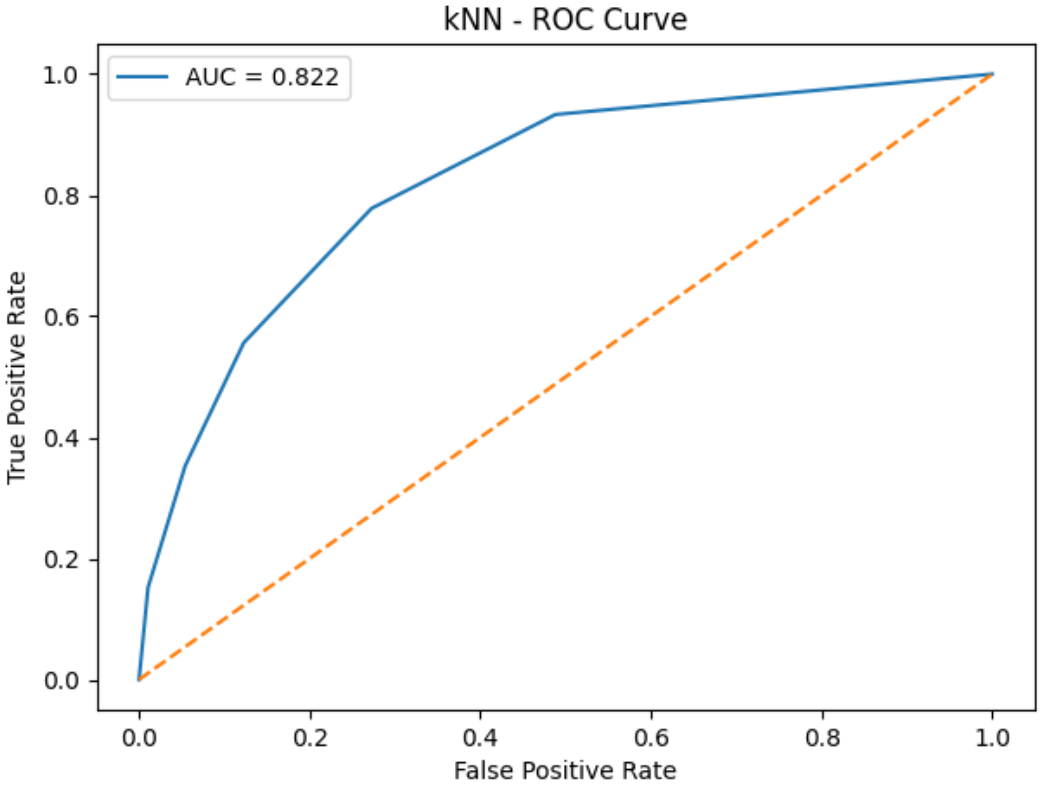


Рисунок 3.36 – ROC-крива для k-NN

```
XGBoost

Accuracy: 0.838
ROC-AUC: 0.904
PR-AUC: 0.777
              precision    recall  f1-score   support

           0       0.87      0.92      0.89      1035
           1       0.73      0.62      0.67       374

    accuracy                           0.84      1409
   macro avg       0.80      0.77      0.78      1409
weighted avg       0.83      0.84      0.83      1409
```

Рисунок 3.37 – Значення метрик для XGBoost

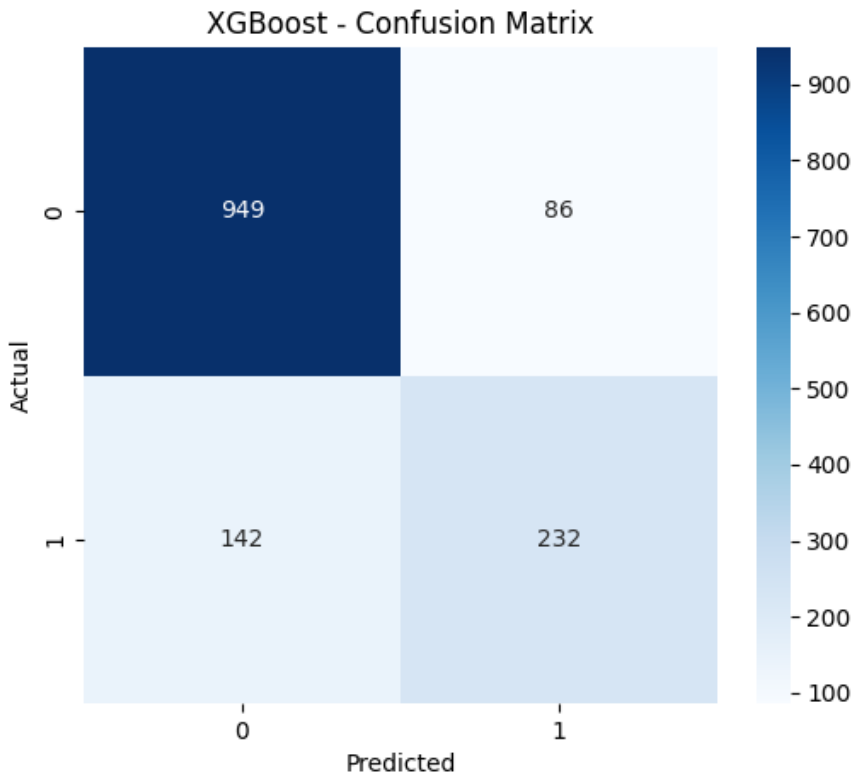


Рисунок 3.38 – Матриця плутанини для XGBoost

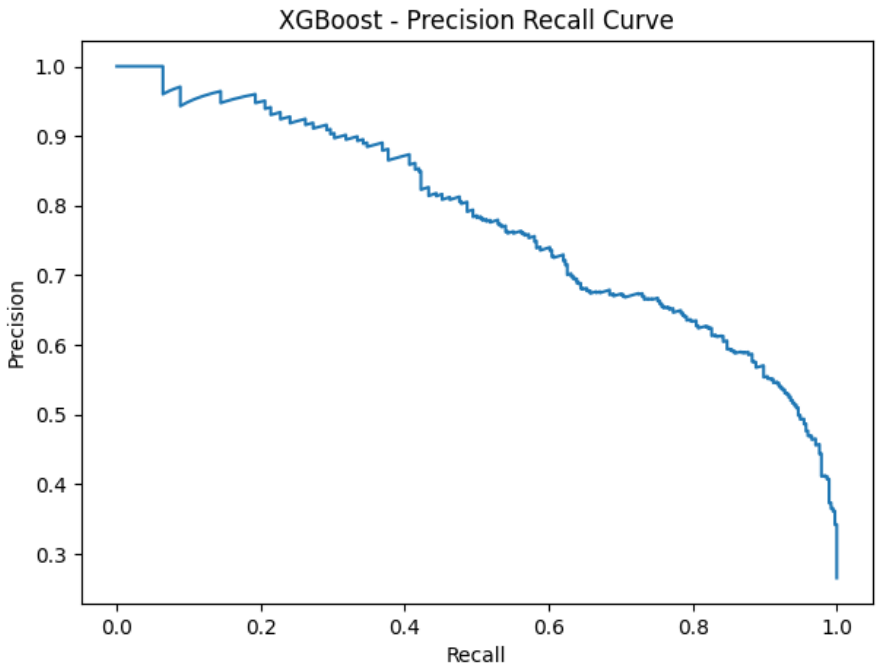


Рисунок 3.39 – PR-крива для XGBoost

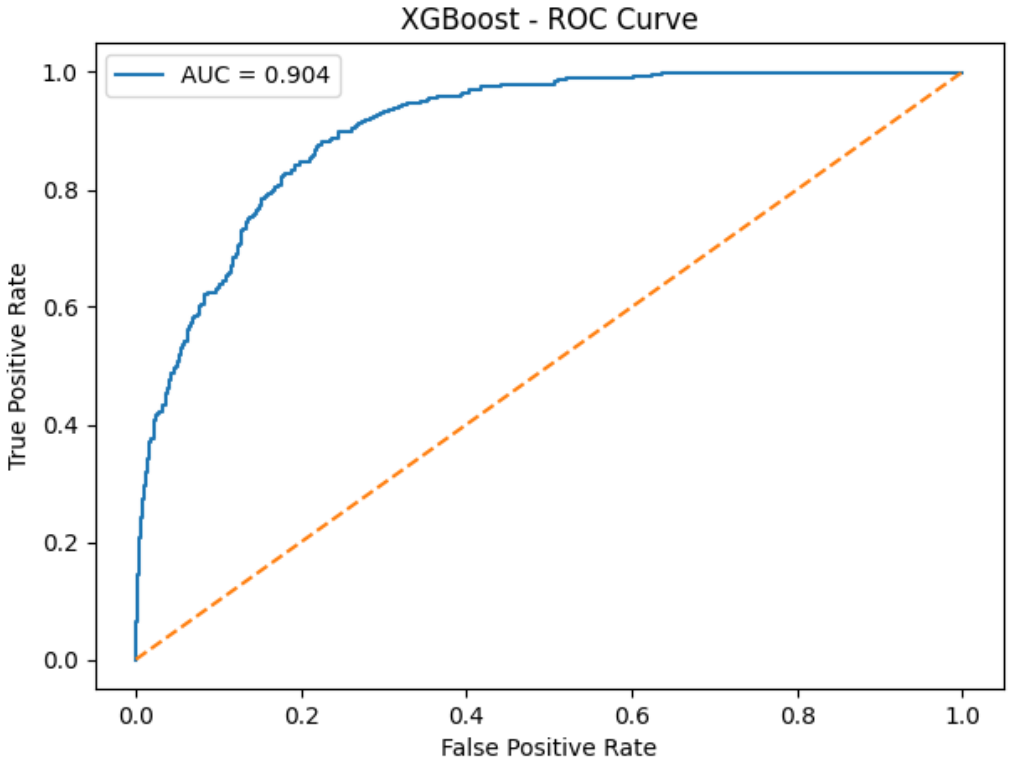


Рисунок 3.40 – ROC-крива для XGBoost

```
LightGBM

Accuracy: 0.848
ROC-AUC: 0.908
PR-AUC: 0.785
              precision    recall  f1-score   support

           0       0.88      0.92      0.90      1035
           1       0.74      0.65      0.70       374

    accuracy                           0.85      1409
   macro avg       0.81      0.79      0.80      1409
weighted avg       0.84      0.85      0.84      1409
```

Рисунок 3.41 – Значення метрик для LightGBM

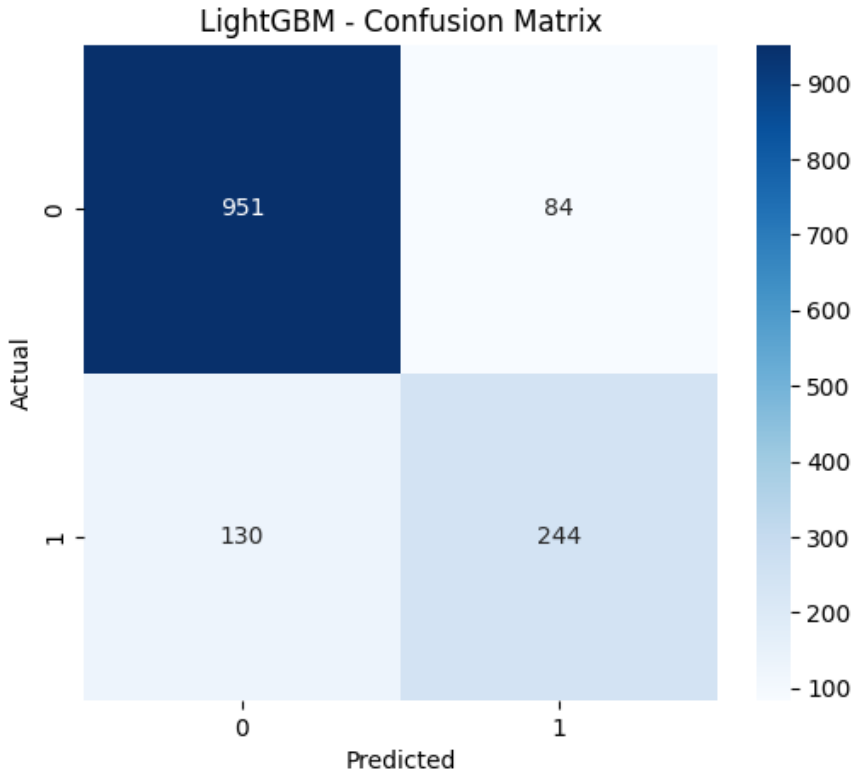


Рисунок 3.42 – Матриця плутанини для LightGBM

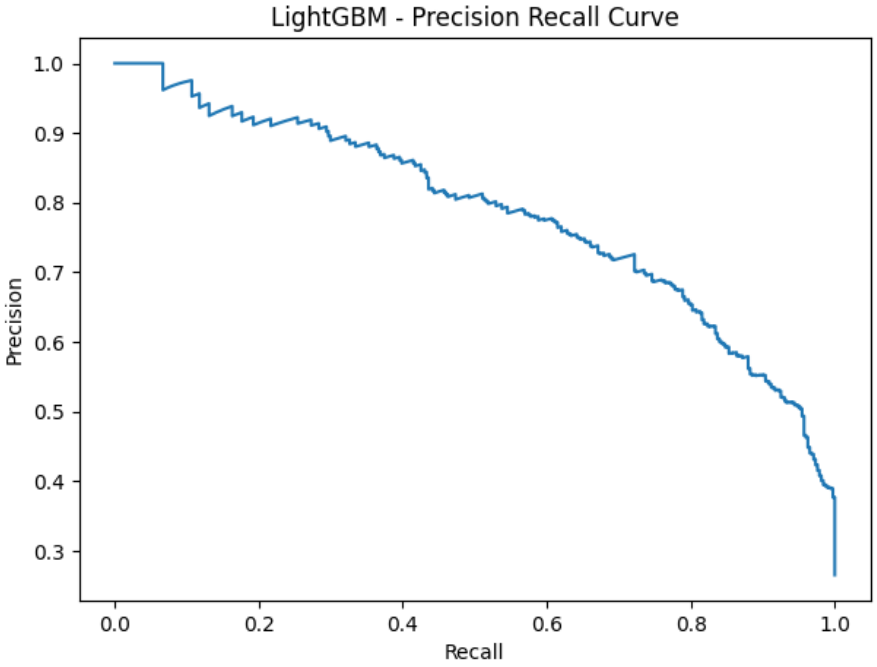


Рисунок 3.43 – PR-крива для LightGBM

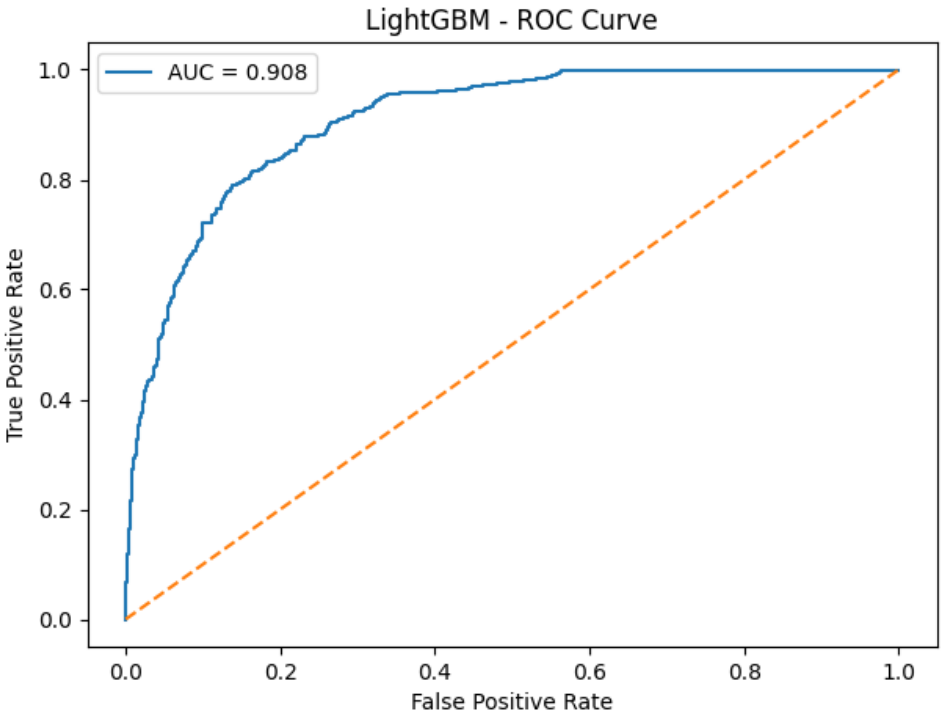


Рисунок 3.44 – ROC-крива для LightGBM

Отримані результати метрик буде зведено у єдину порівняльну таблицю. Ця таблиця показана на рисунку 3.45.

```
Порівняльна таблиця моделей машинного навчання
                Model  Accuracy  Precision    Recall  F1-Score   ROC-AUC  \
0  Logistic Regression  0.848119   0.748447  0.644385  0.692529  0.909804
5             LightGBM  0.848119   0.743902  0.652406  0.695157  0.908285
4              XGBoost  0.838183   0.729560  0.620321  0.670520  0.904154
1        Random Forest  0.843861   0.761905  0.598930  0.670659  0.895417
2                  SVM  0.831086   0.795652  0.489305  0.605960  0.886634
3                  kNN  0.792051   0.620896  0.556150  0.586742  0.822195

     PR-AUC
0  0.784307
5  0.785306
4  0.777447
1  0.755035
2  0.738978
3  0.588549
```

Рисунок 3.45 – Порівняльна таблиця моделей машинного навчання

З отриманих результатів можна побачити, що більшість моделей дають доволі високі результати (ROC-AUC у межах 0,9 – 0,91), а SVM та k-NN трохи поступаються іншим моделям. Найкращою моделлю виявилась логістична регресія, але їй мінімально поступається LightGBM (на 0,17%). Загалом топ-4 моделі (логістична регресія, LightGBM, Random Forest, XGBoost) дають доволі близький результат, оскільки Random Forest (топ-4) поступається логістичній регресії (топ-1) всього на 1,61% по ROC-AUC. Найслабшою моделлю виявилась k-NN, вона має найнижчі значення для всіх метрик, крім Recall, найнижчий Recall має SVM.

Також важливо оцінити, які ознаки виявилися найважливішими для моделей. Тож на рисунках 3.46–3.49 зображені графіки важливості ознак для моделей Logistic Regression, LightGBM, XGBoost, Random Forest.

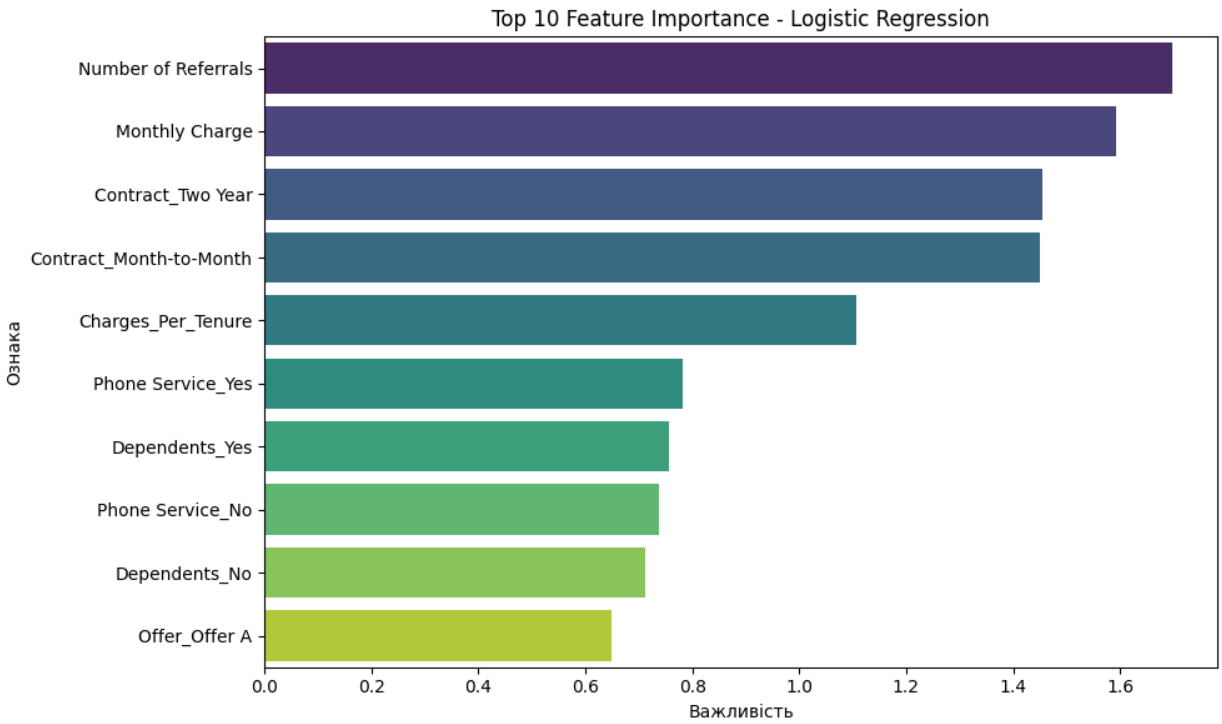


Рисунок 3.46 – Графік важливості ознак для моделі Logistic Regression

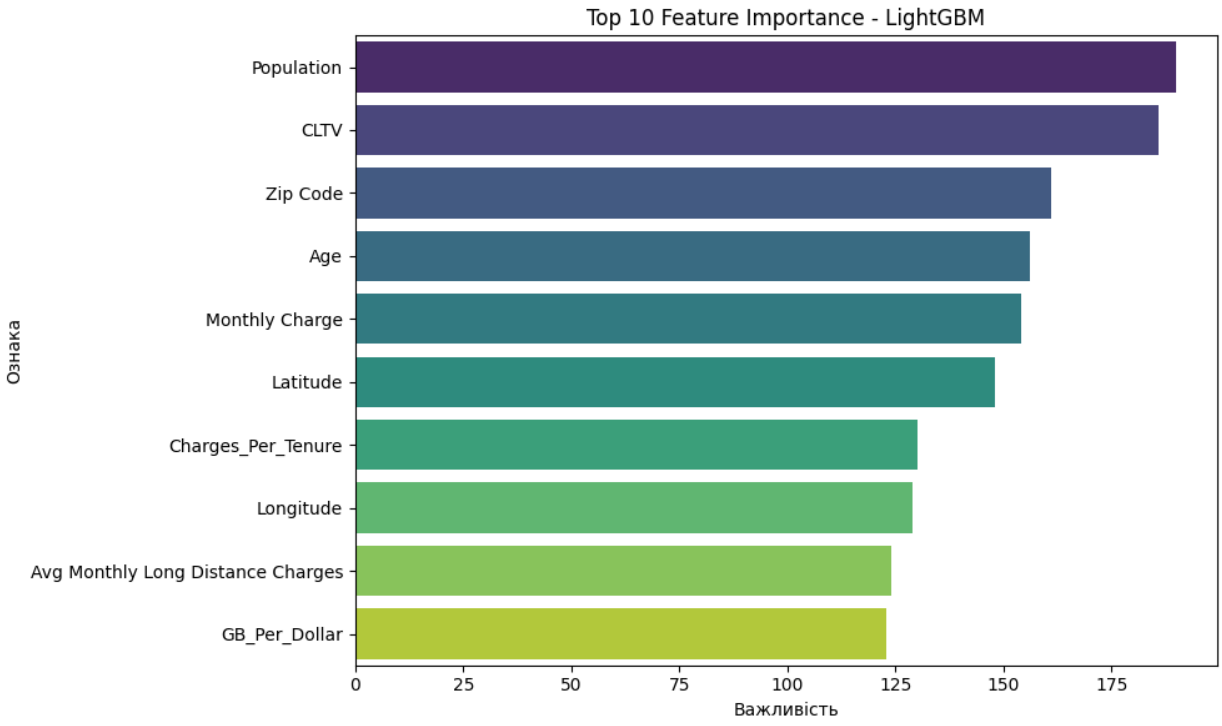


Рисунок 3.47 – Графік важливості ознак для моделі LightGBM

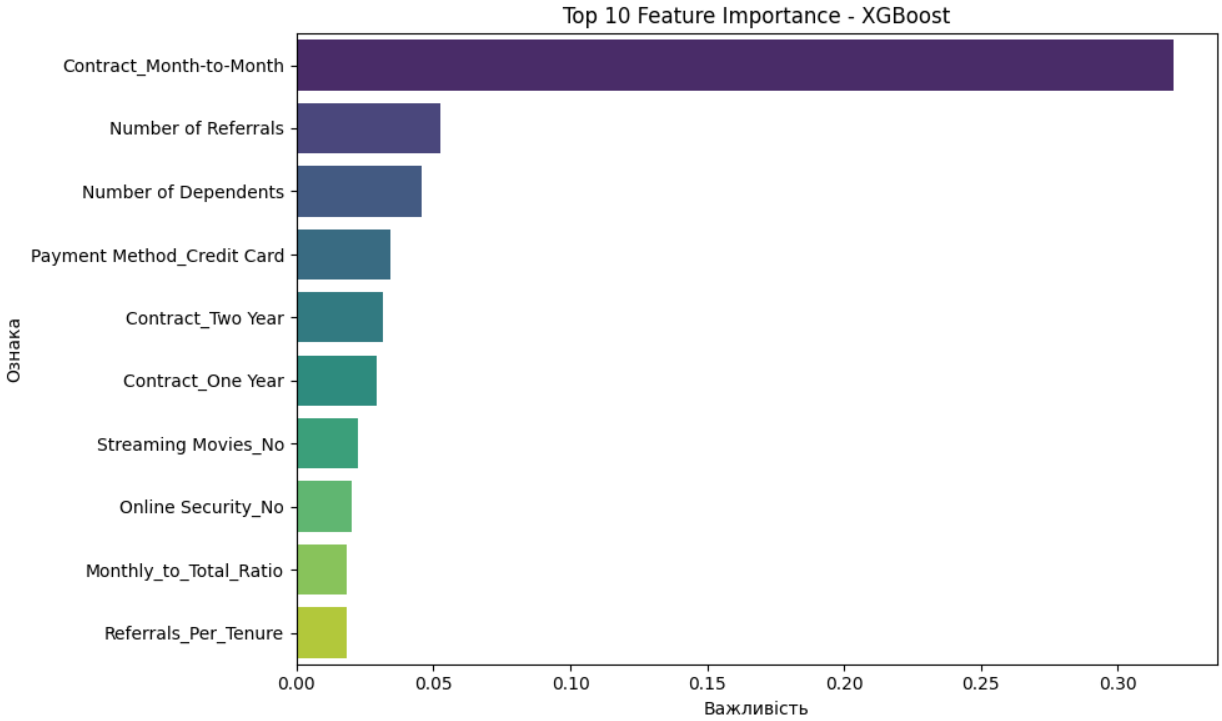


Рисунок 3.48 – Графік важливості ознак для моделі XGBoost

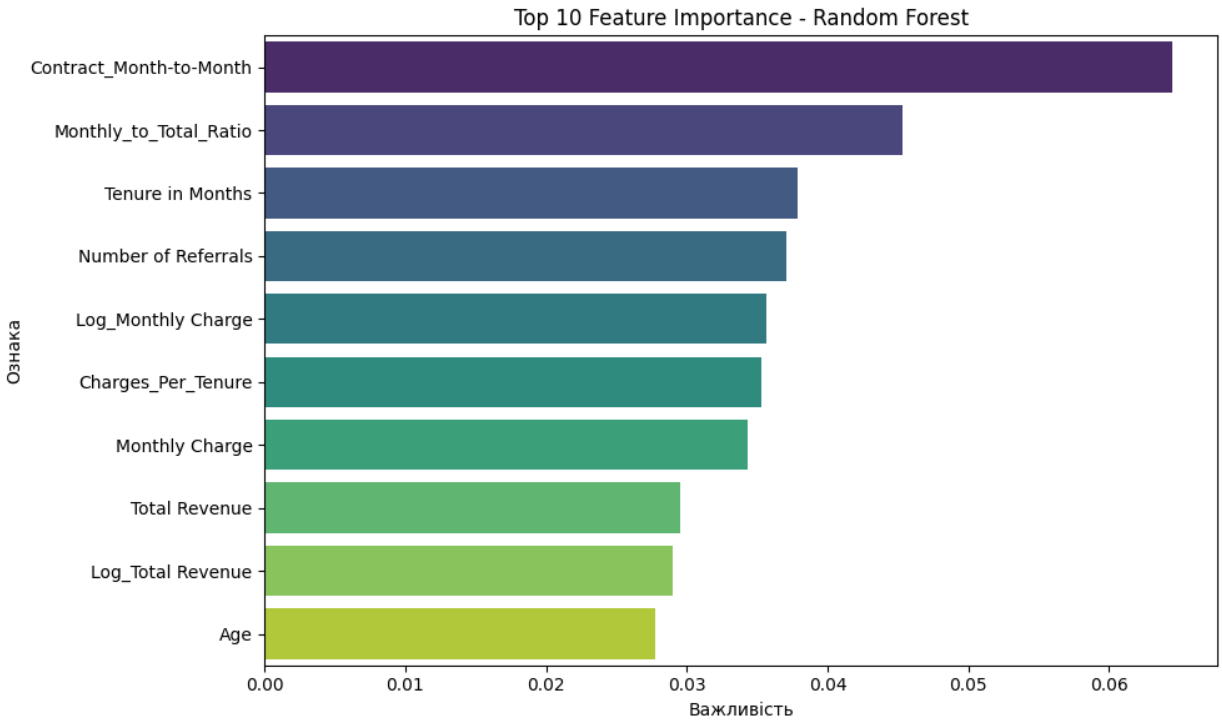


Рисунок 3.49 – Графік важливості ознак для моделі Random Forest

Можна побачити, що до топ-10 найважливіших ознак ввійшли ознаки, які були створені під час процесу попередньої обробки даних, підготовки даних та формування ознак, що вказує на те, що формування ознак пройшло успішно.

Додатково було виконано порівняльний аналіз результатів моделей без застосування таких методів, як PCA та SMOTE, з кожним з цих методів окремо та з двома методами одночасно. Результат порівняння показаний на таблиці 3.6.

Можна побачити, що застосування методу SMOTE дозволило підвищити Recall більшості моделей, однак у ряді випадків це знизило Precision. Використання PCA призвело до суттєвого погіршення результатів для всіх моделей, що свідчить про втрату важливої інформації під час зменшення розмірності ознак. Тож використання PCA для цих даних є недоцільним. Загалом найкращий баланс між всіма метриками забезпечує модель LightGBM із застосуванням SMOTE, яка дала найвищі значення ROC-AUC та PR-AUC – 0,9113 та 0,7873 відповідно.

Висновки до розділу 3

Розділ 3 присвячено практичній частині роботи: аналізу та обробці даних, їх підготовці для навчання моделей, побудові та оцінці шести моделей машинного навчання.

Для дослідження використано датасет Telco Customer Churn (IBM Cognos Analytics), що складається з п'яти таблиць. Після об'єднання таблиць і видалення неінформативних стовпців сформовано фінальний набір даних для навчання. Виявлено наявність пропущених значень у категоріальних змінних та помірний дисбаланс класів, що зумовило застосування відповідних методів обробки.

Таблиця 3.6 – Порівняльна таблиця методів (без PCA і SMOTE vs з PCA vs з SMOTE vs з PCA та SMOTE)

Model	*State*	*Accuracy*	*ROC-AUC*	*PR-AUC*	*Precision*	*Recall*	*F1-Score*
LightGBM	Baseline	0.8481	0.9083	0.7853	0.7439	0.6524	0.6952
	PCA	0.7544	0.7654	0.5458	0.5625	0.3369	0.4214
	PCA + SMOTE	0.6764	0.7637	0.5531	0.4347	0.7299	0.5449
	SMOTE	0.8531	0.9113	0.7873	0.7379	0.6925	0.7145
Logistic Regression	Baseline	0.8481	0.9098	0.7843	0.7484	0.6444	0.6925
	PCA	0.7346	0.5419	0.2807	0.0000	0.0000	0.0000
	PCA + SMOTE	0.3882	0.5361	0.2780	0.2878	0.8850	0.4344
	SMOTE	0.8155	0.9094	0.7828	0.6075	0.8610	0.7124
Random Forest	Baseline	0.8439	0.8954	0.7550	0.7619	0.5989	0.6707
	PCA	0.7580	0.7495	0.5156	0.5647	0.3850	0.4579
	PCA + SMOTE	0.6686	0.7370	0.5277	0.4168	0.6230	0.4995
	SMOTE	0.8474	0.9018	0.7502	0.7252	0.6845	0.7043
SVM	Baseline	0.8311	0.8866	0.7390	0.7957	0.4893	0.6060
	PCA	0.7346	0.6823	0.4180	0.0000	0.0000	0.0000
	PCA + SMOTE	0.6501	0.5826	0.3409	0.3501	0.3717	0.3606
	SMOTE	0.7715	0.8916	0.7466	0.5448	0.8449	0.6625
XGBoost	Baseline	0.8382	0.9042	0.7774	0.7296	0.6203	0.6705
	PCA	0.7466	0.7585	0.5390	0.5339	0.3583	0.4288
	PCA + SMOTE	0.6650	0.7465	0.5213	0.4175	0.6631	0.5124
	SMOTE	0.8474	0.9045	0.7785	0.7387	0.6578	0.6959
kNN	Baseline	0.7921	0.8222	0.5885	0.6209	0.5561	0.5867
	PCA	0.7388	0.7084	0.4463	0.5105	0.3904	0.4424
	PCA + SMOTE	0.6352	0.6989	0.4140	0.3903	0.6658	0.4921
	SMOTE	0.7090	0.8235	0.5618	0.4727	0.8342	0.6035

Підготовка датасету охопила формування ознак, заповнення пропусків модою, кодування категоріальних змінних та масштабування ознак. Додатково застосовано PCA та SMOTE.

За результатами навчання шести моделей встановлено, що більшість із них демонструє високі значення ROC-AUC у діапазоні 0,9 – 0,91. Найвищу ефективність показала логістична регресія, що несподівано перевершила ансамблеві методи на цьому наборі даних, хоча LightGBM із застосуванням SMOTE зміг перевершити логістину регресію і забезпечив найкращий баланс між усіма метриками – ROC-AUC 0,9113 та PR-AUC 0,7873. Найнижчі результати показав метод k-NN. Аналіз важливості ознак підтвердив ефективність формування ознак: серед топ-10 найважливіших присутні ті, що були створені на етапі формування ознак. Порівняльний аналіз показав, що SMOTE підвищує Recall більшості моделей, тоді як PCA призводить до погіршення результатів через втрату важливої інформації.

ВИСНОВКИ

У рамках цієї роботи проведено комплексне дослідження задачі прогнозування відтоку користувачів із застосуванням методів машинного навчання. Виконано аналіз предметної області, визначено сутність явища відтоку, його типи та бізнес-значення, а також сформульовано постановку задачі як задачі бінарної класифікації з дисбалансом класів. Проаналізовано сучасні підходи до прогнозування відтоку у SaaS, телекомунікаціях, банківській сфері та e-commerce. Встановлено, що ансамблеві методи на основі градієнтного бустингу (XGBoost, LightGBM) та Random Forest демонструють стабільно найвищі результати на табличних поведінкових даних, тоді як рекурентні нейронні мережі (LSTM, GRU) є ефективнішими для роботи з послідовними часовими даними. Систематизовано основні категорії ознак для задачі прогнозування відтоку – демографічні, поведінкові, RFM, часові та ознаки підтримки. Визначено сценарії залучення поведінкових характеристик, які будуть застосовані для цієї задачі.

Описано та математично обґрунтовано алгоритми машинного навчання з відповідними формулами та характеристиками. Детально розглянуто методи попередньої обробки даних із математичним формулюванням кожного: формування ознак, обробку пропущених значень, кодування категоріальних змінних, масштабування, PCA та SMOTE.

Розроблено систему прогнозування відтоку на основі відкритого набору даних Telco Customer Churn (IBM). Виконано підготовку даних, формування ознак, навчання шести моделей та їх оцінку за метриками Accuracy, Precision, Recall, F1-score, ROC-AUC та PR-AUC. За результатами порівняльного аналізу встановлено, що більшість моделей демонструє ROC-AUC у діапазоні 0,90–0,91, найвищий результат забезпечує LightGBM із застосуванням SMOTE (ROC-AUC – 0,9113, PR-AUC – 0,7873). Проведений порівняльний аналіз ефективності методів попередньої обробки показав, що

SMOTE підвищує Recall, тоді як PCA призводить до погіршення результатів на цьому наборі даних. Аналіз важливості ознак підтвердив ефективність процесу формування ознак.

Отримані результати підтверджують, що якість прогнозування відтоку залежить не лише від вибору алгоритму, а й від повноти та інформативності ознак і правильності попередньої обробки даних. Розроблений підхід може бути адаптований для інших галузей і даних шляхом відповідного налаштування набору ознак і методів попередньої обробки.

ПЕРЕЛІК ПОСИЛАНЬ

1. Verbeke W., Dejaeger K., Martens D., Hur J., Baesens B. New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. European Journal of Operational Research. 2012. Vol. 218, no. 1. P. 211–229. DOI: 10.1016/j.ejor.2011.09.031.
2. Hadden J., Tiwari A., Roy R., Ruta D. Computer assisted customer churn management: State-of-the-art and future trends. Computers & Operations Research. 2007. Vol. 34, no. 10. P. 2902–2917. DOI: 10.1016/j.cor.2005.11.007.
3. Lalwani P., Mishra M. K., Chadha J. S., Sethi P. Customer churn prediction system: a machine learning approach. Computing. 2021. Vol. 104, no. 8. P. 271–294. DOI: 10.1007/s00607-021-00908-y
4. He H., Garcia E. A. Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering. 2009. Vol. 21, no. 9. P. 1263–1284. DOI: 10.1109/TKDE.2008.239.
5. Reichheld F. F., Sasser W. E. Zero defections: Quality comes to services. Harvard Business Review. 1990. Vol. 68, no. 5. P. 105–111.
6. Gupta S., Lehmann D. R., Stuart J. A. Valuing customers. Journal of Marketing Research. 2004. Vol. 41, no. 1. P. 7–18. URL: https://business.columbia.edu/sites/default/files-efs/pubfiles/534/Valuing_Customers--JMR_2003.pdf (дата звернення: 28.04.2026).
7. Ascarza E. Retention futility: Targeting high-risk customers might be ineffective. Journal of Marketing Research. 2018. Vol. 55, no. 1. P. 80–98. DOI: 10.1509/jmr.16.0163.
8. Fader P. S., Hardie B. G. S., Lee K. L. "Counting your customers" the easy way: An alternative to the Pareto/NBD model. Marketing Science. 2005. Vol. 24, no. 2. P. 275–284. DOI: 10.1287/mksc.1040.0098.

9. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. 2nd ed. Sebastopol: O'Reilly Media, 2019. 851 p.
10. Breiman L. Random forests. Machine Learning. 2001. Vol. 45, no. 1. P. 5–32. DOI: 10.1023/A:1010933404324.
11. Rudd D. H., Huo H., Xu G. Causal Analysis of Customer Churn Using Deep Learning. International Conference on Digital Society and Intelligent Systems. 2021. DOI: 10.1109/DSInS54396.2021.9670561.
12. Imani M., Joudaki M., Beikmohammadi A., Arabnia H. R. Customer churn prediction: A systematic review of recent advances, trends, and challenges in machine learning and deep learning. Machine Learning and Knowledge Extraction. 2025. Vol. 7, no. 3. P. 105. DOI: 10.3390/make7030105.
13. Rautio J. Churn prediction in SaaS using machine learning. Master's thesis. Tampere University. 2019. 62 p. URL: https://trepo.tuni.fi/bitstream/handle/123456789/27579/Rautio.pdf?sequence=4 (дата звернення: 29.04.2026).
14. Ge Y., He S., Xiong J., Brown D. E. Customer churn analysis for a software-as-a-service company. Proceedings of IEEE Systems and Information Engineering Design Symposium (SIEDS). 2017. DOI: 10.1109/SIEDS.2017.7937698.
15. Khan M. R., Manoj J., Singh A., Blumenstock J. Behavioral modeling for churn prediction: Early indicators and accurate predictors of customer defection and loyalty. Proceedings of IEEE BigData. 2015. P. 1–4. DOI: 10.1109/BigDataCongress.2015.107.
16. The value of keeping the right customers. Harvard Business Review. URL: https://hbr.org/2014/10/the-value-of-keeping-the-right-customers (дата звернення: 28.04.2026).
17. Customer retention statistics 2026. SearchLab. URL: https://searchlab.nl/en/statistics/customer-retention-statistics-2026 (дата звернення: 28.04.2026).

18. What is churn rate. Checkout.com. URL: https://www.checkout.com/blog/what-is-churn-rate (дата звернення: 28.04.2026).
19. Hochreiter S., Schmidhuber J. Long Short-Term Memory. Neural Computation. 1997. Vol. 9, no. 8. P. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735
20. Provost F., Fawcett T. Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking. Sebastopol: O'Reilly Media, 2013. 409 p.
21. Chen T., Guestrin C. XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016. P. 785–794. DOI: 10.1145/2939672.2939785
22. Jolliffe I. T. Principal Component Analysis, Second Edition. Springer, 2002. 518 p.
23. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition. Springer, 2009. 764 p.
24. Sanches H. E., Possebom A. T., Aylon L. B. R. Churn prediction for SaaS company with machine learning. Innovation & Management Review. 2025. Vol. 22, no. 2. P. 130-142. DOI: 10.1108/INMR-06-2023-0101
25. Goodfellow I., Bengio Y., Courville A. Deep Learning. MIT Press, 2016. 800 p.
26. Krzanowski W. J., Hand D. J. ROC Curves for Continuous Data. CRC/Chapman and Hall, 2009. 226 p.
27. MinMax vs Standard vs Robust Scaler: Which One Wins for Skewed Data? Machine Learning Mastery. URL: https://machinelearningmastery.com/minmax-vs-standard-vs-robust-scaler-which-one-wins-for-skewed-data/ (дата звернення: 06.05.2026).

ДОДАТОК А ЛІСТИНГ ПРОГРАМИ

```
# 1. Імпортування бібліотек
import os
import warnings
warnings.filterwarnings("ignore")
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.base import clone
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import (
    RobustScaler,
    OneHotEncoder,
    TargetEncoder
)
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.decomposition import PCA
from sklearn.metrics import (
    accuracy_score,
    precision_score,
    recall_score,
    f1_score,
    roc_auc_score,
    average_precision_score,
    confusion_matrix,
    classification_report,
    roc_curve,
    precision_recall_curve
)
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
```

```
from xgboost import XGBClassifier
from imblearn.over_sampling import SMOTE
from lightgbm import LGBMClassifier

# 2. Завантаження датасету
print("Завантаження таблиць:")
demographics_df =
pd.read_excel("Telco_customer_churn_demographics.xlsx")
location_df =
pd.read_excel("Telco_customer_churn_location.xlsx")
population_df =
pd.read_excel("Telco_customer_churn_population.xlsx")
services_df =
pd.read_excel("Telco_customer_churn_services.xlsx")
status_df = pd.read_excel("Telco_customer_churn_status.xlsx")

print(f"Demographics: {demographics_df.shape[0]} рядків і
{demographics_df.shape[1]} стовпців")
print(f"Location: {location_df.shape[0]} рядків і
{location_df.shape[1]} стовпців")
print(f"Population: {population_df.shape[0]} рядків і
{population_df.shape[1]} стовпців")
print(f"Services: {services_df.shape[0]} рядків і
{services_df.shape[1]} стовпців")
print(f"Status: {status_df.shape[0]} рядків і
{status_df.shape[1]} стовпців")

# 3. Створення папки для збереження візуалізацій
os.makedirs("results", exist_ok=True)
os.makedirs("results/plots", exist_ok=True)

# 4. Видалення повторюваних стовпців
if "Count" in location_df.columns:
    location_df.drop(columns=["Count"], inplace=True)
if "Count" in services_df.columns:
    services_df.drop(columns=["Count"], inplace=True)
if "Count" in status_df.columns:
    status_df.drop(columns=["Count"], inplace=True)
if "Quarter" in status_df.columns:
```

```
    status_df.drop(columns=["Quarter"], inplace=True)
if "ID" in population_df.columns:
    population_df.drop(columns=["ID"], inplace=True)

# 5. Об'єднання таблиць
print("Об'єднання таблиць")
df = demographics_df.merge(location_df, on="Customer ID",
how="left")
df = df.merge(population_df, on="Zip Code", how="left")
df = df.merge(services_df, on="Customer ID", how="left")
df = df.merge(status_df, on="Customer ID", how="left")
print("Струткура об'днаної таблиці:", df.shape)
df.to_csv("results/final_merged_dataset.csv", index=False)

# 6. Аналіз датасету
print("Назви стовпців та їх типи даних")
print(df.info())

# 7. Видалення зайвих та неінформативних стовпців у датасеті
print("Видалення зайвих стовпців")
print(f"Розмірність датасету була: {df.shape[0]} рядків,
{df.shape[1]} стовпців\n")
columns_to_drop = [
    "Customer ID", "Lat Long", "Churn Label", "Customer
Status",
    "Churn Score", "Churn Category", "Churn Reason",
"Satisfaction Score"
]
existing_columns_to_drop = [col for col in columns_to_drop if
col in df.columns]

for col in existing_columns_to_drop:
    print(f" - {col}")
df.drop(columns=existing_columns_to_drop, inplace=True)
constant_columns = [col for col in df.columns if
df[col].nunique() <= 1]

if constant_columns:
    for col in constant_columns:
```

```
        print(f" - {col} (містить лише 1 унікальне значення)")
    df.drop(columns=constant_columns, inplace=True)
else:
    print(" - Констант не виявлено.")

print("Оновлена структура датасету")
df.info()
print("\n")
print("Статистика для числових стовпців")
stats =
df.select_dtypes(exclude=["object"]).drop(columns=["Count",
"Zip Code"], errors="ignore")
print(stats.describe().T.round(2).to_string())

# 8. Перевірка датасету на наявність пропущених значень
print("\n")
print("Перевірка наявності пропущених значень")
missing_values = df.isnull().sum()
missing_values = missing_values[missing_values > 0]

if not missing_values.empty:
    for col, count in
missing_values.sort_values(ascending=False).items():
        print(f"Стовпець '{col}' містить {count} пропущених
значень")

    sample_nan_indices = []
    print("\nФрагменти стовпців з пропущеними значеннями:")
    for col in missing_values.index:
        cols_to_show = list(dict.fromkeys([col]))
        cols_to_show = [c for c in cols_to_show if c in
df.columns]
        nan_sample =
df[df[col].isnull()][cols_to_show].head(10)
        sample_nan_indices.extend(nan_sample.index.tolist())
        print(nan_sample.to_string() + "\n")
    sample_nan_indices = list(set(sample_nan_indices))
else:
    print("Пропущених значень у датасеті не виявлено.")
```

```
    sample_nan_indices = []

# 9. Оцінка розподілу класів
target_column = "Churn Value"
print("\n")
print("Перевірка наявності дисбалансу класів")
counts = df[target_column].value_counts()
count_0 = counts.get(0, 0)
count_1 = counts.get(1, 0)
percentages = df[target_column].value_counts(normalize=True) *
100
pct_0 = percentages.get(0, 0.0)
pct_1 = percentages.get(1, 0.0)

print(f"Стовпець '{target_column}' містить {count_0} записи зі
значенням 0 "
      f"і {count_1} записів зі значенням 1, \nщо відповідно
становить "
      f"{pct_0:.2f}% і {pct_1:.2f}% від загальної кількості
записів.")
plt.figure(figsize=(6, 5))
sns.countplot(x=target_column, data=df)
plt.title("Churn Class Distribution")
plt.savefig("results/plots/class_distribution.png")
plt.close()

# 10. Формування ознак
print("Препроцесинг")

# 10.1 Логарифмічне перетворення
columns_to_log = ["Monthly Charge", "Total Charges", "Total
Long Distance Charges", "Total Revenue", "Population"]
for col in columns_to_log:
    if col in df.columns:
        df[f"Log_{col}"] = np.log1p(df[col])

# 10.2 Створення відносних ознак
if "Total Charges" in df.columns and "Tenure in Months" in
df.columns:
```

```
    safe_tenure = np.where(df["Tenure in Months"] == 0, 1,
df["Tenure in Months"])
    df["Charges_Per_Tenure"] = df["Total Charges"] /
safe_tenure

if "Monthly Charge" in df.columns and "Total Charges" in
df.columns:
    safe_total = np.where(df["Total Charges"] == 0, 1,
df["Total Charges"])
    df["Monthly_to_Total_Ratio"] = df["Monthly Charge"] /
safe_total

if "Total Extra Data Charges" in df.columns and "Total Charges"
in df.columns:
    safe_total = np.where(df["Total Charges"] == 0, 1,
df["Total Charges"])
    df["Extra_Charges_Ratio"] = df["Total Extra Data Charges"]
/ safe_total

if "Avg Monthly GB Download" in df.columns and "Monthly Charge"
in df.columns:
    safe_monthly = np.where(df["Monthly Charge"] == 0, 1,
df["Monthly Charge"])
    df["GB_Per_Dollar"] = df["Avg Monthly GB Download"] /
safe_monthly

if "Number of Referrals" in df.columns and "Tenure in Months"
in df.columns:
    safe_tenure = np.where(df["Tenure in Months"] == 0, 1,
df["Tenure in Months"])
    df["Referrals_Per_Tenure"] = df["Number of Referrals"] /
safe_tenure

# 10.3 Створення ознаки, чи клієнт є новим
if "Tenure in Months" in df.columns:
    df["Is_New_Customer"] = (df["Tenure in Months"] <=
12).astype(int)
```

```
# 10.4 Створення ознаки, яка описує кількість сервісів, якими користується клієнт
service_columns = [
    "Phone Service", "Multiple Lines", "Internet Service", "Online Security",
    "Online Backup", "Device Protection Plan", "Premium Tech Support",
    "Streaming TV", "Streaming Movies", "Streaming Music", "Unlimited Data"
]
existing_service_columns = [col for col in service_columns if col in df.columns]
df["NumberOfServicesUsed"] = 0

for col in existing_service_columns:
    df["NumberOfServicesUsed"] += (
        df[col].astype(str).isin(["Yes", "DSL", "Fiber Optic", "Cable"]).astype(int)
    )

# 11. Поділ даних на ознаки та цільову змінну
X = df.drop(columns=[target_column])
y = df[target_column]

# 12. Визначення типу стовпця: числовий, категоріальний з високою кількістю категорій, категоріальний з малою кількістю категорій
high_cardinality_features = ["City", "Zip Code"]
high_cardinality_cols = [col for col in high_cardinality_features if col in X.columns]
low_cardinality_cols = [
    col for col in X.select_dtypes(include=["object"]).columns
    if col not in high_cardinality_cols
]
numerical_columns = X.select_dtypes(exclude=["object"]).columns.tolist()

# 13. Розбиття на тренувальні та тестові дані
X_train, X_test, y_train, y_test = train_test_split(
```

```
    X, y, test_size=0.2, stratify=y, random_state=42
)

# 14. Заповнення пропусків, кодування категоріальних ознак та
масштабування
numerical_pipeline = Pipeline([
    ("imputer", SimpleImputer(strategy="median",
add_indicator=True)),
    ("scaler", RobustScaler())
])
categorical_ohe_pipeline = Pipeline([
    ("imputer", SimpleImputer(strategy="most_frequent",
add_indicator=True)),
    ("encoder", OneHotEncoder(handle_unknown="ignore"))
])
categorical_te_pipeline = Pipeline([
    ("imputer", SimpleImputer(strategy="most_frequent",
add_indicator=True)),
    ("encoder", TargetEncoder(smooth="auto"))
])
transformers = [
    ("num", numerical_pipeline, numerical_columns),
    ("cat_ohe", categorical_ohe_pipeline, low_cardinality_cols)
]

if high_cardinality_cols:
    transformers.append(("cat_te", categorical_te_pipeline,
high_cardinality_cols))
preprocessor = ColumnTransformer(transformers)

# 15. Демонстрація результатів препроцесингу
X_train_processed = preprocessor.fit_transform(X_train,
y_train)
X_test_processed = preprocessor.transform(X_test)
all_feature_names = preprocessor.get_feature_names_out()
clean_feature_names = [name.split('__')[-1] for name in
all_feature_names]
X_train_processed_df = pd.DataFrame(
    X_train_processed,
```

```
    columns=clean_feature_names,
    index=X_train.index
)

# 15.1 Демонстрація різних методів формування ознак
print("\n")
print("1. Формування ознак (feature engineering)")

# 15.1.1 Логарифмічне перетворення
print("\n1.1 Логарифмічне перетворення")
cols_to_log = ["Monthly Charge", "Total Charges", "Total Long
Distance Charges", "Total Revenue", "Population"]
existing_log_base = [col for col in cols_to_log if col in
X_train.columns]
print("До логарифмічного перетворення:")
print(X_train[existing_log_base].head(10).to_string())
log_cols = [f"Log_{col}" for col in existing_log_base if
f"Log_{col}" in X_train.columns]
print("\nПісля логарифмічного перетворення")
print(X_train[log_cols].head(10).round(4).to_string())

# 15.1.2 Створення відносних ознак
print("\n1.2 Відносні ознаки")
ratio_base_cols = ["Monthly Charge", "Total Charges", "Tenure
in Months", "Total Extra Data Charges", "Avg Monthly GB
Download", "Number of Referrals"]
existing_ratio_base = [col for col in ratio_base_cols if col in
X_train.columns]
print("До створення відносних ознак:")
print(X_train[existing_ratio_base].head(10).to_string())
ratio_cols = ['Charges_Per_Tenure', 'Monthly_to_Total_Ratio',
'Extra_Charges_Ratio', 'GB_Per_Dollar', 'Referrals_Per_Tenure']
existing_ratio_cols = [col for col in ratio_cols if col in
X_train.columns]
print("\nПісля створення відносних ознак:")
print(X_train[existing_ratio_cols].head(10).round(4).to_string(
))

# 15.1.3 Сегментація клієнтів
```

```
print("\n1.3 Сегментація клієнтів")
if "Tenure in Months" in X_train.columns:
    print("До сегментації клієнтів:")
    print(X_train[['Tenure in Months']].head(10).to_string())

if "Is_New_Customer" in X_train.columns:
    print("\nПісля сегментації клієнтів ('Новий клієнт' <= 12
місяців):")
    print(X_train[['Is_New_Customer']].head(10).to_string())

# 15.1.3 Підрахунок кількості сервісів, якими користується
клієнт
print("\n1.4 Кількість сервісів, якими користується клієнт")
# Показуємо лише кілька послуг для прикладу, щоб таблиця не
розповзлася на весь екран
services_base_demo = ["Phone Service", "Internet Service",
"Online Security", "Streaming TV"]
existing_services_demo = [col for col in services_base_demo if
col in X_train.columns]
print("Статуси окремих сервісів:")
print(X_train[existing_services_demo].head(10).to_string())

if 'NumberOfServicesUsed' in X_train.columns:
    print("\nАгрегована кількість активних сервісів:")

print(X_train[['NumberOfServicesUsed']].head(10).to_string())

# 15.2 Обробка пропущених значень
print("\n")
print("2. Заповнення пропусків")
missing_train = X_train.isnull().sum()
cols_with_nans = missing_train[missing_train >
0].index.tolist()

if not cols_with_nans:
    print("Пропусків у тренувальній вибірці не виявлено.")
else:
    for col in cols_with_nans:
        missing_count = missing_train[col]
```

```
        print(f"\nСтовпець: {col}")
        print(f"Кількість пропусків у тренувальній вибірці до
заповнення: {missing_count}")
        nan_indices =
X_train[X_train[col].isnull()].head(10).index
        print("До заповнення пропусків:")
        print(X_train.loc[nan_indices, [col]].to_string())

        if X_train[col].dtype == 'object':
            fill_val = X_train[col].mode()[0]
        else:
            fill_val = X_train[col].median()

        print(f"\nПісля заповнення пропусків:")
        demo_imputed = X_train.loc[nan_indices, [col]].copy()
        demo_imputed[col] = demo_imputed[col].fillna(fill_val)
        print(demo_imputed.to_string())

missing_after = X_train_processed_df.isnull().sum().sum()
print(f"\nЗагальна кількість пропусків у всьому датасеті після
відпрацювання Pipeline: {missing_after}")

# 15.3 Кодування категоріальних ознак
print("\n3. Кодування категоріальних ознак")
print("До кодування:")
print(X_train[['Contract']].head(10).to_string())
print("\nПісля кодування:")
contract_cols = [col for col in X_train_processed_df.columns if
'Contract' in col]
print(X_train_processed_df.loc[X_train.head(10).index,
contract_cols].to_string())

# 15.4 Масштабування
print("\n4. Масштабування")
print("До масштабування:")
print(X_train[['Total Charges']].head(10).to_string())
print("\nПісля масштабування:")
print(X_train_processed_df.loc[X_train.head(10).index, ['Total
Charges']].round(4).to_string())
```

```
# 16. PCA
print("\n")
print("PCA Analysis")

if hasattr(X_train_processed, "toarray"):
    X_train_dense = X_train_processed.toarray()
    X_test_dense = X_test_processed.toarray()
else:
    X_train_dense = X_train_processed
    X_test_dense = X_test_processed

pca = PCA(n_components=0.95, random_state=42)
X_train_pca = pca.fit_transform(X_train_dense)
X_test_pca = pca.transform(X_test_dense)
explained_variance = pca.explained_variance_ratio_
cumulative_variance = np.cumsum(explained_variance)
print(f"Кількість PCA-компонент: {pca.n_components_}")
print(f"Пояснений коефіцієнт дисперсії:
{np.round(explained_variance, 4)}")
n_components_to_show = min(3, pca.n_components_)

for i in range(n_components_to_show):
    component_weights = pca.components_[i]
    weights_df = pd.DataFrame({
        'Feature': clean_feature_names,
        'Weight': component_weights,
        'Abs_Weight': np.abs(component_weights)
    })
    top_features = weights_df.sort_values(by='Abs_Weight',
ascending=False).head(5)
    variance_pct = explained_variance[i] * 100
    print(f"\n[Головна Компонента {i+1}] - Пояснює
{variance_pct:.2f}% дисперсії:")
    for _, row in top_features.iterrows():
        print(f"  {row['Feature']:<35} :
{row['Weight']:>8.4f}")

plt.figure(figsize=(10, 6))
```

```
plt.plot(range(1, len(cumulative_variance) + 1),
cumulative_variance, marker='o', linestyle='--')
plt.axhline(y=0.95, color='r', linestyle='-')
plt.xlabel("Кількість компонент")
plt.ylabel("Кумулятивна пояснена дисперсія")
plt.title("PCA: Збереження інформації при стисненні")
plt.grid(True)
plt.savefig("results/plots/pca_explained_variance.png")
plt.close()

# 17. SMOTE
print("\n")
print("SMOTE Analysis")
counts_before = pd.Series(y_train).value_counts()
count_0_before = counts_before.get(0, 0)
count_1_before = counts_before.get(1, 0)
print("До застосування SMOTE:")
print(f"Стовпець 'Churn Value' містить {count_0_before} записи
зі значенням 0 "
      f"і {count_1_before} записів зі значенням 1.")
smote = SMOTE(random_state=42)
X_train_smote, y_train_smote =
smote.fit_resample(X_train_processed, y_train)
counts_after = pd.Series(y_train_smote).value_counts()
count_0_after = counts_after.get(0, 0)
count_1_after = counts_after.get(1, 0)
print("\nПісля застосування SMOTE:")
print(f"Стовпець 'Churn Value' містить {count_0_after} записи
зі значенням 0 "
      f"і {count_1_after} записів зі значенням 1.")

# 18. Побудова моделей
models = {
    "Logistic Regression": LogisticRegression(max_iter=1000,
random_state=42),
    "Random Forest": RandomForestClassifier(n_estimators=200,
random_state=42),
    "SVM": SVC(probability=True, random_state=42),
    "kNN": KNeighborsClassifier(n_neighbors=5),
```

```
    "XGBoost": XGBClassifier(eval_metric="logloss",
random_state=42),
    "LightGBM": LGBMClassifier(random_state=42)
}

# 19. Функція для оцінки моделей
results = []
trained_models = {}

def evaluate_model(model, X_train_data, y_train_data,
X_test_data, y_test_data, model_name):
    model.fit(X_train_data, y_train_data)
    y_pred = model.predict(X_test_data)

    if hasattr(model, "predict_proba"):
        y_prob = model.predict_proba(X_test_data)[:, 1]
    elif hasattr(model, "decision_function"):
        y_prob = model.decision_function(X_test_data)
    else:
        y_prob = y_pred # Запасний варіант

    accuracy = accuracy_score(y_test_data, y_pred)
    precision = precision_score(y_test_data, y_pred,
zero_division=0)
    recall = recall_score(y_test_data, y_pred, zero_division=0)
    f1 = f1_score(y_test_data, y_pred, zero_division=0)
    roc_auc = roc_auc_score(y_test_data, y_prob)
    pr_auc = average_precision_score(y_test_data, y_prob)
    results.append({
        "Model": model_name,
        "Accuracy": accuracy,
        "Precision": precision,
        "Recall": recall,
        "F1-Score": f1,
        "ROC-AUC": roc_auc,
        "PR-AUC": pr_auc
    })
    print(f"\n{model_name}\n")
    print(f"Accuracy: {accuracy:.3f}")
```

```
    print(f"ROC-AUC: {roc_auc:.3f}")
    print(f"PR-AUC: {pr_auc:.3f}")
    print(classification_report(y_test_data, y_pred,
zero_division=0))

    cm = confusion_matrix(y_test_data, y_pred)
    plt.figure(figsize=(6, 5))
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
    plt.title(f"{model_name} - Confusion Matrix")
    plt.xlabel("Predicted")
    plt.ylabel("Actual")
    plt.savefig(f"results/plots/{model_name.replace(' ',
'_').lower()}_confusion_matrix.png")
    plt.close()

    fpr, tpr, _ = roc_curve(y_test_data, y_prob)
    plt.figure(figsize=(7, 5))
    plt.plot(fpr, tpr, label=f"AUC = {roc_auc:.3f}")
    plt.plot([0, 1], [0, 1], linestyle="--")
    plt.xlabel("False Positive Rate")
    plt.ylabel("True Positive Rate")
    plt.title(f"{model_name} - ROC Curve")
    plt.legend()
    plt.savefig(f"results/plots/{model_name.replace(' ',
'_').lower()}_roc_curve.png")
    plt.close()

    precision_curve, recall_curve, _ =
precision_recall_curve(y_test_data, y_prob)
    plt.figure(figsize=(7, 5))
    plt.plot(recall_curve, precision_curve)
    plt.xlabel("Recall")
    plt.ylabel("Precision")
    plt.title(f"{model_name} - Precision Recall Curve")
    plt.savefig(f"results/plots/{model_name.replace(' ',
'_').lower()}_pr_curve.png")
    plt.close()

    return model
```

```
# 20. Тренування та оцінка моделей
print("\n")
print("Тренування моделей")

for model_name, model in models.items():
    trained_model = evaluate_model(
        model,
        X_train_processed,
        y_train,
        X_test_processed,
        y_test,
        model_name
    )
    trained_models[model_name] = trained_model

# 21. Порівняльна таблиця моделей
results_df = pd.DataFrame(results)
results_df = results_df.sort_values(by="ROC-AUC",
ascending=False)
print("\n")
print("Порівняльна таблиця моделей машинного навчання")
print(results_df)
results_df.to_csv("results/model_metrics.csv", index=False)

# 22. Аналіз важливості ознак
print("\n")
print("Аналіз важливості ознак")
all_feature_names = preprocessor.get_feature_names_out()
clean_feature_names = [name.split('__')[-1] for name in
all_feature_names]

for model_name, model in trained_models.items():
    importance_values = None

    if hasattr(model, 'feature_importances_'):
        importance_values = model.feature_importances_
    elif hasattr(model, 'coef_'):
        importance_values = np.abs(model.coef_[0])
```

```
    if importance_values is not None:
        if len(clean_feature_names) == len(importance_values):
            importance_df = pd.DataFrame({
                "Feature": clean_feature_names,
                "Importance": importance_values
            }).sort_values(by="Importance",
ascending=False).head(10)
            print(f"\nТоп-10 найважливіших ознак для моделі:
{model_name}")
            print("-" * 50)
            print(importance_df.to_string(index=False))
            plt.figure(figsize=(10, 6))
            sns.barplot(x="Importance", y="Feature",
data=importance_df, hue="Feature", legend=False,
palette="viridis")
            plt.title(f"Top 10 Feature Importance -
{model_name}")
            plt.xlabel("Важливість")
            plt.ylabel("Ознака")
            plt.tight_layout()
            safe_name = model_name.replace(" ", "_").lower()

plt.savefig(f"results/plots/{safe_name}_feature_importance.png"
)
            plt.close()
        else:
            print(f"\nНеможливо вивести ознаки для
{model_name}: кількість імен ({len(clean_feature_names)}) не
збігається з кількістю ваг ({len(importance_values)}).")

# 23. Порівняльний аналіз застосування PCA та SMOTE
print("\n" + "=" * 70)
print("Порівняльний аналіз: без PCA і SMOTE vs з PCA vs з SMOTE
vs PCA + SMOTE")
print("=" * 70)
pca_for_smote = PCA(n_components=0.95, random_state=42)
X_train_smote_dense = X_train_smote.toarray() if
hasattr(X_train_smote, "toarray") else X_train_smote
```

```
X_test_dense = X_test_processed.toarray() if
hasattr(X_test_processed, "toarray") else X_test_processed
X_train_both = pca_for_smote.fit_transform(X_train_smote_dense)
X_test_both = pca_for_smote.transform(X_test_dense)

def get_proba(model, X):
    if hasattr(model, "predict_proba"): return
model.predict_proba(X)[:, 1]
    elif hasattr(model, "decision_function"): return
model.decision_function(X)
    else: return model.predict(X)

def calc_metrics(y_true, y_pred, y_prob):
    return {
        "Accuracy": round(accuracy_score(y_true, y_pred), 4),
        "ROC-AUC": round(roc_auc_score(y_true, y_prob), 4),
        "PR-AUC": round(average_precision_score(y_true,
y_prob), 4),
        "Precision": round(precision_score(y_true, y_pred,
zero_division=0), 4),
        "Recall": round(recall_score(y_true, y_pred,
zero_division=0), 4),
        "F1-Score": round(f1_score(y_true, y_pred,
zero_division=0), 4)
    }
all_results = []

for model_name, base_model in models.items():
    model_base = clone(base_model).fit(X_train_processed,
y_train)
    metrics_base = calc_metrics(y_test,
model_base.predict(X_test_processed), get_proba(model_base,
X_test_processed))
    all_results.append({"Model": model_name, "State":
"Baseline", **metrics_base})
    model_pca = clone(base_model).fit(X_train_pca, y_train)
    metrics_pca = calc_metrics(y_test,
model_pca.predict(X_test_pca), get_proba(model_pca,
X_test_pca))
```

```
    all_results.append({"Model": model_name, "State": "PCA",
**metrics_pca})
    model_smote = clone(base_model).fit(X_train_smote,
y_train_smote)
    metrics_smote = calc_metrics(y_test,
model_smote.predict(X_test_processed), get_proba(model_smote,
X_test_processed))
    all_results.append({"Model": model_name, "State": "SMOTE",
**metrics_smote})
    model_both = clone(base_model).fit(X_train_both,
y_train_smote)
    metrics_both = calc_metrics(y_test,
model_both.predict(X_test_both), get_proba(model_both,
X_test_both))
    all_results.append({"Model": model_name, "State": "PCA +
SMOTE", **metrics_both})

full_results_df = pd.DataFrame(all_results)
full_results_df = full_results_df.sort_values(by=["Model",
"State"]).set_index(["Model", "State"])
print(full_results_df.to_string())
full_results_df.to_csv("results/comprehensive_ablation_study.cs
v")
```