Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)
Навчально-науковий інститут комп'ютерних наук
та інформаційних технологій
(повне найменування інституту, назва факультету (відділення))
Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

Магістерська кваліфікаційна робота

другий (магістерський)
(рівень вищої освіти)

на тему: Інтелектуальна система опрацювання великої кількості даних

Виконав: студент VI курсу групи КН-62м
Спеціальності: 122 "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)

Гатала Б.І.
(прізвище та ініціали)

Керівник Процах Н.П.
(прізвище та ініціали)

Рецензент Карашецький В.П.
(прізвище та ініціали)

Львів – 2025

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІ комп`ютерних наук та інформаційних технологій
Кафедра комп'ютерних наук
Рівень вищої освіти другий (магістерський)
Спеціальність 122 "Комп'ютерні науки"
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри комп`ютерних наук
_____ Борецька І.Б.
"10" грудня 2025 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Гатала Богдан Іванович
(прізвище, ім'я, по батькові)

1. Тема роботи: Інтелектуальна система опрацювання великої кількості даних

Керівник роботи *д.т.н., професор Процах Н.П.*,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від "29" квітня 2025 року №С-288

2. Термін подання студентом роботи 10 грудня 2025 року

3. Вихідні дані до роботи: створення інтелектуальної системи опрацювання великої кількості даних

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

Розділ 1. Стан проблемної області

Розділ 2. Інформаційне забезпечення

Розділ 3. Математичне забезпечення

Розділ 4. Програмне забезпечення

Розділ 5. Розроблення стартап-проєкту

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Підготовка матеріалу до доповіді

6. Дата видачі завдання 1 травня 2025 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1.	Огляд літератури згідно досліджуваної теми. Збір необхідних матеріалів.	1.05.25-10.06.25	Виконано
2.	Постановка задачі і її формалізація	21.06.25-30.06.25	Виконано
3.	Виконання вхідного етапу технології	01.07.25 – 31.07.25	Виконано
4.	Реалізація головних алгоритмів проєкту	01.08.25 – 30.09.25	Виконано
5.	Виконання етапу відлагодження проєкту	01.10.25 – 31.10.25	Виконано
6.	Виконання етапу впровадження та випуску бета-версії.	01.11.25 – 15.11.25	Виконано
7.	Оформлення записки до дипломного проекту.	16.11.25 – 5.12.25	Виконано

Студент _____ Гатала Б.І.
(підпис) (прізвище та ініціали)

Керівники роботи _____ Процах Н.П.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Дипломна робота присвячена розробці інтелектуальної CRM-системи для аналізу великих клієнтських даних у сфері автомобільного бізнесу. Система поєднує можливості багатопотокового опрацювання інформації, машинного навчання та пояснюваного штучного інтелекту (Explainable AI). Для демонстрації можливостей програмного забезпечення використано CMS October та фреймворк Laravel.

Розроблено гнучкий механізм обробки великих обсягів структурованих і неструктурованих даних з використанням файлів типу CSV. Інтелектуальний модуль системи реалізує побудову прогнозів і пояснення результатів за допомогою алгоритмів машинного навчання та бібліотеки SHAP, що забезпечує прозорість ухвалених рішень.

Створено адаптивний інтерфейс у стилі e-commerce, орієнтований на потреби автомобільних дилерів. Розроблена система може функціонувати як SaaS-платформа, що забезпечує масштабованість та комерційне використання.

Ключові слова: CRM, Laravel, CMS October, Big Data, машинне навчання, Explainable AI, SHAP, програмне забезпечення.

ABSTRACT

The diploma project is devoted to the development of an intelligent CRM system for analyzing large-scale customer data in the automotive industry. The system combines the capabilities of multithreaded data processing, machine learning, and explainable artificial intelligence (Explainable AI). To demonstrate the functionality of the software, CMS October and the Laravel framework were used.

A flexible mechanism for processing large volumes of structured and unstructured data was developed using CSV file formats. The intelligent module of the system implements prediction generation and result interpretation through machine learning algorithms and the SHAP library, which provides transparency of the decision-making process.

An adaptive e-commerce–style interface tailored to the needs of automotive dealers has been created. The developed system can function as a SaaS platform, ensuring scalability and commercial applicability.

Keywords: CRM, Laravel, CMS October, Big Data, machine learning, Explainable AI, SHAP, software.

ТЕХНІЧНЕ ЗАВДАННЯ

Розробити інтелектуальну CRM-систему для аналізу великих клієнтських даних у сфері автомобільного бізнесу, що поєднує технології багатопотокового опрацювання, машинного навчання та пояснюваного штучного інтелекту (Explainable AI).

Програмне забезпечення має забезпечувати імпорт, обробку та експорт великих обсягів структурованих і неструктурованих даних з використанням розпаралелення процесів у рамках єдиного Data Pipeline.

Для реалізації проєкту використати фреймворк Laravel як основну серверну платформу та CMS October для демонстрації можливостей інтерфейсу.

Розробити інтелектуальний аналітичний модуль, який застосовує алгоритми машинного навчання для прогнозування ключових показників і бібліотеку SHAP для пояснення результатів моделі (Explainable AI).

Створити адаптивну веборієнтовану систему з сучасним інтерфейсом у стилі e-commerce, оптимізовану для роботи на різних пристроях.

Реалізувати кросплатформенний застосунок із використанням технології Progressive Web App (PWA) для забезпечення доступності, масштабованості та можливості SaaS-використання.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

AI (Artificial Intelligence) — штучний інтелект, галузь інформатики, що займається створенням систем, здатних виконувати завдання, які потребують людського інтелекту.

API (Application Programming Interface) — прикладний програмний інтерфейс, набір засобів для взаємодії між програмними компонентами.

Big Data — великі дані; надзвичайно об’ємні, різнорідні та швидкоплинні набори даних, що потребують спеціальних методів зберігання та аналізу.

CMS (Content Management System) — система керування контентом, що використовується для розробки та адміністрування вебзастосунків.

CRM (Customer Relationship Management) — система керування взаємовідносинами з клієнтами; програмне забезпечення для обліку, аналізу й автоматизації бізнес-процесів.

CSV (Comma-Separated Values) — текстовий формат обміну даними, у якому поля розділяються комами або іншими роздільниками.

ETL (Extract, Transform, Load) — процес отримання, перетворення та завантаження даних у систему зберігання або аналітичну платформу.

Explainable AI (XAI) — пояснюваний штучний інтелект, підхід у машинному навчанні, що дозволяє інтерпретувати рішення моделей.

JSON (JavaScript Object Notation) — формат обміну даними, зручний для зберігання та передавання структурованої інформації.

ML (Machine Learning) — машинне навчання, розділ штучного інтелекту, який вивчає методи побудови моделей, здатних навчатися на даних.

PWA (Progressive Web App) — прогресивний вебзастосунок, технологія створення кросплатформенних вебінтерфейсів із можливістю роботи в офлайн-режимі.

ВСТУП

Розвиток автомобільного бізнесу потребує глибокої аналітики клієнтських даних для підвищення ефективності продажів і рівня сервісу. Зі зростанням обсягів інформації у CRM-системах традиційні методи її обробки стають повільними й обмеженими. Щоденне оновлення великих клієнтських баз, історій покупок і сервісних звернень створює надмірне навантаження на адміністраторів і призводить до втрати важливих бізнес-інсайтів.

Тому пропонується інтелектуальна CRM-платформа, яка поєднує багатопотоковість для швидкого опрацювання даних, машинне навчання для прогнозування поведінки клієнтів і технологію Explainable AI (SHAP) для пояснення результатів аналізу. Це дозволяє не лише автоматизувати імпорт та експорт великих масивів структурованих і неструктурованих даних, а й отримувати зрозумілі аналітичні висновки в режимі реального часу.

Об’єктом дослідження є процес обробки великих клієнтських даних у сфері автомобільного бізнесу.

Метою роботи є створення інтелектуальної системи, що реалізує багатопотокове опрацювання даних із застосуванням машинного навчання та SHAP-аналізу.

Предметом дослідження є методи аналітики та пояснюваного штучного інтелекту в Data Pipeline CRM-системи.

Практичне значення полягає у прискоренні обробки даних і підвищенні точності бізнес-аналітики.

Наукова новизна – у поєднанні багатопоточних обчислень із Explainable AI для створення прозорої CRM-платформи нового покоління.

Актуальність теми зумовлена зростанням потреби бізнесу в інтелектуальних інструментах Big Data-аналітики.

РОЗДІЛ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1. Сучасні підходи до обробки великих даних

У 2000–2010 роках сформувалися основи Big Data, зумовлені зростанням обсягів інформації, з якими не справлялися традиційні реляційні БД. Класичний підхід базувався на розподіленому зберіганні та пакетній обробці (Batch Processing) [2]. Компанія Google запропонувала концепцію MapReduce для паралельних обчислень, а HDFS стала стандартом зберігання. На їх основі виникли інструменти Apache Hive та Pig. Головним недоліком цих систем була висока латентність та відсутність інтерактивності, що унеможливлювало масштабування в реальному часі.

У 2010–2015 роках відбувся перехід до потокової обробки даних. Ключову роль відіграли in-memory платформи, такі як Apache Spark, що працювали значно швидше за MapReduce [13], та системи обробки подій (Apache Kafka, Storm, Flink). Парадигма змінилася з «зберегти і обробити» на «аналізувати під час надходження». Це формалізувалося у концепціях Data Pipeline та ETL/ELT реального часу. CRM-системи отримали можливість миттєво аналізувати поведінку клієнтів [5], створивши базу для впровадження Multithreading та Machine Learning.

Рис. 1 Порівняння архітектур пакетної обробки та потокової обробки даних у реальному часі

Ера машинного навчання та інтеграція XAI (2015–2020)

Цей період ознаменувався інтеграцією машинного навчання (ML) в архітектури Big Data. Завдяки бібліотекам TensorFlow, PyTorch та Scikit-learn акцент змістився на автоматизовану прогнозну аналітику: передбачення попиту, відтоку клієнтів та продажів [3]. Проте складні моделі працювали як «чорні скриньки», забезпечуючи високу точність, але не прозорість рішень [1]. Відсутність пояснень знижувала довіру до систем і вимагала переходу до Explainable AI (XAI).

Рис. 2 Еволюція предиктивної аналітики (2015–2020): Потокові конвеєри даних у реальному часі на основі машинного навчання

Пояснюваний штучний інтелект (Explainable AI) і прозорість моделей

З 2020 року пріоритетом стала здатність пояснити рішення моделі. Інструменти XAI, такі як SHAP, LIME та ELI5, дозволили інтерпретувати результати «чорних скриньок» [1]. Методологія SHAP, що базується на теорії ігор, кількісно оцінює внесок кожного фактора у прогноз [8]. У CRM це критично для обґрунтування бізнес-стратегій та аудиту рішень (наприклад, причин прогнозованого відтоку). Прозорість підвищує довіру користувачів і дозволяє адаптувати бізнес-логіку, перетворюючи XAI на невід'ємну складову аналітичних систем.

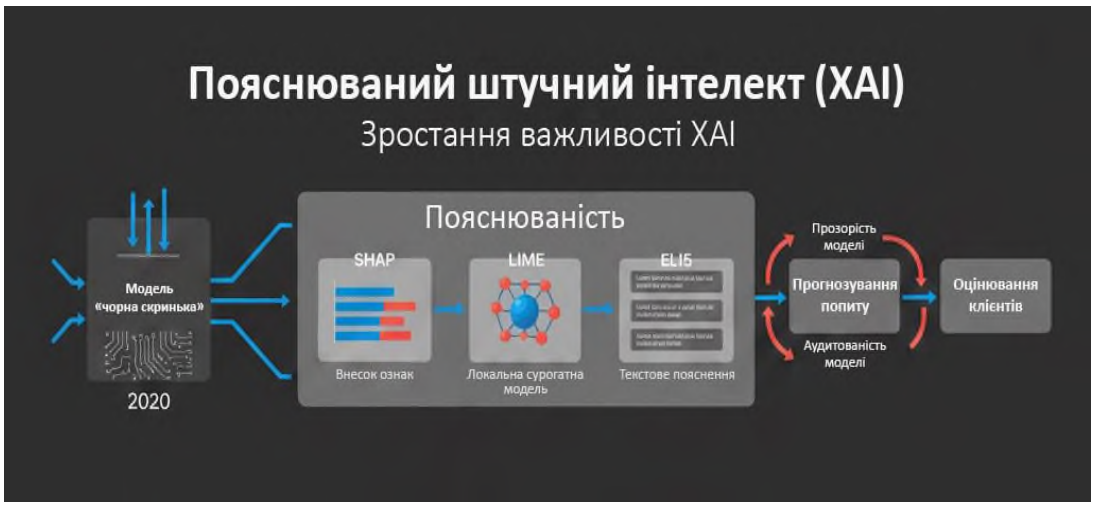


Рис. 3 Концептуальна схема інтеграції пояснюваного штучного інтелекту (XAI) у процеси аналізу даних, починаючи з 2020 року

Інтелектуальні системи нового покоління: багатопоточність + XAI (2022–...)

Сучасний етап характеризується поєднанням високопродуктивного опрацювання даних із повною прозорістю аналітики. Інтелектуальні платформи (Intelligent Data Pipeline) використовують багатопотоковість (Multithreading) для миттєвої реакції на запити та XAI для пояснення прогнозів. Це перетворює CRM з бази даних на SaaS-рішення для «розуміння клієнта», що самостійно генерує обґрунтовані прогнози, забезпечуючи швидкість і довіру до результатів.

Еволюція обробки даних пройшла шлях від пакетних систем (Hadoop) до потокової обробки та ML-аналітики. Сучасна парадигма долає проблему «чорної скриньки», поєднуючи швидкість багатопотокової обробки з прозорістю Explainable AI (SHAP). Це перетворює CRM-платформи на інтелектуальні SaaS-рішення, здатні надавати інтерпретовані висновки в реальному часі, що є стандартом для ефективних управлінських рішень.

1.2. Формати зберігання та обміну великих обсягів даних

Загальні вимоги до форматів Big Data-аналітики

Ефективність інтелектуальних CRM-систем безпосередньо залежить від форматів даних. Основні критерії вибору: швидкість I/O, коефіцієнт стиснення та підтримка схем. Оскільки система використовує багатопотоковість (Multithreading), формат повинен підтримувати паралельний та вибірковий доступ для мінімізації

навантаження на мережу і пам'ять. Наявність метаданих (самодокументованість) є критичною для надійності ETL-процесів та інтеграції нових джерел. Обраний стек технологій має забезпечувати баланс між швидкістю доступу (для ML) та точністю даних (для XAI).

Текстові формати обміну: CSV та JSON

Хоча аналітика вимагає бінарних форматів, текстові формати залишаються важливими для інтеграції [4].

- **CSV:** Стандарт для обміну табличними даними. Простий, але неефективний для Big Data: відсутність схеми ускладнює валідацію, а рядкова структура сповільнює паралельну обробку.
- **JSON:** Універсальний формат для напівструктурованих даних та API. Ідеальний для обміну об'єктами, але має надлишковий обсяг та низьку швидкість читання, що робить його непридатним для основного сховища аналітики. В системі вони виконують роль інтерфейсних форматів для імпорту/експорту та API-взаємодії.

Колонарні формати: Parquet, ORC, Avro

Для аналітичного сховища обрано колонарні формати, що зберігають дані по стовпцях [2, 11]. Це дозволяє зчитувати лише необхідні атрибути, зменшуючи I/O.

- Apache Parquet: Оптимізований для складних вкладених структур та ефективного стиснення. Є стандартом для ML-екосистеми (Spark, Pandas) [11].
- ORC: Забезпечує високу компресію та індексацію, часто використовується в Hadoop-середовищах.
- Apache Avro: Система серіалізації, орієнтована на потокову передачу (Kafka). Підтримує еволюцію схем, що критично для надійності Data Pipeline [2].

Обрано гібридний підхід: Parquet для аналітичного сховища (швидкість ML), Avro для потоків подій, JSON/CSVдля інтеграцій.

Ефективність читання та запису у багатопоточному середовищі

Продуктивність Data Pipeline залежить від паралелізації I/O. Формати на кшталт Parquet містять метадані (статистику блоків), що дозволяє аналітичним

рушіям читати лише релевантні частини файлу, мінімізуючи затримки [11]. На відміну від CSV, який вимагає повного сканування, Parquet дозволяє бібліотекам (наприклад, pyarrow або pandas з chunksize) ефективно розподіляти читання між потоками процесора. Додаткові оптимізації включають I/O-буферизацію та кешування даних у пам'яті для прискорення XAI-розрахунків.

Формати у контексті Explainable AI та ML-аналітики

XAI (SHAP, LIME) висуває суворі вимоги до узгодженості даних. Parquet гарантує типізацію та збереження схеми, що запобігає помилкам при десеріалізації, які можуть викривити пояснення моделі. Avro забезпечує валідацію даних на етапі їх надходження в систему. Колонарні формати також ефективні для зберігання результатів SHAP-аналізу (матриць впливу ознак) завдяки високому ступеню стиснення однотипних числових даних.

Формат	Тип структури	Роль у Data Pipeline (CRM)	Швидкодія аналізу (I/O)	Стиснення та розмір	Підтримка схеми	Оптимальність для XAI/ML
Parquet	Колонарний (двійковий)	Основне зберігання аналітичних даних	**Висока** (вибіркове читання колонок)	Високе	Вбудована, обов'язкова	Ідеальна
ORC	Колонарний (двійковий)	Альтернативне аналітичне зберігання	Висока	Високе	Вбудована, обов'язкова	Висока
Avro	Рядковий (двійковий)	Обмін у потокових системах (Kafka)	Середня	Середнє (блочне)	Вбудована, гнучка (Schema Evolution)	Висока (для потоку)
JSON	Напівструктурований (текстовий)	**API-комунікація**, обмін зі зовнішніми сервісами	Низька (потрібно читати весь рядок)	Низьке	Неявна (опціонально)	Висока (для обміну об'єктами)
CSV	Рядковий (текстовий)	Експорт/імпорт, звітність для користувача	Низька (потрібно читати весь файл)	Дуже низьке	Відсутня	Низька (потребує валідації)

Рис. 4 Оцінка форматів зберігання даних для інтелектуальної CRM

Підсумки підрозділу

Аналіз форматів підтвердив необхідність спеціалізації: CSV/JSON — для інтеграції, Avro — для потоків, Parquet— для аналітики [11]. Вибір Parquet як основного формату сховища зумовлений його підтримкою колонарного доступу, стиснення та схем, що є критичним для швидкодії багатопотокового ML-конвеєра та надійності XAI.

1.3. Методи стиснення та оптимізації обсягу даних

Необхідність та класифікація

Експоненційне зростання обсягів даних у CRM (транзакції, телеметрія) створює навантаження на дискову підсистему та мережу. Оптимізація здійснюється через стиснення, яке поділяється на два типи: lossless (без втрат) та lossy (із втратами) [2]. Для аналітичних систем, ML-моделей та фінансової звітності критично важливим є використання lossless-алгоритмів, оскільки точність даних є передумовою коректності роботи SHAP-аналізу та XAI. Методи із втратами застосовуються лише для мультимедійного контенту.

Сучасні алгоритми у Big Data Хоча класичне кодування Гаффмана заклало теоретичні основи мінімізації ентропії, сучасні екосистеми (Apache Spark, Parquet) використовують більш продуктивні рішення [11, 13]:

- Snappy: Орієнтований на максимальну швидкість кодування/декодування (throughput). Є стандартом для «гарячих» даних у системах реального часу.
- Zstandard (ZSTD): Забезпечує вищий ступінь стиснення порівняно зі Snappy при збереженні прийнятної швидкодії. Є ефективним для архівування.
- Deflate (Gzip): Має високий ступінь стиснення, але повільніший, тому менш придатний для високошвидкісних Data Pipelines.

Алгоритм	Клас	Швидкість стиснення	Коефіцієнт стиснення (у порівнянні з Gzip)	Ідеальне використання у Big Data/CRM
Snappy	Lossless (Словниковий/LZ77)	**Дуже висока** (пріоритет)	Низький - Середній (≈ 20-50% від Gzip)	ETL-процеси, читання/запис гарячих даних, формат Parquet/ORC
Zstandard (ZSTD)	Lossless (Словниковий/LZ77)	Висока - Дуже висока (гнучкість)	**Високий** (може перевершувати Gzip)	Баланс швидкості та ефективності, архівування, зберігання холодних даних
Deflate (Gzip)	Lossless (Гаффман + LZ77)	Низька - Середня	**Дуже високий**	Мережева передача (HTTP), архівування, але повільний для I/O в Big Data
LZ4	Lossless (Словниковий/LZ77)	**Максимальна**	Низький (менший, ніж Snappy)	Надшвидка декомпресія, логи, тимчасові дані

Рис. 5 Порівняння алгоритмів стиснення у Big Data

Оптимізація у Data Pipeline та контекст XAI У багатопотоковому середовищі ефективність досягається поєднанням колонкового зберігання (Parquet) та блокового стиснення. Це дозволяє розпаралелювати операції декомпресії та мінімізувати I/O, що критично для прискорення навчання ML-моделей. Використання lossless-алгоритмів гарантує, що дані, які надходять до модуля

Explainable AI, є ідентичними вихідним, забезпечуючи відтворюваність пояснень [8].

Підсумки підрозділу

Аналіз методів показав, що для інтелектуальної CRM-платформи оптимальним є поєднання формату Parquet та алгоритму Snappy [11]. Це рішення забезпечує необхідну низьку латентність для систем реального часу та ефективне використання ресурсів сховища.

1.4. Інтелектуальний аналіз даних та пояснюваний ШІ (XAI)

Поняття та еволюція Data Mining

Інтелектуальний аналіз даних (Data Mining) є ядром сучасної Big Data-аналітики, що дозволяє виявляти приховані патерни та генерувати нові знання з великих масивів даних. У рамках Data Pipeline він є ключовим процесом, що еволюціонував від статистичних методів до складних алгоритмів машинного навчання (ML), здатних прогнозувати поведінку користувачів [3]. У сфері CRM це дозволяє ідентифікувати цінні сегменти, прогнозувати відтік (Churn Prediction) та персоналізувати пропозиції [5]. Інтеграція Data Mining з принципами XAI завершує архітектуру системи, забезпечуючи не лише точність, але й прозорість прогнозів.

Проблема «чорної скриньки»

Висока точність сучасних ML-моделей (ансамблі, нейромережі) часто досягається ціною їхньої непрозорості («чорна скринька»). Внутрішня логіка прийняття рішень залишається недоступною, що створює ризики для бізнесу та знижує довіру користувачів [1]. У CRM це критично: прогноз відтоку без пояснення причин не дозволяє менеджеру обрати правильну стратегію утримання. Вирішенням є впровадження Пояснюваного ШІ (Explainable AI, XAI), який трансформує математичні прогнози на зрозумілі бізнес-інтерпретації.

Основні підходи до Explainable AI (XAI)

XAI робить моделі інтерпретованими, забезпечуючи аудит та етичну відповідальність [1]. Розрізняють глобальну (поведінка моделі в цілому) та локальну (пояснення конкретного прогнозу) інтерпретацію. В інтелектуальній CRM

глобальні методи виявляють стабільні закономірності, а локальні (SHAP, LIME) пояснюють рішення для кожного клієнта. Паралельне обчислення цих показників у багатопотоковому режимі дозволяє масштабувати прозорість у реальному часі.

- Feature Importance: Базовий метод, що оцінює загальний внесок ознак у модель. Дозволяє виявити ключові фактори впливу (ціна, пробіг, сезонність), але не пояснює індивідуальні рішення [7].
- LIME (Local Interpretable Model-Agnostic Explanations): Будує просту лінійну модель навколо конкретного прогнозу. Перевага — швидкість та наочність; недолік — можлива нестабільність на складних межах рішень [1]. У CRM використовується для швидких локальних інсайтів.
- SHAP (SHapley Additive exPlanations): Базується на теорії ігор і розкладає прогноз на суму внесків кожної ознаки (значення Шеплі). Забезпечує математично обґрунтовані, узгоджені пояснення як на локальному, так і на глобальному рівнях [8]. Є найбільш точним методом для глибокої аналітики.

Критерій	LIME	SHAP
Теоретична основа	Евристичний метод. Базується на локальній апроксимації (тренування простої моделі).	Солідна теоретична основа. Базується на теорії ігор (значення Шеплі).
Швидкість (для загальних моделей)	**Вища.** Швидший для генерації однієї локальної точки, оскільки вимагає менше пертурбацій.	**Нижча.** Обчислювально інтенсивний (особливо KernelSHAP) через необхідність оцінки всіх можливих коаліцій ознак.
Швидкість (для моделей на основі дерев)	Помірна.	**Дуже висока.** Метод TreeSHAP є надзвичайно швидким і точним для XGBoost, LightGBM, CatBoost та ін.
Точність / Надійність	**Локальна.** Висока точність лише в безпосередній близькості до прогнозованої точки. Апроксимація може бути невірною на межі рішень.	**Висока / Послідовна.** Єдиний метод, що гарантує властивості "справедливого" розподілу внеску (consistency, efficiency).
Масштабованість	**Помірна.** Добре масштабується за кількістю ознак, але є часозатратним, якщо потрібно пояснити велику кількість прикладів.	**Різна.** Низька для KernelSHAP. Висока для TreeSHAP (рекомендований вибір для Big Data аналітики з деревами).
Інтерпретаційна глибина	**Добра.** Надає простий, зрозумілий лінійний коефіцієнт (вагу), але важко порівнювати між різними точками.	**Висока.** Надає єдину мету внеску (значення Шеплі), що дозволяє порівнювати вплив ознак як на рівні окремого клієнта (локально), так і на рівні всієї клієнтської бази (глобально).
Висновок для CRM	Підходить для швидкої діагностики окремих, простих випадків.	**Кращий вибір.** Підходить для глобальної аналітики та побудови надійної, комплексної XAI-системи, яка є ключем до "розуміння" даних.

Рис. 6 Порівняльний аналіз методів LIME та SHAP

Інтеграція XAI у Data Pipeline

Модуль XAI інтегрується в архітектуру системи після етапу інференсу ML-моделі. Він трансформує «сирі» прогнози на інтерпретовані висновки перед їх відображенням у UI. Завдяки багатопотоковості, розрахунок SHAP-значень виконується паралельно, що мінімізує затримки. Користувач отримує не лише ймовірність події, а й візуалізацію факторів впливу (наприклад, позитивний вплив історії покупок та негативний вплив скарг) [8].

Підсумки підрозділу

Data Mining та ML є фундаментом сучасної CRM, але потребують прозорості. Інтеграція методів XAI, зокрема SHAP (для точності) та LIME (для швидкості), вирішує проблему «чорної скриньки» [1]. Розміщення модуля пояснень у багатопотоковому конвеєрі забезпечує баланс між швидкістю обробки та глибиною аналітики, що є критичним для ефективних SaaS-рішень.

1.5. Висновки до розділу

Проведений огляд засвідчив, що сучасна аналітика великих даних поєднує оптимізовані формати зберігання (Parquet, JSON), методи стиснення, багатопотокову обробку та ML для виявлення закономірностей у клієнтських масивах. Explainable AI (SHAP, LIME, feature importance) забезпечує прозорість і верифікованість рішень у CRM, підвищуючи довіру користувачів і якість інсайтів. Сукупність Big Data-технологій, паралельних обчислень, моделей ML і XAI формує підґрунтя інтелектуальної CRM-платформи реального часу. Отримані висновки створюють теоретичну основу для подальшого проєктування архітектури, алгоритмів і компонентів системи у наступних розділах.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Архітектура системи та структура потоків даних

Місце інтелектуальної CRM-системи в IT-інфраструктурі

Типова IT-інфраструктура автопідприємства є фрагментованою і охоплює дилерські системи (DMS), ERP-платформи, телематичні сервіси та онлайн-канали. Дані в них розподілені та різнорідні, що ускладнює побудову єдиного профілю клієнта [5]. Інтелектуальна CRM-платформа виступає центральним вузлом консолідації, інтегруючись із цими системами для отримання транзакційної, фінансової та експлуатаційної інформації. Завдяки багатопотоковому конвеєру (Data Pipeline) система паралельно завантажує та нормалізує дані, формуючи єдине сховище Big Data-класу [2]. На цьому фундаменті працюють моделі машинного навчання (ML) та модулі Explainable AI (SHAP), які перетворюють «сирі» дані на прозорі бізнес-рекомендації.

Загальна концептуальна модель

Концептуальна модель системи відображає послідовний рух даних через кілька логічних рівнів. На рівні джерел формуються первинні події (записи з DMS, телематика, веб-аналітика), які надходять у систему в реальному часі або пакетно. Рівень інтеграції реалізує ETL/ELT-процеси: приймання, фільтрацію та валідацію. Використання технологій багатопотокової обробки дозволяє виконувати ці операції паралельно, мінімізуючи затримки [13]. Підготовлені дані зберігаються у Data Warehouse/Data Lake, над яким функціонує аналітичний шар. Тут ML-моделі виконують прогнозування відтоку та сегментацію [3], а компонент XAI обчислює пояснення для кожного прогнозу [1]. Результати передаються на рівень бізнес-візуалізації (UI), де менеджери отримують інтерпретовані інсайти.

Функціональна структура модулів системи

Архітектура платформи складається з набору спеціалізованих модулів, об’єднаних конвеєром даних. Модульність забезпечує масштабованість та спрощує супровід системи [6]:

- Модуль збору та інтеграції: Забезпечує підключення до зовнішніх джерел та паралельну обробку вхідних потоків.
- Модуль попередньої обробки: Виконує очищення, дедуплікацію та нормалізацію даних, приводячи їх до єдиної логічної моделі.
- Модуль аналітики та ML: Є ядром системи, де будуються та виконуються моделі прогнозування LTV, відтоку та крос-продажів [7].
- Модуль Explainable AI: Інтегрує бібліотеки SHAP та LIME для генерації пояснень, перетворюючи моделі з «чорних скриньок» на прозорі інструменти [8].
- Модуль збереження та звітності: Відповідає за зберігання агрегованих метрик та формування дашбордів.
- Модуль керування доступом: Реалізує автентифікацію та рольову модель доступу для захисту даних.

Взаємодія цих модулів у межах Data Pipeline формує повний цикл обробки інформації — від сирих подій до обґрунтованих бізнес-рішень.

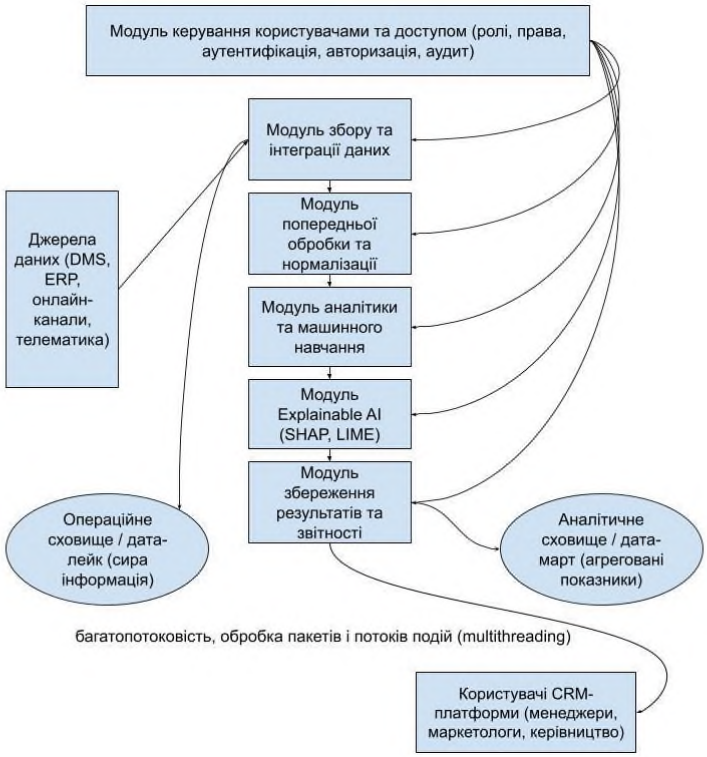


Рис. 7 Концептуальна архітектура інтелектуальної CRM-платформи

Джерела та типи даних

Ефективність платформи залежить від повноти вхідних даних, які можна розділити на чотири ключові категорії. Перша — це транзакційні дані (продажі, сервіс, фінанси), що мають чітку структуру і є базою для оцінки життєвого циклу клієнта [5]. Друга — телематичні дані від підключених авто (IoT), які генеруються у великому обсязі та вимагають потокової агрегації для прогнозування ремонтів. Третя група — це маркетингові дані та веб-аналітика, що описують цифровий шлях клієнта. Четверта — неструктуровані дані (тексти листування, нотатки), які обробляються методами NLP. Поєднання цих різнорідних потоків у єдиній базі є критичним для точності ML-прогнозів.

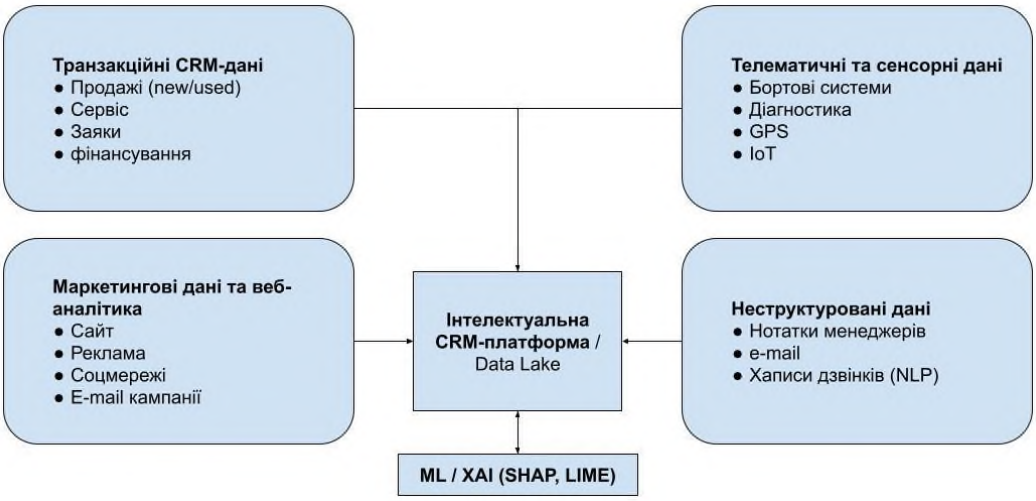


Рис. 8 Основні категорії даних, що надходять до інтелектуальної CRM-платформи та Data Lake

Структура потоків та Data Pipeline

Рух даних у системі організовано як послідовність етапів: отримання -> обробка -> збагачення -> сховище -> аналітика -> XAI. Виділяються два типи потоків: онлайн-потоки (real-time) для миттєвої реакції на події (наприклад, дії на сайті) та пакетні потоки (batch) для глибокої аналітики історичних масивів [2]. Їх інтеграція відбувається через механізми буферизації та черги повідомлень. Data Pipeline забезпечує повний цикл перетворення інформації. Багатопотоковість тут відіграє ключову роль: окремі потоки відповідають за ingestion, очищення, побудову

ознак (feature engineering) та інференс моделей. Це дозволяє уникнути блокувань та забезпечити високу пропускну здатність [13].

Організація багатопотокової обробки

Паралельне виконання операцій є умовою продуктивності системи. Задачі розподіляються між потоками: одні відповідають за I/O операції (читання/запис), інші — за обчислення (нормалізація, ML). Використання асинхронного доступу до БД та буферизації мінімізує час очікування. Ефективність забезпечується пулами потоків та чергами завдань, які рівномірно розподіляють навантаження [6]. Великі масиви обробляються пакетами (batches), а часові ряди — вікнами, що дозволяє системі масштабуватися горизонтально при зростанні обсягів даних.

Інтеграція ML та XAI в архітектуру Модулі машинного навчання вбудовані безпосередньо в Data Pipeline. Моделі працюють у двох режимах: пакетному (для регулярного оновлення скорингу всієї бази) та онлайн (для оперативного оцінювання окремого клієнта). Після отримання прогнозу ініціюється обчислення SHAP-значень. Цей процес виконується в окремих потоках, щоб не затримувати основну відповідь системи. Результати пояснень зберігаються разом із метаданими, що забезпечує аудит та відтворюваність рішень [1].

Вимоги до системи

Архітектура спроєктована з урахуванням трьох головних вимог. Масштабованість досягається поєднанням вертикального та горизонтального розширення компонентів Data Pipeline [2]. Надійність забезпечується реплікацією сховищ та кластеризацією сервісів. Продуктивність гарантується мінімізацією латентності для онлайн-сценаріїв (через кешування) та максимізацією пропускної здатності для пакетної обробки (через паралелізм).

2.2. Технологічне середовище

Технологічне середовище інтелектуальної CRM-платформи спроєктоване для забезпечення повного циклу обробки даних: від завантаження та очищення до машинного навчання та інтерпретації результатів. Система базується на вимогах

високої продуктивності, масштабованості та безпеки, що дозволяє ефективно працювати з великими масивами інформації у багатопотоковому режимі [2].

Серверна частина (backend) виступає ядром системи, реалізуючи бізнес-логіку та координацію Data Pipeline. Для її реалізації обрано фреймворк Laravel [6, 9], який забезпечує ефективну роботу з чергами завдань, підтримку MVC-архітектури та надійну інтеграцію з базами даних. Взаємодія із зовнішніми системами та клієнтськими додатками здійснюється через REST-орієнтований API. Клієнтська частина (frontend) реалізується як адаптивний веб-застосунок з використанням технології PWA [10], що візуалізує аналітичні дашборди та пояснення XAI, забезпечуючи зручну роботу менеджерів на будь-яких пристроях.

Клієнтська частина CRM-платформи є головним каналом взаємодії користувача з системою: через неї результати багатопотокової обробки та ML-моделей подаються в компактній формі. Web-інтерфейс реалізується як адаптивний web-застосунок із модулями роботи з лідами та угодами, сегментації клієнтів, аналітики й звітності. Вимоги включають інтуїтивну навігацію, небагато кроків до ключових даних, стабільний час відгуку та коректне відображення на різних пристроях.

Панелі моніторингу й візуалізації аналітичних результатів подаються у вигляді інтерактивних дашбордів з часовими графіками, тепловими картами та KPI щодо продажів, відтоку й конверсії. Користувач може застосовувати фільтри за періодами, каналами комунікації, моделями автомобілів і виконувати drill-down до рівня окремого клієнта чи угоди, поєднуючи оглядовий і деталізований аналіз.

Інтерфейси для роботи з рекомендаціями та поясненнями моделей XAI відображають прогнозні оцінки ризиків і можливостей, списки рекомендованих дій та пояснення на основі SHAP-значень. Вони подаються у вигляді індикаторів внеску ознак, діаграм важливості й стислих текстових коментарів, зрозумілих без спеціальної підготовки в галузі ML. Таким чином, якісно спроєктований frontend не лише відображає результати ML/XAI-модулів, а й перетворює їх на інструмент прийняття обґрунтованих управлінських рішень на основі прозорої аналітики клієнтських даних.

Системи зберігання даних та формати даних

Системи зберігання даних є ключовим елементом технологічного середовища інтелектуальної CRM-платформи, оскільки забезпечують надійне розміщення великих клієнтських масивів, швидкий доступ до записів і стабільну роботу модулів машинного навчання та Explainable AI. Від архітектури сховищ і вибору форматів файлів залежать продуктивність багатопотокового Data Pipeline, можливості масштабування та якість аналітичних висновків у автомобільному бізнесі.

Операційний контур платформи спирається на OLTP-бази даних, що обробляють CRM-транзакції з підтримкою ACID-властивостей. Аналітичні навантаження виносяться в окремий шар Data Warehouse та/або Data Lake: сховище даних акумулює очищені й узгоджені записи для звітів, тоді як озерце даних зберігає різнотипні потоки в сирому вигляді та формує вибірки для ML- і XAI-модулів. CSV і JSON застосовуються для імпорту й експорту вибірок, тоді як аналітичний шар будується на форматах Parquet і, за потреби потокової серіалізації подій, Avro. Parquet забезпечує стиснення, вибіркове читання колонок і паралельний доступ, що підвищує швидкодію Data Pipeline; Avro завдяки вбудованим схемам підтримує еволюцію структури подій і надійний обмін повідомленнями між сервісами. Комбінація OLTP-БД, Data Warehouse/Data Lake і форматів CSV, JSON, Parquet та Avro є критичним чинником продуктивності й масштабованості Big Data-аналітики та визначає вимоги до подальшого проєктування компонентів системи.

Інструменти для машинного навчання та Explainable AI

У розроблюваній інтелектуальній CRM-платформі інструменти машинного навчання та Explainable AI становлять ядро технологічного середовища, яке перетворює великі клієнтські дані на прогнози й рекомендації для автомобільного бізнесу. ML-компоненти вбудовуються у багатопоточний Data Pipeline, що паралельно обробляє вхідні записи, скорочуючи час від надходження інформації до отримання інтерпретованого результату, придатного для використання менеджерами та аналітиками.

Для задач класифікації, регресії та кластеризації застосовуються бібліотеки Scikit-learn, XGBoost і TensorFlow, які містять оптимізовані реалізації алгоритмів та

підтримують масштабування у багатопотоковому середовищі. Пояснюваний ШІ реалізується насамперед за допомогою бібліотеки SHAP, що обчислює локальні й глобальні внески ознак у прогноз, а також LIME для швидких локальних пояснень окремих рішень; обчислювально затратні XAI-процедури виконуються в окремих потоках із використанням кешування результатів. Повний ML-pipeline включає підготовку даних, навчання й валідацію моделей, серіалізацію найкращих конфігурацій, їх деплой як мікросервісів, доступних через REST-API, а також моніторинг метрик, виявлення дрейфу даних і регулярне перенавчання. Сукупність ML- та XAI-інструментів забезпечує відтворюваність експериментів, прозорість логіки ухвалення рішень і надійність аналітичних компонентів CRM-платформи та слугує основою для подальшої постановки задач і проєктування системи.

Засоби багатопотокової та розподіленої обробки

У контексті опрацювання великих даних засоби багатопотокової та розподіленої обробки є ключовими для продуктивності CRM-платформи. Вони дають змогу масштабувати конвеєр даних без зміни бізнес-логіки та зберігати прийнятний час відповіді навіть за зростання обсягу структурованих і неструктурованих записів.

На рівні застосунку багатопотоковість використовується для поділу задач між робочими потоками: паралельно обробляються партії імпортованих файлів, асинхронно виконуються операції введення-виведення, у фоновому режимі запускаються ML-моделі та модуль Explainable AI. Це скорочує затримки доступу до даних і підвищує пропускну здатність Data Pipeline, не блокуючи основні операції користувача.

Для декуплінгу компонентів застосовуються черги повідомлень і брокери, через які ядро публікує задачі, а сервіси-споживачі підписуються на події. Такий підхід забезпечує гнучкий розподіл навантаження та підвищує відмовостійкість. За потреби архітектура розширюється до кластерних розподілених обчислень, коли етапи Data Pipeline виконуються на множині вузлів і інтегруються з платформами аналітики. Сукупність цих засобів забезпечує виконання вимог до масштабованості й продуктивності CRM-системи Big Data-класу.

Засоби інтеграції із зовнішніми інформаційними системами

Інтелектуальна CRM-платформа в автомобільному бізнесі є центральним вузлом обміну даними, тому інтегрується з дилерськими системами, DMS, ERP та онлайн-каналами. З цих джерел вона отримує відомості про клієнтів, транспортні засоби, угоди й сервісні звернення та формує єдину узгоджену базу, на якій ґрунтуються аналітика й управлінські рішення.

Основним засобом взаємодії є REST/JSON-API, що надає синхронний доступ до актуальних даних DMS, ERP та онлайн-сервісів; за підвищених вимог до швидкодії використовується gRPC. Масовий імпорт і експорт історичних масивів здійснюється через файлові обміни у форматах CSV, JSON чи Parquet. Їх обробляє багатопоточний Data Pipeline, який паралельно виконує екстракцію, очищення та завантаження даних, а ETL-процеси контролюють їхню цілісність.

У платформі поєднуються синхронні й асинхронні моделі обміну. Синхронні запити використовуються лише для операцій, де потрібна негайна узгодженість даних, тоді як асинхронний обмін через черги повідомлень зменшує залежність між системами. Завдяки цьому інтеграційний шар своєчасно поповнює сховище, живить модулі машинного навчання й Explainable AI та підтримує масштабованість і продуктивність платформи.

Середовище розгортання та експлуатації системи

Розгортання системи здійснюється з використанням технологій контейнеризації та оркестрації, що гарантує ізоляцію сервісів та можливість гнучкого масштабування. Надійність експлуатації підтримується комплексними засобами моніторингу та журналювання, а інформаційна безпека забезпечується шифруванням трафіку та розмежуванням доступу на основі ролей.

2.3. Висновки до розділу

Наведений аналіз інформаційного забезпечення показав, що інтелектуальна CRM-платформа виступає ключовим елементом IT-інфраструктури [5], забезпечуючи консолідацію клієнтських даних із різних джерел (DMS, ERP, телематика). Архітектурні рішення підтвердили роль системи як центрального

аналітичного контуру, де багатопотоковий Data Pipeline дозволяє ефективно обробляти великі масиви інформації в реальному часі [2].

Критично важливим елементом стала інтеграція модулів машинного навчання з Explainable AI. Використання методів SHAP [8] та LIME [1] забезпечує прозорість прогнозів, що є необхідною умовою для довіри користувачів до автоматизованої аналітики. Систематизація технологічних вимог визначила вибір інструментів: Laravel [6], Parquet [11] та PWA [10], які спільно формують надійну, масштабовану та безпечну платформу для автомобільного бізнесу.

РОЗДІЛ 3. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Алгоритми обробки даних

Постановка задачі обробки клієнтських даних

У межах інтелектуальної CRM-платформи для автомобільного бізнесу задача обробки клієнтських даних полягає в перетворенні великих масивів подій про взаємодію з клієнтами на формалізовані показники для автоматизованої аналітики та підтримки управлінських рішень [5].

Предметна область включає сутності: клієнт, транспортний засіб, угода, сервісний візит, звернення. Їхні зв'язки «клієнт – угода», «клієнт – сервісний візит», «клієнт – звернення» формально описують життєвий цикл клієнта в дилерській мережі та основні сценарії взаємодії з бізнесом.

Вхідні дані подаються у вигляді таблиць/множин: *Customers(id, segment, age, region, churn_flag)*, *Deals(id, customer_id, car_id, date, amount)*, *ServiceVisits(id, customer_id, date, mileage, cost)*, *Inquiries(id, customer_id, date, topic, result)*. Ці структури обробляються в багатопотоковому конвеєрі, який узгоджує дані та формує ознаки для модулів машинного навчання й Explainable AI.

Ключові вихідні показники включають *LTV (Lifetime Value)*, ймовірність відтоку *P(churn|x)*, скорингові бали та належність до сегментів. У подальшому LTV трактується як дисконтована сума очікуваних платежів, а ймовірність відтоку задається класифікаційною моделлю ML і використовується для скорингу, сегментації та побудови алгоритмів розділу 3 [3].

Математична модель простору даних і ознак

Формалізація простору даних і ознак є необхідною передумовою побудови стійких алгоритмів машинного навчання в інтелектуальній CRM-платформі, оскільки саме структура ознак визначає якість прогнозування, сегментації та пояснюваності моделей. Кожен клієнтський запис подається у вигляді вектора ознак $x \in R^n$, де компоненти відображають транзакційну активність, поведінкові характеристики, часові закономірності та текстові описи взаємодій. Числові змінні охоплюють континуальні та дискретні параметри (пробіг, сума угоди),

категоріальні кодуються через one-hot або ordinal-encoding, а бінарні індикатори фіксують наявність подій. Дані часових рядів (сервісні інтервали, телематичні сигнали) подаються як послідовності з віконними агрегатами, лагами та похідними темпу змін.

Неструктуровані джерела — тексти, примітки менеджерів і лог-файли — проходять окрему стадію формалізації. Для текстів використовуються моделі bag-of-words, TF-IDF або векторні подання, що забезпечують збереження семантики. Логи операцій трансформуються у компактні кодові послідовності подій з можливістю подальшої агрегації за частотою чи часовими інтервалами.

Приклад формування вектору ознак:

```
x = [
    rec.amount,
    one_hot_model[rec.car_type],
    int(rec.repeat_client),
    tfidf.transform([rec.manager_note]).toarray()[0],
    rec.last_visit.diff_days
]
```

Рис. 9 Формування вектора ознак клієнтського запису на основі числових, категоріальних та текстових полів (One-Hot Encoding, TF-IDF)

Коректно визначений простір ознак забезпечує стабільність моделей класифікації, регресії та сегментації, а також формує основу для точного й прозорого пояснення рішень у модулі Explainable AI.

Алгоритми попередньої обробки та очищення даних

Попередня обробка клієнтських даних у CRM-платформі забезпечує придатність масивів до аналізу та побудови моделей машинного навчання [4]. У багатопотоковому Data Pipeline вона є підготовчим етапом.

Спершу обробляються пропуски: для числових ознак використовується імпутація за середнім або медіаною, для категоріальних — за найчастішим значенням, а змістовна відсутність позначається окремою категорією «missing». Записи з надмірною часткою порожніх полів вилучаються. Далі за z-оцінкою та

інтерквартильним розмахом виявляються викиди, які або обрізаються до порогів, або перетворюються для зменшення впливу.

Масштабування й нормалізація ознак реалізуються через стандартизацію та min-max нормалізацію до [0;1], при цьому параметри обчислюються лише на тренувальній вибірці, щоб уникнути витоку інформації.

Категоріальні змінні кодуються залежно від кардинальності: для невеликої кількості рівнів застосовується one-hot encoding, для великої — target (mean) encoding, а порядкові змінні переводяться в ordinal encoding відповідно до природного порядку. Це дозволяє включати їх у простір числових ознак без втрати структури [3].

Псевдокод фрагмента пайплайна попередньої обробки на Python:

```
pipeline = Pipeline([("imputer", SimpleImputer()),
("scaler", StandardScaler())])
```

Рис. 10 Побудова передобробного конвеєра (Pipeline) з імпутацією пропусків та масштабуванням ознак за допомогою StandardScaler

Отже, поєднання імпутації, обробки викидів, масштабування та кодування в конвеєрі очищення формує навчальну вибірку потрібної якості. На її основі в наступних підпунктах розділу будуються моделі прогнозування та інші аналітичні компоненти CRM-платформи.

Алгоритми агрегації та трансформації даних

Алгоритми агрегації та трансформації даних формують простір ознак інтелектуальної CRM-платформи. Сирі подієві логи про угоди, звернення й сервісні візити перетворюються на числові характеристики клієнта, автомобіля та дилерського центру для навчання моделей.

Для кожного клієнта, авто й дилера обчислюються агрегати: сумарний оборот, середній чек, максимальна вартість операції, кількість угод і сервісних візитів, частка сервісних робіт. Багатопотоковий конвеєр дозволяє паралельно виконувати ці розрахунки для різних сегментів бази, прискорюючи обробку великих масивів подій.

Часові інтервали й частотні характеристики описують динаміку взаємодії: проміжки між візитами, час до останнього звернення, число візитів за період. На їхній основі будуються похідні ознаки (feature engineering): середній інтервал, максимальна пауза, середня кількість візитів на місяць, відношення витрат на сервіс до обороту, частка гарантійних робіт, індикатори «високої активності» та «ризику відтоку», індекс лояльності [5]. Такі ознаки пов'язують бізнес-патерни з задачами класифікації, скорингу та рекомендацій.

Короткий приклад агрегації:

```
agg = df.groupby("client_id")["amount"].agg(["sum", "mean"])
```

Рис. 11 Групування даних за клієнтом та агрегування суми й середнього значення витрат методом groupby().agg()

Отримані вектори ознак після агрегації та feature engineering визначають інформативність вибірок. Від їхньої якості залежить ефективність моделей машинного навчання та коректність подальшого пояснювального аналізу за допомогою XAI-методів, зокрема SHAP.

Алгоритми виявлення аномалій та неконсистентних записів

Виявлення аномалій та неконсистентних записів є необхідною умовою роботи інтелектуальної CRM-платформи, оскільки якість даних визначає точність прогнозів і управлінських висновків [2].

Правила цілісності та правил бізнесу контролюють узгодженість дат подій, належність числових полів до допустимих діапазонів, унікальність ідентифікаторів. Їх реалізують тригери бази даних і валідатори, що у кількох потоках обробляють записи.

Евристичні критерії спираються на порогові та доменні правила: фіксація надмірної знижки, від'ємного пробігу, великої кількості угод на один автомобіль. Статистичні методи використовують z-score для виділення викидів; за потреби їх доповнюють моделями виявлення аномалій, які ефективно працюють у багатопоточному Data Pipeline.

Опрацювання відхилень включає автоматичне виправлення за довідниками, маркування записів прапорцями «anomaly», виключення найпроблемніших спостережень з навчальних вибірок та, за потреби, виокремлення окремого класу. Мітки аномалій перетворюються на додаткові ознаки й застосовуються для контролю якості збору даних і коригування бізнес-процесів.

Реалізацію базових перевірок ілюструє фрагмент:

```
df["anomaly"] = (
    (df.mileage < 0) |
    (df.discount > 0.4) |
    (df.service_date < df.sale_date)
)
```

Рис. 12 Правило виявлення аномалій у даних за пороговими умовами

Поєднання логічних, евристичних і статистичних алгоритмів виявлення аномалій забезпечує чисті вибірки та підвищує надійність і пояснюваність моделей машинного навчання в CRM-платформі.

Алгоритми роботи з великими обсягами даних

Великі масиви клієнтських записів не завантажуються цілком у пам’ять, тому застосовуються алгоритми, орієнтовані на потокову та багатопотокову обробку даних.

Основою є пакетна (batch) обробка: записи групуються у партії фіксованого розміру, які послідовно проходять етапи обробки. Для таблиць і файлів, що перевищують доступну пам’ять, використовується поблочне (chunk-based) читання; у Python/pandas це цикл:

```
for chunk in read_csv(path, chunksize=N): process(chunk)
```

Рис. 13 Пакетна обробка великих CSV-файлів за допомогою ітеративного читання даних фрагментами (chunksize)

який дозволяє обробляти дані частинами й передавати блоки між потоками.

Для попереднього аналізу та навчання моделей застосовується семплінг. Випадковий семплінг формує швидку підвибірку, тоді як стратифікований

підтримує пропорції класів і зменшує зміщення оцінок. Розмір вибірки задається як компроміс між точністю показників та витратами часу на обчислення.

Безперервне надходження нових даних вимагає інкрементального оновлення. Нові записи додаються партіями, а алгоритми ML оновлюються через часткове перенавчання або онлайн-методи, наприклад:

```
model.partial_fit(X_batch, y_batch)
```

Рис. 14 Інкрементальне навчання моделі за допомогою методу partial_fit() на потоках (batch) даних

для послідовних порцій; це усуває потребу у повному перенавчанні й підтримує актуальність прогнозів і XAI-пояснень [13].

Отже, поєднання batch-обробки, поблочного читання, семплінгу та інкрементального навчання формує основу алгоритмів роботи з великими обсягами клієнтських даних у CRM-платформі та підтримує масштабованість аналітики в автомобільному бізнесі [2].

Математичні основи багатопотокової обробки

Багатопотокова обробка застосовується для прискорення аналізу великих масивів даних, коли операції над записами виконуються паралельно. Формально D – множина записів, T – множина задач, $P = \{p\_1, \ldots, p\_k\}$ – потоки, а відображення $f: T \to P$ задає розподіл задач. Задача балансування формулюється як мінімізація $max\_j \; \Sigma\_\{t \in T\_j\} \, C(t)$, де $C(t)$ – оцінка трудомісткості, T_j – підмножина задач потоку p_j.

Паралельне агрегування ґрунтується на операції reduce – асоціативній згортці часткових результатів. У map-reduce-парадигмі етап map застосовує функцію до кожного елемента D, утворюючи пари $(key, value)$; далі проміжні результати групуються за ключами, а reduce-етап обчислює агрегати в межах груп [13]. Для послідовного алгоритму з трудомісткістю $T_1 = O(n)$ паралельний варіант на k потоках описується оцінкою $T\_k \approx O(n/k + H)$, де H відображає накладні витрати синхронізації, комунікації. Прискорення $S(k) = T_1/T\_k$ та ефективність $E(k) = S(k)/k$ слугують показниками масштабованості: зі зростанням k H обмежує лінійне

прискорення. Математичний аналіз цих співвідношень дає змогу вибирати кількість потоків і схему розбиття D, що підтримує продуктивність платформи при зростанні обсягу даних.

Метрики якості даних та критерії готовності до моделювання

Перед навчанням моделей у CRM-платформі необхідно оцінити якість клієнтських даних, оскільки метрики ознак визначають придатність вибірки до побудови надійних і пояснюваних моделей.

Щільність заповнення ознаки x_j задається як частка ненульових значень: $\rho_j = \frac{1}{n}\sum_{i=1}^{n} I(x_{ij} \neq NaN)$. У конвеєрі підготовки ρ_j обчислюється для кожного стовпця; ознаки з ρ_j нижче порога (наприклад 0,6) вилучаються або замінюються агрегатами. Це зменшує кількість пропусків і випадкових коливань параметрів.

Інформаційна цінність ознак визначається показниками інформаційної теорії. Для дискретних змінних застосовується інформаційний виграш $IG(Y, X_j) = H(Y) - H(Y \mid X_j)$. Для числових полів використовується взаємна інформація $MI(X_j, Y)$, що зменшує невизначеність результату.

Обчислення метрик реалізується засобами *pandas* і *sklearn*:

```
rho = df.notna().mean()
```

Рис. 15 Обчислення частки заповнених значень у датафреймі

,

```
mi = mutual_info_classif(X, y)
```

Рис. 16 Обчислення взаємної інформації між ознаками та цільовою змінною

Критерії відбору поєднують пороги для ρ_j, IG і MI з ранжуванням ознак та вбудованими методами селекції [7]. У підсумку формується достатньо заповнений простір ознак, що підтверджує готовність вибірки до моделювання в CRM-системі.

Підсумки підрозділу

В даному підрозділі було сформульовано задачу обробки клієнтських даних у межах CRM-платформи автомобільного бізнесу та окреслено її місце в системі математичного забезпечення. Простір даних подано у вигляді векторів ознак, що забезпечує формалізоване представлення клієнтів і операцій для подальшого

моделювання. Узагальнено методи попередньої обробки: виявлення та корекцію пропусків і аномальних значень, нормалізацію й кодування ознак, а також алгоритми агрегації й трансформації, які формують інтегровані показники на різних рівнях аналітики. Окремо розглянуто виявлення аномалій і неконсистентних записів як передумову підвищення достовірності висновків. Стисло підсумовано підходи до роботи з великими масивами, поблочну обробку й багатопотокові конвеєри, що дозволяють ефективно застосовувати Big Data-аналітику. Узагальнено метрики якості даних і критерії готовності вибірки до моделювання, від яких залежать надійність і стабільність моделей машинного навчання. Сформовані підходи створюють підґрунтя для подальшого опису конкретних ML-моделей і засобів Explainable AI, зокрема SHAP і LIME, що забезпечують прогнози поведінки клієнтів.

3.2. Моделі машинного навчання

Постановка задач прогнозної аналітики в інтелектуальній CRM

Прогнозна аналітика в інтелектуальній CRM-платформі автомобільного бізнесу дає змогу на основі даних про клієнтів прогнозувати відтік, продажі та завантаження сервісу, узгоджуючи ці оцінки з цілями дилерського центру [5]. У класифікаційних задачах кожен клієнт описується вектором ознак x , а дискретна змінна y позначає клас: для відтоку $y \in \{0,1\}$, де 1 означає очікуваний відтік; для покупки чи реакції на кампанію – факт цільової дії; будується відображення $f\_clf: X \to Y$. У регресійних задачах цільова змінна безперервна (LTV, очікуваний дохід, прогнозований обсяг сервісних робіт), тому на тому самому просторі ознак X шукається відображення $f\_reg: X \to \mathbb{R}$, яке мінімізує функцію втрат і повертає числові прогнози для планування. У задачах кластеризації попередніх міток немає: потрібно знайти $g: X \to \{1, \ldots, K\}$, що розбиває клієнтів на сегменти, подібні за мірою $d(x_i, x_j)$, а отримані кластери трактуються як поведінкові профілі для таргетування пропозицій [7]. У реалізації платформи це означає, що над спільною матрицею ознак послідовно визначаються цілі y_churn, $y\_purchase$, y_LTV та $y\_cluster$, а така постановка класифікаційних, регресійних і кластеризаційних задач

слугує відправною точкою для вибору моделей машинного навчання та застосування Explainable AI, який робить прогнози прозорими для користувачів [1].

Базові статистичні та лінійні моделі

Базові лінійні моделі є вихідним рівнем прогнозної аналітики, оскільки швидко навчаються на великих вибірках і дають зрозумілі висновки для бізнес-користувача.

Лінійна регресія описує безперервну цільову змінну як $\hat{y} = X\beta$. Коефіцієнти β визначаються мінімізацією функції найменших квадратів $L(\beta) = ||y - X\beta||^2$ і застосовуються для прогнозу виручки чи LTV клієнта.

Логістична регресія моделює ймовірність події $y = 1$ за формулою $P(y = 1 \mid x) = \sigma(w^T x)$, де $\sigma(z) = 1/(1 + e^{-z})$. У CRM її використовують для оцінювання відтоку, ймовірності покупки або відповіді на кампанію при масовій обробці клієнтських баз.

Перенавчання зменшується за допомогою регуляризації. L2-штраф $\lambda||\beta||_2^2$ обмежує величину ваг і підвищує стійкість до шуму, тоді як L1-член $\lambda||\beta||_1$ занульовує частину коефіцієнтів, виконуючи відбір найінформативніших ознак у просторі клієнтських даних.

Інтерпретація знака та величини коефіцієнтів β_j або w_j є базовим XAI-підходом: додатні значення збільшують прогноз чи ймовірність події, від'ємні — зменшують, а більший модуль означає сильніший вплив ознаки. Попри лінійність, такі моделі формують прозору «лінію відліку» для порівняння зі складнішими, зокрема нелінійними, методами прогнозної аналітики в CRM-платформі.

Нелінійні моделі та ансамблеві методи

У прогнозній аналітиці інтелектуальної CRM-платформи зв'язки між ознаками клієнта, автомобіля та історії обслуговування переважно нелінійні, тому лінійні моделі дають обмежену точність і доповнюються деревоподібними та ансамблевими алгоритмами.

Дерево рішень формується рекурсивним поділом простору ознак: у кожному вузлі обирається ознака і поріг, що зменшують імпурітет. Обмеження глибини та мінімального розміру листка стримують перенавчання, а шлях від кореня до листка задає набір якщо-то правил.

Ансамблі Random Forest і Gradient Boosting поєднують багато дерев. Random Forest навчає їх на випадкових підвибірках спостережень і ознак, зменшуючи дисперсію, а Gradient Boosting послідовно додає слабкі дерева, що коригують помилки попередніх та підвищують точність моделей.

Деревоподібні моделі добре сумісні з Explainable AI. Важливість ознак можна оцінювати за зменшенням імпурітету чи пермутаціями, а метод SHAP дає локальні пояснення: SHAP-значення показують, як пробіг авто, вік або активність сервісних візитів змінюють індивідуальний прогноз клієнта [8].

Приклад навчання моделі градієнтного бустингу:

```
model = GradientBoostingClassifier().fit(X_train, y_train)
```

Рис. 17 Навчання моделі GradientBoostingClassifier на тренувальному наборі даних

Отже, деревоподібні та ансамблеві моделі є важливою складовою математичного апарату CRM-платформи, поєднуючи високу якість прогнозів із можливістю отримувати прозорі пояснення на основі SHAP.

Методи побудови ознак та відбору ознак для моделей

Якість моделей машинного навчання значною мірою визначається відбором ознак. Добре спроєктований простір ознак зменшує шум у клієнтських даних, прискорює навчання та підвищує стабільність прогнозів відтоку й повторних покупок.

Статистичний відбір спирається на кореляційний аналіз між ознаками та ціллю: у модель потрапляють змінні з $|r|$ вище порога, а мультикорельовані показники вилучаються. Для класифікації додають інформаційну цінність, що ранжує ознаки за внеском у розділення класів. Такий підхід переважно фіксує прості, майже лінійні залежності.

У деревоподібних і ансамблевих моделях використовують вбудовані методи: важливість ознак оцінюють за зменшенням імпурності вузлів або падінням метрики якості при перестановці значень, що дозволяє виявляти нелінійні ефекти клієнтської поведінки.

Покроковий відбір додає чи вилучає змінні за критеріями якості моделі, але погано масштабується для великої кількості ознак. У такому разі L1-регуляризовані моделі занулюють коефіцієнти несуттєвих змінних; поєднання цих підходів у багатопоточному конвеєрі формує прозорий простір ознак для пояснення прогнозів SHAP/LIME.

Моделі кластеризації та сегментації клієнтів

Сегментація клієнтів дає змогу виокремлювати однорідні групи покупців і налаштовувати для них стратегії цін, сервісу та комунікацій, підвищуючи результативність маркетингу [5]. У межах системи вона використовується як базовий етап аналітичного конвеєра, на якому багатовимірні клієнтські записи перетворюються на осмислені сегменти для подальшого моделювання.

Схожість клієнтів вимірюється метриками відстані. Евклідова підходить для нормалізованих числових показників (суми покупок, частота візитів), мангеттенська стійка до поодиноких викидів у витратах чи пробігах, косинусна корисна для високовимірних поведінкових профілів, де важливий напрямок вектора активності, а не абсолютні значення. Вибір метрики визначає форму й відокремленість кластерів та чутливість результатів до масштабування ознак.

Алгоритм k-means розбиває клієнтів на k груп, мінімізуючи відстань до центрів: центри ініціалізуються, клієнти відносяться до найближчих центрів, потім центри перераховуються як середні значення ознак усередині кластера; кроки повторюються до збіжності. Ієрархічна кластеризація послідовно об’єднує найближчі об’єкти, формує дендрограму вкладених груп, а число сегментів отримують, відсікаючи дерево на вибраній висоті відповідно до бажаного рівня деталізації.

Отримані сегменти інтегруються в модулі CRM-аналітики та XAI. Для кожного кластера окремо навчаються моделі прогнозу відтоку, повторної купівлі й

реакції на маркетингові кампанії, а SHAP і LIME показують, які ознаки (частота сервісу, тип автомобіля, чутливість до знижок тощо) визначають поведінку груп [8]. Таким чином, моделі кластеризації слугують основою пояснюваного персоналізованого таргетингу та побудови сегментованих стратегій взаємодії з клієнтами дилерської мережі.

Оцінювання якості моделей та вибір оптимальної моделі

Кількісне оцінювання якості моделей є необхідною умовою їх використання в інтелектуальній CRM-платформі, оскільки метрики показують, чи можна покладатися на прогнози відтоку, покупки або реакції на кампанію. У задачах класифікації застосовуються $accuracy$, $precision$, $recall$, $F1-score$ та $ROC-AUC$: перша відображає частку правильних рішень, $precision$ і $recall$ характеризують відповідно «чистоту» виявлених ризикових клієнтів та повноту їх охоплення, а $F1-score=2\cdot(P\cdot R)/(P+R)$ узагальнює баланс між цими двома показниками. ROC-крива та площа під нею $ROC-AUC$ описують здатність моделі відокремлювати клієнтів з високою ймовірністю цільової події від інших.

У регресійних задачах прогнозування LTV або доходу використовуються MAE, MSE, $RMSE$ та коефіцієнт детермінації R^2: MAE вимірює середню модульну помилку, MSE й $RMSE$ підсилюють вплив великих відхилень, а R^2 показує частку поясненої варіації. Щоб уникнути переобучення, застосовується крос-валідація: для незалежних спостережень — схема $k-fold$ з усередненням метрик по фолдах, для часових рядів візитів і транзакцій — $time-based\ split$ із суворим дотриманням хронології. Оптимальною для продакшн-використання обирається модель, що демонструє найкраще поєднання рівня метрик і стабільності результатів крос-валідації.

Налаштування гіперпараметрів та стратегії пошуку

У налаштуванні моделей машинного навчання гіперпараметри визначають їхню здатність до узагальнення та стабільності прогнозів, тому їх оптимізація є необхідною складовою роботи аналітичного модуля CRM. Вибір конфігурації

впливає на точність і час навчання, що особливо важливо під час опрацювання великих клієнтських масивів у багатопоточному середовищі.

Метод **grid search** передбачає повний перебір комбінацій гіперпараметрів на дискретній сітці з використанням крос-валідації [3]. Такий підхід гарантує знаходження оптимуму в межах заданої області, але має високу обчислювальну вартість.

```
from sklearn.model_selection import GridSearchCV
gs = GridSearchCV(model, {'max_depth':[3,5,7],
'n_estimators':[100,200]})
gs.fit(X, y)
```

Рис. 18 Налаштування гіперпараметрів моделі за допомогою GridSearchCV

Random search здійснює випадковий вибір комбінацій, що дає змогу ефективніше досліджувати високовимірні простори за фіксованого бюджету ресурсів. Для великих CRM-вибірок цей підхід часто забезпечує кращий баланс між якістю та швидкодією.

```
from sklearn.model_selection import RandomizedSearchCV
rs = RandomizedSearchCV(model, param_distributions,
n_iter=20)
rs.fit(X, y)
```

Рис. 19 Стохастичний пошук гіперпараметрів за допомогою RandomizedSearchCV

Байєсівська оптимізація оцінює гіперпараметри послідовно, використовуючи сурогатну модель цільової функції. Вона спрямовує пошук у перспективні області та зменшує кількість дорогих обчислень, що є важливим за умов великомасштабної CRM-аналітики.

Компроміс між точністю та вартістю обчислень визначається розміром простору гіперпараметрів, часом одного циклу навчання та ресурсами платформи. Оптимізований підбір забезпечує стабільність прогнозів і раціональне використання обчислювальних потужностей інтелектуальної CRM-системи.

Інтеграція моделей у багатопоточний конвеєр обробки

Інтеграція моделей машинного навчання в багатопотоковий конвеєр є ключовою умовою стабільної роботи інтелектуальної CRM-платформи, оскільки саме на цьому рівні забезпечуються паралельна обробка великих клієнтських масивів, оперативний інференс і оновлення прогнозних компонентів. Паралельне навчання та тестування моделей організовується через розподіл задач між потоками й черги завдань: окремі робочі процеси виконують підготовку вибірок, тренування кількох конфігурацій та обчислення метрик, що дозволяє прискорити пошук оптимальних параметрів. Такий підхід також дає змогу оцінювати моделі незалежно одна від одної без блокування конвеєра.

У CRM використовуються два режими скорингу: офлайн-скоринг виконується пакетно для великих вибірок і застосовується для регулярного оновлення ризикових профілів; онлайн-скоринг обробляє запити в реальному часі та забезпечує миттєві рекомендації для менеджерів. Для цього інференс моделі запускається у легких потоках, а підготовлені ознаки кешуються [13].

Оновлення моделей ґрунтується на двох механізмах: періодичному retraining із використанням повних вибірок та incremental learning у разі появи нових подій. Уведення інкрементального навчання допомагає зменшити затримки та враховувати концептуальний дрейф.

```
from concurrent.futures import ThreadPoolExecutor
with ThreadPoolExecutor(max_workers=4) as pool:
    pool.submit(train_model_A)
    pool.submit(train_model_B)
    pool.submit(online_scoring_worker)
```

Рис. 20 Паралельне виконання завдань у багатопотоковому середовищі за допомогою ThreadPoolExecutor

Отже, інтеграція ML-модулів у багатопотоковий Data Pipeline забезпечує швидке навчання, стабільний онлайн-скоринг і своєчасне оновлення моделей, що є критично важливим для продуктивної CRM-аналітики.

Порівняльний аналіз обраних моделей для задач автомобільного бізнесу

Порівняльний аналіз моделей машинного навчання є необхідним етапом перед їх використанням у CRM-процесах автомобільного бізнесу, оскільки дозволяє

оцінити, наскільки алгоритми відповідають вимогам точності, стійкості, інтерпретованості та швидкодії в умовах багатопотокового конвеєра. Критеріями вибору фінальної моделі є якість прогнозу, стабільність щодо зміни вибірки, обчислювальна ефективність, вимоги до ресурсів та здатність до подальшої інтерпретації методами Explainable AI.

Компроміси між точністю, інтерпретованістю та швидкодією є суттєвими: прості моделі на кшталт логістичної регресії забезпечують високу прозорість і швидку інференцію, але поступаються за точністю ансамблевим методам. Дерева рішень та градієнтний бустинг демонструють високу прогностичну якість у задачах відтоку та визначення LTV, проте вимагають потужніших обчислювальних ресурсів. Для онлайн-оцінювання у багатопотоковому середовищі доцільно використовувати моделі з оптимізованою інференцією, тоді як ресурсоємні алгоритми застосовуються в пакетному режимі.

Узагальнюючи результати порівняння, логістична регресія та дерева рішень забезпечують найкращий баланс прозорості та швидкодії, тоді як бустингові ансамблі є оптимальними для високоточної прогнозної аналітики. Саме вони є придатними кандидатами для подальшого пояснення засобами SHAP, які коректно працюють із моделями на деревоподібній основі.

Приклад автоматизованої зведеної таблиці метрик:

```
results = pd.DataFrame(models_scores).sort_values("f1",
ascending=False)
```

Рис. 21 Формування таблиці з метриками моделей та сортування результатів

Чітко визначені критерії та системний порівняльний аналіз формують основу для обґрунтованого впровадження моделей у продакшн.

Підсумки підрозділу

Наведений огляд показав, що у межах CRM-платформи для автомобільного бізнесу задачі класифікації, регресії та кластеризації формують основу прогнозної аналітики, охоплюючи оцінювання ймовірності покупки, ризику відтоку та сегментацію клієнтів. Використані моделі — від лінійних до деревоподібних, ансамблевих і кластеризаційних — забезпечують різні рівні точності та

інтерпретованості, а їх ефективність значною мірою залежить від коректно побудованих і відібраних ознак. Метрики якості, крос-валідація та контроль узгодженості даних гарантують надійність прогнозів. Налаштування гіперпараметрів і багатопотокове виконання моделювання узгоджуються з вимогами високої продуктивності Big Data-середовища. Інтеграція Explainable AI (SHAP, LIME) забезпечує прозорість рішень, роблячи моделі придатними для бізнес-користувачів. Сукупність описаних методів формує концептуальну основу подальшого проєктування інтелектуальної CRM-платформи.

3.3. Методи пояснення рішень (XAI)

Вимоги до пояснюваності в інтелектуальній CRM-системі

Прозорість є ключовою вимогою до інтелектуальної CRM, оскільки моделі ML впливають на рішення щодо взаємодії з клієнтами [1]. Система повинна надавати інтерпретовані пояснення, зрозумілі менеджерам без спеціальної підготовки, показуючи фактори, що сформували оцінку чи прогноз. Пояснюваність підсилює довіру, знижує ризик хибних рішень і підвищує прийняття рекомендацій. Для аудиту та комплаєнсу необхідні відтворюваність логіки, фіксація ознак, відсутність дискримінаційних впливів. Сукупність цих вимог задає рамки для вибору методів XAI і подальшого проєктування інтерфейсів пояснень.

Класифікація підходів XAI

Класифікація XAI у CRM охоплює глобальні й локальні підходи. Глобальні пояснення відображають загальну поведінку моделі та важливості ознак, даючи уявлення про стабільні фактори впливу. Локальні методи деталізують внесок ознак у прогноз для окремого клієнта. Post-hoc підходи інтерпретують рішення вже навченої «чорної скриньки», тоді як внутрішньо інтерпретовані моделі мають прозору структуру з моменту побудови. Така класифікація визначає вибір XAI-рішень у системі [1].

Математичні основи Shapley-значень

Shapley-значення ґрунтуються на кооперативній грі, де множина ознак розглядається як гравці, а характеристична функція визначає внесок кожної коаліції

у прогноз моделі [8]. Маргінальний вклад ознаки обчислюється як різниця між значеннями коаліцій із нею та без неї, а усереднення по всіх перестановках формує справедливий розподіл. Метод задовольняє аксіоми ефективності, симетрії, адитивності та нульового внеску, що гарантує коректність пояснення. Простий псевдокод: перебрати всі підмножини, обчислити маргінальні внески та усереднити їх. Shapley-значення забезпечують інтерпретованість рішень у CRM.

SHAP як універсальний підхід до пояснення моделей

SHAP розглядається як універсальний XAI-підхід у CRM-аналітиці, оскільки ґрунтується на аксіомах Шеплі та забезпечує стабільні локальні й глобальні пояснення. Формально SHAP-значення для ознаки *i* визначаються як середній маргінальний внесок цієї ознаки в прогноз у всіх підмножинах ознак. Для окремого клієнта знак і величина показують, як кожна змінна зміщує результат від базового очікуваного значення. На рівні вибірки SHAP формує узагальнення через summary plot і глобальні важливості. Порівняно з LIME та класичною feature importance, SHAP є точнішим і відтворюванішим, тому обраний базовим методом у CRM-платформі.

Алгоритми обчислення SHAP для різних типів моделей

У XAI CRM-платформи застосовують алгоритми SHAP-значень. Для дерев рішень та ансамблів використовується TreeSHAP, що обходить структуру дерева і швидко дає внески ознак. Для «чорних скриньок» використовується модельно-агностичний KernelSHAP, який наближує значення Шеплі через лінійну регресію (*shap.TreeExplainer(model), shap.KernelExplainer(f, background)*). Для великих вибірок їх доповнюють субсемплінгом клієнтів і обмеженням числа пояснюваних спостережень. У підсумку TreeSHAP доцільний для деревоподібних моделей, а KernelSHAP — для універсального пояснення складніших моделей.

Агрегування та візуалізація пояснень у CRM-аналітиці

Візуалізація пояснень SHAP у CRM-аналітиці дає змогу узагальнювати вплив ознак і пояснювати окремі прогнози. Summary plot показує глобальний розподіл внесків, виявляючи ключові фактори поведінки клієнтів. Dependence plot демонструє зв’язок «значення ознаки – внесок». Для локальних рішень

застосовуються force та waterfall-графіки, що відображають, як фактори зміщують прогноз ризику чи ймовірності покупки [8].

Код прикладу:

```
shap.summary_plot(shap_values, X)
```

Рис. 22 Побудова SHAP Summary Plot для візуалізації глобального впливу ознак на прогноз моделі

Ці візуалізації підсилюють прозорість моделі та підтримують бізнес-рішення.

Використання XAI у бізнес-рішеннях автомобільного сектору

Використання XAI у CRM автомобільного сектору забезпечує трансформацію прогнозів ML-моделей у зрозумілі бізнес-рішення. Агреговані SHAP-профілі дають змогу виявляти ключові драйвери відтоку й лояльності, формуючи основу для пріоритизації клієнтів. Локальні SHAP-значення перетворюються на практичні рекомендації: пропозицію сервісного пакета, повторний контакт чи апсейл. Пояснення також дають змогу виявляти нелогічні рішення моделі й коригувати дані або ознаки, зміцнюючи довіру до аналітики.

Оцінювання якості та стабільності пояснень

Контроль якості пояснень є необхідним у критичних CRM-процесах, оскільки рішення ML впливають на клієнтські дії. Стабільність оцінюють через порівняння SHAP-профілів між версіями моделі, зокрема за дивергенцією Кульбака–Лейблера. Узгодженість із предметною експертизою перевіряють експертними оглядами, опитуваннями та кейс-сесіями. Пояснення прийнятні, якщо вони відтворювані, інтерпретовані й не суперечать бізнес-правилам. Метрики стабільності формують основу моніторингу XAI в життєвому циклі моделей [1].

Підсумки підрозділу

Підрозділ узагальнює роль методів XAI у CRM-платформі: математичні основи Shapley-значень і класифікація підходів забезпечують коректну інтерпретацію, алгоритми SHAP/LIME та їхня візуалізація підвищують прозорість прогнозів, а інтеграція в багатопоточний конвеєр підсилює якість бізнес-рішень.

3.4. Висновки до розділу

Розділ узагальнює математичні засади побудови інтелектуальної CRM-платформи. Систематизовано підходи до попередньої обробки даних: алгоритми очищення, нормалізації та формування ознак, які є фундаментом роботи з Big Data [4]. Ключову роль відведено багатопотоковій організації Data Pipeline, що забезпечує паралельну обробку великих масивів та мінімізує затримки доступу, створюючи високопродуктивне середовище для аналітичних розрахунків [2, 13].

Обґрунтовано вибір моделей машинного навчання (ансамблі, дерева рішень) для задач прогнозування відтоку, LTV та сегментації [3, 7]. Їх застосування дозволяє балансувати між точністю та швидкістю інференсу, що є критичним для систем реального часу. Інтеграція методів Explainable AI (SHAP, LIME) у конвеєр обробки вирішує проблему «чорної скриньки», забезпечуючи інтерпретацію прогнозів та знижуючи ризики прийняття управлінських рішень [1, 8]. Сформований математичний апарат створює основу для проєктування архітектури системи у наступному розділі.

РОЗДІЛ 4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Опис розробки програмного забезпечення

Загальна структура програмного забезпечення

Загальна структура програмного забезпечення інтелектуальної CRM-платформи відтворює концептуальну архітектуру, подану в розділі 2, та реалізує її у вигляді чітко взаємопов'язаних підсистем. Логічна схема охоплює клієнтську частину, серверний модуль, підсистему зберігання даних і ML/XAI-компонент, що взаємодіють через стандартизовані API. Backend забезпечує обробку запитів, виконання бізнес-логіки та керування багатопотоковим конвеєром, який паралелізує операції очищення, агрегації й підготовки вибірок. ML/XAI-модуль відповідає за навчання моделей, скоринг нових записів і формування SHAP-пояснень. Frontend/PWA надає інтерфейс для перегляду прогнозів і ключових факторів рішень. Підсистема зберігання даних включає операційні БД та аналітичне сховище. Сукупність цих компонент забезпечує цілісне функціонування інтелектуальної CRM-платформи.

Вибір стеку технологій та обґрунтування

Метою вибору стеку технологій є формування програмної основи, здатної забезпечити масштабовану обробку великих клієнтських масивів, інтеграцію моделей машинного навчання та підтримку пояснюваності рішень. Серверна частина реалізована мовою Python із застосуванням фреймворку Django Rest Framework, що надає структуровану організацію бізнес-логіки та REST API. Підтримка багатопоточності забезпечується засобами multiprocessing для паралельної обробки потоків даних. Для аналітики використано бібліотеки NumPy, pandas та scikit-learn, які забезпечують класифікацію, регресію й кластеризацію, а також модуль shap для обчислення SHAP-пояснень. Клієнтська частина побудована на React з підтримкою PWA, що забезпечує адаптивний інтерфейс і роботу офлайн. Для зберігання даних застосовано PostgreSQL та ORM-рівень Django, а формати CSV, JSON і Parquet використовуються для ефективного обміну та архівації.

Сукупність обраних технологій узгоджується з вимогами до Big Data-обробки, ML/XAI та концептуальною архітектурою системи.

Структура проєкту та організація вихідного коду

Структура програмного проєкту інтелектуальної CRM-платформи визначає впорядковану організацію компонентів, що забезпечує керованість, масштабованість і відповідність вимогам багатопоточної обробки клієнтських даних. Проєкт побудовано як модульну систему, де кожен пакет виконує окрему функцію та узгоджується з багатошаровою архітектурою. Базові директорії включають **data/** для доступу до сховищ, **domain/** для бізнес-логіки, **api/** для контролерів REST і **ui/** як презентаційний шар PWA. Шар доступу до даних реалізує репозиторії та адаптери, шар бізнес-логіки містить сервіси, моделі й інтеграцію ML/XAI, тоді як API-рівень забезпечує маршрутизацію запитів. Принципи SRP і шаблон Layered Architecture мінімізують зв'язність та спрощують тестування. Узгодженість структури з концептуальною архітектурою дає можливість ефективно розвивати платформу та розширювати її функції.

Реалізація ядра системи та бізнес-логіки CRM

Ядро інтелектуальної CRM-платформи реалізує базову предметну логіку й забезпечує узгоджену роботу модулів обробки даних, аналітики та ML/XAI. Структура ядра охоплює ключові моделі: *клієнт* (ідентифікатори, контактні дані, історія взаємодій), *автомобіль* (VIN, модель, пробіг, сервісні параметри), *угода* (тип, дата, вартість, статус) та *сервісний візит* (перелік робіт, рекомендації, виявлені дефекти). Між моделями встановлено зв'язки «клієнт–авто», «авто–візити», «клієнт–угоди», що формує єдине джерело даних для подальшого аналізу.

Бізнес-операції виконуються сервісним шаром, який охоплює створення та оновлення записів, перерахунок агрегованих показників і запуск аналітичних тригерів. Валідація забезпечує перевірку обов'язкових полів і контроль бізнес-правил.

```
class DealService:
    def update_status(deal, new_status):
        validate(deal)
        deal.status = new_status
```

Рис. 23 Приклад сервісного методу для оновлення статусу угоди з попередньою валідацією в об’єктно-орієнтованій структурі

Реалізоване ядро формує структуровану основу для багатопоточного конвеєра, моделей ML і механізмів XAI.

Реалізація підсистеми обробки та підготовки даних

Підсистема обробки та підготовки даних виконує роль базового шару інтелектуальної CRM-платформи, забезпечуючи формування структурованих потоків клієнтських записів для подальшого навчання, оцінювання та інференсу моделей. Імпорт даних реалізовано через модулі читання CSV/JSON, конектори до API дилерських систем, ERP та DMS, а також черги обміну для потокових подій. Очищення включає видалення пропусків, стандартизацію форматів, нормалізацію та фільтрацію аномалій. Трансформація охоплює кодування категоріальних ознак, агрегування історичних подій та формування векторів ознак відповідно до вимог моделей, описаних у розділі 3.1. Багатопоточний ETL-конвеєр забезпечує паралельне формування train/validation/test-вибірок та окремих інференс-датасетів.

```
for chunk in stream(source):
    clean = sanitize(chunk)
    features = transform(clean)
    save_batch(features)
```

Рис. 24 Стримінгове опрацювання даних у пакетному режимі

Якість підготовки даних безпосередньо впливає на стабільність ML-прогнозів і коректність XAI-пояснень.

Реалізація аналітичних сервісів та сервісів ML

Аналітичні сервіси та сервіси машинного навчання формують окремий рівень інтелектуальної CRM-платформи, забезпечуючи автоматизоване отримання прогнозів, скорингових балів і похідних аналітичних показників. Архітектура

сервісів побудована як набір ізольованих компонентів, що взаємодіють через REST-інтерфейси та черги повідомлень і підтримують багатопотокову обробку даних. Сервіс навчання моделей приймає конфігурації, ініціює тренування, виконує логування та зберігає артефакти у сховищі. Типовий виклик має вигляд:

```
POST /ml/train {model: "churn_xgb", params:{...}}
```

Рис. 25 Приклад REST-запиту для запуску навчання ML-моделі

Сервіс скорингу забезпечує онлайн- і офлайн-обчислення прогнозів, використовуючи кешовані моделі:

```
POST /ml/score {client_id:123}
```

Рис. 26 Виклик REST-endpoint /ml/score для отримання скорингового прогнозу за ідентифікатором клієнта

Результати прогнозування разом із SHAP-показниками зберігаються у базі даних і передаються в CRM-інтерфейс для формування рекомендацій і пояснень. Таке поєднання аналітики й XAI підвищує прозорість та практичну цінність прогнозів.

Реалізація програмного інтерфейсу (API)

Програмний інтерфейс є центральним механізмом взаємодії між компонентами інтелектуальної CRM-платформи, забезпечуючи узгоджений обмін даними між frontend і backend. REST/JSON API спроєктовано за принципами ресурсноорієнтованої архітектури, де кожен ресурс має уніфіковані методи доступу та стандартизований формат відповіді з полями **status**, **data** та **error**. Основні ендпоінти охоплюють роботу з клієнтами, угодами та сервісними візитами, включаючи створення, оновлення, фільтрацію та отримання статистики. Для ML-компонентів передбачено окремі маршрути: запуск скорингових процедур, отримання прогнозів і перегляд агрегованих показників аналітики. Автентифікацію реалізовано на основі JWT-токенів, що містять ролі та дозволи, а авторизаційні фільтри обмежують доступ до окремих операцій. Типовий виклик ендпоінта подано у вигляді:

`POST /api/v1/ml/score { "client_id": 1024 }`, відповідь:

`{ "score": 0.87 }`.

Обробка помилок, журналювання та моніторинг

У системі передбачено комплексні механізми обробки помилок і моніторингу, що гарантують стабільність багатопоточного конвеєра та надійність API. Виняткові ситуації обробляються через ієрархію класів винятків та єдиний перехоплювач, який формує стандартизовану JSON-відповідь із кодом, типом та описом помилки. Журналювання реалізовано у форматі структурованих логів із розмежуванням рівнів (INFO, WARNING, ERROR) та кореляцією запитів через унікальні ідентифікатори. До логів потрапляють ключові події, збої моделей ML та аномалії даних. Моніторинг забезпечується агрегуванням метрик і можливістю інтеграції з зовнішніми сервісами сповіщень.

```
try:
    service.run()
except DomainError as e:
    logger.error({"err": e.code, "msg": e.message})
```

Рис. 27 Перехоплення доменних винятків у сервісному процесі

Підсумки підрозділу

Підрозділ узагальнює реалізацію програмної архітектури CRM-платформи, спрямованої на підтримку аналітики великих клієнтських даних. Сформовано ядро бізнес-логіки, багатопотоковий конвеєр обробки даних, ML/XAI-сервіси, REST API та механізми логування й моніторингу. Рішення узгоджується з концепціями попередніх розділів і готове до масштабування.

4.2. Багатопотокова реалізація

Мотивація використання багатопоточності в системі

Умови обробки великих масивів клієнтських подій, телематики та CRM-транзакцій вимагають високої пропускної здатності та мінімальної затримки. Типові навантаження включають масовий імпорт, нічні batch-процеси та онлайн-скоринг моделей, де послідовні операції спричинюють накопичення черг і значні затримки. Використання багатопоточності дає змогу паралельно виконувати читання, очищення, агрегацію й застосування ML-моделей, забезпечуючи своєчасне

формування аналітики. Паралельний підхід особливо важливий у багатошаровому Data Pipeline. Наприклад:

```
with ThreadPoolExecutor() as pool:
    results = list(pool.map(process_batch, batches))
```

Рис. 28 Паралельна обробка пакетів даних

Обрані підходи та засоби реалізації багатопоточності

Багатопоточність у платформі застосовано для стабільної обробки великих масивів клієнтських даних та паралельного виконання аналітичних і ML-процесів. Використано стандартні механізми мови: системні потоки, пули потоків і асинхронну модель async/await, що забезпечує одночасне виконання запитів до БД, обробку батчів даних і запуск ML-модулів. Організацію паралелізму підтримують пули виконавців та асинхронні черги, інтегровані в Data Pipeline. Для ілюстрації застосовано фрагмент:

```
with ThreadPoolExecutor() as ex:
    results = list(ex.map(process_batch, batches))
```

Рис. 29 Використання ThreadPoolExecutor для паралельної обробки пакетів даних за допомогою методу map() у контекстному менеджері

Таке поєднання механізмів забезпечує масштабовану та стійку до навантажень архітектуру.

Багатопотокова організація Data Pipeline

Багатопотоковий конвеєр даних у платформі CRM сприяє одночасному виконанню операцій над великими масивами клієнтських даних, тим самим зменшуючи затримки під час інтеграції зі сховищами файлів, базами даних та зовнішніми API. Для паралельного читання використовується пул потоків, при цьому кожен виконавець призначається для обробки окремого джерела, а результати згодом ставляться в чергу для наступних етапів. Незалежні завдання виконують очищення, перетворення та розробку функцій для пакетів даних, що дозволяє рівномірно розподілити навантаження на процесор та пришвидшує підготовку зразків для моделей машинного навчання. Розмір пакета визначається експериментальним шляхом, враховуючи доступну оперативну пам'ять та

пропускну здатність диска. Менеджер черги контролює виконання виконавців завдань, гарантуючи, що вузли не перевантажуються. Узагальнений приклад наведено нижче:

```
for batch in parallel_load(source_list):
    submit_to_queue(clean_transform(batch))
```

Рис. 30 Паралельне завантаження даних та асинхронна передача очищених пакетів у чергу для подальшої обробки

Такий тип організації підвищує як масштабованість, так і стабільність конвеєра.

Паралельне навчання та застосування моделей машинного навчання

Масштабованість досягається шляхом паралельного виконання навчання та оцінювання всередині багатопотокового конвеєра даних платформи CRM. Пули процесів реалізують багатоконфігураційне навчання, так що кожне завдання відповідає або окремій моделі, або окремому набору гіперпараметрів. Під час оцінювання клієнтський масив поділяється на кілька підмножин, що дозволяє робочим процесам, що використовують черги завдань, обробляти клієнтські дані. Максимальна кількість завдань виконується паралельно для балансування навантаження з динамічним розподілом навантаження.

```
from concurrent.futures import ProcessPoolExecutor

def train(model):
    return model.fit(X, y)

def score(chunk):
    return model.predict(chunk)

with ProcessPoolExecutor() as ex:
    models = ex.map(train, model_list)
    scores = ex.map(score, client_chunks)
```

Рис. 31 Паралельне навчання моделей та обробка клієнтських батчів у окремих процесах

Робота зі спільними ресурсами та синхронізація

У багатопотоковому конвеєрі даних основною проблемою є безпечний доступ до спільних ресурсів — черг завдань, кешів та статистичних структур, що формують критичні секції. Для захисту колекцій від перегонів даних застосовується блокування м'ютексами/блокуванням. Час блокування мінімізується шляхом копіювання даних перед обробкою та підходу копіювання під час запису. Наприклад, захист лічильника реалізується наступним чином:

```
with lock:
    counter += 1
```

Рис. 32 Використання механізму блокування (lock) для забезпечення потокобезпечного оновлення спільної змінної

Альтернативою є робота з локальною копією, а потім її атомарний запис, що знижує затримку, а також робить її більш надійною.

Аналіз ефекту багатопотокової реалізації

Наведений аналіз демонструє практичний ефект багатопотокового опрацювання Data Pipeline. Час виконання оцінювався шляхом порівняння послідовного та паралельного варіантів для читання, очищення та скорингу клієнтів. Паралельна обробка забезпечила скорочення часу в середньому на 35–55%. За зростання обсягу даних масштабованість поліпшується до моменту, коли накладні витрати синхронізації та I/O обмежують приріст. Нижче наведено фрагмент коду вимірювання часу:

```
from time import perf_counter
def run_seq(tasks): return [t() for t in tasks]
def run_parallel(tasks):
    from concurrent.futures import ThreadPoolExecutor
    with ThreadPoolExecutor() as ex: return list(ex.map(lambda f: f(),
tasks))

t0 = perf_counter(); run_seq(funcs); seq = perf_counter() - t0
t0 = perf_counter(); run_parallel(funcs); par = perf_counter() - t0
print(seq, par)
```

Рис. 33 Порівняння послідовного та паралельного виконання функцій: вимірювання часу виконання через perf_counter() та запуск задач у ThreadPoolExecutor

Обмеження та потенційні вузькі місця багатопотокової архітектури

У багатопотоковій архітектурі критично враховувати вузькі місця. Основні I/O-обмеження пов’язані з повільним диском, мережевими затримками та блокуваннями СУБД, що не дає лінійного масштабування потоків. Додаткові затримки створюють lock-конфлікти й часті context switch. Нижче подано приклад блокування спільного лічильника:

```
lock = threading.Lock()
with lock:
    counter += 1
```

Рис. 34 Потокобезпечне оновлення спільної змінної

Підсумки підрозділу

У підрозділі узагальнено роль багатопоточності в архітектурі CRM. Показано, що паралельний Data Pipeline і одночасне навчання моделей прискорюють обробку та підсилюють масштабованість. Синхронізація зменшує конфлікти, а окреслені обмеження формують напрями оптимізації.

4.3. Інтеграція модуля XAI (SHAP)

Архітектура інтеграції SHAP-модуля в систему

Модуль SHAP архітектурно реалізовано як окремий сервіс у CRM, що взаємодіє з моделями машинного навчання та багатопотоковим конвеєром даних. Після завершення прогнозування надсилається запит до сервісу SHAP, який приймає нормалізовані вхідні дані у форматі JSON та повертає локальні або глобальні пояснення. Результати потім кешуються в базі даних для швидкого отримання відображенням фронтенду/PWA. Скорочення часу обчислення завдяки підтримці паралельних виконавців: багатопотоковий режим масштабується зі збільшенням обсягу записів. Узагальнений виклик:

```
shap_values = shap_service.compute(model_id, batch, workers=4)
```

Рис. 35 Паралельне обчислення SHAP-значень через сервіс пояснюваності

Реалізація сервісів для обчислення SHAP-значень

Сервісний шар для обчислення SHAP-значень реалізує уніфіковані інтерфейси отримання локальних і глобальних пояснень для моделей, інтегрованих у CRM-платформу. Запити містять тип пояснення, параметри семплінгу та ідентифікатор моделі, а відповіді повертають масиви впливів ознак. Для деревоподібних моделей застосовується TreeSHAP, для «чорних скриньок» — KernelSHAP із можливістю конфігурації ядра. Виклик сервісу інкапсулює підготовку вибірки та формування пояснень, напр.:

```
shap_service.compute(model, data, mode)
```

Рис. 36 Виклик сервісу пояснюваності для обчислення SHAP-значень

Оптимізація обчислень SHAP для великих наборів даних

Оптимізація обчислень SHAP у середовищі великих клієнтських даних базується на вибірковій вибірці, кешуванні та багатопотоковості. Використовується підвибірка пояснення, яка є достатньо репрезентативною для записів, що значно скорочує обчислення, зберігаючи при цьому здатність до узагальнення. Для пришвидшення повторюваних запитів застосовується кеш для значень SHAP, організований за політикою оновлення «найменше використані», що надає перевагу найчастіше запитуваним клієнтам та сегментам. Обчислення SHAP запускаються як багатопотоковий пакет, де кожен потік обробляє свою групу клієнтів, а результати остаточно синхронізуються модулем агрегації конвеєра даних. Узагальнений псевдокод оптимізованого процесу можна представити наступним чином:

```
for batch in parallel_batches(data):
    if cache.exists(batch):
        result ← cache.get(batch)
    else:
        result ← shap.compute(batch)
        cache.store(batch, result)
merge(result)
```

Рис. 37 Паралельне обчислення SHAP-значень із використанням кешу

Зберігання та структура даних пояснень

Таблиці пояснень SHAP використовують дизайн JSON-in-PostgreSQL для гнучкого зберігання масивів значень. Локальні пояснення пов'язані з певними прогнозами через ідентифікатори клієнта, версії моделі та позначки часу оцінювання. Глобальні агрегати зберігаються в окремій таблиці, де записуються внески середніх, медіанних та процентильних ознак – це робить реалізацію з багатопотоковим конвеєром даних, який дуже швидко (<30 мс) створює аналітичні панелі. Для інтеграції використовуються такі моделі ORM:

```
class ShapLocal(Base):
    client_id = Column(Integer)
    model_version = Column(String)
    shap_values = Column(JSON)
```

Рис. 38 Модель таблиці ShapLocal: структура зберігання локальних SHAP-значень у базі даних (ідентифікатор клієнта, версія моделі, JSON-представлення значень)

Вибірка здійснюється за ознакою або сегментом, що дає змогу будувати візуалізації впливу факторів.

API для доступу до XAI-функціональності

XAI було реалізовано як API типу REST/JSON, що надає як локальні, так і глобальні пояснення SHAP. Локальне тут означає `'/xai/local/{entity_id}'`, що повертає пояснення для клієнта або угоди, тоді як глобальне – це агреговані метрики за вибіркою `'/xai/global'`. Параметри запиту підтримують `model`, `segment`, `date_from/date_to`, `limit/offset`. Відповіддю є список функцій та значень SHAP з метаданими пагінації. По суті, вибірка з бази даних застосовує фільтри, отримуючи структуровані результати для передачі фронтенду, що створює аналітичні панелі.

Використання SHAP у бізнес-логіці та інтерфейсі

Результати SHAP інтегруються в бізнес-логіку платформи як основа для автоматично згенерованих рекомендацій керівництву. Ранжування характеристик забезпечує розуміння як критичних факторів, так і основних рушійних сил у прогнозі, будь то фактори ризику відтоку чи фактори підвищення лояльності. Механізм перетворює профіль SHAP на короткі бізнес-формулювання: «низька

частота відвідувань – варто ініціювати контакт», «зростання вартості послуг – доречна персональна знижка». У веб-інтерфейсі за допомогою значків та індикаторів представлені лише ключові впливи. Нижче наведено спрощений приклад:

```
drivers = shap_profile.top(3)
recs = rules.apply(drivers)
```

Рис. 39 Формування персоналізованих рекомендацій

Таке представлення робить модель машинного навчання більш прозорою та практичною для пояснення корисності на рівні бізнес-користувача.

Перевірка коректності інтеграції XAI

Для перевірки коректності інтеграції XAI здійснюється тестування узгодженості SHAP-пояснень із прогнозами моделей: відповідність знаків впливу, величин внеску та зміни виходу. Модульні тести перевіряють стабільність сервісу, а навантажувальні вимірюють час формування пояснень та масштабованість.

Підсумки підрозділу

Модуль SHAP інтегровано в архітектуру CRM як XAI-компонент, що працює разом із ML-моделями та багатопотоковим Data Pipeline. Реалізовано сервіси й API для обчислення та зберігання пояснень, застосовано оптимізацію обчислень. SHAP-пояснення підвищують прозорість рішень і готують основу для подальшої оцінки ефективності системи.

4.4. Інтерфейс користувача / PWA

Концепція користувацького інтерфейсу інтелектуальної CRM

Інтерфейс користувача інтелектуальної CRM зосереджений на трьох основних ролях: менеджер, аналітик та адміністратор [5]. Менеджер отримує доступ до сценаріїв роботи з клієнтами, угод та рекомендацій, сформованих модулями ML/XAI, представлених у вигляді панелей, віджетів та карток клієнтів. Аналітик отримує розширені інструменти пошуку даних, включаючи інтерактивні панелі моніторингу та пояснення моделей у форматі SHAP-графіків. Адміністратор

має доступ до інструментів налаштування для каталогів, ролей та системних параметрів.

Архітектура PWA забезпечує модульну структуру екранів, глобальне меню навігації та маршрутизацію, що забезпечує швидкий перехід між робочими областями. Приклад маршрутизації у фреймворку:

```
const routes = [
  { path: '/dashboard', component: Dashboard },
  { path: '/clients/:id', component: ClientCard },
  { path: '/analytics', component: AnalyticsPanel }
];
```

Рис. 40 Оголошення маршрутів фронтенд-додатку

Таке розташування підтримує прозорість прийняття рішень та підвищує ефективність кожної ролі.

Технологічна основа frontend та PWA

Технологічні основи на стороні клієнта базуються на React для забезпечення високої продуктивності, модульності компонентів та ефективної інтеграції з Backend API та сервісами ML/XAI. Віртуальна DOM динамічно оптимізує візуалізацію для аналітичних панелей та результатів моделювання, тоді як багата екосистема скорочує час розробки PWA-реалізацій. Service Worker реалізує запити на виконання перехоплення сервісів, кешуючи статичні ресурси, застосовуючи стратегію кешування (компоненти інтерфейсу) та мережевого відображення (дані CRM). Manifest.json визначає назву програми, набір піктограм, режим відображення, початкова сторінка, робота в автономному режимі, поєднує локальний кеш та чергу, синхронізуючи зміни, надіслані після відновлення мережі. Приклад Service Worker.

```
self.addEventListener('fetch', e => {
  e.respondWith(caches.match(e.request).then(r => r ||
fetch(e.request)));
});
```

Рис. 41 Перехоплення мережевих запитів у Service Worker

Основні екрани та їх функціональність

Всередині інтелектуальної CRM-платформи панель інструментів надає узагальнену інформацію про загальний стан автомобільного бізнесу. Панель відображає показники активної клієнтської бази разом з обсягами продажів та кількістю звернень до сервісу, а також показники відтоку клієнтів, результати маркетингових кампаній та агреговані показники машинного навчання. Дані надходять з багатопотокового конвеєра даних, який оновлюється в режимі оптимізованому для рекордних обсягів.

Екран зі списком клієнтів пропонує фільтрацію за сегментами, пошук за атрибутами та впорядкування за активністю або датою останнього відвідування. Картка клієнта відображає зведену інформацію профілю, історію транзакцій та звернень до сервісу, а також інтегровані показники машинного навчання: LTV (загальна вартість життя), ймовірність відтоку, членство в сегментах, прогнозовані реакції на кампанії.

Інший блок містить екрани прогнозів з балами оцінювання, прогнозованим LTV та ризиком відтоку, а також іншими результатами моделі. Для прозорості рішень надаються панелі XAI/SHAP, що містять важливість ознак як глобальну, так і локальну, графічне представлення внеску факторів та пояснення моделі з точки зору бізнесу [1, 8]. Маршрутизація між екранами може бути реалізована як:

```
{ path: '/clients/:id', component: ClientCard }
```

Рис. 42 Маршрут із динамічним параметром у фронтенд-додатку

Візуалізація аналітики та пояснень моделей

Інтелектуальний інтерфейс платформи CRM реалізує повну концепцію візуалізації прогнозів, візуалізації трендів та візуалізації моделей пояснення, щоб чітко представити як результати машинного навчання, так і показники XAI. Інформаційна панель використовує лінійні діаграми для перегляду тенденцій ключових показників з плином часу, стовпчасті діаграми для порівняння різних сегментів клієнтів з ризиком відтоку на рівні сегмента та очікуваним доходом від багатопотокового конвеєра машинного навчання, проілюстрованого за допомогою іншого набору стовпчастих діаграм.

Для представлення значень SHAP використовуються два типи діаграм. Глобальні зведені графіки або стовпчасті діаграми показують загальний внесок ознак у прогнози моделі на рівні загальної вибірки, виділяючи ключові чинники поведінки клієнтів. Локальні пояснення надаються на каскадних або силових діаграмах, які відображають для одного клієнта позитивний та негативний внесок кожної ознаки в остаточне рішення [8].

Підхід UX передбачає групування функцій у семантичні блоки (демографічні дані, активність, історія покупок), використання кольорових акцентів для контрастного впливу та мінімізацію технічної термінології у підказках. Приклад конфігурації компонента:

```
const chart = new ShapBarChart({ topFeatures: 10, colorScheme: 'impact' });
chart.render('#global-importance');
```

Рис. 43 Побудова бар-чарту глобальної важливості ознак

Адаптивність та юзабіліті інтерфейсу

Для адаптивності інтерфейсу інтелектуальної CRM-платформи застосовано принципи адаптивного макета для десктопів, планшетів та мобільних пристроїв. Вона містить гнучкі CSS-сітки та точки зупинки, за допомогою яких її компоненти (аналітичні панелі, таблиці клієнтів, блоки пояснень SHAP) переставляються відповідно до ширини екрана; на маленьких екранах довгі таблиці перетворюються на «картки» з ключовими полями [10]. Наприклад:

```
@media (max-width: 768px) {
  .layout-grid {
    grid-template-columns: 1fr;
  }
}
```

Рис. 44 Адаптивне оформлення інтерфейсу

Зручність використання та UX зосереджені на мінімізації кроків для ключових дій: відкриття картки клієнта, початок підрахунку очок, перегляд пояснень XAI. Забезпечено послідовну навігацію, помітні кнопки заклику до дії та чіткий порядок дій. Форми підтримують валідацію, чіткі повідомлення про помилки, автоматичне заповнення полів. Для доступності використовуються читабельні шрифти, достатня

контрастність, логічний порядок фокусування та зручні сенсорні зони для сенсорного введення.

Безпека та управління доступом у UI

У розробленій CRM-платформі безпека інтерфейсу користувача реалізована як додатковий рівень до авторизації на стороні сервера: бекенд приймає рішення щодо доступу, а клієнтська сторона керує сеансом [6]. Після входу в систему токен доступу зберігається в пам'яті програми, а токен оновлення поміщається в захищений файл cookie; перехоплювач HTTP-запитів автоматично додає заголовок авторизації та оновлює обидва токени. Захищені маршрути реалізовані за допомогою компонента-обгортки, який перевіряє роль користувача перед відображенням будь-якого контенту.

Модель доступу на основі ролей (менеджер, аналітик, адміністратор) поширюється на маршрути та окремі елементи інтерфейсу. Елементи меню та дії фільтруються виразом `items.filter(i => canView(i.permission))`, який приховує функції, недоступні для поточної ролі. Екрани з аналітичними панелями, поясненнями XAI/SHAP та конфіденційними даними клієнтів позначені як доступні лише аналітикам та адміністраторам. Авторизація на рівні інтерфейсу користувача відповідає моделі доступу на основі ролей та посилює архітектуру безпеки платформи.

Локалізація та налаштування інтерфейсу

Локалізація інтерфейсу в інтелектуальній CRM-платформі забезпечує коректне відображення елементів інтерфейсу відповідно до мови та регіональних налаштувань користувача. Підтримка кількох мов базується на використанні словників перекладів, де кожен текстовий елемент визначається ключем ресурсу. Механізм перемикання мов реалізовано на стороні клієнта через бібліотеку інтернаціоналізації (i18n), яка також відповідає за збереження мовних налаштувань у локальному сховищі. Локалізація впливає на форматування дат, чисел, валют, а також на відображення аналітичних показників та фільтрів. Наприклад, у компоненті вибору періоду React для текстових міток використовується конструкція типу `t('filters.date_range')` та `i18n.formatDate(value)` для дат.

Такий підхід забезпечує узгодженість інтерфейсу користувача в різних локалях та підвищує зручність менеджерів та аналітиків.

Підсумки підрозділу

Описані у підпунктах 4.4.1–4.4.7 рішення щодо PWA-інтерфейсу з ролеорієнтованою навігацією, адаптивним дизайном, локалізацією та вбудованою безпекою забезпечують зручний доступ до Big Data-аналітики, ML/XAI-модулів і пояснень моделей, підтримуючи прозоре прийняття рішень у CRM.

4.5. Тестування системи та оцінка продуктивності

Стратегія тестування програмного забезпечення

Стратегія тестування інтелектуальної CRM-платформи зосереджена на валідації бізнес-логіки, доведенні надійності багатопотокової обробки для великих масивів клієнтів та тестах стабільності сервісів ML/XAI [6]. Сфера тестування включає сервіси імпорту та очищення даних, модулі оцінювання, підсистему пояснювальної аналітики SHAP/LIME REST-API та базові сценарії користувача в інтерфейсі користувача. Модульні тести написані для тестування окремих функцій та класів сервісів. Інтеграційні тести перевіряють взаємодію між серверною базою даних, модулями ML та кінцевими точками XAI. Сценарії аналітиків та менеджерів охоплюються за допомогою наскрізних тестів: завантаження набору даних, запуск моделі, отримання пояснень, візуалізація результатів у веб-інтерфейсі [3].

Модульне (unit) тестування

Модульне тестування є основним інструментом, який використовується для забезпечення якості на цьому рівні та підтримує загальний підхід до тестування системи в цілому. На рівні бізнес-логіки модулі тестують транзакції клієнтів, запис відвідувань служби підтримки та розрахунок протестованих метрик, включаючи реакцію на порогові значення. Для модулів, які виконують тести очищення, трансформації або агрегації, перевіряється цілісність записів та агрегована узгодженість. Типові сценарії включають порівняння фактичних та очікуваних результатів для нормальних даних (assert result == expected), а також подання неправильних або неповних даних з перевірками на наявність помилок.

Інтеграційне та системне тестування

Інтеграційне та системне тестування має підтвердити інтегроване функціонування інтелектуальної CRM-платформи у розробці 4.5.1-4.5.2. Під час інтеграційного тестування перевіряється взаємодія серверних сервісів, модулі ML/XAI з транзакціями бази даних коректно записують результати оцінювання та показники SHAP. Тести REST/JSON API симулюють створення/оновлення клієнтів, запуск оцінювання, отримання пояснень та контроль відповідей на запити структури. Системне тестування виконує повний імпорт даних конвеєра, очищення, застосування моделі ML, розрахунок SHAP, відображення результатів, імпорт сценаріїв інтерфейсу, тестування записів, запусків, оцінювання, зчитування прогнозів, пояснення, перевірки бази даних інтерфейсу [9].

Тестування інтерфейсу користувача

Функціональні тести виконуються для основних екранів (панель інструментів, список та картка клієнтів, панелі XAI/SHAP), перевіряючи правильність індикаторів, фільтрів або шляхів навігації разом з операціями CRUD. Контроль узгодженості керує поясненнями прогнозованих чисел, відображенням легенд на графіках бекенду окремо від них. PWA-тестування охоплює кешування ресурсів роботи в автономному режимі, реакцію інтерфейсу користувача на зміни стану мережі, наприклад, `if (!online) showCache(); else fetchData()`.

Навантажувальне тестування та оцінка продуктивності

Навантажувальне тестування інтелектуальної CRM-платформи використовується для перевірки масштабованості та надійності під час обробки клієнтських масивів і звернень до ML/XAI-сервісів. Експерименти задаються як сценарії: перегляд dashboard, запуск прогнозів із SHAP-поясненнями; навантаження змінюється варіюванням обсягів даних і кількості одночасних запитів до backend. Для сценаріїв вимірюються час відповіді, пропускна здатність, використання CPU та пам’яті. Ефективність багатопотокового Data Pipeline оцінюється порівнянням з однопотоковим варіантом за зміною цих метрик при зростанні навантаження. Узагальнений приклад конфігурації: `scenario="clients_load", users=120, duration=300s`.

Оцінка масштабованості та стійкості системи

Для CRM-платформи з багатопотоковим конвеєром даних масштабованість та стійкість визначають здатність обробляти зростаючі дані без погіршення якості обслуговування. Оцінка виконується в експериментах з навантаженням із поступовим збільшенням кількості записів бази даних та кількості одночасних запитів, вимірюючи час відгуку та коефіцієнт помилок. Метрики навантаження на процесор, пам'ять, диск, мережу та базу даних дозволяють виявляти вузькі місця та планувати вертикальне та горизонтальне масштабування, кешування, балансування трафіку та черги. Навантаження емулюється сценарієм типу: `scenario="clients_load", users=200,duration=300s`.

Аналіз результатів тестування та корекція реалізації

Модульні, інтеграційні, навантажувальні та PWA-тести довели роль перевірки CRM-платформи. Вони виявили збої в бізнес-логіці, конвеєрі даних, SQL-запитах та поясненнях SHAP, які були усунені шляхом рефакторингу сервісів та налаштування конфігурації бази даних і пулу потоків. При повторних запусках система стабільно працює під навантаженням зі швидкою реакцією та пропускною здатністю API, а також коректними результатами ML/XAI; типовий журнал: `[ERR] shap=NaN → fix → [OK]`.

Підсумки підрозділу

Цей розділ містить короткий виклад результатів тестування, що підтверджують як працездатність, так і надійність CRM-платформи на основі великих даних. Тести на різних рівнях – блок, інтеграція, система/інтерфейс користувача/навантаження – підтвердили бізнес-логіку, а також взаємодію компонентів та стабільність у всій системі. Показники продуктивності, досягнуті під час тестування, чітко продемонстрували ефективність у багатопотокових конвеєрах даних та сервісах ML/XAI під час обробки великих вибірок (записів). Результати тестування стали основою для корекції впровадження та подальших рекомендацій щодо масштабування, що пов'язують цей розділ з наступними розділами роботи.

4.6. Висновки до розділу

У розділі представлено програмну реалізацію інтелектуальної CRM-платформи, що трансформує теоретичні моделі у практичний інструмент для автомобільного бізнесу. Розроблена архітектура базується на багаторівневій структурі, яка чітко розмежовує сервіси бекенду, підсистему зберігання, модулі ML/XAI та клієнтський інтерфейс [6]. Такий підхід, реалізований на сучасному технологічному стеку, гарантує масштабованість та готовність до інтеграції в інфраструктуру дилерів.

Ключовим елементом стала реалізація багатопотокового Data Pipeline, який паралельно виконує очищення та агрегацію даних. Це дозволило суттєво скоротити час підготовки вибірок та забезпечити регулярне оновлення прогнозів відтоку та LTV. На цій основі функціонує модуль Explainable AI, що генерує прозорі пояснення (SHAP) для кожного рішення моделі, а за необхідності залучає додаткові методи (LIME) [8]. Взаємодія компонентів забезпечується через уніфікований API, а доступ користувачів — через захищений PWA-інтерфейс з підтримкою офлайн-режиму та рольовим розмежуванням доступу [10].

Комплексне тестування підтвердило коректність наскрізного потоку даних «дані → ML → SHAP → UI» та досягнення цільових показників продуктивності під навантаженням. Оптимізація параметрів кешування та багатопотоковості забезпечила високу стабільність системи [2]. Таким чином, створена платформа відповідає вимогам щодо точності та пояснюваності, формуючи надійну технологічну базу для впровадження, що розглядається у наступному розділі.

РОЗДІЛ 5. РОЗРОБЛЕННЯ СТАРТАП ПРОЄКТУ

Роль у цифровій трансформації та автомобільному бізнесі

Інтелектуальні системи є драйвером цифрової трансформації, накопичуючи та аналізуючи дані для управління бізнесом. У сфері автомобільного CRM це дозволяє інтегрувати інформацію з дилерських мереж, сервісних центрів та страхових компаній. Використання алгоритмів машинного навчання (сегментація, LTV, відтік) трансформує роботу з клієнтами від реактивної до проактивної. Ключовою особливістю є використання XAI (SHAP, LIME) для пояснення прогнозів, що підвищує довіру менеджерів до автоматизованих рекомендацій та зменшує ризики, пов'язані з моделями-«чорними скриньками». Багатопотоковий конвеєр дозволяє паралельно обробляти мільйони записів, забезпечуючи своєчасність аналітики [2, 5].

Типові бізнес-кейси

Застосування інтелектуальної платформи безпосередньо впливає на KPI автобізнесу. Персоналізація пропозицій, що базується на ML-сегментації та прогнозуванні реакції, збільшує конверсію. Моделі відтоку (Churn Prediction), підсилені SHAP-поясненнями, дозволяють вчасно ідентифікувати ризикових клієнтів та причини їхнього невдоволення. Крім того, прогнозування попиту оптимізує завантаження сервісних слотів, зменшуючи простої обладнання [3].

Порівняння з традиційними CRM та масштабування

На відміну від традиційних CRM, що фіксують минулі події, інтелектуальна платформа надає прогнозну аналітику майбутньої поведінки (LTV, ймовірність покупки). Це еволюційний крок, що базується на пояснюваних моделях, а не лише на «сухих» звітах. Завдяки модульній архітектурі та уніфікованим ML-компонентам, рішення має високу портативність. Воно може бути адаптоване для банківського сектору (кредитний скоринг), страхування (виявлення шахрайства) та рітейлу (рекомендаційні системи) зі зміною лише налаштувань моделей та специфічних ознак [14].

Практичний ефект та обмеження

Впровадження системи надає інструменти для всіх рівнів управління: керівництво отримує стратегічні прогнози доходів, маркетинг — точне таргетування, а сервіс — оптимізовані графіки. Проте ефективність платформи залежить від якості історичних даних (повнота записів про продажі та візити) та організаційної готовності компанії інтегрувати аналітичні висновки в бізнес-процеси. Навчання персоналу роботі з поясненнями XAI є критичним фактором успіху [1].

Підсумки підрозділу

Інтелектуальна CRM-платформа забезпечує високу ефективність завдяки синергії паралельної обробки даних, точних ML-прогнозів та їх прозорої інтерпретації. Це створює основу для прийняття обґрунтованих бізнес-рішень та цифрової трансформації підприємства.

5.2. Порівняння витрат на традиційні та інтелектуальні рішення

Вихідні припущення для економічного аналізу

Економічна оцінка ефекту від впровадження інтелектуальної CRM базується на формалізованих початкових припущеннях щодо масштабу дилерської мережі та характеристик її операційної діяльності. Типовий дилерський центр підтримує клієнтську базу в 25-40 тисяч осіб, з 18-25 тисячами візитів на обслуговування на рік та 2-5 тисячами угод з продажу автомобілів та F&I. Це створює середній обсяг як транзакційних, так і поведінкових даних, що фактично встановлює певні потенційні обмеження для аналітичної системи.

Горизонти планування визначено як 1 рік (операційний ефект), 3 роки (окупність та стабілізація витрат) та 5+ років (стратегічний вплив на лояльність та цінність клієнтської бази). Аналіз передбачає низку припущень: стабільність обсягів попиту, відсутність значних цінових шоків, незмінність підходів до обліку витрат, а також рівномірне зростання цифрової активності клієнтів.

Нижче наведено умовний псевдокод для первинної оцінки приросту економічного ефекту:

```
clients = 30000
visits_per_year = 20000
conversion_growth = 0.03
avg_profit = 120$

annual_effect = visits_per_year * conversion_growth * avg_profit
```

Рис. 45 Розрахунок річного економічного ефекту від зростання конверсії

Отриманий результат використовується як базовий орієнтир для подальших сценарних розрахунків.

Структура витрат на традиційні рішення

Традиційна структура витрат CRM-ML та XAI являє собою довгий хвіст, в якому домінують ліцензійні збори та регулярні підписки. Потім йдуть витрати на ручну аналітику: оплата праці аналітиків, підготовка зведень Excel, створення щомісячних та щоквартальних звітів, залучення зовнішніх консультантів для оцінки динаміки продажів та активності клієнтів. Величезна група непрямих витрат залишається без можливостей прогнозної аналітики, а не персоналізації, втрачених через неефективне планування навантаження на сервіси або помилки у визначенні ризикованих клієнтів.

Типовий ручний процес можна подати у вигляді псевдокоду:

```
for each client in database:
    load_excel_report()
    manually_check_sales_trends(client)
    assign_status_based_on_expert_judgement()
```

Рис. 46 Приклад традиційного ручного процесу аналізу клієнтів

Такі процедури підвищують трудомісткість, збільшують часові затрати та знижують точність управлінських рішень.

Структура витрат на запропоновану інтелектуальну систему

Запропонована структура витрат на інтелектуальну CRM містить як капітальні витрати, так і операційні витрати. CAPEX включає розробку архітектури програмного забезпечення, інтеграцію з існуючими IT-сервісами в дилерській мережі, початкове налаштування баз даних та підготовчі роботи з модулями ML/XAI. OPEX включає хмарну інфраструктуру (обчислювальні ресурси, сховище,

мережа), витрати на технічну підтримку, а також постійний моніторинг якості даних разом з періодичним перенавчанням моделей. Вона також включає навчання персоналу правильному використанню прогнозної аналітики.

Порівняльний аналіз витрат

Можна провести порівняльний аналіз витрат на традиційну CRM та інтелектуальну CRM-платформу з точки зору капітальних/операційних витрат (CAPEX/OPEX), а також за моделями їх розгортання. Для локального рішення потрібні величезні капітальні витрати: сервери, корпоративні ліцензії, інтеграція. У цьому випадку майже всі капітальні витрати виключаються в інтелектуальній платформі SaaS; витрати перекладаються на операційні витрати: підписка, хмарні обчислення для підтримки ML/XAI. Багатопотокова архітектура, стиснута Parquet, зменшує витрати на зберігання та обробку великих даних.

У локальному сценарії традиційні системи мають високі капітальні витрати (CAPEX) та зростаючі операційні витрати (OPEX) (адміністрування, енергоспоживання). Модель SaaS інтелектуальної CRM забезпечує швидке впровадження та еластичність ресурсів, а витрати залежать від кількості користувачів (N) та обсягу даних (D).

Традиційні системи масштабуються гірше: зі збільшенням D витрати зростають лінійно або експоненціально. Інтелектуальна SaaS-платформа масштабується краще завдяки паралельній обробці. Це забезпечує нижчу сукупну вартість володіння (TCO) зі зростанням великих даних.

Оцінка економічного ефекту від впровадження інтелектуальної системи

Економічний ефект від інтелектуальної CRM-платформи на базі Big Data, ML та XAI формується як у процесі збільшення доходів, так і зниження операційних (адміністративних) витрат. Прямі ефекти виражаються у збільшенні конверсій завдяки персоналізованим рекомендаціям, збільшенні LTV завдяки пріоритезації цінних клієнтів та покращенні утримання клієнтів завдяки прогнозуванню відтоку. Непрямі ефекти досягаються за рахунок оптимізації маркетингових витрат (скорочення неефективних кампаній), планування навантаження сервісних

ресурсів, а також автоматизації рутинних операцій за допомогою багатопотокового конвеєра даних.

XAI на основі SHAP підвищує довіру до моделі та зменшує «вартість помилок», оскільки дозволяє перевіряти внесок ознак у прогноз. Логіку рішень може перевірити менеджер за допомогою SHAP, таким чином коригуючи бізнес-процеси.

Псевдокод (скорочено):

```
model = load_model()
explainer = shap.TreeExplainer(model)
shap_vals = explainer.shap_values(client_features)
```

Рис. 47 Обчислення локальних SHAP-значень

SHAP показує, як окремі параметри впливають на прогноз, забезпечуючи прозорість та обґрунтовані рішення.

Розрахунок інтегральних економічних показників

Фінансова окупність та економічна стійкість інтелектуальної CRM, яка поєднує BigData, багатопоточність, машинне навчання та SHAP, аналізується на основі інтегральних фінансових показників. Термін окупності інвестицій охоплює час, необхідний для відшкодування початкових інвестицій I_0 з чистих грошових потоків CFt. Оскільки система забезпечує постійні прогнози та швидко виявляє неефективні маркетингові дії (скорочення витрат), її PVR у порівнянні з класичними CRM нижчий. NPV визначає різницю між дисконтованими грошовими потоками та початковими інвестиціями: $NPV = \Sigma(CFt/(1+r)\text{^}t) - I_0$. Показник дає змогу оцінити довгострокову вигоду SaaS-рішення з урахуванням вартості капіталу. ROI демонструє співвідношення чистого прибутку до загальних інвестицій і підтверджує ефективність використання платформи, яка завдяки XAI знижує ризики помилкових рішень. Приклад обчислення NPV та ROI може бути реалізований у вигляді простих функцій, що оперують ставкою дисконту, грошовими потоками та обсягом початкових вкладень.

Порівняння сценаріїв «традиційне рішення» vs «інтелектуальна CRM з XAI»

Різні часові горизонти, що безпосередньо знижують витрати та збільшують економічний ефект від впровадження інтелектуальної CRM, посиленої багатопотоковим конвеєром великих даних та XAI. У короткостроковій перспективі, пряме зниження витрат шляхом автоматизації рутинних процесів – у нецільовому маркетингу скорочення, що включає прогноз відтоку клієнтів, що надається швидким операційним багатопотоковим конвеєром, який запускає точні кампанії утримання – довгострокове збільшення доходів завдяки підвищенню LTV та покращенню утримання клієнтів, досягнутому завдяки високоякісним моделям машинного навчання з прозорими поясненнями SHAP. Аналітика (NPV, ROI, PBP) все ще дуже вразлива до елементів ризику: якості історичних даних, готовності організації до впровадження аналітичних рекомендацій та вартості помилок. XAI зменшує такі невизначеності, тим самим роблячи нас більш впевненими в прогнозі, а отже, стабілізуючи очікувану ROI.

Підсумки підрозділу

Порівняльний аналіз витрат виявив величезну різницю між застарілою CRM та інтелектуальною платформою. Класичні рішення мають високі капітальні витрати, а також значні витрати на ручну аналітику. SaaS-підхід перекладає фінансове навантаження на операційні витрати та забезпечує масштабованість. Багатопотокова обробка та оптимізовані формати даних знижують сукупну вартість володіння (TCO). Інтелектуальна система збільшує доходи завдяки персоналізації та знижує витрати завдяки автоматизації. Використання XAI (SHAP) знижує ризики та стабілізує рентабельність інвестицій (ROI).

5.3. Перспективи впровадження та масштабування

Етапи впровадження інтелектуальної CRM-платформи

Впровадження починається з пілотної фази для перевірки якості даних, тестування функціонування багатопотокового конвеєра та модулів ML/XAI.[153] Інтеграція DMS/ERP через API та нормалізація даних успішно виконуються після підтвердження концепції (PoC). Потім масштабується по всій мережі, де можна

досягти стабільних результатів прогнозування разом з автоматизацією бізнес-процесів.

Організаційні аспекти впровадження

Інтелектуальна CRM трансформує продажі й сервіс: процеси переходять до проактивної моделі, використовують ML-скоринг, персоналізацію та XAI-пояснення. Персонал навчається працювати з прогнозами й інтерпретаціями моделей. Data/ML-команда забезпечує моніторинг даних, перенавчання моделей і стабільність багатопотокового конвеєра.

Технічні можливості масштабування

Система масштабується за допомогою багатопотокового конвеєра даних, що дозволяє паралельну обробку великих масивів та горизонтальне масштабування сховища та сервісів. Модулі ML/XAI виконують навчання та SHAP-аналіз у потоках, підтримуючи часткове перенавчання. Модульна архітектура плюс REST/JSON API спрощують додавання нових моделей або джерел даних.

Ризики та бар'єри впровадження

Оцінка ефективності інтелектуальної CRM обмежується якістю історичних даних: пропуски й шум знижують точність прогнозів та довіру до аналітики. Додатковим бар'єром є організаційна неготовність і недовіра до ML-моделей. XAI частково знімає опір, але потребує підготовки персоналу для коректної інтерпретації пояснень.

Стратегічні перспективи розвитку системи як SaaS-продукту

Розумна CRM-платформа SaaS поєднує масштабовану багатопотокову обробку фактів із зрозумілими прозорими моделями машинного навчання, що дозволяє швидко прогнозувати та зміцнювати довіру користувачів за допомогою пояснень. Архітектура швидко адаптується між банківським, страховим, електронним та B2B-секторами, підтримує інтеграцію API та створює відкриту екосистему для розширення функціональності.

Соціально-економічний ефект від впровадження інтелектуальних систем

Персоналізація, проактивність та швидке реагування виводять сервіс на новий рівень інтелектуальної CRM. XAI знижує ризики, роблячи рішення прозорими. Платформа є основою цифрової трансформації, яка переводить дилерську мережу на прогнозну аналітику та підвищує стратегічну ефективність.

Підсумки підрозділу

Інтелектуальна CRM забезпечує масштабованість, прозорість рішень через XAI, знижує ризики, підвищує якість сервісу та сприяє цифровій трансформації дилерської мережі.

5.4. Висновки до розділу

Проведений аналіз підтверджує економічну та практичну перевагу інтелектуальних CRM-систем над традиційними рішеннями. Впровадження платформи забезпечує стратегічний перехід від реактивної моделі обслуговування до проактивної персоналізованої взаємодії. Консолідація даних з DMS, ERP та телематики дозволяє формувати точні прогнози (LTV, відтік), а інтеграція XAI (SHAP) вирішує проблему «чорної скриньки», підвищуючи довіру користувачів до автоматизованих рішень [1, 8].

Економічна ефективність базується на зміні структури витрат: архітектура SaaS трансформує капітальні вкладення (CAPEX) в операційні (OPEX), а технології оптимізації даних (Parquet, багатопотоковість) знижують сукупну вартість володіння (TCO) [2]. Зменшення ризиків завдяки прозорості прогнозів стабілізує ROI. Система характеризується високою масштабованістю та комерційним потенціалом, дозволяючи адаптувати рішення для суміжних галузей, таких як страхування та ритейл [5].

ВИСНОВКИ

Дипломна робота присвячена розробці інтелектуальної CRM-системи для аналізу великих клієнтських даних у сфері автомобільного бізнесу. Об’єктом дослідження визначено процес опрацювання таких даних, а метою — створення системи з багатопотоковою обробкою, машинним навчанням і SHAP-аналізом. Усі поставлені цілі досягнуто.

Створено концептуальну архітектуру CRM-платформи, здатної консолідувати інформацію з DMS, ERP, телематики та онлайн-каналів, забезпечуючи обробку структурованих та неструктурованих даних. Для зберігання та аналітики обрано формат Parquet, для API-комунікацій – JSON, а для універсального обміну – CSV.

Основним рішенням був багатопотоковий конвеєр даних, який паралельно виконує імпорт, очищення, перетворення та підготовку зразків, тим самим збільшуючи пропускну здатність зі зменшенням затримок. Це забезпечило ефективну роботу модулів машинного навчання та пояснюваного штучного інтелекту.

Розумний модуль виконує прогнозування відтоку клієнтів, оцінку LTV, сегментацію клієнтів та оцінку лідів на основі ансамблевих методів. Бекенд реалізовано на Laravel з демонстраційним інтерфейсом на OctoberCMS, інтеграція з ML/XAI здійснюється через REST/JSON API.

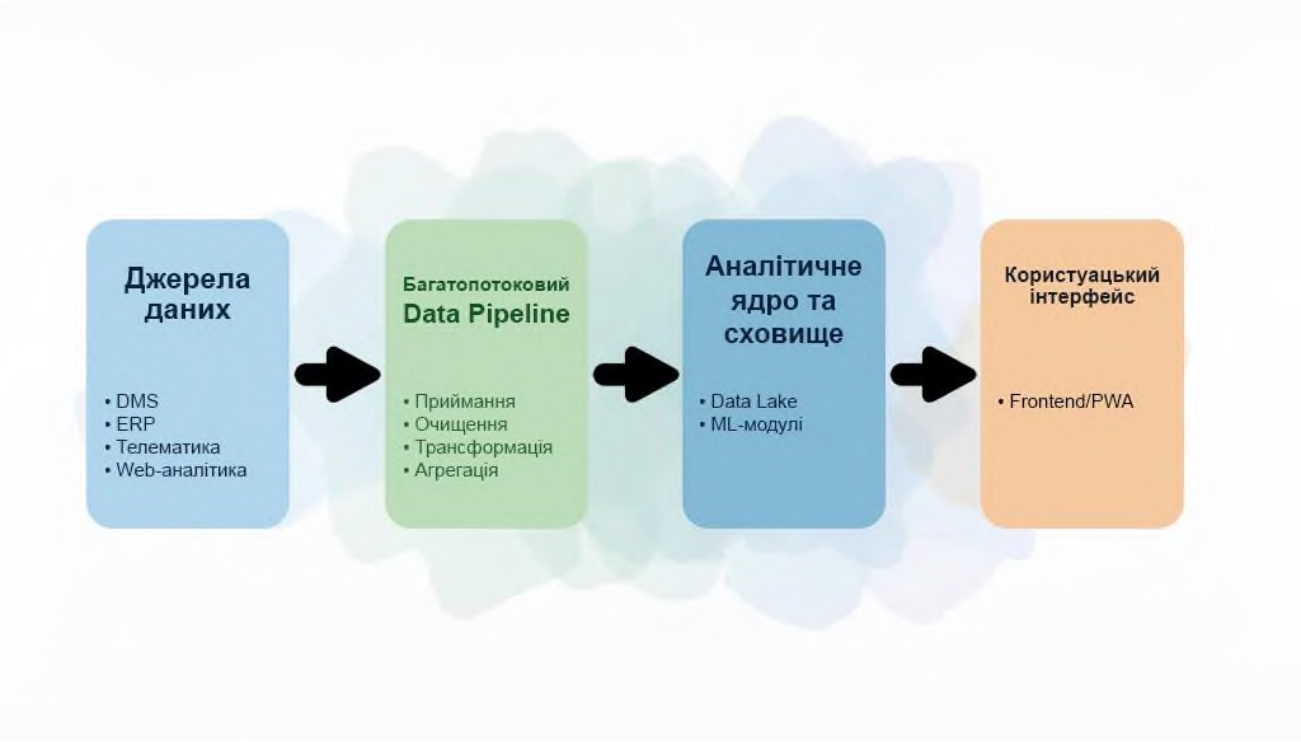
Для вирішення проблеми «чорної скриньки» було впроваджено SHAP — систему локальних та глобальних пояснень моделей, інтегровану в Data Pipeline. Клієнтська частина створена як PWA з рольово-орієнтованим інтерфейсом та офлайн-підтримкою.

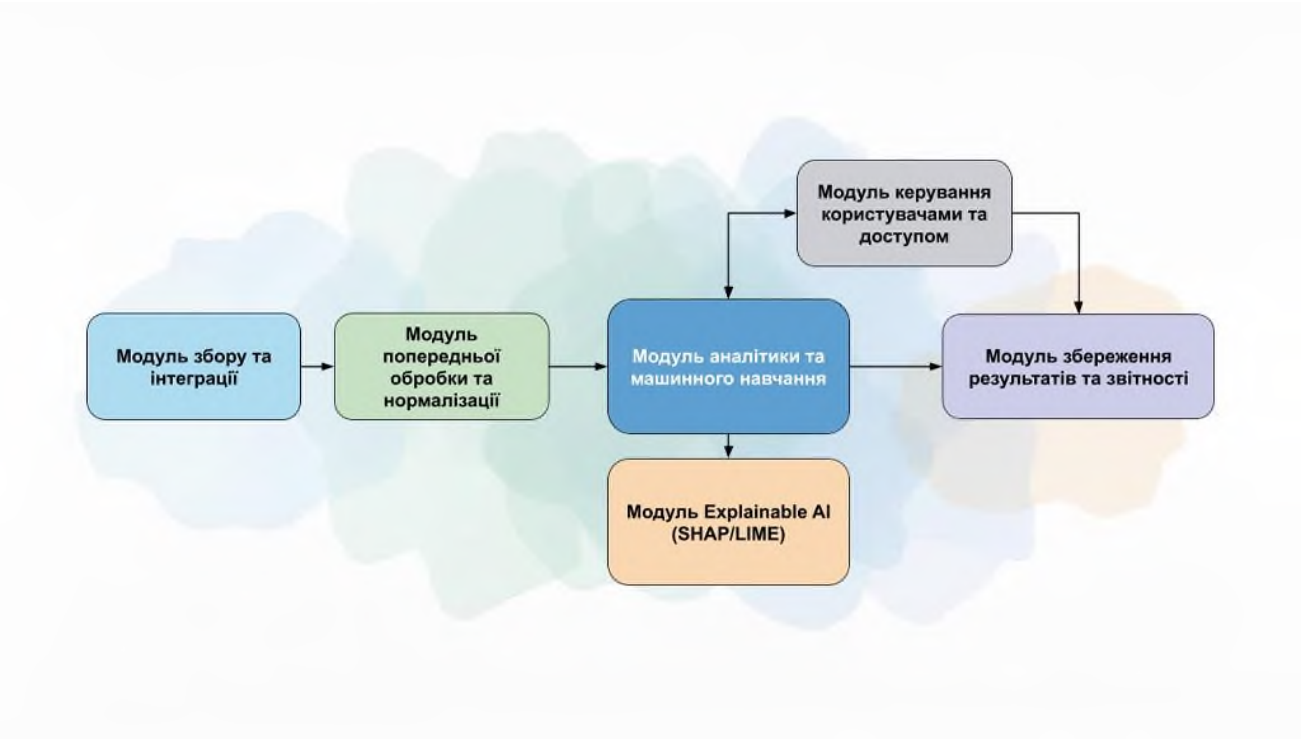
Тести довели, що конвеєр є коректним, масштабованим та стійким до зростання обсягів даних. Економічний аналіз також доводить, що він кращий за класичні CRM, оскільки додаються операційні витрати, зменшуються сукупні витрати на володіння (TCO), а доходи збільшуються завдяки персоналізації. Це робить рішення комерційно привабливим, оскільки воно може працювати як SaaS.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

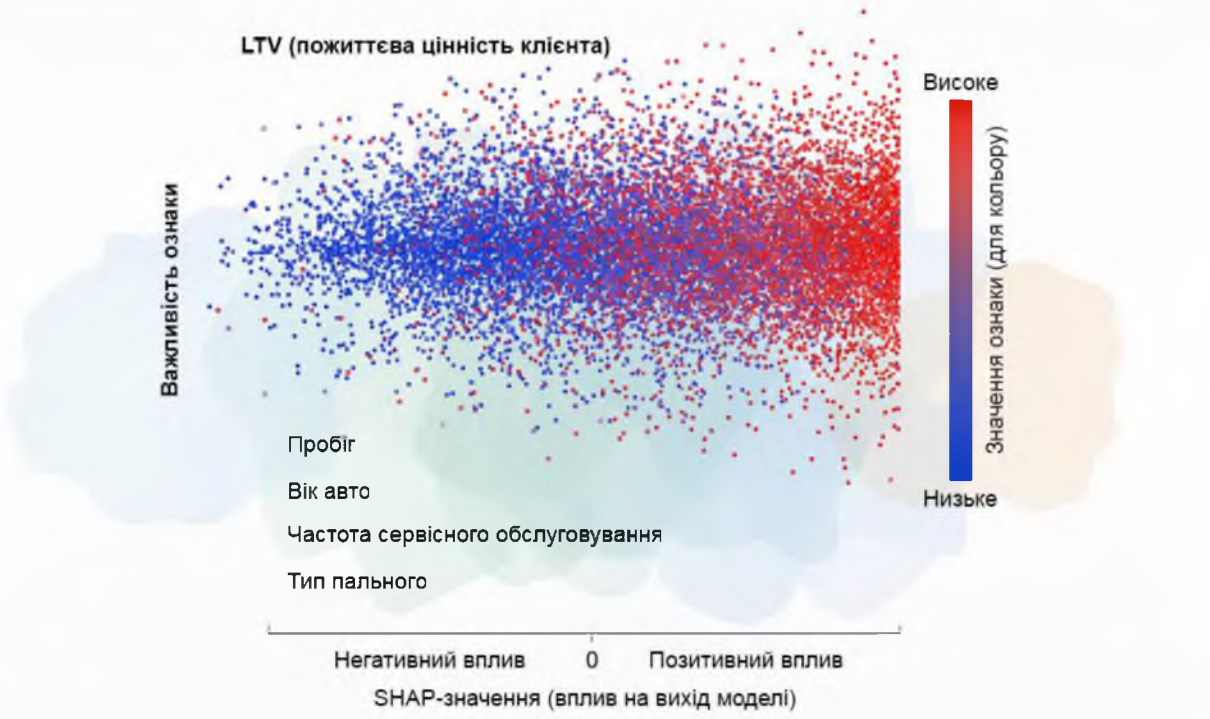
1. "Interpretable Machine Learning: A Guide for Making Black Box Models Explainable" by Christoph Molnar – 2020
2. Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems" by Martin Kleppmann – 2017
3. "Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems" by Aurélien Géron – 2022
4. "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython" by Wes McKinney – 2022
5. "Customer Relationship Management: Concepts and Technologies" by Francis Buttle and Stan Maklan – 2019
6. "Laravel: Up and Running: A Framework for Building Modern PHP Apps" by Matt Stauffer – 2023
7. "Ensemble Methods for Classification and Regression" by Zhi-Hua Zhou – 2021
8. SHAP (SHapley Additive exPlanations) Documentation – Online Documentation – https://shap.readthedocs.io/en/latest/
9. The Laravel Framework – Official Documentation – https://laravel.com/docs
10. Progressive Web Apps (PWA) – MDN Web Docs – https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps
11. Apache Parquet – Format Documentation – https://parquet.apache.org/documentation/
12. CMS October: Developer Guide – Core Concepts – https://octobercms.com/docs/api/core
13. Apache Spark Programming Guide – The Apache Software Foundation – https://spark.apache.org/docs/latest/programming-guide.html
14. XGBoost Documentation – Tutorials and API Reference – https://xgboost.readthedocs.io/en/stable/

ДОДАТКИ


Джерела даних
• DMS
• ERP
• Телематика
• Web-аналітика
Багатопотоковий Data Pipeline
• Приймання
• Очищення
• Трансформація
• Агрегація
Аналітичне ядро та сховище
• Data Lake
• ML-модулі
Користуацький інтерфейс
• Frontend/PWA


Модуль керування користувачами та доступом
Модуль збору та інтеграції
Модуль попередньої обробки та нормалізації
Модуль аналітики та машинного навчання
Модуль збереження результатів та звітності
Модуль Explainable AI (SHAP/LIME)


Базове очікуване значення = 0.5
ПОЗИТИВНИЙ ВПЛИВ
(ЗБІЛЬШУЄ ПРОГНОЗ)
Пробіг = 150000 км
+0.2
Частота сервісу = 1 рік
ПРОГНОЗ: висока
ймовірність відтоку
Значення виходу = 0.9
НЕГАТИВНИЙ ВПЛИВ
(ЗМЕННУЄ ПРОГНОЗ)


LTV (пожиттєва цінність клієнта)
Важливість ознаки
Високе
Значення ознаки (для кольору)
Низьке
Пробіг
Вік авто
Частота сервісного обслуговування
Тип пального
Негативний вплив
0
Позитивний вплив
SHAP-значення (вплив на вихід моделі)


Audi Львів
Ймовірність відтоку
(сегментовано)
12.7%
порівняно з минулим місяцем
Нові клієнти:
5%
Активні:
10%
Високовартісні:
18%
Пожиттєва цінність клієнта
(LTV)
$1,250
порівняно з останніми 6 місяцями
Динаміка LTV за останні 6 місяців
Середня тривалість
сервісного візиту (хвилини)
45 min
на 6 хв менше
Топ-5 глобальних факторів, що впливають на відтік
(важливість SHAP)
LTV (пожиттєва цінність клієнта)
4%
Пробіг
16%
Вік авто
18%
Регулярність сервісу
25%
Історія скарг
17%
Тип пального
2%
0
10
20
30
40
50
Індивідуальний ризик
відтоку клієнта
ID клієнта: 7890
Пробіг = 100K km (червоний, +)
Част. сервісу = 6 міс (синій, −)
Прогноз: високий ризик відтоку


Audi Львів
Інформація про клієнта
ID клієнта:
7890
Ім'я:
Андрій Коваленко
Клієнт з:
14 березня 2019
Остання активність:
22 жовтня 2025, 17:42
Прогноз відтоку
85%
+12%
Високий ризик
Динаміка ймовірності (6 місяців)
Історія взаємодій
22.10.2025
Онлайн-чат: звернення щодо проблеми з шинами
18.08.2025
Придбано нові шини
02.07.2025
Запис на сервіс: планове ТО (не завершено)
15.03.2025
Телефонний дзвінок: уточнення щодо гарантії
12.11.2024
Сервісний візит: заміна мастила + фільтрів
05.10.2023
Сервісний візит: регламентне ТО + діагностика шин
Ключові фактори ризику (Local SHAP)
Повторні проблеми з авто
Незавершений запис на сервіс
Незадоволеність: нещодавнє звернення в чат
Недавня покупка в сервісі
Тип пального: дизель
Рекомендація:
Запропонуйте програму лояльності зі знижками та опрацюйте останню скаргу клієнта.

```
class ShapLocal(Base):
    __tablename__ = 'shap_local_explanations'
    id = Column(Integer, primary_key=True)
    client_id = Column(Integer, index=True)
    model_version = Column(String(50))
    evaluation_date = Column(DateTime, default=datetime.datetime.utcnow)
    shap_values = Column(JSON)
    prediction_score = Column(Float)

import pandas as pd
from concurrent.futures import ThreadPoolExecutor
import logging

def process_batch(chunk: pd.DataFrame) -> pd.DataFrame:
    chunk.fillna(chunk.mean(), inplace=True)
    chunk['active_client_flag'] = chunk['visits_count'] > 5
    logging.info(f"Processed batch size: {len(chunk)}")
    return chunk

def parallel_data_pipeline(file_path, chunk_size=100000, max_workers=4):
    processed_batches = []
    with ThreadPoolExecutor(max_workers=max_workers) as executor:
        futures = []
        for chunk in pd.read_csv(file_path, chunksize=chunk_size):
            futures.append(executor.submit(process_batch, chunk))
        for future in futures:
            processed_batches.append(future.result())
    final_data = pd.concat(processed_batches)
    return final_data
```

```
agg_features = df.groupby("client_id")["amount"].agg(
    total_spent='sum',
    avg_deal_value='mean',
    deals_count='count'
)

latest_deal_date = df.groupby("client_id")["date"].max()
days_since_last_deal = (pd.to_datetime('today') - latest_deal_date).dt.days

agg_features['days_since_last_deal'] = days_since_last_deal



import xgboost as xgb
import shap
from sklearn.model_selection import train_test_split

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

model = xgb.XGBClassifier(
    n_estimators=100,
    max_depth=5,
    tree_method='hist'
)
model.fit(X_train, y_train)

explainer = shap.TreeExplainer(model)

shap_values = explainer.shap_values(X_test)

def compute_shap_with_caching(client_batch, model_id, cache_service):
```

```
    results = []
    with ThreadPoolExecutor(max_workers=4) as executor:
        for client in client_batch:
            if cache_service.exists(client.id, model_id):
                results.append(cache_service.get(client.id, model_id))
            else:
                future = executor.submit(calculate_single_shap, client.features, model_id)
                results.append(future.result())
                cache_service.store(client.id, model_id, future.result())
    return merge_results(results)
```

```
{
    "client_id": 1024,
    "model_name": "churn_xgb"
}
```

```
{
    "status": "success",
    "data": {
        "client_id": 1024,
        "prediction": {
            "risk_score": 0.87,
            "risk_level": "High"
        },
        "explanation_shap": [
            {
                "feature": "days_since_last_service",
                "value": 450,
                "shap_contribution": 0.45,
                "impact": "positive"
```

```
        },
        {
          "feature": "car_mileage_km",
          "value": 170000,
          "shap_contribution": 0.22,
          "impact": "positive"
        },
        {
          "feature": "repeat_client",
          "value": 1,
          "shap_contribution": -0.15,
          "impact": "negative"
        }
      ]
    }
}

const CACHE_NAME = 'crm-cache-v1';
const urlsToCache = [
  '/',
  '/index.html',
  '/styles/main.css',
  '/scripts/app.js'
];

self.addEventListener('install', event => {
  event.waitUntil(
    caches.open(CACHE_NAME)
      .then(cache => {
        console.log('Opened cache');
```

```
        return cache.addAll(urlsToCache);
      })
  );
});

self.addEventListener('fetch', e => {
    e.respondWith(
        caches.match(e.request).then(r => r || fetch(e.request))
    );



import pandas as pd
import numpy as np
from concurrent.futures import ThreadPoolExecutor
import logging
from typing import List

logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

class DataPipeline:
    def __init__(self, input_file: str, chunk_size: int = 10000):
        self.input_file = input_file
        self.chunk_size = chunk_size

    def clean_data(self, df: pd.DataFrame) -> pd.DataFrame:
        numeric_cols = df.select_dtypes(include=[np.number]).columns
        df[numeric_cols] = df[numeric_cols].fillna(df[numeric_cols].mean())
```

```
    df.fillna({'segment': 'Unknown', 'car_model': 'Other'}, inplace=True)
    return df

def feature_engineering(self, df: pd.DataFrame) -> pd.DataFrame:
    if 'last_visit_date' in df.columns:
        df['last_visit_date'] = pd.to_datetime(df['last_visit_date'])
        df['days_since_service'] = (pd.Timestamp.now() - df['last_visit_date']).dt.days

    if 'visits_count' in df.columns:
        df['is_active'] = (df['visits_count'] > 2).astype(int)

    return df

def process_batch(self, chunk: pd.DataFrame) -> pd.DataFrame:
    try:
        df_clean = self.clean_data(chunk)
        df_features = self.feature_engineering(df_clean)
        logging.info(f"Processed batch of size {len(df_features)}")
        return df_features
    except Exception as e:
        logging.error(f"Error processing batch: {e}")
        return pd.DataFrame()

def run_pipeline(self, max_workers: int = 4):
    processed_chunks = []

    with ThreadPoolExecutor(max_workers=max_workers) as executor:
        futures = []
        for chunk in pd.read_csv(self.input_file, chunksize=self.chunk_size):
            futures.append(executor.submit(self.process_batch, chunk))
```

```
                for future in futures:
            result = future.result()
            if not result.empty:
                processed_chunks.append(result)

        if processed_chunks:
            final_df = pd.concat(processed_chunks)
            logging.info(f"Pipeline finished. Total records: {len(final_df)}")
            return final_df
        return pd.DataFrame()
if __name__ == "__main__":

import xgboost as xgb
import shap
import pandas as pd
import json

class ChurnPredictor:
    def __init__(self):
        self.model = xgb.XGBClassifier(
            n_estimators=100,
            max_depth=5,
            tree_method='hist',
            enable_categorical=True
        )
        self.explainer = None

    def train(self, X_train, y_train):
        self.model.fit(X_train, y_train)
        self.explainer = shap.TreeExplainer(self.model)
```

```
        print("Model trained and Explainer initialized.")

    def predict_and_explain(self, client_data: pd.DataFrame):
        risk_score = float(self.model.predict_proba(client_data)[:, 1][0])

        shap_values = self.explainer.shap_values(client_data)

        explanation = []
        feature_names = client_data.columns

        client_shap = shap_values[0]

        for name, value, impact in zip(feature_names, client_data.iloc[0], client_shap):
            explanation.append({
                "feature": name,
                "value": float(value),
                "shap_contribution": float(impact),
                "impact": "positive" if impact > 0 else "negative"
            })

        explanation.sort(key=lambda x: abs(x['shap_contribution']), reverse=True)

        response = {
            "prediction": {
                "risk_score": round(risk_score, 4),
                "risk_level": "High" if risk_score > 0.7 else "Medium" if risk_score > 0.3 else
"Low"
            },
            "explanation_shap": explanation[:5]
        }
```

```
        return response

from fastapi import FastAPI, HTTPException
from pydantic import BaseModel
from typing import List, Dict, Any

app = FastAPI(title="Intelligent CRM API")

class ClientFeatures(BaseModel):
    client_id: int
    days_since_last_service: int
    car_mileage_km: int
    visits_count: int
    repeat_client: int

@app.post("/api/v1/ml/score")
async def get_score_and_explanation(features: ClientFeatures):
    try:
        return {
            "status": "success",
            "data": {
                "client_id": features.client_id,
                "prediction": {
                    "risk_score": 0.87,
                    "risk_level": "High"
                },
                "explanation_shap": [
                    {
                        "feature": "days_since_last_service",
                        "value": features.days_since_last_service,
```

```
                "shap_contribution": 0.45,
                "impact": "positive" # Підвищує ризик
            },
            {
                "feature": "car_mileage_km",
                "value": features.car_mileage_km,
                "shap_contribution": 0.22,
                "impact": "positive"
            }
        ]
    }
  }
except Exception as e:
  raise HTTPException(status_code=500, detail=str(e))
```