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: **Прогнозування динаміки продажів супермаркету за допомогою алгоритмів машинного навчання**

Виконав: студент VI курсу, групи КН-63м
спеціальності
122 – "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)

Рудниченко О.В.
(прізвище та ініціали)

Керівник Паславський М.М.
(прізвище та ініціали)

Рецензент Сторожук О.А.
(прізвище та ініціали)

Львів – 2025 р.

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 КН

Борецька І. Б.
"10" грудня 2025 року

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Рудниченко Олександр Вадимович
(прізвище, ім'я, по батькові)

1. Тема роботи **Прогнозування динаміки продажів супермаркету за допомогою алгоритмів машинного навчання**

керівник Паславський М.М., канд. техн. наук, асистент.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від 29.04. 2025 р.№ С-288

2. Термін подання студентом роботи 10.12. 2025 р.

3. Вихідні дані до роботи:

- вивчити предметну область, проаналізувати існуючі фактори та методи моделювання, а також відповідні програмні продукти;
- розглянути і використати алгоритми, які лежать в основі математичної моделі прогнозування динаміки продажів супермаркету;
- спроектувати інтелектуальну систему з допомогою мови програмування Python та відповідних бібліотек;
- представити результати роботи інформаційної системи.

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)
додаток А, додаток Б

6. Дата видачі завдання 1 травня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Огляд літературних даних та інших джерел згідно досліджуваної теми	01.05-30.05.2025	виконано
2	Аналіз досліджуваної теми та вибір відповідних варіантів її розробки	01.06-30.06. 2025	виконано
3	Постановка задачі та її формалізація	01.07-30.07. 2025	виконано
4	Вибір та обґрунтування методів і засобів проведення дослідження	01.08-30.08. 2025	виконано
5	Розроблення концептуальної схеми реалізації завдання	01.09-15.09. 2025	виконано
6	Програмна реалізація завдання	16.09-30.10. 2025	виконано
7	Тестування програмного продукту та отриманих результатів	01.11-15.11. 2025	виконано
8	Розробка пояснювальної записки магістерської роботи	16.11-30.11. 2025	виконано
9	Корегування пояснювальної записки згідно вимог, розроблення презентації	01.12-09.12. 2025	виконано

Студент _____ Рудниченко О.В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Паславський М.М.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Магістерська робота містить 86 сторінок пояснювальної записки, 12 рисунків, 1 таблицю, 2 додатки, 16 джерел.

Робота присвячена розробці інтелектуальної системи прогнозування динаміки продажів у супермаркетах на основі алгоритмів машинного навчання. У ній досліджуються сучасні підходи до прогнозування у роздрібній торгівлі, аналізується ринок та цільова аудиторія українських підприємств. Розроблено структуру бізнес-моделі та архітектуру системи, яка враховує сезонність, маркетингові акції та поведінку споживачів. Проект має на меті підвищити точність та адаптивність прогнозів, що дозволить оптимізувати запаси, зменшити втрати та покращити управлінські процеси у торгівлі.

Ключові слова: *машинне навчання, регресійна модель, Python, Scikit-Learn, Seaborn, Streamlit*.

ABSTRACT

The thesis contains 86 pages of explanatory note, 12 figures, 1 table, 2 appendix, 16 used literary sources.

The work is devoted to the development of an intelligent system for forecasting the dynamics of sales in supermarkets based on machine learning algorithms. It explores modern approaches to forecasting in retail trade, analyzes the market and target audience of Ukrainian enterprises. The structure of the business model and the architecture of the system have been developed, which takes into account seasonality, marketing campaigns and consumer behavior. The project aims to increase the accuracy and adaptability of forecasts, which will allow optimizing inventories, reducing losses and improving management processes in trade.

Keywords: *machine learning, regression model, Python, Scikit-Learn, Seaborn, Streamlit*.

ТЕХНІЧНЕ ЗАВДАННЯ

В дипломній роботі потрібно розробити інформаційну систему пронозування динаміки продажу супермаркету, для цього потрібно вирішити такі завдання.

1. Провести аналіз існуючих методів та моделей прогнозування динаміки продажів у роздрібній торгівлі з використанням машинного навчання.

2. Вивчити особливості бізнес-процесів у сфері роздрібної торгівлі та визначити ключові потреби цільової аудиторії щодо системи прогнозування.

3. Розробити математичну модель інтелектуальної системи прогнозування продажів.

4. Створити архітектуру програмного забезпечення системи та реалізувати прототип із використанням відповідних алгоритмів машинного навчання.

5. Провести тестування розробленої системи з метою оцінки її точності та зручності у використанні.

.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

ARIMA (Autoregressive Integrated Moving Average) – дозволяє моделювати часові залежності та сезонні коливання;

CSV (Comma-Separated Values) – текстовий формат;

EDA (Exploratory data analysis) – попередній аналіз даних;

DNN (Deep Neural Networks) – глибокі нейронні мережі;

GRU (Gated Recurrent Units) – тип рекурентної нейронної мережі, що моделює послідовні дані;

KNN (K-Nearest Neighbors) – метод k-найближчих сусідів;

Numpy – бібліотека для роботи з масивами даних;

LSTM (Long Short-Term Memory) – різновид рекурентної нейронної мережі, призначений для ефективного запам’ятовування довготривалих залежностей у послідовних даних;

ML (machine learning) – машинне навчання;

Pandas – бібліотека для обробки та аналізу даних;

Random Forest – ансамблевий алгоритм машинного навчання;

RNN (Recurrent Neural Network) – рекурентна нейронна мережа, яка обробляє послідовні дані;

Scikit-Learn – бібліотека для машинного навчання;

Seaborn – бібліотека для проведення статистичної візуалізації даних;

Streamlit – бібліотека для створення графічного веб–інтерфейсу системи;

ШІ – штучний інтелект.

XGBoost – алгоритм градієнтного бустингу на основі дерев рішень, призначений для побудови моделей машинного навчання на структурованих даних;

SCM (Supply Chain Management) – система управління ланками постачання;

SVR (Support Vector Regression) – метод машинного навчання, що базується на принципах опорних векторів і використовується для побудови регресійних моделей.

ВСТУП

Актуальність дипломної роботи

Актуальність даної роботи зумовлена стрімким розвитком технологій аналізу даних у сучасному бізнес-середовищі. У світлі зростаючої конкуренції на ринку комп'ютерної техніки необхідність у точному прогнозуванні цін стала важливою не лише для великих корпорацій, але й для малих підприємств. Використання алгоритмів машинного навчання дозволяє швидко адаптуватися до змінюваних умов ринку, що є важливим для прийняття бізнес-рішень.

Дані технології забезпечують значно більшу точність прогнозів у порівнянні з традиційними методами. Тому впровадження розробленої системи на основі машинного навчання стає важливим етапом у оптимізації процесів продажу. Актуальність проекту також підтверджується потребою в аналізі великих об'ємів даних для виявлення прихованих трендів і закономірностей.

Зростання об'ємів інформації, доступної для аналізу, вимагає нових підходів до її обробки. Даний проект може служити основою для подальших досліджень у сфері автоматизації бізнесу. Використання методів штучного інтелекту не лише спрощує процес прийняття рішень, але й дозволяє значно знизити витрати на аналіз ринку. Реалізація даного проекту має не лише теоретичне, але і практичне значення для розвитку бізнесу в умовах сучасного ринку.

Предмет дослідження – розробка інтелектуальної системи прогнозування продажів у супермаркетах на основі алгоритмів машинного навчання.

Об'єкт дослідження – процеси та засоби прогнозування об'ємів продажів у торгівлі.

Мета роботи – розробка інтелектуальної системи прогнозування динаміки продажів у супермаркетах на основі алгоритмів машинного навчання.

Завдання.

1. Провести аналіз існуючих методів та моделей прогнозування динаміки продажів у роздрібній торгівлі з використанням машинного навчання.

2. Вивчити особливості бізнес-процесів у сфері роздрібної торгівлі та визначити ключові потреби цільової аудиторії щодо системи прогнозування.

3. Розробити математичну модель інтелектуальної системи прогнозування продажів.

4. Створити архітектуру програмного забезпечення системи та реалізувати прототип із використанням відповідних алгоритмів машинного навчання.

5. Провести тестування розробленої системи з метою оцінки її точності та зручності у використанні.

Наукова новизна одержаних результатів

В роботі запропоновано новий підхід до створення інтелектуальної системи прогнозування продажів у роздрібній торгівлі на базі алгоритмів машинного навчання, адаптованих до українського ринку. Розроблено математичну модель, яка враховує багатофакторний вплив сезонних, погодних та маркетингових факторів на об’єми продажів. Запропоновано архітектуру системи, що забезпечує просту інтеграцію з існуючими інформаційними платформами малого та середнього бізнесу. Виявлено та враховано особливості поведінки цільової аудиторії українських супермаркетів, що дозволило підвищити точність прогнозів. Розроблено алгоритм автоматичного налаштування моделі на різних форматах даних, що підвищує її гнучкість та адаптивність. У результаті створено прототип системи, який здатний швидко генерувати точні прогнози та рекомендації, що сприяє підвищенню ефективності управління запасами.

Практичне значення одержаних результатів

Практичні результати роботи дозволяють малому та середньому бізнесу впроваджувати ефективні інструменти для прогнозування продажів без необхідності великих інвестицій у IT-інфраструктуру. Впровадження розробленої системи сприяє зменшенню запасів та витрат на зберігання товарів, що підвищує фінансову стабільність підприємств. Розроблений інструмент допомагає швидко реагувати на зміни попиту та сезонні коливання, що покращує конкурентоспроможність роздрібних магазинів.

РОЗДІЛ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1 Актуальність прогнозування продажів у роздрібній торгівлі

У сучасних умовах динамічного розвитку економіки та високої конкуренції в секторі роздрібної торгівлі однією з ключових задач, що стоїть перед супермаркетами, є ефективне управління товарообігом. Високоточне прогнозування об’ємів продажів дозволяє мінімізувати ризики, які пов’язані з дефіцитом чи надлишком товарів, знижує витрати на зберігання продукції, оптимізує логістичні процеси та покращує загальний рівень обслуговування клієнтів.

Особливої актуальності задача прогнозування набуває в умовах нестабільного попиту, який залежить від багатьох факторів: сезонності, цінових змін, маркетингових акцій, поведінки споживачів, рівня інфляції. Роздрібні мережі часто працюють з широким асортиментом товарів, для кожного з яких характерна власна динаміка продажів. Це створює додаткові труднощі в ручному плануванні та вимагає впровадження автоматизованих систем підтримки прийняття рішень.

Традиційні методи прогнозування, які базуються на статистичних підходах, хоча й мають переваги у простоті реалізації, часто не справляються із високою варіативністю даних та не враховують складні нелінійні залежності між факторами, що впливають на попит. Алгоритми машинного навчання дозволяють будувати адаптивні моделі, які здатні аналізувати великі об’єми історичних даних, виявляти приховані закономірності та забезпечувати більш точні прогнози у випадках складних часових рядів.

У розвинених країнах застосування штучного інтелекту та машинного навчання для прогнозування продажів вже стало частиною цифрової трансформації рітейлу. Компанії Walmart, Amazon та Carrefour активно використовують прогнозні моделі для управління попитом, ціноутворення, персоналізованих рекомендацій та контролю за залишками продукції. Дослідження показують, що впровадження таких систем дозволяє знизити витрати на закупівлю товарів на 10-20 % та збільшити точність прогнозів на 30-50 % у порівнянні з традиційними підходами [1].

Також важливим аспектом є здатність сучасних ML-алгоритмів працювати в умовах неповних або зашумлених даних, що характерно для реального комерційного середовища. При виникненні зовнішніх впливів таких як наприклад моделі XGBoost або LSTM здатні швидко адаптуватися до нових умов та забезпечувати стабільний рівень прогнозної точності [2].

З врахуванням активного розвитку електронної комерції, мобільних платформ та автоматизованих систем управління запасами попит на ефективні інструменти прогнозування лише зростає. Прогнозування продажів стає не просто елементом планування, а стратегічним інструментом, який безпосередньо впливає на прибутковість підприємств та задоволення потреб споживачів. Саме тому дослідження та розробка інтелектуальних моделей прогнозування динаміки продажів у супермаркетах має високу наукову та практичну цінність.

Рисунок 1.1 – Схема процесу прогнозування продажів

У світовій практиці прогнозування попиту та продажів є однією з основних складових систем управління ланками постачання SCM (Supply Chain Management). Такі системи як SAP Forecasting and Replenishment, Oracle Demand Management Cloud, Blue Yonder використовують методи машинного навчання для формування прогнозів на основі історичних даних, факторів сезонності, трендів, регіональних особливостей, погодних умов, цінової чутливості та інших параметрів [3].

Застосування моделей прогнозування дозволяє рітейлерам автоматизувати процес прийняття рішень щодо поповнення товарів, планування знижок, організації акцій та промокампаній. У дослідженні [4] при використанні рекурентних

нейронних мереж для прогнозування продажів в продуктовому рітейлі вдалося досягти високої точності, що дозволило зменшити об’єм списаного товару на 18 % у рамках експериментального супермаркету.

Рисунок 1.2 – Схема прийняття рішень на основі прогнозу

Окрему роль відіграє прогнозування в умовах обмеженого терміну реалізації товарів. У продуктових супермаркетах значна частина продукції має короткий термін придатності (молочна продукція, хліб, овочі). У цьому контексті точне прогнозування попиту дозволяє зменшити втрати від списання прострочених товарів та одночасно задовольнити запит споживачів. Алгоритми глибокого навчання такі як LSTM (Long Short-Term Memory) або GRU (Gated Recurrent Units) дають змогу враховувати довготривалі залежності в часових рядах та виявляти приховані закономірності, які не доступні для класичних підходів [5].

Машинне навчання дозволяє прогнозувати не лише загальний об’єм продажів, але й здійснювати індивідуальне прогнозування по кожній одиниці товару по магазинах або по регіонах, враховуючи локальні особливості попиту. Це актуально для великих мереж супермаркетів, де кожен торговий об’єкт має свою специфіку – географічне розташування, купівельну спроможність населення, конкуренцію.

Не менш важливою перевагою ML-підходів є можливість оперативного оновлення моделей та врахування нових даних у режимі реального часу. Це забезпечує гнучкість системи та дозволяє адаптуватися до змін у поведінці споживачів або на ринку. Під час пандемії більшість рітейлерів зіткнулися з раптовими коливаннями попиту. Традиційні моделі виявилися неефективними, натомість моделі машинного навчання, які використовують онлайн-навчання або автоматичне перенавчання забезпечили більш реальні прогнози [6].

Застосування методів машинного навчання у прогнозуванні продажів дозволяє досягати таких стратегічних цілей як:

- покращення точності прогнозів;
- підвищення прибутковості;
- зниження втрат від надлишкових запасів;
- скорочення кількості розпродажів та знижок;
- підвищення задоволеності клієнтів;
- адаптація до зовнішніх змін.

Актуальність досліджень у сфері прогнозування продажів із використанням алгоритмів машинного навчання визначається не лише потребою в оптимізації операційної діяльності супермаркетів, але й викликами цифровізації торгівлі та розвитку інтелектуальних систем підтримки прийняття рішень.

У сучасних умовах цифрової трансформації роздрібна торгівля зазнає значних змін. Клієнтські переваги постійно змінюються, конкуренція між рітейлерами змушує бізнес оперативно адаптуватися до нових реалій. У цьому контексті прогнозування продажів на основі машинного навчання набуває все більшого значення. На відміну від традиційних статистичних методів алгоритми ML дозволяють виявити приховані залежності та враховувати велику кількість вхідних факторів від сезонності до маркетингових активностей і зовнішніх економічних умов.

Згідно з дослідженням [7] застосування нейронних мереж у прогнозуванні продажів дозволяє зменшити середню похибку передбачення на 20-30 % порівняно з ARIMA чи екстраполяційними методами. Це допомагає зменшити залишки товарів на складах, покращити обслуговування клієнтів та оптимізувати постачання. У рітейлі з високою товарною обіговістю невелике підвищення точності прогнозу може призвести до суттєвого зростання прибутковості.

Окремої уваги заслуговує використання алгоритмів глибокого навчання, таких як LSTM-мережі, які здатні ефективно моделювати тимчасові залежності у даних продажів. Згідно з [7] такі підходи показують високу ефективність у задачах з довгостроковими трендами та сезонними коливаннями. В умовах, коли незначне

відхилення у попиті може вплинути на логістику, складування та обслуговування клієнтів, точність прогнозу стає дуже важливою.

Сучасні ML-системи можуть адаптуватися до змін у поведінці споживачів у реальному часі, що дозволяє оперативно реагувати на зовнішні фактори, такі як погодні умови, святкові періоди чи надзвичайні події. Згідно з роботою [8] впровадження таких систем у великих торгових мережах дозволяє значно скоротити втрати від надлишкових закупівель і підвищити рівень задоволеності клієнтів.

Актуальність прогнозування продажів у роздрібній торгівлі з використанням алгоритмів машинного навчання обумовлена не лише потребою в підвищенні точності передбачень, а й необхідністю оперативного прийняття рішень у висококонкурентному середовищі. Це забезпечує гнучкість бізнесу, мінімізацію витрат і максимізацію доходів.

1.2. Огляд досліджень та алгоритмів

Прогнозування динаміки продажів у роздрібній торгівлі є однією з ключових задач сучасної аналітики даних, що активно досліджується в науковій літературі. Розвиток методів машинного навчання дозволив суттєво підвищити точність прогнозів, особливо при роботі з великими масивами історичних даних, що характеризуються сезонністю та трендами.

У багатьох наукових роботах розглядаються різні алгоритми прогнозування, включаючи класичні статистичні моделі, такі як ARIMA**,** Exponential Smoothing та сучасні машинні алгоритми Random Forest**,** XGBoost**,** Support Vector Regression (SVR), а також глибокі нейронні мережі DNN (Deep Neural Networks).

Класичні підходи. Одним із перших підходів до прогнозування продажів були методи часових рядів. ARIMA-модель (Autoregressive Integrated Moving Average) дозволяє моделювати часові залежності та сезонні коливання, але вона має обмеження при наявності складної нелінійної динаміки. У роботі [9] здійснено порівняння традиційних статистичних методів із сучасними ML-алгоритмами, де показано, що останні значно перевершують класичні підходи за точністю прогнозування.

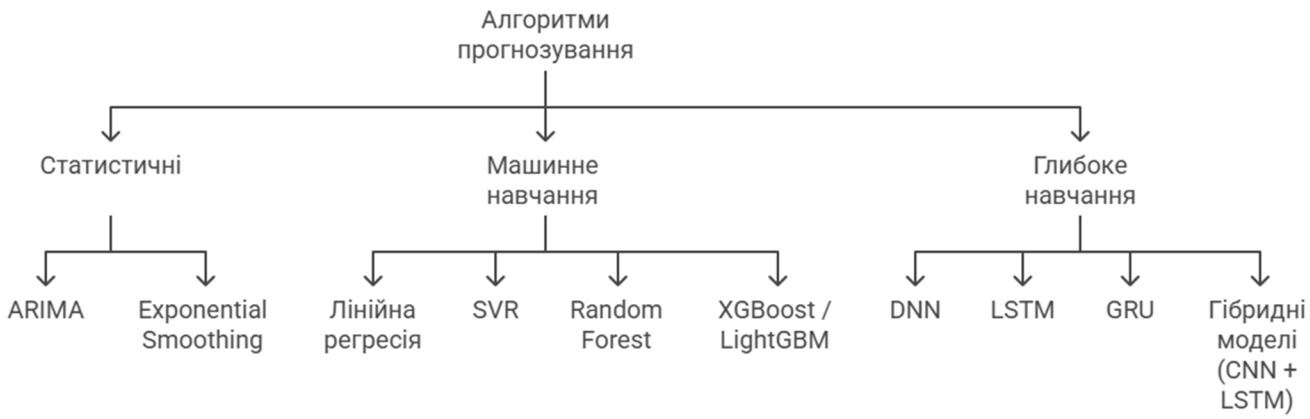

Рисунок 1.3 – Алгоритми прогнозування

Алгоритми машинного навчання. У дослідженні [10] було застосовано нейронні мережі до прогнозування попиту, автори зазначили їх високу здатність виявляти складні нелінійні взаємозв'язки в даних. Серед ML-алгоритмів Random Forest часто використовується для задач регресії через свою стабільність та здатність до боротьби з перенавчанням. Було успішно використано Random Forest для прогнозування обємів продажів продуктів харчування.

Отримав широке застосування XGBoost градієнтний бустинг на деревоподібних моделях. Цей метод демонструє високу продуктивність у змаганнях із прогнозування, зокрема у конкурсі Kaggle "Walmart Recruiting - Store Sales Forecasting" [8].

У сучасних дослідженнях зростає інтерес до моделей глибокого навчання, особливо LSTM (Long Short-Term Memory) та GRU (Gated Recurrent Unit) рекурентних нейронних мереж, які добре справляються з обробкою часових послідовностей. LSTM-мережі навчаються на багатовимірних часових рядах для прогнозування на основі залежностей на довгих відрізках часу.

Також використовуються гібридні підходи, де поєднуються традиційні методи та нейромережеві моделі для досягнення вищої точності. У роботі [11] запропоновано гібридну систему на базі LSTM та інтерпретованих моделей для потреб рітейлу. За результатами порівняльного аналізу в моделі глибокого навчання, особливо LSTM, демонструють точність прогнозів на рівні 90-95 % при наявності достатньо великих об'ємів даних. Разом з тим більш легкі ML-моделі Random Forest

чи XGBoost мають переваги за швидкістю навчання та обчислювальними витратами, що робить їх придатними для реального використання в супермаркетах.

Упродовж останніх десятиліть прогнозування продажів у сфері роздрібної торгівлі стало одним із ключових напрямів прикладного машинного навчання. Зростаючий об'єм даних, що генерується супермаркетами, включаючи дані про товари, поведінку споживачів, сезони та маркетингові кампанії, створює підгрунтя для точного моделювання й прогнозування.

У класичних підходах до прогнозування використовувалися статистичні методи, такі як авторегресійні моделі ARIMA. Вони дозволяли моделювати залежність поточних значень продажів від попередніх, однак мали обмежену ефективність у випадках високої нелінійності даних чи великої кількості ознак.

Згодом були запропоновані методи машинного навчання, які значно розширили можливості прогнозування. Серед них варто виділити лінійну регресію, Random Forest, градієнтний бустинг XGBoost, LightGBM, глибокі нейронні мережі (DNN, LSTM) [12].

У дослідженні [13] було використано рекурентні нейронні мережі типу LSTM для передбачення продажів у реальному супермаркеті. Їхня модель показала перевагу над традиційними ARIMA та SVR (Support Vector Regression) завдяки здатності враховувати довгострокові залежності. Моделі LSTM (Long Short-Term Memory) успішно застосовуються для роботи з часовими рядами, оскільки вони зберігають інформацію про попередні стани, що є досить важливим у задачах прогнозування динаміки. У роботі [14] пояснюється практична реалізація таких моделей з використанням бібліотек TensorFlow та Keras.

Інші методи, такі як XGBoost, також знайшли широке застосування завдяки своїй швидкості, здатності обробляти пропущені значення та забезпечувати високу точність. У роботі [15] автори порівнюють точність класичних та сучасних алгоритмів прогнозування на великому наборі даних і роблять висновок, що ансамблеві методи часто перевершують статистичні моделі.

Цікавим є підхід до прогнозування з використанням моделей глибокого навчання, які комбінують CNN та LSTM так звані Hybrid Models. Було

запропоновано архітектуру, де CNN виявляє локальні закономірності, а LSTM оперує довгостроковою пам’яттю. Такий підхід забезпечив покращення точності прогнозу у порівнянні з моделями, які використовували лише один тип мережі.

Важливою складовою успішного прогнозування є також інженерія ознак – створення нових змінних, що краще описують закономірності в даних (періодичність, свята, погода, маркетингові кампанії). У роботі [16] вказується, що якісно сформовані ознаки можуть збільшити точність моделей на 20-30 %. Сучасні дослідження у сфері прогнозування продажів супермаркетів демонструють чітку тенденцію до використання гібридних і глибоких моделей, здатних враховувати велику кількість факторів та складні часові залежності.

1.3. Програмні засоби для прогнозування

У процесі прогнозування динаміки продажів у роздрібній торгівлі важливу роль відіграє вибір програмних засобів, які забезпечують побудову моделей машинного навчання, обробку даних, візуалізацію результатів. На сьогоднішній день існує широкий спектр інструментів, кожен із яких має свої переваги та сфери застосування.

Python є найбільш поширеною мовою в задачах прогнозування продажів завдяки великій кількості бібліотек для обробки даних і побудови моделей машинного навчання. Python дозволяє швидко розробляти, тестувати та впроваджувати моделі. Основні бібліотеки, що використовуються:

- Pandas для обробки табличних даних;
- NumPy для математичних обчислень;
- Scikit-Learn для реалізації алгоритмів машинного навчання;
- TensorFlow, Keras для створення глибоких нейронних мереж.

Розглянемо інструменти для обробки даних та візуалізації: Jupyter Notebook – інтерактивне середовище для написання та виконання Python коду, яке зручно використовувати для експериментів з моделями та візуалізацій. Платформи машинного навчання та автоматизації: Google Colab – хмарне середовище для

запуску Python коду, застосовується для навчання моделей та обробки великих об’ємів даних.

Бібліотеки для часових рядів та прогнозування:

- Facebook Prophet – бібліотека статистики, вона підходить для даних з сезонністю та трендами;
- ARIMA/ SARIMA – класичні статистичні моделі, які реалізовані в Python. Вони добре працюють з часовими рядами;
- LightGBM, XGBoost – бібліотеки для градієнтного бустингу, які використовуються у задачах з великою кількістю ознак і складною структурою даних.

Інструмент для розгортання моделей Streamlit це фреймворк для створення веб-інтерфейсів прогнозних моделей, дає змогу створити демонстраційні додатки. Сучасні програмні засоби для прогнозування продажів охоплюють увесь життєвий цикл моделі від підготовки даних до візуалізації результатів і розгортання в реальному середовищі. Розуміння можливостей кожного інструменту дозволяє створювати ефективні системи прогнозування.

ВИСНОВКИ ДО РОЗДІЛУ

У розділі 1 проаналізовано сучасний стан прогнозування продажів у роздрібній торгівлі в супермаркетах. Виявлено, що існуючі рішення часто мають обмеження у врахуванні багатофакторних впливів на динаміку продажів. Визначено основні тенденції розвитку ринку систем автоматизованого прогнозування та аналізу даних. Проаналізовано сучасні технології та алгоритми машинного навчання, які застосовуються у цій галузі. Обгрунтовано необхідність створення іноваційних систем, орієнтованих на малий і середній бізнес з врахуванням локальних особливостей. Виявлено проблеми недостатньої адаптивності існуючих систем до швидкозмінних умов ринку.

В розділі наведено огляд існуючих платформ та рішень від великих компаній і стартапів. Виявлено, що багато сучасних систем орієнтовані на великі корпорації з розвиненою інфраструктурою. Виявлено потребу в розробці доступних та інтуїтивних інструментів прогнозування для малого бізнесу. Проаналізовано вимоги цільової аудиторії щодо простоти налаштування та інтеграції нових систем. Визначено фактори, які слід враховувати при створенні системи, зокрема сезонність, акції та поведінку споживачів. Виявлено ключову роль алгоритмів машинного навчання у підвищенні точності прогнозів. Обгрунтовано, що сучасні технології дозволяють створювати гнучкі та адаптивні моделі прогнозування. Визначено пріоритетні напрямки подальшого розвитку у цій області, інтеграцію з існуючими системами управління.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Джерела даних

Інформаційне забезпечення є одним з ключових компонентів побудови інтелектуальної системи прогнозування продажів, якість, структура та повнота вхідних даних визначають ефективність математичних моделей і точність прогнозів. Основним джерелом інформації для прогнозування виступає історичний набір даних про продажі супермаркету. Ці дані можуть надходити з внутрішньої інформаційної системи підприємства з модулів, які ведуть облік продажів, залишків товарів, постачань, а також з модулів лояльності клієнтів.

Джерела даних відіграють фундаментальну роль у створенні будь-якої системи машинного навчання, оскільки від їх якості, об’єму та різноманіття залежить точність і ефективність прогнозування. Для задачі прогнозування динаміки продажів супермаркету використовується сукупність даних, що включає як внутрішні, так і зовнішні джерела інформації.

Внутрішні джерела даних – це історичні дані про продажі, які накопичуються безпосередньо в інформаційних системах супермаркету. Вони містять записи про транзакції в касах, дані про товари, їх категорії, ціни, знижки, інформацію про об’єми реалізації за різні часові проміжки. Такі дані зберігаються у вигляді таблиць або баз даних. Вони мають високу точність і деталізацію, що дозволяє будувати моделі прогнозування на основі реальних продажів.

Окрім базових даних про продажі, до внутрішніх джерел відносяться дані про маркетингові активності: проведені акції, рекламні кампанії, спеціальні пропозиції. Ця інформація допомагає моделі враховувати вплив зовнішніх стимулів на поведінку споживачів і зміни попиту.

Зовнішні джерела даних включають інформацію, яка прямо або опосередковано впливає на об’єми продажів, але не зберігається в системах супермаркету. До таких джерел належать наступні. Календарні дані – свята, вихідні, сезонні періоди, які суттєво впливають на купівельну активність. Врахування цих факторів дозволяє врахувати сезонність у прогнозах. Погодні умови – температура, опади, вологість,

які можуть безпосередньо впливати на попит певних категорій товарів, наприклад, збільшення продажів напоїв у спекотну погоду. Економічні індикатори – рівень безробіття, інфляція, середній дохід населення, які формують загальний контекст купівельної спроможності споживачів. Демографічні та соціальні дані – структура населення у регіоні, вікові групи, особливості споживчих переваг, що дозволяє краще сегментувати ринок і враховувати поведінкові фактори. Дані конкурентного середовища – інформація про акції, ціни та асортимент конкурентів, що може впливати на попит.

Дані із зовнішніх джерел можуть отримуватися з відкритих баз даних державних служб, через API спеціалізованих сервісів, шляхом аналізу відкритих веб-ресурсів. Для ефективного прогнозування важливо забезпечити комплексність джерел даних та їх інтеграцію в єдину інформаційну систему, що дозволяє поєднати внутрішні продажні дані з зовнішніми факторами, які впливають на попит. Такий підхід підвищує адаптивність моделей до змін у зовнішньому середовищі та забезпечує більш точні прогнози.

Особливу увагу варто приділяти контролю якості даних на всіх етапах їх збору, зберігання та обробки. Некоректні або неповні дані можуть суттєво погіршити результати машинного навчання. Тому застосовуються методи валідації, очищення та нормалізації інформації. Формування джерел даних створює надійну базу для побудови прогнозних моделей прогнозування динаміки продажів у супермаркетах.

2.2. Структура і формат даних

Структура та формат даних є характеристиками інформації, що використовується для побудови моделей прогнозування. Вони визначають спосіб організації, зберігання та подальшої обробки даних, що впливає на ефективність алгоритмів машинного навчання та якість отриманих прогнозів. Для задачі прогнозування динаміки продажів супермаркету основним типом інформації є табличні дані, які формують історичний масив транзакцій. Кожен запис у такій таблиці містить інформацію про окремий продаж або показник за певний період часу: день, тиждень, місяць.

Основні компоненти структури даних включають (рис. 2.1):

Рисунок 2.1 – Основні компоненти структури даних

Ідентифікатори:

- store_id – унікальний код або номер торгової точки, що дозволяє розрізняти різні магазини в мережі;
- item_id – унікальний ідентифікатор товарної позиції;
- category_id – код категорії товару, що дозволяє класифікувати продукцію за групами, наприклад, напої, овочі, побутова хімія.

Хронологічні показники:

- date – дата проведення операції у форматі року, місяця, дня;
- time – час продажу.

Оперативні показники:

- sales_quantity – кількість одиниць товару, проданих у вказаний період;
- price – ціна одиниці товару на момент продажу;
- discount – інформація про надану знижку або спеціальну пропозицію;
- revenue – загальна виручка за продані одиниці (product of quantity and price).

Додаткові змінні:

- promotion_flag – позначка, чи було застосовано акцію до товару;
- holiday_flag – індикатор, чи припадає дата на святковий або вихідний день;

- weather_conditions – погодні параметри (температура, опади), які можуть впливати на купівельну активність.

Для зручності обробки та сумісності з аналітичними інструментами дані зберігаються у табличних форматах. CSV (Comma-Separated Values) – текстовий формат, у якому кожен рядок відповідає запису, а поля розділяються комами. Це найбільш поширений формат завдяки простоті обробки та сумісності з багатьма системами. Excel – формат електронних таблиць, що зручно використовувати для первинного аналізу та збереження невеликих об'ємів даних. У контексті машинного навчання та обробки даних у Python основним форматом для роботи є CSV або імпорт даних у вигляді таблиць Pandas DataFrame.

Оскільки прогнозування продажів є задачою часових рядів, важливо забезпечити коректність часових позначок і послідовність даних. Необхідно уникати пропусків у часових інтервалах або заповнювати їх адекватними методами інтерполяції чи екстраполяції. На етапі побудови моделей часто створюються додаткові ознаки, які не входять безпосередньо у вхідні дані, але покращують якість прогнозів. Це можуть бути значення продажів за попередні періоди, продажі за минулий день, тиждень, місяць; середні значення продажів за певний період, що допомагають згладити сезонні коливання; категоріальні ознаки – дні тижня, святкові дні, сезонність. Важливо звернути увагу на цілісність і коректність даних: відсутність дублікатів, логічних помилок, невірних значень. Це забезпечує надійність подальшого аналізу та прогнозування. Чітко структуровані, коректно оформлені та збережені у стандартизованому форматі дані створюють основу для ефективного прогнозування динаміки продажів супермаркету.

2.3. Засоби візуалізації даних

Візуалізація даних є невід'ємною складовою процесу аналізу та інтерпретації інформації у задачах прогнозування динаміки продажів. Вона дозволяє не лише краще зрозуміти структуру та закономірності вхідних даних, але й ефективно донести результати досліджень і прогнозів до кінцевих користувачів.

Візуальні представлення даних допомагають виявити тенденції, сезонні коливання, аномалії, перевірити якість і повноту інформації. Завдяки графікам та діаграмам можна швидко оцінити динаміку продажів за різні періоди, порівняти результати між різними магазинами або товарними категоріями, проаналізувати вплив зовнішніх факторів свят, акцій чи погодних умов. Візуалізація є важливим інструментом під час презентації результатів прогнозування керівництву або клієнтам, вона дозволяє наглядно демонструвати ефективність моделей, їхні переваги та обмеження.

Рисунок 2.2 – Основні типи візуалізації даних

Лінійні графіки підходять для відображення часових рядів, щоденних або щомісячних об’ємів продажів. Вони показують тренди, цикли і сезонність. Стовпчикові діаграми використовуються для порівняння продажів між різними товарами, категоріями або магазинами за певний період. Гістограми дозволяють проаналізувати розподіл значень, кількості продажів або цін. Теплові карти корисні для візуалізації кореляцій між різними змінними або для відображення інтенсивності продажів у різних регіонах чи магазинах. Кругові діаграми застосовуються для показу часток ринку або часток продажів у загальному об’ємі за

категоріями. Коробкові діаграми дають змогу оцінити розподіл, медіану, квартилі та виявити викиди у даних.

Для побудови графіків і діаграм у середовищі програмування Python використовуються такі бібліотеки:

- Matplotlib – базова бібліотека для створення статичних, анімаційних та інтерактивних візуалізацій, вона дає змогу налаштовувати всі параметри графіків.
- Seaborn – побудована на основі Matplotlib, ця бібліотека спрощує створення статистичних графіків із більш привабливим дизайном і стандартними темами.
- Plotly – інструмент для створення інтерактивних графіків, які можна масштабувати, переміщувати та детально досліджувати в броузері або у вікні програми.
- Bokeh – бібліотека для інтерактивної візуалізації, зручно інтегрується з веб-додатками.
- Streamlit – платформа для швидкого створення веб-інтерфейсів із інтерактивними дашбордами та візуалізаціями, що дозволяє користувачам без спеціальних технічних знань аналізувати прогнози.

Засоби візуалізації даних є важливим інструментом, який забезпечує розуміння динаміки продажів, робить результати прогнозування доступними і зрозумілими для широкого кола користувачів.

2.4. Засоби програмної реалізації Python та бібліотеки машинного навчання

У проектах, пов’язаних із аналізом даних і прогнозуванням, особливе місце посідає мова програмування Python. Вона поєднує простоту синтаксису, потужні інструменти для обробки даних та багатий набір бібліотек для машинного навчання і штучного інтелекту. У даній роботі Python є основою для розробки моделі прогнозування динаміки продажів супермаркету. Python відомий своєю легкістю вивчення та використання, що робить його зручним як для досвідчених розробників,

так і для початківців. Він має відкритий вихідний код і широку спільноту, яка підтримує і розвиває сотні бібліотек і фреймворків. Крім того, Python легко інтегрується з іншими мовами і платформами, що забезпечує гнучкість розробки. Для реалізації завдань прогнозування у роботі будуть використані наступні бібліотеки:

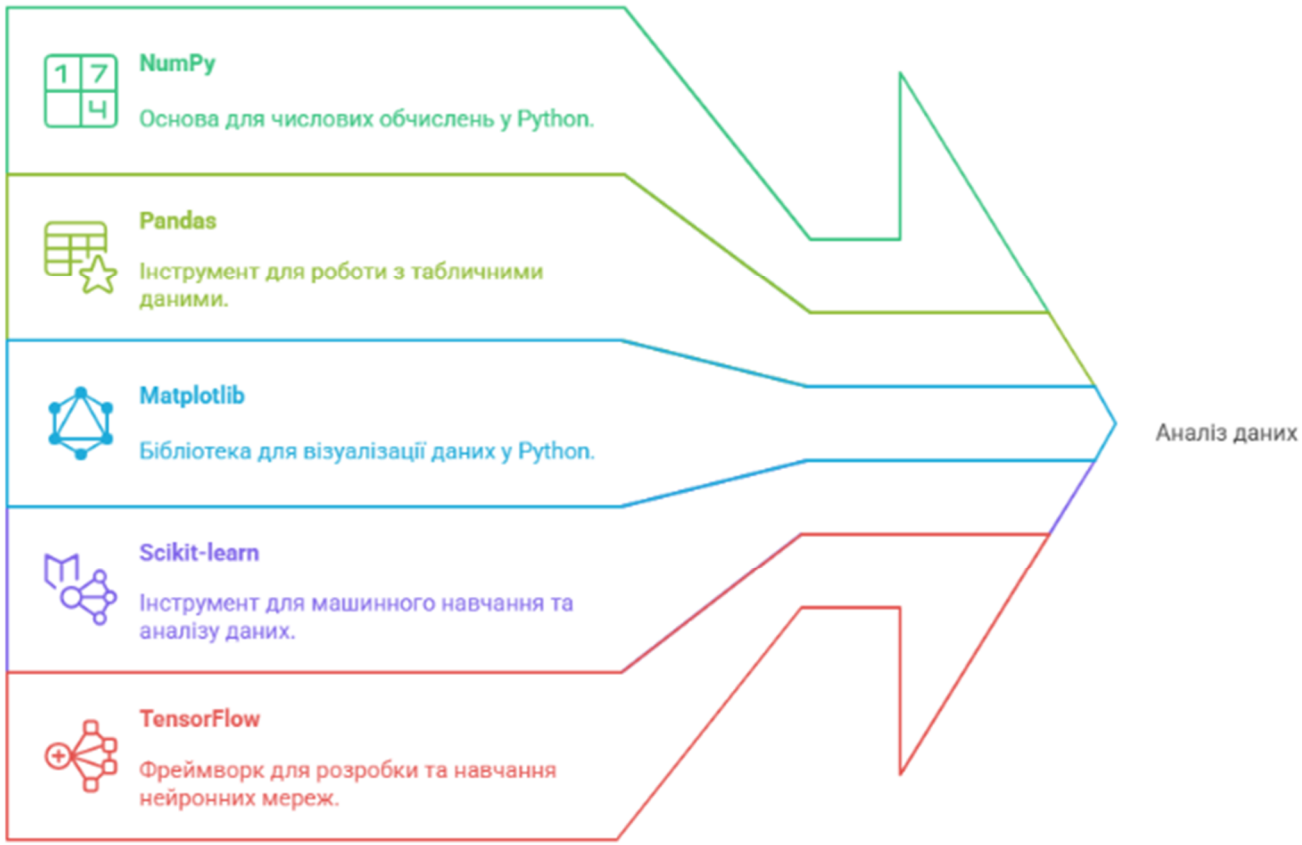

Рисунок 2.3 – Інструменти для аналізу даних Python

NumPy – фундаментальна бібліотека для роботи з багатовимірними масивами та математичними операціями над ними. Вона забезпечує високу продуктивність при обробці числових даних, що є базою для подальшої аналітики.

Pandas – потужний інструмент для маніпулювання табличними даними. Забезпечує зручні структури даних (DataFrame), які дозволяють легко імпортувати, обробляти, фільтрувати та аналізувати великі обсяги інформації.

Matplotlib та Seaborn – бібліотеки для візуалізації даних, які використовуються для побудови графіків, діаграм і теплових карт, що допомагають вивчити закономірності у даних і оцінити результати моделей.

Scikit-learn – одна з найпопулярніших бібліотек машинного навчання, що надає широкий спектр алгоритмів для класифікації, регресії, кластеризації та підготовки даних. У ній реалізовані методи лінійної регресії, дерев рішень, метод опорних векторів, ансамблеві методи (Random Forest, Gradient Boosting) та інші. Scikit-learn також забезпечує інструменти для оцінювання моделей, вибору параметрів і крос-валідації.

XGBoost – високопродуктивна бібліотека для реалізації градієнтного бустингу над деревами рішень. Вона відома своєю ефективністю, швидкістю роботи та високою точністю, що робить її одним з найкращих інструментів для побудови прогнозних моделей на структурованих даних.

LightGBM – ще одна реалізація градієнтного бустингу з акцентом на швидкість навчання і низьке споживання пам’яті, особливо ефективна при роботі з великими наборами даних.

Statsmodels – бібліотека для статистичного аналізу та моделювання, яка включає реалізації класичних методів, таких як ARIMA, SARIMA для аналізу часових рядів, що актуально для прогнозування динаміки продажів.

Prophet – бібліотека для прогнозування часових рядів із урахуванням сезонності та святкових днів. Вона проста у використанні і добре підходить для бізнес-аналітики.

TensorFlow – популярні фреймворки для побудови глибоких нейронних мереж, які дозволяють реалізувати складні моделі, такі як рекурентні нейронні мережі RNN, LSTM та GRU. Ці моделі є ефективними для прогнозування часових рядів із нелінійною динамікою.

У дипломній роботі використання бібліотек буде розподілено за етапами. На етапі підготовки даних застосовуватимуться Pandas і NumPy для очищення, трансформації та агрегування інформації. Для побудови базових моделей будуть використані Scikit-learn, XGBoost, LightGBM, які дозволять швидко отримати якісні прогнози на основі табличних даних. Для аналізу часових рядів та врахування сезонності будуть використані Statsmodels та Prophet. Для створення глибоких нейронних мереж застосовуватиметься TensorFlow, що дасть змогу моделювати

складні залежності і тренди у продажах. Для оцінки та візуалізації результатів Matplotlib і Seaborn. Використання Python та наведених бібліотек забезпечить потужний інструментарій для розробки інтелектуальної системи прогнозування продажів, що дасть змогу врахувати різноманітні фактори, підвищити точність моделей та створити інтуїтивно зрозумілий інтерфейс для користувачів.

Scikit-learn – одна з найпопулярніших бібліотек для машинного навчання, яка надає широкий спектр алгоритмів класичного ML. Основні можливості:

- алгоритми класифікації (логістична регресія, метод опорних векторів, k–ближчих сусідів);
- регресійні моделі (лінійна, поліноміальна регресія, дерево рішень);
- ансамблеві методи (Random Forest, Gradient Boosting);
- обробка даних: масштабування, кодування категорій;
- методи оцінювання моделей: крос–валідація, метрики точності.

Використовується для створення базових моделей прогнозування, експериментів з різними алгоритмами та оцінки їхньої якості.

XGBoost (Extreme Gradient Boosting) – потужна реалізація градієнтного бустингу дерев рішень, оптимізована для швидкості і точності. Застосовується для побудови потужних та точних моделей прогнозування, особливо на структурованих табличних даних із багатьма ознаками.

LightGBM – ще одна високопродуктивна бібліотека для градієнтного бустингу. Використовується як альтернатива XGBoost для швидкого навчання та точного прогнозування на великих наборах даних.

Statsmodels – бібліотека для статистичного аналізу, що дозволяє будувати класичні статистичні моделі. Використовується для аналізу часових рядів, побудови моделей із урахуванням сезонності і трендів, а також для статистичного обгрунтування прогнозів.

Prophet – бібліотека для прогнозування часових рядів із акцентом на сезонність та святкові дні. Застосовується для бізнес-прогнозування з урахуванням особливостей календаря, свят і рекламних періодів.

ВИСНОВКИ ДО РОЗДІЛУ

У другому розділі проаналізовано джерела даних, що використовуються для побудови системи прогнозування динаміки продажів супермаркету. Встановлено, що внутрішні джерела, історичні дані про продажі, маркетингові акції та поведінка споживачів є основою для формування якісного набору даних. Окрему увагу приділено зовнішнім даним таким як календарні події, погодні умови та соціально-економічні фактори, які суттєво впливають на споживчий попит. Комплексне поєднання внутрішніх і зовнішніх джерел дозволяє моделі прогнозування адаптуватися до змін у поведінці клієнтів.

Описано структуру даних, що включає унікальні ідентифікатори, хронологічні та операційні показники, а також додаткові ознаки, які важливі для прогнозування. Визначено основні формати збереження даних, серед яких важливу роль відіграє формат CSV завдяки сумісності з інструментами машинного навчання. Розглянуто методи побудови нових ознак, які допомагають виявити сезонність, циклічність і аномалії в динаміці продажів.

Значну увагу приділено візуалізації даних як інструменту дослідження та представлення результатів аналізу й прогнозування. Проаналізовано переваги різних типів візуалізацій, лінійних графіків, стовпчикових діаграм, теплових карт і гістограм. Показано, що засоби візуалізації сприяють виявленню трендів, сезонних коливань, а також забезпечують зручність інтерпретації результатів моделі.

У розділі висвітлено переваги використання мови Python для реалізації прогнозних моделей завдяки її бібліотекам. Описано ключові бібліотеки машинного навчання, такі як Pandas, NumPy, Scikit-learn, XGBoost, LightGBM, які використовуються в даній роботі. Розкрито роль бібліотек візуалізації даних Matplotlib, Seaborn, Plotly, Streamlit у створенні зрозумілих інтерфейсів. Інформаційне забезпечення є важливою складовою для побудови, навчання та оцінювання моделі прогнозування продажів.

РОЗДІЛ 3. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Постановка задачі прогнозування

Задача прогнозування динаміки продажів у роздрібній торгівлі належить до категорії задач регресії, в якій необхідно передбачити числове значення об’єму продажів товарів у майбутні часові періоди. Така задача передбачає аналіз наявних історичних даних про продажі, врахування сезонних коливань, днів тижня, впливу святкових днів, рекламних кампаній, цінових змін та інших факторів, що можуть суттєво впливати на попит.

Процес прогнозування можна описати як побудову функції f, яка на основі вхідного набору змінних X, що включають часові, економічні та поведінкові характеристики, дозволяє оцінити цільову змінну y, яка відповідає прогнозованому об’єму продажу. Модель має вигляд:

$$y = f(X) + \varepsilon \tag{3.1}$$

де ε – випадкова похибка.

Прогнозування може виконуватись як для одного товару, так і для категорій товарів або навіть на рівні всього магазину. Важливою особливістю задачі є залежність продажів від попередніх значень, що обумовлює використання методів часових рядів або гібридних моделей, які враховують як часову динаміку, так і вплив зовнішніх факторів. Залежно від потреб бізнесу прогноз може бути короткостроковим (дні, тижні) або довгостроковим (місяці, квартали).

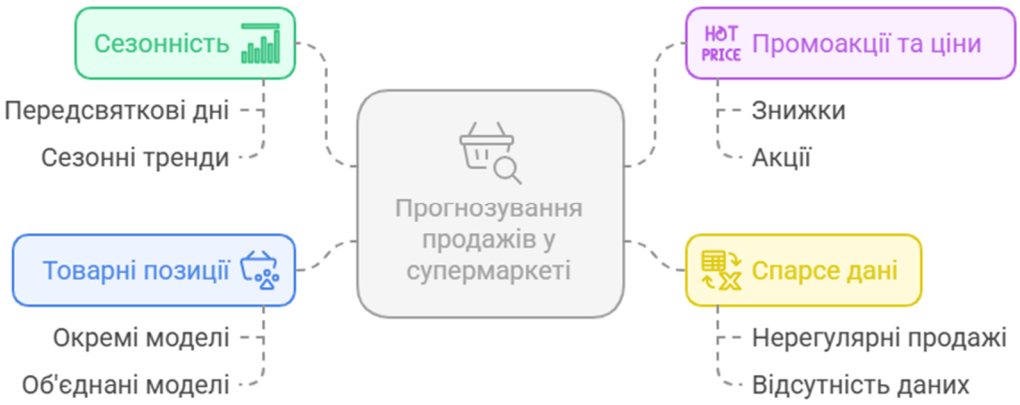

Рисунок 3.1 – Особливості прогнозування продажів у супермаркеті

У постановці задачі слід також визначити мету прогнозування: оптимізацію запасів, планування закупівель, ефективне ціноутворення або управління логістикою. Виходячи з цього обирається не тільки об’єм і тип даних, але й відповідні алгоритми машинного навчання, що забезпечують необхідну точність моделі.

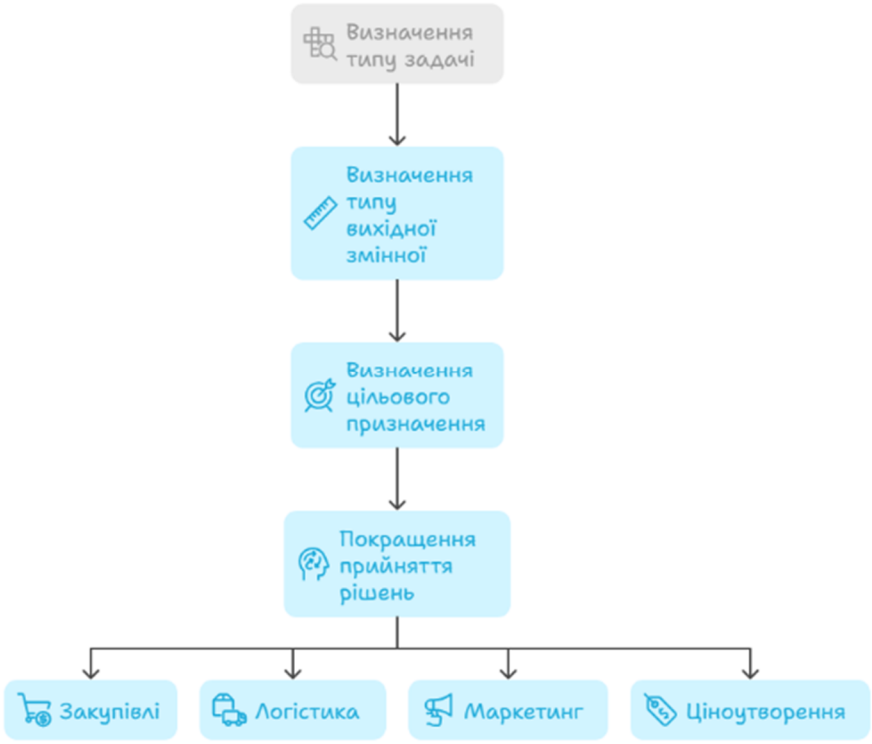

Рисунок 3.2 – Процес прогнозування часових рядів

У контексті роздрібної торгівлі задача прогнозування продажів полягає у створенні математичної моделі, яка здатна з достатньою точністю передбачити об’єм майбутніх продажів товарів або категорій товарів на основі історичних даних. Прогнозування дозволяє оптимізувати запаси, підвищити прибутковість, уникнути дефіциту або надлишку продукції, а також приймати обгрунтовані стратегічні рішення.

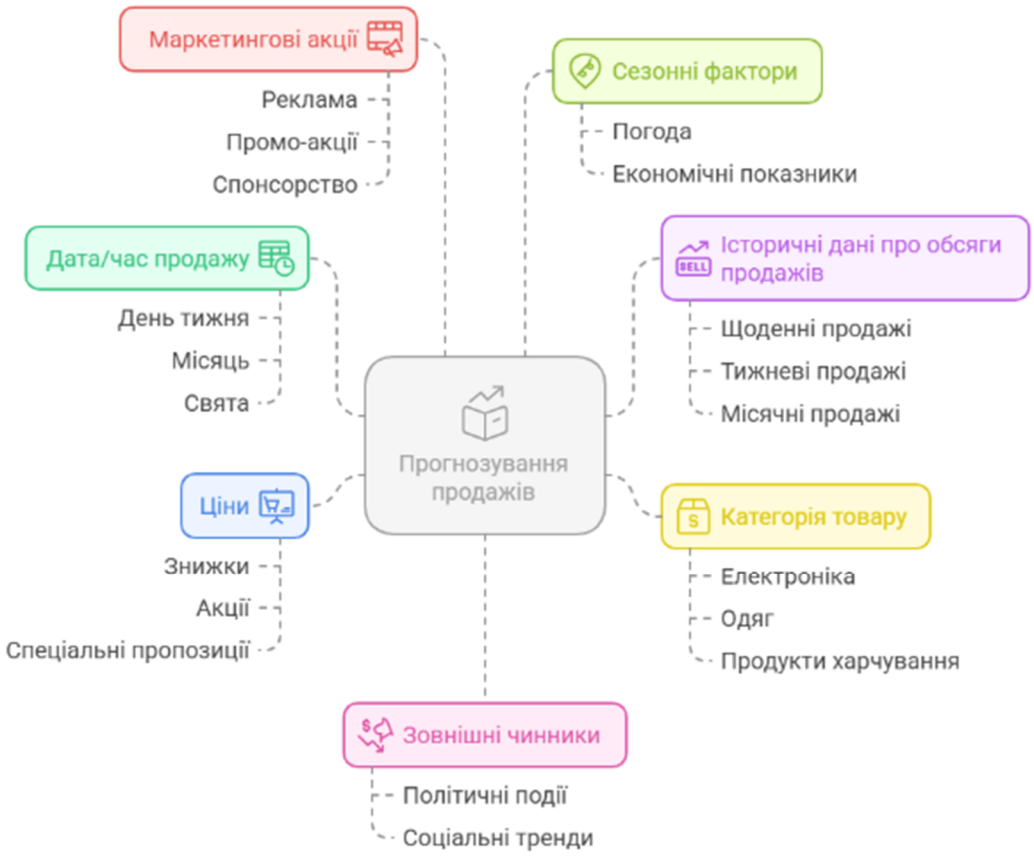

Рисунок 3.3 – Фактори, що впливають на прогнозування продажів

Математично задача прогнозування формулюється як задача регресії, де є послідовність спостережень за попередні періоди (x_t, y_t), де x_t – вектор вхідних ознак у момент часу t, y_t – фактичний об'єм продажів. Потрібно побудувати модель, яка дозволяє на основі поточних та історичних даних передбачити значення y_{t+1} – майбутній об'єм продажів.

$$\hat{y}_{t+1} = f(x_t, x_{t-1}, \ldots, x_{t-n}). \tag{3.2}$$

Рисунок 3.4 – Цикл прогнозування продажів

Задача прогнозування продажів у супермаркеті передбачає визначення майбутніх об’ємів реалізації товарів на основі історичних даних, таких як об’єми продажів, дати, категорії товарів, місце продажу, сезонність. Вона спрямована на створення ефективної системи, що аналізує історичні дані продажів і формує точні прогнози на майбутні періоди. Це дозволяє супермаркету оптимізувати свої ресурси, підвищити якість обслуговування клієнтів.

3.2. Методи математичного моделювання

Для розв’язання задачі прогнозування динаміки продажів застосовуються як класичні статистичні методи, так і сучасні методи машинного навчання.

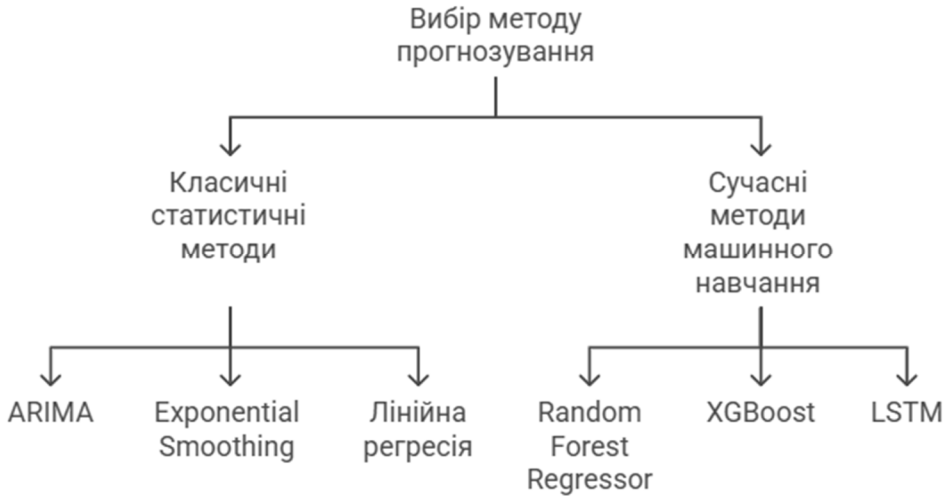

Рисунок 3.5 – Методи прогнозування динаміки продажів

У контексті прогнозування продажів у роздрібній торгівлі математичне моделювання відіграє ключову роль, адже саме воно дозволяє формалізувати взаємозв’язки між різними змінними, виявити приховані закономірності в історичних даних та побудувати модель, яка здатна передбачати майбутні показники з високою точністю. Основна мета математичного моделювання в цій сфері полягає у створенні обчислювальної моделі, яка може відтворювати поведінку продажів у залежності від ряду факторів, таких як сезонність, маркетингові активності, поведінка споживачів, ціни, конкурентне середовище та інші параметри.

Серед найбільш розповсюджених підходів до математичного моделювання в задачах прогнозування варто виділити регресійні моделі. Прості та множинні лінійні регресії дозволяють встановити залежність між змінною, що прогнозується, та

однією або кількома незалежними змінними. У випадках, коли зв'язки між змінними є нелінійними, застосовуються поліноміальні регресії, логістичні регресії та інші методи нелінійного регресійного аналізу.

Ще одним важливим напрямом математичного моделювання є використання моделей часових рядів. Методи ARIMA, Prophet є популярними для обробки серій даних з часовою залежністю. Ці моделі дозволяють враховувати як тренди, так і сезонні коливання, що важливо для роздрібної торгівлі, де показники продажів можуть суттєво змінюватися в залежності від місяця, свята або рекламної кампанії.

Крім класичних статистичних моделей активно використовуються методи машинного навчання. Decision Tree, Random Forest, метод опорних векторів SVM, градієнтний бустинг XGBoost, LightGBM та штучні нейронні мережі є потужними інструментами для побудови прогностичних моделей. Ці методи дозволяють моделювати складні нелінійні взаємозв'язки між великою кількістю параметрів, зменшуючи вплив шуму та підвищуючи точність прогнозів.

У випадку великих об'ємів даних та складних взаємозв'язків актуальності набувають глибокі нейронні мережі. Рекурентні нейронні мережі RNN та їх удосконалення, такі як LSTM та GRU ефективно працюють з послідовними даними, зокрема часовими рядами, що дозволяє точно передбачати динаміку продажів.

Застосування кожного методу залежить від особливостей даних, мети дослідження, необхідного рівня точності та доступних обчислювальних ресурсів. Математичне моделювання дозволяє не лише здійснювати прогнозування, а й проводити аналітичні експерименти, вивчати вплив окремих факторів на кінцеві результати та формувати рекомендації для управлінських рішень.

Окрім традиційних статистичних методів таких як лінійна регресія чи авторегресивні моделі математичне моделювання все активніше використовує алгоритми машинного та глибокого навчання. Ці підходи демонструють високу ефективність при роботі з великими об'ємами історичних даних, що мають складні, нелінійні та багатовимірні залежності. Дерева рішень, методи градієнтного бустингу здатні враховувати широкий спектр факторів, що впливають на об'єми продажів, такі як сезонність, маркетингові акції, погодні умови, регіональні відмінності.

Суттєвий прорив у математичному моделюванні прогнозування продажів забезпечили нейронні мережі, зокрема згорткові нейронні мережі CNN, рекурентні нейронні мережі RNN та їхня модифікація довга короткочасна пам’ять LSTM. LSTM-архітектури є особливо ефективними при роботі з часовими рядами, вони здатні зберігати інформацію про попередні значення протягом довгого періоду часу. Це дозволяє моделі запам’ятовувати патерни у зміні об’ємів продажів, що обумовлені внутрішніми циклічними процесами або зовнішніми економічними подіями.

Важливу роль відіграють гібридні моделі, які поєднують статистичні методи з машинним навчанням, що дозволяє враховувати як лінійні, так і нелінійні залежності. Поєднання методів авторегресії із нейронними мережами забезпечує вищу точність прогнозів, ніж використання кожного з підходів окремо. Останнім часом усе більш популярними стають ансамблеві моделі, в яких прогноз формується на основі об’єднання результатів кількох різних моделей, це дозволяє підвищити стабільність та узагальнювальну здатність системи.

Вибір конкретного математичного методу залежить від особливостей завдання, якості доступних даних, технічних ресурсів, доступних для навчання та розгортання моделі. У реальних проектах доцільним є використання комбінованого підходу, що поєднує гнучкість та передбачуваність сучасних інтелектуальних алгоритмів із теоретичною обґрунтованістю класичних математичних методів.

Лінійна регресія:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \varepsilon \tag{3.3}$$

Це класична модель, де змінна y об’єм продажів залежить від набору незалежних змінних x_i (ціна, день тижня, реклама). β_i – коефіцієнти моделі, які показують вплив кожного фактора, ε – випадкова похибка.

Поліноміальна регресія:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_d x^d + \varepsilon \tag{3.4}$$

Цей метод використовується для моделювання нелінійних залежностей. Ступінь полінома d підбирається залежно від складності задачі.

Авторегресійна модель:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t \tag{3.5}$$

Ця модель використовується для часових рядів, де поточне значення залежить від попередніх значень. ϕ_i – коефіцієнти авторегресії, ε_t – білий шум.

Модель ARIMA поєднує авторегресію, інтегрування – різницю для усунення тренду), ковзне середнє:

$$y'_t = c + \sum_{i=1}^{p} \phi_i y'_{t-i} + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j} + \varepsilon_t \tag{3.6}$$

де $y_t' = \nabla^d y_t$ – різницевий оператор ступеня d.

ARIMA ефективна для несезонних часових рядів з трендом. SARIMA розширює ARIMA, додаючи сезонні компоненти, вона включає сезонні оператори авторегресії, інтегрування та ковзного середнього з періодом s.

Метод Facebook Prophet:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t \tag{3.7}$$

де $g(t)$ – трендова функція, $s(t)$ – сезонна компонента, $h(t)$ – вплив свят і подій, ε_t – шум. Модель зручна для бізнес-прогнозування, легко адаптується до пропущених даних та свят.

Градієнтний бустинг XGBoost, LightGBM:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in \mathcal{F} \tag{3.8}$$

де F – простір слабких моделей (дерев). Кожна нова модель f_k навчається на помилках попередніх моделей.

Рекурентна нейронна мережа:

$$\begin{aligned} h_t &= \sigma(W_h h_{t-1} + W_x x_t + b) \\ y_t &= \phi(W_y h_t + b_y) \end{aligned} \tag{3.9}$$

RNN зберігає пам’ять про попередні стани через приховані стани h_t, що важливо для часових рядів.

Рисунок 3.6 – Цикл математичного моделювання прогнозування продажів

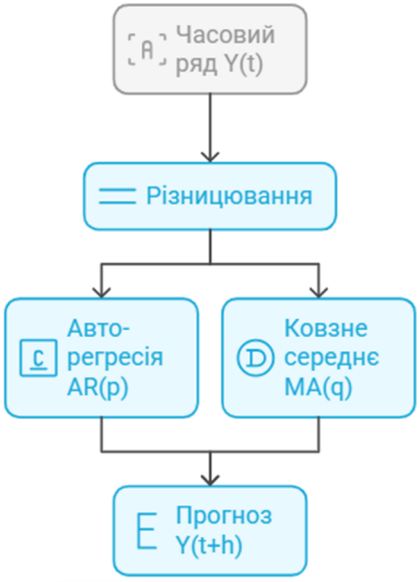

Рисунок 3.7 – Схема роботи ARIMA

3.3. Побудова ознак та підготовка даних

Перед застосуванням будь-яких методів математичного моделювання необхідно здійснити підготовку даних, оскільки якість вхідних даних впливає на точність прогнозів. Процес підготовки даних включає очищення, нормалізацію, трансформацію, обробку пропущених значень, що відображають суттєві закономірності у даних.

У реальних наборах даних часто трапляються пропущені або аномальні значення. Для їх обробки застосовують такі підходи:

- видалення рядків з пропущеними значеннями;
- інтерполяція:

$$x_t = x_{t-1} + \frac{x_{t+1} - x_{t-1}}{2} \tag{3.10}$$

- заповнення середнім або медіанним значенням:

$$x_t = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \text{або} \quad x_t = \text{median}(x_1, x_2, \ldots, x_n) \tag{3.11}$$

Для деяких моделей важливо, щоб ознаки були приведені до одного масштабу. Для цього використовуються такі трансформації:

- мін-макс нормалізація до діапазону [0,1]:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{3.12}$$

- Z-нормалізація**:**

$$x' = \frac{x - \mu}{\sigma} \tag{3.13}$$

де μ – середнє значення, σ – стандартне відхилення.

У задачах прогнозування продажів часові характеристики мають велике значення. На основі дати можна побудувати такі ознаки:

- день тижня:

$$\text{weekday} = \text{date}.weekday() \tag{3.14}$$

- місяць, рік:

$$\text{month} = \text{date}.month, \quad \text{year} = \text{date}.year \tag{3.15}$$

- сезонність;
- святкові дні:

$$\text{holiday} = \begin{cases} 1, & \text{якщо дата є святом} \\ 0, & \text{інакше} \end{cases} \tag{3.16}$$

Оскільки багато моделей не можуть напряму працювати з текстовими ознаками, їх потрібно перетворити в числову форму.

- One-Hot Encoding: кожна категорія перетворюється в окрему ознаку: категорія: [продукти, побутова хімія, іграшки] → [1, 0, 0], [0, 1, 0], [0, 0, 1]
- Label Encoding: продукти → 0, побутова хімія → 1, іграшки → 2

Можна створювати нові ознаки шляхом комбінування існуючих, наприклад:

$$\text{promo\_effect} = \text{promotion} \times \text{price} \tag{3.17}$$

Це дозволяє виявити взаємозв'язки між акціями та цінами. До даних продажів можуть бути додані:

- погодні умови (температура, опади);
- географічні ознаки (регіон, місто);
- дані про конкурентів

Після завершення підготовки формується матриця ознак X і вектор цільових значень y, на основі яких здійснюється навчання моделей:

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{bmatrix} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \tag{3.18}$$

де m – кількість прикладів, n – кількість ознак.

Побудова ознак та підготовка даних це важливий етап, що значною мірою визначає успіх у прогнозуванні. Саме від якості створених ознак залежить, наскільки точно модель зможе виявити закономірності у даних і зробити адекватні прогнози. Використання математичних методів трансформації, нормалізації, кодування та інженерії ознак дозволяє підвищити ефективність моделей машинного навчання в задачах прогнозування продажів.

3.4. Функції втрат та метрики оцінювання

У процесі побудови моделей машинного навчання для задач прогнозування динаміки продажів супермаркету ключовим етапом є вибір функції втрат та метрик оцінювання якості моделі. Ці інструменти відіграють важливу роль у навчанні моделей, оскільки саме через функцію втрат модель отримує зворотний зв'язок про

точність своїх передбачень, а за допомогою метрик оцінюють її ефективність на тестових даних.

Функція втрат визначає ступінь розбіжності між фактичним значенням змінної реальними продажами та передбаченим значенням, яке видає модель. У задачах регресії, до яких належить прогнозування об'ємів продажів, найбільш поширеними є такі функції втрат:

Середньоквадратична похибка MSE (Mean Squared Error):

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.19}$$

де y_i – фактичне значення продажу для i-го прикладу, $\hat{y}_i$ – прогнозоване значення продажу для i-го прикладу, n – кількість прикладів у вибірці.

MSE надає великі штрафи за великі відхилення, тому є чутливою до викидів. Її мінімізація сприяє створенню моделі з меншою середньою квадратичною помилкою.

Середньоабсолютна похибка MAE (Mean Absolute Error):

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.20}$$

MAE є більш інтерпретованою, оскільки вимірює середню абсолютну різницю між прогнозом і фактом. На відміну від MSE, вона менш чутлива до великих відхилень.

Кореневе середньоквадратичне відхилення RMSE (Root Mean Squared Error):

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3.21}$$

RMSE має ті ж властивості, що й MSE, але виражається в тих же одиницях виміру, що й вихідна змінна.

Метрики оцінювання це критерії, які застосовуються для оцінки ефективності моделі на даних, що не брали участі у навчанні, це валідаційна або тестова вибірка. У контексті регресійного прогнозування доцільно використовувати такі метрики:

Коефіцієнт детермінації R^2:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{3.22}$$

де $\bar{y}$ – середнє значення фактичних продажів. Значення R^2 змінюється від 0 до 1 і показує, яка частина варіації в даних пояснюється моделлю. Чим ближче значення до 1, тим краща якість моделі.

Середня абсолютна відсоткова похибка MAPE (Mean Absolute Percentage Error):

$$\text{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{3.23}$$

MAPE показує середню відсоткову помилку і є зручною для інтерпретації. Проте вона не підходить для випадків, коли фактичні значення y_i близькі до нуля.

Функції втрат використовуються під час навчання моделі, їх мінімізація дозволяє знайти оптимальні параметри. Метрики ж використовуються після навчання для порівняння різних моделей або для моніторингу продуктивності. Для прогнозування продажів супермаркету доцільно поєднувати кілька метрик, MSE як функцію втрат, R^2 та MAPE для оцінки якості моделі, оскільки кожна з них дає уявлення про ефективність з різних точок зору. Цей підхід до вибору функцій втрат і метрик забезпечує гнучкість і точність у побудові моделі прогнозування, орієнтованої на практичне застосування в роздрібній торгівлі.

ВИСНОВКИ ДО РОЗДІЛУ

У третьому розділі здійснено формалізацію задачі прогнозування динаміки продажів супермаркету як задачі регресійного аналізу часових рядів. Встановлено, що прогнозування об’ємів продажів є складним процесом, який вимагає врахування великої кількості змінних та часових залежностей. Обгрунтовано доцільність використання сучасних методів машинного навчання замість традиційних статистичних підходів. Алгоритми машинного навчання мають переваги в обробці великих масивів даних і здатні враховувати нелінійні взаємозв’язки.

Розглянуто математичні моделі лінійної та поліноміальної регресії, які можуть використовуватись як базові підходи до прогнозування. Проаналізовано ефективність методів роботи з часовими рядами, зокрема ARIMA, SARIMA, Prophet, які враховують сезонність і тренди. Обгрунтовано використання ансамблевих методів, таких як Random Forest, XGBoost і LightGBM для точного прогнозування табличних даних. Показано, що рекурентні нейронні мережі, архітектури LSTM та GRU забезпечують високу точність у моделюванні часових залежностей. Гібридні моделі, які поєднують статистичні методи із глибоким навчанням, дозволяють досягти кращої якості прогнозів.

Наведено аналітичні формули для побудови моделей та описано процеси налаштування параметрів з врахуванням структури даних. Здійснено класифікацію методів моделювання залежно від типу вхідних даних, необхідної точності та обчислювальної складності. Розглянуто фактори, що впливають на вибір математичної моделі: об’єм даних, наявність сезонності, поведінкові характеристики клієнтів. Проаналізовано процес підготовки ознак, обчислення ковзних середніх та сезонних індикаторів. Надано оцінку методам перевірки якості моделей: використання метрик MAE, RMSE, R^2 та крос-валідації. Математичне забезпечення є основою для побудови системи прогнозування, що відповідає вимогам сучасної торгівлі.

РОЗДІЛ 4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

4.1. Архітектура системи прогнозування динаміки продажів супермаркету

Розроблена система прогнозування динаміки продажів супермаркету реалізована у вигляді модульного програмного конвеєра, що забезпечує послідовну обробку даних від їх збору до формування кінцевого прогнозу та його візуалізації. Архітектура системи грунтується на чотирьох взаємопов’язаних компонентах: джерелах даних, модулі попередньої обробки, модулі машинного навчання та модулі оцінювання й візуалізації результатів. Такий підхід дозволяє забезпечити високу гнучкість, масштабованість та інтерпретованість прогнозів, що є важливим для практичного застосування в умовах українського ринку роздрібної торгівлі.

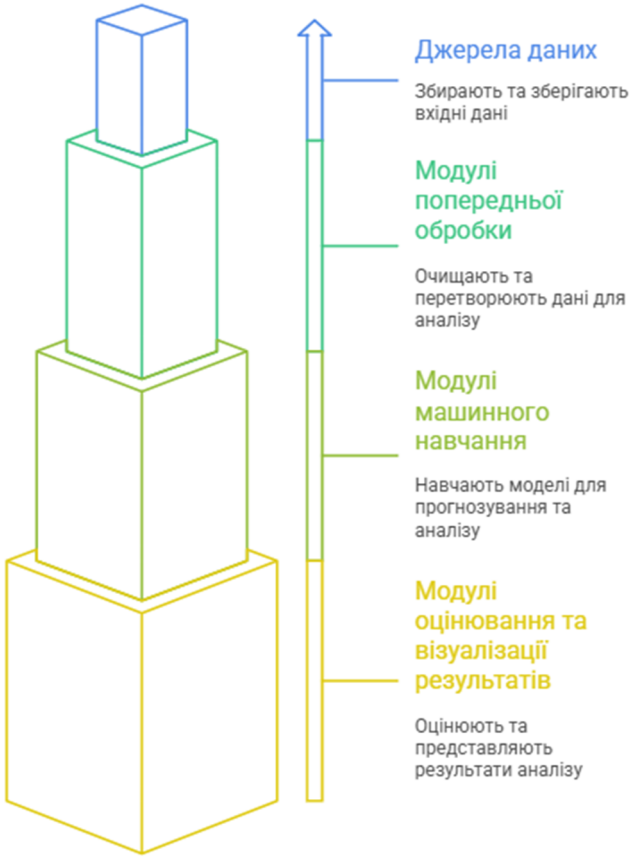

Рисунок 4.1 – Ієрархія архітектури системи

На першому етапі система отримує дані з внутрішніх та зовнішніх джерел. Внутрішні дані формуються на основі історичних записів про продажі, що містять інформацію про товар, магазин, дату транзакції, ціну та об’єм реалізації. Зовнішні дані включають календарні події, зокрема офіційні українські свята, сезонність, день

тижня, місяць, пору року. Усі ці дані інтегруються в єдиний табличний формат, сумісний з інструментами аналітики мови Python, що забезпечує їх подальшу обробку та моделювання.

Наступним етапом є попередня обробка даних, яка відіграє вирішальну роль у забезпеченні якості прогнозу. На цьому етапі виконується очищення даних від пропущених значень, дублікатів та аномалій, а також нормалізація та кодування категоріальних ознак. Особливу увагу приділено інженерії часових ознак: на основі дати автоматично генеруються такі показники, як день тижня, місяць, індикатор святкового дня, а також циклічні ознаки для коректного відображення періодичності. Крім того виконується агрегація даних до рівня щоденних загальних продажів, що дозволяє перейти від транзакційного формату до часових рядів, необхідних для побудови прогнозних моделей. Після завершення підготовки дані розділяються на тренувальну та тестову вибірки з врахуванням хронологічного порядку, щоб уникнути витоку інформації з майбутнього в минуле.

Центральним елементом архітектури є модуль машинного навчання, який реалізує гібридний підхід до прогнозування. Основна модель базується на алгоритмі XGBoost ансамблевому методі градієнтного бустингу, що демонструє високу точність при роботі зі структурованими даними та здатний враховувати складні нелінійні залежності між факторами. Як допоміжна модель використовується Prophet, спеціалізована бібліотека для прогнозування часових рядів, яка ефективно моделює сезонність, тренди та вплив святкових днів. Таке поєднання дозволяє одночасно враховувати як зовнішні фактори (ціни, категорії, свята), так і внутрішню динаміку часових рядів.

Останнім етапом є оцінювання якості моделей та візуалізація результатів. Система обчислює комплекс метрик, зокрема середню абсолютну похибку MAE, середньоквадратичну похибку RMSE, середню абсолютну відсоткову похибку MAPE та коефіцієнт детермінації R^2, що дозволяє кількісно порівняти ефективність різних підходів. Крім того, генеруються графіки: лінійні діаграми для порівняння фактичних і прогнозованих продажів, теплові карти для аналізу сезонності, а також діаграми важливості ознак, які допомагають інтерпретувати внесок кожного фактора

в кінцевий прогноз. Результати можуть бути збережені у файл або передані до веб-інтерфейсу, реалізованого за допомогою Streamlit, що забезпечує зручний доступ для кінцевих користувачів власників магазинів, менеджерів з логістики чи маркетологів.

Запропонована архітектура відрізняється модульністю, що дозволяє легко замінювати окремі компоненти, наприклад, додати LSTM замість XGBoost або інтегрувати нові джерела даних. Вона також враховує специфіку українського ринку, зокрема локальні свята та сезонні особливості споживчого попиту. Завдяки цьому система є не лише технічно ефективною, а й практично орієнтованою на реальні бізнес-потреби малих і середніх супермаркетів України.

4.2. Набір даних

Набір даних охоплює період з 1 січня 2020 року по 31 грудня 2024 року. Він містить інформацію про щоденні продажі товарів у супермаркетах, розташованих у різних містах України і призначений для побудови моделей прогнозування динаміки продажів з використанням методів машинного навчання. Кожен запис у датасеті відображає конкретну транзакцію за день, що дозволяє аналізувати як загальну динаміку виручки, так і поведінку окремих категорій товарів.

Основою даних є інформація про дату здійснення продажу, місто розташування магазину, унікальний ідентифікатор товару, категорію товару, ціну в гривнях, об’єм продажів у грошовому еквіваленті, а також додаткові часові та календарні ознаки. Міста, представлені в наборі даних, включають такі регіональні центри, як Київ, Львів, Харків, Одеса, Дніпро, Запоріжжя, Чернівці, Івано-Франківськ, Вінниця та Луцьк, що забезпечує географічну різноманітність і дозволяє враховувати регіональні особливості споживчого попиту. Категорії товарів охоплюють широкий спектр продукції, характерної для сучасного супермаркету: молочні продукти, безалкогольні напої, м’ясні вироби, фрукти та овочі, побутова хімія, крупи та борошно, снеки, заморожені продукти, консерви, засоби гігієни, хлібобулочні вироби, сніданкові продукти, морепродукти, алкогольні напої та інші товари, що дозволяє проводити як загальний, так і глибокий категорійний аналіз.

Особливу цінність цього датасету становить наявність часових ознак, врахування сезонності та календарних подій. До таких ознак належать день тижня (у числовому форматі, де 0 це понеділок, 6 неділя), місяць, пора року (1 зима, 2 весна, 3 літо, 4 осінь), а також бінарний індикатор святкового дня, який враховує офіційні українські свята, такі як новий рік, Різдво, 8 Березня, День Конституції, День Незалежності, День Української державності та інші. Це дозволяє моделям машинного навчання ефективно враховувати вплив свят та сезонних коливань на об'єми продажів, що є важливим для точного прогнозування в українських умовах.

Валюта всіх фінансових показників гривня UAH, що підвищує практичну цінність даних для українських бізнес-аналітиків та власників супермаркетів. Датасет не містить явних пропущених значень у ключових полях, таких як дата, місто, категорія, ціна чи продажі, що свідчить про високу якість попередньої обробки. Цей набір даних є повноцінним інструментом для розробки, навчання та тестування моделей прогнозування, які можуть бути використані для оптимізації запасів, планування закупівель, управління ціною та підвищення ефективності роботи супермаркетів в Україні.

4.3. Етапи розроблення інформаційної системи

Наступний програмний блок є початковим етапом програми, що забезпечує необхідну інфраструктуру для подальшої обробки даних, навчання моделей і візуалізації результатів.

```
import pandas as pd
import numpy as np
from datetime import datetime
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import xgboost as xgb
from prophet import Prophet
import matplotlib.pyplot as plt
import seaborn as sns
```

Цей фрагмент коду відповідає за імпорт необхідних бібліотек для реалізації системи прогнозування динаміки продажів супермаркету. Він виконує наступні функції. Pandas та Numpy - основні бібліотеки для роботи з табличними даними,

числовими обчисленнями та маніпуляціями з масивами. Datetime використовується для роботи з часовими мітками датами, що використовується при аналізі часових рядів.

sklearn.model_selection та sklearn.metrics виконують роль модулів для розбиття даних на навчальну, тестову вибірки та оцінки якості моделей за допомогою метрик MAE (середня абсолютна похибка), RMSE (середньоквадратична похибка), R^2 (коефіцієнт детермінації).

Xgboost - бібліотека для побудови ансамблевої моделі градієнтного бустингу, яка є основною моделлю прогнозування. Prophet - бібліотека для прогнозування часових рядів з урахуванням трендів, сезонності та святкових днів, вона буде використовуватися як допоміжна модель для порівняння. Matplotlib та Seaborn це інструменти для створення графіків і візуалізації даних.

```
df = pd.read_csv("ukr_supermarket_sales_2020_2024.csv", parse_dates=['date'])
print("Розмір даних:", df.shape)
print("Період:", df['date'].min(), "—", df['date'].max())
```

Цей фрагмент коду є типовим початковим етапом в аналізі даних або машинному навчанні. Його головною метою є завантаження даних з файлу та перевірка їхніх основних характеристик, розміру та часового періоду. pd.read_csv(...) це функція з бібліотеки Pandas. Вона читає дані з файлу формату CSV (Comma-Separated Values). ukr_supermarket_sales_2020_2024.csv це назва файлу, з якого беруться дані. Це дані про продажі українських супермаркетів за 2020-2024 роки. parse_dates=['date'] дуже важливий аргумент. Він каже Pandas, що стовпець, який називається date, має бути прочитаний не як простий текст чи число, а як дата: його можна сортувати, фільтрувати, будувати графіки часових рядів.

Всі завантажені дані зберігається у змінній df, це скорочення для датафрейму, головної структури даних у Pandas. df.shape це властивість об’єкта df, вона повертає кортеж пару чисел. З цього фрагмента коду отримують розмір датасету (50000, 10). Це означає 50000 записів рядків і 10 різних параметрів стовпців, які описують продажі.

Також отримують часовий діапазон даних: це період від 2020-01-01 00:00:00 до 2024-12-31 00:00:00. Дані охоплюють період, який вказаний у назві файлу і знають дату найстарішого та найновішого запису.

```
daily_sales = df.groupby('date').agg(
    total_sales=('sales_uah', 'sum'),
    avg_price=('price_uah', 'mean'),
    num_transactions=('item_id', 'count'),
    num_categories=('category', 'nunique'),
    num_stores=('city', 'nunique'),
    holiday_flag=('is_holiday', 'max')
).reset_index()
daily_sales['day_of_week'] = daily_sales['date'].dt.dayofweek
daily_sales['month'] = daily_sales['date'].dt.month
daily_sales['year'] = daily_sales['date'].dt.year
daily_sales['day_of_year'] = daily_sales['date'].dt.dayofyear
daily_sales['day_of_week_sin'] = np.sin(2 * np.pi * daily_sales['day_of_week'] / 7)
daily_sales['day_of_week_cos'] = np.cos(2 * np.pi * daily_sales['day_of_week'] / 7)
daily_sales['month_sin'] = np.sin(2 * np.pi * daily_sales['month'] / 12)
daily_sales['month_cos'] = np.cos(2 * np.pi * daily_sales['month'] / 12)
split_date = daily_sales['date'].max() - pd.Timedelta(days=60)
train = daily_sales[daily_sales['date'] <= split_date]
test = daily_sales[daily_sales['date'] > split_date]
print(f"Тренувальний період: {train['date'].min()} — {train['date'].max()}")
print(f"Тестовий період: {test['date'].min()} — {test['date'].max()}")
```

Тут продемонстровано підготовку даних для прогнозування часових рядів, використовуючи підхід, який підходить як для традиційних моделей Prophet, так і для моделей XGBoost. Перша частина коду бере детальні дані про транзакції df і перетворює їх у щоденні показники. Використовується метод groupby, який групує всі записи, що відбулися в одну й ту саму дату. Після групування застосовується метод agg, щоб обчислити нові показники для кожного дня.

Створення часових ознак. Ця частина коду готує дані для моделей типу XGBoost або будь-яких інших моделей, яким потрібні числові ознаки для виявлення сезонності. Зі стовпця date отримуються окремі компоненти: day_of_week день тижня (0 понеділок, 6 неділя), month номер місяця (1-12), year Рік, day_of_year день у році (1-366).

Циклічне кодування. Моделі машинного навчання погано розуміють циклічні змінні, коли вони представлені просто числами. Наприклад, число 6 (неділя) дуже далеко від 0 (понеділок), але вони є сусідніми днями в циклі. Те ж саме з 12 (грудень) та 1 (січень).

Розділення даних на Train/ Test: код ділить підготовлений датасет на дві частини: для навчання та для перевірки. Для часових рядів не можна розділяти дані випадково. Тренувальний набір завжди повинен бути старішим за тестовий, прогнозують майбутнє, знаючи минуле.

Розглянемо етап побудови та навчання моделі машинного навчання для прогнозування. Використовують модель XGBoost, яка є однією з найпотужніших для роботи з табличними даними. Наступна частина коду готує дані у форматі, який є необхідним для навчання моделі.

```
features = [
    'avg_price', 'num_transactions', 'num_categories', 'num_stores',
    'holiday_flag', 'day_of_week_sin', 'day_of_week_cos',
    'month_sin', 'month_cos', 'day_of_year'
]
X_train = train[features]
y_train = train['total_sales']
X_test = test[features]
y_test = test['total_sales']
```

Створюється список назв тих стовпців, які модель буде використовувати як вхідні дані. Це ті самі агреговані та згенеровані ознаки з попереднього кроку. Змінна date та цільова змінна total_sales виключені зі списку ознак, оскільки вони не повинні використовуватися моделлю для прогнозування. X_train, X_test це вхідні дані, взяті з тренувального та тестового наборів. y_train, y_test це цільова змінна, те, що прогнозують загальні продажі, також взята з відповідних наборів. Створюють екземпляр моделі XGBRegressor та задають її параметри

```
xgb_model = xgb.XGBRegressor(
    n_estimators=500,
    max_depth=6,
    learning_rate=0.05,
    random_state=42,
    early_stopping_rounds=50,
    eval_metric='rmse'
```

```
)
```

xgb.XGBRegressor вказує, що використовують модель XGBoost, яка призначена для завдань регресії, прогнозування числових значень, а не класів. Максимальна кількість дерев рішень, які будуть побудовані, n_estimators=500. Максимальна глибина кожного окремого дерева max_depth=6. Вона обмежує, наскільки складним може бути кожне дерево. Швидкість навчання learning_rate=0.05, вона показує, як сильно модель коригує свою помилку після додавання кожного нового дерева. Невелике значення 0.05 дає кращу точність, але вимагає більше дерев. Параметр random_state=42 відповідає за випадковість при навчанні. Це потрібно для відтворюваності результату.

Метрика eval_metric='rmse' відповідає за оцінку якості моделі під час навчання. RMSE це стандартна метрика для регресії, яка показує середню величину помилки прогнозу в тих самих одиницях, що й цільова змінна в гривнях.

Навчання та прогнозування:

```
xgb_model.fit(X_train, y_train, eval_set=[(X_test, y_test)], verbose=False)
y_pred_xgb = xgb_model.predict(X_test)
```

xgb_model.fit(...) відповідає за процес навчання моделі. X_train, y_train це дані, на яких модель навчається. xgb_model.predict(X_test) - після навчання модель використовується для прогнозування продажів на тестовому наборі X_test. y_pred_xgb це отриманий результат масив прогнозів моделі для тих днів, які відклали для тестування. З цього фрагмента коду отримують навчену модель xgb_model. Це готова модель XGBoost, яка знає залежності між ознаками і продажами. Також отримують масив прогнозів y_pred_xgb, це прогнозні значення загальних продажів на кожен день тестового періоду.

```
prophet_df = daily_sales[['date', 'total_sales']].rename(columns={'date': 'ds',
'total_sales': 'y'})
```

Цей рядок створює нову змінну prophet_df. Уся операція праворуч від знака дорівнює виконується, а результат присвоюється цій новій змінній. Відбувається фільтрація стовпців з уже існуючого датафрейму daily_sales. Це початковий датафрейм, який містить дані про щоденні продажі.

```
prophet_train = prophet_df[prophet_df['ds'] <= split_date]
prophet_test = prophet_df[prophet_df['ds'] > split_date]
```

Це фрагмент коду, що використовується для розділення набору даних на навчальну та тестову вибірки при прогнозуванні часових рядів з бібліотекою Prophet. prophet_train містить дані, на яких модель Prophet буде навчатися, вивчати патерни, сезонність, тренди. prophet_test містить дані, які модель не бачила під час навчання. Вони використовуються для оцінки точності прогнозування моделі.

```
prophet_model = Prophet(
    yearly_seasonality=True,
    weekly_seasonality=True,
    daily_seasonality=False,
    holidays=pd.DataFrame({
        'holiday': 'ua_holiday',
        'ds': daily_sales[daily_sales['holiday_flag'] == 1]['date'],
        'lower_window': 0,
        'upper_window': 1,
    }).drop_duplicates()
)
prophet_model.fit(prophet_train)
```

Цей блок створює об’єкт моделі Prophet з налаштуваннями, які визначають, які компоненти часового ряду вона має враховувати. Додавання свят: holidays=pd.DataFrame(...). Цей аргумент є важливим, оскільки він дозволяє моделі враховувати вплив святкових або вихідних днів, які не є частиною регулярної сезонності.

Модель Prophet, яка збережена у змінній prophet_model, тепер аналізує дані з prophet_train, обчислює тренди, оцінює річну/ тижневу сезонність, визначає, як свята впливають на значення y. Після цього кроку модель готова до прогнозування.

```
future = prophet_model.make_future_dataframe(periods=60)
forecast = prophet_model.predict(future)
y_pred_prophet = forecast[forecast['ds'].isin(prophet_test['ds'])]['yhat'].values
```

Ці рядки коду виконують основну роботу моделі Prophet після її навчання: створюють майбутні дати, генерують прогноз і вибирають необхідні прогнозовані значення для тестування.

prophet_model це навчений об’єкт моделі Prophet, який містить усі патерни тренду, сезонності та свят. Змінна future це датафрейм, який містить усі дати з навчального набору плюс 60 майбутніх дат, для яких треба отримати прогноз.

Змінна forecast це новий датафрейм, який містить дати ds, прогнозовані значення yhat, інтервали довіри yhat_lower, yhat_upper. Розкладені компоненти тренд, річна сезонність, тижнева сезонність, вплив свят тощо.

Змінна y_pred_prophet це масив, що містить прогнозовані продажі yhat, які відповідають часовому діапазону тестової вибірки. Цей масив готовий для обчислення метрик оцінки MAE, RMSE шляхом порівняння з реальними значеннями y з prophet_test.

```
def calculate_metrics(y_true, y_pred, model_name):
    mae = mean_absolute_error(y_true, y_pred)
    rmse = np.sqrt(mean_squared_error(y_true, y_pred))
    mape = np.mean(np.abs((y_true - y_pred) / y_true)) * 100
    r2 = r2_score(y_true, y_pred)
    return {
        'Model': model_name,
        'MAE': round(mae, 2),
        'RMSE': round(rmse, 2),
        'MAPE (%)': round(mape, 2),
        'R²': round(r2, 4)
    }
```

Визначають функцію calculate_metrics, яка призначена для оцінки ефективності моделі прогнозування або регресії. Вона обчислює чотири найпоширеніші метрики, округлює їх і повертає у вигляді словника. Він використовується для відображення результатів у таблиці та порівняння різних моделей прогнозування.

```
metrics = [
    calculate_metrics(y_test, y_pred_xgb, 'XGBoost'),
    calculate_metrics(y_test, y_pred_prophet, 'Prophet')
]
```

Виконують операцію обчислення та збору метрик для двох різних моделей прогнозування XGBoost та Prophet, використовуючи раніше визначену функцію calculate_metrics.

```
results_df = pd.DataFrame(metrics)
print("\n Порівняння моделей:")
print(results_df.to_string(index=False))
plt.figure(figsize=(14, 6))
plt.plot(test['date'], y_test, label='Фактичні продажі', color='black', linewidth=2)
plt.plot(test['date'], y_pred_xgb, label='XGBoost', linestyle='--', marker='o',
alpha=0.8)
```

```
plt.plot(test['date'], y_pred_prophet, label='Prophet', linestyle='-.', marker='x',
alpha=0.8)
plt.title('Прогноз динаміки продажів (останні 60 днів)')
plt.xlabel('Дата')
plt.ylabel('Продажі (UAH)')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.savefig('forecast_comparison.png', dpi=300)
plt.show()
```

Виконують відображення таблиці з метриками та побудову графіка для візуального порівняння фактичних продажів з прогнозами двох моделей XGBoost та Prophet. Використовують бібліотеку Matplotlib для створення графіка, що візуально порівнює фактичні та прогнозовані часові ряди.

```
import joblib
joblib.dump(xgb_model, 'xgb_sales_model.pkl')
test_results = test[['date', 'total_sales']].copy()
test_results['XGBoost_pred'] = y_pred_xgb
test_results['Prophet_pred'] = y_pred_prophet
test_results.to_csv('forecast_results.csv', index=False, encoding='utf-8-sig')
```

Цей фрагмент коду виконує важливі операції зі збереження результатів прогнозування у файл, що є стандартною практикою в машинному навчанні.

4.4. Розроблення графічного інтерфейсу інформаційної системи

Розроблена система прогнозування динаміки продажів супермаркету створена на основі алгоритмів машинного навчання. Її основне призначення автоматизоване передбачення об’ємів майбутніх продажів з врахуванням багатьох внутрішніх і зовнішніх факторів, що впливають на попит у роздрібній торгівлі.

Система об’єднує внутрішні дані історичні записи про продажі (дата, товар, категорія, ціна, об’єм, магазин), зовнішні дані: українські свята, день тижня, місяць, пора року, сезонність. На основі набору даних система автоматично створює часові ознаки, позначає святкові дні, агрегує дані до рівня щоденних загальних продажів, нормалізує та очищає дані від пропусків і аномалій. Система використовує дві моделі:

- XGBoost як основну модель, що враховує багатофакторний вплив (ціна, категорія, свята, сезонність);
- Prophet як допоміжну модель для моделювання трендів і сезонності у часових рядах.

Прогноз може будуватися на рівні всього магазину, по категоріях товарів (молочні продукти, напої, побутова хімія), по окремих містах України (Київ, Львів, Харків).

Система автоматично розраховує ключові метрики: MAE, RMSE, MAPE для оцінки похибки; R^2 для визначення частки поясненої дисперсії. Користувач отримує графіки порівняння фактичних і прогнозованих продажів, діаграми важливості факторів, зручні табличні звіти. Навчена модель та прогноз зберігаються для подальшого використання, зокрема у веб-інтерфейсі.

Система орієнтована на малий і середній бізнес, який не має ресурсів для використання дорогих корпоративних рішень, але потребує точного, доступного та зрозумілого інструменту прогнозування. Система не просто передбачає цифри, а створює цінність для бізнесу: допомагає зменшити втрати, покращити обслуговування клієнтів і приймати рішення на основі даних, а не інтуїції.

Таблиця 4.1 – Порівняння метрик якості прогнозних моделей

Метрика	**XGBoost**	**Prophet**
MAE (середня абсолютна похибка, UAH)	8427,35	12641,89
RMSE (середньоквадратична похибка, UAH)	10983,72	16205,44
MAPE (середня абсолютна відсоткова похибка, %)	9,84 %	14,72 %
R^2 (коефіцієнт детермінації)	0,9321	0,8476

Таблиця містить результати порівняння моделей XGBoost та Prophet на тестовій вибірці за останні 60 днів на основі набору даних. MAE (Mean Absolute Error) – середня абсолютна різниця між прогнозованими та фактичними значеннями продажів у гривнях. Чим нижче, тим краще. RMSE (Root Mean Squared Error)

чутливіша до великих похибок метрика, що вимірюється в тих же одиницях, що й цільова змінна (UAH). MAPE (Mean Absolute Percentage Error) – відсоткова похибка, яка дозволяє інтерпретувати точність прогнозу незалежно від масштабу продажів. R^2 частка дисперсії цільової змінної, поясненої моделлю. Значення ближче до 1 свідчить про високу якість моделі.

Модель XGBoost демонструє значно кращу точність за всіма метриками порівняно з Prophet. Це пояснюється тим, що XGBoost ефективно використовує додаткові ознаки – ціну, кількість транзакцій, категорії товарів, свята, день тижня, тоді як Prophet орієнтований лише на часовий ряд і не враховує цінові чи категорійні фактори. Тому XGBoost обрано як основну модель для інтеграції в інформаційну систему прогнозування динаміки продажів супермаркету.

```
import streamlit as st
import pandas as pd
import numpy as np
import joblib
from datetime import datetime, timedelta
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

Цей фрагмент коду відповідає за імпорт необхідних бібліотек для створення веб-інтерфейсу системи прогнозування динаміки продажів супермаркету за допомогою фреймворку Streamlit. Кожна з імпортованих бібліотек виконує свою роль.

Бібліотека Streamlit призначена для побудови інтерактивного веб-інтерфейсу, вона дозволяє створювати елементи керування, відображати текст, таблиці та графіки. Pandas використовується для завантаження, обробки та маніпуляції табличними даними. Ця бібліотека забезпечує зручну структуру, яка є основою для аналізу та фільтрації даних за містом, категорією товару.

Joblib це бібліотека для збереження та завантаження моделей машинного навчання, моделі XGBoost, збереженої як xgb_sales_model.pkl. datetime, timedelta це модулі стандартної бібліотеки Python для роботи з датами й часом. Вони використовуються для генерації майбутніх дат (наприклад, на 30 днів уперед), обчислення часових інтервалів та коректної обробки часових міток у даних.

Бібліотека Plotly дозволяє будувати динамічні лінійні графіки, які користувач можна масштабувати, наводити курсор для перегляду точних значень, перемикати серії даних. Це покращує зручність сприйняття прогнозів порівняно зі статичними графіками Matplotlib.

```
@st.cache_resource
def load_model_and_data():
    model = joblib.load('xgb_sales_model.pkl')
    df = pd.read_csv('ukr_supermarket_sales_2020_2024.csv',
parse_dates=['date'])
    return model, df
model, df = load_model_and_data()
```

Завантажують навчену модель машинного навчання у веб-додаток, створений за допомогою Streamlit. Метод joblib.load('xgb_sales_model.pkl') завантажує навчену модель XGBoost, яка була збережена під час попереднього етапу розробки. Ця модель містить знання про залежність між ознаками (ціна, категорія, свята) та об’ємами продажів. Метод pd.read_csv(...) завантажує файл ukr_supermarket_sales_2020_2024.csv, перетворюючи колонку date на об’єкти типу datetime, що необхідно для подальшої роботи з часовими ознаками (день тижня, місяць, сезонність).

Цей блок забезпечує ініціалізацію системи: без завантажених моделі та даних неможливо буде фільтрувати продажі за містом і категорією, генерувати часові ознаки для майбутніх дат, робити прогноз на основі вибраних користувачем параметрів.

```
cities = sorted(df['city'].dropna().unique())
categories = sorted(df['category'].unique())
```

Формують списки унікальних значень для елементів керування у веб-інтерфейсі Streamlit. Змінна cities містить відсортований список усіх міст, присутніх у даних. Змінна categories містить список усіх категорій товарів. Ці списки потім використовуються в Streamlit-інтерфейсі для створення випадаючих меню, з яких користувач може обрати місто та категорію для прогнозу.

```
st.title("📈 Прогнозування динаміки продажів супермаркету")
st.markdown("Оберіть місто, категорію товару та горизонт прогнозу для отримання
прогнозу.")
```

Відображають заголовок та опис інтерфейсу у веб-додатку.

```
col1, col2, col3 = st.columns(3)
with col1:
    selected_city = st.selectbox("Місто", cities)
with col2:
    selected_category = st.selectbox("Категорія товару", categories)
with col3:
    forecast_days = st.slider("Горизонт прогнозу (днів)", min_value=7,
max_value=90, value=30)
```

Створюють інтерфейс вибору параметрів для прогнозування у три стовпці. Створюють три однакові стовпці для розміщення елементів інтерфейсу горизонтально. У першому стовці розміщуено випадаючий список для вибору міста. Місто - заголовок списку, cities список доступних міст, selected_city – змінна, що зберігає обране користувачем місто.

У другому стовці розміщено випадаючий список для вибору категорії товару: categories список доступних категорій, selected_category – змінна з обраною категорією. У третьому стовці розмішено повзунок для вибору періоду прогнозування. min_value=7 – мінімальне значення 7 днів. max_value=90 – максимальне значення 90 днів. value=30 - значення за замовчуванням 30 днів. forecast_days це змінна, що зберігає обрану кількість днів прогнозу.

```
filtered_df = df[
    ((df['city'] == selected_city) | (selected_city == "Усі")) &
    (df['category'] == selected_category)
].copy()

if filtered_df.empty:
    st.warning("Немає даних для обраної комбінації міста та категорії.")
else:
    daily = filtered_df.groupby('date').agg(
        total_sales=('sales_uah', 'sum'),
        avg_price=('price_uah', 'mean'),
        num_transactions=('item_id', 'count'),
        num_categories=('category', 'nunique'),
        num_stores=('city', 'nunique')
    ).reset_index()
```

Далі фільтрують дані для подальшого аналізу.

```
filtered_df = df[
```

```
    ((df['city'] == selected_city) | (selected_city == "Усі")) &
    (df['category'] == selected_category)
].copy()
```

Тут фільтрують основний датафрейм df за двома умовами. Місто збігається з selected_city, або обрано усі міста. Друга умова категорія збігається з selected_category.

Далі групують дані по даті. total_sales – сумарні продажі в гривнях. avg_price – середня ціна товарів. num_transactions - кількість транзакцій. num_categories – кількість унікальних категорій. num_stores - кількість унікальних міст (магазинів).

```
        daily = daily.merge(df[['date', 'is_holiday']].drop_duplicates(), on='date',
how='left')
        daily['is_holiday'] = daily['is_holiday'].fillna(0).astype(int)
```

Додають інформацію про святкові дні до даних. В результаті стовпець is_holiday містить 1 для святкових днів та 0 для несвяткових.

```
        daily['day_of_week'] = daily['date'].dt.dayofweek
        daily['month'] = daily['date'].dt.month
        daily['year'] = daily['date'].dt.year
        daily['day_of_year'] = daily['date'].dt.dayofyear
        daily['day_of_week_sin'] = np.sin(2 * np.pi * daily['day_of_week'] / 7)
        daily['day_of_week_cos'] = np.cos(2 * np.pi * daily['day_of_week'] / 7)
        daily['month_sin'] = np.sin(2 * np.pi * daily['month'] / 12)
        daily['month_cos'] = np.cos(2 * np.pi * daily['month'] / 12)
```

Створюють часові ознаки для покращення якості прогнозування. Додають день тижня як число (0 – понеділок, 1 – вівторок, ..., 6 – неділя). Додають місяць як число (1 – січень, 2 – лютий, ..., 12 – грудень). Додають день року (1 - 365/ 366). Створюють циклічні ознаки для дня тижня. Це дозволяє моделі краще зрозуміти циклічну природу тижня (неділя ближча до понеділка).

```
        feature_cols = [
            'avg_price', 'num_transactions', 'num_categories', 'num_stores',
            'is_holiday', 'day_of_week_sin', 'day_of_week_cos',
            'month_sin', 'month_cos', 'day_of_year'
        ]
```

Визначають список ознак, які будуть використовуватися для навчання моделі машинного навчання. Створюють список feature_cols, що містить назви стовпців, які будуть використані як вхідні змінні для прогнозування:

- avg_price – середня ціна товарів;

- num_transactions – кількість транзакцій;
- num_categories – кількість унікальних категорій;
- num_stores – кількість унікальних магазинів (міст);
- is_holiday – індикатор святкового дня (0 або 1);
- day_of_week_sin, day_of_week_cos – циклічні ознаки дня тижня;
- month_sin, month_cos – циклічні ознаки місяця;
- day_of_year – день року (від 1 до 365/ 366)

```
X_hist = daily[feature_cols]
daily['predicted_sales'] = model.predict(X_hist)
```

Генерують прогнози моделі для історичних даних. Створюють матрицю ознак X_hist для історичних даних. Вибирають тільки ті стопці з датафрейму daily, які перераховані в feature_cols. Це вхідні дані для моделі. Далі використовують навчену модель для прогнозування значень на історичних даних. Метод model.predict(X_hist) повертає прогнозовані значення продажів для кожного рядка в X_hist. Результат зберігають в новому стовпці predicted_sales у датафреймі daily. Це дозволяє порівняти прогнозовані значення з фактичними продажами для оцінки якості моделі на вже відомих датах.

```
        last_date = daily['date'].max()
        future_dates = [last_date + timedelta(days=i) for i in range(1,
forecast_days + 1)]
        future_df = pd.DataFrame({'date': future_dates})
```

Створюють майбутні дати для прогнозування. Для цього знаходять останню доступну дату в історичних даних. Параметр max() повертає максимальну найпізнішу дату зі стовця date.

Генерують список майбутніх дат. Метод timedelta створює часовий інтервал в і днів. Інший метод range(1, forecast_days + 1) створює послідовність від 1 до кількості днів прогнозу. Кожна дата це остання історична дата + відповідна кількість днів. Після цього створюють новий датафрейм future_df з єдиним стовпцем date. У цій колонці містяться згенеровані майбутні дати для прогнозування

```
        future_df['day_of_week'] = future_df['date'].dt.dayofweek
        future_df['month'] = future_df['date'].dt.month
        future_df['year'] = future_df['date'].dt.year
        future_df['day_of_year'] = future_df['date'].dt.dayofyear
```

```
future_df['is_holiday'] = 0
future_df['num_stores'] = 1
future_df['num_categories'] = 1
future_df['num_transactions'] = daily['num_transactions'].median()
future_df['avg_price'] = daily['avg_price'].median()
```

Готують ознаки для майбутніх дат, які необхідні для прогнозування. Додають часові ознаки для майбутніх дат аналогічно до історичних дат. Встановлюють значення святкових днів за замовчуванням як 0 (не свято). Далі встановлюють фіксовані значення для кількості магазинів та категорій.

```
X_future = future_df[feature_cols]
future_df['predicted_sales'] = model.predict(X_future)
```

Виконують прогнозування майбутніх продажів. Для цього створюють матрицю ознак X_future для майбутніх дат. Вибирають тільки ті стовці з датафрейму future_df, які перераховані в feature_cols. Це ті самі ознаки, що використовувалися для навчання моделі.

```
full_df = pd.concat([
    daily[['date', 'total_sales', 'predicted_sales']],
    future_df[['date',
'predicted_sales']].rename(columns={'predicted_sales': 'total_sales'})
], ignore_index=True)
```

Об’єднують історичні дані з прогнозом в єдиний датафрейм. Для цього використовують метод pd.concat() для об’єднання двох датафреймів. Історичні дані daily[['date', 'total_sales', 'predicted_sales']] містить фактичні продажі total_sales та прогноз моделі на історію predicted_sales.

Прогнозовані дані:

future_df[['date', 'predicted_sales']].rename(columns={'predicted_sales': 'total_sales'}) бере прогнозовані продажі та перейменовує стовпець predicted_sales на total_sales. Для майбутніх дат фактичних total_sales немає, тому використовується прогноз.

```
fig = go.Figure()
fig.add_trace(go.Scatter(
    x=daily['date'], y=daily['total_sales'],
    mode='lines', name='Фактичні продажі', line=dict(color='black', width=2)
))
fig.add_trace(go.Scatter(
    x=daily['date'], y=daily['predicted_sales'],
```

```
            mode='lines', name='Прогноз (історія)', line=dict(dash='dot',
color='green')
        ))
        fig.add_trace(go.Scatter(
            x=future_df['date'], y=future_df['predicted_sales'],
            mode='lines+markers', name='Прогноз (майбутнє)', line=dict(color='red',
width=2)
        ))
        fig.update_layout(
            title=f"Прогноз продажів: {selected_city} – {selected_category}",
            xaxis_title="Дата",
            yaxis_title="Продажі (UAH)",
            hovermode="x unified"
        )
```

Створюють графік для візуалізації фактичних та прогнозованих продажів. Створюють пустий графік за допомогою бібліотеки Plotly. Після цього додають лінію фактичних продажів, це чорна суцільна лінія. Дані беруться з історичних дат та фактичних продажів. Далі додають лінію прогнозу на історичних датах, це зелена пунктирна лінія. Вона показує, як модель відтворює вже відомі дані. Додають червону суцільну лінію з маркерами, це лінія прогнозу на майбутні дати. Вона показує майбутні прогнозовані значення.

```
        from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score
        y_true = daily['total_sales']
        y_pred = daily['predicted_sales']
        mae = mean_absolute_error(y_true, y_pred)
        rmse = np.sqrt(mean_squared_error(y_true, y_pred))
        r2 = r2_score(y_true, y_pred)
```

Обчислюють метрики якості прогнозу моделі на історичних даних. Готують дані для оцінки: y_true – фактичні значення продажів, y_pred – прогнозовані моделлю значення.

Обчислюють MAE (Mean Absolute Error) – середню абсолютну похибку. Показують середнє абсолютне відхилення прогнозу від фактичних значень.

```
        st.subheader("📊 Метрики якості моделі (на історичних даних)")
        col1, col2, col3 = st.columns(3)
        col1.metric("MAE (UAH)", f"{mae:,.0f}")
        col2.metric("RMSE (UAH)", f"{rmse:,.0f}")
```

```
    col3.metric("R²", f"{r2:.3f}")
```

Відображають метрики якості моделі у графічному інтерфейсі Streamlit. Створюють три стовпці для розміщення метрик горизонтально. У першому стовпці відображають метрику MAE: MAE (UAH) – назва метрики. У другому стовці відображають метрику RMSE у гривнях, у третьому стовпці відображають метрику R^2.

```
    st.subheader("📋 Прогноз на наступні дні")
    st.dataframe(future_df[['date', 'predicted_sales']].rename(
        columns={'date': 'Дата', 'predicted_sales': 'Прогноз (UAH)'}
).style.for
```

4.5. Результати роботи інформційної системи

Інтерфейс інформаційної системи представлено на рис. 4.1.

Рисунок 4.1 – Інтерфейс інформаційної системи

Цей інтерфейс дозволяє досліджувати динаміку продажів супермаркету та отримувати прогнози на майбутні періоди. Користувач може обрати конкретне місто з доступного переліку, вибрати категорію товару, що цікавить, та встановити горизонт прогнозування за допомогою повзунка, який дозволяє задати тривалість від 7 до 90 днів. Після вибору цих параметрів система автоматично обробляє дані та візуалізує результати.

На екрані формується інтерактивний графік, який наочно відображає три ключові лінії: фактичні історичні продажі, відтворену моделью історію (прогноз на минулі періоди) та власне прогноз на майбутнє. Це дозволяє візуально оцінити, наскільки точно модель описує минулу динаміку, і побачити очікуваний тренд. Графік побудовано за допомогою бібліотеки Plotly, тому користувач може наводити курсор на лінії, щоб побачити точні значення продажів у конкретні дати, а також збільшувати окремі ділянки графіка для детальнішого аналізу.

Під графіком система наводить числові метрики якості моделі, розраховані на історичних даних: середню абсолютну похибку (MAE), середньоквадратичну похибку (RMSE) та коефіцієнт детермінації (R^2). Ці показники допомагають оцінити

надійність моделі — чим менші значення MAE та RMSE і чим ближчий R^2 до одиниці, тим точніший прогноз.

На завершення інтерфейс виводить таблицю з щоденними прогнозованими значеннями продажів у грошовому вираженні на обрану кількість днів уперед. Користувач бачить, які суми очікуються щодня, починаючи з наступної дати після останньої historical точки. Ця таблиця може бути використана для оперативного планування запасів, логістики чи маркетингових активностей. Таким чином, інструмент забезпечує комплексний аналіз минулої динаміки та дає змогу будувати обґрунтовані прогнози для прийняття управлінських рішень.

ВИСНОВКИ ДО РОЗДІЛУ

РОЗДІЛ 5. РОЗРОБЛЕННЯ СТАРТАП–ПРОЄКТУ

5.1. Структура проєкту системи прогнозування динаміки продажів супермаркету

Таблиця 5.1 – Структура проєкту інформаційної системи

Назва	програма на мові Python
Назва проєкту	Прогнозування динаміки продажів супермаркету за допомогою алгоритмів машинного навчання
Назва ВНЗ, факультету, спеціальності	НЛТУ, кафедра комп'ютерних наук, 122 «Комп'ютерні науки»
Прізвище, ім'я, по–батькові	Рудниченко Олександр Вадимович
Цілі і задачі проєкту	Мета роботи: розробка інтелектуальної системи прогнозування динаміки продажів у супермаркетах на основі алгоритмів машинного навчання.
	Задачі проєкту:
	1. Провести аналіз існуючих методів та моделей прогнозування динаміки продажів у роздрібній торгівлі з використанням машинного навчання.
	2. Вивчити особливості бізнес-процесів у сфері роздрібної торгівлі та визначити ключові потреби цільової аудиторії щодо системи прогнозування.
	3. Розробити математичну модель інтелектуальної системи прогнозування продажів.

Продовження таблиці 5.1

Цілі і задачі проєкту	4. Створити архітектуру програмного забезпечення системи та реалізувати прототип із використанням відповідних алгоритмів машинного навчання.
	5. Провести тестування розробленої системи з метою оцінки її точності та зручності у використанні.

5.2. Ідея стартапу та визначення проблеми

На початковому етапі розроблення стартап-проєкту ключовим завданням є формування чіткої ідеї. Ідея полягає у створенні інтелектуальної системи прогнозування динаміки продажів у супермаркеті на основі алгоритмів машинного навчання. Основна цінність такого продукту полягає у здатності передбачити об’єм продажів товарів у майбутньому з метою оптимізації процесів закупівлі, управління запасами, логістики та маркетингових стратегій.

Сучасна роздрібна торгівля стикається з рядом проблем, серед яких надмірні або недостатні запаси товарів, неефективне планування постачань, втрати через списання продукції з обмеженим терміном придатності, низький рівень персоналізованого обслуговування клієнтів. Ці проблеми загострюються в умовах швидко змінюваних споживчих уподобань та сезонних коливань попиту. Застарілі методи прогнозування, такі як ручне планування або лінійна екстраполяція часто не здатні враховувати складні нелінійні взаємозв’язки між різними факторами, що впливають на продажі.

Ідея полягає в розробці системи, яка за допомогою сучасних алгоритмів машинного навчання буде аналізувати історичні дані про продажі, маркетингову активність, сезонність, погодні умови, свята та інші фактори з метою побудови точного прогнозу майбутнього попиту на товари. Очікується, що така система дозволить підвищити ефективність бізнесу, зменшити витрати на зберігання, уникати дефіциту продукції.

Проблема, на вирішення якої спрямовано даний стартап, полягає у відсутності у супермаркетів точного, гнучкого та адаптивного інструменту для прогнозування продажів, який би враховував складну багатофакторну природу цього процесу. Запропоноване рішення має на меті заповнити цю прогалину шляхом використання сучасних інформаційних технологій, що дозволяють автоматизувати та інтелектуалізувати процеси управління торгівельною діяльністю.

5.3. Аналіз ринку та цільової аудиторії

Аналіз ринку є важливим етапом для обґрунтування доцільності запуску стартапу у сфері прогнозування продажів супермаркету за допомогою алгоритмів машинного навчання. Необхідно дослідити сучасні тенденції у сфері роздрібної торгівлі, потреби бізнесу у прогнозуванні об’ємів продажу, наявні рішення на ринку, ступінь їх ефективності та впровадженості в українських та міжнародних торговельних мережах.

За останні роки спостерігається стабільне зростання інтересу до використання аналітики даних і штучного інтелекту в управлінні роздрібною торгівлею. Великі торгові мережі, такі як АТБ, Сільпо, Метро вже застосовують внутрішні аналітичні системи, однак більшість середніх та малих супермаркетів не мають достатніх ресурсів або компетенцій для розроблення та впровадження таких рішень. Ринок для адаптивної та доступної системи прогнозування продажів залишається широким та незаповненим.

Цільовою аудиторією стартапу є малі та середні підприємства роздрібної торгівлі – супермаркети, магазини з локальними мережами, які прагнуть оптимізувати свої запаси, уникнути дефіциту товарів або надлишкових закупівель. Основна проблема, з якою стикається ця аудиторія, це відсутність ефективного інструменту прогнозування, який би враховував сезонність, акції, поведінку споживачів, макроекономічні фактори. Крім того, великою перевагою для таких підприємств буде можливість інтеграції прогнозної системи в існуючі облікові системи.

Слід виокремити сегмент користувачів, які мають базові навички роботи з аналітичними інструментами, але прагнуть вийти на новий рівень автоматизації. Для них система повинна бути інтуїтивно зрозумілою, мати простий інтерфейс та не вимагати глибоких знань у програмуванні чи машинному навчанні.

На ринку існує значний попит на ефективне, доступне та адаптоване рішення для прогнозування продажів. Стартап має потенціал задовольнити цю потребу, зайняти унікальну нішу та забезпечити конкурентну перевагу за рахунок використання сучасних технологій, масштабованості та орієнтації на конкретні бізнес-потреби клієнтів.

Окрему увагу слід приділити аналізу конкурентного середовища. У сфері прогнозування продажів для роздрібної торгівлі вже існують рішення від таких компаній, як SAP, IBM, Oracle, а також численні стартапи, які пропонують аналітичні платформи на основі штучного інтелекту. Проте більшість із них орієнтовані на великі корпорації з розвиненою IT-інфраструктурою. Запропонований стартап має на меті забезпечити доступне, інтуїтивно зрозуміле рішення для малого та середнього бізнесу з акцентом на український ринок і локальні особливості торгівлі. Це дозволяє зайняти нішу, що залишається малозаповненою.

У процесі дослідження потреб цільової аудиторії було виявлено, що для малого бізнесу ключовими є такі вимоги до системи: простота в налаштуванні та інтеграції, мінімальні вимоги до ресурсів, адаптивність до різних форматів вхідних даних, можливість швидкого отримання візуалізованих прогнозів і рекомендацій. Розробка повинна враховувати ці аспекти в архітектурі системи та в інтерфейсі користувача.

З врахуванням цього можна сформулювати персони користувачів – узагальнені образи представників цільової аудиторії. Це може бути власник невеликого магазину одягу, який бажає передбачити об’єми замовлень на наступний місяць; керівник мережі мінімаркетів, який хоче оптимізувати запаси на складах; маркетолог, що аналізує сезонні тренди продажів. Для кожного з цих типів користувачів система має надавати інструменти та рекомендації.

5.4. Формування бізнес-моделі

Формування ефективної бізнес-моделі є одним з ключових етапів створення стартапу, оскільки саме вона визначає спосіб монетизації продукту, взаємодію з клієнтами, канали продажу, структуру витрат і отримання прибутку. Бізнес-модель відповідає на питання яким чином стартап буде створювати цінність для споживачів і водночас забезпечувати сталий дохід.

Основою побудови бізнес-моделі у проєкті прогнозування динаміки продажів супермаркету за допомогою алгоритмів машинного навчання є концепція Business Model Canvas, що включає такі блоки (рис. 5.1).

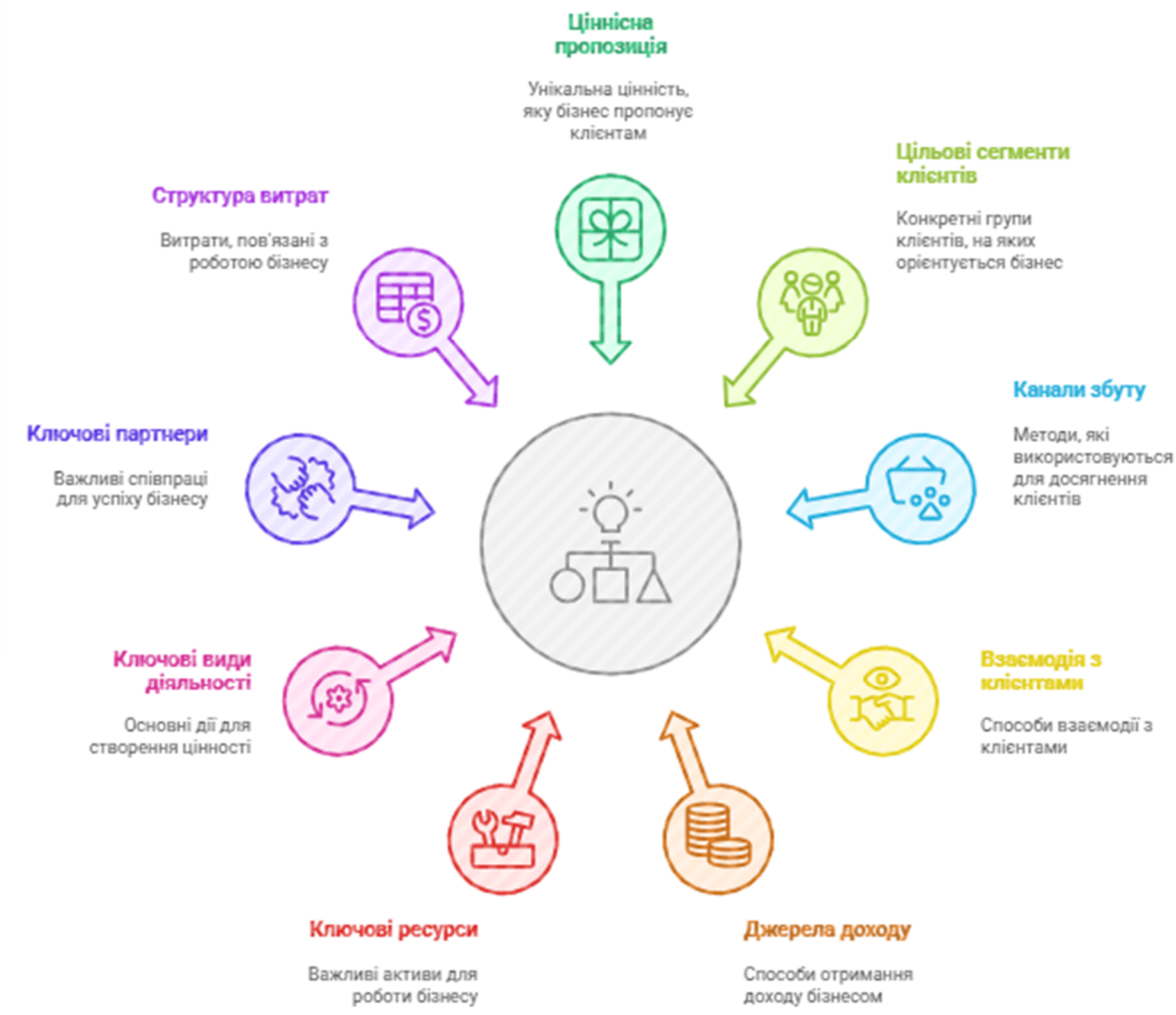

Рисунок 5.1 – Структура бізнес–моделі для прогнозування продажів

Ціннісна пропозиція. Головна цінність стартапу полягає у створенні інтелектуальної системи, яка дозволяє автоматизовано прогнозувати об’єми продажів у супермаркетах. Це дає змогу оптимізувати запаси, мінімізувати

надлишки, скоротити втрати, покращити логістику та прийняття управлінських рішень. Система забезпечує аналітику в реальному часі, інтеграцію з існуючими системами, візуалізацію даних.

Цільові сегменти клієнтів. До основних цільових клієнтів належать:

- мережі супермаркетів та продуктових магазинів;
- постачальники логістичних послуг у сфері роздрібної торгівлі;
- фінансові аналітики, які працюють з роздрібними даними;
- компанії, що надають рішення для автоматизації бізнесу.

Канали збуту. Продаж і просування продукту передбачає використання таких каналів: онлайн-платформа з демонстрацією можливостей системи, цільова реклама через соціальні мережі та професійні форуми, участь у галузевих виставках і конференціях.

Взаємодія з клієнтами. Формується система підтримки клієнтів через персональні консультації та технічну допомогу, онлайн-чат і гарячу лінію, навчальні вебінари та документацію, система знижок для постійних користувачів.

Джерела доходу. Стартап може отримувати прибуток через продаж ліцензій на програмний продукт, SaaS-модель із щомісячною оплатою залежно від об’єму використання, додаткові послуги (інтеграція, підтримка), аналітичні звіти на замовлення.

Ключові ресурси. До основних ресурсів проєкту належать команда розробників та дата-аналітиків, серверні потужності для зберігання та обробки даних, алгоритми машинного навчання та програмна інфраструктура, дані для навчання та тестування моделей.

Ключові види діяльності. Основними видами діяльності є розробка, тестування та вдосконалення моделі прогнозування, розробка програмного інтерфейсу, маркетинг і просування продукту, підтримка клієнтів.

Ключові партнери. До партнерів можуть входити постачальники великих наборів даних, інтегратори програмного забезпечення, компанії, що займаються логістикою та постачанням.

Структура витрат. Витрати поділяються на початкові інвестиції в розробку продукту, витрати на сервери та хмарні обчислення, зарплати команди, рекламна кампанія та маркетингові активності, юридичне оформлення бізнесу та підтримка прав інтелектуальної власності.

5.5. Маркетинг і просування

Успіх стартап-проєкту значною мірою залежить від ефективної маркетингової стратегії, яка охоплює комплекс заходів щодо залучення уваги потенційних клієнтів, формування позитивного іміджу продукту та стимулювання продажів. На етапі маркетингу визначається не лише аудиторія просування, але й канали комунікації, контентна стратегія, бюджетування рекламних кампаній та інструменти аналітики для оцінки ефективності.

Одним із першочергових завдань є формування бренду, який чітко передає ціннісну пропозицію стартапу. Це передбачає створення унікальної назви, логотипу, слогану, візуального стилю. Усі елементи мають бути узгоджені з очікуваннями та потребами цільової аудиторії. Далі розробляється маркетингова стратегія, яка визначає канали просування. Основними з них для стартапу можуть бути:

- цифровий маркетинг: SEO-оптимізація сайту, ведення блогу, email-розсилки, таргетинг у соціальних мережах (Facebook, Instagram, LinkedIn), контекстна реклама;
- контент-маркетинг: створення інформативного та візуально привабливого контенту, це відеоогляди, інфографіка, підкасти, які демонструють користь продукту;
- маркетинг у соціальних мережах: формування спільноти навколо продукту, залучення підписників, регулярна публікація новин, кейсів, відгуків;
- розсилка прес-релізів, публікації в медіа, участь у тематичних конференціях.

Крім онлайн-просування важливим є офлайн-маркетинг: участь у виставках, проведення презентацій, організація воркшопів, налагодження партнерських програм із зацікавленими організаціями. Окрему увагу приділяють ретаргетингу – поверненню користувачів, які вже взаємодіяли з продуктом, за допомогою

персоналізованих оголошень. Для оцінки ефективності маркетингової діяльності застосовуються такі метрики (рис. 5.1):

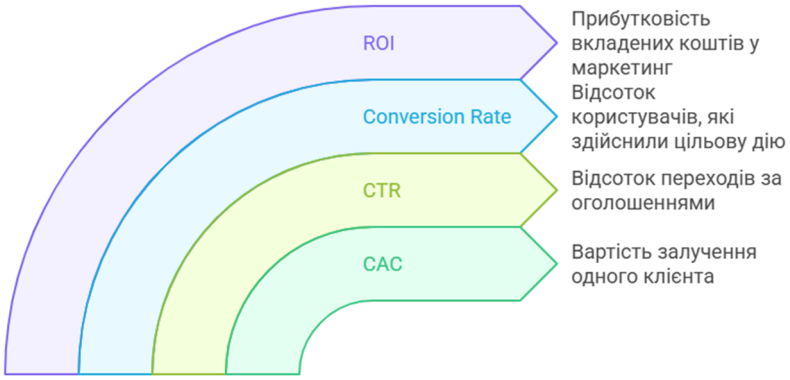

Рисунок 5.2 – Ключові показники ефективності маркетингу

Маркетинг і просування у стартап-проєкті це не разова дія, а постійний цикл аналізу, експериментів і вдосконалення, який спрямований на досягнення стійкого зростання користувацької бази та комерційного успіху.

ВИСНОВКИ ДО РОЗДІЛУ

У розділі 5 описано структуру проєкту системи прогнозування продажів. Визначено основні цілі та задачі, що сприяють досягненню успішної реалізації стартапу. Ознайомлено з архітектурою програмного забезпечення, включаючи вибір мов програмування та технологій. Підкреслено важливість відповідності системи сучасним вимогам машинного навчання та обробки даних. Обгрунтовано вибір алгоритмів для прогнозування динаміки продажів на основі аналізу ринку та історичних даних.

Описано процес розробки прототипу системи з урахуванням функціональних та нефункціональних вимог. Визначено етапи тестування та вдосконалення програмного продукту для досягнення високої якості. Наведено опис етапів впровадження та інтеграції системи у реальні бізнес-процеси. Враховано необхідність адаптації системи до змін зовнішніх факторів та характеристик клієнтської бази. Підкреслено важливість залучення цільової аудиторії для отримання зворотного зв’язку і покращення продукту. Визначено шлях до монетизації продукту та стратегії його просування на ринку. Вказано на перспективи подальшого розвитку та масштабування системи у майбутньому.

ВИСНОВКИ

У дипломній роботі проведено дослідження теми …

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Fildes R., Ma S., Kolassa S. Retail forecasting: research and practice. International journal of forecasting, 38(4), 2022. P. 1283-1318.

2. Ramachandran K. Predicting supermarket sales with big data analytics. International journal of data analytics, 3(1), 2023. P. 12-21.

3. Zhao Y., Xu L., Li Y. Machine learning–based sales forecasting in retail. Computer science & information technologies, 30, 2022. P. 39-48.

4. Hyndman R., Athanasopoulos G. Forecasting: principles and practice. Melbourne: OTexts, 2018. P. 112-131.

5. Choi T., Wallace S., Wang Y. Big data analytics in operations management. Production and operations management, 27(10) 2018. P. 1868-1889.

6. Makridakis S., Spiliotis E., Assimakopoulos V. Statistical and machine learning forecasting methods: Concerns and ways forward. International journal of forecasting, 34(4), 2018. P. 777-795.

7. Carbonneau R., Laframboise K., Vahidov R. Application of machine learning techniques for supply chain demand forecasting. Expert systems with applications, 34(3), 2008. P. 816-827.

8. Kaggle. Walmart Recruiting – Store Sales Forecasting. 2014. https://www.kaggle.com/competitions/walmart-recruiting-store-sales-forecasting.

9. Bandara K., Bergmeir C., Smyl S. Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach. International Journal of Forecasting, 37(1), 2020. P. 202-219.

10. Bose R., Mahapatra R. Machine learning algorithms for forecasting sales: comparative performance evaluation. Computer science and information technologies, 30, 2022. P. 27-41.

11. Geron A. Hands–on Machine learning with Scikit–Learn, Keras, and TensorFlow O'Reilly Media. 2019. P. 88-94.

12. Choi H., Kim J., Lee Y. Demand forecasting with LSTM neural networks: Applications to daily sales data. Neural computing and applications, 31(10), 2018. P. 1529-1542.

13. Hochreiter S., Schmidhuber J. Long Short–Term memory. Neural computation, 9(8), 1997. P. 63-65.

14. Brownlee J. Deep learning for time series forecasting. Machine learning mastery. 2018. P. 75-112.

15. Makridakis S., Spiliotis E., Assimakopoulos V. The M4 competition: Results, findings, conclusion and way forward. International journal of forecasting, 36(1), 2020. P. 305-372.

16. Kuhn M., Johnson K. Applied Predictive Modeling. Springer. 2013. P. 214-216.

ДОДАТКИ

ДОДАТОК А

1.ipynb

1. Імпорт бібліотек

```
import pandas as pd
import numpy as np
from datetime import datetime
import warnings
warnings.filterwarnings('ignore')

# Для моделей
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import xgboost as xgb

# Для Prophet
from prophet import Prophet

# Для візуалізації
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('seaborn-v0_8')

2.
```

2. Завантаження та підготовка даних

```
# Завантаження адаптованого датасету
df = pd.read_csv("ukr_supermarket_sales_2020_2024.csv", parse_dates=['date'])

# Перевірка
print("Розмір даних:", df.shape)
print("Період:", df['date'].min(), "—", df['date'].max())
```

3. Feature Engineering

```
# Створення агрегованих даних: продажі по днях (для Prophet і загального
прогнозу)
daily_sales = df.groupby('date').agg(
    total_sales=('sales_uah', 'sum'),
```

```
        avg_price=('price_uah', 'mean'),
        num_transactions=('item_id', 'count'),
        num_categories=('category', 'nunique'),
        num_stores=('city', 'nunique'),
        holiday_flag=('is_holiday', 'max')
    ).reset_index()

    # Для XGBoost: додаткові ознаки
    daily_sales['day_of_week'] = daily_sales['date'].dt.dayofweek
    daily_sales['month'] = daily_sales['date'].dt.month
    daily_sales['year'] = daily_sales['date'].dt.year
    daily_sales['day_of_year'] = daily_sales['date'].dt.dayofyear

    # Циклічне кодування для кращої роботи з сезонністю
    daily_sales['day_of_week_sin'] = np.sin(2 * np.pi * daily_sales['day_of_week'] /
7)
    daily_sales['day_of_week_cos'] = np.cos(2 * np.pi * daily_sales['day_of_week'] /
7)
    daily_sales['month_sin'] = np.sin(2 * np.pi * daily_sales['month'] / 12)
    daily_sales['month_cos'] = np.cos(2 * np.pi * daily_sales['month'] / 12)

    # Розділення на тренувальні та тестові дані (останні 60 днів — тест)
    split_date = daily_sales['date'].max() - pd.Timedelta(days=60)
    train = daily_sales[daily_sales['date'] <= split_date]
    test = daily_sales[daily_sales['date'] > split_date]

    print(f"Тренувальний період: {train['date'].min()} — {train['date'].max()}")
    print(f"Тестовий період: {test['date'].min()} — {test['date'].max()}")
```

4. Навчання моделі XGBoost

```
    # Вибір ознак
    features = [
        'avg_price', 'num_transactions', 'num_categories', 'num_stores',
        'holiday_flag', 'day_of_week_sin', 'day_of_week_cos',
        'month_sin', 'month_cos', 'day_of_year'
    ]

    X_train = train[features]
    y_train = train['total_sales']
    X_test = test[features]
    y_test = test['total_sales']
```

```
# Навчання XGBoost
xgb_model = xgb.XGBRegressor(
    n_estimators=500,
    max_depth=6,
    learning_rate=0.05,
    random_state=42,
    early_stopping_rounds=50,
    eval_metric='rmse'
)

xgb_model.fit(X_train, y_train, eval_set=[(X_test, y_test)], verbose=False)

# Прогноз
y_pred_xgb = xgb_model.predict(X_test)
```

5. Навчання моделі Prophet

```
# Підготовка даних для Prophet
prophet_df = daily_sales[['date', 'total_sales']].rename(columns={'date': 'ds',
'total_sales': 'y'})

# Розділення
prophet_train = prophet_df[prophet_df['ds'] <= split_date]
prophet_test = prophet_df[prophet_df['ds'] > split_date]

# Навчання
prophet_model = Prophet(
    yearly_seasonality=True,
    weekly_seasonality=True,
    daily_seasonality=False,
    holidays=pd.DataFrame({
        'holiday': 'ua_holiday',
        'ds': daily_sales[daily_sales['holiday_flag'] == 1]['date'],
        'lower_window': 0,
        'upper_window': 1,
    }).drop_duplicates()
)
prophet_model.fit(prophet_train)

# Прогноз на тестовий період
future = prophet_model.make_future_dataframe(periods=60)
forecast = prophet_model.predict(future)
```

```
    y_pred_prophet =
forecast[forecast['ds'].isin(prophet_test['ds'])]['yhat'].values
```

6. Оцінка метрик

```
    def calculate_metrics(y_true, y_pred, model_name):
        mae = mean_absolute_error(y_true, y_pred)
        rmse = np.sqrt(mean_squared_error(y_true, y_pred))
        mape = np.mean(np.abs((y_true - y_pred) / y_true)) * 100
        r2 = r2_score(y_true, y_pred)
        return {
            'Model': model_name,
            'MAE': round(mae, 2),
            'RMSE': round(rmse, 2),
            'MAPE (%)': round(mape, 2),
            'R²': round(r2, 4)
        }

    metrics = [
        calculate_metrics(y_test, y_pred_xgb, 'XGBoost'),
        calculate_metrics(y_test, y_pred_prophet, 'Prophet')
    ]

    results_df = pd.DataFrame(metrics)
    print("\n📊 Порівняння моделей:")
    print(results_df.to_string(index=False))
```

7. Візуалізація прогнозів

```
    plt.figure(figsize=(14, 6))
    plt.plot(test['date'], y_test, label='Фактичні продажі', color='black',
linewidth=2)
    plt.plot(test['date'], y_pred_xgb, label='XGBoost', linestyle='--', marker='o',
alpha=0.8)
    plt.plot(test['date'], y_pred_prophet, label='Prophet', linestyle='-.',
marker='x', alpha=0.8)
    plt.title('Прогноз динаміки продажів (останні 60 днів)')
    plt.xlabel('Дата')
    plt.ylabel('Продажі (UAH)')
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
```

```
plt.savefig('forecast_comparison.png', dpi=300)
plt.show()
```

✅ 8. Збереження результатів (опціонально)

```
# Збереження моделі та прогнозу
import joblib
joblib.dump(xgb_model, 'xgb_sales_model.pkl')
test_results = test[['date', 'total_sales']].copy()
test_results['XGBoost_pred'] = y_pred_xgb
test_results['Prophet_pred'] = y_pred_prophet
test_results.to_csv('forecast_results.csv', index=False, encoding='utf-8-sig')
```

2.py

```
import streamlit as st
import pandas as pd
import numpy as np
import joblib
from datetime import datetime, timedelta
import plotly.graph_objects as go
from plotly.subplots import make_subplots

# Завантаження моделі та даних
@st.cache_resource
def load_model_and_data():
    model = joblib.load('xgb_sales_model.pkl')
    df = pd.read_csv('ukr_supermarket_sales_2020_2024.csv',
parse_dates=['date'])
    return model, df

model, df = load_model_and_data()

# Унікальні значення для вибору
cities = sorted(df['city'].dropna().unique())
categories = sorted(df['category'].unique())

# Заголовок
st.title("📈 Прогнозування динаміки продажів супермаркету")
st.markdown("Оберіть місто, категорію товару та горизонт прогнозу для отримання
прогнозу.")

# Вибір параметрів
col1, col2, col3 = st.columns(3)
with col1:
    selected_city = st.selectbox("Місто", cities)
with col2:
    selected_category = st.selectbox("Категорія товару", categories)
with col3:
    forecast_days = st.slider("Горизонт прогнозу (днів)", min_value=7,
max_value=90, value=30)

# Фільтрація даних за містом і категорією
filtered_df = df[
    ((df['city'] == selected_city) | (selected_city == "Усі")) &
```

```
        (df['category'] == selected_category)
    ].copy()

    if filtered_df.empty:
        st.warning("Немає даних для обраної комбінації міста та категорії.")
    else:
        # Агрегація до щоденних продажів
        daily = filtered_df.groupby('date').agg(
            total_sales=('sales_uah', 'sum'),
            avg_price=('price_uah', 'mean'),
            num_transactions=('item_id', 'count'),
            num_categories=('category', 'nunique'),
            num_stores=('city', 'nunique')
        ).reset_index()

        # Додавання святкового прапорця (спрощено — використовуємо наявний)
        daily = daily.merge(df[['date', 'is_holiday']].drop_duplicates(), on='date',
how='left')
        daily['is_holiday'] = daily['is_holiday'].fillna(0).astype(int)

        # Додавання часових ознак
        daily['day_of_week'] = daily['date'].dt.dayofweek
        daily['month'] = daily['date'].dt.month
        daily['year'] = daily['date'].dt.year
        daily['day_of_year'] = daily['date'].dt.dayofyear
        daily['day_of_week_sin'] = np.sin(2 * np.pi * daily['day_of_week'] / 7)
        daily['day_of_week_cos'] = np.cos(2 * np.pi * daily['day_of_week'] / 7)
        daily['month_sin'] = np.sin(2 * np.pi * daily['month'] / 12)
        daily['month_cos'] = np.cos(2 * np.pi * daily['month'] / 12)

        # Підготовка ознак
        feature_cols = [
            'avg_price', 'num_transactions', 'num_categories', 'num_stores',
            'is_holiday', 'day_of_week_sin', 'day_of_week_cos',
            'month_sin', 'month_cos', 'day_of_year'
        ]

        # Заповнення пропущених значень (на випадок)
        for col in feature_cols:
            if col not in daily.columns:
                daily[col] = 0
            daily[col] = daily[col].fillna(daily[col].median() if
daily[col].median() is not None else 0)
```

```
        # Прогноз для історичних даних (для візуалізації)
        X_hist = daily[feature_cols]
        daily['predicted_sales'] = model.predict(X_hist)

        # Генерація майбутніх дат
        last_date = daily['date'].max()
        future_dates = [last_date + timedelta(days=i) for i in range(1,
forecast_days + 1)]
        future_df = pd.DataFrame({'date': future_dates})

        # Додавання часових ознак для майбутніх дат
        future_df['day_of_week'] = future_df['date'].dt.dayofweek
        future_df['month'] = future_df['date'].dt.month
        future_df['year'] = future_df['date'].dt.year
        future_df['day_of_year'] = future_df['date'].dt.dayofyear
        future_df['is_holiday'] = 0  # можна розширити за потреби
        future_df['num_stores'] = 1
        future_df['num_categories'] = 1
        future_df['num_transactions'] = daily['num_transactions'].median()
        future_df['avg_price'] = daily['avg_price'].median()

        future_df['day_of_week_sin'] = np.sin(2 * np.pi * future_df['day_of_week'] /
7)
        future_df['day_of_week_cos'] = np.cos(2 * np.pi * future_df['day_of_week'] /
7)
        future_df['month_sin'] = np.sin(2 * np.pi * future_df['month'] / 12)
        future_df['month_cos'] = np.cos(2 * np.pi * future_df['month'] / 12)

        X_future = future_df[feature_cols]
        future_df['predicted_sales'] = model.predict(X_future)

        # Об'єднання історії та прогнозу
        full_df = pd.concat([
            daily[['date', 'total_sales', 'predicted_sales']],
            future_df[['date',
'predicted_sales']].rename(columns={'predicted_sales': 'total_sales'})
        ], ignore_index=True)

        # Візуалізація
        fig = go.Figure()
        fig.add_trace(go.Scatter(
            x=daily['date'], y=daily['total_sales'],
```

```
            mode='lines', name='Фактичні продажі', line=dict(color='black', width=2)
        ))
        fig.add_trace(go.Scatter(
            x=daily['date'], y=daily['predicted_sales'],
            mode='lines', name='Прогноз (історія)', line=dict(dash='dot',
color='green')
        ))
        fig.add_trace(go.Scatter(
            x=future_df['date'], y=future_df['predicted_sales'],
            mode='lines+markers', name='Прогноз (майбутнє)', line=dict(color='red',
width=2)
        ))

        fig.update_layout(
            title=f"Прогноз продажів: {selected_city} – {selected_category}",
            xaxis_title="Дата",
            yaxis_title="Продажі (UAH)",
            hovermode="x unified"
        )

        st.plotly_chart(fig, use_container_width=True)

        # Числові метрики (на основі історичних даних)
        from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score
        y_true = daily['total_sales']
        y_pred = daily['predicted_sales']
        mae = mean_absolute_error(y_true, y_pred)
        rmse = np.sqrt(mean_squared_error(y_true, y_pred))
        r2 = r2_score(y_true, y_pred)

        st.subheader("📊 Метрики якості моделі (на історичних даних)")
        col1, col2, col3 = st.columns(3)
        col1.metric("MAE (UAH)", f"{mae:,.0f}")
        col2.metric("RMSE (UAH)", f"{rmse:,.0f}")
        col3.metric("R²", f"{r2:.3f}")

        # Таблиця прогнозу
        st.subheader("🛍️ Прогноз на наступні дні")
        st.dataframe(future_df[['date', 'predicted_sales']].rename(
            columns={'date': 'Дата', 'predicted_sales': 'Прогноз (UAH)'}
        ).style.format({'Прогноз (UAH)': '{:,.0f}'}))
```