МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) автоматизації і комп'ютерних систем імені проф.І.В.Ельперіна

Кафедра інформаційних технологій, штучного інтелекту і кібербезпеки

«До захисту в ЕК»	**«До захисту допущено»**
Директор інституту(декан факультету)	Завідувач кафедри
_____ Андрій ФОРСЮК	_____ Сергій ГРИБКОВ
(підпис) (ім'я та прізвище)	(підпис) (ім'я та прізвище)
«08» грудня 2025р.	«08» грудня 2025р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

зі спеціальності 122 Комп'ютерні науки
(код та назва спеціальності)

освітньо-професійної програми Управління інформацією та аналітика даних

на тему: Інтелектуальна система дослідження впливу інформаційно-новинного фону на котирування акцій компаній

Виконав: здобувач 2 курсу, групи КН-2-3М

Русин Данило Костянтинович _____
(прізвище, ім'я, по батькові повністю) (підпис)

Керівник Костіков Микола Павлович _____
(прізвище , ім'я та по батькові повністю) (підпис)

Консультанти _____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

Рецензент _____ _____
(ім'я та прізвище) (підпис)

Я як здобувач(ка) Національного університету харчових технологій розумію і підтримую політику університету з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволеної допомоги під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Здобувач _____
(підпис)

Київ – 2025р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) автоматизації і комп'ютерних систем імені проф. І.В. Ельперіна
Кафедра інформаційних технологій, штучного інтелекту і кібербезпеки
Освітній ступінь магістр
Спеціальність 122 Комп'ютерні науки
(код і назва)
Освітньо-професійна програма Управління інформацією та аналітика даних
(назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри інформаційних технологій, штучного інтелекту і кібербезпеки

_____ Сергій ГРИБКОВ
«05» листопада 2025 року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА

_____
(прізвище, ім'я, по батькові)

1. Тема роботи Інформаційно-аналітична система прогнозування продажів торговельної мережі
_____
_____
керівник роботи Костіков Микола Павлович , доцент, кандидат технічних наук _____,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від 05 листопада 2025 року №906-кс
2. Строк подання здобувачем роботи 01 грудня 2025 року
3. Вихідні дані до роботи _____
Набір історичних даних продажів _____
Часові ряди продажів _____
Методичні та алгоритмічні основи _____
_____
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити) _____
Розділ 1. Дослідження предметної області та постановка задачі _____
Розділ 2. Дослідження та обґрунтування методів і алгоритмів прогнозування ___
Розділ 3. Реалізація та апробація інформаційно-аналітичної системи прогнозування продажів _____
_____
5. Перелік графічного матеріалу

Діаграми моделювання та архітектури
Графіки результатів прогнозування та візуалізації
Табличний матеріал

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Костіков М. П. доц., к.т.н	07.10.2025	15.10.2025
2	Костіков М. П. доц., к.т.н	20.10.2025	03.11.2025
3	Костіков М. П. доц., к.т.н	11.11.2025	25.11.2025

7. Дата видачі завдання: 01 жовтня 2025 року

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз предметної області та постановка задачі	06.10.2025	Виконано
2	Обґрунтування методології дослідження	12.10.2025	Виконано
3	Моделювання бізнес-процесів	15.10.2025	Виконано
4	Програмна реалізація математичних алгоритмів	20.10.2025	Виконано
5	Розробка архітектури та інтерфейсу системи	03.11.2025	Виконано
6	Експериментальне тестування	11.11.2025	Виконано
7	Порівняльний аналіз результатів	18.11.2025	Виконано
8	Формулювання висновків	19.11.2025	Виконано
9	Підготовка пояснювальної записки, презентації	23.11.2025	Виконано

Здобувач _____ Данило РУСИН
(підпис) (ім'я та прізвище)

Керівник роботи _____ Микола КОСТІКОВ
(підпис) (ім'я та прізвище)

АНОТАЦІЯ

Русин Данило Костянтинович – Інформаційно-аналітична система прогнозування продажів торговельної мережі.

Робота присвячена розробці та дослідженню інформаційно-аналітичної системи для автоматизації прогнозування продажів у роздрібній торгівлі. Об’єктом дослідження є процес прогнозування продажів на основі аналізу часових рядів. У роботі проаналізовано та програмно реалізовано чотири статистичні методи: класична декомпозиція, декомпозиція з лінійним трендом, метод ковзних середніх та метод експоненційного згладжування . Практичним результатом є створення веб-додатку «Sales Forecaster» на базі мови Python та фреймворку Streamlit. Система дозволяє завантажувати історичні дані, автоматично розраховувати коефіцієнти сезонності, генерувати прогнози та проводити порівняльний аналіз точності моделей за метриками MAE, RMSE та MAPE.

Ключові слова: ПРОГНОЗУВАННЯ ПРОДАЖІВ, ЧАСОВІ РЯДИ, СЕЗОННІСТЬ, ДЕКОМПОЗИЦІЯ, ЕКПОНЕНЦІЙНЕ ЗГЛАДЖУВАННЯ, PYTHON, STREAMLIT, WEB-ДОДАТОК.

ABSTRACT

Rusyn Danylo Kostiantynovych – Information and analytical system for sales forecasting of a retail network.

The thesis is devoted to the development and research of an information-analytical system for automating sales forecasting in retail. The object of the study is the process of sales forecasting based on time series analysis. Four statistical methods were analyzed and implemented in the software: classical decomposition, decomposition with linear trend, moving averages method, and exponential smoothing method . The practical result is the creation of the "Sales Forecaster" web application based on the Python language and the Streamlit framework. The system allows uploading historical data, automatically calculating seasonality coefficients, generating forecasts, and conducting a comparative analysis of model accuracy using MAE, RMSE, and MAPE metrics.

Keywords: SALES FORECASTING, TIME SERIES, SEASONALITY, DECOMPOSITION, EXPONENTIAL SMOOTHING, PYTHON, STREAMLIT, WEB APPLICATION.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

BPMN (Business Process Model and Notation) — система умовних позначень (нотація) для моделювання бізнес-процесів.

CSV (Comma-Separated Values) — текстовий формат файлів для зберігання табличних даних, у якому значення розділяються комами.

ETL (Extract, Transform, Load) — процес вилучення, перетворення та завантаження даних з джерела у сховище.

IDEF0 — методологія функціонального моделювання та графічна нотація для опису бізнес-процесів.

MAE (Mean Absolute Error) — середня абсолютна похибка прогнозу.

MAPE (Mean Absolute Percentage Error) — середня абсолютна відсоткова похибка.

MVC (Model-View-Controller) — архітектурний патерн, що розділяє систему на три компоненти: модель даних, інтерфейс користувача та логіку управління.

Python — високорівнева мова програмування загального призначення.

RMSE (Root Mean Square Error) — корінь середньоквадратичної похибки.

SaaS (Software as a Service) — модель надання програмного забезпечення як послуги через інтернет.

Sales Forecaster — назва розробленого веб-додатку для прогнозування продажів.

SME (Small and Medium-sized Enterprises) — малі та середні підприємства.

Streamlit — бібліотека з відкритим кодом для створення веб-додатків мовою Python, орієнтована на задачі Data Science.

Time Series (часовий ряд) — послідовність значень показника, зафіксованих у рівновіддалені моменти часу.

Total Error — загальна сумарна похибка прогнозу.

UI (User Interface) — користувацький інтерфейс.

UX (User Experience) — користувацький досвід взаємодії з програмним продуктом.

Декомпозиція — метод розкладання часового ряду на окремі компоненти (тренд, сезонність, випадкову складову).

Експоненційне згладжування — метод прогнозування, у якому більша вага надається більш сучасним спостереженням.

Інформаційно-аналітична система — програмно-апаратний комплекс, призначений для збору, обробки, аналізу та візуалізації даних.

Коефіцієнти сезонності — числові показники, що відображають відхилення значень часового ряду у певні періоди від середнього рівня.

Метод ковзних середніх — спосіб згладжування часових рядів шляхом обчислення середніх значень на рухомих інтервалах.

Прогнозування (Forecasting) — процес оцінювання майбутніх значень показника на основі історичних даних.

Сезонність (Seasonality) — регулярні періодичні коливання показників, що повторюються через фіксовані проміжки часу.

Тренд (Trend) — довгострокова спрямованість зміни рівня показника у часовому ряді.

ВСТУП

В умовах цифровізації бізнесу та зростання конкуренції на ринку роздрібної торгівлі особливої актуальності набувають питання ефективного планування продажів та управління товарними запасами. Торговельні мережі працюють з великими обсягами даних, що формуються у вигляді часових рядів продажів, які мають виражену сезонну складову. Неврахування сезонних коливань призводить до перевиробництва, дефіциту товарів та фінансових втрат.

Сучасні інформаційно-аналітичні системи дозволяють автоматизувати процеси збору, обробки та аналізу даних. Особливу роль у таких системах відіграють методи прогнозування, що базуються на аналізі часових рядів. Однак практичне застосування різних методів розрахунку сезонності показує, що їх точність та стабільність можуть суттєво відрізнятися залежно від характеристик вихідних даних.

У зв’язку з цим актуальним є створення інформаційно-аналітичної системи, яка дозволяє не лише будувати прогноз продажів, але й порівнювати ефективність різних методів розрахунку сезонності, що дає змогу обґрунтовано обирати найбільш точні математичні моделі для практичного використання у торговельних мережах.

Метою кваліфікаційної роботи є розробка інформаційно-аналітичної системи прогнозування продажів торговельної мережі та оцінка ефективності і точності методів розрахунку сезонності на основі реальних даних продажів.

Для досягнення поставленої мети в роботі необхідно вирішити такі завдання:

- проаналізувати особливості формування сезонності у продажах торговельних мереж;
- дослідити існуючі методи декомпозиції часових рядів;
- реалізувати основні методи розрахунку сезонності засобами мови програмування Python;
- розробити програмний продукт для автоматизованого розрахунку сезонних коефіцієнтів;
- реалізувати алгоритми побудови прогнозів на основі отриманих коефіцієнтів;

- провести експериментальну оцінку точності прогнозів;
- порівняти результати роботи різних методів та сформулювати практичні рекомендації.

Об’єктом дослідження є процес прогнозування продажів торговельної мережі на основі аналізу часових рядів.

Предметом дослідження є методи розрахунку сезонності та алгоритми прогнозування продажів, що використовуються в інформаційно-аналітичних системах.

У роботі використовуються такі методи дослідження:

- методи аналізу часових рядів;
- класична декомпозиція без урахування тренду;
- декомпозиція з використанням лінійної регресії;
- метод ковзних середніх;
- метод експоненційного згладжування;
- методи статистичного аналізу похибок (MAE, MAPE, RMSE);
- методи програмної реалізації та тестування.

Наукова новизна роботи полягає у вдосконаленні процесу прогнозування продажів шляхом інтеграції процедур попередньої обробки даних, розрахунку коефіцієнтів сезонності та валідації результатів у єдиний програмний контур, що дозволяє автоматизувати процес прийняття рішень щодо закупівель.

Практичне значення отриманих результатів полягає у можливості використання розробленої інформаційно-аналітичної системи для автоматизації процесів прогнозування продажів, оптимізації управління товарними запасами та підвищення економічної ефективності діяльності торговельних підприємств.

Усі етапи дослідження, зокрема аналіз літературних джерел, розробка алгоритмів, програмна реалізація та експериментальне тестування системи, виконані автором самостійно.

Результати роботи апробовано шляхом експериментального застосування розробленого програмного забезпечення на реальних даних продажів торговельної мережі.

За матеріалами кваліфікаційної роботи опубліковано тези доповіді на тему «Виявлення та аналіз сезонності продажів торговельних мереж» на XII Міжнародній науково-технічній інтернет-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами» (Київ, НУХТ, 27 листопада 2025 р.)

Кваліфікаційна робота виконувалась згідно із планом науково-дослідних робіт кафедри інформаційних технологій, штучного інтелекту і кібербезпеки Національного університету харчових технологій: НДР «Дослідження та використання сучасних інформаційних технологій для виконання функцій та завдань виробничого і організаційного управління підприємств харчової галузі» № ДР 0120U105386, 2020–2025 рр; «Дослідження та впровадження сучасних методів аналізу даних у харчову промисловість» № ДР 0125U003889, 2025–2031 рр.

Кваліфікаційна робота складається з вступу, трьох розділів, висновків, списку використаних джерел та додатків. Загальний обсяг роботи становить 87 сторінка, у тому числі 16 рисунків, 6 таблиць та 5 додатків.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Аналіз ринку та потенційних користувачів системи

Сучасний стан світової економіки характеризується стрімкою цифровізацією ритейлу та зростанням частки електронної комерції (e-commerce). В умовах глобальної невизначеності та високої волатильності ринків, здатність точно прогнозувати попит стає критичним фактором конкурентоспроможності торговельних підприємств. Ринок програмного забезпечення для бізнес-аналітики (Business Intelligence, BI) демонструє стабільне зростання, однак існує суттєвий дисбаланс у пропозиції інструментів прогнозування.

Основним сегментом потенційних користувачів розробленої системи є підприємства малого та середнього бізнесу (SME — Small and Medium-sized Enterprises), що здійснюють торговельну діяльність як на локальних, так і на міжнародних маркетплейсах. На відміну від великих корпорацій, які мають ресурси для впровадження дороговартісних ERP-систем (Enterprise Resource Planning) та утримання штату аналітиків даних, представники сегменту SME стикаються з проблемою відсутності доступних та зрозумілих інструментів для планування продажів.

Потенційними користувачами системи є:

1. Категорійні менеджери торговельних мереж, які потребують оперативного розрахунку обсягів закупівель для запобігання дефіциту товару (out-of-stock).
2. Власники інтернет-магазинів, для яких критично важливим є управління обіговими коштами та оптимізація складських запасів.
3. Аналітики-початківці, яким необхідний інструмент для швидкої валідації гіпотез щодо сезонності товарів без необхідності написання програмного коду.

Незважаючи на популярність складних методів машинного навчання (Machine Learning), для значної частини цільової аудиторії вони є надлишковими

та економічно неефективними через високу вартість впровадження та непрозорість результатів ("black box problem").

Запропонована інформаційно-аналітична система, що базується на методах класичної декомпозиції, лінійної регресії, ковзних середніх та експоненційного згладжування, вирішує низку специфічних проблем користувачів:

1. Проблема «холодного старту» та обмеженості даних. Багато підприємств малого бізнесу не мають накопиченої історії продажів за 5–10 років, яка необхідна для навчання складних нейромереж (LSTM, ARIMA). Обрані методи (зокрема, метод ковзних середніх та експоненційного згладжування) дозволяють отримувати адекватні прогнозні моделі навіть на коротких часових рядах (від 12 до 24 місяців), що значно знижує бар'єр входу для користувачів.
2. Інтерпретованість та довіра до результатів. Для прийняття фінансових рішень менеджерам важливо розуміти логіку формування прогнозу. Методи декомпозиції з трендом та лінійної регресії надають прозору візуалізацію складових процесу (тренд, сезонність), що дозволяє економічно обґрунтувати план закупівель перед керівництвом або інвесторами.
3. Економічна ефективність (ROI). Впровадження системи дозволяє оптимізувати рівень страхових запасів. Зменшення помилки прогнозу навіть на кілька відсоткових пунктів за допомогою врахування сезонності дозволяє вивільнити значні обсяги обігових коштів, які раніше були "заморожені" у неліквідних товарах, та уникнути втраченої вигоди у пікові періоди продажів.

Розроблений продукт позиціонується як міжнародний (International SaaS Solution), оскільки математичні моделі сезонності та трендів є універсальними і не залежать від мови чи локального законодавства:

- Використання алгоритмів аналізу часових рядів дозволяє системі автоматично адаптуватися до будь-яких регіональних особливостей (наприклад, різниця у датах початку сезонів у Північній та Південній півкулях або специфічні національні свята), оскільки система виявляє патерни безпосередньо з даних, а не базується на жорстко заданих правилах.

- Це відкриває можливості для глобального масштабування продукту без необхідності суттєвих змін у програмній архітектурі, що робить його привабливим для широкого кола міжнародних користувачів, які працюють на платформах Amazon, eBay, Shopify та інших.

Таким чином, розробка системи, що поєднує перевірені часом статистичні методи з сучасним веб-інтерфейсом, є актуальним завданням, що задовольняє наявний попит на ринку доступних аналітичних рішень.

1.2. Особливості продажів у торговельних мережах

Торговельні мережі характеризуються значними обсягами товарообігу та постійною динамікою попиту. Продажі формуються під впливом багатьох чинників, серед яких ключову роль відіграють сезонні, економічні, маркетингові та соціальні фактори. Для більшості товарних категорій характерні періодичні коливання попиту, що повторюються з певною регулярністю.

Сезонність у продажах може проявлятися на різних часових масштабах:

- річна сезонність - зростання або зниження попиту в окремі пори року (наприклад, святкові періоди, літній та зимовий сезони);
- місячна сезонність - циклічні зміни обсягів продажів упродовж року;
- тижнева сезонність - коливання попиту в межах тижня (вихідні та робочі дні).

Наявність сезонності суттєво ускладнює процес управління запасами та планування закупівель, оскільки використання простих середніх значень не дозволяє адекватно врахувати зміну попиту. Саме тому торговельні мережі активно впроваджують інформаційно-аналітичні системи, що дозволяють аналізувати історичні дані та будувати обґрунтовані прогнози.

1.3. Функціональне моделювання та аналіз існуючих бізнес-процесів

Впровадження інформаційно-аналітичної системи «Sales Forecaster» у діяльність торговельної мережі передбачає трансформацію існуючих бізнес-процесів управління запасами. Система виступає ключовим інструментом підтримки прийняття рішень для відділу закупівель та маркетингу.

1.3.1. Сценарії використання системи

Основними користувачами системи є категорійні менеджери та аналітики відділу продажу. Виділено два ключові сценарії використання програмного продукту:

1. Оперативне планування закупівель (щотижневе/щомісячне):
    - Мета*:* Формування замовлення постачальникам для поповнення поточних запасів.
    - Дія*:* Користувач завантажує дані за останні періоди, обирає метод експоненційного згладжування або ковзних середніх (для короткострокового прогнозу) та отримує розрахункову кількість товару, необхідну на наступний місяць.
    - Результат*:* Зниження ризику «out-of-stock» (відсутності товару на полиці).
2. Стратегічне планування сезонних активностей (щоквартальне/річне):
    - Мета*:* Підготовка до пікових періодів (наприклад, «Чорна п'ятниця», новорічні свята) та бюджетування.
    - Дія*:* Користувач використовує метод декомпозиції з трендом для аналізу річних патернів, визначає місяці з найвищими коефіцієнтами сезонності.
    - Результат*:* Оптимізація обігових коштів (не заморожувати гроші в товарі у «тихі» місяці) та завчасне бронювання обсягів у постачальників перед високим сезоном.

1.3.2. Опис функціональної моделі (IDEF0)

Для формалізації роботи системи розроблено функціональну модель у нотації IDEF0. Контекстна діаграма (Рівень А-0) відображає процес «Прогнозування продажів торговельної мережі» (рис. 1.1). Діаграма також наведена у додатку А у повному розмірі.

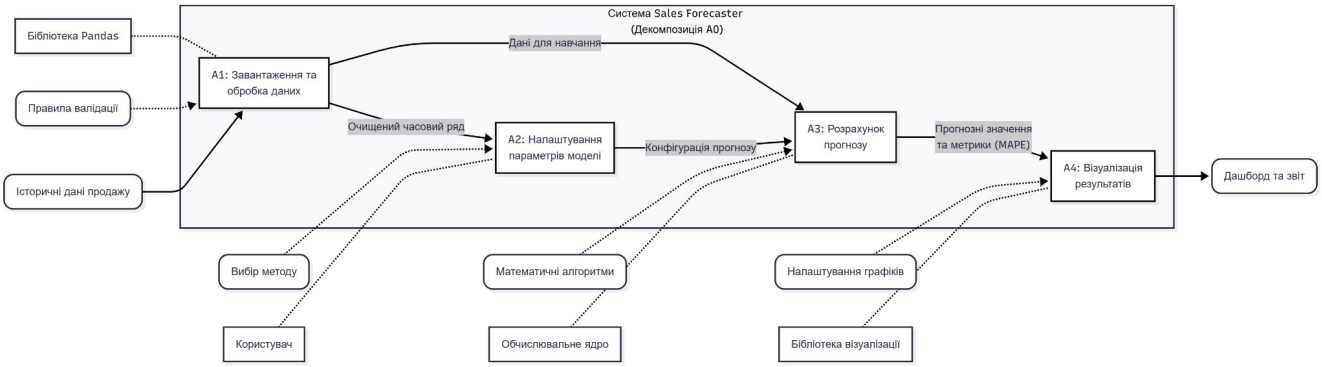

Рисунок 1.1 – Контекстна діаграма процесу «Прогнозування продажів торговельної мережі»

Опис стрілок та блоків діаграми:

- Вхід (Input): Історичні дані про продажі (CSV-файли, вивантаження з ERP), перелік товарних категорій, календарні дані.
- Управління (Control): Бізнес-правила торговельної мережі, маркетинговий календар (акції), бюджетні обмеження, обрана методологія прогнозування (налаштування коефіцієнтів згладжування).
- Механізми (Mechanism): Інформаційно-аналітична система «Sales Forecaster», категорійний менеджер (особа, що приймає рішення), серверні потужності.
- Вихід (Output): Прогноз обсягів продажів, графіки сезонності, аналітичні звіти для формування замовлень.

Декомпозиція контекстної діаграми наведена у Додатку Б і включає підпроцеси: завантаження та валідація даних, вибір математичної моделі, розрахунок коефіцієнтів, візуалізація результатів.

1.3.3. Аналіз документообігу

Взаємодія системи з іншими підрозділами супроводжується обміном інформаційними потоками. Опис документообігу наведено в Таблиці 1.1.

Таблиця 1.1 — Схема документообігу в межах процесу прогнозування

Назва документу / Даних	**Джерело (Від кого)**	**Отримувач (Кому)**	**Частота**	**Формат**	**Призначення**
Звіт про фактичні продажі	ERP-система / Бухгалтерія	Категорійний менеджер (Система)	Щотижня	.CSV / .XLSX	Вхідні дані для розрахунку прогнозу
Прогнозний план продажів	Система «Sales Forecaster»	Відділ закупівель	Щомісяця	Дашборд / PDF	Основа для формування замовлення постачальнику
Аналіз сезонності	Система «Sales Forecaster»	Відділ маркетингу	Щоквартально	Графіки	Планування акцій у періоди спаду попиту
Заявка на закупівлю	Відділ закупівель	Постачальник	За потребою	Електронний документ	Фінальний документ на основі прогнозу

1.3.4. Моделювання бізнес-процесу відділу закупівель (BPMN)

Інтеграція системи «Sales Forecaster» змінює класичний алгоритм роботи менеджера із закупівель. Нижче наведено схему удосконаленого бізнес-процесу (рис. 1.2) та її опис.

Рисунок 1.2 – Діаграма бізнес-процесу використання системи

Представлена діаграма відображає послідовність дій категорійного менеджера при формуванні замовлення. Процес оптимізовано наступним чином:

1. Ініціалізація та підготовка даних: Менеджер отримує актуальні дані про продажі з облікової системи (ERP) та завантажує їх у веб-інтерфейс системи.
2. Автоматизоване моделювання (System Activity): На відміну від ручного підходу, система автоматично виконує паралельні розрахунки:
   - Здійснюється прогнозування методами: класичної декомпозиції, лінійної регресії з сезонністю, ковзних середніх та експоненційного згладжування .
   - Система автоматично порівнює отримані результати з фактичними даними минулого періоду (backtesting) та розраховує метрики помилок (MAE, RMSE, MAPE) .
3. Аналітичний вибір (Human-in-the-loop): Менеджер аналізує візуалізований дашборд, де відображено порівняння всіх методів. Його завдання — не налаштовувати формули, а обрати той метод, який найкраще апроксимує поведінку продажів конкретного товару (має найменший показник MAPE або найкраще відтворює пікові сезонні значення).
4. Прийняття рішення та закупівля: Обраний прогноз стає основою для формування заявки на закупівлю. Після фінансового погодження процес завершується відправкою замовлення постачальнику.

Такий підхід мінімізує ризик вибору невідповідної математичної моделі, оскільки рішення приймається на основі емпіричного порівняння точності алгоритмів на реальних даних.

1.4. Аналіз задач прогнозування в інформаційно-аналітичних системах

Прогнозування продажів є однією з ключових функцій сучасних інформаційно-аналітичних систем. Воно використовується для вирішення таких практичних завдань:

- планування товарних запасів;

- оптимізація логістичних процесів;
- управління фінансовими потоками;
- планування маркетингових кампаній.

Процес прогнозування базується на аналізі часових рядів, які представляють собою послідовність значень показників у часі. Для торговельних мереж такими даними є обсяги продажів у розрізі товарних категорій, магазинів або регіонів.

Основними складовими часового ряду є:

- тренд - довгострокова тенденція зміни значень показника;
- сезонна компонента - регулярні періодичні коливання;
- випадкова компонента - нерегулярні відхилення, зумовлені випадковими факторами.

Ефективне прогнозування можливе лише за умови правильного виокремлення та моделювання кожної з цих складових.

1.5. Огляд існуючих підходів до прогнозування продажів

У практиці прогнозування продажів використовуються різні підходи та методи, які умовно поділяють на кілька груп.

1.5.1. Статистичні методи прогнозування

До цієї групи належать методи, що базуються на математичній статистиці, серед яких:

- методи середніх значень;
- методи екстраполяції трендів;
- методи регресійного аналізу.

Перевагою статистичних методів є їхня відносна простота реалізації та прозорість інтерпретації результатів.

1.5.2. Методи аналізу часових рядів

До цієї групи належать:

- методи декомпозиції часових рядів;
- методи ковзних середніх;
- методи експоненційного згладжування.

Ці методи дозволяють явно описувати структуру часового ряду та окремо моделювати тренд і сезонність.

1.5.3. Інтелектуальні методи прогнозування

Окрім класичних статистичних методів, у сучасних системах використовуються машинне навчання та нейронні мережі. Проте такі методи потребують значних обсягів даних та складніших обчислювальних ресурсів, що не завжди є доцільним для практичного застосування у торговельних мережах середнього масштабу.

У межах даної роботи основна увага приділяється класичним методам аналізу часових рядів, які поєднують достатню точність і простоту практичної реалізації.

1.6. Формування вимог до інформаційно-аналітичної системи

Інформаційно-аналітична система прогнозування продажів повинна забезпечувати виконання таких функцій:

- завантаження історичних даних з файлів або баз даних;
- попередню обробку даних (очищення, агрегація, нормалізація);
- розрахунок сезонних коефіцієнтів кількома методами;
- побудову прогнозів на основі кожного методу;
- оцінку точності прогнозів за допомогою статистичних показників;
- візуалізацію результатів у вигляді графіків та таблиць.

Система повинна бути модульною, масштабованою та зручною для подальшого розширення.

1.7. Постановка задачі дослідження

На основі проведеного аналізу формулюється задача дослідження: розробити інформаційно-аналітичну систему, яка реалізує різні методи розрахунку сезонності та дозволяє на практичних даних продажів торговельної мережі оцінити їх ефективність і точність.

Для розв'язання поставленої задачі необхідно:

1. Реалізувати алгоритми таких методів:

- класична декомпозиція без урахування тренду;
- декомпозиція з трендом, побудованим методом лінійної регресії;
- метод ковзних середніх;
- метод експоненційного згладжування.

2. Побудувати програмну архітектуру системи.
3. Реалізувати механізми порівняння прогнозів із фактичними даними.
4. Розрахувати показники похибок та формувати аналітичні висновки.

1.8. Висновки до розділу 1

У першому розділі виконано комплексний аналіз предметної області прогнозування продажів у торговельних мережах, що дозволило сформувати теоретико-методологічний базис для подальшої розробки системи.

1. Аналіз ринку та потреб користувачів показав, що існує значний попит на доступні інструменти прогнозування серед підприємств малого та середнього бізнесу (SME). Визначено, що для цього сегменту критично важливими є вирішення проблеми «холодного старту» (робота з обмеженою історією даних) та інтерпретованість результатів. Це обґрунтовує вибір класичних статистичних методів (декомпозиція, експоненційне згладжування) як економічно доцільної альтернативи складним алгоритмам Machine Learning.
2. Дослідження особливостей продажів підтвердило ключову роль сезонної та трендової компонент у формуванні попиту. Встановлено, що ігнорування цих факторів призводить до фінансових втрат через дефіцит товарів або їх надлишок на складах.
3. Функціональне моделювання (IDEF0) дозволило формалізувати процес прогнозування та декомпозувати його на етапи: від попередньої обробки даних до візуалізації результатів. Розроблена схема документообігу визначила інформаційні потоки між системою, відділом закупівель та маркетингу.
4. Моделювання бізнес-процесів (BPMN) продемонструвало, як впровадження системи трансформує роботу менеджера із закупівель. Запропоновано

оптимізований сценарій, де рутинні розрахунки автоматизуються, а функція людини зміщується у площину аналітики та прийняття рішень на основі порівняння точності (MAPE) різних математичних моделей.

5. Сформульовано вимоги до інформаційно-аналітичної системи та здійснено постановку задачі дослідження. Визначено, що система повинна реалізовувати стратегію паралельного прогнозування декількома методами для забезпечення можливості вибору найточнішого алгоритму для кожної товарної категорії.

Отримані результати є основою для алгоритмічної та програмної реалізації інформаційно-аналітичної системи «Sales Forecaster» у наступних розділах роботи.

РОЗДІЛ 2. ДОСЛІДЖЕННЯ ТА ОБҐРУНТУВАННЯ МЕТОДІВ І АЛГОРИТМІВ ПРОГНОЗУВАННЯ

2.1. Аналіз структури часових рядів продажів

Часові ряди продажів торговельної мережі являють собою впорядковану послідовність значень обсягів реалізації товарів у дискретні моменти часу. Для дослідження сезонності в роботі використовується мультиплікативна модель часового ряду, яка описується рівнянням (2ю1):

$$Y_t = T_t \cdot S_t \cdot E_t \tag{2.1}$$

де:

- Y_t - фактичний обсяг продажів у момент часу t;
- T_t- трендова компонента;
- S_t - сезонна компонента;
- E_t - випадкова (нерегулярна) складова.

В рамках даного дослідження основна увага приділяється методам оцінювання трендової та сезонної складових без залучення складних стохастичних моделей, що дозволяє забезпечити стабільну роботу алгоритмів при обмежених обсягах даних.

2.2. Обґрунтування вибору методів оцінювання сезонної компоненти для аналітичного дослідження

Процес вибору методів розрахунку сезонності є ключовим етапом побудови системи прогнозування, оскільки правильність визначення сезонної компоненти безпосередньо впливає на точність підсумкових прогнозних значень. У межах даного дослідження основна увага приділяється методам, які поєднують достатню точність результатів із помірною складністю реалізації та можливістю практичного застосування в бізнес-аналітичних системах.

Вибір конкретних методів зумовлений особливостями досліджуваних часових рядів, а саме: наявністю сезонних коливань із періодичністю 12 місяців, можливими структурними зрушеннями в динаміці продажів, а також потребою у стабільних оцінках при відносно невеликій історичній вибірці даних.

Для проведення порівняльного аналізу в роботі обрано чотири підходи до розрахунку сезонності:

- класичну декомпозицію без явного врахування тренду;
- декомпозицію з попереднім виділенням трендової складової методом лінійної регресії;
- метод ковзних середніх;
- метод експоненційного згладжування з сезонною компонентою.

Зазначені методи належать до групи детермінованих та напів стохастичних підходів до аналізу часових рядів, що робить їх придатними для реалізації в прикладних інформаційних системах без залучення складних обчислювальних ресурсів.

Крім того, обрані методи дозволяють оцінити вплив різних підходів до обробки тренду та сезонності на точність прогнозування, що створює підґрунтя для подальшого експериментального порівняння результатів та вибору найбільш ефективного алгоритму.

Таким чином, обґрунтований вибір методів забезпечує методологічну цілісність дослідження та дозволяє сформувати надійну основу для практичної реалізації алгоритмів прогнозування, що розглядається у наступних підрозділах.

2.3. Метод 1 — Класична декомпозиція без урахування тренду

Метод класичної декомпозиції без явного виділення тренду є фундаментальним підходом в аналізі часових рядів, який базується на припущенні про стаціонарність середнього рівня продажів протягом досліджуваного періоду. У прикладному сенсі цей метод найчастіше використовується для аналізу товарних груп зі стабільним попитом (наприклад, товари повсякденного вжитку, хлібобулочні вироби, соціальні групи товарів), де відсутні різкі довгострокові зміни або структурні зрушення ринку.

В інформаційно-аналітичних системах цей алгоритм часто виконує роль "базової лінії" (baseline model). Його основна перевага — обчислювальна простота та висока інтерпретованість: отримані коефіцієнти показують, у скільки разів продажі в конкретний місяць відрізняються від середньорічних. Це дозволяє категорійним менеджерам швидко виявляти сезонні піки без необхідності налаштування складних параметрів моделі. Однак, за наявності сильного тренду росту або падіння продажів, цей метод може давати зміщені оцінки, що потребує застосування більш складних підходів.

2.3.1 Теоретичні основи

Класична декомпозиція без урахування тренду передбачає, що часовий ряд немає вираженого тренду або його вплив є незначним. У цьому випадку модель спрощується до виду (2.2):

$$Y_t = S_t \cdot E_t \tag{2.2}$$

Сезонні коефіцієнти визначаються як середні значення продажів для кожної сезонної позиції (наприклад, місяця) віднесені до загального середнього значення (2.3):

$$S_m = \frac{\underline{Y}_m}{\underline{Y}} \tag{2.3}$$

де:

- S_m — сезонний коефіцієнт для місяця m;
- $\underline{Y}_m$ — середній обсяг продажів у місяці m;
- $\underline{Y}$ — загальне середнє значення ряду.

2.3.2. Алгоритм реалізації

Процес обчислення сезонних коефіцієнтів методом класичної декомпозиції можна представити у вигляді послідовності кроків з агрегації та усереднення даних:

- Обчислення загального середнього значення продажів.

- Групування даних за місяцями.
- Розрахунок середніх значень для кожного місяця.
- Обчислення сезонних коефіцієнтів шляхом ділення місячних середніх на загальне середнє.

2.3.3. Реалізація мовою Python (опис)

Програмна реалізація методу виконана з використанням бібліотеки pandas для маніпуляцій з табличними даними. Це дозволяє використовувати векторизовані операції замість повільних циклів, що критично важливо для веб-додатку.

Повний програмний код методу наведено у додатку В.

2.3.4. Результат симуляції

Для кожного з методів були розраховані коефіцієнти сезонності в двох часових розрізах: для місяців та для тижнів. Це дозволило окремо оцінити ефективність кожного підходу для різних рівнів деталізації часових рядів.

На рисунку 2.1 представлено графічне порівняння результатів прогнозування, отриманих за допомогою методу декомпозиції (Decomposition), із фактичними обсягами продажів (Actual Value) за 2020 рік. Вихідні дані про продажі були отримані з відкритого джерела Kaggle та детально охарактеризовані в третьому розділі роботи.

Рисунок 2.1 – Порівняння прогнозу методом декомпозиції з фактичними продажами в розрізі місяців

Аналіз фактичних значень демонструє наявність вираженої сезонної компоненти, що проявляється у поступовому зростанні обсягів продажів у літній період, їх зниженні восени та різкому підвищенні напередодні новорічних свят.

Застосування класичного методу декомпозиції дозволило відтворити загальний характер сезонних коливань, однак призвело до систематичного заниження прогнозних значень. Графічне порівняння наочно підтверджує відсутність адекватного врахування трендової складової у моделі. Виявлене явище недопрогнозування має критичний характер і в умовах практичного використання могло б призвести до негативних управлінських рішень та суттєвих економічних втрат.

Прогноз обсягів продажів у тижневому розрізі містить 52 точки спостереження, на відміну від 12 точок у місячній агрегації (рис. 2.2), що забезпечує більш детальний аналіз ефектів застосованого методу прогнозування.

Рисунок 2.2 – Порівняння прогнозу методом декомпозиції з фактичними продажами в розрізі тижнів

На графіку за допомогою зеленого прямокутника виділено область максимального річного пікового значення продажів, що є закономірним у період перед різдвяних свят. Метод класичної декомпозиції продемонстрував високу здатність до відтворення даного сезонного піку, ґрунтуючись на історичних даних за попередні три роки.

Водночас виявлено суттєві розбіжності між прогнозними та фактичними значеннями у літній та осінній періоди. Застосування отриманих прогнозних результатів у практичній діяльності підприємства могло б призвести до зниження рівня доступності товарів у торговельній мережі та потенційних втрат обсягів продажів.

2.4. Метод 2 — Декомпозиція з трендом, побудованим за допомогою лінійної регресії

Метод декомпозиції з використанням лінійного тренду є еволюційним розвитком класичного підходу, спрямованим на усунення його основного недоліку - нездатності враховувати довгострокові зміни середнього рівня ряду (не стаціонарність). Якщо класична декомпозиція ефективна лише для стабільних ринків, то даний метод моделює загальну динаміку розвитку (зростання або спад) за допомогою лінійної функції $y = ax + b$, на яку накладаються сезонні коливання.

У прикладному бізнес-аналізі цей підхід є "золотим стандартом" для прогнозування продажів товарів, що мають чітко виражену тенденцію до зміни попиту. Типовими сценаріями використання є:

- Виведення нового товару на ринок: Коли продажі зростають з місяця в місяць не лише через сезон, а й завдяки маркетингу та збільшенню клієнтської бази.
- Аналіз категорій на стадії спаду: Наприклад, падіння попиту на застарілу техніку, де важливо відрізнити сезонний сплеск (наприклад, перед Новим роком) від загального незворотного тренду падіння інтересу.
- Планування бюджету в умовах інфляції: Коли вартісні показники продажів зростають лінійно через підняття цін, навіть якщо фізичний обсяг продажів стабільний.

Використання лінійної регресії дозволяє "очистити" дані від впливу глобального росту/падіння перед розрахунком сезонності, що робить сезонні коефіцієнти більш точними та стійкими у довгостроковій перспективі.

2.4.1. Теоретичні основи

У цьому методі використовується класична модель (2.4):

$$Y_t = T_t \cdot S_t \cdot E_t \quad (2.4)$$

де трендова складова апроксимується лінійною функцією (2.5):

$$T_t = a + bt \quad (2.5)$$

де:

- a - вільний член;
- b - коефіцієнт нахилу прямої;
- t - порядковий номер періоду.

Для оцінювання параметрів a і b використовується метод найменших квадратів.

Після обчислення тренду виконують детрендування ряду (2.6):

$$Y'_t = \frac{Y_t}{T_t} \quad (2.6)$$

Сезонні коефіцієнти визначаються як середні значення Y'_t для кожного періоду сезону.

2.4.2. Алгоритм реалізації

- Формування порядкового індексу часу t.
- Розрахунок коефіцієнтів регресійної прямої.
- Обчислення значень тренду T_t для кожного періоду.
- Детрендування ряду.
- Обчислення сезонних коефіцієнтів.

2.4.3. Реалізація мовою Python (опис)

Програмна реалізація даного методу інкапсульована у функції calculate_trend_linear та базується на використанні бібліотек pandas для обробки даних та numpy для математичних обчислень.

Алгоритм реалізації складається з трьох послідовних етапів:

1. Агрегація даних: Вихідний датасет фільтрується за навчальний період, після чого відбувається групування продажів за роками для кожної товарної категорії. Це дозволяє перейти від детальних транзакційних даних до річних точок, необхідних для побудови тренду.
2. Розрахунок коефіцієнтів регресії: Для кожної категорії обчислюються параметри лінійної функції y = ax + b. У даній реалізації застосовано прямий розрахунок коефіцієнта нахилу (slope) та вільного члена (intercept) через формули методу найменших квадратів. Це забезпечує вищу швидкодію порівняно з ініціалізацією важковагових об'єктів ML-бібліотек для кожної ітерації.
3. Визначення тренд-фактору: На основі отриманого рівняння розраховується прогнозне значення продажів на майбутній рік. Ключовим результатом роботи функції є trend_factor - коефіцієнт, що показує відношення прогнозованого обсягу до фактичного обсягу останнього відомого року.

Важливою особливістю реалізації є механізм запобігання аномальним прогнозам. У коді передбачено примусове обмеження (capping) коефіцієнта тренду в діапазоні від 0.5 до 2.0. Це означає, що система не дозволить спрогнозувати падіння продажів більш ніж у 2 рази або їх зростання більш ніж у 2 рази за один рік, що є запобіжником від мультиплікації помилок на "брудних" даних.

Повний програмний код методу наведено у додатку В.

2.4.4. Результати симуляції

На рисунку 2.3 наведено порівняння фактичних щомісячних обсягів продажів та прогнозних значень, отриманих за допомогою методу декомпозиції з лінійним трендом. Аналіз місячної агрегації дозволяє оцінити здатність моделі відображати довші сезонні цикли та загальну тенденцію зміни попиту.

Рисунок 2.3 – Порівняння прогнозу методом декомпозиції з лінійним трендом і фактичними продажами в розрізі місяців.

Графік демонструє, що прогнозна крива загалом зберігає правильний напрямок руху та відтворює основні сезони зростання і спаду. Найбільш суттєві розбіжності між прогнозом і фактичними даними спостерігаються в літній період, що виділено червоною рамкою. У цьому сегменті модель переоцінює обсяги продажів, не повністю враховуючи тимчасове зниження попиту.

У кінці року прогноз достатньо точно відображає зростання продажів, наближене до фактичних значень, що свідчить про здатність моделі коректно ідентифікувати довгострокові сезонні піки. Загалом метод декомпозиції з лінійним трендом демонструє кращі результати на місячному рівні порівняно з тижневим, однак все ще має обмеження щодо відтворення нестабільних сезонних коливань.

На рисунку 2.4 представлено порівняння фактичних тижневих продажів (Actual Value) із прогнозом, побудованим методом декомпозиції з використанням лінійного тренду (Decomposition with Linear Trend). Використання тижневого агрегування (52 спостереження на рік) дозволяє детальніше проаналізувати поведінку попиту протягом року та виявити короткострокові коливання.

Рисунок 2.4 – Порівняння прогнозу методом декомпозиції з лінійним трендом з фактичними продажами в розрізі тижнів.

На графіку червоними рамками виділено періоди підвищеної волатильності та сезонних піків. Найбільш виражений пік спостерігається наприкінці року, що відповідає передсвятковому періоду. Прогнозна модель з лінійним трендом демонструє здатність відтворювати загальну форму сезонних хвиль і адекватно ідентифікує пікові значення попиту.

Водночас помітні систематичні розбіжності між прогнозними та фактичними значеннями в середині року, зокрема у весняно-літній та осінній періоди. У ці інтервали модель схильна до переоцінювання рівня продажів, що свідчить про обмежену гнучкість лінійного тренду щодо короткострокових змін ринкової динаміки. Незважаючи на загальну коректність відтворення структури сезонності, точність моделі на тижневому рівні є нестабільною.

2.5. Метод 3 — Метод ковзних середніх (розрахунок тренду і сезонності)

Метод ковзних середніх (Moving Average) є одним із найпоширеніших інструментів технічного аналізу та прогнозування, який використовується для згладжування часових рядів з метою виявлення довгострокових трендів. Його основна ідея полягає у фільтрації короткострокових коливань ("шуму") шляхом створення середнього значення для певного під набору даних, яке "ковзає" вздовж часової осі.

У прикладному бізнес-контексті цей метод є незамінним для:

- Очищення даних: Допомагає менеджерам побачити реальну картину продажів, прибираючи вплив випадкових факторів (наприклад, разові великі замовлення або тимчасові спади через погоду).
- Виявлення точок розвороту: Перетин графіків короткострокової та довгострокової ковзних середніх часто сигналізує про зміну ринкового тренду (початок росту або спаду).
- Розрахунку сезонності на "шумних" даних: Якщо продажі мають хаотичний характер, просте усереднення (як у класичній декомпозиції) може бути неточним. Ковзне середнє дозволяє спочатку м'яко виділити тренд, а потім розрахувати сезонні відхилення від цього згладженого тренду, що дає більш стійкі коефіцієнти.

2.5.1. Теоретичні основи

Метод ковзних середніх передбачає згладжування початкового ряду для отримання трендової компоненти. Тренд визначається за формулою (2.7):

$$T_t = \frac{1}{n}\sum_{i=t-k}^{t+k} Y_i \tag{2.7}$$

де:

- n - довжина вікна згладжування;
- k - половина ширини вікна.

Для парної сезонності використовується центрована ковзна середня.

Після визначення тренду сезонна компонента знаходиться як (2.8):

$$S_t = \frac{Y_t}{T_t} \tag{2.8}$$

2.5.2. Алгоритм реалізації

1. Визначення довжини вікна згладжування.
2. Розрахунок ковзної середньої для кожного періоду.
3. Центрування (за потреби).

4. Розрахунок сезонних коефіцієнтів.

2.5.3. Реалізація мовою Python (опис)

Програмна реалізація методу ковзних середніх у системі виконує подвійну функцію: вона використовується як для виділення тренду, так і для розрахунку специфічних сезонних коефіцієнтів. Логіка реалізована у функціях calculate_moving_average_seasonality та calculate_moving_average_trend.

Алгоритм обробки даних включає такі етапи:

1. Підготовка вікна згладжування: Система використовує параметр window (за замовчуванням 2 роки) для визначення ширини вікна усереднення. Це дозволяє згладити випадкові викиди, зберігаючи загальну форму кривої продажів.
2. Розрахунок сезонності: Спочатку дані групуються за роками та періодами (місяцями/тижнями). Для кожного періоду (наприклад, "березень") формується масив значень за всі роки. Замість простого середнього, алгоритм застосовує ковзне вікно вздовж цих значень, що дозволяє нівелювати вплив аномальних років (наприклад, карантинних обмежень).
3. Розрахунок тренду: Для визначення глобального тренду система агрегує дані до річних сум і застосовує ковзне середнє до цих значень. Коефіцієнт тренду визначається як відношення останнього згладженого значення до передостаннього (або до першого, залежно від довжини історії), що показує вектор руху продажів.

Повний програмний код методу наведено у додатку В.

2.5.4. Результати симуляції

Графічне порівняння результатів методу ковзних середніх з фактичними продажами торгівельної мережі виявило майже бездоганне слідування тренду продажів. Однак графік прогнозованих продажів є дещо вищим по осі Value. Для бізнесу це означало б дещо завищені запаси. Це однозначно краще ніж мати недостатній рівень запасів. На рисунку 2.5 також червоними прямокутниками

вказані періоди з найбільшим розходженням прогнозу та факту. найбільше відхилення приблизно дорівнює 300 000 реалів.

Рисунок 2.5 – Порівняння прогнозу методом ковзних середніх з фактичними продажами в розрізі місяців

Тижневий прогноз, наведений на рисунку 2.6, показує майже повне співпадіння з фактичними продажами. Найбільше відхилення складає приблизно 100 000 рупій, а коливання продажів є майже ідентичними до фактичних. За графічним аналізом метод ковзних середніх у розрізі тижнів поки що є найточнішим серед усіх досліджуваних.

Рисунок 2.6 – Порівняння прогнозу методом ковзних середніх з фактичними продажами в розрізі тижнів

2.6. Метод 4 — Метод експоненційного згладжування

Метод експоненційного згладжування (Exponential Smoothing) є одним із найбільш адаптивних підходів до прогнозування короткострокових та середньострокових трендів. Його фундаментальна відмінність від простих ковзних середніх полягає у механізмі зважування історичних даних: замість присвоєння однакової ваги всім спостереженням у вікні, цей метод надає експоненційно спадну вагу минулим значенням. Це означає, що останні події (наприклад, продажі минулого тижня) мають найбільший вплив на прогноз, тоді як дані річної давнини впливають на нього мінімально .

У прикладному бізнес-сенсі цей метод є критично важливим для товарів з високою волатильністю попиту або товарів, життєвий цикл яких швидко змінюється (модна індустрія, електроніка). Основні сценарії використання включають:

- Реагування на раптові зміни ринку: Якщо попит різко зріс через вірусний маркетинг, експоненційне згладжування "схопить" цей тренд швидше, ніж звичайне середнє.
- Прогнозування "свіжих" товарів: Коли історія продажів коротка, метод дозволяє будувати прогноз, спираючись переважно на останні доступні точки, ігноруючи відсутність глибокої історії.
- Коригування запасів (Replenishment): Використовується в логістичних системах для щотижневого оновлення замовлень, оскільки алгоритм потребує мінімальних обчислювальних ресурсів і зберігає лише останнє згладжене значення.

2.6.1. Теоретичні основи

Метод експоненційного згладжування дозволяє будувати оцінки тренду та сезонності з більшим впливом останніх спостережень. Базова формула має вигляд (2.9):

$$L_t = \alpha Y_t + (1 - \alpha) L_{t-1} \tag{2.9}$$

де:

- L_t — згладжене значення;

- α — коефіцієнт згладжування.

Для моделювання тренду використовується подвійне експоненційне згладжування (2.10):

$$T_t = \beta\,(L_t - L_{t-1}) + (1-\beta)\,T_{t-1} \tag{2.10}$$

Сезонна компонента визначається за допомогою коефіцієнта γ (2.11):

$$S_t = \gamma\frac{Y_t}{L_t} + (1+\gamma)S_{t-s} \tag{2.11}$$

де s — довжина сезонного періоду.

2.6.2. Алгоритм реалізації

1. Ініціалізація початкових значень L_0, T_0, S_0.
2. Ітеративний перерахунок компонент для всіх періодів.
3. Формування сезонних коефіцієнтів.

2.6.3. Реалізація мовою Python (опис)

Програмна реалізація методу в системі «Sales Forecaster» базується на ітеративному алгоритмі перерахунку згладжених значень. Логіка інкапсульована у функціях calculate_exponential_smoothing_seasonality та calculate_exponential_smoothing_trend.

Алгоритм реалізації має наступну структуру:

1. Ініціалізація: Як початкове згладжене значення (L_0) береться перше фактичне значення у часовому ряді.
2. Рекурсивне оновлення: Для кожного наступного періоду нове згладжене значення розраховується як зважена сума поточного фактичного значення та попереднього згладженого значення. Вага визначається параметром alpha (коефіцієнт згладжування).
3. Визначення тренду та сезонності:
    - Для сезонності алгоритм застосовує згладжування до значень конкретного періоду (наприклад, усіх січнів) крізь роки.

- Для тренду згладжування застосовується до річних сум продажів, що дозволяє визначити вектор розвитку категорії.

Параметр alpha (за замовчуванням встановлений на рівні 0.3) є змінним: вищі значення роблять модель чутливішою до останніх змін, нижчі — більш консервативною та стійкою до шуму.

Повний програмний код методу наведено у додатку В.

2.6.4. Результати симуляції

На рисунку 2.7 зафіксовано суттєве завищення прогнозних значень, отриманих методом експоненційного згладжування, у місячному розрізі. Візуальний аналіз графіка свідчить, що лише у двох часових періодах прогнозні значення наближалися до фактичних обсягів продажів. Водночас динаміка коливань у прогнозному та фактичному рядах характеризується подібною амплітудою. Таким чином, даний метод доцільно застосовувати для прогнозування загальної тенденції на високому рівні агрегації даних, однак його точність для прогнозування окремих часових точок залишається низькою.

Рисунок 2.7 — Порівняння прогнозу методом експоненційного згладжування з фактичними продажами в розрізі місяців

У тижневому розрізі метод експоненційного згладжування демонструє задовільні результати прогнозування (рис. 2.8). Прогнозна крива в цілому перевищує фактичні значення, що в умовах торговельного бізнесу може розглядатися як прийнятне відхилення. За винятком періоду новорічних свят,

максимальна різниця між прогнозними та фактичними значеннями не перевищує 100 000 рупій. Доцільним є проведення порівняльного аналізу величини похибки між методом ковзних середніх та методом експоненційного згладжування в тижневому розрізі з метою визначення більш точного інструменту прогнозування.

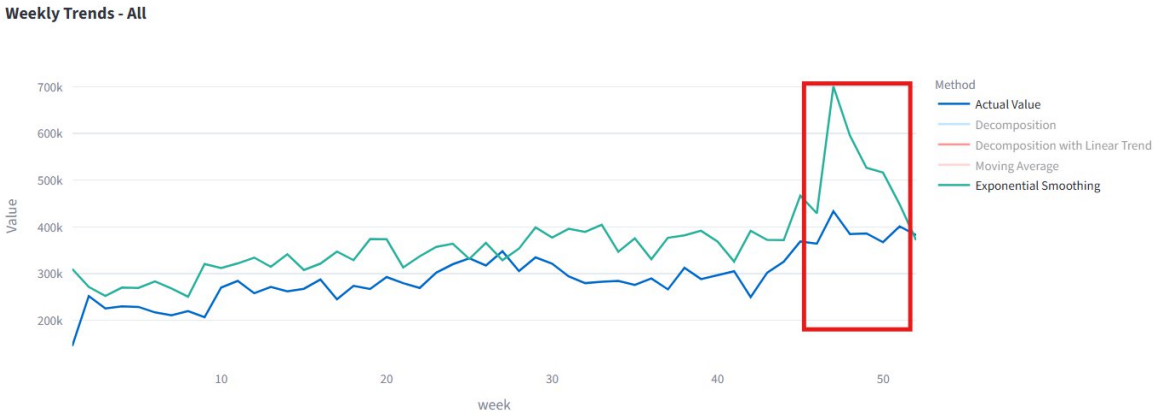

Рисунок 2.8 – Порівняння прогнозу методом експоненційного згладжування з фактичними продажами в розрізі тижнів

Водночас слід зазначити, що метод експоненційного згладжування виявив недостатню ефективність при прогнозуванні святкового попиту. Різкі пікові значення доцільно враховувати шляхом програмної корекції прогнозної моделі, зокрема через додаткове згладжування аномальних відхилень за допомогою коригувальних коефіцієнтів.

2.7. Огляд альтернативних методів

В ході виконання роботи було розглянуто методи, які базуються на складніших стохастичних процесах та алгоритмах машинного навчання.

2.7.1. SARIMA (Seasonal AutoRegressive Integrated Moving Average)

Метод належить до класу параметричних моделей стохастичних процесів. Він моделює майбутнє значення ряду як лінійну комбінацію його минулих значень (авторегресія — AR), минулих помилок прогнозу (ковзне середнє — MA) та інтегрування для досягнення стаціонарності (I).

Математична сутність: Модель описується параметрами$(p, d, q) \times (P, D, Q)_s$, де враховується сезонна та несезонна автокореляція. Процес передбачає аналіз корелограм (ACF/PACF) для ідентифікації порядку моделі.

Вимоги: Передбачає стаціонарність ряду (або приведення до неї через диференціювання) та відсутність структурних зсувів.

2.7.2. Адитивні регресійні моделі (на прикладі Facebook Prophet)

Цей підхід базується на класі узагальнених адитивних моделей (GAM — Generalized Additive Models). Часовий ряд розкладається на суму компонентів (2.12):

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t \qquad (2.12)$$

де:

- $g(t)$ - функція тренду (лінійна або логістична),
- $s(t)$ — сезонність, що апроксимується рядами Фур'є,
- $h(t)$ — ефекти свят,
- ϵ_t — помилка.

Особливість: Використання байєсівського підходу для підгонки параметрів дозволяє гнучко обробляти пропущені дані та зміни тренду (changepoints).

2.7.3. Ансамблеві методи градієнтного бустингу (XGBoost, LightGBM)

Це методи машинного навчання (Supervised Learning), що трансформують задачу часового ряду в задачу регресії на основі ознак (feature-based regression). Алгоритм ітеративно будує ансамбль слабких прогностів (дерев рішень), де кожне наступне дерево мінімізує функцію втрат (loss function) попереднього ансамблю.

Специфіка: Вимагає складного конструювання ознак (feature engineering) — створення лагових змінних (lags), ковзних вікон тощо, щоб модель могла врахувати часову залежність.

2.7.4. Рекурентні нейронні мережі (RNN/LSTM)

Архітектури глибокого навчання (Deep Learning), розроблені для роботи з послідовними даними. LSTM (Long Short-Term Memory) використовує механізм

"вентилів" (gates) для регулювання потоку інформації, що дозволяє вирішити проблему зникаючого градієнта та моделювати довгострокові нелінійні залежності.

Специфіка: Це універсальні апроксиматори функцій, які не потребують попередніх припущень про розподіл даних, але вимагають величезних обсягів навчальної вибірки для конвергенції ваг.

2.8. Порівняльна характеристика методів

Для обґрунтування вибору методів було проведено порівняння за такими критеріями (табл. 2.1):

Таблиця 2.1 – Порівняльна характеристика методів

Метод	Мін. історія даних	Складність обчислень (Ресурси)	Інтерпретованість (Прозорість)	Обробка зовнішніх факторів (Акції, погода)
Ковзні середні (Moving Average)	Дуже мала (3+ точки)	Низька (O(n))	Висока (Інтуїтивна)	Ні
Декомпозиція + Лінійна регресія (Trend)	Мала (5-10 точок)	Низька (Миттєво)	Висока (Геометрична)	Обмежена
Експоненційне згладжування	Мала	Низька (Миттєво)	Середня	Ні
Класична декомпозиція	Середня (1-2 сезони)	Низька	Висока (Наочна)	Ні
Facebook Prophet	Середня	Середня	Висока	Так (Свята, події)
SARIMA	Велика (50+ точок)	Висока (Ітеративна)	Середня (Статистична)	Слабка
XGBoost / LightGBM (Бустинг)	Велика	Середня	Низька	Висока (Будь-які фактори)
LSTM / RNN (Нейромережі)	Величезна (Big Data)	Дуже висока (GPU)	Низька ("Чорна скринька")	Висока

Для задачі створення аналітичного веб додатку для швидкого прогнозування (low-friction entry), поточний вибір методів (Класична декомпозиція, Лінійна

регресія, Ковзні середні, Експоненційне згладжування) є оптимальним з огляду на наступні теоретичні концепції:

1. Принцип Парсимонії (Бритва Оккама)

В економетриці існує консенсус: при рівних умовах простіша модель є кращою. Складні моделі (SARIMA, LSTM) мають велику кількість параметрів, які необхідно оцінити. При обмеженому обсязі даних (короткі часові ряди, характерні для нових користувачів/малого бізнесу) оцінки цих параметрів мають високу дисперсію. Ваші методи мають мінімальну кількість ступенів вільності, що робить їх більш надійними на малих вибірках.

2. Компроміс зміщення та дисперсії (Bias-Variance Trade-off)

Це фундаментальна проблема машинного навчання. Альтернативні методи мають низьке зміщення (можуть описати складну криву), але високу дисперсію (дуже чутливі до шуму в навчальних даних). На "брудних" даних малого бізнесу вони схильні до перенавчання (overfitting), сприймаючи випадковий шум як сигнал.

Методи, що були обрані мають вище зміщення (спрощують реальність), але низьку дисперсію. Вони дають стабільний результат, який краще узагальнюється (generalization) на нових, невідомих даних, мінімізуючи ризик грубих помилок прогнозу.

3. Асимптотична обчислювальна складність

Обрані методи мають складність алгоритмів порядку O(n) або навіть O(1) для оновлення прогнозу (як у експоненційному згладжуванні). Це дозволяє проводити розрахунки на стороні клієнта (Client-side computing) без затримок.

Методи типу LSTM або налаштування гіперпараметрів SARIMA (Grid Search) вимагають ітеративних оптимізаційних процедур, що створює значне обчислювальне навантаження і неприйнятне для миттєвого відгуку інтерфейсу.

4. Інтерпретованість моделі (Model Explainability)

У контексті систем підтримки прийняття рішень (Decision Support Systems), довіра користувача корелює зі зрозумілістю логіки моделі.

Обрані методи є прозорими ("White box models"). Користувач може інтуїтивно зрозуміти геометричну інтерпретацію лінійної регресії (лінія тренду)

або середнього значення. Це критично для непідготовленого користувача, який має валідувати прогноз на основі власного бізнес-досвіду. Складні моделі є "чорними скриньками", що знижує рівень довіри до продукту.

2.9. Висновки до розділу 2

У другому розділі виконано детальне дослідження теоретико-методологічних основ прогнозування продажів та обґрунтовано вибір математичного апарату для інформаційно-аналітичної системи.

1. Аналіз методів прогнозування. Було розглянуто та програмно змодельовано чотири ключові підходи до виділення сезонності та тренду:
    - Класична декомпозиція: визначена як базова модель (baseline), що забезпечує високу наочність для користувача, але має обмеження при роботі з нестаціонарними рядами.
    - Декомпозиція з лінійним трендом: обґрунтована як найоптимальніший метод для товарів зі стабільним зростанням або падінням попиту, дозволяючи "очистити" сезонні коефіцієнти від впливу глобальної динаміки.
    - Метод ковзних середніх: визнаний ефективним інструментом для фільтрації шуму та роботи з волатильними даними, що є типовим для роздрібної торгівлі.
    - Експоненційне згладжування: обрано завдяки його адаптивності та здатності швидко реагувати на структурні зрушення в попиті, надаючи більшу вагу актуальним даним.
2. Обґрунтування технологічного стека. На основі порівняльного аналізу з альтернативними методами (SARIMA, LSTM, Prophet) доведено, що для задачі створення масового веб-продукту (SaaS) для малого та середнього бізнесу обраний набір алгоритмів є оптимальним згідно з такими критеріями:
    - Принцип парсимонії (Бритва Оккама): простіші моделі забезпечують кращу узагальнювальну здатність на коротких часових рядах ("проблема холодного старту"), мінімізуючи ризик перенавчання (overfitting).

- Інтерпретованість (White Box): прозорість логіки розрахунків підвищує довіру користувачів до системи підтримки прийняття рішень, на відміну від моделей "чорної скриньки".
- Обчислювальна ефективність: алгоритмічна складність порядку O(n) дозволяє виконувати розрахунки в реальному часі без залучення високонавантажених серверних потужностей (GPU).

3. Готовність до реалізації. Розроблені описи алгоритмів та їх програмна декомпозиція мовою Python (з використанням бібліотек pandas та numpy) сформували чітке технічне завдання для етапу розробки. Встановлено, що система повинна реалізовувати стратегію ансамблевого прогнозування, генеруючи результати всіма методами паралельно та надаючи користувачеві можливість вибору на основі метрик точності.

Таким чином, у розділі сформовано надійний математичний та алгоритмічний фундамент, який гарантує, що розроблена у наступному розділі система «Sales Forecaster» буде не лише технічно працездатною, а й економічно ефективною для кінцевого користувача.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА АПРОБАЦІЯ ІНФОРМАЦІЙНО - АНАЛІТИЧНОЇ СИСТЕМИ ПРОГНОЗУВАННЯ ПРОДАЖІВ

3.1. Загальна характеристика розробленого програмного забезпечення

У межах кваліфікаційної роботи було розроблено веб-орієнтовану інформаційно-аналітичну систему прогнозування продажів під назвою “Sales Forecaster”. Програмний продукт реалізовано у форматі веб-додатку, що забезпечує доступ до його функціоналу через стандартний веб-браузер без необхідності встановлення додаткового програмного забезпечення на стороні користувача.

Взаємодія користувача з системою здійснюється через графічний веб-інтерфейс, який забезпечує інтуїтивно зрозумілий доступ до основних функцій системи, а саме: завантаження вхідних даних, вибір методів прогнозування, запуск процесу обчислень та перегляд результатів у вигляді таблиць і графіків.

Типовим сценарієм використання системи є процес прогнозування обсягів продажів за окремою товарною категорією. Наприклад, категорійний менеджер торговельної мережі формує файл формату CSV, що містить історичні дані про продажі категорії «Фрукти» за попередні три роки. Після цього користувач завантажує підготовлений файл у веб-інтерфейсі системи Sales Forecaster (дивитись рис. 3.1), ініціює процес аналізу та отримує дашборд з візуальними графіками сезонних коливань, підсумковими метриками, детальними графіками оцінки ефективності кожного з методів прогнозування (рис. 3.2).

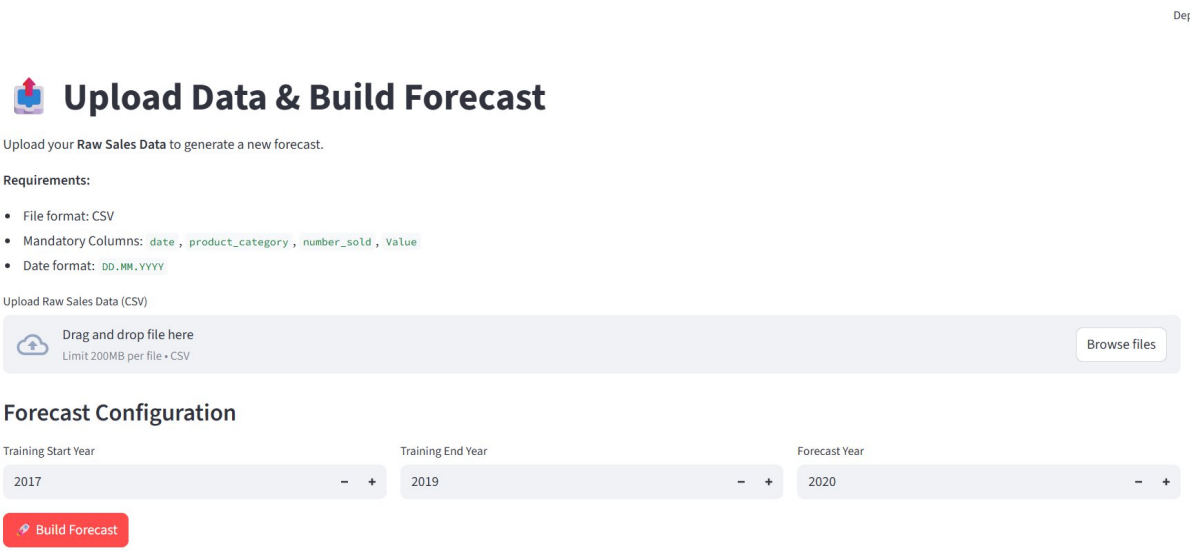

Рисунок 3.1 – Веб-інтерфейс системи Sales Forecaster. Сторінка “Upload data & Build forecast”

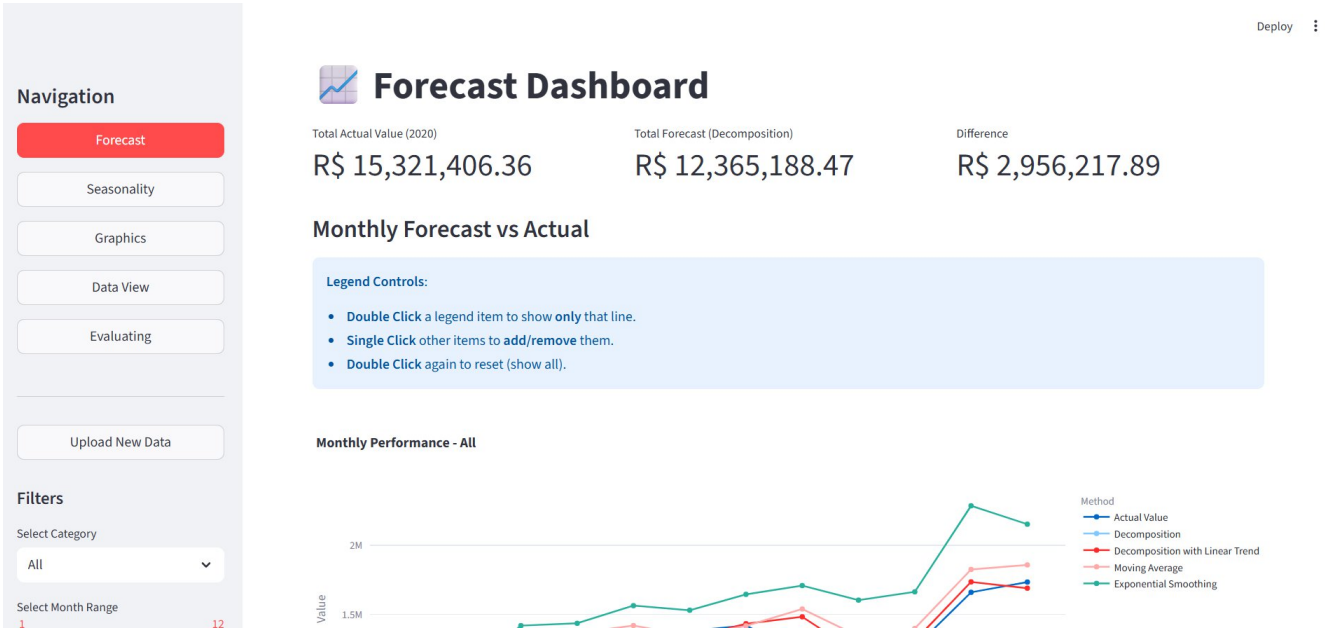

Рисунок 3.2 – Веб-інтерфейс системи Sales Forecaster. Сторінка “Forecast”

Розроблена система орієнтована на використання в умовах реальної бізнес-аналітики та прийняття управлінських рішень, зокрема у сфері управління асортиментом, планування закупівель і оптимізації товарних запасів.

3.1.1. Функціональні можливості системи

Розроблене програмне забезпечення забезпечує виконання таких основних функцій:

- імпорт історичних даних продажів з файлів формату CSV/Excel;
- автоматичну перевірку відповідності даних та їх агрегацію;
- розрахунок коефіцієнтів сезонності чотирма методами;
- побудову прогнозних значень;
- порівняння прогнозних даних із фактичними;
- обчислення похибок прогнозу (MAE, RMSE, MAPE);
- візуалізацію результатів у вигляді графіків та таблиць.

3.2. Архітектура програмної системи

Архітектура розробленої системи «Sales Forecaster» базується на модульному принципі, що забезпечує гнучкість розробки, легкість масштабування та зручність тестування окремих компонентів. В основу архітектурного рішення покладено патерн, адаптований до концепції MVC (Model-View-Controller), реалізований засобами фреймворку Streamlit.

3.2.1 Загальна структура

Система розділена на три логічні рівні, кожен з яких відповідає за специфічні задачі:

1. Рівень управління (Controller): Реалізований у головному модулі app.py. Цей компонент виступає точкою входу в додаток. Його основними функціями є:
    - Управління глобальним станом сесії (session_state), що зберігає інформацію про завантажені дані, обрані параметри прогнозування та поточний стан розрахунків.
    - Маршрутизація користувача (Routing): обробка навігації через бічну панель та перемикання між різними видами (Views).
2. Рівень представлення (View): Зосереджений у директорії views/ і відповідає за інтерфейс користувача. Кожен файл у цій директорії відповідає окремій сторінці або вкладці додатку:

- upload_view.py — інтерфейс завантаження даних та ініціалізації прогнозування;
- forecast_view.py — інтерактивний дашборд з результатами;
- evaluating_view.py — відображення метрик точності моделей;
- graphics_view.py — галерея статичних графіків високої якості.

3. Рівень бізнес-логіки (Model): Інкапсульований у директорії utils/. Цей рівень містить "чисті" Python-функції, які виконують обробку даних та математичні розрахунки, будучи відділеними від інтерфейсу користувача:
   - data_processing.py — відповідає за валідацію вхідних CSV-файлів, очищення даних та їх агрегацію (помісячну/потижневу);
   - forecasting.py — містить реалізацію алгоритмів прогнозування (декомпозиція, ковзні середні, експоненційне згладжування);
   - visualization.py — генерує статичні зображення графіків для звітів.

3.2.2. Схема взаємодії компонентів

Структурна схема системи відображає потоки даних від моменту завантаження користувачем до візуалізації результатів (рис. 3.3).

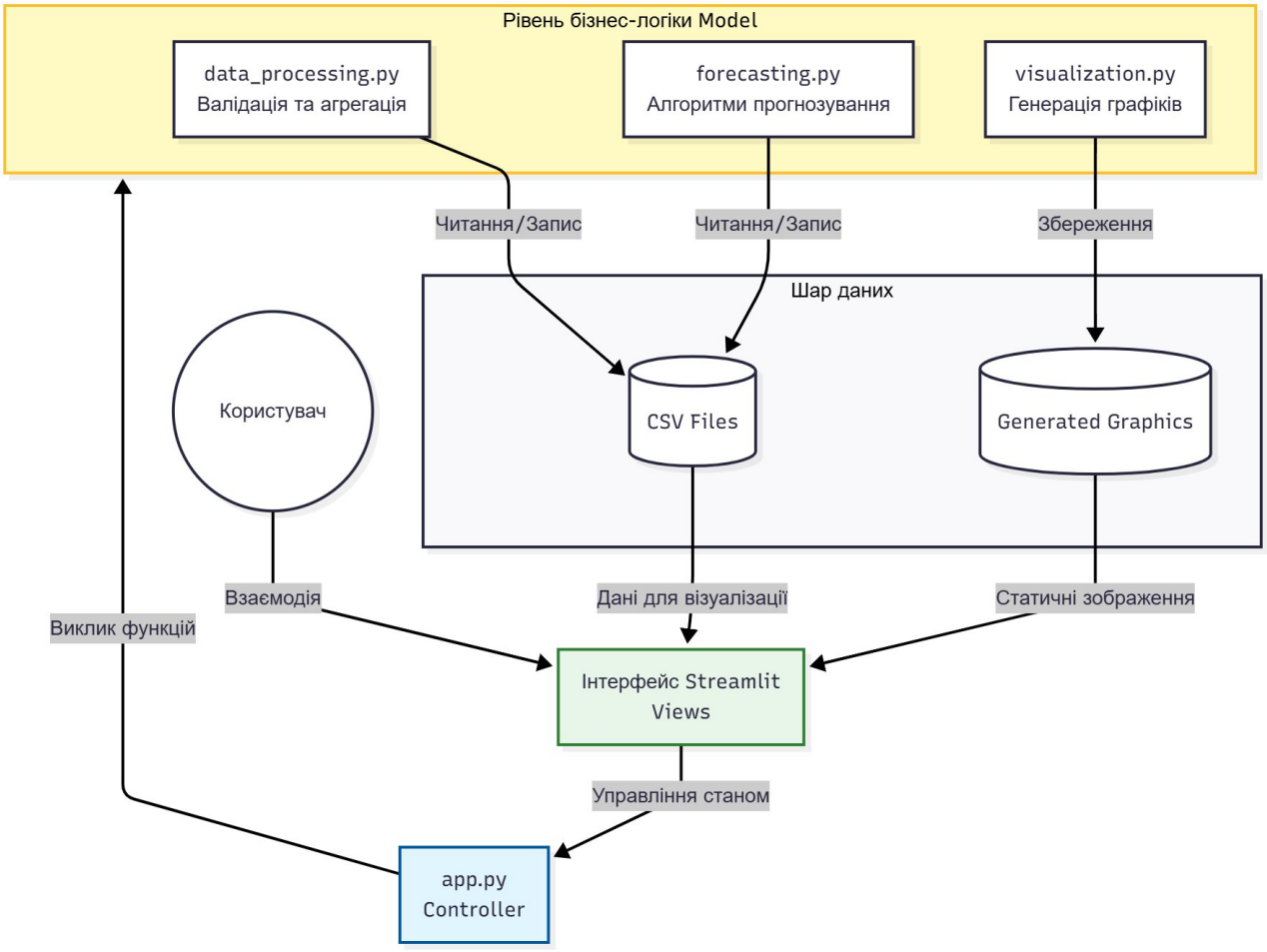

Рисунок 3.3 – схема взаємодії компонентів

3.2.3. Потоки даних (Data Flows)

Функціонування системи забезпечується трьома ключовими процесами обробки даних:

1. Інтеграція даних (Data Ingestion): Користувач завантажує файл через upload_view.py. Модуль data_processing.py перевіряє структуру колонок та трансформує дані, зберігаючи їх у локальному сховищі data/csv_files/.
2. Генерація прогнозів: Процес ініціюється користувачем. Модуль forecasting.py застосовує чотири математичні моделі до вхідних даних, генеруючи файли прогнозів (forecast_{year}_monthly.csv та ін.). Паралельно модуль visualization.py створює статичні графіки.
3. Візуалізація та оцінка: Модуль forecast_view.py зчитує згенеровані файли та візуалізує їх за допомогою бібліотеки Plotly Express, а модуль

evaluating_view.py порівнює прогнози з фактичними даними, розраховуючи показники MAE, MAPE та RMSE.

Така архітектура дозволяє чітко розмежувати зони відповідальності компонентів, що спрощує підтримку коду та дозволяє додавати нові методи прогнозування без необхідності зміни логіки інтерфейсу.

3.3. Технологічний стек і середовище розробки

Система «Sales Forecaster» реалізована мовою програмування Python (версія 3.10), що забезпечує гнучкість, надійність та широкі можливості для аналітичної обробки даних. Вибір Python зумовлений його лідируючими позиціями у сфері науки про дані (Data Science), наявністю потужних бібліотек для статистичного аналізу та розвиненою екосистемою веб-фреймворків.

3.3.1 Використані бібліотеки та фреймворки

Для реалізації функціоналу системи було використано набір спеціалізованих бібліотек. Основні компоненти стека включають:

1. Streamlit — високорівневий веб-фреймворк, що виступає ядром інтерфейсу користувача. Він дозволяє швидко перетворювати скрипти обробки даних у інтерактивні веб-додатки. Використання Streamlit дозволило реалізувати реактивний інтерфейс, де зміни параметрів прогнозування миттєво відображаються на графіках без необхідності перезавантаження сторінки.
2. Pandas — фундаментальна бібліотека для маніпуляцій з табличними даними. У системі вона використовується для:
   - читання та парсингу вхідних CSV-файлів;
   - валідації типів даних;
   - агрегації часових рядів (групування за тижнями/місяцями);
   - виконання векторизованих операцій для розрахунку ковзних середніх.
3. Statsmodels — бібліотека для статистичного моделювання. Зокрема, функція seasonal_decompose використовується для реалізації методу класичної декомпозиції часових рядів, дозволяючи програмно виділити трендову та сезонну складові.

4. Plotly Express — бібліотека для побудови інтерактивних графіків. Вона забезпечує візуалізацію результатів прогнозування на дашборді, надаючи користувачеві можливість масштабувати графіки (zoom), вимикати окремі лінії легенди та переглядати точні значення при наведенні курсору (hover).
5. Matplotlib & Seaborn — використані у фонових процесах (utils/visualization.py) для генерації високоякісних статичних зображень, які зберігаються у форматі PNG для формування звітності.

3.4. Реалізація основних модулів системи

Програмна реалізація системи «Sales Forecaster» виконана з дотриманням принципів слабкої зв'язаності (low coupling), що дозволило розділити логіку обробки даних, математичні обчислення та інтерфейс користувача. Нижче наведено детальний опис реалізації ключових модулів.

3.4.1. Модуль обробки та валідації даних

Цей модуль (utils/data_processing.py) відповідає за етап ETL (Extract, Transform, Load). Його основним завданням є перетворення "сирих" транзакційних даних у структуровані часові ряди, придатні для прогнозування.

Ключові функції модуля:

1. Валідація схеми даних: При завантаженні CSV-файлу система перевіряє наявність обов'язкових колонок (date, product_category, number_sold, Value). Якщо формат не відповідає вимогам, процес зупиняється з відповідним повідомленням користувачеві.
2. Агрегація часових рядів: Оскільки вхідні дані зазвичай містять щоденні транзакції, модуль виконує ресемплінг (resampling) даних до тижневого (W) або місячного (M) рівня. Це реалізовано засобами бібліотеки Pandas, що дозволяє суттєво зменшити розмірність даних та знизити рівень шуму перед прогнозуванням.
3. Очищення даних: Автоматична обробка пропущених значень (NaN) та приведення дат до єдиного формату datetime.

3.4.2. Модуль математичного моделювання

Ядром системи є модуль прогнозування (utils/forecasting.py). Він інкапсулює реалізацію чотирьох обраних математичних алгоритмів та відповідає за генерацію числових прогнозів.

Особливості реалізації:

- Ансамблевий підхід: Функція генерації прогнозу запускає всі чотири методи (класична декомпозиція, декомпозиція з лінійним трендом, ковзні середні, експоненційне згладжування) послідовно для кожного часового ряду.
- Використання Statsmodels: Для виділення сезонної компоненти методом класичної декомпозиції використовується функція seasonal_decompose з бібліотеки statsmodels. Вона розкладає ряд на тренд, сезонність та залишкову компоненту (resid).
- Кастомні алгоритми: Методи ковзних середніх та експоненційного згладжування реалізовані як власні Python-функції (без використання зовнішніх "чорних скриньок"), що дозволило точно налаштувати логіку зважування останніх періодів (параметр α) та адаптувати її до специфіки роздрібної торгівлі.

Результатом роботи модуля є згенеровані CSV-файли прогнозів (наприклад, forecast_2024_monthly.csv), які зберігаються у директорії data/csv_files/ для подальшого використання.

3.4.3. Модуль візуалізації та інтерфейсу

Візуалізація даних реалізована на двох рівнях: інтерактивному (для веб-додатку) та статичному (для звітів).

1. Інтерактивний дашборд (views/forecast_view.py): Використовує бібліотеку Plotly Express для побудови динамічних графіків. Користувач має можливість:
    - Масштабувати графіки (zoom in/out) для детального розгляду пікових періодів.

- Фільтрувати відображення (вмикати/вимикати окремі методи прогнозування на легенді).
- Отримувати точні числові значення при наведенні курсору на точку графіка (hover functionality).

2. Генерація статичних звітів (utils/visualization.py): Використовує бібліотеки Matplotlib та Seaborn для створення високоякісних зображень у форматі PNG. Ці графіки автоматично зберігаються в папці data/graphics/ та використовуються в розділі "Graphics" для експорту у презентації або друковані звіти.

3.4.4. Контролер додатку

Головний файл app.py виконує роль контролера, що зв'язує інтерфейс та бізнес-логіку. Він керує станом сесії (Session State) Streamlit. Це критично важливо для збереження завантажених даних та розрахованих прогнозів при перемиканні між вкладками (наприклад, перехід від завантаження даних до перегляду результатів). Контролер також відповідає за маршрутизацію (Routing) — відображення відповідного модуля view залежно від вибору користувача в меню навігації.

3.5. Тестування та апробація системи

Під час розробки та тестування програми Sales Forecaster було використано реальний набір даних з продажів — Brazilian E-Commerce Public Dataset by Olist, взятий з Kaggle.

Набір даних містить інформацію приблизно про 100 000 замовлень у періоді з 2016 до 2020 років.

Дані представлені у вигляді кількох пов’язаних таблиць на рисунку 3.4.

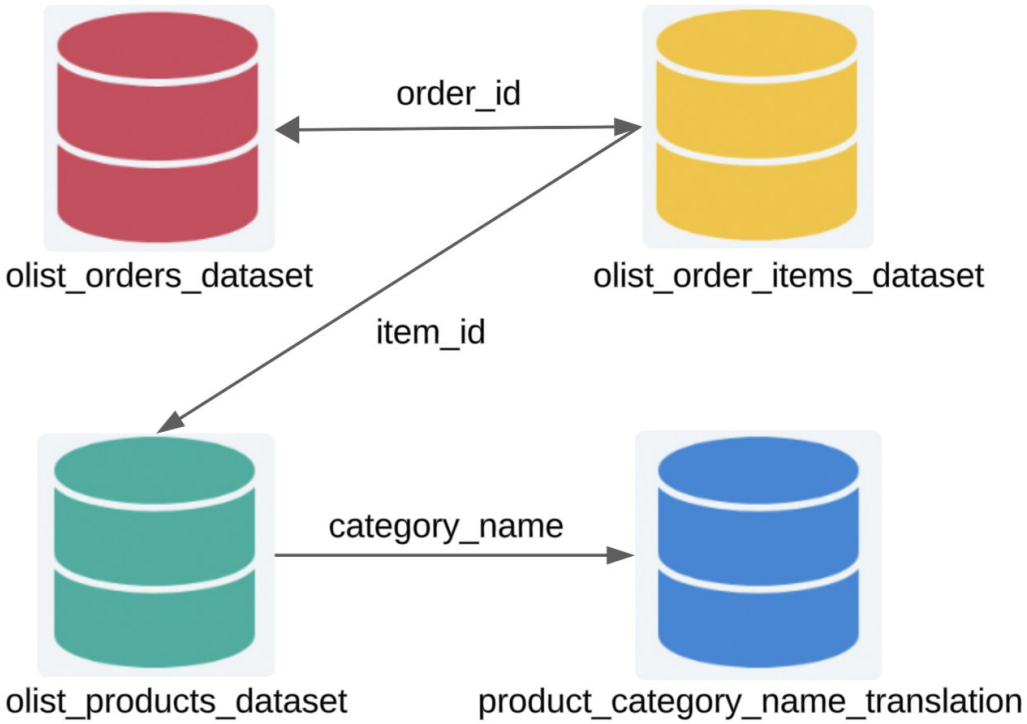

Рисунок 3.4 – Схема зв'язків між таблицями джерела даних

- olist_orders_dataset - з інформацією про замовлення (час покупки, статус замовлення тощо);
- olist_order_items_dataset - дані про товари в кожному замовленні (який товар, кількість, ціна);
- olist_products_dataset - характеристики товарів (категорії, ваги, інші атрибути);
- product_category_name_translation.csv - переклад назв категорій товарів англійською мовою.

Дані анонімізовані: в оригінальному наборі корпоративні назви продавців та інших учасників замінено, щоб зберегти конфіденційність.

Структура набору даних дозволяє аналізувати замовлення з різних вимірів: динаміка замовлень у часі, способи оплати, характеристики товарів, географія клієнтів.

3.5.1. Підготовка даних та розрахунок прогнозів

- Сформовано csv-файл з консолідованою інформацією з джерела даних. Далі цей файл буде імпортовано у програмне забезпечення Sales Forecaster. Код для консолідації даних наведено в додатку Г.

- Виконано очищення: обробка пропущених значень, приведення форматів дат.
- Об’єднання даних: сумування продажів за категоріями на кожну дату.

Апробація прогнозування:

- Використано частину історичних даних (перші 36 місяці) для розрахунку коефіцієнтів сезонності і прогнозування наступного року, а решту (останні 12 місяців) - для порівняння з прогнозом і обчислення похибки.
- Для кожного з чотирьох методів (декомпозиція класична, регресійна, ковзні середні, експоненційне згладжування) побудовано прогноз продажів для кожної категорії (дивитись рис. 3.5).

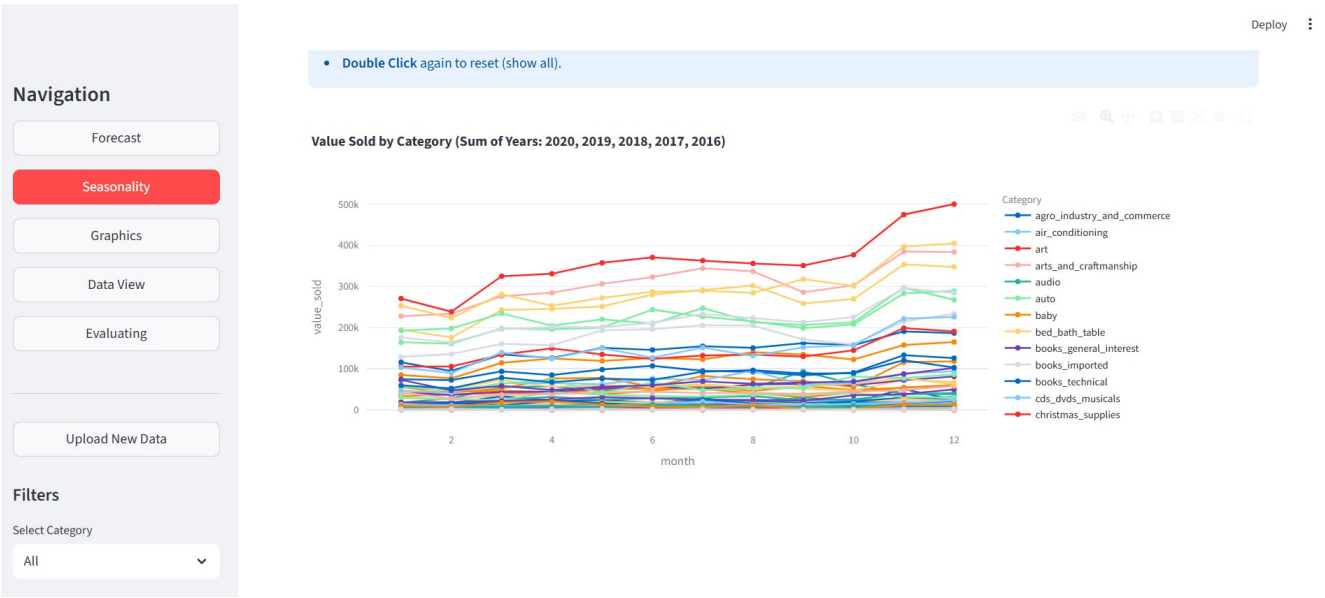

Рисунок 3.5 – Прогноз продажів на 2020 рік для кожної категорії

3.5.2 Оцінка результатів

Для оцінювання якості прогнозних моделей у роботі використано низку статистичних показників похибок, які дозволяють комплексно проаналізувати відхилення прогнозних значень від фактичних даних.

MAE (Mean Absolute Error, середня абсолютна похибка) — це середнє арифметичне модулів різниці між фактичними та прогнозованими значеннями. Показує середній розмір помилки в тих самих одиницях виміру, що й самі дані (наприклад, кількість проданих одиниць товару). Чим менше значення MAE, тим

вищою є точність моделі прогнозування. Перевагою цієї метрики є її зрозумілість та стійкість до поодиноких викидів.

RMSE (Root Mean Square Error, корінь середньоквадратичної похибки) — це квадратний корінь із середнього значення квадратів відхилень між фактами та прогнозами. На відміну від MAE, ця метрика надає більшу вагу великим помилкам, оскільки відхилення зводяться до квадрата. RMSE є чутливою до викидів і добре підходить для оцінки моделей, у яких критично важливі великі похибки.

MAPE (Mean Absolute Percentage Error, середня абсолютна відсоткова похибка) — показник, що відображає середню величину відносної помилки у відсотках. Він дає змогу оцінити точність прогнозу незалежно від масштабу даних, що особливо зручно при порівнянні результатів для різних товарних категорій. Нижчі значення MAPE відповідають більш точному прогнозуванню, однак цей показник може спотворювати результати у випадку наявності дуже малих або нульових фактичних значень.

Total Error (загальна похибка) — агрегований показник, який характеризує сумарне відхилення прогнозованих значень від фактичних за весь досліджуваний період. Зазвичай обчислюється як сума різниць між прогнозом і фактом або як сума абсолютних відхилень. Цей показник дозволяє оцінити, наскільки модель систематично переоцінює або недооцінює фактичний рівень продажів.

Побудовано таблиці з результатами по кожному методу (табл. 3.1).

Таблиця 3.1. – Показники точності прогнозування

Метод	MAE	RMSE	MAPE (%)	Total Error (%)
Класична декомпозиція	4016.62	8167.69	19.29	-19.29
Декомпозиція з лінійним трендом	3145.85	6112.41	5.74	5.74
Метод ковзних середніх	4038.14	7830.45	11.47	11.47
Метод експ. згладжування	5760.92	11971.91	25.05	25.05

Отримані результати свідчать про суттєві відмінності в точності прогнозування між досліджуваними методами розрахунку сезонності.

Найкращі показники за абсолютними похибками (MAE та RMSE) продемонстрував метод декомпозиції з лінійним трендом (MAE = 3145,85; RMSE = 6112,41). Це свідчить про те, що врахування трендової складової через лінійну регресію дозволяє значно підвищити точність прогнозу в порівнянні з простою класичною декомпозицією (рис. 3.6).

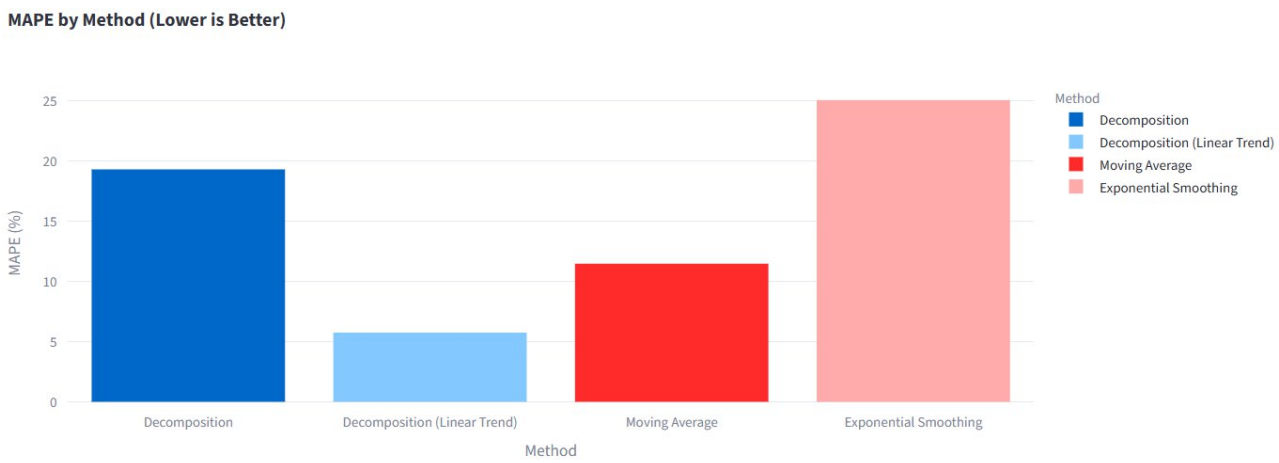

Рисунок 3.6 – Діаграма MAPE кожного з методів прогнозування

Класична декомпозиція без урахування тренду показала гірші результати (MAE = 4016,62; RMSE = 8167,69), проте відзначається від'ємним значенням Total Error (-19,29%), що свідчить про відсутність трендової складової, адже згідно даних продажі мережі в середньому виросли на 15% з минулого року.

Метод ковзних середніх продемонстрував подібний рівень абсолютної похибки до класичної декомпозиції (MAE = 4038,14; RMSE = 7830,45), проте має дещо вищий показник MAPE (11.47%), що свідчить про низьку точність у відносному вимірі та слабку стабільність прогнозів, особливо при великих коливаннях попиту.

Метод експоненційного згладжування показав найгірші результати серед усіх досліджуваних методів (MAE = 5760,92; RMSE = 11971,91), а також найбільше

позитивне значення Total Error (25,05%), що вказує на систематичне завищення прогнозованих обсягів продажів.

Також в роботі проведено візуалізацію прогнозів проти фактичних значень, щоб перевірити, наскільки добре моделі відтворюють сезонні коливання (дивитись рис. 3.7 і 3.8).

Рисунок 3.7– Порівняння щотижневих прогнозів продажів на 2020 рік з фактичними продажами (Actual Value)

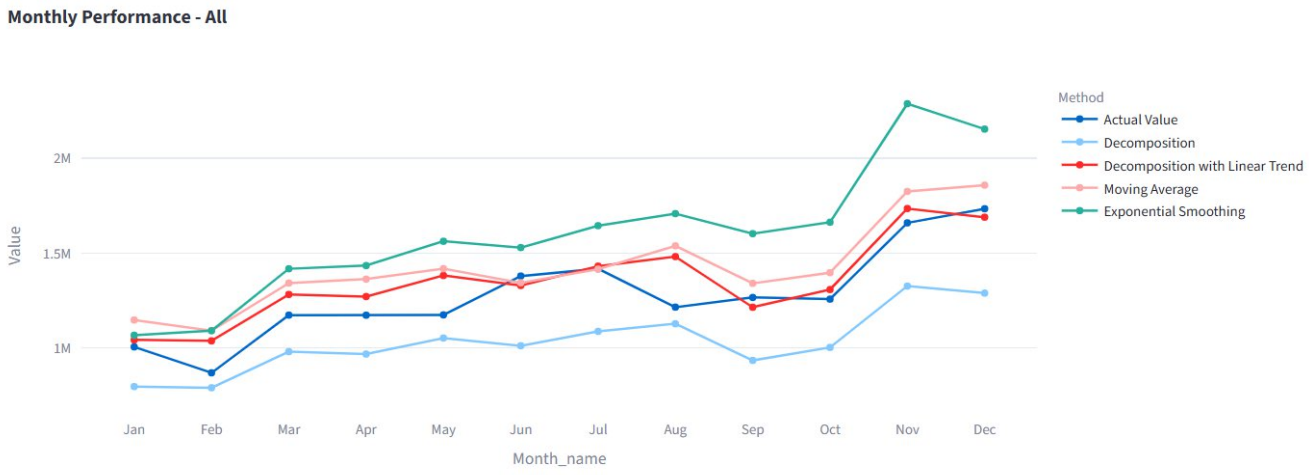

Рисунок 3.8 – Порівняння щомісячних прогнозів продажів на 2020 рік з фактичними продажами (Actual Value)

На основі комплексного аналізу всіх чотирьох метрик можна зробити такі висновки:

- Найбільш точним методом є декомпозиція з лінійним трендом, яка забезпечує мінімальні абсолютні похибки та не демонструє значних систематичних перекосів у прогнозах.
- Класична декомпозиція демонструє прийнятну точність, але без урахування тренду має схильність до заниження прогнозу.
- Метод ковзних середніх виявився нестабільним у відносних помилках, що обмежує його практичну застосовність для точного прогнозування.
- Метод експоненційного згладжування виявився найменш ефективним для обраного набору даних через значні відхилення та систематичне перевищення прогнозних значень.

3.5.3 Висновки тестування

Набір даних Olist з Kaggle є достатньо багатим для задач прогнозування сезонності, оскільки містить об’ємні часові ряди продажів, інформацію про товари, клієнтів та транзакції.

Використання реальних комерційних даних підвищує практичну значимість розробленої системи.

Тестування на цих даних дозволяє адекватно оцінити застосовані методи прогнозування та порівняти їх ефективність у реальних умовах.

Результати експериментального дослідження підтверджують доцільність використання методів, які явно враховують трендову складову часових рядів. У рамках даного дослідження найбільш ефективним виявився метод декомпозиції з лінійною регресією, який рекомендовано як базовий для практичного використання в інформаційно-аналітичній системі прогнозування продажів.

3.6 Аналіз результатів прогнозування

У межах даного дослідження було виконано прогнозування обсягів продажів із використанням чотирьох методів врахування сезонності:

1. класична декомпозиція без урахування тренду;
2. декомпозиція з трендом, побудованим методом лінійної регресії;
3. метод ковзних середніх (із побудовою як тренду, так і сезонної компоненти);

4. метод експоненційного зважування (з інтегрованим врахуванням тренду та сезонності).

Для кожного з методів були розраховані коефіцієнти сезонності в двох часових розрізах: помісячному та потижневому. Це дозволило окремо оцінити ефективність кожного підходу для різних рівнів деталізації часових рядів.

3.6.1. Аналіз місячної сезонності

На основі розрахованих коефіцієнтів було побудовано графіки сезонних коливань для кожного методу (рис. 3.9).

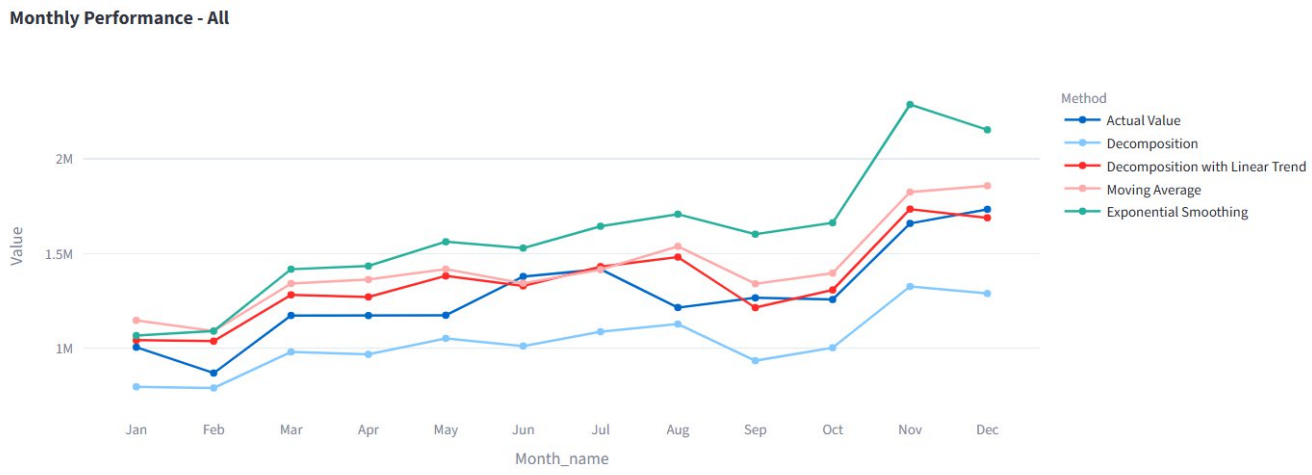

Рисунок 3.9 – Порівняння графіків усіх методів прогнозування з фактичними продажами в розрізі місяців

Аналіз графіків показав наявність чітко виражених сезонних піків у періоди підвищеного попиту, а також сезонних спадів у міжсезонні проміжки. Найбільш згладжену криву продемонстрував метод ковзних середніх, що свідчить про його кращу адаптацію до випадкових коливань даних. Метод класичної декомпозиції без тренду, навпаки, показав більш різкі коливання, що може призводити до завищення або заниження прогнозних значень.

Порівняння прогнозних значень із фактичними даними за контрольний рік засвідчило (дивитися таблицю 3.2), що методи з урахуванням тренду (лінійна регресія, ковзні середні, експоненційне зважування) демонструють вищу точність прогнозування порівняно з моделями без трендової складової.

Таблиця 3.2. – Порівняння прогнозів з фактичними продажами торговельної мережі

Місяць	Фактичні продажі	Декомпозиція	Лінійна декомпозиція	Метод ковзних середніх	Експоненційне згладжування
Січень	1 005 572,41	796 321,94	1 042 352,14	1 147 189,20	1 066 972,63
Лютий	868 805,19	790 103,14	1 037 268,40	1 091 127,78	1 090 919,70
Березень	1 173 066,37	980 467,72	1 281 773,51	1 342 028,06	1 417 326,39
Квітень	1 172 468,73	967 821,87	1 270 880,18	1 362 619,84	1 434 103,26
Травень	1 174 163,31	1 052 152,09	1 381 668,33	1 417 871,48	1 562 284,29
Червень	1 378 907,99	1 011 535,19	1 329 039,00	1 343 474,67	1 529 034,40
Липень	1 417 179,68	1 087 300,86	1 431 172,48	1 415 627,05	1 644 411,62
Серпень	1 215 034,06	1 127 468,52	1 481 423,43	1 538 091,52	1 708 042,27
Вересень	1 265 998,11	934 291,74	1 214 361,72	1 340 649,92	1 602 321,25
Жовтень	1 257 276,34	1 002 801,40	1 307 768,81	1 396 513,57	1 662 658,19
Листопад	1 659 189,39	1 325 978,93	1 734 759,14	1 824 900,69	2 287 256,76
Грудень	1 733 744,78	1 288 945,07	1 689 177,52	1 858 073,87	2 153 568,72

3.6.2. Аналіз тижневої сезонності

Порівняння прогнозних даних з фактичними в розрізі тижнів можна подивитися в додатку Д.

Окремо було побудовано графіки тижневих коефіцієнтів сезонності (рис. 3.10).

Рисунок 3.10 - Порівняння графіків усіх методів прогнозування з фактичними продажами в розрізі тижнів

Тижнева сезонність виявилася більш волатильною, ніж місячна, що обумовлено коротшим інтервалом агрегації даних та впливом короткострокових факторів (акції, розпродажі, святкові періоди). Найменшу амплітуду випадкових коливань продемонстрував метод експоненційного зважування, що свідчить про його високу адаптивність до локальних змін у часовому ряді.

Порівняння прогнозів із фактичними даними показало, що тижневі прогнози загалом мають вищу похибку, ніж місячні, однак забезпечують більш детальне уявлення про структуру попиту.

3.6.3. Оцінка точності методів прогнозування

Аналіз таблиць показав, що:

- для місячної сезонності найкращі результати продемонстрували методи декомпозиції з лінійним трендом та ковзних середніх;
- для тижневої сезонності найменші похибки показав метод експоненційного зважування;
- класична декомпозиція без тренду має найнижчу точність через ігнорування довгострокової динаміки ряду.

3.6.4. Висновки до підрозділу

За результатами проведеного аналізу встановлено, що вибір методу розрахунку сезонності суттєво впливає на точність прогнозування. Методи, які

враховують як тренд, так і сезонну компоненту (ковзні середні та експоненційне зважування), забезпечують більш стабільні та точні результати порівняно з класичною декомпозицією.

Окреме оцінювання місячної та тижневої сезонності дозволило виявити специфіку поведінки попиту на різних часових горизонтах та підтвердило доцільність використання адаптивних методів для коротких періодів прогнозування. Отримані результати можуть бути практично використані в інформаційно-аналітичних системах прогнозування продажів торговельних мереж для підвищення ефективності планування запасів і управління асортиментом.

3.7 Висновки до розділу 3

У межах третього розділу було реалізовано та апробовано інформаційно-аналітичну систему прогнозування продажів Sales Forecaster, яка забезпечує автоматизований розрахунок коефіцієнтів сезонності та побудову прогнозів на основі різних методів аналізу часових рядів. Розроблений програмний продукт продемонстрував працездатність, стабільність та практичну придатність для аналізу історичних даних продажів.

У даному розділі проведено комплексний кількісний і візуальний аналіз точності прогнозування, що включав обчислення метрик похибок (MAE, RMSE, MAPE, Total Error), а також графічне порівняння прогнозних значень із фактичними обсягами продажів. Це дозволило виявити характерні особливості поведінки кожного з досліджуваних методів, а саме схильність до систематичного завищення або заниження результатів та здатність адекватно відтворювати сезонні коливання попиту.

Отримані результати підтвердили, що ефективність методів прогнозування суттєво залежить від способу побудови трендової та сезонної складових. Найвищу точність для досліджуваного набору даних продемонструвала декомпозиція з використанням лінійного тренду, тоді як методи ковзних середніх та експоненційного згладжування показали нижчу стабільність при роботі з даними, що мають значні коливання.

Таким чином, у третьому розділі було не лише реалізовано програмний продукт, а й експериментально доведено доцільність використання методів прогнозування, що враховують трендову складову. Результати розділу слугують обґрунтуванням практичної цінності розробленої системи та створюють основу для формування загальних висновків кваліфікаційної роботи.

ВИСНОВКИ

У дипломній роботі вирішено актуальне науково-прикладне завдання створення інформаційно-аналітичної системи для прогнозування продажів торговельної мережі. На основі проведеного теоретичного узагальнення та експериментального дослідження зроблено наступні висновки:

1. Проаналізовано особливості формування попиту в роздрібній торгівлі та встановлено, що для підприємств малого та середнього бізнесу (SME) критично важливим є використання методів, здатних працювати в умовах обмеженої історії даних («проблема холодного старту»). Доведено, що ігнорування сезонної та трендової компонент при плануванні закупівель призводить до фінансових втрат через ефекти out-of-stock або затоварення складів.
2. Експериментально перевірено ефективність чотирьох статистичних методів на реальному наборі даних Brazilian E-Commerce Public Dataset. Результати апробації спростували гіпотезу про універсальність методу експоненційного згладжування для всіх типів товарів, оскільки він продемонстрував найвищу похибку (MAPE = 25,05%) через схильність до переоцінки тренду. Натомість, найбільш точним виявився метод декомпозиції з лінійним трендом, який забезпечив мінімальну похибку прогнозу (MAPE = 5,74%, MAE = 3145,85). Це підтверджує доцільність попереднього очищення даних від тренду перед розрахунком сезонних коефіцієнтів.
3. Розроблено веб-орієнтовану інформаційно-аналітичну систему «Sales Forecaster» на базі мови Python та фреймворку Streamlit. Архітектура системи, побудована за принципом MVC (Model-View-Controller), забезпечує модульність та масштабованість рішення. Реалізований функціонал дозволяє автоматизувати повний цикл прогнозування: від валідації "сирих" даних до візуалізації порівняльного аналізу моделей, що значно скорочує час на прийняття управлінських рішень.
4. Запропонований ансамблевий підхід, при якому система розраховує прогнози одночасно кількома методами та автоматично визначає похибки

(RMSE, MAE), дозволяє категорійним менеджерам обирати оптимальну стратегію закупівель для кожної товарної групи індивідуально, спираючись на об'єктивні метрики точності, а не на інтуїцію.

Пропозиції щодо подальшого розвитку системи:

1. Реалізувати API-шлюзи для прямої інтеграції з популярними маркетплейсами (Amazon, Shopify) та ERP-системами, що дозволить завантажувати дані про продажі в режимі реального часу без необхідності ручного експорту CSV-файлів.
2. Доповнити модуль прогнозування алгоритмом Facebook Prophet для кращого врахування ефектів свят та маркетингових акцій, які складно моделювати лінійними методами.
3. Розробити модуль «Best Fit», який автоматично обиратиме та підсвічуватиме користувачу метод з найменшою похибкою на тестовому періоді, виключаючи необхідність ручного аналізу графіків.
4. Розширити аналітичний блок функцією розрахунку оптимального страхового запасу (Safety Stock) та точки перезамовлення (Reorder Point) на основі отриманого прогнозу та варіативності попиту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Box G. E. P., Jenkins G. M., Reinsel G. C., Ljung G. M. (2015) Time Series Analysis: Forecasting and Control. 5th ed. Hoboken: Wiley.
2. Brockwell P. J., Davis R. A. (2016) Introduction to Time Series and Forecasting. 3rd ed. New York: Springer.
3. Chatfield C. (2004) The Analysis of Time Series: An Introduction. 6th ed. Boca Raton: Chapman & Hall/CRC.
4. Cowpertwait P. S. P., Metcalfe A. V. (2009) Introductory Time Series with R. New York: Springer.
5. Gujarati D. N., Porter D. C. (2009) Basic Econometrics. 5th ed. New York: McGraw-Hill Education.
6. Hamilton J. D. (1994) Time Series Analysis. Princeton: Princeton University Press.
7. Hyndman R. J., Athanasopoulos G. (2021) Forecasting: Principles and Practice. 3rd ed. Melbourne: OTexts. [Електронний ресурс]. URL: https://otexts.com/fpp3/ (дата звернення: 13.11.2025).
8. Montgomery D. C., Jennings C. L., Kulahci M. (2015) Introduction to Time Series Analysis and Forecasting. 2nd ed. Hoboken: Wiley.
9. Shumway R. H., Stoffer D. S. (2017) Time Series Analysis and Its Applications: With R Examples. 4th ed. Cham: Springer.
10. Wei W. W. S. (2006) Time Series Analysis: Univariate and Multivariate Methods. 2nd ed. Boston: Addison-Wesley.
11. Armstrong J. S. (2001) Principles of Forecasting: A Handbook for Researchers and Practitioners. New York: Springer.
12. Everitt B. S., Skrondal A. (2010) The Cambridge Dictionary of Statistics. 4th ed. Cambridge: Cambridge University Press.
13. Stock J. H., Watson M. W. (2019) Introduction to Econometrics. 4th ed. Boston: Pearson.
14. James G., Witten D., Hastie T., Tibshirani R. (2021) An Introduction to Statistical Learning: With Applications in R. 2nd ed. New York: Springer.

15. Bishop C. M. (2006) Pattern Recognition and Machine Learning. New York: Springer.
16. Goodfellow I., Bengio Y., Courville A. (2016) Deep Learning. Cambridge: MIT Press.
17. Han J., Kamber M., Pei J. (2022) Data Mining: Concepts and Techniques. 4th ed. Cambridge: Morgan Kaufmann.
18. McKinney W. (2017) Python for Data Analysis: Data Wrangling with pandas, NumPy, and Jupyter. 2nd ed. Sebastopol: O'Reilly Media.
19. VanderPlas J. (2016) Python Data Science Handbook: Essential Tools for Working with Data. Sebastopol: O'Reilly Media.
20. Géron A. (2022) Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. 3rd ed. Sebastopol: O'Reilly Media.
21. Brownlee J. (2020) Deep Learning for Time Series Forecasting. Machine Learning Mastery.
22. Seabold S., Perktold J. (2010) Statsmodels: Econometric and Statistical Modeling with Python. Proceedings of the 9th Python in Science Conference. P. 92–96.
23. Brazilian E-Commerce Public Dataset by Olist [Електронний ресурс]. URL: https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce (дата звернення: 07.10.2025).
24. NIST/SEMATECH. e-Handbook of Statistical Methods. National Institute of Standards and Technology [Електронний ресурс]. URL: https://www.itl.nist.gov/div898/handbook/ (дата звернення: 15.11.2025).
25. Jupyter Project. Project Jupyter Documentation [Електронний ресурс]. URL: https://jupyter.org/ (дата звернення: 07.12.2025).
26. Power BI Documentation. Microsoft Learn [Електронний ресурс]. URL: https://learn.microsoft.com/power-bi/ (дата звернення: 21.11.2025).
27. Russo M., Ferrari A. (2019) The Definitive Guide to DAX. 2nd ed. Redmond: Microsoft Press.
28. Collie R., Singh A. (2016) Pro Power BI Desktop. New York: Apress.

29. Olist. About Olist [Електронний ресурс]. URL: https://olist.com/ (дата звернення: 09.10.2025).
30. Datacamp. Time Series Analysis in Python [Електронний ресурс]. URL: https://www.datacamp.com/courses/time-series-analysis-in-python (дата звернення: 25.10.2025).

ДОДАТКИ

Додаток А. Контекстна діаграма «Прогнозування продажів торгівельної мережі»

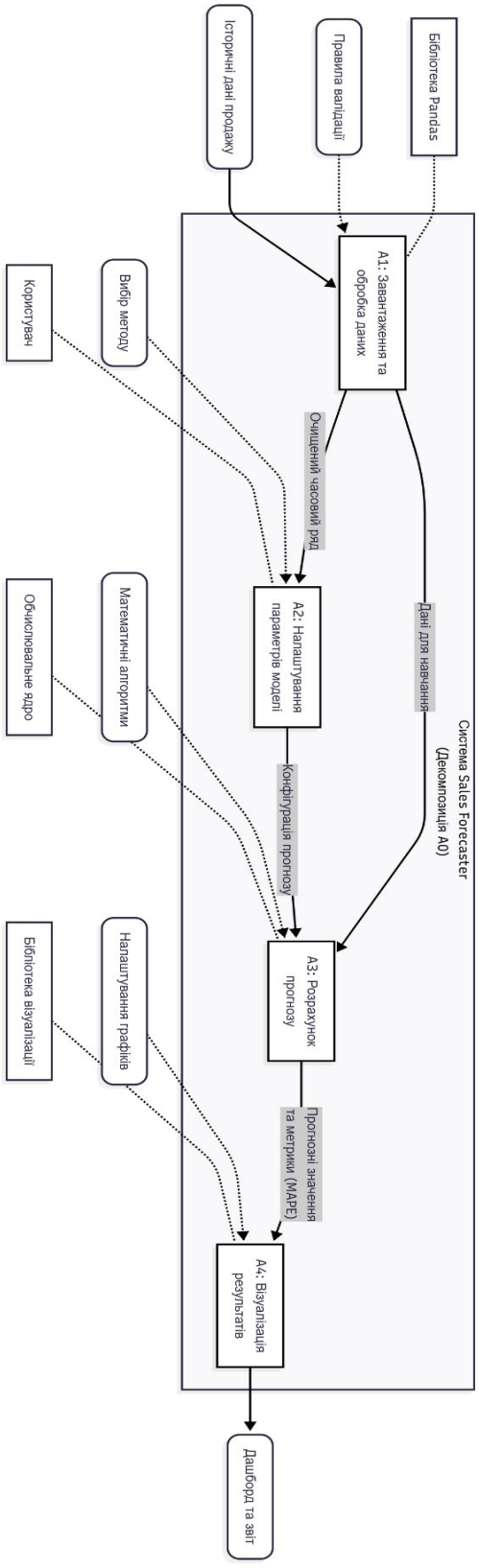

Рисунок А.1 – Контекстна діаграма процесу «Прогнозування продажів торговельної мережі»

Додаток Б. Декомпозиція контекстної діаграми «Прогнозування продажів торгівельної мережі»

- Завантаження та попереднитя обробка даних (Блок A1):
    - Вхід: Сирі історичні дані (CSV-файли).
    - Управління: Формат даних, правила валідації.
    - Механізм: Модуль Data Processing (Pandas), Інтерфейс завантаження.
    - Вихід: Агрегований та очищений часовий ряд.
- Вибір методу та налаштування параметрів (Блок A2):
    - Вхід: Очищений часовий ряд.
    - Управління: Вибір користувача (Декомпозиція/Ковзне середнє), параметри згладжування (α, β).
    - Механізм: Категорійний менеджер (користувач).
    - Вихід: Конфігурація прогнозу.
- Розрахунок коефіцієнтів та генерація прогнозу (Блок A3):
    - Вхід: Очищений часовий ряд, конфігурація.
    - Управління: Алгоритми прогнозування (математичні формули).
    - Механізм: Обчислювальне ядро (Python/Numpy).
    - Вихід: Масив прогнозних значень, розраховані метрики помилок (MAPE, RMSE).
- Візуалізація та експорт результатів (Блок A4):
    - Вхід: Прогнозні значення, метрики помилок.
    - Управління: Шаблони звітів, налаштування відображення графіків.
    - Механізм: Модуль візуалізації (Plotly/Matplotlib).
    - Вихід: Дашборд, графіки сезонності, звіт для закупівлі.

Додаток В. Приклад реалізації методів прогнозування на python

```
import pandas as pd
import numpy as np
import warnings

warnings.filterwarnings('ignore')

MONTH_NAMES = {
    1: 'Jan', 2: 'Feb', 3: 'Mar', 4: 'Apr', 5: 'May', 6: 'Jun',
    7: 'Jul', 8: 'Aug', 9: 'Sep', 10: 'Oct', 11: 'Nov', 12: 'Dec'
}

def get_freq_params(freq):
    """Return group column and divisor based on frequency."""
    if freq == 'month':
        return 'month', 12
    else:
        return 'week', 52

# --- DECOMPOSITION METHODS ---

def calculate_seasonality_ratios(df, start_year, end_year, freq='month'):
    """
    Calculate seasonality ratios based on historic data (Standard Decomposition).
    """
    historic = df[(df['year'] >= start_year) & (df['year'] <= end_year)].copy()
    group_col, _ = get_freq_params(freq)

    # Calculate Overall Avg per category
    overall_avg = historic.groupby('category_name')['quantity_sold'].mean().reset_index()
    overall_avg.columns = ['category_name', 'Overall_Avg']

    # Calculate periodic averages per category
periodic_avg=historic.groupby([group_col, 'category_name'])['quantity_sold'].mean().reset_index()
    periodic_avg.columns = [group_col, 'category_name', 'periodic_avg_quantity']

    # Merge
    ratios = periodic_avg.merge(overall_avg, on='category_name', how='left')
    ratios['ratio'] = ratios['periodic_avg_quantity'] / ratios['Overall_Avg']
    ratios['ratio'] = ratios['ratio'].fillna(0).replace([np.inf, -np.inf], 0)

    if freq == 'month':
        ratios['Month_name'] = ratios['month'].map(MONTH_NAMES)

    return ratios[[group_col, 'category_name', 'ratio'] + (['Month_name'] if freq == 'month' else
[])]

def calculate_trend_linear(df, start_year, end_year, forecast_year):
    """Calculate linear trend factor."""
    historic = df[(df['year'] >= start_year) & (df['year'] <= end_year)].copy()
    yearly_totals = historic.groupby(['year', 'category_name'])['value_sold'].sum().reset_index()
    trend_factors = []
```

```
    for category in yearly_totals['category_name'].unique():
        cat_data = yearly_totals[yearly_totals['category_name'] == category].sort_values('year')
        if len(cat_data) < 2:
            trend_factors.append({'category_name': category, 'trend_factor': 1.0})
            continue

        years = cat_data['year'].values
        values = cat_data['value_sold'].values

        # Linear Regression
        n = len(years)
        sum_x = np.sum(years)
        sum_y = np.sum(values)
        sum_xy = np.sum(years * values)
        sum_x2 = np.sum(years ** 2)

        denominator = (n * sum_x2 - sum_x ** 2)
        if denominator == 0:
            slope = 0
        else:
            slope = (n * sum_xy - sum_x * sum_y) / denominator

        intercept = (sum_y - slope * sum_x) / n

        projected = slope * forecast_year + intercept
        last_actual = values[-1]

        factor = projected / last_actual if last_actual > 0 else 1.0
        factor = max(0.5, min(2.0, factor)) # Cap

        trend_factors.append({'category_name': category, 'trend_factor': factor})

    return pd.DataFrame(trend_factors)

# --- MOVING AVERAGE METHODS ---

def calculate_moving_average_seasonality(df, start_year, end_year, freq='month', window=2):
    """
    Calculate seasonality ratios using Moving Average method.
    Applies moving average to periodic values across years, then calculates ratios.
    """
    historic = df[(df['year'] >= start_year) & (df['year'] <= end_year)].copy()
    group_col, period_count = get_freq_params(freq)

    seasonality_ratios = []

    for category in historic['category_name'].unique():
      cat_data = historic[historic['category_name'] == category].sort_values(['year', group_col])

        # Get periodic values for all years
        periodic_values = {}
        periods = range(1, period_count + 1)

        for p in periods:
            p_data = cat_data[cat_data[group_col] == p]['value_sold'].values
            periodic_values[p] = p_data
```

```
        # Apply moving average to smooth periodic values across years
        smoothed_periodic = {}
        for p in periods:
            values = periodic_values[p]
            if len(values) >= window:
                # Moving average across years
                ma_values = []
                for i in range(len(values) - window + 1):
                    ma = np.mean(values[i:i+window])
                    ma_values.append(ma)
                smoothed_periodic[p] = np.mean(ma_values) if ma_values else np.mean(values)
            else:
                smoothed_periodic[p] = np.mean(values) if len(values) > 0 else 0

        # Calculate overall average of smoothed values
        overall_avg = np.mean(list(smoothed_periodic.values()))

        # Calculate ratios
        for p in periods:
            ratio = smoothed_periodic[p] / overall_avg if overall_avg > 0 else 0
            seasonality_ratios.append({
                group_col: p,
                'category_name': category,
                'ratio': ratio
            })

    ratios_df = pd.DataFrame(seasonality_ratios)
    if freq == 'month':
        ratios_df['Month_name'] = ratios_df['month'].map(MONTH_NAMES)
    return ratios_df

def calculate_moving_average_trend(df, start_year, end_year, freq='month', window=2):
    """
    Calculate trend factor using Moving Average method on yearly totals.
    """
    historic = df[(df['year'] >= start_year) & (df['year'] <= end_year)].copy()
    yearly_totals = historic.groupby(['year', 'category_name'])['value_sold'].sum().reset_index()
    trend_factors = []

    for category in yearly_totals['category_name'].unique():
        cat_data = yearly_totals[yearly_totals['category_name'] == category].sort_values('year')

        if len(cat_data) < 2:
            trend_factors.append({'category_name': category, 'trend_factor': 1.0})
            continue

        values = cat_data['value_sold'].values

        # Calculate moving averages
        ma_values = []
        for i in range(len(values) - window + 1):
            ma = np.mean(values[i:i+window])
            ma_values.append(ma)

        # Calculate trend from moving averages
```

```
        if len(ma_values) >= 2:
            trend = ma_values[-1] / ma_values[-2] if ma_values[-2] > 0 else 1.0
        else:
            trend = values[-1] / values[0] if values[0] > 0 else 1.0

        trend_factor = max(0.5, min(2.0, trend))

        trend_factors.append({
            'category_name': category,
            'trend_factor': trend_factor
        })

    return pd.DataFrame(trend_factors)

# --- EXPONENTIAL SMOOTHING METHODS ---

def calculate_exponential_smoothing_seasonality(df, start_year, end_year, freq='month',
alpha=0.3):
    """
    Calculate seasonality ratios using Exponential Smoothing method.
    Applies exponential smoothing to periodic values across years.
    """
    historic = df[(df['year'] >= start_year) & (df['year'] <= end_year)].copy()
    group_col, period_count = get_freq_params(freq)

    seasonality_ratios = []

    for category in historic['category_name'].unique():
      cat_data = historic[historic['category_name'] == category].sort_values(['year', group_col])

        # Get periodic values for all years
        periodic_values = {}
        periods = range(1, period_count + 1)

        for p in periods:
            p_data = cat_data[cat_data[group_col] == p]['value_sold'].values
            periodic_values[p] = p_data

        # Apply exponential smoothing across years
        smoothed_periodic = {}
        for p in periods:
            values = periodic_values[p]
            if len(values) > 0:
                # Exponential smoothing
                smoothed = values[0]
                for i in range(1, len(values)):
                    smoothed = alpha * values[i] + (1 - alpha) * smoothed
                smoothed_periodic[p] = smoothed
            else:
                smoothed_periodic[p] = 0

        # Calculate overall average of smoothed values
        overall_avg = np.mean(list(smoothed_periodic.values()))

        # Calculate ratios
        for p in periods:
```

```
            ratio = smoothed_periodic[p] / overall_avg if overall_avg > 0 else 0
            seasonality_ratios.append({
                group_col: p,
                'category_name': category,
                'ratio': ratio
            })

    ratios_df = pd.DataFrame(seasonality_ratios)
    if freq == 'month':
        ratios_df['Month_name'] = ratios_df['month'].map(MONTH_NAMES)
    return ratios_df

def calculate_exponential_smoothing_trend(df, start_year, end_year, freq='month', alpha=0.3):
    """
    Calculate trend factor using Exponential Smoothing method on yearly totals.
    """
    historic = df[(df['year'] >= start_year) & (df['year'] <= end_year)].copy()
    yearly_totals = historic.groupby(['year', 'category_name'])['value_sold'].sum().reset_index()
    trend_factors = []

    for category in yearly_totals['category_name'].unique():
        cat_data = yearly_totals[yearly_totals['category_name'] == category].sort_values('year')

        if len(cat_data) < 2:
            trend_factors.append({'category_name': category, 'trend_factor': 1.0})
            continue

        values = cat_data['value_sold'].values

        # Exponential smoothing
        smoothed = values[0]
        for i in range(1, len(values)):
            smoothed = alpha * values[i] + (1 - alpha) * smoothed

        # Calculate trend from last smoothed value vs first
        if values[0] > 0:
            trend_factor = smoothed / values[0]
        else:
            trend_factor = 1.0

        trend_factor = max(0.5, min(2.0, trend_factor))

        trend_factors.append({
            'category_name': category,
            'trend_factor': trend_factor
        })

    return pd.DataFrame(trend_factors)

# --- MAIN GENERATION FUNCTION ---

def generate_forecast(df, train_start, train_end, forecast_year, freq='month'):
    """
    Generate forecasts for a specific year using multiple methods.
    """
    group_col, divisor = get_freq_params(freq)
```

```
    # 1. Calculate Base (Average of last training year)
    last_train_year = df[df['year'] == train_end].copy()
    base_calc = last_train_year.groupby('category_name')['value_sold'].sum().reset_index()
    base_calc['Base'] = base_calc['value_sold'] / divisor
    base_calc = base_calc[['category_name', 'Base']]

    # --- Method 1: Decomposition (No Trend) ---
    ratios_decomp = calculate_seasonality_ratios(df, train_start, train_end, freq)

    m1 = ratios_decomp.merge(base_calc, on='category_name', how='left')
    m1['forecast_decomposition'] = m1['Base'] * m1['ratio']
    m1['forecast_decomposition'] = m1['forecast_decomposition'].fillna(0)

    # --- Method 2: Decomposition (Linear Trend) ---
    trend_linear = calculate_trend_linear(df, train_start, train_end, forecast_year)

    m2 = m1.merge(trend_linear, on='category_name', how='left')
    m2['trend_factor'] = m2['trend_factor'].fillna(1.0)
    m2['forecast_decomposition_linear'] = m2['Base'] * m2['trend_factor'] * m2['ratio']
    m2['forecast_decomposition_linear'] = m2['forecast_decomposition_linear'].fillna(0)

    # --- Method 3: Moving Average ---
    ratios_ma = calculate_moving_average_seasonality(df, train_start, train_end, freq, window=2)
    trend_ma = calculate_moving_average_trend(df, train_start, train_end, freq, window=2)

    # We need to merge back to a clean slate or handle the merge carefully
    # Let's create a separate dataframe for MA and then merge
    m3_base = ratios_ma.merge(base_calc, on='category_name', how='left')
    m3_base = m3_base.merge(trend_ma, on='category_name', how='left')
    m3_base['trend_factor'] = m3_base['trend_factor'].fillna(1.0)
    m3_base['forecast_MovAvg'] = m3_base['Base'] * m3_base['trend_factor'] * m3_base['ratio']
    m3_base['forecast_MovAvg'] = m3_base['forecast_MovAvg'].fillna(0)

    # --- Method 4: Exponential Smoothing ---
    ratios_exp = calculate_exponential_smoothing_seasonality(df, train_start, train_end, freq,
alpha=0.3)
   trend_exp = calculate_exponential_smoothing_trend(df, train_start, train_end, freq, alpha=0.3)

    m4_base = ratios_exp.merge(base_calc, on='category_name', how='left')
    m4_base = m4_base.merge(trend_exp, on='category_name', how='left')
    m4_base['trend_factor'] = m4_base['trend_factor'].fillna(1.0)
    m4_base['forecast_ExponentialSmoothing'] = m4_base['Base'] * m4_base['trend_factor'] *
m4_base['ratio']
    m4_base['forecast_ExponentialSmoothing'] = m4_base['forecast_ExponentialSmoothing'].fillna(0)

    # --- Final Merge ---
    # Start with the decomposition result (m2) which has the grid
    final = m2.copy()

    # Merge MA
    final = final.merge(m3_base[[group_col, 'category_name', 'forecast_MovAvg']],
                        on=[group_col, 'category_name'], how='left')

    # Merge Exp
    final = final.merge(m4_base[[group_col, 'category_name', 'forecast_ExponentialSmoothing']],
```

```
                    on=[group_col, 'category_name'], how='left')

    # Final Cleanup
    final_cols = [group_col, 'category_name', 'forecast_decomposition',
              'forecast_decomposition_linear', 'forecast_MovAvg', 'forecast_ExponentialSmoothing']
    if freq == 'month':
        final_cols.insert(1, 'Month_name')

    result = final[final_cols].copy()

    # Add Actuals if available (for comparison)
    actuals = df[df['year'] == forecast_year].copy()
    if not actuals.empty:
        act_col = f'Value_actual_{forecast_year}'
      actuals = actuals[[group_col, 'category_name', 'value_sold']].rename(columns={'value_sold':
act_col})
        result = result.merge(actuals, on=[group_col, 'category_name'], how='left')
        result[act_col] = result[act_col].fillna(0)

    return result

def generate_evaluation_report(df_forecast, actual_year, freq='month'):
    """
    Calculate accuracy metrics for all forecast methods in the dataframe.
    """
    methods = {
        'Decomposition': 'forecast_decomposition',
        'Decomposition (Linear Trend)': 'forecast_decomposition_linear',
        'Moving Average': 'forecast_MovAvg',
        'Exponential Smoothing': 'forecast_ExponentialSmoothing'
    }

    actual_col = f'Value_actual_{actual_year}'
    if actual_col not in df_forecast.columns:
        return None

    results = []

    for method_name, pred_col in methods.items():
        if pred_col not in df_forecast.columns:
            continue

        # Filter valid data (remove zeros/nans)
        mask = (df_forecast[actual_col] > 0) & (df_forecast[pred_col] >= 0)
        valid_data = df_forecast[mask].copy()

        if valid_data.empty:
            continue

        y_true = valid_data[actual_col].values
        y_pred = valid_data[pred_col].values

        # Calculate Metrics
        errors = y_pred - y_true
        abs_errors = np.abs(errors)
        pct_errors = np.abs(errors / y_true) * 100
```

```
        mae = np.mean(abs_errors)
        rmse = np.sqrt(np.mean(errors**2))
        mape = np.mean(pct_errors)

        total_actual = np.sum(y_true)
        total_pred = np.sum(y_pred)
        total_error_pct = ((total_pred / total_actual) - 1) * 100

        results.append({
            'Method': method_name,
            'MAE': mae,
            'RMSE': rmse,
            'MAPE (%)': mape,
            'Total Error (%)': total_error_pct
        })

    return pd.DataFrame(results)
```

Додаток Г. Код функції створення агрегованої таблиці замовлень та товарів

```
"""
Create orders_products_calculated.csv table showing daily product sales
"""

import pandas as pd
from pathlib import Path
from datetime import datetime


def main():
    print("Loading required datasets...")

    # Load orders dataset
    print("1. Loading orders dataset...")
    orders  =  pd.read_csv('data/csv_files/olist_orders_dataset_extended.csv',
low_memory=False)
    print(f"   Loaded {len(orders)} orders")

    # Parse order_purchase_timestamp to get date
    orders['order_purchase_timestamp_dt'] = pd.to_datetime(
        orders['order_purchase_timestamp'],
        format='%d.%m.%Y %H:%M:%S',
        errors='coerce'
    )

    # Extract date in format dd.MM.yyyy
    orders['date']                                                    =
orders['order_purchase_timestamp_dt'].dt.strftime('%d.%m.%Y')

    # Keep only order_id and date
    orders_subset = orders[['order_id', 'date']].copy()

    # Load order items dataset
    print("2. Loading order items dataset...")
    order_items  =  pd.read_csv('data/csv_files/olist_order_items_dataset.csv',
low_memory=False)
    print(f"   Loaded {len(order_items)} order items")

    # Merge orders with order items to get date for each item
    print("3. Merging orders with order items...")
    items_with_dates = order_items.merge(
        orders_subset,
```

```
        on='order_id',
        how='inner'
    )
    print(f"   Merged: {len(items_with_dates)} items with dates")

    # Load products dataset
    print("4. Loading products dataset...")
    products = pd.read_csv('data/csv_files/olist_products_dataset.csv',
low_memory=False)
    print(f"   Loaded {len(products)} products")

    # Merge with products to get category
    print("5. Merging with products to get categories...")
    items_with_categories = items_with_dates.merge(
        products[['product_id', 'product_category_name']],
        on='product_id',
        how='left'
    )
    print(f"   Merged: {len(items_with_categories)} items with categories")

    # Load translation table
    print("6. Loading category translation table...")
    translations =
pd.read_csv('data/csv_files/product_category_name_translation.csv',
low_memory=False)
    print(f"   Loaded {len(translations)} translations")

    # Merge with translations to get English category name
    print("7. Translating categories to English...")
    items_with_translations = items_with_categories.merge(
        translations[['product_category_name',
'product_category_name_english']],
        on='product_category_name',
        how='left'
    )
   print(f"  Merged: {len(items_with_translations)} items with translations")

    # Use English name if available, otherwise use original
    items_with_translations['product_category'] =
items_with_translations['product_category_name_english'].fillna(
        items_with_translations['product_category_name']
    )

    # Group by date and product_id to calculate metrics
    print("8. Calculating daily product sales...")
```

```
    # Group by date and product_id
    daily_products = items_with_translations.groupby(['date', 'product_id']).agg({
        'order_item_id': 'count',  # Number of items sold
        'price': 'first',  # Price per product (should be same for same product_id)
        'product_category': 'first'  # Category (should be same for same product_id)
    }).reset_index()

    # Rename columns
    daily_products.rename(columns={
        'order_item_id': 'number_sold'
    }, inplace=True)

    # Calculate Value (price * number_sold)
    daily_products['Value'] = daily_products['price'] * daily_products['number_sold']

    # Select and reorder columns
    result = daily_products[[
        'date',
        'product_id',
        'product_category',
        'number_sold',
        'Value'
    ]].copy()

    # Sort by date and product_id
    result = result.sort_values(['date', 'product_id']).reset_index(drop=True)

    # Round Value to 2 decimal places
    result['Value'] = result['Value'].round(2)

    # Save to CSV
    output_file = Path("data/csv_files/orders_products_calculated.csv")
    print(f"\n9. Saving to {output_file}...")
    result.to_csv(output_file, index=False)

    print(f"\nComplete!")
    print(f"  Total records: {len(result)}")
    print(f"  Date range: {result['date'].min()} to {result['date'].max()}")
    print(f"  Unique dates: {result['date'].nunique()}")
    print(f"  Unique products: {result['product_id'].nunique()}")
    print(f"  Total value: {result['Value'].sum():.2f}")
```

```
    # Statistics
    print(f"\nStatistics:")
    print(f"         Average     products     per     day:     {len(result)     /
result['date'].nunique():.1f}")
    print(f"         Average     number_sold     per     product     per     day:
{result['number_sold'].mean():.2f}")
    print(f"  Max number_sold in one day: {result['number_sold'].max()}")
   print(f"  Average Value per product per day: {result['Value'].mean():.2f}")
    print(f"  Max Value in one day: {result['Value'].max():.2f}")

    # Sample data
    print(f"\nSample data (first 10 rows):")
    print(result.head(10).to_string(index=False))

    # Check for missing categories
    missing_categories = result['product_category'].isna().sum()
    if missing_categories > 0:
        print(f"\nWarning:    {missing_categories}    records    have    missing
product_category")


if __name__ == "__main__":
    main()
```

Додаток Д. Таблиця порівняння методів прогнозування з фактичними продажами за 2020 рік

Таблиця Д.1 – актуальні та прогнозовані дані

Тиждень	Класична декомпозиція	Лінійна декомпозиція	Метод ковзних середніх	Метод експоненційного згладжування	Фактичні продажі за 2020 рік
1	$217 804,64	$281 195,75	$310 243,62	$310 033,75	$238 166,31
2	$195 034,18	$250 753,23	$287 554,94	$271 537,35	$252 049,38
3	$191 259,24	$246 743,34	$267 901,30	$252 093,85	$271 426,72
4	$194 780,44	$251 128,18	$272 485,57	$270 140,95	$229 867,04
5	$201 917,80	$261 359,93	$260 905,45	$269 252,63	$274 432,52
6	$197 241,79	$254 521,58	$273 222,69	$283 228,12	$240 999,18
7	$197 487,19	$255 347,59	$259 636,28	$268 092,36	$210 625,46
8	$201 208,58	$261 252,35	$261 630,37	$250 547,31	$219 767,59
9	$227 483,91	$294 333,63	$325 171,63	$320 603,20	$252 481,89
10	$223 987,70	$288 935,83	$303 313,90	$312 084,64	$315 920,31
11	$222 818,81	$287 675,93	$300 198,54	$321 796,02	$330 237,89
12	$238 318,46	$308 280,57	$317 950,99	$334 237,22	$257 967,52
13	$217 702,89	$281 455,19	$289 887,65	$314 644,28	$271 280,17
14	$225 397,87	$291 350,64	$321 726,52	$341 423,15	$294 487,07
15	$214 444,78	$277 724,42	$303 324,52	$307 672,62	$267 527,28
16	$229 881,24	$297 842,36	$306 731,08	$321 390,29	$287 223,22
17	$243 412,41	$316 557,19	$329 309,57	$347 155,39	$290 814,63
18	$241 706,62	$314 036,39	$319 866,10	$328 711,26	$306 354,33
19	$271 251,66	$352 872,25	$373 156,85	$373 867,80	$313 023,05
20	$261 124,23	$338 817,83	$348 945,41	$373 446,36	$292 650,53
21	$199 673,95	$257 778,22	$249 357,12	$313 281,70	$279 522,60
22	$209 953,66	$272 810,01	$289 916,28	$337 037,47	$315 053,84
23	$239 780,27	$311 249,10	$314 026,80	$357 230,76	$302 126,88
24	$242 427,42	$313 959,28	$325 515,14	$363 891,62	$365 823,07
25	$237 323,97	$309 029,58	$292 545,83	$331 423,53	$332 691,13
26	$246 721,52	$321 024,03	$317 553,68	$365 822,59	$317 480,63
27	$218 760,93	$283 714,98	$274 385,69	$328 641,16	$347 841,57
28	$223 292,51	$289 714,49	$292 151,10	$353 656,33	$305 321,89

29	$275 542,92	$358 488,85	$345 483,87	$398 970,49	$334 709,39
30	$263 085,87	$342 821,66	$343 926,05	$377 115,40	$367 277,28
31	$285 173,18	$371 859,54	$367 952,06	$395 889,70	$294 151,29
32	$271 590,64	$352 791,83	$367 638,81	$389 341,43	$279 607,90
33	$271 794,58	$354 962,23	$359 876,07	$404 720,35	$282 488,87
34	$223 388,87	$289 168,99	$306 092,84	$346 656,76	$284 264,62
35	$232 860,37	$295 006,24	$317 437,17	$375 566,08	$275 893,68
36	$200 496,62	$257 759,26	$277 672,62	$330 859,61	$289 526,34
37	$228 978,68	$295 890,77	$316 685,52	$376 630,11	$287 271,55
38	$234 626,35	$304 636,24	$334 338,60	$381 879,34	$312 299,48
39	$219 820,09	$283 740,29	$320 993,46	$391 761,07	$379 774,16
40	$220 608,29	$285 437,20	$314 681,82	$368 842,06	$342 203,26
41	$204 713,83	$264 085,25	$264 238,72	$325 750,00	$305 118,98
42	$241 515,46	$312 346,24	$324 240,78	$391 686,44	$295 460,51
43	$233 447,64	$301 109,94	$331 938,12	$372 294,34	$347 729,23
44	$226 879,64	$292 246,85	$313 142,74	$371 658,10	$325 572,29
45	$289 068,46	$375 709,24	$403 643,73	$466 873,35	$368 738,08
46	$261 507,24	$339 020,35	$362 752,75	$428 954,28	$409 728,48
47	$399 182,96	$515 370,33	$495 454,36	$700 990,58	$433 388,30
48	$345 660,98	$447 029,23	$479 954,46	$595 393,28	$430 155,03
49	$321 558,32	$415 322,83	$448 205,25	$526 385,32	$410 035,16
50	$312 460,05	$405 461,26	$446 505,42	$516 196,71	$413 062,44
51	$277 654,12	$362 168,68	$408 377,09	$448 528,53	$446 860,96
52	$243 824,53	$316 432,74	$344 211,20	$371 533,81	$382 300,57
Усього	**$12 517 638,36**	**$16 210 329,90**	**$16 984 058,15**	**$18 977 420,87**	**$16 280 781,55**