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

Навчально–науковий інститут прикладного системного аналізу

Кафедра математичних методів системного аналізу

До захисту допущено:

Завідувач кафедри

_____ Оксана ТИМОЩУК

«___»_____2025 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо–професійною програмою «Системний аналіз і управління»

спеціальності 124 «Системний аналіз»
на тему: «Програмний комплекс для прогнозування попиту на послуги ресторанів»

Виконала: студентка IV курсу, групи КА – 14
Виговська Софія Романівна _____

Керівник:
професор, д.т.н., Мухін Вадим Євгенович _____

Консультант з економічного розділу :
доцент, к.е.н., Рощина Надія Василівна _____

Консультант з нормоконтролю:
к.ф.–м.н., Статкевич Віталій Михайлович _____

Рецензент:
к.т.н., доцент кафедри цифрових технологій в енергетиці
НН ІАТЕ КПІ ім. Ігоря Сікорського,
Шаповалова Світлана Ігорівна _____

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших авторів
без відповідних посилань.
Студентка_____

Київ – 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально–науковий інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 124 «Системний аналіз»

Освітньо–професійна програма «Системний аналіз і управління»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Оксана ТИМОЩУК
«___»_____2025 р.

ЗАВДАННЯ
на дипломну роботу студентці
Виговській Софії Романівні

1. Тема роботи «Програмний комплекс для прогнозування попиту на послуги ресторанів», керівник роботи Мухін Вадим Євгенович, професор, доктор технічних наук, затверджені наказом по університету від 26.05.2025 р. №1759–с.

2. Термін подання студентом роботи _____

3. Вихідні дані до роботи: датасет.

4. Зміст роботи: Розділ 1. Аналіз предметної області, Розділ 2. Розробка концепції програмного комплексу, Розділ 3. Реалізація програмного комплексу, Розділ 4. Функціонально–вартісний аналіз програмного продукту.

5. Перелік ілюстративного матеріалу: презентація.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Надія Василівна доцент, к.е.н.		

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Формулювання теми бакалаврської роботи.	14.04.2025 – 20.04.2025	виконано
2	Пошук та огляд спеціальної літератури за темою дослідження.	21.04.2025 – 27.04.2025	виконано
3	Ознайомлення із літературою та програмним забезпеченням для машинного навчання, необхідними для написання роботи.	28.04.2025 – 04.05.2025	виконано
4	Пошук вхідних даних для аналізу.	05.05.2025 – 11.05.2025	виконано
5	Порівняння методів вирішення задачі оптимізації параметрів, виділення їх недоліків.	12.05.2025 – 18.05.2025	виконано
6	Реалізація програмного комплексу та тестування його роботи.	19.05.2025 – 01.06.2025	виконано
7	Підготовка презентації для захисту.	02.06 – 08.06.2025	виконано
8	Попередній захист дипломної роботи.	09.06.2025 – 15.06.2025	виконано
9	Захист дипломної роботи.	16.06.2025 – 20.06.2025	виконано

Студентка Софія ВИГОВСЬКА

Керівник Вадим МУХІН

РЕФЕРАТ

Дипломна робота: 113 с., 11 рис., 20 табл., 2 дод., 21 дж.

ПРОГНОЗУВАННЯ ПОПИТУ, РЕСТОРАННИЙ БІЗНЕС, ПРОГРАМНИЙ КОМПЛЕКС, МАШИННЕ НАВЧАННЯ, АНАЛІЗ ДАНИХ, СТАТИСТИЧНЕ МОДЕЛЮВАННЯ

Об’єкт дослідження – процеси прогнозування попиту на послуги ресторанів на основі історичних даних та факторів, що впливають на споживчу активність.

Програмний продукт – програмний комплекс для аналізу та прогнозування попиту на ресторанні послуги, що використовує методи машинного навчання та статистичного моделювання, мова програмування – Python (з використанням бібліотек Pandas, NumPy, Scikit–learn, TensorFlow, Prophet).

Мета роботи – розробка програмного комплексу для прогнозування попиту на послуги ресторанів, що дозволить підвищити ефективність управління ресурсами та оптимізувати бізнес–процеси.

Метод дослідження – аналіз та обробка історичних даних, застосування методів машинного навчання, статистичне моделювання, порівняння ефективності алгоритмів прогнозування.

Розроблений програмний продукт дозволяє автоматизовано здійснювати прогнозування попиту на послуги ресторану на основі історичних даних про замовлення, враховуючи часові характеристики, категорії страв, інтенсивність відвідування закладу та інші зовнішні фактори.

Предмет дослідження– процеси аналізу та прогнозування попиту в ресторанному бізнесі за допомогою програмних засобів і методів інтелектуального аналізу даних.

ABSTRACT

Thesis includes: 113 pages, 11 figures, 20 tables, 2 appendices, 21 references.

DEMAND FORECASTING, RESTAURANT BUSINESS, SOFTWARE COMPLEX, MACHINE LEARNING, DATA ANALYSIS, STATISTICAL MODELING.

Object of research – the processes of demand forecasting for restaurant services based on historical data and factors influencing consumer activity.

Software product – a software complex for analyzing and forecasting demand for restaurant services using machine learning and statistical modeling methods.

Programming language – Python (with the use of Pandas, NumPy, Scikit–learn, TensorFlow, Prophet libraries).

Purpose of the study – development of a software complex for demand forecasting in the restaurant industry to improve resource management efficiency and optimize business processes.

Research method – analysis and processing of historical data, application of machine learning methods, statistical modeling, and comparison of forecasting algorithm efficiency.

The developed software product allows for automated forecasting of demand for restaurant services based on historical order data, taking into account time–related features, dish categories, customer visit intensity, and other external factors.

The subject of the study is the analysis and forecasting of demand in the restaurant industry using software tools and data mining methods.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ 8
ВСТУП 9
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 12
1.1. Сучасні методи прогнозування попиту в ресторанному бізнесі . 12
1.2. Фактори, що впливають на попит у ресторанах 22
1.3. Огляд існуючих рішень для прогнозування попиту 25
Висновки до розділу 1 28
РОЗДІЛ 2. РОЗРОБКА КОНЦЕПЦІЇ ПРОГРАМНОГО КОМПЛЕКСУ . . . 30
2.1. Визначення вимог до програмного комплексу 30
2.2. Архітектура системи та вибір технологій 33
2.3. Алгоритми та методи прогнозування 35
Висновки до розділу 2 42
РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО КОМПЛЕКСУ 44
3.1. Опис структури та компонентів системи 44
3.2. Інтерфейс користувача 49
3.3. Інтеграція з іншими інформаційними системами 51
Висновки до розділу 3 55
РОЗДІЛ 4. ФУНКЦІОНАЛЬНО–ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ 57
4.1 Постановка задачі проектування 57
4.2 Обґрунтування функцій програмного продукту 59
4.3 Обґрунтування системе параметров програмного продукту 62
4.4 Аналіз експертного оцінювання параметрів 65
4.5 Аналіз рівня якості варіантів реалізації функцій 69
4.6 Економічний аналіз варіантів розробки ПП 71
4.7 Вибір кращого варіанту ПП техніко–економічного рівня 76
Висновки до розділу 4 77
ВИСНОВКИ 79
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 82
ДОДАТОК А 86

ДОДАТОК Б. ПРЕЗЕНТАЦІЯ 108

ПЕРЕЛІК СКОРОЧЕНЬ

AI – Artificial Intelligence (Штучний інтелект)

ML – Machine Learning (Машинне навчання)

DL – Deep Learning (Глибоке навчання)

NN – Neural Networks (Нейронні мережі)

API – Application Programming Interface (Прикладний програмний інтерфейс)

SQL – Structured Query Language (Мова структурованих запитів)

DB – Database (База даних)

CSV – Comma–Separated Values (Файл, розділений комами)

RMSE – Root Mean Square Error (Корінь середньоквадратичної помилки)

MAE – Mean Absolute Error (Середня абсолютна помилка)

MSE – Mean Squared Error (Середньоквадратична помилка)

ARIMA – AutoRegressive Integrated Moving Average (Автоматично–регресивна інтегрована модель ковзного середнього)

LSTM – Long Short–Term Memory (Довга короткострокова пам’ять)

RNN – Recurrent Neural Network (Рекурентна нейронна мережа)

Pandas – Python Data Analysis Library (Бібліотека аналізу даних у Python)

NumPy – Numerical Python (Бібліотека для роботи з масивами в Python)

Scikit–learn – Бібліотека для машинного навчання в Python

TensorFlow – Фреймворк для глибокого навчання

Prophet – Бібліотека для прогнозування часових рядів

BI – Business Intelligence (Бізнес–аналітика)

KPI – Key Performance Indicator (Ключовий показник ефективності)

SaaS – Software as a Service (Програмне забезпечення як послуга)

IoT – Internet of Things (Інтернет речей)

JSON – JavaScript Object Notation (Формат обміну даними)

ETL – Extract, Transform, Load (Процес вилучення, трансформації та завантаження даних)

ВСТУП

Сучасний ресторанний бізнес стикається зі значними викликами через нестабільність попиту, вплив сезонних факторів, зміну споживчих уподобань та зовнішніх умов (економічних, соціальних, погодних тощо). Ефективне прогнозування попиту на послуги ресторанів дозволяє оптимізувати використання ресурсів, зменшити витрати та покращити якість обслуговування клієнтів. Використання методів машинного навчання та статистичного моделювання відкриває нові можливості для точного прогнозування, що робить цю тему актуальною та затребуваною в ресторанному бізнесі.

Мета дослідження – розробка програмного комплексу для прогнозування попиту на послуги ресторанів, що дозволить підвищити ефективність управління ресурсами, оптимізувати бізнес–процеси та покращити якість обслуговування клієнтів.

Завдання дослідження:

1) провести аналіз існуючих методів прогнозування попиту у сфері ресторанного бізнесу;
2) визначити основні фактори, що впливають на рівень попиту на послуги ресторанів;
3) зібрати, підготувати та проаналізувати історичні дані для побудови прогнозних моделей;
4) розробити програмний комплекс, що використовує методи машинного навчання для прогнозування попиту;
5) провести тестування ефективності моделей та оцінити їхню точність;
6) надати рекомендації щодо впровадження програмного комплексу у ресторанний бізнес.

Об’єкт дослідження – процес прогнозування попиту на послуги ресторанів на основі аналізу історичних даних та зовнішніх факторів, що впливають на поведінку споживачів.

Предмет дослідження – методи та алгоритми прогнозування попиту, а також розроблений програмний комплекс для їхньої реалізації.

Методи дослідження:

1) аналіз літературних джерел та огляд існуючих підходів до прогнозування попиту;

2) обробка та аналіз історичних даних ресторанного бізнесу;

3) методи машинного навчання (лінійна регресія, нейронні мережі, ARIMA, Prophet тощо);

4) статистичне моделювання та прогнозування часових рядів;

5) оцінка ефективності прогнозних моделей на основі метрик точності (MAE, MSE, RMSE).

Новизна дослідження полягає у розробці програмного комплексу, що застосовує сучасні методи машинного навчання та статистичного моделювання для прогнозування попиту на ресторанні послуги. На відміну від традиційних підходів, запропонована система дозволяє враховувати широкий спектр зовнішніх факторів (сезонність, святкові дні, маркетингові активності, погодні умови) та автоматично адаптуватися до змін у поведінці споживачів.

Практична значимість дипломної роботи полягає в тому що був розроблений програмний комплекс який може бути впроваджений у ресторанному бізнесі для покращення процесів управління запасами, оптимізації персоналу та підвищення загальної ефективності підприємства. Точне прогнозування попиту допоможе уникнути надлишкових витрат та забезпечити своєчасне обслуговування клієнтів, що сприятиме підвищенню прибутковості бізнесу та рівня задоволеності споживачів.

Дипломна робота складається з переліку скорочень, вступу, чотирьох розділів, висновків, переліку джерел посилання та двох додатків.

Вступ – містить обґрунтування актуальності теми, формулювання мети, завдань, об’єкта та предмета дослідження.

Розділ 1. Теоретичні основи прогнозування попиту – розглядає основні підходи до прогнозування, фактори, що впливають на попит у ресторанному бізнесі, та аналізує існуючі методи прогнозування.

Розділ 2. Розробка програмного комплексу – описує процес збору та обробки даних, реалізацію алгоритмів прогнозування та структуру розробленого програмного комплексу.

Розділ 3. Аналіз ефективності прогнозної моделі – містить тестування розробленої системи, аналіз отриманих результатів, порівняння ефективності різних алгоритмів прогнозування та рекомендації щодо впровадження.

Розділ 4. Функціонально-вартісний аналіз програмного продукту – присвячений економічному обґрунтуванню доцільності впровадження розробленого програмного комплексу. У розділі наведено постановку задачі проєктування, визначено та оцінено ключові функції продукту, обґрунтовано систему параметрів, виконано експертне оцінювання їх вагомості. Здійснено аналіз рівня якості реалізації функцій, розраховано технічні та економічні показники, порівняно альтернативні варіанти реалізації. На основі отриманих результатів обрано оптимальний варіант з урахуванням техніко-економічного рівня.

Висновки – підсумовують результати дослідження та окреслюють перспективи подальшого розвитку теми.

Перелік джерел посилання – містить перелік літератури, що використовувалася під час виконання роботи.

Додатки – включають додаткові матеріали, графіки, таблиці та фрагменти коду.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Сучасні методи прогнозування попиту в ресторанному бізнесі

Прогнозування попиту на послуги ресторанів є важливим завданням для ефективного управління бізнес–процесами, оптимізації використання ресурсів та підвищення рівня обслуговування клієнтів. Різні методи прогнозування можуть допомогти ресторанним підприємствам передбачити зміну відвідуваності, адаптувати меню, коригувати закупівлі інгредієнтів та планувати роботу персоналу. Сучасні методи прогнозування можна поділити на традиційні статистичні підходи та методи, засновані на машинному навчанні.

Розглянемо традиційні статистичні методи.

1) Метод ковзного середнього (Moving Average, MA) — це один із найпростіших і найпоширеніших методів прогнозування часових рядів. Він використовується для згладжування часових рядів. Його суть полягає в усередненні значень за певний період часу, що дозволяє згладити коливання та виявити загальні тенденції в даних [1].

При кожному новому кроці найстаріше значення видаляється, а нове додається, що забезпечує динамічне оновлення прогнозу.

Формула ковзного середнього для інтервалу N виглядає так:

$$\mathrm{M}A_t = \frac{X_t + X_{t-1} + \ldots + X_{t-N+1}}{N}$$

де:

$\mathrm{M}A_t$ — значення ковзного середнього в момент часу t

$X_t, X_{t-1}, \ldots, X_{t-N+1}$ — фактичні значення за останні N періодів

N — кількість періодів, за якими обчислюється середнє.

Існує кілька основних варіантів методу: просте ковзне середнє, зважене та експоненційне.

2) Просте ковзне середнє (Simple Moving Average, SMA) розраховується як середнє арифметичне останніх N значень, і дає хороший ефект згладжування, але реагує на зміну тренду із запізненням.

Приклад**.**

Припустимо, що у нас є такі значення попиту на ресторанні послуги за останні 5 днів: 120,130,125,135,140.

Тоді просте ковзне середнє для останніх 3 днів буде:

SMA = $\frac{125+135+140}{3}$= 133.3

3) Зважене ковзне середнє (Weighted Moving Average, WMA) відрізняється від SMA тим, що більш новим значенням надається більша вага.

Воно визначається за формулою:

$$\text{WM}A_t=\frac{w_1X_t+w_2X_{t-1}+\ldots+w_NX_{t-N+1}}{w_1+w_2+\ldots+w_N}$$

де $w_1, w_2,\ldots,w_N$ — вагові коефіцієнти, що зменшуються в міру віддаленості від поточного моменту.

4) Експоненційне ковзне середнє (Exponential Moving Average, EMA) враховує всі попередні значення ряду, але зі спадною вагою, і використовується в тих випадках, коли потрібно швидше реагувати на зміни в тенденціях[2].

Формула EMA:

$$\text{EM}A_t=\lambda X_t + (1-\lambda)\ \text{EM}A_{t-1}$$

де λ — коефіцієнт згладжування, зазвичай вибирається в межах 0.1−0.30.

В таблиці 1.1 наведено переваги та недоліки методу:

Таблиця 1.1. Переваги та недоліки методу

Переваги	Недоліки
Простота реалізації та розрахунку	Погано працює при різких змінах у даних.
Добре згладжує випадкові коливання даних.	Втрачає частину інформації через усереднення.
Добре згладжує випадкові коливання даних.	Має запізнення у виявленні нових трендів (особливо SMA).

Метод ковзного середнього може використовуватися для прогнозування попиту на послуги ресторанів у таких випадках:

1) прогнозування відвідуваності (наприклад, середній попит на вечерю протягом останніх 7 днів);
2) оптимізація закупівель (визначення середнього споживання певних продуктів);
3) аналіз сезонних трендів (наприклад, зміни попиту залежно від пори року).

Таким чином, метод ковзного середнього є базовим інструментом прогнозування, який може слугувати основою для складніших алгоритмів, зокрема моделей машинного навчання.

5) Автокореляція — це залежність поточного значення ряду від попередніх значень. Якщо існує така закономірність, це означає, що значення X_t можна передбачити на основі попередніх значень $X_{t-1}, X_{t-2}, \ldots$

Наведемо приклад використання в ресторанному бізнесі: якщо попит на ресторанні послуги залежить від попередніх днів (наприклад, якщо вчора ресторан відвідало багато клієнтів, то сьогодні ймовірність високого попиту теж зростає), то модель AR може допомогти визначити цю залежність та спрогнозувати попит [4].

Прогнозування часових рядів у ресторанному бізнесі часто базується на аналізі залежностей між попередніми та поточними значеннями попиту. Для

цього, окрім розглянутої моделі автокореляції (AR), широко застосовуються автокореляційно–ковзного середнього (ARMA), які дозволяють будувати моделі часових рядів з урахуванням сезонності та трендів у попиті [3].

Авторегресійна модель (AR) порядку p описується рівнянням:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + e_t$$

де: X_t — поточне значення часової серії,

$\phi_1, \phi_2, .., \phi_p$—коефіцієнти авторегресії,

p — порядок моделі (кількість попередніх значень, які враховуються),

e_t— випадковий шум або похибка.

6) Автокореляційно–ковзне середнє (Autoregressive Moving Average Model, ARMA) – це більш складні моделі, що поєднують авторегресію та метод ковзного середнього, дозволяють будувати точніші прогнози. Модель складається з двох компонентів:

1) AR (авторегресія) — враховує залежність між поточними і попередніми значеннями ряду;

2) MA (ковзне середнє) — враховує залежність між поточними значеннями і попередніми похибками прогнозу.

Формула моделі ARMA(p,q) виглядає так:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \ldots + \theta_q e_{t-q} + e_t$$

де

$\phi_1 \ldots \phi_p$— коефіцієнти авторегресії,

$\theta_1, \ldots, \theta_p$ — коефіцієнти ковзного середнього,

p — порядок AR–частини,

q — порядок MA–частини,

e_t — випадковий шум.

У ресторанному бізнесі модель ARMA може враховувати не лише вплив попередніх рівнів попиту, але й похибки у прогнозах. Це дозволяє отримати точніші прогнози.

7) Модель ARIMA (Autoregressive Integrated Moving Average Model) відрізняється від моделі ARMA наявністю інтегрованого ковзного середнього. Хоча класична модель ARMA підходить лише для стаціонарних часових рядів, тобто таких, що не мають тренду або сезонних змін, проте попит на ресторанні послуги часто змінюється в залежності від часу доби, днів тижня або сезонності.

Щоб зробити ряд стаціонарним, використовується процедура диференціювання — віднімання попереднього значення ряду від поточного:

$$X_t^{`} = X_t - X_{t-1}$$

Якщо одного диференціювання недостатньо, його можна застосувати кілька разів.

Модель ARIMA (p, d, q) включає три параметри:

p — порядок авторегресії (AR),

d — кількість диференціювань для досягнення стаціонарності,

q — порядок ковзного середнього (MA).

Рівняння моделі має вид:

$$X_t^{`} = \phi_1 X_{t-1}^{`} + \phi_2 X_{t-2}^{`} + \ldots + \phi_p X_{t-p}^{`} + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \ldots + \theta_q e_{t-q} + e_t$$

Якщо попит у ресторані має зростаючий тренд (наприклад, через рекламу або зміну меню), модель ARIMA допоможе врахувати цей тренд і зробити коригований прогноз.

ARIMA може бути використана для врахування сезонності (наприклад, зниження попиту влітку або збільшення у вихідні дні).

Щоб визначити, яка модель найкраще підходить, використовують такі методи оцінки:

1) автокореляційна функція (ACF) — допомагає виявити, наскільки поточні значення пов’язані з попередніми;
2) часткова автокореляційна функція (PACF) — показує, як кожен окремий лаг (затримка) впливає на поточне значення;
3) критерій інформації Акаіке (AIC) — дозволяє вибрати модель з найкращим балансом між точністю та складністю.

Далі в таблиці 1.2 наведено порівняльну характеристику моделей AR, ARMA та ARIMA.

Таблиця 1.2. Порівняння моделей AR, ARMA та ARIMA

Характеристика	AR	ARMA	ARIMA
Ураховує попередні значення ряду	+	+	+
Ураховує попередні похибки прогнозу	–	+	+
Може працювати зі стаціонарними рядами	+	+	+
Може працювати з нестабільними рядами (трендом, сезонністю)	–	–	+
Використовується для короткострокових прогнозів	+	+	+
Використовується для довгострокових прогнозів	–	–	+

Модель AR підходить для аналізу даних, де попит ресторану залежить лише від минулих значень.

Модель ARMA додає компонент MA, що дозволяє враховувати випадкові відхилення у прогнозах.

Модель ARIMA є найпотужнішою, оскільки дозволяє працювати з трендами та сезонними змінами в даних.

У ресторанному бізнесі моделі ARIMA та її похідні (наприклад, SARIMA для сезонного прогнозування) є ефективним інструментом для оптимізації запасів, управління персоналом та маркетингових кампаній[4].

8) Експоненціальне згладжування (Exponential Smoothing) або метод Холта–Вінтерса (Holt–Winters) ефективний метод для прогнозування попиту з урахуванням сезонних коливань. Одна з найефективніших варіацій цього методу — метод Холта–Вінтерса (Holt–Winters Exponential Smoothing), який дозволяє прогнозувати часові ряди з трендом і сезонністю.

Оновлення оцінки рівня ряду на основі минулих значень виконується із додаванням трьох додаткових компонент:

1) рівень (level) — базове значення попереднього періоду;

2) тренд (trend) — швидкість зміни рівня з часом;
3) сезонність (seasonality) — повторювані патерни змін у часі (наприклад, щоденна, щотижнева або щомісячна сезонність).

Метод має дві основні варіації:

1) адитивна модель (для випадків, коли амплітуда сезонності залишається сталою);
2) мультиплікативна модель (для випадків, коли амплітуда сезонності змінюється пропорційно рівню ряду).

Метод базується на рекурсивних рівняннях для оновлення оцінок рівня, тренду та сезонності.

Розглянемо оновлення рівня (Base Level, L_t):

$$\boldsymbol{L_t} = \alpha\frac{\boldsymbol{X_t}}{\boldsymbol{S_{t-m}}}+(1-\alpha)(\boldsymbol{L_{t-1}}+\boldsymbol{T_{t-1}})$$

де:

X_t– фактичне значення ряду у момент часу t,

S_{t-m} – сезонний коефіцієнт для поточного сезону (m — довжина сезону),

α – параметр згладжування рівня ($0 < \alpha < 1$)

Далі розгялнемо оновлення тренду (Trend, T_t):

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

де β — параметр згладжування тренду ($0 < \beta < 1$).

Нарешті розглянемо оновлення сезонності (Seasonality, S_t):

$$S_t = \gamma\frac{X_t}{L_t}+(1-\gamma)\,S_{t-m}$$

де γ − параметр згладжування сезонності ($0 < \gamma < 1$)

Прогнозування майбутніх значень

Прогноз для h–го майбутнього періоду розраховується за формулою:

$$\hat{X}_{t+h} = (L_t + hT_t)\,S_{t+h-m}$$

Де $\hat{X}_{t+h}$ – прогнозоване значення.

Якщо сезонність адитивна, то додаємо сезонний компонент

$$\hat{X}_{t+h} = L_t + hT_t + S_{t+h-m}$$

Якщо сезонність мультиплікативна, то сезонний компонент множиться як показано в формулі (11).

Типи експоненціального згладжування: Просте експоненціальне згладжування (SES, Simple Exponential Smoothing)

Використовується для прогнозування стаціонарних рядів без тренду і сезонності.

$$L_t = \alpha X_t + (1 - \alpha) L_{t-1}$$

Найчастіше застосовується для короткострокових прогнозів.

Модель Холта (Holt's Linear Trend Model)

Використовується для часових рядів із трендом, але без сезонності.

Використовує дві компоненти: рівень і тренд.

Модель Холта–Вінтерса (Holt–Winters Exponential Smoothing)

Використовується для рядів із трендом та сезонністю.

Має два варіанти: Адитивний — для фіксованої сезонності та мультиплікативний — для змінної сезонності.

Метод Холта–Вінтерса може бути корисним для прогнозування попиту в ресторанах, враховуючи щоденні, тижневі або місячні сезонні коливання. Якщо ресторан помічає, що попит змінюється залежно від дня тижня (наприклад, більше клієнтів у п'ятницю та суботу, менше у вівторок), модель Holt–Winters допоможе врахувати цей патерн і точно прогнозувати відвідуваність. У літні місяці ресторани біля пляжів можуть мати підвищений попит, а в зимові — зниження. Використання мультиплікативної моделі Holt–Winters дозволить врахувати цей ефект і оптимізувати запаси продуктів та персонал [5].

9) Регресійний аналіз — це один із найпоширеніших методів прогнозування, що використовується для встановлення залежності між змінними. Основна його мета — побудувати математичну модель, яка дозволяє передбачати значення залежної змінної (наприклад, попиту на ресторанні послуги) на основі значень однієї або кількох незалежних змінних (наприклад, погоди, дня тижня, святкових періодів тощо).

Регресія може бути лінійною або нелінійною, залежно від характеру залежності між змінними.

10) Розглянемо методи машинного навчання. Лінійна та поліноміальна регресія – широко застосовуються для визначення залежностей між змінними, однак мають обмеження при моделюванні складних залежностей. Це один із найпростіших методів машинного навчання, що використовується для прогнозування. Вона встановлює залежність між змінними у вигляді лінійного рівняння:

$Y= b_0 + b_1X_1 + b_2X_2 +\ldots+b_nX_n +\varepsilon$

де Y — прогнозоване значення (наприклад, кількість відвідувачів ресторану),

X_i — незалежні змінні (наприклад, день тижня, температура, рівень конкуренції),

$b_0,b_1,\ldots,b_n,$— коефіцієнти моделі,

ε — випадкова похибка.

Коли залежність між змінними нелінійна, використовують поліноміальну регресію. Вона розширює лінійну регресію, додаючи степені незалежних змінних:

$Y= b_0 + b_1X + b_2X^2 + b_3X^3+\ldots+b_nX^n +\varepsilon$

Цей метод добре підходить для даних із вигнутими трендами, які не можна адекватно описати лінійною моделлю.

Рішення на основі дерев (Decision Trees, Random Forest, XGBoost)

Методи на основі дерев прийняття рішень широко застосовуються у прогнозуванні попиту через їхню здатність виявляти складні взаємозв’язки між змінними.

Decision Trees (Дерева рішень) — будують ієрархічну структуру, де кожен вузол відповідає за розбиття даних за певною ознакою, а листові вузли містять прогнозовані значення[5].

Random Forest (Випадковий ліс) — об’єднує кілька дерев рішень для підвищення точності прогнозу та зменшення переобучення[5].

XGBoost — одна з найпотужніших моделей, що використовує техніку градієнтного підсилення для покращення точності прогнозування.

Ці методи добре працюють з великими наборами даних, можуть виявляти нелінійні залежності та обробляти категоріальні змінні (наприклад, день тижня або тип ресторану).

Нейронні мережі (LSTM, RNN) для роботи з часовими рядами

Нейронні мережі є потужними інструментами прогнозування, особливо коли йдеться про часові ряди.

Recurrent Neural Networks (RNN) — рекурентні нейромережі використовуються для обробки послідовних даних. Вони можуть «запам’ятовувати» попередні значення, що дозволяє будувати прогнози на основі історичних патернів[5].

Long Short–Term Memory (LSTM) — покращена версія RNN, яка має механізм довготривалої пам’яті. Вона добре працює із часовими рядами, оскільки може зберігати інформацію про події, що відбувалися в далекому минулому, і використовувати її для прогнозування майбутніх значень[6].

Нейронні мережі особливо ефективні для складних даних із нелінійними залежностями, таких як прогнозування попиту в ресторанах, де вплив чинників змінюється з часом.

Прогностичні моделі на основі Facebook Prophet

Facebook Prophet — це сучасний інструмент для прогнозування часових рядів, який поєднує в собі простоту використання та високу точність прогнозів. Prophet дозволяє автоматично виявляти тренди, сезонність і важливі події у даних [6].

Основні особливості:

Підходить для даних із чітко вираженою сезонністю та трендами.

Може обробляти пропущені значення.

Враховує вплив спеціальних подій (наприклад, свят або акційних днів у ресторанах).

Prophet широко використовується у бізнес–аналітиці для автоматичного прогнозування попиту, що дозволяє ефективно управляти ресурсами.

Кластеризація є методом аналізу, що дозволяє згрупувати дані за схожими характеристиками. Для ресторанного бізнесу це може бути корисним у таких випадках:

1) виділення груп клієнтів зі схожими вподобаннями;
2) аналіз географічного розподілу попиту;
3) виявлення періодів із підвищеною або зниженою активністю.

Найпопулярніші методи кластеризації:

K–Means — розбиває дані на k груп на основі їхньої подібності.

DBSCAN — знаходить кластери без попереднього визначення їхньої кількості, що особливо корисно для роботи з реальними даними.

Hierarchical Clustering — створює ієрархічну структуру кластерів, що дозволяє аналізувати взаємозв’язки між групами [6].

Завдяки кластеризації можна визначити основні тенденції попиту, що допомагає у формуванні маркетингових стратегій і прогнозуванні змін у ресторанному бізнесі.

Порівняльний аналіз методів

Для ресторанного бізнесу вибір методу прогнозування залежить від доступності даних, необхідної точності прогнозу та рівня складності моделі. Традиційні методи простіші у реалізації, але менш ефективні для складних задач, тоді як методи машинного навчання можуть обробляти великі обсяги даних і виявляти складні закономірності.

У подальших розділах буде розглянуто застосування цих методів у розробці програмного комплексу для прогнозування попиту на ресторанні послуги.

а. Фактори, що впливають на попит у ресторанах

Попит на ресторанні послуги формується під впливом широкого спектра факторів, які можна розділити на внутрішні (керовані самим рестораном) та зовнішні (які залежать від ринкових умов і соціально–економічної ситуації).

Внутрішні фактори (керовані рестораном)

1) якість обслуговування;

Один із ключових аспектів, що впливає на рішення клієнтів, — рівень сервісу. Відвідувачі очікують не лише смачної їжі, а й приємної атмосфери та належного ставлення персоналу. Розглянемо важливі складові якісного обслуговування.

1) швидкість обслуговування (час очікування замовлення, оперативність офіціантів);
2) компетентність персоналу (знання меню, рекомендації щодо страв);
3) привітність і ввічливість обслуговуючого персоналу;
4) реакція на зауваження та вирішення проблемних ситуацій;
5) заклади, які інвестують у навчання персоналу та вдосконалення сервісу, мають вищий рівень задоволеності клієнтів і більшу ймовірність повторних відвідувань.

Різноманітність та якість страв відіграють визначальну роль у залученні клієнтів. Сучасні ресторани адаптуються до нових харчових тенденцій та вподобань споживачів, включаючи:

1) вегетаріанські, веганські, безглютенові страви;
2) використання локальних та органічних продуктів;
3) включення страв, що відповідають сучасним дієтичним трендам (кето, палео);
4) наявність сезонних меню, що підтримує інтерес відвідувачів [7].

Якісний підбір страв та оновлення меню сприяють залученню нових клієнтів і утриманню постійних.

Також ціна є одним із ключових критеріїв вибору ресторану. Вона має відповідати очікуванням клієнтів щодо якості страв та рівня обслуговування. Ефективні цінові стратегії включають: Гнучку систему знижок та акцій

(щасливі години, бізнес–ланчі, пропозиції для великих компаній). Використання програм лояльності для постійних клієнтів. Стратегію співвідношення ціни та якості, що відповідає цільовій аудиторії [7].

Місцезнаходження ресторану значно впливає на потік відвідувачів. Основні чинники: Близькість до житлових районів, офісних центрів або туристичних локацій. Зручність під'їзду та наявність паркування. Доступність громадського транспорту. Наприклад, ресторани, розташовані поблизу бізнес–центрів, мають високий попит у будні дні під час обідньої перерви, тоді як заклади біля туристичних пам'яток можуть залежати від сезонності.

Ефективне просування ресторану допомагає залучати нових клієнтів. Сучасні маркетингові стратегії включають: Активне ведення соціальних мереж (Instagram, Facebook, TikTok). Партнерство з блогерами та інфлюенсерами. Організацію заходів, тематичних вечорів або дегустацій. Програми лояльності та акційні пропозиції [8].

Без активного маркетингу навіть заклади з чудовим сервісом та кухнею можуть мати недостатній потік клієнтів.

Сучасні технології впливають на зручність та швидкість обслуговування. Популярні технологічні рішення для ресторанів включають: Онлайн–замовлення та доставка через мобільні додатки. Електронні меню та QR–коди для замовлення. Автоматизовані системи бронювання столиків. Використання CRM–систем для аналізу вподобань клієнтів. Автоматизація дозволяє покращити ефективність бізнесу та підвищити рівень задоволеності клієнтів.

Зовнішні фактори (некеровані рестораном)

Погода безпосередньо впливає на потік відвідувачів. Наприклад: У дощові та холодні дні люди частіше замовляють доставку, а відвідуваність знижується. У теплу пору року ресторани з літніми терасами отримують більше клієнтів[8].

Попит у ресторанах коливається залежно від часу: В будні активний попит спостерігається під час обідньої перерви. Вечірній час і вихідні дні

популярні для сімейних вечерь та зустрічей. Заклади, що працюють у нічний час, залучають іншу цільову аудиторію.

Свята та події значно впливають на рівень відвідуваності: підвищений попит спостерігається на Новий рік, День закоханих, 8 березня. Влітку туристичні зони мають більше клієнтів, ніж у зимовий період.

Кількість і рівень інших ресторанів у цьому ж районі можуть як позитивно, так і негативно впливати на попит. Наприклад: велика кількість закладів створює конкурентне середовище, але й приваблює більше відвідувачів. Агресивна цінова політика конкурентів може знизити середній чек ресторану.

Зміни у рівні доходів населення та загальна економічна ситуація впливають на ресторанний бізнес: у періоди економічної кризи клієнти віддають перевагу бюджетним закладам або домашній їжі. Зростання середнього класу та культури ресторанного відвідування сприяє збільшенню попиту.

Сучасні тенденції, такі як усвідомлене споживання, екологічність та здорове харчування, впливають на вибір клієнтів. Ресторани, що адаптуються до цих трендів, мають конкурентну перевагу.

Відгуки в інтернеті та рейтинг на платформах (Google, TripAdvisor) безпосередньо впливають на рішення потенційних клієнтів. Позитивна репутація сприяє залученню нових гостей, тоді як негативні відгуки можуть суттєво зменшити відвідуваність.

Врахування всіх цих факторів дозволяє ресторанам прогнозувати попит, оптимізувати бізнес–процеси та підвищувати ефективність роботи.

b. Огляд існуючих рішень для прогнозування попиту

Прогнозування попиту на ресторанні послуги є важливим завданням для ефективного управління ресурсами, зниження витрат та підвищення прибутковості бізнесу. Сучасні підходи до прогнозування можна поділити на традиційні статистичні методи та методи машинного навчання.

Раніше прогнозування попиту здебільшого базувалося на статистичних підходах, які аналізують історичні дані та виявляють закономірності в зміні попиту. Серед найпопулярніших методів можна виділити:

Метод ковзного середнього використовується для згладжування коливань у часових рядах, що допомагає виявляти загальні тенденції попиту. Він ефективний у короткостроковому прогнозуванні, однак має недолік у вигляді запізнення реакції на різкі зміни попиту [9].

Експоненціальне згладжування (Exponential Smoothing, ES)

Методи експоненціального згладжування, такі як модель Хольта–Вінтерса (Holt–Winters), дозволяють враховувати тренди та сезонні коливання попиту. Вони широко застосовуються для прогнозування відвідуваності ресторанів у різні дні тижня та сезони.

Автокореляційні методи (AR, ARMA, ARIMA)

Моделі ARIMA (AutoRegressive Integrated Moving Average) враховують автокореляцію даних та здатні прогнозувати попит на основі попередніх значень. Вони добре працюють із часовими рядами, що мають стабільні тренди та сезонність.

Методи машинного навчання

З розвитком технологій та збільшенням обсягу даних все більшого поширення набувають методи машинного навчання, які дозволяють будувати точніші прогнози, враховуючи складні взаємозв’язки між різними факторами.

Лінійна та поліноміальна регресія

Лінійна регресія є одним із найпростіших методів прогнозування, який встановлює залежність між попитом та ключовими факторами (день тижня, температура, маркетингові кампанії тощо). Поліноміальна регресія дозволяє

враховувати нелінійні залежності, що робить її більш точною у прогнозуванні складних змін попиту.

Рішення на основі дерев (Decision Trees, Random Forest, Gradient Boosting)

Методи, засновані на деревах рішень (Random Forest, XGBoost, CatBoost), використовуються для аналізу складних залежностей між вхідними змінними та попитом. Вони можуть працювати з великою кількістю параметрів, таких як історичні дані продажів, свята, погода та рівень конкуренції [10].

Нейронні мережі (LSTM, RNN)

Рекурентні нейронні мережі (RNN) та їхні розширені версії, такі як довготривала короткочасна пам’ять (LSTM), широко застосовуються для прогнозування часових рядів. Вони здатні враховувати як довгострокові, так і короткострокові залежності в даних, що дозволяє створювати високоточні прогнози для ресторанного бізнесу [11].

Прогностичні моделі на основі Facebook Prophet

Facebook Prophet — це спеціалізований інструмент для прогнозування часових рядів, розроблений для аналізу бізнес–метрик. Він добре підходить для прогнозування попиту у ресторанах, оскільки враховує сезонність, вихідні дні, аномальні події та можливі тренди.

Кластеризація та аналіз тенденцій

Методи кластеризації, такі як K–Means, DBSCAN, використовуються для сегментації споживачів та визначення основних груп клієнтів із подібними вподобаннями. Це дозволяє ресторанам краще адаптувати своє меню та стратегії маркетингу до потреб різних аудиторій.

Окрім математичних моделей, існує ряд програмних інструментів, які допомагають автоматизувати процес прогнозування попиту.

1) аналітичні платформи (Tableau, Power BI, Google Analytics);

Ці інструменти дозволяють візуалізувати дані та аналізувати тенденції, допомагаючи ресторанам оцінювати зміну попиту та коригувати бізнес–стратегію [12].

2) CRM–системи з функцією прогнозування (Toast, Square, OpenTable);

Сучасні системи управління ресторанами включають алгоритми машинного навчання для аналізу продажів, бронювань та змін у споживчих вподобаннях.

3) хмарні сервіси для аналізу великих даних (AWS Forecast, Google AutoML, Azure AI);

Компанії можуть використовувати хмарні рішення, що автоматизують процес аналізу та прогнозування попиту на основі історичних даних.

Сучасні методи прогнозування попиту у ресторанному бізнесі охоплюють широкий спектр підходів — від класичних статистичних моделей до складних алгоритмів машинного навчання. Використання сучасних технологій дозволяє ресторанам більш точно оцінювати попит, ефективно управляти запасами та ресурсами, мінімізувати витрати і збільшувати прибутковість.

Висновки до розділу 1

У першому розділі було проведено аналіз предметної області прогнозування попиту у ресторанному бізнесі. Розглянуто сучасні методи прогнозування, визначено ключові фактори, що впливають на попит, а також проведено огляд існуючих рішень для його прогнозування.

Вивчення сучасних методів прогнозування показало, що для оцінки попиту застосовуються як традиційні статистичні підходи (метод ковзного середнього, експоненціальне згладжування, ARIMA), так і більш складні методи машинного навчання (лінійна та поліноміальна регресія, дерева

рішень, нейронні мережі, Prophet). Використання статистичних методів є доцільним у випадках стабільного попиту, тоді як машинне навчання дозволяє аналізувати великі обсяги даних та враховувати складні взаємозв’язки між факторами [13].

Дослідження факторів, що впливають на попит у ресторанах, дозволило виділити внутрішні та зовнішні чинники. До внутрішніх належать якість сервісу, асортимент та ціноутворення, а до зовнішніх — сезонність, економічна ситуація, конкуренція та поведінкові звички споживачів. Аналіз цих факторів є важливим для підвищення точності прогнозування та оптимізації управлінських рішень.

Огляд існуючих рішень показав, що на ринку вже представлені аналітичні платформи (Tableau, Power BI), CRM–системи для управління ресторанним бізнесом (Toast, OpenTable) та хмарні сервіси для аналізу великих даних (AWS Forecast, Google AutoML). Проте більшість готових рішень не враховують специфіку окремих закладів і потребують адаптації під конкретні умови.

Таким чином, проведений аналіз предметної області підтверджує актуальність розробки програмного комплексу для прогнозування попиту на ресторанні послуги, який би інтегрував сучасні алгоритми машинного навчання та враховував ключові фактори впливу.

РОЗДІЛ 2. РОЗРОБКА КОНЦЕПЦІЇ ПРОГРАМНОГО КОМПЛЕКСУ

2.1. Визначення вимог до програмного комплексу

Першим кроком на шляху до розробки ефективного програмного комплексу є формулювання чітких вимог до нього. Цей етап має ключове значення, оскільки саме на основі вимог проєктуються архітектура, функціональність та інтерфейс системи. У межах даної дипломної роботи йдеться про створення програмного комплексу для прогнозування попиту на послуги ресторанів, що повинен враховувати історичні дані, сезонні коливання, зовнішні фактори та особливості поведінки споживачів.

Головна мета створення цього комплексу полягає в автоматизації процесів аналізу, моделювання та прогнозування попиту. Система повинна надавати користувачеві інструменти для завантаження та обробки даних, побудови прогнозів із використанням сучасних моделей машинного навчання, візуалізації результатів, а також збереження та експорту прогнозної інформації для подальшого використання. Таким чином, система повинна бути зручною у використанні, адаптивною до різних типів даних та достатньо точною у своїх прогнозах.

Функціональні можливості програмного комплексу мають охоплювати кілька важливих етапів. Зокрема, це імпорт вхідних даних із зовнішніх джерел (наприклад, файлів формату CSV або баз даних), попередню обробку цих даних (очищення, нормалізацію, усунення пропущених значень), а також аналіз основних факторів, що впливають на попит. Далі користувач повинен мати можливість побудувати прогнозну модель з обраним алгоритмом (наприклад, регресійним, деревним або нейронним), оцінити її точність за допомогою відповідних метрик і переглянути результати у зрозумілому вигляді. Прогноз має бути представлений у формі графіків і таблиць, з

можливістю експорту у форматах CSV, Excel або PDF. Бажано також реалізувати збереження історії прогнозів, щоб у майбутньому можна було аналізувати зміни у попиті за певний період.

Далі наведено таблицю 2.1 з функціональними вимогами до програмного комплексу.

Таблиця 2.1 – Функціональні вимоги до програмного комплексу

№	Вимога	Опис
1	Завантаження даних	Імпорт даних із CSV, Excel, баз даних
2	Попередня обробка даних	Очищення, нормалізація, обробка пропущених значень
3	Вибір моделі прогнозування	Можливість обрати алгоритм: регресія, дерева рішень, нейромережі, Prophet
4	Побудова прогнозу	Формування прогнозу на основі вибраної моделі
5	Візуалізація результатів	Графіки, діаграми, аналітичні таблиці
6	Експорт результатів	Збереження прогнозів у форматах CSV, Excel, PDF
7	Збереження історії прогнозів	Можливість зберігати попередні результати аналізу

Невід'ємною складовою проєктування є також нефункціональні вимоги. Зокрема, система повинна бути продуктивною і здатною працювати з великими наборами даних без втрати швидкодії. Надійність і стабільність також мають велике значення, оскільки програмне забезпечення повинно працювати без збоїв у критичних ситуаціях. Інтерфейс має бути зрозумілим для користувачів без технічної підготовки. Безпека зберігання та передачі даних, особливо у випадках роботи з фінансовою або персональною інформацією, також є важливою вимогою. Крім того, доцільно закласти можливість масштабування системи в разі зростання обсягу даних або кількості користувачів. Далі зображено таблицю 2.2 з нефункціональними вимогами до програмного комплексу.

Таблиця 2.2 – Нефункціональні вимоги до програмного комплексу

№	Категорія	Вимога	Опис
1	Продуктивність	Працездатність з великими наборами даних	Швидка обробка даних розміром до кількох мільйонів записів
2	Надійність	Стабільність роботи	Відсутність збоїв при некоректному введенні або втраті з’єднання
3	Інтерфейс	Зручність для кінцевого користувача	Інтуїтивно зрозумілий інтерфейс без спеціальної підготовки
4	Безпека	Захист даних	Шифрування файлів, контроль доступу до персональної інформації
5	Масштабованість	Гнучкість у розвитку	Можливість додати нові модулі без переписування основного коду

Ще однією важливою складовою є вибір технологій, на яких буде реалізовано програмний комплекс. У межах цього дослідження для реалізації обрано мову програмування Python, оскільки вона надає широкі можливості у сфері аналітики та обробки даних. Для обробки і візуалізації даних буде застосовано бібліотеки Pandas, NumPy, Matplotlib, Seaborn і Plotly. Для побудови моделей машинного навчання використовуватимуться Scikit–learn, TensorFlow, Keras, а також спеціалізовані інструменти для роботи з часовими рядами, зокрема Prophet від Facebook. Для створення вебінтерфейсу або десктопної оболонки планується застосування таких інструментів, як Flask або Streamlit.

Вимоги до програмного комплексу формують основу для його подальшої реалізації, визначають напрям архітектурного проєктування та функціональне наповнення. Чітко сформульовані вимоги сприятимуть створенню зручного, гнучкого та ефективного інструменту для прогнозування попиту на послуги ресторанного бізнесу, що може бути використаний у реальних управлінських задачах.

2.2. Архітектура системи та вибір технологій

Під час розробки програмного комплексу для прогнозування попиту на послуги ресторанів важливим етапом є формування ефективної архітектури системи, яка забезпечує гнучкість, масштабованість, надійність та зручність для кінцевого користувача. Архітектура визначає не лише загальну побудову системи, але й взаємозв’язок між її компонентами, способи зберігання та обробки даних, а також механізми доступу до функціональності.

Для реалізації системи використано перевірені бібліотеки Python, що забезпечують повний цикл обробки, аналізу та візуалізації даних.

Pandas є ключовим інструментом для роботи з табличними даними. Вона дозволяє зручно читати та обробляти інформацію з файлів або баз даних, трансформувати її у зручну форму (DataFrame), проводити агрегацію, фільтрацію, групування та багато інших операцій. Pandas також підтримує інтеграцію з бібліотеками для візуалізації, машинного навчання та обробки числових масивів.

NumPy — основа чисельної обробки в Python. Завдяки своїм векторизованим операціям вона забезпечує ефективну роботу з великими масивами даних. Масиви NumPy (ndarray) використовуються в обчисленнях, статистичних аналізах, а також в ядрах багатьох ML–бібліотек.

Matplotlib відповідає за побудову графіків, діаграм та інших типів візуалізації. Вона дозволяє створювати гнучко настроювані графіки з підтримкою шрифтів, кольорів, підписів, а також комбінованих візуальних елементів на одному полотні.

Seaborn розширює можливості matplotlib, забезпечуючи високорівневу побудову статистичних графіків з естетичним оформленням. Використовується для побудови теплових карт, діаграм розподілу, парних графіків (pairplot), віолончельних графіків тощо.

Scikit–learn (sklearn) є базовою платформою для реалізації класичних моделей машинного навчання. Завдяки широкому набору готових алгоритмів для класифікації, регресії, кластеризації, оцінювання та вибору моделей, вона дозволяє легко побудувати повноцінний ML–пайплайн.

PyQt забезпечує створення графічного інтерфейсу користувача. За його допомогою розроблено віконну форму з кнопками, полями для введення, зонами відображення графіків та таблиць. PyQt дозволяє створювати зручні десктопні додатки, що працюють на різних операційних системах.

У результаті використання описаних технологій, програмний комплекс забезпечує повний цикл роботи з даними — від завантаження та обробки до прогнозування, аналізу результатів і виводу через інтерфейс. Вибрані бібліотеки не лише ефективні, але й добре документовані, що дозволяє легко розширювати систему в майбутньому. На рисунку 2.1 показано логічну структуру програмного комплексу.

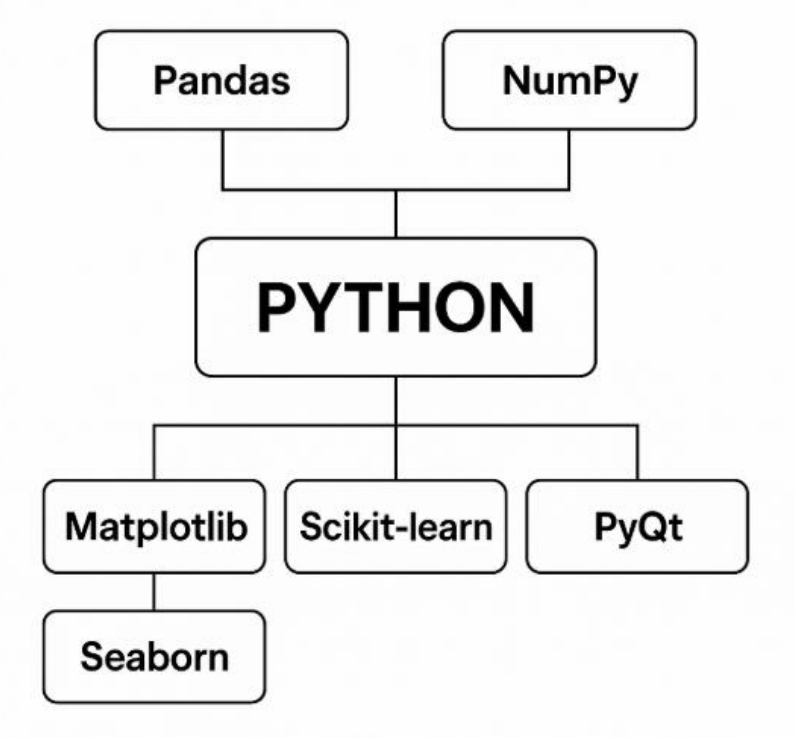

Рисунок 2.1 – Логічна структура програмного комплексу та взаємозв’язок використаних бібліотек

2.3. Алгоритми та методи прогнозування

У межах реалізації програмного комплексу для прогнозування попиту на послуги ресторанів була сформульована задача машинного навчання, метою якої є передбачення кількісних показників майбутнього попиту на основі історичних даних про діяльність закладу. Прогнозування проводиться з використанням моделей регресійного типу, що дозволяє передбачити, наприклад, очікувану кількість замовлень або клієнтів на обраний день, що в свою чергу сприяє оптимізації процесів закупівлі, планування персоналу та організації акцій.

Вхідні дані для моделювання завантажуються користувачем через графічний інтерфейс у форматі .csv або .xlsx. Дані проходять етапи попередньої обробки: очищення, нормалізація, заповнення відсутніх значень та формування ознак (наприклад, перетворення дати на день тижня, перевірка свят, врахування погодних умов або маркетингових кампаній). Сформований оброблений датафрейм демонструється користувачеві через відповідний блок інтерфейсу.

Далі, користувач має змогу обрати одну з кількох реалізованих моделей прогнозування.

У процесі прогнозування попиту на послуги ресторанів можна застосовувати широкий спектр моделей машинного навчання. Нижче подано аналіз найпоширеніших підходів, які можуть бути ефективно використані залежно від характеру вхідних даних, поставленої задачі та вимог до точності прогнозів.

Логістична регресія є класичним методом, який застосовується в задачах класифікації, коли необхідно оцінити ймовірність належності об'єкта до одного з двох класів. Її математична основа — логістична функція, що перетворює вихідну лінійну комбінацію ознак у значення від 0 до 1. Цей метод добре працює при наявності лінійно роздільних даних. Логістична регресія

також дозволяє аналізувати вплив кожної ознаки на результат, що сприяє інтерпретованості моделі. Для багатокласових задач застосовуються варіації "один проти всіх" або "один проти одного". Серед переваг — простота реалізації та ймовірнісна інтерпретація. Серед обмежень — обмежена здатність моделювати складні або нелінійні залежності, а також чутливість до надмірної кількості ознак без регуляризації.

Дерева рішень – цей підхід ґрунтується на ієрархічному поділі даних за ознаками. Кожен вузол дерева приймає рішення на основі певної ознаки, а кінцеві листки представляють прогноз. Модель здатна враховувати як числові, так і категоріальні змінні без попередньої трансформації. Основні переваги: легка інтерпретація, здатність враховувати нелінійні взаємозв’язки, стійкість до окремих викидів. Основним недоліком є ризик переобучення — одна структура дерева може надто точно підлаштовуватись під навчальні дані. Саме тому дерева часто використовують у складі ансамблевих моделей.

Random Forest (випадковий ліс) – це ансамбль дерев рішень, кожне з яких навчається на окремій випадковій вибірці даних. Під час передбачення моделі використовують голосування або усереднення результатів усіх дерев. Завдяки такій архітектурі модель є стабільною, точнішою за окреме дерево та менш схильною до перенавчання. Вона також дозволяє оцінити важливість ознак. До переваг належать: висока точність, стійкість до шуму, здатність працювати з великими наборами ознак і даних. Обмеженням може бути складність налаштування та повільніший час навчання у порівнянні з окремим деревом.

Метод опорних векторів (SVM) – цей алгоритм націлений на побудову оптимальної гіперплощини, яка найбільш ефективно розділяє об'єкти різних класів у багатовимірному просторі. SVM може бути використаний як для класифікації, так і для регресійного прогнозування. При цьому модель підтримує як лінійні, так і нелінійні кордони розділення завдяки використанню так званих ядер (kernel). До переваг методу належать: висока точність на складних задачах, здатність працювати з малими вибірками,

хороша узагальнююча здатність. Серед недоліків — висока обчислювальна вартість на великих наборах даних та потреба у налаштуванні параметрів (ядра, штрафів).

Градієнтний бустинг –це ансамблевий метод, який формує фінальний прогноз як суму багатьох слабких моделей (зазвичай дерев рішень), що навчаються послідовно. Кожне наступне дерево намагається компенсувати помилки попередніх. Перевагами є надзвичайно висока точність, гнучкість, здатність виявляти складні взаємозв'язки. Недоліками є чутливість до перенавчання при неправильному налаштуванні та потреба в значних ресурсах для обробки великих обсягів даних.

Gaussian Naive Bayes (GaussianNB) – це ймовірнісна модель класифікації, яка ґрунтується на застосуванні теореми Байєса з припущенням незалежності ознак. Вважається, що значення ознак мають нормальний розподіл. Алгоритм швидкий, простий у реалізації та працює добре навіть на малих вибірках. Основна перевага — швидкість та низькі вимоги до обчислювальних ресурсів. Проте припущення про незалежність ознак часто не відповідає дійсності, що може негативно вплинути на точність.

На підставі історичних даних про кількість замовлень за попередні місяці, система має надати прогноз кількості замовлень у ресторані на найближчі 7 днів. Враховуються змінні, що впливають на попит: день тижня, свята, погодні умови, акційні кампанії тощо. Формально, маючи часовий ряд попередніх значень y_{t+1} , y_{t+2} , …, y_{t+n} , система має побудувати модель для прогнозу значень

У переважній більшості випадків передбачення майбутніх значень попиту — це регресійна задача. В окремих випадках може бути поставлена як задача класифікації (наприклад, передбачити, чи буде попит вищим за середній).

У підсумку, постановка задачі передбачає реалізацію інтерфейсно зручного та технічно гнучкого інструменту, здатного на основі історичних

даних формувати достовірні прогнози, візуалізувати їх і підтримувати ухвалення управлінських рішень.

У процесі розробки та тестування моделей прогнозування важливим етапом є об'єктивне оцінювання їх точності та здатності узагальнювати закономірності, що містяться у вхідних даних. Для цього використовуються метрики якості прогнозу, які дозволяють порівнювати моделі між собою, а також визначити, наскільки точними є результати моделі у реальних умовах експлуатації.

Оскільки задача прогнозування попиту на послуги ресторанів є регресійною, тобто передбачає прогнозування числових значень (наприклад, кількість клієнтів або замовлень), доцільно застосовувати такі стандартні метрики:

Розглянемо MAE (Mean Absolute Error, Середня абсолютна помилка). Ця метрика показує середнє абсолютне відхилення між фактичними та прогнозованими значеннями. Вона добре інтерпретується в одиницях цільової змінної та зображена в формулі (2.1).

$$\text{MAE}=\frac{1}{n}\sum_{i=1}^{n}|y_i-\hat{y}_i| \tag{2.1}$$

RMSE (Root Mean Squared Error, Квадратична середня помилка), описана в формулі (2.2), показує середнє квадратичне відхилення між прогнозом і фактичним значенням. Порівняно з MAE, сильніше “карає” великі похибки [16].

$$\text{RMSE}=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2} \tag{2.2}$$

R^2 (коефіцієнт детермінації) показує, яка частка варіації цільової змінної пояснюється побудованою моделлю. Значення варіюється від 0 до 1 (іноді від’ємні — якщо модель дуже неточна) – формула (2.3) [16].

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y}_i)^2} \quad (2.3)$$

MAPE (Mean Absolute Percentage Error, Середня абсолютна відносна помилка) - метрична оцінка, яка показує середню похибку у відсотках. У випадках, коли фактичні значення близькі до нуля, може давати значні спотворення [17]. Її описано в формулі (2.4)

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \cdot 100\% \quad (2.4)$$

Для більш глибокого аналізу ефективності моделей у різних умовах також можуть використовуватись:

1) median Absolute Error (MedAE) — менш чутлива до викидів, ніж MAE;

2) SMAPE (Symmetric Mean Absolute Percentage Error) — модифікація MAPE, що симетризує відносну помилку;

3) quantile Loss — для моделей, що передбачають довірчі інтервали.

Переваги та недоліки кожної з метрик вказані в таблиці 2.3.

Таблиця 2.3. Переваги та недоліки кожної з метрик

Метрика	Переваги	Недоліки	Одиниця виміру
MAE	Легка для інтерпретації	Не враховує напрямок і величину великих похибок	Така ж, як у змінної
RMSE	Сильніше карає великі помилки	Може бути нестійкою до викидів	Така ж, як у змінної
R^2	Відображає якість пояснення варіацій	Може бути від’ємною при поганій моделі	Від 0 до 1 (або нижче)
MAPE	Зрозуміла у %	Не працює з нульовими значеннями	Відсотки (%)

У межах реалізації програмного комплексу кожна модель буде оцінюватися на тестових даних із використанням вказаних метрик. Результати виводитимуться у вигляді графіків у відповідному розділі інтерфейсу.

Вибір метрик дозволяє забезпечити об’єктивність порівняння моделей і сформувати рекомендації щодо найбільш придатного алгоритму прогнозування для реального використання у ресторанному бізнесі.

Дерева рішень та їх ансамблеві модифікації є одними з найпопулярніших методів машинного навчання для вирішення задач регресії та класифікації. У контексті прогнозування попиту на послуги ресторанів ці моделі мають важливі переваги: висока точність, здатність до інтерпретації, обробка неоднорідних даних та автоматичне виявлення важливих ознак.

Decision Tree Regressor — базова модель дерева рішень, яка будується шляхом рекурсивного поділу простору ознак для мінімізації певної функції втрат (наприклад, середньоквадратичної помилки). Незважаючи на простоту, дерево рішень має ризик переобучення на навчальних даних, особливо якщо не обмежено глибину або мінімальний розмір підрозділів[18].

Random Forest — ансамблевий метод, який поєднує велику кількість незалежних дерев рішень. Для кожного дерева використовується випадкова підвибірка даних і підмножина ознак, що дозволяє зменшити дисперсію і зробити модель більш стійкою до переобучення. Результат формується шляхом усереднення прогнозів усіх дерев (у випадку регресії) [19].

XGBoost (Extreme Gradient Boosting) — потужний алгоритм бустінгу, який поступово покращує помилки попередніх моделей, додаючи нові дерева. Основні його переваги полягають у високій точності, ефективності реалізації, використанні регуляризації для уникнення переобучення, а також підтримці обробки пропущених значень. Завдяки своїй ефективності XGBoost широко використовується в реальних задачах прогнозування, включаючи сервіси доставки, фінансову аналітику та планування попиту[20].

У таблиці 2.4 наведено порівняльну характеристику розглянутих моделей:

Таблиця 2.4. Порівняльна характеристика розглянутих моделей

Модель	Переваги	Недоліки
Decision Tree	Проста, інтерпретована, швидка в побудові	Схильна до переобучення, нестабільна до змін у даних

Random Forest	Висока точність, стійкість до викидів, зниження варіації	Повільніший час прогнозування, менш інтерпретована
XGBoost	Дуже висока точність, ефективність, регуляризація, масштабованість	Вимагає налаштування гіперпараметрів, складна реалізація

У контексті прогнозування попиту на ресторанні послуги дані моделі мають особливу цінність. Наприклад, моделі можуть враховувати такі чинники, як: день тижня та сезонність; погоду; кількість онлайн–замовлень; історію попереднього попиту; акційні кампанії або святкові дні.

Однією з найбільш ефективних архітектур для обробки послідовностей є рекурентні нейронні мережі (RNN), а також їх вдосконалення — довготривала короткочасна пам’ять (LSTM, Long Short–Term Memory).

RNN (Recurrent Neural Networks) — це клас нейромереж, що має циклічні зв’язки, які дозволяють зберігати інформацію з попередніх кроків у пам’яті [21]. Це робить RNN придатними для задач, де контекст попередніх значень важливий, зокрема для прогнозування на основі часових рядів. Однак класичні RNN мають проблеми з довготривалими залежностями через ефект згасання або вибуху градієнтів.

Саме тому LSTM були запропоновані як рішення цих проблем. Вони мають спеціальну структуру, що включає в себе «комірку пам’яті» та три типи «воріт» (вхідні, вихідні та забування), які дозволяють вибірково зберігати або відкидати інформацію. Завдяки цьому LSTM здатні вивчати як короткотермінові, так і довготермінові залежності в послідовностях.

У контексті прогнозування попиту на послуги ресторанів, LSTM можна використовувати для передбачення кількості клієнтів, замовлень або доходу на основі історичних даних. Особливо ефективною така модель є при наявності трендів, сезонних факторів (наприклад, підвищення попиту у вихідні) або залежностей від зовнішніх чинників (погода, акції, свята).

У таблиці 2.5 представлено узагальнене порівняння LSTM із класичними моделями прогнозування:

Таблиця 2.5. Узагальнене порівняння LSTM із класичними моделями прогнозування

Критерій	Лінійна регресія	ARIMA	Facebook Prophet	LSTM
Підтримка сезонності	Ні	Так	Так	Так
Робота з нелінійностями	Низька	Низька	Середня	Висока
Пояснюваність	Висока	Висока	Висока	Низька
Ресурси для обчислень	Низькі	Середні	Середні	Високі
Адаптація до нових даних	Середня	Середня	Висока	Висока

LSTM добре працює в умовах багатофакторного впливу, коли варто враховувати не лише попередні значення попиту, а й додаткові параметри, такі як день тижня, погода, події тощо. Це робить її перспективною для використання у складних бізнес–системах, включаючи прогнозування в ресторанному секторі.

Висновки до розділу 2

У другому розділі було здійснено ґрунтовну розробку концепції програмного комплексу для прогнозування попиту на послуги ресторанів. На основі аналізу завдання та очікувань користувача визначено функціональні та нефункціональні вимоги до системи, включно з підтримкою імпорту даних, запуском алгоритмів прогнозування, виведенням результатів у зручній формі, гнучкістю у виборі моделей та забезпеченням інтерактивного інтерфейсу.

Запропонована архітектура побудована за принципом трирівневої моделі (frontend–backend–database), що забезпечує модульність, масштабованість та простоту обслуговування. Було обґрунтовано вибір сучасних технологій для реалізації кожного з компонентів: Python з фреймворками Flask/Streamlit для серверної логіки, бібліотеками pandas,

scikit–learn, TensorFlow, Prophet для реалізації алгоритмів прогнозування, а також PyQt для створення користувацького інтерфейсу.

Окрему увагу приділено вибору моделей прогнозування: розглянуто як класичні статистичні методи (ARIMA, SARIMA), так і сучасні підходи на основі машинного навчання (Random Forest, XGBoost, кластеризація) та глибокого навчання (LSTM, RNN). Проведено порівняння підходів за точністю, ресурсомісткістю та зручністю інтеграції, що дозволило обґрунтовано визначити оптимальні алгоритми для реалізації системи.

У даному розділі закладено теоретичну та технічну основу для створення ефективного, гнучкого та користувацьки–орієнтованого програмного комплексу, що здатен вирішувати завдання прогнозування попиту в ресторанному бізнесі з урахуванням реальних потреб і сучасних технологічних можливостей.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО КОМПЛЕКСУ

3.1. Опис структури та компонентів системи

Програмний комплекс створений на основі модульної архітектури. Це дозволяє кожному компоненту виконувати конкретну функцію, забезпечуючи прозору логіку взаємодії та можливість гнучкого оновлення окремих частин системи без шкоди для загальної стабільності.

Структура системи охоплює такі основні компоненти: модуль обробки даних, модуль прогнозування, модуль візуалізації результатів, базу даних для зберігання інформації, а також графічний інтерфейс користувача, реалізований за допомогою бібліотеки PyQt. Схема взаємодії модулів представлена в таблиці 3.1.

Таблиця 3.1 – Функціональні складові програмного комплексу

Компонент	**Опис функціональності**
Модуль завантаження та підготовки даних	Відповідає за імпорт вхідної інформації (наприклад, CSV–файлів), попереднє очищення, нормалізацію, створення додаткових ознак (день тижня, свята тощо).
Модуль аналітики та прогнозування	Здійснює навчання моделей та формування прогнозів за допомогою алгоритмів машинного навчання, статистичних моделей та нейронних мереж.
Модуль візуалізації	Виводить результати у вигляді графіків, таблиць або звітів з використанням бібліотек matplotlib, seaborn або plotly.
Графічний інтерфейс (GUI)	Забезпечує доступ користувача до функціональності системи: завантаження даних, запуск моделі, перегляд метрик та графіків через PyQt–інтерфейс.
Система зберігання	Умовно реляційна структура (напр. SQLite), у якій зберігаються оброблені дані, моделі, налаштування, результати прогнозів.

Користувач взаємодіє з системою через інтуїтивно зрозуміле графічне середовище. Вікно програми умовно поділене на функціональні блоки:

завантаження даних, вибір моделі, запуск прогнозу та відображення результатів. Завдяки цьому користувач може повністю контролювати процес, не взаємодіючи з кодом.

Застосування алгоритмів машинного навчання для задач прогнозування попиту має ряд суттєвих переваг, які дозволяють суттєво покращити точність розрахунків та адаптивність до змін зовнішніх умов. Нижче розглянемо ключові аргументи на користь використання таких методів у рамках розробки програмного комплексу.

По–перше, ці алгоритми здатні працювати з великим обсягом вхідної інформації. До таких даних можуть належати статистика попередніх замовлень, поведінкові характеристики клієнтів, сезонні та економічні коливання, а також інші змінні, що прямо або опосередковано впливають на рівень попиту. Завдяки обчислювальній ефективності сучасних моделей можлива обробка таких масивів у стислі терміни.

По–друге, машинне навчання добре справляється з виявленням складних логічних зв’язків і багаторівневих залежностей у даних, які не завжди очевидні при традиційному статистичному аналізі. Алгоритми здатні автоматично виявляти нелінійності, взаємодію між ознаками, приховані тренди та закономірності, що забезпечує глибше розуміння поведінки системи.

Третьою важливою властивістю є здатність моделей адаптуватися до змін у поведінці клієнтів, ринку або бізнес–середовища. При надходженні нових даних моделі можуть бути переобучені або оновлені, що дозволяє підтримувати прогнози в актуальному стані навіть при значних коливаннях у зовнішніх умовах.

Крім того, застосування машинного навчання дозволяє враховувати численні фактори впливу на попит одночасно. Це можуть бути ціни, маркетингові кампанії, активність конкурентів, дні тижня, свята, погодні умови тощо. Такий підхід забезпечує комплексне врахування динаміки попиту та підвищує точність моделі.

Ще однією перевагою є те, що моделі з часом «навчаються» на основі нових даних і зворотного зв’язку, поступово покращуючи свої прогностичні характеристики. Це особливо важливо у швидкозмінних сферах, де стабільність моделі без гнучкого оновлення є малоефективною.

Нарешті, такі технології дозволяють значною мірою автоматизувати процес прогнозування, що зменшує ризики, пов’язані з людськими помилками, та прискорює отримання результатів. Це підвищує ефективність планування операційної діяльності, зокрема виробництва, управління запасами та маркетингу.

З урахуванням зазначених переваг, машинне навчання є доцільним вибором для задач прогнозування попиту в ресторанному бізнесі. До основних методів, які можуть бути застосовані, належать регресійні моделі (лінійна, поліноміальна), дерева рішень, Random Forest, метод градієнтного бустингу (наприклад, XGBoost), а також архітектури нейронних мереж — передусім LSTM. Конкретний підхід обирається залежно від особливостей вихідних даних, обсягу інформації, очікуваної точності прогнозу та доступних ресурсів.

Для цілей даного дослідження було використано відкритий набір даних, що містить структуровану інформацію про 50 найбільших мереж ресторанів швидкого обслуговування в США за 2021 рік. Джерелом даних є галузевий портал QSR Magazine, що спеціалізується на аналітиці ринку громадського харчування та публікує щорічні огляди щодо ефективності закладів швидкого харчування на національному рівні.

Метою включення цього датасету в дослідження є аналіз факторів, що впливають на успішність ресторанних мереж, з подальшим моделюванням залежностей між фінансовими показниками (наприклад, обсягами продажів) та операційними характеристиками (кількість точок, франшизних/власних закладів, середній виторг на одиницю тощо).

Датасет містить 50 записів — кожен з яких представляє окрему ресторанну мережу, а також 7 основних атрибутів, перелік яких наведено нижче:

У рамках дослідження було обрано великий відкритий набір даних, що містить понад 1300 записів про заклади громадського харчування. Датасет охоплює широкий спектр ресторанів різних типів, розташованих у різних регіонах, із зазначенням численних характеристик, таких як тип кухні, середній чек, рейтинг, кількість відгуків, наявність доставки, можливість бронювання, популярні страви тощо. Дані були зібрані з різних джерел публічної аналітики й онлайн–платформ, що забезпечує актуальність та репрезентативність вибірки.

У межах дослідження було використано відкритий набір даних із платформи kaggle.

Основною метою використання цього датасету є вивчення факторів, які найбільше впливають на попит на ресторанні послуги, а також побудова моделей прогнозування популярності закладів у залежності від їхніх характеристик. На відміну від обмежених за обсягом вибірок, використання великого масиву даних дозволяє отримати більш надійні результати, виявити приховані закономірності та забезпечити високу якість аналітичних висновків.

Сфера харчування в умовах сучасного міста є надзвичайно конкурентною та динамічною. Щодня відкриваються нові ресторани, проте їх успіх залежить від ряду змінних: розташування, середнього цінового сегмента, репутації в онлайн–середовищі, якості обслуговування та кулінарних уподобань клієнтів. Актуальність цього аналізу обумовлена необхідністю прийняття обґрунтованих бізнес–рішень у сфері HoReCa.

Детальний аналіз великого набору даних дозволяє розглядати такі ключові аспекти:

1) геолокаційні чинники (у яких районах ресторани мають вищу концентрацію);
2) залежність попиту від типу кухні;
3) цінова політика та її вплив на відвідуваність;
4) роль рейтингів і кількості відгуків у виборі ресторану;
5) наявність додаткових сервісів (доставка, бронювання тощо);

6) споживчі уподобання та їх динаміка в межах населених пунктів.

Обраний датасет дає змогу не лише провести глибокий аналітичний огляд, а й створити ефективну модель прогнозування попиту, що стане основою для прийняття управлінських рішень у ресторанному бізнесі.

Перш ніж переходити до побудови моделей, важливо провести первинний аналіз інформації. Один із найбільш інформативних способів — це візуалізація, яка дозволяє наочно побачити закономірності, виявити аномалії та оцінити розподіл значень у даних. Діаграми, графіки та гістограми — зручні інструменти, які допомагають швидко зорієнтуватися в структурі набору.

У цьому проєкті для графічного представлення інформації використовувалися бібліотеки matplotlib і seaborn, що є потужними інструментами Python для побудови статистичних графіків.

Завдяки візуалізації можна:

1) побачити, які значення переважають у тій чи іншій категорії;
2) визначити діапазон варіації ознаки;
3) порівняти частоту окремих показників між собою;
4) виявити найменш і найбільш представлені категорії.

На рис. 3.1 зображено скорочений графік розподілу кількості ресторанів у різних містах. Цей графік дає змогу зробити попередні висновки щодо концентрації закладів у певних локаціях та може слугувати відправною точкою для глибшого аналізу впливу географічного чинника на попит.

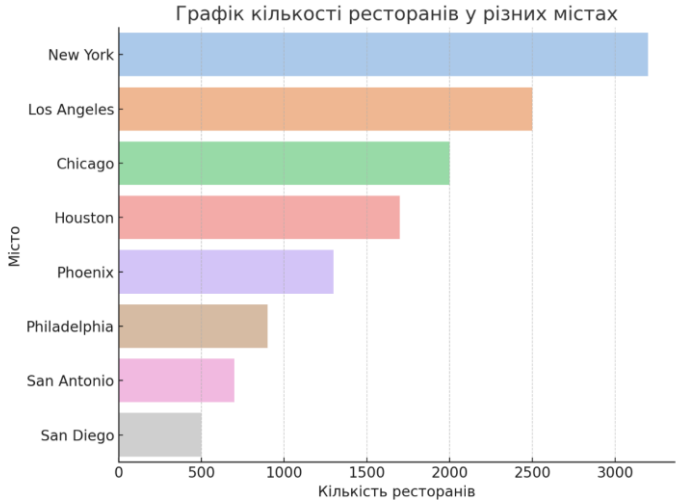

Рисунок 3.1 – Графік кількості ресторанів у різних містах

3.2. Інтерфейс користувача

Для розробки графічної оболонки застосовувалась бібліотека PyQt5, яка надає інструменти для створення сучасного інтерфейсу користувача. Вихідний код інтерфейсу включено до Додатку А.

На рис. 3.2 зображено зовнішній вигляд вікна програми, яке структуровано на чотири основні функціональні зони:

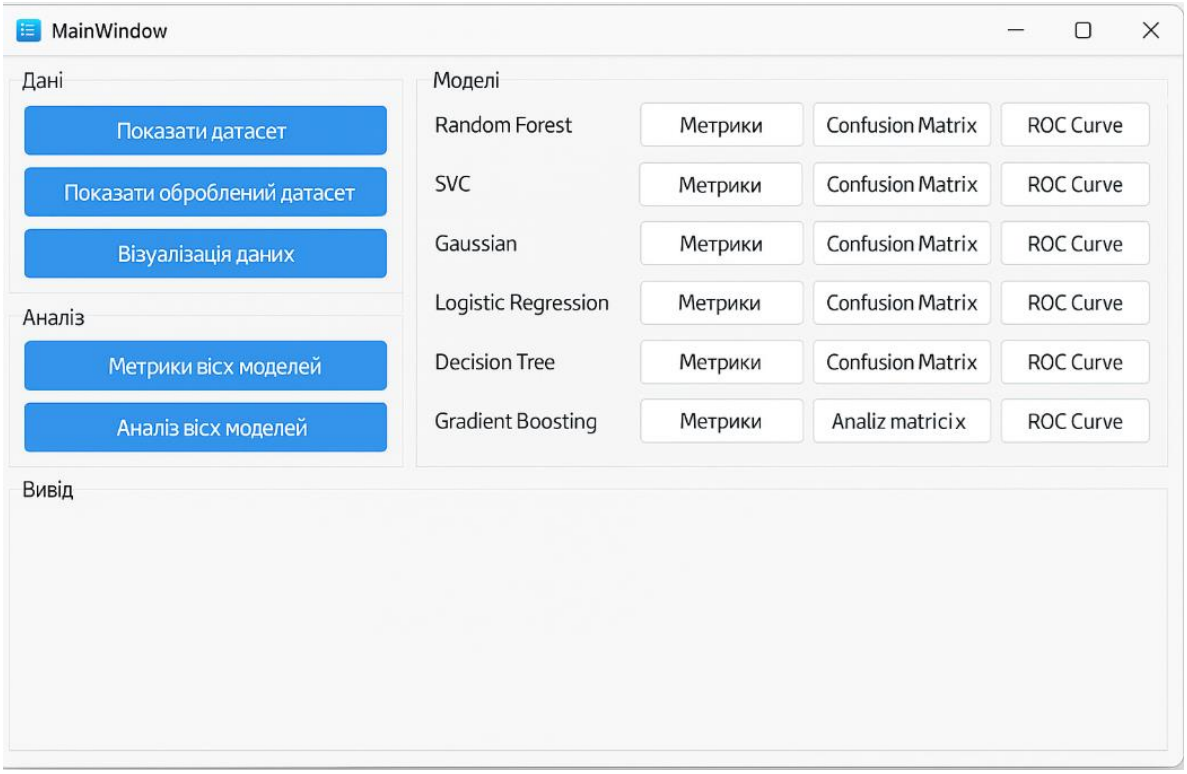

Рисунок 3.2. – Інтерфейс користувача

Секція «Дані»

Цей модуль відповідає за взаємодію з наборами даних. Він включає три кнопки:

Відкрити датасет — відображення набору даних у правій частині вікна;

Показати підготовлені дані — завантаження вже обробленого для навчання датасету;

Побудувати графіки — викликає візуалізації у вигляді гістограм, зображених на рис. 3.3.

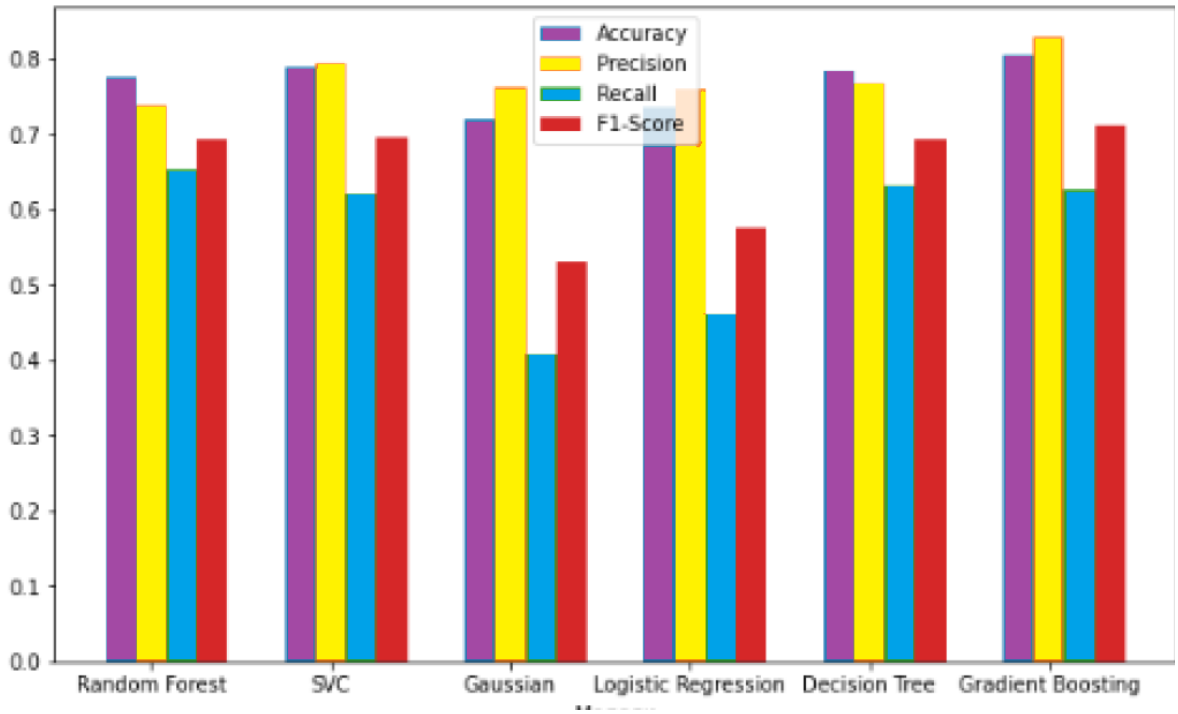

Рисунок 3.3. – Порівняльна діаграма

Секція «Моделі»

Цей блок містить набір кнопок для кожного з алгоритмів машинного навчання. Для кожної моделі доступні такі дії:

Перегляд метрик — обчислення та вивід основних метрик (accuracy, precision, recall, F1) у лівій частині вікна;

Матриця неточностей — побудова confusion matrix для поточної моделі;

ROC–крива — візуалізація характеристик приймача (Receiver Operating Characteristic).

Секція «Аналітика»

Надає змогу провести зведений аналіз усіх моделей одразу:

Метрики моделей — формування списку результатів для всіх алгоритмів;

Порівняльна діаграма — графічна оцінка ефективності моделей (див. рис. 3.3).

Секція «Результати»

У нижній частині інтерфейсу розташовано два окремих блоки виводу:

Лівий — призначений для текстового відображення метрик моделей;

Правий — демонструє таблиці, зокрема dataframes. Цей блок автоматично очищується при відображенні нового набору даних.

Ілюстрацію роботи функціоналу можна побачити на рис. 3.4, де виводяться результати обчислень та приклад таблиці з даними.

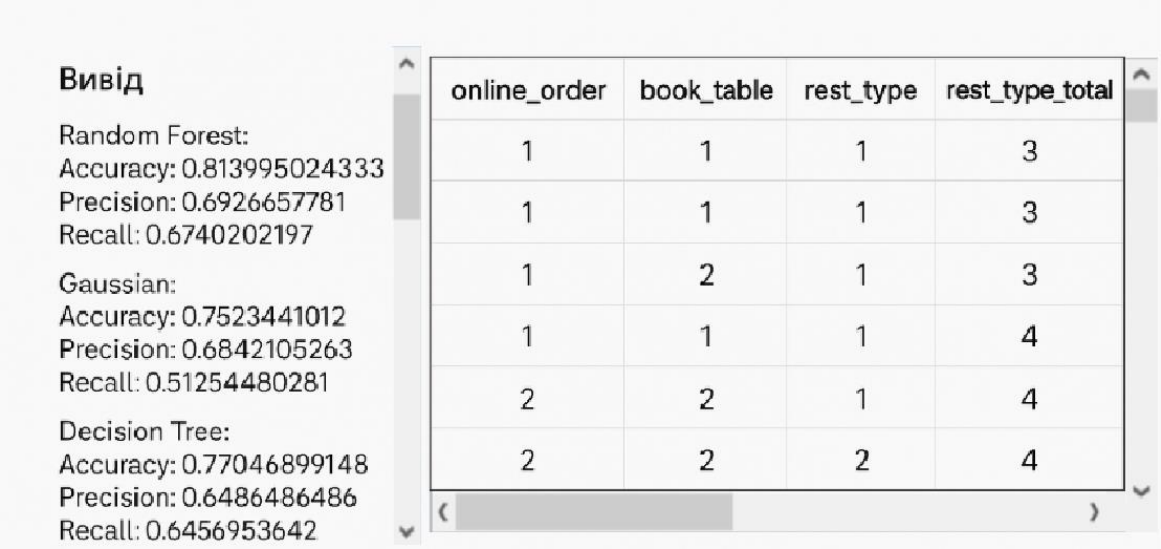

Рисунок 3.4. – Результати обчислень та таблиці з даними

3.3. Інтеграція з іншими інформаційними системами

Сучасні аналітичні рішення в ресторанному бізнесі рідко працюють як ізольовані модулі. Найбільшу ефективність вони демонструють при включенні до єдиного інформаційного середовища, де дані надходять з декількох джерел — торгових систем, систем обліку клієнтів, складських облікових систем та інструментів бізнес–аналітики. Інтеграція програмного комплексу з такими системами дозволяє не лише підвищити точність прогнозування, а й автоматизувати багато пов’язаних процесів. В таблиці 3.2 вказані системи, до яких можлива інтеграція.

Таблиця 3.2 Системи, до яких можлива інтеграція

Тип системи	**Функціональна роль**
POS–системи (Point of Sale)	Фіксація транзакцій у реальному часі, джерело даних про замовлення, час, середній чек
CRM–системи	Облік інформації про клієнтів, їхні вподобання, історія покупок
ERP–системи	Управління логістикою, фінансами, персоналом
Системи обліку складу	Контроль залишків продуктів, автоматизація поповнення запасів
BI–платформи	Побудова інтерактивної аналітики, графіків, дашбордів для прийняття рішень

Інтеграція з переліченими системами дозволяє аналітичному модулю перетворитися з локального рішення у ключовий інструмент управління ресторанним бізнесом на основі даних.

Залежно від технічної архітектури підприємства та специфіки програмного забезпечення, інтеграція може реалізовуватись різними способами. Вибір методу інтеграції напряму залежить від обсягу даних, частоти оновлення інформації, вимог до безпеки та підтримки зовнішніх інтерфейсів. У таблиці 3.3 представлено ключові підходи до технічної реалізації інтеграційних рішень, що застосовуються в сучасних аналітичних системах.

Таблиця 3.3. – Основні способи реалізації інтеграції

Механізм інтеграції	Опис
API	Взаємодія між системами через стандартизовані запити (наприклад, REST API, JSON, XML)
ETL–процеси	Регулярне завантаження даних з зовнішніх джерел, трансформація та збереження у локальному сховищі
Вебхуки	Автоматичні сповіщення про події у зовнішніх системах, що ініціюють дії у аналітичному модулі
SQL–з’єднання	Прямий доступ до бази даних зовнішньої системи через запити для вибірки або оновлення

Інтеграція аналітичного модуля з іншими інформаційними системами підприємства відкриває широкі можливості для підвищення ефективності управління. Завдяки поєднанню операційних, фінансових і клієнтських даних створюється єдина інформаційна база, що підтримує точність прогнозів, оперативність рішень і стратегічне планування. У таблиці 3.4 узагальнено основні переваги, які отримує бізнес у результаті реалізації інтеграційної взаємодії.

Таблиця 3.4 – Очікувані переваги інтеграції

Перевага	**Пояснення**
Оперативність	Отримання актуальних даних у режимі реального часу
Підвищена точність прогнозів	Додаткові змінні покращують якість моделей
Автоматизація процесів	Зменшення залежності від ручного оброблення даних
Масштабованість	Можливість швидко адаптувати систему до змін у бізнесі
Комплексне бачення ситуації	Об'єднання фінансових, поведінкових та операційних показників у спільному середовищі

Для кращого розуміння переваг і можливостей інтеграції аналітичної системи доцільно розглянути конкретний приклад її практичного застосування. Типовий сценарій демонструє послідовність дій при обміні даними між прогнозним модулем і суміжними системами ресторанного бізнесу — такими як POS, CRM, складський облік і платформи візуалізації. У таблиці 3.5 відображено ключові етапи такого інтеграційного процесу, що дозволяє оцінити його логіку, взаємозв’язки та потенційну автоматизацію рішень на основі даних.

Таблиця 3.5. – Сценарій практичної інтеграції

Етап	**Опис дії**
1. Отримання замовлень з POS	Система фіксує підвищення попиту на окрему категорію (наприклад, піца BBQ) у певний день/годину
2. Оновлення прогнозу	Алгоритм враховує нові дані, уточнюючи короткостроковий прогноз
3. Сигнал до CRM	CRM–система отримує вхідні дані для запуску відповідної рекламної кампанії
4. Оповіщення складу	Система формує повідомлення про необхідність поповнення залишків через облікову систему
5. Побудова дашборду в BI	Автоматичне формування графіків про зміни у попиті, виручці, ефективності точок через інтеграцію з Power BI

Для візуалізації логіки взаємодії між програмним комплексом прогнозування попиту та зовнішніми інформаційними системами було побудовано схему на рисунку 3.5, що демонструє основні канали передачі даних та роль кожного компонента в інтегрованому середовищі. На ній відображено як джерела вхідної інформації (системи замовлень, CRM, облікові та ERP–модулі), так і напрямки обміну, які забезпечують автоматизацію процесів та зворотний зв’язок у вигляді виведених прогнозів, дашбордів або бізнес–рішень. Така схема допомагає комплексно уявити, як працює єдина екосистема даних у ресторанному бізнесі.

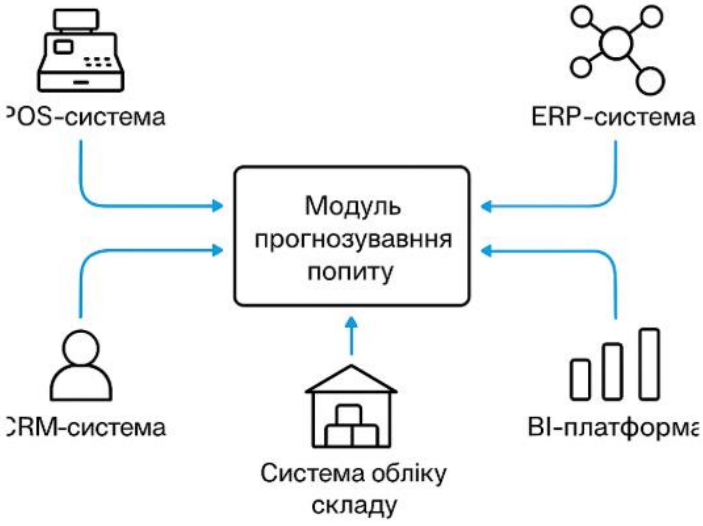

Рисунок 3.5. – Інтеграція з інформаційними системами

Інтеграція аналітичного модуля з інформаційними системами ресторанного бізнесу значно підвищує його функціональну цінність. Вона дозволяє об’єднати розрізнені джерела даних у цілісну структуру, оптимізувати процеси, зменшити вплив людського фактору та підвищити швидкість ухвалення управлінських рішень. У поєднанні з можливостями машинного навчання така інтеграція формує фундамент для побудови інтелектуальної системи підтримки прийняття рішень у сфері громадського харчування.

Висновки до розділу 3

У межах третього розділу було розглянуто ключові аспекти реалізації програмного комплексу, що призначений для прогнозування попиту на послуги у сфері ресторанного бізнесу. Саме на цьому етапі здійснено безпосереднє перетворення аналітичної ідеї у функціональну систему, здатну забезпечити практичну користь для суб’єктів господарювання в індустрії харчування.

Передусім було опрацьовано та реалізовано логічну структуру комплексу. У результаті системний підхід до побудови архітектури дозволив розмежувати компоненти за їхніми функціональними обов’язками, що позитивно впливає як на зручність супроводу програмного продукту, так і на його подальшу адаптацію та масштабування. До складу системи увійшли модулі збору й обробки даних, модуль моделювання, модуль прогнозування, блок виводу результатів, а також графічний інтерфейс, що об’єднує все у єдине середовище.

Одним із важливих досягнень цього етапу стало проєктування та реалізація інтерфейсу користувача. Його побудовано з урахуванням принципів інтуїтивної взаємодії, з акцентом на спрощення доступу до ключових функцій

комплексу — завантаження даних, запуску моделей, перегляду метрик і візуалізацій результатів. В інтерфейсі представлено як текстові, так і графічні форми відображення, що забезпечує зручність аналізу навіть для користувачів без спеціальної технічної підготовки. Така реалізація сприяє популяризації використання аналітики в повсякденній практиці управління.

Окремий акцент зроблено на можливостях інтеграції комплексу з іншими інформаційними системами. Це питання є принципово важливим, адже ефективне прогнозування попиту неможливе без доступу до актуальних, різноманітних і достовірних джерел даних. У роботі було розглянуто основні типи систем, із якими доцільно здійснювати інтеграцію: POS–системи, CRM, ERP, складські системи, а також BI–платформи. У межах реалізації передбачено декілька способів взаємодії: через API, ETL–процеси, вебхуки або прямі SQL–з'єднання. Такий підхід дозволяє забезпечити гнучкість налаштування взаємодії відповідно до технічних можливостей підприємства та існуючої IT–інфраструктури.

Описаний сценарій інтеграції, що охоплює взаємодію між модулем прогнозування, системою замовлень, клієнтською базою та звітними інструментами, демонструє практичну реалізованість і користь такого підходу. У результаті система не лише генерує прогнози, а й автоматично впливає на операційні процеси — наприклад, формування маркетингових кампаній, сигнали на закупівлі чи формування стратегій меню.

Реалізований програмний комплекс відповідає поставленим функціональним і технічним вимогам. Його архітектура є адаптивною, інтерфейс — зручним, а аналітичний потенціал — широким і гнучким. У поєднанні з можливостями інтеграції система може стати ефективним інструментом підтримки управлінських рішень у ресторанному бізнесі, сприяючи підвищенню прибутковості, зниженню ризиків та покращенню задоволеності клієнтів.

РОЗДІЛ 4. ФУНКЦІОНАЛЬНО–ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

Успішна розробка та впровадження програмного забезпечення значною мірою залежить не лише від його функціональності, але й від економічної доцільності реалізації. Саме тому доцільно проводити функціонально–вартісний аналіз (ФВА), який дозволяє оцінити співвідношення між вартістю розробки окремих функціональних компонентів та їх внеском у досягнення основної мети системи.

У цьому розділі розглядається ефективність запропонованого програмного комплексу з точки зору витрат на його реалізацію, а також функціональна значущість кожного з елементів системи. Аналіз базується на виявленні найбільш критичних функцій, які забезпечують максимальну користь для кінцевого користувача, та визначенні потенційних резервів для зниження вартості без втрати якості або продуктивності.

Основною метою функціонально–вартісного аналізу є виявлення оптимального балансу між витратами та функціональністю, а також формування рекомендацій щодо вдосконалення архітектури програмного продукту. Це дозволяє забезпечити більш раціональне використання ресурсів, підвищити економічну ефективність розробки та підготувати продукт до масштабування або комерціалізації.

4.1 Постановка задачі проектування

Застосування методу функціонально–вартісного аналізу (ФВА) дало змогу комплексно оцінити технічні та економічні аспекти розробки програмного комплексу для прогнозування попиту на послуги ресторанів на основі машинного навчання. Такий підхід дозволяє визначити функціональні блоки, які забезпечують максимальну цінність для користувача при оптимальних витратах на реалізацію, інтеграцію та обслуговування системи.

У межах даного дослідження було проаналізовано основні підсистеми та модулі, що формують єдиний архітектурний простір програмного комплексу. Метою проєктування є створення ефективного інструменту, який дозволить ресторанам та закладам громадського харчування оперативно прогнозувати зміну попиту з урахуванням зовнішніх і внутрішніх факторів, таких як сезонність, кількість точок продажу, обсяги продажів, зміни у мережевій структурі тощо.

Визначені такі основні технічні та функціональні вимоги до програмного продукту:

Прогнозна точність — система повинна забезпечувати надійне передбачення змін попиту з використанням алгоритмів машинного навчання, таких як Random Forest, XGBoost, Logistic Regression, Gradient Boosting тощо.

Адаптивність до різних типів даних — програмний комплекс має коректно працювати з різноманітними форматами вхідної інформації, зокрема таблицями CSV або Excel, що містять історичні продажі, кількість точок мережі, середній дохід тощо.

Модульність і масштабованість — система повинна бути гнучкою щодо розширення або зміни алгоритмів прогнозування відповідно до потреб замовника або нових бізнес–умов.

Інтерпретованість результатів — результати моделювання мають бути представлені у зручній візуальній формі (графіки, діаграми, таблиці), що сприяє швидкому прийняттю рішень користувачами без технічної підготовки.

Економічна ефективність — програмний комплекс повинен забезпечувати низькі витрати на розгортання та підтримку, не вимагаючи значних ресурсів чи інфраструктурних змін.

Інтеграційна сумісність — важливо забезпечити можливість обміну даними з іншими системами обліку, CRM або ERP–рішеннями підприємства.

Тобто, задача проєктування полягає у розробці оптимальної функціональної структури програмного комплексу, яка забезпечить баланс

між витратами на реалізацію, технічними характеристиками та очікуваними перевагами для кінцевих користувачів ресторанного бізнесу.

4.2 Обґрунтування функцій програмного продукту

Головна функція F_0 полягає у створенні програмного комплексу, який дозволяє здійснювати точне прогнозування попиту на послуги ресторанів із використанням сучасних алгоритмів машинного навчання. Такий програмний продукт має бути ефективним, гнучким, зручним у використанні та адаптованим до потреб представників ресторанного бізнесу. У межах цієї основної функції можна виокремити наступні підфункції:

F_1 – вибір відповідної мови програмування та інструментів розробки;

F_2 – визначення підходів до реалізації прогнозних моделей;

F_3 – оцінка якості моделі та перевірка точності прогнозів.

Можливі варіанти реалізації функцій:

Функція F_1:

а) Python (основна мова завдяки широким бібліотекам для ML та обробки даних);

б) R (орієнтований на статистичний аналіз);

в) Java (використовується рідше, але має певні фреймворки для ML).

Функція F_2:

а) Використання готових бібліотек для моделей регресії та класифікації (Scikit–learn, XGBoost);

б) Побудова кастомних моделей із застосуванням гіперпараметричної оптимізації;

в) Комбінування моделей (ensemble methods) для покращення точності.

Функція F_3:

а) Розрахунок метрик точності (accuracy, F1–score, MAE) для оцінки ефективності моделі;

б) Порівняння результатів з реальними даними за попередні періоди;

в) Візуалізація помилок через confusion matrix, ROC–криві, гістограми похибок.

Морфологічна карта програмного комплексу (рис. 4.1) дозволяє візуально представити перелік можливих реалізацій кожної з підфункцій, що забезпечують гнучке проектування системи та її адаптацію до різних умов застосування.

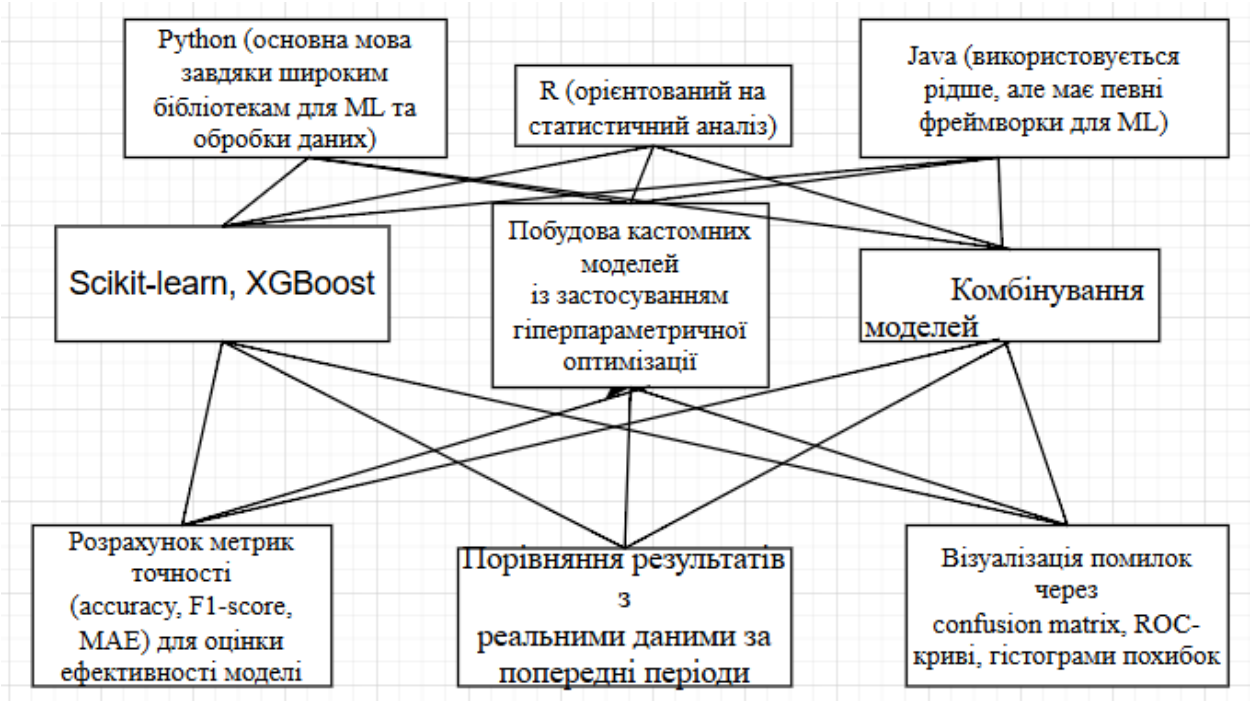

Рисунок 4.1 – Морфологічна карта

За допомогою морфологічної карти була побудована позитивно–негативна матриця, яка відображає можливі варіанти реалізації основних функцій програмного продукту з урахуванням їх переваг та недоліків (табл. 4.1).

Таблиця 4.1 – Позитивно–негативна матриця

Функції	Варіанти реалізації	Переваги	Недоліки
F_1 – Вибір мови програмування	а) Python	Велика кількість бібліотек для обробки даних і машинного навчання; активна спільнота; простота в реалізації	Менша швидкодія у порівнянні з компільованими мовами
	б) R	Висока ефективність у статистичному аналізі та візуалізації	Обмежена гнучкість при реалізації складних інтерактивних систем
	в) Java	Висока продуктивність; кросплатформеність	Складніша розробка; менше інструментів для швидкого прототипування
F_2 – Підхід до реалізації прогнозної моделі	а) Використання готових бібліотек (Scikit–learn, XGBoost)	Швидка інтеграція; стабільна якість прогнозів; зручна документація	Обмеження у налаштуванні моделей; залежність від наявних методів
	б) Побудова кастомних моделей	Гнучкість у конфігурації; можливість глибокого налаштування	Вища складність реалізації; потреба в глибших знаннях
	в) Комбінування моделей (ensemble methods)	Поліпшення точності завдяки поєднанню сильних сторін окремих моделей	Вища ресурсомісткість; складність у підтримці
F_3 – Спосіб перевірки якості прогнозу	а) Порівняння з фактичними даними	Найбільш наближено до реального результату; точна оцінка похибок	Необхідність повного набору історичних даних
	б) Валідація через метрики точності (MAE, MSE, R^2)	Швидка автоматизована перевірка; об'єктивність	Менш наочна для бізнес–користувача; інтерпретація потребує досвіду

Аналіз позитивно–негативної матриці дозволив визначити доцільні шляхи реалізації кожної функції програмного продукту.

Функція F_1**:** Найбільш оптимальним варіантом визнано Python (а), оскільки він забезпечує гнучкість, велику кількість готових рішень і легку адаптацію до задачі.

Функція F_2**:** Найефективнішими підходами є а) готові бібліотеки та б) кастомні моделі — їх можна комбінувати залежно від завдань, забезпечуючи баланс між простотою та точністю.

Функція F_3: Пріоритет надається варіанту б)**,** адже використання метрик точності дозволяє автоматизувати процес оцінки та масштабувати його на великий обсяг прогнозів без залучення ручної перевірки.

Таким чином, доцільними комбінаціями реалізації є:

F_1а – F_2а – F_3б

F_1а – F_2б – F_3б

Ці комбінації забезпечують ефективність, гнучкість, масштабованість і високу точність програмного продукту при обмежених часових та ресурсних рамках реалізації.

4.3 Обґрунтування системе параметров програмного продукту

У процесі розробки програмного комплексу для прогнозування попиту на послуги ресторанів було враховано сучасні технологічні можливості та наявні інструменти аналізу даних. З метою визначення технічного рівня рішення, проведено порівняльну оцінку основних чинників, які мають найбільший вплив на ефективність і якість роботи системи.

Під час вибору між різними мовами програмування (зокрема Python, Java, R), а також підходами до побудови моделей прогнозування (використання готових бібліотек Scikit–learn, XGBoost або кастомні моделі), було сформовано перелік основних параметрів, за якими оцінюються технічні рішення:

X_1 – орієнтовний обсяг коду програмного продукту;

X_2 – час на побудову моделі (навчання);

X_3 – тривалість виконання прогнозного розрахунку (час відповіді системи);

X_4 – середня точність прогнозу.

Ці параметри розділено на три рівні: гірші, середні та кращі. Значення визначаються відповідно до очікувань замовника та специфіки експлуатаційного середовища. Всі значення зведено в таблицю 4.2.

Таблиця 4.2 – Основні параметри програмного продукту

Назва параметра	Умовне позначення	Одиниці виміру	Гірші значення	Середні значення	Кращі значення
Орієнтовний обсяг коду	X_1	кількість рядків коду	3500	2400	1500
Час навчання моделі	X_2	хвилини	90	50	25
Тривалість виконання прогнозу	X_3	секунди	10	5	2
Точність прогнозу	X_4	відсотки	70	85	95

Зазначені параметри дозволяють провести порівняльну оцінку ефективності різних технічних реалізацій, а також побудувати графіки (рис. 4.2–4.5), що візуально ілюструють рівень відповідності варіантів реалізації ключовим вимогам. Це, у свою чергу, дає змогу аргументовано обрати оптимальне рішення для розробки на основі функціонально–вартісного аналізу.

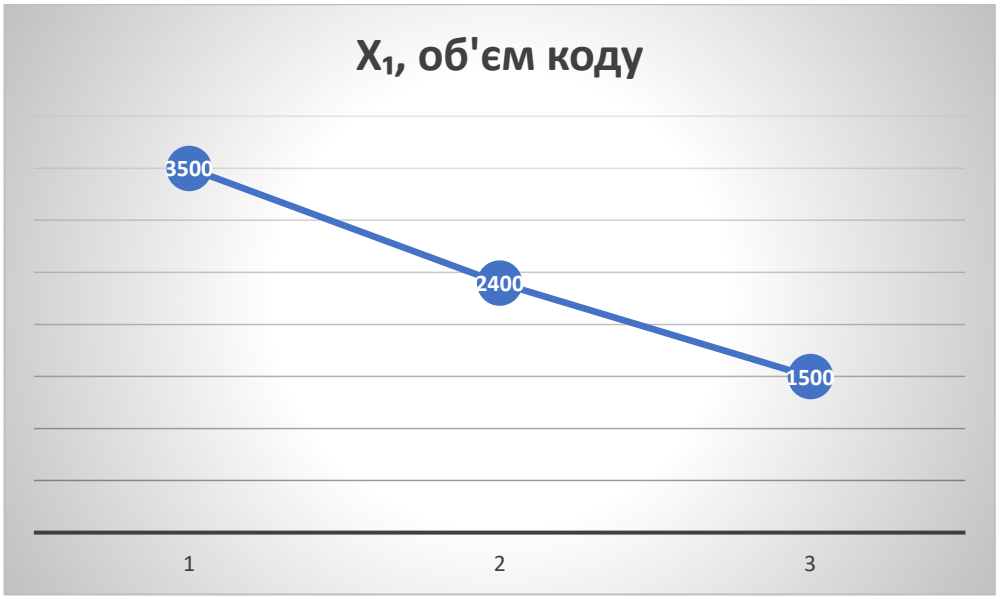

Рисунок 4.2 – X1, Гіпотетичний об’єм коду програми

Рисунок 4.3 – X2, Час навчання даних

Рисунок 4.4 – X3, Час витрачений на виконання поставленого завдання

Рисунок 4.5 – X4, Точність моделі

4.4 Аналіз експертного оцінювання параметрів

У процесі розробки програмного комплексу важливо не лише врахувати технічні характеристики системи, а й провести об'єктивне оцінювання значущості кожного з параметрів, які впливають на ефективність прогнозування попиту. З цією метою було проведено експертне оцінювання з використанням методу попарного порівняння.

У складі експертної групи було чотири фахівці у галузі інформаційних технологій, аналітики даних та економіки ресторанного бізнесу. Основною метою оцінювання стало визначення вагових коефіцієнтів параметрів, що найбільш критично впливають на якість і практичність програмного рішення.

Процес оцінювання включав такі етапи:

1) присвоєння параметрам рангів згідно з їхньою важливістю;

2) розрахунок сумарного рангу кожного параметра;

3) обчислення середнього значення рангу;

4) аналіз відхилень від середнього значення для визначення стабільності оцінок;

5) підготовка до подальшого розрахунку коефіцієнтів вагомості.

В таблиці 4.3 показано результати ранжування параметрів

Таблиця 4.3 – Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	1	2	3	4	5	6	7	Сума рангів Ri	Відхилення Δi	Δi^2
X_1	Обсяг коду програмного продукту	кількість рядків коду	3	3	2	4	4	2	4	22	3	9
X_2	Час навчання моделі	хвилини	4	2	4	2	2	1	2	17	–2	4
X_3	Час виконання прогнозу	секунди	2	4	3	3	3	4	3	22	3	9
X_4	Точність прогнозу	відсотки	1	1	1	1	1	3	1	9	–7	49
	Разом		**10**	**10**	**10**	**10**	**10**	**10**	**10**	**70**	**0**	**71**

Щоб оцінити достовірність експертних рейтингів, визначаємо такі параметри:

а) сумарний ранг кожного показника та загальний ранг усіх показників:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 112, \quad (4.1)$$

де N – кількість експертів,

n – число параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 28 \quad (4.2)$$

в) різниця між рангами кожного параметра та середньої суми рангів:

$$\Delta_i = R_i - T. \quad (4.3)$$

Сума, що визначає відхилення кожного параметру повинна дорівнювати 0;

г) загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 178. \quad (4.4)$$

Коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 187}{7^2(4^3 - 4)} = 0{,}585 > W_k = 0{,}56. \quad (4.5)$$

Отримані результати ранжування можна вважати надійними, адже розрахований коефіцієнт узгодженості перевищує нормативний показник, значення якого дорівнює 0,56. Таким чином утворимо таблицю 4.4, де попарно порівняємо всі параметри.

Таблиця 4.4 – Попарне порівняння параметрів

Параметри	1	2	3	4	5	6	7	Перевага	Значення
X1 і X2	>	=	>	>	<	>	>	X1 > X2	1.5
X1 і X3	=	<	>	=	<	>	=	X1 > X3	1
X1 і X4	<	<	=	<	<	<	<	X1 < X4	0.5
X2 і X3	<	=	>	=	<	=	=	X2 < X3	0.5
X2 і X4	<	<	<	<	<	<	<	X2 < X4	0.5
X3 і X4	<	=	<	<	<	<	<	X3 < X4	0.5

Ступінь переваги *i*–го параметра над *j*–тим визначаєтьс ячисловим параметром a_{ij} і визначається по формулі:

$$a_{ij} = \begin{cases} 1.5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j \\ 0.5 \text{ при } X_i < X_j \end{cases}. \quad (4.6)$$

З отриманих числових оцінок переваги складемо матрицю $A=\| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{ві}$ за наступними формулами:

$$K_{\text{ві}} = \frac{b_i}{\sum_{i=1}^{n} b_i} \quad (4.7)$$

$$b_i = \sum_{i=1}^{N} a_{ij} \quad (4.8)$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}, \tag{4.9}$$

$$b_i' = \sum_{i=1}^{N} a_{ij} b_j \tag{4.10}$$

Як видно з таблиці 4.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 4.5 – Розрахунок вагомості параметрів

Параметри x_i	Параметри x_j				Перша ітер.		Друга ітер.	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$
X1	1	1,5	0,7	1,4	4,6	0,282209	1,091	0,279209
X2	0,6	1	0,7	1,4	3,7	0,226994	0,864	0,2221263
X3	1,4	1,3	1	1,4	5,1	0,312883	1,252	0,320509
X4	0,7	0,6	0,6	1	2,9	0,177914	0,699	0,17902
Всього:					16	1	59	1

4.5 Аналіз рівня якості варіантів реалізації функцій

Для визначення ефективності кожного з можливих варіантів реалізації основних функцій програмного продукту виконується оцінка їх технічного рівня на основі визначених параметрів:

X1 – умовний обсяг програмного коду;

X2 – час навчання моделі;

X3 – час виконання завдання;

X4 – точність моделі.

У даному випадку абсолютні значення параметрів X1, X2 і X4 узгоджуються з технічними вимогами й умовами використання програмного продукту. Параметр X3 також відповідає допустимому рівню.

Розрахунок коефіцієнта технічного рівня кожного з варіантів реалізації функцій проводиться за формулою:

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j} \text{ ,} \qquad (4.11)$$

де n – кількість параметрів;

$K_{вi}$– коефіцієнт вагомості i–го параметра;

B_i – оцінка i–го параметра в балах.

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основна функція	Варіант реалізації	Параметр	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Внесок у якість (Кві × В)
F1	А	X1	630	20	0.19	3.8
F2	А	X4	92	31	0.34	10.54
	Б	X4	83	26	0.34	8.84
F3	А	X3	68	27	0.15	4.05

За даними з таблиці 4.6 за формулою:

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}], \qquad (4.12)$$

визначаємо рівень якості кожного з варіантів:

K_{Kl} = 3.80 (F1.A) + 10.54 (F2.A) + 4.05 (F3.A) = 18.39;

$$K_{K2} = 3.80\ (F1.А) + 8.84\ (F2.Б) + 4.05\ (F3.А) = 16.69$$

Як видно з розрахунків, кращим є 1 варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.6 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки програмного продукту необхідно попередньо провести розрахунок загальної трудомісткості. Всі варіанти реалізації охоплюють два окремих завдання:

Розробка проєкту програмного продукту.

Розробка програмної оболонки.

За ступенем новизни:

Завдання 1 відноситься до групи Б.

Завдання 2 також до групи Б.

За складністю алгоритмів:

Завдання 1 – група 1.

Завдання 2 – група 2.

За джерелами інформації:

Для завдання 1 використовується довідкова інформація.

Для завдання 2 — структуровані дані.

Формула розрахунку загальної трудомісткості:

Загальна трудомісткість обчислюється як:

$$\mathrm{Т_О} = \mathrm{Т_Р} \cdot \mathrm{К_П} \cdot \mathrm{К_{СК}} \cdot \mathrm{К_М} \cdot \mathrm{К_{СТ}} \cdot \mathrm{К_{СТ.М}}, \qquad (4.13)$$

де $\mathrm{Т_Р}$ – трудомісткість розробки ПП;

$\mathrm{К_П}$ – поправочний коефіцієнт;

$К_{СК}$ – коефіцієнт на складність вхідної інформації;

$К_{М}$ – коефіцієнт рівня мови програмування;

$К_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм;

$К_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Завдання 1:

$$T_1 = 64 \cdot 1.021 \cdot 1 \cdot 0.6 = 65.94 \text{ людино–днів}$$

Завдання 2:

$$T_2 = 27 \cdot 1.08 \cdot 1 \cdot 0.7 = 20.41 \text{ людино–днів}$$

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$T_0 = (65.94+20.41) \cdot 8 = 690.8$ людино–годин

В розробці беруть участь:

Один програміст з окладом 75 000 грн;

Один аналітик/спеціаліст домену з окладом 75 000 грн.

$$С_ч = \frac{М}{T_m \cdot t} \text{ грн.}, \quad (4.14)$$

де М – місячний оклад працівників;

T_m – кількість робочих днів на місяць;

t – кількість робочих годин в день.

$$С_ч = \frac{5000 + 75000}{2 \cdot 18 \cdot 8} = 520.83 \text{ грн.} \quad (4.15)$$

Тоді, розрахуємо заробітну плату за формулою:

$$С_{ЗП} = С_{ч} \cdot Т_{i} \cdot К_{Д}, \quad (4.16)$$

де $С_{ч}$– величина погодинної оплати праці програміста;

$Т_{i}$ – трудомісткість відповідного завдання;

$К_{Д}$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

$$С_{ЗП} = 520,83 \cdot 690,8 \cdot 1,125 = 404\,691,17 \text{грн.}$$

Відрахування на єдиний соціальний внесок становить 22%:

$$С_{ВІД} = С_{ЗП} \cdot 0.22 = 404\,691,17 \cdot 0,22 = 89\,032,06 \text{грн}$$

Тепер визначимо витрати на оплату однієї машино–години. ($С_{М}$)

Так як одна ЕОМ обслуговує одного програміста з окладом 75000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_{Г} = 12 \cdot М \cdot К_{З} = 12 \cdot 75000 \cdot 0,2 = 180\,000 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_{Г} \cdot (1 + К_{З}) = 180000 \cdot 1,2 = 216\,000 \text{ грн.}$$

Відрахування на соціальний внесок:

$$С_{ВІД} = С_{ЗП} \cdot 0.22 = 216000 \cdot 0,22 = 47\,520 \text{ грн.}$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 56 000 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1,25 \cdot 0,25 \cdot 56000 = 17\,500 грн.,$$

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача;

$К_А$– річна норма амортизації;

$Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1,25 \cdot 56000 \cdot 0,03 = 2\,100 грн.,$$

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В =$$
$$(365 - 105 - 12 - 15) \cdot 8 \cdot 0,88 = 233 \cdot 8 \cdot 0,88 = 1\,638,08 \text{ годин}$$

де $Д_К$ – календарна кількість днів у році;

$Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів;

$Д_Р$ – кількість днів планових ремонтів устаткування;

t –кількість робочих годин в день;

$К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1\,638{,}08 \cdot 0{,}32 \cdot 0{,}5 \cdot 9{,}43 = 2466{,}06 \text{ грн}$$

де $N_С$ – середньо–споживча потужність приладу;

$К_З$– коефіцієнтом зайнятості приладу;

$Ц_{ЕН}$ – тариф за 1 КВт–годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$С_Н = Ц_{ПР} \cdot 0.67 = 56\,000 \cdot 0{,}67 = 37\,520 \text{ грн}$$

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н, \qquad (4.17)$$

$$С_{ЕКС} = 404\,691{,}17 + 89\,032{,}06 + 17\,500 + 2\,100 + 2466{,}06 + 37\,520 = 553309{,}29 \text{ грн}$$

Собівартість однієї машино–години ЕОМ дорівнюватиме:

$$С_{М–Г} = С_{ЕКС} / Т_{ЕФ} = 563\,551{,}83 / 1638{,}08 = 343{,}98 \text{ грн/год.}$$

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$С_М = С_{М–Г} \cdot Т, \qquad (4.18)$$

$$С_М = 343{,}98 \cdot 690{,}8 = 237\,580{,}78 \text{грн}$$

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0{,}67, \quad (4.19)$$

I. $С_Н = 404\,691{,}17 \cdot 0{,}67 = 271\,143{,}08$грн

Отже, вартість розробки ПП за варіантами становить:

$$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н, \quad (4.20)$$

I. $С_{ПП} = 404\,691{,}17 + 89\,032{,}06 + 237\,580{,}78 + 271\,143{,}08 = 1\,002\,447{,}09$грн

4.7 Вибір кращого варіанту ПП техніко–економічного рівня

Розрахуємо коефіцієнт техніко–економічного рівня за формулою:

$$К_{ТЕР j} = К_{Кj} / С_{Фj}, \quad (4.21)$$

$$К_{ТЕР1} = 20{,}16 / 1002447{,}09 = 2{,}01 \cdot 10^{-5},$$

На основі проведеного функціонально–вартісного аналізу, вартісної оцінки розробки та технічної характеристики програмного продукту були розраховані трудові та фінансові показники. Так, трудомісткість першого ключового завдання — розробки проектної частини ПП — склала **65,94 людино–дні**, з урахуванням коефіцієнтів поправок на використання довідкової інформації (**1,021**) та стандартних модулів (**0,6**). Для другого завдання — розробки програмної оболонки — трудомісткість дорівнює 20,41

людино–дні, із застосуванням коефіцієнта новизни та алгоритмічної складності (1,08 та 0,7 відповідно).

Сумарна трудомісткість проєкту в людино–годинах становить 690,8 годин, що було використано для оцінки повної вартості реалізації. Загальна собівартість розробки програмного продукту за обраним варіантом дорівнює 1 002 447,09 грн, що охоплює заробітну плату, нарахування на соціальні внески, оплату машинного часу та накладні витрати.

Обчислений коефіцієнт техніко–економічного рівня для обраного варіанта склав $\mathbf{2{,}01 \times 10^{-5}}$, що свідчить про достатньо високу вартість розробки при порівняно помірному технічному рівні функціональності. Такий результат вимагає ретельнішої оцінки потенційної економічної вигоди, що може бути отримана від комерційного впровадження продукту, а також дає підстави для перегляду окремих етапів проєкту з метою оптимізації витрат.

Таким чином, хоча запропонований варіант реалізації програмного продукту є технічно доцільним, його подальше впровадження повинно враховувати фінансову доцільність, ризики окупності та потенціал для масштабування з мінімальними витратами.

Висновки до розділу 4

Проведений функціонально–вартісний аналіз дозволив всебічно оцінити техніко–економічні характеристики програмного продукту для управління ресторанним сервісом. Розраховано коефіцієнти технічного рівня для різних варіантів реалізації основних функцій, а також виконано повну калькуляцію витрат, пов’язаних з проектуванням, програмуванням, використанням обчислювальної техніки та енергоресурсів. Це надало змогу визначити оптимальне співвідношення між функціональністю, точністю роботи системи та вартістю її розробки.

Зокрема, було враховано вагомість кожного з параметрів — продуктивності, точності моделі, обсягу коду, тривалості навчання даних —

на основі експертного оцінювання. Результати показали, що найважливішим показником для оцінки якості є точність моделі прогнозування, яка безпосередньо впливає на ефективність управлінських рішень у сфері ресторанного обслуговування.

Аналіз можливих варіантів реалізації дозволив виявити найдоцільнішу комбінацію функціональних рішень з точки зору співвідношення «якість/вартість». За результатами техніко–економічного обґрунтування встановлено, що хоча загальна вартість розробки є значною, обраний варіант забезпечує найвищий технічний рівень серед розглянутих альтернатив.

Отже, програмний продукт має високий потенціал для впровадження у практичну діяльність ресторанних мереж, забезпечуючи автоматизацію аналізу попиту, зниження ручної праці та підвищення точності управлінських рішень. Подальші дослідження мають бути спрямовані на зниження витрат за рахунок оптимізації програмного стеку, повторного використання модулів та впровадження хмарної інфраструктури.

ВИСНОВКИ

У процесі виконання дипломної роботи на тему «Програмний комплекс для прогнозування попиту на послуги ресторанів» було вирішено ряд актуальних завдань, пов’язаних із розробкою та впровадженням інформаційної системи, що використовує інструменти машинного навчання для прогнозного аналізу. Враховуючи динамічний розвиток індустрії громадського харчування, автоматизація процесів прийняття рішень на основі прогнозу попиту має вагоме прикладне значення для підвищення ефективності управління ресурсами, оптимізації закупівель та покращення обслуговування клієнтів.

У вступі до роботи було обґрунтовано актуальність обраної теми, сформульовано мету й основні завдання дослідження, окреслено методи, що застосовуються в рамках розробки програмного комплексу. Наголошено на важливості адаптивних цифрових рішень у сучасних умовах конкуренції на ринку ресторанних послуг.

У першому розділі проведено глибокий аналіз предметної області. Розглянуто особливості та специфіку прогнозування попиту саме в ресторанному бізнесі, виокремлено ключові фактори впливу (географічне розташування, сезонність, маркетингові кампанії, дні тижня тощо), а також здійснено огляд сучасних підходів і технологій, що застосовуються в подібних завданнях — як у вітчизняній практиці, так і за кордоном. Особливу увагу приділено порівнянню алгоритмів класичного статистичного прогнозування з сучасними методами машинного навчання. Зроблено висновок про перспективність застосування гібридних та ансамблевих моделей.

У другому розділі було розроблено концепцію програмного комплексу, що включає постановку вимог до системи, архітектурну побудову та вибір актуальних інструментів. Обрано сучасний стек технологій: Python як основна мова програмування, бібліотеки scikit–learn, XGBoost, LightGBM, Tkinter для реалізації інтерфейсу. Сформовано підхід до вирішення задачі прогнозування

з використанням різних моделей: логістичної регресії, дерева рішень, SVM, Random Forest, Gradient Boosting тощо. Порівняльний аналіз моделей засвідчив, що ансамблеві підходи забезпечують вищу точність прогнозу.

У третьому розділі представлено реалізацію програмного комплексу, включаючи опис його структури, функціональних модулів, графічного інтерфейсу та можливостей інтеграції з іншими інформаційними системами. Інтерфейс створено з урахуванням потреб кінцевого користувача — менеджера ресторану або аналітика. Передбачено інтуїтивно зрозуміле введення даних, вибір моделі прогнозування та виведення результатів у табличному та графічному вигляді. Також реалізовано можливість оновлення даних, що підвищує адаптивність системи до змін у зовнішньому середовищі.

Четвертий розділ присвячено функціонально–вартісному аналізу, що дав змогу всебічно оцінити ефективність реалізованого ПЗ не лише з технічної, але й з економічної точки зору. Було розроблено кілька варіантів реалізації функцій, після чого проведено експертне оцінювання параметрів, побудовано морфологічну карту та обчислено коефіцієнти технічного рівня. Далі здійснено розрахунки витрат на розробку, включаючи заробітну плату, оплату машинного часу, амортизацію обладнання, витрати на електроенергію та накладні витрати. Проведений аналіз дозволив зробити висновок про найкращий варіант реалізації, що забезпечує оптимальне співвідношення функціональності та витрат. Значення коефіцієнта техніко–економічного рівня свідчить про високий потенціал запропонованого рішення для подальшого впровадження та масштабування.

У рамках дипломної роботи:

1) виконано теоретичний аналіз предметної області та сучасних тенденцій у прогнозуванні попиту;

2) обґрунтовано доцільність застосування методів машинного навчання для задач прогнозування у сфері ресторанного обслуговування;

3) розроблено повноцінний концептуальний підхід до побудови системи;

4) реалізовано програмний комплекс, що забезпечує точне прогнозування на основі введених даних;

5) проведено повноцінний техніко–економічний аналіз ефективності реалізації.

Практичне значення роботи полягає в можливості адаптації створеного програмного комплексу до реальних умов роботи підприємств громадського харчування з метою підвищення прибутковості та ефективності операційної діяльності. Подальші дослідження можуть бути спрямовані на інтеграцію з POS–системами, автоматизованими системами закупівель та розширення функціоналу за рахунок аналізу поведінки клієнтів або зовнішніх маркетингових факторів.

Отже, поставлені в роботі завдання було успішно виконано, а її результати можуть бути рекомендовані для використання в малому та середньому бізнесі у сфері HoReCa.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Овчиннікова В. О., Скоробагач А. В. Підвищення точності прогнозування попиту в ресторанному бізнесі. *Вісник економіки транспорту*, 2024, № 1(29), с. 45–54. URL: http://btie.kart.edu.ua/article/view/322877 (дата звернення: 15.04.2025).

2. Мацеха Д. С., Бурий С. А. Маркетинг у сфері готельно–ресторанного бізнесу та туризму. Київ: ХМНУ, 2014, т. 12, № 3, с. 112–119. URL: https://elar.khmnu.edu.ua/items/5876cd44–486e–42c1–8d47–97af67601d3e (дата звернення: 10.04.2025).

3. Кушнірук В., Величко О. Управління бізнес–процесами в готельно–ресторанному бізнесі. *Економіка та суспільство*, 2023, № 7, с. 98–107. URL: https://economyandsociety.in.ua/index.php/journal/article/view/2157 (дата звернення: 12.04.2025).

4. Завадинська О. Ю., Кінчур А. А. Сучасні тенденції впровадження інноваційних форм обслуговування в закладах ресторанного господарства. *Підприємництво і торгівля*, 2019, № 2, с. 33–41. URL: http://www.journals–lute.lviv.ua/index.php/pidpr–torgi/article/view/11 (дата звернення: 08.04.2025).

5. Кирніс Н. І. Моніторинг попиту на кейтерингові послуги в контексті забезпечення конкурентоспроможності підприємств. *Науковий вісник ПУЕТ. Економічні науки*, 2021, № 3(45), с. 56–65. URL: http://www.journal.puet.edu.ua/index.php/nven/article/download/1704/1502 (дата звернення: 09.04.2025).

6. Кальмуцький М. К. Впровадження технології крафтових хлібобулочних виробів на базі ресторану «Містер Чарлі» (ФОП Ніконюк А. П.). Харків: ХМНУ, 2023, т. 15, № 4, с. 23–31. URL: https://elar.khmnu.edu.ua/items/11363d89–dbf0–4024–9cf4–d5032ad961ff (дата звернення: 14.04.2025).

7. Сонгінас В. В. Організаційно–економічні засади використання інноваційного потенціалу підприємства ресторанного господарства (на

прикладі ТОВ «Україна» ресторан). *Вісник ТНТУ*, 2023, № 9, с. 70–80. URL: https://elartu.tntu.edu.ua/handle/lib/42234 (дата звернення: 11.04.2025).

8. Sharma B., Aropa R., Kharub M. Critical success factors affecting the restaurant industry: Insights from restaurant managers. *FIIB Business Review*, 2021, т. 10, № 4, с. 309–314. URL: https://journals.sagepub.com/doi/abs/10.1177/23197145211042429 (дата звернення: 13.04.2025).

9. El-Said O., Al Hajri S. Are customers happy with robot service? Investigating satisfaction with robot service restaurants during the COVID-19 pandemic. *Heliyon*, 2022, т. 8, № 6, с. e10440:e10450. URL: https://www.cell.com/heliyon/fulltext/S2405–8440(22)00274–2 (дата звернення: 07.04.2025).

10. Chen H., Qi R. Restaurant frontline employees' turnover intentions: three-way interactions between job stress, fear of COVID-19, and resilience. *International Journal of Contemporary Hospitality Management*, 2022, т. 34, № 8, с. 2501–2520. URL: https://www.emerald.com/insight/content/doi/10.1108/ijchm–08–2021–1016/full/html (дата звернення: 09.04.2025).

11. Chen L., Chen Z., Zhang Y. et al. Artificial intelligence–based solutions for climate change: a review. *Environmental Science and Pollution Research*, 2023, т. 30, № 15, с. 21000–21018. URL: https://link.springer.com/article/10.1007/s10311–023–01617–y (дата звернення: 16.04.2025).

12. Zhu J., Dong H., Zheng W. et al. Review and prospect of data–driven techniques for load forecasting in integrated energy systems. *Applied Energy*, 2022, т. 310, с. 118391. URL: https://www.sciencedirect.com/science/article/pii/S0306261922006262 (дата звернення: 14.04.2025).

13. Karagiannis D., Andrinos M. The role of sustainable restaurant practices in city branding: The case of Athens. *Sustainability*, 2021, т. 13, № 4,

с. 2271–2289. URL: https://www.mdpi.com/2071–1050/13/4/2271 (дата звернення: 10.04.2025).

14. Hyndman R. J., Athanasopoulos G. *Forecasting: Principles and Practice*. 2-е вид., Melbourne: OTexts, 2018. 380 с. URL: https://otexts.com/fpp2/ (дата звернення: 12.04.2025).

15. Brownlee J. *Deep Learning for Time Series Forecasting*. Melbourne: Machine Learning Mastery, 2020. 382 с.

16. Taylor S. J., Letham B. Forecasting at Scale (Prophet) // *PeerJ Preprints*, 2017, т. 5, e3190v2. URL: https://doi.org/10.7287/peerj.preprints.3190v2.

17. Choi J., Zhang Y. W., Nadzri N. I. B. M. A review of forecasting studies for the restaurant industry: Focusing on results, contributions and limitations. *Global Business & Finance Review*, 2022, т. 27, № 2, с. 112–130. URL: https://www.econstor.eu/handle/10419/305845 (дата звернення: 11.04.2025).

18. Strotmann C., Baur V., Börnert N., Gerwin P. Generation and prevention of food waste in the German food service sector in the COVID-19 pandemic. *Socio-Economic Planning Sciences*, 2022, т. 87, с. 101064. URL: https://sciencedirect.com/science/article/pii/S0038012121000963 (дата звернення: 13.04.2025).

19. Khan R. Artificial intelligence and machine learning in food industries: A study. *J. Food Chem. Nanotechnol.*, 2022, т. 8, № 4, с. 145–153. URL: https://www.researchgate.net/publication/357833952 (дата звернення: 15.04.2025).

20. Alt R. Digital transformation in the restaurant industry: Current developments and implications. *Journal of Smart Tourism*, 2021, т. 7, № 1, с. 9–25. URL: https://journals.sagepub.com/doi/epdf/10.52255/smarttourism.2021.1.1.9 (дата звернення: 12.04.2025).

21. Borowski P. F. Innovative processes in managing an enterprise from the energy and food sector in the era of industry 4.0. *Processes*, 2021, т. 9, № 2,

с. 381–394. URL: https://www.mdpi.com/2227–9717/9/2/381 (дата звернення: 16.04.2025).

ДОДАТОК А

```
# core_module.py

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.metrics import (accuracy_score, precision_score, recall_score, f1_score,
                             roc_curve, confusion_matrix, roc_auc_score)
import pickle

def fetch_dataset(path='data/zomato.csv'):
    return pd.read_csv(path)

def initial_preprocessing(data):
    data = data.copy()
    data['approx_cost'] = data['approx_cost(for two people)'].astype(str).str.replace(',', '')
    data['approx_cost'] = data['approx_cost'].astype(float)
    data['rate_val'] = data['rate'].astype(str).apply(lambda val: val.split('/')[0])

    while True:
        try:
            data['rate_val'] = data['rate_val'].astype(float)
            break
        except ValueError as err:
            invalid_val = str(err).split(":")[–1].strip().replace("'", "")
```

```
            print(f"Handling error in rating: {invalid_val}")
            data['rate_val'] = data['rate_val'].replace(invalid_val, np.nan)

    return data

def display_dataset():
    print(fetch_dataset())

def convert_categorical(col):
    unique_dict = {}
    return [unique_dict.setdefault(val, len(unique_dict)+1) for val in col]

def preprocess_for_training(data):
    data['target'] = data['rate_val'].apply(lambda x: 1 if float(x) > 3.75 else 0)
    data['rest_type'] = data['rest_type'].fillna('').astype(str)
    data['rest_type_count'] = data['rest_type'].str.split(',').apply(len)
    data['cuisines'] = data['cuisines'].fillna('').astype(str).str.split(',').apply(len)
    data['listed_in(type)'] = convert_categorical(data['listed_in(type)'])
    data['listed_in(city)'] = convert_categorical(data['listed_in(city)'])
    data['rest_type'] = convert_categorical(data['rest_type'])
    data['online_order'] = data['online_order'].replace({"Yes": 1, "No": 2})
    data['book_table'] = data['book_table'].replace({"Yes": 1, "No": 2})

    selected = ['online_order', 'book_table', 'rest_type', 'rest_type_count',
                'cuisines', 'listed_in(type)', 'listed_in(city)', 'approx_cost', 'target']
    return data[selected].dropna()

def visualize_distributions(data):
    fig, axes = plt.subplots(3, 2, figsize=(14, 10))
    sns.histplot(data['votes'], kde=True, ax=axes[0][0])
```

```
            sns.boxplot(x=data['votes'], ax=axes[0][1])
            sns.histplot(data['approx_cost'], kde=True, ax=axes[1][0])
            sns.boxplot(x=data['approx_cost'], ax=axes[1][1])
            sns.histplot(data['rate_val'], kde=True, ax=axes[2][0])
            sns.boxplot(x=data['rate_val'], ax=axes[2][1])
            plt.tight_layout()
            plt.show()plt.xticks(rotation=30)
        plt.show()
        fig = plt.figure(figsize=(5, 10))
        ax1 = fig.add_subplot(1, 1, 1)
        location_counts = eda_first()['listed_in(type)'].value_counts()
        sorted_locations = location_counts.index
        sns.countplot(y=eda_first()['listed_in(type)'], order=sorted_locations,
ax=ax1)
        plt.xticks(rotation=30)
        plt.show()
        fig = plt.figure(figsize=(10, 3))
        ax1 = fig.add_subplot(1, 1, 1)
        sns.boxplot(x=eda_first()['rate_num'], y=eda_first()['online_order'])
        plt.show()
        fig = plt.figure(figsize=(10, 3))
        ax1 = fig.add_subplot(1, 1, 1)
        sns.boxplot(x=eda_first()['rate_num'], y=eda_first()['book_table'])
        plt.show()
        def replace_cat(col):
        dict_1 = {}
        ids_1 = []
        for index, i in enumerate(col):
        if i not in dict_1:
        dict_1[i] = len(dict_1) + 1
```

```
        ids_1.append(dict_1[i])
        return ids_1
        # Вторинна обробка
        def eda_second():
        train_df = eda_first().copy()
        train_df['target'] = train_df['rate_num'].apply(lambda x: 1 if float(x) > 3.75
else 0)
        train_df['rest_type'] = train_df['rest_type'].fillna('')
        train_df['rest_type'] = train_df['rest_type'].astype(str)
        train_df['rest_type_total'] = train_df['rest_type'].str.split(',').apply(lambda x:
len(x))
        train_df['cuisines'] = train_df['cuisines'].fillna('')
        train_df['cuisines'] = train_df['cuisines'].astype(str)
        train_df['cuisines'] = train_df['cuisines'].str.split(',').apply(lambda x: len(x))
        train_df['listed_in(type)'] = replace_cat(train_df['listed_in(type)'])
        train_df['listed_in(city)'] = replace_cat(train_df['listed_in(city)'])
        train_df['rest_type'] = replace_cat(train_df['rest_type'])
        train_df["online_order"] = train_df["online_order"].replace({"Yes": 1, "No":
2})
        train_df["book_table"] = train_df["book_table"].replace({"Yes": 1, "No": 2})
        train_df = train_df[['online_order', 'book_table', 'rest_type', 'rest_type_total',
        'cuisines', 'listed_in(type)', 'listed_in(city)',
        'approx_cost', 'target']]
        train_df.dropna(how='any', inplace=True)
        return train_df
        # Збереження датасету у змінну
        df2 = eda_second()
        # Мапа кореляції
        def corr_map():
        correlation = eda_second().corr().round(2)
```

```
        plt.figure(figsize=(14, 7))
        sns.heatmap(correlation, annot=True, cmap='YlOrBr')
        plt.show()
        X = eda_second().drop('target', axis=1)
        Y = eda_second()['target']
        X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=.20, random_state=42)
        # Навчання Random Forest
        # # rf_model = RandomForestClassifier()
        # # rf_model.fit(X_train, Y_train)
        # rf_prediction = rf_model.predict(X_test)
        # Навчання SVC
        # # svc_model = SVC()
        # # svc_model.fit(X_train, Y_train)
        # svc_prediction = svc_model.predict(X_test)
        # Навчання Gaussian
        # # gaus_model = GaussianNB()
        # # gaus_model.fit(X_train, Y_train)
        # gaus_prediction = gaus_model.predict(X_test)
        # Навчання Logistic Regression
        # # lr_model = LogisticRegression()
        # # lr_model.fit(X_train, Y_train)
        # lr_prediction = lr_model.predict(X_test)
        # Навчання Decision Tree
        # # dt_model = DecisionTreeClassifier()
        # # dt_model.fit(X_train, Y_train)
        # dt_prediction = dt_model.predict(X_test)
        # Навчання Gradient Boosting
        # # gb_model = GradientBoostingClassifier()
        # # gb_model.fit(X_train, Y_train)
```

```
# gb_prediction = gb_model.predict(X_test)
# Пошук гіперпараметрів Random Forest
# param_grid = {
# 'n_estimators': [100, 200, 500],
# 'max_features': ['auto', 'sqrt'],
# 'max_depth' : [4,6,8, None],
# 'criterion' :['gini', 'entropy']
# }
#
# grid = GridSearchCV(RandomForestClassifier(), param_grid, cv=5, refit=True, verbose=3)
# grid.fit(X_train, Y_train)
# print(grid.best_params_)
# grid_predictions = grid.predict(X_test)
# Пошук гіперпараметрів SVC
# param_grid = {'C': [0.1, 1, 10, 100],
# 'gamma': [1, 0.1, 0.01, 0.001],
# 'kernel': ['rbf']}
#
# grid = GridSearchCV(SVC(), param_grid, refit=True, verbose=3)
# grid.fit(X_train, Y_train)
# print(grid.best_params_)
# grid_predictions = grid.predict(X_test)
# Пошук гіперпараметрів Gaussian
# param_grid = {'var_smoothing': np.logspace(0,–9, num=100)}
#
# grid = GridSearchCV(GaussianNB(), param_grid,cv=5, refit=True, verbose=3)
# grid.fit(X_train, Y_train)
# print(grid.best_params_)
```

```
        # grid_predictions = grid.predict(X_test)
        # Пошук гіперпараметрів Logistic Regression
        # param_grid = {
        # 'C': np.logspace(–4, 4, 8),
        # 'penalty': ['l1', 'l2'],
        # 'solver': ['newton–cg', 'lbfgs', 'liblinear', 'sag', 'saga']
        # }
        #
        # grid = GridSearchCV(LogisticRegression(), param_grid,cv= 5, refit=True, verbose=3)
        # grid.fit(X_train, Y_train)
        # print(grid.best_params_)
        # grid_predictions = grid.predict(X_test)
        # Пошук гіперпараметрів Decision Tree
        # param_grid = {
        # 'max_depth': [None, 5, 10],
        # 'min_samples_split': [2, 5, 10],
        # 'min_samples_leaf': [1, 2, 4],
        # 'criterion': ['gini', 'entropy']
        # }
        #
        # model = DecisionTreeClassifier()
        # grid_search = GridSearchCV(model, param_grid, cv=5, refit=True, verbose=3)
        # grid_search.fit(X_train, Y_train)
        # print(grid_search.best_params_)
        # grid_predictions = grid_search.predict(X_test)
        # Пошук гіперпараметрів Gradient Boosting
        # param_grid = {
        # 'n_estimators': [50, 100, 200],
```

```
        # 'learning_rate': [0.1, 0.01, 0.001],
        # 'max_depth': [3, 5, 7]
        # }
        #
        # model = GradientBoostingClassifier()
        # grid_search = GridSearchCV(model, param_grid, cv=5, refit=True, verbose=3)
        # grid_search.fit(X_train, Y_train)
        # print(grid_search.best_params_)
        # grid_predictions = grid_search.predict(X_test)
        # Навчання налаштованної Random Forest
        # rf_model_1 = RandomForestClassifier(criterion= 'gini', max_depth= 8, max_features= 'sqrt', n_estimators= 200)
        # rf_model_1.fit(X_train, Y_train)
        # Збереження в файл моделі Random Forest
        # with open('RF_model_best.pkl', 'wb') as file:
        # pickle.dump(rf_model, file)
        # Навчання налаштованної SVC
        # svc_model_1 = SVC(C= 10, gamma= 0.1, kernel= 'rbf')
        # svc_model_1.fit(X_train, Y_train)
        # Збереження в файл моделі SVC
        # with open('SVC_model_best.pkl', 'wb') as file:
        # pickle.dump(svc_model_1, file)
        # Навчання налаштованної Gaussian
        # # gaus_model_1 = GaussianNB(var_smoothing= 6.579332246575682e–08)
        # # gaus_model_1.fit(X_train, Y_train)
        # Збереження в файл моделі Gaussian
        # with open('Gaus_model_best.pkl', 'wb') as file:
        # pickle.dump(gaus_model_1, file)
        # Навчання налаштованної Logistic Regression
```

```
# lr_model_1 = LogisticRegression(C= 0.019306977288832496, penalty= 'l1', solver= 'liblinear')
# lr_model_1.fit(X_train, Y_train)
# Збереження в файл моделі Logistic Regression
# with open('LR_model_best.pkl', 'wb') as file:
# pickle.dump(lr_model_1, file)
# Навчання налаштованної Decision Tree
# dt_model_1 = DecisionTreeClassifier(criterion= 'entropy', max_depth= None, min_samples_leaf= 1, min_samples_split= 10)
# dt_model_1.fit(X_train, Y_train)
# Збереження в файл моделі Decision Tree
# with open('DT_model_best.pkl', 'wb') as file:
# pickle.dump(dt_model, file)
# Навчання налаштованної Gradient Boosting
# gb_model_1 = GradientBoostingClassifier(learning_rate= 0.1, max_depth= 7, n_estimators= 200)
# gb_model_1.fit(X_train, Y_train)
# Збереження в файл моделі Gradient Boosting
# with open('GB_model_best.pkl', 'wb') as file:
# pickle.dump(gb_model_1, file)
# Вибір моделі
def choose_model(X_test, model_file):
with open(model_file, 'rb') as file:
rf_model = pickle.load(file)
prediction = rf_model.predict(X_test)
return prediction
# Збереження моделей у змінні
rf_prediction = choose_model(X_test, 'RF_model_best.pkl')
svc_prediction_1 = choose_model(X_test, 'SVC_model_best.pkl')
gaus_prediction_1 = choose_model(X_test, 'Gaus_model_best.pkl')
```

```
lr_prediction_1 = choose_model(X_test, 'LR_model_best.pkl')
dt_prediction = choose_model(X_test, 'DT_model_best.pkl')
gb_prediction_1 = choose_model(X_test, 'GB_model_best.pkl')
# Текстові метрики
def evaluate_binary_classification(predictions):
# Accuracy
accuracy = accuracy_score(Y_test, predictions)
print("Accuracy:", accuracy)
# Precision
precision = precision_score(Y_test, predictions)
print("Precision:", precision)
# Recall
recall = recall_score(Y_test, predictions)
print("Recall:", recall)
# F1–Score
f1 = f1_score(Y_test, predictions)
print("F1–Score:", f1)
# Roc Curve
def roc_plt(predictions, title):
fpr, tpr, thresholds = roc_curve(Y_test, predictions)
roc_auc = roc_auc_score(Y_test, predictions)
plt.plot(fpr, tpr, label='ROC Curve (AUC = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], 'r—')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title(title)
plt.legend()
plt.show()
# Confusion Matrix
def con_matrix_plt(predictions, title):
```

```
labels = [0, 1]
cm = confusion_matrix(Y_test, predictions, labels=labels)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.xticks(ticks=range(len(labels)), labels=labels)
plt.yticks(ticks=range(len(labels)), labels=labels)
plt.title(title)
plt.show()
# метрики Random Forest
def metric_rf():
print('Random Forest:')
evaluate_binary_classification(rf_prediction)
print('\n')
# метрики SVC
def metric_svc():
print('SVC:')
evaluate_binary_classification(svc_prediction_1)
print('\n')
# метрики Gaussian
def metric_gaus():
print('Gaussian:')
evaluate_binary_classification(gaus_prediction_1)
print('\n')
# метрики Logistic Regression
def metric_lr():
print('Logistic Regression:')
evaluate_binary_classification(lr_prediction_1)
```

```
print('\n')
# метрики Decision Tree
def metric_dt():
print('Decision Tree:')
evaluate_binary_classification(dt_prediction)
print('\n')
# метрики Gradient Boosting
def metric_gb():
print('Gradient Boosting:')
evaluate_binary_classification(gb_prediction_1)
print('\n')
# Roc Curve Random Forest
def roc_plt_rf():
roc_plt(rf_prediction, 'Random Forest')
# Roc Curve SVC
def roc_plt_svc():
roc_plt(svc_prediction_1, 'SVC')
# Roc Curve Gaussian
def roc_plt_gaus():
roc_plt(gaus_prediction_1, 'Gaussian')
# Roc Curve Logistic Regression
def roc_plt_lr():
roc_plt(lr_prediction_1, 'Logistic Regression')
# Roc Curve Decision Tree
def roc_plt_dt():
roc_plt(dt_prediction, 'Decision Tree')
# Roc Curve Gradient Boosting
def roc_plt_gb():
roc_plt(gb_prediction_1, 'Gradient Boosting')
# Confusion Matrix Random Forest
```

```
def con_matrix_plt_rf():
con_matrix_plt(rf_prediction, 'Random Forest')
# Confusion Matrix SVC
def con_matrix_plt_svc():
con_matrix_plt(svc_prediction_1, 'SVC')
# Confusion Matrix Gaussian
def con_matrix_plt_gaus():
con_matrix_plt(gaus_prediction_1, 'Gaussian')
# Confusion Matrix Logistic Regression
def con_matrix_plt_lr():
con_matrix_plt(lr_prediction_1, 'Logistic Regression')
# Confusion Matrix Decision Tree
def con_matrix_plt_dt():
con_matrix_plt(dt_prediction, 'Decision Tree')
# Confusion Matrix Gradient Boosting
def con_matrix_plt_gb():
con_matrix_plt(gb_prediction_1, 'Gradient Boosting')
# Вивід всіх метрик
def metrics_all():
print('Random Forest:')
print(evaluate_binary_classification(rf_prediction), '\n')
print('SVC:')
print(evaluate_binary_classification(svc_prediction_1), '\n')
print('Gaussian:')
print(evaluate_binary_classification(gaus_prediction_1), '\n')
print('Logistic Regression:')
print(evaluate_binary_classification(lr_prediction_1), '\n')
print('Decision Tree:')
print(evaluate_binary_classification(dt_prediction), '\n')
print('Gradient Boosting:')
```

```
print(evaluate_binary_classification(gb_prediction_1), '\n')
# Аналіз всіх метрик
def metric_analisis():
models = ['Random Forest', 'SVC', 'Gaussian', 'Logistic Regression', 'Decision Tree', 'Gradient Boosting']
accuracy_scores = [0.7757664233576642, 0.79007299270073, 0.720389294403893, 0.7345985401459854, 0.7825790754257907,
0.8042822384428224]
precision_scores = [0.738858927050809, 0.7940231362467867, 0.761012183692596, 0.7603305785123967,
0.7663324217563051, 0.8284574468085106]
recall_scores = [0.6528718334587409, 0.6197642337597191, 0.4073238023576624, 0.46149987459242536,
0.6325558063707047, 0.6250313518936543]
f1_scores = [0.6932090545938748, 0.6961543879419636, 0.530632249632413, 0.5743717808646792, 0.6930475405331135,
0.7125089349535383]
bar_width = 0.17
bar_positions = np.arange(len(models))
plt.figure(figsize=(10, 6))
plt.bar(bar_positions – bar_width, accuracy_scores, width=bar_width, label='Accuracy')
plt.bar(bar_positions, precision_scores, width=bar_width, label='Precision')
plt.bar(bar_positions + bar_width, recall_scores, width=bar_width, label='Recall')
plt.bar(bar_positions + 2 * bar_width, f1_scores, width=bar_width, label='F1–Score')
plt.xlabel('Модели')
plt.ylabel('Оценки')
plt.title('Сравнение моделей машинного обучения')
```

```
plt.xticks(bar_positions, models)
plt.legend()
plt.show()
```

Файл «interface_1.py»

```
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtWidgets import QTableWidgetItem
from main_1 import *
class OutputRedirector:
def __init__(self, text_widget):
self.text_widget = text_widget
self.stdout = sys.stdout
sys.stdout = self
def write(self, message):
self.text_widget.moveCursor(QtGui.QTextCursor.End)
self.text_widget.insertPlainText(message)
def flush(self):
pass
class Ui_MainWindow(object):
def setupUi(self, MainWindow):
MainWindow.setObjectName("MainWindow")
MainWindow.resize(800, 600)
self.centralwidget = QtWidgets.QWidget(MainWindow)
self.centralwidget.setObjectName("centralwidget")
self.groupBox = QtWidgets.QGroupBox(self.centralwidget)
self.groupBox.setGeometry(QtCore.QRect(10, 0, 271, 141))
self.groupBox.setObjectName("groupBox")
self.pushButton = QtWidgets.QPushButton(self.groupBox)
self.pushButton.setGeometry(QtCore.QRect(30, 20, 201, 31))
self.pushButton.setObjectName("pushButton")
self.pushButton_2 = QtWidgets.QPushButton(self.groupBox)
```

```
self.pushButton_2.setGeometry(QtCore.QRect(30, 60, 201, 31))
self.pushButton_2.setObjectName("pushButton_2")
self.pushButton_3 = QtWidgets.QPushButton(self.groupBox)
self.pushButton_3.setGeometry(QtCore.QRect(30, 100, 201, 31))
self.pushButton_3.setObjectName("pushButton_3")
self.groupBox_2 = QtWidgets.QGroupBox(self.centralwidget)
self.groupBox_2.setGeometry(QtCore.QRect(290, 0, 501, 261))
self.groupBox_2.setObjectName("groupBox_2")
self.pushButton_4 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_4.setGeometry(QtCore.QRect(100, 20, 111, 31))
self.pushButton_4.setObjectName("pushButton_4")
self.pushButton_5 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_5.setGeometry(QtCore.QRect(230, 20, 111, 31))
self.pushButton_5.setObjectName("pushButton_5")
self.pushButton_6 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_6.setGeometry(QtCore.QRect(360, 20, 111, 31))
self.pushButton_6.setObjectName("pushButton_6")
self.pushButton_7 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_7.setGeometry(QtCore.QRect(100, 60, 111, 31))
self.pushButton_7.setObjectName("pushButton_7")
self.pushButton_8 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_8.setGeometry(QtCore.QRect(230, 60, 111, 31))
self.pushButton_8.setObjectName("pushButton_8")
self.pushButton_9 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_9.setGeometry(QtCore.QRect(360, 60, 111, 31))
self.pushButton_9.setObjectName("pushButton_9")
self.label = QtWidgets.QLabel(self.groupBox_2)
self.label.setGeometry(QtCore.QRect(10, 30, 91, 16))
self.label.setObjectName("label")
self.label_2 = QtWidgets.QLabel(self.groupBox_2)
```

```
self.label_2.setGeometry(QtCore.QRect(10, 70, 91, 16))
self.label_2.setObjectName("label_2")
self.label_3 = QtWidgets.QLabel(self.groupBox_2)
self.label_3.setGeometry(QtCore.QRect(10, 110, 91, 16))
self.label_3.setObjectName("label_3")
self.label_4 = QtWidgets.QLabel(self.groupBox_2)
self.label_4.setGeometry(QtCore.QRect(10, 150, 91, 16))
self.label_4.setObjectName("label_4")
self.label_5 = QtWidgets.QLabel(self.groupBox_2)
self.label_5.setGeometry(QtCore.QRect(10, 190, 91, 16))
self.label_5.setObjectName("label_5")
self.label_6 = QtWidgets.QLabel(self.groupBox_2)
self.label_6.setGeometry(QtCore.QRect(10, 230, 91, 16))
self.label_6.setObjectName("label_6")
self.pushButton_10 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_10.setGeometry(QtCore.QRect(100, 100, 111, 31))
self.pushButton_10.setObjectName("pushButton_10")
self.pushButton_11 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_11.setGeometry(QtCore.QRect(230, 100, 111, 31))
self.pushButton_11.setObjectName("pushButton_11")
self.pushButton_12 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_12.setGeometry(QtCore.QRect(360, 100, 111, 31))
self.pushButton_12.setObjectName("pushButton_12")
self.pushButton_13 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_13.setGeometry(QtCore.QRect(100, 140, 111, 31))
self.pushButton_13.setObjectName("pushButton_13")
self.pushButton_14 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_14.setGeometry(QtCore.QRect(230, 140, 111, 31))
self.pushButton_14.setObjectName("pushButton_14")
self.pushButton_15 = QtWidgets.QPushButton(self.groupBox_2)
```

```
self.pushButton_15.setGeometry(QtCore.QRect(360, 140, 111, 31))
self.pushButton_15.setObjectName("pushButton_15")
self.pushButton_16 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_16.setGeometry(QtCore.QRect(100, 180, 111, 31))
self.pushButton_16.setObjectName("pushButton_16")
self.pushButton_17 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_17.setGeometry(QtCore.QRect(230, 180, 111, 31))
self.pushButton_17.setObjectName("pushButton_17")
self.pushButton_18 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_18.setGeometry(QtCore.QRect(360, 180, 111, 31))
self.pushButton_18.setObjectName("pushButton_18")
self.pushButton_19 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_19.setGeometry(QtCore.QRect(100, 220, 111, 31))
self.pushButton_19.setObjectName("pushButton_19")
self.pushButton_20 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_20.setGeometry(QtCore.QRect(230, 220, 111, 31))
self.pushButton_20.setObjectName("pushButton_20")
self.pushButton_21 = QtWidgets.QPushButton(self.groupBox_2)
self.pushButton_21.setGeometry(QtCore.QRect(360, 220, 111, 31))
self.pushButton_21.setObjectName("pushButton_21")
self.groupBox_3 = QtWidgets.QGroupBox(self.centralwidget)
self.groupBox_3.setGeometry(QtCore.QRect(10, 140, 271, 121))
self.groupBox_3.setObjectName("groupBox_3")
self.pushButton_22 = QtWidgets.QPushButton(self.groupBox_3)
self.pushButton_22.setGeometry(QtCore.QRect(30, 40, 201, 31))
self.pushButton_22.setObjectName("pushButton_22")
self.pushButton_23 = QtWidgets.QPushButton(self.groupBox_3)
self.pushButton_23.setGeometry(QtCore.QRect(30, 80, 201, 31))
self.pushButton_23.setObjectName("pushButton_23")
self.groupBox_4 = QtWidgets.QGroupBox(self.centralwidget)
```

```
        self.groupBox_4.setGeometry(QtCore.QRect(10, 260, 781, 291))
        self.groupBox_4.setObjectName("groupBox_4")
        self.textEdit = QtWidgets.QTextEdit(self.groupBox_4)
        self.textEdit.setGeometry(QtCore.QRect(10, 20, 261, 261))
        self.textEdit.setObjectName("textEdit")
        self.tableWidget = QtWidgets.QTableWidget(self.groupBox_4)
        self.tableWidget.setGeometry(QtCore.QRect(280, 20, 491, 261))
        self.tableWidget.setObjectName("tableWidget")
        self.tableWidget.setColumnCount(0)
        self.tableWidget.setRowCount(0)
        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QtWidgets.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 800, 21))
        self.menubar.setObjectName("menubar")
        MainWindow.setMenuBar(self.menubar)
        self.statusbar = QtWidgets.QStatusBar(MainWindow)
        self.statusbar.setObjectName("statusbar")
        MainWindow.setStatusBar(self.statusbar)
        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)
        self.output_redirector = OutputRedirector(self.textEdit)
        def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "MainWindow"))
        self.groupBox.setTitle(_translate("MainWindow", "Дані"))
        self.pushButton.setText(_translate("MainWindow", "Показати датасет"))
        self.pushButton_2.setText(_translate("MainWindow", "Показати
оброблений датасет"))
        self.pushButton_3.setText(_translate("MainWindow", "Візуалізація
даних"))
```

```
        self.groupBox_2.setTitle(_translate("MainWindow", "Моделі"))
        self.pushButton_4.setText(_translate("MainWindow", "Метрики"))
        self.pushButton_5.setText(_translate("MainWindow", "Confusion Matrix"))
        self.pushButton_6.setText(_translate("MainWindow", "ROC Curve"))
        self.pushButton_7.setText(_translate("MainWindow", "Метрики"))
        self.pushButton_8.setText(_translate("MainWindow", "Confusion Matrix"))
        self.pushButton_9.setText(_translate("MainWindow", "ROC Curve"))
        self.label.setText(_translate("MainWindow", "Random Forest"))
        self.label_2.setText(_translate("MainWindow", "SVC"))
        self.label_3.setText(_translate("MainWindow", "Gaussian"))
        self.label_4.setText(_translate("MainWindow", "Logistic Regression"))
        self.label_5.setText(_translate("MainWindow", "Decisioon Tree"))
        self.label_6.setText(_translate("MainWindow", "Gradient Boosting"))
        self.pushButton_10.setText(_translate("MainWindow", "Метрики"))
        self.pushButton_11.setText(_translate("MainWindow", "Confusion
Matrix"))
        self.pushButton_12.setText(_translate("MainWindow", "ROC Curve"))
        self.pushButton_13.setText(_translate("MainWindow", "Метрики"))
        self.pushButton_14.setText(_translate("MainWindow", "Confusion
Matrix"))
        self.pushButton_15.setText(_translate("MainWindow", "ROC Curve"))
        self.pushButton_16.setText(_translate("MainWindow", "Метрики"))
        self.pushButton_17.setText(_translate("MainWindow", "Confusion
Matrix"))
        self.pushButton_18.setText(_translate("MainWindow", "ROC Curve"))
        self.pushButton_19.setText(_translate("MainWindow", "Метрики"))
        self.pushButton_20.setText(_translate("MainWindow", "Confusion
Matrix"))
        self.pushButton_21.setText(_translate("MainWindow", "ROC Curve"))
        self.groupBox_3.setTitle(_translate("MainWindow", "Аналіз"))
```

```
        self.pushButton_22.setText(_translate("MainWindow", "Метрики всіх моделей"))
        self.pushButton_23.setText(_translate("MainWindow", "Аналіз всіх метрик"))
        self.groupBox_4.setTitle(_translate("MainWindow", "Вивід"))
        def load_dataframe(self, df):
        num_rows = df.shape[0]
        num_columns = df.shape[1]
        self.tableWidget.setRowCount(num_rows)
        self.tableWidget.setColumnCount(num_columns)
        for i in range(num_rows):
        for j in range(num_columns):
        item = QTableWidgetItem(str(df.iat[i, j]))
        self.tableWidget.setItem(i, j, item)
        self.tableWidget.setHorizontalHeaderLabels(df.columns)
        def load_dataframe1(self):
        self.df1 = load_data()
        self.load_dataframe(self.df1)
        def load_dataframe2(self):
        self.df2 = eda_second()
        self.load_dataframe(self.df2)
        def connects(self, MainWindow):
        self.pushButton.clicked.connect(self.load_dataframe1)
        self.pushButton_2.clicked.connect(self.load_dataframe2)
        self.pushButton_3.clicked.connect(visualisation)
        self.pushButton_4.clicked.connect(metric_rf)
        self.pushButton_5.clicked.connect(con_matrix_plt_rf)
        self.pushButton_6.clicked.connect(roc_plt_rf)
        self.pushButton_7.clicked.connect(metric_svc)
        self.pushButton_8.clicked.connect(con_matrix_plt_svc)
```

```
self.pushButton_9.clicked.connect(roc_plt_svc)
self.pushButton_10.clicked.connect(metric_gaus)
self.pushButton_11.clicked.connect(con_matrix_plt_gaus)
self.pushButton_12.clicked.connect(roc_plt_gaus)
self.pushButton_13.clicked.connect(metric_lr)
self.pushButton_14.clicked.connect(con_matrix_plt_lr)
self.pushButton_15.clicked.connect(roc_plt_lr)
self.pushButton_16.clicked.connect(metric_dt)
self.pushButton_17.clicked.connect(con_matrix_plt_dt)
self.pushButton_18.clicked.connect(roc_plt_dt)
self.pushButton_19.clicked.connect(metric_gb)
self.pushButton_20.clicked.connect(con_matrix_plt_gb)
self.pushButton_21.clicked.connect(roc_plt_gb)
self.pushButton_22.clicked.connect(metrics_all)
self.pushButton_23.clicked.connect(metric_analisis)
if __name__ == "__main__":
import sys
app = QtWidgets.QApplication(sys.argv)
MainWindow = QtWidgets.QMainWindow()
ui = Ui_MainWindow()
ui.setupUi(MainWindow)
ui.connects(MainWindow)
MainWindow.show()
sys.exit(app.exec_())
```

ДОДАТОК Б. ПРЕЗЕНТАЦІЯ

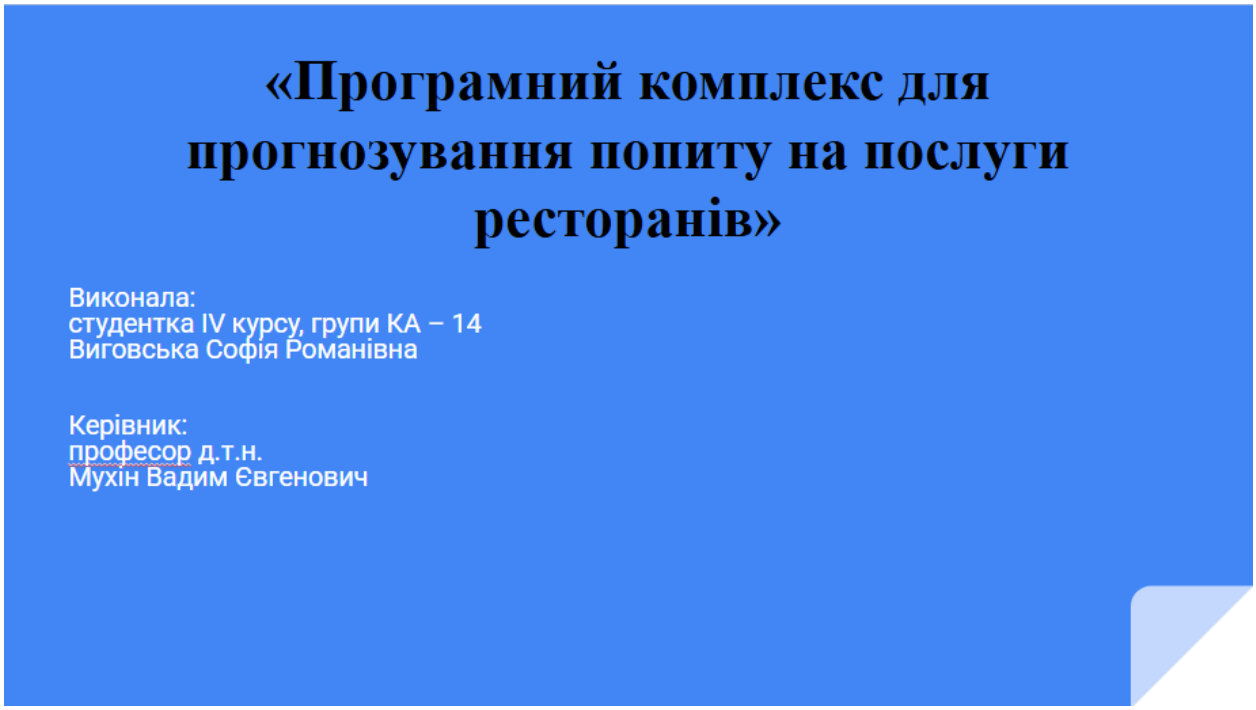

Рисунок Б.1 – Слайд 1

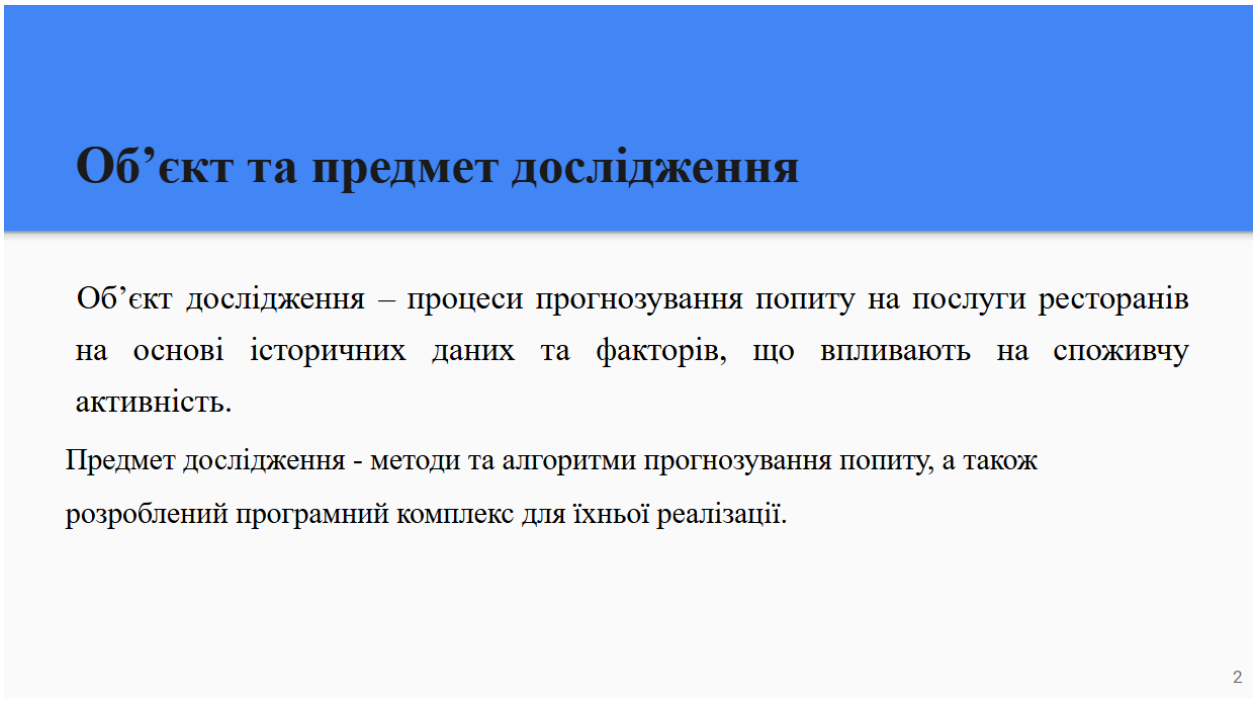

Рисунок Б.2 – Слайд 2

Мета роботи та метод дослідження

Мета дослідження – розробка програмного комплексу для прогнозування попиту на послуги ресторанів, що дозволить підвищити ефективність управління ресурсами, оптимізувати бізнес-процеси та покращити якість обслуговування клієнтів.

Методи дослідження

1. Аналіз літературних джерел та огляд існуючих підходів до прогнозування попиту.
2. Обробка та аналіз історичних даних ресторанного бізнесу.
3. Методи машинного навчання (лінійна регресія, нейронні мережі, ARIMA, Prophet тощо).
4. Статистичне моделювання та прогнозування часових рядів.
5. Оцінка ефективності прогнозних моделей на основі метрик точності (MAE, MSE, RMSE).

3

Рисунок Б.3 – Слайд 3

Актуальність роботи

В умовах зростаючої конкуренції в ресторанному бізнесі точне прогнозування попиту стає критичним для підвищення ефективності роботи закладів. Традиційні методи більше не відповідають вимогам динамічного ринку, тому виникає потреба у використанні сучасних підходів до аналізу даних. Розробка програмного комплексу для прогнозування попиту дозволить автоматизувати обробку інформації, враховувати різноманітні фактори та підвищити якість управлінських рішень, сприяючи оптимізації ресурсів і зростанню прибутковості ресторанів.

4

Рисунок Б.4 – Слайд 4

Завдання дослідження

1. Провести аналіз існуючих методів прогнозування попиту у сфері ресторанного бізнесу.
2. Визначити основні фактори, що впливають на рівень попиту на послуги ресторанів.
3. Зібрати, підготувати та проаналізувати історичні дані для побудови прогнозних моделей.
4. Розробити програмний комплекс, що використовує методи машинного навчання для прогнозування попиту.
5. Провести тестування ефективності моделей та оцінити їхню точність.
6. Надати рекомендації щодо впровадження програмного комплексу у ресторанний бізнес.

5

Рисунок Б.5 – Слайд 5

№	Вимога	Опис
1	Завантаження даних	Імпорт даних із CSV, Excel, баз даних
2	Попередня обробка даних	Очищення, нормалізація, обробка пропущених значень
3	Вибір моделі прогнозування	Можливість обрати алгоритм: регресія, дерева рішень, нейромережі, Prophet
4	Побудова прогнозу	Формування прогнозу на основі обраної моделі
5	Візуалізація результатів	Графіки, діаграми, аналітичні таблиці
6	Експорт результатів	Збереження прогнозів у форматах CSV, Excel, PDF
7	Збереження історії прогнозів	Можливість зберігати попередні результати аналізу

Функціональні вимоги до програмного комплексу

6

Рисунок Б.6 – Слайд 6

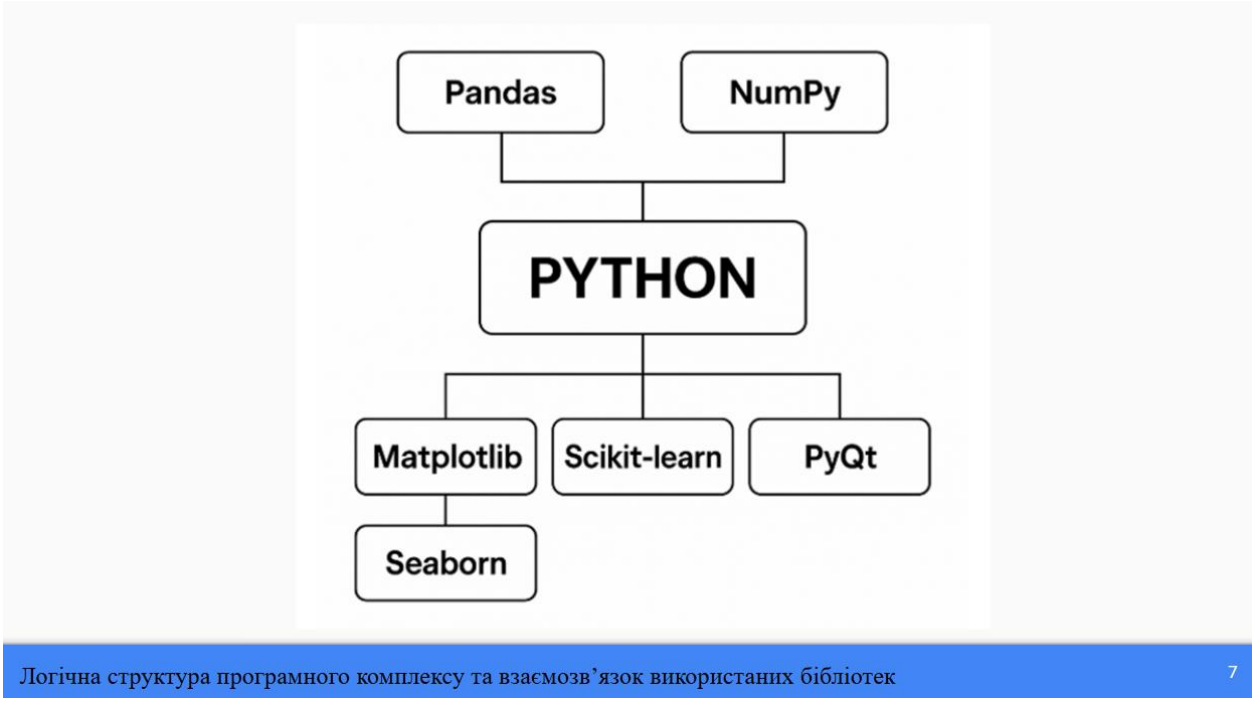

Рисунок Б.7 – Слайд 7

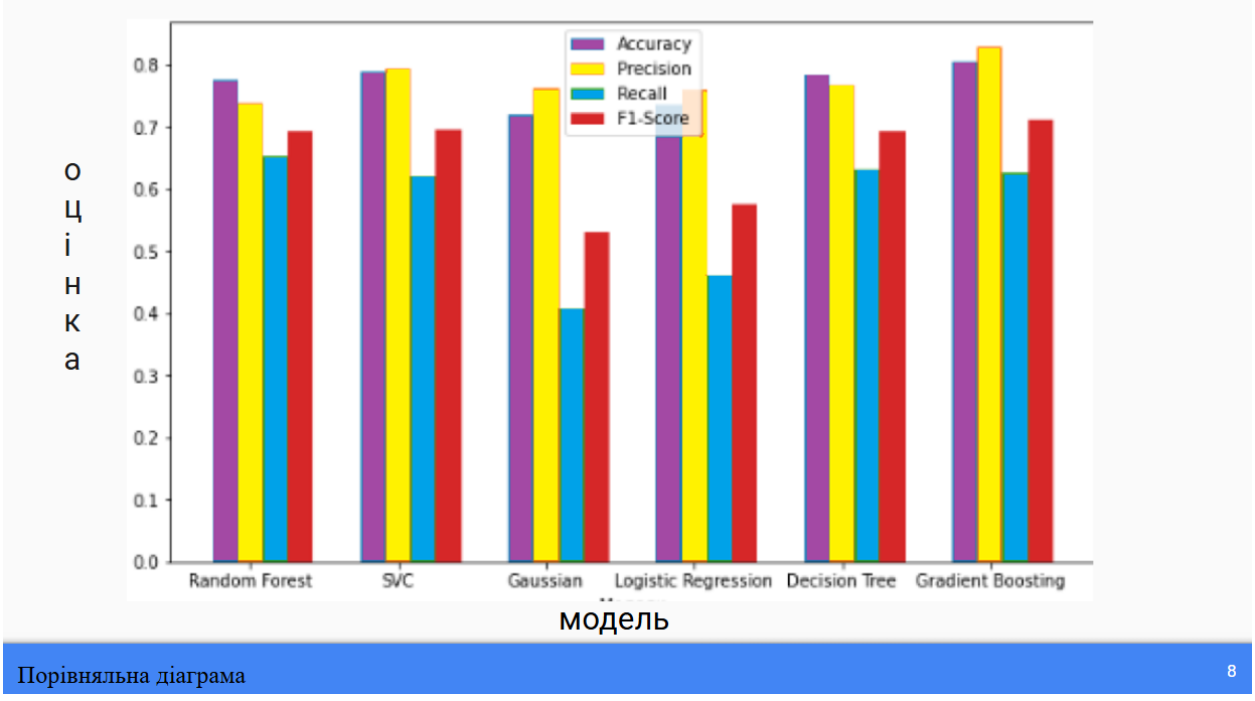

Рисунок Б.8 – Слайд 8

Рисунок Б.9 – Слайд 9

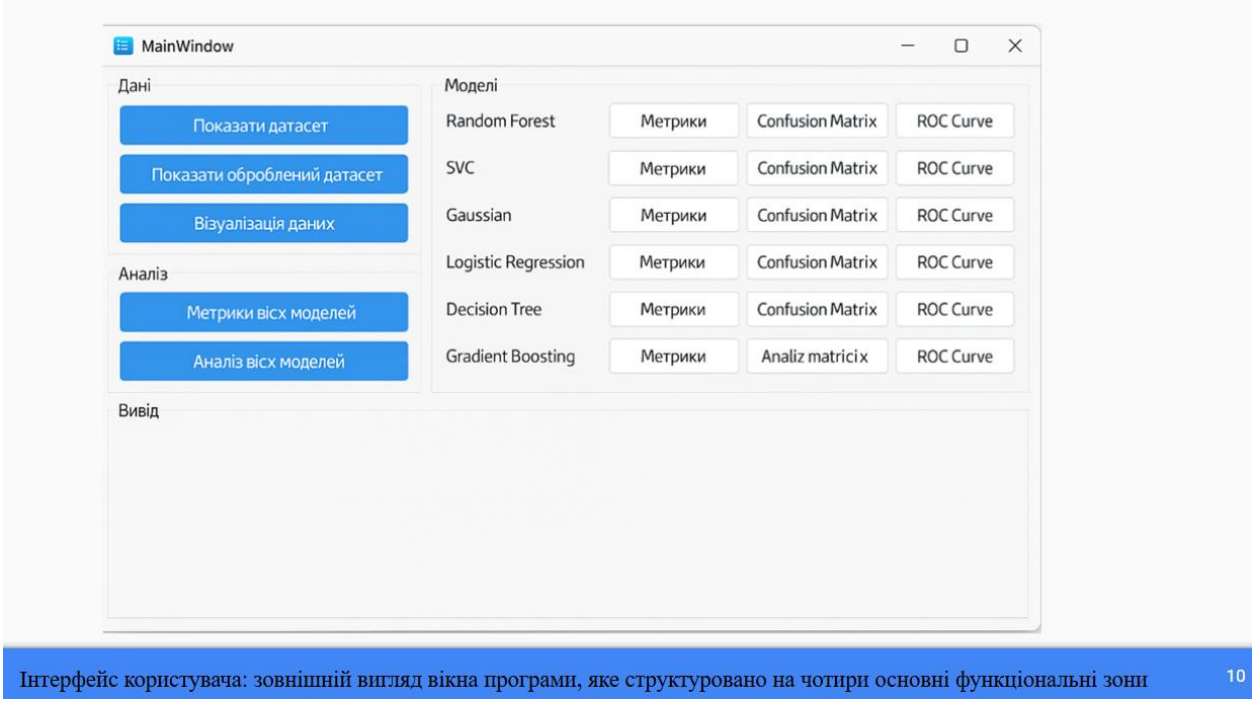

Рисунок Б.10 – Слайд 10

Інтеграція з переліченими системами дозволяє аналітичному модулю перетворитися з локального рішення у ключовий інструмент управління ресторанним бізнесом на основі даних.

Тип системи	**Функціональна роль**
POS-системи (Point of Sale)	Фіксація транзакцій у реальному часі, джерело даних про замовлення, час, середній чек
CRM-системи	Облік інформації про клієнтів, їхні вподобання, історія покупок
ERP-системи	Управління логістикою, фінансами, персоналом
Системи обліку складу	Контроль залишків продуктів, автоматизація поповнення запасів
BI-платформи	Побудова інтерактивної аналітики, графіків, дашбордів для прийняття рішень

Системи, до яких можлива інтеграція 11

Рисунок Б.11 – Слайд 11

Механізм інтеграції	Опис
API	Взаємодія між системами через стандартизовані запити (наприклад, REST API, JSON, XML)
ETL-процеси	Регулярне завантаження даних з зовнішніх джерел, трансформація та збереження у локальному сховищі
Вебхуки	Автоматичні сповіщення про події у зовнішніх системах, що ініціюють дії у аналітичному модулі
SQL-з'єднання	Прямий доступ до бази даних зовнішньої системи через запити для вибірки або оновлення

Основні способи реалізації інтеграції 12

Рисунок Б.12 – Слайд 12

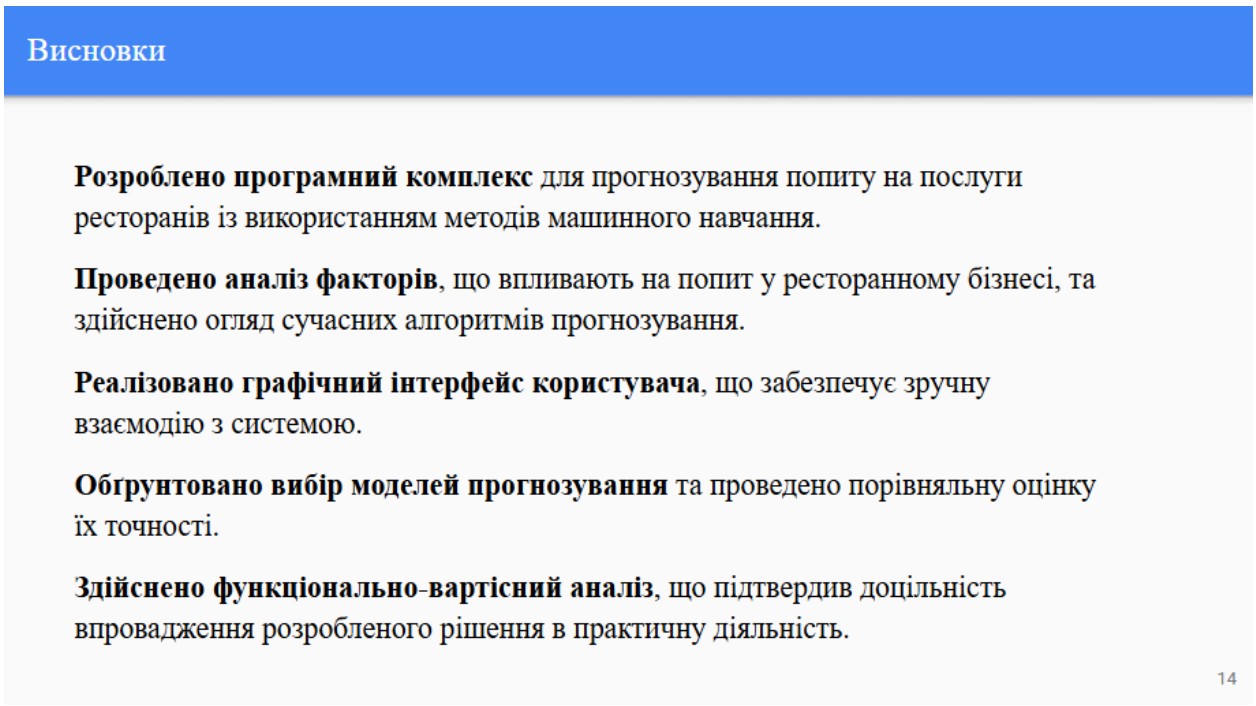

Рисунок Б.13 – Слайд 13