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

Магістерська кваліфікаційна робота

другий (магістерський)
(рівень вищої освіти)

на тему: **Інформаційна система прогнозування попиту на харчові продукти**

Виконав: студент VI курсу, групи КН-61м
спеціальності
122 – "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)

Бродзінський Ю.Р.
(прізвище та ініціали)

Керівник Процах Н.П.
(прізвище та ініціали)

Рецензент Салапак В.М.
(прізвище та ініціали)

Львів – 2025 р.

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІ комп'ютерних наук та інформаційних технологій

Кафедра комп'ютерних наук

Рівень вищої освіти другий (магістерський)

Спеціальність 122 "Комп'ютерні науки"
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри КН

Борецька І. Б.
"10" грудня 2025 року

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

Бродзінський Юрій Русланович
(прізвище, ім'я, по батькові)

1. Тема роботи **Інформаційна система прогнозування попиту на харчові продукти**

керівник роботи Процах Н.П, доктор. фіз.-мат. наук, професор.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

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

2. Термін подання студентом роботи 10.12. 2025 р.

3. Вихідні дані до роботи:

- вивчити предметну область, проаналізувати існуючі фактори які впливають на попит харчових продуктів, а також відповідні програмні продукти;
- розглянути і використати алгоритми, які лежать в основі математичної моделі інформаційної системи;
- спроектувати інформаційну систему з допомогою мови програмування Python та відповідних бібліотек;
- представити результати роботи інформаційної системи.

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

Розділ 1. Стан проблемної області

Розділ 2. Інформаційне забезпечення

Розділ 3. Математичне забезпечення

Розділ 4. Програмне забезпечення

Розділ 5. Розроблення стартап-проєкту

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

додаток А, додаток Б

6. Дата видачі завдання 1 травня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

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

Студент _____ Бродзінський Ю.Р.
(підпис) (прізвище та ініціали)

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

АНОТАЦІЯ

Магістерська робота містить 66 сторінок пояснювальної записки, 5 рисунків, 5 таблиць, 2 додатки, 16 джерел.

Розроблено інформаційну систему прогнозування попиту на харчові продукти з використанням сучасних методів аналізу даних та машинного навчання. В роботі створено математичну модель, яка враховує трендові та сезонні коливання споживчого попиту, а також реалізовано програмне забезпечення на базі бібліотек Python. Впроваджена система дозволяє автоматизувати процес обробки даних і забезпечує зручний інтерфейс для користувачів. Результати тестування підтверджують високий рівень точності прогнозів, що має значний практичний потенціал.

Ключові слова:*часові ряди, Python, прогнозування попиту на продукти, індикатори тренду, сезонність*.

ABSTRACT

The thesis contains 66 pages of explanatory note, 5 figures, 5 tables, 2 appendix, 16 used literary sources.

An information system for forecasting demand for food products has been developed using modern methods of data analysis and machine learning. The work has created a mathematical model that takes into account trend and seasonal fluctuations in consumer demand, and also implemented software based on Python libraries. The implemented system allows you to automate the data processing process and provides a convenient interface for users. The testing results confirm the high level of forecast accuracy, which has significant practical potential.

Keywords:*time series, Python, forecasting demand for products, trend indicators, seasonality*.

ТЕХНІЧНЕ ЗАВДАННЯ

В дипломній роботі потрібно розробити інформаційну систему прогнозування попиту на продукти харчування, для цього потрібно вирішити такі завдання.

1. Провести огляд сучасних методів і моделей прогнозування попиту на харчові продукти та їх технічного застосування.

2. Розробити математичну модель системи прогнозування, що враховує трендові та сезонні коливання на продукти харчування.

3. Реалізувати програмний модуль обробки та аналізу даних із використанням бібліотек Python, таких як Pandas, Numpy, Statsmodels.

4. Створити інтерфейс користувача на основі Streamlit для зручного доступу до функціоналу системи.

5. Провести тестування та валідацію розробленої системи на історичних даних для оцінки її точності та стабільності.

6. Проаналізувати результати прогнозування та запропонувати рекомендації щодо використання системи у практичній діяльності.

.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ ……. 8
ВСТУП……. 9
РОЗДІЛ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ…… 11
1.1. Методи прогнозування ринку харчових продуктів …….. 11
1.2. Аналіз та прогнозування рівня продовольчого забезпечення населення ….. 14
Висновки до розділу……... 18
РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ……. 19
2.1. Опис набору даних ……... 19
2.2. Метод машинного навчання LSTM …… 20
2.3. Алгоритмічне забезпечення інформаційної системи ……... 21
Висновки до розділу……... 24
РОЗДІЛ 3. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ……. 25
3.1. Математичне забезпечення для інформаційної системи прогнозування попиту на продукти харчування …… 25
Висновки до розділу……... 30
РОЗДІЛ 4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ…… 31
4.1. Розроблення інформаційної системи для прогнозування попиту на продукти харчування ……. 31
4.2. Розроблення графічного інтерфейсу інформаційної системи ……. 37
Висновки до розділу……... 43
РОЗДІЛ 5. РОЗРОБЛЕННЯ СТАРТАП-ПРОЄКТУ …… 44
5.1. Структура проєкту інформаційної системи для прогнозування попиту на продукти харчування …….. 44
5.2. Бюджетне планування та ресурсне забезпечення ……. 46
5.3. Етапи розвитку стартапу ……. 46
Висновки до розділу ……... 49
ВИСНОВКИ …… 50
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ……... 51

ДОДАТКИ 53
ДОДАТОК А 53
ДОДАТОК Б 65

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

- ANN (Artificial Neural Network) – штучна нейронна мережа;
- API (Application Programming Interface) – інтерфейс прикладного програмування;
- CNN (Convolutional Neural Network) – згорткова нейронна мережа;
- GPU Graphical Processing Unit – графічний процесор;
- ML (machine learning)–машинне навчання;
- ReLU (Rectified Linear Unit) – активаційна функція;
- ІСЦ – індекс споживчих цін;
- БІСЦ – базовийіндексспоживчихцін;
- SVM – методопорнихвекторів (Support Vector Machine);
- SVR – регресіяопорноговектора (Support Vector Regression).

ВСТУП

Актуальністьдипломної роботи

В сучасних умовах швидких змін на ринку харчових продуктів важливо мати точні інструменти для прогнозування попиту. Ефективне управління запасами та планування виробництва залежить від надійних моделей прогнозування. Застосування сучасних аналітичних методів дозволяє зменшити втрати та підвищити конкурентоспроможність підприємств. Аналіз ринкових тенденцій допомагає адаптуватися до змін уподобань покупців і сезонних коливань. Розробка автоматизованих систем прогнозування сприяє зменшенню людських помилок у процесах планування. Використання машинного навчання та статистичних моделей дозволяє отримати більш точні та обґрунтовані прогнози. Врахування соціально-економічних факторів і регіональних особливостей підвищує достовірність прогнозів. Впровадження таких систем має велике значення для підвищення ефективності роботи підприємств харчової галузі. Розробка системи прогнозування є сучасним необхідним інструментом для прийняття стратегічних рішень у сфері харчової промисловості.

Предмет дослідження – динаміка цін на продукти харчування, а також методи прогнозування їхньої зміни з використанням моделей часових рядів.

Об’єктдослідження – часові ряди цін на продукти харчування.

Метароботи – розроблення інформаційної системи для аналізу цін на продукти, застосування моделей прогнозування для оцінки майбутніх змін вартості цих показників.

Завдання:

1. Провести огляд сучасних методів і моделей прогнозування попиту на харчові продукти та їх технічного застосування.

2. Розробити математичну модель системи прогнозування, що враховує трендові та сезонні коливання на продукти харчування.

3. Реалізувати програмний модуль обробки та аналізу даних із використанням бібліотек Python, таких як Pandas, Numpy, Statsmodels.

4. Створити інтерфейс користувача на основі Streamlit для зручного доступу до функціоналу системи.

5. Провести тестування та валідацію розробленої системи на історичних даних для оцінки її точності та стабільності.

6. Проаналізувати результати прогнозування та запропонувати рекомендації щодо використання системи у практичній діяльності.

Наукова новизна одержаних результатів

Наукова новизна одержаних результатів полягає у впровадженні нових моделей прогнозування для аналізу динаміки цін на харчові продукти. Застосовуються сучасні підходи обробки часового ряду для визначення трендів і сезонних коливань. Впроваджено методи аналізу даних, зокрема, на основі використання моделей машинного навчання для більш точної оцінки перспективних змін цін. Проведено порівняльний аналіз традиційних і сучасних підходів до прогнозування, що сприяє підвищенню їх ефективності. Результати досліджень мають практичне значення для формулювання рекомендацій щодо планування цінових стратегій у сфері харчової промисловості.

Практичне значення одержаних результатів

Практичне значення одержаних результатів полягає у можливості використання отриманих моделей для оперативного прогнозування цін на харчові продукти. Це дозволяє підприємствам більш ефективно планувати запаси та формувати цінову політику. Результати досліджень сприяють вдосконаленню систем моніторингу змін ринку та своєчасному реагуванню на коливання попиту та пропозиції.

РОЗДІЛ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1. Методи прогнозування ринку харчових продуктів

Вибір методу прогнозування ринку продовольчих товарів залежить від мети прогнозу, рівня деталізації та наявності вхідної інформації. Якщо прогноз можливого продажу продовольчих товарів здійснюється для визначення перспектив розвитку роздрібної торгової мережі, можуть використовуватися оціночні методи прогнозування. Якщо ж він застосовується для обгрунтування закупівлі конкретних продовольчих товарів на найближчий місяць, слід використовувати точніші методи. Якщо є дані про попит за останні кілька років, а також матеріали, що характеризують зміну факторів, що формують попит, можна використовувати надійні методи прогнозування.

Прогнози продовольчого ринку можуть класифікуватися за кількома ознаками. За часом попередження вони поділяються на короткострокові, середньострокові та довгострокові[1].

За товарною ознакою розрізняють прогнози ринку: конкретної харчової продукції, виду продукції, товарної групи. За регіональною ознакою здійснюють прогнози ринку для конкретних споживачів, адміністративних районів, великих регіонів, країни.

По суті методів, що застосовуються, виділяють групи прогнозів, основою яких є: екстраполяція ряду динаміки, коефіцієнти попиту, цін і товарної пропозиції, багатофакторні регресійні моделі.

Екстраполяційні моделі дають добрі результати лише на найближчу (один-два роки) перспективу. Продовження тенденцій зниження (зростання) обсягів споживання на душу населення окремих видів продовольчих товарів на більш віддалену перспективу, що склалися у звітному періоді, може призвести до негативних результатів [2].

Якщо в якості фактора, що визначає попит, прийняти доходи на душу населення, то неважко обчислити коефіцієнти еластичності з будь-якого продукту.

Враховуючи, що попит формується переважно під впливом зміни доходів та цін, можна прогнозувати коефіцієнт еластичності від зміни цих факторів.

Прогнозування попиту можна здійснювати з врахуванням однофакторних моделей. Їх доцільно використовувати за необхідності врахування впливу найважливішого фактора на попит. Наприклад, за стабільного рівня цін визначають залежність попиту на харчову продукцію від зміни доходів населення.

У перехідний період при посиленні диференціації доходів населення доцільно використовувати для прогнозування попиту багатофакторну регресійну модель. Очікуваний попит на харчову продукцію визначається підстановкою в рівняння регресії прогнозних значень факторів [3].

Вивчення попиту методом багатофакторного прогнозування базується на даних про реалізований попит (продаж окремих видів продовольчих товарів) і формують його фактори (доходи, ціни, склад населення) за кожен аналізований рік в цілому по країні або в територіальному аспекті. Для оцінки перспектив насичення продовольчих товарів за рахунок вітчизняних товаровиробників необхідно визначити балансові моделі. Пропозиція харчової продукції на продовольчому ринку формується з продукції, виробленої в країні (регіоні), за її межами (імпорт продукції) та товарних запасів.

У сучасних умовах господарювання обсяг виробітку харчової продукції формується двома способами, а саме шляхом: укладання харчовими підприємствами прямих договорів на постачання продукції; доведення до підприємств замовлення на постачання продукції для державних потреб. Крім того, частину харчової продукції, виробленої понад замовлення держави та прямих договорів, підприємства можуть реалізувати на вільному ринку [4]. У процесі прогнозування розвитку продовольчого ринку необхідно враховувати зміни у структурі реалізації продукції.

На більшості товарних ринків криві пропозиції та попиту зазнають певних змін. Дохід споживачів змінюється у міру економічного зростання. Аналогічним чином змінюються обсяг заробітної плати, ціни на сільськогосподарську сировину та продовольчі товари, що значною мірою впливає на пропозицію продовольчих товарів.

На обсяг пропозиції також великий вплив має науково-технічний прогрес. Він дозволяє знижувати витрати виробництва та змінювати обсяг пропозиції продовольчих товарів. Іноді підвищення технічного рівня стає єдиним шляхом, з якого харчове підприємство може збільшити свій прибуток. Попит населення та його доходи визначають виробничу програму, кількісні характеристики та якість харчової продукції підприємств [5].

Нерідко навіть за умов переходу до ринку частина виробленої харчової продукції поставляється на експорт. Тому прогнози імпорту та експорту дозволяють внести важливі корективи до результатів прогнозування попиту (споживання) та виробництва на внутрішньому продовольчому ринку.

Для динамічного продовольчого ринку характерне явище циклічності, тобто повторюваності тенденцій та інтенсивності розвитку. Сезонні коливання ринку зумовлені сезонністю сільськогосподарського виробництва, сезонно-кліматичними змінами потреб. Сезонні зміни попиту та пропозиції охоплюють далеко не всі продовольчі товари, але для багатьох із них характерний значний розмах сезонних коливань. Причому сезонність різних товарів має особливості [6]. Це створює ряд економічних та організаційно-технологічних проблем: утворення сезонних товарних запасів, нерівномірності навантаження на працівників торгівлі та торгове обладнання, простої транспортних засобів тощо. У зв’язку з цим необхідно прогнозування сезонності продажу продовольчих товарів виконувати з використанням методів математичної статистки.

Для моделювання повторюваності сезонних хвиль доцільно використовувати формулу гармоніки Фур’є.Визначення сезонної компоненти є окремим випадком гармонійного аналізу, при якому період коливання дорівнює 12 місяців. Коефіцієнти Фур’є визначаються із застосуванням формул Бесселя [7].

Використання моделі сезонної хвилі дозволяє визначити обсяги продажу продовольчих товарів на найближчу перспективу за місяцями. Вибір економічно обґрунтованого методу прогнозування ринку продовольчих товарів дозволить підвищити ефективність розрахунків у сфері планування розвитку цього сектора економіки.

1.2. Аналіз та прогнозування рівня продовольчого забезпечення населення

Основою моделювання є механізм, що забезпечує взаємозв'язок усіх сфер продовольчого комплексу як цілісної системи з взаємопов'язаними галузями-виробниками продовольства. Центральною ланкою продовольчої системи є аграрна галузь, ефективність якої значною мірою залежить від територіального фактора.

Моделювання територіального розміщення підгалузей сільського господарства здійснюється за допомогою оптимізаційної задачі з критерієм оптимальності максимум виробленої продукції з урахуванням асортиментної структури при мінімумі матеріально-грошових і трудових витрат, а також із заданими обмеженнями за середньодушовим споживанням кожного продукту [8].

Залежно від різних гіпотез можна отримати кілька варіантів розміщення регіональної структури та спеціалізації виробництва харчових продуктів. Рекомендації щодо територіального розміщення аграрних підгалузей пов'язуються з напрямами інтенсифікації, вдосконалення технологій виробництва та залучення трудових ресурсів до аграрної галузі за рахунок стимулюючих заходів. Моделювання розміщення виробництва дозволить задовольнити потреби особистого та виробничого споживання, створити відповідні резервні фонди та забезпечити експортний потенціал регіону.

Програмний підхід дає змогу розробити комплекс узгоджених і взаємопов'язаних за строками та виконавцями заходів щодо оптимізації продовольчої системи. Ресурси на розвиток системи та її очікувана ефективність розраховуються за спеціальними програмами, у яких зміна кожного показника прогнозується як результат конкретних заходів, тоді як при розробці прогнозів показники розвитку виробництва визначаються методом екстраполяції [9].

Попередньо виявляються тенденції розвитку динамічних рядів прогнозованих показників за попередній період достатньої тривалості. При цьому доцільно розраховувати альтернативні варіанти розвитку продовольчого ринку за різних методів державного регулювання та умов зовнішнього середовища. Необхідність подолання існуючої технічної та технологічної відсталості підгалузей рослинництва

і тваринництва передбачає використання перспективних можливостей впровадження імпортних розробок із застосуванням світових досягнень [10].

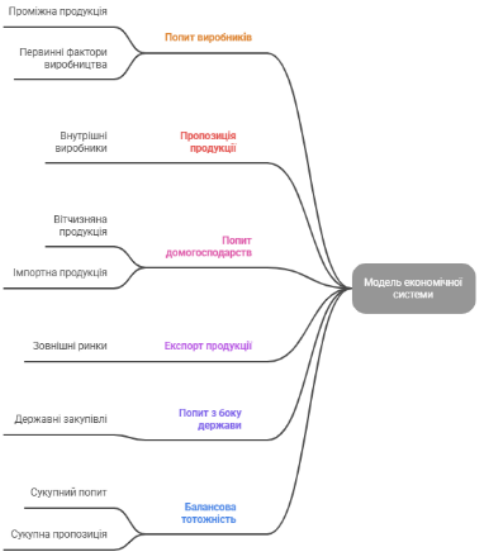


Рисунок 1.1 –Модель економічної системи

Моделювання включає кілька етапів[11]:

1. Моделювання попиту. Одним із найважливіших законів конкурентного ринку є закон попиту, який визначає обернену залежність між ціною та об’ємом купівлі продукту: чим вища ціна, тим меншу кількість товару даного виду споживачі готові придбати.

Попит на продовольчому ринку формується під впливом комплексу факторів:

- чисельності, статевої та вікової структури населення;

- споживчих уподобань і традицій, що визначаються історично сформованими харчовими звичаями;

- рівня платоспроможного попиту, який залежить від доходів населення та цінової політики;

- наявності супутніх товарів, товарів-субститутів та рівня цін на них.

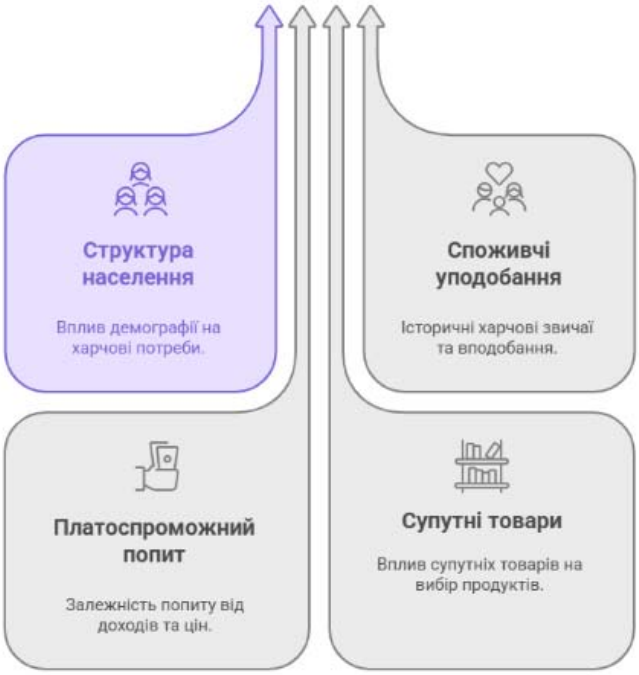


Рисунок 1.2 –Фактори впливу на попит

У економічній літературі пропонується значна кількість систем функцій попиту. Для обґрунтування виду рівнянь попиту висувається припущення про форму функції корисності споживача, після чого визначається функціональна форма рівняння попиту, яка підлягає економетричній оцінці.

Споживач має можливість замінити набір із двох продуктів третім, який адекватно компенсує їх сукупну корисність. При цьому перехід покупця від одних виробників до інших зменшує частку перших на продовольчому ринку [12]:

Залежність попиту від доходу та ціни:

- Попит на будь-який продукт харчування є прямо пропорційним до середньодушового доходу споживача.

- Попит обернено пропорційний до ціни продукту.

Комплементарні товари:

- Досконалі комплементарні товари (наприклад, кава та цукор) завжди споживаються разом у фіксованій пропорції, що відображає їхню взаємодоповнюваність.

- Ефект заміщення: можливість заміни продуктів (наприклад, гречка ↔ рис) залежить від їхньої граничної норми заміщення у функції корисності.

- Ринкова динаміка: зниження частки певних виробників через перехід споживачів до інших брендів ілюструє конкуренцію та еластичність попиту.

Якщо розглядати функцію Кобба-Дугласа для двох продуктів x_1 і x_2 з можливістю заміни на x_3 [13]:

$$U(x_1, x_2) = x_1^{\alpha} \cdot x_2^{\beta} \quad \text{або} \quad U(x_3) = k \cdot x_3^{\gamma}, \tag{1.1}$$

де k - коефіцієнт, що враховує еквівалентність корисності.

Приклад: якщо ціна яблук зростає, споживач може частково замінити їх грушами (субститутами), що призведе до зменшення частки виробників яблук на ринку.

Функція прогнозування попиту на агреговані групи продуктів. Ця модель широко використовується для аналізу поведінки споживачів при зміні цін та доходів. Умови застосування моделі: модель передбачає обов'язкову наявність продуктів-субститутів, здатних задовольняти однакові потреби (мандарини ↔ апельсини, курятина ↔ яловичина).Критерій взаємозамінності: перехресна еластичність попиту між товарами має бути додатною [14].

ВИСНОВКИ ДО РОЗДІЛУ

У розділі 1 проаналізовано сучасний стан проблемної області прогнозування попиту на харчові продукти.Виявлено, що ринок харчової продукції зазнає постійних коливань через сезонні та економічні фактори.Встановлено, що існуючі методи прогнозування часто недостатньо точні або не враховують комплекс факторів, що впливають на попит.Визначено, що недостатня автоматизація та низька швидкість аналізу призводять до втрат ресурсів і можливості швидко реагувати на зміни ринку.Виявлено прогалини у залученні сучасних технологій машинного навчання для покращення точності прогнозів.

Встановлено, що більшість існуючих систем прогнозування мають обмежену адаптивність до змін у ринковій ситуації.Виявлено, що відсутність сучасних автоматизованих систем ускладнює процес планування запасів та виробництва.Обгрунтовано необхідність розробки нових систем, які базуються на сучасних методах і моделях машинного навчання для подолання існуючих недоліків.

Визначено, що покращення якості прогнозів сприяло б підвищенню конкурентоспроможності підприємств харчової галузі.Важливо враховувати сезонні і трендові коливання для досягнення більшої точності у прогнозуванні.

Розгляд сучасних тенденцій і технологій показав, що використання математичних моделей та машинного навчання має великий потенціал для покращення прогнозних характеристик.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Опис набору даних

Набір даних «Індекс споживчих цін на продукти харчування» є офіційним ресурсом, оприлюдненим Державною службою статистики України. Цей набір містить інформацію про зміни цін на основні продукти харчування в Україні за період з 2015 по 2024 рік [12].

Дані представлені з наступними основними стовпцями:

- рік, до якого відноситься індекс;
- місяць, до якого відноситься індекс;
- продукт: назва продукту харчування;
- індекс цін: числове значення індексу цін для відповідного продукту в зазначений період.

Цей набір даних [13] дозволяє аналізувати динаміку цін на різні продукти харчування в Україні протягом зазначеного періоду.Індекс цін для відповідного продукту - це числовий показник, який відображає зміну цін на цей продукт у певний період часу відносно базового періоду (попередній рік або місяць, взятий за основу).Якщо індекс цін дорівнює 100%, це означає, що ціна продукту не змінилась порівняно з базовим періодом.Якщо індекс цін більший за 100%, це означає, що ціна зросла. Наприклад, індекс 110% означає, що ціна зросла на 10%.Якщо індекс цін менший за 100%, це означає, що ціна знизилась. Наприклад, індекс 90% - ціна впала на 10%.

Таблиця 2.1–Приклад рядка з набору даних

Рік	Місяць	Продукт	Індекс цін
2024	Січень	Картопля	115,3

Це означає, що у січні 2024 року ціна на картоплю була на 15,3% вищою за ціну в базовому періоді.Індекс цін допомагає відстежувати тенденції росту чи падіння цін. Він дозволяє економістам, аналітикам та бізнесу робити висновки щодо інфляції, попиту, сезонності, є основою для прогнозування цін у майбутньому.Ці дані можуть бути корисними для аналізу тенденцій зміни цін на продукти

харчування, прогнозування майбутніх цінових коливань, для розробки моделей прогнозування за допомогою методів машинного навчання, таких як LSTM.

2.2. Метод машинного навчання LSTM

LSTM (Long Short-Term Memory)- різновид рекурентних нейронних мереж (RNN), спеціально розроблений для роботи з послідовними даними.LSTM може зберігати інформацію протягом довгого часу завдяки спеціальним структурам, які називаються комірками пам’яті.Вони керуються трьома ключовими елементами - input gate, forget gate та output gate, які контролюють потік інформації.Завдяки їм мережа вибірково запам’ятовує або забуває інформацію, що допомагає ефективно працювати з довгими послідовностями.LSTMшироко використовують у задачах, де важлива послідовність або часові залежності. Це прогнозування часових рядівцін на продукти[14].

Принцип роботи LSTM. Вхідний гейт input gatевирішує, яку нову інформацію додати в комірку пам’яті.Гейт забування Forget Gate вирішує, яку інформацію видалити з комірки пам’яті.Гейт виходу Output Gate)вирішує, яку частину комірки пам’яті використовувати для вихідного сигналу мережі.Це нейронні шари з функцією активації sigmoid, значення якої від 0 до 1, які регулюють, скільки інформації пропускати.

Нижче знаходиться програмний код для створення моделі LSTM.

```
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
from sklearn.preprocessing import MinMaxScaler

time_steps = 100
x = np.linspace(0, 50, time_steps)
data = np.sin(x)

scaler = MinMaxScaler(feature_range=(0, 1))
data = scaler.fit_transform(data.reshape(-1,1))
def create_dataset(dataset, look_back=10):
    X, Y = [], []
```

```
    for i in range(len(dataset)-look_back):
        X.append(dataset[i:i+look_back, 0])
        Y.append(dataset[i+look_back, 0])
    return np.array(X), np.array(Y)

look_back = 10
X, Y = create_dataset(data, look_back)

X = X.reshape(X.shape[0], X.shape[1], 1)

model = Sequential()
model.add(LSTM(50, input_shape=(look_back, 1)))
model.add(Dense(1))

model.compile(optimizer='adam', loss='mean_squared_error')
model.fit(X, Y, epochs=100, batch_size=1, verbose=0)

predicted = model.predict(X)
predicted = scaler.inverse_transform(predicted)
actual = scaler.inverse_transform(Y.reshape(-1,1))

plt.plot(actual, label='Фактичні дані')
plt.plot(predicted, label='Прогноз LSTM')
plt.legend()
plt.show()
```

2.3. Алгоритмічне забезпечення інформаційної системи

Алгоритмічне забезпечення є ключовим компонентом інформаційної системи прогнозування цін на продукти харчування. Воно включає в себе сукупність алгоритмів, що реалізують математичну модель прогнозу та забезпечують її інтеграцію в програмне середовище. У цьому проекті реалізовано модуль побудови прогнозу на основі глибокої рекурентної нейронної мережі типу LSTM(Long Short-Term Memory)**.**

Розглянемо основні етапи алгоритмічного забезпечення [15].

Завантаження та обробка даних:

- завантаження історичних цін з публічного ресурсу у форматі Excel;
- фільтрація за обраним продуктом;

- приведення даних до формату, придатного для машинного навчання;
- масштабування цін у діапазон [0, 1] за допомогою MinMaxScaler.

Формування навчальних послідовностей:

- побудова послідовностей фіксованої довжини (30 днів) для навчання моделі;
- формування вхідних X та вихідних у даних на основі ковзного вікна.

Побудова та тренування LSTM-моделі:

- архітектура включає два LSTM-шари по 50 нейронів та один вихідний шар;
- модель компілюється з оптимізатором Adam та функцією втрат mean_squared_error;
- навчання відбувається протягом 20 епох на розбитті тренувальних і тестових даних у співвідношенні 80/20.

Прогнозування та обернене масштабування[16]:

- отримані прогнозовані значення масштабуються назад до реальних цін;
- розраховуються метрики якості: MAE, MSE, R^2.

Інтеграція з інтерфейсом користувача (Streamlit):

- використовується бібліотека Streamlit для реалізації веб-інтерфейсу інформаційної системи;
- інтерфейс дозволяє обрати продукт, задати період прогнозу, відобразити графіки, таблиці та ключові метрики;
- передбачено візуалізацію результатів у вигляді графіків і таблиць.

Рисунок 2.1– Процес прогнозування часових рядів

ВИСНОВКИ ДО РОЗДІЛУ

У розділі 2 проаналізовано сучасний стан інформаційного забезпечення систем прогнозування.Визначені основні вимоги до збору та обробки даних для системи прогнозування попиту на продукти харчування.Вибрано інструменти та технології для автоматизації збору, збереження та обробки даних.Вивчено особливості роботи з відкритими базами даних та їхню інтеграцію у систему.

В результаті виконано обґрунтування вибору технологій і архітектури системи для надійної роботи у реальних умовах. Інформація у цьому розділі заклала основу для подальшого розвитку математичного та програмного забезпечення системи.

РОЗДІЛ 3. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Математичне забезпечення для інформаційної системи прогнозування попиту на продукти харчування

Постановка задачі прогнозування. Завдання полягає в побудові моделі, яка здатна прогнозувати майбутню вартість певного продукту харчування (наприклад, картоплі) на основі історичних даних про ціни. Нехай y_t - ціна продукту у момент часу t. Необхідно передбачити значення $y_{t+1}, y_{t+2}, \ldots, y_{t+n}$, використовуючи попередні значення $y_t, y_{t-1}, \ldots, y_{t-k}$.Це є типовою задачею прогнозування часових рядів.

У рамках даного дослідження розглядається задача прогнозування цін на продукти харчування, зокрема на картоплю, на основі аналізу часових рядів історичних цін. Ціль полягає в побудові моделі, яка здатна на основі попередніх значень цін передбачити їхні майбутні значення з максимальною точністю.

Нехай маємо часовий ряд реальних значень цін на певний продукт:

$$Y = \{y_1, y_2, \ldots, y_T\}, \quad y_t \in \mathbb{R} \tag{3.1}$$

деy_t–ціна продукту у момент часу t,T– загальна кількість спостережень у часовому ряду.Задача прогнозування полягає у побудові функції:

$$\hat{y}_{t+1} = f(y_t, y_{t-1}, \ldots, y_{t-k+1}) \tag{3.2}$$

деk– кількість попередніх часових точок, що використовуються для прогнозу (довжина вхідного вікна),$\hat{y}_{t+1}$–прогнозоване значення ціни у момент часу t+1, f– функція, яку реалізує модель машинного навчання LSTM-мережа.

Модель має навчитися визначати приховані закономірності у зміні ціни за певний проміжок часу, включаючи сезонні коливання, довготривалі тенденції (тренди), локальні флуктуації.

Ця задача відноситься до одновимірного прогнозування часових рядів, де прогнозується лише одна змінна ціна продукту. На відміну від багатовимірного прогнозування, у цьому випадку не враховуються зовнішні фактори (погодні умови, логістичні витрати, інфляція), але це можна враховувати у майбутньому.

При розв’язанні задачі зроблено наступні припущення:

- стаціонарність у короткостроковому періоді– ціна змінюється за певними шаблонами, які можуть бути виявлені на підставі попередніх значень;
- регулярність спостережень– значення цін фіксуються через однакові проміжки часу, щотижня або щомісяця;
- наявність історичних даних достатньої довжини для навчання моделі.

Метою прогнозуванняє отримання прогнозованих значень цін у майбутні часові моменти $t+1,t+2,\ldots,t+n$з мінімальною похибкою. Це дозволитьоцінювати цінові коливання,планувати закупівлі/продажі продуктів.

Вартість сільськогосподарських продуктів, таких як картопля, залежить від багатьох факторів (урожайність, сезон, економічна ситуація). Тому прогноз цін є важливим інструментом як для виробників, так і для споживачів. Автоматизована модель прогнозування дозволяє реагувати на можливі коливання на ринку та зменшити економічні ризики.

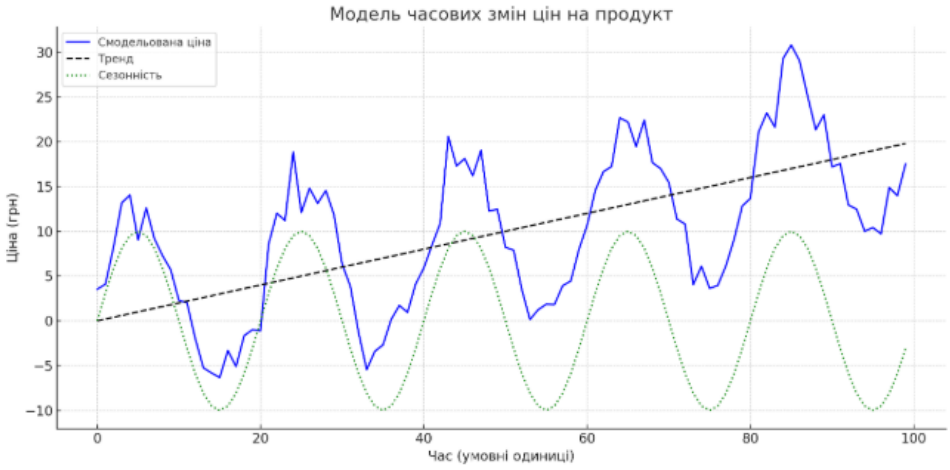


Рисунок 3.1–Графік математичної моделі зміни ціни з часом.

Цей графік відображає:

- тренд (чорна пунктирна лінія) – довгострокову тенденцію зростання ціни;
- сезонність (зелена пунктирна лінія) – періодичні коливання через пори року;
- випадковий шум– флуктуації, які додають реалізму до моделі.

Цей графік ілюструє принцип формування прогнозу.Перед подачею даних до нейромережі здійснюється нормалізація значень цін:

$$x_i' = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \tag{3.3}$$

деx_i–оригінальне значення ціни,x' – масштабоване значення,x_{min}, x_{max}–мінімальні та максимальні значення в наборі даних.Цей підхід зменшує розкид даних і прискорює навчання моделі.

Перед використанням даних для навчання моделі прогнозування необхідно провести попередню обробку масштабування числових значень. Ціни на продукти харчування можуть коливатись у широкому діапазоні, наприклад, від кількох десятків гривень (овочі) до сотень гривень (м'ясо). Якщо передавати такі значення без обробки безпосередньо до моделі, велика варіативність може призвести до некоректного навчання.

Використовується мін-макс нормалізація – один з найбільш поширених методів масштабування, що перетворює всі значення у заданий діапазонвід 0 до 1. Формула масштабування:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3.4}$$

де x – початкове значення ціни, x_{min}– мінімальне значення в наборі даних, x_{max}– максимальне значення в наборі даних, x_{scaled} – нормалізоване значення.

Для реалізації масштабування було використано клас MinMaxScaler з бібліотеки sklearn.preprocessing:

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range=(0, 1))
prices_scaled = scaler.fit_transform(product_data['Price'].values.reshape(-1, 1))
```

Всі значення будуть приведені до діапазону [0, 1], після цього перетворюють одномірний масив у двовимірний, що потрібно для роботи з бібліотекою Scikit-Learn.

Після того як модель виконає прогнозування, значення також необхідно зворотно перетворити до оригінального масштабу, щоби результати мали зміст у контексті ціну гривнях:

```
predictions = scaler.inverse_transform(predictions_scaled)
```

Це забезпечує інтерпретованість результатів для користувача.

Формування навчальних послідовностей. Для створення вибірки з часових рядів використовується ковзне вікно довжини L. Вхід X_i–це L попередніх значень, вихід y_i–наступне значення:

$$X_i = [x_i, x_{i+1}, \ldots, x_{i+L-1}], \quad y_i = x_{i+L} \tag{3.5}$$

Це дозволяє перетворити одномірний часовий ряд у множину пар вхід-вихід для навчання.Оскільки завдання прогнозування цін є завданням аналізу часових рядів, модель має враховувати залежність між значеннями, що розташовані в часі послідовно. Для цього дані необхідно перетворити у вигляді послідовностей фіксованої довжини вікон часу, які ковзають по ряду. Це дасть змогу моделі навчитися закономірностям змін цін з урахуванням контексту попередніх днів.

Розглянемо побудова вікон часу, послідовностей. Навчальні дані формуються наступним чином: із масштабованого ряду цін створюються підпослідовності довжиною SEQ_LENGTH. Кожна така підпослідовність використовується як вхід до моделі, а значення, що йде після неї, як цільове значення.

Нехай маємо нормалізований часовий ряд:

$$S = [s_1, s_2, s_3, \ldots, s_T] \tag{3.6}$$

Довжина вікна часуL. Тоді кожен навчальний приклад можна подати у вигляді:

$$X^{(i)} = [s_i, s_{i+1}, \ldots, s_{i+L-1}], \quad y^{(i)} = s_{i+L} \tag{3.7}$$

де$X^{(i)}$–послідовність нормалізованих цін, вхід до моделі,$y^{(i)}$–наступне значення, яке потрібно передбачити.

Цей підхід дозволяє LSTM-моделі ефективно аналізувати динаміку змін у часі та формувати узагальнені закономірності для прогнозування.Для прогнозування використовується рекурентна нейронна мережа типу LSTM (LongShort-TermMemory), яка ефективно працює з послідовними даними завдяки механізму довгої пам’яті.Вхідний гейт:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{3.8}$$

Забуття:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3.9}$$

Новий кандидат:

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{3.10}$$

Оновлення стану пам’яті:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{3.11}$$

Вихідний гейт:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{3.12}$$

Вихід:

$$h_t = o_t * \tanh(C_t) \tag{3.13}$$

деx_t–вхід у момент часу t,h_t–вихід прихованого шару,C_t–внутрішній стан пам’яті,σ–сигмоїдна функція активації,*–поелементне множення.

Функція втрат. Для оцінки точності моделі використовувалась середньоквадратична помилка (MSE):

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.14}$$

деy_i–реальні значення, $\hat{y}_i$–прогнозовані значення.

У процесі навчання нейронної мережі ключовим елементом є функція втрат.Це математичний інструмент, який вимірює відмінність між прогнозованими моделлю значеннями та фактичними значеннями з навчальних даних.У задачах регресії, де необхідно передбачити неперервне числове значення, ціну картоплі у певний день, функція втрат дозволяє навчальному алгоритму адаптувати ваги нейронів так, щоб зменшити похибку передбачення.

Метрики якості моделі. Для оцінки ефективності моделі використовуються:

MAE (Mean Absolute Error):

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.15}$$

R^2 (коефіцієнт детермінації):

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \tag{3.16}$$

ВИСНОВКИ ДО РОЗДІЛУ

У розділі 3 сформульовано математичну модель системи прогнозування попиту на харчові продукти.Розглянуто основні підходи до обробки часового ряду, включаючи визначення трендових та сезонних компонент.Вибрано методи статистичного аналізудля врахування динаміки даних.Детально описані алгоритми для розпізнавання та врахування сезонних коливань у даних.Проведено оцінку вибраних моделей на основі історичних даних та визначено їхню точність.

Враховано фактори трендового зростання та таких сезонних змін для підвищення якості прогнозів.Використано методи апроксимації для адаптації моделі до змін у ринковій ситуації.Розробка математичної моделі дозволила автоматизувати процес прогнозування попиту в інформаційній системі.Обґрунтовано вибір конкретних математичних методів на основі аналізу характеристик даних.Математичне забезпечення створює підґрунтя для подальшої реалізації прогнозної моделі у програмному середовищі.Успішна реалізація математичного залучення дозволила підвищити надійність системи прогнозування та її практичне застосування.

РОЗДІЛ 4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

4.1. Розроблення інформаційної системи для прогнозування попиту на продукти харчування

Розглянемо процес реалізації проекту прогнозування цін на харчові продукти в Україні за допомогою LSTM, використовуючи реальні дані.Для цього проекту використано набір даних «Індекс споживчих цін на продукти харчування» від Державної служби статистики України. Цей набір містить інформацію про зміни цін на основні продукти харчування в Україні за період з 2015 по 2024 рік:

https://data.gov.ua/en/dataset/b5df316f-dfbe-4cb0-b596-807a301dd0bc

Встановлення необхідних бібліотек:

```
pip install pandas numpy matplotlib scikit-learn tensorflow openpyxl
```

Це команда для встановлення Python-бібліотек через пакетний менеджер pip.

- pip install - команда для інсталяції пакетів;
- pandas - бібліотека для роботи з таблицями та аналізу даних;
- numpy - бібліотека для числових обчислень, робота з масивами;
- matplotlib - бібліотека для створення графіків і візуалізацій;
- scikit-learn бібліотека для машинного навчання;
- tensorflow - бібліотека для глибокого навчання (нейронні мережі);
- openpyxl - бібліотека для роботи з Excel-файлами формату .xlsx.

Цю команду виконують в командному рядку, щоб встановити всі потрібні залежності для проекту.

Завантаження та попередня обробка даних.

```
import pandas as pd
url = 'https://data.gov.ua/dataset/b5df316f-dfbe-4cb0-b596-807a301dd0bc/download'
data = pd.read_excel(url)
print(data.head())
```

Імпортують бібліотеку Pandas,вона потрібна для роботи з наборами даних.

url = 'https://data.gov.ua/dataset/b5df316f-dfbe-4cb0-b596-807a301dd0bc/download' - зберігають в змінній url посилання на Excel-файл, який містить дані.

За допомогою Pandas завантажують Excel-файл безпосередньо з інтернету і зберігають його в змінній data як датафрейм.Виводять перші 5 рядків таблиці для

перегляду структури і вмісту даних.Цей код дозволяє імпортувати та подивитися, що міститься у файлі з даними.

Підготовка даних для моделі LSTM.

```
import numpy as np
from sklearn.preprocessing import MinMaxScaler
```

Готують інструменти для числової обробки та масштабування даних.Імпортуютьбібліотеку NumPy,яка використовується для роботи з багатовимірними масивами, математичних операцій та обробки числових даних.Імпортують клас MinMaxScaler з бібліотеки Scikit-Learn.MinMaxScaler використовується для масштабування числових даних у діапазон від 0 до 1. Це допомагає підготувати дані для подальшої обробкидля моделей машинного навчання.

```
data = data[['Date', 'Product', 'Price']]
```

Вибирають з датафрейму data три стовпці: Date, Product та Price.Потім цей відфільтрований датафрейм присвоюється назад у змінну data, тобто тепер data містить тільки ці три стовпці.Це робиться, щоб працювати тільки з потрібними для аналізу даними і виключити зайві стовпці.

```
product_data = data[data['Product'] == 'Картопля'].sort_values('Date')
```

Створюють масив, де True для рядків, у яких у стовпці Product значення дорівнює картопля.Фільтрують таблицю data, залишаючи лише ті рядки, де продукт єкартопля.Сортуютьвідфільтровані рядки за зростанням дати у стовпці Date.Результат зберігається у змінній product_data.product_data - це таблиця з цінами на картоплю, яка відсортована за датою.

```
scaler = MinMaxScaler(feature_range=(0, 1))
prices_scaled = scaler.fit_transform(product_data['Price'].values.reshape(-1, 1))
```

Створюють об’єкт MinMaxScaler, який масштабуватиме дані у діапазон від 0 до 1.Береться стовпець Price з product_data, отримується його значення у вигляді масиву NumPy і перетворюється у форму стовпця двовимірного масиву з однимстовпцем. Це потрібно, бо fit_transform очікує двовимірний масив.Метод fit_transform одночасно навчає масштабувач на даних (визначає мінімум і максимум) і трансформує значення у новий масштаб 0 до 1. Результат є нормалізовані

ціни.prices_scaled -масив з масштабованими цінами від 0 до 1, готовий для подальшої роботидля навчання моделі.

```
def create_sequences(data, seq_length):
    X, y = [], []
    for i in range(len(data) - seq_length):
        X.append(data[i:i+seq_length])
        y.append(data[i+seq_length])
    return np.array(X), np.array(y)
```

Ця функція потрібна для підготовки даних у формат, зручний для навчання моделей часових рядів. Вона приймає два аргументи:

- data- послідовність (масив цін);
- seq_length- довжина послідовності, яку потрібно використовувати для прогнозу наступного значення.

Всередині функції створюються два списки:

- X - для вхідних послідовностей довжиною seq_length;
- y - для відповідних цільових значень (наступних після послідовності).

Цикл ітерує по позиціях у масиві, щоб взяти підмасиви довжини seq_length для навчання.Повертаються два масиви: матрицю вхідних послідовностей і вектор цільових значень.

```
SEQ_LENGTH = 30
X, y = create_sequences(prices_scaled, SEQ_LENGTH)
```

Проводять підготовку даних для навчання моделі прогнозування часових рядів.Встановлюютьдовжину послідовності для створення вхідних даних рівною 30. Тобто модель буде дивитися на 30 попередніх значень, щоб передбачити наступне.Викликаютьфункцію create_sequences, передаючи їй масштабовані ціни prices_scaled та довжину послідовності SEQ_LENGTH.Функція повертає:

- X - масив із послідовностей по 30 значень вхідні дані;
- y - масив з відповідними наступними значеннями після цих послідовностей цільові дані.

```
split = int(0.8 * len(X))
X_train, X_test = X[:split], X[split:]
y_train, y_test = y[:split], y[split:]
```

Це крок для оцінки якості моделі на нових, раніше не бачених даних.Обчислюють індекс, який відповідає 80% довжини масиву X. Це точка розбиття даних на тренувальні та тестові.Розбивають вхідні дані X на дві частини:X_train -перші 80% для тренування моделі,X_test - останні 20% для перевірки та тестування моделі.Аналогічно розбивають цільові значення y на тренувальні y_train та тестові y_test.

Побудова та тренування моделі LSTM.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
```

Імпортують з бібліотеки TensorFlow компоненти для побудови нейронної мережі:

- Sequential - модель, у якій шари накладаються послідовно один за одним;
- LSTM - шар довгої короткочасної пам’яті (Long Short-Term Memory), який добре підходить для роботи з часовими рядами та послідовностями;
- Dense - звичайний повнозв’язний шар нейронної мережі.

Ці імпорти потрібні для створення моделі, яка буде прогнозувати ціни на основі послідовностей цін.

```
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape=(SEQ_LENGTH, 1)))
model.add(LSTM(50))
model.add(Dense(1))
```

Ця модель LSTM з двома шарами та одним вихідним шаром призначена для прогнозування числової величини на основі послідовних вхідних даних.Створюють порожню послідовну модель, куди поступово додаються шари.Додають перший LSTM шар з 50 нейронами.Очікується вхід у вигляді послідовності довжиною SEQ_LENGTH (30), кожен крок має 1 ознаку (значення ціни).return_sequences=True - цей шар повертає послідовність (всі проміжні виходи), щоб наступний LSTM шар теж міг її обробити.

model.add(LSTM(50)) - другий LSTM шар з 50 нейронами. Тут return_sequences за замовчуванням False, тому цей шар повертає тільки останнє значення послідовності, підходить для передачі у вихідний шар.model.add(Dense(1)) - вихідний шар, повнозв’язний, з одним нейроном - це прогнозоване значення ціни.

```
model.compile(optimizer='adam', loss='mean_squared_error')
```

Виконуютькомпіляцію моделі перед тренуванням.Використовується оптимізатор Adam, це один з найпопулярніших, якийдобре підходить для часових рядів.Вказують функцію втрат -середньоквадратичну помилку (MSE). Це стандартна функція для задач регресії, яка мінімізує різницю між фактичними та прогнозованими цінами.Після цього модель готова до навчання на тренувальних даних.

```
history = model.fit(X_train, y_train, epochs=20, batch_size=32,
validation_data=(X_test, y_test))
```

Цей рядок відповідає за навчання (тренування) моделі. model.fit(...) запускає процес навчання моделі.X_train, y_train -вхідні та цільові (ціни) дані для навчання.Модель проходить по всіх тренувальних даних 20 разів, 20 епох.Дані обробляються пакетами по 32.Під час кожної епохи модель також перевіряється на тестових даних.history – об’єкт, у якому зберігається інформація про тренування: значення функції втрат на кожній епосі для тренувальних і валідаційних даних. Його можна використати для побудови графіка навчання.

Оцінка моделі.

```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
```

Імпортують метрики оцінки якості моделі з бібліотеки sklearn.metrics.Ці метрики використовуються після прогнозу моделі для оцінки її точності. mean_squared_error (MSE) - середньоквадратична помилка, вонавимірює середній квадрат різниці між фактичними та прогнозованими значеннями,чим менше, тим краща модель.mean_absolute_error (MAE) - середня абсолютна помилка: середнє абсолютне відхилення передбачень від справжніх значень, такожчим менше, тим краще.r2_score - коефіцієнт детермінації: відображає, яку частину варіації цільової змінної пояснює модель (значення ближче до 1 - краще, 0 - погано, менше 0 - дуже погано).

```
predictions_scaled = model.predict(X_test)
predictions = scaler.inverse_transform(predictions_scaled)
actual = scaler.inverse_transform(y_test)
```

Виконуютьпрогнозування цін та зворотне масштабування, щоб отримати реальні значення цін у гривнях. Модель використовує тестові послідовності X_test,

щоб передбачити майбутні ціни.Результат - це масштабовані від 0 до 1 значення, оскільки навчання проводилось на масштабованих даних.Перетворюють масштабовані передбачення назад до реальних цін у гривнях.Без цього кроку значення залишаться в нормалізованому вигляді (0.3, 0.8 тощо).Те саме робиться і з тестовими правильними значеннями y_test, щоб їх можна було порівнювати з передбаченнями.Після цього будуть два масиви:

- predictions - передбачені ціни (у гривнях);
- actual - справжні ціни (у гривнях),які можна використовувати для побудови графіків та обчислення метрик точності.

```
mse = mean_squared_error(actual, predictions)
mae = mean_absolute_error(actual, predictions)
r2 = r2_score(actual, predictions)
```

Обчислюють точність прогнозної моделі за допомогою трьох основних метрик.

mean_squared_error обчислює середню квадратичну помилку MSE між фактичними і передбаченими цінами.Менше значення означає кращу точність.mean_absolute_error обчислює середню абсолютну похибку MAE, це середнє арифметичне модулів різниці між фактичними і прогнозованими значеннями. Вона легше інтерпретується, бо має ту ж одиницю виміру, що й ціни (гривні).r2_score обчислює коефіцієнт детермінації R^2, який показує, наскільки добре модель пояснює варіацію в даних.

```
print(f'MSE: {mse:.3f}')
print(f'MAE: {mae:.3f}')
print(f'R2 Score: {r2:.3f}')
```

Виводять на екран значення метрик оцінки моделі.

Візуалізація результатів.

```
import matplotlib.pyplot as plt
```

matplotlib.pyplot - це модуль із бібліотеки Matplotlib, який використовується для побудови графіків та візуалізації даних.

```
plt.figure(figsize=(12, 6))
plt.plot(actual, label='Реальні ціни')
plt.plot(predictions, label='Прогнозовані ціни')
plt.title('Прогноз цін на картоплю в Україні')
plt.xlabel('Час')
plt.ylabel('Ціна (грн)')
```

```
plt.legend()
plt.show()
```

Візуалізують результати прогнозу цін. Проводять побудову двох лінійних графіків:actual реальні значення цін, predictionsпередбачені моделью значення.label задає назву для кожної лінії.Підпис осі X час,Y ціна в гривнях.Виводять легенду графіка, яка показує, яка лінія що означає.Цей графік дає змогу порівняти, наскільки точно модель передбачає зміну цін у часі, а також чи слідує прогноз тенденціям у реальних даних.

Таким чином розроблено проект для прогнозування цін на картоплю в Україні за допомогою LSTM. Можна адаптувати цей підхід для інших продуктів, змінюючи фільтрацію даних за назвою продукту.

4.2. Розроблення графічного інтерфейсу інформаційної системи

Для створення інтерфейсу інформаційної системи було використано бібліотеку Streamlit, яка дозволяє реалізувати веб-додаток. Програмний код графічного інтерфейсу інформаційної системи представлений в додатку Б.

```
import streamlit as st
import numpy as np
import pandas as pd
products = ['Картопля', 'Молоко', 'Хліб', 'М’ясо', 'Овочі']
```

Підключають бібліотеку Streamlit, яка використовується для створення веб-інтерфейсів на Python.NumPувикористовується для роботи з масивами, матрицями та математичними операціями.Використовується у машинному навчанні, генерації випадкових даних, статистичних розрахунках.Pandasслужить для обробки табличних даних,використовується для завантаження, фільтрації, аналізу CSV-файлів.

Далі йде Python-список продуктів харчування, якібудуть використані для вибору користувачем або для відображення прогнозу цін. Він використовується у виді випадаючого списку:

```
def predict_price(product, period_value, period_type):
    base_prices = {
        'Картопля': 30,
        'Молоко': 60,
```

```
    'Хліб': 48,
    'М’ясо': 250,
    'Овочі': 75
}
```

Створюється функція для прогнозу ціниpredict_price на продукт.Вона приймає три аргументи:

- product - назва продукту;
- period_value - кількість часу;
- period_type - тип часу: дні, тижні або місяці.

Для базових цін створено словник з початковими цінами для кожного продукту.

Таблиця 4.1 –Ціни на продукти з набору даних

Продукт	Ціна (грн)
картопля	30
молоко	60
хліб	48
м’ясо	250
овочі	75

Ці значення використовуються як відправна точка для розрахунку майбутньої ціни.Ця функція повинна на основі базової ціни, кількості днів/тижнів/місяців і певного алгоритму прогнозувати нову ціну на продукти харчування.

```
base_price = base_prices.get(product, 10)
factor = 1 + 0.01 * period_value
if period_type == 'Дні':
    factor = 1 + 0.01 * period_value
elif period_type == 'Місяці':
    factor = 1 + 0.03 * period_value
elif period_type == 'Роки':
    factor = 1 + 0.2 * period_value
```

Ця частина функції predict_price розраховує прогнозований коефіцієнт зміни ціни залежно від часу. Отримується базова ціна з раніше створеного словника base_prices.Якщо назва продукту знайдена, отримують відповідне значення (наприклад, хліб → 48).Якщо продукту немає у словнику, використовується значення за замовчуванням 10.Спочатку коефіцієнт збільшується на 1 % за кожну

одиницю часу.Проте далі він перезаписується в залежності від типу періоду (дні, місяці, роки).

Далі йде основа моделі зростання цін. Залежно від вибраного типу періоду змінюється коефіцієнт:

Таблиця 4.2 –Коефіцієнти зростання цін у різних часових інтервалах

Тип періоду	Формула	Зростання
дні	1 + 0.01 * кількість_днів	1% за день
місяці	1 + 0.03 * кількість_місяців	3% за місяць
роки	1 + 0.2 * кількість_років	20% за рік

Цей коефіцієнт далі використовується, щоби обчислити прогнозовану ціну на продукти харчування.

```
predicted_price = base_price * factor
return round(predicted_price, 2)
```

Обчислютьпрогнозовану ціну шляхом множення базової ціни на розрахований коефіцієнт factor, який залежить від обраного періоду (днів, місяців або років).

Наприклад:

base_price = 60 # молоко

factor = 1.09 # якщо 3 місяці (1 + 0.03*3)

predicted_price = 60 * 1.09 = 65.4

Функція повертає результат, округлений до 2 знаків після коми, це стандартне відображення цін у гривнях.Функціяpredict_price отримує базову ціну продукту. Далі залежно від вибраного періоду (дні/місяці/роки) розраховує коефіцієнт зміни. Після цього збільшує базову ціну відповідно до коефіцієнта та повертає фінальну прогнозовану ціну.

```
product = st.selectbox('Оберіть продукт:', products)
```

Selectbox- це інтерактивний елемент у Streamlit, який створює випадаючий список. Його аргументи наступні. Оберіть продукт - це підпис/пояснення, яке бачить користувач поруч із списком.products - список варіантів для вибору:['Картопля', 'Молоко', 'Хліб', 'М’ясо', 'Овочі']. З допомогою цього віджета на сторінці відобразиться випадаючий список з назвами продуктів.Коли користувач обирає

якийсь продукт, ця назва записується у змінnyproduct.Потім цю змінну можна використовувати для прогнозу ціни, передати у функцію прогнозу:

```
period_type = st.selectbox('Виберіть тип періоду прогнозу:', ['Дні', 'Місяці',
'Роки'])
```

st.selectbox створює випадаючий список у інтерфейсі Streamlit.Підпис для користувача: Виберіть тип періоду прогнозу.Варіанти вибору елементи списку: ['Дні', 'Місяці', 'Роки'].Коли користувач вибирає один із варіантів, його значення зберігається у змінній period_type.Ця змінна потім використовується у функції predict_price, щоб визначити, як саме розраховувати коефіцієнт зміни цінизалежно від періоду часу.

```
period_value = st.number_input(f'Введіть кількість {period_type.lower()}:',
min_value=1, max_value=365, value=1, step=1)
```

st.number_input – це елемент інтерфейсу Streamlit для введення числового значення користувачем.Текст-підпис, який бачить користувач, динамічно формується: Введіть кількість. Додають вибраний раніше тип періоду:дні, місяці або роки. Параметри віджету number_input:

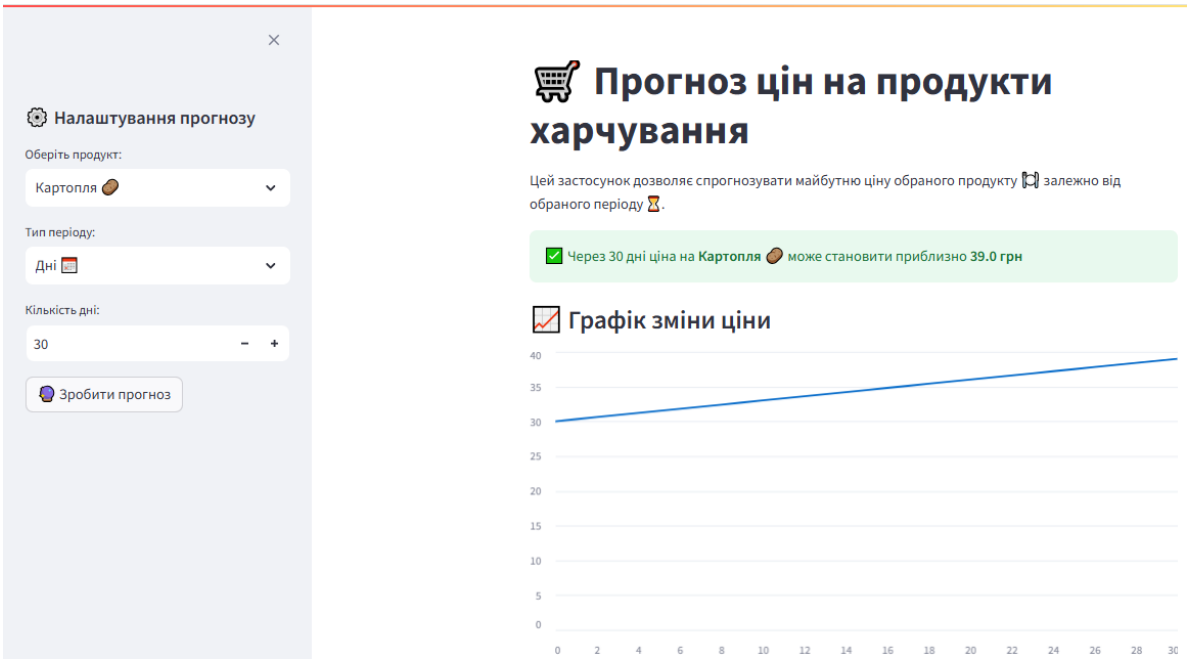
- min_value=1 – мінімально можливе число для вводу;
- max_value=365 – максимальне число;
- value=1 – значення за замовчуванням;
- step=1 – крок збільшення/зменшення при кліках на стрілки.

Користувач бачить інтуїтивний запит типу:введіть кількість днів або введіть кількість місяців. Вводить число, воно зберігається у змінній period_value.Потім це число разом з period_type і product передається у функцію прогнозу ціни.

```
if st.button('Зробити прогноз'):
    price = predict_price(product, period_value, period_type)
    st.success(f'Прогнозована ціна на {product} через {period_value}
{period_type.lower()} становить: {price} грн')
```

Створюють кнопку з написом Зробити прогноз в інтерфейсі Streamlit.Коли користувач натискає цю кнопку, умова в if стає істинною, код всередині виконується.Функція отримує вибраний продукт, період (значення і тип)і обчислює прогнозовану ціну.Виводиться повідомлення, яке містить прогнозовану ціну: прогнозована ціна на молоко через 3 місяці становить65,4 грн.

Користувач обирає продукт, тип періоду, вводить період часу, після цього натискає кнопку Зробити прогноз. Після цього бачить результат з прогнозованою ціною на вибраний товар.



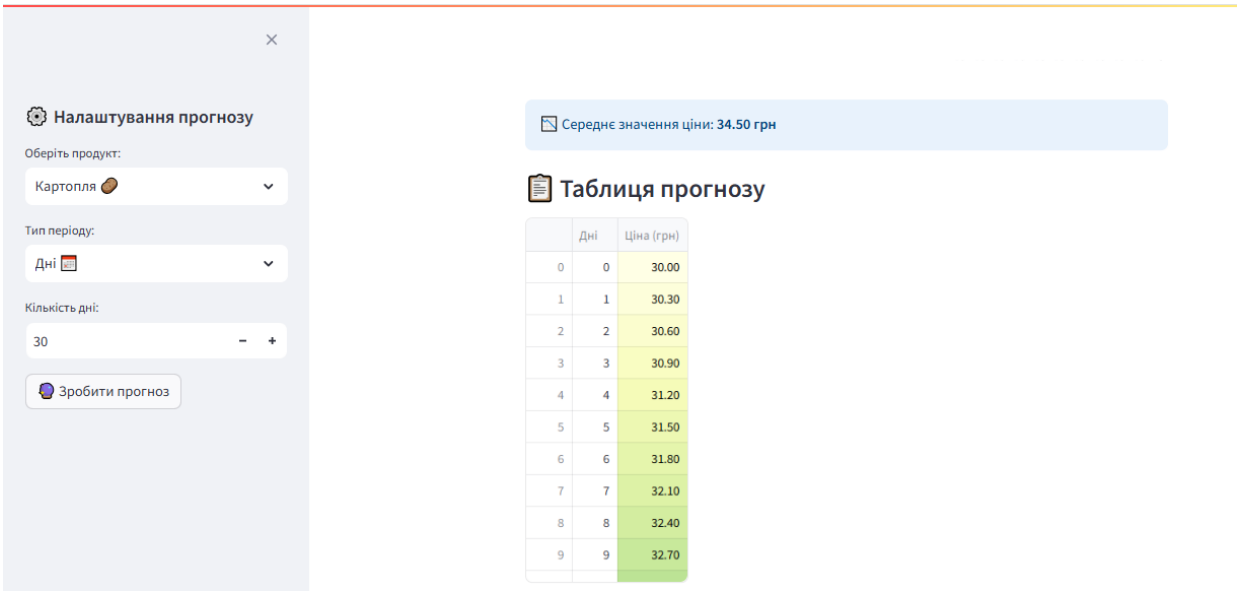


Рисунок 4.1–Графічний інтерфейс інформаційної системипрогнозу цін на продукти харчування.

ВИСНОВКИ ДО РОЗДІЛУ

У розділі 4 описано процес розробки програмного забезпечення для системи прогнозування попиту на продукти харчування.Визначено основні інструменти та бібліотеки мови Python, що використовуються для реалізації функціоналу системи. Вибрано архітектуру інформаційної системи, що забезпечує обробку, аналіз та візуалізацію даних.Розроблено інтерфейс користувача на базі бібліотеки Streamlit, що дозволяє взаємодіяти з системою.Створено алгоритми для реалізації математичного моделювання сезонних та трендових коливань.Проведено тестування розроблених функцій для виявлення та усунення помилок і підвищення їх надійності.Проведена перевірка роботи системи на історичних даних для підтвердження її точності і стабільності.Оцінено ефективність реалізованих алгоритмів у контексті точності прогнозування.Реалізована система дозволяє здійснювати аналіз та візуалізацію результатів прогнозування для прийняття обґрунтованих рішень.

РОЗДІЛ 5. РОЗРОБЛЕННЯ СТАРТАП-ПРОЄКТУ

5.1. Структура проекту інформаційної системи для прогнозування попиту на продукти харчування

Перед запуском проекту як стартапу необхідно виконати низку підготовчих заходів. Першочергово слід розробити структуру проєкту, яка відображає ключові його характеристики та приблизний бюджет, потрібний для його реалізації. До цього бюджету включають витрати на оплату праці розробників, що обчислюються відповідно до фаз розробки, а також витрати на оренду необхідного обладнання. Структура проєкту наведена в таблиці 5.1.

Таблиця 5.1–Структурапроєктуінформаційної системи

Назва	програма на мові Python
Назва проєкту	Інформаційна система прогнозування попиту на продукти харчування
Назва ВНЗ, факультету, спеціальності	НЛТУ, кафедра комп'ютерних наук, 122 «Комп'ютерні науки»
Прізвище, ім'я, по-батькові	Бродзінський Юрій Русланович
Цілі і задачі проєкту	Мета роботи – розроблення інформаційної системи для аналізу цін на продукти, застосування моделей прогнозування для оцінки майбутніх змін вартості цих показників.
	Задачі проєкту:
	1. Провести огляд сучасних методів і моделей прогнозування попиту на харчові продукти та їх технічного застосування.

Продовження таблиці 5.1

Цілі і задачі проєкту	2. Розробити математичну модель системи прогнозування, що враховує трендові та сезонні коливання на продукти харчування. 3. Реалізувати програмний модуль обробки та аналізу даних із використанням бібліотек Python, таких як Pandas, Numpy, Statsmodels. 4. Створити інтерфейс користувача на основі Streamlit для зручного доступу до функціоналу системи. 5. Провести тестування та валідацію розробленої системи на історичних даних для оцінки її точності та стабільності. 6. Проаналізувати результати прогнозування та запропонувати рекомендації щодо використання системи у практичній діяльності.

Перед запуском стартапу з впровадження системи прогнозування попиту на харчові продукти необхідно спланувати її структуру та визначити ключові характеристики. Це включає формування концептуальної моделі, бюджетування, оцінку ресурсних і технологічних витрат.

Базовими компонентами проекту є:

Математична модель прогнозування, що враховує трендові та сезонні коливання цін і споживчого попиту.Програмне забезпечення, реалізоване з використанням бібліотек Python (Pandas, Numpy, Statsmodels) для обробки даних та машинного навчання.Інтерфейс користувача, створений із застосуванням платформи Streamlit для зручного доступу та взаємодії.

Розподіл структури проекту дозволяє забезпечити гнучкість і масштабованість системи, а також можливість її інтеграції з існуючими інформаційними ресурсами роздрібних мереж і виробників.

5.2. Бюджетне планування та ресурсне забезпечення

Основні статті витрат включають:

- витрати на оплату праці розробників, які реалізують програмний модуль та інтерфейс;
- витрати на оренду серверного обладнання для розгортання системи;
- витрати на закупівлю супутніх ліцензійних та програмних ресурсів за потребою;
- оцінка протяжності та відповідних строків реалізації кожної фази проекту.

Згідно з графіком робіт, реалізація запланована протягом 2025 року з послідовним виконанням етапів від дослідження і постановки задачі до тестування й узгодження кінцевого результату.

5.3. Етапи розвитку стартапу

Залежно від позначених цілей і задач, визначені наступні етапи:

- аналіз ринку та потреб цільової аудиторії;
- розробка моделі та технічної архітектури системи;
- реалізація програмного модуля та інтерфейсу;
- тестування і валідація системи на історичних даних.
- проведення пілотних запусків у партнерських підприємствах та збір зворотного зв'язку.

Майбутня реалізація системи відкриває можливості для додаткового аналітичного моделювання за участю багатофакторних впливів, інтеграції з системами управління запасами та логістики, впровадження штучного інтелекту та автоматизації прийняття рішень, створення платформи для консультацій та аналізу ринку для виробників і роздрібних мереж.

На цьому початковому етапі відбувається формування ідеї, яка має потенціал рішення конкретної проблеми або заповнення існуючої прогалини на ринку. Важливо провести глибокий аналіз предметної області, визначити цільову аудиторію та її потреби. Проводиться збір первинної інформації, аналіз конкурентів

та існуючих альтернатив, а також визначення унікальної ціннісної пропозиції. Важливо створити концептуальний прототип або опис майбутньої системи, наприклад, у вигляді схем або ескізів інтерфейсу.

На етапі розробки бізнес-моделіформулюється деталізована бізнес-ідея і розробляється бізнес-модель. Визначаються джерела доходів, структура витрат, ключові ресурси і партнерства. Створюється бізнес-план, який містить фінансову оцінку, аналіз ринку та стратегію входу на ринок. У випадку з інформаційною системою прогнозування попиту на харчові продукти особливу увагу приділяється визначенню цільової нішевої аудиторії та користувачів системи, а також ключових функцій платформи.

На наступному етапі зосереджуються на створенні мінімально функціональної версії системи. Це дозволить протестувати ключові ідеї, залучити перших користувачів та отримати їх зворотній зв’язок. В процесі розробки визначаються і реалізуються найнеобхідніші функції: базові моделі прогнозування цін, інтерфейс користувача, базові механізми збору та обробки даних. Тестування системи допомагає виявити слабкі місця і коригує фундаментальні моменти проекту.

Після створення прототипу стартап починає активно шукати фінансування — через можливості інвесторів, венчурних фондів, державних грантів або приватних інвестицій. Важливим є підготовка презентаційних матеріалів - пітчингу, бізнес-плану, технічної документації. Залежно від рівня розвитку проектуготуються демо-версії або попередні моделі для демонстрації потенційним інвесторам.

Якщо стартап довів свою життєздатність і отримав перших користувачів та інвестиції, починається етап масштабування. Це означає додавання нових функцій, ускладнення моделей прогнозування, інтерфейсів, інтеграцію з іншими системами та розширення цільової аудиторії. Це також включає масштабування технічної інфраструктури, перехід на більш потужні сервери, хмарні сервіси, налагодження процесів підтримки користувачів та підвищення стабільності системи.

Щоб привернути широку аудиторію, активізується маркетингова стратегія: залучення користувачів через рекламні кампанії, публікації у засобах масової інформації, участь у тематичних виставках і конференціях, створення партнерських

відносин. Важливо також аналізувати поведінкові дані користувачів і вдосконалювати систему на основі зворотного зв’язку.

Стартап на етапі підтримка і вдосконаленнямає функціонуючу систему з стабільною базою користувачів, але постійно рухається до покращення послуг. Ведеться активне оновлення програми, виправлення помилок, додаються нові модулі та можливості, що відповідають новим вимогам ринку. Важливо також здійснювати моніторинг ключових показників ефективності, використовувати аналітичні дані для прийняття рішень.

Наступний етап - переведення проєкту у прибуткову і стабільну компанію. Це включає оптимізацію витрат, налаштування каналів збуту, можливості масштабування на інші регіони або країни. Оптимальною є стратегія постійного вдосконалення, пошук нових ринкових сегментів та парнерських можливостей, а також залучення додаткових інвестицій для розширення.

ВИСНОВКИ ДО РОЗДІЛУ

У розділі 5 сформульовано структуру стартап-проекту для реалізації системи прогнозування попиту на продукти харчування.Визначено ключові етапи підготовки, які включають розробку концепції, бюджетування та планування ресурсів.Окреслено основні заходи для створення команди розробників і залучення необхідного обладнання.Враховано необхідність аналізу цільової аудиторії та визначення її потреб у функціоналі системи.

Важливим аспектом стала оцінка фінансових витрат та складання бюджету на реалізацію проекту.Окрім технічних аспектів, розглянуто питання маркетингу та поширення продукту на ринку.Розроблена структура стартапу забезпечує логічний та послідовний підхід до його запуску та розвитку.Важливим результатом стала формалізація цілей і завдань, що сприятиме ефективному управлінню проєктом.Відповідна організаційна і фінансова підготовка створює умови для успішної реалізації і адаптації системи.

ВИСНОВКИ

Розроблено інформаційну систему для прогнозування попиту на харчові продукти в Україні, що базується на сучасних методах аналізу даних та машинного навчання.В рамках роботи було сформулювано та реалізовано математичну модель, яка враховує тренд і сезонні коливання вдинаміці цін.Реалізація інформаційної системи з використанням бібліотек Python, таких як Pandas, Numpy і Statsmodels, забезпечила ефективну обробку та аналіз даних.

Створений веб-інтерфейс на основі Streamlit дозволяє користувачам отримувати прогнози та аналізувати інформацію.Проведене тестування системи показало високий рівень точності прогнозів та стабільність роботи на історичних даних.Зїї допомогою системи виробники та роздрібні мережі можуть більш точно планувати запаси та формувати цінову політику.Впровадження системи сприяє підвищенню ефективності моніторингу ринку та швидкості реагування на зміни у попиті.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Юрченко М. Є.Прогнозування та аналіз часових рядів.Чернігів: ЧНТУ, 2018. – 88 с.

2.Сичевський М.П. Харчова промисловість як основа продовольчої безпеки та розвитку держави. Монографія. Київ: Аграр. Наука, 2019. –388 с.

3. Лупенко Ю. О., Пугачов М. І., Духницький Б. В. Формування глобального і регіонального ринків сільськогосподарської сировини та продовольства: монографія. Київ: ННЦ «ІАЕ», 2015. –320 с.

4.Andreyeva T., Long M.W., Brownell K. The Impact of Food Prices on Consumption: ASystematic Review of Research on the Price Elasticity of Demand for Food. American Journal of Public Health.2010.100(2). pp. 216-222.

5. Janssen M., Hamm U. The mandatory EU logo for organic food: consumer perceptions. British Food Journal. 2012. 114(3). pp. 335-352.

6. Lem A., Bjorndal T.,Lappo A. Economic analysis of supply and demand for food up to 2030 –Special focus on fish and fishery products. FAO Fisheries and Aquaculture Circular. 2014. No1089. Rome, FAO. –106 pp.

7.BolotovaY.V. Recent price developments in the United States potato industry. American Journal of Potato Research. 2017. Vol.94.pp.567-571.

8.LoyJ., RiekertS., SteinhagenC. Potato prices as affected by demand and yearly production: German perspective. American Journal of Potato Research. 2011. Vol.88. pp.195-198.

9.PavlistaA., FeuzD. Potato prices as affected by demand and yearly production. American Journal of Potato Research.2005. Vol.82. pp.339-343.

10.FirlejaK.,KubalaS.Determinants of variationof potato prices in the European Union. Economia Agro-Alimentare. 2020. Vol.3. pp.697-707.

11.DrachalK. Analysisof agricultural commodities prices with new Bayesianmodel combination schemes. Sustainability.2019. Vol.11(19). pp.5305-5328.

12.Офіційний сайт Державної служби статистики України. Реалізація продукції сільського господарства підприємствами та господарствами населення за 2012-2021рр.URL:

http://www.ukrstat.gov.ua/operativ/menu/menu_u/cg.htm

13.Офіційний сайтДержавної служби статистики України. Індекси цін продукції сільського господарства, реалізованої підприємствами за місяць (продукція рослинництва, 2012–2024рр.). URL:

http://www.ukrstat.gov.ua/operativ/operativ2020/sg/icsh/icsh2020_u.htm.

14. FAOSTAT(2020). Statistical database of food and agriculture organization of the United Nations, available at:http://faostat.fao.org/faostat

15. ЛубкоД.В., ШаровС.В. Методи та системи штучного інтелекту. Мелітополь: ФОП Однорог Т.В., 2019. –264 с.

16. Кудін О. В. Технології машинного навчання в обробці даних. Запоріжжя: Запорізький національний університет, 2022. –100 с.

ДОДАТКИ

ДОДАТОК А

1.ipynb

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime, timedelta
import warnings
warnings.filterwarnings('ignore')

from sklearn.preprocessing import MinMaxScaler, StandardScaler, RobustScaler
from sklearn.model_selection import train_test_split, TimeSeriesSplit
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR

import tensorflow as tf
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import LSTM, Dense, Dropout, Bidirectional, Conv1D,
MaxPooling1D, Flatten
from tensorflow.keras.optimizers import Adam, RMSprop
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau,
ModelCheckpoint
from tensorflow.keras.regularizers import l1_l2

import plotly.graph_objects as go
from plotly.subplots import make_subplots
import plotly.express as px
import statsmodels.api as sm
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.stattools import adfuller
import scipy.stats as stats

class Config:
def __init__(self):
self.data_urls = {
            'primary': 'https://data.gov.ua/dataset/b5df316f-dfbe-4cb0-b596-
807a301dd0bc/download',
```

```
            'backup': 'https://data.gov.ua/dataset/alternative-source/data.xlsx'
        }

        self.SEQ_LENGTH = 30
        self.TEST_SIZE = 0.2
        self.VALIDATION_SIZE = 0.1
        self.RANDOM_STATE = 42

        self.BATCH_SIZE = 32
        self.EPOCHS = 100
        self.LEARNING_RATE = 0.001

        self.LSTM_UNITS = [64, 32]
        self.DENSE_UNITS = [16, 8, 1]
        self.DROPOUT_RATE = 0.2

        self.PRODUCTS = ['Картопля', 'Капуста', 'Цибуля', 'Морква', 'Яблука']

        self.COLORS = {
            'primary': '#2E86AB',
            'secondary': '#A23B72',
            'tertiary': '#F18F01',
            'background': '#F7F7F7'
        }

config = Config()

class DataProcessor:
def __init__(self, config):
        self.config = config
        self.data = None
        self.processed_data = {}

def load_data(self):
print("Завантаженняданих...")
try:
            self.data = pd.read_excel(self.config.data_urls['primary'])
print("✓ Дані успішно завантажено з основного джерела")
exceptExceptionase:
print(f"✗ Помилка завантаження з основного джерела: {e}")
try:
                self.data = pd.read_excel(self.config.data_urls['backup'])
```

```
print("✓ Дані завантажено з резервного джерела")
exceptExceptionase2:
print(f"✗ Помилка завантаження з резервного джерела: {e2}")
return self.data


def explore_data(self):
print("\n" + "="*50)
print("АНАЛІЗ ДАНИХ")
print("="*50)
if self.data is None:
print("Данінезавантажені!")
return

print(f"Розмірдатасету: {self.data.shape}")
print(f"Колонки: {list(self.data.columns)}")
print("\nПерші 5 записів:")
print(self.data.head())
print("\nОстанні 5 записів:")
print(self.data.tail())
print("\nТипи даних:")
print(self.data.dtypes)
print("\nСтатистика даних:")
print(self.data.describe())

print("\nПропущенізначення:")
missing_data = self.data.isnull().sum()
print(missing_data[missing_data > 0])

print(f"\nУнікальні продукти: {self.data['Product'].unique()}")

print(f"Діапазон дат: від {self.data['Date'].min()} до {self.data['Date'].max()}")

def preprocess_data(self):
print("\nПопередняобробкаданих...")
self.data['Date'] = pd.to_datetime(self.data['Date'])
        self.data = self.data.sort_values(['Product', 'Date']).reset_index(drop=True)
self.data['Year'] = self.data['Date'].dt.year
self.data['Month'] = self.data['Date'].dt.month
self.data['Day'] = self.data['Date'].dt.day
self.data['DayOfWeek'] = self.data['Date'].dt.dayofweek
self.data['DayOfYear'] = self.data['Date'].dt.dayofyear
```

```
self.data['Week'] = self.data['Date'].dt.isocalendar().week
self.data['Quarter'] = self.data['Date'].dt.quarter

self.data['IsWeekend'] = self.data['DayOfWeek'].isin([5, 6]).astype(int)
self.data['Season'] = self.data['Month'] % 12 // 3 + 1

        self.data = self.data.ffill()  # Заповнення вперед

        self.data = self.data.drop_duplicates()
print("✓ Дані успішно оброблено")
return self.data

def create_sequences(self, data, seq_length, target_col='Price'):
        X, y = [], []
for i in range(len(data) - seq_length):
X.append(data[i:(i + seq_length)])
y.append(data[i + seq_length][target_col])
return np.array(X), np.array(y)

def prepare_product_data(self, product_name):
print(f"Підготовка даних для продукту: {product_name}")
product_data = self.data[self.data['Product'] == product_name].copy()
if len(product_data) == 0:
print(f"✗ Дані для продукту {product_name} не знайдено!")
return None, None, None, None
        feature_columns = ['Price', 'Month', 'DayOfWeek', 'DayOfYear', 'IsWeekend',
'Season']
scalers = {}
        scaled_features = []
for col in feature_columns:
if col == 'Price':
scaler = RobustScaler()
else:
scaler = MinMaxScaler()
            scaled_col = scaler.fit_transform(product_data[col].values.reshape(-1,
1))
            scaled_features.append(scaled_col)
scalers[col] = scaler

        features_scaled = np.hstack(scaled_features)

        X, y = self.create_sequences(features_scaled, self.config.SEQ_LENGTH)
```

```
        split_idx = int(len(X) * (1 - self.config.TEST_SIZE))
        X_train = X[:split_idx]
        X_test = X[split_idx:]
        y_train = y[:split_idx]
        y_test = y[split_idx:]

        self.processed_data[product_name] = {
            'X_train': X_train, 'X_test': X_test,
            'y_train': y_train, 'y_test': y_test,
            'scalers': scalers,
            'original_data': product_data
        }
print(f"✓ Дані для {product_name} підготовлено: "
f"тренувальних {len(X_train)}, тестових {len(X_test)}")
return X_train, X_test, y_train, y_test, scalers

classStatisticalAnalyzer:
def __init__(self, data_processor):
        self.data_processor = data_processor
        self.results = {}

def perform_time_series_analysis(self, product_name):
"""Виконання повного аналізу часових рядів"""
print(f"\nСтатистичний аналіз для {product_name}...")
product_data = self.data_processor.data[
            self.data_processor.data['Product'] == product_name
        ].copy()
if len(product_data) == 0:
return None
        price_series = product_data.set_index('Date')['Price']
        adf_result = adfuller(price_series.dropna())
decomposition = seasonal_decompose(
            price_series,
model='additive',
period=365,  # Річна сезонність
            extrapolate_trend='freq'
        )
        correlation_features = ['Price', 'Month', 'DayOfWeek', 'DayOfYear', 'Season']
        correlation_matrix = product_data[correlation_features].corr()
self.results[product_name] = {
            'adf_test': {
                'statistic': adf_result[0],
                'p_value': adf_result[1],
```

```
                'is_stationary': adf_result[1] < 0.05
            },
            'decomposition': decomposition,
            'correlation_matrix': correlation_matrix,
            'basic_stats': {
                'mean': price_series.mean(),
                'std': price_series.std(),
                'min': price_series.min(),
                'max': price_series.max(),
                'trend': 'Зростаючий' if price_series.iloc[-1] > price_series.iloc[0]
else 'Спадний'
            }
        }
return self.results[product_name]

def generate_statistical_report(self, product_name):
if product_name not in self.results:
return "Аналіз не виконано для даного продукту"
result = self.results[product_name]

        report = f"""
        СТАТИСТИЧНИЙ ЗВІТ ДЛЯ {product_name.upper()}
        ==========================================
        ОСНОВНІ СТАТИСТИКИ:
        - Середнє значення: {result['basic_stats']['mean']:.2f}
        - Стандартне відхилення: {result['basic_stats']['std']:.2f}
        - Мінімальна ціна: {result['basic_stats']['min']:.2f}
        - Максимальна ціна: {result['basic_stats']['max']:.2f}
        - Загальний тренд: {result['basic_stats']['trend']}
        ТЕСТ СТАЦІОНАРНОСТІ (Augmented Dickey-Fuller):
        - Статистика: {result['adf_test']['statistic']:.4f}
        - P-значення: {result['adf_test']['p_value']:.4f}
        - Стаціонарний: {'Так' if result['adf_test']['is_stationary'] else 'Ні'}
        ВИСНОВКИ:
        {self._generate_insights(result)}
        """
return report

class ModelFactory:
def __init__(self, config):
self.config = config

def create_lstm_model(self, input_shape):
```

```
model = Sequential([
LSTM(self.config.LSTM_UNITS[0],
                 return_sequences=True,
                 input_shape=input_shape,
                 kernel_regularizer=l1_l2(0.01, 0.01)),
Dropout(self.config.DROPOUT_RATE),
LSTM(self.config.LSTM_UNITS[1],
                 return_sequences=False,
                 kernel_regularizer=l1_l2(0.01, 0.01)),
Dropout(self.config.DROPOUT_RATE),
Dense(self.config.DENSE_UNITS[0], activation='relu'),
Dropout(self.config.DROPOUT_RATE),
Dense(self.config.DENSE_UNITS[1], activation='relu'),
Dense(self.config.DENSE_UNITS[2])
        ])
optimizer = Adam(learning_rate=self.config.LEARNING_RATE)
model.compile(optimizer=optimizer, loss='mse', metrics=['mae'])
return model

def create_conv_lstm_model(self, input_shape):
model = Sequential([
Conv1D(filters=64, kernel_size=3, activation='relu', input_shape=input_shape),
MaxPooling1D(pool_size=2),
LSTM(50, return_sequences=True),
Dropout(0.2),
LSTM(50, return_sequences=False),
Dropout(0.2),
Dense(50, activation='relu'),
Dense(1)
        ])
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
return model

def create_bidirectional_lstm(self, input_shape):
model = Sequential([
Bidirectional(LSTM(64, return_sequences=True), input_shape=input_shape),
Dropout(0.2),
Bidirectional(LSTM(32)),
Dropout(0.2),
Dense(16, activation='relu'),
Dense(1)
        ])
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
```

```
return model

class ModelTrainer:
def __init__(self, config):
        self.config = config
        self.model_factory = ModelFactory(config)
        self.trained_models = {}
        self.training_history = {}

def train_models(self, X_train, y_train, X_val, y_val, product_name):
print(f"\nНавчаннямоделейдля {product_name}...")
models = {}
callbacks = [
EarlyStopping(monitor='val_loss', patience=15, restore_best_weights=True),
ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=10, min_lr=
1e-7),
ModelCheckpoint(
f'best_model_{product_name}.h5',
monitor='val_loss',
                save_best_only=True
            )
        ]

        lstm_model = self.model_factory.create_lstm_model((X_train.shape[1],
X_train.shape[2]))
print("Навчання LSTM моделі...")
        history_lstm = lstm_model.fit(
            X_train, y_train,
            batch_size=self.config.BATCH_SIZE,
epochs=self.config.EPOCHS,
            validation_data=(X_val, y_val),
callbacks=callbacks,
verbose=1
        )
models['LSTM'] = {
            'model': lstm_model,
            'history': history_lstm
        }


def evaluate_models(self, X_test, y_test, scalers, product_name):
print(f"\nОцінка моделей для {product_name}...")
if product_name not in self.trained_models:
```

```
print("Моделі не навчені!")
return None
        evaluation_results = {}
for model_name, model_data in self.trained_models[product_name].items():
model = model_data['model']
            predictions_scaled = model.predict(X_test)
            price_scaler = scalers['Price']
predictions = price_scaler.inverse_transform(predictions_scaled)
actual = price_scaler.inverse_transform(y_test.reshape(-1, 1))
mse = mean_squared_error(actual, predictions)
mae = mean_absolute_error(actual, predictions)
rmse = np.sqrt(mse)
            r2 = r2_score(actual, predictions)
mape = np.mean(np.abs((actual - predictions) / actual)) * 100
            evaluation_results[model_name] = {
                'predictions': predictions,
                'actual': actual,
                'metrics': {
                    'MSE': mse,
                    'MAE': mae,
                    'RMSE': rmse,
                    'R2': r2,
                    'MAPE': mape
                }
            }
print(f"\n{model_name} - Результати оцінки:")
print(f"  MSE: {mse:.3f}")
print(f"  MAE: {mae:.3f}")
print(f"  RMSE: {rmse:.3f}")
print(f"  R²: {r2:.3f}")
print(f"  MAPE: {mape:.2f}%")
return evaluation_results

class Visualizer:
def __init__(self, config):
        self.config = config
def create_comprehensive_plots(self, data_processor, statistical_analyzer,
model_trainer):
for product in data_processor.processed_data.keys():
            self._create_product_analysis_plots(product, data_processor,
statistical_analyzer, model_trainer)
```

```
def _create_product_analysis_plots(self, product_name, data_processor,
statistical_analyzer, model_trainer):
product_data = data_processor.processed_data[product_name]['original_data']
fig = make_subplots(
rows=3, cols=2,
            subplot_titles=(
f'Динаміка цін на {product_name}',
'Сезонна декомпозиція',
                'Гістограма розподілу цін',
                'Прогнозування vs Реальні значення',
                'Помилки прогнозування',
                'Коефіцієнти кореляції'
            ),
specs=[
[{"secondary_y":False}, {"secondary_y": False}],
[{"secondary_y": False}, {"secondary_y": False}],
                [{"secondary_y": False}, {"secondary_y": False}]
            ]
        )
        fig.add_trace(
            go.Scatter(x=product_data['Date'], y=product_data['Price'],
name='Ціна', line=dict(color=self.config.COLORS['primary'])),
row=1, col=1
        )

if product_name in statistical_analyzer.results:
decomposition = statistical_analyzer.results[product_name]['decomposition']
dates = product_data['Date'][:len(decomposition.trend)]
            fig.add_trace(
                go.Scatter(x=dates, y=decomposition.trend, name='Тренд',
line=dict(color=self.config.COLORS['secondary'])),
row=1, col=2
            )

        fig.add_trace(
            go.Histogram(x=product_data['Price'], name='Розподіл цін',
                         marker_color=self.config.COLORS['tertiary']),
row=2, col=1
        )
fig.update_layout(
height=1200,
showlegend=True,
            title_text=f"Комплексний аналіз {product_name}",
```

```
template="plotly_white"
        )
fig.show()

def generate_final_report(self, data_processor, statistical_analyzer, model_trainer):
report = """
        ЗВІТ ПРО РОБОТУ ІНФОРМАЦІЙНОЇ СИСТЕМИ
        ПРОГНОЗУВАННЯ ПОПИТУ НА ХАРЧОВІ ПРОДУКТИ
        ========================================
"""
for product in data_processor.processed_data.keys():
report += f"\nАНАЛІЗ ДЛЯ ПРОДУКТУ: {product.upper()}\n"
report += "-" * 40 + "\n"
if product in statistical_analyzer.results:
                stats_report =
statistical_analyzer.generate_statistical_report(product)
report += stats_report + "\n"
if product in model_trainer.trained_models:
report += "РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ:\n"
report += "\n" + "="*50 + "\n"
return report

def main():
print("="*70)
print("ІНФОРМАЦІЙНА СИСТЕМА ПРОГНОЗУВАННЯ ПОПИТУ НА ХАРЧОВІ ПРОДУКТИ")
print("ДИПЛОМНА РОБОТА - РОЗШИРЕНА ВЕРСІЯ")
print("="*70)
config = Config()
data_processor = DataProcessor(config)
    statistical_analyzer = StatisticalAnalyzer(data_processor)
    model_trainer = ModelTrainer(config)
visualizer = Visualizer(config)

print("\nЕТАП 1: ЗБІР ТА ПІДГОТОВКА ДАНИХ")
data_processor.load_data()
    data_processor.explore_data()
    data_processor.preprocess_data()

print("\nЕТАП 2: СТАТИСТИЧНИЙ АНАЛІЗ")
for product in config.PRODUCTS:
        statistical_analyzer.perform_time_series_analysis(product)
report = statistical_analyzer.generate_statistical_report(product)
print(report)
```

```
print("\nЕТАП 3: ПІДГОТОВКАДАНИХДЛЯМОДЕЛЮВАННЯ")
for product in config.PRODUCTS:
        data_processor.prepare_product_data(product)

print("\nЕТАП 4: НАВЧАННЯ МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ")
for product in config.PRODUCTS:
if product in data_processor.processed_data:
data = data_processor.processed_data[product]
            X_temp, X_val, y_temp, y_val = train_test_split(
data['X_train'], data['y_train'],
                test_size=config.VALIDATION_SIZE,
                random_state=config.RANDOM_STATE
            )
            model_trainer.train_models(X_temp, y_temp, X_val, y_val, product)

print("\nЕТАП 5: ОЦІНКА ЯКОСТІ МОДЕЛЕЙ")
for product in config.PRODUCTS:
if product in data_processor.processed_data:
data = data_processor.processed_data[product]
            model_trainer.evaluate_models(
data['X_test'], data['y_test'], data['scalers'], product
            )

print("\nЕТАП 6: ВІЗУАЛІЗАЦІЯ РЕЗУЛЬТАТІВ")
    visualizer.create_comprehensive_plots(data_processor, statistical_analyzer,
model_trainer)

print("\nЕТАП 7: ГЕНЕРАЦІЯ ФІНАЛЬНОГО ЗВІТУ")
final_report = visualizer.generate_final_report(data_processor, statistical_analyzer,
model_trainer)
print(final_report)
print("\n" + "="*70)
print("РОБОТУ СИСТЕМИ ЗАВЕРШЕНО!")
print("="*70)

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

ДОДАТОК Б

2.py

```
import streamlit as st
import numpy as np
import pandas as pd

products = ['Картопля', 'Молоко', 'Хліб', 'М’ясо', 'Овочі']

def predict_price(product, period_value, period_type):
base_prices = {
        'Картопля': 10,
        'Молоко': 20,
        'Хліб': 15,
'М’ясо': 50,
        'Овочі': 12
    }

    base_price = base_prices.get(product, 10)

factor = 1 + 0.01 * period_value

if period_type == 'Дні':
factor = 1 + 0.01 * period_value
elif period_type == 'Місяці':
factor = 1 + 0.03 * period_value
elif period_type == 'Роки':
factor = 1 + 0.2 * period_value

    predicted_price = base_price * factor
return round(predicted_price, 2)

st.title('Прогнозціннапродуктихарчування')

product = st.selectbox('Оберітьпродукт:', products)

period_type = st.selectbox('Виберітьтипперіодупрогнозу:', ['Дні', 'Місяці', 'Роки'])

period_value = st.number_input(f'Введітькількість {period_type.lower()}:',
min_value=1, max_value=365, value=1, step=1)
ifst.button('Зробитипрогноз'):
```

```
price = predict_price(product, period_value, period_type)
st.success(f'Прогнозована ціна на {product} через {period_value}
{period_type.lower()} становить: {price} грн')
```