НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

На правах рукопису
УДК 004.942+001.5

До захисту допущено
Завідувач кафедри ММСА
_____ Оксана ТИМОЩУК
«___» _____2024 р.

Магістерська дисертація
на здобуття ступеня магістра
за освітньо-професійною програмою «Системний аналіз фінансового ринку»
зі спеціальності 124 «Системний аналіз»
на тему: «Система прогнозування метеорологічних умов на основі методів аналізу даних та штучного інтелекту»

Виконав:
студент ІІ курсу, групи КА-22мп
Іванійчук Арсен Павлович _____

Керівник:
доцент кафедри ШІ, PhD
Гуськова Віра Генадіївна _____

Рецензент:
старший викладач НТУУ КПІ ім. Ігоря Сікорського
к.т.н. Шаповал Наталія Віталіївна _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань
Студент (*підпис*): _____

Київ-2024

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

Рівень вищої освіти — другий (магістерський)
Спеціальність — 124 «Системний аналіз»
Освітньо-професійною програмою «Системний аналіз фінансового ринку»

ЗАТВЕРДЖУЮ
Завідувач кафедри ММСА
_____ Оксана ТИМОЩУК
«___» _____2024 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Іванійчуку Арсену Павловичу

1. Тема дисертації: «Система прогнозування метеорологічних умов на основі методів аналізу даних та штучного інтелекту», науковий керівник дисертації доцент кафедри ШІ, PhD Гуськова Віра Генадіївна, затверджені наказом по університету від «08» листопада 2023 р. № 5200-с

2. Строк подання студентом дисертації: _____2024 р.

3. Об’єкт дослідження: метеорологічні дані.

4. Предмет дослідження: розробка і валідація моделей штучного інтелекту, здатних точно прогнозувати метеорологічні умови.

5. Перелік завдань, які потрібно розробити:
1) дослідити моделі і методи штучного інтелекту та аналізу даних для створення системи;
2) зібрати необхідний набір даних;
3) створити (вибрати) програмне забезпечення для виконання обчислювальних експериментів;
4) побудувати модель і обчислити характеристики її адекватності та якості прогнозів;
5) виконати аналіз результатів.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу:
1) блок-схеми (рис.);

2) графіки отриманих результатів (рис.);
3) таблиці у розділі стартап-проекту.

7. Орієнтовний перелік публікацій:

1. Іванійчук А.П., Гуськова В.Г. Система прогнозування метеорологічних умов на основі методів аналізу даних та штучного інтелекту. II Всеукраїнська науково-практична конференція «Системні науки та інформатика» з нагоди 125-річчя КПІ ім. Ігоря Сікорського, м. Київ, 04-08 грудня 2023 року. С. 121-126.

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання: 01.09.2023 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Інструктаж з техніки безпеки	01.09.2023-06.09.2023	виконано
2	Затвердження теми магістерської дисертації (МД)	07.09.2023-13.09.2023	виконано
3	Ознайомлення з ДСТУ 3008:2015 та стандарти Єдиної системи програмної документації	14.09.2023-20.09.2023	виконано
4	Ознайомлення з державним стандартом України ДСТУ ГОСТ 7.1:2006	21.09.2023-27.09.2023	виконано
5	Проведення дослідження за темою під керівництвом наукового керівника	28.09.2023-04.10.2023	виконано
6	Завершення роботи над першим варіантом основної частини МД	05.10.2023-11.10.2023	виконано
7	Продовження роботи над експериментальною частиною МД та програмним забезпеченням, оформлення розділу із стартап-проекту	12.10.2023-18.10.2023	виконано
8	Завершення розділу стартап-проекту та оформлення першого варіанту МД	19.10.2023-04.12.2023	виконано
9	Виконання умов нормоконтролю та оформлення остаточного варіанту МД	05.12.2023-31.12.2023	виконано

Студент Арсен ІВАНІЙЧУК

Науковий керівник дисертації Віра ГУСЬКОВА

РЕФЕРАТ

Магістерська дисертація містить 76 с., 14 табл., 18 рис., 1 додаток, 23 джерела.

Актуальність теми: актуальність теми полягає у значній перспективності застосування новітніх досягнень у сфері штучного інтелекту та аналізу даних для прогнозування метеорологічних умов. Проблема отримання точного прогнозу погодних умов є та залишатиметься дуже важливою через значний вплив на усі аспекти життя.

Об’єкт дослідження: метеорологічні дані.

Предмет дослідження: розробка і валідація моделей штучного інтелекту, здатних точно прогнозувати метеорологічні умови.

Мета дослідження: створення системи прогнозування метеорологічних показників на основі моделей та методів штучного інтелекту.

Постановка задачі: основною поставленою задачею було створення системи прогнозування метеорологічних показників на основі моделей та методів штучного інтелекту, яка б дозволила отримувати надійний прогноз бажаних показників.

За допомогою мови програмування Python було створено бажану систему, яка на наборі вхідних даних про метеорологічні показники дозволяє виконати прогнозування за допомогою моделі LSTM.

Ключові слова: RCNN, LSTM, ПРОГНОЗУВАННЯ ПОГОДНИХ УМОВ

ABSTRACT

The master's thesis contains 76 pages, 14 tables, 18 figures, 1 appendix, 23 sources.

Relevance of the topic: The relevance of the topic lies in the significant prospects of applying the latest achievements in the field of artificial intelligence and data analysis for predicting meteorological conditions. The problem of obtaining an accurate weather forecast is and will remain very important due to its significant impact on all aspects of life.

Object of research: meteorological data.

Subject of research: development and validation of artificial intelligence models capable of accurately predicting meteorological conditions.

Purpose of the study: to create a system for forecasting meteorological indicators based on models and methods of artificial intelligence.

Setting the task: the main task set was to create a system for forecasting meteorological indicators based on models and methods of artificial intelligence, which would allow for reliable forecasting of desired indicators.

We used the Python programming language to design the desired system, which obtains a set of input data about meteorological indicators and allows the forecasting using the LSTM model.

Keywords: RCNN, LSTM, WEATHER FORECASTING

ЗМІСТ

ВСТУП

Штучний інтелект (ШІ) та передові техніки аналізу даних революціонізують область метеорології, вносячи значні поліпшення у точність погодних прогнозів. Використання ШІ дозволяє перейти від традиційних лінійних моделей, які не завжди ефективно відображають складність атмосферних процесів, до більш комплексних підходів. Це стає можливим завдяки здатності ШІ аналізувати великі масиви даних, виявляючи складні закономірності і роблячи прогнози на підставі глибокого навчання, яке враховує нелінійні зв'язки.

Сучасні потреби різних галузей економіки та повсякденного життя роблять точні погодні прогнози надзвичайно важливими. Наприклад, у аграрному секторі, де планування посіву, зрошення та збору врожаю сильно залежить від погоди, точні прогнози допомагають мінімізувати ризики від негативних кліматичних умов. В енергетичній галузі, з особливим акцентом на відновлювані джерела енергії, як-от вітрові та сонячні електростанції, прогнози погоди є ключовими для оптимізації виробництва та розподілу енергії. Також у транспорті точні прогнози відіграють роль у плануванні маршрутів, допомагаючи уникати затримок і аварій через погані погодні умови.

Дисертація складається із вступу, чотирьох розділів, висновків, переліку джерел посилання та одного додатку. У першому розділі розглядається актуальність теми та існуючі рішення у даній області, а також міститься постановка задачі. У другому розділі розглядаються моделі і методи штучного інтелекту та аналізу даних, що можуть бути використані для вирішення проблеми метеорологічного прогнозування. У третьому розділі описано процес розробки системи прогнозування метеорологічних умов на основі методів штучного інтелекту та аналізу даних. У четвертому розділі міститься

план розробки стартап-проєкту на основі ідеї такої системи. У додатку А наведено лістинг коду реалізованого програмного забезпечення.

РОЗДІЛ 1 АКТУАЛЬНІСТЬ ТЕМИ ДОСЛІДЖЕННЯ, ОГЛЯД ІСНУЮЧИХ РІШЕНЬ. ПОСТАНОВКА ЗАДАЧІ

1.1 Актуальність теми дослідження

Актуальність теми дослідження можна сформулювати наступним чином: постійні зміни клімату, зростання екстремальних погодних явищ, таких як урагани, зливи, посухи, та їхній вплив на різні сфери людської діяльності зробили точне та ефективне прогнозування погоди одним із ключових викликів сучасності.

Застосування штучного інтелекту (ШІ) та методів аналізу даних в метеорології відкриває нові можливості для підвищення точності прогнозів. Традиційні підходи до прогнозування погоди часто обмежені лінійними моделями та не можуть адекватно враховувати складні взаємозв'язки в атмосферних процесах. Впровадження ШІ дозволяє аналізувати великі обсяги даних, виявляти нетривіальні закономірності та робити прогнози на основі глибоких навчальних мереж, які здатні моделювати нелінійні залежності.

Актуальність цього напрямку дослідження також обумовлена зростаючою потребою в точних метеорологічних прогнозах у різних галузях. Наприклад, в сільському господарстві точний прогноз допомагає планувати посівні роботи, зрошення та збирання врожаю, мінімізуючи втрати від несприятливих погодних умов. В енергетиці прогнозування погоди важливе для управління виробництвом та розподілом енергії, особливо з урахуванням зростаючої частки відновлюваних джерел енергії, таких як вітрова та сонячна енергія. У транспортній сфері точні погодні прогнози дозволяють ефективно планувати маршрути, запобігаючи затримкам та аваріям, пов'язаним з поганими погодними умовами.

Крім того, покращення систем прогнозування погоди має велике значення для підвищення громадської безпеки. Ефективні системи

прогнозування можуть надавати своєчасні попередження про небезпечні погодні явища, такі як шторми, тайфуни та повені, дозволяючи своєчасно евакуювати людей та вживати заходів щодо захисту майна. Також це важливо для реагування на надзвичайні ситуації, де швидкість та точність прогнозу можуть врятувати життя.

1.2 Постановка задачі

Початковим кроком є проведення огляду існуючих інструментів прогнозування погоди, аналізуючи наукові та технічні джерела для оцінки сучасних технологій у галузі метеорологічного прогнозування. На основі цього огляду необхідно розробити та тестувати моделі штучного інтелекту, які можуть ефективно прогнозувати метеорологічні умови, використовуючи як історичні, так і актуальні дані. Це передбачає створення прототипу системи, що інтегрує обрані алгоритми штучного інтелекту для точного прогнозування метеорологічних умов. Важливим етапом буде тестування та оцінка системи через проведення ряду тестів для визначення точності, надійності та ефективності розробленої системи в реальних метеорологічних сценаріях. Нарешті, дослідження має включати аналіз потенційного впливу розробленої системи на різні сфери, такі як сільське господарство, транспорт, планування в екстрених ситуаціях, та оцінювання її можливостей для подальшого розширення та вдосконалення.

1.3 Огляд існуючих рішень

Проблема прогнозування погодних умов є однією з найбільш давніх та важливих проблем, які вирішує людство. Це зумовлено тим, що адекватне прогнозування погодних умов має вирішальне значення у багатьох аспектах людського життя, включаючи сільське господарство, транспорт, будівництво, а також для забезпечення безпеки населення в умовах екстремальних метеорологічних явищ.

З найдавніших часів люди спостерігали за природними явищами, намагаючись передбачити зміни погоди. Це було здійснено шляхом спостереження за поведінкою тварин, рослин, змінами у зовнішньому вигляді неба та іншими природними індикаторами. Проте, ці методи були досить обмежені та ненадійні, оскільки базувалися переважно на суб'єктивних оцінках та відсутності систематичного підходу.

Розвиток науки та технологій привів до значного удосконалення методів прогнозування погоди. Одним із ключових моментів у цьому контексті є використання історичних даних. Збір та аналіз історичних даних дозволили виявити певні закономірності та тенденції у погодних умовах, що стало основою для розробки більш точних методів прогнозування.

Важливою статистичною технікою, яка використовує історичні дані, є тест Манна-Кендала. Цей непараметричний статистичний тест дозволяє визначити наявність та напрямок трендів у часових рядах, що є особливо корисним у метеорології для аналізу змін клімату та погодних умов протягом тривалого часу. Застосування тесту Манна-Кендала в метеорології дозволяє виявити, наприклад, тенденції до зміни температур, опадів, вітрових умов, що є важливим для розуміння довготривалих змін у погодних патернах.

Ще одним важливим інструментом у прогнозуванні погоди є регресійний аналіз. Цей метод дозволяє виявити взаємозв'язки між різними метеорологічними параметрами та використовувати ці зв'язки для

прогнозування майбутніх погодних умов. Регресійний аналіз вимагає великої кількості історичних даних, що дозволяє створювати більш точні та надійні моделі прогнозування.

Серед моделей регресійного аналізу важливе місце займає модель Ангстрома-Прескотта, яка використовується для оцінки сонячної радіації на поверхні Землі на основі кількох метеорологічних параметрів, таких як хмарність та сонячне сяйво. Ця модель є особливо важливою для сільського господарства та відновлювальної енергетики, оскільки дозволяє з високою точністю прогнозувати сонячну радіацію, що є ключовим фактором у цих галузях.

Іншим потужним інструментом сучасного прогнозування погоди є дискретизація. Дискретизація в метеорології - це математична техніка, яка є важливою для моделювання складних фізичних явищ, таких як погода. Цей процес включає поділ атмосфери Землі на сітку з тисяч тривимірних кубів або комірок. Кожна комірка у цій сітці представляє окрему частину атмосфери.

У кожному кубі вчені відстежують різні метеорологічні змінні, такі як температура, тиск, вологість та швидкість вітру. Поведінка в одній комірці впливає на сусідні комірки, відображаючи взаємопов'язаний характер атмосферних процесів. Цей підхід є ключовим для розуміння та прогнозування погодних умов, оскільки він дозволяє отримати детальний, всебічний огляд атмосферних умов на глобальному рівні.

Точність прогностичних моделей погоди покращується з більш дрібною дискретизацією, що означає менші та численніші куби. Ця більш дрібна сітка дозволяє точніше і детальніше представити атмосферу, що призводить до кращих прогнозів погодних подій.

Ефективність дискретизації значною мірою залежить від потужності обчислювальної техніки та математичних методів. Розвинені обчислювальні можливості дозволяють моделювати та аналізувати ці дискретні елементи, роблячи можливим моделювання складних погодних систем з більшою точністю.

1.4 Обґрунтування вибору мови програмування. Мова програмування Python

Мова програмування Python - це високорівнева, інтерпретована мова з простим і читабельним синтаксисом, що робить її особливо популярною серед розробників по всьому світу. Вкажемо кілька ключових переваг Python, які сприяли його широкому використанню, особливо в областях статистики та нейронних мереж.

1. Легкість вивчення і читабельність: Python має простий і зрозумілий синтаксис, що полегшує вивчення мови новачками та сприяє ефективному написанню коду. Чітка структура і наявність великої кількості документації роблять Python доступним для широкого кола розробників.
2. Велика стандартна бібліотека і спільнота: Python має одну з найбільших бібліотек розширень та модулів, які покривають широкий спектр потреб у програмуванні, включаючи статистику та обробку даних. Бібліотеки, такі як NumPy, Pandas, Matplotlib, Scikit-learn та TensorFlow, забезпечують потужні інструменти для аналізу даних, статистичного моделювання та роботи з нейронними мережами.
3. Мультипарадигмальність: Python підтримує різні стилі програмування - об'єктно-орієнтоване, процедурне та функціональне. Це дає розробникам гнучкість у виборі найбільш ефективного підходу для кожної конкретної задачі.
4. Інтеграція і розширюваність: Python легко інтегрується з іншими мовами програмування та системами, що дозволяє використовувати його в багатомовних проектах. Це особливо корисно при роботі з великими наборами даних та складними алгоритмами в області машинного навчання.

5. Широке застосування в наукових дослідженнях: велика кількість наукових бібліотек і підтримка спільноти роблять Python ідеальним вибором для наукових досліджень, особливо в області статистики, біоінформатики, астрофізики та інших.
6. Популярність у розробці нейронних мереж: Python став основною мовою для розробки нейронних мереж завдяки своїй гнучкості та широкому спектру бібліотек, як TensorFlow, Keras, PyTorch. Ці інструменти дозволяють розробникам ефективно будувати, навчати та розгортати моделі глибокого навчання.

Завдяки цим перевагам, Python зайняв провідні позиції у сфері обробки даних, машинного навчання та інших високотехнологічних областях, ставши однією з найпопулярніших мов програмування сучасності.

1.5 Висновки до розділу

У розділі було підтверджено актуальність обраної теми. Розгляд сучасних підходів до прогнозування погоди виявив, що вони значною мірою базуються на моделюванні процесів високої складності, а також використовують класичні статистичні методи. Ці методи, хоч і ефективні, часто мають обмеження через складність та динамічність метеорологічних процесів. Використання штучного інтелекту в цій сфері являє собою новий та перспективний підхід, який може принести значні переваги, зокрема в точності та швидкості обробки даних. Однак цей напрямок потребує детального дослідження, щоб повністю розкрити свій потенціал та ефективно інтегрувати його у системи прогнозування погоди. Враховуючи ці аспекти, стає очевидною важливість подальшого дослідження в цій області, особливо з точки зору розробки та впровадження інноваційних методів, заснованих на

штучному інтелекті, для підвищення точності та надійності метеорологічних прогнозів.

РОЗДІЛ 2 МЕТОДИ ШТУЧНОГО ІНТЕЛЕКТУ ТА АНАЛІЗУ ДАНИХ ДЛЯ ПРОГНОЗУВАННЯ ПОГОДНИХ УМОВ. РЕКУРЕНТІ НЕЙРОННІ МЕРЕЖІ. LSTM

2.1 Огляд поширених моделей та методів аналізу даних та штучного інтелекту

Кожен з підходів, описаних нижче, пропонує унікальні можливості для різних аспектів прогнозування погоди. Вибір моделі в значній мірі залежить від конкретних характеристик проблеми прогнозування, таких як наявність даних, необхідний горизонт прогнозування та обчислювальні ресурси.

2.1.1 ARIMA

Зважаючи на те, що історичні дані про метеорологічні показники за своєю суттю являють собою часові ряди, для прогнозування таких показників можна використовувати класичну модель аналізу часових рядів на кшталт ARIMA (Авторегресійне Інтегроване Ковзне Середнє). Ця модель особливо корисна для короткострокового прогнозування. Вона може обробляти дані з тенденціями та сезонністю, включаючи введення зсувних спостережень як вхідних даних [16]. Однак ARIMA вимагає стаціонарних даних і може мати проблеми з нелінійними шаблонами.

Модель ARIMA(p, d, q) може бути математично представлена наступним чином:

$$\left(1 - \sum_{i=1}^{p} \phi_i L^i\right)(1 - L)^d X_t = \left(1 + \sum_{j=1}^{q} \theta_j L^j\right)\epsilon_t,$$

де:

- p, q, d – параметри порядків моделі;
- X_t – значення часового ряду в момент часу t;
- L – оператор лагу;
- ϕ_i – параметри моделі авторегресії (AR);
- θ_j – параметри моделі ковзного середнього (MA);
- ϵ_t – певне значення шуму.

2.1.2 Дерева рішень та випадкові ліси

Обґрунтованим підходом може бути також використання дерев рішень та випадкових лісів. Дерева рішень використовуються як для класифікації, так і для регресії. Вони працюють, розділяючи дані на гілки на основі правил прийняття рішень, що робить їх легкими для інтерпретації. Однак вони схильні до перенавчання, особливо зі складними, шумними даними.

Як ансамбль дерев рішень, випадкові ліси пом'якшують проблему перенавчання окремих дерев рішень. Вони більш надійні та точні, але менш інтерпретовані, ніж одне дерево рішень [18]. На рис. 2.1 представлену архітектуру моделі випадкового лісу.

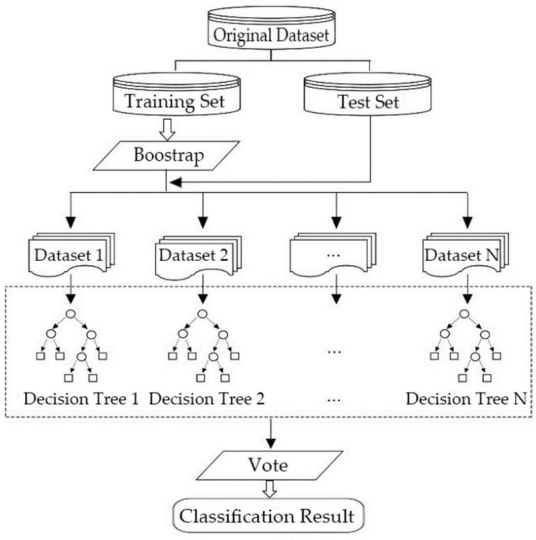

Рисунок 2.1 – Архітектура моделі випадкового лісу [19]

2.1.3 Регресійний аналіз

Іншим підходом, як зазначалось у першому розділі роботи, є використання регресійного аналізу. Цей класичний метод для кількісної оцінки зв’язку між залежною змінною та однією або набором незалежних змінних є потужним інструментом для здійснення прогнозування. Виділимо основні типи цього методу [17, с. 27-46].

1. Лінійна Регресія: використовується для прогнозування безперервної змінної, наприклад температури, на основі лінійних зв'язків між змінними. Це просто і легко інтерпретувати, але припускає лінійний зв'язок, який може не завжди бути правдивим у складних погодних системах.
2. Поліноміальна Регресія: розширює лінійну регресію, вводячи поліноміальні характеристики, що дозволяє краще підігнати в нелінійних зв'язках. Вона може моделювати більш складні шаблони, ніж лінійна регресія, але ризикує перенавчанням, якщо ступінь полінома занадто висока.

У найбільш базовій формі, лінійна регресія математично представляється формулою:

$$y = \beta_0 + \beta_1 x + \epsilon,$$

де β_0, β_1 – параметри моделі, а ϵ – певне значення шуму. При цьому параметри моделі оцінюються за наступними формулами:

$$\beta_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})\,(y_i - \bar{y})}{(x_i - \bar{x})^2},$$

$$\beta_0 = \bar{y} - \beta_1 x,$$

де $\bar{x}, \bar{y}$ – середні значення змінних x та y.

Оскільки поліноміальна регресія є розширенням лінійної, то принцип її побудови є таким же.

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \ldots + \beta_n x^n + \epsilon.$$

Для оцінки значень параметрів моделі будується матриця:

$$X = \begin{pmatrix} x_0^0 & \cdots & x_0^n \\ \vdots & \ddots & \vdots \\ x_m^0 & \cdots & x_m^n \end{pmatrix},$$

де m – кількість спостережень, а n – порядок полінома. Далі:

$$\hat{\beta} = (X^T X)^{-1} X^T \boldsymbol{y},$$

де $\hat{\beta}$ – вектор параметрів моделі, а $\boldsymbol{y}$ – вектор спостережень y.

2.1.4 Метод опорних векторів (SVM)

Метод опорних векторів (SVM, від англ. Support Vector Machine) – це тип навчального алгоритму, який використовується у машинному навчанні для класифікації та регресійного аналізу. Основна ідея SVM полягає у знаходженні гіперплощини (або набору гіперплощин) у високовимірному просторі, яка найкращим чином розділяє різні класи даних.

Метод опорних векторів широко використовується в задачах класифікації, регресії та виявлення аномалій у різних галузях, від фінансового аналізу до біоінформатики. Його популярність обумовлена високою точністю прогнозування та здатністю ефективно працювати з великими та складними наборами даних.

SVM можуть бути використані як для прогнозування категоріальних результатів (наприклад, дощ чи без дощу), так і для безперервних змінних. Вони добре працюють з невеликими або середніми наборами даних і можуть обробляти нелінійні зв'язки за допомогою ядрового трюку [3].

У випадку лінійно роздільних даних, задача класифікації за допомогою SVM зводиться до побудови двох гіперплощин, які описуються рівняннями:

$$\vec{w} \cdot \vec{x} - b = 1,$$

$$\vec{w} \cdot \vec{x} - b = -1,$$

де $\vec{w}$ – вектор вагових коефіцієнтів. З геометричної точки зору, відстань між двома цими гіперплощинами дорівнює $\frac{2}{\|w\|}$, тому оскільки суть SVM полягає у максимізації цієї відстані, то задача зводиться до мінімізації $\|w\|$.

При цьому, необхідно ввести обмеження:

$$\vec{w} \cdot \vec{x_i} - b \geq 1, \text{ якщо } \vec{x_i} \text{ належить до класу А,}$$

$$\vec{w} \cdot \vec{x_i} - b \leq -1, \text{ якщо } \vec{x_i} \text{ належить до класу Б,}$$

оскільки потрібно запобігати потраплянню точок даних до розділення.

Таким чином, з математичної точки зору задача SVM може бути сформульована наступним чином:

«Мінімізувати $\|w\|$ за умови $y_i(\vec{w} \cdot \vec{x} - b) \geq 1 \; \forall i = 1, \ldots, n$»,

де $y_i \in \{-1, 1\}$ – індикатор належності точки даних до класу А або Б.

2.1.5 Нейронні мережі та моделі глибинного навчання

Згорткові нейронні мережі (CNN, від. англ. Convolutional Neural Networks) переважно використовуються в обробці зображень. CNN можуть аналізувати супутникові та радарні зображення для ідентифікації формацій хмар, штормових шаблонів та інших метеорологічних особливостей. Вони автоматично виявляють важливі характеристики без необхідності ручного вилучення [5]. Схематичне зображення архітектури згорткової нейронної мережі зображено на рис. 2.2.

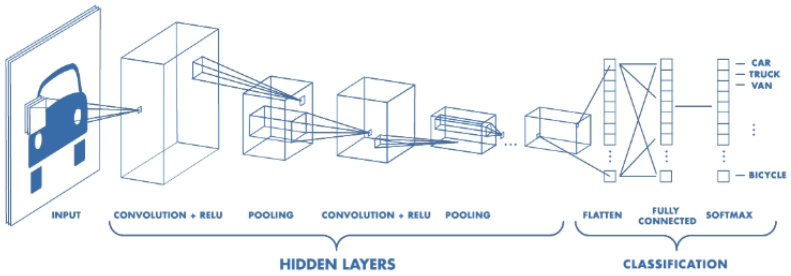

Рисунок 2.2 – Схематичне зображення архітектури CNN [20]

Повнозв’язні Нейронні Мережі (DNN, від. англ. Dense Neural Networks) складаються з багатьох шарів взаємозв'язаних вузлів і здатні моделювати складні нелінійні зв'язки. DNN можуть обробляти великі обсяги даних з різних джерел, але вимагають значних обчислювальних ресурсів і даних для навчання [6]. На рис. 2.3 зображено схематичний приклад архітектури DNN.

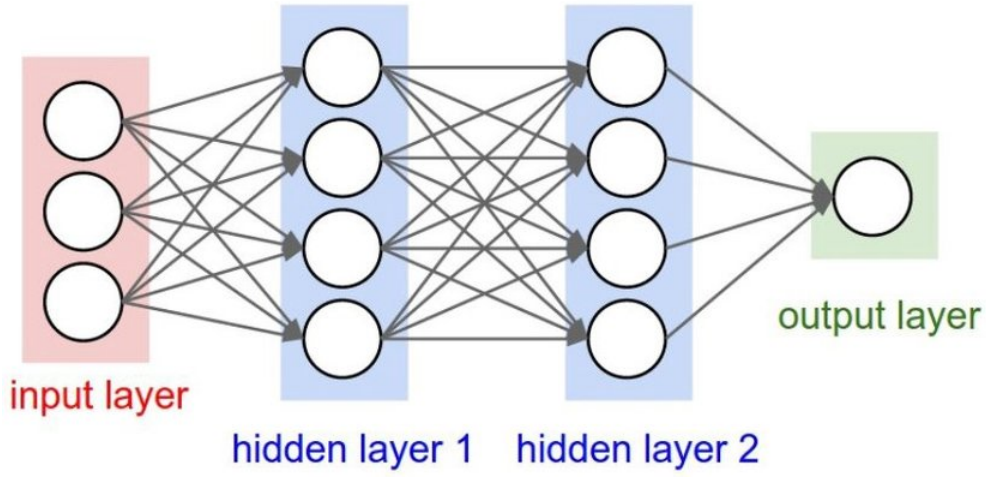

Рисунок 2.3 - Схематичний приклад архітектури DNN [21]

2.2 Рекурентні нейронні мережі

Рекурентні нейронні мережі (RNN, від англ. Recurrent Neural Networks) – це тип нейронної мережі, який особливо підходить для обробки послідовних даних. Вони відрізняються від інших нейронних мереж своєю здатністю підтримувати "пам'ять" про попередні входи у своєму внутрішньому стані, що використовується для впливу на вихід мережі. Ця особливість робить їх особливо корисними для завдань, таких як моделювання мови, розпізнавання мовлення та аналіз часових рядів.

Вкажемо ключові відмінності між RNN та іншими типами нейронних мереж.

1. Обробка послідовних даних: на відміну від прямолінійних нейронних мереж, RNN розроблені для обробки послідовностей даних. Це означає, що вони можуть брати не лише поточні дані, але й те, що вони сприймали раніше у часі.

2. Внутрішня пам'ять: RNN мають форму внутрішньої пам'яті, що дозволяє їм зберігати інформацію про те, що було оброблено до цього часу. Це критично важливо для завдань, де важливий контекст і порядок даних.

3. Спільні ваги: у RNN одні й ті ж ваги використовуються на кожному кроці часу. Це спільне використання параметрів робить RNN ефективнішими і зменшує кількість параметрів, які вони повинні вивчити.

4. Зворотне розповсюдження помилки через час (BPTT від англ. Backpropagation trough time): RNN навчаються за допомогою специфічного типу зворотного розповсюдження, що називається BPTT, який включає розгортання мережі в часі, а потім зворотнє розповсюдження помилки.

Варто порівняти найпростішу штучну нейронну мережу прямого поширення (FNN, від англ. FeedForward Neural Network) та RNN (рис. 2.4, 2.5).

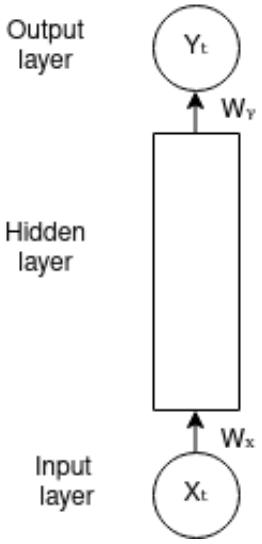

Рисунок 2.4 – Схематичне представлення архітектури FNN [22]

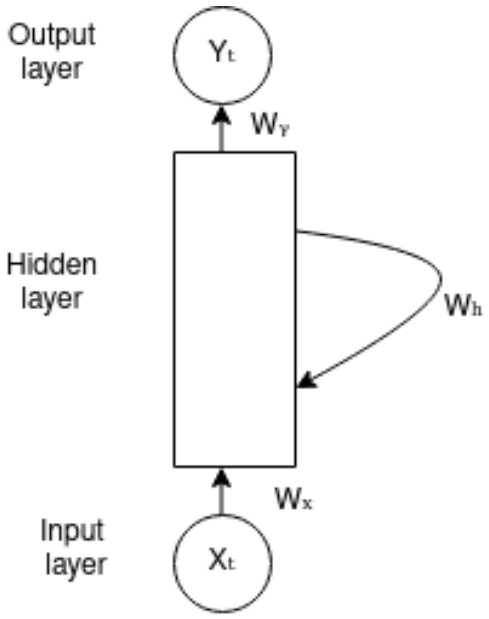

Рисунок 2.5 – Схематичне представлення архітектури RNN [22]

На схемі (рис. 2.5) відображено концепцію «внутрішньої пам'яті», що притаманна рекурентним нейронним мережам. На кожному кроці часу мережа використовує значення прихованого стану, що передається з попереднього кроку, для оновлення вихідного значення. Цей механізм дозволяє враховувати часові залежності в даних при здійсненні прогнозу [1, 8].

Для кращого розуміння принципу роботи RNN можна розглянути схему «розгорнутої» мережі (рис. 2.6)

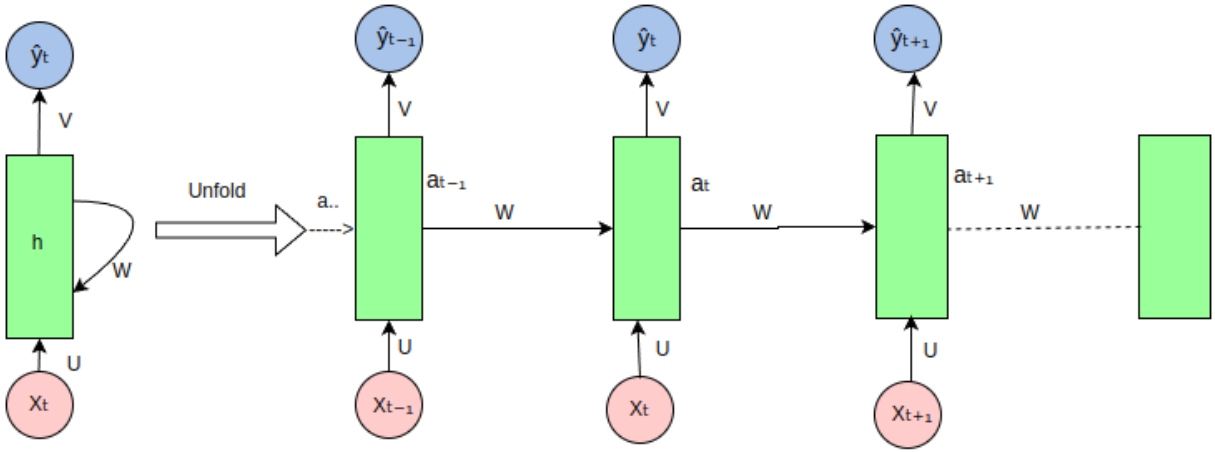

Рисунок 2.6 – Схематичне представлення «розгорнутої» архітектури RNN [22]

Мають місце наступні формули [1]:

$$\boldsymbol{a}_t = f(W_{aa}\boldsymbol{a}_{t-1} + W_{xa}\boldsymbol{x}_t + \boldsymbol{b}_a)$$

$$\boldsymbol{y}_t = W_{ay}\boldsymbol{a}_t + \boldsymbol{b}_y$$

де:

$\boldsymbol{a}_t$ – прихований стан у момент часу *t*,

$\boldsymbol{x}_t$ – вхід у момент часу *t*,

$\boldsymbol{y}_t$ – вихід у момент часу *t*,

W_{aa}, W_{xa}, W_{ay} – матриці ваг,

$\boldsymbol{b}_a, \boldsymbol{b}_y$ – вектори зсуву,

$f(\cdot)$ – функція активації.

В якості функції активації найчастіше використовується гіперболічний тангенс:

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

При цьому, можуть бути використані й інші функції активації, до прикладу сигмоїд або ReLU:

$$sigm(x) = \frac{1}{1 + e^{-x}}$$

$$ReLU(x) = max(0, x)$$

Однак, RNN мають обмеження, такі як труднощі у захопленні довгострокових залежностей у послідовностях через проблеми, такі як зникаючі та вибухові градієнти. Це призвело до розробки більш передових архітектур RNN, таких як LSTM (Long Short-Term Memory) та GRU (Gated Recurrent Units).

2.2.1 Зворотне розповсюдження помилки через час

Зворотне поширення помилки полягає у регулюванні параметрів моделі (ваг і зсувів) на основі помилки між передбаченим виводом та фактичним цільовим значенням. Метою зворотнього поширення є покращення продуктивності моделі шляхом мінімізації функції втрат. Зворотне поширення у часі є спеціальною варіацією зворотнього поширення, яке використовується для тренування RNN, де помилка розповсюджується назад у часі до початкового кроку часу $t = 1$. Зворотне поширення включає два ключові кроки: прямий прохід та зворотний прохід.

1. Прямий прохід: під час прямого проходу, RNN обробляє вхідну послідовність у часі, як описано вище, від $t = 1$ до $t = n$, де n є довжиною вхідної послідовності. Після обробки всієї послідовності, RNN генерує послідовність передбачених виводів $\hat{y} = [\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_t]$. Із цієї послідовності обраховується значення функції втрат:

$$L(y, \hat{y}) = \frac{1}{t}\sum_{t}(y_t - \hat{y}_t)^2$$

2. Зворотній прохід: цей крок включає в себе розрахунок градієнтів функції втрат і передачу цих значень у попередні ланки мережі.

2.3 LSTM

Проблема «згасаючого» та «вибухового» градієнту (рис. 2.7), що сильно звужує можливості навчання рекурентних нейронних мереж, змусила розробити модифікацію цього типу мереж, яка б дозволила уникнути подібних проблем.

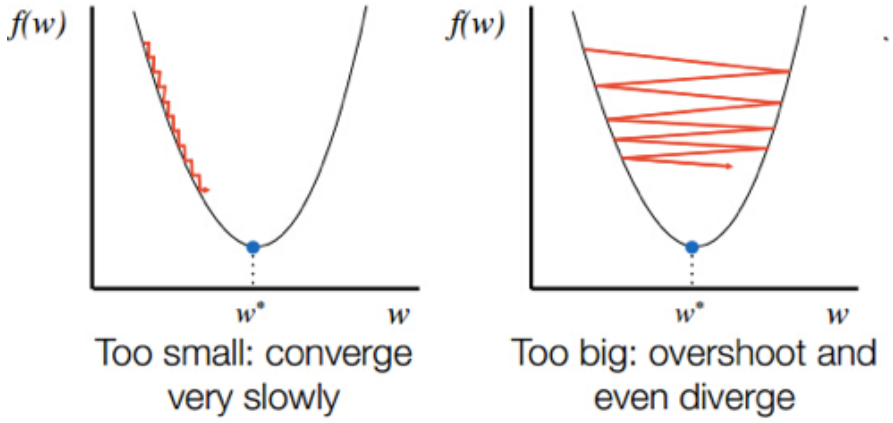

Рисунок 2.7 – Ілюстрація випадків «згасаючого» та «вибухового» градієнтів [22]

Довготривала короткочасна пам'ять (LSTM, від англ. Long short-term memory) – це особливий вид архітектури рекурентних нейронних мереж, який здатний навчатися довготривалим залежностям у даних. Вони були вперше представлені Цеппом Хохрайтером та Юргеном Шмідхубером у 1997 році [4]. LSTM ефективно вирішують широкий спектр різноманітних завдань і зараз широко використовуються.

Структура LSTM, як і в RNN, нагадує ланцюжок, але її модулі мають інший вигляд (рис. 2.8). Замість одного шару нейронної мережі, вони містять чотири шари, які взаємодіють у особливий спосіб. Ці чотири шари в LSTM включають «ворота» входу, виходу та забуття, а також шар, який регулює стан клітини [2]. Кожен з цих шарів виконує певну роль у регуляції потоку інформації всередині LSTM-модуля, що дозволяє ефективно зберігати, забувати або передавати інформацію протягом тривалого часу.

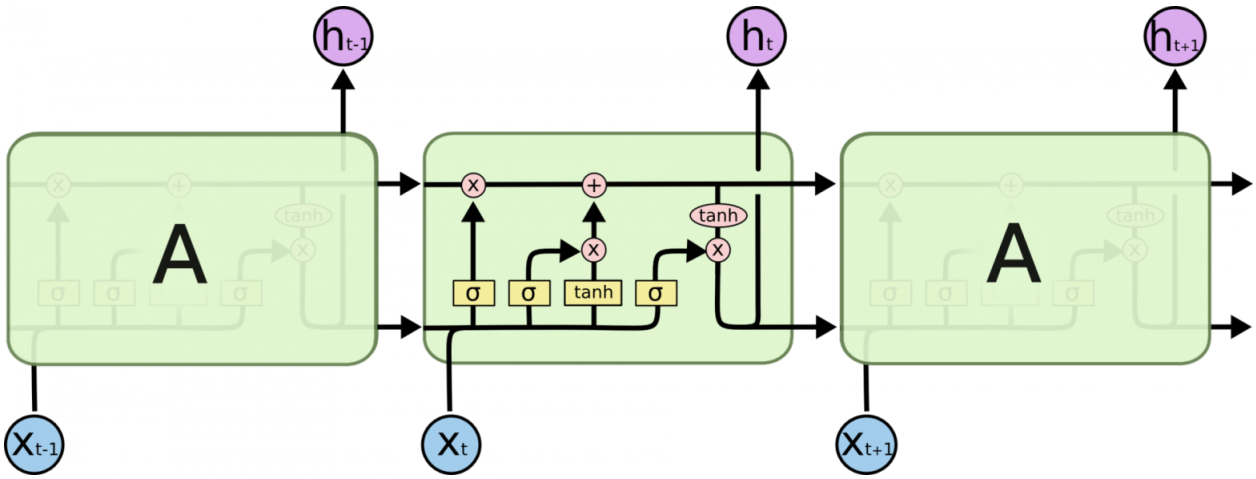

Рисунок 2.8 – Схематичне зображення архітектури LSTM [23]

Ключовим компонентом LSTM є стан клітини (cell state) – горизонтальна лінія, що пролягає вздовж верхньої частини схеми.

Цей стан проходить безпосередньо через всю ланцюгову структуру, зазнаючи лише декількох лінійних перетворень. Це дозволяє зберігати важливу інформацію на тривалі періоди часу.

LSTM може також видаляти («забувати») інформацію зі стану клітини, цей процес регулюється спеціальними фільтрами, або ж «воротами» (gates). Фільтри являють собою комбінацію сигмоїдального шару нейронної мережі та операції поточкового множення. Таким чином, кожен фільтр відкидає частину вхідної інформації, пропускаючи далі решту. Кожна клітина містить три таких фільтри [2, 13].

Перший крок у LSTM - це визначення, яку інформацію потрібно видалити зі стану клітини (рис. 2.9). Це рішення приймається за допомогою сигмоїдного шару, який називається "шаром забування" (forget gate layer) [7]. Цей шар аналізує h_{t-1} та x_t і вирішує, яке число від 0 до 1 присвоїти кожному елементу попереднього стану клітини C_{t-1}. Число 1 означає "зберегти повністю", а 0 - "повністю викинути".

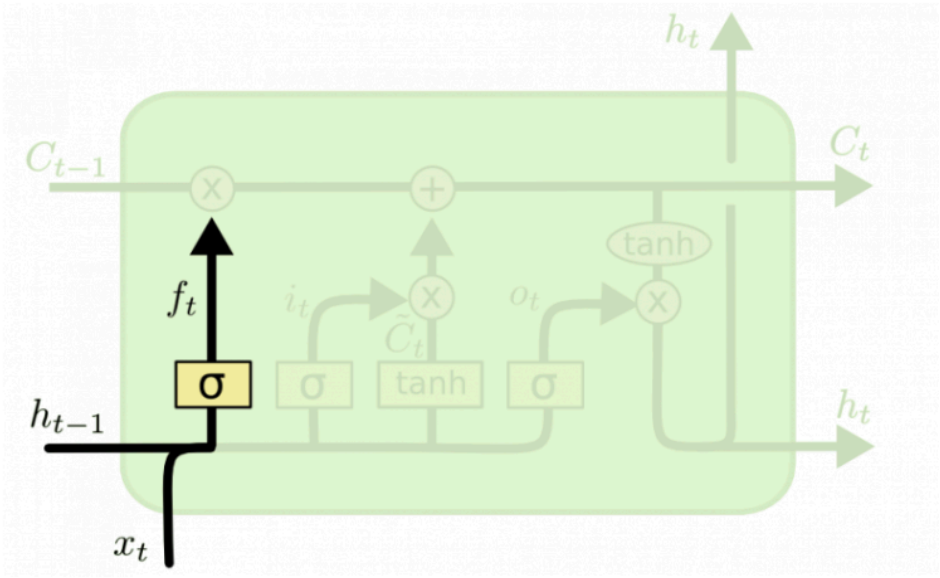

Рисунок 2.9 – Ілюстрація першого кроку LSTM [23]

Значення f_t розраховується наступним чином:

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

Далі визначається, яка нова інформація буде зберігатися у стані клітини. Цей етап складається з двох частин. По-перше, сигмоїдний шар, відомий як "шар вхідного фільтра" (input layer gate) [7], вирішує, які значення потрібно оновити. Потім шар *tanh* формує вектор нових потенційних значень $\tilde{C}_t$, які можуть бути додані до стану клітини (рис. 2.10).

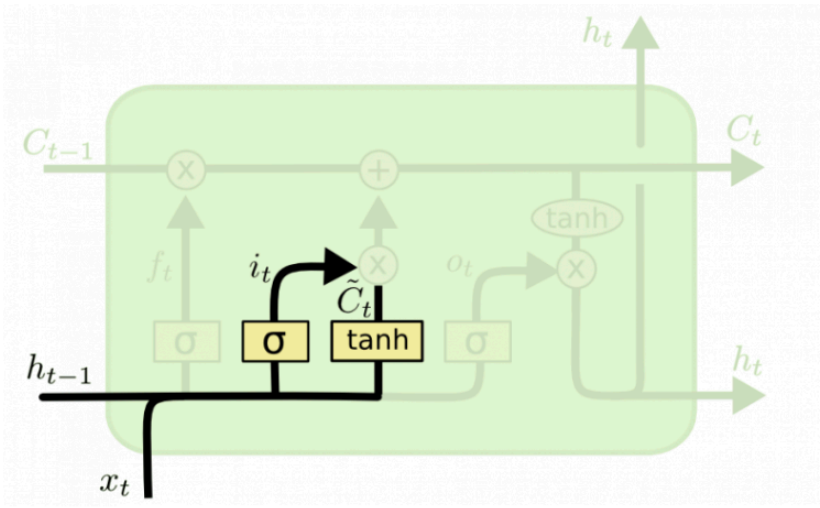

Рисунок 2.10 – Ілюстрація другого кроку LSTM [23]

Значення i_t та $\tilde{C}_t$ розраховується наступним чином:

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

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

Далі розраховується нове значення стану клітини. Це відбувається за рахунок «забування» з першого кроку, тобто видаленням частити інформації зі старого значення стану клітини, а також оновленням стану клітини новою інформацією з другого кроку (рис. 2.11).

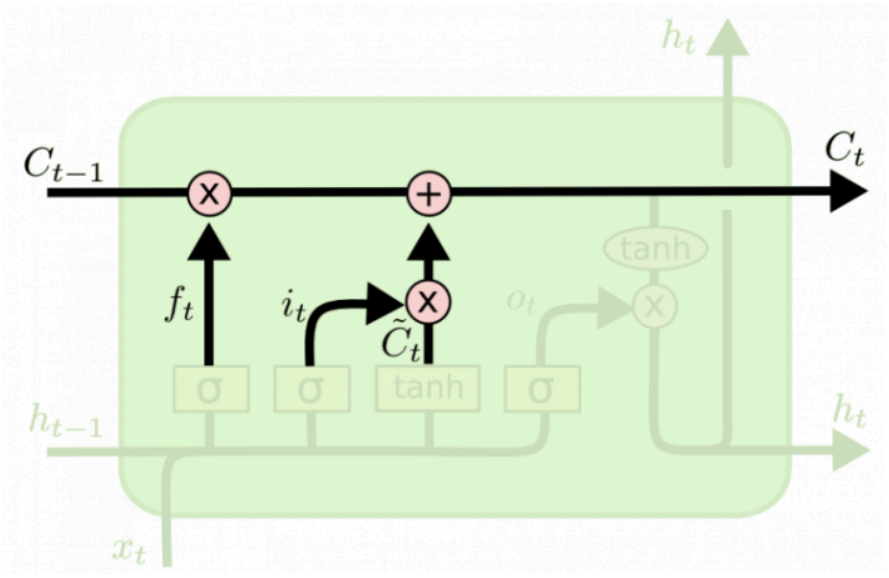

Рисунок 2.11 – Ілюстрація третього кроку LSTM [23]

Значення C_t розраховується наступним чином:

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

У кінці LSTM визначає, яка інформація буде виході. Вихідні дані базуються на стані клітини, до якого застосовуються певні фільтри. Спочатку застосовується сигмоїдний шар, який визначає, яку інформацію зі стану клітини ми буде виводитись. Потім значення стану клітини проходять через шар *tanh*, щоб на виході отримати значення в діапазоні від -1 до 1, і ці значення множаться на вихідні значення сигмоїдного шару, що дозволяє виводити лише потрібну інформацію (рис. 2.12).

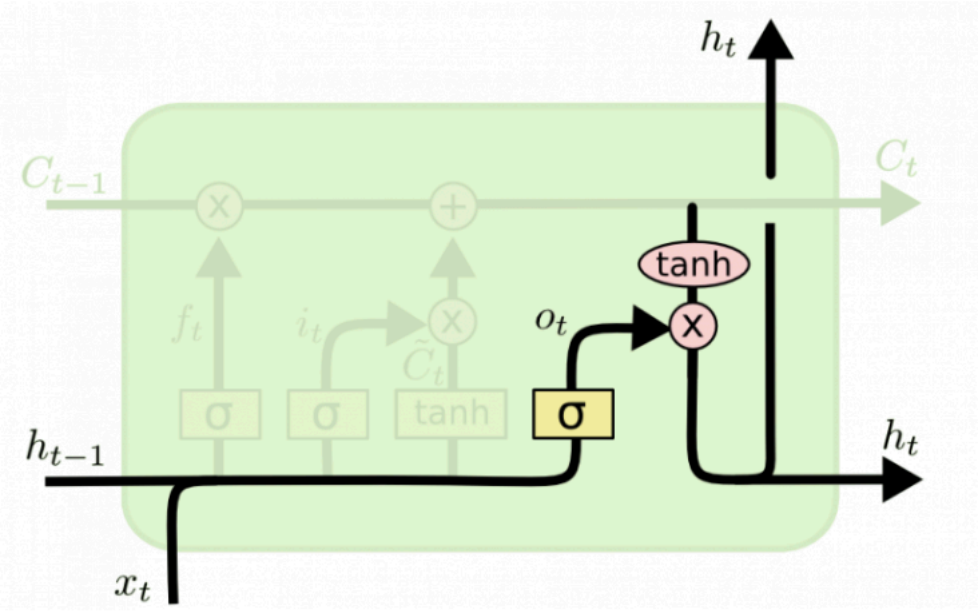

Рисунок 2.12 – Ілюстрація четвертого кроку LSTM [23]

Значення o_t, h_t розраховується наступним чином:

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

$$h_t = o_t * \tanh(C_t)$$

2.4 Висновки до розділу

Рекурентні нейронні мережі є одним з найпотужніших інструментів для аналізу та прогнозування даних у часових рядах. Ці мережі ефективно використовуються в різноманітних сферах, від фінансового моделювання до прогнозування погодних умов, завдяки їх здатності до обробки послідовностей даних та зберігання інформації про попередні спостереження. Така особливість робить їх підходящими для роботи з часовими рядами, де важливим є врахування історичних даних для розуміння поточних та прогнозування майбутніх тенденцій.

Однак, RNN мають певні обмеження, зокрема, проблему зникнення або вибуху градієнтів під час навчання, що ускладнює збереження інформації на тривалі періоди. Ця проблема особливо виражена у задачах, де необхідно враховувати довготривалі залежності між даними.

LSTM – це більш потужна модифікація RNN, яка була спеціально розроблена для вирішення цієї проблеми. Структура LSTM дозволяє ефективніше управляти потоком інформації всередині мережі. Завдяки цьому LSTM може зберігати важливу інформацію на тривалі періоди часу та ефективніше працювати з довготривалими залежностями. Ця особливість робить LSTM підходящою для складних завдань прогнозування в часових рядах, де важлива здатність до аналізу даних, які розтягуються на довгі періоди часу [10].

РОЗДІЛ 3 СИСТЕМА ПРОГНОЗУВАННЯ МЕТЕОРОЛОГІЧНИХ УМОВ НА ОСНОВІ МЕТОДІВ АНАЛІЗУ ДАНИХ ТА ШТУЧНОГО ІНТЕЛЕКТУ

Метою даного розділу є розробка системи прогнозування погодних умов в Києві на основі історичних метеорологічних даних за допомогою методів аналізу даних та штучного інтелекту, а саме моделі машинного навчання LSTM.

Даний розділ містить інформацію про використаний дата сет, а також огляд результатів роботи системи на тестовому наборі даних.

3.1 Огляд дата сету

Дослідницькі проекти в сфері кліматології та екології часто базуються на історичних даних про погодні умови. Отримання достовірної та повної інформації є ключовим для глибокого розуміння змін клімату та їх впливу на різні аспекти навколишнього середовища. В рамках даного огляду ми розглядаємо датасет, що представляє докладну інформацію про погодні умови в Києві, столиці України, з 1 січня 1881 року.

Джерелом даних є російська федеральна гідрометеорологічна служба. Незважаючи на різноманіття джерел метеорологічної інформації, що доступні сьогодні, такий обширний історичний масив інформації може надавати в основному лише ця служба. Це зумовлено історичними, організаційними та науковими особливостями збору та зберігання даних.

Структура датасету чітко організована та містить наступні ключові колонки.

1. Year: рік замірів. Ця колонка дозволяє аналізувати довготривалі тренди та сезонність.
2. Month: місяць, що забезпечує можливість аналізу місячних змін у кліматі.
3. Day: конкретний день місяця.
4. Hour: година доби.
5. t_min_c: мінімальна температура повітря визначена для кожного дня. Це важлива інформація для вивчення екстремальних погодних явищ, таких як морози чи холодні періоди.
6. t_mean_c: середнє температурне значення дня. Ця метрика часто використовується як базовий показник для аналізу температурних трендів.
7. t_max_c: максимальне температурне значення дня, що дозволяє вивчати періоди спеки або несподівано теплих днів.
8. precipitation_mm: кількість опадів, виміряна в міліметрах. Основний показник для дослідження опадів та їх змін в контексті кліматичних змін.

Цей датасет є цінним ресурсом для науковців, екологів та інших спеціалістів, що працюють у сфері вивчення кліматичних змін. Він надає можливість досліджувати довготривалі температурні та опадові тренди, виявляти аномалії та розуміти їх можливі причини та наслідки.

3.2 Результати тренування моделі та роботи на тестовій вибірці

Було проведено тренування та тестування моделі для отримання 48-годинного прогнозу на основі даних про 18 попередніх діб.

Для оцінки якості використовуються метрики Mean Absolute Error (MAE), Mean Squared Error (MSE), і Mean Absolute Percentage Error (MAPE) -

це метрики, які використовуються для оцінки точності прогнозування або моделювання в наукових дослідженнях і аналізі даних.

Mean Absolute Error (MAE): MAE є однією з найпростіших метрик для вимірювання рівня точності моделі. Ця метрика вимірює середню абсолютну різницю між спостережуваними і прогнозованими значеннями. MAE обчислюється наступним чином:

$$MAE = \frac{1}{n}\sum |y - \hat{y}|,$$

де:

- n - кількість спостережень,
- y - спостережуване значення,
- ŷ - прогнозоване значення.

MAE вимірюється в тих же одиницях, що і вихідні дані, і чим менше MAE, тим точніше модель.

MSE є іншою метрикою для оцінки точності моделі, і вона вимірює середню квадратичну різницю між спостережуваними і прогнозованими значеннями. MSE обчислюється так:

$$MSE = \frac{1}{n}\sum (y - \hat{y})^2$$

MSE також вимірюється в квадратних одиницях вихідних даних, і він більше чутливий до великих відхилень між спостережуваними і прогнозованими значеннями.

MAPE є метрикою, яка вимірює відсоткову середню абсолютну похибку між спостережуваними і прогнозованими значеннями. MAPE обчислюється так:

$$MAPE = \frac{1}{n}\sum \left|\frac{y - \hat{y}}{y}\right| * 100$$

MAPE виражає похибку моделі у відсотках і дозволяє порівнювати точність прогнозів для різних змінних та масштабів даних.

Нижче наведено графіки передбачень моделі для показника середньої температури на тренувальній та тестовій вибірках, а також значення метрик для кожного випадку.

На рисунку 3.1 зображено порівняння графіків середньої температури для тренувальної вибірки (80% дата сету).

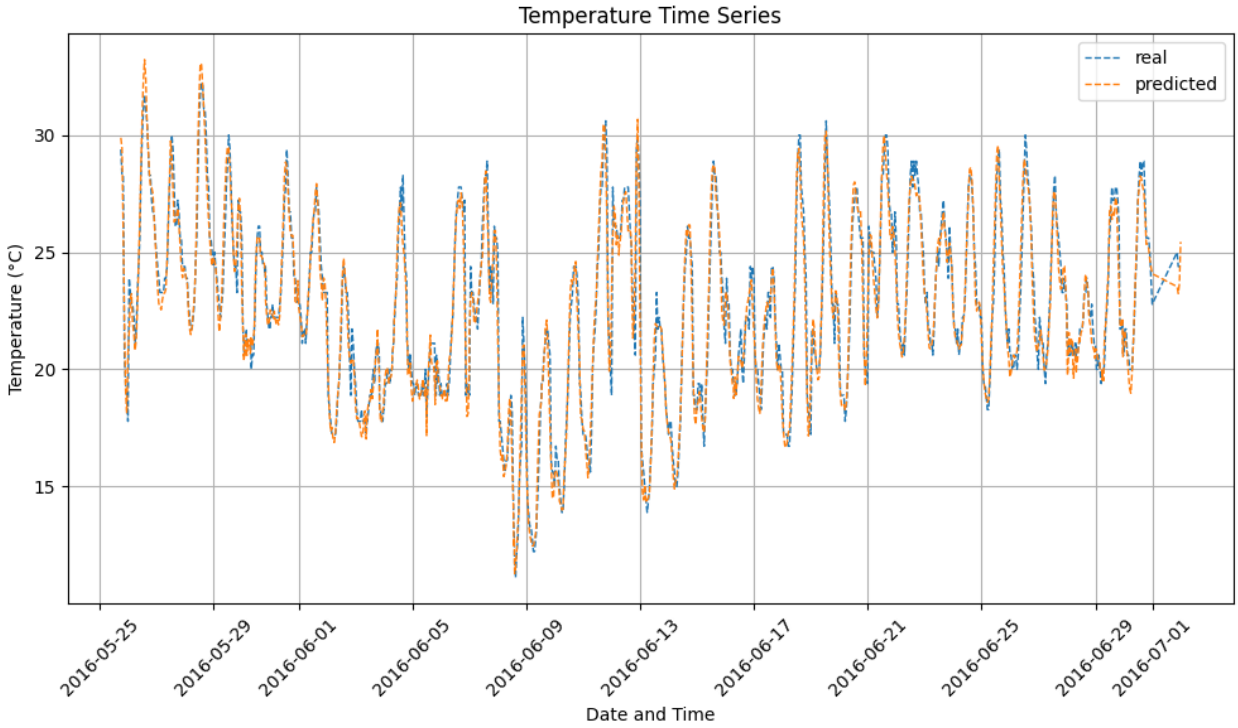

Рисунок 3.1 Порівняння графіків середньої температури для тренувальної вибірки

Показники метрик:

- MAE (Середня абсолютна похибка): 0.53,
- MSE (Середня квадратична похибка) : 0.60,
- MAPE (Середня абсолютна похибка у відсотках): 2,9 %.

На рисунку 3.2 зображено порівняння графіків середньої температури для тестової вибірки (20% дата сету).

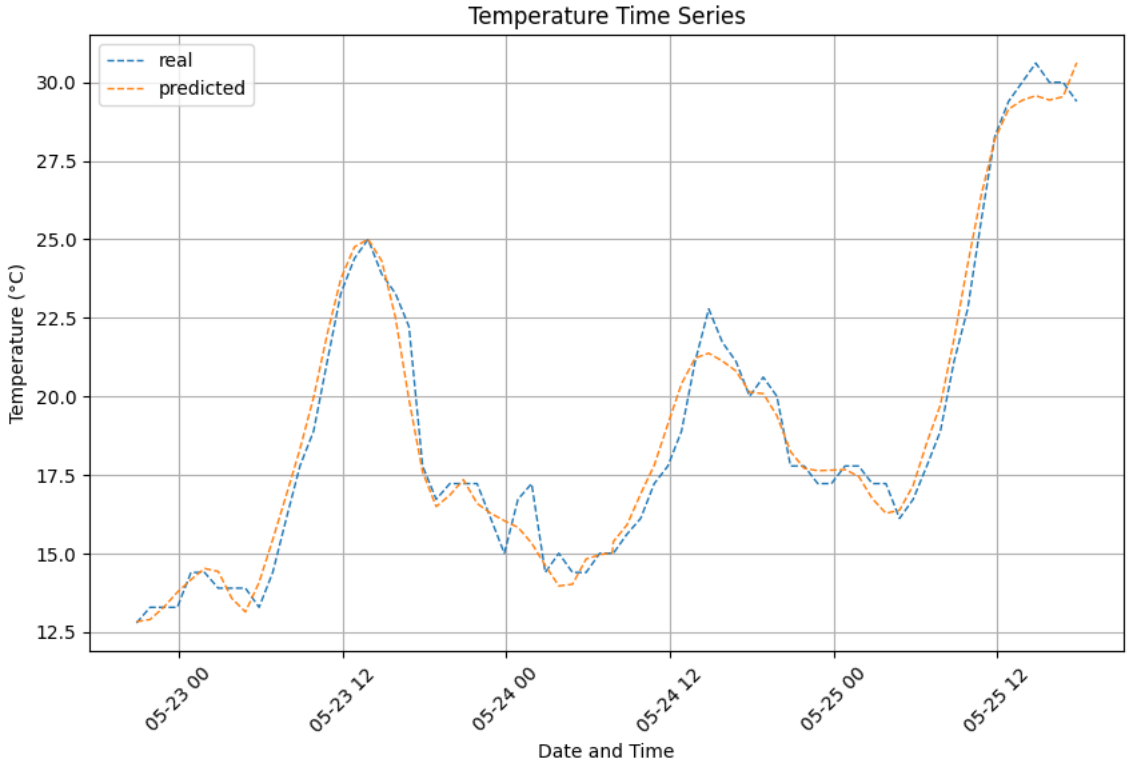

Рисунок 3.2 Порівняння графіків середньої температури для тестової вибірки

Показники метрик:

- MAE (Середня абсолютна похибка): 0.59,
- MSE (Середня квадратична похибка) : 0.82,
- MAPE (Середня абсолютна похибка у відсотках): 2,5 %.

Остаточні результати експериментів наведено у табл. 3.1.

Таблиця 3.1 – Результати проведених експериментів

Вибірка	MAE	MSE	MAPE
Тренувальна	0.53	0.60	2.9%
Тестова	0.59	0.82	2.5%

3.3 Розробка системи прогнозування

В сучасному цифровому світі, де інтеграція між системами і взаємодія сервісів є ключовими для досягнення максимальної ефективності та гнучкості,

технологія REST API відіграє важливу роль. API (Application Programming Interface) є мостом між різними програмними системами, дозволяючи їм спілкуватися та обмінюватися даними. REST (Representational State Transfer) є популярним стандартом для створення API, який використовує HTTP запити для отримання, відправки та видалення даних. У контексті метеорології та прогнозування погоди, використання архітектури LSTM (Long Short-Term Memory) для аналізу та прогнозування часових рядів може надати значущі переваги завдяки високій точності й здатності розпізнавати складні шаблони в даних. Поєднуючи потужність LSTM із можливостями REST API, в даній роботі отримуємо систему, яка може динамічно аналізувати, прогнозувати та надавати інформацію про погоду через веб-запити, що робить її доступною для різних додатків та платформ.

3.3.1 REST API

REST API – це визначений набір правил та протоколів, що дозволяє різним програмним системам комунікувати між собою через Інтернет. API (Application Programming Interface) відіграє роль "мосту" між програмами, дозволяючи одній системі отримувати доступ до функцій або даних іншої системи. REST (Representational State Transfer) є архітектурним стилем, що описує як такі взаємодії мають відбуватися.

Серцевина REST базується на багатьох принципах.

1. Безстановість (Statelessness): кожний запит від клієнта до сервера має містити всю необхідну інформацію для його обробки. Сервер не має зберігати контекст клієнта між запитами.
2. Клієнт-серверна архітектура: взаємодія відбувається між клієнтом (який здійснює запит) та сервером (який обробляє запит і надсилає відповідь).

3. Безпосередня співставимість (Cacheability): відповіді сервера можуть бути кешовані на клієнтській стороні для оптимізації роботи.
4. Єдиний інтерфейс: система повинна мати єдиний, зрозумілий і стійкий інтерфейс, що спрощує взаємодію між компонентами.

Основний механізм взаємодії в REST API – це HTTP запит. HTTP (Hypertext Transfer Protocol) є основним протоколом передачі даних в Інтернеті. Коли ми говоримо про HTTP запити в контексті REST, маємо на увазі наступні методи:

- GET: отримання інформації з сервера,
- POST: відправлення нових даних на сервер,
- PUT: оновлення існуючих даних на сервері,
- DELETE: видалення даних з сервера.

Кожний з цих запитів взаємодіє з ресурсами, які ідентифікуються за допомогою URL (Uniform Resource Locator). Ресурс може бути будь-якою сутністю, наприклад, користувачем, товаром або інформацією про погоду.

Відповідь на HTTP запит містить статус (успішний, помилковий тощо) та, зазвичай, дані у вигляді JSON (JavaScript Object Notation) або XML (eXtensible Markup Language), які легко обробляються програмами.

JSON (JavaScript Object Notation) є легковісним форматом обміну даними, широко використовуваним для передачі даних між клієнтом та сервером. Він був розроблений на основі підмножини мови програмування JavaScript, але в даний час підтримується більшістю сучасних мов програмування.

Вкажемо основні характеристики JSON.

1. Легкість читання для людей: JSON є текстовим форматом, який легко читається і розуміється людьми.
2. Легкість генерації та обробки для машин: більшість мов програмування мають вбудовані функції для генерації та обробки JSON.

JSON даними можуть бути:

- об'єкти: набори пар "ключ-значення", заключені в фігурні дужки «{ }»;

- масиви: списки значень, заключені в квадратні дужки «[ ]».

На рисунку 3.3 зображено приклад JSON.

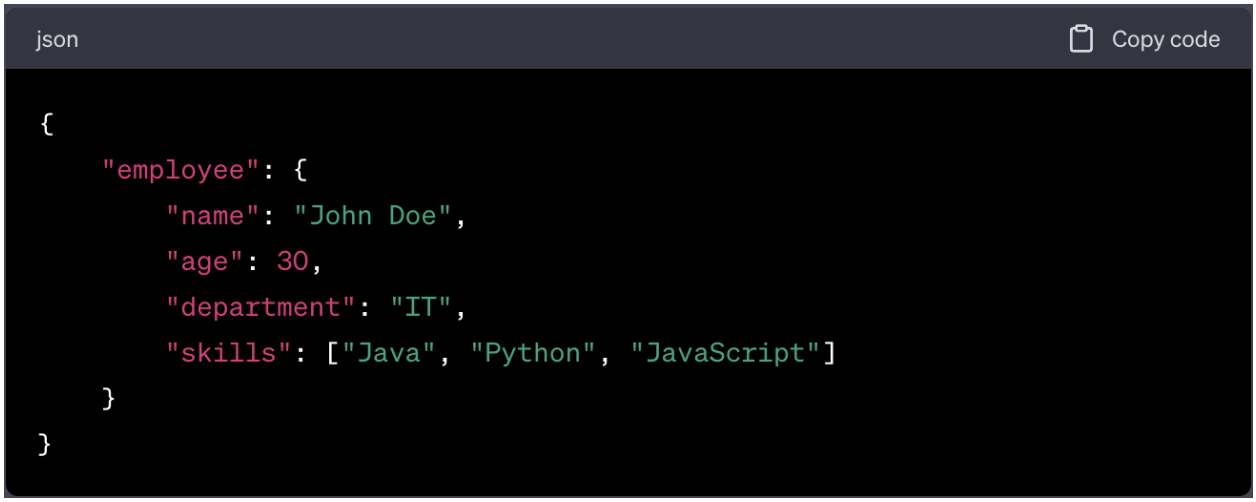

Рисунок 3.3 Приклад JSON

У цьому прикладі міститься наступна інформація:

- «employee» є об'єктом з чотирма ключами: «name», «age», «department» та «skills».
- Значення «skills» є масивом, що містить три рядкові значення.

JSON є особливо корисним для серіалізації та десеріалізації структурованих даних, що робить його відмінним вибором для використання в REST API та інших веб-застосунках. Його простота, гнучкість та широка підтримка роблять його незамінним інструментом в арсеналі сучасного розробника.

Вкажемо основні переваги REST API.

1. Незалежність платформи: REST є незалежним від мови програмування або платформи, що дозволяє системам, створеним на різних технологіях, взаємодіяти між собою.
2. Масштабованість: завдяки безстановій архітектурі, системи, засновані на REST, легко масштабуються.
3. Прозорість та інтуїтивність: інтерфейси REST API часто є легкими для розуміння та використання.

Підсумовуючи, REST API – це сучасний, гнучкий та ефективний спосіб реалізації взаємодії між різними програмними системами в Інтернеті, що

дозволяє гармонійно інтегрувати сервіси, обмінюватися даними та розширювати можливості додатків.

Нижче наведено приклад вхідних даних для системи (рис. 3.4) та результату роботи системи (рис. 3.5).

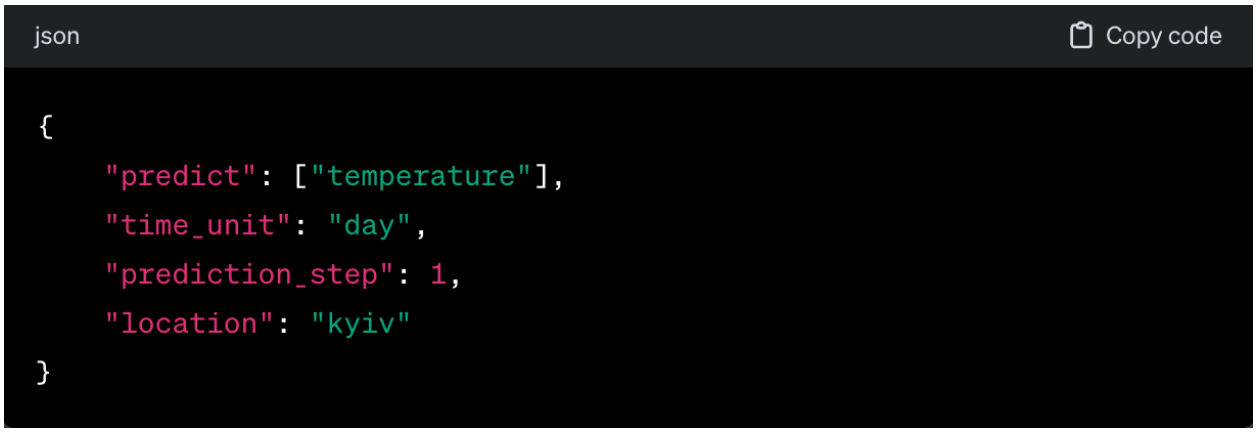

Рисунок 3.4 – Приклад вхідних даних для системи

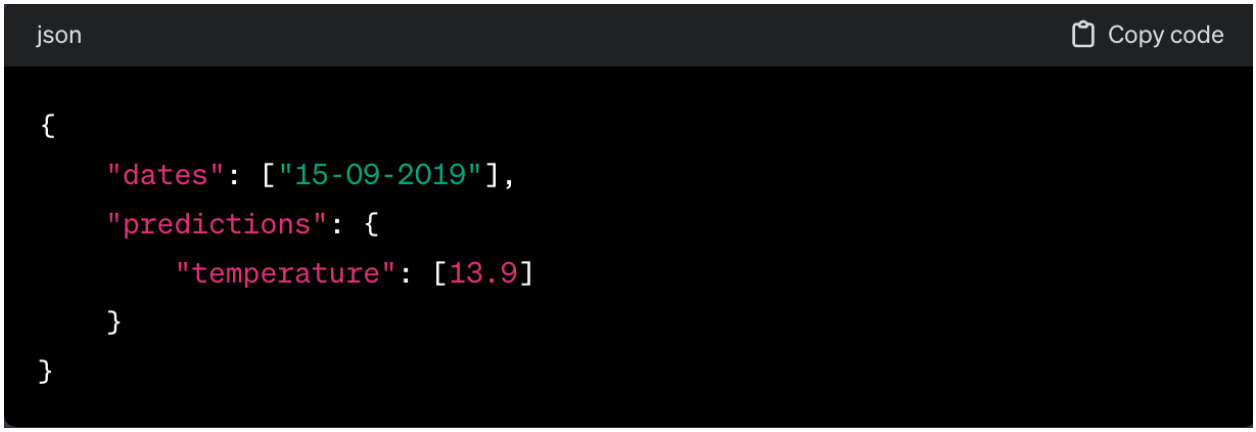

Рисунок 3.5 – Приклад результату роботи системи

3.3.2 Архітектура системи прогнозування погоди на основі моделі LSTM із використанням технології REST API

Сучасні системи прогнозування погоди вимагають ефективної інтеграції моделей машинного навчання, джерел даних та клієнтських запитів. Розглянемо архітектуру системи, яка використовує модель LSTM для

прогнозування погодних умов, інтегровану через REST API для обміну даними.

На рисунку 3.6 зображено архітектуру розробленої системи.

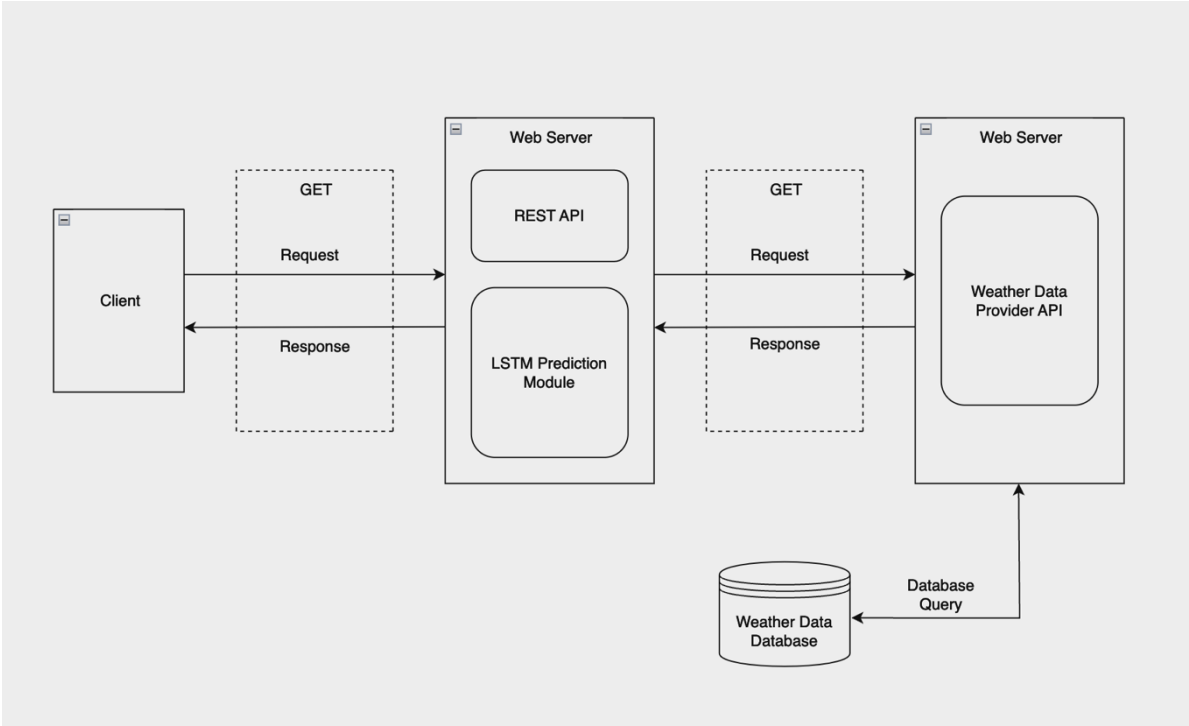

Рисунок 3.6 Архітектура системи

Вкажемо принцип її роботи.

1. Ініціація процесу клієнтом: коли користувач бажає отримати прогноз погоди, він робить GET запит до сервера через REST API.
2. Серверний запит до постачальника даних: після отримання запиту від клієнта, сервер, який має інтегровану LSTM модель, відправляє власний GET запит до іншого сервера — постачальника історичних метеорологічних даних. Цей сервер, у свою чергу, здійснює доступ до бази даних з метеорологічною інформацією.
3. Отримання історичних даних: постачальник даних виконує запит до своєї бази даних, щоб зібрати метеорологічні показники за останні 18 днів. Ці дані після отримання конвертуються у формат JSON для оптимізованої передачі даних.

4. Повернення даних до первинного сервера: історичні дані, отримані від постачальника, передаються назад до первинного сервера через REST API у форматі JSON. Цей сервер, вже маючи необхідні дані, приступає до обробки цих даних за допомогою LSTM моделі.
5. Прогнозування з допомогою LSTM: первинний сервер використовує LSTM модель для аналізу наданих історичних даних. LSTM, завдяки своїй архітектурі, здатна розпізнавати та враховувати довгострокові залежності в часових рядах, що робить її ідеальною для прогнозування погоди на основі історичних даних. Після обробки інформації, модель генерує прогноз погоди на наступні 3 дні.
6. Відправка прогнозу клієнту: прогнозовані дані конвертуються у формат JSON і, за допомогою REST API, відправляються назад клієнту, який ініціював первинний запит.

Отже, запропонована архітектура системи забезпечує ефективний обмін даними між клієнтом, сервером прогнозування та постачальником історичних метеорологічних даних. Використання LSTM моделі в комбінації із технологією REST API створює гнучку, масштабовану та високоефективну систему прогнозування погоди.

3.4 Висновки до розділу

Розглядаючи реалізацію системи прогнозування метеорологічних умов у Києві на основі REST API та LSTM, можна відзначити ключову роль комбінації цих технологій. Інтеграція REST API забезпечує гнучкість та масштабованість системи, дозволяючи ефективно обмінюватися даними між її компонентами. Ця гнучкість стає особливо актуальною, оскільки система

автоматично звертається до постачальника метеорологічної інформації через його власний API для отримання актуальних даних.

LSTM, в свою чергу, демонструє високу ефективність при аналізі часових рядів, зокрема в метеорологічному прогнозуванні. Ця модель враховує довгострокові залежності в даних, що підтверджується точністю прогнозів системи щодо погодних умов в Києві.

Архітектура системи розроблена таким чином, що вона легко адаптується до різних джерел даних, дозволяючи масштабувати систему не лише для Києва, а й для інших регіонів. Крім того, завдяки її відкритості для інновацій, можливе впровадження нових методів аналізу даних або алгоритмів штучного інтелекту в майбутньому, що робить систему довгостроково стійкою до технологічних змін.

Таким чином, розроблена система є вдалим прикладом ефективного застосування сучасних технологій для рішення конкретної практичної задачі прогнозування погоди.

РОЗДІЛ 4 РОЗРОБКА ВЛАСНОГО СТАРТАП ПРОЕКТУ

4.1 Опис ідеї проекту

Проект, який пропонується, передбачає створення і впровадження системи прогнозування погодних умов на основі історичних метеорологічних даних та використання глибоких нейронних мереж, зокрема Long Short-Term Memory (LSTM) моделі. Ця система буде використовувати REST API архітектуру для забезпечення доступу до прогнозів погоди для користувачів.

Основна ідея цього проекту полягає в створенні точного та надійного інструменту прогнозування погодних умов на майбутнє, що дозволить користувачам отримувати актуальну інформацію про погоду в будь-якому регіоні світу. Модель LSTM виявилася дуже ефективною у роботі з послідовними даними, такими, як дані про погоду, завдяки її здатності враховувати довгострокові залежності і тенденції в даних.

Система буде використовувати історичні дані метеорологічних спостережень для навчання LSTM моделі. Це дозволить моделі розробити глибокі зв'язки між різними метеорологічними параметрами і враховувати сезонність та інші фактори, що впливають на погоду. Після навчання моделі буде створено REST API, яке буде надавати доступ до прогнозів погоди через інтернет.

Основні переваги цього проекту включають точність прогнозів завдяки використанню глибокого навчання, можливість масштабування системи для обробки великої кількості запитів в реальному часі і постійне вдосконалення точності прогнозів завдяки використанню нових даних для періодичного перенавчання моделі. Крім того, система буде здатна підтримувати різні формати видачі прогнозів, такі як текстові описи, графіки та API для розробників додатків.

Проект також передбачає можливість розширення функціональності, додавання додаткових метеорологічних параметрів і створення додаткових модулів для врахування специфічних потреб користувачів, таких як сільські господарства, транспортні компанії, туристичні підприємства і інші. Це дозволить системі стати важливим інструментом для прийняття рішень у різних галузях, які залежать від точних прогнозів погоди.

4.2 План розробки стартап проекту

План розробки та впровадження стартапу та його виходу на ринок може бути структурованим наступним чином.

1. Маркетинговий аналіз:
   - проведення конкурентного аналізу для вивчення методів, якими вже користуються люди для вирішення схожих проблем;
   - формування концепції проекту та визначення цільової аудиторії;
   - розробка стратегії виходу на ринок, базованої на аналізі ринкового середовища.
2. Організація стартапу:
   - складання детального плану та розробка таймлайну розробки та запуску продукту;
   - планування обсягу виробництва та оцінка потреби в ресурсах для виконання плану;
   - розрахунок фінансових витрат, необхідних для втілення проекту, та витрат на запуск.
3. фінансово-економічний аналіз та ризики:
   - визначення обсягу інвестиційних втрат;
   - розрахунок ключових фінансово-економічних показників проекту, включаючи собівартість, ціну продукту/послуги, податковий збір та чистий прибуток;

- визначення показників інвестиційної привабливості проекту, таких як рентабельність продажів та період окупності проекту;
- виділення основних ризиків проекту та визначення методів їх управління та запобігання;

4. Розробка заходів з комерціалізації продукту:

- проведення дослідження для виявлення інтересів потенційних інвесторів та бізнес-партнерів;
- складання інвестиційної пропозиції, включаючи опис продукту, його поточні характеристики та можливості розширення та розвитку;
- вибір каналів комунікації для взаємодії з потенційними зацікавленими сторонами.

4.3 Дослідження ринкових можливостей

Для ефективного планування введення проекту на ринок, необхідно провести аналіз ринкових можливостей та загроз. Цей аналіз дозволить ідентифікувати потенційні напрями розвитку проекту та врахувати фактори, які можуть вплинути на його успішність.

Розпочнемо з аналізу попиту на продукт або послугу, яку надає стартап. Проведемо оцінку наявності попиту на ринку, його обсягу та динаміки розвитку (табл. 4.1). Основна мета полягає в тому, щоб визначити, чи існує попит на конкретну продукцію чи послугу, яку маємо намір надавати.

Таблиця 4.1 Оцінка параметрів стану ринку

№	Параметр стану ринку	Характеристика
1	Кількість головних гравців, од	5
2	Загальний обсяг продаж, грн/ум.од	2000
3	Динаміка ринку (якісна оцінка)	Зростає

Продовження таблиці 4.1

4	Наявність обмежень для входу (вказати характер обмежень)	Відсутні
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні
6	Середня норма рентабельності в галузі (або по ринку), %	15%

Рентабельність - це фінансовий показник, який вимірює прибутковість і ефективність діяльності підприємства або проекту. Вона вказує на те, як ефективно компанія використовує свої ресурси для отримання прибутку.

Рентабельність може бути виражена у відсотках і розраховується як відношення чистого прибутку (прибутку після відрахування всіх витрат і податків) до загальних витрат чи витрат на виробництво. Зазвичай цей показник розраховується на річній основі.

Висока рентабельність свідчить про те, що компанія ефективно використовує свої ресурси і генерує прибуток. Низька рентабельність може свідчити про проблеми в управлінні, високі витрати або недостатню генерацію прибутку.

Необхідно провести аналіз цільової аудиторії проекту, розглянути фактори, які впливають на привабливість кінцевого продукту для цільової аудиторії.

Проведення аналізу цільової аудиторії проекту є критично важливим етапом в розробці та впровадженні будь-якого продукту чи послуги. Цей аналіз допомагає глибше зрозуміти характеристики і потреби цільової групи, що в свою чергу сприяє створенню більш привабливого та ефективного кінцевого продукту.

В таблиці 4.2 наведено характеристику цільової аудиторії, аналіз потреб, особливостей поведінки потенційних клієнтів, а також фактори, що впливають на вибір.

Таблиця 4.2 Характеристика потенційної цільової аудиторії

Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
Необхідний сервіс, який дозволить отримувати точні прогнози погодних умов як короткотривалі, так і більш віддалені в часі	Середні та великі сільськогосподарські підприємства, індивідуальні користувачі	Вимоги до часових рамок прогнозу, кількість затребуваних показників, які прогнозуються	Точність прогнозу, можливість отримувати прогноз великої кількості різних метеорологічних показників у різні часові проміжки

Тепер необхідно провести аналіз ринкового середовища, розглянути фактори, що сприяють ринковому впровадженню проекту, а також тих, які йому перешкоджають.

Ринкові загрози - це потенційні фактори або обставини, які можуть негативно вплинути на бізнес чи проект на ринку і призвести до втрат, зниження прибутку або загрози його стійкості і конкурентоспроможності. Ці загрози можуть виникнути внаслідок зовнішніх чи внутрішніх чинників та навколишнього середовища.

Нижче, в таблиці 4.3 наведено аналіз можливих ринкових загроз для впровадження проекту, а в таблиці 4.4 – аналіз ринкових можливостей.

Таблиця 4.3 Фактори ринкових загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Конкуренція	Ринок насичений постачальниками метеорологічної інформації	Акцент в маркетингу на сильних сторонах власного продукту, таких як інноваційність, можливість отримання більшої кількості різноманітних показників, ніж пропонують конкуренти
2	Висока вартість користування сервісом для індивідуальних користувачів	За наявності безкоштовних альтернатив, можуть виникнути труднощі з переконанням потенційних користувачів у необхідності оформлення платної підписки	А) Створення безкоштовної версії для індивідуальних користувачів із отриманням вигоди через розміщення реклами Б) Акцент на роботі з підприємствами, які будуть готові платити за послуги, щоб розкрити потенціал продукту

Таблиця 4.4 Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Гнучка архітектура	Проста, та водночас гнучка архітектура системи відкриває широкий спектр можливостей постійного вдосконалення та масштабування продукту	Постійна та безперервна робота над вдосконаленням архітектури та моделі, інвестиції в розробку та дослідження задля розкриття повного потенціалу системи
2	Широкий спектр потреб потенційних клієнтів	Інформація про прогноз різноманітних метеорологічних показників необхідна різноплановим представникам середнього та великого бізнесу	Розширення різноманіття надаваної інформації задля ширшого охоплення різних сегментів ринку
3	Популярність теми штучного інтелекту	Нейронні мережі та штучний інтелект перебувають на піку популярності та легко викликають зацікавлення в учасників ринку	Посилена експлуатація теми нейронних мереж та штучного інтелекту в стратегії маркетингу продукту

Модель «5 сил Портера» (Porter's Five Forces Model) - це аналітичний інструмент, розроблений відомим американським стратегічним консультантом Майклом Портером в 1979 році. Ця модель використовується для аналізу конкурентної ситуації в галузях та на ринках і допомагає

підприємствам розуміти та оцінювати рівень конкуренції та можливості для успішної діяльності.

Модель «5 сил Портера» включає наступні п'ять основних сил, які визначають структуру і конкурентну динаміку галузі чи ринку.

1. Загроза нових учасників (Threat of New Entrants): ця сила оцінює, наскільки легко нові компанії можуть ввійти на ринок. Висока загроза нових учасників може підвищити конкуренцію та знизити прибутковість, а низька - створювати бар'єри для вступу.
2. Постачальники (Supplier Power): ця сила вказує на вплив постачальників на підприємство. Якщо постачальники мають великий вплив, вони можуть здійснювати тиск на ціни та умови поставки.
3. Покупці (Buyer Power): ця сила оцінює вплив покупців (споживачів) на підприємство. Якщо покупці мають великий вплив, вони можуть вимагати зниження цін та покращення якості товарів чи послуг.
4. Загроза заміщення (Threat of Substitution): ця сила вказує на те, наскільки легко споживачі можуть замінити продукт чи послугу іншими альтернативами. Висока загроза заміщення може обмежувати здатність підприємства збільшувати ціни.
5. Конкуренція в галузі (Competitive Rivalry): ця сила оцінює рівень конкуренції між існуючими гравцями на ринку. Велика конкуренція може призвести до падіння цін та зниження прибутковості.

Модель "5 сил Портера" допомагає підприємствам зрозуміти фактори, які впливають на їхнє становище на ринку та визначити стратегії для збереження конкурентних переваг. Вона є популярним інструментом для стратегічного аналізу і прийняття рішень в бізнесі.

Нижче, в таблиці 4.5 наведено розроблений перелік факторів конкурентоспроможності за моделлю «5 сил Портера».

Таблиця 4.5 Аналіз за моделлю «5 сил Портера»

Складова аналізу	Загроза нових учасників	Постачальники	Покупці	Загроза заміщення	Конкуренція в галузі
Зміст складової	Оскільки ринок насичений пропозиціями, нові учасники малоймовірно активно заходитимуть в цю нішу ринку	Постачальниками виступають лише компанії, що збирають історичні метеорологічні дані, за таких обставин тиск постачальників відсутній	Покупці чинять високий рівень впливу, оскільки від їх бажання платити за сервіс прямо залежить прибутковість. За умови високої конкуренції, цей чинник має велику вагу	Індивідуальні споживачі можуть замінити продукт без проблем. У випадку з метео-залежними бізнесами, що є клієнтами, ситуація є дещо іншою, втім вплив чинника залишається великим	Конкуренція в даному сегменті ринку є високою, втім ідея проекту пропонує надавати ексклюзивні за рахунок інноваційності послуги, що дозволяє дещо знизити вплив чинника

В таблиці 4.6 міститься інформація про фактори конкурентоспроможності та їх обґрунтування

Таблиця 4.6 Аналіз факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування
1	Інноваційність	Інноваційність та новизна пропонованого підходу до вирішення проблеми прогнозування погодних умов створює можливості для підвищеного інтересу до продукту з боку потенційних клієнтів
2	Широка вибірка прогнозованих показників	Можливість, в перспективі, здійснювати прогнозування таких показників як сила вітру, глибина снігового покриття, зволоженість ґрунту та інших, створює перспективу підвищеного інтересу до продукту з боку середнього та великого бізнесу

SWOT-аналіз (SWOT analysis) - це інструмент стратегічного аналізу, який використовується для оцінки внутрішніх сильних і слабких сторін підприємства чи проекту, а також зовнішніх можливостей і загроз.

Вкажемо компоненти аналізу, які визначає абревіатура SWOT.

1. S (Strengths) – сильні сторони: це внутрішні позитивні аспекти, які надають переваги підприємству чи проекту. Це можуть бути, наприклад, висока якість продукту, досвідний персонал, сильний бренд, ефективність виробництва тощо.
2. W (Weaknesses) – слабкі сторони: це внутрішні негативні аспекти, які можуть бути обмеженнями для підприємства чи проекту. Сюди входять, наприклад, недостатня інфраструктура, високі витрати, неефективні процеси, низька якість продукту тощо.
3. O (Opportunities) – можливості: це зовнішні фактори, які можуть бути вигідними для підприємства чи проекту. Можливості можуть включати

в себе нові ринки, зростання попиту на продукт, зміни в законодавстві, тенденції ринку, нові технології і таке інше.

4. T (Threats) – загрози: це зовнішні чинники, які можуть становити загрозу для підприємства чи проекту. Загрози можуть включати в себе конкуренцію, зміни в законодавстві, економічні кризи, зміни в споживчих уподобаннях, негативну репутацію тощо.

SWOT-аналіз допомагає підприємству або проекту ліпше розуміти своє оточення, ідентифікувати ключові фактори і розробляти стратегію на основі їхнього розуміння. Цей інструмент дозволяє виявити можливості для покращення та мінімізації загроз, а також робити вибіркові рішення для досягнення стратегічних цілей.

В таблиці 4.7 міститься матриця SWOT-аналізу проекту

Таблиця 4.7 SWOT-аналіз проекту

Сильні сторони:	Слабкі сторони:
Інноваційність, широка вибірка показників, що прогнозуються, підвищений інтерес до теми штучного інтелекту на ринку	Вартість розробки рішення в сфері штучного інтелекту, дороговизна створення проекту у порівнянні з конкурентами
Можливості:	Загрози:
Гнучкість архітектури, відкритість системи до вдосконалення, інтеграція нових моделей, розширення вибірки показників, що прогнозуються	Підвищена конкуренція в цьому сегменті ринку, певна ймовірність невдачі в залученні користувачів

Під час проведеного аналізу ринкових можливостей та загроз для стартап-проекту, який орієнтується на розробку і впровадження системи прогнозування метеорологічних показників на основі рекурентних нейронних мереж (LSTM), було виконано ґрунтовну оцінку різних факторів, що впливають на цей проект.

Однією з головних знахідок аналізу є те, що наявні ринкові можливості для такого стартапу є досить значущими. Постійна потреба у точних та надійних метеорологічних прогнозах створює підґрунтя для успішного впровадження даної технології. Переваги системи на основі LSTM включають в себе здатність адаптуватися до складних нестаціонарних метеорологічних явищ, що робить її привабливою для потенційних користувачів.

Однак аналіз також виявив загрози та виклики, з якими стикається стартап-проект. Конкуренція в галузі метеорології може бути досить високою, і важливо буде розробити стратегію виходу на ринок, щоб виділитися серед інших гравців.

4.4 Розробка ринкової стратегії проекту

Розділ "Розробка ринкової стратегії проекту" відіграє важливу роль у стратегічному плануванні та успішному впровадженні стартапу чи проекту. Розробка ринкової стратегії є ключовим кроком, який дозволяє підприємству чи команді проекту визначити, як вони мають намір взаємодіяти з ринком, конкурентами та клієнтами.

У даному розділі ми розглянемо процес розробки ринкової стратегії для нашого стартап-проекту, який спрямований на створення системи прогнозування метеорологічних показників на основі LSTM. Ми проаналізуємо внутрішні та зовнішні фактори, які впливають на нашу стратегію, і визначимо стратегічні кроки, спрямовані на досягнення наших цілей на ринку.

В таблиці 4.8 міститься аналіз можливих цільових учасників ринку

Таблиця 4.8 Можливі цільові учасники ринку

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит у межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Персональні користувачі	Низька / Середня	5%	Висока	Низька
2	Великі бізнеси	Середня	25%	Середня	Середня
3	Малі та середні бізнеси	Середня	30%	Середня	Середня
4	Держава	Низька	--	--	Низька

Оскільки держава в питаннях, що стосуються прогнозування метеорологічних показників для власних потреб, користується власними можливостями, то її не варто розглядати як можливу цільову групу.

З цільовими групами 1-3 можна працювати, при цьому варто зробити акцент саме на підприємствах-учасниках ринку.

Визначимо базову стратегію розвитку продукту (табл. 4.9-4.11).

Таблиця 4.9 Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитк
1	1, 2, 3	Змішаного маркетингу	Масштабування	Стратегія інновацій

Таблиця 4.10 Визначення базової стратегії конкурентної поведінки

Чи є проект «піонером» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
Ні	Так	Ні	Фокусування

Таблиця 4.11 Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
Якість результатів Широка вибірка показників, що прогнозуються	Стратегія інновацій	Інноваційність, Гнучкість	Система, створена на базі найсучасніших підходів Широкий асортимент показників, Точність прогнозів

4.5 Розробка маркетингової програми

Перейдемо до опису ключових переваг концепції продукту та побудуємо концепцію маркетингових комунікацій (табл. 4.12, 4.13).

Таблиця 4.12 Ключові переваги концепції продукту

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1	Якісне прогнозування	Висока якість прогнозу показників	Використання сучасних моделей штучного інтелекту дозволяє робити високоточні прогнози
2	Широкий набір показників, що прогнозуються	Прогноз практично будь-яких показників, по яких існують історичні дані в достатньому обсязі	Використання моделей машинного навчання та штучного інтелекту має значні переваги над традиційними методами метеорологічного прогнозування
3	Гнучкість системи, можливість отримувати ті дані, які потрібно	Можливе створення персоналізованих пакетів показників для конкретного клієнта	У разі, якщо клієнт потребує отримання регулярних прогнозів певного набору показників, для нього можна створити персоналізований пакет послуг

Таблиця 4.13 Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Пошук системних рішень, співвідношення ціна-якість	Реклама у соціальних мережах; Публікація реклами в профільній кореспонденції; Особисті комунікації	Точність Якість Інноваційність Постійне вдосконалення	Наголосити на унікальності нового підходу, висвітлити його переваги	Презентація можливостей системи, протиставлення з продуктом-конкурентом

4.5 Висновки до розділу

У даному розділі була розглянута можливість розробки та впровадження стартап-проекту, який спрямований на створення високоякісного програмного забезпечення для аналізу та прогнозування метеорологічних даних. Цей проект відображає важливість інновацій та передових технологій у цій сфері та має потенціал змінити підхід до прогнозування погодних умов.

Успішний вихід на ринок вимагає не лише відмінного продукту, але й ефективної маркетингової стратегії. Створення унікальної пропозиції для клієнтів, розробка якісного бренду та маркетингова кампанія стануть важливими складовими успіху. Аналіз цільової аудиторії та визначення факторів, які впливають на привабливість продукту для цієї аудиторії, є ключовими етапами розробки маркетингової стратегії.

Оскільки потенційні переваги проекту є реалістичними та прогнозованими, з урахуванням потенційних клієнтів та ринкових умов, проект виявляється в перевагах перед конкурентами. Поєднання інноваційних рішень та якісного програмного забезпечення для фінансового аналізу робить проект привабливим для підприємств та індивідуальних користувачів.

Загалом можна зробити висновок, що подальша імплементація проекту є доцільною та обіцяючою. Розробка високоякісного програмного забезпечення для вирішення поставлених задач має потенціал змінити галузь та надати користувачам ефективний інструмент.

ВИСНОВКИ

У даній магістерській дисертації була розглянута дуже актуальна та нагальна проблема, якою є прогнозування погодних умов. Сучасний розвиток технологій і постійні зміни в кліматичних умовах ставлять перед науковцями та практиками важливе завдання забезпечення точного і надійного прогнозу погоди. Погодні умови мають великий вплив на різні аспекти нашого життя, включаючи сільське господарство, транспорт, енергетику та безпеку. Тому розвиток підходів та технологій для ефективного прогнозування погоди є актуальною та важливою задачею, і ця робота відділяється на тлі цієї актуальності.

У світі науки та досліджень останні роки характеризуються інтенсивним розвитком моделей штучного інтелекту, які виявили значний потенціал у вирішенні проблем прогнозування погоди. Використання нейронних мереж, зокрема моделей з довгочасною короткочасною пам'яттю (LSTM), стало популярним та перспективним напрямком в цій галузі. Це дозволило покращити точність прогнозування та зменшити похибки, пов'язані з погодними передбаченнями.

Модель LSTM була вибрана через її унікальні особливості та здатність моделювати послідовні дані, які є характерними для погодних процесів. Ця модель добре адаптується до змінних умов і має вбудований механізм для роботи з даними, що змінюються в часі. Це робить її ідеальним інструментом для прогнозування погоди, де важливо брати до уваги зміни в часі та взаємозв'язки між погодними параметрами.

При реалізації даної моделі однією з найскладніших проблем традиційно вважають велику обчислювальну складність процесу тренування моделі. В даній роботі вказана проблема була виявлена, і слід зазначити, що для досягнення підвищення точності прогнозування необхідно було б в

подальшому проводити складні обчислення, і це вимагало б більших обчислювальних ресурсів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Rumelhart D. E., Hinton G. E., Williams R. J. Learning internal representations by error propagation. 1985. Tech. rep. ICS 8504. San Diego, California: Institute for Cognitive Science, University of California.
2. Hochreiter S., Schmidhuber J. Long Short-Term Memory. 1997. Neural Comput. 9, 8 (November 15), P. 1735–1780.
3. Evgeniou T., Pontil M. Support Vector Machines: Theory and Applications. 2001. 2049, P. 249-257. DOI: 10.1007/3-540-44673-7_12.
4. Sherstinsky A. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. 2018. ArXiv, abs/1808.03314.
5. Chen J., Huang C. A Comprehensive Review of Weather Forecasting Using Machine Learning. 2020. IEEE Access, 8, 65288-65311.
6. Paramasivan S. K. Deep learning based recurrent neural networks to enhance the performance of wind energy forecasting: A review. Revue d'Intelligence Artificielle. 2021.
7. Ly R., Traoré F., Dia K. Forecasting Commodity Prices Using Long-Short-Term Memory Neural Networks. Social Sciences Education eJournal. 2021.
8. Han J. M., Ang Y. Q., Malkawi A., Samuelson H. W. Using recurrent neural networks for localized weather prediction with combined use of public airport data and on-site measurements. Building and Environment, 192. 2021. 107601.
9. Sabzipour B., Arsenault R., Troin M., Martel J.-L., Brissette F., Brunet F., Mai J. Comparing a long short-term memory (LSTM) neural network with a physically-based hydrological model for streamflow forecasting over a Canadian catchment. Journal of Hydrology, 130380. 2023.
10. Іванійчук А.П., Гуськова В.Г. Система прогнозування метеорологічних умов на основі методів аналізу даних та штучного інтелекту. II Всеукраїнська науково-практична конференція «Системні науки та

інформатика» з нагоди 125-річчя КПІ ім. Ігоря Сікорського, м. Київ, 04-08 грудня 2023 року. С. 121-126.

11. REST API Documentation. (n.d.). URL: https://developer.mozilla.org/en-US/docs/Glossary/REST.
12. OpenWeather. (n.d.). Official Website. URL: https://openweathermap.org/.
13. LSTM: Long Short-Term Memory. (n.d.). URL: https://www.tensorflow.org/guide/keras/rnn#lstm_long_short-term_memory.
14. PyTorch Documentation. (n.d.). URL: https://pytorch.org/docs/stable/index.html.
15. Pandas Documentation. (n.d.). URL: https://pandas.pydata.org/docs/.
16. AL-Chalabi Hussan, Al-Douri Yamur, Lundberg, Jan. Time Series Forecasting using ARIMA Model: A case study of mining face drilling rig. 2018.
17. Пашко А.О. Статистичний аналіз даних. КНУ ім. Т.Г. Шевченка, Київ. 2019. 55 с.
18. Ali Jehad, Khan Rehanullah, Ahmad Nasir, Maqsood Imran. Random Forests and Decision Trees. International Journal of Computer Science Issues(IJCSI). 9. 2012.
19. Chen Wei, Luo Yu-Feng, Wen Xinyu, Zhang Yiyi Yin, Wu Yingdan Yao. Pre-evacuation Time Estimation Based Emergency Evacuation Simulation in Urban Residential Communities. International Journal of Environmental Research and Public Health. 2019. 16. 4599. 10.3390/ijerph16234599.
20. Nordin Md Jan, Xin Ooi, Aziz NorShakirah. Food Image Recognition for Price Calculation using Convolutional Neural Network. 2019. P. 80-85. 10.1145/3316551.3316557.
21. Lokhandwala Mufaddal, Saif Ahmed, Ghasemi Alireza. Machine Learning Models for Storm Damage Prediction in Nova Scotia, Canada. 2023. 10.1109/DASA59624.2023.10286795.

22. Recurrent Neural Network (RNN) Architecture Explained. URL: https://medium.com/@poudelsushmita878/recurrent-neural-network-rnn-architecture-explained-1d69560541ef
23. Christopher Olah. Understanding LSTM Networks. 2015. URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

ДОДАТОК А

```
import numpy as np
import torch
import torch.nn as nn

from sklearn.preprocessing import StandardScaler

device = torch.device('cuda:0')

original_df = pd.read_csv("/content/kyiv-ukraine.csv")
original_df.columns

#Train size: 80% of the total data size
train_size = int(len(input_data) * 0.80)

# Normalization of the inputs
scaler = StandardScaler()
scaler.fit(input_data[:train_size + T - 1])
input_data = scaler.transform(input_data)

# Preparing X_train and y_train
X_train = np.zeros((train_size, T, D))
y_train = np.zeros((train_size, 1))

for t in range(train_size):
  X_train[t, :, :] = input_data[t:t+T]
  y_train[t] = (targets[t+T])

# Preparing X_test and y_test
X_test = np.zeros((N - train_size, T, D))
y_test = np.zeros((N - train_size, 1))

for i in range(N - train_size):
  t = i + train_size
  X_test[i, :, :] = input_data[t:t+T]
  y_test[i] = (targets[t+T])

# Make inputs and targets
X_train = torch.from_numpy(X_train.astype(np.float32))
y_train = torch.from_numpy(y_train.astype(np.float32))
X_test = torch.from_numpy(X_test.astype(np.float32))
y_test = torch.from_numpy(y_test.astype(np.float32))

class LSTM(nn.Module):
  def __init__(self, input_dim, hidden_dim, layer_dim, output_dim):
    super(LSTM, self).__init__()
```

```
    self.M = hidden_dim
    self.L = layer_dim

    self.rnn = nn.LSTM(
        input_size=input_dim,
        hidden_size=hidden_dim,
        num_layers=layer_dim,
        batch_first=True)
    #batch_first to have (batch_dim, seq_dim, feature_dim)
    self.fc = nn.Linear(hidden_dim, output_dim)

  def forward(self, X):
    # initial hidden state and cell state
    h0 = torch.zeros(self.L, X.size(0), self.M).to(device)
    c0 = torch.zeros(self.L, X.size(0), self.M).to(device)

    out, (hn, cn) = self.rnn(X, (h0.detach(), c0.detach()))

    # h(T) at the final time step
    out = self.fc(out[:, -1, :])
    return out

model = LSTM(D, 512, 2, 1)
model.to(device)

new_df = original_df[["Year", "Month", "Day", "t_min_c", "t_mean_c",
"t_max_c", "precipitation_mm"]]
new_df.head()

new_df.loc[:, 'date'] = pd.to_datetime(new_df.loc[:, ("Year", "Month",
"Day")], errors='coerce')
new_df = new_df[~new_df['date'].isna()]
len(new_df.index)

main_df = new_df.drop(columns=['Year', 'Month', 'Day'])

weather = main_df.copy(deep=True)
weather['month'] = weather['date'].dt.month

weather.head(2)

weather_naive = weather[['date', 't_mean_c']].copy(deep=True)
weather_naive['prev_temperature'] = weather_naive['t_mean_c'].shift(1)
weather_naive.head(2)

weather_naive.drop([0], inplace=True)
weather_naive.head(2)

weather_naive['difference'] = weather_naive['t_mean_c'] -
weather_naive['prev_temperature']
weather_naive['square_error'] = weather_naive['difference'] ** 2
```

```
print(weather_naive.head(2))

square_error = weather_naive['square_error'].mean()
print(f'Square Error of the Naive Approach is {square_error:.3f}')

weather_LSTM = weather.copy(deep=True)
weather_LSTM = pd.get_dummies(weather, columns = ['month'])
weather_LSTM.columns

input_data = weather_LSTM.drop(['date'], axis=1)
targets = weather_LSTM['t_mean_c'].values
T = 20                    #Number of timesteps to look while
predicting
D = input_data.shape[1]   #Dimensionality of the input
N = len(input_data) – T

import os
from constants import SCRIPTS_PATH

CWD_PATH = os.getcwd()


class ValidationError(Exception):
    """
    Script was marked by the model as invalid
    """
    def __init__(self, user_prompt: str):
        self.msg = f"PROMPT: {user_prompt}\nThe model failed to
generate a valid script\n"

    def __str__(self) -> str:
        return self.msg


def save_script(script: str):
    """
    Save the generated Node.js script to the file system.

    This function saves the generated Node.js script to the file
system in the directory specified by the SCRIPTS_PATH
    constant. The saved script filename follows the naming convention
script_<index>.js, where <index> is an integer
    larger than any previously saved script.

    :param script: The Node.js script to be saved.
    :return: The index of the saved script.
    """
    # Create the save to directory if it does not exist
    save_to_path = os.path.join(CWD_PATH, SCRIPTS_PATH)
    if not os.path.isdir(save_to_path):
        os.makedirs(save_to_path)
```

```
    # Get a list of all the files in the directory that match the
naming convention
    files = [
        f for f in os.listdir(save_to_path) if f.startswith("script_")
and f.endswith(".js")
    ]

    # Extract the index from each filename and find the maximum value
    indices = [int(f.lstrip("script_").rstrip('.js')) for f in files]
    max_index = max(indices) if indices else 0

    # Generate a new filename using the maximum index + 1
    new_file_name = f"script_{max_index + 1}.js"

    # Save the script
    with open(os.path.join(save_to_path, new_file_name), "w") as file:
        file.write(script)
        file.write("\nconst args =
process.argv.slice(2)\ntest(...args)")

    return str(max_index + 1)



import os
import subprocess
from flask import Flask
from flask_restful import Resource, Api, reqparse

from models.llm import ScriptGenerator
from constants import SCRIPTS_PATH

app = Flask(__name__)
api = Api(app)

# Request parser for script generator
generator_parser = reqparse.RequestParser()
generator_parser.add_argument("description", type=str, required=True,
help="Enter the prompt!")

# Request parser for script runner
runner_parser = reqparse.RequestParser()
runner_parser.add_argument("script_id", type=str, required=True,
help="Provide a script id to run the script")
runner_parser.add_argument("parameters", help="Provide parameters for
the script", action="append")


class GeneratorBot(Resource):
    """
```

```
    This class represents an API endpoint for generating and saving
node.js scripts based on the prompt
    received in a POST request. It uses the ScriptGenerator object to
generate and validate the script.
    """
    def __init__(self):
        self.generator: ScriptGenerator = ScriptGenerator()

    @staticmethod
    def get():
        return "Opposite to great success, use POST instead"

    def post(self):
        """
        Generates and saves node.js scripts based on the prompt
received in the POST request. Uses the
        ScriptGenerator object to generate and validate the script.
Returns the script ID.
        :return: The script ID.
        """
        args = generator_parser.parse_args()
        script_id = self.generator.run(args["description"])
        return script_id


class ScriptRunner(Resource):
    """
    A Flask resource that runs a script with given
    """
    @staticmethod
    def get():
        return "Opposite to great success, use POST instead"

    @staticmethod
    def post():
        """
        Runs a script with given parameters using.
        :return: The output of the script as a string
        """
        args = runner_parser.parse_args()
        subprocess_command = ["node", os.path.join(SCRIPTS_PATH,
f"script_{args['script_id']}.js")]
        if args["parameters"]:
            subprocess_command = subprocess_command +
args["parameters"]
        sub = subprocess.run(
            subprocess_command,
            capture_output=True
        )
        return sub.stdout.decode("utf-8").strip()
```

```
api.add_resource(GeneratorBot, "/generate")
api.add_resource(ScriptRunner, "/test")


if name == '__main__':
    app.run()


import pandas as pd

from sklearn.preprocessing import StandardScaler, OneHotEncoder

from sklearn.impute import SimpleImputer

from sklearn.compose import ColumnTransformer

from sklearn.pipeline import Pipeline


class DatasetProcessor:

    def __init__(self, numerical_features, categorical_features):

        """

        Initialize the DatasetProcessor.


        :param numerical_features: List of names of numerical features.

        :param categorical_features: List of names of categorical features.

        """

        self.numerical_features = numerical_features

        self.categorical_features = categorical_features

        self.pipeline = None


    def _create_processing_pipeline(self):

        """

        Create a data processing pipeline with separate steps for numerical and categorical features.
```

```
        """
        # Numerical features pipeline
        numerical_pipeline = Pipeline([
            ('imputer',  SimpleImputer(strategy='mean')),    #  Handle
missing values
            ('scaler', StandardScaler())                  # Standardize
features
        ])

        # Categorical features pipeline
        categorical_pipeline = Pipeline([
            ('imputer',   SimpleImputer(strategy='most_frequent')),   #
Handle missing values
            ('encoder', OneHotEncoder(handle_unknown='ignore'))   # One
hot encoding
        ])

        # Full pipeline
        self.pipeline = ColumnTransformer([
            ('num', numerical_pipeline, self.numerical_features),
            ('cat', categorical_pipeline, self.categorical_features)
        ])

    def fit_transform(self, df):
        """
        Fit the processing pipeline to the data and transform it.

        :param df: The Pandas DataFrame to process.
        :return: Transformed DataFrame.
```

```
        """
        if self.pipeline is None:
            self._create_processing_pipeline()

        return pd.DataFrame(self.pipeline.fit_transform(df))

    def transform(self, df):
        """
        Transform the data using the already fitted pipeline.

        :param df: The Pandas DataFrame to process.
        :return: Transformed DataFrame.
        """
        if self.pipeline is None:
            raise RuntimeError("Pipeline not fitted. Call 'fit_transform' first.")
        return pd.DataFrame(self.pipeline.transform(df))
processor = DatasetProcessor(numerical_features, categorical_features)
# Assuming df is your raw DataFrame
# processed_df = processor.fit_transform(df)
```