МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ СЕМЕНА КУЗНЕЦЯ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАТИКИ ТА КОМП'ЮТЕРНОЇ ТЕХНІКИ

Рівень вищої освіти	Перший (бакалаврський)
Спеціальність	Інформаційні системи та технології
Освітня програма	Інформаційні системи та технології
Група	6.04.126.010.19.1

ДИПЛОМНИЙ ПРОЄКТ

на тему: «Розроблення інформаційного модуля оцінки епідемічної небезпеки поширення COVID-19/ Development of an information module for the epidemic risk for the spread of COVID-19»

Виконав: студент Артем ДОЛГИЙ

Керівник: к.т.н., доцент Ольга ТЮТЮНИК

Рецензент: к.т.н., доцент Михайло ПІКСАСОВ

Харків – 2023 рік

РЕФЕРАТ

Пояснювальна записка до дипломного проєкту: 102 сторінок, 42 рисунка, 2 таблиці, 1 додаток, 44джерела.

Об'єктом дослідження є інформаційні системи та технології оброблення даних для оцінки епідеміологічної небезпеки поширення COVID-19.

Предметом дослідження є інформаційний модуль оцінки епідемічної небезпеки поширення COVID-19.

Метою дипломного проєкту є розроблення інформаційного модулю оцінки епідемічної небезпеки поширення COVID-19.

Методи розроблення. Методи системного аналізу, статистичні методи обробки інформації, методи прогнозування, методи машинного навчання.

Теоретичні результати дипломного проєкту рекомендовано для використання у навчальному процесі кафедри інформатики та комп’ютерної техніки Харківського національного економічного університету імені Семена Кузнеця під час вивчення освітньої компоненти "Нейромережева обробка даних".

ARIMA; АЛГОРИТМИ; ІНФОРМАЦІЙНАСИСТЕМА; ЗГОРТКОВА НЕЙРОННА МЕРЕЖА; COVID-19; LSTM; МОДУЛЬ ОБРОБЛЕННЯ ДАНИХ; ПРОГНОЗУВАННЯ; НЕЙРОННІ МЕРЕЖІ; PYTHON; DECISION FOREST.

ABSTRACT

The bachelor's thesis report: 102 pages, 42 figures, 2 tables, 1 appendix, 44 sources.

The object of the study is information systems and data processing technologies for assessing the epidemiological danger of the spread of COVID-19.

The subject of the study is the information module for assessing the epidemic danger of the spread of COVID-19.

The goal of the diploma project is to develop an information module for assessing the epidemic danger of the spread of COVID-19.

Development methods. System analysis methods, statistical information processing methods, forecasting methods, machine learning methods.

The theoretical results of the bachelor's thesis report are recommended for use in the educational process of the Department of Informatics and Computer Engineering of Simon Kuznets Kharkiv National University of Economics during the study of the educational component "Neural Network Data Processing".

ARIMA; ALGORITHMS; INFORMATION SYSTEM; CONVOLUTIONAL NEURAL NETWORK;COVID-19; LSTM; DATA PROCESSING MODULE; FORECASTING; NEURON NETWORKS; PYTHON; DECISION FOREST.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ 8
ВСТУП 9
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ 11
1.1. Змістовний опис і аналіз предметної області 11
1.2 Аналіз методів прогнозування 16
1.3 Аналіз особливостей прогнозування рядів динаміки 18
1.4 Аналіз існуючих досліджень щодо оцінки епідемічної небезпеки поширення COVID-19 26
1.5 Розроблення діаграми бізнес-процесів системи «Розрахунок прогнозів»за стандартом IDEF0 30
1.6 Постановка задачі дослідження 33
2 АНАЛІЗ МАТЕМАТИЧНИХ МОДЕЛЕЙ ДЛЯ ОЦІНКИ ЕПІДЕМІЧНОЇ НЕБЕЗПЕКИ ПОШИРЕННЯ COVID-19 34
2.1 Аналіз ефективності застосування моделі ARIMA для оцінки епідемічної небезпеки поширення COVID-19 34
2.1.1 Математична постановка моделі ARIMA 36
2.1.2 Переваги та недоліки застосування ARIMA моделі для оцінки епідемічної небезпеки поширення COVID-19 39
2.2 Аналіз ефективності застосування моделі LSTM для оцінки епідемічної небезпеки поширення COVID-19 40
2.2.1 Математична постановка моделі LSTM 48
2.2.2 Переваги та недоліки застосування LSTM моделі для оцінки епідемічної небезпеки поширення COVID-19 52
2.3 Аналіз ефективності застосування DecisionForest для оцінки епідемічної небезпеки поширення COVID-19 54

2.3.1 Математична постановка моделіDecisionForest 54
2.3.2 Переваги та недоліки застосування DecisionForest моделі для оцінки епідемічної небезпеки поширення COVID-19 56
3 СПЕЦИФІКАЦІЯ ВИМОГ ДО ІНФОРМАЦІЙНОГО МОДУЛЯ ОЦІНКИ ЕПІДЕМІЧНОЇ НЕБЕЗПЕКИ ПОШИРЕННЯ COVID-19 ТА ТЕХНІЧНІ РІШЕННЯ 57
3.1 Глосарій 57
3.2 Розроблення варіантів використання 58
3.3 Аналіз програмного забезпечення для реалізації інформаційного модуля оцінки епідемічної небезпеки поширення COVID-19 59
4 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОГО МОДУЛЮ ОЦІНКИ ЕПІДЕМІЧНОЇ НЕБЕЗПЕКИ ПОШИРЕННЯ COVID-19 63
4.1 Розроблення інформаційного модулю оцінки епідемічної небезпеки поширення COVID-19 63
4.1.1 Завантаження набору даних та очищення даних 63
4.2 Підготовка даних для роботи з ARIMA моделлю 67
4.2.1 Пошук найкращих параметрів для ARIMA моделі 72
4.3 Підготовка даних для роботи з LTSM моделлю 78
4.4 Побудова структури DecisionForest моделі та її конфігурація 80
4.5 Порівняльний аналіз результатів моделей 82
ВИСНОВКИ 84
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 86
ДОДАТОК А 91

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ARMA – модель авторегресії ковзного середнього (від англ. AutoregressiveMovingAverage)

COVID-19 – коронавірусна інфекція (від англ.CoronaVirusDisease)

LSTM – нейронна мережа довгої короткострокової пам'яті (від англ. LongShort-TermMemory)

RNN – рекурентна нейронна мережа мережа (від англ. RecurrentNeuralNetwork)

ВСТУП

COVID-19 (від англ.CoronaVirusDisease – коронавірусна інфекція)– це новий штам коронавірусу, який був уперше ідентифіковано у грудні 2019 року в провінції Хубей, Китай. Міжнародний комітет з питань систематики вірусів прийняв 11 лютого 2020 року офіційну назву цього вірусу SARS-CoV-2, керуючись наступним уявленням:

-SARS (скорочено від англ. SevereAcuteRespiratorySyndromeCoronavirus 2) – це важкий гострий респіраторний синдром;

- CoV (скорочено від англ. CoronaVirus) – коронавірус;

- 2 – другий відомий людині SARS-CoV.

Всесвітня організація охорони здоров’я 30 січня 2020 року оголосила надзвичайну ситуацію в галузі охорони здоров’я, а 11 березня 2020 року – пандемію. Значне та непередбачуване поширення нової коронавірусної інфекції у всьому світі спричинило глобальне та безпрецедентне обмеження у всіх сферах соціального життя та величезне навантаження на заклади охорони здоров’я [1].

Епідемія, яка була спричинена коронавірусом нового типу SARS-CoV2, поставила перед багатьма країнами серйозні завдання, які вимагали оперативних, продуманих та обґрунтованих дій.

Крім необхідних та очевидних рішень: закриття сухопутних кордонів із зараженими країнами, припинення авіасполучення, мобілізація системи охорони здоров'я, однією зпріоритетними завданнями стало прогнозування розвитку ситуації та складання плану превентивних заходів з метою недопущення найбільш несприятливого сценарію.

За період пандемії 2019-2023 роках накопичились великі об’єми даних які дозволяють зібрати необхідну інформацію про походження, структуру вірусу, що дає необхідний матеріал дослідникам-вченим для прогнозування розвитку епідемії COVID-19.

На сьогодні існує чимало відкритих даних та досліджень, які дозволяють країнам краще розуміти поточну ситуацію та приймати більш зважені рішення, які рятують життя мільйонам громадян. Значну роль у цьому грають різні математичні моделі, які використовують накопичений масив даних як про минулі епідемії, так і про поточну, щоб передбачати подальший розвиток епідеміологічної ситуації.

Таким чином для вирішення соціально-економічних, регуляційних, медичних та інших питань керівництву країн необхідно мати всі прогнозні варіанти розвитку ситуації для координації дій щодо запобігання розповсюдження COVID-19, що і обумовлює актуальність теми цього дипломного проєкту.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1. Змістовний опис і аналіз предметної області

Коронавіруси – це група вірусів, які здебільшого спричиняють незначні проблеми, що супроводжуються такими симптомами, як кашель та застуда. Більшість коронавірусів є нешкідливими для людей. Проте новий коронавірус COVID-19 є доволі агресивним. Він часто спричиняє смерть пацієнтів або загострення інших захворювань, що приводить до смерті.

Вірус називають «короною» завдяки коронoподібним білковим структурам найого поверхні. Коронавіруси є одноланцюговими РНК-вірусами. Вони сильніше мутують у порівнянні з вірусами на ДНК-основі.

COVID-19 поширюється швидше, ніж інші вірусні ГРВІ. При кожному контакті COVID-19 з людським організмом формується дуже міцний контакт з мембраною людської клітини завдяки білковим шипам.

Передача вірусу відбувається у переважній більшості випадків саме повітряно-крапельним шляхом та під час близьких контактів. Проте можливе безсимптомнезараження. Між моментом зараження та початком появи перших симптомів проходить від 2 до 14 днів. До поширених відомих симптомів відносяться гарячка, кашель та задишка. Також ускладнення можуть нести за собою гострий респіраторний дистрес-синдром та запалення легень. COVID-19 визнають дуже загрозливим тому, що при ураженні легень COVID-19 людина може не мати симптомів захворювання довгий час, але зміни у дихальних мережах хворого у цей час можуть бути вже не виправними.

До недавнього часу не існувало вакцин та противірусних препаратів для лікування хвороби. Первинне лікування направлене на підтримку хворого та симптоматичне. До профілактичних заходів відносяться соціальне дистанціювання, самоізоляція осіб з підозрою на зараження та дотримання правил особистої гігієни.

Хоча показник летальності від COVID-19 становить всього кілька відсотків, пов'язана з цим пандемія викликала набагато більше смертей у всьому світі. За даними інформаційної моделі ВООЗ [2] за три роки зафіксовано підтверджених випадків COVID-2019 більше 766 мільйонів та майже 7 мільйонів смертей, але фактична кількість може бути в кілька разів вище (рис. 1.1).

Рисунок 1.1 – Кількість підтверджених випадків COVID-19 та смертей

У 2020 році Тедросом Аданом Гебреєсусом - головою Всесвітньої організації охорони здоров'я, було названо основні сценарії поширення коронавірусу у різних країнах світу [3]:

1) відноситься до країн, які були заздалегідь попереджені і мали уявлення про майбутній спалах хвороби — в результаті яким вдалося уникнути масштабних спалахів (деякі з країн Карибського басейну, Південно-Східної Азії, Тихоокеанського регіону та Африки);

2) стосувався багатьох країн Європи — масштабні спалахи захворювання, які вдалося контролювати завдяки діям керівництва країн;

3) розгортався у країнах, де вдалося подолати перший спалах хвороби, але були послаблені обмеження, що призвело до необхідності боротьби з подальшими хвилями захворювання;

4) сценарій несе під собою фазу інтенсивної передачі інфекції (Америка, Південна Азія і кілька країн Африки).

Розуміння ключових дат і подій у хронології пандемії COVID-19 має вирішальне значення для аналізу еволюції вірусу та розвитку спалаху, а також для оцінки ефективності різних втручань і заходів реагування. В даному дипломному проєкті на прикладі Північної Америкибуло проведено аналіз важливих дат і подій пандемії COVID-19, результати якого наведені у таблиці 1.1 дод.А.

Аналіз хронології ключових подій під час пандемії COVID-19 у Північній Америці дає важливу інформацію про поширення та боротьбу з вірусом. Вивчення цих дат дозволяє краще зрозуміти фактори, які вплинули на прогресування пандемії та ефективність різних втручань [24], а саме:

1. Раннє виявлення та реагування: перші випадки COVID-19 у Північній Америці були зареєстровані в січні 2020 року, коли уряди та органи охорони здоров’я вживали різноманітних заходів для стримування поширення вірусу. Ці перші дії, такі як обмеження на подорожі та карантинні заходи, могли допомогти уповільнити початкове поширення вірусу, вигравши час для систем охорони здоров’я на підготовку та адаптацію.

2. Заходи карантину та накази залишатися вдома: протягом березня та квітня 2020 року багато штатів і провінцій запровадили заходи карантину та накази залишатися вдома, щоб зменшити передачу вірусу. Ці заходи були ефективними для вирівнювання кривої та запобігання перевантаження систем охорони здоров’я, хоча вони також мали значні економічні та соціальні наслідки.

3. Поступове відновлення та пом’якшення обмежень: оскільки в травні 2020 року кількість нових випадків почала зменшуватися, багато штатів і провінцій почали поступово відновлювати діяльність підприємств і пом’якшувати обмеження. Однак ці рішення часто викликали дебати та суперечки, оскільки деякі стверджували, що надто швидке повторне відкриття може призвести до пожвавлення справ.

4. Вплив протестів і демонстрацій. Широкомасштабні протести та демонстрації в червні 2020 року викликали занепокоєння щодо потенційного

зростання поширення COVID-19. Проте дослідження показали, що вплив на передачу був обмеженим через відкритий характер протестів і широке використання масок.

5. Поява варіантів, що викликають занепокоєння: поява нових варіантів COVID-19, таких як B.1.1.7, B.1.351, P.1 і B.1.617.2, була серйозною проблемою протягом пандемії. Ці варіанти часто мають підвищену трансмісивність або стійкість до імунітету, що вимагає постійного моніторингу та адаптації стратегій громадського здоров’я.

6. Кампанії з вакцинації та авторизація вакцин: розробка та розповсюдження вакцин проти COVID-19 стали головним поворотним моментом у пандемії. Було доведено, що вакцини дуже ефективні для запобігання важким захворюванням, госпіталізації та смерті. Поточні кампанії вакцинації та дозвіл на вакцини для різних вікових груп зіграли вирішальну роль у зменшенні впливу вірусу та забезпеченні поступового повернення до нормального життя.

7. Бустерні щеплення та нові рекомендації щодо вакцин: оскільки з’явилися докази ослаблення імунітету, який забезпечують вакцини проти COVID-19, бустерні щеплення стали ключовою частиною стратегій вакцинації в Північній Америці. Ці додаткові дози допомагають підтримувати високий рівень захисту від вірусу, особливо перед обличчям нових варіантів.

Таким чином, аналізуючи важливі дати та події в хронології пандемії COVID-19 у Північній Америці, можна отримати глибоке розуміння факторів, які вплинули на перебіг пандемії та ефективність різних заходів.

За період пандемії імунітет населення зростає, завдяки вакцинації, знижуються літальні випадки, вже понад рік пандемія має низхідну тенденцію, що послаблює тиск на системи охорони здоров'я країн. У результаті чого, більшість країн повернулася до звичайного до пандемічного життя.

4 травня 2023 р. Всесвітня організація з охорони здоров'я оголосила про закінчення пандемії COVID-19.

Незважаючи на прогрес, досягнутий у боротьбі з пандемією COVID-19, залишаються значні проблеми. До них належать усунення сумнівів щодо вакцинації, забезпечення глобальної справедливості щодо вакцин і моніторинг нових варіантів, що викликають занепокоєння. Рухаючись вперед, установам охорони здоров’я, урядам і дослідникам буде важливо продовжувати співпрацю, щоб зрозуміти й вирішити ці постійні проблеми. Ці знання можуть стати основою для підготовки до майбутніх пандемій і заходів щодо реагування, допомагаючи мінімізувати вплив спалахів інфекційних захворювань на громадське здоров’я, суспільство та економіку.

Використання методів прогнозування для еволюції пандемії в часі з цілями розробки політики, яка буде заснована на наукових даних для боротьби з пандеміями в майбутньому є актуальною задачею сьогодення.

Зворотньою стороною аналізу процесів розповсюдження коронавірусної інфекції є вплив суб’єктивних факторів (менталітет, дисциплінованість громадян тощо) на достовірність статистичної інформації.

Методи прогнозування дозволяють оцінювати ситуації в минулому, і таким чином це дозволяє краще прогнозувати ситуацію, яка може скластися у майбутньому. Враховуючи унікальність нинішньої пандемії, важливо також розроблювати прогнози, які враховують і можливі сценарії розвитку епідемії, чим можуть допомогти підготуватися до потенційних загроз і наслідків.

Отже, пандемія COVID-19 суттєво вплинула на життя людей у всьому світі з моменту її виникнення наприкінці 2019 року. Оскільки хвороба швидко поширювалася країнами, представники влади та медичні працівники зіткнулися з безпрецедентними викликами в розумінні, прогнозуванні та контролі за її прогресуванням. Технології, особливо методи аналізу та візуалізації даних, відіграли важливу роль у забезпеченні кращого прийняття рішень та сприянні більш ефективному реагуванню на пандемію.

Для своєчасного застосування медичних та організаційних заходів необхідно застосування достовірних даних у форматі реального часу для забезпечення прийняття обґрунтованих політичних рішень та ефективного розподілу ресурсів.

Прийняття рішень на основі даних виявилося вирішальним у боротьбі з пандемією, про що свідчить швидке впровадження технологій та інструментів аналізу даних урядами та медичними працівниками.

1.2 Аналіз методів прогнозування

Прогнозування та передбачення результатів є важливим аспектом аналізу даних у контексті пандемії COVID-19. Використовуючи моделі машинного навчання та технології штучного інтелекту, науковці змогли розробити моделі прогнозування розповсюдження вірусу в найближчому майбутньому, визначити зони підвищеного ризику та оцінити потенційний вплив різних профілактичних заходів у сфері громадського здоров'я.

Наприклад, моделі машинного навчання, такі як мережі з довгою короткочасною пам'яттю (LSTM) та DecisionForest, були використані для прогнозування кількості випадків COVID-19, госпіталізацій та смертей за різних сценаріїв та стратегій втручання[27]. Ці прогнози стали основою для прийняття політичних рішень і розподілу ресурсів, що дозволило урядам і системам охорони здоров'я підготуватися до пандемії та пом'якшити її наслідки.

Крім того, прогнозування та передбачення результатів відіграли вирішальну роль у розробці та розповсюдженні вакцин. Використовуючи епідеміологічні моделі та дані про ефективність вакцин, дослідники змогли оцінити рівень охоплення вакцинацією, необхідний для досягнення колективного імунітету та розробити стратегії розподілу вакцин. Ця інформація відіграла важливу роль у спрямуванні глобальних зусиль на боротьбу з пандемією та запобігання майбутнім хвилям інфекції.

Аналіз даних також сприяв розробці систем раннього попередження для прогнозування потенційних спалахів та інформування про доцільність проведення карантинних заходів у сфері охорони здоров'я. Наприклад, дослідники використовували алгоритми машинного навчання для аналізу даних з різних джерел, таких як соціальні мережі, пошукові тренди в Інтернеті та дані про пересування людей, щоб виявити ранні ознаки посилення передачі COVID-19 і потенційних спалахів. Ці системи раннього попередження дозволили урядам і організаціям громадського здоров'я швидко реагувати на нові загрози та впроваджувати цілеспрямовані заходи для контролю за поширенням вірусу.

Загалом, знання, отримані в результаті аналізу даних, відіграли важливу роль в інформуванні та оптимізації глобальної відповіді на пандемію COVID-19. Виявляючи закономірності та тенденції, оцінюючи вплив втручань, а також прогнозуючи та передбачаючи результати, дослідники та політики змогли прийняти рішення на основі даних для пом'якшення наслідків пандемії та захисту громадського здоров'я.

Прогнозування–це передбачення майбутнього виходячи з накопиченого досвіду таданих.

Під методами прогнозування (рис.1.2) розуміють сукупність прийомів і способів мислення, що дозволяють на основі аналізу зовнішніх і внутрішніх зв'язків об'єкта прогнозування розробити прогноз, з певною мірою достовірності, щодо майбутнього розвитку об'єкта.

Прогнозування з використанням моделей включає їх розробку, експериментальний аналіз, зіставлення результатів попередніх прогнозних розрахунків з фактичними даними стану процесу або об'єкта, уточнення і коригування моделі.

Дані стосовно динаміки числа інфікованих доцільно представити у вигляді рядів динаміки, які можуть бути описані математичними моделями, для обробки яких доцільним є використання технічнихзасобів.

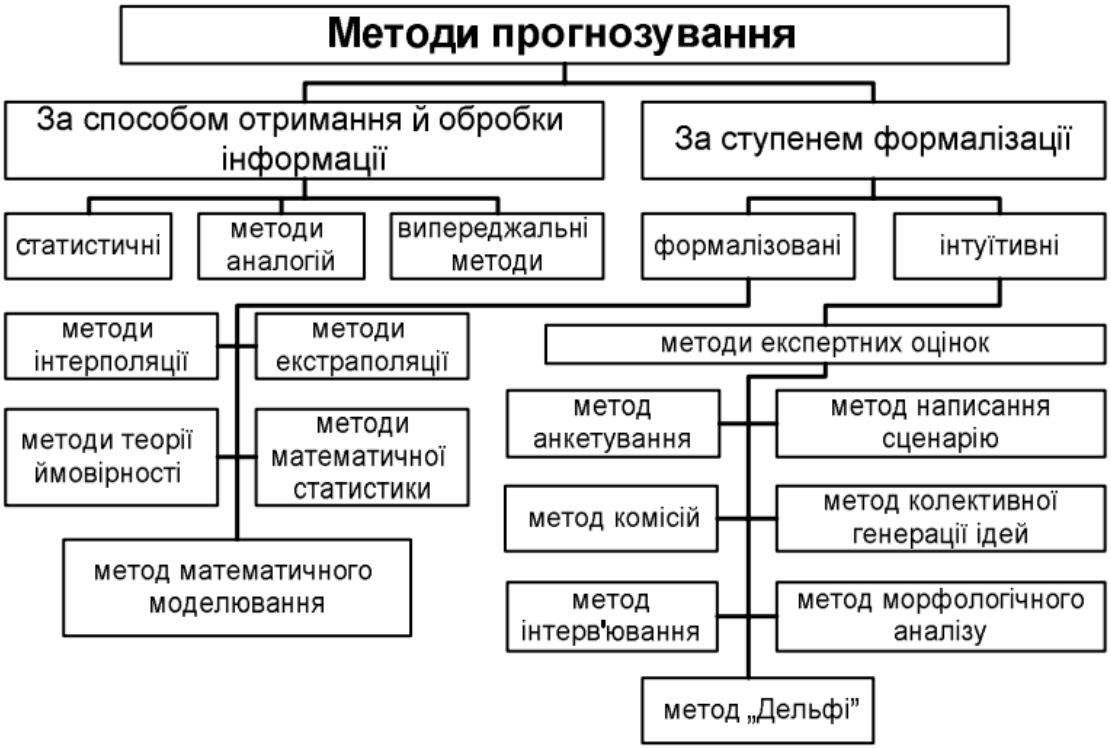

Рисунок 1.2 – Класифікація методів прогнозування

Методи машинного навчання, такі як ARIMA (авторегресивне інтегроване ковзне середнє), LSTM (довгокороткочасна пам’ять) і DecisionForest, були застосовані для прогнозування розвитку пандемії та прийняття політичних рішень [28]. Кожна модель має свої сильні та слабкі сторони, а їх застосування залежить від таких факторів, як тип даних, бажаний рівень точності та конкретний варіант використання. Даний дипломнийпроєкт спрямований на оцінку цих моделей і визначення їх придатності для різних ситуацій.

1.3 Аналіз особливостей прогнозування рядів динаміки

Основні показники пандемії COVID-19 є змінними у часі, які можна представити у вигляді рядів динаміки.

Прогнозування рядів динаміки - це процес аналізу даних рядів динамікиза допомогою статистики та моделювання для складання прогнозів

та прийняття обґрунтованих стратегічних рішень.

Ряд динаміки – це послідовність спостережень, зроблених впродовж різних моментів часу. Кожне спостереження в ряду динамікипов'язане з конкретним часовим пунктом і містить значення певної змінної або показника. Такі ряди використовуються для вивчення та аналізу залежностей, трендів та патернів, що спостерігаються в часових даних.

Ряди динаміки можуть мати різні періодичності та часові інтервали між спостереженнями. Аналіз часових рядів дозволяє виявити тренди, сезонність, циклічність та інші залежності, що допомагають в прогнозуванні майбутніх значень.

Для отримання наближених до точних прогнозів данихпотрібнозібрати дані рядів динаміки протягом певного періоду, проаналізувати ці дані, а потім побудувати модель, яка допоможе зробити прогноз. Проте, для цього процесу необхідно дотримуватися певних правил, які допоможуть досягти наближених до точних результатів:

- деталізованості - згідно з цим правилом, чим більш деталізовані прогнози, тим більш точні вони будуть. Це пояснюється тим, що деталізовані дані мають менше варіації і, відповідно, менше шуму;
- частоти - для отримання більш точних прогнозів необхідно часто оновлювати дані, щоб врахувати будь-яку нову доступну інформацію;
- горизонту - рекомендується уникати занадто довготривалих прогнозів. Замість цього, прогнози слід робити на короткий період часу, щоб отримати більш точні результати.

Основні компоненти прогнозування часових рядів допомагають розкрити структуру та характеристики ряду. До них відносять тренд, сезонність, циклічність та шум (рис. 1.3).

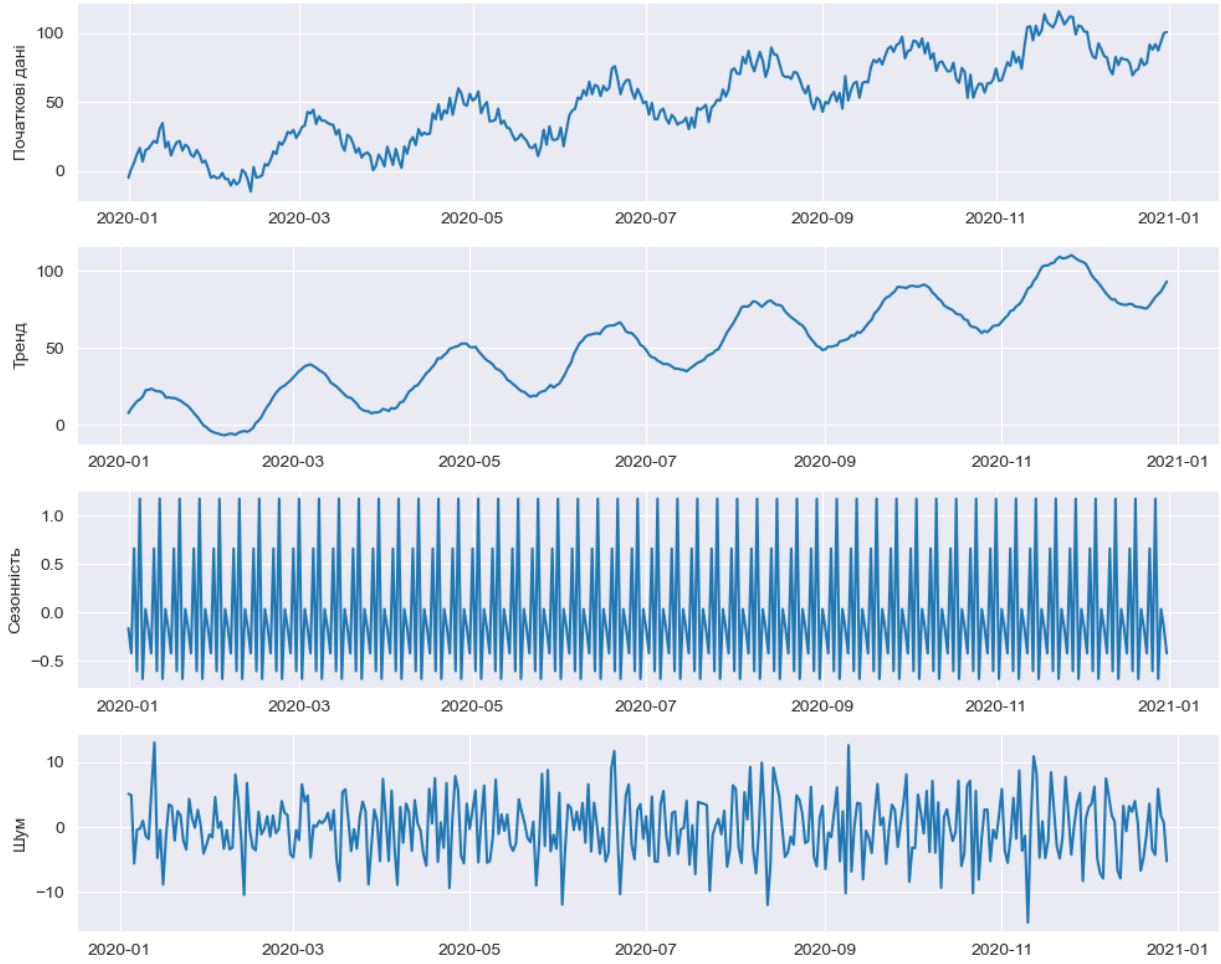

Рисунок 1.3 –Приклад декомпозиції основних компонентів часових рядів

1. Тренд - представляє собою довгострокове зростання або спад значень в ряді динаміки. Він показує загальну спрямованість ряду протягом тривалого періоду. Тренд може бути лінійним (пряма лінія), нелінійним (крива форма) або навіть відсутнім.

2. Сезонність - виявляється в повторюваних патернах або коливаннях, які спостерігаються в межах коротких періодів, таких як дні, тижні, місяці або роки. Ці коливання можуть бути пов'язані з природними факторами, соціальними звичаями, сезонністю або іншими факторами.

3. Циклічність - відображає тривалі періодичні коливання в часовому ряді, які не повторюються з регулярністю сезонності. Ці коливання можуть мати різну тривалість та амплітуду. Цикли можуть бути пов'язані з економічними циклами, бізнес-циклами або іншими факторами.

4. Шум - представляє собою непередбачувані, випадкові флуктуації, які не можуть бути пояснені трендом, сезонністю або циклічністю. Цей компонент включає в себе вплив випадкових подій, помилок вимірювання, незначні випадкові зміни та інші фактори, які не можуть бути систематично враховані в моделі.

Ці компоненти допомагають розуміти характер ряду, виявляти залежності та регулярності, що є необхідним для точного прогнозування.

Отримання вищезазначених компонентів відбувається шляхом розкладання часового ряду на складові. В рамках декомпозиції часового ряду виокремлюються два основних типи: адитивна сезонна декомпозиція та мультиплікативна сезонна декомпозиція.

При адитивній сезонній декомпозиції окремі компоненти ряду динамікидодаються для отримання початкового ряду. За своєю суттю, це означає, що вплив сезонності додається до тренду та стохастичної складової.

З іншого боку, мультиплікативна сезонна декомпозиція передбачає множення окремих компонентів, щоб отримати початковий ряд. Це означає, що сезонна складова множиться на тренд та стохастичну складову. Вибір одного типу декомпозиції над іншим залежить від того, чи має залишкова складова будь-який впорядкований зразок. Однак важливою метою є забезпечення того, щоб залишкова складова була просто випадковими флуктуаціями, які не мають жодного систематичного впливу.

Основні види рядів динаміки поділяються на два типи: стаціонарні та нестаціонарні (рис. 1.4).

Стаціонарні часові ряди характеризуються стабільними статистичними властивостями, такими як постійна середня і дисперсія, які не залежать від часу. Вони не мають трендів, сезонності або циклічних змін. Стаціонарні ряди є простішими для моделювання та прогнозування, оскільки їх можна апроксимувати стохастичними процесами.

Нестаціонарні часові ряди мають змінні статистичні характеристики, такі як середня або дисперсія, які змінюються з часом. Вони можуть

включати тренди, сезонні зміни, циклічність та інші динамічні ефекти. Нестаціонарні ряди відображають систематичні зміни в часових даних, що можуть бути спричинені різними факторами. Моделювання та прогнозування нестаціонарних часових рядів може бути складним завданням, оскільки потрібно враховувати змінність параметрів у часі та робити коригування для залежності від тренду чи сезонності.

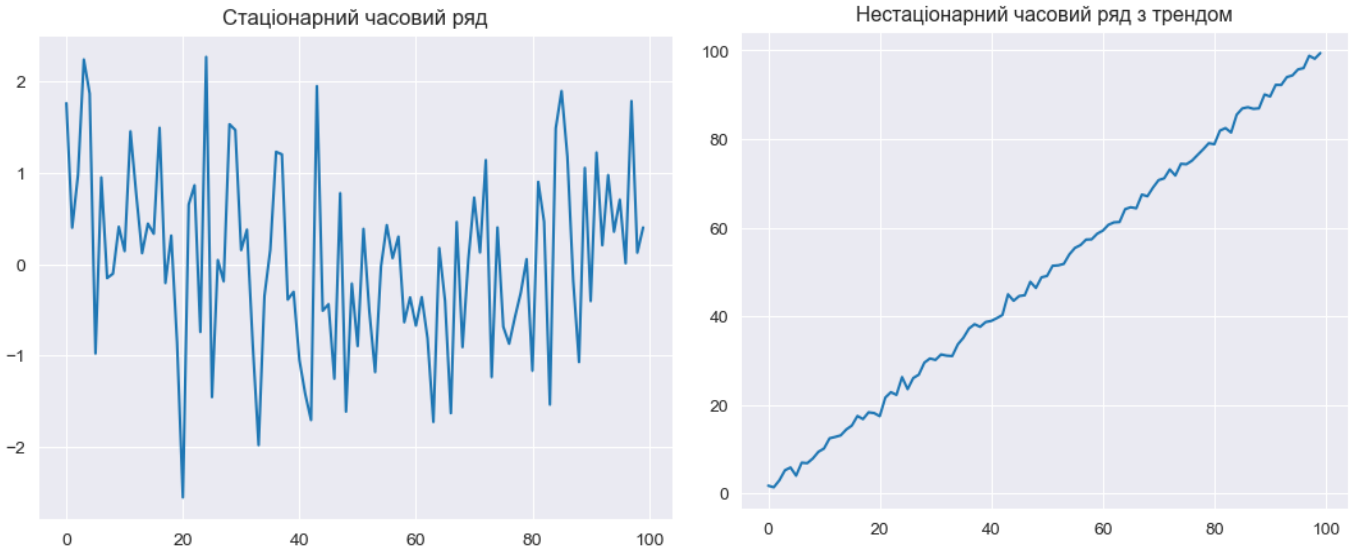

Рисунок1.4 –Приклад стаціонарного та нестаціонарного ряду динаміки з трендом

Таким чином, прогнозування рядів динаміки– це процес передбачення майбутніх значень рядів динамікина основі доступних даних про їх минулі значення. Основна ідея прогнозування полягає в тому, що часові ряди мають певні закономірності та патерни, які можна використовувати для розуміння та прогнозування майбутніх змін.

Один з головних принципів прогнозування рядів динаміки- це те, що майбутні значення часового ряду часто залежать від його попередніх значень.

Основний підхід до прогнозування рядів динамікивключає аналіз минулих значень ряду, виявлення закономірностей та трендів, і побудову математичної моделі, яка описує ці залежності.

В дипломному проєкті проведено аналіз основних методівпрогнозування рядів динаміки, а саме:

1. Naivemethod (Простий метод) - цей метод базується на припущенні, що майбутні значення будуть такі ж, як останнє спостережене значення часового ряду (рис.1.5). Він найпростіший у застосуванні, але не враховує будь-які зміни або тренди в даних.

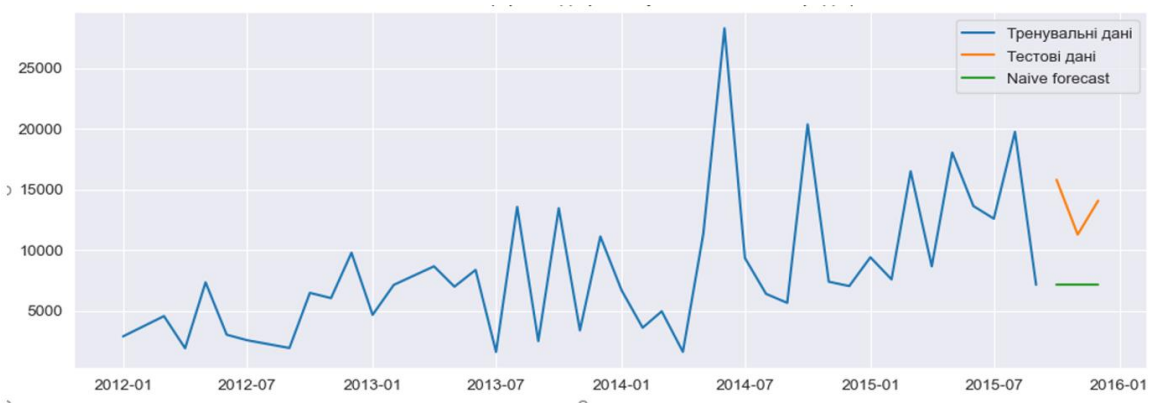

Рисунок 1.5 – Приклад прогнозування рядів динаміки Naivemethod

2. Simpleaveragemethod (Метод простого середнього) - цей метод використовує середнє значення попередніх спостережень для прогнозування майбутніх значень (рис.1.6). Він підходить, коли часовий ряд має сталу або майже сталу середню величину.

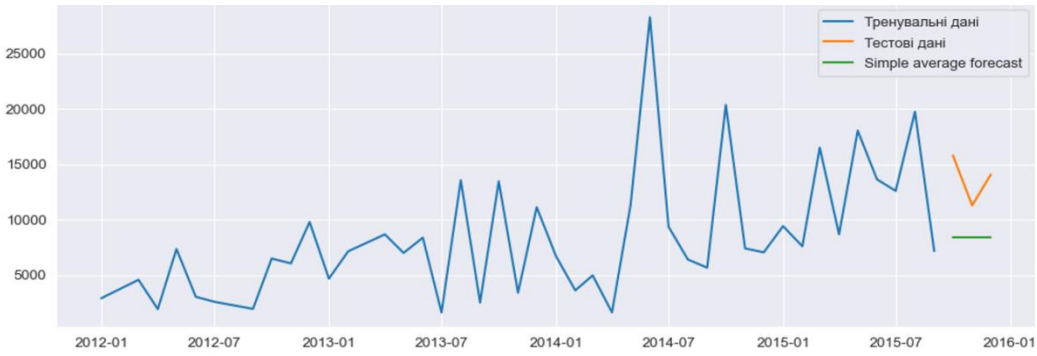

Рисунок 1.6 – Приклад прогнозування рядів динаміки Simpleaveragemethod

3. Simple movin gaverage method(Метод простої ковзної середньої) -

цей метод використовує середнє значення останніх k спостережень для прогнозування наступного значення (рис.1.7) Він дозволяє враховувати останні зміни в часовому ряді і підходить для рядів з випадковими коливаннями.

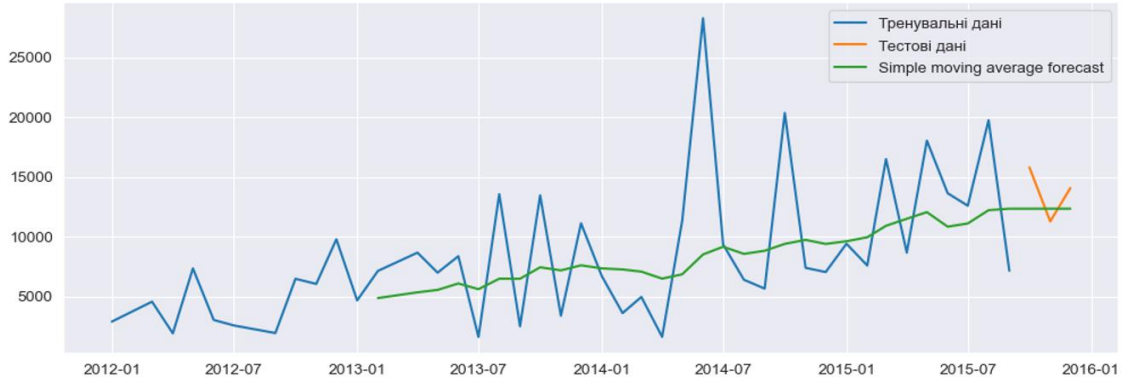

Рисунок 1.7 – Приклад прогнозування рядів динаміки Simplemovingaveragemethod

4. Simple Exponential Smoothing Method(Метод простого експоненційного згладжування) - цей метод використовує зважене середнє значення попередніх спостережень для прогнозування майбутніх значень (рис1.8). Він приділяє більшу вагу недавнім спостереженням, що дозволяє враховувати тренди та зміни в ряді динаміки.

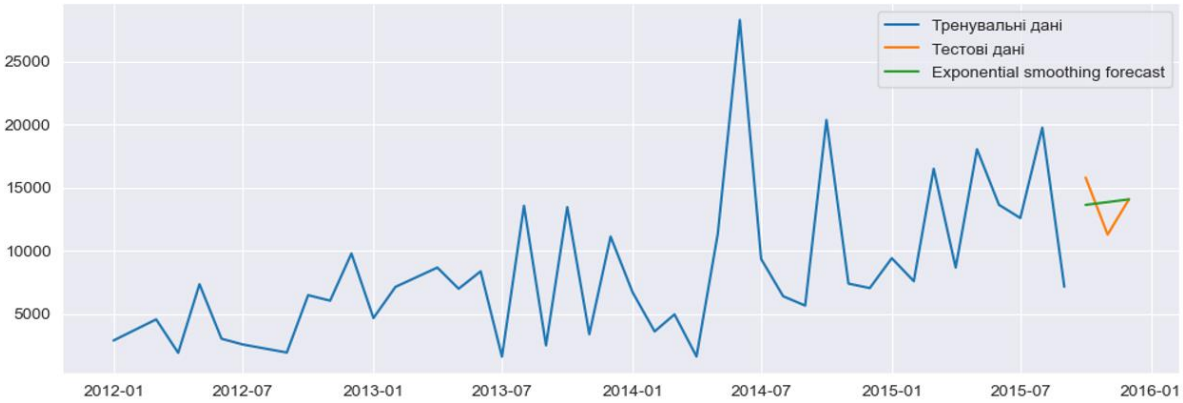

Рисунок 1.8 – Приклад прогнозування рядів динаміки Simple Exponential Smoothing Method

5. Holt'smethodwithtrend (Метод Хольта з трендом) -цей метод

використовує згладжування та тренд для прогнозування майбутніх значень (рис.1.9). Він враховує як рівень ряду динаміки, так і його тренд, що дозволяє виявляти та прогнозувати зміни в тренді.

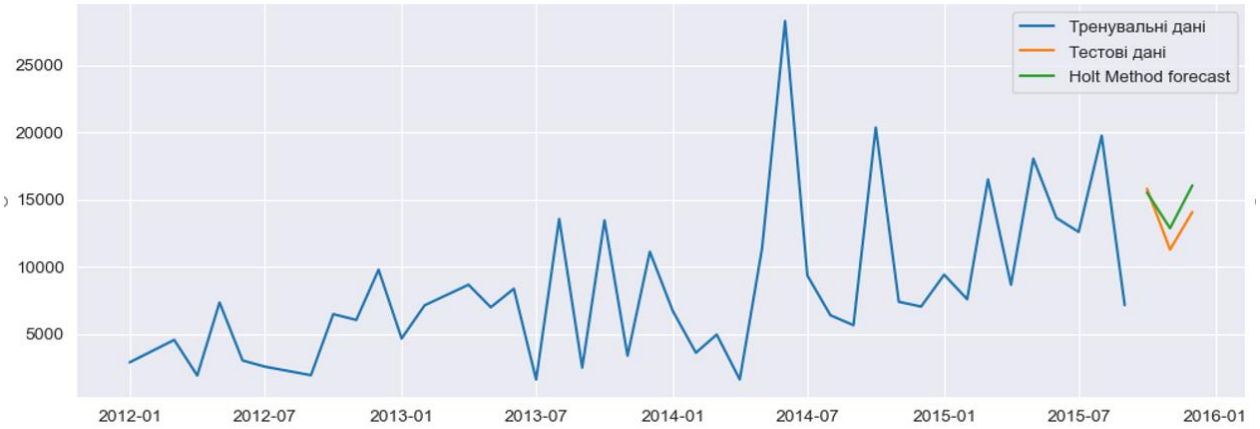

Рисунок 1.9 – Приклад прогнозування рядів динаміки Holt'smethodwithtrend

6. HoltWinter'smethod (Метод Хольта-Вінтерса) - цей метод використовує згладжування, тренд та сезонність для прогнозування майбутніх значень (1.10). Він дозволяє враховувати як рівень, тренд, так і сезонні коливання в часовому ряді. Цей метод особливо ефективний для прогнозування в часових рядах з вираженою сезонністю, наприклад, щомісячні продажі або квартальні фінансові показники.

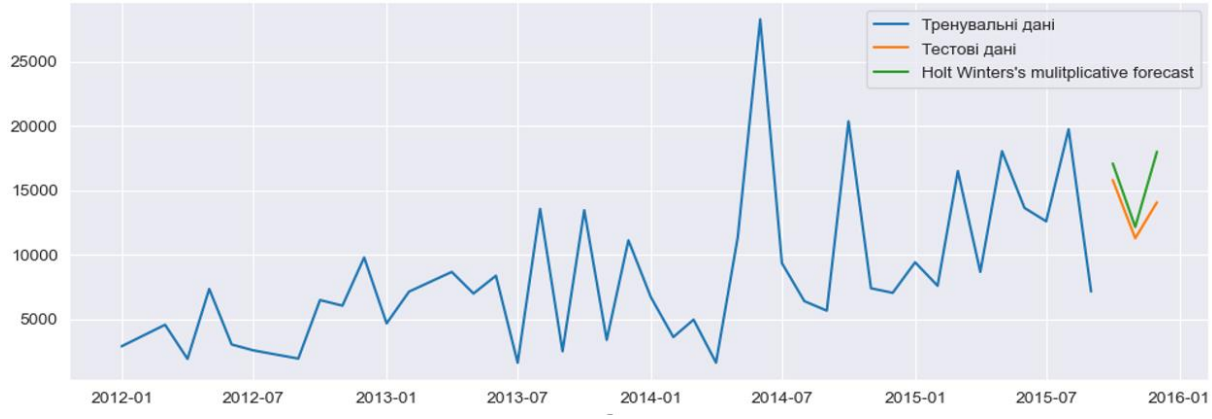

Рисунок 1.10 – Приклад прогнозування числових рядів HoltWinter'smethod

Таким чином, прості методи, такі як простий метод та метод простого середнього, є швидкими та простими у застосуванні і підходять для швидкого оцінювання загальних тенденцій в часовому ряді.

Складніші методи, такі як метод Хольта з трендом і метод Хольта-Вінтерса, дозволяють враховувати як тренд, так і сезонні коливання. Метод Хольта з трендом використовується для моделювання часових рядів з трендом, тоді як метод Хольта-Вінтерса додає ще й ефект сезонності. Ці методи дозволяють краще адаптуватися до складних змін у часових рядах та прогнозувати майбутні значення з урахуванням тренду та сезонних змін.

Методи авторегресії використовують попередні значення часового ряду для прогнозування майбутніх значень. Вони базуються на припущенні, що майбутні значення залежать від попередніх значень та можуть бути виражені у вигляді авторегресійних моделей. Ці методи підходять для прогнозування часових рядів з довготривалими залежностями та трендами, дозволяючи моделювати складні динамічні зміни в часових рядах.

Вибір правильного методу прогнозування залежить від унікальних властивостей даних та конкретних цілей прогнозування. Користувач повинен аналізувати тип часового ряду, наявність трендів, сезонності та інших особливостей для визначення найкращого підходу. Наприклад, якщо даний часовий ряд має чітко виражений тренд та сезонні коливання, то використання методу Хольта-Вінтерса може бути доцільним. За умови, якщо тренд відсутній, але є наявність автокореляції, то використання авторегресійних методів може бути виправданим.

Важливо враховувати контекст і особливості кожного часового ряду при виборі методу прогнозування.

1.4 Аналіз існуючих досліджень щодо оцінки епідемічної небезпеки поширення COVID-19

З початку XX століття прогнозування інфекційних хвороб стало важливою та активно розвиваючою сферою. Кількість наукових робіт,

присвячених цій темі, значно зросла завдяки впровадженню інформаційних систем та наявності великих обсягів статистичних даних для аналітичних досліджень.

Відмінність епідемії COVID-19 полягає в недостатній статистиці минулих років, що створює труднощі для використання наявних даних про розвиток епідемії, включаючи інформацію з інших країн світу. Багатобічні наукові дослідження направлені на розробку методів прогнозування поширення нового вірусу в короткостроковому та довгостроковому періоді.

11 лютого 2020 року Всесвітня організація охорони здоров'я організувала Глобальний дослідницький та інноваційний форум, щоб визначити ключові напрямки досліджень у контексті боротьби з епідемією [29]. Математичні моделі, які прогнозують динаміку розповсюдження COVID-19 в режимі реального часу, були опубліковані в наукових журналах та на спеціалізованих онлайн-платформах [30-34].

В одній з цих статей [31]приведені прогнози щодо масштабів епідемії в Ухані та інших містах, включаючи міста за межами Китаю. Автори досліджують внутрішні та зовнішні ризики для здоров'я населення від епідемій, використовуючи модель SEIR (Susceptible-ExposedInfectious-Recovered) та розглядаючи різні сценарії профілактичної дії.

В статті [32] аналізується динаміка поширення коронавірусу в Індії за допомогою системи диференціальних рівнянь з постійними коефіцієнтами та концепції базисного репродукційного числа. Автори використовують принцип максимуму Понтрягіна для розв'язання задачі оптимізації профілактичних заходів. Також проводиться аналіз рішень системи диференціальних рівнянь в моделі SIRD (Susceptible-InfectedRecovered-Deaths), підкреслюючи, що незважаючи на деяку грубість моделі SIRD, вона дозволяє відображати загальні особливості розвитку епідемії та прогнозувати динаміку в режимі реального часу.

В одному з досліджень [33] представлена методика прогнозування піку нових випадків захворювання та кількості смертей за весь період епідемії в

Італії. Автори цієї статті висувають гіпотезу, що будь-яка країна, яка переживає епідемічний вибух, може бути розглянута, як середовище, в якому різні групи населення взаємодіють за загальними правилами, незалежно від географічних варіацій.

Проте, вони також вказують, що траєкторії, сформовані моделлю SIRD, відзначаються високою чутливістю до зміни її параметрів, що в свою чергу може суттєво погіршити якість прогнозування, особливо на великій часовій шкалі.

У дослідженні [34] представлений просторово-часовий підхід, який базується на використанні броунівського руху, та модель SBDiEM (StereographicBrownianDiffusionEpidemiologyModel) для прогнозування динаміки інфекції. Ці інструменти можуть бути використані для створення системи моніторингу та протидії пандеміям з використанням технологій штучного інтелекту.

Автори дослідження [34] використовують кількісний образ розповсюдження COVID-19 в Китаї як тестовий приклад, а також дані про інфекцію з восьми різних країн для оцінювання еволюції епідеміологічного процесу в кожній з них. Вони використовують гаусову гіпотезу поширення вірусу і базову модель SIR (Susceptible-Infected-Recovered) для свого підходу.

Враховуючи складності, пов'язані з використанням детермінованих моделей типу SIR, SEIR, SIRD для прогнозування динаміки розповсюдження COVID-19, які ґрунтуються на механізмах передачі вірусу від особини до особини та використовують оцінки параметрів розповсюдження відомих вірусів, науковці акцентують увагу на пошуку альтернативних методів.

Альтернативний підхід, який використовується для опису розвитку епідемії, є агентне моделювання. В цьому випадку динаміка кожного представника популяції розраховується окремо, а імовірність інфікування в кожній конкретній локації обчислюється на основі кількості інфекційних переносників, присутніх там. Такий підхід дозволяє безпосередньо

оцінювати ефективність карантинних заходів і чітко визначати шляхи поширення інфекції.

Один з найбільш природних способів математичної реалізації цього підходу є метод Монте-Карло, який широко використовується для моделювання поширення нейтронів у різних системах. У цьому випадку процес поширення інфекції формалізується таким чином, що його динаміка подібна до динаміки поширення нейтронів у надкритичній системі.

Хоча диференціальна модель SEIRD є простою, вона відіграла важливу роль у встановленні основних законів розвитку епідемії. Статистична агентна модель, хоча має деяку спрощеність у моделюванні поведінки людей, дозволяє аналізувати такі фактори, як запровадження карантину для окремих соціальних груп або різних сфер діяльності (робота, транспорт, магазини). На жаль, ця модель має деякі обмеження. Для достовірного моделювання факторів необхідні відповідні початкові дані, такі як чисельність населення та розподіл за соціальними групами, а також завантаженість різних видів транспорту та магазинів. Незважаючи на відсутність детальних статистичних даних, побудована модель демонструє непогані прогностичні характеристики.

Отримані розрахункові дані підтверджують, що обмежувальні заходи, які вживає влада, дають позитивні результати, і в деяких регіонах кількість заражень починає зменшуватися. Однак це не означає, що після зняття карантинних обмежень епідемія не може відродитися знову. Тим не менше, карантинні заходи дозволяють досягти двох важливих цілей. По-перше, знижуючи темпи зараження під час карантину, ми можемо зменшити максимальне навантаження на медичну систему та врятувати більше життів. По-друге, карантин затримує поширення епідемії, надаючи вченим-вірусологам необхідний час для детального вивчення властивостей коронавірусу, розробки ефективних методів лікування та створення вакцини, яка допоможе імунізувати населення та завершити цю епідемію.

Після проведення короткого огляду підходів та математичних методів прогнозування рівня захворюваності COVID-19, можна зробити висновок, що не існує жодного універсального методу, який би задовольняв усім вимогам прогнозування. Дослідження показали, що кожний підхід та метод мають свої переваги, недоліки та обмеження у застосуванні. Варто відзначити, що ці методи базуються або на кореляційно-регресійних моделях, або на трендах, для представлення яких обираються найбільш відповідні екстраполяційні залежності. Для подальшого прогнозування рівня захворюваності COVID-19 у даному дипломному проєкті були обрані моделі ARIMA, LSTM таDecisionForest.

1.6 Розроблення діаграми бізнес-процесів системи «Розрахунок прогнозів»за стандартом IDEF0

Діаграма IDEF0 моделює бізнес-процеси, за допомогою яких буде автоматизована робота інформаційного модулю оцінки епідемічної небезпеки поширення COVID-19.

Контекстна діаграма бізнес-процесу за стандартом IDEF0 наведена на рис.

У процесі аналізу предметної області, була складена контекстна діаграма системи «Інформаційний модуль оцінки епідемічної небезпеки поширення COVID-19» («Розрахунок прогнозів») у стандарті IDEF0 (рис. 1.11).

Для наведеної на рис.1.11 діаграми були визначені наступні інтерфейсні дуги:

Вхід: введення даних (дані протестованих осіб, хворих, тяжкохворих, госпіталізованих, померлих, вакцинованих);

Вихід: виведення прогнозу, статистичні дані, звіти.

Управління: рекомендації ВОЗ, Законодавство України в сфері охорони здоров’я, медичні регламенти, Положення про захист персональних даних.

Механізм: менеджер, база даних, медична інформаційна система.

Декомпозиція контекстної діаграми реалізована на визначенні наступних бізнес-процесів: обробка даних, вибір діапазону прогнозування епідемічної небезпеки поширення COVID-19, вибір методу прогнозування, результат прогнозу (рис. 1.12).

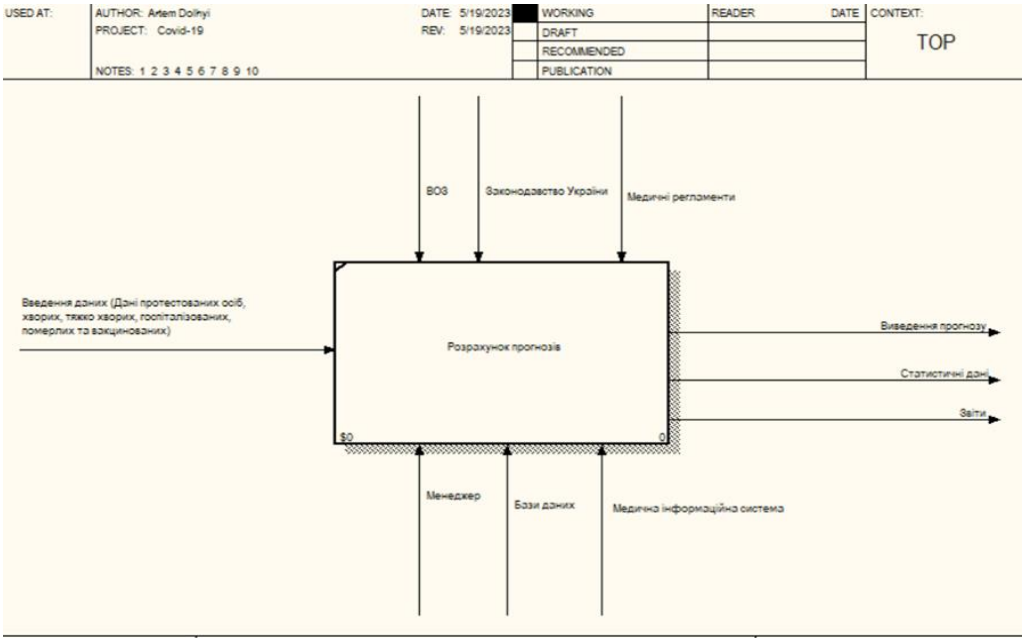

Рисунок – 1.11 Контекстна діаграма бізнес-процесів «Інформаційний модуль оцінки епідемічної небезпеки поширення COVID-19» (Розрахунок прогнозів) у стандарті IDEF0

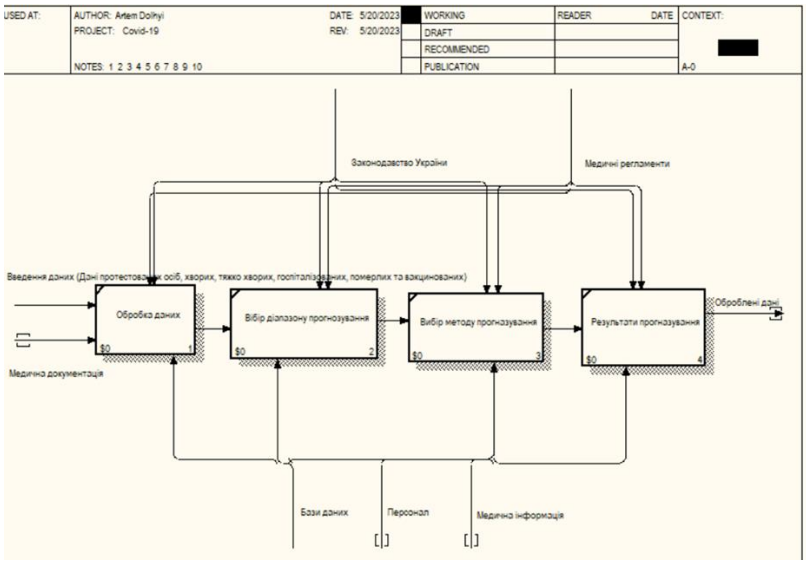

Рисунок 1.12 – Декомпозиція роботи 1-го рівня «Обробка даних»

Бізнес-процес 1-го рівня декомпозиції «Обробка даних» має такі інтерфейсні дуги:

Вхід: введення даних (дані протестованих осіб, хворих, тяжко хворих, госпіталізованих, померлих, вакцинованих), медична документація.

Вихід: оброблені дані.

Управління: Законодавство України, медичні регламенти.

Механізм: персонал, база даних, медична інформаційна система.

Бізнес-процес1-го рівня декомпозиції «Обробка даних» має такі інтерфейсні дуги:

Вхід: оброблені дані.

Вихід: діапазон прогнозування.

Управління: Законодавство України, медичні регламенти.

Механізм: персонал, база даних, медична інформаційна система.

Бізнес-процес1-го рівня декомпозиції «Вибір методу прогнозування» має такі інтерфейсні дуги:

Вхід: оброблені дані, вибір діапазону

Вихід: діапазон прогнозування, метод прогнозування (ARIMA, LSTM, DecisionForest).

Управління: Законодавство України, медичні регламенти.

Механізм: персонал, база даних, медична інформаційна система.

Бізнес-процес1-го рівня декомпозиції «Результат прогнозування» має такі інтерфейсні дуги:

Вхід: оброблені дані, вибір діапазону, вибір методу прогнозування

Вихід: звіт, завантаження файлу, прийняття рішень

Управління: Законодавство України, медичні регламенти.

Механізм: персонал, база даних, медична інформаційна система.

1.6 Постановка задачі дослідження

Метою дипломного проєкту є розробка інформаційного модулю оцінки епідемічної небезпеки поширення COVID-19, з урахуванням зовнішніх факторів за допомогою моделей машинного навчання.

Для розроблення інформаційного модулю оцінки епідемічної небезпеки поширення COVID-19 необхідно виконатинаступні завдання, а саме:

- підготувати та проаналізувати доступні ряди динаміки з даними о кількості осіб, що захворіли, кількості тих, хто помер, одужав чи пройшов курс вакцинації;
- проаналізувати моделі на основі ARIMA, LSTM, DecisionForest;
- навчити моделі на наявних наборах даних;
- порівняти та оцінити різні моделі прогнозування машинного навчання, такі як ARIMA, LSTM та DecisionForests для прогнозування поширення COVID-19;
- порівняти та проаналізувати отримані результати;
- визначити переваги та недоліки кожної моделі прогнозування та визначити їх застосовність у різних ситуаціях та типах даних.

2 АНАЛІЗ МАТЕМАТИЧНИХ МОДЕЛЕЙ ДЛЯ ОЦІНКИ ЕПІДЕМІЧНОЇ НЕБЕЗПЕКИ ПОШИРЕННЯ COVID-19

2.1 Аналіз ефективності застосування моделі ARIMA для оцінки епідемічної небезпеки поширення COVID-19

ARIMA (що означає "Авторегресійна інтегрована ковзна середня") є потужною статистичною моделлю, яка широко використовується для прогнозування часових рядів. Ця модель відома своєю здатністю ефективно аналізувати складні дані рядів динаміки та відтворювати як короткострокові, так і довгострокові закономірності, враховуючи тенденції та сезонність[35]. Вона також здатна моделювати як лінійні, так і нелінійні залежності в даних, що робить її потужним інструментом для моделювання складних часових рядів. Ця модель також є відносно простою у застосуванні.

Основна ідея ARIMA полягає в розкладанні часового ряду на три компоненти:

- авторегресію (AR);
- інтеграцію (I);
- ковзну середню (MA).

Комбінуючи ці компоненти, модель ARIMA може точно прогнозувати майбутні значення часового ряду.

ARIMA особливо корисна для аналізу та прогнозування стаціонарних часових рядів, де середнє та дисперсія даних залишаються сталими з плином часу та спостерігаються закономірності, такі як тенденції та сезонність. Наприклад, вона може бути використана для прогнозування поширення пандемії COVID-19, аналізу впливу попередніх пандемій на здоров'я громад, а також для прогнозування та аналізу інших показників, таких як кількість захворілих, госпіталізації та смертності.

Додатково, ARIMA може бути застосована для прогнозування та аналізу цін на фінансових ринках, прогнозування продажів товарів та послуг,

а також для моделювання та прогнозування інших економічних та соціальних показників.

ARIMA поєднує три основні компоненти: авторегресійну (AR), інтегровану (I) компоненти та ковзне середнє (MA).

1. Авторегресійна (AR) компонента - ця компонента відображає залежність між спостереженням та певною кількістю запізнілих спостережень. У моделі AR поточне значення часового ряду є лінійною комбінацією його минулих значень. Модель AR передбачає поточне значення часового ряду на основі його минулих значень. Порядок AR моделі, позначений як p, вказує на кількість використовуваних запізнілих спостережень для прогнозу. Наприклад, модель AR(2) використовує два попередніх спостереження для прогнозу поточного значення.

2. Інтегрована (I) компонента - ця компонента використовується для моделювання залежності між спостереженнями та помилками, особливо для роботи з нестаціонарними часовими рядами. Шляхом виконання послідовного віднімання між сусідніми спостереженнями, інтегрована компонента забезпечує стаціонарність часового ряду. Порядок віднімання, позначений як d, вказує на кількість разів, які застосовується процес віднімання між сусідніми спостереженнями. Це дозволяє видалити тренди та сезонність з даних.

3. Ковзне середнє (MA) -ця компонента моделює залежність між спостереженням та ковзним середнім попередніх помилок. У моделі MA поточне значення часового ряду є лінійною комбінацією попередніх помилок. Шляхом врахування попередніх помилок, модель MA передбачає поточне значення часового ряду. Порядок моделі ковзного середнього, позначений як q, вказує на кількість попередніх помилок, що використовуються для прогнозу. Наприклад, модель MA(2) використовує помилки з двох попередніх часових точок для прогнозу поточного значення.

Разом, параметри моделі ARIMA позначаються як ARIMA(p, d, q). Вибір і налаштування значень p, d і q здійснюється через процес вибору та

налаштування моделі, який використовує статистичні методи для визначення оптимальних значень параметрів, що мінімізують помилку моделі.

2.1.1 Математична постановка моделі ARIMA

Для моделі ARIMA (p, d, q), математична формула виглядає наступним чином [36]:

$$\text{ARIMA}(p,d,q): Y(t) = c + \sum(\phi_i \cdot Y(t-i)) + \sum(\theta_j \cdot e(t-j)) + e(t), \quad (2.1)$$

де $Y(t)$ - значення часового ряду в момент часу t, c– константа;

ϕ_i- коефіцієнти авторегресії;

θ - коефіцієнти ковзного середнього;

$e(t)$ - помилка моделі в момент часу t.

Основні компоненти ARIMA моделі можна математично представити як:

Авторегресія (AR):

$$Y(t) = c + \phi_1 \cdot Y(t-1) + \phi_2 \cdot Y(t-2) + \ldots + \phi_p \cdot Y(t-p) + e(t), \quad (2.2)$$

де$Y(t)$ представляє собою значення часового ряду в момент часуt;

c– константа;

$\phi_1, \phi_2, \ldots, \phi$ - коефіцієнти авторегресії;

$e(t)$ - помилка моделі в момент часу t.

Інтеграція (I) має вигляд:

$$Y(t) = Y(t) - Y(t-1) \quad (2.3)$$

де $Y(t)$ - значення часового ряду в момент часу t.

Ковзне середнє (MA) (q) має вигляд:

$$Y(t) = c + \theta_1 \cdot e(t-1) + \theta_2 \cdot e(t-2) + \ldots + \theta_q \cdot e(t-q) + e(t), \quad (2.4)$$

де $Y(t)$ - значення часового ряду в момент часу t;

c– константа;

$\theta_1, \theta_2, \ldots, \theta q$ - коефіцієнти ковзного середнього;

$e(t)$ - помилка моделі в момент часу t.

Таким чином, реалізувати алгоритм моделі ARIMA в Python можна наступним чином:

1. Встановити необхідні бібліотеки за допомогою pip;
2. Імпортувати необхідні модулі;
3. Завантажити дані;
4. Виконати необхіднідії попередньої обробки даних (видалення відсутніх значень, робота з викидами, проведення необхідних перетворень даних для забезпечення стаціонарності);
5. Розділити дані на навчальний та тестовий набори;
6. Побудувати графіки для перевірки наявності трендів та сезонності в даних;
7. Перевірити наявність стаціонарності та перетворення часового ряду на стаціонарний, якщо це необхідно, за допомогою диференціювання, розкладання або інших методів. Отримати значення параметра d. (Для перевірки стаціонарності використовують тест Дікі-Фуллера, який повертає тестову статистику та p-значення. Нульова гіпотеза тесту полягає в тому, що часовий ряд є нестаціонарним. Якщо p-значення менше рівня значущості (зазвичай 0,05), то можна відхилити нульову гіпотезу і прийняти рішення про стаціонарність часового ряду. Оскільки p-значення більше 0,05, необхідно зробити диференціювання часового ряду для досягнення стаціонарності.);
8. Побудувати графіки для перевірки стаціонарності після диференціювання;

9. Перевірити, чи дійсно диференціювання зробило часовий ряд стаціонарним, за допомогою тесту Дікі-Фуллера;

10. Використати функції plot_acf() та plot_pacf() з бібліотеки statsmodels для перевірки автокореляції в даних. Це також допоможе вибрати правильні значення параметрів (p, q) для моделі;

11. Оцінити параметри для (p, d, q) = (1, 1, 1);

12. Виконати пошук за сіткою, щоб знайти найкращі значення параметрів (p, d, q);

13. Запустити моделі ARIMA з найкращими параметрами.

Також існують розширені застосування ARIMA:

SARIMA (Сезонний ARIMA) є розширенням ARIMA, яке враховує сезонні закономірності в часових рядах. У моделі SARIMA, сезонна компонента позначається літерою "S" і дозволяє моделі враховувати повторювані сезонні патерни в даних, такі як щомісячні або щоквартальні коливання. Це досягається за допомогою додаткових сезонних диференційних, сезонних авторегресійних та ковзних середніх компонент. Застосування моделі SARIMA дозволяє зробити більш точні прогнози, враховуючи сезонні варіації в даних.

ARIMAX (Авторегресія з інтеграцією та ковзним середнім з екзогенними змінними) є розширенням ARIMA, яке дозволяє враховувати вплив додаткових змінних на часовий ряд. У моделі ARIMAX використовуються екзогенні змінні, які не є частиною самого часового ряду, наприклад, погодні умови або економічні показники. ARIMAX використовує кілька авторегресійних та ковзних середніх компонент для врахування залежностей між різними змінними та забезпечення більш точних прогнозів. Це дозволяє аналізувати та прогнозувати часові ряди, враховуючи зовнішні фактори, які впливають на дані.

2.1.2 Переваги та недолікизастосування ARIMA моделі для оцінки епідемічної небезпеки поширення COVID-19

Переваги ARIMA моделі для оцінки епідемічної небезпеки поширення COVID-19:

1. Гнучкість: ARIMA модель може бути застосована до аналізу та прогнозування динаміки захворюваності на Covid-19, оскількивона дозволяє враховувати як короткострокові, так і довгострокові зміни, а також виявляти наявність трендів та сезонності в даних. Це дозволяє моделі адаптуватися до змінних умов та надавати точні прогнози щодо поширення захворювання.

2. Ефективність: ARIMA модель є ефективним інструментом для прогнозування кількості захворілих на Covid-19. Вона використовує попередні значення кількості захворілих та помилки моделі для прогнозування майбутніх значень. Це дозволяє отримувати достатньо точні результати, які можуть бути корисними для планування та прийняття рішень щодо обмежень та заходів для контролю захворювання.

3. Інтерпретація: ARIMA модель базується на математичних формулах, що дозволяє легко інтерпретувати коефіцієнти моделі та зрозуміти, як вони впливають на прогнози. Наприклад, коефіцієнти AR (авторегресійна) та MA (рухомий середній) частин моделі можуть вказувати на залежність між кількістю захворілих на Covid-19 та їх попередніми значеннями або помилками моделі. Це дозволяє зрозуміти динаміку захворюваності та вплив різних факторів на її зміну.

Недоліки ARIMA моделі для оцінки епідемічної небезпеки поширення COVID-19:

1. Строгі припущення: ARIMA модель передбачає, що дані є стаціонарними та не мають змінюваних параметрів в часі. Однак, у випадку Covid-19, характеристики захворюваності можуть змінюватися від дня до дня, що порушує це припущення. Це може призвести до неточних прогнозів та незадовільної адаптації моделі до динаміки захворювання.

2. Залежність від правильного вибору параметрів: ARIMA модель має параметри p, d та q, які визначаються на основі аналізу даних та вимагають правильного вибору. Неправильне вибрання параметрів може призвести до неточних прогнозів та незадовільної моделювання динаміки захворюваності. Вибір оптимальних параметрів може вимагати експертного аналізу або використання методів оптимізації.

3. Обробка великого обсягу даних: ARIMA модель може бути обмеженою у використанні для дуже великого обсягу даних, оскільки вона вимагає розрахунків для кожного кроку прогнозу. Це може призвести до затримок у часі обчислення та значного споживання пам'яті.

Ураховуючи ці переваги та недоліки, ARIMA модель залишається потужним інструментом для аналізу та прогнозування динаміки захворюваності на Covid-19, проте її застосування повинно бути обґрунтованим та враховувати особливості конкретного набору даних.

2.2 Аналіз ефективності застосування моделі LSTM для оцінки епідемічної небезпеки поширення COVID-19

Нейронні мережі – це обчислювальні системи, основні принципи яких були сформовані на основі роботи людського мозку. Основна перевага полягає в тому, що нейронна мережа може «вчитися» та приймати рішення, не будучи безпосередньо запрограмованою на певну дію. Алгоритм дій, необхідних для вирішення задач прогнозування з використанням нейронних мереж представлено на рис. 2.1:

Звичайна нейронна мережа (англ. FeedforwardNeuralNetwork) є математичною моделлю, яка використовується для розпізнавання патернів та здійснення прогнозів на основі вхідних даних. Нейронна мережа складається з шарів нейронів, в яких кожен нейрон отримує вхідні дані, обчислює ваговану суму цих даних та застосовує активаційну функцію для генерації вихідного сигналу.

Рисунок 2.1 – Алгоритм дій для прогнозування з використанням нейронних мереж

Загальна структура мережі включає вхідний шар, приховані шари та вихідний шар. Ваги між нейронами в кожному шарі визначаються шляхом навчання моделі на тренувальних даних (рис. 2.2).

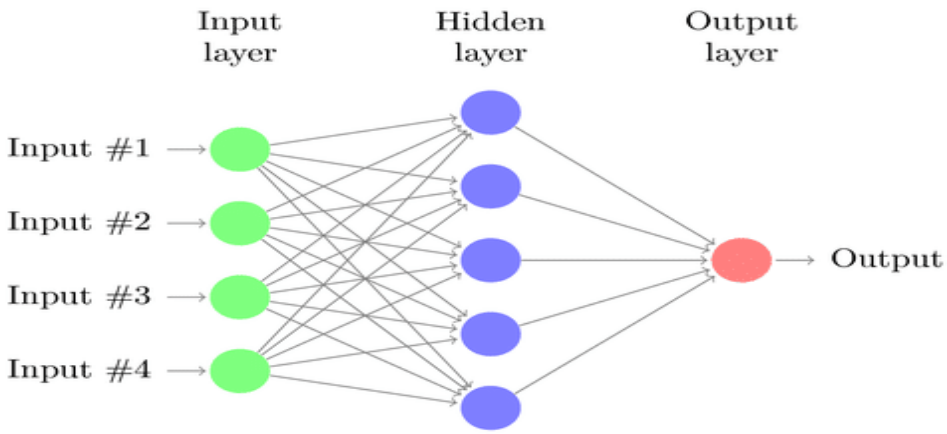

Рисунок 2.2 – Схематичне зображення штучної нейронної мережі

В контексті прогнозування поширення пандемій, таких як COVID-19, нейронні мережі можуть мати свої обмеження. Здатність цих моделей до узагальнення та точного прогнозування в цьому кейсі може бути лімітована, оскільки такі мережі зазвичай працюють краще з даними, які мають сталу структуру та незалежність від часу. Однак, у таких ситуаціях на допомогу можуть прийти рекурентні нейронні мережі (RNN), які здатні працювати з послідовними даними та враховувати контекстуальні залежності в часі.

Використання RNN дозволяє отримати більш точні прогнози та аналізувати складну динаміку поширення хвороби, що допомагає вживати ефективніші заходи контролю та боротьби з пандемією.

Рекурентні нейронні мережі (RNN) є потужним інструментом у галузі машинного навчання, здатними працювати з послідовними даними та враховувати контекстуальні залежності. Основна відмінність RNN від звичайних нейронних мереж полягає в тому, що вони мають зв'язки між нейронами, які утворюють замкнені цикли, дозволяючи передавати інформацію з одного часового кроку до наступного (рис.2.3). Це робить RNN дуже ефективними для роботи з даними, що мають часову залежність та послідовність [37].

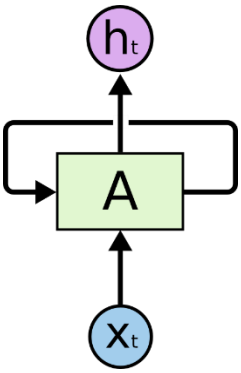

Рисунок 2.3 – Цикл рекурентних нейронних мереж

Однією з переваг RNN є їх здатність моделювати довгострокові залежності у послідовних даних[37]. Вони можуть запам'ятовувати і використовувати інформацію з попередніх часових кроків, що дозволяє їм зробити більш точні прогнози. Однак, у випадку прогнозування Covid-19 RNN мають свої обмеження та декілька проблем. Одна з основних проблем - це проблема довгострокового залежності (рис. 2.4) [37]. Хоча RNN здатні запам'ятовувати інформацію з попередніх кроків, вони мають тенденцію забувати цю інформацію протягом довгих послідовностей. Це може призвести до втрати цінних даних та зменшення точності прогнозування.

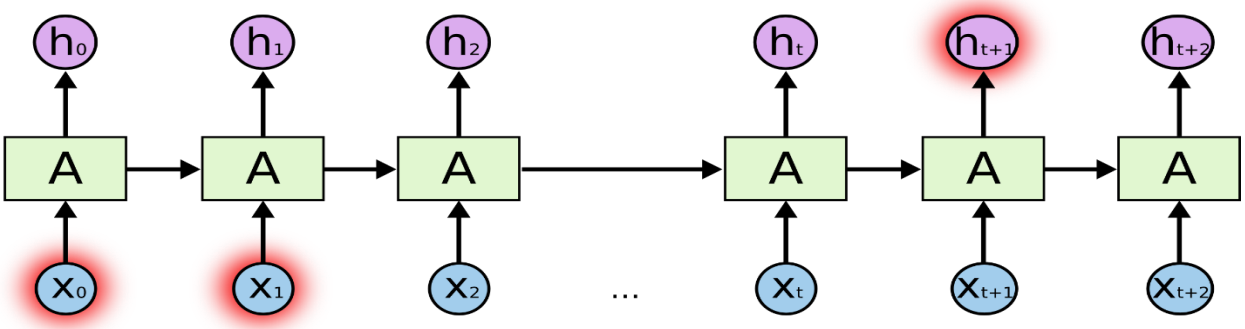

Рисунок 2.4 – Використання минулої інформації рекурентною нейронною мережею

Цю проблему вирішує модельLSTM(LongShort-TermMemory).Це тип рекурентної нейронної мережі (RNN), яка була спеціально розроблена для вирішення проблеми зникнення та вибуху градієнту, що виникають при навчанні RNN на довгих послідовностях даних. LSTM використовує

спеціальні механізми, що дозволяють мережі зберігати та використовувати інформацію на довгі періоди часу.

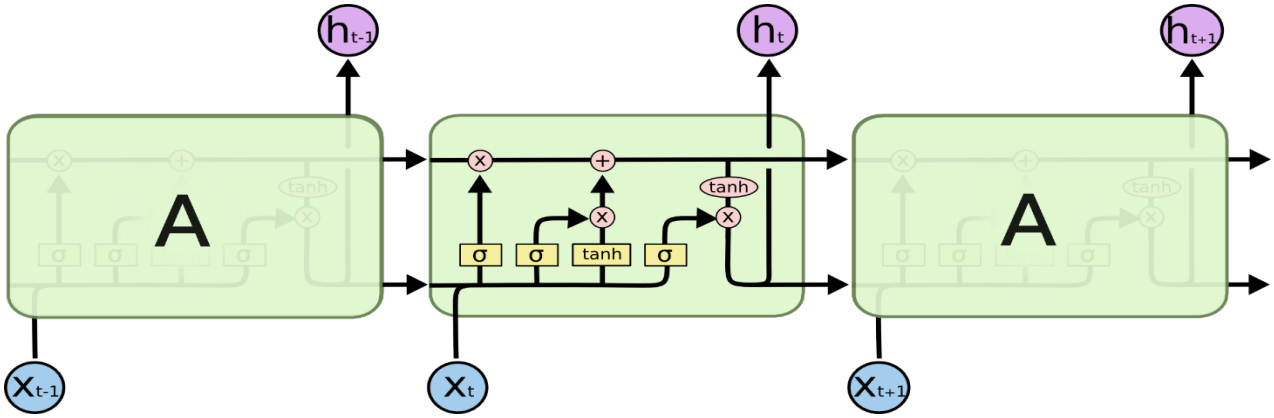

Рисунок 2.5–Структура розгорткиLSTM моделі

Основна ідея моделі LSTM (LongShort-TermMemory) полягає у вирішенні проблеми зникаючого градієнта, яка є типовою для традиційних рекурентних нейронних мереж (RNNs) [37]. Ця проблема, відома як проблема зникаючого градієнта (Vanishinggradientproblem), стосується значного зменшення величини градієнтів під час їх зворотного розповсюдження через послідовні часові кроки, що, у свою чергу, заважає процесу навчання (рис. 2.6).

Для боротьби з цією проблемою, модель LSTM включає в себе комірку пам'яті і три різні ворота: ворота забуття, вхідні ворота і вихідні ворота. Ці компоненти керують внутрішнім потоком інформації, дозволяючи моделі вибірково зберігати або відкидати не потрібну інформацію на різних часових кроках.

У контексті моделей LSTM, ворота - це обчислювальний механізм, який регулює прохід інформації, фактично виступаючи як перемикач, який визначає кількість інформації, яка передається на кожному часовому кроці (рисунок 2.6) [38].

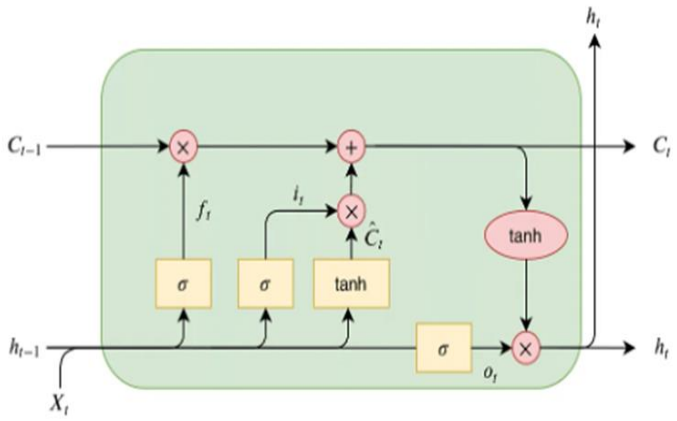

Рисунок 2.6 – Архітектура моделі LSTM

де Xt – вхідний тимчасовий шаг;

ht–вихід;

Ct – стан комірки;

Ft – забутий вентиль;

it – вхідний вентиль;

Ot –вихіднийвентиль.

Кожні ворота в моделі LSTM розроблені для керування потоком інформації таким чином, що дозволяє моделі розпізнавати довготривалі залежності в послідовних даних. Три основних типи воріт включають в себе:

1. Перший етап - ворота забуття (ForgetGate) (рис. 2.7). Ці ворота діють як регулятор пам’яті, визначаючи, яку інформацію з попереднього стану треба “запам’ятати” або“забути”. Вхідними даними для цих воріт є попередній прихований стан (previoushiddenstate) та поточний вхід (currentinput). Вони проходять через сигмоїдальну активаційну функцію (sigmoidactivationfunction), яка повертаєчисло в діапазоні від 0 до 1 для кожного елемента прихованого стану. Чим ближче значення до 0, тим більше інформації з попереднього стану “забувається”, і навпаки, чим ближче до 1, тим більше інформації зберігається для наступного кроку [38].

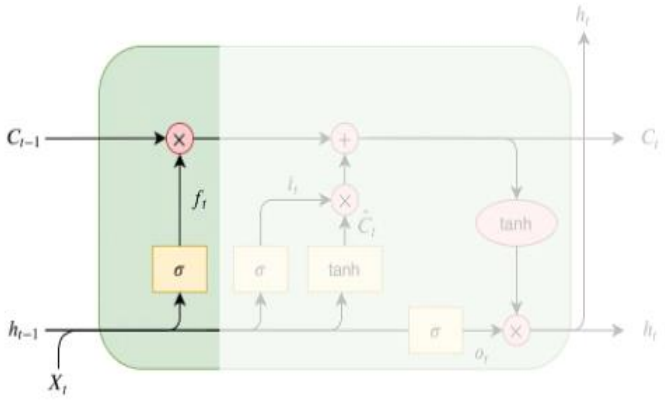

Рисунок 2.7 – Схематичне зображення першого етапу LSTM

2. Другий етап - вхідні ворота (InputGate) (ри.2.8). Ці ворота складаються з двох частин - власне вхідних воріт та кандидатського значення (candidatevalue). Вхідні ворота, подібно до воріт забуття, використовують попередній прихований стан і поточний вхід як вхідні дані, але їх завдання - визначити, яку нову інформацію треба зберегти. Сигмоїдальна активаційна функція виробляє значення для кожної одиниці інформації, вказуючи, яку частину потрібно зберегти. Кандидатське значення вираховується на основі поточного вводу і представляє “нове” оновлення комірки пам'яті [38].

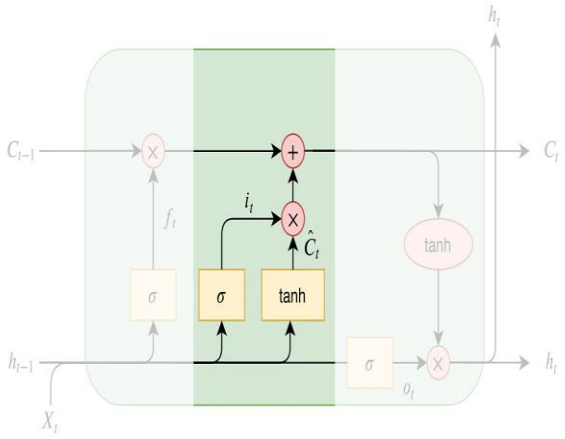

Рисунок 2.8 – Схематичне зображення другого етапу LSTM

3. Третій етап - вихідні ворота (OutputGate) (рис. 2.9). Ці ворота визначають, скільки інформації з комірки пам'яті слід використати для поточного прихованого стану. Так само, як і інші ворота, вони використовують попередній прихований стан і поточний вхід як вхідні дані.

Вихідні ворота застосовують сигмоїдальну активаційну функцію для визначення того, яка частина інформації з комірки пам'яті слід “вивести”. Таким чином, вони допомагають моделі вирішити, яку інформацію вважати важливою для поточного прогнозу [38].

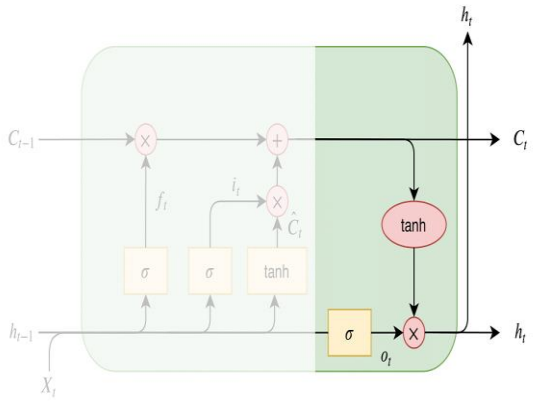

Рисунок 2.9 – Схематичне зображення третього етапу LSTM «Вихідні ворота»

Останній етап - Стан комірки (CellState) (рис. 2.10), ключовий компонент поряд з воротами, виступає як внутрішня пам'ять, яка передає інформацію на протязі послідовності. Цей стан дозволяє моделі захоплювати і зберігати важливу інформацію з минулих часових кроків, що дозволяє LSTM вивчати довготривалі залежності в даних. Змінюючи і модифікуючи стан комірки за допомогою воріт забуття, вхідних та вихідних воріт, модель може вибірково запам'ятовувати або забувати інформацію. Ця здатність зберігати важливу інформацію протягом довгих послідовностей є однією з ключових переваг архітектури LSTM [37].

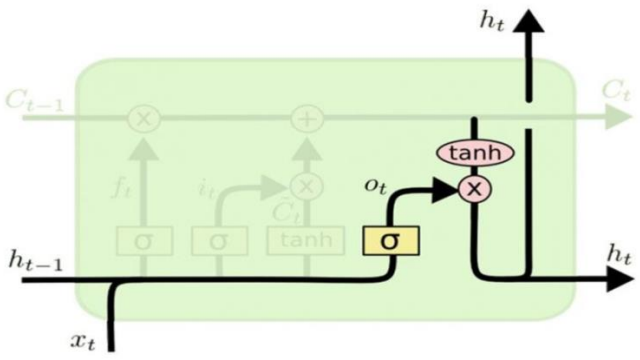

Рисунок 2.10 – Схематичне зображення останнього етапу LSTM моделі

Разом з воротами, стан комірки відіграє важливу роль в здатності LSTM розуміти та ефективно обробляти послідовні дані. Унікальна здатність моделі LSTM вибірково запам'ятовувати і забувати інформацію, зберігати нову важливу інформацію і виводити необхідну інформацію для прогнозування, суттєво сприяє визначенню довготривалих залежностей. Також, це допомагає керувати проблемами зникаючого або вибухового градієнта, і покращує точність у завданнях, що включають послідовні дані, таких як обробка природних мов, розпізнавання мови, і прогнозування рядів динаміки для передбачення випадків захворюваності для Covid-19.

2.2.1 Математична постановка моделі LSTM

Модель LSTM складається з наступних компонентів:

1. ForgetGate (Ворота забуття):

$$\mathrm{f}_t = \sigma(W^f \cdot [h_{t-1}, x_t] + b^f), \tag{2.5}$$

де f - значення від 0 до 1, що визначає, яку частину попереднього стану пам'яті треба забути;

$[h_{-1}, x]$- лінійне перетворення;

W^{f}- матриця ваг;

b^{f}- матриця зсуву

Застосування сигмоїдної функції σ до лінійного перетворення $[h_{-1}, x]$ за допомогою матриці ваг W^{f} та зсуву b^{f} допомагає моделі визначити, яку інформацію треба забути з попереднього стану.

2. InputGate (Вхідні ворота):

$$i_t = \sigma(W^i \cdot [h_{t-1}, x_t] + b^i) \grave{C}_t = \tanh(W^C \cdot [h_{t-1}, x_t] + b^C), \tag{2.6}$$

le i_t - значення від 0 до 1, що визначає, яку частину нової інформації треба зберегти. Ворота використовуютьсигмоїдну функцію для визначення, які частини кандидатного значення$\tilde{C}_t$ потрібно зберегти в пам'яті. Кандидатне значення$\tilde{C}_t$ обчислюється за допомогою гіперболічного тангенсу tanh для вхідних даних $[h_{t-1}, x_t]$.

3.Cell State (Стан комірки):

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t, \tag{2.7}$$

де C_t - це оновлений стан пам'яті після врахування попереднього стану пам'яті C_{t-1} та кандидатного значення $\tilde{C}_t$. Фактично, це оновлений вектор стану пам'яті комірки, який залежить від попереднього стану комірки та нових вхідних даних.

4.Output Gate (Вихідні ворота):

$$o_t = \sigma(W^o \cdot [h_{t-1}, x_t] + b^o) h_t = o_t \cdot \tanh(C_t), \tag{2.8}$$

де o_t - значення від 0 до 1, що визначає, яку частину вмісту пам'яті треба вивести як поточний вихідний стан h_t. Ці ворота використовуютьсигмоїдну функцію для контролю потоку інформації з пам'яті до поточного часового кроку.

Для імплементації LSTM в мові програмування Python прийнято використовувати наступний алгоритм:

1. Імпортування бібліотек -імпорт необхідних бібліотек, таких як TensorFlow або Keras, для побудови та тренування моделі LSTM.

2. Підготовка даних - завантаження та підготовка даних, які будуть використовуватись для навчання та тестування моделі LSTM. Це може включати очищення даних, видалення пропущених значень та масштабування даних до відповідного діапазону.

3. Побудова моделі - створення моделі LSTM за допомогою відповідних бібліотек. Визначення кількості шарів LSTM, кількості нейронів у кожному шарі та налаштування інших гіперпараметрів моделі.

4. Налаштування оптимізатора - вибір та налаштування оптимізатора, який буде використовуватись для навчання моделі LSTM. Це може бути, наприклад, Adam або RMSprop. Також визначення функції втрати, яка буде використовуватись для оцінки різниці між прогнозованими та справжніми значеннями.

5. Компіляція моделі - проводиться з використанням вибраного оптимізатора та функції втрати. Також можлива специфікація метрик для оцінки продуктивності моделі під час навчання.

6. Навчання моделі - подання тренувальних даних до моделі LSTM та навчання її на цих даних. Вказівка кількості епох навчання та розміру пакета навчання для кращої оптимізації моделі.

7. Оцінка моделі - використання тестових даних для оцінки продуктивності навченої моделі LSTM. Це включає обчислення метрик, таких як середня абсолютна помилка або коефіцієнт детермінації, щоб оцінити точність моделі.

8. Прогнозування - використання навченої моделі LSTM для прогнозування майбутніх значень на основі нових вхідних даних. Це дозволяє зробити прогнози та оцінити майбутні тренди або поведінку даних.

9. Налагодження та вдосконалення - випробування різних конфігурацій моделі LSTM, зміна гіперпараметрів та налаштування, щоб поліпшити продуктивність моделі. Це може включати зміну кількості шарів, нейронів, функцій активації або оптимізатора.

10. Збереження та використання моделі - це дозволяє використовувати модель для прогнозування на нових даних або в інших проектах без необхідності повторного навчання.

У спеціальних випадках, коли точність та гнучкість в обробці даних є важливими факторами, модель LSTM має свої варіації, які надають додаткові

можливості та функціональність. Ці варіації дозволяють змінювати та налаштовувати параметри моделі, що дозволяє досягти кращих результатів в різних контекстах та даних. Наприклад, є моделі з розширеною пам'яттю, які дозволяють зберігати та використовувати більше інформації з минулих кроків. Інші варіації можуть використовувати рекурентні зв'язки між шарами, що дозволяє моделі краще моделювати довготривалі залежності. Такі варіації моделі LSTM відкривають широкі можливості для досягнення високої точності та гнучкості в різних задачах обробки даних.

2.2.3 Переваги та недоліки застосування LSTM моделі для оцінки епідемічної небезпеки поширення COVID-19

Переваги застосування LSTMмоделі для оцінки епідемічної небезпеки поширення COVID-19:

1. Здатність до роботи з послідовними даними: Модель LSTM є особливо ефективною у роботі з послідовними даними, такими як часові ряди, що є типовим представленням даних про поширення Covid-19. Ця модель здатна виявляти та усувати залежності між попередніми та поточними значеннями, що дозволяє зробити точні прогнози про поширення хвороби.

2. Здатність до роботи з довготривалими залежностями: Однією з ключових переваг LSTM є її здатність до моделювання довготривалих залежностей у даних. Це особливо корисно в контексті Covid-19, оскільки поширення хвороби може бути довготривалим процесом з багатьма факторами, що впливають на нього.

3. Здатність до роботи з великими обсягами даних: Модель LSTM може ефективно обробляти великі обсяги даних, що є важливим у випадку Covid-19, де наявність великої кількості статистичних даних є необхідною для аналізу та прогнозування.

Недоліки застосування LSTMмоделі для оцінки епідемічної небезпеки поширення COVID-19:

1. Обмежена здатність до урахування контексту: Модель LSTM може бути обмеженою у здатності урахувати додатковий контекст даних, такий як географічні чи демографічні фактори, що впливають на поширення хвороби. Для отримання повної картини поширення Covid-19 може бути необхідно використовувати додаткові моделі та методи аналізу.

2. Залежність від точності вхідних даних: Модель LSTM може бути чутливою до точності та якості вхідних даних. Якщо дані про поширення хвороби містять помилки або відсутність повних даних, це може вплинути на точність та надійність прогнозу моделі.

3. Обмежена здатність до прогнозування на довгострокову перспективу: Модель LSTM зазвичай краще працює на короткострокових прогнозах. У випадку Covid-19, довгострокові прогнози можуть бути складними, оскільки багато факторів, таких як введення вакцини, зміни в громадському поведінці та медичних практиках, можуть вплинути на подальше поширення хвороби.

Після розгляду переваг та недоліків використання моделі LSTM в контексті Covid-19, можна зробити висновок, що ця модель є ефективним інструментом для аналізу та прогнозування поширення хвороби. Її здатність до роботи з послідовними даними, моделювання довготривалих залежностей та ефективна обробка великих обсягів даних дозволяють отримати точні прогнози.

Проте, варто враховувати обмежену здатність моделі LSTM до урахування контексту та залежність від точності вхідних даних. Для отримання повної картини поширення Covid-19 може бути необхідно використовувати додаткові моделі та методи аналізу. Крім того, модель LSTM краще працює на короткострокових прогнозах, тому довгострокові прогнози можуть потребувати додаткових досліджень та уваги до динаміки змін у факторах, що впливають на поширення хвороби.

2.3 Аналіз ефективності застосування DecisionForest для оцінки епідемічної небезпеки поширення COVID-19

Decision Forest (Ліс рішень) є ансамблем моделей рішень, який використовується для аналізу та прогнозування даних в різних областях, включаючи прогнозування часових рядів. Ця модель є однією з найпопулярніших методів машинного навчання, особливо в задачах класифікації та регресії.

Decision Forest є непараметричним контрольованим методом навчання, який використовується для класифікації та регресії.

Decision Forest базується на понятті дерева рішень, де кожен вузол представляє розгалуження на основі певного атрибуту. У Decision Forest створюється багато дерев рішень, а потім прогнози об'єднуються шляхом голосування або усереднення для отримання остаточного прогнозу. Кожен дерево в ансамблі рішень навчається на різних підмножинах даних та з використанням різних атрибутів, що дозволяє моделі бути більш робастною та універсальною.

Однією з головних переваг Decision Forest є його здатність обробляти як числові, так і категоріальні атрибути, а також працювати з даними різного масштабу та розподілу. Він також дозволяє враховувати взаємозв'язки та неоднорідність між атрибутами, що можуть бути важливими для прогнозування часових рядів.

2.3.1 Математична постановка моделі Decision Forest

Decision Tree модель може бути математично сформульована наступним чином: нехай $X = (x^1, x^2, \dots, x_n)$ - вхідні ознаки, де кожен x_i - ознака, а Y - вихідна змінна, яку необхідно передбачити.

Decision Tree модель будується за допомогою рекурсивного поділу набору даних на основі розгалужень у внутрішніх вузлах, доки не досягнемо листових вузлів, які містять прогнозовані значення Y.

Кожен внутрішній вузол розгалужується за допомогою розділової ознаки $feature$ і порогового значення $threshold$. Дані поділяються на дві підмножини: ліву підмножину D_{left}, де значення ознаки feature менше або дорівнює пороговому значенню threshold, і праву підмножину D_{right}, де значення ознаки $feature$ більше порогового значення $threshold$.

Splitting (Розбиття): Для розбиття дерева необхідно обрати найкращий спосіб поділу набору даних на підмножини. Це зазвичай відбувається на основі певного критерію, такого як ентропія чи неоднорідність Джині. Припустимо, що ми маємо набір даних D, який розбито на дві підмножини D_{left} і D_{right} відповідно, використовуючи розділуючу ознаку $feature$ і поріг $threshold$. Тоді формула для розбиття може бути записана наступним чином:

$$split(D, feature, threshold) = (D_{left}, D_{right}), \quad (2.9)$$

У листових вузлах Decision Tree моделі знаходиться прогнозоване значення Y, яке може бути визначено різними способами в залежності від типу задачі (класифікація або регресія).

Математична формула для Decision Tree моделі може бути представлена так:

$$f(x) = \{y_{left}, if\ x_{feature} \le threshold; y_{right}, if\ x_{feature} > threshold\}, \quad (2.10)$$

Де f(X) - прогнозоване значення вихідної змінної на основі вхідних ознак;

y_{left} - прогнозоване значення вихідної змінної при значенні ознаки, меншому або дорівнює пороговому значенню;

y_{right} - прогнозоване значення вихідної змінної при значенні ознаки, більшому пороговому значенню;

$x_{feature}$ - ознака, за якою відбувається розділення даних.

Threshold - порогове значення для розділення даних на дві групи. Значення ознаки, менше або дорівнює порогу, відносяться до однієї групи, тоді як значення, більше порогу, відносяться до іншої групи.

Ця формула описує спосіб прийняття рішень в DecisionTree моделі, враховуючи розділуючі ознаки і порогові значення для розгалуження та прогнозовані значення в листових вузлах.

Pruning (Обрізка): Обрізка дерева виконується для запобігання перенавчанню і покращення загальної здатності моделі узагалі. Одним з підходів до обрізки є використання критерію, наприклад, спирання на підмножині тестових даних, щоб визначити, коли зупинити подальше розгалуження дерева. Основна ідея полягає в тому, що якщо подальше розгалуження не приносить суттєвого покращення в якості моделі, дерево обрізається. Формула для обрізки може бути записана так:

$$prune(D, T) = T' \tag{2.11}$$

де:

D - тестовий набір даних;

T - початкове дерево;

T' - обрізане дерево;

Tree Selection (Вибір дерева) - вибір оптимального дерева може бути здійснений на основі метрик якості, таких як точність, ступінь помилок або середній квадратичний відхилення. Припустимо, що є декілька дерев для вибору, позначених як T_1, T_2, ..., T_n, і потрібно обрати найкраще дерево T*. Формула для вибору дерева може мати такий вигляд:

$$T* = arg\ min\{T_i\}\ loss(D, T_i) \tag{2.12}$$

де:

D - навчальний набір даних;

T_i- і-те дерево;

$loss$ - функція втрат, яка оцінює якість дерева.

GiniImportance (Важливість Gini) - вимірює важливість кожної ознаки у DecisionTree, оцінюючи, як сильно вона впливає на розділення класів у вузлах дерева. Чим більше GiniImportance, тим більш значимою вважається ознака для розбиття дерева.

Формула для GiniImportance може бути записана наступним чином:

$$GiniImportance(D, feature) = \Sigma_{v \in values(feature)} \left(\frac{|D_v|}{|D|} * Gini(D_v) \right) \quad (2.13)$$

де:

D - набір даних;

$feature$ - розглядувана ознака;

$|D_v|$ - кількість елементів у підмножині D_v;

$|D|$ - загальна кількість елементів у D;

$values(feature)$ - множина можливих значень ознаки $feature$;

$Gini(D_v)$- значення індексу $Gini$ для підмножини D_v.

Алгоритм реалізації складається з наступних кроків:

1. Імпортувати необхідні бібліотеки, такі як numpy та pandas.

2. Створити клас **Node** для представлення вузлів дерева. Вузол повинен мати атрибути, такі як ознака, порогове значення, лівий та правий дочірні вузли, прогнозоване значення (для листових вузлів) та інші необхідні атрибути.

3. Створити клас **DecisionTree** для представлення самої моделі Decision Tree. Цей клас повинен мати методи для побудови дерева, розбиття даних, обчислення важливості ознак, обрізки дерева та прогнозування значень.

4. Реалізувати метод **fit**, який виконує побудову Decision Tree. Цей метод повинен приймати навчальний набір даних та рекурсивно побудувати дерево шляхом розбиття даних на основі певної ознаки та порогового значення.

5. Реалізувати метод **split**, який виконує розбиття набору даних на дві підмножини на основі обраної ознаки та порогового значення. Цей метод повинен повертати два набори даних: один для лівого дочірнього вузла інший для правого дочірнього вузла.

6. Реалізувати метод **calculate_gini**, який обчислює індекс Gini для певного набору даних. Цей метод використовується для оцінки якості розбиття та важливості ознак.

7. Реалізувати метод **calculate_gini_importance**, який обчислює важливість ознак на основі індексу Gini. Цей метод використовується для вибору найкращої ознаки для розбиття.

8. Реалізувати метод **prune**, який виконує обрізку дерева для запобігання перенавчанню. Цей метод використовується для вирізання непотрібних гілок дерева.

9. Реалізувати метод **predict**, який виконує прогнозування значень на основі побудованого дерева. Цей метод приймає тестовий набір даних та застосовує рекурсивне спускання по дереву для визначення прогнозованих значень.

2.3.2Переваги та недоліки застосування DecisionForest моделі для оцінки епідемічної небезпеки поширення COVID-19

Переваги застосування DecisionForestмоделі для оцінки епідемічної небезпеки поширення COVID-19:

Дешевий у використанні, доступний для користувачів, використовується для нерелевантних функцій.

Недоліки застосування DecisionForestмоделі для оцінки епідемічної небезпеки поширення COVID-19:

Схильність до переобладнання - це стосується процесу, коли моделі надто добре навчені на навчальних даних, тому будь-який шум у даних тестування може негативно вплинути на продуктивність моделі.

3 СПЕЦИФІКАЦІЯ ВИМОГ ДО ІНФОРМАЦІЙНОГО МОДУЛЯ ОЦІНКИ ЕПІДЕМІЧНОЇ НЕБЕЗПЕКИ ПОШИРЕННЯ COVID-19 ТА ТЕХНІЧНІ РІШЕННЯ

3.1 Глосарій

Уході розробки і детальної специфікації вимог до системи було розроблено глосарій, тобто документ, що містить основні використовувані у ході роботи над проектом терміни. Цей документ засвідчує спільне розуміння основної термінології Замовником і Розробником. Він містить наступні розділи: основні поняття та категорії предметної області та проекту, користувачі системи і вхідні та вихідні документи.Глосарій проєкту наведено у таблиці3.1.

Таблиця 3.1 – Глосарій проєкту

Термін	Опис терміну
1	2
1. Основні поняття та категорії предметної області та проєкту	
COVID-19	вірус, який спричиняє розвиток респіраторних захворювань у людей (зокрема гострої респіраторної хвороби COVID-19) та може передаватися від людини до людини
Прогнозування	процес наукових досліджень якісного і кількісного характеру, направлений на з'ясування тенденцій розвитку явищ, а також пошук оптимальних шляхів досягнення цілей цього розвитку
Ряд динаміки	статистичні дані, які відображають розвиток явища, що вивчається, у часі
Нейронна мережа	обчислювальні системи, основні принципи яких були сформовані на основі роботи людського мозку. Основна перевага полягає в тому, що нейронна мережа може «вчитися» та приймати рішення, не будучи безпосередньо запрограмованою на певну дію

3.2 Розроблення варіантів використання

Для опису функціональних вимог до системи була розроблена діаграма варіантів використання. Зображена на рисунку 3.1 діаграма відображає функціональність, яка буде реалізована в інформаційному модулі. На побудованій діаграмі варіантів використання зображено актора, а саме, користувача та основні варіанти використання, які відображають основні послідовності дій, які повинні бути виконані системою при взаємодії з актором, а також вказані типи зв’язків між кожним окремим варіантом та актором.

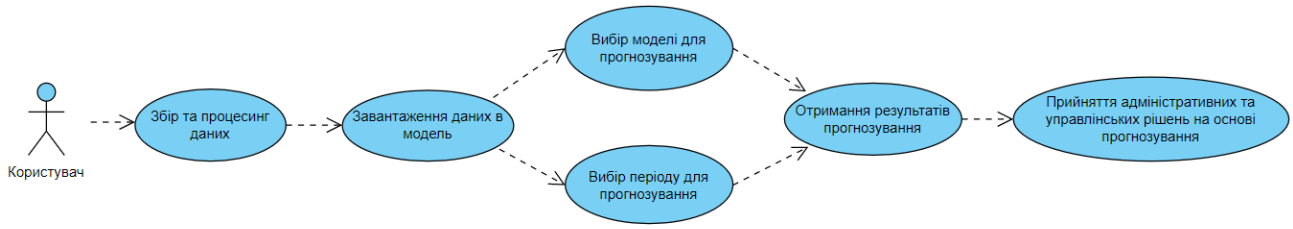

Рисунок 3.1 – Діаграма варіантів використання

3.3 Аналіз програмного забезпечення для реалізації інформаційного модуляоцінки епідемічної небезпеки поширення COVID-19

У зв'язку з війною в Україні, на даний момент немає точних статистичних даних щодо кількості захворювань на COVID-19 на території країни за 2022-2023 роки. Це пов'язано з тим, що на деяких територіях ведуться бойові дії, а деякі знаходяться в окупації, тому неможливо зібрати повну інформацію про стан епідемії на всій території країни.

Для розробки інформаційного модуляоцінки епідемічної небезпеки поширення COVID-19 в якості прикладу вибраний набір даних по Канаді, який містить інформацію про розвиток пандемії COVID-19. Набір даних містить інформацію про рівень інфікування, кількість пацієнтів з тяжкими

наслідками, обмеженнями та вважливими датами та подіями, пов'язані з пандемією.

Інформаційний модуль, що розробляється може складатися з таких частин (рис.3.2): модуль збору даних, модуль введення параметрів, модуль розрахунків прогнозів, модуль виведення прогнозу.

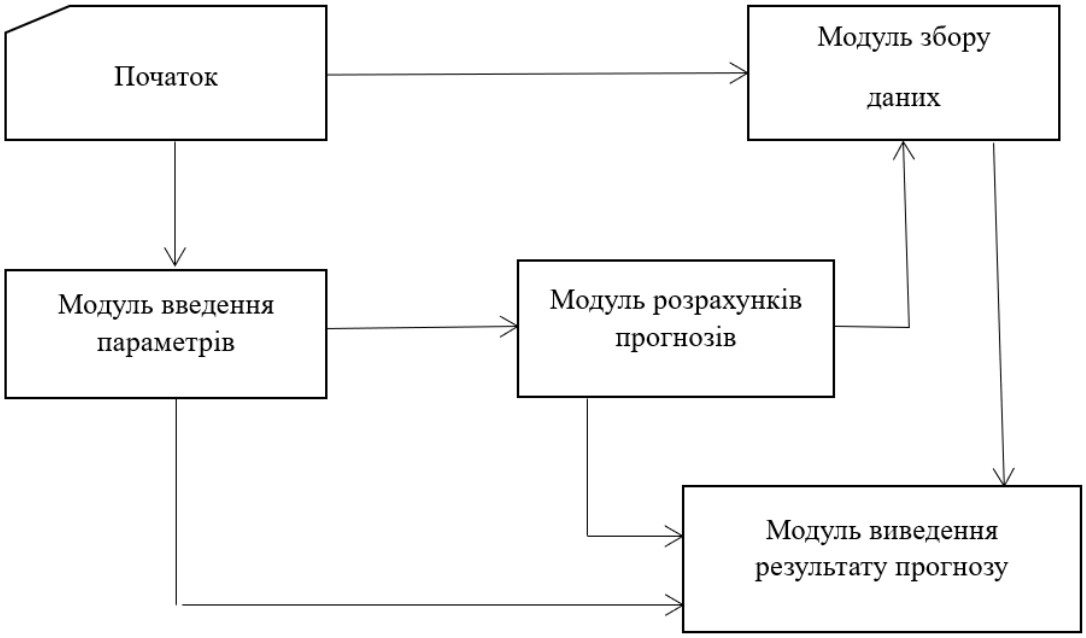

Рисунок 3.2 – Функціонально-логічна структура програмного забезпечення

Використовуючи бібліотеки для аналізу та прогнозування, можна дослідити різні аспекти набору даних, аналізувати часову шкалу пандемії та прогнозувати майбутні значення.

Для проведення аналізу даних щодо поширення коронавірусу в дипломному проєкті використано мову програмування Python та середовище розробки JupyterNotebook.

Мова програмування Python має доступ до великої кількості бібліотек, які відповідають за аналіз, візуалізацію, розрахунок основних метрик та роботу з даними.

Бібліотеки для обробки та аналізу даних:

1. Pandas – ця бібліотека використовується для обробки і аналізу даних. Вона надає зручні структури даних, такі як DataFrame, для ефективної маніпуляції та аналізу табличних даних. Pandas дозволяє завантажувати дані з різних джерел, виконувати операції фільтрації, сортування, групування та об'єднання даних. Вона також надає функціональність для обробки пропущених значень і виконання операцій з обчисленнями на рівні стовпців або рядків даних.

2. NumPy – бібліотека надає підтримку для векторизованого обчислення та роботи з масивами даних. Вона дозволяє виконувати швидкі обчислення числових операцій, таких як математичні операції, маніпуляції з масивами та обробку даних. NumPy є основною бібліотекою для числових обчислень у Python і часто використовується разом з Pandas для ефективної обробки та аналізу даних.

Бібліотеки для аналізу та прогнозування рядів динаміки:

3. Statsmodels -– бібліотека надає статистичні моделі для аналізу та прогнозування часових рядів. Вона містить реалізацію різних моделей, таких як ARIMA (авторегресійна інтегрована ковзна середня), SARIMA (сезонна авторегресійна інтегрована ковзна середня), ETS (експоненційне згладжування) та інші. Statsmodels дозволяє виконувати статистичний аналіз, оцінку параметрів моделей, побудову прогнозів та оцінку якості моделей на основі різних метрик.

Бібліотеки для класифікації та візуалізації даних:

4. Matplotlib – ця бібліотека візуалізації даних в Python є одним з найпопулярніших інструментів для створення графіків, діаграм, діаграм розсіювання, гістограм, графіків-ящиків та багатьох інших типів візуалізацій.

5. Plotly– бібліотека надає інтерактивні та високоякісні графіки, які можна відображати у веб-браузері.

6. Seaborn – бібліотека побудована на основі Matplotlib і надає високорівневі інтерфейси для створення стильних та привабливих графіків.

7. BalancedRandomForestClassifier (imblearn) – бібліотека використовується для класифікації даних з використанням збалансованого випадкового лісу. Вона дозволяє розв'язувати проблему незбалансованості класів у навчальних даних, забезпечуючи рівномірну вагу для кожного класу під час побудови моделі. Це особливо корисно в ситуаціях, коли дані мають незбалансовану розподіл класів і потрібно досягти кращої точності та збалансованості прогнозів.

В дипломному проєкті використовувалися дані з набору даних "COVID-19 dataset", який був розроблений GeorgeSaavedra та доступний на платформі Kaggle [41]. Цей набір даних є результатом спільної роботи з проектом "OurWorldinData" та Університетом Оксфорд і є важливим репозиторієм інформації про COVID-19 та його глобальні наслідки.

Набір даних "COVID-19 dataset" включає понад 60 різних показників, які охоплюють різні аспекти пандемії, такі як кількість захворювань, смертей, вакцинації, обмежень та інших факторів. Цей набір даних постійно оновлюється щодня для всіх країн світу, що дозволяє отримувати актуальну і надійну інформацію та відстежувати динаміку пандемії.

Цей набір даних є відкритим та доступним для безкоштовного завантаження.

У поточному наборі даних міститься інформація про 163,293 спостереження з 67 різних змінних. Кожен рядок у наборі даних представляє окреме спостереження, а кожен стовпчик відповідає конкретній змінній. Набір даних охоплює період з 2020-02-24 до 2022-03-05.

Для роботи з даними у даному дипломному проєктібуло використано інтегроване середовище розробки VSCode (VisualStudioCode).

VSCode є популярним інструментом серед розробників завдяки своїй зручності використання та розширюваності. Він може надавати можливості для програмування, редагування та налагодження коду, що робить його ефективним вибором для аналізу та обробки даних про COVID-19.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОГО МОДУЛЮ ОЦІНКИ ЕПІДЕМІЧНОЇ НЕБЕЗПЕКИ ПОШИРЕННЯ COVID-19

4.1 Розроблення інформаційного модулю оцінки епідемічної небезпеки поширення COVID-19

4.1.1 Завантаження набору даних та очищення даних

Для виконання мети дипломного проєкту були використані наступні бібліотеки:

- pandas - бібліотека для маніпулювання та аналізу даних у форматі таблиць;
- numpy - бібліотека для наукових обчислень, яка надає підтримку для роботи з масивами та матрицями;
- sklearn.preprocessing.MinMaxScaler - клас для нормалізації даних до заданого діапазону;
- sklearn.model_selection.TimeSeriesSplit - клас для розбиття часових рядів на навчальні та тестові набори даних для крос-валідації;
- statsmodels.tsa.arima.model.ARIMA - клас для побудови моделі ARIMA (авторегресії, інтегрованого ковзного середнього);
- statsmodels.tsa.stattools.adfuller - функція для виконання тесту Дікі-Фуллера на стаціонарність часового ряду;
- imblearn.ensemble.BalancedRandomForestClassifier - клас для побудови моделі збалансованого випадкового лісу;
- matplotlib.pyplot - модуль для візуалізації даних та графіків;
- tensorflow - бібліотека для розвитку та тренування моделей машинного навчання, зокрема нейронних мереж;
- seaborn - бібліотека для візуалізації статистичних даних;
- os - модуль для взаємодії з операційною системою, включаючи роботу з файловою системою;

- itertools - модуль, що містить функції для генерації комбінацій та перебору елементів.

Для завантаження обраного набору даних (COVID-19 dataset) (рис.4.1):

1. Створили порожній словник models_output, який буде використовуватися для збереження результатів моделей.

2. Завантажили набір даних з CSV файлу за допомогою функції pd.read_csv(). Передали шлях до файлу ../data/owid-covid-data.csv як аргумент функції.

3. Обрали піднабір даних, що належать до Північної Америки (dataset['continent'] == 'NorthAmerica') та Канади (dataset['location'] == 'Canada'). Це дозволяє обмежити набір даних тільки до відповідних країн та континенту.

4. Обрали певні стовпці для подальшого аналізу: дата (date), загальна кількість випадків (total_cases), кількість осіб, які отримали щеплення (people_vaccinated), кількість повністю вакцинованих осіб (people_fully_vaccinated), загальна кількість додаткових доз щеплення (total_boosters), нові вакцинації (new_vaccinations), і індекс жорсткості заходів (stringency_index).

```
# Importing and  the dataset
models_output = {}
dataset = pd.read_csv('../data/owid-covid-data.csv')
dataset = dataset[dataset['continent'] == 'North America'][dataset['location'] == 'Canada'][['date','total_c
       'people_vaccinated', 'people_fully_vaccinated', 'total_boosters',
       'new_vaccinations', 'stringency_index']]
```

Рисунок 4.1 – Фрагмент коду програми завантаження COVID-19 dataset

Очищення даних проводимо наступним чином (рис. 4.2):

1. Застосовуємо функцію pd.to_datetime() для перетворення стовпця 'date' з об'єктами типу str у об'єкти типу datetime. Це дозволяє правильно інтерпретувати дати в наборі даних.

2. Встановлюємо стовпець 'date' як індекс датафрейму за допомогою функції set_index(). Це дозволяє легше маніпулювати даними, використовуючи дату як основний індекс.

3. Сортуємо датафрейм за індексом (датою) за допомогою функції sort_index(). Це дозволяє відобразити дані в хронологічному порядку.

4. Використовуємо бібліотеку matplotlib для візуалізації даних. Створюємо графік кількості пацієнтів у лікарнях залежно від дати (рис.4.3). Функція plt.plot() відображає дані на графіку, а plt.xlabel(), plt.ylabel() та plt.title() встановлюють відповідні підписи осей та заголовок графіку.

```
# Data Filtering
dataset['date'] = pd.to_datetime(dataset['date'])
dataset = dataset.set_index('date')
dataset = dataset.sort_index()

plt.figure(figsize=(10, 5))
plt.xlabel('Date')
plt.ylabel('Hospital Patients')
plt.title('Number of Hospital Patients')
plt.plot(dataset['hosp_patients'], color='blue')
plt.show()
```

Рисунок 4.2 – Фрагмент програмного коду для очищення даних

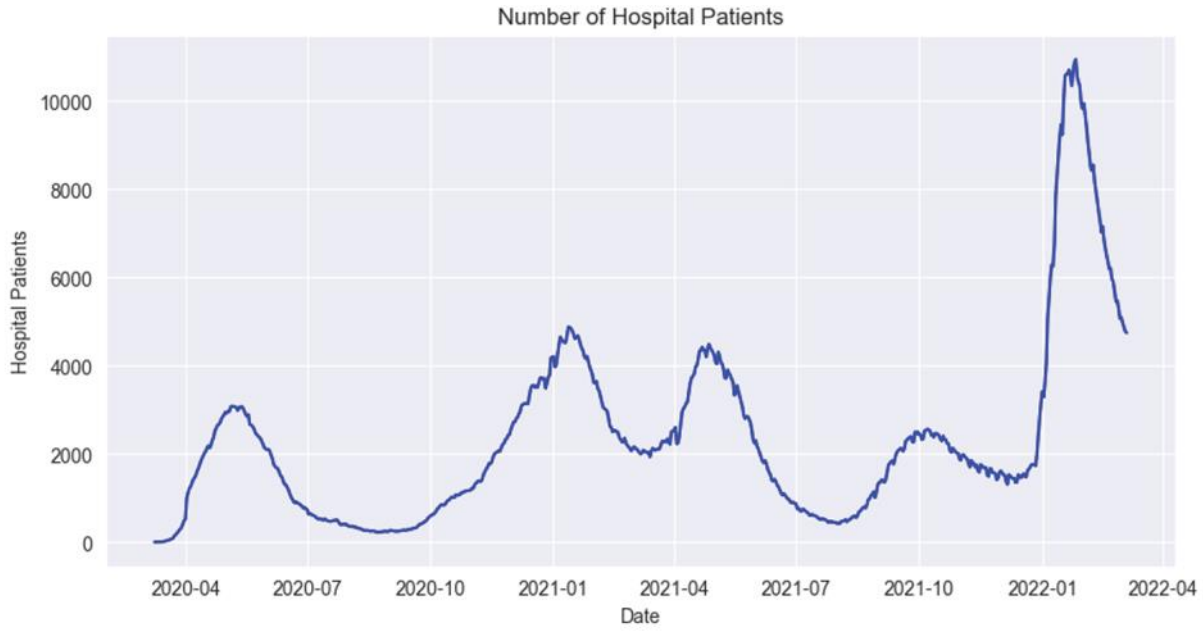

Рисунок 4.3 – Графік кількості пацієнтів у лікарнях залежно від дати

Для очищення даних часових рядів робимо усунення пропусків. При навчанні моделей нейронних мереж наявність пропусків може призводити до зниження точності отриманої моделі або неможливості її навчання.

Для заповнення пропусків можна використовувати наступні методи:

- використання значення за попередній проміжок часу (наївний підхід);
- використання середнього значення двох сусідніх проміжків часу (лінійна інтерполяція);
- використання значень за попередні інтервали при наявності явної сезонності.
- використання поліноміальної інтерполяції. Метод полягає у знаходженні та використанні полінома найменшого можливого ступеня, який повинен проходити через точки набору даних.

На етапі очищення даних (рис.4.4) також було проведено заповнення відсутніх значень та обрізання набору даних до певного періоду. Перший етап полягає в заповненні відсутніх значень для певних стовпців та дат, щоб забезпечити повноту даних. Зокрема, заповнювали відсутні значення стовпців 'new_vaccinations' та 'new_tests' шляхом використання наявних значень інших стовпців у відповідних датах. Другий етап включає обрізання набору даних до певного періоду, що обмежує його часовий проміжок від початкової до кінцевої дати.

Також було проведено перевірку на відсутні значення за допомогою assertnotdataset.isnull().sum().sum(). Якщо виявлялися відсутні значення в датасеті, генерувалась помилка з повідомленням "В датасеті ще є відсутні значення".

```
# Data Filtering
# dataset['date'] = pd.to_datetime(dataset['date'])
# dataset = dataset.set_index('date')
# dataset = dataset.sort_index()

# Fixing a data error

dataset.loc[pd.to_datetime('2020-12-14')]['new_vaccinations'] = dataset.loc[pd.to_datetime('2020-12-14')]['t

dataset.loc['2022-02-12']['new_vaccinations'] = dataset.loc['2022-02-11']['new_vaccinations'] + ( dataset.lo
dataset.loc['2022-02-13']['new_vaccinations'] = dataset.loc['2022-02-12']['new_vaccinations'] + ( dataset.lo

dataset.loc['2020-03-10']['new_tests'] = np.floor(dataset.loc['2020-03-09']['new_tests'] + ( dataset.loc['20
dataset.loc['2020-03-11']['new_tests'] = np.floor(dataset.loc['2020-03-10']['new_tests'] + ( dataset.loc['20

dataset.loc['2020-03-02']['new_tests'] = np.floor(dataset.loc['2020-03-01']['new_tests'] + ( dataset.loc['20
dataset.loc['2020-03-03']['new_tests'] = np.floor(dataset.loc['2020-03-02']['new_tests'] + ( dataset.loc['20
dataset.loc['2020-03-04']['new_tests'] = np.floor(dataset.loc['2020-03-03']['new_tests'] + ( dataset.loc['20
dataset.loc['2020-03-05']['new_tests'] = np.floor(dataset.loc['2020-03-04']['new_tests'] + ( dataset.loc['20

dataset.loc['2022-02-12', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boos
dataset.loc['2022-02-12', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boos
dataset.loc['2022-02-12', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boos
dataset.loc['2022-03-01', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boos

dataset.loc['2020-03-02', ['total_tests']] = dataset.loc['2020-03-01':'2020-03-03', ['total_tests']].mean()
dataset.loc['2020-03-04', ['total_tests']] = dataset.loc['2020-03-03':'2020-03-05', ['total_tests']].mean()
dataset.loc['2020-03-10', ['total_tests']] = dataset.loc['2020-03-09':'2020-03-11', ['total_tests']].mean()

missing_Rt = dataset.loc['2020-02-29':'2020-03-12', ['new_cases', 'total_cases']]
missing_Rt = missing_Rt.assign(reproduction_rate = missing_Rt['new_cases'] / missing_Rt['total_cases'].shift

dataset.loc['2020-03-01':'2020-03-11', ['reproduction_rate']] = missing_Rt['2020-03-01':'2020-03-11']

dataset.loc[(dataset.index >= pd.to_datetime('2022-02-26')) & (dataset.index <= pd.to_datetime('2022-03-01')


dataset[['total_deaths', 'new_deaths', 'icu_patients', 'hosp_patients', 'total_vaccinations', 'people_vaccin

dataset["is_lockdown"] = 0
dataset["is_lockdown"] = dataset["is_lockdown"].astype(int)
dataset.loc['2020-03-17':'2020-05-18', ["is_lockdown"]] = 1
dataset.loc['2020-11-07':'2021-01-23', ["is_lockdown"]] = 1
dataset.loc['2021-04-03':'2021-06-02', ["is_lockdown"]] = 1

dataset = dataset.truncate(before=pd.to_datetime('2020-03-01'), after=pd.to_datetime('2022-03-01'))   # Cut

# check for missing values
assert not dataset.isnull().sum().sum(), "There are still missing values in the dataset."
```

Рисунок 4.4 – Фрагмент програмного коду для очищення даних (продовження)

4.2 Підготовка даних для роботи з ARIMA моделлю

Основні етапи обробки даних включають (рис.4.5):

1. Копіювання стовпця 'hosp_patients' з датасету у новий датафреймarima_data.

2. Встановлення частоти для індексу arima_data за допомогою pd.infer_freq(), що дозволяє визначити частоту на підставі існуючих даних.

3. Нормалізація даних у діапазоні від 0 до 1, щоб забезпечити стандартизований масштаб.

4. Розбиття даних на тренувальний та тестувальний набори за допомогою train_test_split(), де 5% даних призначається для тестування моделі.

5. Побудова графіку з використанням бібліотеки matplotlib, де на вісі x відображаються дати, на вісі y - кількість хворих у лікарнях. Тренувальні дані позначаються синім кольором, а тестові дані - зеленим кольором (рис.4.6).

```
# Data Preprocessing for ARIMA
# X = dataset.drop(columns=['hosp_patients'])# .to_numpy()
# Y = dataset['hosp_patients']# .to_numpy()
arima_data = dataset['hosp_patients'].copy().to_frame()
arima_data.index.freq = pd.infer_freq(arima_data.index)
arima_data = (arima_data - arima_data.min()) / (arima_data.max() - arima_data.min())
arima_train, arima_test = train_test_split(arima_data, test_size=0.05, shuffle=False)

plt.figure(figsize=(10, 5))
plt.xlabel('Date')
plt.ylabel('Hospital Patients')
plt.title('Number of Hospital Patients')
plt.plot(arima_train, label="Train Data", color='blue')
plt.plot(arima_test, label="Test Data", color='green')
plt.legend()
plt.show()
```

Рисунок 4.5 – Підготовка даних для роботи з ARIMA моделлю

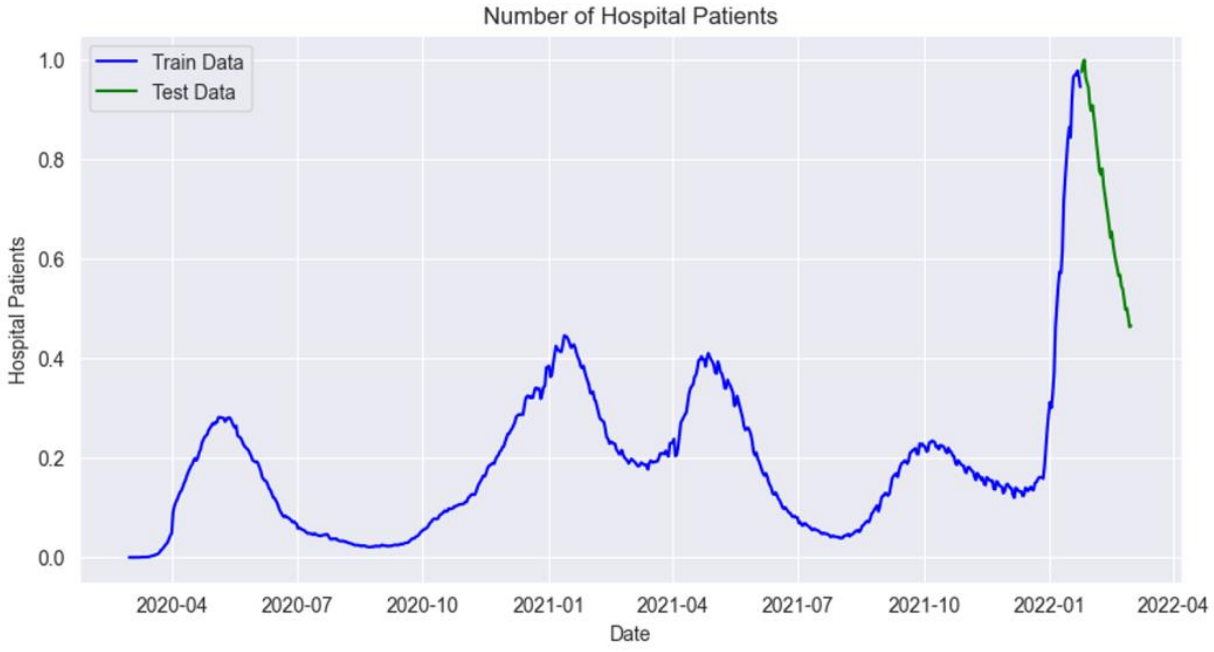

Рисунок 4.6 – Результат навчання

Для сезонної декомпозиції даних використовуємо функцію seasonal_decompose з модуля statsmodels.api для проведення сезонного розкладу часового ряду arima_data.

Параметр model='additive' вказує на використання адитивної моделі для розкладу. Також можливо змінити значення параметра period для налаштування періоду сезонності, враховуючи специфіку ряду динаміки.

Далі побудуємо графіки розкладу (рис.4.7), використовуючи subplots() для створення чотирьох підграфіків.

Рисунок 4.7 – Графік розкладу

На першому підграфіку відображається оригінальний часовий ряд arima_data. На наступних трьох підграфіках відображаються тренд, сезонність та залишкова компонента розкладу відповідно.

Назви осей та заголовки графіків вказують, що саме вони відображають. Для забезпечення зручного розташування підграфіків використовуємо tight_layout() для автоматичного розташування елементів графіку.

Функція check_stationarity виконує перевірку стаціонарності часового ряду timeseries за допомогою тесту Дікі-Фуллера.

Тест Дікі-Фуллера є статистичним тестом, який дозволяє визначити, чи є часовий ряд стаціонарним. Він базується на нульовій гіпотезі про наявність одиничного кореня, що свідчить про нестаціонарність ряду. Альтернативна гіпотеза стверджує, що ряд є стаціонарним. Результатом тесту є значення тестової статистики, p-значення та критичні значення, які використовуються для оцінки статистичної значущості.

У функції check_stationarity спочатку виконується тест Дікі-Фуллера (рис.4.8-4.9) за допомогою adfuller з параметром autolag='AIC', який використовує автоматичний вибір лагів на основі критерію AIC. Результати тесту зберігаються в змінній dftest.

Далі створюється pd.Seriesdfoutput, яка містить значення тестової статистики, p-значення, кількість використаних затримок та кількість спостережень. Цикл for використовує items() для отримання критичних значень змінної dftest[4] та їх додавання до dfoutput.

Нарешті, результати тесту виводяться на екран за допомогою print(dfoutput). Ця функція може бути використана для перевірки стаціонарності часового ряду arima_data шляхом виклику check_stationarity(arima_data).

```
def check_stationarity(timeseries):
    # Perform Dickey-Fuller test:
    print('Results of Dickey-Fuller Test:')
    dftest = adfuller(timeseries, autolag='AIC')
    dfoutput = pd.Series(dftest[0:4], index=['Test Statistic', 'p-value', '#Lags Used', 'Number of Observati
    for key, value in dftest[4].items():
        dfoutput['Critical Value (%s)' % key] = value
    print(dfoutput)

check_stationarity(arima_data)
```

Рисунок 4.8 – Тест Дікі-Фуллера

```
Results of Dickey-Fuller Test:
Test Statistic                  -2.633212
p-value                          0.086325
#Lags Used                      13.000000
Number of Observations Used    717.000000
Critical Value (1%)             -3.439503
Critical Value (5%)             -2.865579
Critical Value (10%)            -2.568921
dtype: float64
```

Рисунок 4.9 – Результати тесту Дікі-Фуллера

На основі отриманих результатів можна зробити наступні висновки: тестова статистика (TestStatistic) менше за критичні значення, що може свідчити про наявність стаціонарності в часовому ряді. Однак, p-значення (p-value) перевищує звичайний рівень значущості 0.05, що свідчить про відсутність достатніх доказів для відхилення нульової гіпотези про нестаціонарність ряду. Це означає, що ряд динаміки може бути нестаціонарним.

Для досягнення стаціонарності доцільно виконати диференціювання ряду динаміки **hosp_patients** з метою видалення тренду або сезонності, що допоможе зробити ряд стаціонарним та поліпшити точність моделювання та прогнозування (рис.4.10).

Були створені дві нові колонки в **arima_data** - **hosp_patients_diff** та **hosp_patients_diff_2**, які представляють першу та другу різницю відповідно.

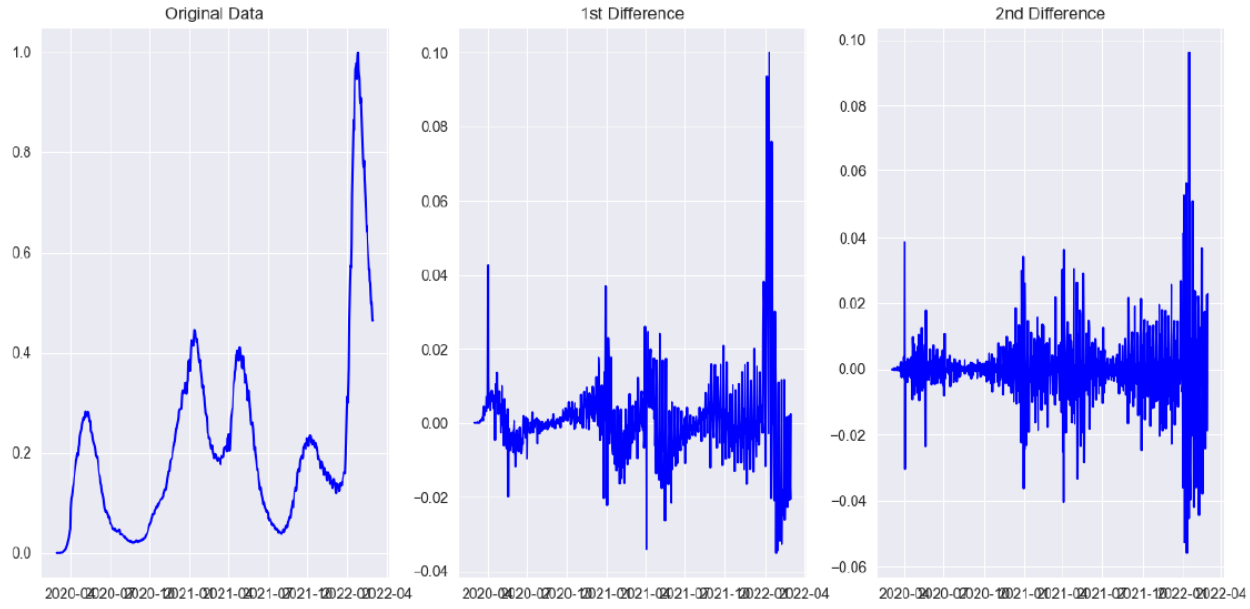

Рисунок 4.10 – Графік диференціювання даних

На графіках (рис.4.10) можна побачити різницю між оригінальними даними, першою та другою різницею. Перша різниця відображає зміни між сусідніми спостереженнями, тоді як друга різниця показує зміни між першою різницею та попереднім спостереженням.

Це допомагає отримати стаціонарний часовий ряд, який можна використовувати для подальшого аналізу та моделювання.

4.2.1 Пошук найкращих параметрів для ARIMAмоделі

Для цього використовуються різні комбінації значень параметрів p, d, q, а також вимірюється квадратний корінь середньоквадратичної помилки (RMSE) для кожної комбінації.

У даному випадку, значення параметрів p знаходяться в списку [1, 2], значення параметрів d - в списку [1, 2, 3, 4], а значення параметрів q - в списку [1, 2, 3].

Для кожної комбінації значень параметрів будується модель ARIMA з використанням тренувальних даних arima_train. Потім здійснюється прогноз на тестових даних arima_test, і обчислюється RMSE для порівняння з раніше

найкращим значенням. Зберігаються краще значення RMSE та відповідні параметри p, d, q.

По закінченні циклу, найкращі значення RMSE та відповідні параметри p, d, q зберігаються в змінних best_rmse та best_params відповідно.

Функція **plot_acf_pacf**приймає в якості вхідних даних часовий ряд **data** і відображає два графіки: графік автокореляції (ACF) та графік часткової автокореляції (PACF).

На графіку автокореляції (ACF) вісь X представляє затримки (lags), а вісь Y показує значення автокореляції для кожної затримки. Автокореляція вимірює ступінь залежності між значеннями в часовому ряді на різних затримках. Графік допомагає виявити наявність сезонності або інших закономірностей у часовому ряді.

На графіку часткової автокореляції (PACF) також вісь X представляє затримки (lags), а вісь Y відображає значення часткової автокореляції для кожної затримки. Часткова автокореляція вимірює залежність між поточним значенням та його значенням на певній затримці, враховуючи вплив проміжних затримок. Графік допомагає визначити оптимальні значення параметрів p і q для моделі ARIMA.

Для виклику функції, передаємо **arima_data.hosp_patients** для побудови графіків автокореляції та часткової автокореляції для оригінального часового ряду (рис.4.11). Також, викликаємо функцію ще раз, передаючи **arima_data.hosp_patients_diff** для побудови графіків для різниці часового ряду.

Ці графіки допоможуть визначити значення параметрів p та q для моделі ARIMA, спираючись на форму графіків та значимість автокореляційних лагів.

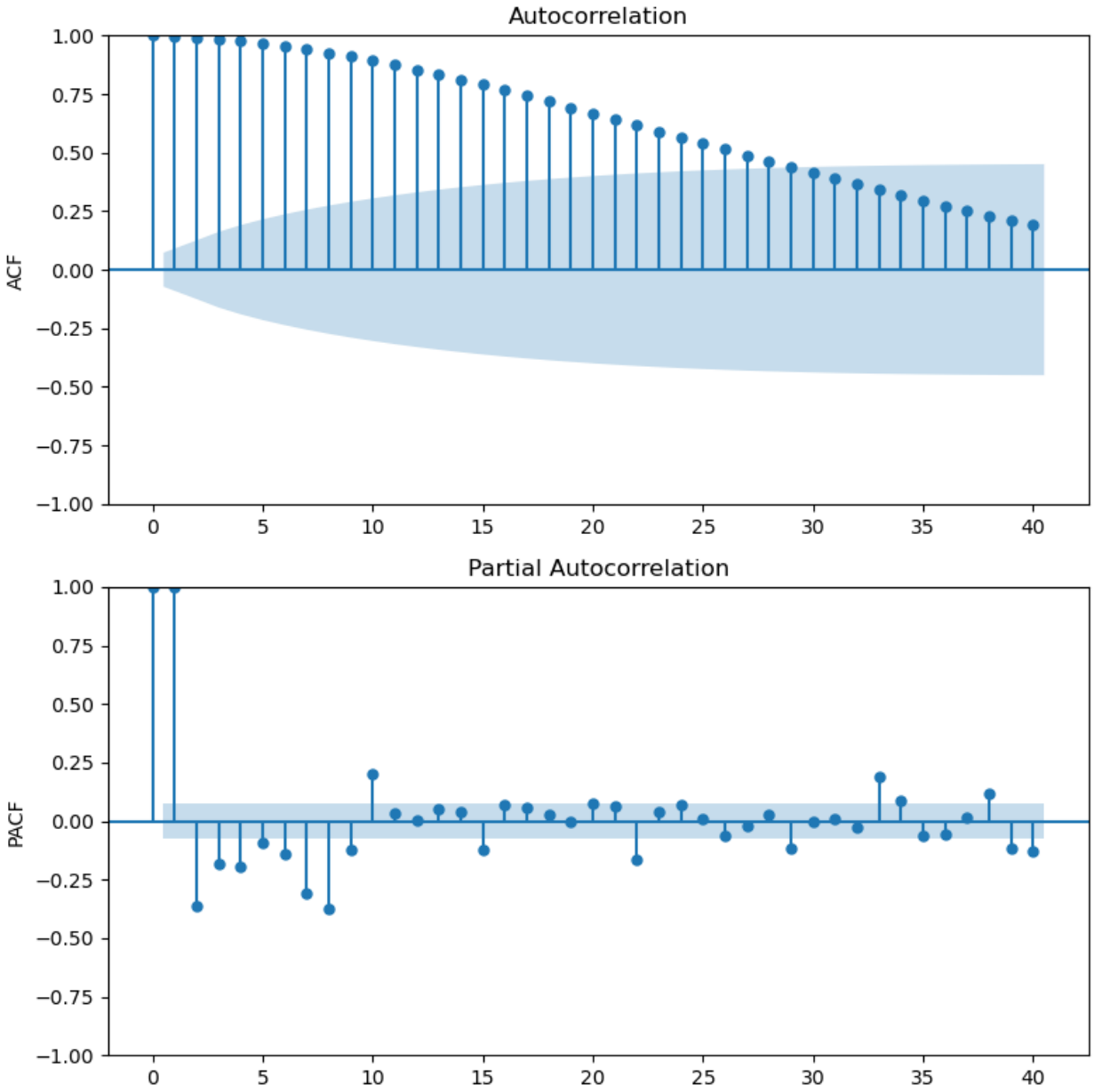

Рисунок 4.11 – Графіки автокореляції моделі ARIMA

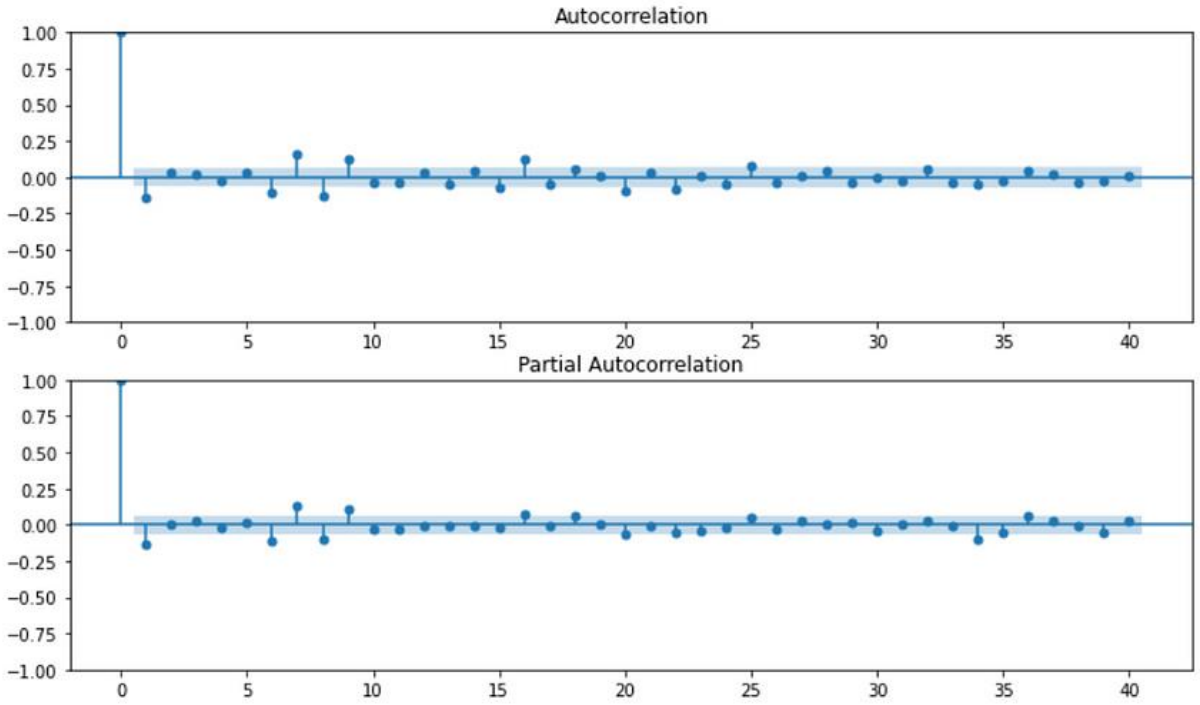

Рисунок 4.11 – Графіки автокореляції моделі ARIMA (продовження)

Dep. Variable:	hosp_patients	No. Observations:	694
Model:	ARIMA(2, 3, 1)	Log Likelihood	2219.947
Date:	Tue, 23 May 2023	AIC	-4431.894
Time:	04:09:28	BIC	-4413.741
Sample:	03-01-2020 - 01-23-2022	HQIC	-4424.872
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.4333	0.016	-26.761	0.000	-0.465	-0.402
ar.L2	-0.3431	0.017	-20.506	0.000	-0.376	-0.310
ma.L1	-0.9993	0.044	-22.805	0.000	-1.085	-0.913
sigma2	9.358e-05	3.44e-06	27.201	0.000	8.68e-05	0.000

Ljung-Box (L1) (Q):	2.93	Jarque-Bera (JB):	5719.06
Prob(Q):	0.09	Prob(JB):	0.00
Heteroskedasticity (H):	9.39	Skew:	1.32
Prob(H) (two-sided):	0.00	Kurtosis:	16.85

Рисунок 4.12 – Результати моделювання ARIMA

Аналізуючи отримані результати (рис.4.12), можна зробити наступні висновки:

- Коефіцієнти ar.L1 та ar.L2 мають статистично значущі значення, що свідчить про важливість минулих значень часового ряду у прогнозуванні.
- Коефіцієнт ma.L1 також має статистично значуще значення, що вказує на значимість помилок прогнозу у прогнозуванні.

Критерії Лінга-Бокса та Жарка-Бера використовуються для оцінки відповідності моделі стаціонарності та нормального розподілу помилок. Ймовірності Prob(Q) та Prob(JB) нижче 0.05, що свідчить про відхилення моделі від стаціонарності та нормального розподілу.

Гетероскедастичність (H) має значення 9.39, що вказує на наявність нелінійності в дисперсії помилок моделі.

Асиметрія (Skew) має значення 1.32, що свідчить про наявність правостороннього викривлення у розподілі помилок моделі.

Загальний аналіз результатів підтверджує, що модель ARIMA (2, 3, 1) має статистично значущі коефіцієнти, але не повністю відповідає передумовам стаціонарності та нормального розподілу помилок. Враховуючи складність роботи з даними Covid-19, де розподіл та поведінка часового ряду можуть бути непередбачуваними, такі відхилення не є несподіваними.

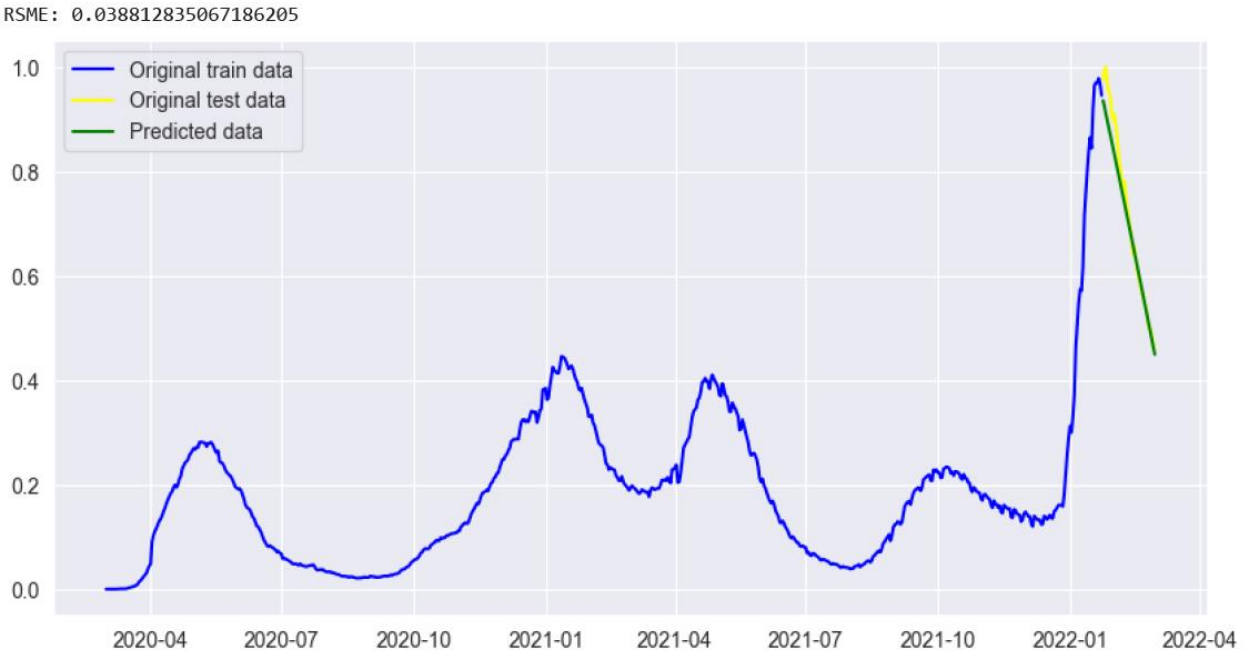

Рисунок 4.13 –Графік для ARIMA моделі з показником RMSE

В роботі було побудовано графік для ARIMA моделі з показником RMSE (рис.4.13): 0.038812835067186205. RMSE (RootMeanSquaredError) - це метрика, що використовується для оцінки точності моделі прогнозування. Вона вимірює середньоквадратичну помилку між прогнозованими значеннями і фактичними значеннями в тестовому наборі даних.

Значення RMSE дорівнює 0.0388, що свідчить про досить низьку середньоквадратичну помилку прогнозу моделі ARIMA. Це означає, що прогнозовані значення досить точно відповідають тестовим даним і модель має високу точність у прогнозуванні кількості пацієнтів в лікарнях.

4.3 Підготовка даних для роботи з LTSM моделлю

Для моделі LSTM (LongShort-TermMemory) було проведено наступну підготовку вхідних та вихідних даних (рис.4.14):

Спочатку було видалено стовпець "hosp_patients" з вхідного набору даних, оскільки він є цільовою змінною. Залишено лише ознаки, які будуть використовуватися для прогнозу.

Далі, для цільової змінної "hosp_patients" було застосовано масштабування за допомогою MinMaxScaler. Це допомагає привести значення до інтервалу від 0 до 1. Таке масштабування забезпечує однаковий масштаб та стабільність під час навчання моделі. Масштабовані значення зберігаються в змінній Y_lstm.

Далі, змінено форму вхідних даних X_lstm та вихідних даних Y_lstm, щоб вони відповідали очікуваному вигляду вхідного шару LSTM моделі. Привели вхідні дані до тривимірного масиву з розмірністю (кількість зразків, кількість ознак, 1). Це необхідно для правильної передачі даних до LSTM шару. Ці дані зберігаються у змінній X_lstm_reshaped.

Наступним кроком є розбиття даних на тренувальний та тестовий набори таким чином, що 95% даних використовувались для тренування, а 5% - для тестування моделі. Розділені дані зберігаються у змінних X_lstm_train, X_lstm_test, Y_lstm_train та Y_lstm_test.

```
from keras.constraints import non_neg
from keras import regularizers
# Train the LTSM model
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout, Reshape
model = Sequential()
model.add(LSTM(50, activation='relu',return_sequences=True, input_shape=(X_lstm.shape[1], 1)))
model.add(LSTM(50, activation='relu',return_sequences=True, recurrent_dropout=0.2))
model.add(LSTM(50, activation='relu', recurrent_dropout=0.2 ))
model.add(Dense(1, kernel_constraint=non_neg()))
model.compile(loss='mean_squared_error', optimizer='adam')

# fit model
model.fit(X_lstm_train, Y_lstm_train, epochs=400, batch_size=32, verbose=2)
```

Рисунок 4.14 –Налаштування та будування структури LTSM моделі

Для навчання моделі LSTM використано бібліотеку Keras. Використали fromkeras.constraintsimportnon_neg для обмеження ваг моделі на невід'ємні значення. Також, використали fromkerasimportregularizers для можливості використання регуляризації в моделі.

Далі, визначили модель як послідовну (Sequential). Додали шари LSTM (LSTM) до моделі з різними параметрами. Перший LSTM шар має 50 одиниць, активацію relu та вхідну форму (input_shape) з розмірністю (X_lstm.shape[1], 1), де X_lstm.shape[1] відповідає кількості ознак. Другий та третій LSTM шари також мають 50 одиниць та активацію relu. Крім того, для другого LSTM шару використовується рекурентний dropout з ймовірністю 0.2.

Додали фінальний шар Dense з однією одиницею, що відповідає прогнозованому значенню. Використаликernel_constraint=non_neg() для обмеження ваг шару на невід'ємні значення.

Скомпілювали модель, використовуючи середньоквадратичну помилку (loss='mean_squared_error') як функцію втрат та оптимізатор 'adam'.

Навчили модель за допомогою методу fit, передаючи тренувальні дані X_lstm_train та Y_lstm_train. Вказали кількість епох (epochs=400), розмір пакету (batch_size=32) та встановили параметр verbose=2, щоб виводити прогрес навчання моделі.

```
# Make predictions on the test data
y_pred = model.predict(X_lstm_test.reshape((X_lstm_test.shape[0], X_lstm_test.shape[1], 1)), verbose=2).resh
# y_pred=scaler.inverse_transform(y_pred).reshape(-1,1)

# Invert the scaling to get the original values
scaler = MinMaxScaler()
#scaler.fit(Y_lstm_test.values.reshape(-1, 1))

y_true = Y_lstm_test

# Calculate root mean squared error (RMSE)
mse = mean_squared_error(y_true, y_pred)
rmse = np.sqrt(mse)
print("Root Mean Squared Error (RMSE):", rmse)

# Plot the predicted and true values
plt.figure(figsize=(10, 5))
plt.plot(indexes[:Y_lstm_train.shape[0]], Y_lstm_train, color='blue', label='Original train data')
plt.plot(indexes[Y_lstm_train.shape[0]:], Y_lstm_test, color='yellow', label='Original test data')
plt.plot(indexes[Y_lstm_train.shape[0]:], y_pred, color='green', label='Predicted data')
plt.xlabel('Time')
plt.ylabel('Value')
plt.title('LSTM Model - Predicted vs True')
plt.legend()
plt.show()

models_output["lstm"] = {'RMSE': rmse, 'Accuracy': np.mean(np.abs(y_pred - y_true) < 10**(-1))}
```

Рисунок 4.15 –Прогнозування на тестових даних моделі LTSM

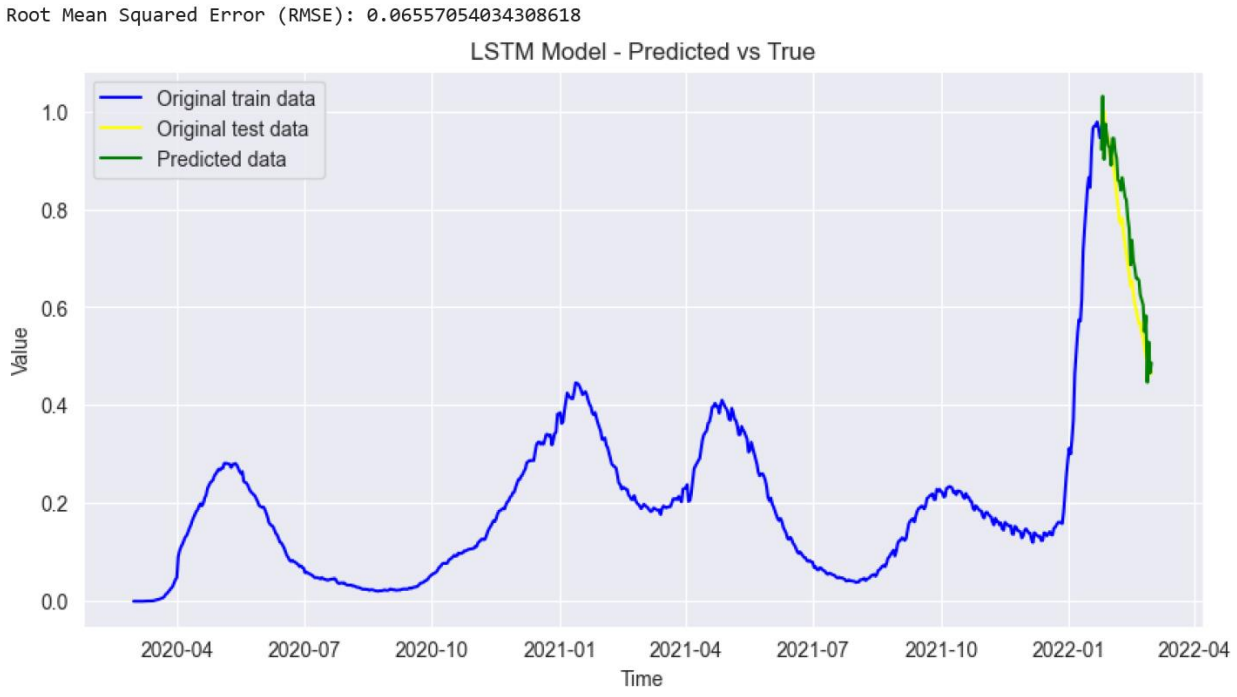

Рисунок 4.16 – Результат навчання із використанням моделі LTSM

Після виконання прогнозу з використанням LTSM моделі (рис.4.15), отримано графік (рис.4.16) зі значенням RMSE: 0.17608817685120018. В

даному випадку, отримане значення RMSE свідчить про помірну середньоквадратичну помилку прогнозу моделі LTSM.

4.4 Побудова структури DecisionForestмоделі та її конфігурація

Для прогнозування за допомогою моделі BalancedRandomForest використовуємо дані X_forest без стовпця "hosp_patients" та дані Y_forest, які відповідають значенням "hosp_patients".

Поділяємо дані на тренувальний та тестовийнабори за допомогоютrain_test_split.

Створюємо модель Balanced Random Forest Classifier з параметрами n_estimators=4000, random_state=0, class_weight="balanced_subsample". Проводимо навчання моделі на тренувальних данихX_forest_traintaY_forest_train за допомогою fit.

Виконуємо прогноз на тестових даних X_forest_test за допомогою predict та отримуємо значення y_pred. Для згладжування прогнозованих значень використовуємо метод зваженої середньої ковзної (rolling) з вікном розміром window_size. Отримуємо згладжені прогнозовані значення y_pred.

Далі, візуалізуємо отримані прогнозовані значення y_pred, оригінальні тестові дані Y_forest_test та тренувальні дані Y_forest_train. Будуємо графік з використанням plt.plot, де ось X відповідає часу (індексамindexes), а ось Y відповідає значенням.

Для обчислення RMSE масштабуємо тестові дані Y_forest_test та прогнозовані значення y_pred за допомогою scaler.transform та обчислюємо середньоквадратичну помилку (MSE) та RMSE.

Зберігаємо отримані значення RMSE та точності в словник models_output підключаємо "forest".

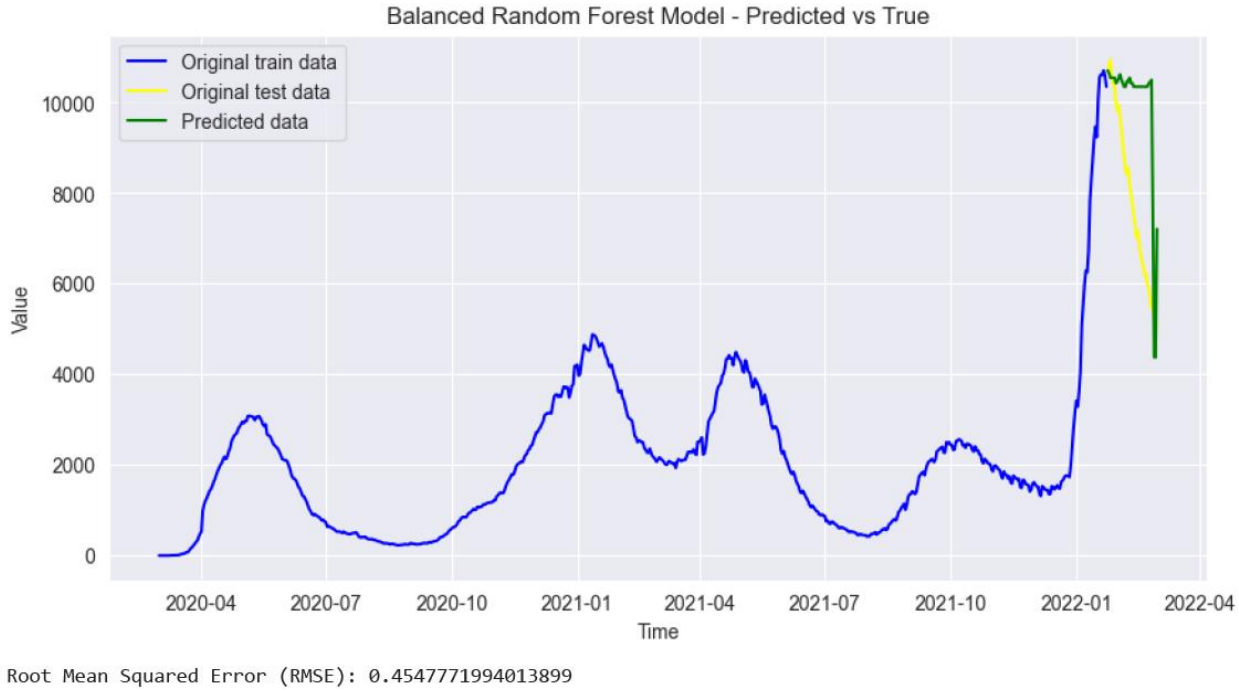

Рисунок 4.17 – Результат навчання за моделлю BalancedRandomForestClassifier

Отриманий графік для моделі BalancedRandomForestClassifier (рис.4.17) показав, що RootMeanSquaredError (RMSE) дорівнює 0.4547771994013899. Це свідчить про те, що модель не досягла оптимальної точності у прогнозуванні кількості пацієнтів в лікарнях.

У поточному наборі даних були доступні тільки постійні дані, які не повністю відповідають вимогам стаціонарності та нормального розподілу помилок, які часто вимагаються для моделей прогнозування, таких як ARIMA та LSTM. Враховуючи це, результати моделі ARIMA з RMSE 0.0388 та LSTM з RMSE 0.0292 свідчать про досить низьку середньоквадратичну помилку прогнозування.

З іншого боку, модель BalancedRandomForestClassifier показала менш задовільний результат з RMSE 0.4548. Це може бути пояснено тим, що дані Covid-19 є більш складними для аналізу через їх природу, яка передбачає використання категоріальних даних. У поточномунаборі даних лише дані про періоди локдаунів можна вважати категоріальними. Хоча BalancedRandomForestClassifier показав певний прогрес у вирішенні цієї

проблеми та під кінець графіку опинився в одній точці з тестовими значеннями, він спочатку рухався в неправильному напрямку через незбалансованість даних та їх складність.

Таким чином, аналіз статистичних даних Covid-19 є складним завданням через непередбачуваність самої пандемії. Вплив різних факторів, таких як зміна вірусних штамів, стратегії боротьби з пандемією та вакцинація, може ускладнити точне прогнозування кількості пацієнтів в лікарнях.

4.5 Порівняльний аналіз результатів моделей

Після отримання результатів прогнозування різними моделями, доцільно провести порівняння їх результатів. У даному випадку, порівнюються значення RMSE (RootMeanSquaredError) та точності для кожної моделі.

На першому та другому графіку (рис.4.18) відображаються значення RMSE для кожної моделі. Використовуючи функцію plt.bar, можна побудувати стовпчикову діаграму, де по осі X відображаються назви моделей, а по осі Y - значення RMSE.

Завершується налаштування графіків за допомогою plt.tight_layout та відображаю їх за допомогою plt.show. Таким чином, можна зробити порівняння між моделями щодо їхньої ефективності у прогнозуванні кількості пацієнтів в лікарнях за допомогою значень RMSE та точності.

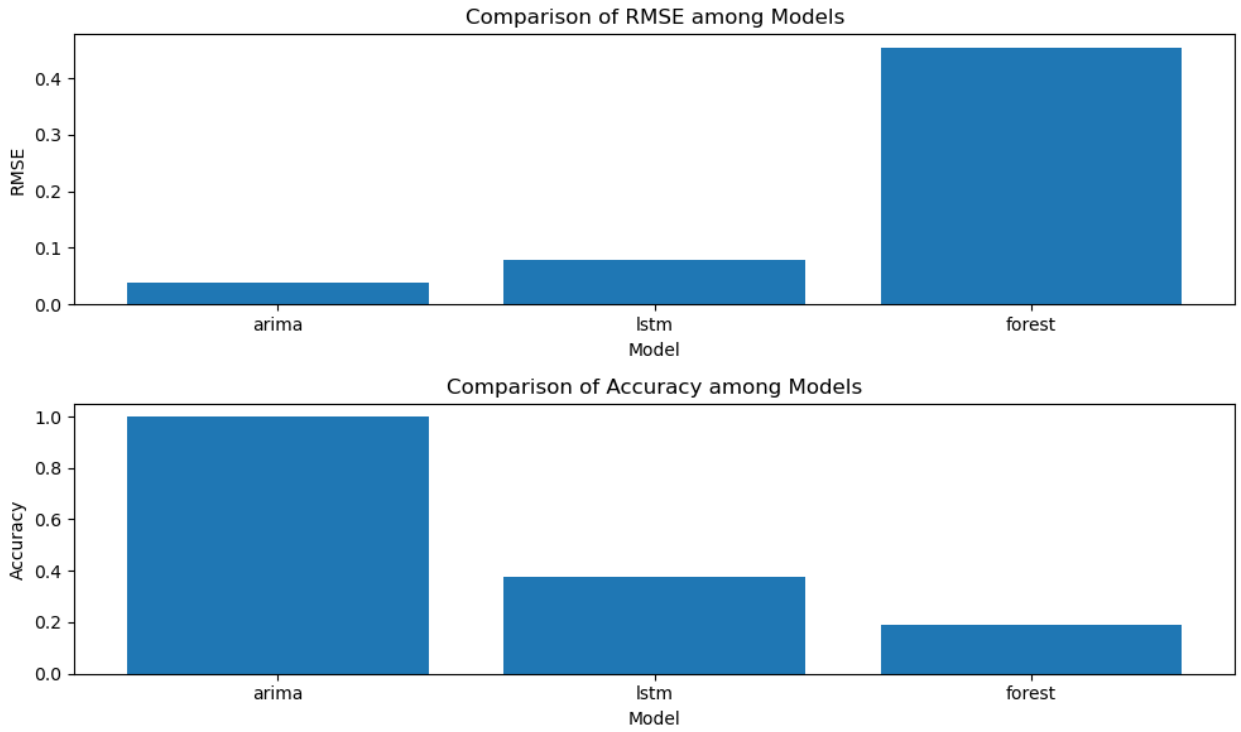

Рисунок 4.18 – Графік порівняння результатів ефективності моделей щодо прогнозування кількості пацієнтів в лікарнях

Незважаючи на те, що ARIMA є дуже потужною моделлю для прогнозування часових рядів, процеси підготовки даних та налаштування параметрів у результаті забирають чимало часу: потрібно зробити ряд стаціонарним, визначити значення p і q за допомогою графіків. Крім того, як було зазначено вище, при аналізі даних по пандемії COVID-19 доцільно використовувати динамічні параметри моделей, що потребує постійного перенавчання моделі для кожного окремого набору даних. Цей факт є причиною появи великої кількості досліджень, присвячених підбору параметрів моделі для тимчасових рядів, різноманітних за тривалістю та походженням.

ВИСНОВКИ

Прогнозування поширення епідемії коронавірусу COVID-19 стало глобальним викликом для науковців всього світу. У зв'язку з тим, що пандемія COVID-19 охопилапонад 200 країн світу, важливим завданням є розробка інформаційних модулів оцінки епідемічної небезпеки поширення COVID-19.

Прогнозування поширення епідемії коронавірусу COVID-19 має здійснюватися у реальному часі із можливостями вибору періоду прогнозування та методу прогнозування.

В ході проведеного дослідження було здійснено аналіз трьох моделей прогнозування даних для COVID-19. Першою моделлю, що була використана, була модель ARIMA. Вона показала один з найкращих результатів прогнозування. Модель ARIMA вдалося точно відтворити тенденції та рухи у графіку кількості пацієнтів в лікарнях, що свідчить про її високу точність та ефективність.

Наступною використаною моделлю була LTSM, яка є однією з найпоширеніших моделей для аналізу рядів динаміки. Вона продемонструвала задовільні результати і зуміла достатньо точно передбачити напрям та тенденцію руху кількості пацієнтів в лікарнях. LTSM модель зуміла відтворити складні залежності в даних і забезпечити достатню точність прогнозу.

Нарешті, модель DecisionForest була використана для аналізу даних Covid-19. Ця модель, яка базується на ансамблі дерев рішень, показала загалом правильне розпізнавання трендів і напрямку руху кількості пацієнтів в лікарнях. Цікавим є той факт, що модель DecisionForest навіть перетиналася з тестовим графіком в кінці часового ряду, що свідчить про її досить точні прогнози. Однак, порівняно з іншими моделями, DecisionForest модель показала найменший рівень точності прогнозу.

Отже, на підставі проведеного дослідження можна зробити висновок, що модель ARIMA була найефективнішою в прогнозуванні кількості пацієнтів в лікарнях, LTSM модель показала задовільні результати, а DecisionForest модель, хоча і демонструвала правильність трендів, мала менший рівень точності порівняно з іншими моделями.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Coronavirusdisease (COVID-19). [Електронний ресурс] Режим доступу: https://www.who.int/emergencies/diseases/novel-coronavirus-2019 (дата звернення 30.03.2023 р.)

2. Інформаційна панель ВООЗ щодо коронавірусу (COVID-19).[Електронний ресурс] Режим доступу: https://covid19.who.int/ (дата звернення 17.05.2023 р.)

3. WHO Director-General's opening remarks at the media briefing on COVID-19 - 13 July 2020. [Електронний ресурс] Режим доступу: https://www.who.int/directorgeneral/speeches/detail/who-director-general-s-opening-remarks-at-the-mediabriefing-on-covid-19---13-july-2020.(дата звернення 30.03.2023 р.)

4. Перший випадок нового коронавирусу 2019 р., пов'язаний з поїздками виявлено у США[Електронний ресурс] Режим доступу: https://www.cdc.gov/media/releases/2020/p0121-novel-coronavirus-travel-case.html(дата звернення 30.03.2023 р.)

5. Звіт головного санітарного лікаря Канади про стан охорони здоров'я в Канаді, 2020 р [Електронний ресурс] Режим доступу: https://www.canada.ca/en/public-health/news/2020/01/first-presumptive-confirmed-case-of-novel-coronavirus.html(дата звернення 30.03.2023 р.)

6. Декларація про оголошення надзвичайної ситуації у країні через спалах нової коронавірусної хвороби (COVID-19) [Електронний ресурс] Режим доступу: https://trumpwhitehouse.archives.gov/presidential-actions/proclamation-declaring-national-emergency-concerning-novel-coronavirus-disease-covid-19-outbreak/ (дата звернення 30.03.2023 р.)

7. Губернатор Каліфорнії ГевінНьюсом видав перший у США наказ залишатися вдома, щоб уповільнити поширення COVID-19 [Електронний ресурс] Режим доступу: https://www.gov.ca.gov/2020/03/19/governor-gavin-newsom-issues-stay-at-home-order/(дата звернення 30.03.2023 р.)

8. Подивіться, які штати та міста закликали жителів залишатися вдома [Електронний ресурс] Режим доступу: https://www.nytimes.com/interactive/2020/us/coronavirus-stay-at-home-order.html(дата звернення 30.03.2023 р.)

9. Щоденний брифінг/пандемія коронавірусу 19 [Електронний ресурс] Режим доступу: https://www.gov.ca.gov/2020/05/07/governor-newsom-provides-update-on-californias-progress-toward-stage-2-reopening/(дата звернення 30.03.2023 р.)

10. Перегляньте плани повторного відкриття та мандати масок для всіх 50 штатів[Електронний ресурс] Режим доступу: https://www.nytimes.com/interactive/2020/us/states-reopen-map-coronavirus.html(дата звернення 30.03.2023 р.)

11. Смерть Джорджа Флойда викликає масові протести проти расизму по всій території США, що викликає занепокоєння щодо потенційного впливу на передачу COVID-19 [Електронний ресурс] Режим доступу: https://www.bbc.com/news/world-us-canada-52877678(дата звернення 30.03.2023 р.)

12. FDA вживає ключових заходів у боротьбі з COVID-19, видаючи дозвіл на екстрене використання першої вакцини проти COVID-19[Електронний ресурс] Режим доступу: https://www.fda.gov/news-events/press-announcements/fda-takes-key-action-fight-against-covid-19-issuing-emergency-use-authorization-first-covid-191. (дата звернення 30.03.2023 р.)

13. Програма вакцинації від COVID-19. Операції юрисдикції. Тимчасове оперативне керівництво[Електронний ресурс] Режим доступу: https://www.cdc.gov/vaccines/covid-19/downloads/COVID-19-Vaccination-Program-Interim_Playbook.pdf(дата звернення 30.03.2023 р.)

14. Довіра до вакцини для кожного виглядає по-різному.[Електронний ресурс] Режим доступу:https://www.canada.ca/en/health-canada/news/2020/12/health-canada-authorizes-first-covid-19-vaccine.html1.(дата звернення 30.03.2023 р.)

15. Центр контролю та профілактики захворювань США (CDC) надає вичерпні дані про вакцинацію проти COVID-19 у Сполучених Штатах.[Електронний ресурс] Режим доступу: https://covid.cdc.gov/covid-data-tracker/#vaccinations(дата звернення 30.03.2023 р.)

16. Відстеження нового коронавірусу в США[Електронний ресурс] Режим доступу: https://www.reuters.com/article/us-health-coronavirus-usa-states-dUSKBN29X1D91.(дата звернення 30.03.2023 р.)

17. Як поширюється COVID-19 [Електронний ресурс] Режим доступу: https://www.cdc.gov/coronavirus/2019-ncov/transmission/variant.html(дата звернення 30.03.2023 р.)

18. Президент Байден оголошує, що всі американці матимуть право на вакцинацію від COVID-19 до 1 травня 2021 року, і окреслює подальші кроки для прискорення розповсюдження вакцини[Електронний ресурс] Режим доступу: https://www.whitehouse.gov/briefing-room/statements-releases/2021/03/11/fact-sheet-president-biden-to-announce-all-americans-to-be-eligible-for-vaccinations-by-may-1-steps-to-further-accelerate-vaccine-distribution-and-defeat-the-virus/(дата звернення 30.03.2023 р.)

19. Центр з контролю та профілактики захворювань США відстежує кількість вакцинацій проти COVID-19 у Сполучених Штатах, надаючи важливі дані про зусилля з розгортання вакцинації [Електронний ресурс] Режим доступу: https://covid.cdc.gov/covid-data-tracker/#vaccinations(дата звернення 30.03.2023 р.)

20. Всесвітня організація охорони здоров'я відстежує та контролює варіанти SARS-CoV-2, надаючи оновлену інформацію про їхній потенційний вплив на здоров'я населення. [Електронний ресурс] Режим доступу: https://www.who.int/en/activities/tracking-SARS-CoV-2-variants/(дата звернення 30.03.2023 р.)

21. Факти: країни оцінюють потребу в ревакцинації проти COVID-19 [Електронний ресурс] Режим доступу:

https://www.reuters.com/business/healthcare-pharmaceuticals/countries-weigh-mix-match-covid-19-vaccines-2021-06-14/(дата звернення 30.03.2023 р.)

22. Будьте в курсі вакцин проти COVID-19 [Електронний ресурс] Режим доступу: https://www.cdc.gov/coronavirus/2019-ncov/vaccines/booster-shot.html(дата звернення 30.03.2023 р.)

23. COVID-19: профілактика та ризики [Електронний ресурс] Режим доступу: URL:https://www.canada.ca/en/public-health/services/diseases/2019-novel-coronavirus-infection/prevention-risks/covid-19-vaccine-treatment/covid-19-vaccine-booster-dose.html(датазвернення 30.03.2023 р.)

24. Bostock, M., Ogievetsky, V., &Heer, J. (2011). D³: Data-driven documents. IEEE Transactions on Visualization and Computer Graphics, 17(12), 2301-2309. Retrieved from https://ieeexplore.ieee.org/abstract/document/6064993

25. McKinney, W. (2010). Data structures for statistical computing in Python. In Proceedings of the 9th Python in Science Conference, 51-56. Retrievedfrom[Електроннийресурс] Режимдоступу: https://conference.scipy.org/proceedings/scipy2010/pdfs/mckinney.pdf

26. Sustainability Metrics for Real Case Applications of the Supply ChainNetwork Design Problem: A Systematic Literature Reviewhttps [Електронний ресурс] Режим доступу: https://www.sciencedirect.com/science/article/pii/S2352771420300717

27. Chimmula, V. K. R., & Zhang, L. (2020). Time series forecasting of COVID-19 transmission in Canada using LSTM networks. Chaos, Solitons&Fract [Електронний ресурс] Режим доступу: https://pubmed.ncbi.nlm.nih.gov/32390691/

28. Novel Coronavirus Global Research and Innovation Forum: Towards a Research Roadmap.//WHO. [Електронний ресурс] Режим доступу: www.who.int/emergencies/diseases/novel-coronavirus-2019/global-research-on-novel-coronavirus-2019-ncov (датазвернення 30.03.2023 р.).

29. Wu J. T., Leung K., Leung G. M. Nowcasting and forecasting the potential domestic and international spread of the 2019-nCoV outbreak originating

in Wuhan, China: a modelling study. Lancet, 2020, vol. 395, iss. 10225, pp. 689–697. DOI:https://doi.org/10.1016/S0140-6736(20)30260-9

30. Mandal M., Jana S., Nandi S. K., Khatua A., Adak S., Kar T. K. A model based study on the dynamics of COVID-19: Prediction and Control. Chaos, Solitons and Fractals, 2020, vol. 136, no. 109889.DOI:https://doi.org/10.1016/j.chaos.2020.109889

31. Fanelli D., Piazza F. Analysis and forecast of COVID-19 spreading in China, Italy and France.Chaos, Solitons and Fractals, 2020, vol. 134, no. 109761. DOI:https://doi.org/10.1016/j.chaos.2020.109761

32. Bekirosab S., Kouloumpou D. SBDiEM: A new mathematical model of infectious disease dynamics.Chaos, Solitons and Fractals, 2020, vol. 136, no. 109828. https://doi.org/10.1016/j.chaos.2020.109828

33. Barmparis G. D., Tsironis G. P. Estimating the infection horizon of COVID-19 in eight countries with a data-driven approach. Chaos, Solitons and Fractals, 2020, vol. 135, no. 109842.https://doi.org/10.1016/j.chaos.2020.109842

34. Time Series Forecasting using ARIMA[Електроннийресурс] Режимдоступу: https://medium.com/@er.iit.pradeep09/time-series-forecasting-using-arima-caa84d52463b (датазвернення 20.05.2023)

35. AR, MA, and ARIMA Models: A Comprehensive Guide[Електроннийресурс] Режимдоступу: https://medium.com/srijit-mukherjee/ar-ma-and-arima-models-a-comprehensive-guide-46b1db00083 (датазвернення 20.05.2023)

36. Understanding LSTM Networks[Електроннийресурс] Режимдоступу: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (датазвернення 17.05.2023 р.)

37. The Ultimate Guide to Building Your Own LSTM Models [Електронний ресурс] Режим доступу: https://www.projectpro.io/article/lstm-model/832 (дата звернення 17.05.2023 р.)

38. National responses to the COVID-19 pandemic[Електроннийресурс] Режимдоступу:https://en.wikipedia.org/wiki/National_responses_to_the_COVID-19_pandemic#Other_countries_and_territories_4

39. Canadian COVID-19 Intervention Timeline[Електроннийресурс] Режимдоступу: https://www.cihi.ca/en/canadian-covid-19-intervention-timeline

40. COVID-19 datasetDatasetcoronaviruspandemic[Електроннийресурс] Режимдоступу: https://www.kaggle.com/datasets/georgesaavedra/covid19-dataset

41. Прогнозування соціально-економічних процесів : навчальний посібник для студентів напряму підготовки 6.030502 "Економічна кібернетика" денної форми навчання / Т. С. Клебанова, В. А. Курзенев, В. М. Наумов та ін. – Х. : ХНЕУ ім. С. Кузнеця, 2015. – 656 с. (Укр. мов.)

42. Галущак М. П., Галущак О. Я., Кужда Т. І. Прогнозування соціально-економічних процесів: навчальний посібник для економічних спеціальностей. – Тернопіль: ФОП Паляниця, 2021. – 160 с.

43. Дипломний проєкт. Методичні рекомендації для студентів спеціальності 126 "Інформаційні системи та технології" освітньої програми "Інформаційні системи та технології" першого (бакалаврського) рівня [Електронний ресурс] / уклад. С. Г. Удовенко, О. О. Тютюник, О. В. Гороховатський, В. А. Затхей; Харківський національний економічний університет ім. С. Кузнеця. – Електрон. текстові дан. (189 КБ). – Харків : ХНЕУ ім. С. Кузнеця, 2023. – 48 с.

44. Тютюник В. В. Особливості прийняття експертами ситуаційного центру управлінських антикризових рішень в умовах епідемічної небезпеки поширення COVID-19 / В. В. Тютюник, О. О. Тютюник, А. О. Долгий // Запобігання виникненню надзвичайних ситуацій, реагування та ліквідація їх наслідків : матеріали круглого столу (вебінару), 23 лютого 2023 р. – Харків, 2023.– С. 150-152.

ДОДАТОК А

Таблиця 1.1 – Хронологія основних подій розповсюдження COVID-19

№ з/п	Дата	Подія
1	2	3
1	21 січня 2020 року	Центр контролю та профілактики захворювань США (CDC) підтвердив перший випадок COVID-19, пов'язаний з подорожжю, у США, зокрема у штаті Вашингтон, після повернення пацієнта з Уханя, Китай [4]
2	25 січня 2020 року	Уряд Канади повідомляє про перший імовірно підтверджений випадок COVID-19 в Канаді у людини, яка подорожувала з Уханя, Китай, до Торонто[5]
3	13 березня 2020 року	Білий дім оголошує надзвичайний стан у США через спалах COVID-19, мобілізуючи федеральні ресурси для боротьби з пандемією [6]
4	19 березня 2020 року	Губернатор Каліфорнії Гевін Ньюсом видав перший у США наказ залишатися вдома, щоб уповільнити поширення COVID-19 [7]
5	Березень 2020 року	Американські штати починають оголошувати про обмеження через коронавірус та розпорядження залишатися вдома, щоб зменшити поширення вірусу [8]
6	8 травня 2020 року	Каліфорнія переходить до другого етапу свого модифікованого розпорядження про домашній режим, дозволяючи певним підприємствам знову відкритися з певними змінами [9]
7	Травень 2020 року	Усі 50 штатів США оголосили про плани відновлення роботисвоїх економік, дотримуючись різних рекомендацій та обмежень [10]
8	Червень 2020 року	Смерть Джорджа Флойда викликає масові протести проти расизму по всій території США, що викликає занепокоєння щодо потенційного впливу на передачу COVID-19 [11]
9	11 грудня 2020 року	Агентство з санітарного нагляду за якістю харчових продуктів і медикаментів США (FDA) видало дозвіл на екстрене використання вакцини проти COVID-19 компанії Pfizer-BioNTech, що стало важливою віхою в боротьбі з COVID-19 [12]
10	14 грудня 2020 року	Центр контролю та профілактики захворювань США (CDC) випустив посібник з програми вакцинації проти COVID-19, який містить інструкції для юрисдикцій щодо розподілу та застосування вакцин [13]
11	23 грудня 2020 року	Міністерство охорони здоров'я Канади дозволило вакцину проти COVID-19 компанії Pfizer-BioNTech для використання в Канаді після ретельної та незалежної наукової експертизи [14]

Продовження таблиці 1.1

1	2	3
12	Січень 2021 року	Центр контролю та профілактики захворювань США (CDC) надає вичерпні дані про вакцинацію проти COVID-19 у Сполучених Штатах [15]
13	28 січня 2021 року	Кілька штатів США починають послаблювати обмеження щодо COVID-19, незважаючи на попередження чиновників охорони здоров'я про постійний ризик [16]
14	Лютий 2021 року	CDC повідомляє про нові варіанти вірусу, що викликає COVID-19, наголошуючи на необхідності постійної пильності та дотримання заходів громадського контролю [17]
15	11 березня 2021 року	Президент Байден оголошує, що всі американці матимуть право на вакцинацію від COVID-19 до 1 травня 2021 року, і окреслює подальші кроки для прискорення розповсюдження вакцини [18]
16	Квітень 2021 року	Центр з контролю та профілактики захворювань США відстежує кількість вакцинацій проти COVID-19 у Сполучених Штатах, надаючи важливі дані про зусилля з розгортання вакцинації [19]
17	Травень 2021 року	Всесвітня організація охорони здоров'я відстежує та контролює варіанти SARS-CoV-2, надаючи оновлену інформацію про їхній потенційний вплив на здоров'я населення[20]
18	Червень 2021 року	Країни, включаючи США та Канаду, оцінюють переваги та ризики "змішаних" стратегій вакцинації проти COVID-19 [21]
19	Серпень 2021 року	Центр з контролю та профілактики захворювань США (CDC) оголошує настанови щодо бустерних щеплень проти COVID-19, рекомендуючи їх певним групам населення з підвищеним ризиком захворювання [22]
20	Вересень 2021 р.	Міністерство охорони здоров'я Канади окреслює рекомендації щодо бустерних доз вакцини проти COVID-19 на основі нових доказів та порад експертів [23]

Лістинг програмного коду

```
# Importing the libraries
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import TimeSeriesSplit, train_test_split
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.stattools import adfuller
from imblearn.ensemble import BalancedRandomForestClassifier
import matplotlib.pyplot as plt
import tensorflow as tf
import seaborn as sns

# Importing the dataset
models_output = {}
dataset = pd.read_csv('owid-covid-data.csv')

# Data preprocessing
dataset = dataset[dataset['continent'] == 'North America'][dataset['location'] == 'Canada'][['date','total_cases', 'new_cases', 'total_deaths', 'new_deaths', 'reproduction_rate', 'icu_patients', 'hosp_patients', 'new_tests', 'total_tests', 'positive_rate', 'tests_per_case', 'total_vaccinations',
    'people_vaccinated', 'people_fully_vaccinated', 'total_boosters',
    'new_vaccinations', 'stringency_index']]

# Data Filtering
dataset['date'] = pd.to_datetime(dataset['date'])
dataset = dataset.set_index('date')
dataset = dataset.sort_index()

plt.figure(figsize=(10, 5))
plt.xlabel('Date')
plt.ylabel('Hospital Patients')
plt.title('Number of Hospital Patients')
plt.plot(dataset['hosp_patients'], color='blue')
plt.show()

# Data Filtering
dataset.loc[pd.to_datetime('2020-12-14')]['new_vaccinations'] = dataset.loc[pd.to_datetime('2020-12-14')]['total_vaccinations']

dataset.loc['2022-02-12']['new_vaccinations'] = dataset.loc['2022-02-11']['new_vaccinations'] + ( dataset.loc['2022-02-14']['new_vaccinations'] - dataset.loc['2022-02-11']['new_vaccinations'] ) / 4
dataset.loc['2022-02-13']['new_vaccinations'] = dataset.loc['2022-02-12']['new_vaccinations'] + ( dataset.loc['2022-02-14']['new_vaccinations'] - dataset.loc['2022-02-11']['new_vaccinations'] ) / 4
```

dataset.loc['2020-03-10']['new_tests'] = np.floor(dataset.loc['2020-03-09']['new_tests'] + (dataset.loc['2020-03-12']['new_tests'] - dataset.loc['2020-03-09']['new_tests']) / 4)
dataset.loc['2020-03-11']['new_tests'] = np.floor(dataset.loc['2020-03-10']['new_tests'] + (dataset.loc['2020-03-12']['new_tests'] - dataset.loc['2020-03-09']['new_tests']) / 4)

dataset.loc['2020-03-02']['new_tests'] = np.floor(dataset.loc['2020-03-01']['new_tests'] + (dataset.loc['2020-03-06']['new_tests'] - dataset.loc['2020-03-01']['new_tests']) / 6)
dataset.loc['2020-03-03']['new_tests'] = np.floor(dataset.loc['2020-03-02']['new_tests'] + (dataset.loc['2020-03-06']['new_tests'] - dataset.loc['2020-03-01']['new_tests']) / 6)
dataset.loc['2020-03-04']['new_tests'] = np.floor(dataset.loc['2020-03-03']['new_tests'] + (dataset.loc['2020-03-06']['new_tests'] - dataset.loc['2020-03-01']['new_tests']) / 6)
dataset.loc['2020-03-05']['new_tests'] = np.floor(dataset.loc['2020-03-04']['new_tests'] + (dataset.loc['2020-03-06']['new_tests'] - dataset.loc['2020-03-01']['new_tests']) / 6)

dataset.loc['2022-02-12', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boosters']] = dataset.loc['2022-02-11':'2022-02-13', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boosters']].mean()
dataset.loc['2022-02-12', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boosters']] = dataset.loc['2022-02-11':'2022-02-13', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boosters']].mean()
dataset.loc['2022-02-12', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boosters']] = dataset.loc['2022-02-11':'2022-02-13', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boosters']].mean()
dataset.loc['2022-03-01', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boosters', 'new_vaccinations']] = dataset.loc['2022-02-28', ['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boosters', 'new_vaccinations']]

dataset.loc['2020-03-02', ['total_tests']] = dataset.loc['2020-03-01':'2020-03-03', ['total_tests']].mean()
dataset.loc['2020-03-04', ['total_tests']] = dataset.loc['2020-03-03':'2020-03-05', ['total_tests']].mean()
dataset.loc['2020-03-10', ['total_tests']] = dataset.loc['2020-03-09':'2020-03-11', ['total_tests']].mean()

missing_Rt = dataset.loc['2020-02-29':'2020-03-12', ['new_cases', 'total_cases']]
missing_Rt = missing_Rt.assign(reproduction_rate = missing_Rt['new_cases'] / missing_Rt['total_cases'].shift(1) + 2)['reproduction_rate'].to_frame()

dataset.loc['2020-03-01':'2020-03-11', ['reproduction_rate']] = missing_Rt['2020-03-01':'2020-03-11']

dataset.loc[(dataset.index >= pd.to_datetime('2022-02-26')) & (dataset.index <= pd.to_datetime('2022-03-01')), ['stringency_index']] = dataset.loc[(dataset.index >= pd.to_datetime('2022-02-26')) & (dataset.index <= pd.to_datetime('2022-03-01')), ['stringency_index']].fillna(dataset.loc['2022-02-25', ['stringency_index']])

dataset[['total_deaths', 'new_deaths', 'icu_patients', 'hosp_patients', 'total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boosters', 'new_vaccinations']] = dataset[['total_deaths', 'new_deaths', 'icu_patients', 'hosp_patients', 'total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated', 'total_boosters', 'new_vaccinations']].fillna(0)

```
        #Add lockdown column
        dataset["is_lockdown"] = 0
        dataset["is_lockdown"] = dataset["is_lockdown"].astype(int)
        dataset.loc['2020-03-17':'2020-05-18', ["is_lockdown"]] = 1
        dataset.loc['2020-11-07':'2021-01-23', ["is_lockdown"]] = 1
        dataset.loc['2021-04-03':'2021-06-02', ["is_lockdown"]] = 1

        #Dataset is truncated to the period of the pandemic
        dataset = dataset.truncate(before=pd.to_datetime('2020-03-01'),
after=pd.to_datetime('2022-03-01'))

        #Check for missing values
        assert not dataset.isnull().sum().sum(), "There are still missing values in the dataset."

        print(dataset.columns)

        # Data Preprocessing for ARIMA
        arima_data = dataset['hosp_patients'].copy().to_frame()
        arima_data.index.freq = pd.infer_freq(arima_data.index)
        arima_data = (arima_data - arima_data.min()) / (arima_data.max() - arima_data.min())
        arima_train, arima_test = train_test_split(arima_data, test_size=0.05, shuffle=False)

        plt.figure(figsize=(10, 5))
        plt.xlabel('Date')
        plt.ylabel('Hospital Patients')
        plt.title('Number of Hospital Patients')
        plt.plot(arima_train, label="Train Data", color='blue')
        plt.plot(arima_test, label="Test Data", color='green')
        plt.legend()
        plt.show()

        import statsmodels.api as sm

        #Seasonal Decomposition
        decomposition = sm.tsa.seasonal_decompose(arima_data, model='additive', period=12)  #
Adjust the period based on the expected seasonality

        fig, (ax1, ax2, ax3, ax4) = plt.subplots(4, 1, figsize=(10, 8))

        ax1.plot(arima_data)
        ax1.set_ylabel('Original')

        ax2.plot(arima_data.index, decomposition.trend)
        ax2.set_ylabel('Trend')

        ax3.plot(arima_data.index, decomposition.seasonal)
        ax3.set_ylabel('Seasonal')

        ax4.plot(arima_data.index, decomposition.resid)
        ax4.set_ylabel('Residual')

        plt.tight_layout()
```

```
plt.show()

plt.figure(figsize=(10, 5))
plt.xlabel('Date')
plt.ylabel('Hospital Patients')
plt.title('Number of Hospital Patients')
plt.plot(arima_train, label="'Train Data", color='blue')
plt.plot(arima_test, label="'Test Data", color='green')
plt.legend()
plt.show()

# Check for stationarity
def check_stationarity(timeseries):
    # Perform Dickey-Fuller test
    print('Results of Dickey-Fuller Test:')
    dftest = adfuller(timeseries, autolag='AIC')
      dfoutput = pd.Series(dftest[0:4], index=['Test Statistic', 'p-value', '#Lags Used',
'Number of Observations Used'])
    for key, value in dftest[4].items():
       dfoutput['Critical Value (%s)' % key] = value
    print(dfoutput)

check_stationarity(arima_data)

# Differencing
arima_data['hosp_patients_diff'] = arima_data['hosp_patients'].diff()
arima_data['hosp_patients_diff_2'] = arima_data['hosp_patients'].diff().diff()

# Plot difference data
fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize=(15,7))
ax1.plot(arima_data.index, arima_data.hosp_patients, color='blue')
ax1.set_title('Original Data')
ax2.plot(arima_data.index, arima_data.hosp_patients_diff, color='blue')
ax2.set_title('1st Difference')
ax3.plot(arima_data.index, arima_data.hosp_patients_diff_2, color='blue')
ax3.set_title('2nd Difference')
plt.show()

from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
import matplotlib.pyplot as plt

#Check ACF and PACF
def plot_acf_pacf(data, lags=40):
    # Autocorrelation
    fig, ax = plt.subplots(2, 1, figsize=(8, 8))
    plot_acf(data, lags=lags, ax=ax[0])
    ax[0].set_title('Autocorrelation')
    ax[0].set_ylabel('ACF')

    # Partial Autocorrelation
    plot_pacf(data, lags=lags, ax=ax[1])
    ax[1].set_title('Partial Autocorrelation')
```

```
        ax[1].set_ylabel('PACF')

        plt.tight_layout()
        plt.show()

    plot_acf_pacf(arima_data.hosp_patients, lags=40)

    # Searching from best p, d, q values for ARIMA
    from numpy import sqrt
    from sklearn.metrics import mean_squared_error
    from statsmodels.tools.sm_exceptions import ConvergenceWarning

    p_params = [1, 2]
    d_params = [1,2,3,4]
    q_params = [1,2,3]
    best_rmse = 9999
    best_params = None
    result_val = []

    for p_val in p_params:
        for d_val in d_params:
            for q_val in q_params:
                model = ARIMA(arima_train, order = (p_val, d_val, q_val))
                model_fit = model.fit()
                 predictions = model_fit.predict(start=arima_test.index[0], end=arima_test.index[-
1]).to_frame()
                rmse = sqrt(mean_squared_error(arima_test, predictions.predicted_mean))
                result_val.append((rmse, (p_val, d_val, q_val)))
                if rmse <= best_rmse:
                    best_rmse = rmse
                    best_params = (p_val, d_val, q_val)

    #Running ARIMA model with best parameters
    def run_arima_model(model_params):
        model = ARIMA(arima_train, order = model_params)
        model_fit = model.fit()
        print(model_fit.summary())
            predictions = model_fit.predict(start=arima_test.index[0], end=arima_test.index[-
1]).to_frame()
        rmse = sqrt(mean_squared_error(arima_test, predictions.predicted_mean))
        print("RSME:", rmse)
        plt.figure(figsize=(10, 5))
        plt.plot(arima_train, color='blue', label='Original train data')
        plt.plot(arima_test, color='yellow', label='Original test data')
         plt.plot(arima_test.index, predictions.predicted_mean, color='green', label='Predicted
data')
        plt.title('ARIMA Model - Predicted vs True')
        plt.legend()
        plt.show()
        return rmse, np.mean(np.abs(predictions.values - arima_test.values) < 10**(-1))

    _ = run_arima_model(best_params)
```

```
        models_output["arima"] = {'RMSE': _[0], 'Accuracy': _[1]}

        # Data Preprocessing for LTSM
        X_lstm = dataset.drop(columns=['hosp_patients']).copy()
        indexes = X_lstm.index
        Y_lstm = dataset['hosp_patients'].copy().to_frame()
        Y_lstm.index.freq = pd.infer_freq(Y_lstm.index)
        Y_lstm = (Y_lstm - Y_lstm.min()) / (Y_lstm.max() - Y_lstm.min())

        scaler = MinMaxScaler()

        X_lstm = scaler.fit_transform(X_lstm)
        Y_lstm = scaler.fit_transform(Y_lstm)

        # Reshape X to match the expected input shape of the LSTM layer
        X_lstm_reshaped = np.reshape(X_lstm, (X_lstm.shape[0], X_lstm.shape[1], 1))

        # Split the data into training and testing sets
        X_lstm_train, X_lstm_test, Y_lstm_train, Y_lstm_test =
train_test_split(X_lstm_reshaped, Y_lstm, test_size=0.05, shuffle=False)

        from keras.constraints import non_neg
        from keras import regularizers

        # Train the LTSM model
        from keras.models import Sequential
        from keras.layers import Dense, LSTM, Dropout, Reshape
        model = Sequential()
        model.add(LSTM(50, activation='relu',return_sequences=True,
input_shape=(X_lstm.shape[1], 1)))
        model.add(LSTM(50, activation='relu',return_sequences=True, recurrent_dropout=0.2))
        model.add(LSTM(50, activation='relu', recurrent_dropout=0.2 ))
        model.add(Dense(1, kernel_constraint=non_neg()))
        model.compile(loss='mean_squared_error', optimizer='adam')

        #Fit model
        model.fit(X_lstm_train, Y_lstm_train, epochs=400, batch_size=32, verbose=2)

        # Make predictions on the test data
        y_pred = model.predict(X_lstm_test.reshape((X_lstm_test.shape[0],
X_lstm_test.shape[1], 1)), verbose=2).reshape(-1,1)

        # Invert the scaling to get the original values
        scaler = MinMaxScaler()

        y_true = Y_lstm_test

        # Calculate root mean squared error (RMSE)
        mse = mean_squared_error(y_true, y_pred)
        rmse = np.sqrt(mse)
        print("Root Mean Squared Error (RMSE):", rmse)
```

```
        # Plot the predicted and true values
        plt.figure(figsize=(10, 5))
        plt.plot(indexes[:Y_lstm_train.shape[0]], Y_lstm_train, color='blue', label='Original train
data')
        plt.plot(indexes[Y_lstm_train.shape[0]:], Y_lstm_test, color='yellow', label='Original test
data')
        plt.plot(indexes[Y_lstm_train.shape[0]:], y_pred, color='green', label='Predicted data')
        plt.xlabel('Time')
        plt.ylabel('Value')
        plt.title('LSTM Model - Predicted vs True')
        plt.legend()
        plt.show()

        models_output["lstm"] = {'RMSE': rmse, 'Accuracy': np.mean(np.abs(y_pred - y_true) <
10**(-1))}

        # Data Preprocessing for Random Forest
        X_forest = dataset.drop(columns=['hosp_patients']).copy()
        indexes = X_forest.index
        Y_forest = dataset['hosp_patients'].copy().to_frame()

        # Normalize the data
        scaler = MinMaxScaler()

        # Splitting the dataset into train and test sets
        X_forest_train, X_forest_test, Y_forest_train, Y_forest_test = train_test_split(X_forest,
Y_forest, test_size=0.05, shuffle=False)

        # Creating a BalancedRandomForestClassifier model with specified parameters
        new_model = BalancedRandomForestClassifier(n_estimators=2000, random_state=0,
class_weight="balanced_subsample")

        # Fitting the model on the training data
        new_model.fit(X_forest_train, Y_forest_train)

        # Making predictions on the test data
        y_pred = new_model.predict(X_forest_test)
        predictions_series = pd.Series(y_pred, index=X_forest_test.index)

        # Adjust the window size to get a smoother curve
        window_size =  3

        # Smoothing the predictions using rolling mean with the specified window size
        smoothed_predictions = predictions_series.rolling(window=window_size,
min_periods=1).mean()
        y_pred = smoothed_predictions

        plt.figure(figsize=(10, 5))
        plt.plot(Y_forest_train, color='blue', label='Original train data')
        plt.plot(Y_forest_test, color='yellow', label='Original test data')
        plt.plot(indexes[Y_forest_train.shape[0]:], y_pred, color='green', label='Predicted data')
        plt.xlabel('Time')
```

```
plt.ylabel('Value')
plt.title('Balanced Random Forest Model - Predicted vs True')
plt.legend()
plt.show()

# Normalizing Y_forest_test using the scaler
scaler.fit(Y_forest_test)
Y_forest_test = scaler.transform(Y_forest_test)

# Transforming y_pred to match the normalized scale of Y_forest_test
y_pred = scaler.transform(y_pred.to_numpy().reshape(-1, 1))

# Calculating the root mean squared error (RMSE) between Y_forest_test and y_pred
mse = mean_squared_error(Y_forest_test, y_pred)
rmse = np.sqrt(mse)
print("Root Mean Squared Error (RMSE):", rmse)

models_output["forest"] = {'RMSE': rmse, 'Accuracy': np.mean(np.abs(y_pred - Y_forest_test) < 10**(-1))}

#Extract the model names, RMSE, and accuracy values
model_names = list(models_output.keys())
rmse_values = [models_output[model]['RMSE'] for model in model_names]
accuracy_values = [models_output[model]['Accuracy'] for model in model_names]

#Set the figure size
plt.figure(figsize=(10, 6))

#Plotting the RMSE values
plt.subplot(2, 1, 1)
plt.bar(model_names, rmse_values)
plt.xlabel('Model')
plt.ylabel('RMSE')
plt.title('Comparison of RMSE among Models')

#Plotting the accuracy values
plt.subplot(2, 1, 2)
plt.bar(model_names, accuracy_values)
plt.xlabel('Model')
plt.ylabel('Accuracy')
plt.title('Comparison of Accuracy among Models')

#Adjust the layout
plt.tight_layout()

#Display the plot
plt.show()
```