МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КРИВОРІЗЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ
Фізико-математичний факультет
Кафедра інформатики та прикладної математики

«Допущено до захисту»
В.о. завідувача кафедри
_____ Семеріков С.О.
«____» _____ 2022 р.

Реєстраційний № _____
«____» _____ 2022 р.

ПРОГНОЗУВАННЯ МАЙБУТНІХ ЗНАЧЕНЬ ФІНАНСОВИХ РИНКІВ ІЗ ВИКОРИСТАННЯМ РЕКУРЕНТНИХ НЕЙРОННИХ МЕРЕЖ ТА ЇХ МОДИФІКАЦІЙ

Кваліфікаційна робота студента
групи Ім-17
ступінь вищої освіти «магістр»
спеціальності 014 Середня освіта (Інформатика)
Кравця Олексія Сергійовича

Керівник: доктор фізико-математичних наук, професор
Соловйов Володимир Миколайович

Оцінка:
Національна шкала _____
Шкала ECTS ___ Кількість балів ___

Голова ЕК _____ _____
Члени ЕК _____ _____
_____ _____
_____ _____

Кривий Ріг — 2022

ЗАПЕВНЕННЯ

Я, Кравець Олексій Сергійович, розумію і підтримую політику Криворізького державного педагогічного університету з академічної доброчесності. Запевняю, що ця кваліфікаційна робота виконана самостійно, не містить академічного плагіату, фабрикації, фальсифікації. Я не надавав і не одержував недозволену допомогу під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають покликання на відповідне джерело.

Із чинним Положенням про запобігання та виявлення академічного плагіату в роботах здобувачів вищої освіти Криворізького державного педагогічного університету ознайомлений. Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі порушення академічної доброчесності робота не допускається до захисту або оцінюється незадовільно.

_____

(підпис)

ЗМІСТ

ВСТУП

Прогнозування часових рядів є однією з найважливіших задач машинного навчання та штучного інтелекту, так як удосконалення методів прогнозування дозволить більш точно передбачити вплив різних факторів у багатьох галузях. Основною метою будь-якого прогнозу є побудова, ідентифікація та перевірка моделі поведінки часового ряду.

В наш час поряд з традиційними методами прогнозування широко використовуються методи машинного навчання, з яких найбільш популярними є нейромережеві моделі. Вони добре себе показали у задачах класифікації, розпізнавання образів та аналізу часових рядів. За допомогою нейромереж є можливим моделювання нелінійної залежності майбутнього значення часового ряду від його попередніх значень і від значень зовнішніх факторів [26]. Відмінність цього підходу від стандартних полягає у тому, що нейромережева модель навчається самостійно, що є важливим при вирішенні складних задач.

За допомогою нейронних мереж можна розпізнавати об’єкти, мову, моделювати процеси побудови клітин та відновлювати пошкоджені дані. Нейромережі широко використовуються у сфері економіки та фінансів: за їх допомогою стає можливим оптимізація матеріальних та робочих ресурсів, які витрачаються на специфічні задачі цих галузей.

Актуальність теми. Задача прогнозування часових рядів є досить актуальною у багатьох предметних областях. Завдяки високій швидкості та точності прогнозування такі методи прогнозування можуть допомогти знизити робочі та матеріальні затрати . Нейронні мережі широко використовуються у торгівлі акціями для отримання прибутку, для алгоритмів розпізнавання обличчя та моделюванні хімічних процесів у медицині. Прогнозування часових рядів є найпопулярнішою

задачею машинного навчання, а штучні нейронні мережі є одним із найперспективніших напрямів у області аналізу даних.

Метою кваліфікаційної роботи є прогнозування майбутніх значень часових рядів, представлених індексами нафтопереробних компаній за допомогою нейромережних моделей.

Для досягнення мети необхідно виконати наступні **завдання**:

1. провести аналіз предметної області глибокого навчання та прогнозування значень часових рядів;
2. проаналізувати способи прогнозування часових рядів;
3. проаналізувати принципи роботи нейронних мереж;
4. реалізувати та навчити нейромережні моделі LSTM, GRU та RNN;
5. дослідити результати навчання, порівняти результати виконання та точність вихідних даних.

Об’єкт дослідження: сегмент світового фондового ринку, представлених індексами нафтопереробних компаній.

Предмет дослідження: методи інтелектуального аналізу даних з використанням штучних нейронних мереж та глибокого навчання - рекурентні та згорткові нейронні мережі.

Структура роботи. Кваліфікаційна робота складається зі вступу, трьох розділів, висновків до кожного з розділів, висновку , списку використаних джерел

РОЗДІЛ 1. ДОСЛІДЖЕННЯ КОНЦЕПЦІЙ ГЛИБОКОГО НАВЧАННЯ ДЛЯ ЗАДАЧ ПРОГНОЗУВАННЯ МАЙБУТНІХ ЗНАЧЕНЬ ЧАСОВИХ РЯДІВ

1.1 Означення та історія розвитку нейронних мереж

У сучасному світі технології штучного інтелекту охоплюють практично всі сфери нашого життя і представлені як досягнення наднових. Насправді концепція штучного інтелекту з'явилася в 1950-х роках і з тих пір основні питання практично не змінилися.

До 1949 року в комп'ютерах був відсутній ключовий елемент, що дозволяє говорити про інтелект, а саме пам'ять. Вони могли виконувати команди, але не могли зберігати інформацію про вже виконані операції. До того ж використання таких комп'ютерів було доступне тільки найпрестижнішим університетам і великим технічним компаніям. Серйозні дослідження в області штучного інтелекту вимагали значного фінансування, яке могло стати реальністю тільки при наявності перших вагомих результатів в цій області, зображень розумної комп'ютерної програми в науковій фантастиці. Однак перші успіхи алгоритмів штучного інтелекту продемонстрували величезну кількість перешкод на шляху створення машинного інтелекту. Найбільшою проблемою була відсутність обчислювальної потужності. Наприклад, щоб діяти як віртуальний співрозмовник, машина повинна запам'ятовувати значення багатьох слів і розуміти їх значення в багатьох комбінаціях. Комп'ютери в той час не мали технічної можливості зберігати необхідний обсяг інформації і досить швидко її обробляти.

Уряди різних країн, зацікавлені у відповідних технологіях, почали активно фінансувати галузь. У першу чергу це було зумовлено цікавістю до застосування даних технологій у воєнній галузі. Однак уже в 1987 році фінансування практично припинилося, і розробка цих технологій практично не проводилася до 1993 року. Протягом 1990-х і 2000-х років штучний інтелект розвивався, незважаючи на відсутність державного фінансування і суспільної уваги. Основна роль у розвитку технологій машинного навчання була передана IT-гігантам, яким вдалося домогтися значних успіхів на шляху до розумної машини. Наприклад, в 1997 році комп'ютерна програма від IBM перемогла чинного чемпіона світу Гаррі Каспарова з шахів. Звичайно, у вчених є деякі уявлення про подальші перспективи в області штучного

інтелекту, але поки немає очевидного способу створення по-справжньому інтелектуальних і універсальних машин.

1.1.1 Біологічний нейрон

На початку досліджень в області штучного інтелекту вчені ставили за мету відтворити людський інтелект для вирішення конкретних задач. Ідея створення штучних нейронних мереж полягала у відтворені діяльності нейрону (рис. 1.1).

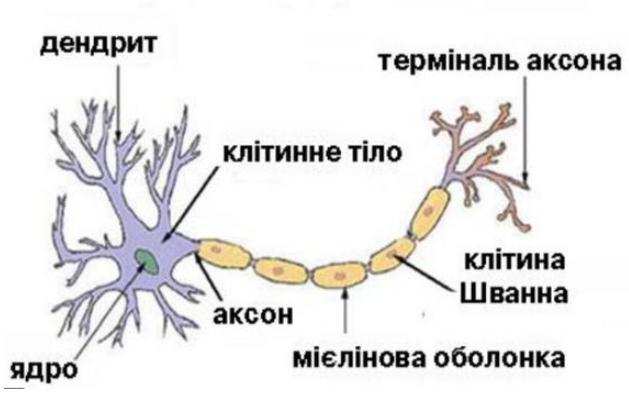


Рис. 1.1 – Типова структура нейрона

Як бачимо з рисунка, кожен нейрон має декілька типів нервових волокон: дендрити, які отримують інформацію в вигляді імпульсів і аксон, який виконує передавальну функцію.

У нейрона є декілька функцій [21]:

- Функція прийому: синапси отримують інформацію;
- інтеграційна функція: на виході нейрону отримуємо сигнал , який містить суму сигналів;
- провідникова функція: по аксону проходить інформація до синапсу;
- функція передачі: імпульс посилає медіатор до іншого нейрону.

Очевидно, що швидкість відповіді конкретного нейрона мережі є меншою ніж у цифрових, але завдяки цьому мозок здатний до розв’язання до комплексних задач, як наприклад обробка зорової інформації та мовлення. Це досягається за допомогою великого об’єму повільних нейронів, які формують близько 100 зв'язків з іншими

нейронами[13]. Завдяки такій кількості нейронних зв'язків біологічні нейронні мережі здатні до розподілення задач, які можуть вирішуватися одночасно на різних рівнях системи. Прийнято вважати, що при потраплянні сигналу на аксон від декількох дендридів нейрон активується і передає імпульс далі, сила якого є сумою вхідних сигналів, які він отримав. Таким чином може визначатись пріорітетність задач, які опрацьовує нейронна мережа. Одним із основних механізмів вирішення складних комплексних завдань є розбиття останнього на безліч субзавдань, які оброблюються одночасно. Це чимось нагадує багатопотоковість у сучасних процесорах, де обчислювальний елемент швидко переключається між завданнями для їх одночасного вирішення виділяючи рівні долі процесорного часу на кожну із них.

Хоча робота біологічних нейронних мереж є досить розвиненою для підтримки функціонування і розвитку живих організмів у неї є свої недоліки. Наприклад у людському організмі мозок є найбільшим споживачем енергії і в той же час найвразливішим органом у порівняні з іншими. Якщо нервова система будь-якого живого організму не буде отримувати достатньо ресурсів для стабільної роботи це може призвести до непоправних пошкоджень.

1.1.2 Штучний нейрон

Штучний нейрон є структурним елементом штучної нейронної мережі, який є базовим обчислювальним елементом і являє собою математичну модель біологічного нейрона[13].

Штучний нейрон має декілька входів (аналог дендридів біологічного нейрона) і єдиний вихід (аналог аксона). Кожен вхід має деяку вагу, який помножується на ваговий коеффіцієнт w_i, який поступив по даному входу. В тілі нейрона відбувається додавання взважених входів, після чого отримана сума S перетворюється за допомогою активаційної, у більшості випадків нелінійної функції нейрона. Таким чином роботу штучного нейрона можна описати за формулою:

$$S = \sum_{i=1}^{n} w_i\, x_i + w_0 x_0$$

Де n – розмірність вхідного вектора, w_i – ваговий кофефіцієнт i-го входу нейрона,

x_i – значення, яке проходить по і входу нейрона, w_0 та x_0 є додатковими ваговими коефіцієнтами та входом відповідно, необхідним для ініціалізації нейрона.

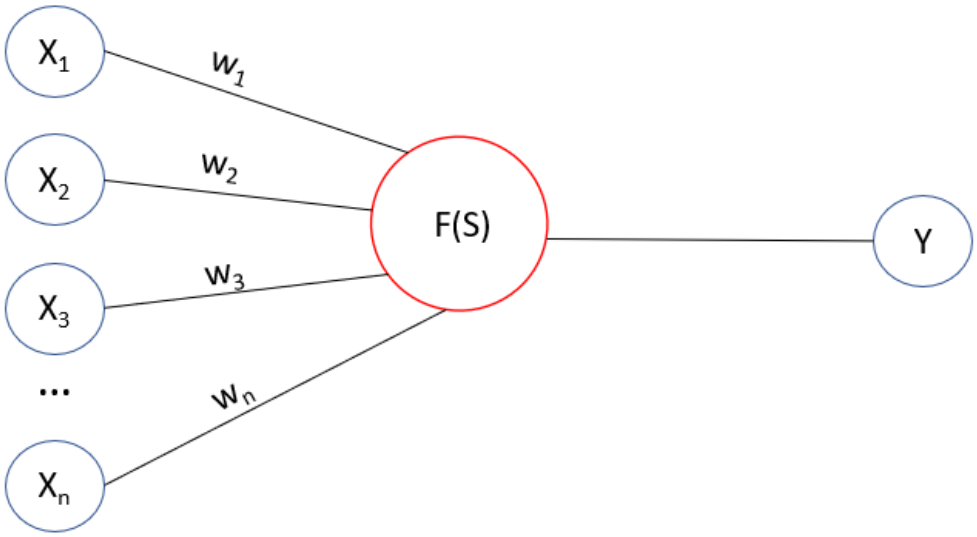


Рис. 1.2 – Типова структура штучного нейрона

Функція F(S) є нелінійною і визначає вихідне значення нейрона в залежності від суми входів і порогового значення. Одними з найпоширеніших функцій активації є[28]:

- Лінійна;
- ступінчаста;
- сигмоїдальна;
- ReLU;
- гіперболічний тангенс.

Лінійна функція активації містить дві лінійних ділянки, де її значення дорівнює мінімально і максимально допустимому значенню. Також функція містить ділянку, на якій вона монотонно зростає.

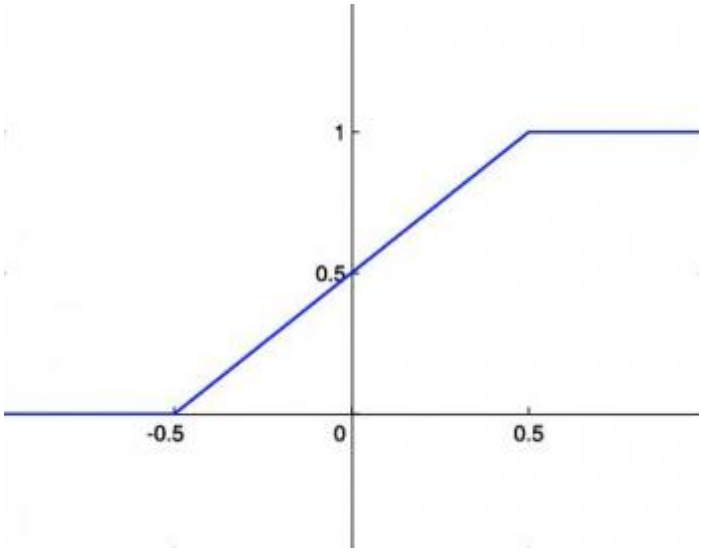


Рис. 1.3 – Лінійна функція

Ця функція дає певний спектр значень, а не бінарну відповідь. Якщо в мережі було активовано декілька нейронів одночасно, то відповідь видається на основі максимального значення[16].

Ступінчаста функція працює на основі певного порогового значення *Y* – якщо значення більше за Y, то нейрон можна вважати активованим, якщо менше за Y, то нейрон неактивний.

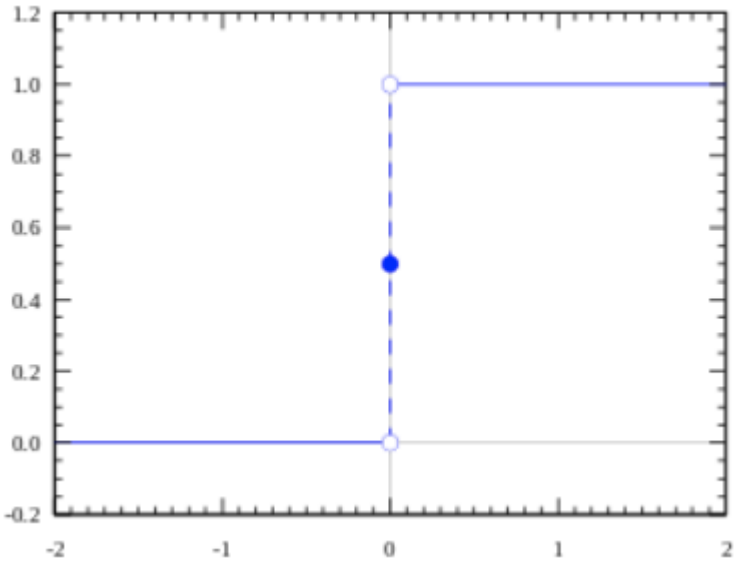


Рис. 1.3 – Ступінчаста функція

Ступінчасту функцію використовують для створення класифікаторів, так як вона повертає значення 0 або 1. Але у цьому є і певні недоліки: якщо для визначення дається більше ніж 2 класи, то одночасно від декількох нейронів штучної нейронної мережі може бути отримано значення 1. В такому разі незрозуміло, до якого саме

класу відноситься дане значення. Виходить, що для точного визначення класу предмета має активовуватися лише нейрон, який відноситься до даного класу, а значення інших має дорівнювати 0[16].

Сигмоїдна функція - це гладка монотонна зростаюча нелінійна функція [4].

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

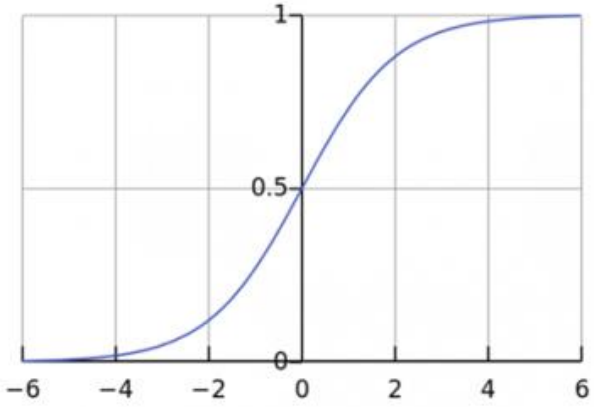


Рис. 1.4 – Сигмоїдна функція

Розглянемо її переваги. По-перше, сигмоїда має нелінійний характер, а результат поєднання набору таких функцій є нелінійним, що дозволяє побудувати мережу з декількох шарів. Основна перевага цієї функції полягає в тому, що вона не є двійковою, на відміну від покрокової. Сигмоїдальна функція також характеризується плавним градієнтом. *У* діапазоні значень *X* (-2; 2) значення *Y* дуже швидко змінюються. Це означає, що будь-яка незначна зміна значення *X* у цій області тягне за собою значну зміну значення Y. Така поведінка функції вказує на те, що *Y* прагне до одного з країв діапазону.

На даний момент сигмоїд є однією з найбільш використовуваних функцій активації в нейронних мережах.

ReLU функція.

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

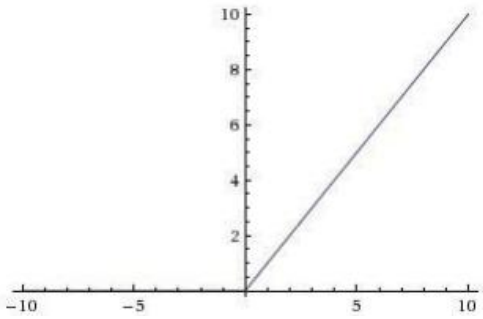


Рис. 1.5 – ReLU функція

Ця функція вважається найбільш успішною і широко використовуваною передавальною функцією. ReLU показує кращу продуктивність в глибокому навчанні порівняно з сігмоїдою та гіперболічним тангенсом. ReLU являє собою майже лінійну функцію і тому зберігає властивості лінійних моделей, які роблять їх легко оптимізувати методами градієнтного спуску. Основна перевага використання ReLU в обчисленні полягає в тому, що функція не потребує обчислення експоненти чи ділення, отже гарантує більш швидке виконання.

Гіперболічний тангенс.

$$f(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$

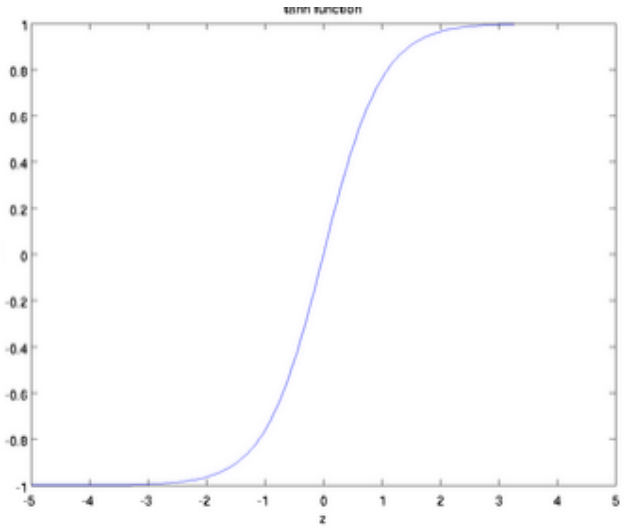


Рис. 1.6 – Гіперболічний тангенс

Іншою часто використовуваною функцією активації є **гіперболічний тангенс**. Гіперболічний тангенс є скорегованою сигмовидною функцією [4]. Ця функція має ті самі характеристики, що і сигмоїдна, про яку говорилося раніше. Вона також має нелінійний характер, добре підходить для поєднання шарів та визначена в діапазоні (-1, 1) [4]. Вузький діапазон значень запобігає перевантаженню функції активації. Однак слід зазначити, що градієнт дотичної функції більший, ніж сигмоїди. Рішення щодо вибору функції залежить від вимог до амплітуди градієнта. Гіперболічний тангенс також є дуже популярною і часто використовуваною функцією активації.

1.1.3 Моделі прогнозування на основі нейронних мереж

Нейронні мережі (НМ) – математичні моделі, зроблені за принципом організації функціонування біологічних нейронних мереж – мереж нервових клітин живого організму [13, 46-47]. Щоб вибудувати математичну модель процесів, що проходять всередині мозку, можна зазначити декілька припущень:

- кожен нейрон має передавальну функцію, яка визначає умову його збудження в залежності від сили прийнятих сигналів; окрім цього, передавальні функції не мають залежності від часу;
- при проходженні синапсу сигнал змінюється лінійно, це число називають «вагою» синапсу або вагою відповідного входу нейрона;
- діяльність нейронів синхронізована, також це відноситься до часу обробки прийнятих сигналів. Варто відмітити, що ваги синапсів мають змінюватися з часом – це принципова ознака, що дає перевагу цій моделі прогнозу над всіма іншими. Нейронні мережі можуть витягувати приховані закономірності з потоку даних. Проте, дані можуть бути неповні, протилежні та навіть спотворені.

Якщо між вхідними та вихідними даними існує певний зв’язок НМ здатна самостійно налаштуватися на неї із заданим степенем точності [4]. На НМ варто дивитись як на мережу «нейронів», організованих в шарах. Вхідні нейрони утворюють нижній шар, а прогнози (або виходи) створюють верхній шар. Можуть проявлятись і проміжні шари, які містять «приховані нейрони».

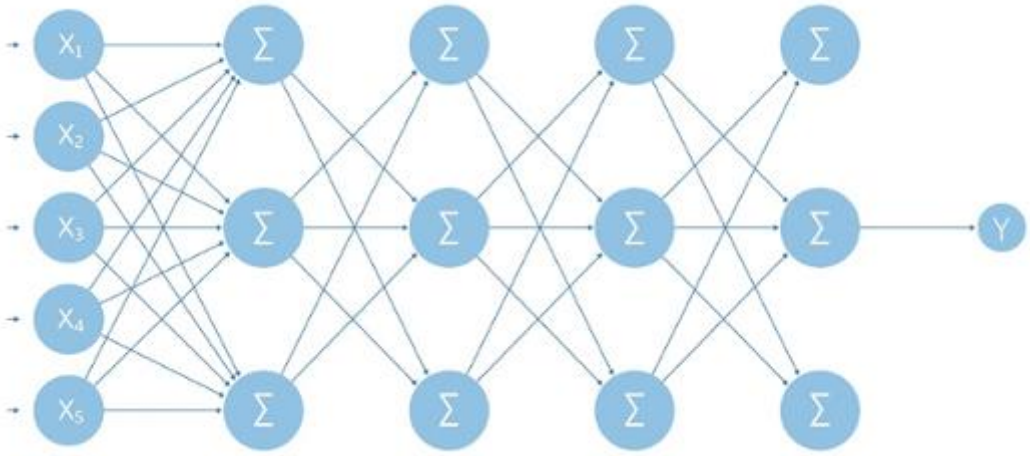


Рис. 1.7 – Приклад архітектури нейронної мережі

На старті функціонування нейронної мережі її ваги приймають спонтанні значення та оновлюються із застосуванням спостережених даних. Тому зазвичай, в прогнозах, отриманих за допомогою НМ є елемент випадковості. Роботу НМ перевіряють певну кількість разів із застосуванням різних вхідних даних. Процес усереднення ваг штучних нейронів з ціллю отримати потрібний результат на виході, називається навчанням НМ, а вхідні дані для навчання – навчальні параметри або навчальна вибірка. Навчальна вибірка – це вибірка, за допомогою якої зазвичай вибудовується модель залежностей. Отриманий результат має назву тестова вибірка. За його допомогою оцінюється якість моделей залежностей. Оцінку якості проводять для вибору найкращої моделі. Для ефективної перевірки моделі її перевіряють на даних, які не брали участь у навчанні. Здебільшого формування методів навчання відбувається за принципом «ковзного вікна»: береться деякий відрізок ряду і з нього утворюється спостереження, які і будуть являти собою вхідний вектор. Значенням отриманого бажаного виходу в навчальному прикладі буде наступне за порядком спостереження. Потім «ковзне вікно» переміщається на один щабель в напрямку зросту часу, і процес утворення наступної пари навчальної вибірки повторюється як і раніше.

Навчання нейронних мереж ділиться на категорії: навчання з учителем і навчання без учителя [18, 37-45]. Для навчання з учителем треба розмітити дані, що представляє із себе пари значень. Перше значення кожної пари - вхідні дані, друге значення – еталон. Еталон - це ідеальні значення нейронної мережі, результат, до

якого вона прагне. Дані подаються на вхід нейронної мережі, потім результати виходу мережі звіряються з еталоном [16]. Для того, щоб знайти відхилення від необхідного результату, вводиться функція помилки, яка є мінімізованою цільовою функцією:

$$E(w) = \frac{1}{2}\sum_{jk}(y_{jk} - d_{jk})^2$$

де, y − реальне вихідне значення нейрона j вихідного шару при подачі на jk вхід значення k, а d_{jk} − потрібне вихідне значення цього нейрона. В процесі навчання нейромережі мінімізація помилки відбувається шляхом поновлення ваги кожного шару нейромережі. Оновлене значення розраховується за формулою:

$$\Delta w_{ij}^{q} = -n\frac{\sigma E}{\sigma w_{ij}},$$

де w_{ij}^{q} − це ваговий коефіцієнт, який з'єднує i-тий нейрон шару q з j-тим нейроном, n − коефіцієнт швидкості навчання, $0 < n < 1$.

Неконтрольоване навчання – це машинне навчанання з використанням наборів даних без визначеної структури. Неконтрольоване навчання зазвичай використовується для транзакційних даних [23]. У вас може бути великий набір даних про клієнтів і їх покупках, але ви, як людина, швидше за все, не зможете зрозуміти, які аналогічні атрибути можна витягти з профілів клієнтів і їх типів покупок. З урахуванням того, що ці дані введені в алгоритм навчання без нагляду, можна визначити, що жінки певного вікового діапазону, які купують мило без запаху, можуть бути вагітними, і, отже, маркетингова кампанія, пов'язана з вагітністю та дитячими продуктами, може бути націлена на цю аудиторію, щоб збільшити їх кількість покупок.

1.2 Основні поняття аналізу часових рядів

Часовий ряд – це послідовність даних у хронологічному порядку [8]. Часові ряди можуть містити в собі декілька значень, які змінюються з часом. Такі ряди називаються багатовимірними, якщо вони містять лише одну складову, то вони є одновимірними.

Часові ряди широко використовуються у багатьох сферах діяльності: фінанси, медицина, сейсмологія та інші. Часові ряди знаходять застосування всюди, де спостерігаються зміни певних процесів у часі.

Проводячи аналіз часового ряду можна зробити більш точний прогноз його поведінки. Аналіз часового ряду проводиться для розуміння його природи.

Основним завданням прогнозування часового ряду є підбір моделі на основі історичних даних для прогнозування майбутніх значень .

За своїми властивостями часові ряди можна розділити на декілька типів: детерміновані, недетерміновані, стаціонарні та нестаціонарні [2].

Детермінований часовий ряд – це ряд, який можна виразити аналітичним виразом. Ряд можна описати точно за весь період розкладанням в ряд Тейлора, якщо відомі всі його похідні в деякий момент часу. Ряд не містить в собі ймовірносних аспектів.

Недетермінований часовий ряд – це ряд, який містить у собі певний випадковий аспект , що унеможливлює опис даного ряду аналітичним виразом. Ряд може бути нетермінований через випадкову природу генерації значень, або певна частина значень недоступна. Такий ряд визначається розподілом ймовірностей та іншими статистичними характеристиками.

Стаціонарний часовий ряд – це ряд, статистичні значення якого є незалежними від часу. Прогнозування значень стаціонарного ряду зводиться до передбачення ймовірності появи значення, які уже були отримані в минулому.

Нестаціонарний часовий ряд – це ряд, статистичні властивості якого змінюються з часом.

Часові ряди можуть бути представлені у графічному та в табличному вигляді.

На відміну від статистичних вибірок часові ряди мають певні ознаки [24]:

- послідовні в часі значення є взаємопов’язаними, це можна відстежити на близько розташованих значеннях;
- зі збільшенням показників часового ряду точність його характеристик не буде збільшуватися, а може навпаки зменшитись;

- інформаційна цінність спостережень спадає в міру їх віддалення від поточного моменту часу.

При дослідженні часових рядів застосовують основні методи:

1. **Кореляційний аналіз**, який дає змогу виявляти істотні періодичні залежності та їх затримки всередині певного процесу (автокореляція) або між декількома процесами (кроскореляція).

2. **Спектральний аналіз**, що застосовують для визначення періодичних та квазіперіодичних компонент часового ряду.

3. Методи згладжування та фільтрації, призначені для перетворення часових рядів з метою видалення з них високочастотних та сезонних коливань.

4. **Методи авторегресії та ковзних середніх**, які використовують для опису і прогнозування процесів, що здійснюють випадкові коливан ня навколо певного середнього значення.

5. **Методи прогнозування**, що дають можливість на основі обраної моделі часового ряду оцінювати його найбільш імовірні значення в майбутньому.

В сучасній статистичній теорії існує багато різноманітних методів прогнозування економічної інформації. Модель часових рядів якраз враховує минулу поведінку даної змінної і використовує цю інформацію для прогнозування її майбутньої поведінки. Особливістю прогнозування часових рядів є те, що аналізуються лише дані спостережень без додаткової інформації, без аналізу впливу зовнішніх сил.

Для побудови часового ряду всі його елементи мають характеризувати певне явище за різні відрізки часу або фіксувати стан ознаки через одинакові інтервали. Для визначення стійкої тенденції треба провести велику кількість вимірювань, щоб в подальшому ми могли спираючись на неї прогнозувати майбутні значення, але водночас використання досить великого набору даних може призвести до нечутливості тренду до змін.

Для детальнішого аналізу та прогнозу часових рядів, потрібно визначити основні компоненти. Давайте спробуємо визначити компоненти, які зазвичай виділяють в часових рядах [29]:

- Циклічний – динаміка, яка містить фазу зросту та спаду, часові рамки, які займають достатньо великий проміжок часу.
- Тренд – це динаміка, яка визначає загальний розвиток часового ряду.
- Сезонні компоненти – це постійні коливання рівнів ряду в певний час.
- Аномальні явища – це непередбачувані стрибки, які призводять до короткочасних відхилень ряду від загальної тенденції розвитку.
- Календарні ефекти – це стрибки часового ряду, пов’язані з деякими визначеними подіями у календарі (для прикладу, свята або вихідні).
- Структурні компоненти – непередбачувані стрибки, які ведуть до відхилення ряду від основного напрямку розвитку, які впливають на його поведінку.
- Випадкові компоненти – хаотичні комплексні рухи в міру високої частоти, які пов’язані з впливом значної кількості невідомих факторів. Деякі з часових рядів мають той чи інший компонент в чистому вигляді, але на практиці ряди такого типу з’являються досить нечасто.

Часовий ряд в більшості поєднанє декілька компонентів. Ряди, що являють певний компонент в чистому вигляді зустрічаються досить рідко.

Аналіз часового ряду бере початок з побудови і вивчення його графіку. Якщо нефункціональність часового ряду видно одразу, тоді потрібно визначити його нестаціонарну частину [30]. Процес виокремлення тренду та інших складових ряду, які ведуть до порушення стаціонарності, може проходити в кілька етапів. На кожному з них визначається ряд залишків, виведений в результаті вирахування з вихідного ряду підібраної моделі тренду або результат різницевих перетворень ряду. Окрім графіків, ознаками нефункціональності часового ряду можуть бути автокореляційна функція, яка веде не до нуля (вийняток − дуже великі значення лагів) і наявність чітко визначених піків на низьких частотах у періодограмі. За допомогою автокореляційної функції досліджують і внутрішні зв'язки між частинами часових рядів [3].

У деяких дослідженнях найпростіші числові ознаки описової статистики (середнє, медіана, дисперсія, стандартне відхилення, коефіцієнти асиметрії й ексцесу) дають досить інформативне уявлення про вибірку. Графічні методи зображення й аналізу вибірок при цьому відіграють тільки допоміжну роль, дозволяючи чіткіше зрозуміти локалізацію і концентрацію даних, закон їхнього розподілу.

Роль графічних методів при аналізі часових рядів зовсім не така. Табличне представлення часового ряду й описових статистик не дозволяє визначити характер процесу, у той час як за графіком часового ряду можна вивести достатньо висновків. Потім вони можуть бути перевірені й уточнені за допомогою підрахунків.

1.3 Огляд бібліотек компонентів нейромережного прогнозування

Бібліотека в програмуванні – збірка програм або об'єктів, які застосовуються для розробки програмного забезпечення [2]. Загальним призначенням представлених нижче бібліотек є інтеграція нейромережних технологій в особисті інформаційні та програмні системи, для того щоб розширити їхні аналітичні можливості. Реалізація НМ у вигляді складових і наявність відкритого коду дає змогу вбудовувати подібні бібліотеки в інші програми без жодних проблем. Об'єктно орієнтоване виконання дає їх особливу гнучкість для оптимізації під різні задачі.

Найбільш поширені нейромережні бібліотеки для роботи з часовими рядами:

1. ANN – безкоштовна бібліотека з відкритим вихідним кодом для утворення нейронної мережі на мові PHP. Вихідний код заснований на роботі Едді Янга. ANN є вільно розповсюдженою бібліотекою за умови зберігання інформації про авторство в вихідному коді.

2. FANN – це безкоштовна бібліотека з відкритим вихідним кодом для створення нейронних мереж. Сьогодні бібліотека FANN доступна практично для всіх мов програмування і середовищ розробки. Вона досить проста у використанні, універсальна і добре документована.

3. Encog – безкоштовна бібліотека для створення нейронних мереж на мовах Java, C# і C++. Encog підтримує різноманітні алгоритми навчання. Його основна сила проявляється в нейромережних алгоритмах. Encog має класи для створення мереж з

різноманітними архітектурами, а також допоміжні класи для нормалізації та обробки НМ.

4. NeuralBase – це бібліотека з відкритим вихідним кодом, реалізована мережа Хопфілда та багатошарова нейронна мережа, що навчається за алгоритмом зворотного розповсюдження. Бібліотека є інтелектуальною власністю компанії BaseGroup Labs – професіонального постачальника програмних продуктів і рішень в сфері аналізу даних.

5. Keras – відкрита нейромережева бібліотека, яка написана мовою Python. Вона здатна працювати поверх Deeplearning4j, TensorFlow та Theano. Спроектовану для уможливлення швидких експериментів з мережами глибинного навчання, її покладено на тому, щоб вона була мінімальною, модульною та розширюваною. Її було створено як частину дослідницьких зусиль проекту ONEIROS (англ. Open-ended Neuro-Electronic Intelligent Robot Operating System), а її основним автором є Франсуа Шоллє (фр. François Chollet), інженер Google.

6. Darts – це бібліотека Python для зручного маніпулювання та прогнозування часових рядів. Він містить безліч моделей, від класичних, таких як ARIMA, до глибоких нейронних мереж. Усі моделі можна використовувати однаково, використовуючи fit()і predict () функції, подібні до scikit-learn. Бібліотека також полегшує тестування моделей, об'єднання прогнозів декількох моделей та облік зовнішніх даних. Darts підтримує як одновимірні, так і багатовимірні часові ряди та моделі. Моделі на основі машинного навчання можна навчати на потенційно великих наборах даних, що містять кілька часових рядів, а деякі моделі пропонують багату підтримку ймовірнісного прогнозування.

Різноманітність даних бібліотек дає змогу застосовувати їх у розв’язку будь-якого завдання незалежно від обраної мови програмування та архітектури НМ. Відкритий вихідний код дає змогу для налаштування розробленої програмної системи, що дуже важливо в завданнях про прийняття рішень. В подальшому, Keras і слугуватиме для реалізації нейромережевих моделей прогнозування.

Висновки до розділу 1

Останнім часом точність штучних нейронних мереж дуже швидко зростає. Існує множина прикладів, коли нейронні мереж з’являються в якійсь області та повністю замінюють класичні алгоритми вирішення задач. Для того щоб використання штучний нейронних мереж було виправдано, необхідно щоб задача мала наступні ознаки:

- достатньо велика множина прикладів, не відомі принципи рішення задачі (відсутній алгоритм);
- рішення задачі передбачає розгляд великого об’єму вхідної інформації;
- дані неповні, надлишкові, частково суперечливі.

Виходить, що використання штучний нейронних мереж добре підходять для рішення задач класифікації, розпізнавання образів, проведення оптимізації та прогнозування.

РОЗДІЛ 2. ПРОГРАМНА РЕАЛІЗАЦІЯ НЕЙРОМЕРЕЖНИХ МОДЕЛЕЙ

2.1 Представлення вхідних даних

Модель рекурентних нейронних мереж постає дуже корисною при прогнозуванні масиву часових рядів, оскільки даний підхід дозволяє враховувати пам'ять прогнозованої системи за довготривалий період часу. У нашому випадку модель збиратиме інформацію зі своїх власних атрибутів (в даному випадку наших акцій) при прогнозуванні конкретно обраного індексу. Ми припускаємо, що довготривала пам'ять інших рядів впливатиме на подальшу динаміку прогнозованого ряду цін акцій.

Послуговуючись інструментарієм мови програмування Python та API сайту Yahoo Finance (https://finance.yahoo.com/), ми вилучаємо значення таких змінних часових рядів:

- Ціна XOM: ціна акцій компанії Exxon.
- S&P 500: індекс фондового ринку S&P 500.
- Ціна WTI: ціна бареля сирої нафти West Texas Intermediate (США).
- Ціна на нафту марки Brent: ціна бареля сирої нафти марки Brent (ЄС).

Код для вилучення показників за їх максимальний період часу та фільтрації зайвих параметрів представлений на рис. 1.

```
xom = yf.Ticker("XOM")
sp500 = yf.Ticker("^GSPC")
bre = yf.Ticker("BZ=F")
wt = yf.Ticker("CL=F")
```

(а)

```
histxom = xom.history(period="max")
histsp = sp500.history(period="max")
histbre = bre.history(period="max")
histwt = wt.history(period="max")
exxon = histxom.drop(columns = ['Open', 'High', 'Low', 'Volume', 'Dividends', 'Stock Splits'])
exxon = exxon.rename(columns={'Close': 'XOM Price'})
market = histsp.drop(columns = ['Open', 'High', 'Low', 'Volume', 'Dividends', 'Stock Splits'])
market = market.rename(columns={'Close': 'S&P 500'})
brent = histbre.drop(columns = ['Open', 'High', 'Low', 'Volume', 'Dividends', 'Stock Splits'])
brent = brent.rename(columns={'Close': 'Brent Price'})
wti = histwt.drop(columns = ['Open', 'High', 'Low', 'Volume', 'Dividends', 'Stock Splits'])
wti = wti.rename(columns={'Close': 'WTI Price'})
fin = pd.concat([exxon,market,brent,wti], axis = 1)
```

(б)

Рис. 2.1. Вилучення та фільтрація значень індексів XOM, S&P 500, Brent та WTI (а) та їх фільтрація (б).

Окрім цього, ми вилучимо додаткові показники, що можуть визначати динаміку нафтових індексів. Для цього скористаємось публічним API ресурсу «Управління енергетичної інформації США». Для вилучення значень потребується згенерувати публічний API-ключ та задати кодові значення нових змінних. У даному випадку вилучатимемо наступні змінні:

- Видобуток (змінна Production): видобуток сирої нафти в США.
- Споживання (змінна Consumption): споживання нафтопродуктів у США.
- Запаси (Inventory (Crude)): запаси сирої нафти в США.
- Витрати на нафтопереробні заводи (Input to Refineries): споживання сирої нафти на нафтопереробних заводах США.
- Запаси (Inventory (Gas)): запаси бензину в США.

  Код для об’єднання та фільтрації представлений на рис. 3.2.

```
api_key = 'QVz5PaUgJZM1zgMo3NoOpqmmsNsfcQGKvLSq1ZdV'
```

(а)

```
titles = ['Production',
          'Consumption',
          'Inventory (Crude)',
          'Input to Refineries',
          'Inventory (Gas)']

codes = ['PET.MCRFPUS2.M',
         'PET.WRPUPUS2.W',
         'PET.WTTSTUS1.W',
         'PET.WCRRIUS2.W',
         'PET.WGTSTUS1.W']

eia_dat = []
```

(б)

Рис. 2.2. Визначення публічного API-ключа (а) та оголошення кодових значень нафтових змінних (б).

Парсинг здійснюється згідно наступного фрагменту, де задається посилання (змінна url) на відповідний ресурс із нашим API-ключем та йде перевірка, чи є вилучення даних успішним. На рис. 3.3 представлено фрагмент ключового фрагменту.

```
for i in range(len(codes)):
    url = 'https://api.eia.gov/series/?api_key=' + api_key + '&series_id=' + codes[i]
    r = requests.get(url)
    json_data = r.json()

    if r.status_code == 200:
        print('Success!')
    else:
        print('Error')

    df = pd.DataFrame(json_data.get('series')[0].get('data'),
                      columns = ['Date', titles[i]])
    df.set_index('Date', drop=True, inplace=True)
    eia_dat.append(df)
```

Рис. 2.3. Парсинг значень нафтових змінних із сайту Управління енергетичної інформації США.

Наші подальші дії полягають у згладжуванні даних для видалення зайвого шуму, поєднання за найменшим із досліджуваних рядів та заповнення негативного значення ціни нафти WTI. Код даних дій представлений на рис. 3.4.

```
data['XOM Price'] = data['XOM Price'].rolling(5).mean().round(decimals = 2)
data['S&P 500'] = data['S&P 500'].rolling(5).mean().round(decimals = 2)
data['WTI Price'] = data['WTI Price'].rolling(5).mean().round(decimals = 2)
data['Brent Price'] = data['Brent Price'].rolling(5).mean().round(decimals = 2)
data['Production'] = data['Production'].rolling(60).mean().round(decimals = 2)
data['Consumption'] = data['Consumption'].rolling(15).mean().round(decimals = 2)
data['Inventory (Crude)'] = data['Inventory (Crude)'].rolling(15).mean().round(decimals = 2)
data['Input to Refineries'] = data['Input to Refineries'].rolling(15).mean().round(decimals = 2)
data['Inventory (Gas)'] = data['Inventory (Gas)'].rolling(15).mean().round(decimals = 2)
```

(а)

```
startDate = data['Brent Price'].first_valid_index()
mask = (data.index >= startDate) & (data.index <= endDate)
data = data.loc[mask]
```

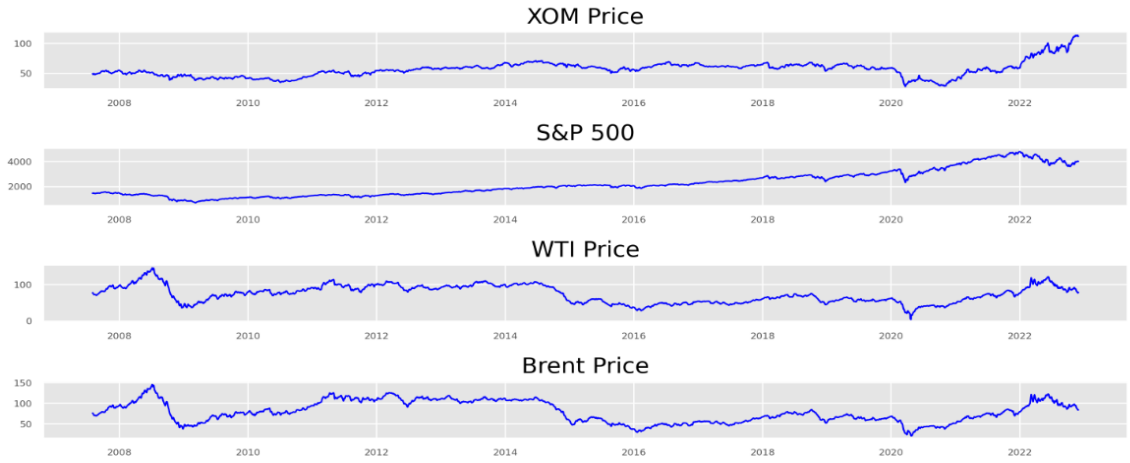
(б)

```
data['WTI Price'] = data['WTI Price'].mask(data['WTI Price'].lt(0)).ffill().fillna(0).convert_dtypes()
```

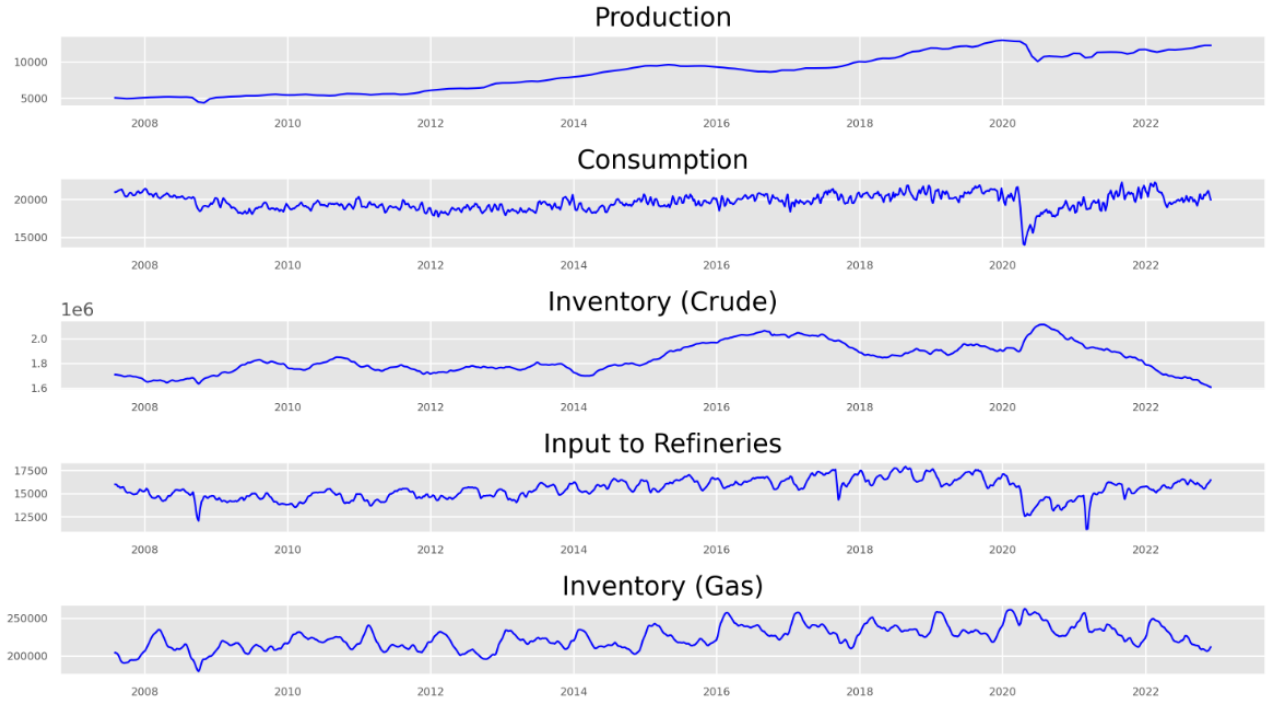
(в)

Рис. 2.4. Фільтрація часових рядів за допомогою процедури ковзного середнього (а) співставлення всіх за першим ненульовим значенням (б) та замінна від'ємної ціни нульовим значенням (в).

Динаміку значень нафтових та фондових індексів представлено на рис. 3.5.



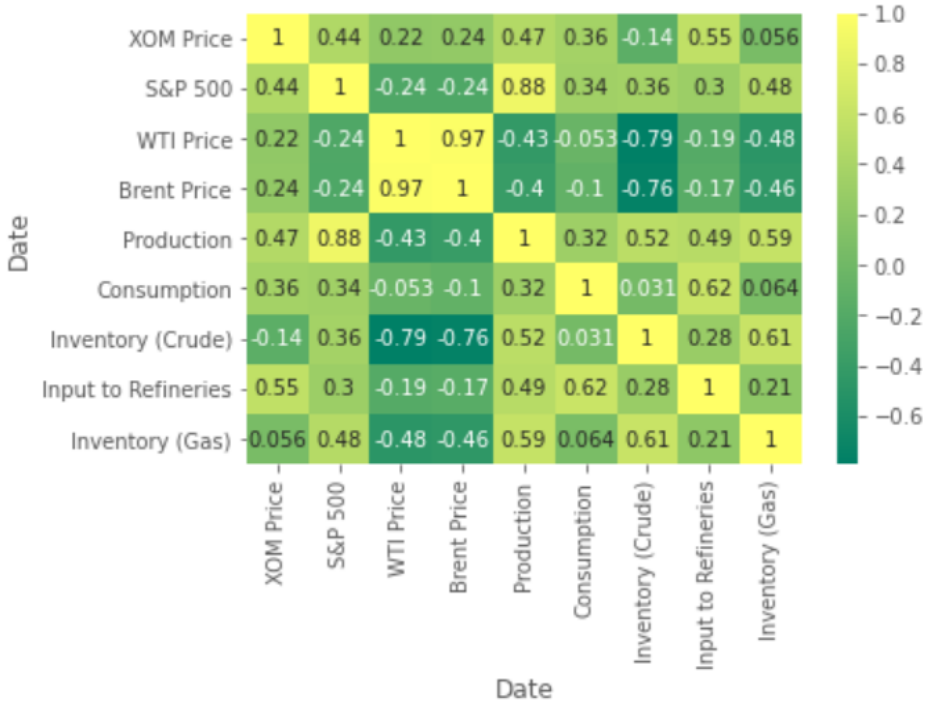
(а)



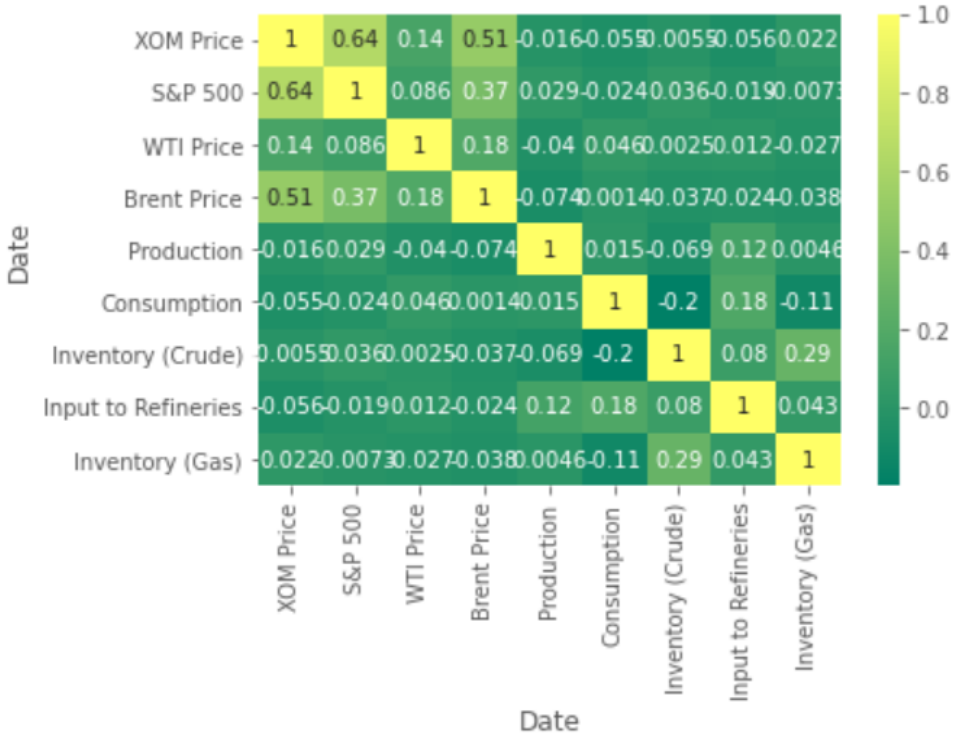
(б)

Рис. 2.5. Динаміка значень індексів XOM, S&P 500, WTI та Brent (а). Динаміка показників видобутку сирої нафти в США (а), споживання нафтопродуктів (б), запасів сирої нафти (в), витрат на нафтопереробні заводи (г) та запасів бензину (д).

Розглянемо матрицю корреляцій між досліджуваними рядами.



(а)



(б)

Рис. 2.6. Кореляційна матриця для вхідних даних (а) та прибутковостей (б).

Із данних матриці для вихідних значень (рис. 3.6) можемо спостерігати середню та високу лінійну корреляцію між наступними показниками:

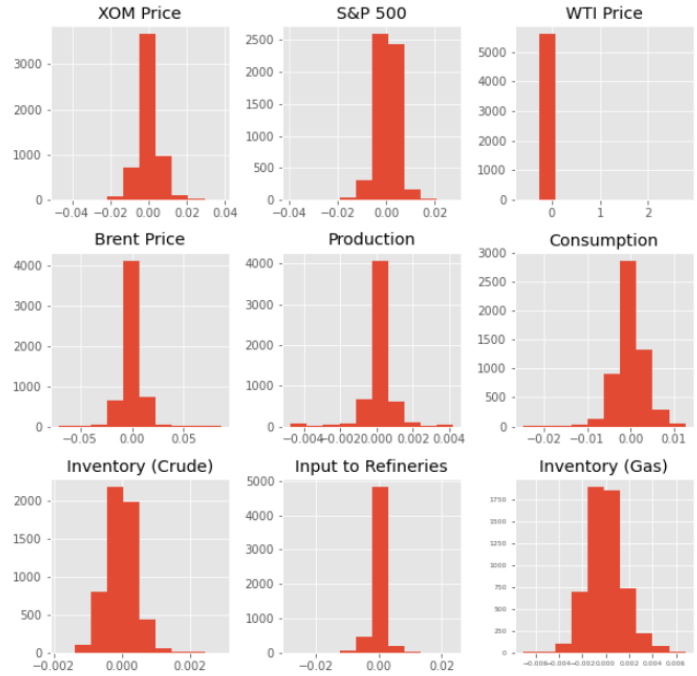
- ціна індексу S&P 500 та видобуток сирої нафти в США (0.88);
- Ціна WTI та ціна Brent (0.97);
- Видобуток сирої нафти в США та запаси бензину в США (0.59);
- Споживання сирої нафти на нафтопереробних заводах США та споживання нафтопродуктів у США (0.62);
- Запаси сирої нафти в США та запаси бензину в США (0.61);
- Видобуток сирої нафти в США та запаси сирої нафти в США (0.52);
- Ціна акцій компанії Exxon та витрати на нафтопереробні заводи (0.55).

Для прибутковостей залишається залежність між

- ціна Brent та ціна акцій компанії Exxon (0.51);
- ціна індексу S&P 500 та ціна акцій компанії Exxon (0.64).

Якщо поглянемо на більшість наших рядів та їх частотні розподіли (рис. 3.7), то побачимо, що білшість наших даних є нестаціонарними. А їх прибутковості характеризуються гостровершинним ексцесом та важкими хвостами.

(a)



(б)

Рис. 2.7. Частотні розподіли вихідних значень досліджуваних рядів (а) та їх прибутковостей (б).

Виходячи з парних графіків регресії залежність між величинами стає більш наглядніщою.

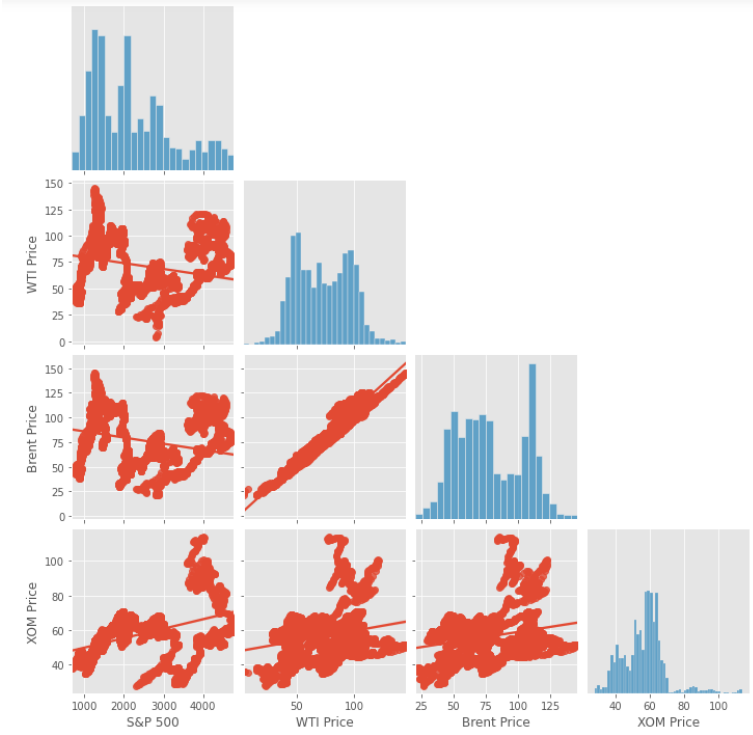


Рис. 2.8. Парна діаграма ліній регресії та взаємних розподілів індексів XOM, Brent, WTI та S&P 500.

2.2 Вибір мови програмування та середовища розробки

Для реалізації програмного додатку для аналізу часових рядів було обрано Python – мова програмування високого рівня загального призначення, орієнтована на підвищення швидкодії розробника та читаності коду[27].

Синтаксис ядра Python мінімалістичний. Водночас стандартна бібліотека включає великий об’єм корисних функцій.

Python підтримує декілька парадигм програмування, в тому числі структурне, об’єктно-орієнтоване, функціональне, імперативне і аспектно-орієнтоване.

Основі архітектурні риси – динамічна типізація, автоматичне управління пам’яттю, повна інтроспекція, механізм обробки виключень, підтримка багато потокових обчислень і зручні високо рівневі структури даних. Код в Python організовується в функції та класи, які можуть об’єднуватися в модулі (вони в свою чергу можуть об’єднуватися в пакети).

Основними вимогами до вибору середовища розробки є підтримка мови програмування Python і можливість інтерактивного вводу та виводу даних, таким середовищем являється IPython – потужний інструмент для роботи з мовою Python. Базові компоненти IPython – це інтерактивна оболонка з широким набором можливостей і ядро для Jupyter. Jupyter notebook являє собою графічну веб оболонку для IPython, яка розширює ідею консольного підходу до інтерактивних обчислень.

Основі особливості даної платформи – це комплексна інтроспекція об’єктів, зберігання історії вводу протягом усіх сеансів, хешування вихідних результатів, розширювана система «магічних» команд, додатковий командний синтаксис, підсвічування коду, доступ до системної оболонки, стикування з pdb відладчиком і Python профайлером.

IPython дозволяє підключати багатьох клієнтів до одного обчислювального ядра і завдяки своїй архітектурі, може працювати в паралельному кластері.

В Jupyter notebook можна розробляти, документувати та виконувати додатки на мові Python, він складається з двох компонентів: веб-додаток і ноутбуки – файли, в яких можна працювати з вихідним кодом програми, запускати його, вводити та виводити дані та ін.

Веб додаток дозволяє:

- редагувати Python код в браузері, з підсвічуванням синтаксису, автовідступами та автодоповненням;
- запускати код в браузері;
- відображати результати обчислень з медіа представленням (схеми, графіки);
- працювати з мовою розмітки Markdown і LaTeX.

Ноутбуки – це файли в яких зберігається вихідний код і вихідні дані, отримані в рамках сесії. Фактично він є записом вашої роботи, але при цьому дозволяє заново виконати код, присутній в ньому. Ноутбуки можна експортувати в формати PDF, HTML.

2.3 Побудова нейромережної моделі RNN, LSTM, GRU та її програмна реалізація

Рекурентні нейронні мережі − вид нейронних мереж, де зв'язки між складовими створюють певну послідовність. Завдяки цьому утворюється можливість переробляти частини подій у часі або просторові ланцюги [6, 48-56]. На відміну від перцептронів з великою кількістю шарів, рекурентні мережі використовують свою внутрішню пам'ять для розробки послідовностей довільної довжини. Тому мережі RNN застосовані в задачах, де щось одне розбите на частини, для прикладу, виявлення тексту від руки або розпізнавання мови. Було запропоновано велику кількість різносторонніх архітектурних рішень для рекурентних мереж від найпростіших до найскладніших. Останнім часом найбільшого розголосу здобули мережі з довготривалою і короткочасною пам'яттю (LSTM) і керований рекурентний блок (GRU).

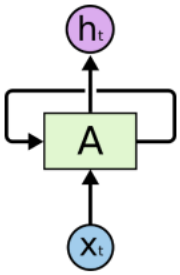


Рис. 2.9. Схема логічного елементу RNN

Сегмент нейронної мережі A отримує дані X на вхід і подає на вихід деяке значення H. Циклічний зв'язок дає змогу передавати інформацію від даного кроку мережі до наступного. Існує багато різновидів, рішень і конструктивних елементів рекурентних НМ. Складнощі застосування рекурентної мережі полягає в тому, що якщо враховувати кожен крок часу, то стає потрібним для кожного кроку часу утворювати свій шар нейронів, що спонукає серйозні обчислювальні складнощі. Окрім цього, багатошарові реалізації є обчислювально нестійкими, так як в них зникають або зашкалюють ваги. Якщо, наприклад, обмежити розрахунок тимчасовим вікном, то виведені моделі не будуть відображати довгострокових трендів. Різні методи, як правило, пробують покращити модель історичної пам'яті і механізм запам'ятовування і забування. Рекурентні нейронні мережі не сильно відрізняються від звичайних нейронних мереж. Їх можна уявити собі, як велика кількість копій тої самої мережі, причому, кожна копія передає повідомлення наступної копії. Що може вийти, якщо ми розглянемо даний цикл:

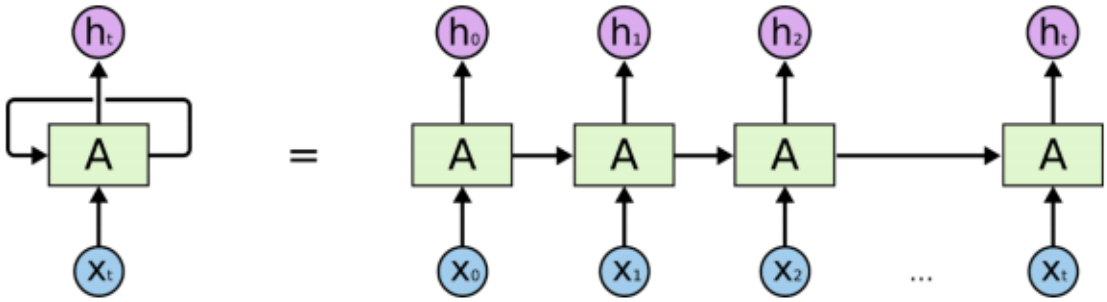


Рис. 2.10. Рекурентна нейронна мережа в розгорнутому вигляді

Дана "ланцюгова" сутність показує, що рекурентні нейронні мережі за своєю природою дуже пов'язані з послідовностями.

Послідовна модель, як правило, створена для переробку вхідної послідовності однієї предметної області у вихідну послідовність [18].

Архітектури мереж представленні як у звичайному так і з використанням регуляризаторів. Зазвичай точність на тренувальній вибірці набагато менша ніж на навчальній. Відбувається це, оскільки модель могла перенавчитися на тренувальній вибірці, що робить її дизадаптованою для нових даних. В області Глибокого навчання

використовується техніка під назвою *dropout* для запобігання перенавчання. *Dropout* вимикає нейрони з ймовірністю p та залишає увімкненими з ймовірністю $q=1-p$. Таким чином, модель дивиться на дещо видозмінену структуру самої себе щоб оптимізуватися.

Моделі RNN, LSTM та GRU представляються наступним чином:

- кількість вхідних значень (днів), які передаються в модуль прогнозування під час прогнозування – 12;
- кількість вихідних значень мереж – 10;
- кількість нейронів для кожного прихованого шару RNN – 20;
- кількість рекурентних шарів – 3;
- значення Dropout, що виступає в якості регуляризатора – 0.1;
- довжина як вхідних (цільових або коваріаційних) змінних, так і вихідних (цільових) часових рядів, що використовуються під час навчання – 22;
- цільова функція втрат – середньоквадратичне відхилення;
- для градієнтного спуску використовувать Adam оптимізатор;
- коефіцієнт швидкості навчання - 10^{-3};
- кількість епох (ітерацій градієнтного спуску) – 100.

Представимо результати прогнозів. У якості основної бібліотеки мови програмування Python для глибинного навчання буде *Darts*.

2.4 Розробка та реалізація нейромережної моделі LSTM та GRU

Мережі довго-короткострокової пам'яті (Long Short Term Memory) − зазвичай просто називають "LSTM" − особливий вид РНС, здатних до навчання довгостроковим залежностям [5]. Вони працюють неймовірно добре на великому різноманітності проблем і в даний момент широко застосовуються.

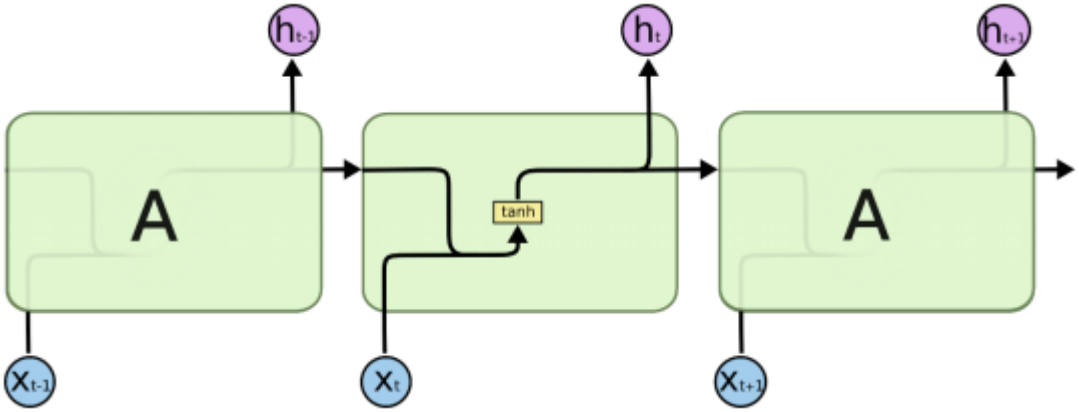


Рис. 2.11. Схема комірок LSTM

LSTM спеціально спроектовані таким чином, щоб уникнути проблеми довгострокових залежностей. Запам'ятовувати інформацію на тривалий період часу - це практично їх поведінку за замовчуванням, а не щось таке, що вони тільки намагаються зробити. У LSTM мережах вдалося обійти проблему зникнення або зашкалювання градієнтів в процесі навчання методом зворотного поширення помилки. Мережа LSTM зазвичай управляється за допомогою рекурентних вентилів, які називаються вентилі (gates) «забування» [21]. Помилки поширюються назад по часу через потенційно необмежену кількість віртуальних шарів. Таким чином відбувається навчання в LSTM, при цьому зберігаючи пам'ять про тисячі і навіть мільйони тимчасових інтервалів в минулому. топології мереж типу LSTM можуть розроблятися відповідно до специфіки завдання. В мережі LSTM навіть великі затримки між значущими подіями можуть враховуватися, і тим самим високочастотні і низькочастотні компоненти можуть змішуватися.

Шар керованих зворотних блоків (Gated Recurrent Unit, GRU) заснований на тому ж принципі, що і шар LSTM, однак рекурентні блоки представляють собою більш прості структури в порівнянні з LSTM і, відповідно, менш затратні в обчислювальному сенсі [1]. У традиційних рекурентних нейронних мережах для реалізації зворотного зв'язку використовується комбінація прихованого стану на попередньому кроці і поточних вхідних даних в шарі з нелінійної функцією активації, наприклад такий, як гіперболічний тангенс.

Комірка GRU подібна до комірки LSTM, але з декількома важливими відмінностями. По-перше, немає прихованого стану. Стан комірки приймає функціональність прихованого стану з конструкції комірки LSTM. Далі процеси визначення того, про що забуваються стани клітин і для якої частини стану комірки записані, об'єднуються в єдині вентилі. Записується лише частина стану клітинки, яка була стерта. Нарешті, весь стан комірки подається як вихідний результат. Це відрізняється від комірки LSTM, яка вибирає, що читати із стану комірки для отримання вихідних даних. Всі ці зміни разом забезпечують простіший дизайн з меншими параметрами, ніж LSTM. Однак менша кількість параметрів може спричинити зменшення вираженості.

Висновки до розділу 2

В результаті даної роботи були розглянуті варіанти побудови архітектури нейронних мереж для прогнозування майбутніх значень ціни індексів. На основі аналізу існуючих програмних продуктів, було обрано мову програмування Python. За допомогою зазначеної мови та нейромережевої бібліотеки Darts, що в свою чергу надає високорівневий, більш інтуїтивний набір абстракцій, було реалізовано найбільш підходящі мережі – RNN, LSTM та GRU. Для роботи із багатовимірними послідовностями, математичними алгоритмами, обробки даних та їх візуалізації були обрані такі програмні засоби як *Numpy, Pandas, Sci-Py*, та *Matplotlib*.

Зазначені програмні продукти дозволили нам:

- виокремити описову статистику досліджуваних індексів, розподіли прибутковостей;
- розбити досліджуваний ряд на навчальну, затверджувальну та тестову вибірку;
- спроектувати архітектуру мережі: визначити кількість шарів моделей, кількість нейронів у кожному шарі, кількість регуляризаторів та ймовірність вимкнення кожного нейрону, функції оцінки якості навчання (метрику), оптимізатор тощо.

Варто зазначити, що зазначена мова програмування Python та бібліотеки обранні нами не є єдинним еталоном для роботи із сьогоднішніми задачами штучного інтелекту, так само як і використовувані нами у подальшому нейромережеві архітектури. Однак, згідно поставлених нами задач, ці програмні засоби роблять наше дослідження більш інтуїтивно зрзумілим, а використовувані моделі швидкореалізуємими.

РОЗДІЛ 3. РЕЗУЛЬТАТИ НЕЙРОМЕРЕЖНОГО ПРОЗНОЗУВАННЯ ЗНАЧЕНЬ ФІНАНСОВИХ РИНКІВ

3.1 Прогнозування значень фінансових активів із використанням нейромережевих підходів

Спершу імпортуємо наступні модулі бібліотеки Darts, що будуть корисні нам для подальшого прогнозування.

```
from darts.models import RNNModel
from darts import TimeSeries
from darts.dataprocessing.transformers import Scaler
from darts.metrics import mape, smape
```

Рис. 3.1. Підключення основних модулів для побудови рекурентних мереж, роботи з часовими рядами, нормалізації значень та вимірювання точності прогнозів.

Скористаємось класом *RNNModel* для ініціалізації нашої моделі.

```
my_model = RNNModel(
    model="RNN",
    hidden_dim=20,
    n_rnn_layers=3,
    dropout=0.1,
    batch_size=32,
    n_epochs=100,
    optimizer_kwargs={"lr": 1e-3},
    model_name="Air_RNN",
    log_tensorboard=True,
    random_state=42,
    training_length=22,
    input_chunk_length=12,
    output_chunk_length=10,
    force_reset=True,
    save_checkpoints=True,
)
```

Рис. 3.2. Ініціалізація екземпляру класу RNNModel для навчання та прогнозування рекурентних мереж.

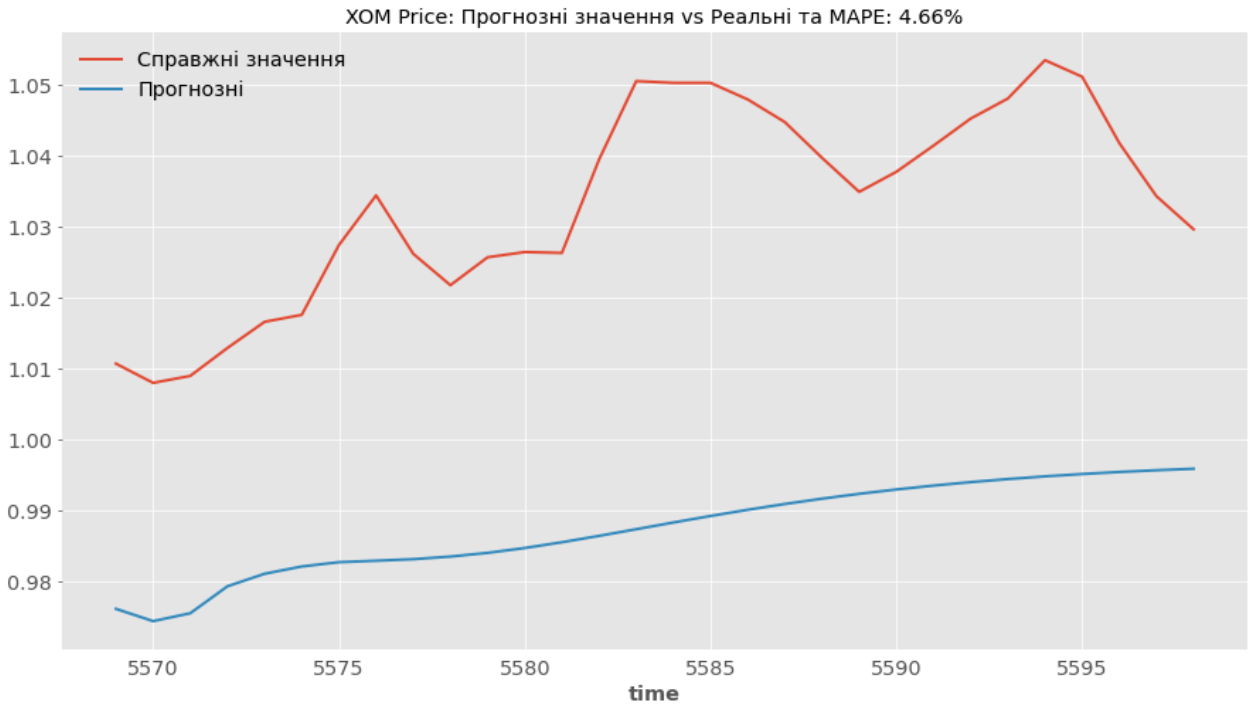
Даний екзмпляр тепер матиме метод fit, котрий прийматиме навчальну вибірку *train_transformed* та тестову вибірку *val_transformed*. У ході навчання нам відображатимуться доступні апартні можливості для прискорення навчання, цільова функція, тип рекурентної мережі, кількість налаштовуваних параметрів тощо.

```
my_model.fit(
    train_transformed,
    val_series=val_transformed,
    verbose=True,
)
```

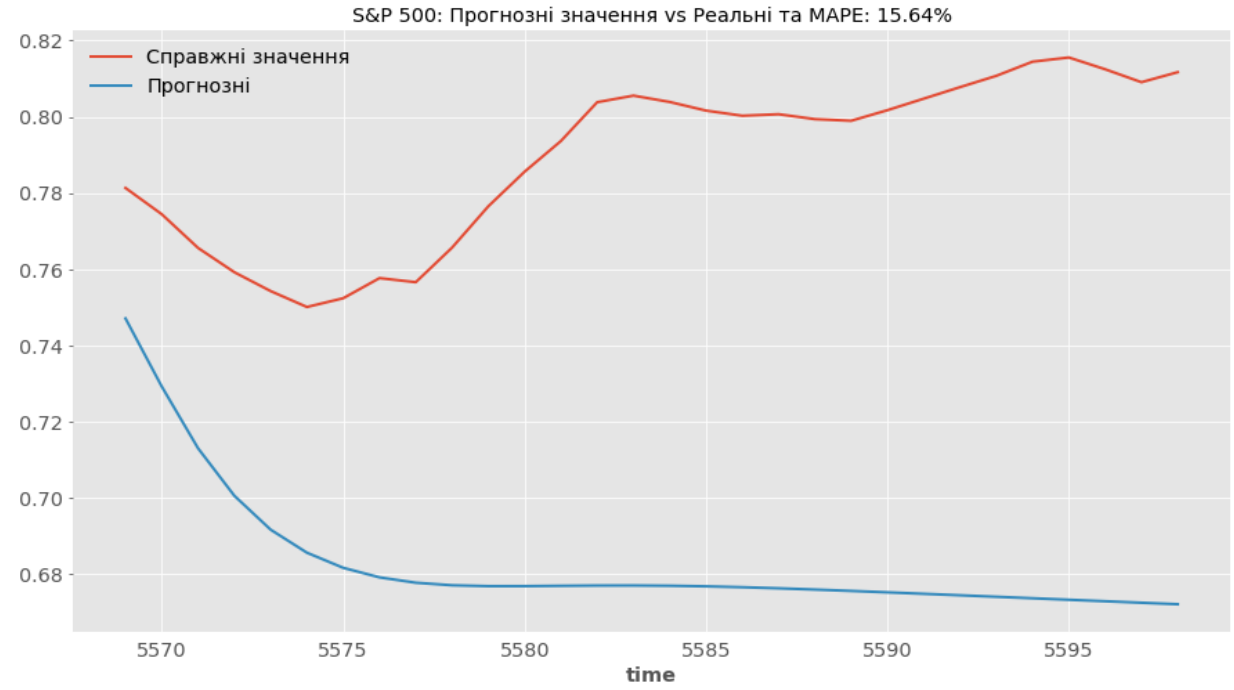
```
GPU available: False, used: False
TPU available: False, using: 0 TPU cores
IPU available: False, using: 0 IPUs
HPU available: False, using: 0 HPUs

  | Name          | Type              | Params
-------------------------------------------------
0 | criterion     | MSELoss           | 0
1 | train_metrics | MetricCollection  | 0
2 | val_metrics   | MetricCollection  | 0
3 | rnn           | GRU               | 6.9 K
4 | V             | Linear            | 189
-------------------------------------------------
7.1 K     Trainable params
0         Non-trainable params
7.1 K     Total params
0.028     Total estimated model params size (MB)
```

Рис. 3.3. Синтаксис бібліотеки Darts для навчання мережі.

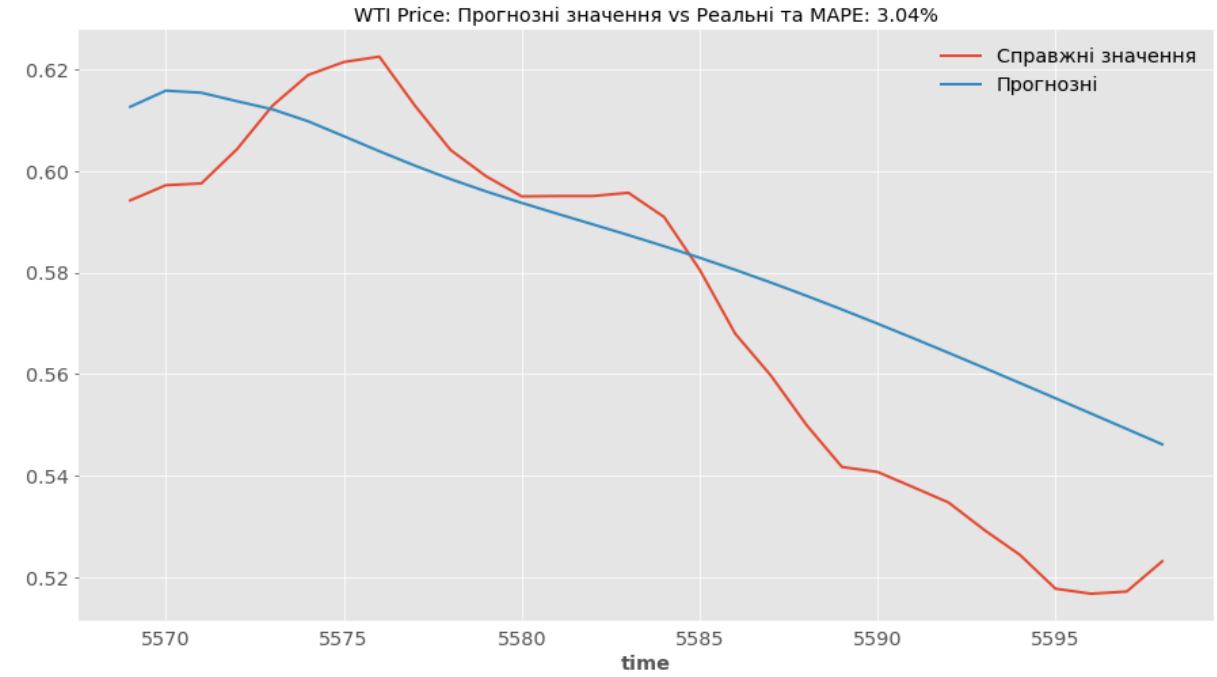
Визначивши всі необхідні параметри, можна приступити до прогнозування. На рис. 3.13 представлено результати прогнозу класичного блоку RNN для індексів XOM, S&P 500, WTI та Brent.


XOM Price: Прогнозні значення vs Реальні та MAPE: 4.66%
Справжні значення
Прогнозні
1.05
1.04
1.03
1.02
1.01
1.00
0.99
0.98
5570
5575
5580
5585
5590
5595
time

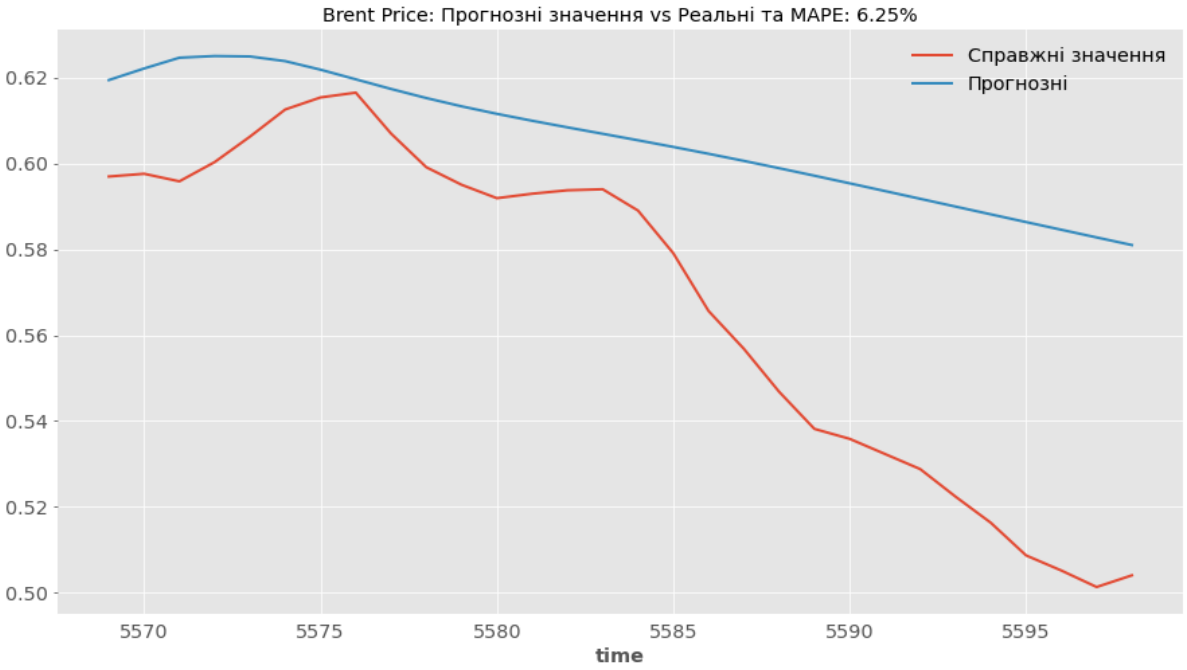
(а)


S&P 500: Прогнозні значення vs Реальні та MAPE: 15.64%
Справжні значення
Прогнозні
0.82
0.80
0.78
0.76
0.74
0.72
0.70
0.68
5570
5575
5580
5585
5590
5595
time

(б)



(в)

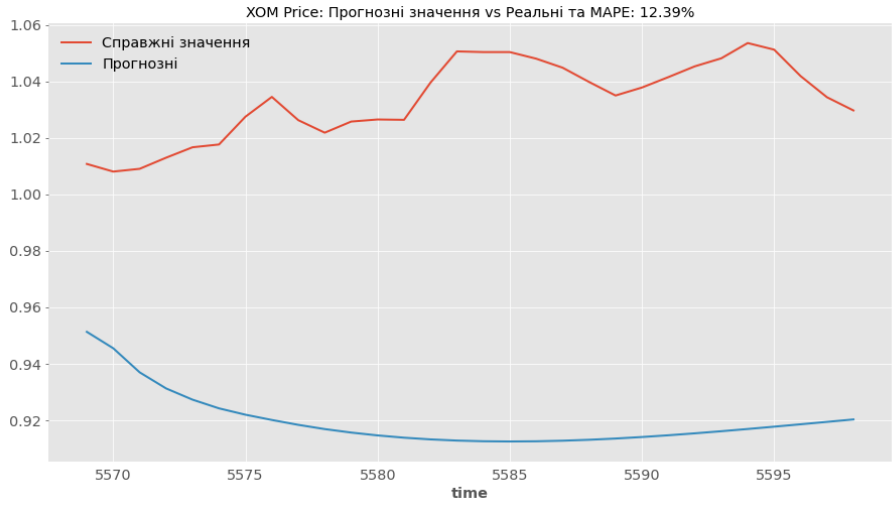


(г)

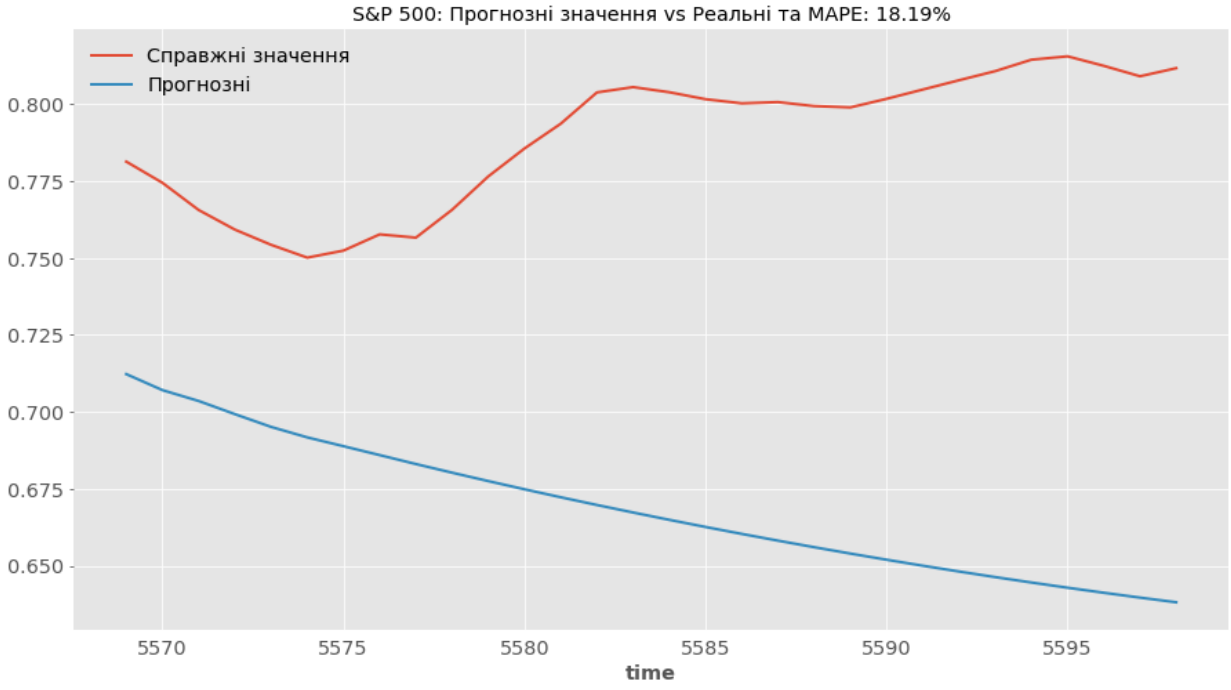
Рис. 3.4. Результати прогнозування з використанням звичайної RNN для індексів XOM, S&P 500, WTI та Brent.

З представлених рисунків у нормалізованому масштабі можна бачити, що RNN погано підлаштовується під вихідні значення для всіх обраних наборів даних. Це свідчить про те, потребуються подальші експерименти з архітектурою мережі.

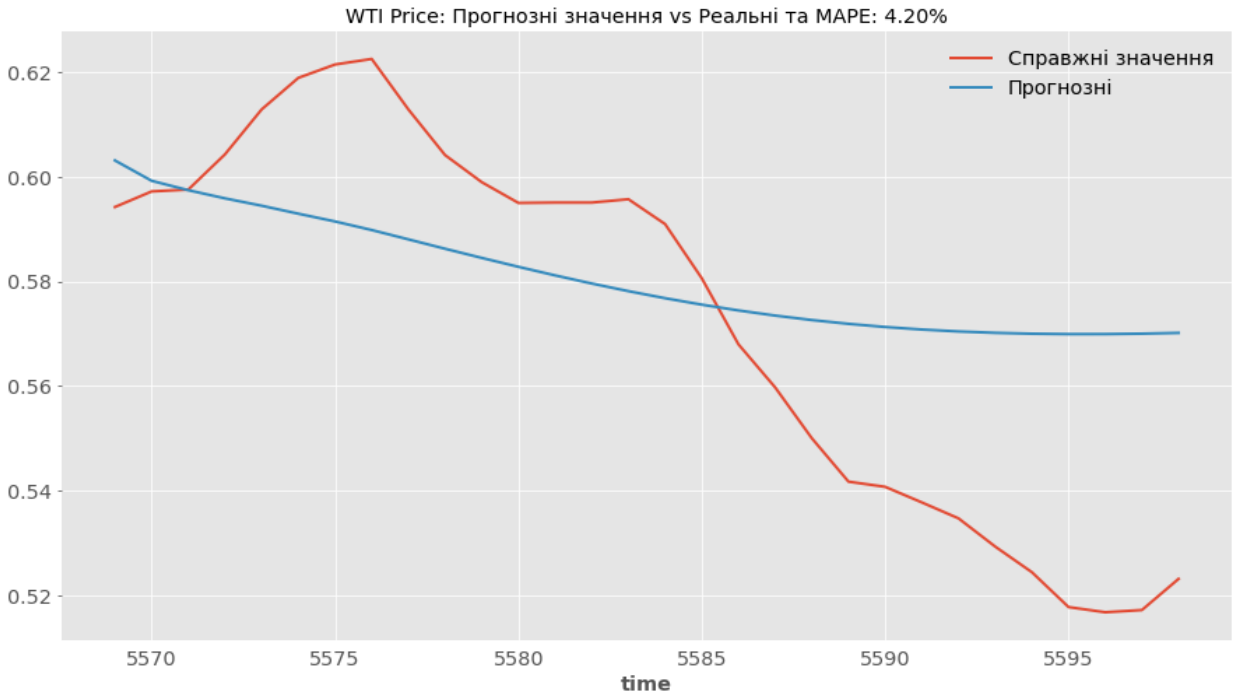
На рис. 3.5 представлено результати прогнозу класичного блоку LSTM для індексів XOM, S&P 500, WTI та Brent.



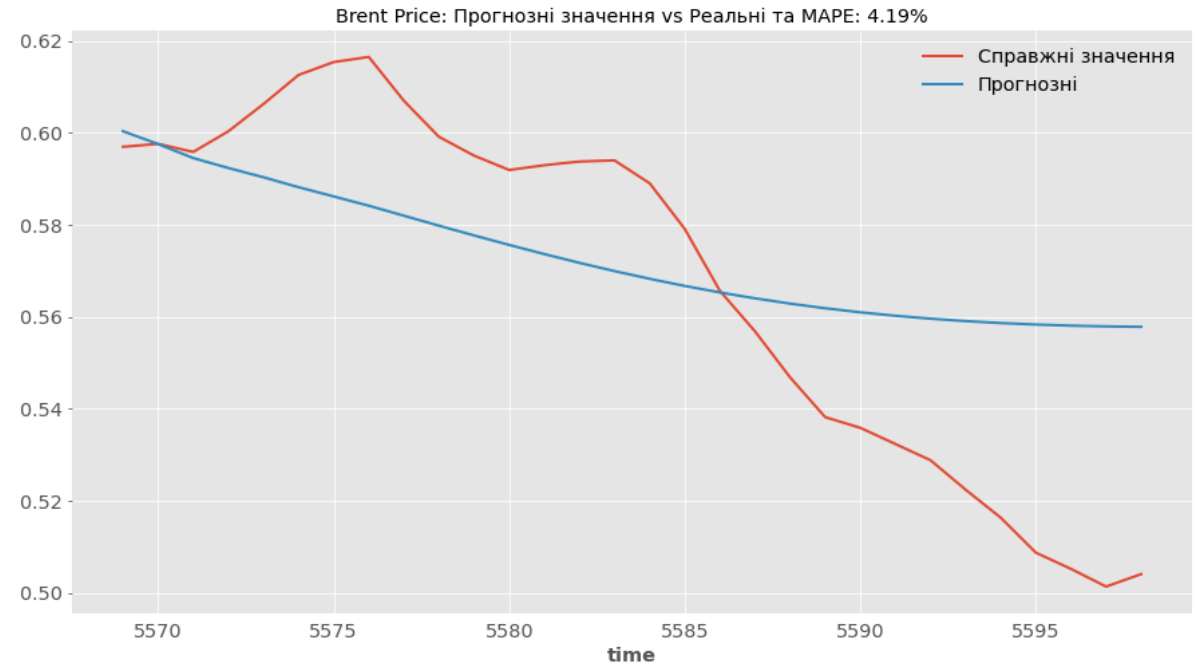
(а)



(б)



(в)



(г)

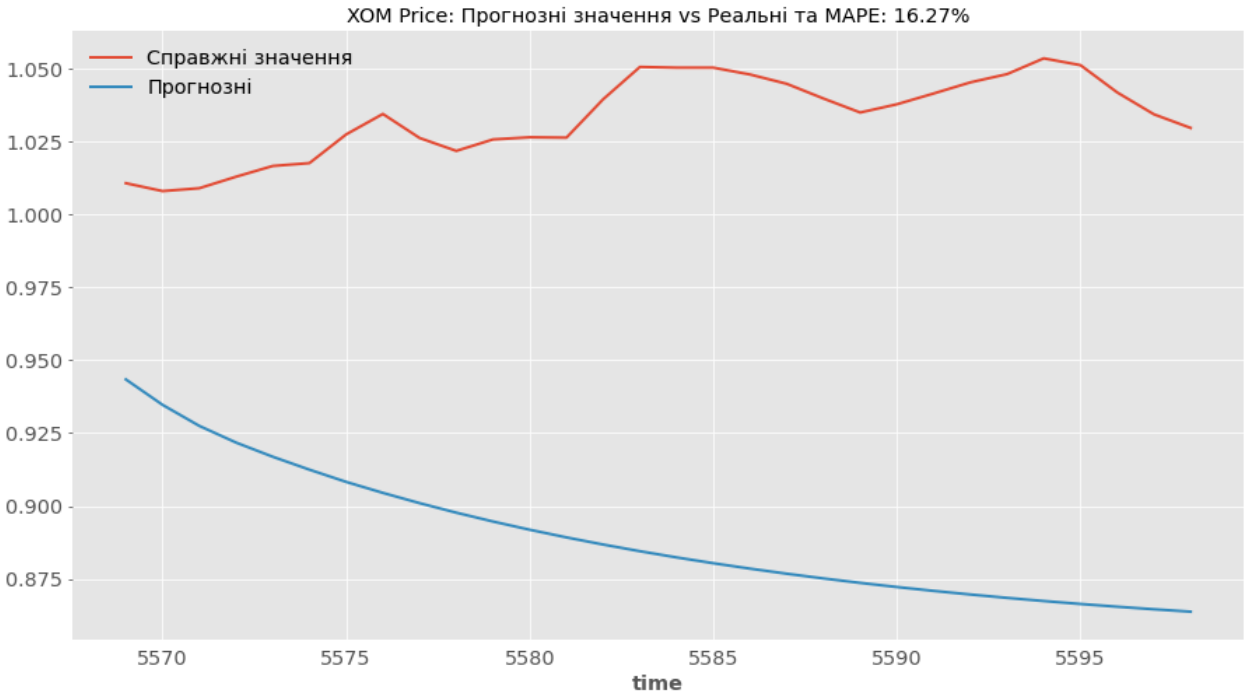
Рис. 3.5. Результати прогнозування з використанням LSTM для індексів XOM, S&P 500, WTI та Brent.

Видно, що представлена мережа може спрогнозувати загальний тренд для останніх двох індексів. Мережа найгірше спрогнозувала значення індексу S&P 500 з

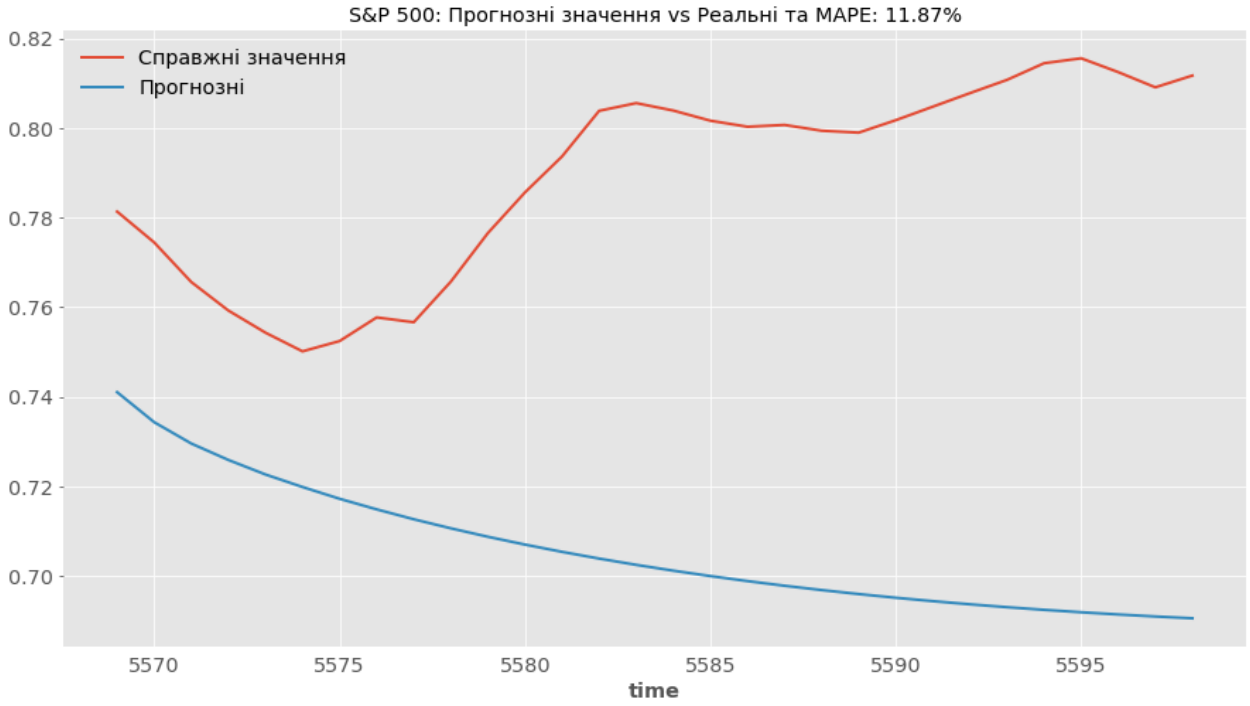
абсолютною похибкою у 18,19%, що вказує на те, що архітектура даної нейронної мережі потребує доопрацювання.

Модель GRU показує схожі результати з LSTM, хоча значення MAPE у середньому є менше, ніж у попередньої моделі. Варто зазначити, що останні дві моделі зовсім корректно спрогнозували загальний тренд на повному відрізкі тестової вибірки значень для індексів XOM та S&P 500.

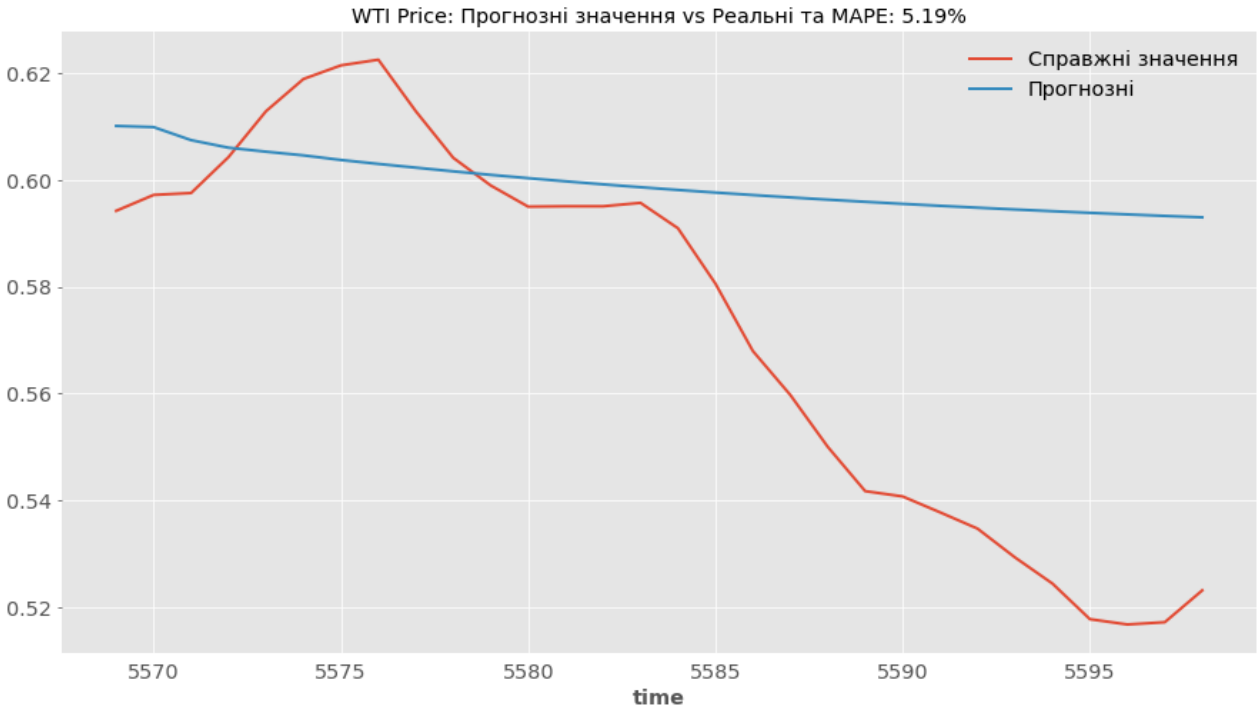
На рис. 3.15 представлено результати прогнозу блоків GRU для індексів XOM, S&P 500, WTI та Brent.



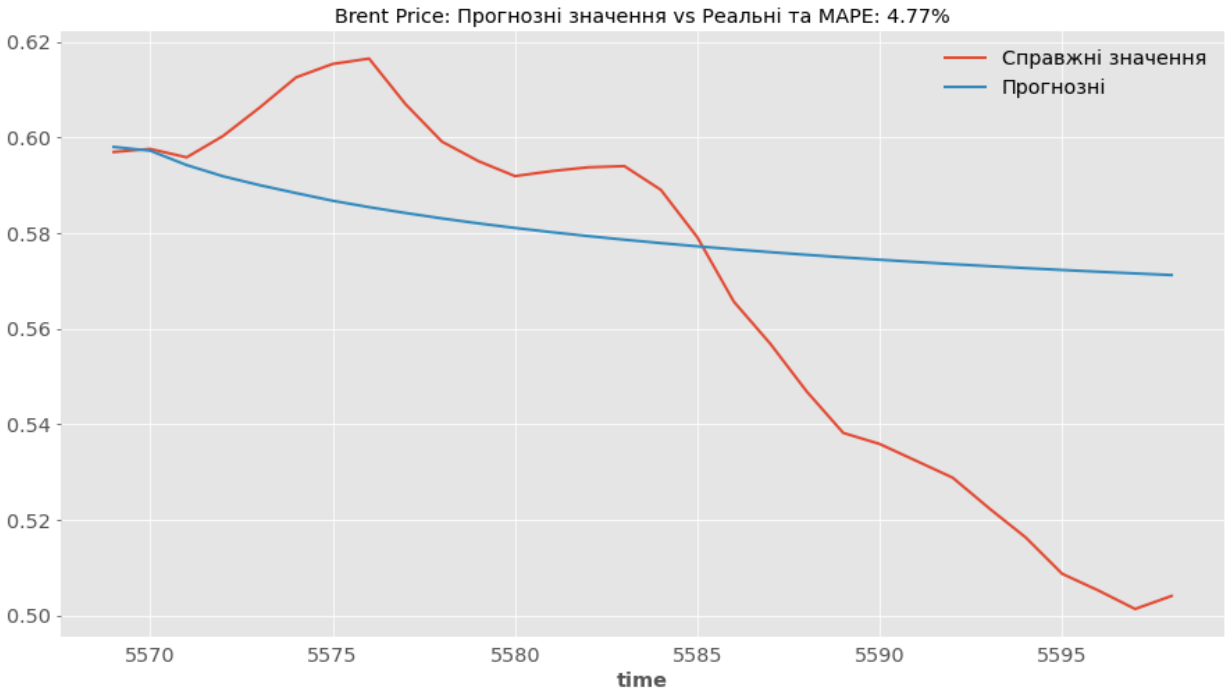
(а)


S&P 500: Прогнозні значення vs Реальні та MAPE: 11.87%
Справжні значення
Прогнозні
0.82
0.80
0.78
0.76
0.74
0.72
0.70
5570
5575
5580
5585
5590
5595
time

(б)


WTI Price: Прогнозні значення vs Реальні та MAPE: 5.19%
Справжні значення
Прогнозні
0.62
0.60
0.58
0.56
0.54
0.52
5570
5575
5580
5585
5590
5595
time

(в)



(г)

Рис. 3.6. Результати прогнозування з використанням GRU для індексів XOM, S&P 500, WTI та Brent.

3.2 Побудова прогнозів поза вибірки навчання

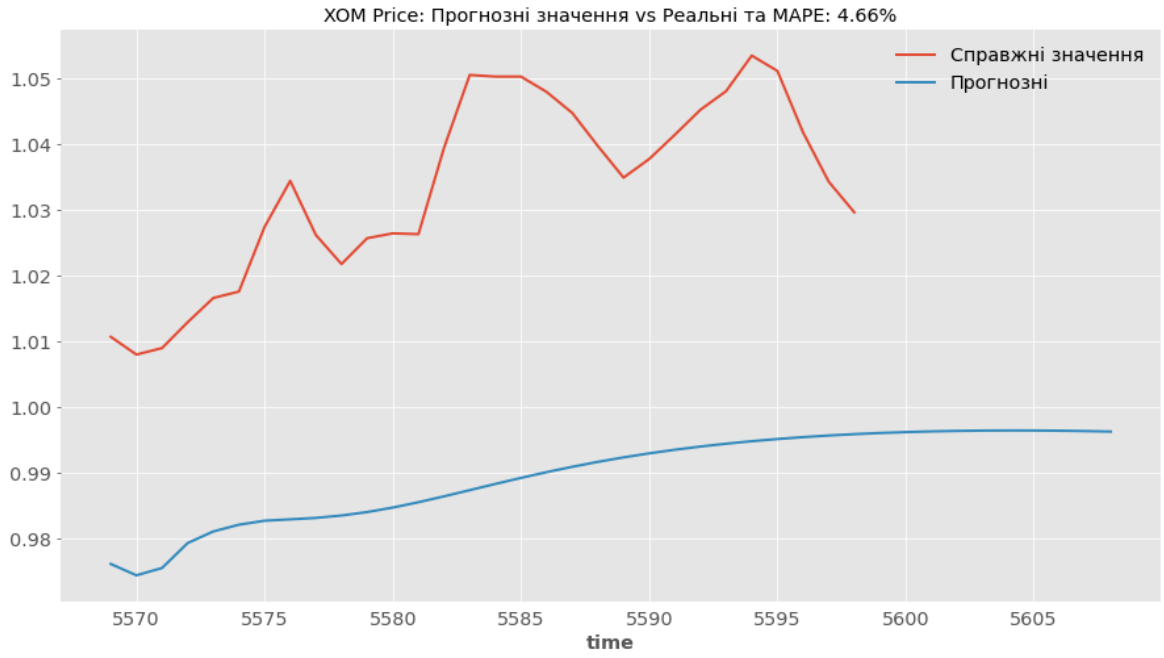
Можливості бібліотеки Darts не обмежуються лише прогнозами, що можна співставити з існуючим рядом, але й тими, що виходять за межі існуючих вибірок. Метод *predict* виконує прогнозування на довільно задану кількість значень, незважаючи на кількість вихідних значень, що були встановлені. Якщо кількість значень перевищує кількість вихідних нейронів – модель виконуватиме прогноз рекурсивним чином.

```
index_fin = 'Brent Price'


pred_series = my_model.predict(n=40, series=train_transformed)

plt.figure(figsize=(15, 8))
val_transformed[index_fin].plot(label='Справжні значення')
pred_series[index_fin].plot(label="Прогнозні")
plt.title(index_fin + ": Прогнозні значення vs Реальні та MAPE: {:.2f}%".format(mape(pred_series[index_fin], val_transformed[inde
plt.legend()
plt.show()
```

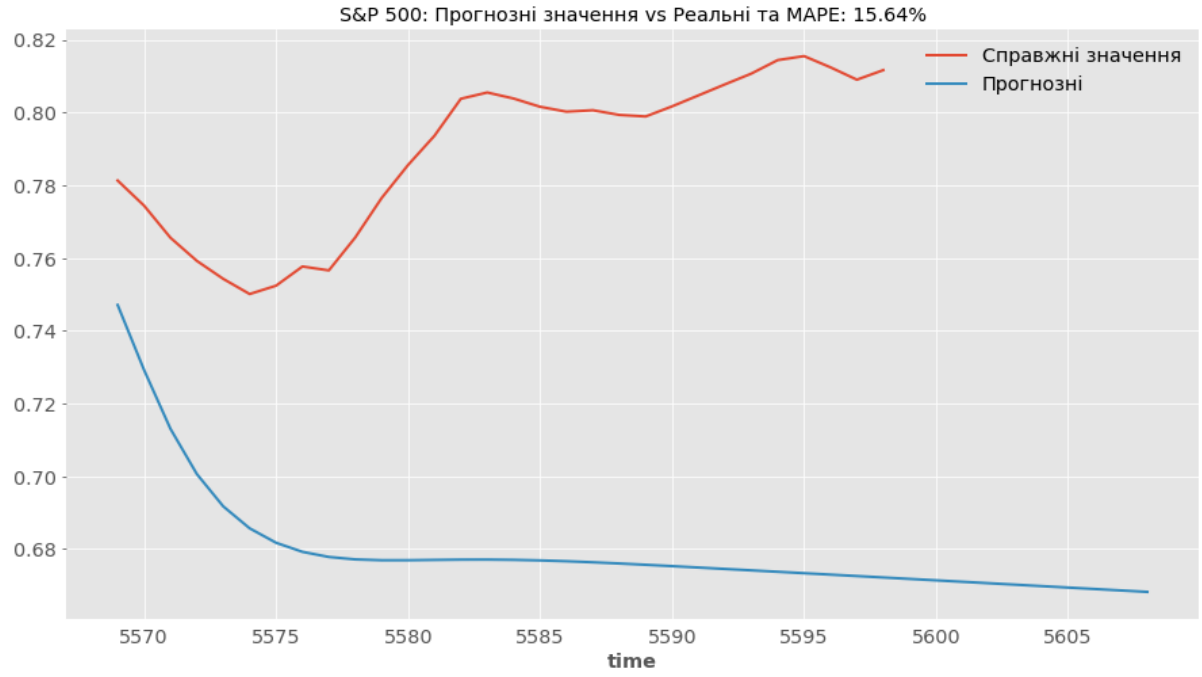
```
Predicting DataLoader 0: 100% 1/1 [00:00<00:00, 90.89it/s]
```

Рис. 3.7. Код побудови прогнозів у майбутнє та виводу результатів.

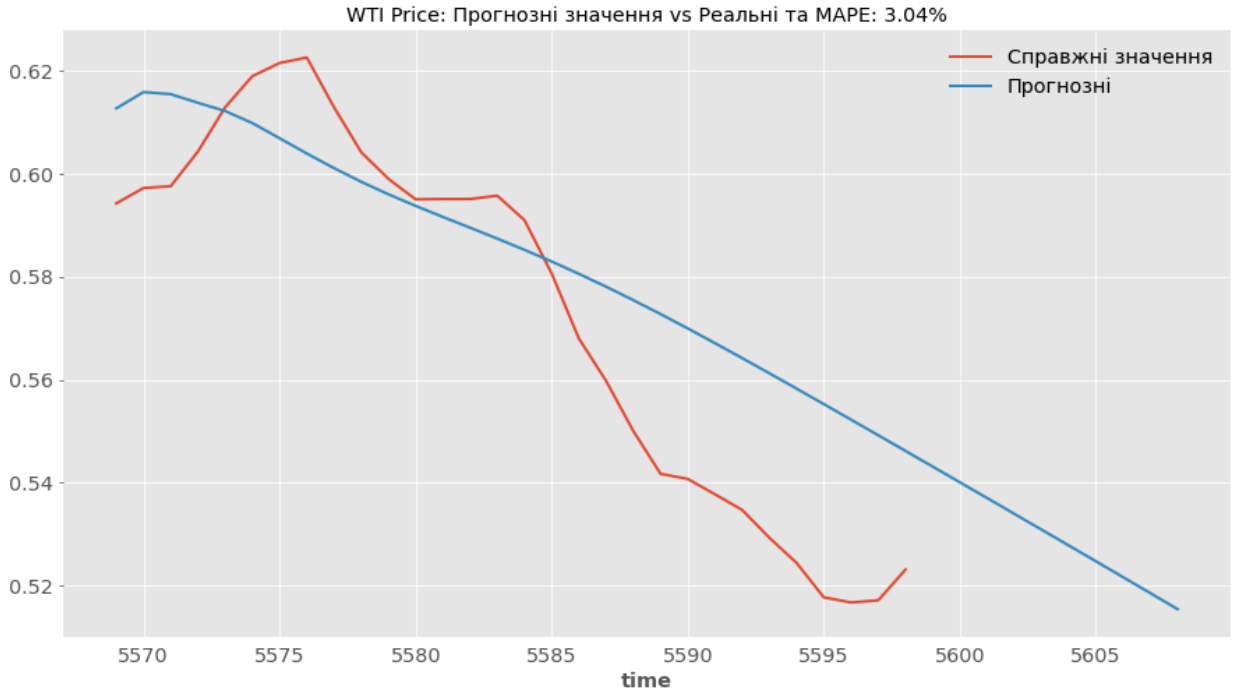
Розглянемо відповідні результати прогнозів поза наших даних. На рис. 3.8 представлено прогноз на 10 днів уперед із використання класичної RNN для XOM, S&P 500, WTI та Brent.



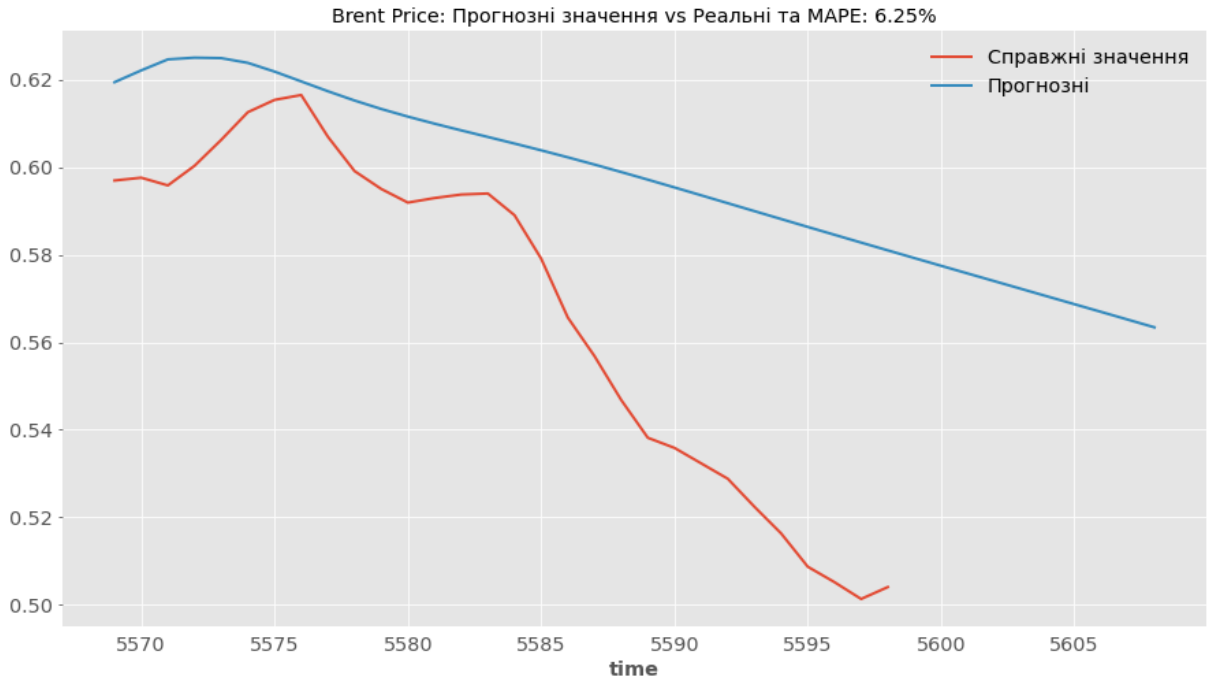
(a)


S&P 500: Прогнозні значення vs Реальні та MAPE: 15.64%
Справжні значення
Прогнозні
0.82
0.80
0.78
0.76
0.74
0.72
0.70
0.68
5570
5575
5580
5585
5590
5595
5600
5605
time

(б)


WTI Price: Прогнозні значення vs Реальні та MAPE: 3.04%
Справжні значення
Прогнозні
0.62
0.60
0.58
0.56
0.54
0.52
5570
5575
5580
5585
5590
5595
5600
5605
time

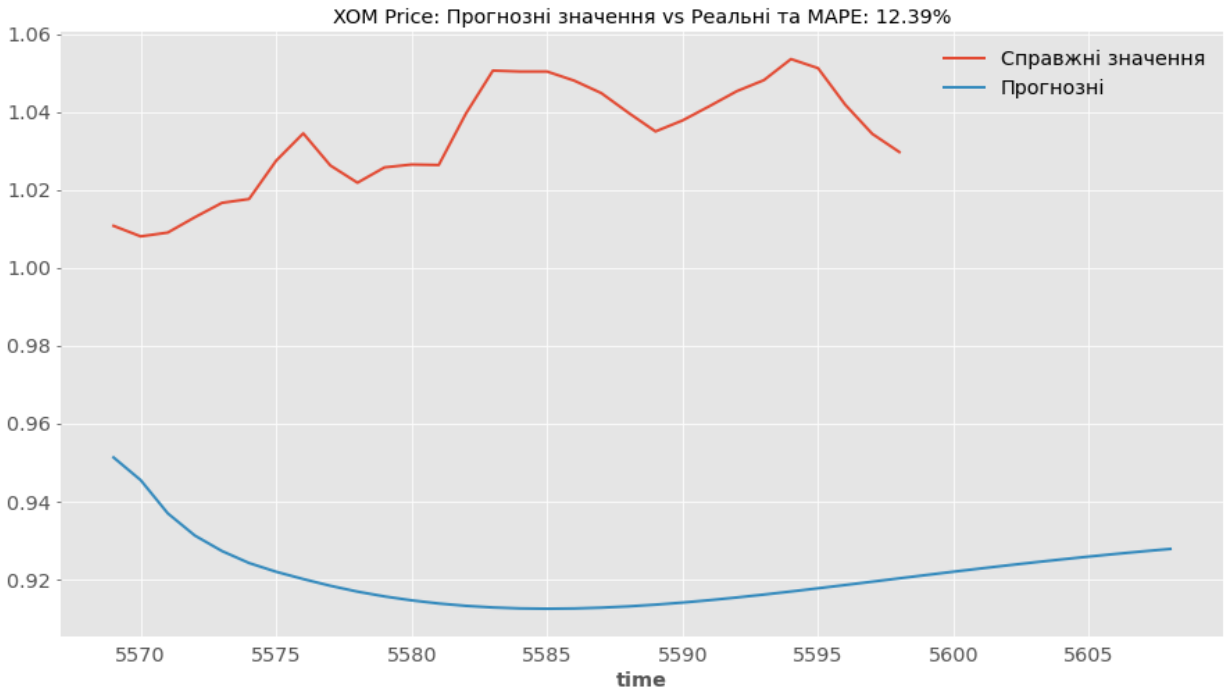
(в)



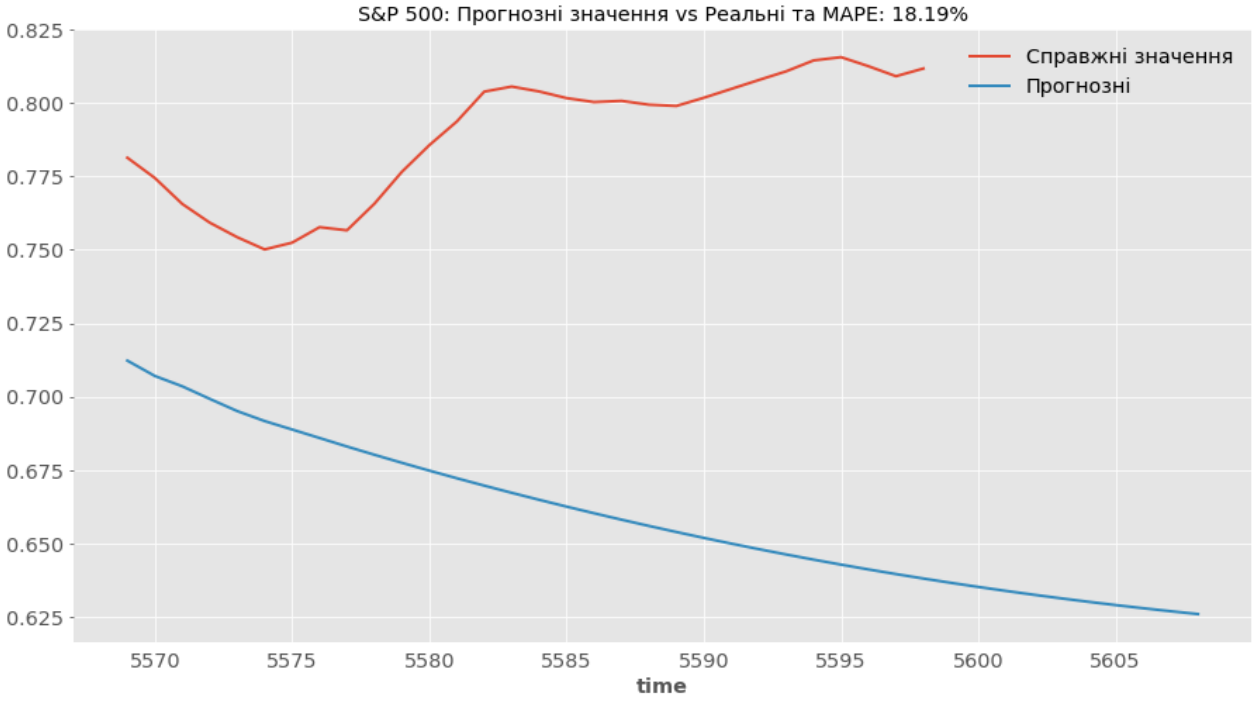
(г)

Рис. 3.8. Прогноз на 10 днів уперед із використання класичної RNN для XOM (а), S&P 500 (б), WTI (в) та Brent (г).

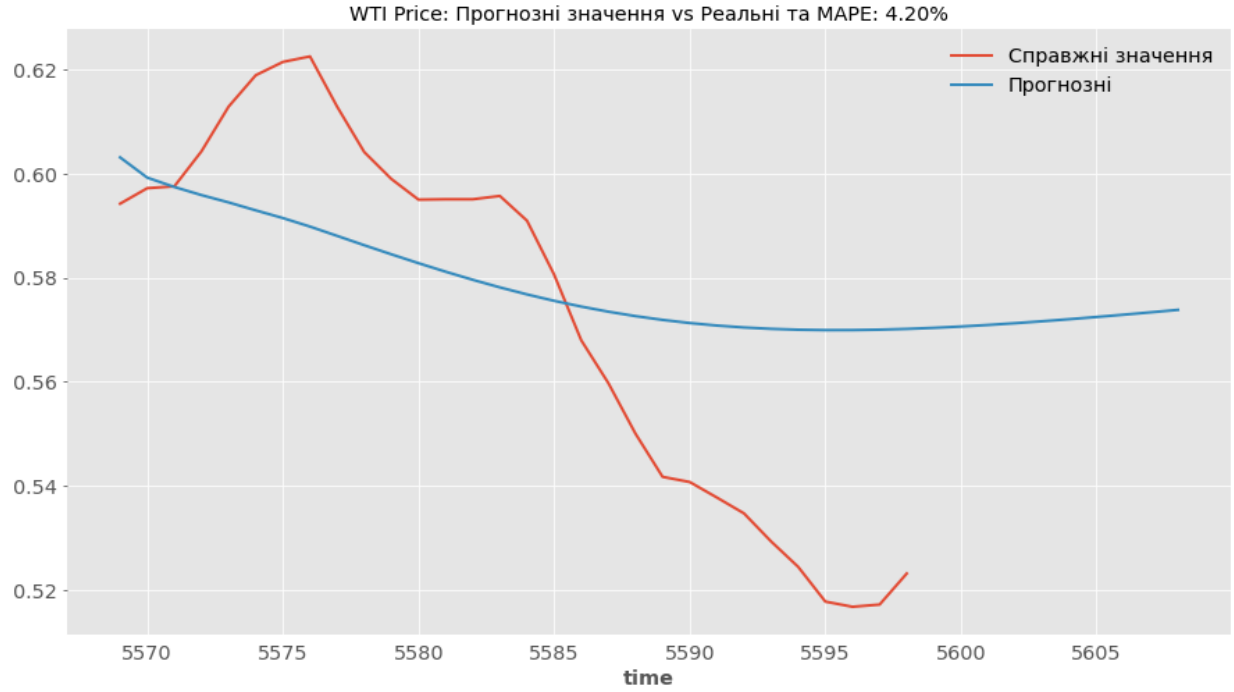
На рис. 3.9 представлено прогноз на 10 днів уперед із використання класичної LSTM для XOM, S&P 500, WTI та Brent.



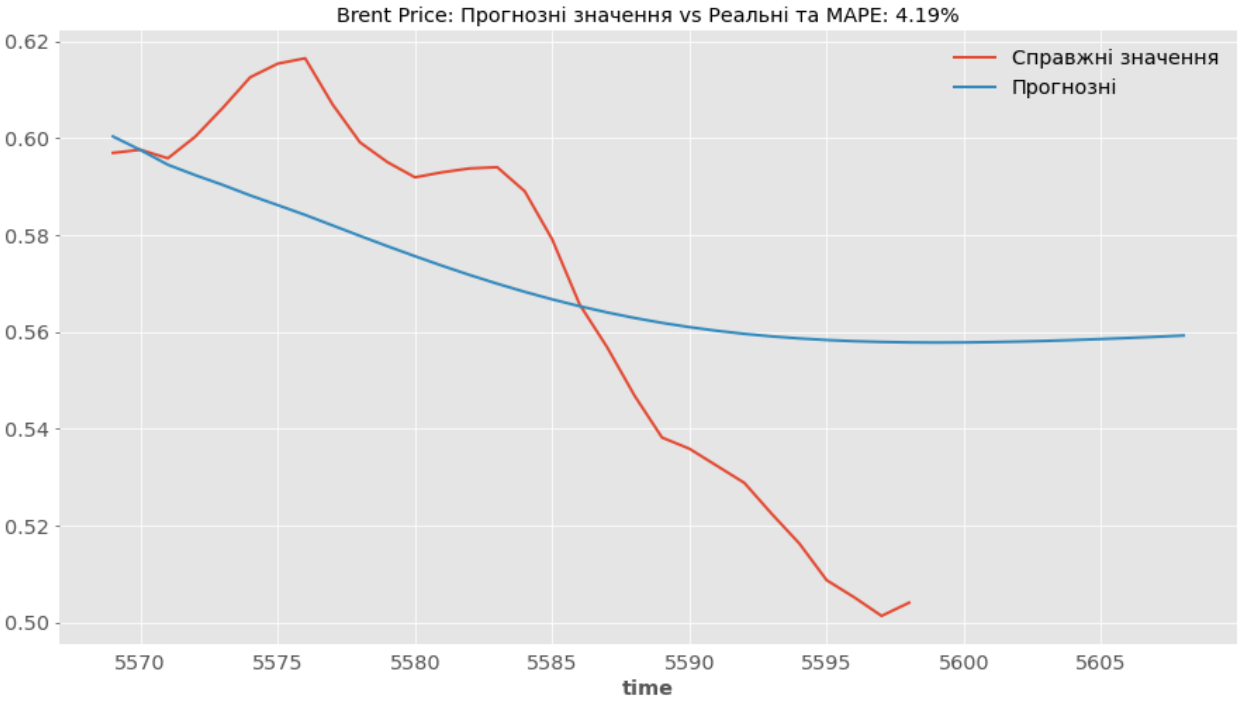
(а)


S&P 500: Прогнозні значення vs Реальні та MAPE: 18.19%
Справжні значення
Прогнозні
0.825
0.800
0.775
0.750
0.725
0.700
0.675
0.650
0.625
5570
5575
5580
5585
5590
5595
5600
5605
time

(б)


WTI Price: Прогнозні значення vs Реальні та MAPE: 4.20%
Справжні значення
Прогнозні
0.62
0.60
0.58
0.56
0.54
0.52
5570
5575
5580
5585
5590
5595
5600
5605
time

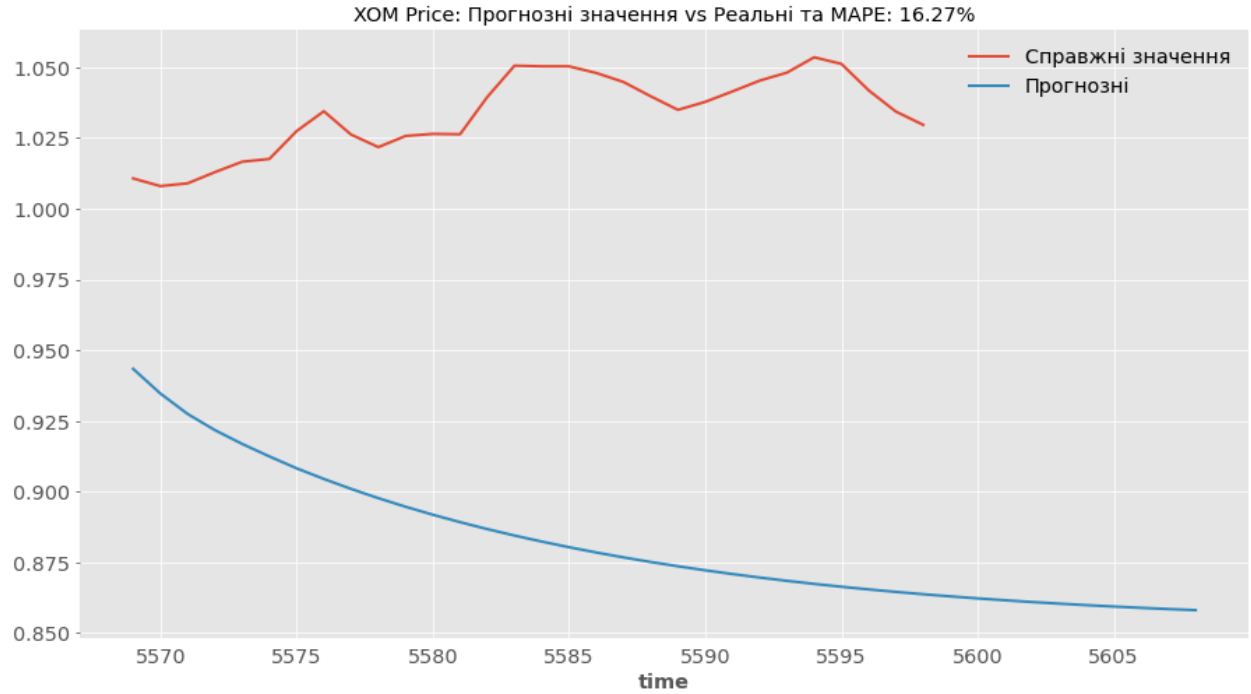
(в)



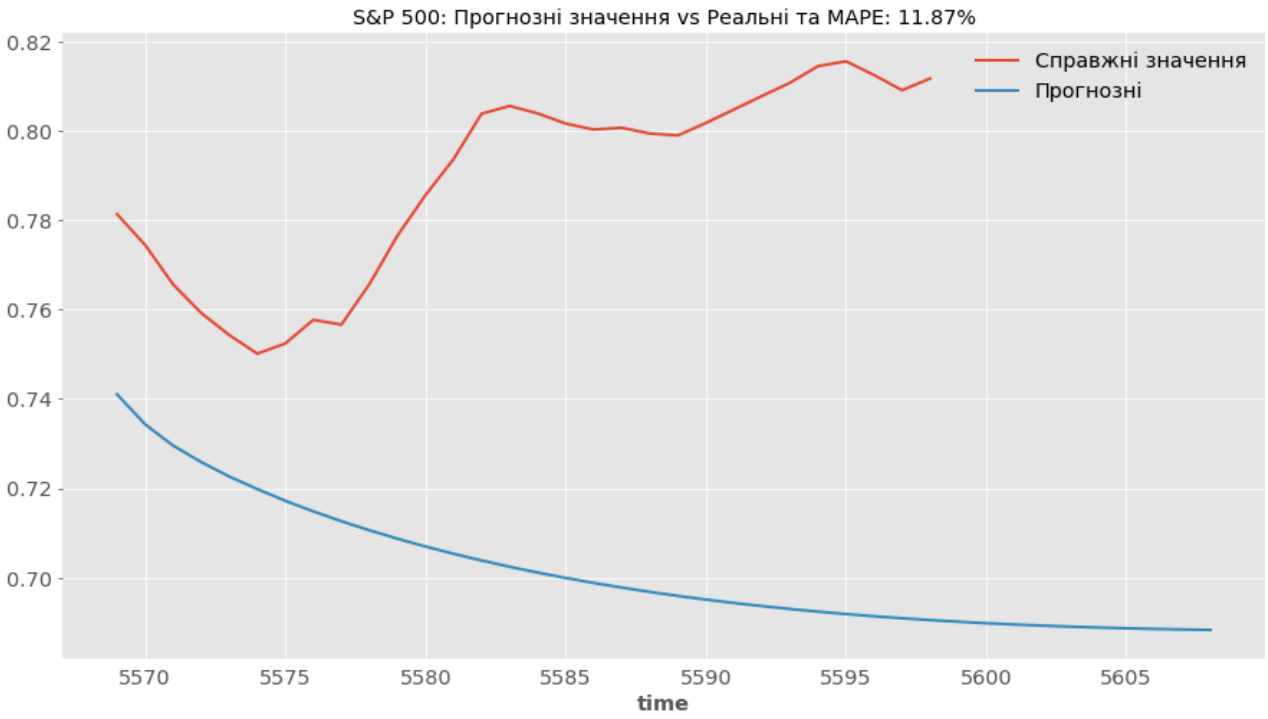
(г)

Рис. 3.10. Прогноз на 10 днів уперед із використання LSTM для індексів XOM (а), S&P 500 (б), WTI (в) та Brent (г).

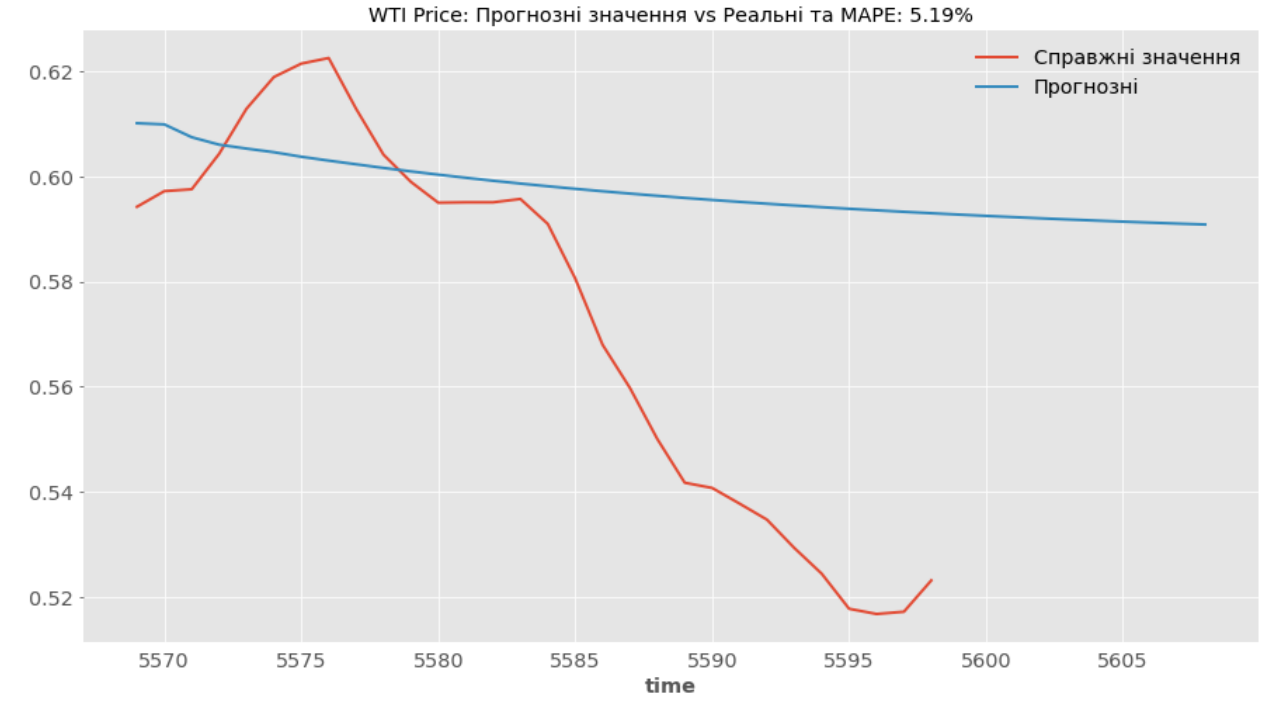
На рис. 3.11 представлено прогноз на 10 днів уперед із використання класичної GRU для XOM, S&P 500, WTI та Brent.



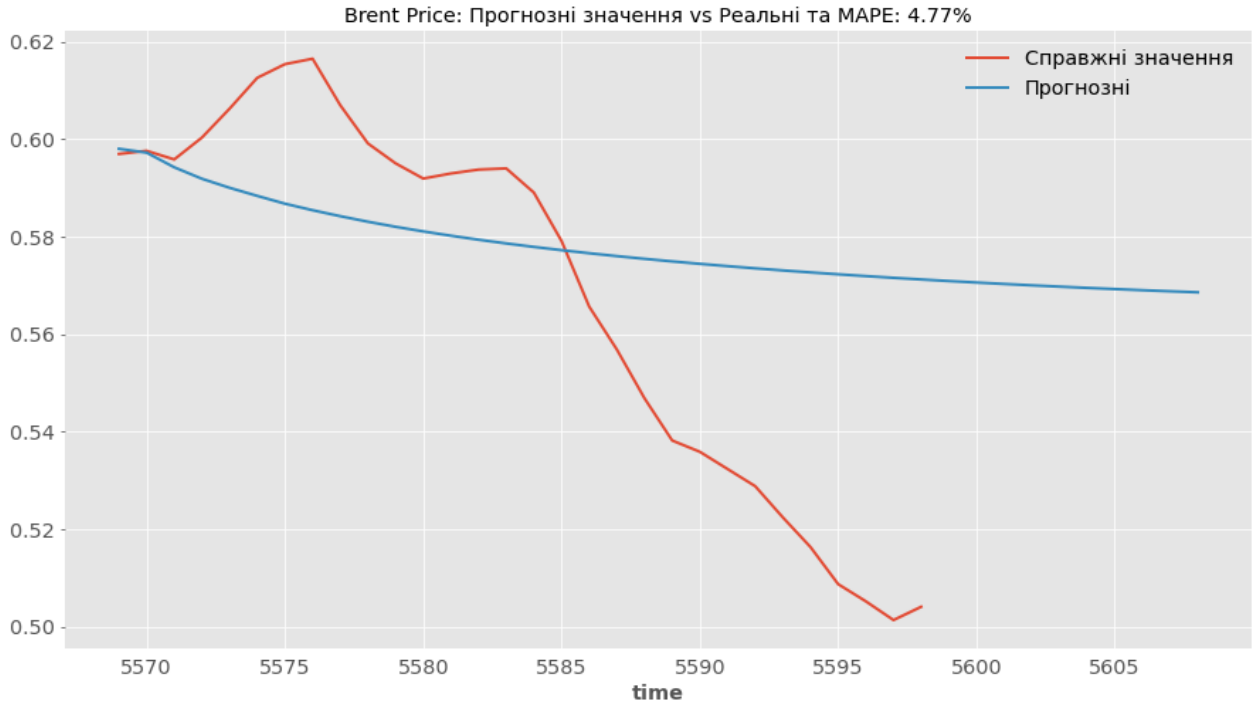
(а)


S&P 500: Прогнозні значення vs Реальні та MAPE: 11.87%
Справжні значення
Прогнозні
0.82
0.80
0.78
0.76
0.74
0.72
0.70
5570
5575
5580
5585
5590
5595
5600
5605
time

(б)


WTI Price: Прогнозні значення vs Реальні та MAPE: 5.19%
Справжні значення
Прогнозні
0.62
0.60
0.58
0.56
0.54
0.52
5570
5575
5580
5585
5590
5595
5600
5605
time

(в)



(г)

Рис. 3.11. Прогноз на 10 днів уперед із використання GRU для індексів XOM (а), S&P 500 (б), WTI (в) та Brent (г).

З представленних вище рисунків можна побачити, що моделі не є чутливими до різких флуктуацій та мають досить велике значення MAPE для прогногнозованних значень, що вказує на доцільність доопрацювання нейромережних моделей для їх подальшого використання. Загалом можна зазначити що для деяких наборів даних, представленних в даному дослідженні нейромережі вдало прогнозували тренд, але якщо порівнювати реальні та прогнозованні значення результат є розбіжним.

Висновки до розділу 3

Мета даного дослідження полягала у тому, щоб дослідити чи можемо ми спрогнозувати майбутні значення індексів фондового ринку за допомогою рекурентих нейронних мереж на прикладі індексів фондового ринку XOM, S&P 500, WTI та Brent.

Для цього ми реалізували нейронні мережі RNN, LSTM та GRU, даючи для навчання історичні данні ціни. Отримані моделі є не завжди придатні для реальної торгівлі.

Мережі можуть ефективно навчатися, але обрана у даному дослідженні стратегія прогнозування не є успішною.

Для більш ефективного навчання треба використовувати більш складні підходи. Можливе використання великої кількості рядів, оптимізація архітектур нейронних мереж – регулювання кількості та значень *dropout* на кожному шарі, збільшення (зменшення) кількості шарів у мережі, обрання іншого оптимізатора як, наприклад, стохастичного градієнтного спуску тощо, − подання до навчання значень корелюючих рядів.

ВИСНОВКИ

Нейромережі набули широкого розповсюдження в області прогнозування динамічних систем: їх використання застосовуються для цілого ряду задач інтелектуального аналізу даних. Для успішного прогнозування майбутніх значень фінансових ринків потрібна велика вибірка однотипних прикладів, які мають приховані взаємозв’язки.

Для використання методів глибокого навчання для вирішення завдання прогнозування ми провели аналіз предметної області прогнозування часових рядів та глибокого навчання. Після ознайомлення з принципами роботи нейронних мереж ми побудували та реалізували обрані нами архітектури, а саме RNN, LSTM та GRU та провели їх навчання.

У результаті прогнозованні значення мають значну похибку і не можуть використовуватися на практиці . Тим не менш, моделі слідували загальному тренду, що свідчить про перспективи подальших досліджень прогнозних моделей на основі нейронних мереж.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Машинное обучение / Сайт «Habr» [Электронний ресурс]. / URL: https://habr.com/company/wunderfund/blog/331310

2. Садовникова Н. Анализ временных рядов и прогнозирование: Учебное пособие / Н.А. Садовникова, Р.А. Шмойлова – М.: Синергия, 2016 – 152 с.

3. Ярушкина Н. Интеллектуальный анализ временных рядов: Учебное пособие / Н.Г. Ярушкина, Т.В. Афанасьева, И.Г. Перфильева – М.: Инфра-М, 2015 – 160 с.1. Горбачевская, Е. Н. Классификация нейронных сетей [Электронний ресурс] / Е. Н. Горбачевская // Вестник ВУиТ. – 2012. – № 2 (19). – URL: https://cyberleninka.ru/article/n/ klassifikatsiya-neyronnyh-setey

4. Сигмоида[Электронный ресурс].URL:https://www.wikiwand.com/ru/%D0%A1%D0%B8%D0%B3%D0%BC%D0%BE%D0%B8%D0%B4%D0%B

5. Хошрейтер С., Шмідхубер Ю., Довга коростроковa пам’ять // Neural Computation. 9 – 1997. – 1735-1780 с.

6. Барский, А.Б. Логические нейронные сети: Учебное пособие / А.Б. Барский. - М.: Бином, 2013. - 352 с.

7. Ширяев, В.И. Финансовые рынки: Нейронные сети, хаос и нелинейная динамика / В.И. Ширяев. - М.: КД Либроком, 2016. - 232 с

8. Бідюк П.І. Аналіз часових рядів: навчальний посібник. К: Політехніка, 2010. 317 с.

9. Robert J. Van Eyden The Application of Neural Networks in the Forecasting of Share Prices. New York: Finance and Technology Publishing, 1996. 326 p.

10. Yoon Y., G. Swales Applying Artificial Neural Networks to Investment Analysis. London: Taylor & Francis, 2011. 80 с.

11. Бенджио, Гудфеллоу, Курвилль: Глубокое обучение. Издательство: ДМК-Пресс, 2018 г. - 652 с.

12. Горелова В. Л., Мельникова Е. Н. Основы прогнозирования систем. — М.: Высшая школа, 1986.

13. Нейронні мережі: їх застосування [Електронний ресурс] – Режим доступу до ресурсу: http://www.poznavayka.org/uk/nauka-i-tehnika-2/ neyronn imerezhi-yih-zastosuvannya-robota/.

14. Samarasinghe S. Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition / Samarasinghe S., 2006.

15. Бринк Х. Машинное обучение / Х. Бринк, Д. Ричардс, М. Феверолф., 2018. – 336 с.

16. Короткий С. Нейронные сети: Основные положения / С. Короткий., 2017. – 69 с.

17. Wolfgang G., Sascha S. Predicting Time Series with SpaceTime Convolutional and Recurrent Neural Networks. ESANN 2017: Proceedings of the technical sessions presented at the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Bruges, Belgium, 2628 April, 2017. Munich: Schwabing, 2018. P. 71–85.

18. Рутковская, Д. Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М. Пилиньский, Л. Рутковский. - М.: РиС, 2013. - 384 с.

19. Ширяев, В.И. Финансовые рынки: Нейронные сети, хаос и нелинейная динамика / В.И. Ширяев. - М.: КД Либроком, 2016. - 232 с

20. Уоссермен Ф. Нейрокомпьютерная техника: теория и практика. — М.: Мир, 1992.

21. Искусственные нейронные сети и приложения: учеб. пособие / Ф.М. Гафаров, А.Ф. Галимянов. – Казань: Изд-во Казан. ун-та, 2018. – 121 с.

22. Прогнозування та аналіз часових рядів. Методичні вказівки до практичних занять та самостійної роботи студентів спеціальності 051 «Економіка» освітня програма «Економічна кібернетика», «Економічна аналітика» / Укл.: Юрченко М. Є. – Чернігів: ЧНТУ, 2018. – 88 с.

23. Филатова, Т. В. Применение нейронных сетей для аппроксимации данных [Электронный ресурс] / Т. В. Филатова // Вестник Томского

государственного университета. – 2004. – № 284. – URL: https://cyberleninka.ru/article/n/primenenie-neyronnyh-setey-dlya-approksimatsii-dannyh

24. Григорьева Д.Р. Методы статистического прогнозирования // Экономический анализ: теория и практика. – 2015 . – №17. – С. 21.

25. Shengdong Du, Tianrui Li, Yan Yang, Shi-Jinn Horng: Deep Air Quality Forecasting Using Hybrid Deep Learning Framework. CoRR abs/1812.04783 (2018).

26. Григорьева Д.Р. Методы статистического прогнозирования // Экономический анализ: теория и практика. – 2015 . – №17. – С. 21.

27. Маккінні У. Python for Data Analysis. / У. Маккінні., 2015. – 482 с.

Вандер Д. П. Python Data Science Handbook: Essential Tools for Working with Data / Дж. Плас Вандер., 2017. – 576 с.

28. Ian Goodfellow, Yoshua Bengio, and Aaron Courville (2016). Deep Learning. MIT Press

29. Deng, L.; Yu, D. (2014). Deep Learning: Methods and Applications. Foundations and Trends in Signal Processing **7** (3-4): 1–199. doi:10.1561/2000000039

30. Bengio, Yoshua (2009). Learning Deep Architectures for AI. Foundations and Trends in Machine Learning **2** (1): 1–127.

31. Bengio, Y.; Courville, A.; Vincent, P. (2013). Representation Learning: A Review and New Perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence **35** (8): 1798–1828. arXiv:1206.5538. doi:10.1109/tpami.2013.50

32. Schmidhuber, J. (2015). Deep Learning in Neural Networks: An Overview. Neural Networks 61: 85–117. arXiv:1404.7828. doi:10.1016/j.neunet.2014.09.003

33. Bengio, Yoshua; LeCun, Yann; Hinton, Geoffrey (2015). Deep Learning. Nature 521: 436–444. doi:10.1038/nature14539

34. Deep Machine Learning — A New Frontier in Artificial Intelligence Research — a survey paper by Itamar Arel, Derek C. Rose, and Thomas P. Karnowski. IEEE Computational Intelligence Magazine, 2013

35. Schmidhuber, Jürgen (2015). Deep Learning. Scholarpedia **10**(11): 32832. doi:10.4249/scholarpedia.32832

36. Carlos E. Perez. A Pattern Language for Deep Learning

37. Zheng Zhao and Huan Liu. (2007). Spectral feature selection for supervised and unsupervised learning. In Proceedings of the 24th international conference on Machine learning (ICML '07). Association for Computing Machinery, New York, NY, USA, 1151–1157. DOI:https://doi.org/10.1145/1273496.1273641

38. Sathya, R. & Abraham, Annamma. (2013). Comparison of Supervised and Unsupervised Learning Algorithms for Pattern Classification. International Journal of Advanced Research in Artificial Intelligence. **2**(2). 10.14569/IJARAI.2013.020206.

39. Love, B.C. Comparing supervised and unsupervised category learning. Psychonomic Bulletin & Review 9, 829–835 (2002). https://doi.org/10.3758/BF03196342

40. Turner, Jenine & Charniak, Eugene. (2005). Supervised and Unsupervised Learning for Sentence Compression. 10.3115/1219840.1219876.

41. J. Corchado, C. Fyfe and B. Lees, "Unsupervised learning for financial forecasting," Proceedings of the IEEE/IAFE/INFORMS 1998 Conference on Computational Intelligence for Financial Engineering (CIFEr) (Cat. No.98TH8367), 1998, pp. 259-263, doi: 10.1109/CIFER.1998.690316.

42. Salvador B, Oosterlee CW, van der Meer R. Financial Option Valuation by Unsupervised Learning with Artificial Neural Networks. Mathematics. 2021; 9(1):46. https://doi.org/10.3390/math9010046

43. Wang, Bao & Kong, Yue & Zhang, Yongtao & Liu, Dapeng & Ning, Lianju. (2019). Integration of Unsupervised and Supervised Machine Learning Algorithms for Credit Risk Assessment. Expert Systems with Applications. 128. 10.1016/j.eswa.2019.02.033.

44. Dixon, Matthew Francis and Halperin, Igor, The Four Horsemen of Machine Learning in Finance (September 15, 2019). http://dx.doi.org/10.2139/ssrn.3453564

45. M. Heidari, S. Zad and S. Rafatirad, "Ensemble of Supervised and Unsupervised Learning Models to Predict a Profitable Business Decision," 2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), 2021, pp. 1-6, doi: 10.1109/IEMTRONICS52119.2021.9422649.

46. Olshausen, B. A. (1996). "Emergence of simple-cell receptive field properties by learning a sparse code for natural images". Nature. 381 (6583): 607–609. Bibcode:1996Natur.381..607O. doi:10.1038/381607a0. PMID 8637596. S2CID 4358477.

47. Bengio, Yoshua; Lee, Dong-Hyun; Bornschein, Jorg; Mesnard, Thomas; Lin, Zhouhan (13 February 2015). "Towards Biologically Plausible Deep Learning". arXiv:1502.04156

48. K. Bandara, P. Shi, C. Bergmeir, H. Hewamalage, Q. Tran, and B. Seaman, "Sales demand forecast in E-commerce using a long short-term memory neural network methodology," in Proc. ICONIP, Sydney, NSW, Australia, 2019, pp. 462–474.

49. Z. Hu, Y. Zhao, and M. Khushi, "A Survey of Forex and stock price prediction using deep learning," Appl.Syst.Innov., vol. 4, no. 9, 2021. DOI: 10.3390/asi4010009.

50. R. Zhang, Z. Yuan, and X. Shao, "A new combined CNN-RNN model for sector stock price analysis," in Proc. COMPSAC, Tokyo, Japan, 2018.

51. A. Dolatabadi, H. Abdeltawab, and Y. Mohamed, "Hybrid deep learningbased model for wind speed forecasting based on DWPT and bidirectional LSTM Network," IEEE Access., vol. 8, pp. 229219–229232, 2020. DOI: 10.1109/ACCESS.2020.3047077.

52. M. Alazab, S. Khan, S. Krishnan, Q. Pham, M.Reddy, and T. Gadekallu, "A multidirectional LSTM model for predicting the stability of a smart grid," IEEE Access., vol. 8, pp. 85454–85463, 2020. DOI: 10.1109/ACCESS.2020.2991067.

53. T. Liu, T. Wu, M. Wang, M. Fu, J. Kang, and H. Zhang, "Recurrent neural networks based on LSTM for predicting geomagnetic field," in Proc. ICARES, Bali, Indonesia, 2018.

54. G. Lai, W. Chang, Y. Yang, and H. Liu, "Modelling long-and short-term temporal patterns with deep neural networks," in Proc. SIGIR, 2018.

55. M. Alhussein, K. Aurangzeb, and S. Haider, “Hybrid CNN-LSTM model for short-term individual household load forecasting,” IEEE Access., vol. 8, pp. 180544–180557, 2020. DOI: 10.1109/ACCESS.2020.3028281.

56. J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, ”Empiral evaluation of gated recurrent neural networks on sequence modelling,” 2014, arXiv: 1412.3555. [Online]. Available: https://arxiv.org/abs/1412.3555