МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«РОЗРОБКА ВЕБ ЗАСТОСУНКУ ПРОГНОЗУВАННЯ БАГАТОВАРІАНТНИХ ЧАСОВИХ РЯДІВ»**

Виконала: студентка 2 курсу, групи 8.1210-з

спеціальності 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

освітньої програми Інженерія програмної забезпечення
(назва освітньої програми)

А.С. Преподобна
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, доцент, к.ф.-м.н. Кудін О.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри фундаментальної та прикладної математики, доцент, д.т.н. Гребенюк С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти магістр

Спеціальність 121 Інженерія програмного забезпечення

Освітня програма Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.

(підпис)

« ____ » _____ 2021 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТЦІ

Преподобній Анастасії Сергіївні

(прізвище, ім'я та по– батькові)

1. Тема роботи (проєкт) Розробка веб застосунку прогнозування багатоваріантних часових рядів

керівник роботи (проєкту) Кудін Олексій Володимирович, к.ф.-м.н., доцент

(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 09 » червня 2021 року № 850-с

2. Строк подання студентом роботи 25.11.2021

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково – пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі, аналіз предметної області.
2. Моделювання та проєктування програмного доповнення.
3. Розробка веб застосунку прогнозування багатоваріантних часових рядів.

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

6. Консультанти розділів роботи

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

7. Дата видачі завдання 09.06.2021

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану виконання кваліфікаційної роботи магістра.	02.09.2021	
2.	Збір вихідних даних та аналіз предметної області.	17.09.2021	
3.	Обробка методичних та теоретичних джерел. Робота над першим розділом.	08.10.2021	
4.	Моделювання та проєктування системи. Робота над другим розділом.	29.10.2021	
5.	Реалізація та тестування системи. Робота над третім розділом.	12.11.2021	
6.	Оформлення та нормоконтроль кваліфікаційної роботи.	23.11.2021	
7.	Захист кваліфікаційної роботи.	09.12.2021	

Студент _____ А.С. Преподобна
(підпис) (ініціали та прізвище)

Керівник роботи _____ О.В. Кудін
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ С.П. Швидка

РЕФЕРАТ

Кваліфікаційна робота магістра «Розробка веб застосунку прогнозування багатоваріантних часових рядів»: 45 с., 16 рис., 24 джерела.

БАГАТОВАРІАНТНИЙ ЧАСОВИЙ РЯД, КОМБІНОВАНЕ МОДЕЛЮВАННЯ, НЕЙРОННА МЕРЕЖА, СТАТИСТИЧНА МОДЕЛЬ, ПРОГНОЗУВАННЯ, МАШИННЕ НАВЧАННЯ, ГЛИБИННЕ НАВЧАННЯ.

Об’єкт дослідження – часові ряди.

Предмет дослідження – сервіси прогнозування багатоваріантних часових рядів.

Мета роботи: розробити веб-додаток для прогнозування багатоваріантних часових рядів.

Методи дослідження – методи збору та аналізу вимог до програмного забезпечення, методи моделювання, проєктування, конструювання та тестування програмного забезпечення.

У кваліфікаційній роботі викладено підхід до прогнозування багатоваріантних часових рядів шляхом створення веб застосунку.

Розглянуто основні методи аналізу та прогнозування часових рядів. Розглянуто концепцію побудови алгоритму машинного навчання та нейронної мережі для прогнозування багатоваріантного часового ряду. На основі вивченого матеріалу розроблено веб застосунок, за допомогою якого користувач може прогнозувати поведінку певного сценарію подій.

Результати роботи можуть бути використані для побудови нових алгоритмів машинного навчання.

SUMMARY

Master’s qualifying paper «Development of the Web Application for Multivariate Time Series Forecasting»: 45 pages, 16 figures, 24 references.

MULTIVARIATE TIME-SERIES, COMBINED MODELING, NEURAL NETWORK, STATISTICAL MODEL, FORECASTING, MACHINE LEARNING, DEEP LEARNING.

The object of the study is time series.

The subject of the study is multivariate time series forecasting services.

The aim of the study: to develop a web application for predicting multivariate time series.

The methods of research are methods of collection and analysis of software requirements, modeling, design, construction and testing of software.

The qualification work presents an approach to predicting multivariate time series by creating a web application.

The main methods of analysis and forecasting of time series are considered. The concept of building of machine learning algorithm and a neural network for predicting a multivariate time series is considered. Based on the studied material, a web application was developed, with the help of which one of the users can predict the behavior of a certain event scenario.

The results of the work can be used to build new machine learning algorithms.

ЗМІСТ

ВСТУП

Сутність та стан наукової проблеми. Прогнозування багатоваріантних часових рядів широко вивчається в метеорології, енергетиці, фінансах, транспортній галузі тощо. Ціноутворення фондових ринків, почасові об’єми використання електроенергії побутовими та комерційними споживачами, прогнозування метеорологічних явищ, народжуваність, трафік, прогнозування якості повітря – це лише декілька прикладів часових рядів.

Основна мета аналізу часового ряду – прогнозувати його значення на майбутні періоди. Що, в свою чергу, напряму пов’язано з плануванням та поняттям ефективного прийняття рішень. Постає питання – яким чином досягти найбільш достовірного прогнозування?

Оскільки безперервний моніторинг і збір даних стають все більш поширеними, потреба в компетентному аналізі часових рядів за допомогою методів статистики та машинного навчання збільшиться. Найперспективніші нові моделі прогнозування поєднують обидві методології. Саме тому обрана предметна область представляє неабиякий інтерес до вивчення.

Ступінь розробленості наукової проблеми (огляд інформаційних джерел). Новатором, одним з перших, хто зробив спробу прогнозування часового ряду, вважається Джон Граунт (John Graunt), англієць за походженням [1]. У XII столітті він, обробивши записи про смертність за останні приблизно сто років, у 1662 році опублікував наукову працю, яка пізніше стане початком аналізу та прогнозування демографічної ситуації.

Прогнозуванню поведінки часових рядів присвячено ряд наукових робіт. Одними з останніх, які при глибинному вивченні, розкривають поняття, характеристики часового ряду, моделі прогнозування, методи оцінки параметрів коливань, зміну стійкості рівнів ряду, містять приклади математичних моделей одно- та багатоваріантних часових рядів та їх реалізацію мовою програмування Python, є роботи [2, 3]. Вони складають

теоретичну основу проведення дослідження в межах даної кваліфікаційної роботи.

Серед досліджень, опублікованих у періодичних наукових виданнях, слід виділити [4–10]. Нейромережевий підхід до багатовимірного аналізу часових рядів представлено у [4]. На прикладі зміни ціноутворення на борошно, автори показали, що такий метод аналізу є найперспективнішим серед підходів статистичного моделювання.

Перевагу багатовимірних моделей часових рядів над одновимірними у напрямку отримання більш точних прогнозів наведено у [5]. Дослідження проводилося у сфері міжнародного туристичного попиту.

Для найбільш точного прогнозування, важливо моделювати довгострокову залежність часових рядів. Досягнути бажаних результатів точності можливо використовуючи моделі глибинного навчання, засновані на методах рекурентної нейронної мережі (RNN) та згорткової нейронної мережі (CNN). Так, автори [6] використовують RNN з механізмом уваги. Підхід базується на використанні набору фільтрів для виділення незмінних у часі тимчасових моделей.

У [7] набір даних аналізується з використанням моделі багатовимірної мережі часової згортки (M-TCN).

Прогнозування багатоваріантного часового ряду на прикладі аналізу даних попиту у відділені невідкладної допомоги наведено у [8]. Автори дійшли висновку, що багатоваріантні моделі забезпечують більш точні прогнози в порівнянні з одноваріантними.

Порівняння моделей ARIMA, LSTM та GRU для прогнозування часового ряду ціноутворення біткоїну (Bitcoin’s) показують, що модель авторегресійного інтегрованого ковзного середнього (ARIMA) дає кращі результати в порівнянні регресійними моделями [9].

Актуальному питанню сьогодення – прогнозуванню щодо COVID-19, використовуючи моделі глибинного навчання LSTM, GRU та Bi-LSTM, присвятили своє дослідження автори [10]. Проводилося прогнозування

часових рядів підтверджених випадків, смертності, та одужання на прикладі десяти найбільших країн світу. З'ясувалося, що у більшості випадків модель Bi-LSTM дає найкращий результат у порівнянні з іншими. Тому саме цю модель автори рекомендують для прогнозування пандемії для кращого планування та управління.

Узагальнюючи наведені вище статті, можна дійти висновку, що не дивлячись на широке використання аналізу часових рядів у найрізноманітніших сферах життя, саме прогнозування часового ряду – є одним з найважчих та найменш вивчених методів машинного навчання.

Нестабільність, швидкоплинність, закономірні та випадкові фактори впливу на об'єкт дослідження обумовлюють **актуальність роботи** в обраній предметній сфері. Оскільки, опис математичних моделей багатоваріантних часових рядів мовами програмування досить складний для сприйняття пересічним користувачем, актуальним напрямом стає розробка веб додатків та застосунків.

Метою дослідження є розробити веб застосунок для прогнозування багатоваріантних часових рядів.

Завдання дослідження. Досягнення поставленої мети передбачає реалізацію наступних завдань:

- проаналізувати існуючі методи дослідження часових рядів;
- розробити нейронну мережу для прогнозування багатоваріантних часових рядів;
- протестувати розроблений веб застосунок та проаналізувати отримані результати.

Об'єкт та предмет дослідження. Об'єкт дослідження – часові ряди. Предмет дослідження – сервіси прогнозування багатоваріантних часових рядів.

Вибір об'єкта дослідження здійснено на основі аналізу як класичних, так і сучасних літературних джерел у напрямку можливості прогнозування тих чи

інших явищ з високою надійністю співпадіння очікуваного результату з реальним.

Наукова новизна та практична значущість отриманих результатів. Наукова новизна дослідження полягає у наступному: проаналізовано сучасні сервіси прогнозування багатоваріантних часових рядів; на основі проведеного аналізу, розроблено веб застосунок для прогнозування багатоваріантних часових рядів на основі нейронної мережі. Практична значущість полягає у можливості безпосереднього використання розробленого веб застосунку для прогнозування багатоваріантних часових рядів для різних наборів типів даних.

Структура роботи. Логіка дослідження зумовила таку структуру кваліфікаційної роботи: вступ, 3 розділи, висновки, перелік посилань із 24 найменувань.

Перший розділ присвячений загальному опису поняття часового ряду та його характеристик. Представлено огляд моделей прогнозування та вимоги до програмного забезпечення.

У другому розділі висвітлюються традиційні методи підготовки даних (серед інших метод згладжування, нормалізації).

Програмна реалізація веб застосунку прогнозування багатоваріантних часових рядів, реалізована бібліотекою Python скриптів, представлена у третьому розділі.

1 ОСОБЛИВОСТІ ЧАСОВИХ РЯДІВ

1.1 Часові ряди та їх характеристики

Часовий ряд – це послідовність упорядкованих у часі числових показників, що характеризують рівень стану та зміни досліджуваного явища [11].

Нехай $y_1, y_2, y_3, \dots, y_T$ – значення спостережень за процесом протягом T періодів [12]. Ця послідовність є числовими значеннями, кожне з яких має відповідний індекс, який залежить від номера періоду, в який він спостерігався. Така послідовність, записана у порядку зростання індексу, називається часовим рядом. Позначимо часовий ряд з T елементами:

$$\{Y_T\}. \tag{1.1}$$

Задача прогнозу полягає в побудові математичної моделі ряду, за допомогою якої можна пояснити поведінку ряду і здійснити прогноз на майбутні періоди. Тобто, знайти

$$y_{T+l}, \tag{1.2}$$

де l – час упередження.

Зрозуміло, що точно спрогнозувати значення випадкового процесу, яким є часовий ряд, неможливо в принципі, а тому прогноз здійснюють, домагаючись мінімуму якогось критерію адекватності прогнозної моделі.

Якщо значення T є малим (1, 2 кроки), то одним із таких критеріїв може бути дисперсія відхилення $y_T(l)$ від y_{T+l}, яка повинна бути для оптимальної моделі прогнозу мінімальною, тобто

$$E\left\{\left(y_{T+l} - y_T(l)\right)^2\right\} \to min, \tag{1.3}$$

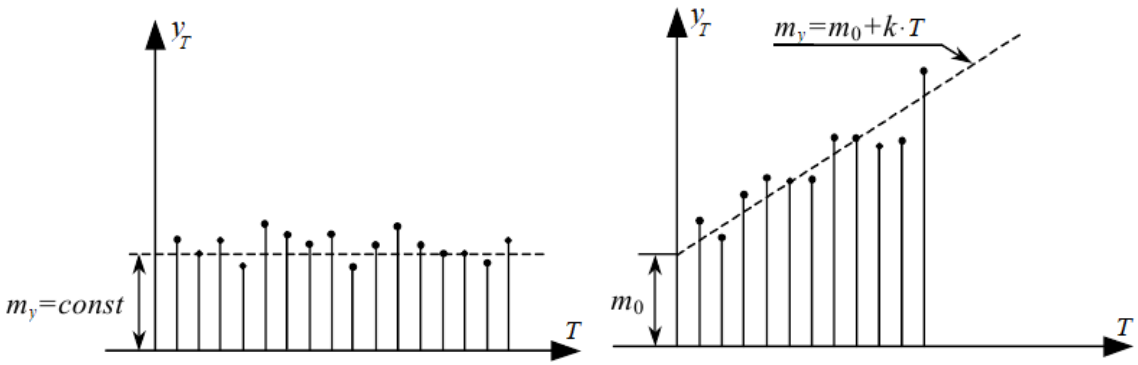
де E – символ операції математичного очікування [13].

Існують й інші критерії оптимальності моделей.

Це найпростіше визначення поняття часового ряду. Зрозуміло, що на реальний часовий ряд, окрім зміни в часі досліджуваного явища, накладається низка додаткових факторів, нециклічних чинників, що можуть суттєво вплинути на прогнозування подальшого перебігу подій. Тому, доцільним надалі буде навести основні характеристики часових рядів.

Виділяють три основні характеристики часових рядів: стаціонарність, тенденція, сезонність.

Стаціонарність. Як і будь-який інший випадковий процес, часовий ряд y_T може бути стаціонарним (див. рис. 1.1, а) або нестаціонарним (див. рис. 1.1, б) [13].



а *б*

Рисунок 1.1 – Графік реалізації часового ряду:
а – стаціонарного; *б* – нестаціонарного

Для стаціонарного часового ряду характерною є рівновага його значень y_T біля середнього значення m_y, яке є константою.

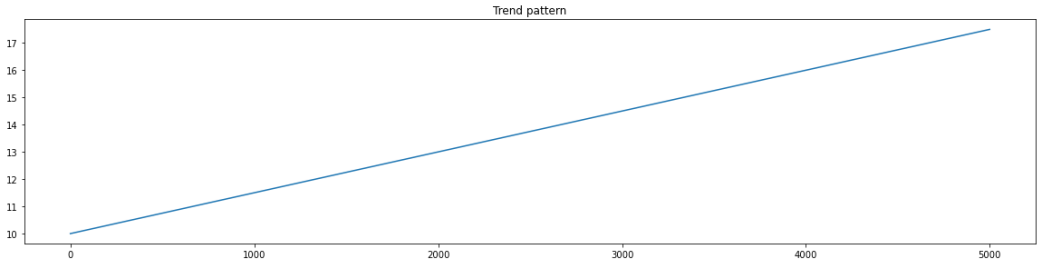
Для нестаціонарного часового ряду ковзне середнє значення $m_y(T)$ процесу є функцією часу T, як показано на рисунку 1.1, б.

Тенденція. У багатьох випадках у світі, коли дані демонструють тенденцію до зростання або зниження з часом, цікаво проаналізувати ці закономірності. Коли ряд містить неявну тенденцію, спостерігається зміна середнього значення з часом.

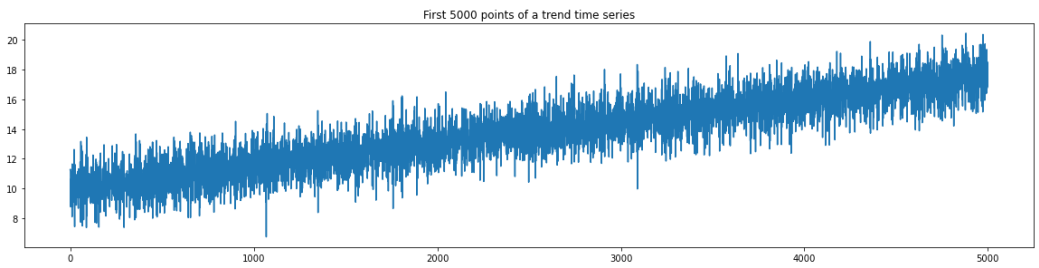
У довгостроковій перспективі, якщо тенденція є передбачуваною, це може дозволити нам охопити основний напрямок даних часового ряду, що призведе до кращого прогнозу на майбутнє. Уявіть собі, хоча стаціонарність дозволяє науковцям з даних включати щоденний (період за періодом) ефект ряду, саме ця тенденція допомагає виявити довгостроковий рух.

Існує кілька типових типів тенденцій, з якими ми можемо зіткнутися на практиці. Найпопулярнішим є лінійний тренд, коли дані, здається, коливаються навколо лінії. Крім того, квадратична тенденція, експоненціальна тенденція іноді може мати місце для часового ряду, де збільшення часових кроків відноситься до більш швидкого та швидшого зростання або зниження спостережуваних значень.

Приклад часового ряду тренду (див. рис. 1.2, б) з лінійним шаблоном тренду (див. рис. 1.2, а) представлено на рисунку 1.2 [14].



а

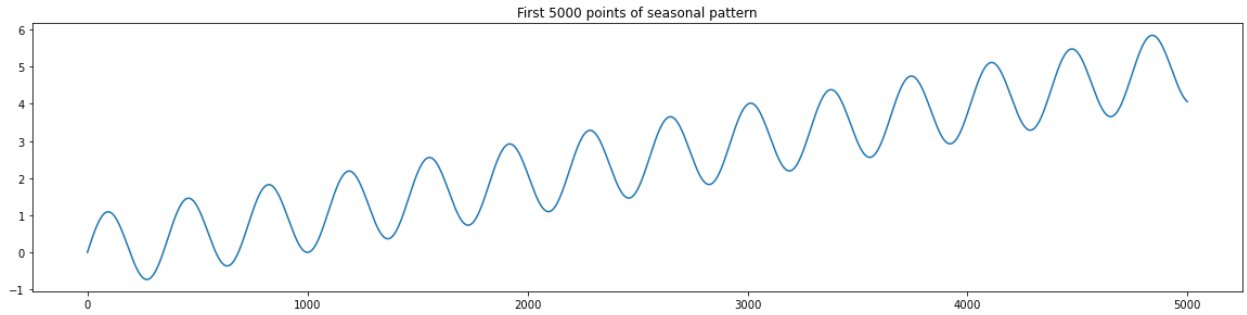


б

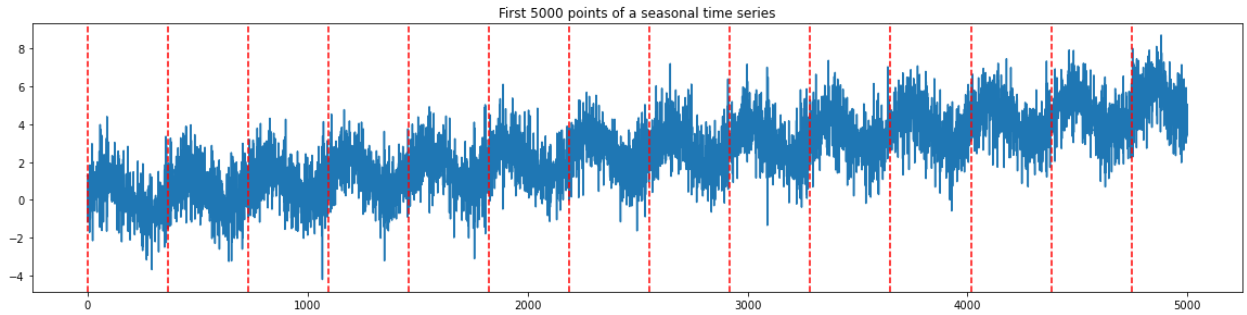
Рисунок 1.2 – Приклад часового ряду зі зростаючою тенденцією

Сезонність. У той час як стаціонарна характеристика спрямована на аналіз щоденних (період за періодом) взаємозв'язків, сезонність фіксує регулярну закономірність у межах інтервалу (зазвичай менше року).

Приклад сезонного часового ряду тренду (див. рис. 1.3, б) з розкладеним сезонним шаблоном (див. рис. 1.3, а) представлено на рисунку 1.3 [14].



а



б

Рисунок 1.3 – Приклад сезонного часового ряду

Сезонність призводить до того, що дані часових рядів змінюються за сезонами, що є ознакою залежності від часу. Отже, сезонний часовий ряд є нестаціонарним.

1.2 Огляд моделей прогнозування

Під прогнозом розуміють науково обґрунтоване міркування про можливий перебіг подій у майбутньому або про альтернативні варіанти

розвитку подій і терміни, в які ці події можуть трапитися (або і те і інше одночасно) [15].

В залежності від мети дослідження, прогнози діляться на пошукові та нормативні. Пошуковий прогноз відповідає на питання: «Чого, вірогідніше за все, чекати в майбутньому?» Нормативний прогноз вказує на можливі шляхи перебігу подій та термін досягнення бажаного результату, тобто розробляється на основі завчасно визначених цілей та задач.

До основних статистичних моделей часових рядів відносять наступні:

1) авторегресійна модель (AR);
2) модель ковзного середнього (MA);
3) авторегресійна модель інтегрованого ковзного середнього (ARIMA);
4) векторна авторегресія (VAR).

Переваги статистичних моделей [1]:

- прості та прозорі, тому їх можна чітко зрозуміти з точки зору вхідних параметрів;
- характеризуються відносно простим математичним представленням часового ряду;
- добре працюють на невеликих масивах даних;
- можуть конкурувати з більш складними моделями машинного навчання.

Недоліки статистичних моделей [1]:

- поступаються складним моделям машинного навчання для дуже великих наборів даних;
- акцентують увагу на точкових оцінках середнього значення розподілу, а не на самому розподілі;
- погано описують дані, де домінують нелінійні зв’язки.

Дамо короткий опис кожної з моделей.

1.2.1 Авторегресійна модель (AR)

Найпростіша модель часового ряду – авторегресійна (Autoregressive Models / AR). Для моделі AR(p) значення ряду в момент часу t є функцією значень ряду в попередні моменти часу [1]:

$$y_t = \phi_0 + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + e_t, \quad (1.4)$$

де t – момент часу, ϕ – коефіцієнт авторегресії, e_t – похибка.

На рисунку 1.4 графічно представлено кореляцію прогнозованих та реальних значень часового ряду (для побудови рисунку використані деякі дані прогнозування попиту, опубліковано в репозитарії машинного навчання UCI).

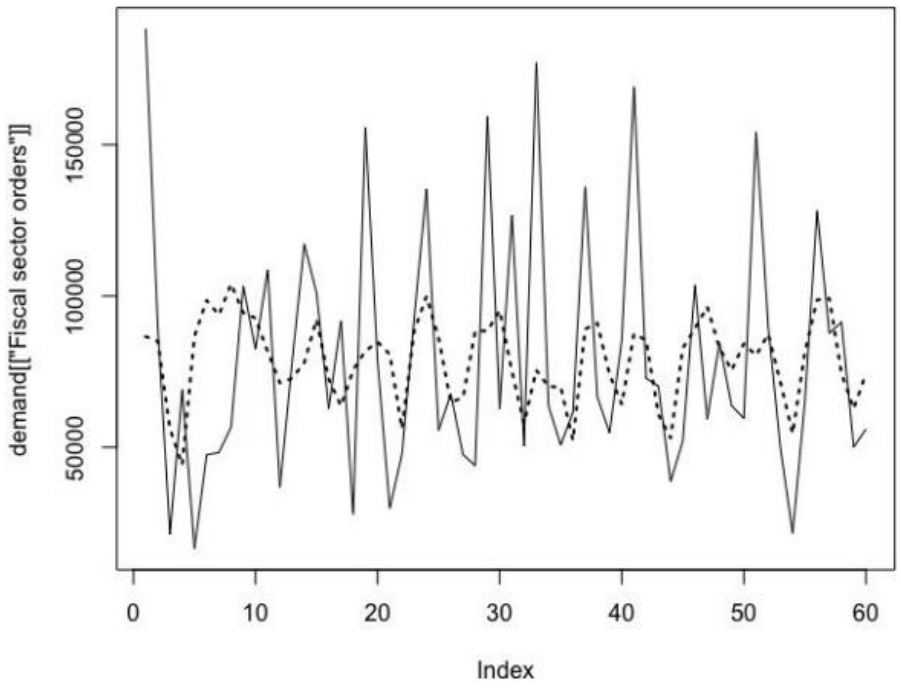


Рисунок 1.4 – Прогнозування з використанням AR моделі: суцільна лінія – вихідний часовий ряд; пунктирна лінія – прогнозований часовий ряд

Прогнозування (див. рис. 1.4) побудовано за допомогою функції fitted() з пакету прогнозів. Програмна реалізація виглядає наступним чином:

```
## R
> require(forecast)
> plot(demand[["Banking orders (2)"]], type = 'l')
> lines(fitted(est.1), col = 3, lwd = 2) ## використовувати пакет прогнозів.
```

1.2.2 Модель ковзного середнього (MA)

Модель ковзного середнього (Moving Average Models / MA) – модель, в якій значення ряду в кожен момент часу є функцією термінів «похибки» недавнього минулого значення, кожен з яких незалежний від інших. Модель порядку q (інша форма запису – MA (q)) описується рівнянням [1]:

$$y_t = \mu + e_t + \theta_1 e_{t-1} + \theta_2 e_{t-1} + \cdots + \theta_q e_{t-q}, \quad (1.5)$$

де q – кількість елементів ковзного середнього.

Програмна реалізація, як і при прогнозуванні AR моделі, будується за допомогою функції fitted().

1.2.3 Авторегресійна інтегрована модель ковзного середнього (ARIMA)

Авторегресійна модель інтегрованого ковзного середнього (Autoregressive Integrated Moving Average Models / ARIMA) – це вдосконалене поєднання моделей AR та MA в ARMA (Auto Regressive – Moving Average). Це означає, що один і той самий часовий ряд може мати як базову динаміку моделі AR, так і MA.

В загальному вигляді модель ARMA(p, q) описується наступним чином [16]:

$$y_t = \alpha_1 y_{t-1} + \cdots + \alpha_p y_{t-p} + e_t + \theta_1 y_{t-1} + \cdots + \theta_q y_{t-q}, \qquad (1.6)$$

де p – порядок авторегресії, q – порядок ковзного середнього.

Якщо процес виявляється нестаціонарним і для приведення до стаціонарного виду доводиться брати декілька різниць, то модель ARMA трансформується в модель ARIMA(p, d, q), де d – порядок різниці.

Модель ARIMA досить ефективна у випадку невеликих наборів даних, де більш складні моделі машинного або глибинного навчання не найкращі.

На мові програмування Python, функції, що підходять для моделі ARIMA, підключаються бібліотекою statsmodels.

1.2.4 Векторна авторегресія (VAR)

Векторна авторегресія (Vector Autoregression / VAR) належить до моделі багатоваріантного часового ряду і є розширенням концепції ARIMA моделювання окремого часового ряду. Загалом моделі цього типу належать до атеоретичних моделей, тобто базуються не на економічній теорії, а на відтворенні динаміки часових рядів, так званій довгостроковій пам'яті ряду даних. Принцип «нехай дані пояснюють самі себе» є основою VAR моделювання [17].

Модель VAR будується за стаціонарними часовими рядами. Це система рівнянь, у якій кожна змінна (компонента багатовимірного часового ряду) представлена у вигляді лінійної комбінації всіх змінних у попередні моменти часу. Порядок такої моделі визначається порядком запізнюваних значень (лагів).

У найпростішому випадку двох часових рядів з одним лагом, модель VAR має вигляд [18]:

$$x_1(t) = \theta_{11} x_1(t-1) + \theta_{12} x_2(t-1),$$

$$x_2(t) = \theta_{21}x_1(t-1) + \theta_{22}x_2(t-1), \quad (1.7)$$

де θ_{ij}, $i,j = 1,2$ – параметри моделі.

У загальному випадку для m тимчасових рядів і k лагів така модель має вигляд системи m рівнянь:

$$\begin{aligned}
&\mathrm{x_1(t)} = \theta_{11}\mathrm{x_1(t-1)} + \cdots + \theta_{1k}\mathrm{x_1(t-k)} + \\
&\quad +\theta_{1,k+1}\mathrm{x_2(t-1)} + \cdots + \theta_{1,2k}\mathrm{x_2(t-k)} + \theta_{1,mk}\mathrm{x_m(t-k)}, \\
&\cdots\cdots\cdots \\
&\mathrm{x_m(t)} = \theta_{m1}\mathrm{x_1(t-1)} + \cdots + \theta_{mk}\mathrm{x_1(t-k)} + \\
&\quad +\theta_{m,k+1}\mathrm{x_2(t-1)} + \cdots + \theta_{m,2k}\mathrm{x_2(t-k)} + \theta_{m,mk}\mathrm{x_m(t-k)}. \quad (1.8)
\end{aligned}$$

1.3 Застосування нейронних мереж

Процес прогнозування часового ряду засобами нейронних мереж можна розбити на три етапи:

1. Підготовка даних.

2. Навчання нейронної мережі (створення моделі).

3. Тестування одержаної моделі.

Для навчання будь-якої нейронної мережі потрібно мати суттєвий об’єм даних (екземплярів) для навчання. Після завершення етапу підготовки даних можна безпосередньо займатись навчанням моделі. Модель будується на основі архітектури нейронної мережі, що використовується при машинному перекладі [19].

Найбільш популярними є такі нейронні мережі, як RNN, CNN, та LSTM. Розглянемо їх та їхні варіації.

Рекурентні нейронні мережі (Recurrent Neural Networks, RNN) – це мережі, що містять зворотні зв’язки і дозволяють зберігати інформацію.

Згорткова нейронна мережа CNN (Convolutional Neural Network – основний інструмент для класифікації та розпізнавання об’єктів, обличь на фотографіях, розпізнавання мови.

LSTM (Long short-term memory, дослівно – довга короткострокова пам’ять) – різновид RNN, здатний навчатися довгостроковим залежностям. Можливість LSTM-мереж успішно вивчати дані з довготривалими залежностями робить їх природним вибором для розв’язання задач, у яких як вхідна, так і вихідна інформація представляються у вигляді послідовностей деяких елементів (наприклад, літер, слів, речень).

Vanilla LSTM – це модель LSTM, яка має один прихований шар одиниць LSTM та вихідний шар, що використовується для передбачення. Ключ до LSTM полягає в тому, що вони пропонують власну підтримку послідовності.

На відміну від CNN, яка зчитує весь вхідний вектор, модель LSTM читає один тимчасовий крок послідовності за раз і створює внутрішнє представлення стану, яке можна використовувати як вивчений контекст для прогнозування.

Приклад визначення моделі Vanilla LSTM виглядатиме наступним чином:

```
# define model
model = Sequential()
model.add(LSTM(50, activation='relu', input_shape=(n_steps, n_features)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse').
```

Кількість часових кроків як вхідних даних – це число, яке вибрається під час підготовки нашого набору даних як аргумент функції split sequence().

Форма вхідних даних для кожного зразка вказується в аргументі input_shape визначення першого прихованого шару. У нас майже завжди є кілька зразків, отже, модель буде очікувати, що вхідний компонент навчальних даних матиме розміри або форму: [samples, timesteps, features].

Функція split_sequence() з формою [samples, timesteps], тому ми легко змінюємо її, щоб мати додатковий вимір для однієї функції. Приклад зміни форми навчання даних для LSTM:

```
# reshape from [samples, timesteps] into [samples, timesteps, features]
n_features = 1
X = X.reshape((X.shape[0], X.shape[1], n_features)).
```

Наведемо скрипт Vanilla LSTM для одновимірного прогнозування часових рядів:

```
# univariate lstm example
from numpy import array
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers import Dense
# split a univariate sequence into samples
def split_sequence(sequence, n_steps):
	X, y = list(), list()
	for i in range(len(sequence)):
		# find the end of this pattern
		end_ix = i + n_steps
		# check if we are beyond the sequence
		if end_ix > len(sequence)-1:
			break
		# gather input and output parts of the pattern
		seq_x, seq_y = sequence[i:end_ix], sequence[end_ix]
		X.append(seq_x)
		y.append(seq_y)
	return array(X), array(y)
```

```
# define input sequence
raw_seq = [10, 20, 30, 40, 50, 60, 70, 80, 90]
# choose a number of time steps
n_steps = 3
# split into samples
X, y = split_sequence(raw_seq, n_steps)
# reshape from [samples, timesteps] into [samples, timesteps, features]
n_features = 1
X = X.reshape((X.shape[0], X.shape[1], n_features))
# define model
model = Sequential()
model.add(LSTM(50, activation='relu', input_shape=(n_steps, n_features)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
# fit model
model.fit(X, y, epochs=200, verbose=0)
# demonstrate prediction
x_input = array([70, 80, 90])
x_input = x_input.reshape((1, n_steps, n_features))
yhat = model.predict(x_input, verbose=0)
print(yhat).
```

Запуск наведеного прикладу готує дані, створює модель і робить прогноз. Можна побачити, що модель передбачає наступне значення в послідовності.

Bidirectional LSTM. У деяких задачах передбачення послідовності, може бути корисно дозволити моделі LSTM навчати вхідну послідовність як вперед, так і назад, або об’єднати обидві інтерпретації. Така модель називається двонаправленою LSTM (Bidirectional LSTM). Ми можемо реалізувати двонаправлену LSTM для одновимірного прогнозування часових рядів

шляхом обгортання першого прихованого шару в шар-оболонку, який називається двонаправленим.

Нижче наведено приклад визначення двонаправленого LSTM для читання вхідних даних як вперед, так і назад:

```
# define model
model = Sequential()
model.add(Bidirectional(LSTM(50, activation='relu'), input_shape=(n_steps, n_features)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse').
```

Наведемо скрипт Bidirectional LSTM для одновимірного прогнозування часових рядів:

```
# univariate bidirectional lstm example
from numpy import array
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers import Dense
from keras.layers import Bidirectional
# split a univariate sequence
def split_sequence(sequence, n_steps):
	X, y = list(), list()
	for i in range(len(sequence)):
		# find the end of this pattern
		end_ix = i + n_steps
		# check if we are beyond the sequence
		if end_ix > len(sequence)-1:
			break
		# gather input and output parts of the pattern
		seq_x, seq_y = sequence[i:end_ix], sequence[end_ix]
		X.append(seq_x)
		y.append(seq_y)
	return array(X), array(y)
# define input sequence
raw_seq = [10, 20, 30, 40, 50, 60, 70, 80, 90]
```

```
# choose a number of time steps
n_steps = 3
# split into samples
X, y = split_sequence(raw_seq, n_steps)
# reshape from [samples, timesteps] into [samples, timesteps, features]
n_features = 1
X = X.reshape((X.shape[0], X.shape[1], n_features))
# define model
model = Sequential()
model.add(Bidirectional(LSTM(50, activation='relu'), input_shape=(n_steps, n_features)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
# fit model
model.fit(X, y, epochs=200, verbose=0)
# demonstrate prediction
x_input = array([70, 80, 90])
x_input = x_input.reshape((1, n_steps, n_features))
yhat = model.predict(x_input, verbose=0)
print(yhat).
```

Запуск прикладу передбачає наступне значення в послідовності, яке, як ми очікуємо, становитиме 100.

Підготовка даних – один з основних етапів прогнозування часових рядів нейронними мережами. Наведемо основні етапи їх підготовки [20]:

1. Import the data. Імпорт даних. Іноді буває так, що сирих даних достатньо для формування більш-менш виразного прогнозу.

2. Aggregate the data. Якщо дані представлені нерівними часовими інтервалами, необхідно їх агрегувати. Наприклад, дані із торгів цінними паперами, валютою та іншими фінансовими інструментами необхідно агрегувати. Зазвичай беруть середнє в інтервалі, але можна і максимальне, мінімальне, стандартне відхилення та інші статистики.

3. Preprocessing the data. Передобробку даних, завдяки якій тимчасовий ряд набуває властивостей гомоскедастичності (через логарифмування даних) і стає стаціонарним (через диференціювання ряду).

4. Split to train, test & forecast. У цьому блоці коду тимчасові ряди розбиваються на періоди навчання, тестування та прогнозування шляхом додавання нового стовпця з відповідними значеннями train, test, forecast.

5. Extraction exogenous time-series features. Буває корисним виділити додаткові зовнішні (екзогенні) ознаки з часового ряду. Наприклад, вказати вихідний це день чи ні, вказати кількість днів на місяці (або кількість робочих днів на місяці) та ін. Як правило, ці ознаки «витягуються» з самого тимчасового ряду без будь-якого ручного втручання.

6. Create/import exogenous data. Не всю інформацію можна «витягнути» з часового ряду. Іноді можуть знадобитися додаткові зовнішні дані. Наприклад, якісь епізодичні події, які мають сильний вплив на значення часового ряду. Такими подіями можуть бути дати початку військових дій, запровадження санкцій, природні катаклізми та ін.

7. Exogenous values. У цьому блоці коду поєднуються всі екзогенні дані в одну таблицю.

8. Union the data (create dataset). У цьому блоці коду поєднуються значення часового ряду та екзогенних ознак в одну таблицю. Іншими словами – готується датасет, на підставі якого буде «навчатись» модель, тестувати якість і формувати прогноз.

9. Learning the model. «Навчання» моделі.

10. Preprocessing data: predict & forecast. У разі, якщо для навчання моделі використовувались передоброблені дані (логарифмовані, оброблені функцією боксу-коксу, стаціонарний ряд та ін.), то якість моделі для початку оцінюється на оброблених даних і тільки потім вже на «сирих» даних. Якщо дані не передоброблялись, то даний етап пропускається.

11. Row data: predict & forecast. Цей етап є заключним. Якщо модель навчалася на передоброблених даних, наприклад, їх прологарифмували, то для отримання прогнозу слід перевести прогноз назад в одиниці вимірювання часового ряду.

2 МЕТОДИ ПІДГОТОВКИ ДАНИХ

2.1 Декомпозиція рядів

Декомпозиція (розкладання) часових рядів передбачає розгляд ряду як комбінації компонентів рівня (середнього значення в серії), тенденції (збільшення або зменшення значення в ряді), сезонності (повторюваний короткостроковий цикл у серії) та шуму (випадкова варіація в серії). При цьому, перші три компоненти ряду – систематичні, а шум – несистематичний компонент.

Адитивна модель часового ряду передбачає, що усі чотири компоненти ряду додаються [2]:

$$y(t) = Level + Trend + Seasonality + Noise. \quad (2.1)$$

Мультиплікативна модель передбачає, що усі компоненти перемножуються:

$$y(t) = Level * Trend * Seasonality * Noise. \quad (2.2)$$

Декомпозувати часовий ряд можна автоматично за допомогою Python. Бібліотека statsmodels забезпечує автоматичну реалізацію класичного методу декомпозиції у функції під назвою seasonal_decompose() . Для цього потрібно вказати, чи є модель адитивною чи мультиплікативною.

Функція seasonal_decompose() повертає об’єкт результату. Об’єкт результату містить масиви для доступу до чотирьох компонентів даних з розкладання.

Наприклад, у фрагменті нижче показано, як розкласти серію на тенденційні, сезонні та залишкові компоненти, припускаючи адитивну модель.

```
from statsmodels.tsa.seasonal import seasonal_decompose
series = ...
result = seasonal_decompose(series, model='additive')
print(result.trend)
print(result.seasonal)
print(result.resid)
print(result.observed).
```

Ці чотири часові ряди можна побудувати безпосередньо з об’єкта результату, викликавши функцію plot(). Наприклад:

```
from statsmodels.tsa.seasonal import seasonal_decompose
from matplotlib import pyplot
series = ...
result = seasonal_decompose(series, model='additive')
result.plot()
pyplot.show().
```

2.2 Метод нормалізації

Нормалізація – це зміна масштабу даних із вихідного діапазону до нового діапазону від 0 до 1.

Нормалізація може бути корисною і навіть необхідною в деяких алгоритмах машинного навчання, коли дані часових рядів мають вхідні значення з різними масштабами. Однак, якщо часовий ряд має тенденцію до

підвищення або зниження, оцінка очікуваних значень може бути ускладнена, і нормалізація може виявитися не найкращим методом для вирішення проблеми.

На мові Python нормалізацію можна реалізувати за допомогою об’єкта перетворення з бібліотеки scikit-learn, зокрема класу MinMaxScaler.

Окрім нормалізації, цей клас також можна використовувати для масштабування даних до будь-якого діапазону, вказавши бажаний діапазон у конструкторі об’єкта.

Наведемо приклад [2]:

```
# example of normalization
from sklearn.preprocessing import MinMaxScaler
from numpy import array
# define dataset
data = [x for x in range(1, 10)]
data = array(data).reshape(len(data), 1)
print(data)
# fit transform
transformer = MinMaxScaler()
transformer.fit(data)
# difference transform
transformed = transformer.transform(data)
print(transformed)
# invert difference
inverted = transformer.inverse_transform(transformed)
print(inverted).
```

2.3 Метод згладжування

Ковзне середнє. Найпростіший метод згладжування рядів – ковзне середнє. Для будь-якої непарної кількості точок послідовності ряду центральна точка замінюється на середнє арифметичне інших точок [21]:

$$s_i = \frac{1}{2k+1} \sum_{j=-k}^{k} x_{i+j}, \tag{2.3}$$

де x_i – вихідний ряд, s_i – згладжений ряд.

Метод ковзного середнього має певні недоліки:

- неефективний у обчисленні, оскільки для кожної точки ряду середнє необхідно перераховувати по новій;
- метод мне можна продовжити на перші та останні точки ряду;
- ковзне середнє не визначено за межами ряду і, як наслідок, не може використовуватися для прогнозування.

Експоненціальне згладжування. Більш універсальним методом згладжування є експоненціальне згладжування. Експоненціальне згладжування – метод прогнозування часових рядів для одновимірних даних.

Існує три варіації даного методу:

- одинарне згладжування (Single Exponential Smoothing / SES) для рядів, у яких немає тренду та сезонності;
- подвійне згладжування (Double Exponential Smoothing) для рядів, які мають тренд, але немає сезонності;
- потрійне згладжування (Triple Exponential Smoothing або Holt-Winters Exponential Smoothing) для рядів, які мають і тренд, і сезонність.

Метод експоненціального згладжування обчислює значення згладженого ряду шляхом оновлення значень, розрахованих на попередньому кроці, використовуючи інформацію з поточного кроку. Інформація з

попереднього та поточного кроків береться з різними вагами, якими можна керувати.

У найпростішому випадку одинарного згладжування, маємо наступне співвідношення [21]:

$$s_i = \alpha x_i + (1 - \alpha) s_{i-1}, \tag{2.4}$$

де $0 \leq \alpha \leq 1$.

Параметр α визначає співвідношення між незгладженим значенням на поточному кроці та згладженим значенням попереднього кроку. При $\alpha = 1$ братимуться лише точки вихідного ряду, тобто ніякого згладжування не буде. При $\alpha = 0$ братимуться лише згладжені значення з попередніх кроків, тобто. ряд перетвориться на константу.

Розглянемо детальніше модель потрійного експоненціального згладжування. Для її реалізації скористаємося бібліотекою statsmodels.

Дана модель має гіперпараметри, від яких залежить характер експоненціального перетворення. Зокрема [2]:

smoothing_level (alpha): коефіцієнт згладжування для рівня;

smoothing_slope (бета): коефіцієнт згладжування для тренду;

smoothing_seasonal (gamma): коефіцієнт згладжування для сезонного компонента;

damping_slope (phi): коефіцієнт для демпфованого тренду.

Усі чотири з цих гіперпараметрів можна вказати при визначенні моделі. Якщо вони не вказані, бібліотека автоматично налаштує модель і знайде оптимальні для них значення (наприклад, optimized=True).

Існують і інші гіперпараметри, які модель автоматично налаштовувати не буде, але їх можна задати самостійно. Це:

trend: тип компонента тренду, наприклад, “add” для адитивного або “mul” для мультиплікативного. Моделювання тренду можна вимкнути, встановивши значення “None”;

demped: чи потрібно демпфувати компонент тренду, чи ні;

seasonal: тип сезонного компонента, наприклад “add” для адитивного або “mul” для мультиплікативного. Моделювання сезонного компонента можна вимкнути, встановивши для нього значення “None”;

seasonal_periods: кількість кроків у часі в сезонному періоді;

use_boxcox: виконувати чи не виконувати перетворення ряду (True/False) чи вказувати лямбда для перетворення.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ

3.1 Дані та методи

Для тестування розробленої системи прогнозування використовувались дані про вартість металів та інших товарів ресурсу finance.yahoo.com. Зокрема, було проаналізовано такі часові ряди: ціни на золото, S&P індекс, ціни на сталь, ціни на паладіум, ціни на алюміній.

В першому розділі розглянуто різні підходи до прогнозування часових рядів. Ми могли б розділити чинні методи на дві категорії: класичні статистичні методи (наприклад, ARIMA, ARCH тощо) та методи на основі штучних нейронних мереж (ШНМ).

Підготовка даних для навчання нейронної мережі зазвичай включає такі кроки:

– нормалізація даних;

– згладжування даних;

– перерахунок цін з урахуванням інфляції;

– формування тренувальної та тестової вибірки.

Початкова модель включає нейронну мережу зворотного зв’язку для однофакторного прогнозування часових рядів з використанням однокрокового прогнозування.

Однофакторне прогнозування означає, що ми використовуємо лише один часовий ряд, зокрема ціну на метал.

При використанні однокрокового прогнозу зазвичай прогнозується одне значення, яке потім в свою чергу використовується для прогнозу наступного значення.

Наприклад, на рисунку 3.1 наведено оригінальні ціни на золото (див. рис. 3.1) з частотою оновлення даних – рік.

	Date	Price
0	1950-12	34.720
1	1951-12	34.660
2	1952-12	34.790
3	1953-12	34.850
4	1954-12	35.040
...	...	...
65	2015-12	1068.317
66	2016-12	1152.165
67	2017-12	1265.674
68	2018-12	1249.887
69	2019-12	1480.025

Рисунок 3.1 – Ціни на золото та описові статистики

Дані після нормалізації (див. рис. 3.2).

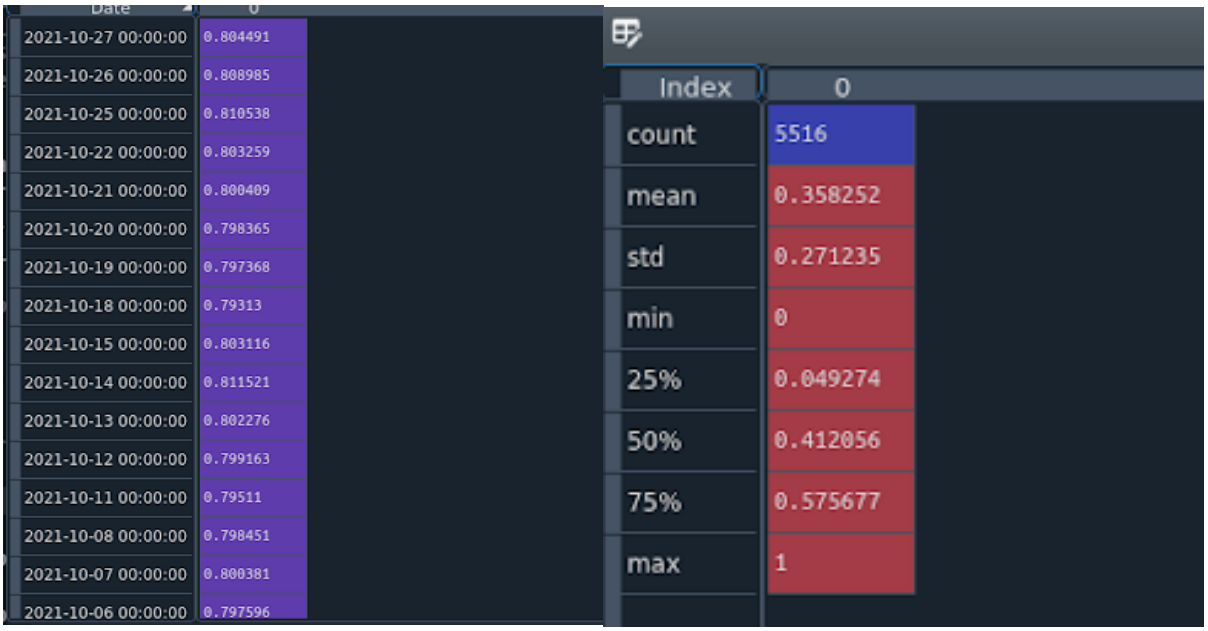


Рисунок 3.2 – Дані після нормалізації

Методи згладжування – це методи видалення шуму з набору даних (див. розділ 2). Це дозволяє виділити важливі шаблони та залежності в даних. У роботі використано такі підходи:

– Moving average smoothing (MAS);

– Exponential smoothing (ES);

– Double exponential smoothing (DES);

– Triple exponential smoothing (TES).

Наприклад, оригінальні дані зображено на рисунку 3.3.

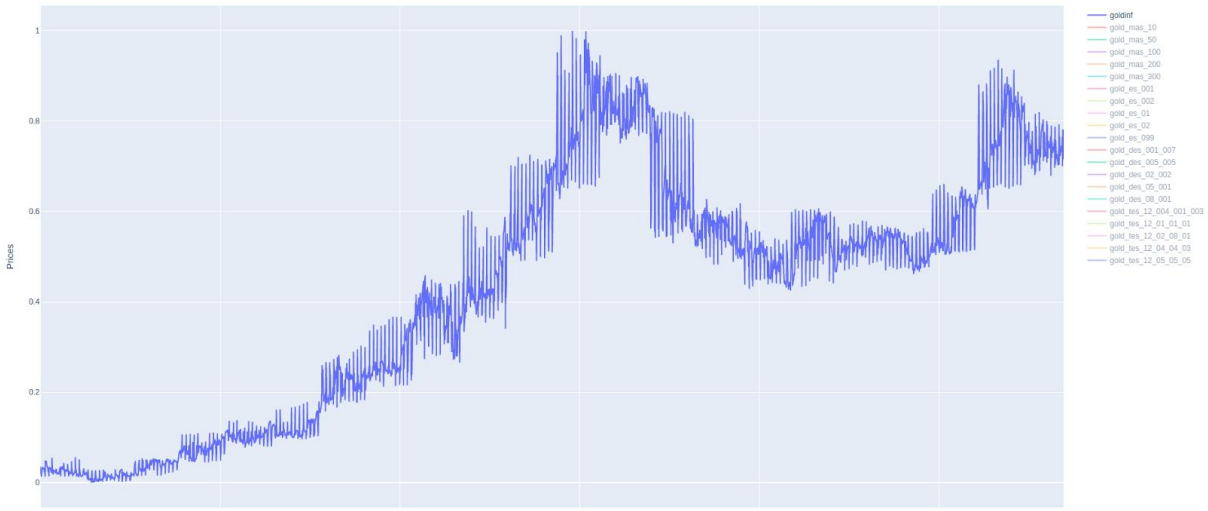


Рисунок 3.3 – Початкові дані

Дані після застосування методів згладжування (див. рис. 3.4).

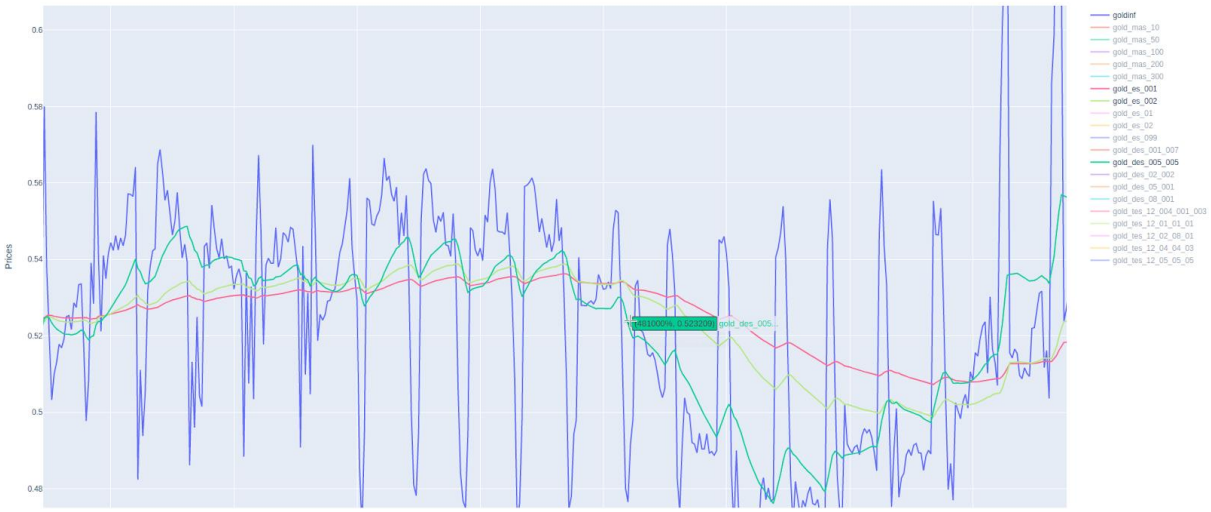


Рисунок 3.4 – Дані після згладжування

Можна зробити висновок, що для подальшого аналізу найбільш прийнятними є такі методи: ES_001, ES_002 та DES_005_005. Оскільки ці

методи з наведеними параметрами дозволяють згладити коливання ринку та перейти до застосування прогнозних моделей.

Ще одним важливим етапом попередньої обробки даних є декомпозиція часових рядів (див. підрозділ 2.1). На рисунку 3.5 зображено декомпозицію на трендову та циклічну компоненту, а також виділено часовий ряд без цих компонент. На практиці далі можна застосовувати різні підходи. Наприклад, будувати прогнозні моделі для наближення тренду, циклів та ряду без цих компонентів, а потім комбінувати відповідь як суму цих прогнозів.

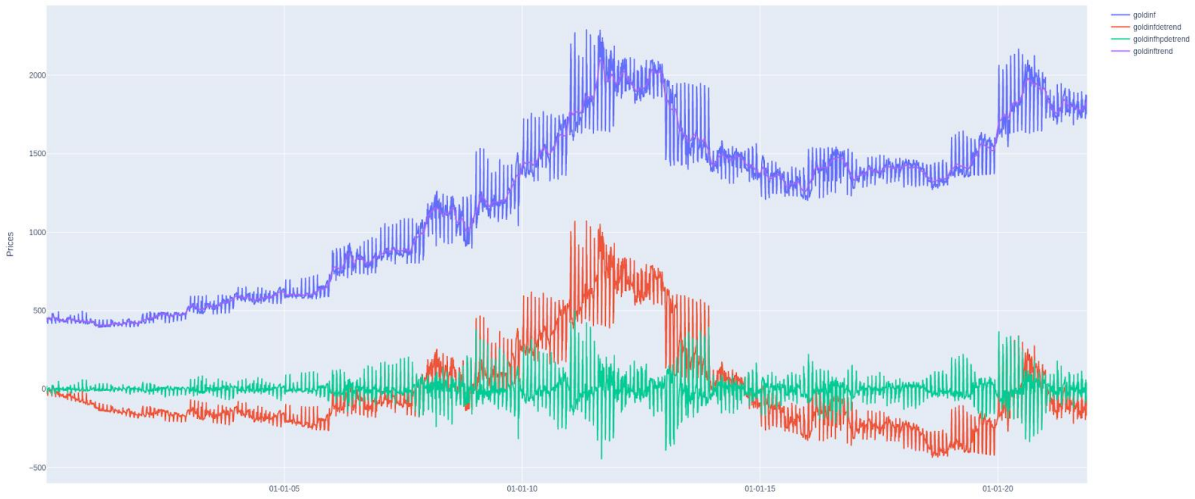


Рисунок 3.5 – Декомпозиція та трендову та циклічну компоненти

Наступним важливим кроком для застосування нейронних мереж є формування вибірки даних для тренування та тестування системи для задачі навчання з вчителем. Тут можуть бути відмінності в залежності від типу моделей: одно та багато- етапне прогнозування, застосування екзогенних даних. На рисунку 3.6 зображено поділ даних у випадку однокрового прогнозування з кроком історії 7 років.

Date	Price	shift_0	shift_1	shift_2	shift_3	shift_4	shift_5	shift_6	shift_7	shift_8	shift_9
1950-12-01	34.72	34.72	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1951-12-01	34.66	34.66	34.72	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1952-12-01	34.79	34.79	34.66	34.72	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1953-12-01	34.85	34.85	34.79	34.66	34.72	NaN	NaN	NaN	NaN	NaN	NaN
1954-12-01	35.04	35.04	34.85	34.79	34.66	34.72	NaN	NaN	NaN	NaN	NaN
1955-12-01	34.97	34.97	35.04	34.85	34.79	34.66	34.72	NaN	NaN	NaN	NaN
1956-12-01	34.90	34.90	34.97	35.04	34.85	34.79	34.66	34.72	NaN	NaN	NaN
1957-12-01	34.99	34.99	34.90	34.97	35.04	34.85	34.79	34.66	34.72	NaN	NaN
1958-12-01	35.09	35.09	34.99	34.90	34.97	35.04	34.85	34.79	34.66	34.72	NaN
1959-12-01	35.05	35.05	35.09	34.99	34.90	34.97	35.04	34.85	34.79	34.66	34.72

Рисунок 3.6 – Підготовка часових рядів для навчання з вчителем

Рисунки 3.7 та 3.8 зображують аналогічний процес при врахуванні екзогенних даних для багатоетапного прогнозування.

Date	shift_1	shift_2	shift_3	shift_4	shift_5	shift_6	shift_7
1959-12-01	35.090	34.990	34.900	34.970	35.040	34.850	34.790
1960-12-01	35.050	35.090	34.990	34.900	34.970	35.040	34.850
1961-12-01	35.540	35.050	35.090	34.990	34.900	34.970	35.040
1962-12-01	35.150	35.540	35.050	35.090	34.990	34.900	34.970
1963-12-01	35.080	35.150	35.540	35.050	35.090	34.990	34.900
1964-12-01	35.080	35.080	35.150	35.540	35.050	35.090	34.990
1965-12-01	35.120	35.080	35.080	35.150	35.540	35.050	35.090
1966-12-01	35.130	35.120	35.080	35.080	35.150	35.540	35.050
1967-12-01	35.180	35.130	35.120	35.080	35.080	35.150	35.540
1968-12-01	35.190	35.180	35.130	35.120	35.080	35.080	35.150
1969-12-01	41.113	35.190	35.180	35.130	35.120	35.080	35.080
1970-12-01	35.189	41.113	35.190	35.180	35.130	35.120	35.080
1971-12-01	37.434	35.189	41.113	35.190	35.180	35.130	35.120
1972-12-01	43.455	37.434	35.189	41.113	35.190	35.180	35.130
1973-12-01	63.779	43.455	37.434	35.189	41.113	35.190	35.180
1974-12-01	106.236	63.779	43.455	37.434	35.189	41.113	35.190
1975-12-01	183.683	106.236	63.779	43.455	37.434	35.189	41.113
1976-12-01	139.279	183.683	106.236	63.779	43.455	37.434	35.189
1977-12-01	133.674	139.279	183.683	106.236	63.779	43.455	37.434
1978-12-01	160.480	133.674	139.279	183.683	106.236	63.779	43.455
1979-12-01	207.895	160.480	133.674	139.279	183.683	106.236	63.779
1980-12-01	463.666	207.895	160.480	133.674	139.279	183.683	106.236
1981-12-01	596.712	463.666	207.895	160.480	133.674	139.279	183.683
1982-12-01	410.119	596.712	463.666	207.895	160.480	133.674	139.279
1983-12-01	444.776	410.119	596.712	463.666	207.895	160.480	133.674
1984-12-01	388.060	444.776	410.119	596.712	463.666	207.895	160.480

Рисунок 3.7 – Вхідні дані тренування (TrainX)

Date	Price
1959-12-01	35.050
1960-12-01	35.540
1961-12-01	35.150
1962-12-01	35.080
1963-12-01	35.080
1964-12-01	35.120
1965-12-01	35.130
1966-12-01	35.180
1967-12-01	35.190
1968-12-01	41.113
1969-12-01	35.189
1970-12-01	37.434
1971-12-01	43.455
1972-12-01	63.779
1973-12-01	106.236
1974-12-01	183.683
1975-12-01	139.279
1976-12-01	133.674
1977-12-01	160.480
1978-12-01	207.895
1979-12-01	463.666
1980-12-01	596.712
1981-12-01	410.119

Рисунок 3.8 – Вихідні дані тренування (TrainY)

3.2 Реалізація моделей

Огляд пов’язаних публікацій показує, що методи на основі нейронних мереж є досить перспективними для нелінійного прогнозування часових рядів. Зокрема, архітектура довгострокової пам’яті (LSTM) у поєднанні зі згортковою нейронною мережею використовується для прогнозування цін на золото. Хорошим прикладом є стаття під авторством I. E. Livieris, E. Pintelas, та P. A. Pintelas [22]. Згорткові шари використовуються для вилучення шаблону із значень історії часових рядів. Шари LSTM використовуються для прогнозування з використанням виділених ознак із згорткових шарів (див. рис. 3.1).

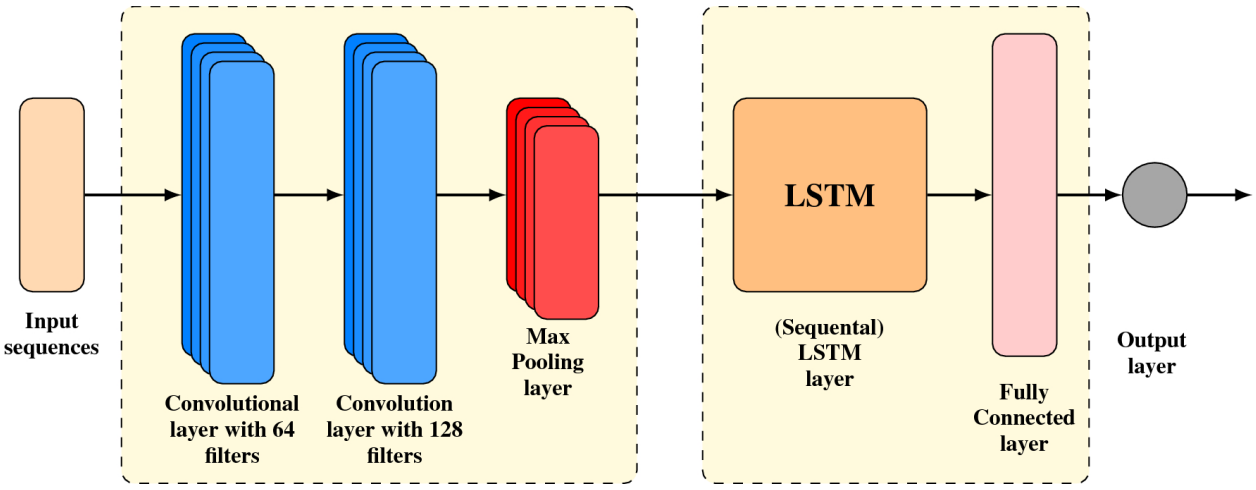


Рисунок 3.9 – CNN LSTM архітектура [22]

Також ця мережа демонструє задовільні результати щодо цін на золото (з січня 2014 року по квітень 2018 року, отримані з http://finance.yahoo.com).

Отже, окрім стандартних архітектур нейромереж для прогнозування часових рядів у нашій системі реалізовано модель Conv + LSTM.

Згідно з іншими роботами з аналізу фінансових даних, існує кілька додаткових екзогенних параметрів, які можна використовувати з історичними цінами [23]:

– ціна на нафту;

– індекс долара США;

– індекс Доу Джонса;

– індекс Standard&Poor`s 500.

Отже, для тестування системи також використані зазначені та інше часові ряди.

З іншої точки зору, ШНМ мають деякі недоліки, а саме:

– вимоги до великої кількості даних для навчання мережі;

– традиційна нейронна мережа - це чорний ящик. Це означає, що ми не можемо виділити правила, якими керується нейронна мережа при прогнозуванні;

– потрібна додаткова настройка гіперпараметрів нейронної мережі (кількість шарів, тип шарів, кількість нейронів тощо).

Водночас існують методики усунення кожного з описаних вище недоліків (метанавчання, нейронечіткі мережі, нейроеволюція). Реалізація цих методів може стати предметом подальших досліджень.

Функція втрат є вирішальним параметром для будь-яких моделей прогнозування, оскільки ця функція показує, наскільки прогнозовані результати відрізняються від основної істини. А мета прогнозних моделей – мінімізувати величину втрат.

Отже, для прогнозування часових рядів важливими критеріями є форма та зсув у часі прогнозованих значень. Функція розширених втрат є однією з втрат, придатних для прогнозування часових рядів. Оригінальна реалізація паперу та PyTorch доступна тут: [https://github.com/vincent-leguen/DILATE].

Ми впровадили PyTorch Feed Forward Network з DILATE і перевіримо це в наступних підрозділах.

Основна ідея втрати DILATE (Втрата викривлення, включаючи форму і час) для часових рядів полягає в тому, щоб забезпечити функцію втрат, яка враховує форму та час. Наприклад, випадки на рисунку 3.10 ілюструють різницю між стандартними втратами MSE (середня квадратична помилка) і DILATE.

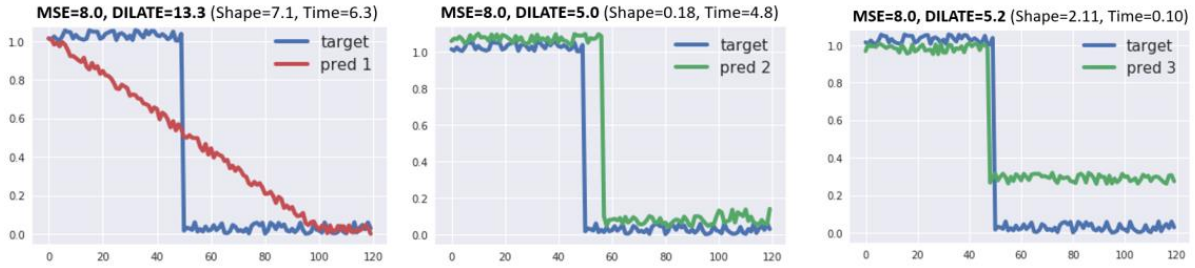


Рисунок 3.10 – MSE та DILATE метрики похибки прогнозування [24]

Отже, застосування наведеної функції втрат DILATE може бути перспективним саме при реалізації систем прогнозування часових рядів.

В системі реалізовано ряд нейромережевих моделей, серед яких: мережа прямого поширення сигналу, LSTM модель, CNN модель, комбінована CNN–LSTM модель. Наприклад, на рисунку 3.11 зображено гібридну архітектуру.

```
n_steps = 4
# split into samples
X, y = split_sequence(raw_seq, n_steps)
# reshape from [samples, timesteps] into [samples, subsequences, timesteps, features]
n_features = 1
n_seq = 2
n_steps = 2
X = X.reshape((X.shape[0], n_seq, n_steps, n_features))
# define model
model = Sequential()
model.add(TimeDistributed(Conv1D(filters=64, kernel_size=1, activation='relu'),
    input_shape=(None, n_steps, n_features)))
model.add(TimeDistributed(MaxPooling1D(pool_size=2)))
model.add(TimeDistributed(Flatten()))
model.add(LSTM(50, activation='relu'))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
# fit model
model.fit(X, y, epochs=500, verbose=0)
```

Рисунок 3.11 – Комбінована CNN–LSTM модель

Наведений варіант є реалізацією архітектури, запропонованої в роботі [22].

3.3 Приклади роботи системи

Далі наведено декілька прикладів прогнозів, що отримано системою для різних часових рядів.

Наприклад, прогнозування цін на золото із використанням S&P біржового індексу зображено на рисунку 3.12.

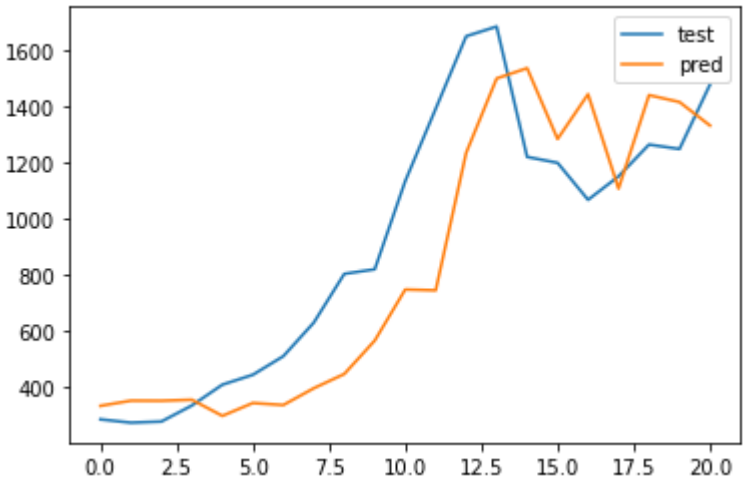


Рисунок 3.12 – Приклад прогнозу

Слід зазначити, що розроблена система прогнозування цін на товари має такі гіперпараметри:

– кількість кроків для формування навчальної вибірки з історії;

– значення горизонту прогнозування, скільки кроків передбачить система;

– тип моделі;

– структура моделі;

– параметри моделі: ініціалізатори, оптимізатори, активації, втрати;

– для прогнозування використовуються специфічні екзогенні часові ряди;

– метод згладжування та його параметри;

– ансамблевий метод та його параметри.

Отже, застосування розробленої системи на практиці потребує налаштування всіх наведених гіперпараметрів, що не є тривіальною задачею.

ВИСНОВКИ

Застосування нейронних мереж в різних галузях IT сфери дозволяє істотно підвищити якість моделювання та дослідження різноманітних процесів. Розробка відповідного програмного забезпечення є вельми актуальною та своєчасною задачею, яка передбачає володіння широким спектром компетентностей – від аналізу даних та математичної статистики до сучасних інформаційних технологій.

Таким чином, побудова нейронних мереж дозволяє підвищити якість моделювання та прогнозування часових рядів.

Для досягнення поставленої мети було проаналізувати існуючі моделі прогнозування часових рядів. Серед класичних моделей особливої уваги заслуговує авторегресійна інтегрована модель ковзного середнього ARIMA. Серед моделей нейронних мереж – модель LSTM.

В результаті роботи було розроблено веб застосунок для прогнозування багатоваріантних часових рядів. Система реалізована засобами бібліотека Python скриптів, для роботи зі сторонніми ресурсами реалізовано Rest API засобами Python та Flask для доступу до основних функцій.

В результаті тестування розробленої системи, проведено прогнозування цін на золото із використанням S&P біржового індексу.

ПЕРЕЛІК ПОСИЛАНЬ

1. Nielsen A. Practical Time Series Analysis. Prediction with Statistics and Machine Learning, 2019. 500 p.
2. Brownlee J. Deep Learning for Time Series Forecasting. Predict the Future with MLPs, CNNs and LSTMs in Python, 2019. 572 p.
3. Vishwas B. V., Patel A. Hands-on Time Series Analysis with Python. From Basics to Bleeding Edge Techniques, 2020. 420 p. doi: 10.1007/978-1-4842-5992-4.
4. Chakraborty K., Mehrotra K., Mohan C. K., Ranka S. Forecasting the behavior of multivariate time series using neural networks // *Neural Networks*. 1992. Vol. 5(6). P. 961–970. doi: 10.1016/s0893-6080(05)80092-9.
5. Du Preez J., Witt S. F. Univariate versus multivariate time series forecasting: an application to international tourism demand // *International Journal of Forecasting*. 2003. Vol. 19(3). P. 435–451. doi: 10.1016/s0169-2070(02)00057-2.
6. Shih S.-Y., Sun F.-K., Lee H. Temporal pattern attention for multivariate time series forecasting // *Machine Learning*. 2019. Vol. 108. P. 1421–1441. doi: 10.1007/s10994-019-05815-0.
7. Wan R., Mei S., Wang J., Liu M., Yang F. Multivariate Temporal Convolutional Network: A Deep Neural Networks Approach for Multivariate Time Series Forecasting // *Electronics*. 2019. Vol. 8(8) 876. P. 1–18. doi: 10.3390/electronics8080876.
8. Jones S. S., Evans R. S., Allen T. L., Thomas A., Haug P. J., Welch S. J., Snow G. L. A multivariate time series approach to modeling and forecasting demand in the emergency department // *Journal of Biomedical Informatics*. 2009. Vol. 42(1). P. 123–139. doi: 10.1016/j.jbi.2008.05.003.
9. Yamak P. T., Yujian L., Gadosey P. K. A Comparison between ARIMA, LSTM, and GRU for Time Series Forecasting. *Proceedings of the 2019 2nd*

International Conference on Algorithms, Computing and Artificial Intelligence (Sanya, China, December 20–22, 2019). Sanya, 2019. P. 49–55. doi: 10.1145/3377713.3377722.

10. Shahid F., Zameer A., & Muneeb M. Predictions for COVID-19 with deep learning models of LSTM, GRU and Bi-LSTM // *Chaos, Solitons & Fractals*. 2020. Vol. 140. P. 110212. doi: 10.1016/j.chaos.2020.110212.
11. Афанасьев В. Н., Юзбашев М. М. Анализ временных рядов и прогнозирование. М.: Финансы и статистика, 2010. 320 с.
12. Ставицький А. В. Теорія часових рядів. URL: http://andriystav.cc.ua/Downloads/AppliedEco/02_Time_Series.pdf (дата звернення: 08.09.2021).
13. Часові ряди та їх основні характеристики. URL: https://web.posibnyky.vntu.edu.ua/feeem/1mokin_matmetody_identifikaciyi_dinamsystem/5-1.html (дата звернення: 08.09.2021).
14. Nguyen N. Time series data characteristics. URL: https://medium.com/@namnguyenthe/time-series-data-characteristics-994e43c470c6 (дата звернення: 27.09.2021).
15. Сажин Ю. В., Катынь А. В., Сарайкин Ю. В. Анализ временных рядов и прогнозирование. Саранск: Изд-во Мордов, 2013. 192 с.
16. Доронина А. И. Модели временного ряда: AR(p), MA(q), ARIMA(p,d,q). Пример исследования потребления нефтепродуктов во Франции. *Студенческий научный форум – 2014*: Материалы VI Международной студенческой электронной научной конференции. (Москва, 15 февр.–31 марта 2014). Москва, 2014. URL: https://scienceforum.ru/2014/article/2014001560 (дата звернення: 24.09.2021).
17. Лук’яненко І. Г., Жук В. М. Аналіз часових рядів. Ч. 2: Побудова VAR і VECM моделей з використанням пакета E.Views 6.0. К.: НаУКМА; Аграр Медіа Груп, 2013. 174 с.

18. Ефименко С. Н. Построение систем прогнозных моделей многомерных взаимосвязанных процессов // *УСиМ*. 2016. № 4. С. 80–85. URL: http://usim.org.ua/arch/2016/4/10.pdf (дата звернення: 24.09.2021).
19. Шеремет О. І., Запорожець В. С. Застосування рекурентних нейронних мереж для виконання машинного рерайту // *Научный вестник ДГМА*. 2018. № 1(25Е). С. 62–68.
20. Петренко А. Временные ряды. Простые решения. URL: https://habr.com/ru/post/553658/ (дата звернення: 05.10.2021).
21. Методы анализа временных рядов: сглаживание. URL: https://asu-analitika.ru/metody-analiza-vremennyh-rjadov-sglazhivanie/ (дата звернення: 21.10.2021).
22. Livieris I. E., Pintelas E., Pintelas P.A. CNN-LSTM model for gold price time-series forecasting // *Neural Comput & Applic*. 2020. Vol. 32. P. 17351–17360. doi: 10.1007/s00521-020-04867-x.
23. Liu D., Li Z. Gold Price Forecasting and Related Influence Factors Analysis Based on Random Forest // *Proceedings of the Tenth International Conference on Management Science and Engineering Management*. 2016. Vol. 59. P. 711–723. doi: 10.1007/978-981-10-1837-4_59.
24. Le Guen V., Thome N. DILATE: DIstortion Loss with shApe and tImE. URL: https://github.com/vincent-leguen/DILATE (дата звернення: 14.10.2021).