НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

«До захисту допущено»
Завідувач кафедри
_____ О.Л. ТИМОЩУК
07 червня 2021 р.

Дипломна робота

на здобуття ступеня бакалавра
за освітньо-професійною програмою «Системний аналіз і управління»
спеціальності 124 "Системний аналіз"
на тему: «Моделювання нелінійних нестаціонарних процесів в економіці»

Виконав:
Студент IV курсу, групи КА-77
Жук Володимир Миколайович _____

Керівник:
старший викладач кафедри ММСА
к.т.н., Селін Юрій Миколайович _____

Консультант з економічного розділу:
доцент кафедри ТПЕ
к.е.н., доцент Надія Василівна Рощіна _____

Консультант з нормоконтролю:
доцент кафедри ММСА
к.т.н., доцент Анатолій Єпіфанович Коваленко _____

Рецензент:
Професор кафедри АУТС ФІОТ,
д. т. н., доцент Корнієнко Богдан Ярославович _____

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших авторів
без відповідних посилань.
Студент _____

Київ–2021 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут прикладного системного аналізу

Кафедра математичних методів системного аналізу

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 124 "Системний аналіз"

Освітньо-професійна програма «Системний аналіз і управління»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ О.Л. Тимощук
«___»_____20__ р.

ЗАВДАННЯ

на дипломну роботу студенту

Жуку Володимиру Миколайовичу

1. Тема роботи «Моделювання нелінійних нестаціонарних процесів в економіці», керівник роботи Селін Юрій Миколайович, к.т.н., старший викладач, затверджені наказом по університету від 26 травня 2021 р. № 1344-с.

2. Термін подання студентом роботи: 07.06.2021 р.

3. Вихідні дані до роботи: Дані про ціни акції компанії NETFLIX з 2019 року по 2021 рік.

4. Зміст роботи: Огляд та моделювання економічних процесів різними засобами. Порівняння лінгвістичного методу моделювання з авторегресійними моделями.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо) : презентація

6. Консультанти розділів роботи

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

Економічний	Рощіна Н.В., доцент кафедри ТПЕ		

7. Дата видачі завдання: 24.02.2021.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Формулювання тематики (напряму) дослідження.	03.09.2020 – 30.09.2020	виконано
2	Аналіз актуальності задач стосовно тематики дослідження	01.10.2020 – 30.10.2020	виконано
3	Аналіз відомих результатів стосовно тематики дослідження	01.11.2020 – 30.11.2020	виконано
4	Формулювання задач дослідження	25.02.2021	виконано
5	Уточнення теми дипломної роботи	25.02.2021	виконано
6	Збір статичних даних, попередній аналіз даних	01.03.2021 – 30.03.2021	виконано
7	Розробка програмного продукту для виконання обчислювальних експериментів	01.03.2021 – 30.04.2021	виконано
8	Виконання обчислювальних експериментів, аналіз та оформлення результатів	01.05.2021 – 20.05.2021	виконано
9	Оформлення пояснювальної записки у цілому	21.05.2021 – 31.05.2021	виконано
10	Підготовка презентації для захисту	01.06.2021	виконано
11	Попередній захист дипломної роботи	02.06.2021	виконано
12	Захист дипломної роботи	17.06.2021	виконано

Студент Володимир Жук

Керівник Юрій Селін

РЕФЕРАТ

Дипломна робота: 113 с., 35 рис., 8 табл., 2 додатки, 21 джерело.

ЛІНГВІСТИЧНЕ МОДЕЛЮВАННЯ, МАШИННЕ НАВЧАННЯ, PYTHON, ПРОГНОЗУВАННЯ ЧАСОВИХ РЯДІВ

Тема бакалаврської роботи «Моделювання нелінійних нестаціонарних процесів в екноміці».

Велика кількість часових даних різних процесів за своїм походження та плином є неламінарними, тобто основні статистичні дані не є постійними у часі. Створення та розробка методів прогнозування рядів будь-якого походження без необхідної попередньої обробки та з достатньою якістю прогнозу є необхідною умовою для розуміння багатьох процесів, в тому числі і економічних, що дозволяє досить точно визначати причини та плин процесу, виявляти та прогнозувати майбутню поведінку, усувати існуючі проблеми тощо.

Об’єкт дослідження – статистичні дані, графік часового ряду акції компанії NETFLIX.

Предмет дослідження – методи прогнозування часових рядів, лінгвістичне моделювання.

Метою бакалаврської роботи є реалізація методу лінгвістичного моделювання та порівняння його роботи з існуючими методами прогнозу економічних часових рядів.

Результати роботи – розроблення програмного продукту, який містить в собі реалізацію лінгвістичного моделювання.

Новизна роботи полягає у розробці та програмній реалізації алгоритму лінгвістичного моделювання та порівняння його з існуючими методами прогнозування.

ABSTRACT

The work consists of 113 p., 35 fig., 8 tables, 2 append 21 sources. LINGUISTIC MODELING, MACHINE LEARNING, PYTHON, TIME SERIES FORECASTING

The topic of the bachelor`s work is "Modeling of nonlinear nonstationary processes in economics".

A large number of time series data of different processes are non-laminar, so the main statistical parameters are not constant in time. Creation and development of methods for forecasting series of any processes without the necessary data pre-processing and with sufficient forecast quality is a necessary condition for understanding many time series processes, including economic, which allows to accurately determine the causes and understanding of the process, identify and predict future behavior, eliminate existing problems, etc.

Object of research – statistical data, such as time series of stock prices of NETFLIX company.

The subject of research - methods of time series forecasting, linguistic modeling.

The aim of the bachelor's dissertation is to create and develop linguistic modeling method and to compare the results with existing models, which are used to predict economical time series data.

The results of the work - the development of an application system for the analysis of budget data using a wide range of SAS tools.

The novelty of the work - after data processing, the user will be able to track the positive and negative factors affecting the budget of the local community, to consider the prospects for cost optimization.

ЗМІСТ

РОЗДІЛ 1 ІСТОРІЯ ТА ОСОБЛИВОСТІ РОЗВИТКУ ЕКОНОМІЧНИХ ПРОЦЕСІВ СУЧАСНОСТІ ТА ЇХ МОДЕЛЮВАННЯ 10
1.1 Характеристика розвитку економічних процесів в Україні та світі 10
1.2 Способи моделювання нелінійних нестаціонарних процесів 12
1.3 Програмні продукти для аналізу і моделювання нелінійних процесів .. 19
1.4 Постановка задачі і висновки до розділу 20
РОЗДІЛ 2 ОГЛЯД ТА ХАРАКТЕРИСТИКА СТРУКТУР МАТЕМАТИЧНИХ МОДЕЛЕЙ ПРОГНОЗУВАННЯ І ЇХ ВИКОРИСТАННЯ 22
2.1 Статистичні тести для аналізу процесів 22
2.1.1 Тести на стаціонарність 22
2.1.2 Тести на нелінійність 24
2.1.3 Тести на гетероскедастичність 27
2.2 Модель ARIMA 28
2.3 Лінгвістичне моделювання 31
2.4 Метод Хольта-Вінтерса 34
2.5 Моделі умовної гетероскедастичності 36
2.6 Критерії для оцінок адекватності побудованих моделей та оцінок якості прогнозів 40
2.6.1 Критерії адекватності моделей 40
2.6.2 Критерії оцінки якості прогнозів 43
2.7 Висновки до розділу 44
РОЗДІЛ 3. ПОБУДОВА МОДЕЛЕЙ ТА ОЦІНКА ПРОГНОЗІВ ВИБРАНИХ ЕКОНОМІЧНИХ ПРОЦЕСІВ 46
3.1 Обрання прогромного середовища 46
3.2 Функціональна схема розробленого програмного продукту 46
3.3 Побудова моделей та оцінювання прогнозів вибраних процесів 49
3.3.1 Побудова моделей та прогнозів вартості акцій компанії NETFLIX . 49

3.4 Аналіз та порівняння отриманих результатів 66
3.4.1 Порівняння побудованих моделей часового рядку NETFLIX..... 66
3.5 Висновки до розділу 67
РОЗДІЛ 4. ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ 68
4.1 Постановка завдання техніко-економічного дослідження..... 68
4.2 Обґрунтування функцій та параметрів програмного продукту 68
4.3 Обґрунтування системи параметрів 71
4.4 Визначення коефіцієнтів значимості параметрів..... 73
4.5 Оцінка рівня якості варіантів реалізації програмного продукту..... 76
4.6 Економічний аналіз варіантів програмного продукту 76
4.7 Вибір кращого варіанту програмного продукту техніко-економічного рівня. 81
4.8 Висновки до розділу..... 81
ВИСНОВКИ..... 83
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 84
ДОДАТОК А. ЛІСТИНГ ПРОГРАМИ 86
ДОДАТОК Б. ІЛЮСТРАТИВНИЙ МАТЕРІАЛ..... 102

ВСТУП

В сьогоденні вивчення та аналіз економічних процесів – рушій розвитку багатьох країн та народів. Увесь світ давно став транснаціональним . Торгівля товарами виробництва займає всю географію земної кулі. Розвиток промисловості, високих технологій, зміна структури виробництва – все це впливає на стан суб'єктів економічних відносин. Багато аналітиків та вчених розглядають та намагаються зрозуміти сутність окремих економічних процесів для прогнозування та коригування складових показників не тільки суб'єкта торгівлі, а і цілих країн. Ці процеси зазвичай мають нелінійну структуру, оскільки на їх плин можуть впливати як внутрішні, так і зовнішні фактори. Існують цілі наукові інститути та підрозділи для розробки прогнозів та вивчення поведінки економічних процесів. Майже всі процеси які зустрічаються у економіці мають нелінійну структуру, як от ціни на ріст акцій, чи динаміка розвитку ВВП країни.

Апарат аналізу часових рядів широко застосовується для вивчення нелінійних процесів різної природи, а вже досліджена теорія застосування та розвиток технологій дозволяє ефективно будувати математичні моделі високої точності, які описують майже будь-який процес різних явищ виникнення. Часовий ряд – набір даних будь-якого процесу, отриманий за допомогою спостереження у рівновіддалені проміжки часу. Цей спосіб фіксації даних дозволяє ефективно отримувати та застосовувати елементи статистичного, функціонального аналізу для вивчення причин поведінки процесу та побудови математичної моделі. Кожний процес можна описати та приблизити його математичною моделлю, яка будується на основі попереднього аналізу даних на багато факторів : стаціонарність, лінійність, гетероскедастичність, репрезентативність, присутність трендів, циклів тощо.

Після детального аналізу підбирається модель, яка найкраще описує отриманий процес. Результати застосування моделі повинно бути перевірені

за допомогою критеріїв адекватності моделі. Точність прогнозу також визначатиметься відповідними критеріями оцінки.

РОЗДІЛ 1 ІСТОРІЯ ТА ОСОБЛИВОСТІ РОЗВИТКУ ЕКОНОМІЧНИХ ПРОЦЕСІВ СУЧАСНОСТІ ТА ЇХ МОДЕЛЮВАННЯ

1.1 Характеристика розвитку економічних процесів в Україні та світі

Сучасний розвиток суспільства можна розглянути через призму глобалізації та відкритий і багатогранний світ, у якому неможливо жити без нескінченої кількості найрізноманітніших послуг.

Ще за часів кам'яного віку людське існування було тісно пов'язано з обміном, придбанням, виготовленням різноманітних товарів різного призначення. З розвитком суспільства та зростанням кількості населення з'явилася необхідність розбиття праці та обліку виготовленої продукції. Так почали виникати перші економічні статистичні дані різної природи.

Епоха Відродження сприяла більшій глобалізації тогочасних країн, прискореному видобуту цінних металів та як наслідок появі перших фінансових бірж, де можна було придбати або продати той чи інший метал, виріб тощо. Поява класу малих ремісників зумовила пожвавлення процесу вироблення різної продукції , а їх значна кількість посприяла створенню перших задокументованих даних про обсяги та тип виробництва, що невдовзі дозволить застосувати перші статистичні методи для аналізу побудови математичних моделей процесів економічної природи.

Велика кількість факторів могла впливати на вартість привезених з країн Америки металів, як ось наприклад частка металу в грошах, темпи видобутку, розвиток супутніх технологій, війни.

На рисунку (рис.1.1) можемо бачити графік знецінення американської валюти за останні 386 років[1]. Даний малюнок зображує експоненційний ріст знецінення, беручи 1635 рік як базовий. Таким чином стає помітний експоненційний тренд, та з'являється можливість застосувати математичні моделі які описуються рівняннями з наявністю експоненти. У регресійному аналізі класичною моделлю для побудови прогнозу цього нелінійного

нестаціонарного економічного процесу є модель ARIMA, яка більш детально буде описана у другому розділі. Саме графічний спосіб аналізу вплинув на подальші кроки вивчення способів побудови математичних моделей для прогнозу процесів.

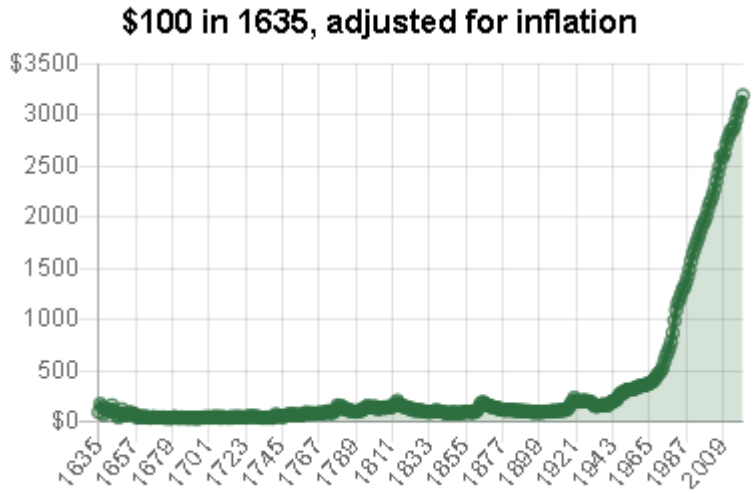


Рисунок 1.1, аркуш 9 – Знецінення долара за останні 386 років

Варто зазначити, що з появою дрібної торгівлі, під впливом індустріальної революції з'явилася необхідність фіксувати, зберігати та аналізувати економічні дані для створення та зведення бюджету, планування фінансування державних секторів, аналізу економіки виробництва, прогнозування та попередження фінансових криз, проведення військових кампаній тощо.

Найпростіші методі аналізу являли собою побудову таблиць з виробничими значеннями (вартість, кількість, об'єм, вага) та зображенням цих самих значень на графіку. Кожний процес економічної природи має свою особливу поведінку, на нього впливає різна кількість чинників, він може описуватися різними типами рівнянь, може містити тренд, зростати чи спадати, мати лінійний графік чи експоненційний. На рисунку (рис.1.2) [2] зображений графік щорічного приросту ВВП України на душу населення протягом останніх тридцяти років. З малюнку можна зробити висновок, що графік відсоткового росту ВВП нашої країни – складне поліноміальне

рівняння на окремих ділянках яких вдається помітити порядок росту та тренд та проаналізувати його. Очевидно, що довготривалий динамічний прогноз в такому випадку – задача пошуку системи багатьох рівнянь , а даний ряд буде нестаціонарним та нелінійним зі змінною дисперсією протягом всього часу проведення вимірів.

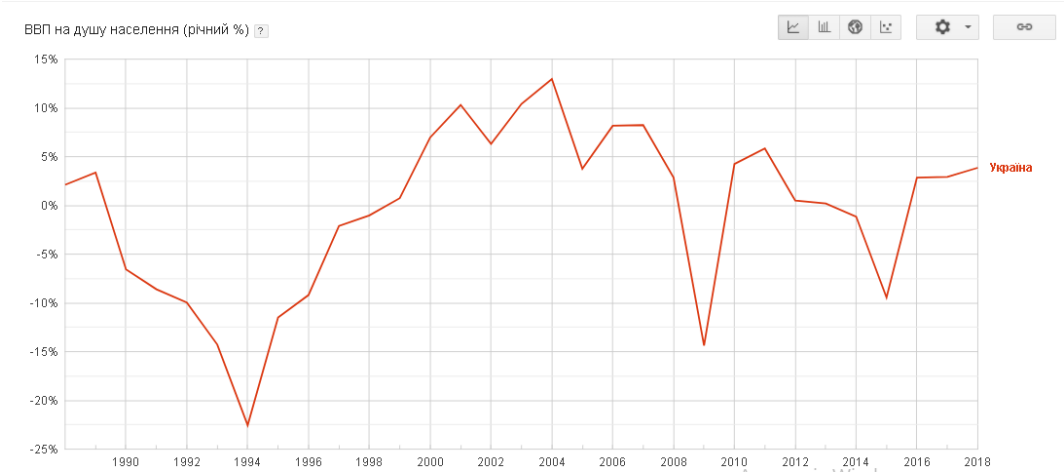


Рисунок 1.2, аркуш 11 – Річна зміна зросту ВВП України на душу населенняпротягом 30 років

Попередній малюнок чітко демонструє нестаціонарність економічного процесу.

1.2 Способи моделювання нелінійних нестаціонарних процесів

Розвиток програмних технологій надає можливість використання та застосування широкого математичного апарату для моделювання нелінійних нестаціонарних процесів в економіці. Правильно підібрана математична модель дозволяє з високою точністю прогнозувати поведінку економічного процесу будь-якої природи, а висока продуктивність ЕОМ сприяє обробленню та аналізу великих масивів даних за короткий проміжок часу.

1.2.1 Регресійний аналіз

Найбільш розповсюджені методи побудови математичних моделей здійснюються за допомогою регресійного аналізу. Ці методи зазвичай дозволяють побудувати точний прогноз із лінійними або експоненційними залежностями. Хоча існують і поліноміальні види регресії для побудови більш складної моделі. В окремих випадках застосовують “лінеаризацію” процесу. Також може бути застосований сегментований підхід, коли для кожного окремого сегменту намагаються побудувати регресійну модель яка буде описувати економічний нелінійний процес.[13] Підхід побудови регресії базується на знаходженні відношення між залежною змінною y та однією (або декількома) незалежними змінними x. При цьому для збільшення ефективності моделі, рівняння може містити значення шуканої змінної затриманої у часі на визначену кількість кроків (лаги). Нехай маємо n незалежних змінних $x_i, i = 1,..,n$. Тоді рівняння мішаної регресії буде визначатися за формулою 1.1 [3].

$$y(k) = a_0 + a_1 y(k-1) + a_2 y(k-2) + \cdots + a_n(k) y(k-n) + b_1 x_1(k) + b_2 x_2(k) + \cdots + b_p x_p(k) + e(k), \quad (1.1)$$

де $a_i,\ i = 1,..,n$ – коефіцієнти моделі, які оцінюються на основі значень часового ряду;

$e(k)$ – випадкова величина, що може зумовлюватися впливом збурень на процес.

Регресію використовують як інструмент для прогнозування процесів, тому задані вектори $x_i,\ i = 1,..,n$ називають екзогенними змінними (незалежні змінні), а вектор (або вектори) y – ендогенними змінними (залежна змінна).

Похибка моделі визначається як різниця між реальними та прогнозованими значеннями залежної змінної. За формулою 1.2 [3].

$$e(k) = y - \left(a_0 + a_1 x_1(k) + a_2 x_2(k) + \cdots + a_n x_n(k)\right) \quad (1.2)$$

де a_i, $i = 1,..,n$ – коефіцієнти моделі, які оцінюються на основі значень часового ряду;

$e(k)$ – випадкова величина, що може зумовлюватися впливом збурень на процес.

Для знаходження оптимальних значень параметрів $a_0, a_1, \dots, a_n$ в якості оптимізаційної функції мінімізується сума квадратів похибок. За формулою 1.3:

$$\sum_{i=1}^{n} e_i^2 \to min \tag{1.3}$$

де e_i– похибка прогнозу на і-ому кроці;

У разі існування поліноміальної залежності між незалежною та залежною змінною, використовують варіант псевдолінійної (поліноміальної регресії). Рівняння поліноміальної регресії визначається за формулою 1.4:

$$y(k) = a_0 + a_1 x_1(k) + a_2 x_1^2(k) + \cdots + a_n x_1^n(k) + e(\mathrm{k}) \tag{1.4}$$

де n – порядок регресії.

Коефіцієнти цього рівняння також знаходяться за допомого МНК чи методу максимальної правдоподібності.

1.2.2 Метод групового урахування аргументів

Метод групового урахування аргументів іноді вважають узагальненням підходу регресійного аналізу. Ця методика використовується у дуже різноманітних галузях аналізу даних (наприклад, розпізнавання образів, моделювання різноманітних систем, оптимізації тощо). У ході виконання алгоритму відбувається перебір різних варіантів моделювання вхідного

процесу, при цьому автоматично обирається найкращий варіант моделі за спеціальними критеріями адекватності (коефіцієнт детермінації, суми квадратів похибки тощо). В загальному вигляді модель має вигляд полінома за формулою 1.5:

$$y = a_0 + \sum_{i=1}^{n} a_i x_i + \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} x_i x_j + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} a_{ijk} x_i x_j x_k \ , \qquad (1.5)$$

де $a_i,\ a_{ij},\ a_{ijk}, \ldots$ – параметри моделі;

$x_i\ ,\ i = 1,..,n$ – незалежні змінні (предиктори).

Постановка завдання для застосування методу повинна включати наступні елементи:

1. вибір додаткових описів, які створюють основу для можливого остаточного вибору моделі;
2. відбір та адаптація параметрів моделі для конкретного застосування
3. розробка нової або застосування відомої моделі оцінки;
4. вибір критеріїв якості моделі для використання в проміжних кроках обчислень і для остаточного вибору моделі;

Моделі, побудовані з відповідним чином налаштованим МГУА, зазвичай забезпечують середню або високу якість короткострокових прогнозів [10].

В останнє десятиліття інтерес до МГУА активно зростає в усьому світі, що можна пояснити, окрім відомої ефективності методу, також зростанням популярності технології штучних нейромереж. Річ у тім, що структуру МГУА можна інтерпретувати як нейромережу, оригінальність якої полягає в самоорганізації як її структури, так і параметрів. При цьому виявляється, що до явних переваг МГУА належать автоматичне формування структури мережі, простота і швидкодія настроювання параметрів, а також можливість «згорнути» побудовану мережу безпосередньо в явний математичний вираз [7].

1.2.3 Байєсівські мережі

Байєсівські мережі застосовуються для побудови моделі і прогнозу нелінійних нестаціонарних процесів. Мережа являє собою пару <G,B>, де компонента G - спрямований нециклічний граф, що відповідає змінним, які досліджуються і записується у вигляди причинно-наслідкової мережі. Друга компонента B – це множина параметрів, що визначають мережу. З математичної точки зору байєсівська мережа – це модель подання наявних і відсутніх ймовірнісних залежностей. При цьому зв'язок (1.6):

$$A \to B \tag{1.6}$$

де A – подія, яка відбулася раніше;

B- подія, яка відбулася після події A.

є причинним, якщо подія A є причиною виникнення B , тобто коли існує механізм, відповідно до якого значення, прийняте A , впливає на значення, прийняте B . Байєсівську мережу називають причинною (каузальною), якщо всі її зв'язки причинні. Реалізація алгоритму для побудови структури байєсівської мережі може бути виконана на основі використання тестів на умовну незалежність [4]. Загальна постановка задачі прогнозування за допомогою мереж Байєса складається з наступних етапів:

1. доскональне вивчення процесу модельованої величини;
2. збір даних та експертна оцінка збору;
3. вибір відомого або будівництво нового методу модельної структури;
4. навчання параметрів байєсівської мережі (складання таблиць умовних ймовірностей);
5. розробка нового або підбір відомого способа виводу;
6. розробка нового або підбір відомого способа виводу;
7. застосування моделі для практичного вирішення задач;

Ступінь успішності застосування даного методу моделювання та формування статистичного висновку залежить від вміння коректно сформулювати постановку задачі, вибрати змінні процесу, які в достатній мірі характеризують його динаміку або статику, зібрати статистичні дані та використати їх для навчання мережі, а також коректно сформувати результат – висновок за допомогою побудованої мережі. Побудова БМ пов’язана з необхідністю послідовного розв’язання декількох задач, зокрема це задачі обчислювального характеру, що зустрічаються при навчанні мережі. В загальному випадку навчання мережі відноситься до NP -повних задач, тобто об’єм обчислень зростає поліноміально із збільшенням кількості вузлів (змінних) мережі [8].

Незважаючи на те, що загальна теорія БМ розроблена досить добре, виникають, як правило, багато питань, коли конкретна практична проблема вирішена. Це особливо вірно щодо завдань прогнозування, тому що вимоги до якості оцінок прогнозування постійно зростає, що призводить до подальшого уточнення обчислювальних методів і алгоритмів.

1.2.4 Узагальнені лінійні моделі

Узагальнені лінійні моделі (УЛМ) це клас моделей, які розширюють уявлення про лінійне моделювання і прогнозування в тих випадках, коли чистий лінійний підхід до встановлення відносин між змінними процесу не може бути застосований. Підхід УЛМ також розширює можливості для моделювання у випадках, коли розподіли статистичних даних відрізняються від звичайних. Конструювання УЛМ можна розглядати з точки зору класичної статистики або з байесівской точки зору. Зазвичай проблема побудови такого типу моделі має враховувати такі елементи: тип попереднього розподілу для параметрів моделі; метод оцінки параметрів, що використовує відповідні методи моделювання; необхідність ієрархічного моделювання, апостеріорне моделювання, тощо [11]. УЛМ можуть бути успішно застосовані для

вирішенні проблем класифікації та прогнозування нелінійних процесів. Установлено, що для оцінювання параметрів узагальнених лінійних моделей доцільно застосовувати узагальнений зважений метод найменших квадратів, який у цьому випадку забезпечує отримання незміщених ефективних оцінок. Альтернативою є метод Монте-Карло для марковських ланцюгів [9].

1.2.5 Метод подібних траєкторій

Ідея методу полягає в наступному: маємо ряд спостережень екологічного процесу, що їх зроблено за якийсь час $\{y(1), y(2), \dots, y(n)\}$, графік якого наведено на рисунку 1.3:

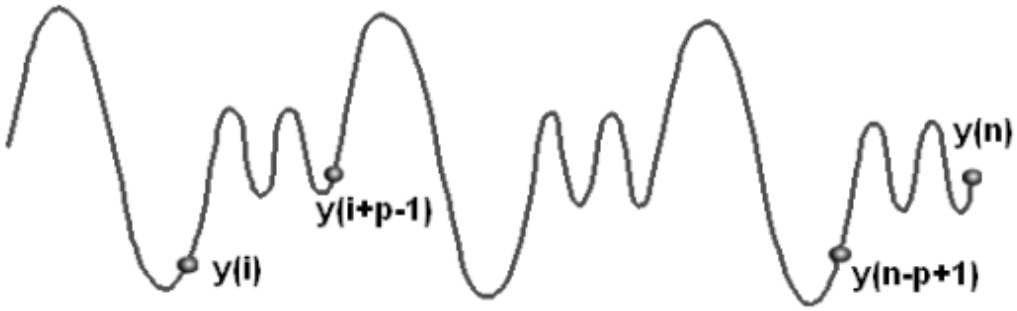


Рисунок 1.3 аркуш 17 – Динаміка ряду спостережень

Змінна y(i) тут представлена фізичними значеннями відповідного процесу (наприклад, сила вітру, інтенсивність стоку води, сила підземних поштовхів). За обраним критерієм обирається ділянка траєкторії “найближча” до ділянки, яка передує прогнозованій точці [12]. Надалі оцінюється прогноз за формулою 1.7 - 1.8 [6]:

$$I = \min\left\{\sum_{i=1}^{p} |y(j+i-1) - y(n-p+i)|, J = 1, \dots, n-p\right., \qquad (1.7)$$

$$J = \min_{i} |y(i+j-1) - y(n)|, i = I, I+1, \dots, I+p-1\,, \qquad (1.8)$$

Формалізувати метод можна наступним чином. Нехай ми маємо наступні вектори спостережень 1.9 [6]:

$$Y_1 = (y, y_2, \dots, y_p)^T, \dots, Y_n = (y_{n-p+1}, y_{n-p+2}, \dots, y_n)^T, \quad (1.9)$$

Знаходимо найближчу точку із умови мінімальної відстані за формулою 1.11 [6]:

$$Y_k = arg \min_j d(Y_n, Y_j), \quad (1.10)$$

де Y_n – вектор спостереження у момент часу n.

1.3 Програмні продукти для аналізу і моделювання нелінійних процесів

Сьогодні на ринку представлена велика кількість розроблених статистичних пакетів для обробки, аналізу та побудови моделі часових рядів. Найбільш відомим з них є : Eviews, SAS, STATISTICA, пакетні інструменти мови програмування Python.

SAS- програмне забезпечення для виконання статистичного, текстового аналізу даних. Цей програмний продукт має власну мову програмування, яка дозволяє розробляти та модифікувати стандартні пакетні засоби обробки та аналізу. Проста інтеграція з базами даних, чіткий та дружній до користувача інтерфейс дозволяє швидко розібратися у роботі. Існування версії для студентів робить програмне забезпечення непоганим варіантом для проведення дослідження.

Eviews – статистичний пакет для економетричного аналізу даних. Найбільші його переваги – велика кількість вбудованих моделей для застосування, наявність пакету для статистичної обробки вхідних даних,

інтуїтивно зрозумілий інтерфейс, гарно написана документація. Вбудовані програмні засоби дозволяють проводити аналіз часових рядів. Наявна можливість виконання тестів на дослідження типу вхідного ряду (вбудовані такі тести, як тест Дікі-Фуллера на стаціонарність, bds independence test, Хаусмана тощо). В останніх версіях програми з'явилася можливість отримати ліцензію для студентів, що робить можливість використання цього пакету для проведення та оцінці досліджень.

STATISTICA – статистичний пакет, розроблений компанією StatSoft. Вбудовані інструменти дозволяють будувати найрізноманітніші моделі, будувати нейронні мережі та проводити попередню обробку даних. Однак відсутність вбудованих тестів для аналізу вхідних даних робить роботу дослідника більш складною через необхідність використання сторонніх продуктів для проведення статистичних тестів "сирих" даних.

Jupiter Notebook –середовище розробки програмного коду, яке поєднує у собі зручний інтерфейс та вбудовані пакетні розширення для обробки, аналізу та побудови математичних моделей. Швидкість роботи та простота встановлення, безкоштовна ліцензія та доступність навчальних матеріалів робить Jupyter Notebook справжнім інструментом для розробки власного програмного рішення. Проте існує обмеження у вигляді знання базового синтаксису та структур даних мови програмування Python.

1.4 Постановка задачі і висновки до розділу

Постановка задачі

У першому розділі було проаналізовано чинники виникнення перших економічних даних, обгрунтовано важливість розуміння поведінки економічних процесів, наведено способи побудови та оцінки математичних моделей. Моделювання економічних процесів дозволяє зрозуміти природу та чинники, які впливають на виникнення та протікання процесів, запобігти погіршенню прогнозованих оцінок, змінити спосіб обліку або стратегію

економічної поведінки. Також було оглянуто основні підходи до моделювання нелінійних процесів, описані способи та базові алгоритми реалізації кожного з них.

Для досягнення мети дослідження було поставлено і виконано такі завдання:

- виявити нелінійності і нестаціонарності у сучасних фінансово-економічних процесах;
- виконати аналіз деяких методів дослідження нелінійних нестаціонарних процесів;
- вибрати процеси для дослідження та зібрати необхідні статистичні дані;
- виконати порівняльний аналіз отриманих результатів і виробити рекомендації стосовно їх практичного застосування.

РОЗДІЛ 2 ОГЛЯД ТА ХАРАКТЕРИСТИКА СТРУКТУР МАТЕМАТИЧНИХ МОДЕЛЕЙ ПРОГНОЗУВАННЯ І ЇХ ВИКОРИСТАННЯ

2.1 Статистичні тести для аналізу процесів

2.1.1 Тести на стаціонарність

Математичні методи побудови моделі прогнозу часових рядів мають найбільшу ефективність при використанні з даними, значення математичного сподівання якого не змінюється з часом, тобто зі стаціонарним. Найбільш уживаний статистичний тест для визначення цього критерію є тест Діккі-Фуллера.

Маємо часовий ряд спостережень $\{X_t\}$. Такий ряд має одиничний корінь у випадку, коли його перша різниця є стаціонарним рядом. Ряд перших різниць складається за допомогою наступного перетворення (утворений ряд має розмір на одиницю менше за оригінальний). За формулою 2.1 маємо:

$$\Delta X_t = X_t - X_{t-1}, \tag{2.1}$$

де X_t- значення часового ряду, в момент t;

X_{t-1}- значення часового ряду в момент t-1.

Далі проводиться дослідження авторегресії першого AR(1)

$$X_t = a \cdot X_{t-1} + \varepsilon_t\,, \tag{2.2}$$

де ε_t – похибка.

Рівняння (2.2) можна переписати, використовуючи (2.1) в наступному вигляді:

$$\Delta X_t = b \cdot X_{t-1} + \varepsilon_t, \tag{2.3}$$

де $b = a - 1$.

Далі виконується перевірка наступних гіпотез:

H_0: $b = 0$, справдження якої свідчить про нестаціонарність ряду.

$H_1 : b < 0$, справдження якої свідчить про стаціонарність ряду.

Існує багато версій та модифікацій теста Діккі-Фуллера. Також існують інші версії тесту Дікі-Фулера, у яких використовуються наступні рівняння:

$$\Delta X_t = b_0 + b \cdot X_{t-1} + \varepsilon_t, \tag{2.4}$$

$$\Delta X_t = b_0 + b_1 t + b \cdot X_{t-1} + \varepsilon_t, \tag{2.5}$$

де X_{t-1}- значення часового ряду в момент t-1.

У випадку (2.4) наявне зміщення на константу b_0, проте немає лінійного тренду. При використанні тестової регресії (2.5) ми включаємо ще лінійний тренд. Від вибору варіації тесту Дікі-Фулера залежить критичне значення DF-статистики, що береться з таблиці МакКіннона [1].

У випадку, якщо процес може бути авторегресією не першого, а більш високого порядку, доцільніше використовувати розширений тест Дікі-Фулера, який включає в себе лаги перших різниць.

Наприклад, у випадку авторегресії другого порядку AR(2) маємо:

$$X_t = a_1 \cdot X_{t-1} + a_2 \cdot X_{t-2} + \varepsilon_t \tag{2.6}$$

де X_{t-1}- значення часового ряду в момент t-1;

X_{t-2}- значення часового ряду в момент t-2;

$a_1, a_2 -$ коефіцієнти.

Перепишемо рівняння (2.6) в наступному вигляді:

$$\Delta X_t = (a_1 + a_2 - 1) \cdot X_{t-1} - a_2 \Delta X_{t-1} + \varepsilon_t \tag{2.7}$$

де X_{t-1}- значення часового ряду в момент t-1;

$\varepsilon_t -$ похибка моделі прогнозу в момент часу t;

$a_1, a_2 -$ коефіцієнти.

В цьому випадку перевіряється гіпотеза, що коефіцієнт при X_{t-1} рівний нулю аналогічно звичайному тесту Дікі-Фулера.

У загальному випадку для розширеного тесту Дікі-Фулера використовується наступна тестова регресія (авторегресія порядку k з включеним зміщенням та лінійним трендом):

$$\Delta X_t = a_0 + a_1 t + bX_{t-1} + \sum_{i=1}^{k} c_i \Delta X_{t-i} + \varepsilon_t, \quad (2.8)$$

де a_0, a_1, b, c_i , $i = 1, .. k$ – невідомі параметри.

Розраховується наступна статистика за формулою 2.9:

$$|t_{\text{стат}}| = \left| \frac{\hat{b}}{\sigma_{\hat{b}}} \right|, \quad (2.9)$$

де $\hat{b}$ – оцінка параметра b;

$\sigma_{\hat{b}}$ – стандартне відхилення оцінки.

У випадку, якщо значення цієї t-статистики більше або рівне, ніж критичне значення, то нульова гіпотеза відкидається, отримуємо стаціонарний ряд. Інакше, ряд нестаціонарний. Відповідний порядок інтеграції часового ряду визначається шляхом зведення ряду до стаціонарного (почергового взяття операціями перших різниць).

2.1.2 Тести на нелінійність

Існує досить багато тестів на встановлення нелінійності часового ряду, зокрема: Keenan test (1985), Tsay test (1986), Brock, Hsieh, and LeBaron test (1991), Brock test (1996) тощо. Розглянемо один із найпоширеніших тестів на встановлення нелінійності часового ряду – BDS (Brock-Dechert-Scheinkman, 1996) тест.

Алгоритм виконання тесту полягає в наступному. Спочатку необхідно обчислити залишки для моделі авторегресії $AR(p)$. Оптимальний порядок p авторегресії визначається за допомогою часткової автокореляційної функції.

Нехай $\varepsilon(k)$ – отриманий ряд залишків часового ряду $y(k)$.

Формуються всі можливі послідовності із m елементів: $(\varepsilon(k), \varepsilon(k+$ $+1), \dots \varepsilon(k+m-1))$ для $k = 1, \dots, n-m$ (n – обсяг вибірки, $m \geq 2$ – обрана розмірність послідовності).

Після цього підраховується величина $C_{t,s}$ для $t = 1, \dots, n-m$ та для $s = 1, \dots, n-t-m+1$.

$C_{t,s} = 1$, якщо дві послідовності $(\varepsilon(t), \varepsilon(t+1), \dots \varepsilon(t+m-1))$ та $(\varepsilon(t+s), \varepsilon(t+s+1), \dots \varepsilon(t+s+m-1))$ знаходяться близько одна до одної.

$C_{t,s} = 0$, в іншому випадку.

Міру «близькості» двох послідовностей автори тесту визначають як максимальну евклідову відстань між членами цих послідовностей, що знаходяться на тих самих місцях. Тобто, якщо ми розглядаємо дві послідовності: $(\varepsilon(t), \varepsilon(t+1), \dots \varepsilon(t+m-1))$ та $(\varepsilon(t+s), \varepsilon(t+s+$ $+1), \dots \varepsilon(t+s+m-1))$, то визначаються евклідові відстані між членами цих послідовностей за формулою (2.9) [2].

$$d_j = |\varepsilon(t+j) - \varepsilon(t+s+j)|, \ \ j = 1, \dots, m-1, \tag{2.9}$$

де $\varepsilon(t+j)$ – отриманий ряд залишків часового ряду $y(t+j)$

$\varepsilon(t+s+j)$ – отриманий ряд залишків часового ряду $y(t+s+j)$.

У випадку $\max_j d_j \leq C$ послідовності вважаються близькими одна до одної. Параметр C визначається на власний розсуд. Досить часто у якості параметру m тесту BDS беруть число від 2 до 5, а у якості параметру C – значення від $0.5\sigma_x$ до $2\sigma_x$, де σ_x – середньоквадратичне відхилення вихідних даних [3].

Наступним кроком обчислюється величина C_m за формулою:

$$C_{m,T} = \frac{2}{(T-m)(T-m-1)} \sum_{t=1}^{T-m} \sum_{s=t+1}^{T-m-1} C_{t,s}, \tag{2.10}$$

де T – обсяг вибірки, який ми досліджуємо.

BDS-статистика – це стандартизоване значення $C_{m,T}$ [2]:

$$w_{m,T} = \sqrt{T-m-1}\,\frac{(C_{m,T} - C^m_{1,T-m+1})}{\sigma_{m,T-m+1}} \tag{2.11}$$

де T – обсяг вибірки, який ми досліджуємо.

Нульова гіпотеза H_0: $w_{m,T} \sim N(0,1)$. Тобто, у випадку, стандартного нормального розподілу статистики BDS часовий ряд є лінійним. У випадку відхилення нульової гіпотези, приймається альтернативна гіпотеза H_1, яка говорить про нелінійність вихідного ряду.

Науковцями *McLeod* A.I. та *Li* W.K. була запропонована дещо інша статистика для дослідження наявності нелінійності. Вона базується на дослідження квадратів залишків та їх автокореляцій [17]:

$$r_k = \frac{\sum_{t=k+1}^{T} (\varepsilon^2(t) - \hat{\sigma}^2)(\varepsilon^2(t-k) - \hat{\sigma}^2)}{\sum_{t=1}^{T} (\varepsilon^2(t) - \hat{\sigma}^2)^2}, \tag{2.12}$$

де $\hat{\sigma} = \sum \frac{\varepsilon^2(t)}{T}, k = 1, .., m,$

m – це параметр моделі, який обирається на власний розсуд.

В залежності від його значення, граничні значення статистики будуть змінюватись. Нелінійність перевіряється за допомогою статистики Ljung-Box.

$$Q_{ML} = T(T+2)\sum_{k=1}^{m}(T-k)^{-1}r_k^2 \qquad (2.13)$$

де T – обсяг вибірки, який ми досліджуємо.

Перевіряється гіпотеза H_0: статистика Q_{ML} асимптотично розподілена χ_m^2 , вихідний часовий ряд лінійний. У випадку відхилення нульової гіпотези, приймається альтернативна гіпотеза H_1, щодо нелінійності вихідного ряду.

2.1.3 Тести на гетероскедастичність

Гетероскедастичні процеси, або процеси із змінною дисперсією дуже поширені в нелінійних часових рядах, особливо фінансових. Сам процес побудових математичних моделей для таких часових рядів відрізняється від принципів побудови стаціонарних процесів. Існує велика кількість статистичних тестів для визначення того, чи є ряд гетероскедастичним. Відомими вважають наступні тести: тест Глейзера, тест Уайта тощо. Однак найбільш прикладним є тест Уайта, який і буде розглянуто.

На початку треба обчислити залишки моделі та зробити оцінку вихідного рівняння регресії. Ряд лишків $\{\varepsilon(k), k = 1,..,n\}$ буде застосований в подальшому кроці.

Потім будуємо рівняння регресії квадрату лишків всіх змінних,квадратів самих змінних, а також їх попарно узятих добутків.

Можемо зобразити рівняння авторегресії порядку p

$$(k) = a_0 + \sum_{i=1}^{p} a_i y(k-i) + \varepsilon(k), \qquad (2.14)$$

де $\varepsilon(k)$ – залишки моделі.

Вигляд допоміжного рівняння буде наступним:

$$\varepsilon^2(k) = b_0 + \sum_{i=1}^{p} b_i y(k-i) + \sum_{i \le j}^{p} b_{ij} y(k-i)y(k-j) + \varepsilon(k), \qquad (2.15)$$

де $\varepsilon(k)$ – залишки моделі.

Обчислюється коефіцієнт R^2 допоміжної моделі. У якості статистики береться величина nR^2. У випадку, коли справедлива гіпотеза про гомоскедастичність, дана статистика асимптотично розподілена за χ_k^2, де ступінь свободи k дорівнює числу регресорів в допоміжній моделі для квадратів залишків $\varepsilon(k)$.

Тобто, при заданому рівні значущості α гіпотеза про гомоскедастичність відхиляється у випадку $nR^2 \geq \chi^2_{\text{кр}}$. Критичне значення береться з відповідної таблиці χ^2 з заданим рівнем значущості α та k ступенями вільності.

2.2 Модель ARIMA

Модель ARIMA – інтегрована модель широко відомої моделі ARMA, яка застосовується до нестаціонарних процесів зі змінною дисперсією та коваріацією елементів часового ряду.

Дана абревіатура є описовою, що фіксує ключові аспекти самої моделі. Домоделі $ARIMA$ входять наступні складові:

1. AR- авторегресія. Дана модель враховує зв’язок прогнозованої

змінної з її значеннями у попередні проміжки часу. Окремо авторегресійна модель може бути використана в якості найпростішої моделі прогнозування;

2. I- інтегрованість. Даний параметр визначає зв'язок між авторегресією та ковзним середнім. Порядок інтегрування зазвичай описує кількість необхідної диференціації ряду для приведення його до стаціонарного;
3. MA- ковзне середнє. Ця модель використовує залежність між прогнозованою змінною і залишковими помилками моделі ковзного середнього, затриманої у часі.

Кожен компонент моделі є параметром. Загально прийнята форма запису – ARIMA(p,d,q) – де параметри у дужках – цілі числа, які відповідають конкретній моделі прогнозу. Найкращий підбір компонент моделі забезпечує найбільш точний прогноз. Існують багато способів визначення параметрів : від використання автокореляційних та частково-автокореляційних функцій до використання алгоритмів типу GreedSearch з автопідбором необхідних параметрів.

Визначення параметрів моделі ARIMA:

- p- кількість значень змінної на попередніх кроках, яка буде враховуватися у моделі;
- d- порядок диференціювання, тобто застосування оператору віднімання для приведення часового ряду до стаціонарного;
- q- розмір вікна ковзного середнього, яке буде використовуватися при побудові;

Загалом, дана модель описується інтуїтивно зрозумілим рівнянням 2.16:

$$y(k) = a_0 + a_1 y(k-1) + a_2 y(k-2) + \cdots + a_p(k) y(k-p) + b_1 \varepsilon_1(k-1) + b_2 \varepsilon_2(k-2) + \cdots + b_q \varepsilon_q(k-q) + e(k), \quad (2.16)$$

де $a_i,\ i = 1,..,n$ – коефіцієнти моделі, які оцінюються на основі значень часового ряду,

$b_i,\ i = 1,..,q$ – коефіцієнти моделі, як оцінюються на основі значень похибок прогнозу моделі середнього ковзного,

$e(k)$ – випадкова величина, що може зумовлюватися впливом збурень на процес.

Однак необхідно розуміти, що вищерозглянута математична модель не може бути універсальною та має ряд недоліків, які роблять її використання неможливим :

1. присутність у часовому ряду “сезонного” фактору;
2. дані часовогу ряду описують дуже нелінійний процес, який приближується лінійною регресіє з поганою точністю;
3. присутність у великій кількості в часовому ряді “аномальних даних”, що мають велику дисперсію;
4. для інтегрованих моделей характерна наступна особливість – припущення про постіну дисперсію, в той час як більшість фінансових даних демонструє змінну волатильність і ця особливість даних не може бути реалізована при такому припущенні ;

На рисунку 2.1 зображений приклад прогнозу часового ряду.

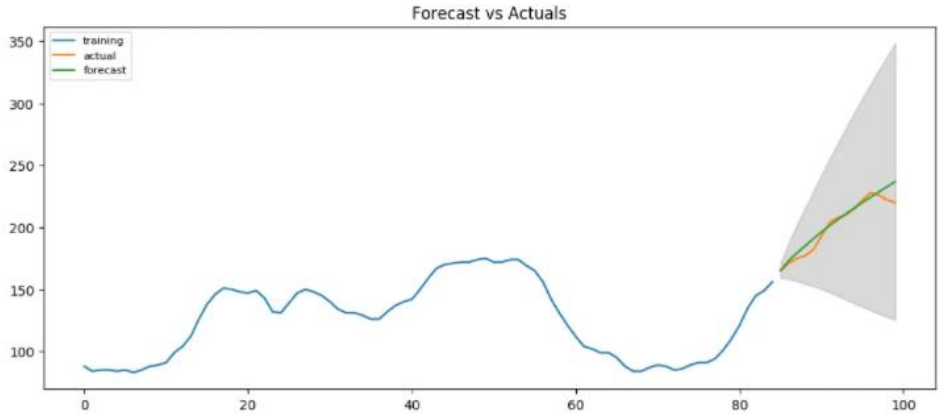


Рисунок 2.1, аркуш 29 –прогноз моделі ARIMA

Часто окрім кривої прогнозу зображують довірчий інтервал.

2.3 Лінгвістичне моделювання

Метод лінгвістичного моделювання належить до родини статистичних методі прогнозування, що зумовлює широку можливість його використання з даними абсолютно різної природи. Даний тип методів прогнозування відрізняється універсальність, проте трохи гіршою точністю.

Для побудови системи, необхідно розв'язати задачу пошуку лінгвістичного образу для досліджуваного часового ряду, яка складається з декількох етапів:

- обрахунок різницевих рядів;
- вибір значення критерію інтервального розбиття різницевого ряду;
- співставлення часовому інтервалу ряду певної літери алфавіту;
- побудова матриці переходу між будь-якими двома літерами;

Для побудови системи, сам процес лінгвістичного моделювання буде розподілений на окремі задачі.

Задача побудови першої різниці часового ряду – аналогу першої похідної [14]. Цей спосіб дозволяє прибрати тренд часового ряду та зробити його стаціонарним для подальшої побудови моделі для прогнозу, оскільки майже все методі прогнозування дають успішний результат при використанні їх зі стаціонарними часовими рядами. Слід зазначити, що використання перших різниць не завжди зробить ряд стаціонарним. Для визначення порядку тренду можна проаналізувати коефіцієнти автокореляції часового ряду.

Маємо : вектор з цілих чисел X потужністю $n = |X|$

Отримаємо : вектор з цілих чисел D потужністю $k = |D|$

$$\forall d_i \in D : d_i = x_{i+1} - x_i \qquad (2.17)$$

$$i \in [0; n-1); x_i, x_{i+1} \in X \qquad (2.18)$$

Задача вибору оптимального значення критерія інтервального розбиття різницевого ряду. Розв'язок задачі дозволяє побудувати алфавіт користувача шляхом розділення відсортованого ряду першої (зазвичай, якщо порядок тренду лінійний) різниці на безліч інтервалів, в якому кожний елемент характеризує певну літеру заданого алфавіту [14]. Варто розуміти, що дуже велика або дуже маленька кількість інтервалів розбиття негативно вплине на отриманий результат, тому що при побудові матриці переходу або дуже багато значень будуть потрапляти в один проміжок (при малому розбитті) або значення ймовірності переходу від одної літери до іншої будуть майже однаковими (при великому розбитті).

Маємо : алфавіт α, вектор з цілих чисел D потужністю $k = |D|$

Отримаємо : вектор цілочисельних пар I потужністю $n = |I|$

Вектор з цілих чисел D потужністю $k = |D|$

Обмеження :

$$\forall x \in I : x^1 \leq x^2 , \qquad (2.19)$$

$$\forall x_i, x_{i+1} \in I : x_i^2 \leq x_{i+1}^1, i \in [0; n-1), \qquad (2.20)$$

$$n \leq \alpha, \alpha \ll k, \qquad (2.21)$$

$$\forall d_i, d_{i+1} \in D : \ d_i \leq d_{i+1}, i \in [0; k-1), \qquad (2.22)$$

$$\forall x_0 \in I : \ x_0 = (-\infty; x_1^1), \qquad (2.23)$$

$$\forall x_n \in I : \ x_n = (x_{n-1}^2; +\infty), \qquad (2.24)$$

$$\exists x \in I \ : \ \forall d \in D, d \in [x^1, x^2] \tag{2.25}$$

Лінгвістична задача. Метою цієї задачі є формування лінгвістичного ланцюга шляхом співставлення відповідної букви алфавіту для кожного спостережуваного значення різницевого ряду. Кожна буква у визначеному алфавіті однозначно відповідає певному інтервалу з набору інтервалів, які були отримані при розв'язанні минулої задачі розбиття ряду відповідними інтервалами.

Маємо : вектор з цілих чисел D потужністю $k = |D|$,вектор цілочисельних пар I з потужністю $n = |I|$.

Отримаємо : вектор з літер A потужністю $k = |A|$

Обмеження :

$$\forall x_i \epsilon \bar{A}: \exists d_i \epsilon D, \exists y_j \epsilon \bar{I}, d_i \epsilon \left[y_j^1, y_j^2\right], x_i = j, \text{де } i \in [0; k), j \in [0; n) \tag{2.26}$$

На рисунку 2.2 зображений приклад отриманого лінгвістичного ланцюга.

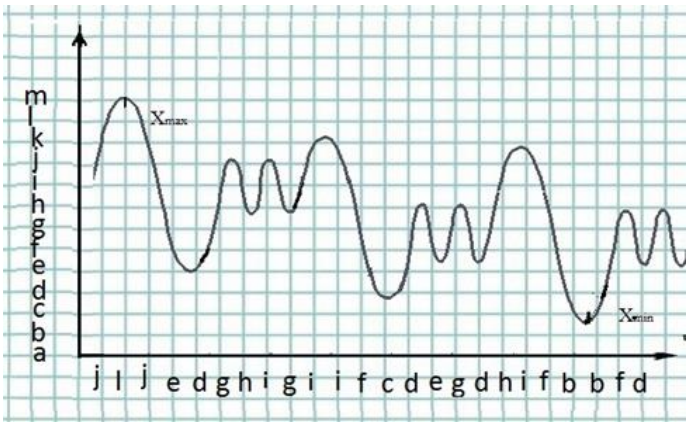


Рисунок 2.2 – Схема отриманого лінгвістичного ланцюга

Задача побудови матриці переходу між будь-якими двома літерами. Отримана послідовність аналізується на наявність граматичних конструкцій. На виході ми отримуємо список граматичних конструкцій з ймовірностями їх присутності в процесі, а також матрицю ймовірностей переходу від одного

символу до іншого символу. Цей етап тісно пов’язаний із моделюванням (прихованих) процесів Маркова, а також методом подібних траєкторій [15].

Маємо вектор з цілих чисел A потужністю $k = |\mathrm{A}|$, потужність інтервалів множини n, отриманою після виконання задачі розбиття на інтервали.

Отримаємо квадратну матрицю дійсних чисел P розмірністю n×n.

Обмеження:

$$\forall x_{ij} \epsilon \bar{P}: x_{ij} \epsilon [0.0; 1.0], \text{де } i, j \in [0, n) \tag{2.27}$$

Можна зазначити, що розглянутий метод моделювання відноситься до родини статистичних методів прогнозування, що робить можливий спектр його застосування з даними абсолютно різної природи та для рядів як стаціонарними так і нестаціонарними.

2.4 Метод Хольта-Вінтерса

Даний метод являє собою модифікацію методу ексоненційного згладжування, який враховує експоненційний тренд та адитивну сезонність. Сама модель складається з рівняння прогнозу та трьох наступних рівнянь згладжування : для тренду b_t, для сезонної компоненти та для рівня l_t та з відповідними параметрами згладжування α, β та γ. Для врахування сезонної компоненти існують два типа рівнянь: адитивний та мультиплікативний.

Адитивний метод використовують, якщо сезонні коливання приблизно постійні по всьому ряду, а мультиплікативний коли сезонні коливання змінюються пропорційно рівню ряду. При адитивному методі сезонна компонента виражається в абсолютному значенні в масштабі спостережуваного ряду, а в рівнянні рівня ряд коригується шляхом віднімання сезонної компоненти.

Рівняння адитивного методу:

$$\begin{gathered}\hat{y}_{t+h|t} = l_t + hb_t + s_{t+h-m(k+1)} \\ l_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(l_{t-1} + b_{t-1}) \\ b_t = \beta(l_t + l_{t-1}) + (1-\beta)\, b_{t-1} \\ s_t = \gamma(y_t - l_{t-1} - b_{t-1}) + (1-\gamma)\, s_{t-1}\end{gathered} \tag{2.28}$$

де *m* - довжина сезону;

k - ціла частина від (h-1)/m, що гарантує, що оцінка сезонних індексів буде отримана з останнього року з вибірки.

Рівняння мультиплікативного методу:

$$\begin{gathered}\hat{y}_{t+h|t} = (l_t + hb_t)s_{t+h-m(k+1)} \\ l_t = \alpha\frac{y_t}{(y_t - s_{t-m})} + (1-\alpha)(l_{t-1} + b_{t-1}) \\ b_t = \beta(l_t + l_{t-1}) + (1-\beta)\, b_{t-1} \\ s_t = \gamma\frac{y_t}{l_{t-1} + b_{t-1}} + (1-\gamma)\, s_{t-1},\end{gathered} \tag{2.29}$$

де *m* - довжина сезону;

k - ціла частина від (h-1)/m (занести в формулу), що гарантує, що оцінка сезонних індексів буде отримана з останнього року з вибірки [16].

Рівняння тренду тут таке саме як і в методі Хольта, тобто вказує на зважене значення тренду в момент часу *t* та оцінку тренду в момент часу *t-1*. Рівняння рівня показує зважене значення між сезонно скоригованими результатами та прогнозу без урахування сезону на момент часу *t*. Рівняння сезону є зваженим значенням між поточним значенням сезонного індексу та значенням сезонного індексу рік назад.

Усі значення в цій моделі залежать від попередніх, тому на першому кроці моделі виникає питання звідки брати початкові значення. Зазвичай початкові оцінки компонент отримуються за допомогою усереднення всіх

вхідних даних. Параметри згладжування зазвичай обирають такими, щоб мінімізувати похибку прогнозу.

Перевагами даного метода є те,що для використання експоненційного щгладжування не потрібна велика кількість даних, хоча підбори необхідних параметрів складають більшу частину роботи зі створеня прогнозу. Інколи варто використовувати вбудовані статистични пакети мови прогнозувань для виокремлення з часового ряду тренду, сезонності та похибок та працювати вже з ними. На рисунку 2.3 наведений приклад прогнозу за допомогою методу Хольта-Вінтерса. Можна побачити досить велику похибку прогнозу, яка пояснюється застосування експоненційного згладжування.

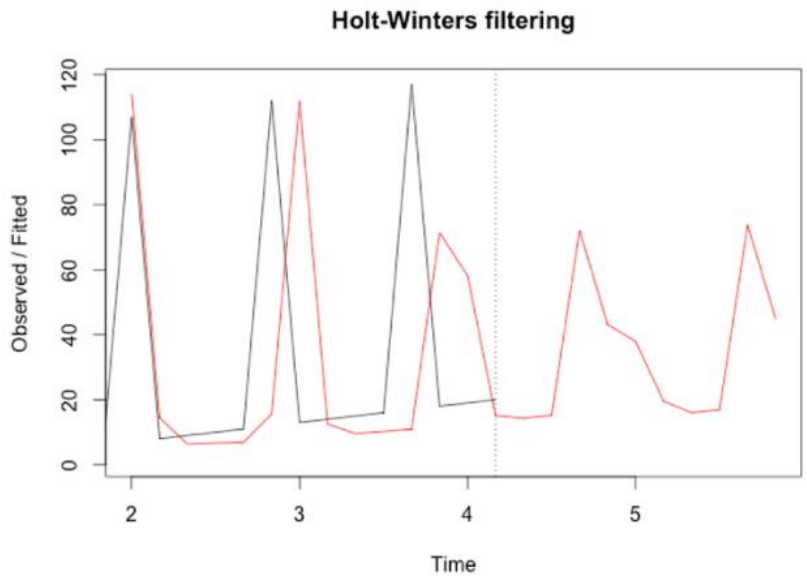


Рисунок 2.3 – Прогнозування за допомогою методу Хольта-Вінтерса

Даний метод прогнозу може застосовуватися в якості початкового приближення.

2.5 Моделі умовної гетероскедастичності

Зміна дисперсії або волатильності з часом може спричинити проблеми при моделюванні часових рядів класичними методами, такими як ARIMA.

Математична модель авторегресії умовної гетероскедастичності-АРУГ, дозволяє побудувати модель для прогнозування зміни дисперсії для часового ряду в різні проміжки часу. Узагальнена модель –УАРУГ, узагальнює модель, використовуючи той факт, що умовна дисперсія залежить також від попередніх значень самої умовної дисперсії. В часових рядах, де існує тренд на постійну зміну дисперсії, цю властивість називають гетероскедастичністю.

Модель АРУГ застосовується в економічній моделі для аналізу часових рядів (у першій черзі фінансових), у яких умовна (за минулим значенням ряду) дисперсія ряду залежить від минулих значень часового ряду, минулих значень дисперсії цього ряду та інших факторів. Ця модель, як і УАРУГ “пояснюють” кластеризацію волатильності на фінансових ринках коли періоди високої волатильності змінюються періодами низької волатильності, при чому середня (умовна) волатильність майже не змінюється з часом.

Нехай часовий ряд можна виразити наступним процесом:

$$y(k) = \varepsilon(k)\sqrt{a_0 + \sum_{i=1}^{q} y^2(k-i)a_i} \qquad (2.30)$$

де $\varepsilon(k)$ - білий шум, а $y^2(k-i)$ – квадрат спостережень затриманих з лагами, $a_i, i = 1, \dots, q$ - коефіцієнти моделі, які оцінюються на основі значень часового ряду.

Тоді як умовне та безумовне математичне сподівання буде рівним нулю,умовна дисперсія даного процесу буде:

$$\sigma^2(k) = a_0 + \sum_{i=1}^{q} y^2(k-i)a_i \qquad (2.31)$$

де $a_i, i = 1, \dots, q$ - коефіцієнти моделі, які оцінюються на основі значень часового ряду.

Така модель умовної дисперсії називається АРУГ(q). Для того, щоб значення дисперсії не було від’ємним робиться припущення, що всі

коефіцієнти моделі невід'ємні, при чому константа буде строго додатньою. Якщо спостерігаємий процес буде стаціонарним – то , вочевидь, дисперсія буде визначатися наступним чином:

$$\sigma^2(k) = \frac{\sigma^2(\varepsilon)}{1-\sum_{i=1}^{q} a_i} \tag{2.32}$$

де $a_i, i = 1, \dots, q$ - коефіцієнти моделі, які оцінюються на основі значень часового ряду.

Необхідна умова стаціонарності – сума коефіцієнтів моделі без константи строго менше за одиницю.

Модель УАРУГ покращує попередню за рахунок врахування умовної дисперсії на минулих кроках. Таке припущення дозволяє зменшити похибку прогнозу при дуже швидкій зміні волатильності часових даних, що особливо корисно при роботі з фінансами. В цьому випадку модель УАРУГ(p,q), де p – порядок АРУГ членів y(k), а q – порядок квадрату попередніх значень, затриманих з лагом. Тоді УАРУГ модель описується наступним чином :

$$\sigma^2(k) = a_0 + \sum_{i=1}^{q} y^2(k-i)a_i + \sum_{j=1}^{p} \sigma^2(k-j)b_j \tag{2.33}$$

де $\varepsilon(k)$ - білий шум, а $y^2(k-i)$ – квадрат спостережень затриманих з лагами, $a_i, i = 1, \dots, q$ - коефіцієнти моделі, які оцінюються на основі значень часового ряду,

$b_i, i = 1, \dots, q$ - коефіцієнти моделі, які оцінюються на основі значень умовної дисперсії на попередніх кроках.

Необхідна умова стаціонарності:

$$\sum_{i=1}^{q} a_i + \sum_{i=1}^{p} b_i < 1 \tag{2.34}$$

де $a_i, i = 1, \dots, q$ - коефіцієнти моделі, які оцінюються на основі значень часового ряду,

$b_i, i = 1, \dots, q$ - коефіцієнти моделі, які оцінюються на основі значень умовної дисперсії на попередніх кроках.

Безумовна дисперсія УАРУГ (p,q) буде постійною та дорівнюватиме :

$$\sigma^2(k) = \frac{\sigma^2(\varepsilon)}{1-\sum_{i=1}^{p} b_i - \sum_{i=1}^{q} a_i} \tag{2.35}$$

де $a_i, i = 1, \dots, q$ - коефіцієнти моделі, які оцінюються на основі значень часового ряду,

$b_i, i = 1, \dots, q$ - коефіцієнти моделі, які оцінюються на основі значень умовної дисперсії на попередніх кроках.

Використання моделі АРУГ та УАРУГ із нестаціонарними процесами ускладнює прогозування волатильності. Описані вище моделі можуть застосовуватися для прогнозування зміни волатильності часових рядів, однак прогнозування цінових значень ускладнюється.

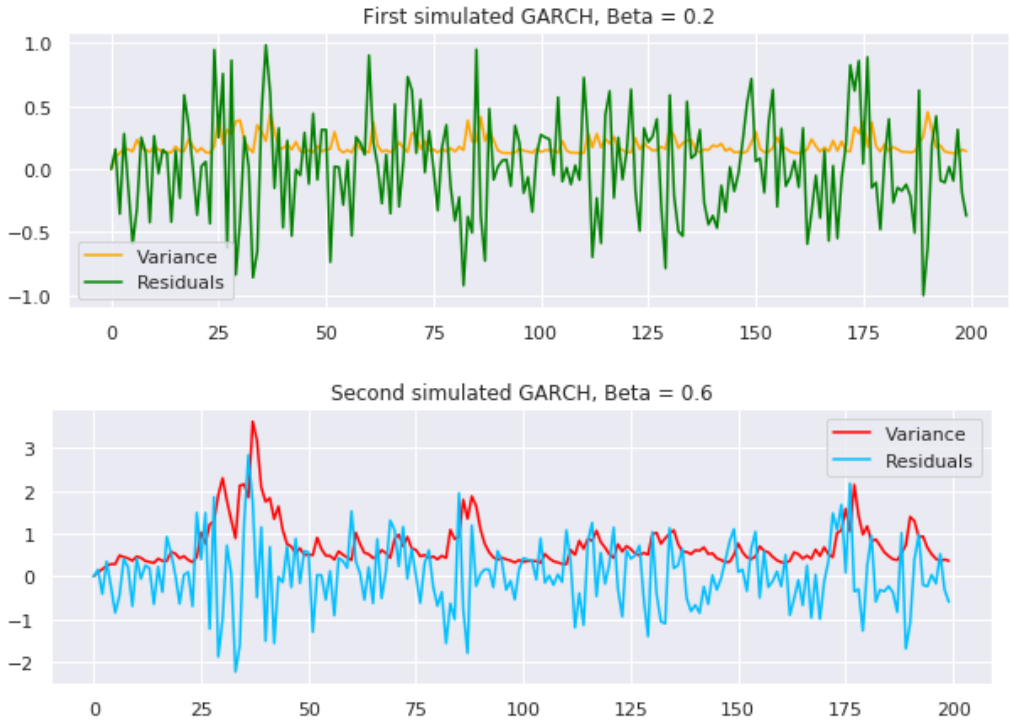


Рисунок 2.4 – Прогнозування волатильності методом УАРУГ

Оглянуті методи використовуються для дослідження зміни дисперсії часових рядів.

2.6 Критерії для оцінок адекватності побудованих моделей та оцінок якості прогнозів

2.6.1 Критерії адекватності моделей

Для виконання задачі прогнозування даних будь-якої природи, дослідник повинен знати критерії адекватності моделі, які характеризують те, наскільки побудована модель прогнозу є адекватною. Останнім часом з'явилася купа критеріїв для визначення ступеня адекватності. В даному розділі будуть згадані найбільш широко розповсюджені з них.

Статистика Дурбіна-Уотсона:

$$DW = \frac{\sum_{k=2}^{N}\left(\varepsilon(k)-\varepsilon(k-1)\right)^2}{\sum_{k=1}^{N}\varepsilon^2(k)}, \tag{2.36}$$

де N – розмір вибірки;

$\varepsilon(k)$ – лишки побудованої моделі.

Статистика Дурбіна-Уотсона може приймати значенні в діапазоні від 0 до 4. Цей критерій свідчіть про ступінь автокореляції лишків побудованої моделі. Якщо статистика DW=2, то це означає повну відсутність кореляції.

Коефіцієнт детермінації:

$$R^2 = \frac{Var(\hat{y})}{Var(y)}, \tag{2.37}$$

де $Var(\hat{y})$ – дисперсія часового ряду, оціненого побудованою моделлю;
$Var(y)$ – дисперсія точних значень часового ряду.

За означенням, коефіцієнт детермінації може приймати значення від 0 до 1. Він показує, наскільки дисперсія ряду побудована використаною моделі близька до дисперсії справжніх значень часового ряду. Наближення значення R^2 до 1 свідчить про малий вплив сторонніх факторів та досить непогану точність прогнозу.

Середньоквадратична похибка (СКП):

$$\text{СКП} = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y_i})^2}{N}}, \tag{2.38}$$

де N – обсяг вибірки;
$y_i, i = 1,..,N$ – реальні значення часового ряду;
$\hat{y_i}, i = 1,..,N$ – значення ряду, які спрогнозовані моделлю.

Середньоквадратична похибка зображує міру відхилення дійсних значень ряду від прогнозованих. Тобто чим менше значення СКП, тим кращою є побудована модель.

Скоригований коефіцієнт детермінації:

$$adjusted\ R^2 = 1 - (1 - R^2)\frac{N-1}{N-k}, \tag{2.39}$$

де N – кількість спостережень в часовому ряді,
k – кількість параметрів моделі.

Скоригований коефіцієнт детермінації усуває недолік звичайного – при додаванні в модель нових змінних залежна змінна не завжди буде залежати від

щойно уведених. Тому новий коефіцієнт штрафує коефіцієнт за введення нових змінних.

Сума квадратів похибок (SSE):

$$SSE = \sum_{i=1}^{N} (y_i - \widehat{y_i})^2 , \tag{2.40}$$

де N – обсяг вибірки;

$y_i,\ i = 1,..,N$ – реальні значення часового ряду;

$\widehat{y_i},\ i = 1,..,N$ – значення часового ряду, оцінені побудованою моделлю.

Даний критерій визначає суму усіх похибок, прогнозованих моделлю. Чим ближче даний показник до 0, тим краще спрогнозовані моделлю дані.

t-Статистика:

$$tStat = \frac{a}{\hat{\sigma}} , \tag{2.41}$$

де a – оцінка коефіцієнта,

$\hat{\sigma}$ – стандартна похибка оцінки.

t-Статистика (інколи згадується як статистика Стьюдента) використовується для перевірки гіпотези про рівність коефіцієнта нулю. Для того,щоб порахувати статистику Стьюдента необхідно обрати як рівень значущості так і кількість ступенів вільності , де α – рівень значущості, і кількість ступенів вільності $f = N - p$, де $N -$ обсяг вибірки, p – кількість параметрів моделі. Використовуючи ці значення у спеціальній таблиці Стьюдента ми знаходимо критичне значення статистики $t_{\text{крит}}$. У випадку $|tStat| \leq |t_{\text{крит}}|$ гіпотеза щодо нульового значення коефіцієнту моделі приймається .

2.6.2 Критерії оцінки якості прогнозів

Для розуміння точності прогнозування за допомогою побудованої моделі існує багато критеріїв. Нижчу буде наведено основні метрики для здійснення оцінки якості прогнозів.

Середня похибка (ME):

$$ME = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i), \tag{2.42}$$

Середня абсолютна похибка у відсотках (MAPE):

$$MAPE = \frac{1}{N}\frac{\sum_{i=1}^{N}|y_i - \hat{y}_i|}{y_i} \cdot 100\%, \tag{2.43}$$

Середньоквадратична похибка (RMSE):

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{T}(y_i - \hat{y}_i)^2}\ , \tag{2.44}$$

Середня похибка у відсотках (MPE):

$$MPE = \frac{1}{N}\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)}{y_i} \cdot 100\%\ , \tag{2.45}$$

Для найкращого прогнозу значення критеріїв (2.42), (2.43), (2.44), (2.45) повинні прямувати до нуля. Всі ці критерії показують величину відхилення прогнозованого значення від реального.

Коефіцієнт Тейла:

$$Theil = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{T}(y_i - \hat{y}_i)^2}}{\sqrt{\frac{1}{N}\sum_{i=1}^{T} y_i^2} + \sqrt{\frac{1}{N}\sum_{i=1}^{T} \hat{y}_i^{\,2}}}, \qquad (2.46)$$

Коефіцієнт Тейла приймає значення від 0 до 1. Чим менше значення коефіцієнту Тейла, тим якіснішим є прогноз. У випадку $Theil = 0$ прогнозовані значення точно співпадають з реальними і модель є ідеальною. Метрики оцінок якості прогнозу дозволяють зробити правильний висновок стосовно аналізу та підбору параметрів побудованої моделі.

```
forecast_accuracy(fc, test.values)

#> {'mape': 0.02250131357314834,
#>  'me': 3.230783108990054,
#>  'mae': 4.548322194530069,
#>  'mpe': 0.016421001932706705,
#>  'rmse': 6.373238534601827,
#>  'acf1': 0.5105506325288692,
#>  'corr': 0.9674576513924394,
#>  'minmax': 0.02163154777672227}
```

Рисунок. 2.2 – Приклад метрик оцінок якості прогнозу

2.7 Висновки до розділу

У другому розділі була описана методика оцінки вхідних даних, були розглянуті основні методи прогнозування нелінійних нестаціонарних процесів та представлений новий метод лінгвістичного моделювання , що являє собою

статистичний метод прогнозу з використанням елементів прихованих марковських процесів. Були розглянуті найбільш популярні тести для перевірки часового ряду на нестаціонарність, нелінійність та гетероскедастичність.

Були описані такі моделі : лінгвістична модель, ARIMA,УАРУГ, модель Хольта-Вінтерса. Також були наведені основні критерії адекватності моделей (такі як Дарбіна-Уотсона, коефіцієнт детермінації, SSR тощо). Також наведені критерії якості прогнозів, що обчислюються за побудованими моделями, зокрема, RMSE, ME, MPE, MAPE, Theil.

РОЗДІЛ 3. ПОБУДОВА МОДЕЛЕЙ ТА ОЦІНКА ПРОГНОЗІВ ВИБРАНИХ ЕКОНОМІЧНИХ ПРОЦЕСІВ

3.1 Обрання прогромного середовища

Розробка власного програмного продукту для моделювання та прогнозу нелінійних нестаціонарних процесів в економіці велася у інтегрованому середовищі Jupyter Notebook за допомогою мови програмування Python. Рішення про вибір доцільного інструменту реалізації програмного продукту було обрано на основі поширеності даної мови програмування, існуванні багатої кількості вбудованих бібліотек для роботи з часовими рядами та велика активна кількість користувачів. Наразі мова програмування Python найбільш поширена для аналізу даних.

Програма була написана у середовищі Jupyter Notebook, що дає змогу зручно писати окремі функції та запускати їх у потрібному порядку, легко завантажувати вхідні дані, інтегрувати та синхронізувати результати побудованих моделей у базу даних. Найбільшою перевагою даного середовища – є можливість ведення розробки на локальному сервері, що дозволяє швидко встановлювати необхідні бібліотеки, перезавантажувати вже написану програму, зберігати її на віддалені репозиторії тощо.

3.2 Функціональна схема розробленого програмного продукту

Програмний продукт реалізовано мовою програмування Python. Основна структура програмного продукту складається з окремих модулів, які виконують різні завдання та мають наступні етапи роботи:

- Етап завантаження вхідних часових рядів з комп’ютера або репозиторія у форматі .csv;

- Етап обробки та фільтрації даних, розбиття вхідної вибірки на тренувальну та тестувальну;
- Етап зображення графікиів вхідних часових рядів до їхнього перетворення;
- Етап проведення статистичних тестів на стаціонарність (тест Дікі-Фулера) та присутність нелінійностей (BDS);
- Етап побудови АКФ та ЧАКФ для коректного визначення порядку авторегресії та рухомого середнього;
- Етап побудови лінгвістичного моделювання та оцінювання параметрів моделі;
- Етап побудови авторегресійних моделей та авторегресійних моделей із сковзним середнім (AR,ARMA,ARIMA);
- Етап побудови та зображення спрогнозованих значень моделі та порівняння з існуючими даними;
- Етап оцінювання адекватності моделей та якості прогнозу;

Усі етапи виконаної роботи є окремим модулями, які не мають чіткої прив’язки один до одного, тобто стиль написання коду враховує можливе перевикористання вже написаних функціональних блоків у іншій програмній розробці. Варто зазначити, що етапи виконання та побудови прогнозу грунтуються на застосуванні так званих “best practice” , тобто найбільш ефективного та розповсюдженого алгоритму роботи з часовими рядами , починаючи від завантаження даних закінчуючи побудовою моделей прогнозу та використанні спеціальних метрик для їх оцінки. Більша частина функціоналу для прогнозування авторегресійних моделей використовує вбудовані бібліотеки. Метод лінгвістичного прогнозування було реалізовано за допомогою розробленого алгоритму, де всі кроки виконання моделювання і прогнозу були розроблені з нуля.

Функціональна схема програмного продукту представлена на рисунку 3.1:

Рисунок 3.1 – Функціональна схема програмного продукту

Для пришвидшення написання і реалізації обрахунків метрик та побудови математичних моделей були застосовані вбудовані бібліотеки мови Python, які користуються високою популярністю та мають великий рівень надійності роботи.

- Pandas – для перетворення вхідних даних у структури типу Series (часовий ряд) та DataFrame;
- Statsmodels – для побудови моделей і обчислення адекватності моделей;
- Matplotlib.pyplot – для загальної візуалізації роботи з часовими рядами та їхніми характеристиками;
- Math - для виконання математичних операцій;
- Sklearn – для реалізації лінійної регресії;

- Numpy – для роботи обробки та використання вбудованих функцій роботи з масивами, а також їхнього застосування у побудові математичних моделей;

3.3 Побудова моделей та оцінювання прогнозів вибраних процесів

Для побудови моделей та оцінювання прогнозів нелінійних нестаціонарних процесів був обраний наступний часовий ряд:
1) Ціна акцій компанії NETFLIX. Вибірка складається з 468 елементів.

3.3.1 Побудова моделей та прогнозів вартості акцій компанії NETFLIX

Первинний аналіз графіків часових рядів дає велику кількість необхідної інформації, яку можна застосувати при прогнозуванні та побудові моделей. Проаналізуємо зображення останніх річних денних цін акцій компанії NETFLIX на рисунку 3.2. Дані, які описують річну зміну акцій були взяті з офіційного сайту Yahoo:Yahoo Finance []. Легко можна визначити помітний позитивний тренд, який має місце з середини липня 2020 року та пояснюється відновленням активності після чотирьох місяців пандемії COVID-19. Також можна виявити деяку циклічність цін акції, коли умовна різниця в рівні цін повторюється щомісячно.

Графік демонструє чіткий лінійний тренд приросту цін акцій, що зумовлюється багатьма факторами. На основі цього можна зробити висновок щодо доцільності використання авторегресійних моделей прогнозування економічних даних.

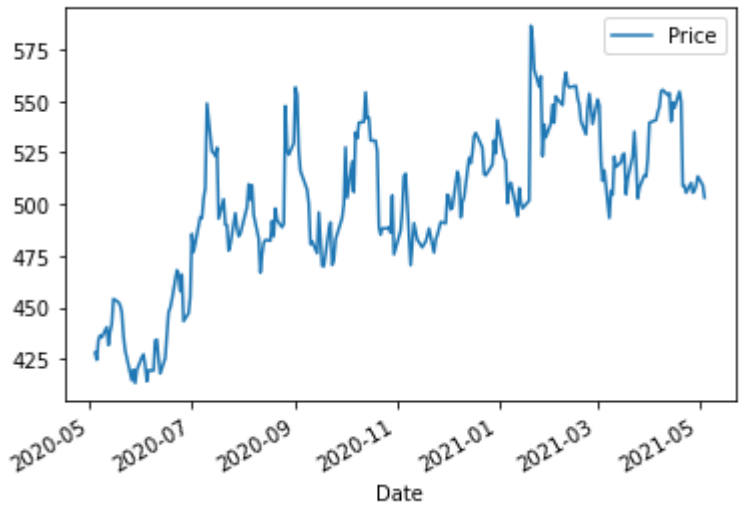


Рисунок 3.2 – Графічне зображення часового ряду «NETFLIX»

Одразу можемо побачити нестаціонарність часового ряду.

Побудуємо гістограму часового ряду (рис.3.3) :

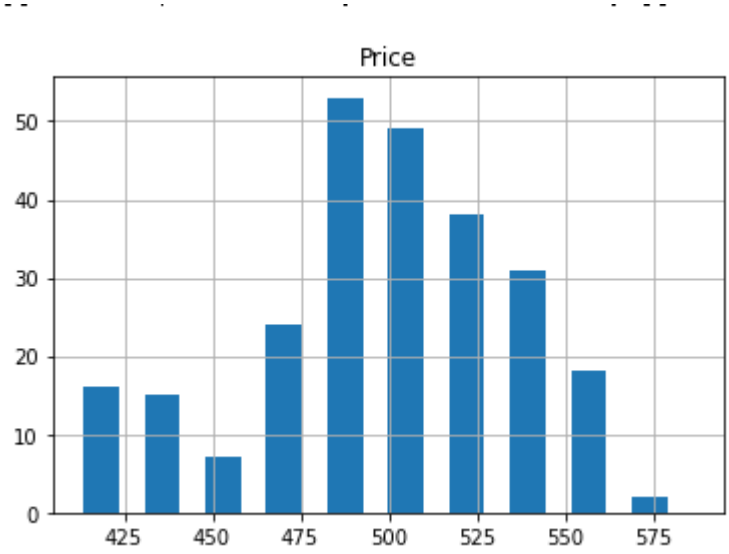


Рисунок 3.3 – Гістограма часового ряду «NETFLIX»

Згідно даних гістограми також можна побачити тренд зміні ціни акцій компанії, що вказує на нестаціонарність фінансового процесу. Далі необхідно проаналізувати основні статистики часового ряду, такі як середнє значення, дисперсія, мінімальні та максимальні ціни тощо (рис.3.4):

Out[154]:

	Price
count	253.000000
mean	501.075336
std	37.400579
min	413.440002
25%	482.839996
50%	503.179993
75%	527.330017
max	586.340027

Рисунок 3.4 – Основні статистики часового ряду «NETFLIX»

Для розуміння вибору параметрів авторегресії та ковзного середнього необхідно побудувати відповідно АКФ та ЧАКФ функції (рис.3.5), (рис3.6):

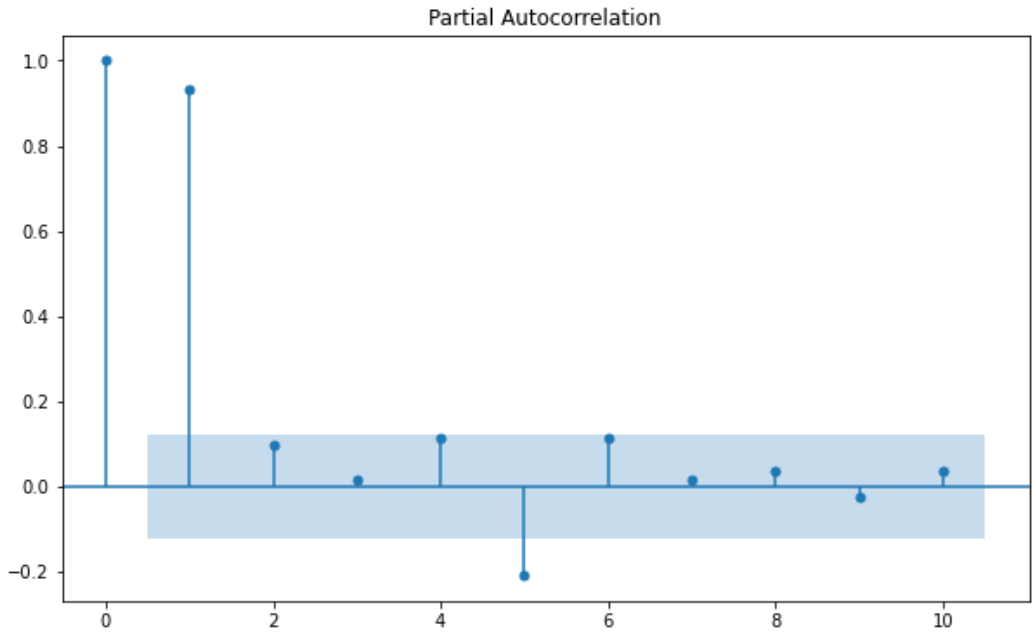


Рисунок 3.5, аркуш 52 – ЧАКФ часового ряду «NETFLIX»

Можемо побачити, що перший та п’ятий лаг буде мати найбільший вплив.

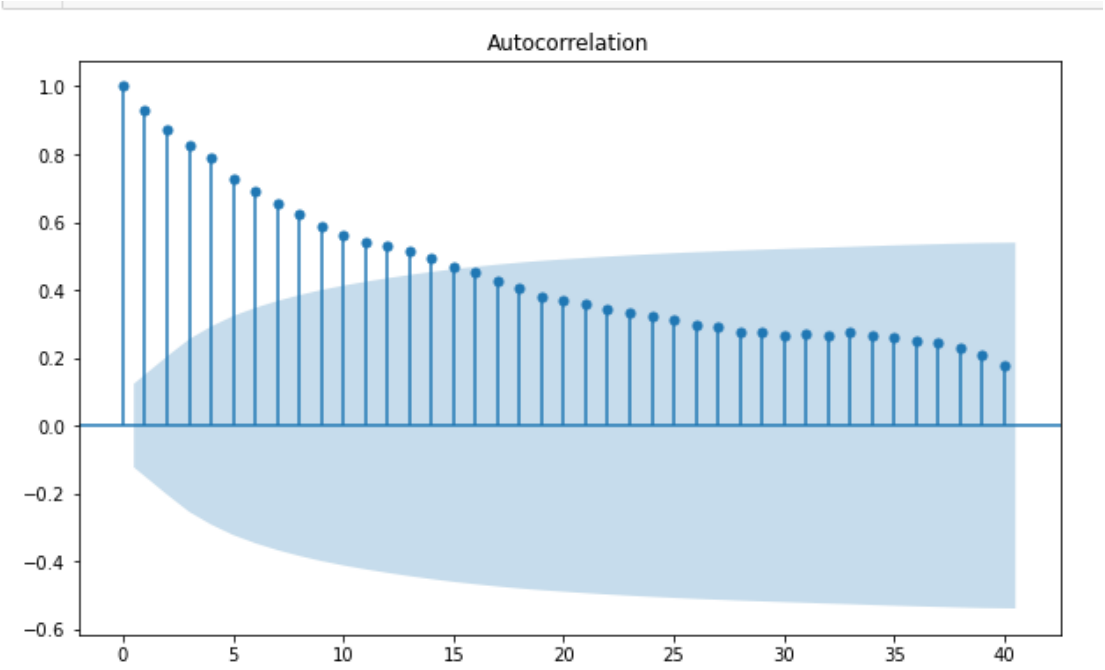


Рисунок 3.6 – АКФ часового ряду «NETFLIX»

Побудова моделі авторегресії першого порядку AP(1):

Проводячи аналіз ЧАКФ, можна помітити, що перший лаг має найбільшу статистичну значущість, тому буде побудована авторегресія першого порядку (рис.3.7):

```
Результати та рівняння моделі авторегресії порядку 1
y(k)= 36.34261646593541 +
0.9280727548898484 y( k - 1 )
Параметри адекватності моделі:
SSR =  43540.40774446886
DW =  2.21994705414396
R Squared =  0.8744725770722599
predicted=505.529787, true=506.519989
predicted=506.434029, true=509.000000
predicted=508.746370, true=513.469971
predicted=512.915257, true=509.109985
predicted=508.851921, true=503.179993
Якість прогнозу:
RMSE =  3.912236596853888
MAPE =  [0.71877105 0.61273452 0.51739445 0.96440231 0.51335234]
Theil =  0.0038477136400038993
Моделювання ряду NETFLIX AR(3)
```

Рисунок 3.7 – Модель AP(1) для часового ряду «NETFLIX»

Рівняння моделі:

$$y(k) = 36.3426 + 0.928072754 \cdot y(k-1) + \varepsilon(k), \qquad (3.8)$$

де $\varepsilon(k)$ – похибка моделі.

Зобразимо основні статистики для оцінки адекватності моделі AP(1) часового ряду «NETFLIX»:

$$R^2 = 0.87447$$
$$Sum\ squared\ resid = 43540.31$$
$$Durbin - Watson = 2.21994$$

Спробуємо спрогнозувати ріст цін акцій на 5 днів. Результати прогнозування зображені на рисунку 3.9:

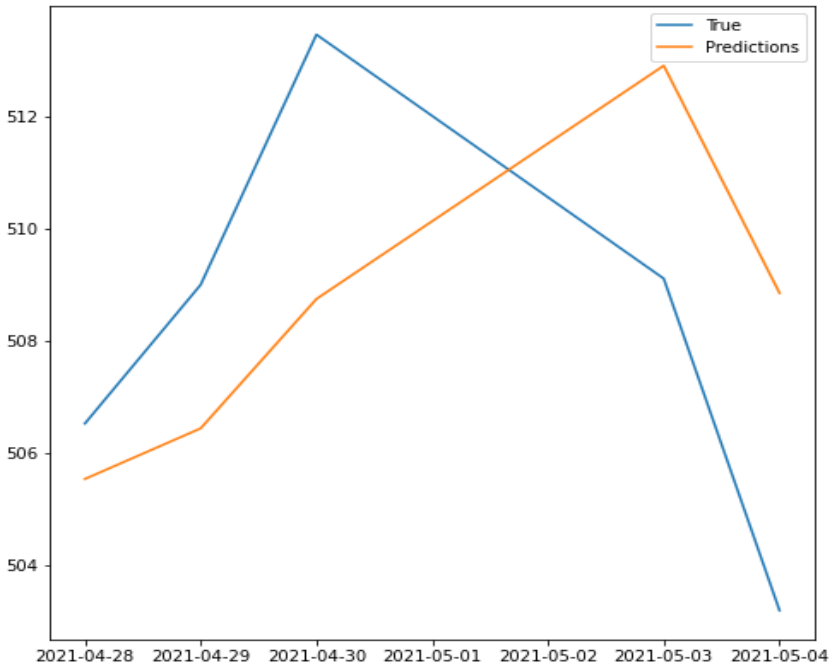


Рисунок 3.9 – Прогноз на 5 кроків вперед моделі AP(1)

Основні критерії якості прогнозу:

$$RMSE = 3.9122$$
$$MAPE = 0.61273452$$

$$Theil = 0.00384$$

З результатів видно, що прогнозовані дані трохи відрізняються від реальних, та значущість лагів зображених на рисунку ЧАКФ (рис.3.5) не відповідає числуодин, тому ми можемо спробувати збільшити порядок авторегресії.

Побудова моделі авторегресії першого порядку AP(4):

Проводячи аналіз ЧАКФ, можна помітити, що окрім першого лагу статистично значущим є четвертий лаг, тому буде побудована авторегресія четвертого порядку на рисунку 3.10:

```
Результати та рівняння моделі авторегресії порядку 4
y(k)= 30.629412004389646 +
0.8147446874961216 y( k - 1 )
0.09259820738511855 y( k - 2 )
-0.08528656940840622 y( k - 3 )
0.11764423709015054 y( k - 4 )
Параметри адекватності моделі:
SSR =  42237.67797426811
DW =  1.9503273716442497
R Squared =  0.8728367119646518
predicted=506.514854, true=506.519989
predicted=506.080240, true=509.000000
predicted=509.165340, true=513.469971
predicted=512.415174, true=509.109985
predicted=509.163123, true=503.179993
Якість прогнозу:
RMSE =  3.8412754935593245
MAPE =  [0.6032566  0.65422179 0.51869238 0.90496308 0.51842886]
Theil =  0.0037772879246484935
```

Рисунок 3.10 – Модель AP(4) для часового ряду «NETFLIX»

Рівняння моделі:

$$y(k) = 30.62941 + 0.814744 \cdot y(k-1) + 0.09259 \cdot y(k-2) - 0.08528 \cdot y(k-3) + 0.11764 \cdot y(k-4) + \varepsilon(k), \quad (3.11)$$

де $\varepsilon(k)$ – похибка моделі.

Зобразимо основні статистики для оцінки адекватності моделі AP(4) часового ряду «NETFLIX»:

$$R^2 = 0.872836$$
$$Sum\ squared\ resid = 42237.67$$
$$Durbin - Watson = 1.95032$$

Спробуємо спрогнозувати ріст цін акцій на 5 днів. Результати прогнозування на рисунку 3.12:

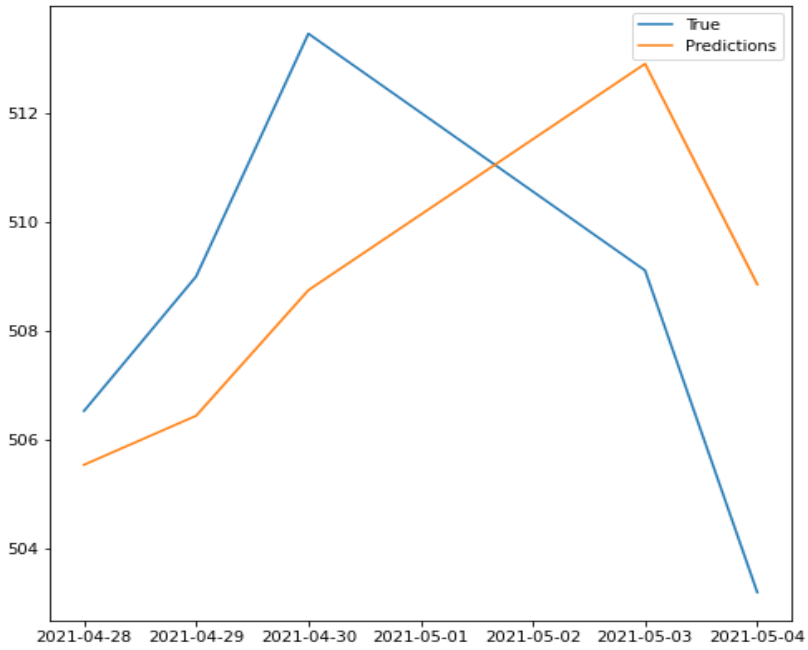


Рисунок 3.12 – Прогноз на 5 кроків вперед моделі AP(4)

Основні критерії якості прогнозу:

$$RMSE = 3.8412$$
$$MAPE = 0.51842$$
$$Theil = 0.00377$$

Отриманий прогноз є трохи кращим за попередній, хоча графік ЧАКФ (рис.3.5) чітко вказує на необхідність побудови моделі AP(5).

Зобразимо АКФ залишків на рисунку 3.13:

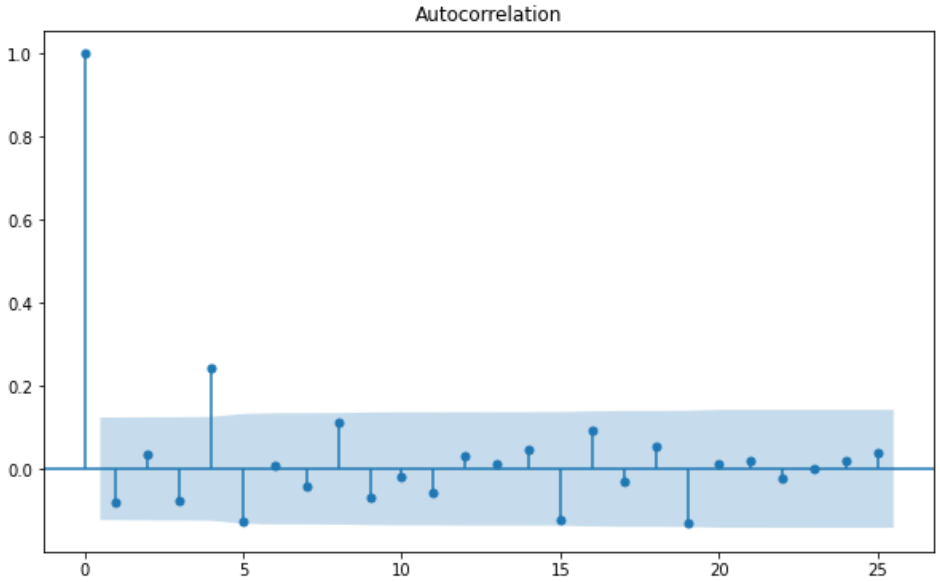


Рисунок 3.13 –АКФ залишків моделі AP(4) для часового ряду «NETFLIX»

Можна зробити висновок про доцільність побудови моделі ARMA(4,4).

Побудова моделі авторегресії першого порядку AP(5):

Проводячи аналіз ЧАКФ, можна помітити, що окрім першого лагу статистично значущим є п’ятий лаг, тому буде побудована авторегресія п’ятого порядку (рис.3.14):

```
Результати та рівняння моделі авторегресії порядку 5
y(k)= 35.278971343650895 +
0.8391031923724924 y( k - 1 )
0.07582451445004906 y( k - 2 )
-0.06743247423268053 y( k - 3 )
0.28766628032789365 y( k - 4 )
-0.2048700092887384 y( k - 5 )
Параметри адекватності моделі:
SSR =  40409.329183724905
DW =  1.9528902396062968
R Squared =  0.8766913595780785
predicted=506.190846, true=506.519989
predicted=505.420949, true=509.000000
predicted=509.936412, true=513.469971
predicted=511.487734, true=509.109985
predicted=509.241381, true=503.179993
Якість прогнозу:
RMSE =  3.6823391141154698
MAPE =  [0.64125158 0.73153399 0.61034084 0.79472893 0.5277305 ]
Theil =  0.003621754776062661
```

Рисунок 3.14 – Модель AP(5) для часового ряду «NETFLIX»

Рівняння моделі:

$$y(k) = 35.27897 + 0.83910 \cdot y(k-1) + 0.07582 \cdot y(k-2) - 0.06743 \cdot y(k-3) + 0.28766 \cdot y(k-4) - 0.20487 \cdot y(k-5) + \varepsilon(k), \quad (3.15)$$

де $\varepsilon(k)$ – похибка моделі.

Зобразимо основні статистики для оцінки адекватності моделі AP(5) часового ряду «NETFLIX»:

$$R^2 = 0.87669$$
$$Sum\ squared\ resid = 40409.32$$
$$Durbin - Watson = 1.95289$$

Спробуємо спрогнозувати ріст цін акцій на 5 днів. Результати прогнозування на рисунку 3.16:

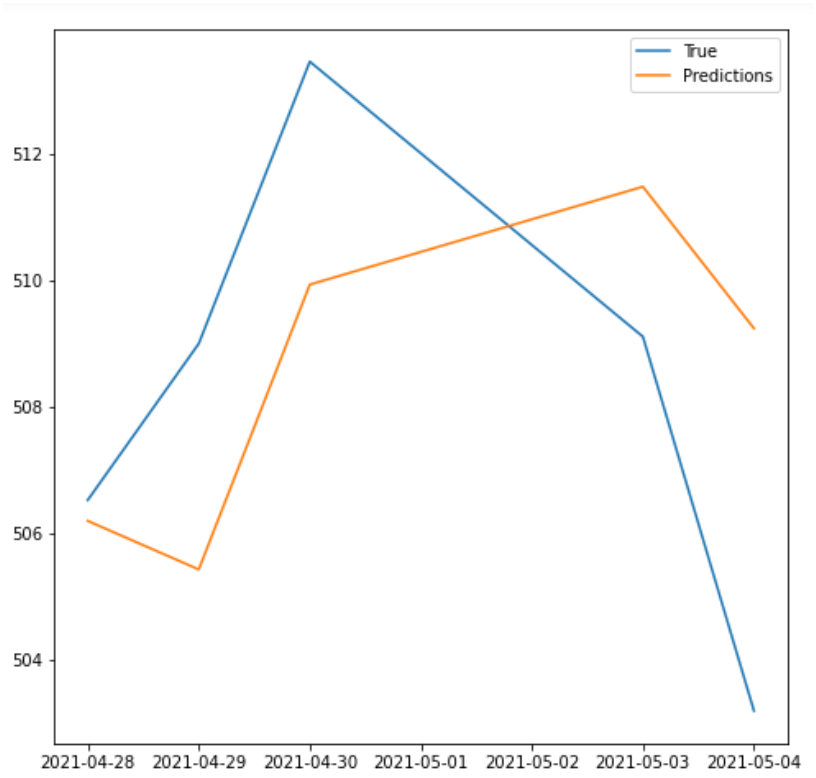


Рисунок 3.16, аркуш 58 – Прогноз на 5 кроків вперед моделі AP(5)

Основні критерії якості прогнозу:

$$RMSE = 3.68233$$
$$MAPE = 0.5237$$
$$Theil = 0.003621$$

Отриманий прогноз є досить задовільним, однак додавання сковзного середнього покращить результати моделювання.

Модель авторегресії з ковзним середнім АРКС(4,4):

Побудуємо модель авторегресії із ковзним середнім, включаючи 4 порядок авторегресії та четвертий порядок ковзного середнього (рис.3.17):

```
Результати та рівняння моделі ARMA порядку 4 4 :

y(k)= -902.3159672186498 +
9
0.2393169905275211 y( k - 1 )
-0.013555830824234848 y( k - 2 )
-0.032795594054308264 y( k - 3 )
0.6152227309489412 y( k - 4 )
+ 0.6234934891140202 ma( k- 1 )
+ 0.6363164838777607 ma( k- 2 )
+ 0.5744388788937261 ma( k- 3 )
+ 0.14089985910278593 ma( k- 4 )
Параметри адекватності моделі:
SSR =  39770.352445761724
DW =  2.0099348251100473
predicted=510.103596, true=506.519989
predicted=507.749113, true=509.000000
predicted=510.344768, true=513.469971
predicted=508.588528, true=509.109985
predicted=509.373027, true=503.179993
```

Рисунок 3.17 – Модель АРКС(4,4) для часового ряду «NETFLIX»

Рівняння моделі:

$$y(k) = -902.315 + 0.23931 \cdot y(k-1) - 0.01355 \cdot y(k-2) - 0.03279 \cdot$$

$$\cdot y(k-3) + 0.61522 \cdot y(k-4) + 0.62349 \cdot ma(k-1) + 0.63631 \cdot ma(k-2) + 0.574438 \cdot ma(k-3) + 0.14089 \cdot ma(k-4) \quad (3.18)$$

Запишемо основні статистики адекватності моделі авторегресії АРКС(4,4) часового ряду «NETFLIX»:

$$R^2 = 0.90509$$
$$Sum\ squared\ resid = 39770.3524$$
$$Durbin - Watson = 2.00993$$

Спрогнозуємо ціну акцій на 5 кроків уперед. Результати прогнозування на рисунку 3.19:

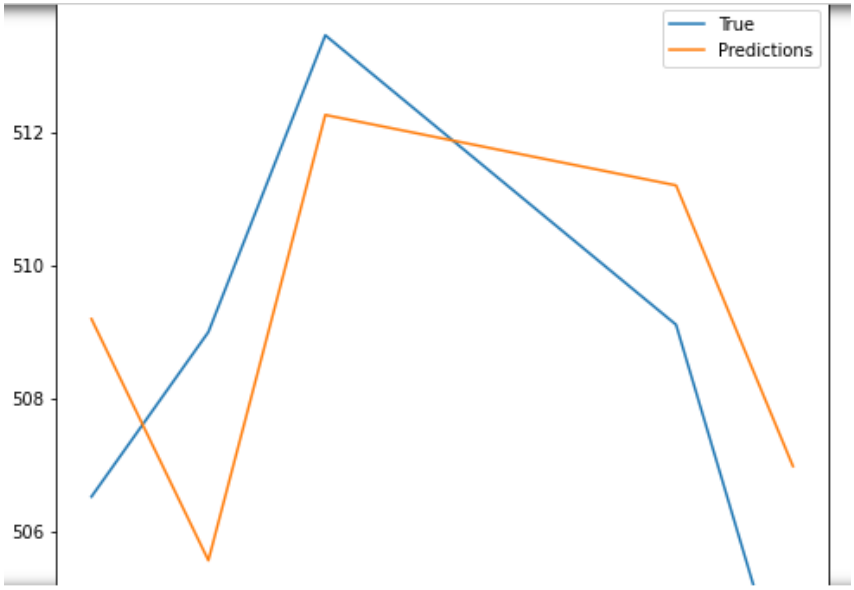


Рисунок 3.19 - Прогноз на 5 кроків вперед з використанням моделі АРКС(4,4)

Основні критерії якості прогнозу:

$$RMSE = 3.54399 \qquad MAPE = 0.54337 \qquad Theil = 0.00348$$

Перевіримо нелінійність у вихідному часовому ряду, застосовуючи BDS-тест на рисунку 3.20:

```
bds stat:  [34.74885546 36.48847964 38.57582952 41.71979798 45.89345104]
p-value:  [1.44227584e-264 1.68902477e-291 0.00000000e+000 0.00000000e+000
 0.00000000e+000]
Часовий ряд нелінійний
```

Рисунок 3.20 – Результати виконання BDS-тесту часового ряду «NETFLIX»

Можемо побачити, що гіпотеза про лінійність процесу відхиляється, отже є доцільним побудови моделей з трендом, так як всі p-value значення менші 0.05.

Модель інтегрованої авторегресії з ковзним середнім АРІКС(4,1,4):

Оскільки часовий ряд є нестаціонарним, буде доцільно використовувати модель АРІКС. Побудуємо інтегровану модель авторегресії із ковзним середнім, включаючи 4 порядок авторегресії та четвертий порядок ковзного середнього та продиференціюємо ряд один раз на рисунку 3.21:

```
Моделювання ряду NETFLIX ARIMA(4,1,4)
Результати та рівняння моделі ARIMA порядку 4 1 4 :

y(k)= 0.6036877070112345 +
9
-0.7822975683163123 y( k - 1 )
-0.16922907221270833 y( k - 2 )
0.5739942735116267 y( k - 3 )
0.7687461510246363 y( k - 4 )
+ 0.6342221645345891 ma( k- 1 )
+ 0.033292028893041614 ma( k- 2 )
+ -0.8379019391515665 ma( k- 3 )
+ -0.8296122542760641 ma( k- 4 )
Параметри адекватності моделі:
SSR =  40037.91769866158
DW =  2.0055141265525465
R Squared =  0.95530979791
```

Рисунок 3.21 – Модель АРІКС(4,1,4) для часового ряду «NETFLIX»

Рівняння моделі:

$$y(k) = 0.636877 - 0.7822975 \cdot y(k-1) - 0.1692290 \cdot y(k-2) + 0.573994 \cdot y(k-3) + 0.768746 \cdot y(k-4) + 0.6342221 \cdot ma(k-1) + 0.0332920 \cdot ma(k-2) - 0.8379019 \cdot ma(k-3) - 0.82961 \cdot ma(k-4) \quad (3.22)$$

Запишемо основні статистики адекватності моделі авторегресії АРІКС(4,1,4) часового ряду «NETFLIX»:

$$R^2 = 0.955309$$
$$Sum\ squared\ resid = 40037.917$$
$$Durbin - Watson = 2.005514$$

Спрогнозуємо ціну акцій на 5 кроків уперед. Результати прогнозування на рисунку 3.33:

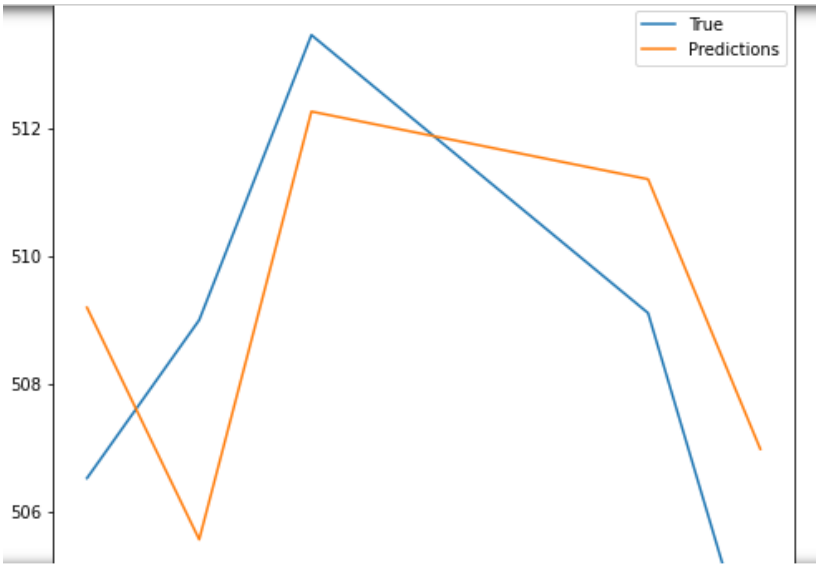


Рисунок 3.33 - Прогноз на 5 кроків вперед з використанням моделі АРІКС(4,1,4)

Основні критерії якості прогнозу:

$$RMSE = 5.49226 \qquad MAPE = 0.51599 \qquad Theil = 0.0054$$

Застосування моделі АРІКС трохи покращила наші попередні результати та відповідні статистики.

Модель інтегрованої авторегресії з ковзним середнім АРІКС(5,1,1):

Оскільки часовий ряд є нестаціонарним, буде доцільно використовувати модель АРІКС. Побудуємо інтегровану модель авторегресії із ковзним середнім, включаючи 5 порядок авторегресії, який є найбільш оптимальним з огляду на аналіз ЧАКФ (рис.3.5)та перший порядок ковзного середнього та продиференціюємо ряд один раз на рисунку 3.34:

```
Моделювання з найкраще підібраними параметрами ARIMA(p,d,q)
Результати та рівняння моделі ARIMA порядку 5 1 1 :

y(k)= 0.4327059987772963 +
7
0.8201171952521378 y( k - 1 )
0.06793715276832331 y( k - 2 )
-0.06875764679621679 y( k - 3 )
0.27604333904279676 y( k - 4 )
-0.23474178021811762 y( k - 5 )
+ -0.9999999822040176 ma( k- 1 )
Параметри адекватності моделі:
SSR =  39818.990840229235
DW =  1.9625608563170167
R Squared =  0.948889
predicted=508.889860, true=506.519989
predicted=504.517199, true=509.000000
predicted=510.593358, true=513.469971
predicted=511.019280, true=509.109985
```

Рисунок 3.34 – Модель АРІКС(5,1,1) для часового ряду «NETFLIX»

Рівняння моделі:

$$y(k) = 0.432705 + 0.82011 \cdot y(k-1) + 0.06793 \cdot y(k-2) - 0.006875 \cdot y(k-3) + 0.27604 \cdot y(k-4) - 0.234741 \cdot y(k-5) - 0.9999998 \cdot ma(k-1) \quad (3.35)$$

Запишемо основні статистики адекватності моделі авторегресії АРІКС(5,1,1) часового ряду «NETFLIX»:

$$R^2 = 0.968889$$

$$Sum\ squared\ resid = 41037.917$$

$$Durbin - Watson = 1.96256$$

Спрогнозуємо ціну акцій на 5 кроків уперед. Результати прогнозування на рисунку 3.36

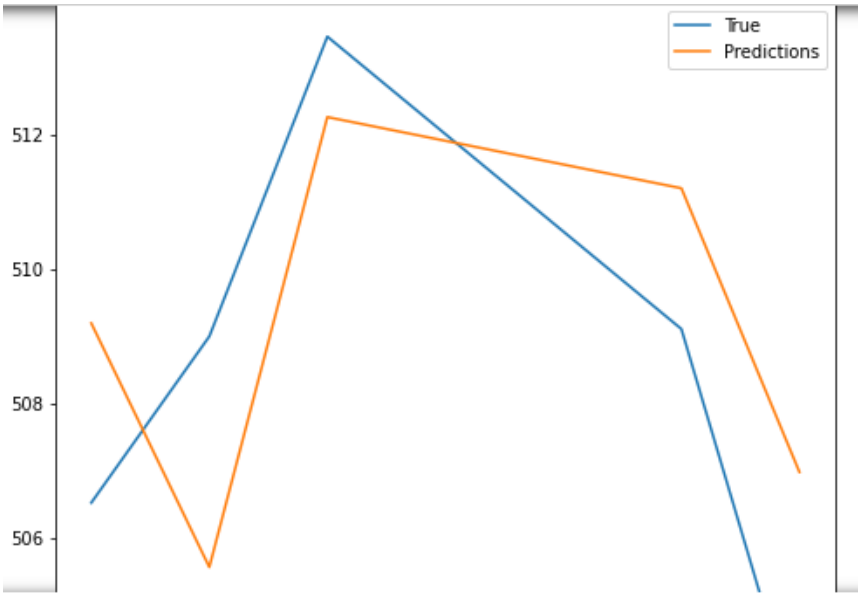


Рисунок 3.36 - Прогноз на 5 кроків вперед з використанням моделі APIKC(5,1,1)

Основні критерії якості прогнозу:

$$RMSE = 4.008855 \quad MAPE = 0.51189 \quad Theil = 0.00394$$

Після проведення моделювання даний набір параметрів вийшов найкращим, що збігається з попереднім статистичним аналізом, який проводився при дослідженні часового ряду.

Linguistic modeling

Метод лінгвістичного моделювання дозволяє застосовувати мінімальну необхідну попередню обробку даних перед використанням даного алгоритму. Необхідність диференціювання ряду для зведення його до стаціонарного є єдиною необхідною умовою для подальшої побудови математичної моделі.

Після приведення ряду до стаціонарного ми визначаємо критерій розбиття, тобто на скільки інтервалів нам розділити значення часового ряду та крок з яким ми будемо це робити. Крок визначається як різниця максимального та мінімального значення диференційованого ряду, поділеного на кількість інтервалів. Зазвичай оптимальною є кількість 25-35 інтервалів, де добре підходять літери англійського алфавіту.

Далі ми будуємо відповідні інтервали, співставляючи кожній літері відрізок значень на рисунку 3.37:

```
In [179]:
dictionary = initialize(minValue,maxValue,step,alphabet)
print(dictionary)

{'a': (-24.270004, -22.448465499999998), 'b': (-22.448465499999998, -20.626926999999995), 'c': (-20.626926999999995, -18.805388499999992), 'd': (-18.805388499999992, -16.98384999999999), 'e': (-16.98384999999999, -15.162311499999989), 'f': (-15.162311499999989, -13.340772999999988), 'g': (-13.340772999999988, -11.519234499999987), 'h': (-11.519234499999987, -9.697695999999986), 'i': (-9.697695999999986, -7.876157499999985), 'j': (-7.876157499999985, -6.054618999999983), 'k': (-6.054618999999983, -4.233080499999981), 'l': (-4.233080499999981, -2.4115419999999794), 'm': (-2.4115419999999794, -0.5900034999999779), 'n': (-0.5900034999999779, 1.2315350000000236), 'o': (1.2315350000000236, 3.053073500000025), 'p': (3.053073500000025, 4.874612000000027), 'q': (4.874612000000027, 6.696150500000028), 'r': (6.696150500000028, 8.51768900000003), 's': (8.51768900000003, 10.33922750000003), 't': (10.33922750000003, 12.16076600000003), 'u': (12.16076600000003, 13.982304500000032), 'v': (13.982304500000032, 15.803843000000033), 'w': (15.803843000000033, 17.625381500000035), 'x': (17.625381500000035, 19.446920000000038), 'y': (19.446920000000038, 21.26845850000004), 'z': (21.26845850000004, 23.089997000000043)}
```

Рисунок 3.37 – Інтервальне розбиття для часового ряду «NETFLIX»

Після чого ми перетворюємо наш числовий ряд у лінгвістичну послідовність, яка зображена на рисунку 3.38:

```
In [180]:
print(returnLinguisticChain(dictionary,diff))

['q', 'n', 'm', 'l', 'o', 'n', 'k', 'n', 'p', 'm', 'n', 'k', 'g', 'p', 'n', 'n', 'q', 'o', 'p', 'l', 'o', 'p', 'r', 'n',
'm', 'o', 'o', 'o', 'l', 'm', 'o', 'p', 'n', 'o', 'o', 'm', 'm', 'p', 'p', 'm', 'p', 'm', 'n', 'm', 'm', 'l', 'p', 'n',
'o', 'l', 'm', 'l', 'n', 'l', 'm', 'l', 'j', 'o', 'n', 'q', 'm', 'l', 'p', 'i', 'o', 'n', 'o', 'm', 'n', 'm', 'k', 'o',
'n', 'n', 'p', 'n', 'l', 'p', 'p', 'l', 'o', 'l', 'o', 'm', 'n', 'm', 'o', 'n', 'o', 'n', 'l', 'k', 'o', 'l', 'l', 'n',
'm', 'm', 'p', 'n', 'm', 'm', 'o', 'n', 'p', 'm', 'q', 'n', 'n', 'k', 'p', 'j', 'p', 'n', 'o', 'o', 'n', 'm', 'p', 'o',
'n', 'n', 'l', 'm', 'n', 'm', 'p', 'm', 'n', 'o', 'o', 'o', 'm', 'n', 'n', 'n', 'm', 'o', 'n', 'm', 'm', 'n', 'n', 'o',
'n', 'n', 'o', 'k', 'l', 'p', 'n', 'p', 'p', 'n', 'n', 'm', 'o', 'n', 'l', 'n', 'p', 'm', 'p', 'n', 'o', 'p', 'n', 'p',
'l', 'o', 'n', 'n', 'm', 'n', 'm', 'm', 'l', 'o', 'q', 'f', 'j', 'o', 'q', 'n', 'n', 'o', 'n', 'k', 'p', 'o', 'n', 'p',
'n', 'l', 'k', 'i', 'l', 'n', 'j', 'o', 'p', 'h', 'q', 'j', 'l', 'g', 's', 'i', 'e', 'w', 'a', 'p', 'l', 'q', 'l', 'm',
'u', 'k', 'r', 'j', 's', 'n', 'j', 'm', 'l', 't', 'p', 'q', 'n', 'n', 'p', 'm', 'm', 'o', 'm', 'j', 't', 'o', 'q', 'l',
'k', 't', 't', 'l', 'o', 'o', 'o', 'o', 'n', 'n', 'l', 'k', 'o', 'p', 'o', 'p', 'u', 'o', 'p', 'l', 'l', 'l', 'n', 'n',
'n', 'l', 'l', 'p', 'n', 'r', 'm', 'g', 'p', 'p', 'p', 'n', 'n', 'o', 'n', 'o', 'i', 'o', 'c', 'p', 'q', 't', 'l', 'r',
'n', 'o', 'n', 'n', 'j', 'n', 'n', 'n', 'n', 'p', 'l', 'm', 'j', 'm', 'o', 'l', 'p', 'n', 'x', 'm', 'm', 'm', 'w', 'p',
'k', 'j', 'p', 'o', 'n', 'n', 'n', 'n', 'q', 'm', 'p', 's', 'z', 'h', 'n', 'n', 'o', 'r', 'h', 'i', 'g', 'o', 'k', 'm',
'n', 'q', 'i', 'i', 'm', 'k', 'q', 'k', 'n', 'q', 'o', 'q', 'n', 'p', 'j', 'p', 'k', 'n', 'q', 'n', 't', 'n', 'k', 'k',
'm', 'k', 'q', 't', 'm', 'q', 'j', 'q', 'e', 'u', 'd', 'm', 'p', 'z', 'r', 'm', 'f', 'j', 'p', 'm', 'o', 'o', 'l', 'l',
'n', 'l', 'm', 'r', 'm', 'o', 'm', 's', 'n', 'k', 'm', 'q', 'm', 'k', 'm', 'l', 'n', 'o', 'n', 'm', 'o', 'l', 'k', 'n',
'm', 's', 'n', 'k', 'o', 'l', 'o', 'j', 'q', 'm', 'h', 'k', 'n', 'k', 'q', 's', 'q', 'q', 'o', 'p', 'p', 'h', 'j', 'j',
'p', 'q', 'n', 'n', 'o', 'm', 'o', 'o', 'm', 'n', 'p', 'n', 'l', 'j', 'm', 'q', 'm', 'i', 'o', 'r', 'k', 'l', 'o', 'q',
'i', 't', 'm', 's', 'k', 'q', 'q', 'p', 'k', 't', 'p', 'l', 'i', 'l', 'p', 'r', 'l', 'q', 'p', 's', 'l', 'r', 'n', 'n',
'm', 'm', 'j', 'q', 'm', 'l', 'n', 'm', 'k', 'p', 'o', 'n', 'p', 'z', 'k', 'l', 'l', 'l', 'q', 'm', 'g', 'n', 'l', 'o']
```

Рисунок 3.38, аркуш 63 – Лінгвістичний ланцюг часового ряду «NETFLIX»

Далі побудуємо матрицю переходів яка містить ймовірнісні значення переходу від однієї літери до іншої, що зображено на рисунку 3.39:

```
[[0.         0.         0.         0.         0.05263158 0.
  0.05263158 0.15789474 0.05263158 0.15789474 0.26315789 0.15789474
  0.         0.05263158 0.05263158 0.         0.         0.
  0.         0.         0.        ]
 [0.         0.03333333 0.         0.03333333 0.03333333 0.03333333
  0.1        0.13333333 0.16666667 0.13333333 0.1        0.13333333
  0.03333333 0.         0.06666667 0.         0.         0.
  0.         0.         0.        ]
 [0.         0.01851852 0.         0.01851852 0.03703704 0.09259259
  0.12962963 0.12962963 0.14814815 0.18518519 0.12962963 0.05555556
  0.03703704 0.         0.01851852 0.         0.         0.
  0.         0.         0.        ]
 [0.01369863 0.02739726 0.01369863 0.01369863 0.04109589 0.06849315
  0.12328767 0.1369863  0.15068493 0.16438356 0.10958904 0.05479452
  0.01369863 0.04109589 0.         0.01369863 0.         0.01369863
  0.         0.         0.        ]
 [0.         0.         0.         0.         0.02941176 0.07843137
  0.10784314 0.17647059 0.23529412 0.15686275 0.11764706 0.05882353
  0.01960784 0.         0.00980392 0.         0.         0.
  0.00980392 0.         0.        ]
 [0.         0.         0.         0.01449275 0.01449275 0.02898551
  0.11594203 0.11594203 0.30434783 0.15942029 0.13043478 0.07246377
  0.02898551 0.         0.         0.         0.         0.
  0.         0.         0.        ]
```

Рисунок 3.39 – Матриця переходів часового ряду «NETFLIX»

Запишемо основні статистики адекватності лінгвістичної моделі часового ряду «NETFLIX»:

$$R^2 = 0.9348136$$

$$Sum\ squared\ resid = 11.637$$

$$Durbin - Watson = 1.4256213$$

Спрогнозуємо ціну акцій на 5 кроків уперед. Результати прогнозування на рисунку 3.40:

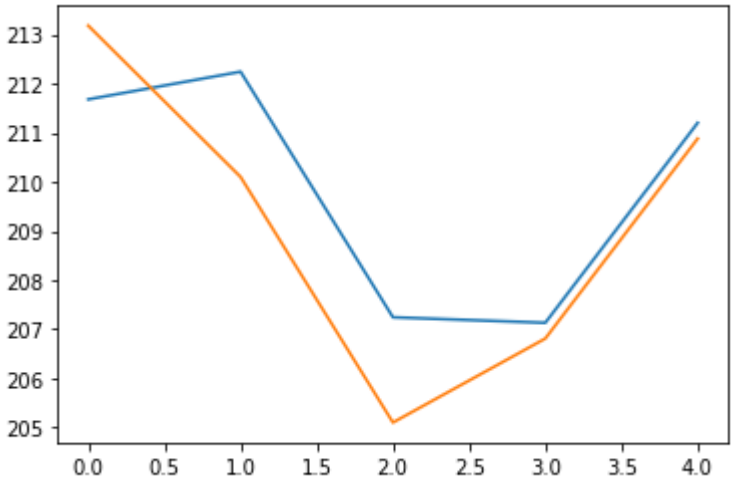


Рисунок 3.40 - Прогноз на 5 кроків вперед з використанням моделі лінгвістичного моделювання

Основні критерії якості прогнозу:

$$RMSE = 2.5935\ MAPE = 0.008\ \ Theil = 0.0036397$$

3.4 Аналіз та порівняння отриманих результатів

3.4.1 Порівняння побудованих моделей часового рядку NETFLIX

Спершу зобразимо порівняльну таблицю адекватності побудованих моделей часового ряду «NETFLIX» (табл.3.1):

Таблиця 3.1 – Таблиця для порівняння статистичних характеристик побудованих моделей для часового ряду «NETFLIX»

Модель часового ряду	R^2	Sum squared resid	Durbin – Watson
AP(1)	0.87447	43540.31	2.21994
AP(4)	0.872836	42237.67	1.95032
AP(5)	0.87669	40409.32	1.95289

АРКС(4,4)	0.90509	39770.35	2.00993
ARIMA(4,1,4)	0.955309	40037.91	2.005514
ARIMA(5,1,1)	0.968889	41037.91	2.017372
Linguistic modeling	0.9348136	11.637035	1.425621

Відобразимо таблицю, яка проілюструє порівняння якості прогнозів (табл.3.2).

Таблиця 3.2 – Порівняльна таблиця основних критеріїв якості прогнозу побудованих моделей для часового ряду «NETFLIX»

Модель часового ряду	$RMSE$	$MAPE$	$Theil$
AP(1)	3.9122	0.61273	0.00384
AP(4)	3.8412	0.51842	0.00377
AP(5)	3.68233	0.5237	0.003621
АРКС(4,4)	3.54399	0.54337	0.00348
ARIMA(4,1,4)	0.0084001	0.51599	0.0044297
ARIMA(5,1,1)	4.008855	0.511891	0.00394
Linguistic modeling	2.5935	0.008	0.0036397

3.5 Висновки до розділу

У третьому розділі були побудовані моделі економічних часових рядів акції компанії NETFLIX за період з 04.05.2020 до 04.05.2021. Були побудовані моделі АР(1), АР(4), АР(5). Також для процесу були побудовані моделі із ковзним середнім АРКС(4,4), АРКС(5,1). Також були побудовані інтегровані моделі АРІКС(4,1,4) та АРІКС(5,1,1). Був застосований новітній метод лінгвістичного моделювання, який показав гарні результати.

РОЗДІЛ 4. ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

4.1 Постановка завдання техніко-економічного дослідження

Програмний продукт розробляється з метою прогнозування нелінійних нестаціонарних процесів в економіці, оцінки параметрів моделей, прогнозу та їх подальшого аналізу.

4.2 Обґрунтування функцій та параметрів програмного продукту

4.2.1 Формування варіантів функцій

Виходячи з конкретних цілей, які реалізуються, виділимо основні функції:

F1: вибір мови програмування:

- вибір мови програмування Python в програмному продукті Jupyter Notebook;
- вибір мови програмування C++;
- вибір мови програмування R;

F2: оцінювання параметрів моделей:

- підбір моделей за допомогою вбудованих засобів;
- за допомогою власне розроблених алгоритмів;

F3: інтерфейс користувача:

- веб-інтерфейс;
- віконний інтерфейс;
- консольний інтерфейс у веб-додатку;

4.2.2 Варіанти реалізації основних функцій.

Складемо морфологічну карту варіантів реалізації основних функцій (рис. 4.1):

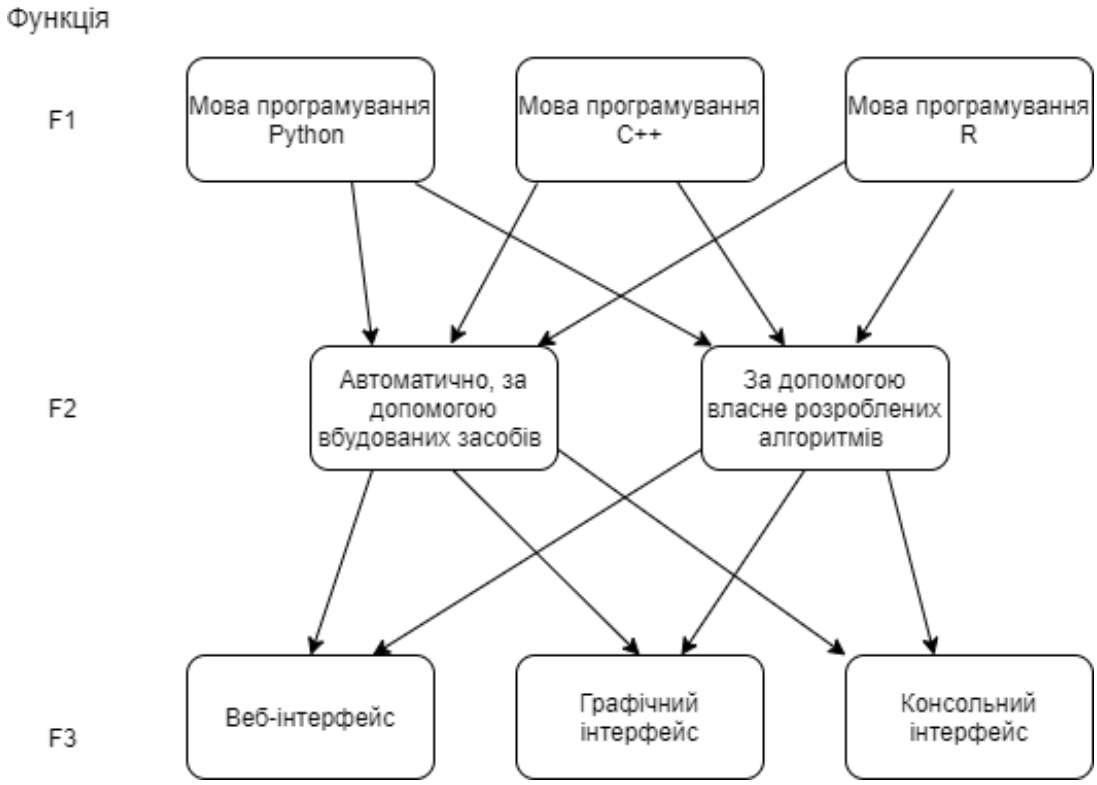


Рисунок 4.1, аркуш 71 – Морфологічна карта

Складемо морфологічну карту варіантів реалізації основних функцій (рис. 4.1):

Таблиця 4.1 – позитивно-негативна матрицю варіантів реалізації основних функцій

Основні функції	Варіанти реалізації	Переваги	Недоліки
F1	а)	Простий синтаксис з великою кількістю вбудованих бібліотек	Маленька швидкодія, динамічна типізація, що ускладнює процес написання програми
	б)	Швидкодія, строга типізація	Важка робота з пам’яттю, необхідність реалізації власних алгоритмів

Продовження таблиці 4.1

	в)	Наявність бібліотечних засобів роботи з даними, простота реалізації алгоритмів	Невисока швидкодія, обмеженість засобів опису даних
F2	а)	Простота реалізації	Потреба у попередній підготовці даних
	б)	Можливість модифікації та покращення алгоритму	Складність реалізації, потребує багато часових ресурсів на реалізацію
F3	а)	Простота реалізації	Необхідний сумісний веб-браузер
	б)	Інтуїтивність при використанні, не потребує сторонніх програм	Складність реалізації, потребує багато часових ресурсів на реалізацію
	в)	Простота реалізації, можливість додавання візуалізації	Необхідність встановлення програмного засобу

Проаналізувавши позитивно-негативну матрицю варіантів реалізації основних функцію можемо одразу виключити наступні варіанти: F1 б) Вибір мови програмування C++, F2 б) за допомогою власне розроблених алгоритмів, F3 б) віконний інтерфейс, F3 а) веб-інтерфейс.

Варіанти, що залишилися:

- F1 а) + F2 а) + F3 в)
- F1 в) + F2 а) + F3 в)

4.3 Обґрунтування системи параметрів

На основі змісту основних функцій, виділимо наступну систему параметрів: X1 – потенційний об’єм коду, X2 – час обробки даних, X3 – час на освоєння мови програмування, X4 – рівень складності алгоритму, X5 – затрати оперативної пам’яті. Для функції F1 застосовуються параметри X1, X2, X3, для функції F2 застосовується параметр X4, для функції F3 застосовується параметр X5. Встановимо граничні значення кожного параметру (кращі, середні, гірші) і занесемо у таблицю (табл. 4.2):

Таблиця 4.2 – Основні параметри дослідження

Назва параметру	Умовне позначення	Одиниці виміру	Значення параметру		
			Краще	Середнє	Гірше
потенційний об’єм коду	X1	Кількість рядків коду	300	600	1000
час обробки даних	X2	Секунди	1	5	10
час на освоєння мови програмування	X3	Години	3	10	24
рівень складності алгоритму	X4	Частка від 1	0.2	0.6	1
затрати оперативної пам’яті	X5	Мб	16	32	64

За даними таблиці будуємо графічні характеристики параметрів X1-X5 (рис. 4.2-4.6). На осі ординат зображено кількість балів, а на осі абсцис – відповідні значення параметру:

За даними таблиці 4.2 будуються графічні характеристики параметрів – рисунок 4.2 – рисунок 4.5.

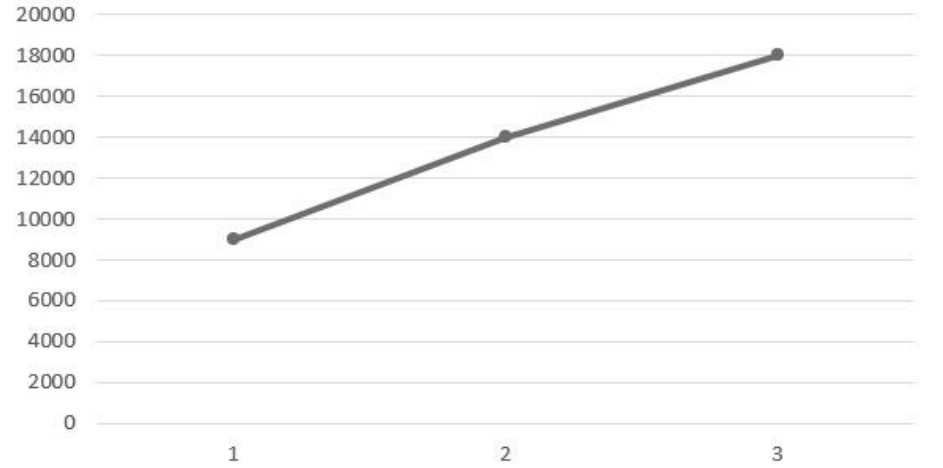


Рисунок 4.2, аркуш 72 - X1, швидкодія мови програмування

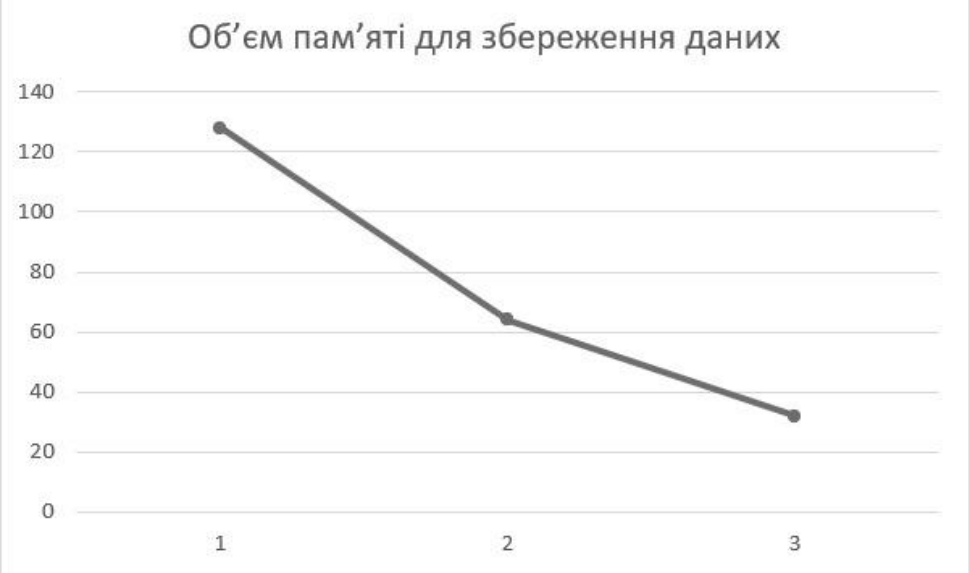


Рисунок 4.3, аркуш 72 - X2, об’єм пам’яті для збереження даних

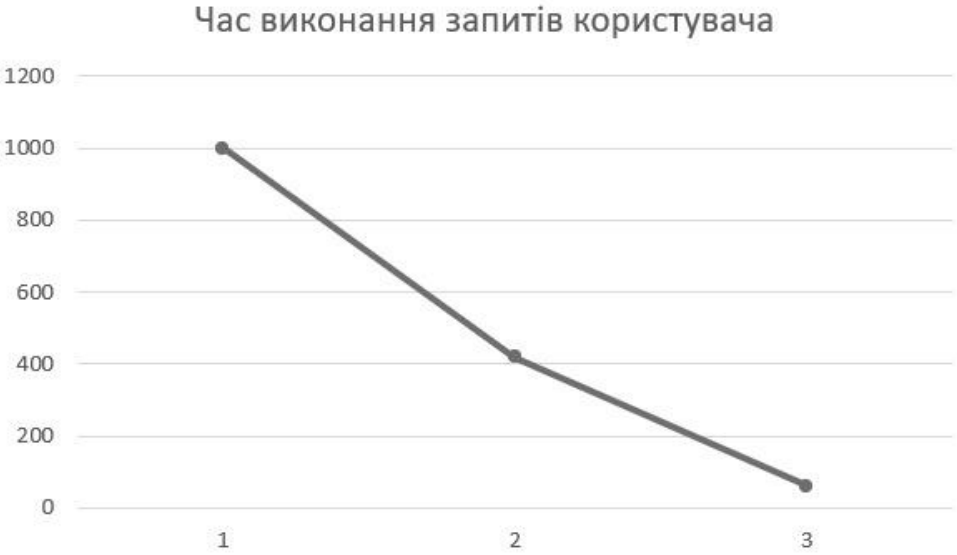


Рисунок 4.4, аркуш 72 - X3, час виконання запитів користувача

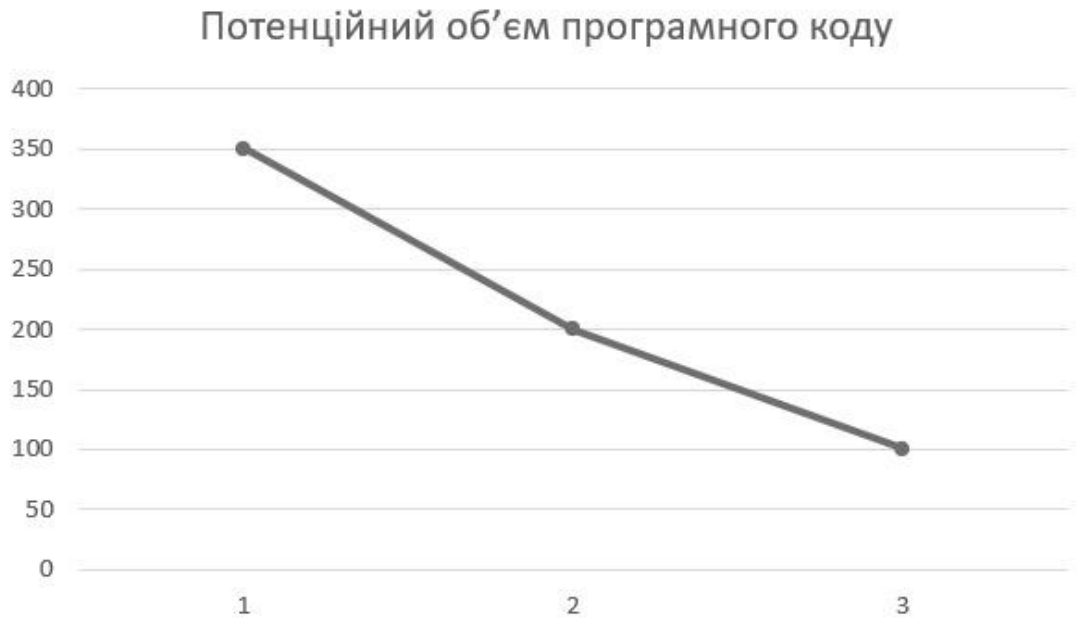


Рисунок 4.5, аркуш 72 - X4, потенційний об’єм програмного коду

4.4 Визначення коефіцієнтів значимості параметрів

Вагомість параметрів визначається методом попарного їх порівняння на основі результатів ранжування експертами та попарного порівняння параметрів. Наведемо результати експертного ранжування (табл. 4.3):

Таблиця 4.3 – Результати ранжування показників

Параметр	Од. виміру	Ранг показника по оцінці експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
		1	2	3	4	5	6	7			
X1	Рядки коду	3	1	5	3	3	3	2	20	-1	1
X2	Секунди	5	4	4	5	4	4	5	31	10	100
X3	Години	4	5	3	4	5	5	4	30	9	81
X4	Частка від 1	2	3	1	2	1	1	3	13	-8	64
X5	Мб	1	2	2	1	2	2	1	11	-10	100
		15	15	15	15	15	15	15	105	0	346

У таблиці 4.3 найбільш вагомим є коефіцієнт 1, найменш вагомий – 5.

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 105, \quad (4.1)$$

де N – число експертів;

n – кількість параметрів

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 21 \quad (4.2)$$

де n – кількість параметрів

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T. \quad (4.3)$$

де R_i - загальна сума рангів, T - середня сума рангів

Сума відхилень по всім параметрам повинна дорівнювати 0;

г) загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 346. \quad (4.4)$$

де N – число експертів

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 346}{7^2 \cdot (5^3 - 5)} \approx 0.706 > W_k = 0.67, \qquad (4.5)$$

де N – число експертів

Отже, експертне ранжування достовірне. Результати порівняння параметрів занесемо у таблицю (табл. 4.4):

Таблиця 4.4 – Експертне порівняння параметрів

Параметри	Експерти							Підсумкова оцінка	Числове значення
	1	2	3	4	5	6	7		
X1,X2	>	>	<	>	>	>	>	>	1.5
X1,X3	>	>	<	>	>	>	>	>	1.5
X1,X4	<	>	<	<	<	<	>	<	0.5
X1,X5	<	>	<	<	<	<	<	<	0.5
X2,X3	<	>	<	<	>	>	<	<	0.5
X2,X4	<	<	<	<	<	<	<	<	0.5
X2,X5	<	<	<	<	<	<	<	<	0.5
X3,X4	<	<	<	<	<	<	<	<	0.5
X3,X5	<	<	<	<	<	<	<	<	0.5
X4,X5	<	<	>	<	>	>	<	<	0.5

Виконаємо розрахунок вагомості параметрів, результати занесемо у таблицю (табл. 4.5):

Таблиця 4.5 – Розрахунок вагомості параметрів

Параметри	Параметри					Перший крок		Другий крок		Третій крок	
	X1	X2	X3	X4	X5	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$
X1	1.0	1.5	1.5	0.5	0.5	5	0.2	22	0.191	100	0.19
X2	0.5	1.0	0.5	0.5	0.5	3	0.12	14	0.122	64.5	0.123
X3	0.5	1.5	1.0	0.5	0.5	4	0.16	17.5	0.152	80.25	0.153
X4	1.5	1.5	1.5	1.0	0.5	6	0.24	27.5	0.239	124.75	0.238
X5	1.5	1.5	1.5	1.5	1.0	7	0.28	34	0.296	155.5	0.296
Сума						25	1	115	1	525	1

4.5 Оцінка рівня якості варіантів реалізації програмного продукту

Виконаємо розрахунок показників рівня якості варіантів реалізації основних функцій ПП, результати занесемо у таблицю (табл. 4.6):

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	а)	X1	500	6.5	0.19	1.235
		X2	7	3	0.123	0.369
		X3	8	6	0.153	0.918
	в)	X1	500	6.5	0.19	1.235
		X2	5	5	0.123	0.615
		X3	15	3.5	0.153	0.536
F2	а)	X4	0.2	10	0.238	2.38
F3	в)	X5	16	10	0.296	2.96

За даними таблиці визначимо показники рівня якості кожного з варіантів ПП:

$$K_{я1} = 1.235 + 0.369 + 0.918 + 2.38 + 2.96 = 7.862 \qquad (4.6)$$

$$K_{я2} = 1.235 + 0.615 + 0.536 + 2.38 + 2.96 = 7.726 \qquad (4.7)$$

Отже, найбільш якісним є перший варіант, оскільки його показник рівня якості – найбільший.

4.6 Економічний аналіз варіантів програмного продукту

Усі варіанти реалізації основних функцій включають в себе 3 наступні завдання: 1) Розробка проекту програмного продукту; 2) Програмування моделей-кандидатів; 3) Виконання прогнозу обраних процесів.

Крім того, кожний з варіантів має додаткове завдання. Далі наведені варіанти розробки додаткового завдання: 4.1) Використання вбудованих бібліотек аналізу Python для виконання обчислень; 4.2) Виконання моделювання на основі великого масиву даних за допомогою вбудованих бібліотек R.

За ступенем новизни завдання 1) відноситься до групи А, завдання 2) та 3) – до групи Б, завдання 4.1) та 4.2) – до групи В. За складністю алгоритми, які використовуються в завданнях 1), 2) та 3) належать до групи 1, в завданнях 4.1) та 4.2) – до групи 3. За видом інформації завдання 1), 4.1) та 4.2) відносяться до НДІ (нормативно-довідкова), завдання 2) та 3) відносяться до ПІ (перемінної). Обчислимо загальну трудомісткість завдань:

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань. Загальна трудомісткість обчислюється як:

$$T_o = T_p * K_{\text{П}} * K_{CK} * K_M * K_{CT} * K_{CT.M} \quad (4.8)$$

де T_P – трудомісткість розробки ПП;

$K_{\text{П}}$ – поправочний коефіцієнт;

K_{CK} – коефіцієнт на складність вхідної інформації;

K_M – коефіцієнт рівня мови програмування;

K_{CT} – коефіцієнт використання стандартних модулів і прикладних програм;

$K_{CT.M}$ – коефіцієнт стандартного математичного забезпечення.

$$T_1 = 90 \cdot 1.7 \cdot 1 \cdot 1 \cdot 1 \cdot 1 = 153 \text{ людино} - \text{днів} \quad (4.9)$$

$$T_2 = 64 \cdot 2.02 \cdot 1 \cdot 1 \cdot 0.8 \cdot 1 = 103.424 \text{ людино} - \text{днів} \quad (4.10)$$

$$T_3 = 64 \cdot 2.02 \cdot 1 \cdot 1 \cdot 0.8 \cdot 1 = 103.424 \text{ людино} - \text{днів} \quad (4.11)$$

$$T_{4.1} = 12 \cdot 0.6 \cdot 1 \cdot 1 \cdot 0.8 \cdot 0.8 = 4.608 \text{ людино} - \text{днів} \quad (4.12)$$

$$T_{4.2} = 12 \cdot 0.6 \cdot 1 \cdot 1 \cdot 0.8 \cdot 1 = 5.76 \text{ людино} - \text{днів} \quad (4.13)$$

Визначимо повну трудомісткість кожного з варіантів:

$$T_1 = 153 + 103.424 + 103.424 + 4.608 = 364.456 \text{ людино} - \text{днів} \quad (4.14)$$

$$T_2 = 153 + 103.424 + 103.424 + 5.76 = 365.608 \quad (4.15)$$

Вважаємо, що робочий день складає 8 годин, в тижні 5 робочих днів. В розробці бере участь один програміст з окладом 20000 грн та тестувальник з окладом 15000 грн. Визначимо середню заробітну плату за годину (грн):

$$C_{\text{ч}} = \frac{20000 + 15000}{2 \cdot 22 \cdot 8} = 99.43 \quad (4.16)$$

Тоді заробітна плата кожного з варіантів реалізації (грн):

$$C_{\text{ЗП1}} = 99.43 \cdot 8 \cdot 364.456 = 289902.88 \quad (4.17)$$

$$C_{\text{ЗП2}} = 99.43 \cdot 8 \cdot 365.608 \ = 290819.22 \quad (4.18)$$

Відрахування на соціальне страхування (грн):

$$C_{\text{ВІД1}} = 289902.88 \cdot 0.22 = 63778.63 \quad (4.19)$$

$$C_{\text{ВІД2}} = 290819.22 \cdot 0.22 = 63980.23 \quad (4.20)$$

Розрахуємо витрати на оплату однієї машино-години. Враховуючи, що вона обслуговує одного спеціаліста з окладом 20000 грн та другого з окладом 15000 грн з коефіцієнтом зайнятості $K_З = 0.4$, то для двох машин отримаємо:

$$C_r = 12 \cdot 20000 \cdot 0.4 + 12 \cdot 15000 \cdot 0.4 = 168000 \text{ грн} \quad (4.21)$$

Враховуючи додаткову заробітну плату (25%):

$$C_{\text{ЗП}} = 168800 \cdot (1 + 0.25) = 210000 \text{ грн} \quad (4.22)$$

Відрахування на соціальне страхування:

$$C_{\text{ВІД}} = 210000 \cdot 0.22 = 46200 \quad (4.23)$$

Розрахуємо амортизаційні підрахунки (амортизація 25%, вартість ЕОМ - 30000 грн):

$$C_A = 1.15 \cdot 0.25 \cdot 30000 = 8625 \quad (4.24)$$

Розрахуємо витрати на ремонт та профілактику:

$$C_{\text{Р}} = 1.15 \cdot 30000 \cdot 0.05 = 1725 \quad (4.25)$$

Розрахуємо ефективний годинний фонд часу ПК за рік:

$$\text{Т}_{\text{ЕФ}} = (365 - 104 - 11 - 12) \cdot 8 \cdot 0.8 = 1523.2 \quad (4.26)$$

Обрахуємо витрати на електроенергію (з ПДВ):

$$C_{\text{ел}} = 1523.2 \cdot 0.6 \cdot 2 \cdot 3{,}52 = 6433.99 \text{ грн.} \quad (4.27)$$

Накладні витрати рівні:

$$C_{\text{Н}} = 30000 \cdot 0.67 = 20100 \text{ грн.} \quad (4.28)$$

Обчислимо річні експлуатаційні витрати:

$$C_{\text{ЕКС}} = 210000 + 46200 + 8625 + 1725 + 6433.99 + 20100 = \\ = 293083.99 \quad (4.29)$$

Тоді собівартість однієї машино-години ЕОМ дорівнюватиме:

$$C_{\text{М-Г}} = \frac{293083.99}{1523.2} = 192.4 \text{ грн/год.} \quad (4.30)$$

Враховуючи, що всі роботи ведуться на ЕОМ, витрати на оплату машинного часу:

$$C_{\text{М}} = 192.4 \cdot 8 \cdot 364.456 = 560970.68 \quad (4.31)$$

$$C_{\text{М}} = 192.4 \cdot 8 \cdot 365.608 = 562743.83 \quad (4.32)$$

Накладні витрати відповідно:

$$C_{\text{Н}} = 560970.68 \cdot 0.67 = 375850.35 \quad (4.33)$$

$$\text{C}_{\text{Н}} = 562743.83 \cdot 0.67 = 377038.06 \quad (4.34)$$

Розрахуємо повну вартість розробки за варіантами:

$$C_{\text{ПП}} = 289902.88 + 63778.63 + 554556.25 \ + 375850.35 \ = 1284088.11 \quad (4.35)$$

$$C_{\text{ПП}} = 290819.22 + 63980.23 + 556309.13 + 377038.06 = 1288146.41 \quad (4.36)$$

4.7 Вибір кращого варіанту програмного продукту техніко-економічного рівня.

Розрахуємо коефіцієнт техніко-економічного рівня:

$$K_{\text{ТЕР1}} = \frac{14.622}{1284088.11} = 1.138 \cdot 10^{-5}, \quad (4.37)$$

$$K_{\text{ТЕР2}} = \frac{13.006}{1288146.41} = 1.009 \cdot 10^{-5} \quad (4.38)$$

З техніко-економічної точки зору перший варіант є більш ефективним.

4.8 Висновки до розділу

У ході виконання функціонально-вартісного аналізу програмного продукту були проаналізовані декілька підходів його реалізації.

На першому етапі був виконаний технічний аналіз продукту. Були сформульовані функції, які повинна виконувати програма, альтернативні варіанти їх реалізації. Для кожного з варіантів реалізації були обчислені основні параметри, виконане експертне оцінювання. Обчислений коефіцієнт технічного рівня, завдяки якому відбувся вибір найкращої альтернативи.

На другому етапі варіанти реалізації основних функцій були досліджені на оптимальність з економічної точки зору. Для кожної з

досліджуваних альтернатив знайдені такі параметри, як трудомісткість, витрати на трудові ресурси, витрати на матеріальне обладнання, накладні витрати, витрати на оплату машинного часу. За результатами аналізу визначено, що найкращою альтернативою є наступна: Мова програмування – Python з програмним продуктом Jupyter Notebook; Оцінювання параметрів моделей – автоматично, за допомогою вбудованих засобів; Інтерфейс користувача – консольний інтерфейс у веб-браузері.

ВИСНОВКИ

У дипломній роботі було досліджено найбільш важливе та цікаве питання – прогнозування економічних даних, які в свою чергу безпосередньо впливають на фінансове життя цілих країн та всього світу.

Побудова моделей часового ряду здійснювалася зза допомогою основних методів опису нелінійних процесів, а також за допомогою лінгвістичного моделювання. Було побудовані зрозумілі таблиці оцінок моделей, що використовуються. Також було здійснено прогнозування на декілька діб вперед, що дало можливість оцінити точність та здастність моделей прогнозувати значення у майбутньому.

Для проведення аналізу та прогнозування був розроблений власний програмний продукт, який має зручні способи роботи з даними та використовує інтуїтивно зрозумілі графічні інструменти зображення здійснених прогнозів.

Отримані результати доводять досить високу точність опису процесу та непогано здійснюють короткостроковий прогноз економічних даних різної природи. Подальші покращення можливі за умови використання методів “глибокого навчання”, застосуванню нейронних мереж та побудови інтелектуальної інформаційної системи аналізу отриманої інформації. Окремим шляхом покращення є більш детальний аналіз вхідного ряду, та застосування спеціалізованих програмних засобів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Inflation calculator. URL: https://www.in2013dollars.com.
2. Annual GDP per pesrong growth in Ukraine from 1989 till 2020 years. URL: https://www.google.com/publicdata/explore?ds=d5bncppjof8f9_#!ctype=l&strail=false&bcs=d&nselm=h&met_y=ny_gdp_pcap_kd_zg&scale_y=lin&ind_y=false&rdim=country&idim=country:UKR&ifdim=country&hl=uk&dl=uk&ind=false
3. Бідюк П. І, Романенко В. Д., Тимощук. О. Л. Аналіз часових рядів: підручник / НТУУ КПІ ім. Ігоря Сікорського. Київ: ВПК "Політехніка", 2013. 599 с.
4. Байєсівські мережі в технологіях інтелектуального аналізу даних : публікація / Наукова електронна бібліотека періодичниз видань НАН України,2010.10с.URL:http://dspace.nbuv.gov.ua/bitstream/handle/123456789/56141/13-Bidyuk.pdf?sequence=1
5. Івахненко А. Г. Індуктивний метод самоорганізації моделей складних систем: навч. посібник. Київ: Наукова думка, 1982. 245 с.
6. Баклан І.В., Селін Ю.М., Шулькевич Т.В. Математичні моделі прогнозування часових рядів різної природи // Вестн. Херсонского национального техн. ун-та. - Херсон: ХНТУ, 2014. - Вып. 3 (50).– С.213-218
7. Коротка характеристика МГУА, 2008 URL: http://www.mgua.irtc.org.ua/ukr/index.php?page=gmdh
8. Бідюк П. І., Гожий О. П., Проектування комп'ютерних інформаційних систем підтримки прийняття рішень , 2010, 335 с.
9. Бідюк П.І., Трухан С.В. Прогнозування актуарних процесів за допомогою узагальнених лінійних моделей, 2014, 7с. URL: http://old.bulletin.kpi.ua/files/2014-2-2.pdf
10. Зайченко Ю.П., Кебкал О.Г., Крачковський В.Ф. Нечіткий метод групового врахування аргументів та його застосування в задачах прогнозування макроекономічних показників. //Наукові вісті НТУУ «КПІ», №2, 2000р. с.18-26.
11. Generalized linear model. URL:https://towardsdatascience.com/generalized-linear-models-9ec4dfe3dc3f
12. Review on trajectory similarity measures 2015,6c. URL:https://www.researchgate.net/publication/293823957_Review_on_trajectory_similarity_measures
13. Юрченко М. Є Прогнозування та аналіз часових рядів. Методичні вказівки до практичних занять та самостійної роботи студентів спеціальності 051 «Економіка» освітня програма «Економічна кібернетика», «Економічна аналітика». 2018. – 88 с.

14. Селін Ю.М. Системний аналіз екологічно небезпечних процесів різної природи / Ю.М. Селін // Системні дослідження та інформаційні технології. — 2007. — № 2. — С.22–32
15. Селін Ю.М. Лінгвістичне моделювання динамічних процесів різної природи.
16. Holt-Winter`s seasonal smoothing URL: https://otexts.com/fpp2/holt-winters.html
17. Pena D., Rodriguez J. Detecting nonlinearity in time series by model selection criteria. *International Journal of Forecasting*, 2005. 18 p.
18. Berry M. J. A., Linoff G. S. Data Mining Techniques. 2nd edition. New York : Wiley Publishing, Inc., 2004. 670 p.
19. Bidyuk P., Prosyankina-Zharova T., Terentiev O. Modelling Nonlinear Nonstationary Processes in Macroeconomy and Finances. Advances in Computer Science for Engineering and Education. ICCSEEA 2018. Advances in Intelligent Systems and Computing / Ed. Hu Z., Petoukhov S., Dychka I., He M. Cham : Springer, 2019. Vol. 754. P. 735–745. URL: http://doi.org/10.1007/978-3-319-91008-6_72
20. Коршевнюк Л. О., Терентьєв О. М., Бідюк П. І. Методика побудови математичних моделей динамічних процесів. Системний аналіз та інформаційні технології(SAIT 2013) : матеріали 15-ї міжнар. наук.-техн. конф. (Київ, 27–31 трав. 2013 р.). Київ : ННК “ІПСА” НТУУ “КПІ”, 2013. С. 288–289.
21. Снитюк В. Є. Прогнозування. Моделі. Методи. Алгоритми: навч. посіб. Київ : Маклаут, 2008. 364 с.

ДОДАТОК А. ЛІСТИНГ ПРОГРАМИ

```
import pandas as pd
import string
import numpy
import math
from pandas import read_csv
from pandas import DataFrame
from matplotlib import pyplot
from statsmodels.tsa.stattools import adfuller
from numpy import log
from numpy import zeros
from numpy import sum
from numpy import set_printoptions

series = read_csv('FB.csv', header=0, parse_dates=[0], index_col=0,
squeeze=True)
df = DataFrame(series)

df.drop('Open',inplace=True,axis=1)
df.drop('High',inplace=True,axis=1)
df.drop('Low',inplace=True,axis=1)
df.drop('Close',inplace=True,axis=1)
df.drop('Volume',inplace=True,axis=1)
df.columns =['Price']

df.describe()

print(df.hist(width=10))

df.head()
data1 = df.iloc[:,0].values
result = adfuller(log(data1))
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
print('Critical Values:')
for key, value in result[4].items():
   print('\t%s: %.3f' % (key, value))

diff = []
X = df.values
for i in range(1, len(X)):
        value = X[i] - X[i - 1]
        diff.append(value)
pyplot.plot(diff)
```

```
result = adfuller(diff)
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
print('Critical Values:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))

maxValue = (max(diff))[0]
minValue = (min(diff))[0]
alphabetSize = 26
print(minValue)
print(maxValue)
step=(maxValue-minValue)/alphabetSize
print(step)

alphabet = ('a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z')
print(alphabet)

def initialize(minValue,maxValue,step,alphabet):
    empty={}
    start = minValue
    end = step
    for i in range(len(alphabet)):
        case = {alphabet[i]:(start, start+step)}
        empty.update(case)
        start = start + step
    return empty

dictionary = initialize(minValue,maxValue,step,alphabet)
print(dictionary)

def listToString(s):

    # initialize an empty string
    str1 = ""

    # traverse in the string
    for ele in s:
        str1 += ele

    # return string
    return str1

def levenshtein(seq1, seq2):
```

```
    size_x = len(seq1) + 1
    size_y = len(seq2) + 1
    matrix = numpy.zeros ((size_x, size_y))
    for x in range(size_x):
        matrix [x, 0] = x
    for y in range(size_y):
        matrix [0, y] = y

    for x in range(1, size_x):
        for y in range(1, size_y):
            if seq1[x-1] == seq2[y-1]:
                matrix [x,y] = min(
                    matrix[x-1, y] + 1,
                    matrix[x-1, y-1],
                    matrix[x, y-1] + 1
                )
            else:
                matrix [x,y] = min(
                    matrix[x-1,y] + 1,
                    matrix[x-1,y-1] + 1,
                    matrix[x,y-1] + 1
                )
    #print (matrix)
    return (matrix[size_x - 1, size_y - 1])

def returnValueFromLetter(dictionary, letter,step):

    return (dictionary[letter][1] - 0.5*step );

def returnLinguisticPosition(alphabet,dictionary,value):
    for key in alphabet:
        if(dictionary[key][0]<=value and dictionary[key][1]>value):
            return key;
    return ' '

def returnLinguisticChain(dictionary, data):
    new_arr = []
    for i in range(len(data)):
        for key in alphabet:
            if(dictionary[key][0]<=data[i] and dictionary[key][1]>data[i]):
                new_arr.append(key)
                break;
    return new_arr

def createTransitionMatrix(alphabet,transformedTS):
```

```
    arr = numpy.zeros((len(alphabet),len(alphabet)))
    for i in range(len(alphabet)):
        counter=0
        for j in range(len(alphabet)):
            for k in range(len(transformedTS)-1):
                if(transformedTS[k]==alphabet[i] and
transformedTS[k+1]==alphabet[j]):
                    counter=counter+1
            arr[i][j]=counter
            counter=0
    return arr

def probabilityMatrix(matrix):
    matr = matrix/matrix.sum(axis=1)[:,None]
    matr = numpy.nan_to_num(matr)
    return matr

def
buildPrediction(minValue,maxValue,step,alphabet,arr,windowSize,baseSize,real_a
rr):
    pred_values=[]
    real_values=[]
    next_values=[]
    error = 0
    dictionary = initialize(minValue,maxValue,step,alphabet)
    pred_letter_from_method='';
    for i in range(windowSize):
        lingChain = returnLinguisticChain(dictionary, arr[i:baseSize+i])
        if(i > 0):
            lingChain[baseSize-2]=real_values[i-1]
            #print(probMatrix)
#         print(probMatrix)
        transMatrix = createTransitionMatrix(alphabet,lingChain)
        probMatrix = probabilityMatrix(transMatrix)
        last_known_letter_index = lingChain[baseSize-1]
        last_known_value = arr[baseSize+i-1]
        last_known_index_number = alphabet.index(last_known_letter_index)
#         print(last_known_letter_index)
#         print(last_known_value)
        real_next_letter =
returnLinguisticPosition(alphabet,dictionary,arr[baseSize+i])

prognozed_index=numpy.where(probMatrix[last_known_index_number]==numpy
.amax(probMatrix[last_known_index_number]))
        if(prognozed_index[0][0]==0):
```

```
            print("TRUE")
            predicted_letter = real_next_letter
            pred_values.append(predicted_letter)
        else:
            predicted_letter = alphabet[prognozed_index[0][0]]
            pred_values.append(predicted_letter)
        #print(returnValueFromLetter(dictionary,predicted_letter,step))
        real_values.append(real_next_letter)
        #arr[baseSize+i]=returnValueFromLetter(dictionary,predicted_letter,step)
        #print("predicted next letter: "+ predicted_letter + " real next letter: " +
real_next_letter)
        distance = levenshtein(predicted_letter,real_next_letter)
        #print(abs(alphabet.index(predicted_letter)-alphabet.index(real_next_letter)))
        error = error + abs(alphabet.index(predicted_letter)-
alphabet.index(real_next_letter))
        if(i==0):

next_values.append(real_arr[baseSize+i]+returnValueFromLetter(dictionary,predi
cted_letter,step))
        else:

next_values.append(next_values[i+1]+returnValueFromLetter(dictionary,predicted
_letter,step))
    b = numpy.sort(alphabet)
    v = numpy.searchsorted(b, pred_values)
    pyplot.plot(v)
    pyplot.yticks(numpy.arange(b.size), b)

    b = numpy.sort(alphabet)
    v = numpy.searchsorted(b, real_values)
    pyplot.plot(v)
    pyplot.yticks(numpy.arange(b.size), b)
#     pyplot.plot.yticks(np.arange(26), alphabet)
#     pyplot.plot(pred_values)
#     pyplot.plot(real_values)
    print(next_values)
    print(error/5)
    return numpy.array(next_values)

value=buildPrediction(minValue,maxValue,step,alphabet,diff,5,250,df.values)
print(value)

pyplot.plot(value)
pyplot.plot(df.values[250:255])
print(value)
```

```
predicted = []
for i in range(len(value)):
   predicted.append(value[i][0])
real = []
for i in range(len(value)):
   real.append(df['Price'][250+i])
correlation_matrix = numpy.corrcoef(predicted, real)
correlation_xy = correlation_matrix[0,1]
r_squared = correlation_xy**2
print(r_squared)

def forecast_accuracy(forecast, actual):
   mape = numpy.mean(numpy.abs(forecast - actual)/numpy.abs(actual))  # MAPE
   me = numpy.mean(forecast - actual)             # ME
   mae = numpy.mean(numpy.abs(forecast - actual))    # MAE
   mpe = numpy.mean((forecast - actual)/actual)   # MPE
   rmse = numpy.mean((forecast - actual)**2)**.5  # RMSE
   corr = numpy.corrcoef(forecast, actual)[0,1]   # corr
   mins = numpy.amin(numpy.hstack([forecast[:,None],
                        actual[:,None]]), axis=1)
   maxs = numpy.amax(numpy.hstack([forecast[:,None],
                        actual[:,None]]), axis=1)
   minmax = 1 - numpy.mean(mins/maxs)             # minmax
  # acf1 = acf(fc-test)[1]                      # ACF1
   return({'mape':mape, 'me':me, 'mae': mae,
         'mpe': mpe, 'rmse':rmse,
         'corr':corr, 'minmax':minmax})

forecast_accuracy(numpy.array(predicted), numpy.array(real))

def ssr(y_true, y_pred):
   squared_resid=0
   for i in range(len(y_true)):
      squared_resid=squared_resid+(y_true[i]-y_pred[i])**2
   return  squared_resid
print(ssr(value,df.values[250:255]))

def Theil(y_true, y_pred):
   y_true_2 = []
   y_pred_2 = []
   for i in range(len(y_true)):
      y_true_2.append(y_true[i]**2)
      y_pred_2.append(y_pred[i]**2)
```

```
    return rmse(y_true,
y_pred)/(math.sqrt((1/len(y_true))*sum(y_true_2))+math.sqrt((1/len(y_true))*sum
(y_pred_2)))

print(Theil(value,df.values[250:255]))

def read_data (name, number_of_predict_data):
    series = pd.read_csv(name, header=0, parse_dates=[0], index_col=0,
squeeze=True)
    series.drop('Open',inplace=True,axis=1)
    series.drop('High',inplace=True,axis=1)
    series.drop('Low',inplace=True,axis=1)
    series.drop('Close',inplace=True,axis=1)
    series.drop('Volume',inplace=True,axis=1)
    series.columns =['Price']
    split_point = len(series) - number_of_predict_data
    training_set, test_set = series.iloc[0:split_point], series.iloc[split_point:]
    print (len(test_set),len(training_set))
    return series,training_set,test_set

def dikky_fuller(series):
    test = sm.tsa.adfuller(series)
    print ('adf: ', test[0])
    print ('p-value: ', test[1])
    print ('Critical values: ', test[4])
    if (test[0]> test[4]['5%']):
        print ('часовий ряд має одиничні корені, ряд не стаціонарний')

    else:
        print ('ряд стаціонарний, одиничних коренів немає')
        return True
    return False
class PACF(object):
    def __init__(self, time_s):
        self.time_s = np.array(time_s)
        self.pacf_results = {}
        self.time_s_mean = self.time_s.mean()
        self.time_s_var = self.count_var()

    def count_var(self):
        return  ((self.time_s - self.time_s_mean)**2).sum() / (self.time_s.shape[0] - 1)

    def count_r(self, s):
        to_div = sum( (self.time_s[i] - self.time_s_mean) * (self.time_s[i-s] -
self.time_s_mean) for i in range(s, len(self.time_s)))
```

```
        divider = (self.time_s_var)*(len(self.time_s) - 1)
        return to_div / divider

    def F(self, k,j):

        if (k,j) in self.pacf_results.keys():
            return self.pacf_results[(k,j)]


        if k == j == 1 :
            f = self.count_r(1)
        else:
            to_div = self.count_r(k) - sum(self.F(k - 1, i) * self.count_r(k - i) for i in
range(1, k))
            divider = 1 - sum(self.F(k - 1, i) * self.count_r(i) for i in range(1, k))

            f = to_div/divider

        if k == j :
            self.pacf_results[(k, j)] = f
            return self.pacf_results[(k, j)]
        else:
            self.pacf_results[(k, j)] = self.F(k - 1, j) - f * self.F(k - 1, k - j)
            return self.pacf_results[(k, j)]

    def simmetric_F(self, k):
        return self.F(k,k)

    pacf_results = [pacf_object.simmetric_F(i) for i in range(12)]

def moving_average_order(resid):
    pacf_object = PACF(resid)
    pacf_results = [pacf_object.simmetric_F(i) for i in range(12)]
    print ('ACF: \n')
    for i in range (12):
        print (pacf_results[i])
    sm.graphics.tsa.plot_acf(resid, lags = 12)
    p = 0
    for i in range (12):
        if (abs(pacf_results[i]) > 0.15):
            p = i + 1
    print ('Доцільний порядок ковзного середнього = ',  p)
    return p

def analysis_of_series(series):
```

```
    print(series.describe())
    bds_test(series)
    if (dikky_fuller(series)):
        print('Порядок тренду = 0')
    else:
        i = 1
        temp_series = series.diff().dropna()

        while (not dikky_fuller(temp_series)):
            temp_series = temp_series.diff().dropna()
            i = i + 1
        #print('Порядок тренду = ', i)

def results_of_modeling_AR(params,y_true,y_pred,resid,p):
    print('Результати та рівняння моделі авторегресії порядку',p)
    #print('Рівняння моделі авторегресії порядку',p,':\n')
    print('len(params)= ',len(params))
    print('y(k)=',params[0]*(1-(sum(params)-params[0])),'+')
    for i in range(len(params)-1):
        if (i <= p): print(params[i + 1],'y( k -',i + 1,')')
        else:  print('+',params[i + 1],'ma( k-',i + 1,')')
    print('Параметри адекватності моделі:')
    print('SSR = ', ssr(resid))
    print('DW = ', durbin_watson(resid) )

    if (len(params) == 2) :
        print('R Squared = ', r2_score(y_true[1:],y_pred))
    else:
        print('R Squared = ', r2_score(y_true[p:],y_pred))

def results_of_modeling_ARMA(params,y_true,y_pred,resid,p,q):
    print('Результати та рівняння моделі ARMA порядку',p,q,':\n')
    #print('Рівняння моделі авторегресії порядку',p,':\n')
    print('y(k)=',params[0]*(1-(sum(params)-params[0])),'+')
    print(len(params))
    k=0
    while(k<=p-1):
        print(params[k + 1],'y( k -',k + 1,')')
        k=k+1
    if(k<len(params)):
        for j in range(len(params)-p-1):
            print('+',params[p+1+j],'ma( k-',j+1,')')
    print('Параметри адекватності моделі:')
    print('SSR = ', ssr(resid))
    print('DW = ', durbin_watson(resid) )
```

```
    if (len(params) == 3) : print('R Squared = ', r2_score(y_true[1:],y_pred))
    else: print('R Squared = ', r2_score(y_true[p:],y_pred))

def results_of_modeling_ARIMA(params,y_true,y_pred,resid,p,q):
    print('Результати та рівняння моделі ARIMA порядку',p,1,q,':\n')
    #print('Рівняння моделі авторегресії порядку',p,':\n')
    print('y(k)=',params[0]*(1-(sum(params)-params[0])),'+')
    print(len(params))
    k=0
    while(k<=p-1):
        print(params[k + 1],'y( k -',k + 1,')')
        k=k+1
    if(k<len(params)):
        for j in range(q):
            print('+',params[p+1+j],'ma( k-',j+1,')')
    print('Параметри адекватності моделі:')
    print('SSR = ', ssr(resid))
    print('DW = ', durbin_watson(resid) )
    gap=len(y_true.values)-len(y_pred.values)
    if (len(params) == 3) : print('R Squared = ', r2_score(y_true[2:],y_pred))
    else: print('R Squared = ', r2_score(y_true[gap:],y_pred))

def predictions_ARIMA(train, test,
model_ar_order,integration_order,model_ma_order):
    temp = [x for x in train.values]
    pred = []
    predict = []
    for t in range(len(test)):
        model =
sm.tsa.ARIMA(train,order=(model_ar_order,integration_order,model_ma_order))
        model_fitted = model.fit(disp=-1)
        output = model_fitted.forecast()
        yhat = output[0]
        pred.append(yhat)
        obs = test['Price'][t]
        temp.append(obs)
        print('predicted=%f, true=%f' % (yhat, obs))
    plt.figure(figsize=(15,10))
    pred=pd.Series(pred,index=test.index)
#     plt.plot(test, label = 'True')
#     plt.plot(pred, label = 'Predictions')
#     plt.legend()
#     plt.show()
    #model_fitted.plot_predict(dynamic=False)
```

```
    for i in range(len(pred)):
        predict.append(pred[i][0])
    print('Якість прогнозу:')
    print ('RMSE = ',rmse(test.values,predict))
    print ('MAPE = ',MAPE(test.values,predict))
    print ('Theil = ',Theil(test.values,predict))
    return predict

def predictions_ARMA(train, test, model_ar_order,model_ma_order):
    temp = [x for x in train.values]
    pred = []
    predict = []
    for t in range(len(test)):
        model = sm.tsa.ARMA(temp,order=(model_ar_order,model_ma_order))
        model_fitted = model.fit(method='css')
        output = model_fitted.forecast()
        yhat = output[0]
        pred.append(yhat)
        obs = test['Price'][t]
        temp.append(obs)
        print('predicted=%f, true=%f' % (yhat, obs))
    plt.figure(figsize=(8,8))
    pred=pd.Series(pred,index=test.index)
    plt.plot(test, label = 'True')
    plt.plot(pred, label = 'Predictions')
    plt.legend()
#     plt.show()
    for i in range(len(pred)):
        predict.append(pred[i][0])
    print('Якість прогнозу:')
    print ('RMSE = ',rmse(test.values,predict))
    print ('MAPE = ',MAPE(test.values,predict))
    print ('Theil = ',Theil(test.values,predict))
    return predict

nflx, nflx_train, nflx_test = read_data('NFLX.csv',5)

#графіки часових рядів

plt.figure(figsize=(15,10))
plt.subplot(223, title = 'Часовий ряд цін акцій NETFLIX')
plt.plot(nflx)
plt.show()

#дослідження вхідних рядів
```

```
print('\nДослідження часового ряду цін компанії NETFLIX \n')
analysis_of_series(nflx)


#Часткова автокореляційна функція

print('Визначення порядку авторегресії часового ряду "NETFLIX"')
nflx_ar_order = autoregressive_order(nflx_train)
nflx_ma_order=moving_average_order(model_fitted.resid)

print('Моделювання ряду NETFLIX AR(1)')
#Результати моделювання AP(1) NETFLIX
model = sm.tsa.ARMA(nflx_train,order=(1,0))
model_fitted = model.fit(method='css')
print(model_fitted.summary())
plt.figure(figsize=(8,8))
plt.plot(nflx_train,label = 'true')
plt.plot(model_fitted.predict(), label='predicted')
plt.legend()
plt.show()
results_of_modeling_AR(model_fitted.params,nflx_train,model_fitted.predict(),m
odel_fitted.resid,1)

#Результати виконання прогнозування AP(1)
nflx_result_AR1 = predictions_AR(nflx_train, nflx_test, 1)


print('Моделювання ряду NETFLIX AR(4)')
#Результати моделювання AP(4) NETFLIX
model = sm.tsa.ARMA(nflx_train,order=(4,0))
model_fitted = model.fit(method='css')
print(model_fitted.summary())
plt.figure(figsize=(8,8))
plt.plot(nflx_train,label = 'true')
plt.plot(model_fitted.predict(), label='predicted')
plt.legend()
plt.show()
results_of_modeling_AR(model_fitted.params,nflx_train,model_fitted.predict(),m
odel_fitted.resid,4)

#Результати виконання прогнозування AP(4)
nflx_result_AR3 = predictions_AR(nflx_train, nflx_test, 4)

print('Моделювання з найкраще підібраними параметрами AR(p)')
```

```
#Результати моделювання AP(1) NETFLIX
model = sm.tsa.ARMA(nflx_train,order=(nflx_ar_order,0))
model_fitted = model.fit(method='css')
print(model_fitted.summary())
plt.figure(figsize=(8,8))
plt.plot(nflx_train,label = 'true')
plt.plot(model_fitted.predict(), label='predicted')
plt.legend()
plt.show()
results_of_modeling_AR(model_fitted.params,nflx_train,model_fitted.predict(),m
odel_fitted.resid,nflx_ar_order)

#Результати виконання прогнозування AP(1)
nflx_result_AR = predictions_AR(nflx_train, nflx_test, nflx_ar_order)

nflx, nflx_train, nflx_test = read_data('NFLX.csv',5)

#графіки часових рядів

plt.figure(figsize=(15,10))
plt.subplot(223, title = 'Часовий ряд цін акцій NETFLIX')
plt.plot(nflx)
plt.show()

#дослідження вхідних рядів

print('\nДослідження часового ряду цін компанії NETFLIX \n')
analysis_of_series(nflx)


#Часткова автокореляційна функція

print('Визначення порядку авторегресії часового ряду "NETFLIX"')
nflx_ar_order = autoregressive_order(nflx_train)
nflx_ma_order=moving_average_order(model_fitted.resid)

print('Моделювання ряду NETFLIX ARMA(1,1)')
#Результати моделювання ARMA(1,1) NETFLIX
model = sm.tsa.ARMA(nflx_train,order=(1,1))
model_fitted = model.fit(method='css')
print(model_fitted.summary())
plt.figure(figsize=(8,8))
plt.plot(nflx_train,label = 'true')
plt.plot(model_fitted.predict(), label='predicted')
plt.legend()
```

```
plt.show()
print('TEST')
print(model_fitted.params)
results_of_modeling_ARMA(model_fitted.params,nflx_train,model_fitted.predict(
),model_fitted.resid,1,1)

#Результати виконання прогнозування ARMA(1,1)
nflx_result_ARMA = predictions_ARMA(nflx_train, nflx_test, 1,1)


print('Моделювання ряду NETFLIX ARMA(3,3)')
#Результати моделювання APMA(3,3) NETFLIX
model = sm.tsa.ARMA(nflx_train,order=(4,4))
model_fitted = model.fit(method='css')
plt.figure(figsize=(8,8))
plt.plot(nflx_train,label = 'true')
plt.plot(model_fitted.predict(), label='predicted')
plt.legend()
plt.show()
results_of_modeling_ARMA(model_fitted.params,nflx_train,model_fitted.predict(
),model_fitted.resid,4,5)

#Результати виконання прогнозування APMA(4,5)
nflx_result_ARMA3 = predictions_ARMA(nflx_train, nflx_test, 4,5)

print('Моделювання з найкраще підібраними параметрами ARMA(p,q)')
#Результати моделювання APMA(p,q) NETFLIX
model = sm.tsa.ARMA(nflx_train,order=(nflx_ar_order,nflx_ma_order))
model_fitted = model.fit(method='css')
plt.figure(figsize=(8,8))
plt.plot(nflx_train,label = 'true')
plt.plot(model_fitted.predict(), label='predicted')
plt.legend()
plt.show()
results_of_modeling_ARMA(model_fitted.params,nflx_train,model_fitted.predict(
),model_fitted.resid,nflx_ar_order,nflx_ma_order)

#Результати виконання прогнозування APMA(p,q)
nflx_result_ARMA = predictions_ARMA(nflx_train, nflx_test,
nflx_ar_order,nflx_ma_order)

nflx, nflx_train, nflx_test = read_data('NFLX.csv',5)

#графіки часових рядів
```

```
plt.figure(figsize=(15,10))
plt.subplot(223, title = 'Часовий ряд цін акцій NETFLIX')
plt.plot(nflx)
plt.show()

#дослідження вхідних рядів

print('\nДослідження часового ряду цін компанії NETFLIX \n')
analysis_of_series(nflx)


#Часткова автокореляційна функція

print('Визначення порядку авторегресії часового ряду "NETFLIX"')
nflx_ar_order = autoregressive_order(nflx_train)
nflx_ma_order=moving_average_order(model_fitted.resid)

print('Моделювання ряду NETFLIX ARIMA(1,1,1)')
#Результати моделювання ARIMA(1,1,1) NETFLIX
model = sm.tsa.ARIMA(nflx_train,order=(1,1,1))
model_fitted = model.fit(method='css')
# plt.figure(figsize=(8,8))
# plt.plot(nflx_train,label = 'true')
# plt.plot(model_fitted.predict(), label='predicted')
# plt.legend()
# plt.show()
model_fitted.plot_predict(dynamic=False,alpha=0.05)
print('TEST')
print(model_fitted.params)
results_of_modeling_ARIMA(model_fitted.params,nflx_train,model_fitted.predict
(),model_fitted.resid,1,1)

#Результати виконання прогнозування ARMA(1,1)
nflx_result_ARIMA = predictions_ARMA(nflx_train, nflx_test, 1,1)


print('Моделювання ряду NETFLIX ARIMA(4,1,4)')
#Результати моделювання ARIMA(4,1,4) NETFLIX
model = sm.tsa.ARIMA(nflx_train,order=(4,1,4))
model_fitted = model.fit()
# plt.figure(figsize=(8,8))
# plt.plot(nflx_train,label = 'true')
# plt.plot(model_fitted.predict(), label='predicted')
# plt.legend()
# plt.show()
```

```
model_fitted.plot_predict(dynamic=False)
results_of_modeling_ARIMA(model_fitted.params,nflx_train,model_fitted.predict
(),model_fitted.resid,4,4)

#Результати виконання прогнозування ARIMA(4,1,4)
nflx_result_ARIMA3 = predictions_ARMA(nflx_train, nflx_test, 3,3)

print('Моделювання з найкраще підібраними параметрами ARIMA(p,d,q)')
#Результати моделювання ARIMA(p,d,q) NETFLIX
model = sm.tsa.ARIMA(nflx_train,order=(nflx_ar_order,1,nflx_ma_order))
model_fitted = model.fit()
# plt.figure(figsize=(8,8))
# plt.plot(nflx_train,label = 'true')
# plt.plot(model_fitted.predict(), label='predicted')
# plt.legend()
# plt.show()
model_fitted.plot_predict(dynamic=False)
results_of_modeling_ARIMA(model_fitted.params,nflx_train,model_fitted.predict
(),model_fitted.resid,nflx_ar_order,nflx_ma_order)

#Результати виконання прогнозування ARIMA(p,d,q)
nflx_result_ARIMA = predictions_ARMA(nflx_train, nflx_test,
nflx_ar_order,nflx_ma_order)
```

ДОДАТОК Б. ІЛЮСТРАТИВНИЙ МАТЕРІАЛ

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. ІГОРЯ СІКОРСЬКОГО»

МОДЕЛЮВАННЯ НЕЛІНІЙНИХ НЕСТАЦІОНАРНИХ ПРОЦЕСІВ В ЕКОНОМІЦІ

Виконав:
студент групи КА-77, Жук В.М.
Науковий керівник:
к.т.н., старший викладач, Селін Ю.М.

Актуальність

Основні проблеми – прогнозування нелінійних нестаціонарних економічних процесів.

Майже всі економічні процеси, які спостерігає людство за своєю природою є нелінійними та нестаціонарними. Існують лише частково стаціонарні процеси на певних часових ділянках.

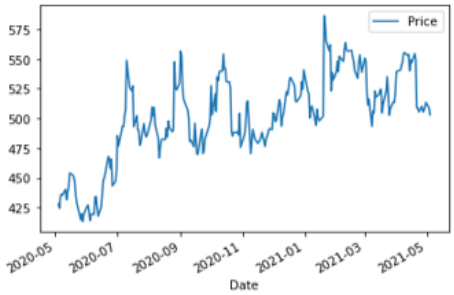


Рис.1 – Графік часового ряду річних цін акцій компанії «NETFLIX»

Мета роботи

Метою роботи є покращення оцінки прогнозів нелінійних нестаціонарних процесів у порівнянні з методами, які вже розроблені та активно використовуються.

Для досягнення мети дослідження було поставлено і виконано такі завдання:

- Виконати аналіз деяких методів дослідження нелінійних нестаціонарних процесів.
- Виявити нелінійності і нестаціонарності у сучасних фінансово-економічних процесах.
- Вибрати процеси для дослідження та зібрати необхідні статистичні дані.
- Виконати порівняльний аналіз отриманих результатів і виробити рекомендації стосовно їх практичного застосування.

Об’єкт, предмет і мета дослідження

Об’єкт дослідження – нелінійні нестаціонарні процеси в економіці.

Предмет дослідження – методи моделювання і прогнозування часових рядів.

Мета дослідження - покращення оцінки прогнозів нелінійних нестаціонарних процесів у порівнянні з методами, які вже розроблені та активно використовуються.

Моделі прогнозування

- Авторегресія порядку p AP(p).

$$y(k) = a_0 + \sum_{i=1}^{p} a_i y(k-i) + \varepsilon(k)$$

- Авторегресія з ковзним середнім АРКС(p,q).

$$y(k) = a_0 + \sum_{i=1}^{p} a_i y(k-i) + \sum_{j=1}^{q} b_j ma(k-i) + \varepsilon(k)$$

- Інтегрована авторегресія з ковзним середнім АРІКС(p,d,q)
- МГУА

$$Y(x_1, \ldots, x_n) = a_0 + \sum_{i=1}^{n} a_i x_i + \sum_{i=1}^{n} \sum_{j=i}^{n} a_{ij} x_i x_j + \sum_{i=1}^{n} \sum_{j=i}^{n} \sum_{k=j}^{n} a_{ijk} x_i x_j x_k + \cdots$$

Метод лінгвістичного моделювання

Під час виконання дипломної роботи було реалізовано статистичний метод лінгвістичного моделювання для побудови прогнозів нелінійних нестаціонарних процесів економічної природи.

Слід зазначити, що даний метод розроблений разом з науковим керівником Селіним Ю.М., а його реалізація була виконана за допомогою середовища програмування Jupyter Notebook, основною мовою якого є мова програмування Python.

Опис методу лінгвістичного моделювання

Загальна характеристика

Даний метод належить до родини так званих статистичних методів, що використовують попередньо отримані результати часових рядів. Слід зауважити, що реалізований метод використовує приховані марковські мережі для побудови ймовірнисних матриць переходів.

Для побудови системи, необхідно розв'язати задачу пошуку лінгвістичного образу для досліджуваного часового ряду, яка складається з декількох етапів:

- Обрахунок різницевих рядів.
- Вибір значення критерію інтервального розбиття різницевого ряду.
- Співставлення часовому інтервалу ряду певної літери алфавіту.
- Побудова матриці переходу між будь-якими двома літерами.

Для побудови системи, сам процес лінгвістичного моделювання буде розподілений на окремі задачі.

Опис методу лінгвістичного моделювання

Обрахунок різнецевих рядів

Якщо вхідний часовий ряд (у більшості випадків) є нестаціонарним – єдине, що потрібно зробити це продиференціювати існуюючий ряд для того, щоб основні статистичні характеристики, такі як математичне сподівання, дисперсія, коваріація були сталими в часі.

Найпростіший спосіб звести ряд до стаціонарного – відняти від наступного значення ряду попереднє. Довжина отриманого ряду буде менше на кількість разів, скільки була застосована операція віднімання. Даний показник визначається відповідними тестами.

$$\forall d_i \in D : d_i = x_{i+1} - x_i$$

$$i \in [0; n-1); x_i, x_{i+1} \in X$$

Отриманий ряд перевіряється на умови стаціонарності (наприклад, тест Дікі-Фуллера)

Опис методу лінгвістичного моделювання

Вибір значення критерію інтервального розбиття різницевого ряду

Розв'язок задачі дозволяє побудувати алфавіт користувача шляхом розділення відсортованого ряду першої (зазвичай, якщо порядок тренду лінійний) різниці на безліч інтервалів, в якому кожний елемент характеризує певну літеру заданого алфавіту.

Завичай довжина алфавіту обирається рівною 26. Для цього ідеально підходить алфавіт англійської мови. Довжина інтервалу, що буде відповідати кожній літері визначається наступним чином:

$$\frac{maxValue - minValue}{length(alphabet)},$$

Де $minValue$ – мінімальне значення продефиренційованого ряду, $maxValue$ – максимальне значення продиференційованого ряду, $length(alphabet)$ – довжина алфавіту.

Варто розуміти, що дуже велика або дуже маленька кількість інтервалів розбиття негативно вплине на отриманий результат, тому що при побудові матриці переходу або дуже багато значень будуть потрапляти в один проміжок (при малому розбитті) або значення ймовірності переходу від одної літери до іншої будуть майже однаковими (при великому розбитті).

Опис методу лінгвістичного моделювання

Співставлення часовому інтервалу ряду певної літери алфавіту

Після проведення операції інтервалізації даних, необхідно співставити кожній літері алфавіту часовий інтервальний проміжок.

```
In [179]: 1 dictionary = initialize(minValue,maxValue,step,alphabet)
          2 print(dictionary)
```

```
{'a': (-24.270004, -22.448465499999998), 'b': (-22.448465499999998, -20.626926999999995), 'c': (-20.626926999999995, -18.8
05388499999992), 'd': (-18.805388499999992, -16.98384999999999), 'e': (-16.98384999999999, -15.162311499999989), 'f': (-1
5.162311499999989, -13.340772999999988), 'g': (-13.340772999999988, -11.519234499999987), 'h': (-11.519234499999987, -9.69
7695999999986), 'i': (-9.697695999999986, -7.876157499999985), 'j': (-7.876157499999985, -6.054618999999983), 'k': (-6.054
618999999983, -4.233080499999981), 'l': (-4.233080499999981, -2.411541999999794), 'm': (-2.411541999999794, -0.590003499
9999779), 'n': (-0.5900034999999779, 1.2315350000000236), 'o': (1.2315350000000236, 3.053073500000025), 'p': (3.0530735000
00025, 4.874612000000027), 'q': (4.874612000000027, 6.696150500000028), 'r': (6.696150500000028, 8.5176890000000003), 's':
(8.517689000000003, 10.339227500000003), 't': (10.339227500000003, 12.160766000000003), 'u': (12.160766000000003, 13.9823845000
00032), 'v': (13.982384500000032, 15.803843000000033), 'w': (15.803843000000033, 17.625301500000035), 'x': (17.625301500000
035, 19.446920000000038), 'y': (19.446920000000038, 21.26845850000004), 'z': (21.26845850000004, 23.089997000000043)}
```

Рис.2 – Інтервальне розбиття часового ряду «NETFLIX»

Опис методу лінгвістичного моделювання

Побудова матриці переходу між будь-якими двома літерами

Далі будуємо матрицю переходів, де стовпчики та рядки являють собою літери алфавіту, що використовується. На перетинах рядків та стовпців (i,j) – ймовірність переходу від літери і до j.

Сума ймовірнестей у рядку дорівнює 1.

```
[[0.         0.         0.         0.         0.05263158 0.
  0.05263158 0.15789474 0.05263158 0.15789474 0.26315789 0.15789474
  0.         0.05263158 0.05263158 0.         0.         0.
  0.         0.         0.        ]
 [0.         0.03333333 0.         0.03333333 0.03333333 0.03333333
  0.1        0.13333333 0.16666667 0.13333333 0.1        0.13333333
  0.03333333 0.         0.06666667 0.         0.         0.
  0.         0.         0.        ]
 [0.         0.01851852 0.         0.01851852 0.03703704 0.09259259
  0.12962963 0.12962963 0.14814815 0.18518519 0.12962963 0.05555556
  0.03703704 0.         0.01851852 0.         0.         0.
  0.         0.         0.        ]
 [0.01369863 0.02739726 0.01369863 0.01369863 0.04109589 0.06849315
  0.12328767 0.1369863  0.15068493 0.16438356 0.10958904 0.05479452
  0.01369863 0.04109589 0.         0.01369863 0.         0.01369863
  0.         0.         0.        ]
 [0.         0.         0.         0.         0.02941176 0.07843137
  0.10784314 0.17647059 0.23529412 0.15686275 0.11764706 0.05882353
  0.01960784 0.         0.00980392 0.         0.         0.
  0.00980392 0.         0.        ]
 [0.         0.         0.         0.01449275 0.01449275 0.02898551
  0.11594203 0.11594203 0.30434783 0.15942029 0.13043478 0.07246377
  0.02898551 0.         0.         0.         0.         0.
  0.         0.         0.        ]
```

$\forall x_{ij} \epsilon \bar{P}: x_{ij} \epsilon [0.0; 1.0]$, де $i, j \in [0, n)$

Рис.3 – Інтервальне розбиття часового ряду «NETFLIX»

Опис методу лінгвістичного моделювання

Лінгвістичний ланцюг та подальші обчислення

Після виконання наступних операцій ми отримали лінгвістичний ланцюг з побудованою ймовірнісною матрицею переходів. Далі прогноз визначається наступним чином :

- Формується вибірка довжиною N та визначається кількість прогнозованих елементів k.
- Будується лінгвістичний ланцюг довжиною N та матриця переходів P.
- Береться останнє відоме значення ряду та перетворюється в літеру.
- Аналізується матриця переходів та визначається найбільш ймовірна літера.
- Робиться прогноз на один крок, визначається похибка та зсувається вікно даних на одиницю, враховуючи спрогнозоване значення.
- Ітерація виконується k кроків.

Опис методу лінгвістичного моделювання

Приклад лінгвістичного ланцюга

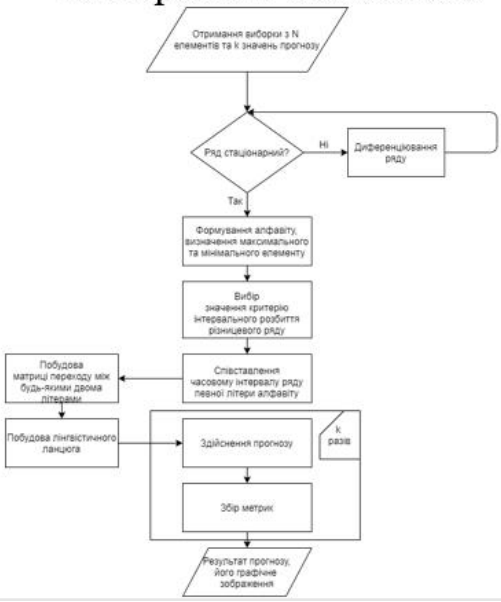
```
In [100]: print(returnLinguisticChain(dictionary,diff))
```

```
['g', 'n', 'm', 'l', 'o', 'n', 'k', 'n', 'p', 'm', 'n', 'k', 'g', 'p', 'n', 'n', 'q', 'o', 'p', 'l', 'o', 'p', 'r', 'n',
'm', 'o', 'o', 'o', 'l', 'm', 'o', 'p', 'n', 'o', 'o', 'm', 'm', 'p', 'p', 'm', 'p', 'm', 'n', 'm', 'm', 'l', 'p', 'n',
'o', 'l', 'm', 'l', 'n', 'l', 'm', 'l', 'j', 'o', 'n', 'q', 'm', 'l', 'p', 'l', 'o', 'n', 'o', 'm', 'n', 'm', 'k', 'o',
'n', 'n', 'p', 'n', 'l', 'p', 'p', 'l', 'o', 'l', 'o', 'm', 'n', 'm', 'o', 'n', 'o', 'n', 'l', 'k', 'o', 'l', 'l', 'n',
'm', 'm', 'p', 'n', 'm', 'm', 'o', 'n', 'p', 'm', 'q', 'n', 'n', 'k', 'p', 'j', 'p', 'n', 'o', 'o', 'n', 'm', 'p', 'o',
'n', 'n', 'l', 'm', 'n', 'm', 'p', 'm', 'n', 'o', 'o', 'o', 'm', 'n', 'n', 'n', 'm', 'o', 'n', 'm', 'm', 'n', 'n', 'o',
'n', 'n', 'o', 'k', 'l', 'p', 'n', 'p', 'p', 'n', 'n', 'm', 'o', 'n', 'l', 'n', 'p', 'm', 'p', 'n', 'o', 'p', 'n', 'p',
'l', 'o', 'n', 'n', 'm', 'n', 'm', 'm', 'l', 'o', 'q', 'f', 'j', 'o', 'q', 'n', 'n', 'o', 'n', 'n', 'p', 'o', 'n', 'p',
'n', 'l', 'k', 'l', 'l', 'n', 'j', 'o', 'p', 'h', 'q', 'j', 'l', 'g', 's', 'l', 'e', 'w', 'x', 'p', 'l', 'q', 'l', 'm',
'u', 'k', 'r', 'j', 's', 'n', 'j', 'm', 'l', 't', 'p', 'q', 'n', 'n', 'p', 'm', 'm', 'o', 'm', 'j', 't', 'o', 'q', 'l',
'k', 't', 't', 'l', 'o', 'o', 'o', 'o', 'n', 'n', 'l', 'k', 'o', 'p', 'o', 'p', 'x', 'o', 'p', 'l', 'l', 'l', 'n', 'r',
'n', 'l', 'l', 'p', 'n', 'r', 'm', 'g', 'p', 'p', 'p', 'n', 'n', 'o', 'n', 'o', 'l', 'o', 'c', 'p', 'q', 't', 'l', 'r',
'n', 'o', 'n', 'j', 'n', 'n', 'n', 'n', 'p', 'l', 'm', 'j', 'm', 'q', 'l', 'p', 'n', 'o', 'm', 'm', 'm', 'x', 'p', 'n',
'k', 'j', 'p', 'o', 'n', 'n', 'n', 'q', 'm', 'p', 'q', 'z', 'b', 'z', 'r', 'n', 'n', 'q', 'n', 'n', 'h', 'l', 'g', 'o', 'l', 'm',
'n', 'q', 'l', 'm', 'l', 'q', 'l', 'q', 'o', 'q', 'n', 'p', 'j', 'p', 'n', 'q', 'q', 'l', 't', 'n', 'k', 'k',
'm', 'k', 'q', 't', 'm', 'q', 'j', 'q', 'e', 'u', 'd', 'm', 'p', 'z', 'r', 'm', 'f', 'j', 'p', 'm', 'o', 'o', 'l', 'l',
'n', 'l', 'm', 'r', 'm', 'o', 'm', 'j', 'n', 'k', 'm', 'q', 'm', 'k', 'm', 'l', 'x', 'o', 'n', 'm', 'o', 'l', 'k', 'n',
'm', 'n', 'k', 'o', 'l', 'o', 'j', 'o', 'm', 'h', 'k', 'n', 'k', 'q', 's', 'q', 'q', 'o', 'p', 'p', 'h', 'j', 'j',
'p', 'q', 'n', 'o', 'm', 'o', 'q', 'm', 'n', 'p', 'n', 'l', 'q', 'm', 'q', 'm', 'l', 'o', 'r', 'k', 'l', 'o', 'q',
'l', 't', 'm', 's', 'l', 'q', 'q', 'p', 'k', 't', 'p', 'l', 'l', 'l', 'p', 'r', 'l', 'q', 'p', 's', 'l', 'r', 'n', 'n',
'm', 'm', 'j', 'q', 'm', 'l', 'n', 'm', 'k', 'p', 'o', 'n', 'p', 'r', 'k', 'l', 'l', 'l', 'q', 'm', 'g', 'n', 'l', 'o']
```

Рис.4 – Лінгвістичний ланцюг часового ряду «NETFLIX»

$$\forall x_i \epsilon \bar{A}: \exists d_i \epsilon D, \exists y_j \epsilon \bar{I}, d_i \epsilon [y_j^1, y_j^2], x_i = j, \text{де } i \in [0; k), j \in [0; n)$$

Поверхнева блок-схема



Функціональна схема програмного продукту

Вигляд вхідних отриманих даних

Date	Price
2019-05-14	180.729996
2019-05-15	186.270004
2019-05-16	186.990005
2019-05-17	185.300003
2019-05-20	182.720001
2019-05-21	184.820007
2019-05-22	185.320007
2019-05-23	180.869995
2019-05-24	181.059998
2019-05-28	184.309998
2019-05-29	182.190002
2019-05-30	183.009995
2019-05-31	177.470001
2019-06-03	164.149994
2019-06-04	167.500000
2019-06-05	168.169998
2019-06-06	168.330002
2019-06-07	173.350006
2019-06-10	174.820007
2019-06-11	178.100006
2019-06-12	175.039993
2019-06-13	177.470001
2019-06-14	181.330002
2019-06-17	189.009995
2019-06-18	188.470001
2019-06-19	187.479996
2019-06-20	189.529999

Результати моделювання лінгвістичним методом

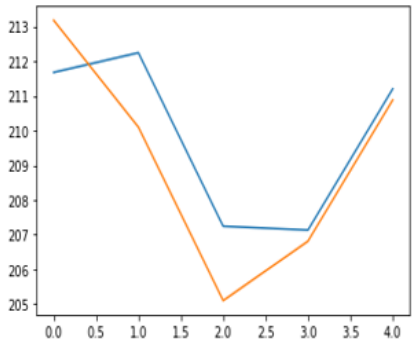


Рис.4 – Прогноз на 5 кроків вперед з використанням моделі лінгвістичного моделювання часового ряду «NETFLIX»

Результати моделювання лінгвістичним методом

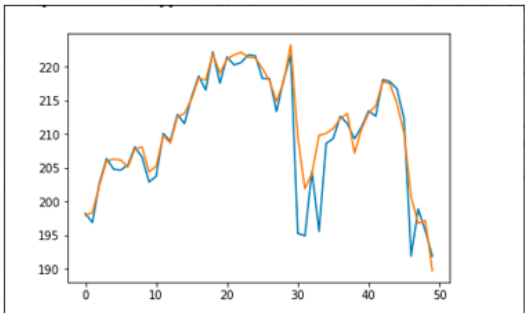


Рис.5 – Прогноз на 50 кроків вперед з використанням моделі лінгвістичного моделювання часового ряду «NETFLIX»

Результати моделювання лінгвістичним методом

Основні критерії якості прогнозу

$RMSE = 2.5935 \quad MAPE = 0.008 \quad Theil = 0.0036397$

Основні статистики адекватності лінгвістичної моделі

$R^2 = 0.9348136$

$Sum\ squared\ resid = 11.637$

$Durbin - Watson = 1.4256213$

Порівняння статистичних характеристик з існуючими методами прогнозування

Модель часового ряду	R^2	Sum squared resid	Durbin – Watson
AP(1)	0.87447	43540.31	2.21994
AP(4)	0.872836	42237.67	1.95032
AP(5)	0.87669	40409.32	1.95289
APKC(4,4)	0.90509	39770.35	2.00993
ARIMA(4,1,4)	0.955309	40037.91	2.005514
ARIMA(5,1,1)	0.968889	41037.91	2.017372
Linguistic modeling	0.9348136	11.637035	1.425621

Розроблений метод має один з найкращих статистичних характеристик в порівнянні з іншими методами

Порівняння основних критеріїв якості прогнозу побудованих моделей

Модель часового ряду	*RMSE*	*MAPE*	*Theil*
AP(1)	3.9122	0.61273	0.00384
AP(4)	3.8412	0.51842	0.00377
AP(5)	3.68233	0.5237	0.003621
APKC(4,4)	3.54399	0.54337	0.00348
ARIMA(4,1,4)	0.0084001	0.51599	0.0044297
ARIMA(5,1,1)	4.008855	0.511891	0.00394
Linguistic modeling	2.5935	0.008	0.0036397

Розроблений метод має один з найкращих критерії якості прогнозу в порівнянні з іншими методами

Висновки по роботі

- Розроблений метод є універсальним.
- Для використання та здійснення прогнозу необхідна лише мінімальна підготовка даних.
- Лінгвістичний метод дає непогані та прийнятні результати прогнозу.
- Найпопулярніші статистичні метрики оцінки моделі є досить високими.

Всі задачі, які були поставлені під час роботи виконано:

- виконано аналіз деяких методів дослідження нелінійних нестаціонарних процесів.
- виявлено нелінійності і нестаціонарності у сучасних фінансово-економічних процесах.
- вибрано процеси для дослідження та зібрано необхідні статистичні дані.
- виконано порівняльний аналіз отриманих результатів і виробити рекомендації стосовно їх практичного застосування.

Дякую за увагу!