Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра автоматизації та інтелектуальних інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«ВИКОРИСТАННЯ КЛАСТЕРНОГО АНАЛІЗУ ДЛЯ ІДЕНТИФІКАЦІЇ ЗАЛЕЖНОСТЕЙ НА ЧАСОВИХ РЯДАХ»

Виконав: студент 2 курсу, групи 1ІСТ-22м
спеціальності 126 – «Інформаційні системи та технології»
_____ Радислав ГРІНЧАК

Керівник: к.т.н., доц. каф. АІІТ
_____ Владислав КАБАЧІЙ
«18» 12 2023 р.

Опонент: к.т.н., доц. каф. КН
_____ Ігор АРСЕНЮК
«18» грудня 2023 р.

Допущено до захисту

Завідувач кафедри АІІТ
_____ д.т.н., проф. Олег БІСІКАЛО
«18» грудня 2023 р.

Вінниця ВНТУ – 2023 рік

Вінницький національний технічний університет

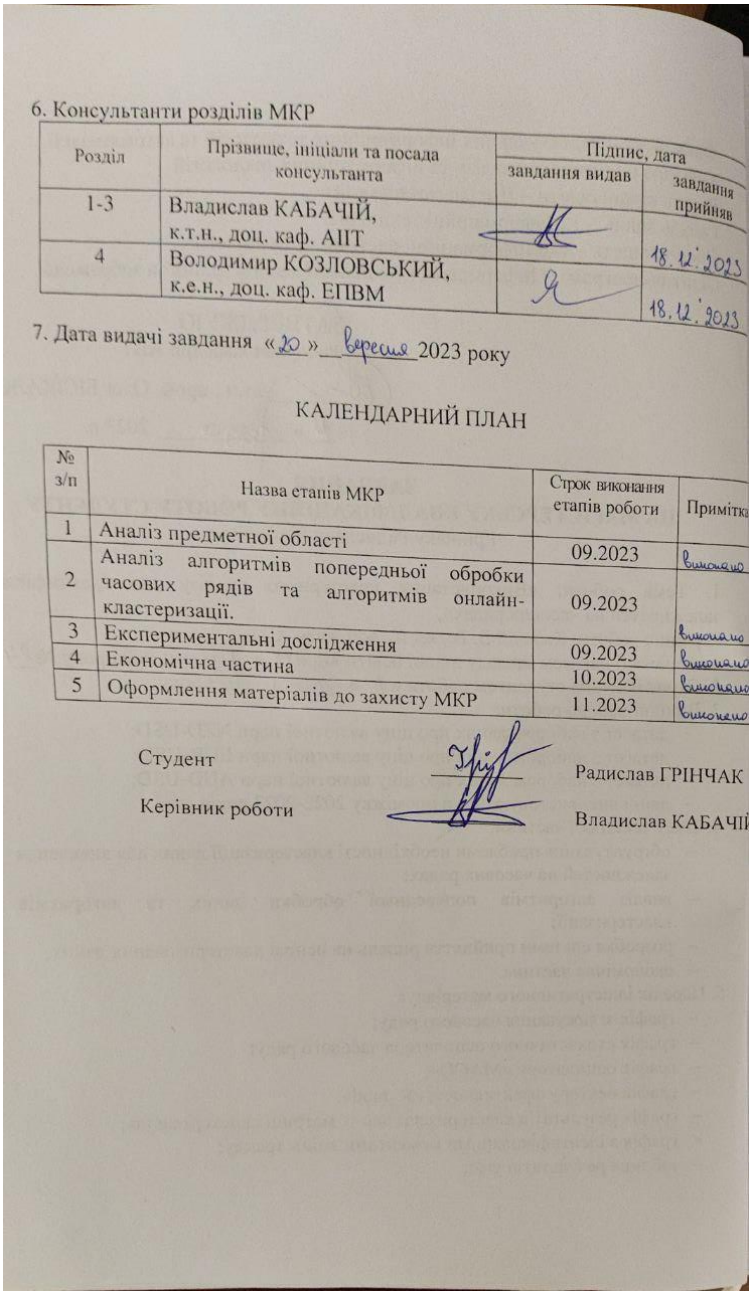
2

Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій
Рівень вищої освіти – II-й (магістерський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 126 Інформаційні системи та технології
Освітня програма – Інформаційні технології аналізу даних та зображень

ЗАТВЕРДЖУЮ
Завідувач кафедри АІІТ
_____ д.т.н., проф. Олег БІСІКАЛО
«18» вересня 2023 р.

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Грінчаку Радиславу Юрієвичу

1. Тема роботи: «Використання кластерного аналізу для ідентифікації залежностей на часових рядах»,
керівник роботи: Кабачій В. В., к.т.н., доц. каф. АІІТ,
затверджені наказом закладу вищої освіти від «18» вересень 2023 року №247
2. Строк подання студентом роботи «10» грудня 2023 року
3. Вихідні дані до роботи:
   - датасет з набором даних про ціну валютної пари NZD-USD;
   - датасет з набором даних про ціну валютної пари EUR-USD;
   - датасет з набором даних про ціну валютної пари AUD-USD;
   - данні використовуються в проміжку 2022-2023 років.
4. Зміст текстової частини:
   - обґрунтування проблеми необхідності кластеризації даних для виявлення залежностей на часових рядах;
   - аналіз алгоритмів попередньої обробки даних та алгоритмів кластеризації;
   - розробка системи прийняття рішень на основі кластеризованих даних;
   - економічна частина.
5. Перелік ілюстративного матеріалу:
   - графік зглажування часового ряду;
   - графік стохастичного осцилятора часового ряду;
   - графік осцилятору «MACD»;
   - графік вектору ефективності «K_mod»;
   - графік результатів кластеризації даних матриці спостереження;
   - графік з ідентифікованими моментами зміни тренду;
   - таблиця результатів угод.



6. Консультанти розділів МКР

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-3	Владислав КАБАЧІЙ, к.т.н., доц. каф. АІІТ		18.12.2023
4	Володимир КОЗЛОВСЬКИЙ, к.е.н., доц. каф. ЕПВМ		18.12.2023

7. Дата видачі завдання «20» вересня 2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів МКР	Строк виконання етапів роботи	Примітка
1	Аналіз предметної області	09.2023	Виконано
2	Аналіз алгоритмів попередньої обробки часових рядів та алгоритмів онлайн-кластеризації.	09.2023	
3	Експериментальні дослідження	09.2023	Виконано
4	Економічна частина	10.2023	Виконано
5	Оформлення матеріалів до захисту МКР	11.2023	Виконано

Студент Радислав ГРІНЧАК

Керівник роботи Владислав КАБАЧІЙ

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра автоматизації та інтелектуальних інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«ВИКОРИСТАННЯ КЛАСТЕРНОГО АНАЛІЗУ ДЛЯ ІДЕНТИФІКАЦІЇ ЗАЛЕЖНОСТЕЙ НА ЧАСОВИХ РЯДАХ»

Виконав: студент 2 курсу, групи 1ІСТ-22м
спеціальності 126 – «Інформаційні системи та технології»
_____ Радислав ГРІНЧАК

Керівник: к.т.н., доц. каф. АІІТ
_____Владислав КАБАЧІЙ
«___» _____ 2023 р.

Опонент: к.т.н., доц. каф. КН
_____ Ігор АРСЕНЮК
«___» _____ 2023 р.

Допущено до захисту

Завідувач кафедри АІІТ
_____ д.т.н., проф. Олег БІСІКАЛО
«___» _____ 2023 р.

Вінниця ВНТУ – 2023 рік

Вінницький національний технічний університет

Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій
Рівень вищої освіти – ІІ-й (магістерський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 126 Інформаційні системи та технології
Освітня програма – Інформаційні технології аналізу даних та зображень

ЗАТВЕРДЖУЮ
Завідувач кафедри АІІТ

_____д.т.н., проф. Олег БІСІКАЛО

«___» _____ 2023 р.

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Грінчаку Радиславу Юрієвичу

1. Тема роботи: «Використання кластерного аналізу для ідентифікації залежностей на часових рядах»,
керівник роботи: Кабачій В. В., к.т.н., доц. каф. АІІТ,
затверджені наказом закладу вищої освіти від «___»__ _____2023 року №__
2. Строк подання студентом роботи «___»_____2023 року
3. Вихідні дані до роботи:
   - датасет з набором даних про ціну валютної пари NZD-USD;
   - датасет з набором даних про ціну валютної пари EUR-USD;
   - датасет з набором даних про ціну валютної пари AUD-USD;
   - данні використовуються в проміжку 2022-2023 років.
4. Зміст текстової частини:
   - обґрунтування проблеми необхідності кластеризації даних для виявлення залежностей на часових рядах;
   - аналіз алгоритмів попередньої обробки даних та алгоритмів кластеризації;
   - розробка системи прийняття рішень на основі кластеризованих даних;
   - економічна частина.
5. Перелік ілюстративного матеріалу:
   - графік зглажування часового ряду;
   - графік стохастичного осцилятора часового ряду;
   - графік осцилятору «MACD»;
   - графік вектору ефективності «K_mod»;
   - графік результатів кластеризації даних матриці спостереження;
   - графік з ідентифікованими моментами зміни тренду;

– таблиця результатів угод.

6. Консультанти розділів МКР

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-3	Владислав КАБАЧІЙ, к.т.н., доц. каф. АІІТ		
4	Володимир КОЗЛОВСЬКИЙ, к.е.н., доц. каф. ЕПВМ		

7. Дата видачі завдання «___»_____2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів МКР	Строк виконання етапів роботи	Примітка
1	Аналіз предметної області	09.2023	
2	Аналіз алгоритмів попередньої обробки часових рядів та алгоритмів онлайн-кластеризації.	09.2023	
3	Експериментальні дослідження	09.2023	
4	Економічна частина	10.2023	
5	Оформлення матеріалів до захисту МКР	11.2023	

Студент _____ Радислав ГРІНЧАК

Керівник роботи _____ Владислав КАБАЧІЙ

АНОТАЦІЯ

УДК004.65:519.237.2

Грінчак Р.Ю. Використання кластерного аналізу для ідентифікації залежностей на часових рядах. Магістерська кваліфікаційна робота зі спеціальності 126 – Інформаційні системи та технології, освітня програма – Інформаційні технології аналізу даних та зображень. Вінниця: ВНТУ, 2023. 115 с.

На укр. мові. Бібліогр.: 34 назв; рис.: 40.

В даній роботі представлено огляд існуючих методів та засобів аналізу часових рядів. Обґрунтовується вибір методу та програмного середовища для розробки та реалізації математичної моделі та алгоритму прийняття рішень за використанням кластерного аналізу. Представлена покрокова реалізація ідентифікації моментів прийняття рішень по зміні тенденції часового ряду. Наведені результати роботи експертної системи на прикладі історичних даних декількох фінансових інструментів.

ABSTRACT

Grinchak Radyslav. Using cluster analysis to identify addiction on time rows. Master's qualification work in specialty 126 – information systems and technologies, educational program – information systems and Internet things. Vinnitsa: VNTU, 2023. 115 p.

At the Ukrainian. language. Bibliogr: 34 titles; Fig.: 40.

This work provides an overview of existing methods and tools for time series analysis. The choice of method and software environment for the development and implementation of a mathematical model and decision-making algorithm using cluster analysis is justified. A step-by-step implementation of the identification of decision-making moments based on the change in the trend of the time series is presented. The results of the expert system's performance are demonstrated using historical data from several financial instruments.

ЗМІСТ

АНОТАЦІЯ 7
ВСТУП 6
1.АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ 9
1.1. Характеристика об’єкту досліджень 9
1.2 Кластеризація часових рядів 14
1.3 Класифікація методів кластеризації часових рядів 17
1.4 Постановка задачі дослідження 26
1.5 Висновки 26
2. АНАЛІЗ АЛГОРИТМІВ ПОПЕРЕДНЬОЇ ОБРОБКИ ЧАСОВИХ РЯДІВ ТА АЛГОРИТМІВ ОНЛАЙН КЛАСТЕРИЗАЦІЇ 28
2.1. Перевірка статистичних гіпотез властивостей часового ряду у процесі попереднього аналізу вибірки 28
2.2. Методи зменшення розмірності 35
2.3. Міри відстані для послідовних даних 38
2.4. Динамічна деформація часу 39
2.5. Висновки 42
3. ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ 43
3.1. Обґрунтування вибору середовища програмної реалізації 43
3.2. Аналіз обраних бібліотек 48
3.3. Підготовка та аналіз датасету 52
3.4. Програмна реалізація 60
3.5. Висновки 73
4. ЕКОНОМІЧНИЙ РОЗДІЛ 74
4.1 Використання Кластерного аналізу для ідентифікації залежностей на часових рядах 74
4.2 Розрахунок витрат на розроблення алгоритму та методики ідентифікації ринкових залежностей на основі впровадженої математичної моделі 80
4.3 Розрахунок економічного ефекту від можливої комерціалізації розробки 85
ВИСНОВКИ 92
ПЕРЕЛІК ПОСИЛАНЬ 93

Додаток А (обов’язковий). Технічне завдання..... 96
Додаток Б (обов’язковий). Ілюстративна частина 100
Додаток В (обов’язковий). Лістинг програмного забезпечення..... 109
Додаток Г (обов’язковий). Протокол перевірки на плагіат 125

ВСТУП

Актуальність. Оскільки головною метою трейдингу завжди є отримання стабільного прибутку та його підвищення в довгостроковій перспективі, необхідно враховувати те, що успіх в торгівлі фінансовими інструментами залежить від того, наскільки краще та швидше трейдер розумітиме зміни на ринку, тобто в тому числі зміни тенденцій на фінансових часових рядах.

Проблеми аналізу та прогнозування вивчалися вже протягом багатьох століть і залишаються актуальними і найближчим часом. Володіння ефективними інструментами аналізу надає велику перевагу, оскільки той, хто володіє найкращими засобами аналізу, отримує конкурентну перевагу і, отже, може розраховувати на більш передбачуваний прибуток.

Для досягнення найкращих результатів потрібно комбінувати існуючі методи аналізу ринків, що призводить до необхідності опрацювання обширного спектру ознак для ухвалення рішення. Популярні методи технічного аналізу, які широко використовуються для вивчення фінансового ринку, не завжди можуть враховувати всі аспекти, або цей процес може бути надто складним. У таких обставинах особливо важливим стає використання кластерного аналізу, який дозволяє ефективно зменшувати розмір великих обсягів соціально-економічної інформації, здійснювати їх конденсацію та зробити їх більш доступними.

Протягом останніх років кластерний аналіз і нейронні мережі стали широко використовуваними для аналізу фінансових показників, оскільки вони ефективно опрацьовують великий обсяг різноманітних показників і ознак одночасно.

Після огляду літературних джерел та наукових публікацій, виявлено, що обмежена увага приділяється використанню кластерного аналізу та нейронних мереж для ідентифікації конкретних закономірностей на фінансовому часовому ряді, у той час як їх головне використання в цій сфері зосереджується на прогнозуванні. Таким чином, є важливим проведення подальших досліджень в цьому контексті.

Метою дослідження є підвищення ефективності передбачення динаміки фінансового часового ряду шляхом ідентифікації залежностей його поведінки. Цей аналіз здійснюється на основі використання кластерного аналізу та розробленої системи прийняття рішень.

Об’єктом дослідження даної роботи є фінансовий часовий ряд.

Предмет дослідження – математична модель для ідентифікації залежностей на часових рядах. Для досягнення поставленої мети використані такі методи дослідження, як метод технічного аналізу фінансових інструментів, кластерні методи, метод статистичної обробки даних та методи прикладної математики.

Методи дослідження:

1. Аналіз та огляд різних підходів до кластерного аналізу та їх можливе використання у відношенні до обробки часових рядів.
2. Дослідження застосування кластерного аналізу в завданнях фінансового прогнозування..
3. Розробка математичної моделі для кластеризації вибраних об'єктів.
4. Створення системи прийняття рішень.
5. Перевірка розробленої системи на історичних даних певних фінансових інструментів.

Наукова новизна одержаних результатів.

1. Запропоновано новий підхід до обробки сигналів індикаторів, що ґрунтується на використанні методу кластерного аналізу, для визначення моментів прийняття рішень по зміні тенденції часового.
2. Розроблено математичну модель ідентифікації ринкових ситуацій, яка використовує методи кластерного аналізу та низку ознак та показників, як самого часового ряду, так і похідних від нього індикаторів.

Практичне значення одержаних результатів полягає у розробці алгоритму та методики ідентифікації ринкових залежностей на основі впровадженої математичної моделі. Цей підхід був конкретизований та

впроваджений у формі, яка може бути покладена в основу автоматичної системи прийняття рішень.

Апробація результатів дослідження: основні результати виконання магістерської кваліфікаційної роботи було подано в матеріали науково інтернет-конференції «LII Науково-технічна конференція підрозділів Вінницького національного технічного університету (2023)», м. Вінниця, 2023 р. Режим доступу: https://press.vntu.edu.ua/index.php/vntu/catalog/book/788

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Характеристика об’єкту досліджень

Часові ряди в аналізі даних використовуються для вивчення змін певної величини в часі. Це може бути будь-що, від цін на акції до температури повітря. Вони грають важливу роль у прогнозуванні та прийнятті рішень. Важливими концепціями є тренди, сезонність та випадкові величини, які впливають на часові ряди. Аналізуючи їх, можна отримати корисну інформацію для прийняття рішень у різних галузях, таких як економіка, фінанси та метеорологія.

Часові ряди можна розділити на стаціонарні та нестаціонарні. Стаціонарні часові ряди мають постійні статистичні властивості, що спрощує їхній аналіз. Нестаціонарні часові ряди, навпаки, можуть мати змінювані статистичні характеристики, що може ускладнити їхню інтерпретацію[1].

Аналіз часових рядів включає в себе використання різних методів, таких як експоненційне згладжування, авторегресійні моделі, а також методи для виявлення трендів і сезонних коливань. Важливою частиною аналізу є також визначення прогнозів та оцінка їх точності.

Часові ряди використовуються в різних галузях, включаючи економіку для прогнозування економічних показників, у фінансах для аналізу ринкових тенденцій та у науці про клімат для моделювання та прогнозування погоди. Вони є потужним інструментом для розуміння та передбачення змін у часі.

Важливими концепціями в аналізі часових рядів є тренди, які показують загальний напрямок змін, сезонність – періодичні коливання, і випадкові величини, які враховують випадкові чинники. Зрозуміти ці елементи дозволяє краще виявляти закономірності та прогнозувати майбутні значення.

Розділяють часові ряди на стаціонарні та нестаціонарні. Стационарні ряди мають постійні статистичні характеристики, що спрощує їхнє вивчення. Нестаціонарні ряди можуть демонструвати змінювані статистичні параметри, що ускладнює їхню обробку.

Використання різних методів аналізу, таких як експоненційне згладжування та авторегресійні моделі, дозволяє отримати більш точні прогнози та здійснити більш ефективний контроль за динамікою змін у часі. Часові ряди є необхідним інструментом для прийняття рішень та розуміння великого спектру явищ у різних галузях.

Часові ряди визначають динаміку подій у часі, надаючи можливість прогнозування та адаптації до змін. Вони широко використовуються в бізнесі для аналізу споживчої активності, прогнозування продажів і оптимізації ланцюга постачання. Крім того, у сфері медицини вони використовуються для вивчення тенденцій захворюваності та розробки стратегій лікування[1].

Важливим аспектом їх використання є можливість виявлення аномалій та попередження можливих проблем. Аналізуючи динаміку часових рядів, можна вчасно реагувати на зміни, що є важливим у сучасному динамічному середовищі.

Завдяки зростанню обчислювальних можливостей і розвитку алгоритмів машинного навчання, аналіз часових рядів стає все більш точним і автоматизованим. Це відкриває нові перспективи для вивчення складних взаємозв'язків у даних та розробки передових стратегій управління на основі часових змін.

Усе це підкреслює важливість подальшого дослідження та вдосконалення методів аналізу часових рядів, які відіграють ключову роль у сучасному аналітичному та стратегічному прийнятті рішень.

Часові ряди, фундаментально, є вікном у світ зміни та розвитку. Вони дозволяють нам розглядати події не лише в ізоляції, але й у контексті часу, виявляючи справжні патерни та динаміку. Це особливо актуально в умовах швидкозмінюваних ринків, де усі аспекти суспільства, економіки та науки піддаються постійному перетворенню[2].

Аналіз часових рядів є важливим інструментом для прогнозування та адаптації до майбутніх трендів. У світі бізнесу це означає здатність оперативно реагувати на зміни у попиті, розуміти сезонні коливання та ефективно управляти

ресурсами. В науці це відкриває можливості для розуміння еволюції явищ, таких як зміни клімату та пандемії.

Продовжуючи дослідження та вдосконалюючи методи аналізу часових рядів, ми можемо не лише краще розуміти минуле, а й здатні ефективніше приймати рішення в невизначеному майбутньому. Використання новітніх технологій, таких як штучний інтелект та глибоке навчання, надає нові горизонти для точного прогнозування та стратегічного планування.

Часові ряди, як засіб вивчення динаміки подій у часі, є ключовим інструментом для сучасного суспільства, що стикається з неспрогнозованими викликами та швидкими змінами. Вони стають основою для прийняття стратегічних рішень у різних галузях, де час визначає успіх чи невдачу.

Застосування аналізу часових рядів у науці важливе для розуміння тривалих тенденцій та прогнозування можливих наслідків змін. У галузі економіки це допомагає компаніям виявляти можливості для росту та уникати ризиків[3].

Постійний розвиток методів аналізу, врахування новітніх технологій та вдосконалення стратегій роботи з часовими рядами є важливим завданням для ефективного використання цього інструменту. Такий підхід відкриває шлях до нових можливостей для вивчення та передбачення явищ, що визначають наші життя, а також дозволяє нам краще розуміти і адаптуватися до складних змін в сучасному світі.

У галузі науки про дані, часові ряди є неоціненим ресурсом для вивчення інтервалів невизначеності. Вони вказують на моменти високого попиту, економічних зрушень, аномалій у здоров'ї громад та багато іншого. Адаптація новітніх методів аналізу та використання штучного інтелекту дозволяють нам усе точніше передбачати й реагувати на ці динамічні процеси[2].

Очевидно, що подальший розвиток аналізу часових рядів та їхнє вдосконалення важливі для розвитку суспільства. Це не тільки забезпечує нас конкретними інструментами для ефективного управління ризиками та прийняття

рішень, але і створює фундамент для подальших наукових досліджень та інновацій.

На рисунку 1.1 показано приклад дискретного часового ряду.

Рисунок 1.1 – Приклад дискретного часового ряду

Часові ряди можна класифікувати за різними ознаками, такими як стаціонарність, тренд, сезонність та випадкові коливання.

Стаціонарні часові ряди мають постійні статистичні характеристики, такі як середнє значення та варіабельність, протягом часу. Нестаціонарні ряди, навпаки, можуть виявляти зміни у цих характеристиках, що може ускладнити їх аналіз[3].

На рисунку 1.2 показано приклад стаціонарного часового ряду.

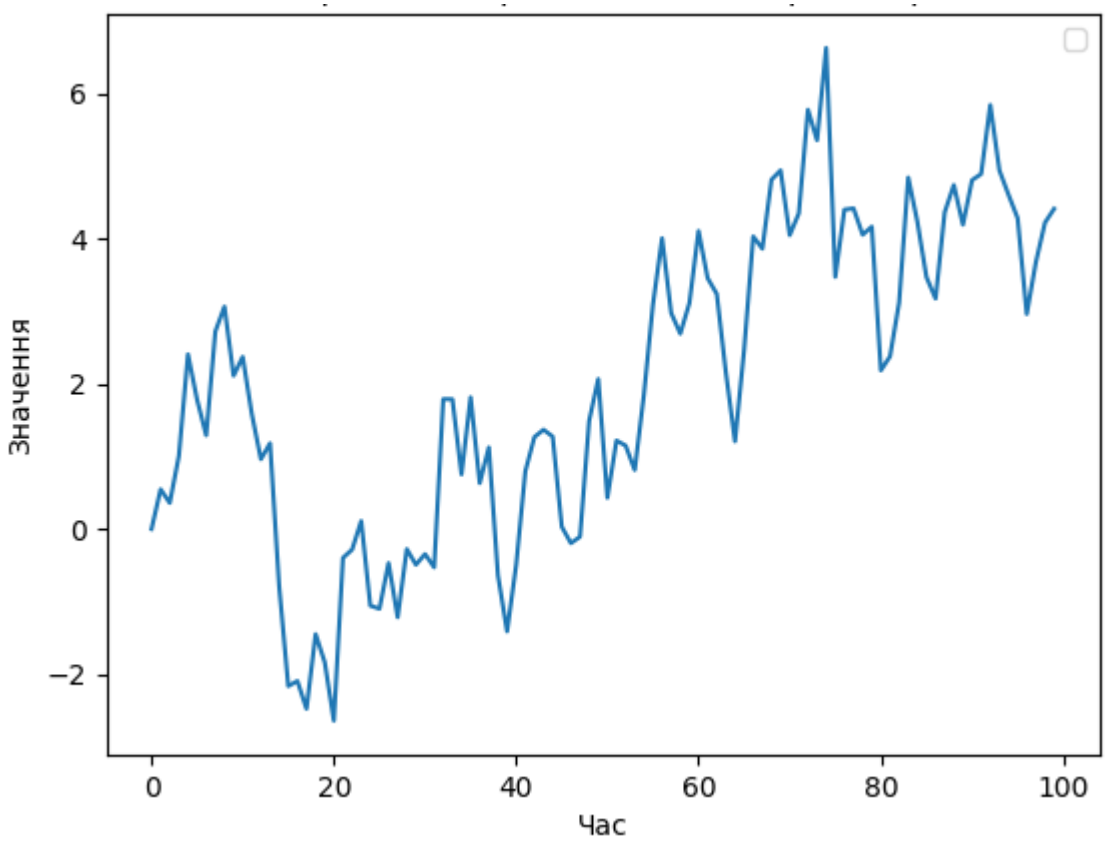


Рисунок 1.2 – Приклад стаціонарного часового ряду.

Тренд вказує на загальний напрямок змін у часовому ряді. Він може бути зростаючим, спадаючим або плоским, в залежності від змін у середньому значенні величини з часом.

Сезонні коливання повторюються в певні періоди, наприклад, щоденні, щотижневі або щорічні зміни. Вони можуть бути обумовлені різноманітними факторами, такими як погода, свята чи шкільні канікули.

Випадкові коливання представляють собою випадкові впливи або шум у часовому ряді, які не піддаються систематичному аналізу та прогнозу[4].

Залежно від конкретного дослідження чи задачі, може виникати потреба враховувати всі ці аспекти для точного аналізу та прогнозу часових рядів в різних галузях.

1.2 Кластеризація часових рядів

Кластеризація часових рядів - це процес групування схожих часових рядів разом з метою виявлення певних закономірностей, шаблонів або характеристик. Цей метод важливий в аналізі даних та прогнозуванні, де велика кількість часових рядів може бути важко аналізувати індивідуально. Кластеризація може допомогти зрозуміти подібність між рядами та групувати їх залежно від їхніх характеристик.

На рисунку 1.3 показано приклад часового ряду що подається на перевірку.

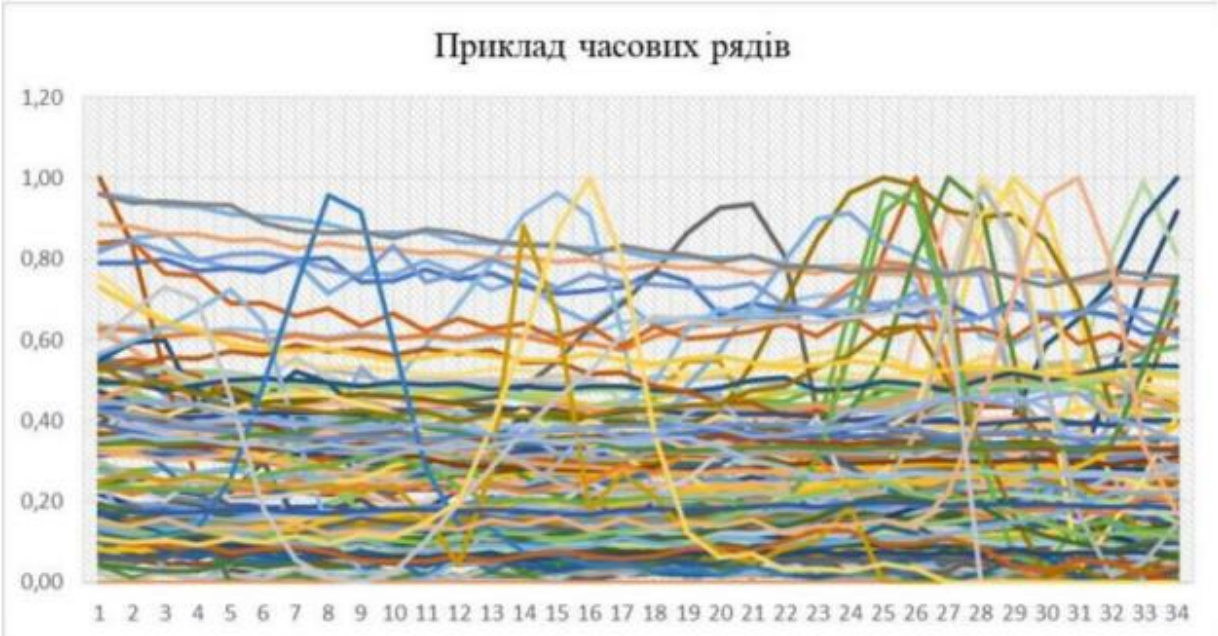


Рисунок 1.3 – Приклад послідовних даних, що подаються на обробку.

Для кластеризації часових рядів можна використовувати різні методи, такі як методи знаходження схожості (наприклад, кореляційні коефіцієнти), методи агломеративної чи роздільної кластеризації, або методи машинного навчання, такі як кластеризація K-середніх.

Зазвичай, процес кластеризації часових рядів включає в себе такі етапи, як підготовка даних, вибір метрик схожості, визначення кількості кластерів, виконання самої кластеризації та оцінка результатів. Отримані кластери можуть вказати на різні зміни, цикли або тенденції в різних групах часових рядів[5].

Кластеризація часових рядів також є актуальною в області інтернету речей (IoT), де датчики та пристрої генерують велику кількість данних в реальному часі. Аналіз цих часових рядів може допомогти виявити аномалії, прогнозувати витрати енергії, покращувати ефективність систем та розуміти патерни споживання ресурсів.

На рисунку 1.4 показано графічне відображення кластеризованого часового рядку.

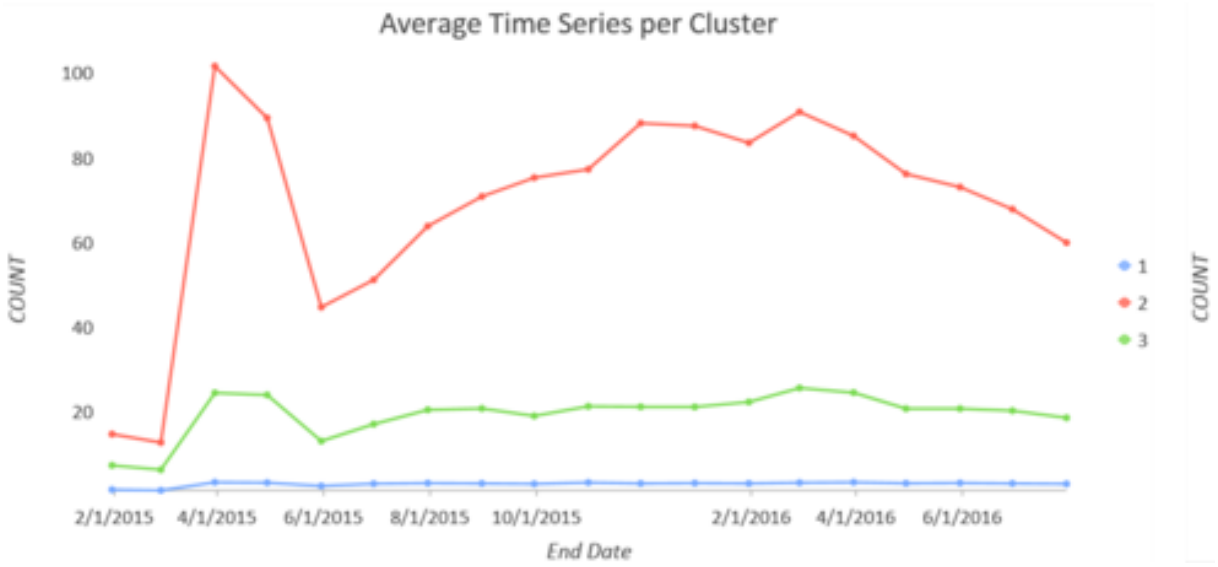


Рисунок 1.4 – Кластеризований часовий ряд

Однією з ключових вигод кластеризації часових рядів є здатність автоматизувати процеси та виділяти схожі характеристики без необхідності ручного аналізу. Це особливо важливо в умовах великих обсягів даних, де ручний аналіз стає часово і ресурснозатратним завданням.

Важливим етапом в кластеризації часових рядів є визначення оптимального методу та параметрів кластеризації, щоб отримати найкращі результати. Це може включати в себе експерименти з різними алгоритмами, визначенням оптимальної кількості кластерів, та врахуванням особливостей конкретного домену дослідження[4].

Додатково, важливо зазначити, що використання методів кластеризації часових рядів відкриває можливості для удосконалення стратегій прогнозування.

Результати кластеризації можуть бути використані для розробки індивідуальних моделей прогнозування для кожного кластеру, що підвищує точність прогнозування в порівнянні з універсальними моделями.

Також важливо враховувати, що в процесі кластеризації часових рядів можна ефективно використовувати інформацію про контекст. Наприклад, враховуючи взаємодію між рядами, можна отримати більш точні та значущі кластери. Інтеграція контекстуальної інформації може покращити якість кластеризації та робити отримані результати більш інтерпретованими.

У світлі зростаючої кількості даних в реальному часі, особливо в контексті Інтернету речей, важливо розвивати ефективні методи кластеризації, які можуть обробляти стрім даних. Спроможність адаптуватися до великого обсягу даних у реальному часі робить кластеризацію часових рядів надзвичайно корисною в сучасних областях, де швидкість обробки і аналізу грають ключову роль[5].

Нарешті, важливим аспектом є постійне вдосконалення методів кластеризації часових рядів, оскільки це дозволяє враховувати сучасні технологічні та методологічні розробки. Здатність адаптувати та вдосконалювати алгоритми дозволяє забезпечити актуальні та точні результати, що має велике значення в сферах дослідження та практичного використання.

1.3 Класифікація методів кластеризації часових рядів

Існує кілька типів алгоритмів кластеризації, кожен з яких має свої унікальні особливості та використовується в різних сферах досліджень та застосувань. На рисунку 1.5 показано загальну класифікацію підходів до кластеризації часових рядів.

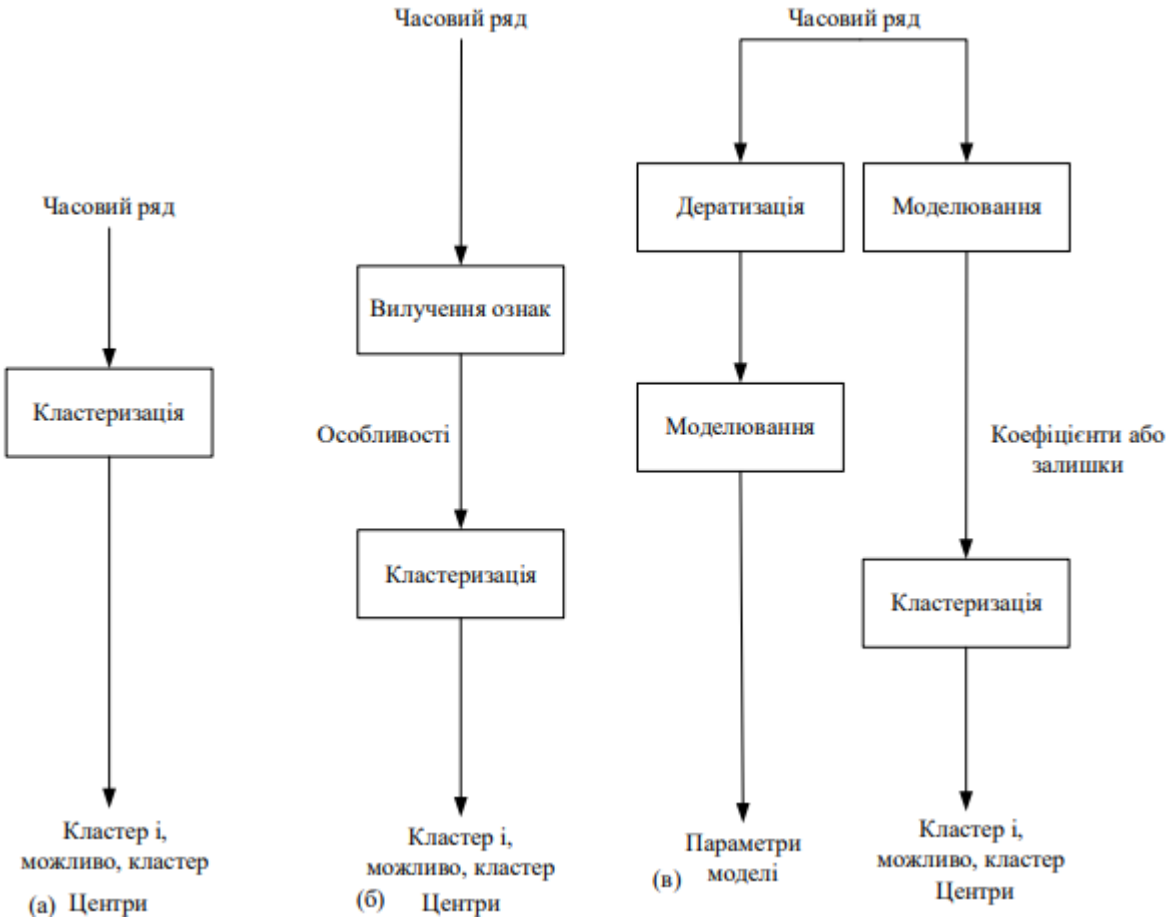


Рисунок 1.5 – Класифікація підходів до кластеризації часових рядів.

Кластеризація K-середніх - це алгоритм машинного навчання, який використовується для групування схожих об'єктів в наборі даних у K кластерів. Основна ідея полягає в розділенні точок даних на групи таким чином, щоб в межах кожного кластера об'єкти були схожими між собою, а відмінності між кластерами були максимальними[6].

На рисунку 1.6 показано приклад роботи алгоритму K-середніх.

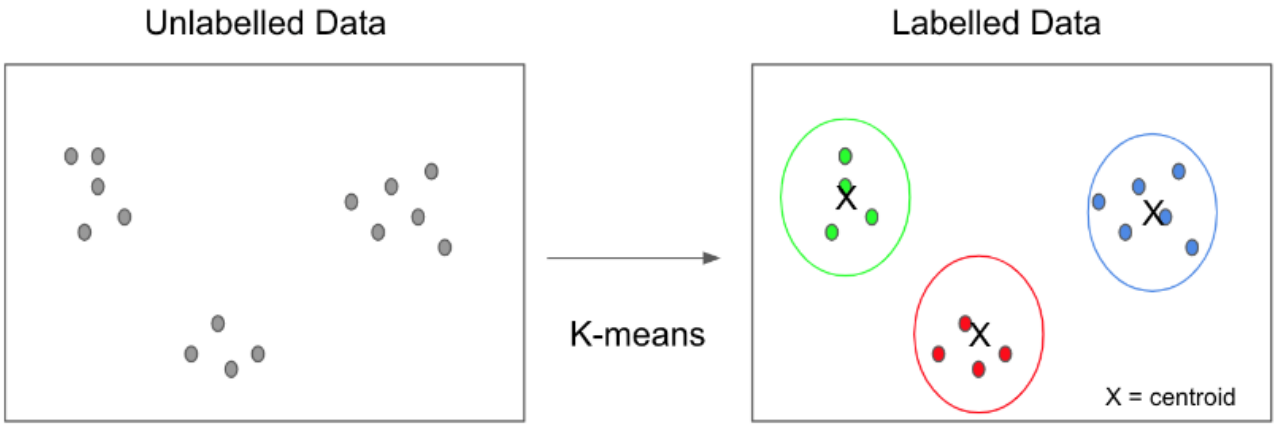


Рисунок 1.6 – Принцип роботи алгоритму K-середніх.

Процес роботи алгоритму K-середніх починається з вибору K початкових центрів кластерів. Потім кожна точка даних призначається тому кластеру, центр якого знаходиться найближче. Після цього центри кластерів перераховуються, взявши середнє значення всіх точок у кожному кластері. Цей процес повторюється до тих пір, поки центри кластерів не стабілізуються або досягнеться максимальна кількість ітерацій.

Критерії вибору центрів кластерів та визначення кількості кластерів є ключовими частинами алгоритму. Оптимальні значення K можуть визначатися на підставі експертного досвіду, або використовуватися методи, такі як "ліктеве правило", що базується на визначенні точки, де спостерігається різке зменшення дисперсії між точками та центроїдами[6].

Алгоритм K-середніх є ефективним і широко використовується в аналізі даних, візуалізації та вирішенні завдань групування. Його простота та швидкодія робить його популярним в багатьох областях, таких як маркетингові дослідження, біологія, фінанси та інші.

Ієрархічна кластеризація - це метод групування об'єктів, що використовується для створення ієрархії кластерів. У цьому підході кожен об'єкт спочатку вважається окремим кластером, а потім об'єднується чи розділяється з іншими кластерами до досягнення бажаного стану. Одна з головних відмінностей між агломеративною та дивізійною ієрархічною кластеризацією

DBSCAN, або Density-Based Spatial Clustering of Applications with Noise, є алгоритмом кластеризації, який базується на густині точок у просторі ознак. Він відзначається своєю здатністю виявляти кластери будь-якої форми та розміру, а також здатністю виділяти окремі точки як шум. Робота алгоритму ґрунтується на оцінці густини точок та їх просторовому розташуванні.

DBSCAN визначає кластери наступним чином: для кожної точки визначається його окіл з вказаною радіусом і мінімальною кількістю точок у цьому околі. Якщо кількість точок у цьому околі більша за задане мінімальне значення, то вони утворюють кластер. Таким чином, точки, які знаходяться достатньо близько одна до одної, вважаються частиною кластера. Крім того, точки, які не належать жодному кластеру та не мають достатньої кількості сусідів, вважаються шумовими[7].

Важливою особливістю DBSCAN є його здатність автоматично виявляти кластери будь-якої форми та виявляти аномальні точки, які не входять до жодного кластера. Це робить його ефективним у випадках, коли кластери можуть мати складні форми та розміри, а також коли є присутність шуму в даних.

Однією з великих переваг DBSCAN є те, що він не потребує заздалегідь визначеної кількості кластерів у даних. У порівнянні з іншими алгоритмами кластеризації, такими як k-середні, де користувач повинен вказати кількість кластерів перед виконанням алгоритму, DBSCAN визначає кількість кластерів автоматично на основі густини даних.

На рисунку 1.7 показано принцип роботи DBSCAN.

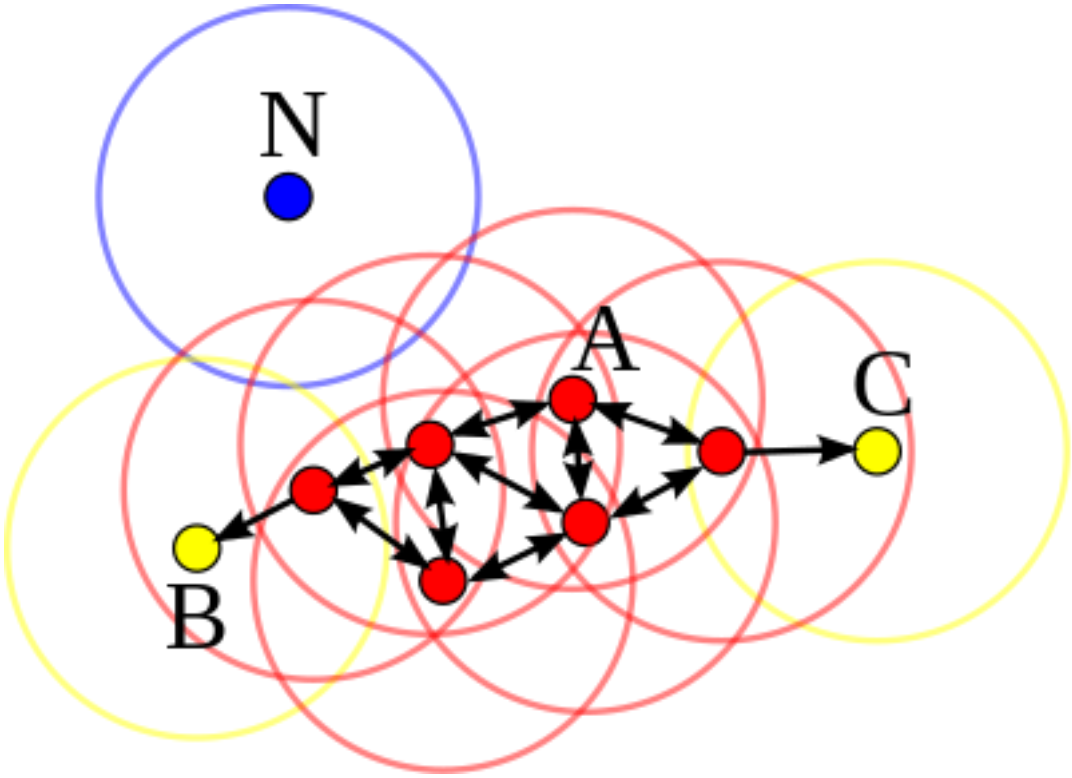


Рисунок 1.7 – Принцип роботи алгоритму DBSCAN

Однак алгоритм також має свої обмеження. Наприклад, йому важко впоратися з кластерами різної густини, і він може мати проблеми з визначенням кластерів у випадках, коли границі між ними неправильно визначені. Також, вибір параметрів, таких як радіус і мінімальна кількість точок для визначення кластера, може суттєво впливати на результат.

Усупереч цим обмеженням, DBSCAN залишається ефективним інструментом для кластеризації в задачах, де інші алгоритми можуть показати менш задовільні результати. Його використання особливо доцільне у випадках, коли форма та розмір кластерів невідомі, і коли важливо виділити якнайбільше структури в даних[8].

Спектральна кластеризація є потужним методом кластеризації, який використовує інформацію про граф співвідношень між об'єктами для розділення їх на групи або кластери. У контексті кластеризації часових рядів без пунктів та підпунктів, спектральна кластеризація може бути корисною, оскільки вона

дозволяє аналізувати схожість часових рядів на основі їх спектральних властивостей.

Основна ідея полягає в тому, щоб побудувати граф, в якому вузли представляють об'єкти (у цьому випадку - часові ряди), а ребра відображають ступінь подібності між ними. Зазвичай, для часових рядів, подібність може вимірюватися за допомогою кореляції, відстані Мінковського або інших метрик схожості.

Рисунок 1.8 показано принцип роботи спектральної кластеризації.

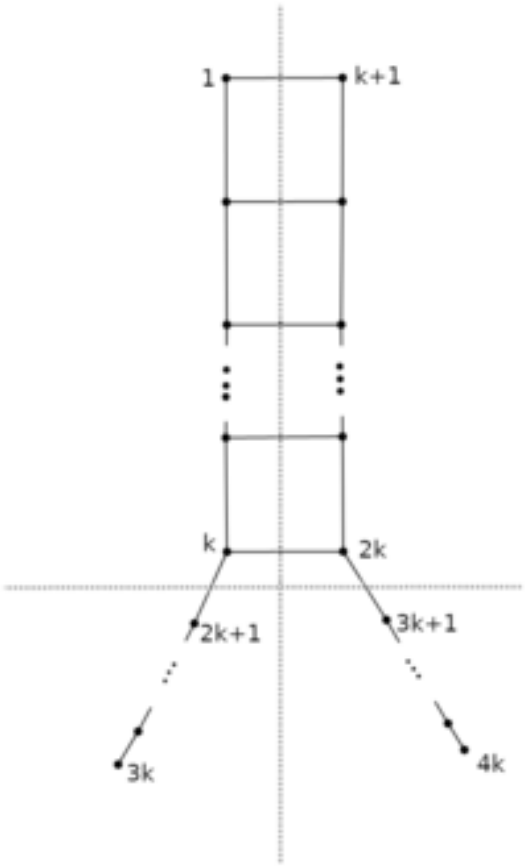


Рисунок 1.8 – Принцип роботи ієрархічної кластеризації.

Ієрархічна кластеризація може бути використана для створення дендрограми, яка візуально відображає структуру ієрархії кластерів. Кожен вузол дендрограми представляє собою кластер, а віддаленість між вузлами вказує на схожість чи відмінність між кластерами.

Принцип роботи алгоритму показано на рисунку 1.9.

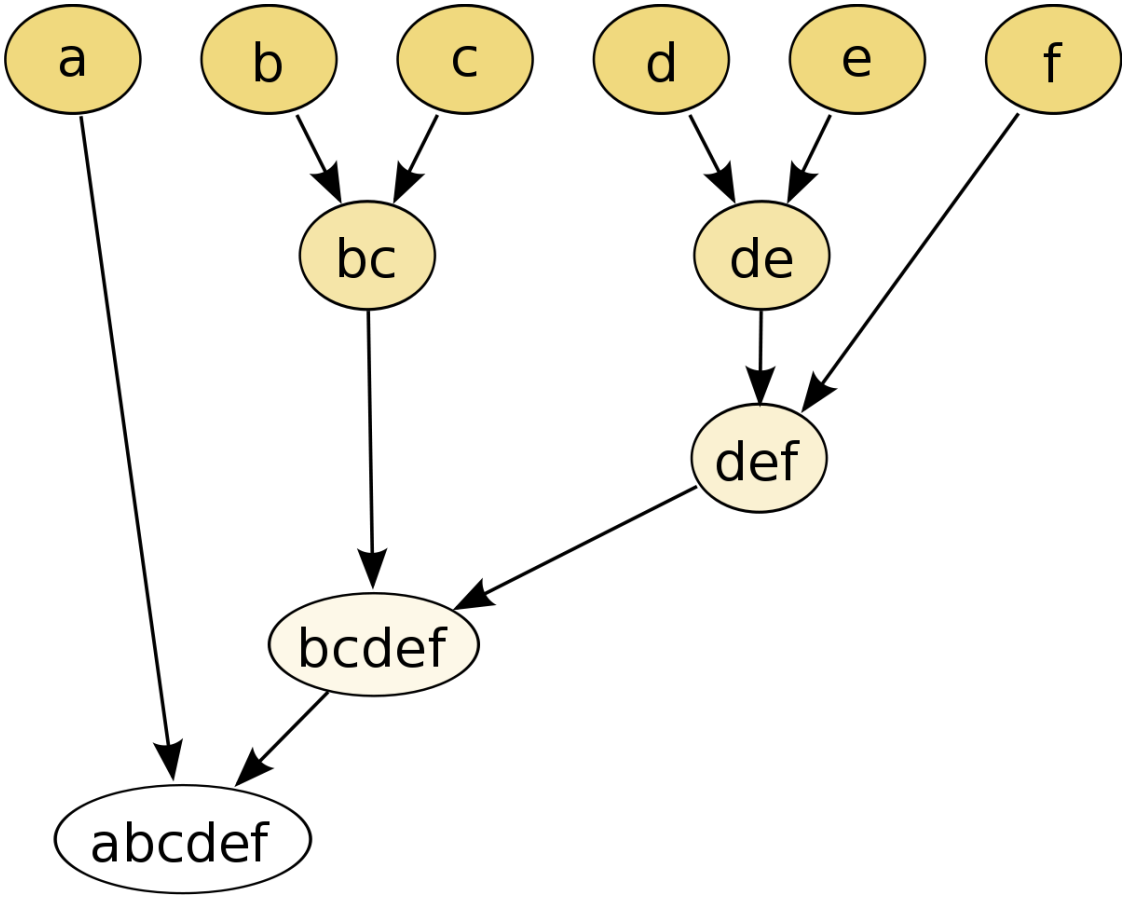


Рисунок 1.9 – Принцип роботи алгоритму ієрархічної кластеризації.

Агломеративна ієрархічна кластеризація починається з того, що кожен об'єкт розглядається як окремий кластер. На кожному кроці об'єднуються два найближчих кластера, і такий процес триває до того моменту, коли всі об'єкти утворюють один великий кластер.

Дивізійна ієрархічна кластеризація, навпаки, починається з того, що всі об'єкти належать до одного кластера, а потім на кожному кроці відбувається розділення кластерів на менші. Цей процес триває до досягнення бажаного рівня деталізації кластерів[9].

Ієрархічна кластеризація широко використовується в областях, де важливо враховувати структурні зв'язки між кластерами та візуалізувати їхню ієрархію. Цей метод може бути ефективним для виявлення подібностей та різниці між

підгрупами даних, що допомагає у кращому розумінні структури досліджуваних об'єктів.

Ще однією важливою характеристикою ієрархічної кластеризації є можливість адаптації до різних рівнів деталізації. Залежно від конкретного застосування та завдань дослідження, ієрархічні методи можуть бути налаштовані для створення більш загальних або деталізованих кластерних структур.

Цей підхід особливо корисний у випадках, коли об'єкти в даних можуть організовуватися на різних рівнях подібності. Наприклад, у біології ієрархічна кластеризація може використовуватися для класифікації організмів на великі таксономічні групи та більш дрібні підгрупи, що відображають більш детальні схожості[10].

Також важливо відзначити, що ієрархічна кластеризація може бути обчислювально витратною, особливо при роботі з великими обсягами даних. Тому ефективність і швидкодія алгоритму можуть залежати від конкретної реалізації та властивостей набору даних.

Fuzzy C-means (FCM) - це метод кластеризації, який використовує техніку "розмиття" (fuzzification), щоб дозволити об'єктам даних приналежати не тільки одному, а й усім кластерам з певною ймовірністю чи ступенем належності. У порівнянні з традиційними методами, де об'єкт повністю належить одному кластеру, FCM надає більш гнучкий підхід до моделювання невизначеності в даних.

Основна ідея FCM полягає в тому, щоб кожен об'єкт матеріально належав кожному кластеру, але з різними ступенями належності. Алгоритм мінімізує функціонал, який враховує відстані між об'єктами і центрами кластерів, враховуючи ймовірність належності[9].

FCM добре підходить для задач, де об'єкти можуть мати неоднозначні чи перехрещені принципи належності до кластерів. Наприклад, в медичному аналізі він може бути використаний для класифікації пацієнтів на групи за ступенем

подібності до різних захворювань, де одна людина може мати ознаки, що вказують на кілька захворювань одночасно.

На рисунку 1.10 показано принцип роботи Fuzzy C-means (FCM) кластеризації.

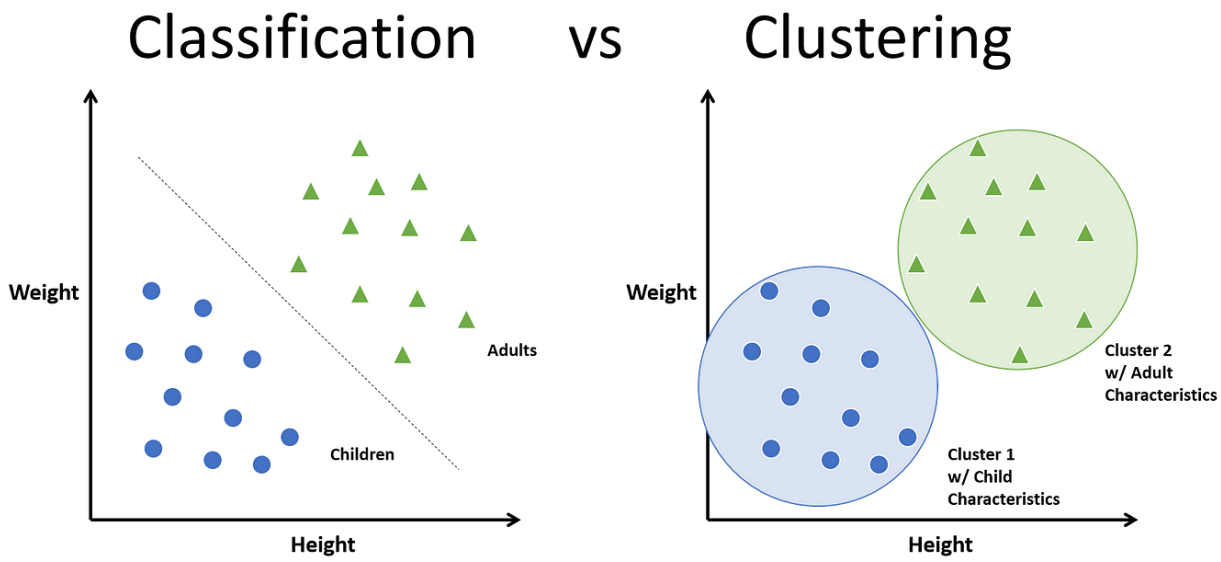


Рисунок 1.10 – Принцип роботи алгоритму Fuzzy C-means (FCM) кластеризації.

Принцип роботи Fuzzy C-means (FCM) полягає в тому, щоб кожен об'єкт даних приналежав не тільки одному кластеру, але й усім кластерам з різними ступенями належності. Цей метод дозволяє враховувати невизначеність в даних та розглядати об'єкти з певною ймовірністю належності до різних кластерів.

Алгоритм починається з ініціалізації параметрів, включаючи кількість кластерів та початкові центри. На кожній ітерації алгоритму визначаються ступені належності для кожного об'єкта до кожного кластера, з урахуванням відстаней між об'єктом і центрами кластерів. Потім центри кластерів оновлюються на основі зважених значень об'єктів, де ваги - це розраховані ступені належності[10].

Процес ітерацій повторюється до досягнення збіжності, тобто до тих пір, поки ступені належності та центри кластерів не залишаться стабільними.

Оптимальні значення ступенів належності та центрів кластерів визначаються так, щоб мінімізувати цільову функцію, що враховує відстані між об'єктами та центрами кластерів.

FCM дозволяє враховувати невизначеність та гнучко вирішувати завдання кластеризації в умовах, де об'єкти можуть мати багатозначні приналежності до різних груп.

FCM є особливо ефективним для вирішення завдань, де границі між кластерами нечітко визначені або коли об'єкти можуть відноситися до кількох груп одночасно. Цей підхід важливий в тих випадках, коли чітке приналежність до одного кластеру не може адекватно відобразити структуру даних.

Однією з переваг Fuzzy C-means є його здатність враховувати вагомість об'єктів при визначенні центрів кластерів. Ступені належності служать ваговими коефіцієнтами для об'єктів під час оновлення центрів. Це може бути корисним, коли деякі об'єкти мають більший вплив або важливість для кластеризації.

Однак слід враховувати, що FCM може бути чутливим до вибору початкових значень та параметрів, таких як параметр "fuzziness" (ступінь розмитості). При неправильному налаштуванні ці параметри можуть призвести до нестійких результатів або витратних обчислень[11].

Онлайн кластеризація - це метод групування даних в режимі реального часу, де модель кластеризації постійно оновлюється з потоком нових даних. Це дозволяє ефективно адаптувати модель до змінюючогося характеру даних і негайно виявляти нові патерни чи кластери. Основні переваги цього підходу включають можливість обробки великих потоків даних в реальному часі та адаптивність до змін в структурі даних. Онлайн кластеризація часто застосовується в сферах, де важлива оперативна реакція на нову інформацію, таких як моніторинг, аналітика в реальному часі та системи обробки стрімів даних.

Цей метод кластеризації особливо корисний в сценаріях, де дані надходять безперервно, і важливо оперативно аналізувати їхню структуру. Онлайн кластеризація здатна пристосовуватися до різноманітних змін, таких як

зростання або спад інтенсивності потоку даних, або зміни у характері з'являючихся кластерів.

Однією з основних вимог до алгоритмів онлайн кластеризації є ефективність, оскільки вони повинні працювати в реальному часі. Деякі алгоритми використовують прийоми для зменшення обчислювального бар'єру, такі як обмежені обчислення чи апроксимації, щоб забезпечити швидку реакцію на нові дані[12].

Онлайн кластеризація також знаходить своє застосування в інтерактивних системах, аналізі стріму соціальних мереж, моніторингу великих мереж датчиків та в інших областях, де важлива миттєва обробка та реакція на зміни.

1.4 Постановка задачі дослідження

Розглядаючи важливість завдання кластеризації часових рядів у сфері інтелектуального аналізу даних, важливо акцентувати увагу на проблемах та особливостях цього процесу. У контексті сучасного світу, адаптація технологій та алгоритмів до умов сучасних вимог вимагає уважного розгляду.

Об'єктом дослідження є послідовність даних у форматі часових рядів. Метою є розробка методу кластеризації, який базується на використанні непрямих ознак часового ряду. Ці ознаки дозволяють ефективно розподіляти вхідні дані на класи – групи подібних екземплярів вибірки для подальшого передбачення зміни тенденції.

1.5 Висновки

В данному розділі було проведено аналіз предметної області, пов'язаної із використанням часових рядів у задачах кластеризації. Розглянуто основні аспекти використання часових рядів, визначено ключові поняття та методи, зокрема кластеризацію часових рядів. Проведений огляд методів кластеризації надав чітке уявлення про різноманітність підходів до цієї задачі.

На основі проведеного аналізу була сформульована постановка задачі дослідження, яка визначає цільові напрямки та завдання, що вирішуються у подальших розділах роботи. Отже, розділ надає чітку основу для подальших розділів дослідження, визначаючи контекст, у якому вирішуються конкретні завдання та формулюються рекомендації для вирішення проблем в обраній предметній області.

2. АНАЛІЗ АЛГОРИТМІВ ПОПЕРЕДНЬОЇ ОБРОБКИ ЧАСОВИХ РЯДІВ ТА АЛГОРИТМІВ ОНЛАЙН КЛАСТЕРИЗАЦІЇ

2.1. Перевірка статистичних гіпотез властивостей часового ряду у процесі попереднього аналізу вибірки

Рівність часових рядів можуть включати аномальні значення, які можуть виникнути внаслідок помилок у процесі збору, запису або передачі інформації. Ці помилки можуть бути технічного характеру або виникати в результаті перепадів, які не відображають будь-якої систематичної тенденції. Важливо відрізняти такі аномалії від реальних процесів, таких як різкі зміни валютного курсу, які вважаються аномальними значеннями другого роду і не піддаються виправленню.

Для оцінки спостережень часового ряду формулюються дві гіпотези:

- H0: і-е спостереження не є аномальним;
- H1: і-е спостереження є аномальним.

Математично це можна виразити як обчислення середнього значення (μ) та стандартного відхилення (σ) для часового ряду та розрахунок відстані між кожним спостереженням і середнім значенням у виразі Ірвіна.

Додатковий контекст важливий для розуміння методу Ірвіна та виявлення аномальних значень у часових рядах. При використанні цього методу важливо враховувати, що він базується на статистичних величинах, таких як середнє значення та стандартне відхилення[13].

Після розрахунку значення Ірвіна для кожного елемента часового ряду можна встановити поріг, за яким будь-яке значення, що перевищує цей поріг, вважається аномальним. Величина порогу може визначатися емпірично або за допомогою статистичних методів.

Такий підхід дозволяє відокремити технічні помилки від реальних аномалій у часових рядах, сприяючи точнішому аналізу даних. Важливо враховувати контекст дослідження та особливості конкретної області

застосування цього методу для належного визначення аномальних значень у часових рядах.

На рисунку 2.1 показано приклад використання алгоритмів для виявлення аномалій в часових рядах.

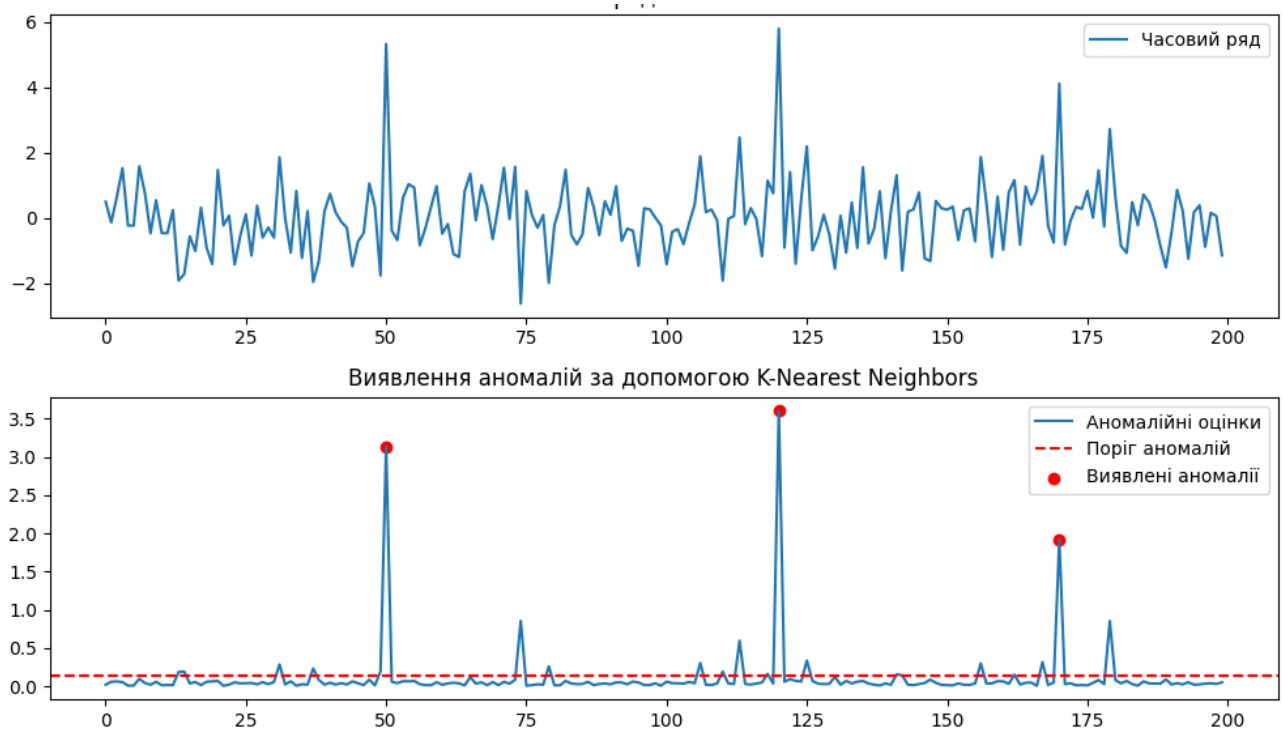


Рисунок 2.1 – Приклад виявлення аномалій.

Гіпотеза про постійність середнього значення послідовності є однією з основних при перевірці наявності невипадкової величини часового ряду, такої як тренд. Розглянемо дві статичні гіпотези, що використовуються при цьому аналізі:

Гіпотеза про стаціонарність передбачає, що статистичні характеристики часового ряду залишаються постійними з часом. Це означає, що середнє значення, дисперсія та інші параметри ряду не змінюються від періоду до періоду[14].

Гіпотеза про нестаціонарність, навпаки, вказує на те, що статистичні характеристики ряду змінюються з часом. Це може бути викликано наявністю тренду, сезонності чи інших систематичних варіацій в часовому ряді. Гіпотеза

про нестаціонарність часто враховує можливість наявності тренду чи циклічності в динаміці даних[13].

Для перевірки цих гіпотез часто використовують статистичні методи, такі як тест Дікі-Фуллера. Якщо результат тесту вказує на відхилення від гіпотези про стаціонарність, це може свідчити про наявність тренду або інших систематичних змін у часовому ряді.

Враховуючи результати тесту, можна зробити висновок щодо наявності чи відсутності тренду в часовому ряді, що є важливим для подальшого аналізу та прогнозування.

Для успішного використання векторів-ознак у статистичних дослідженнях важливо провести ретельну обробку та аналіз початкових даних. Цей етап включає в себе виявлення потенційних грубих помилок у дослідженні, а також виявлення помилок, що можуть виникнути при кодуванні і трансформації даних. Додатково, важливо виявити можливі шуми або аномалії у спостереженнях.

Одним із ключових етапів в цьому процесі є формування векторів-ознак на основі статистичних методів. Це включає в себе визначення релевантних характеристик даних, які найкраще відображають їхню структуру та властивості[15].

Також важливим етапом обробки даних для успішного використання векторів-ознак є нормалізація та стандартизація даних. Це допомагає уникнути проблем, пов'язаних з різницею в масштабах та варіації між різними характеристиками. Нормалізація зазвичай використовується для приведення значень ознак до діапазону від 0 до 1, щоб усунути вплив різниці в масштабах. Стандартизація, у свою чергу, дозволяє зробити розподіл значень ознак більш нормальним (з середнім значенням 0 і стандартним відхиленням 1).

Додатково, важливо враховувати взаємозв'язки між ознаками та відбирати ті, які найбільше впливають на цільову змінну. Це може включати аналіз кореляцій, використання методів відбору ознак, таких як дерева рішень чи алгоритми важливості ознак, для визначення вагомості кожної ознаки в моделі.

Окрім того, слід враховувати можливість виникнення пропущених даних та вибрати стратегію їх обробки, яка найкраще підходить до конкретного дослідження (наприклад, видалення, заміна середнім значенням, інтерполяція тощо).

Крім того, важливим етапом у формуванні векторів-ознак є обробка категоріальних ознак. Категоріальні дані, такі як типи продуктів, кольори або регіони, потребують спеціальної обробки для використання у моделях машинного навчання. Це може включати кодування категоріальних ознак у числовий формат за допомогою методів, таких як one-hot encoding чи label encoding[15].

Також важливо враховувати можливі викиди або виняткові значення в даних, які можуть впливати на результати аналізу. Виявлення та обробка цих викидів допомагає покращити стабільність та достовірність моделей.

Додатково, у випадках великої кількості ознак може бути корисним використання методів зменшення розмірності, таких як метод головних компонент (PCA). Це дозволяє зберегти основні властивості даних, зменшуючи кількість ознак і сприяючи уникненню перевантаження моделі[16].

Загалом, ретельна обробка та аналіз початкових даних, включаючи нормалізацію, обробку категоріальних ознак, виявлення викидів та використання методів зменшення розмірності, створюють підґрунтя для ефективного використання векторів-ознак у статистичних дослідженнях та моделях машинного навчання.

Статистичні методи дозволяють враховувати розподіл та взаємозв'язки між різними змінними, що сприяє точнішому виокремленню інформації та зменшенню впливу шуму у векторах-ознак.

Програмну реалізацію формування векторів-ознак для часового ряду з використанням мови програмування Python показано на рисунку 2.2.

```
time_series = np.sin(np.linspace(0, 4 * np.pi, n_samples)) + np.random.normal(loc=0, scale=0.1, size=n_samples)

# Функція для формування векторів-ознак на основі статистичних методів
def extract_features(time_series):
    features = []

    # Додамо різні статистичні характеристики
    features.append(np.mean(time_series))  # Середнє значення
    features.append(np.std(time_series))   # Середньоквадратичне відхилення
    features.append(np.median(time_series))  # Медіана
    features.append(np.max(time_series))   # Максимальне значення
    features.append(np.min(time_series))   # Мінімальне значення

    # Додамо можливі інші характеристики
    features.append(np.percentile(time_series, 25))  # 25-й персентиль
    features.append(np.percentile(time_series, 75))  # 75-й персентиль
    features.append(np.var(time_series))   # Дисперсія
    features.append(pd.Series(time_series).autocorr())  # Автокореляція

    return features

# Формування векторів-ознак
feature_vector = extract_features(time_series)

# Друк векторів-ознак
print("Вектори-ознак:")
print(feature_vector)

# Візуалізація часового ряду
plt.figure(figsize=(10, 4))
plt.plot(time_series, label='Часовий ряд')
plt.title('Часовий ряд')
plt.legend()
plt.show()
```

```
Вектори-ознак:
[-0.004077096517208511, 0.7004049017482131, -0.0386604840196475, 1.2182668319822736, -1.26119577981025, -0.687610412672853, 0.662063529598047, 0.4905670263929241, 0.979736179332170
2]
```

Рисунок 2.2 – Програмна реалізація алгоритму формування ознак на Python.

Код на рисунку 2.2 генерує часовий ряд, а потім використовує статистичні методи для створення векторів-ознак для цього часового ряду. Вектори-ознак, які створюються цим кодом, містять різні статистичні характеристики, включаючи середнє значення, середньоквадратичне відхилення, медіану, максимальне значення, мінімальне значення, 25-й та 75-й персентиль, дисперсію та автокореляцію[17].

Ці вектори-ознак можна використовувати для різних завдань машинного навчання, таких як прогнозування, класифікація та кластеризація. Наприклад, можна використовувати ці вектори-ознак для прогнозування майбутнього значення часового ряду. Також можна використовувати ці вектори-ознак для класифікації часового ряду в один із декількох класів. Крім того, можна

використовувати ці вектори-ознак для кластеризації часових рядів на основі їх схожості.

Основні підходи до формування ознак, що базуються на статистичних дослідженнях, включають в себе ефективні методи аналізу даних. Один із таких методів є регресійний аналіз, і серед його інструментів використовується метод найменших квадратів. Цей метод дозволяє знаходити оптимальні параметри моделі, які найкраще відповідають спостережуваним даним.

Ще одним важливим методом є метод максимальної правдоподібності, який базується на пошуку параметрів моделі так, щоб ймовірність спостережень була максимальною. Це дозволяє ефективно оцінювати параметри та робити прогнози на основі ймовірнісних розподілів[18].

Додатково, методи згладжування використовуються для зменшення впливу випадкових аномалій або шумів у даних, що може покращити точність аналізу. Ці методи враховують закономірності у данних, використовуючи фільтрацію та усереднення для стабілізації та поліпшення якості векторів-ознак.

Такі статистичні методи формування ознак є важливою складовою аналізу даних, допомагаючи отримати достовірні та інформативні результати у різних областях дослідження.

У разі наявності тренду в даних просте згладжування може відстежувати його затримку або вимагати встановлення значень, близьких до 1. У таких випадках застосування подвійного експоненційного згладжування є доцільним. Цей метод використовує два рівняння: одне для оцінки тренду через різницю між поточним і попереднім згладженими значеннями, а інше для згладжування самого тренду[19].

На рисунку 2.3 показано приклад використання ковзкої середньої та експоненціальним раніше методів згладжування.

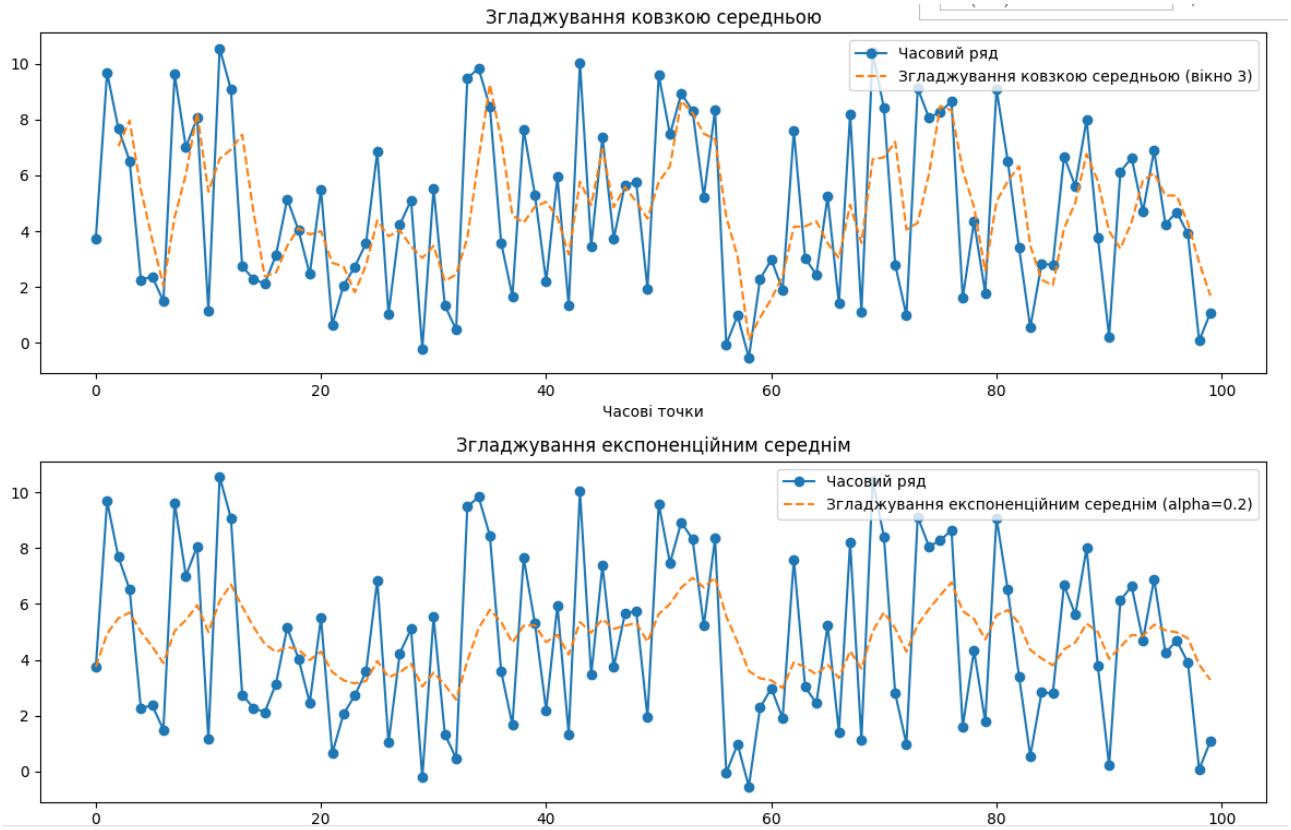


Рисунок 2.3 – Приклад використання методів згладжування ковзкої середньої та експоненціальним

Третє рівняння додається в метод потрійного згладжування, яке враховує сезонність у даних. Існують два варіанти сезонного компонента: адитивний, де амплітуда не залежить від базової амплітуди ряду, та мультиплікативний, де амплітуда змінюється разом із зміною базової амплітуди низки.

На рисунку 2.4 показано приклад роботи методу потрійного згладжування.

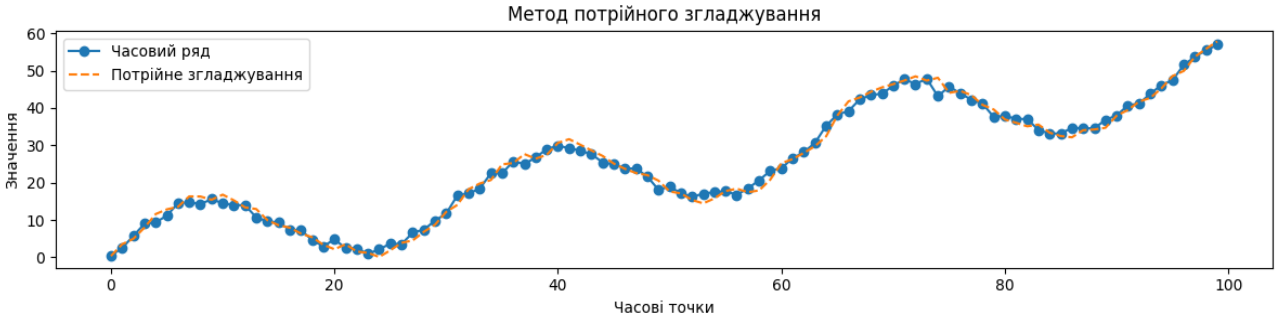


Рисунок 2.4 – Приклад використання методу потрійного згладжування.

У контексті обробки часових рядів важливо враховувати особливості використання та реалізації цих методів. Використання експоненційного згладжування, зокрема, може бути вигідним, оскільки воно призначене для аналізу даних в режимі реального часу, оброблюючи кожну точку ряду та адаптуючись до їхньої динаміки.

2.2. Методи зменшення розмірності

Зменшення розмірності відноситься до процесу зменшення кількості змінних або ознак у наборі даних. Це може бути важливим етапом в аналізі даних та машинному навчанні з декількох причин. Зокрема, це допомагає уникнути перевантаження моделі, зменшити вимірювальну складність та покращити продуктивність. Ось деякі методи зменшення розмірності[20]:

Вбудований відбір ознак - це метод відбору ознак у машинному навчанні, де сам алгоритм визначає, які ознаки є найбільш важливими під час навчання моделі. Це може бути здійснено різними алгоритмами, такими як дерева рішень, метод опорних векторів, лінійні моделі, ансамблеві методи чи нейронні мережі. Вбудований відбір ознак дозволяє моделі автоматично взаємодіяти з відбором ознак, що може полегшити процес і зменшити ризик перевантаження моделі.

Однак варто враховувати обчислювальну складність цього підходу, особливо при роботі з великими обсягами даних.

Вбудований відбір ознак використовується для автоматичного визначення того, які аспекти даних є найбільш інформативними для конкретної задачі. Цей підхід особливо корисний, оскільки дозволяє моделі адаптуватися до особливостей даних без втручання користувача.

Методи вбудованого відбору ознак ефективно враховують взаємозв'язки та важливість різних ознак, забезпечуючи гнучкість в процесі визначення, які з них слід враховувати. Основна ідея полягає в тому, що сам алгоритм навчання враховує вагомість кожної ознаки під час побудови моделі, визначаючи її вплив на точність прогнозування.

Використання вбудованого відбору ознак допомагає уникнути надмірного вивчення, сприяє покращенню узагальнених властивостей моделі і може зробити її більш інтерпретованою. Однак слід враховувати, що вбудований відбір ознак може виявитися менш ефективним в тих випадках, коли взаємозв'язки між ознаками складні або коли існують нелінійні залежності в даних. Також важливо тестувати різні алгоритми вбудованого відбору ознак для конкретної задачі та оцінювати їхню ефективність[21].

Зменшення розмірності за допомогою методу головних компонент (PCA) відзначає важливість оптимізації обробки послідовних даних, особливо при їхній великій кількості та корельованості. В контексті цього виникає необхідність використання методів, які здатні зберегти корисну інформацію при зменшенні розмірності.

Одним із часто використовуваних методів є метод головних компонент (PCA). Цей метод використовує лінійне перетворення для переходу від початкових корельованих характеристик до нової системи ознак, які є некорельованими та називаються основними компонентами. При нормальному розподілі це перетворення приводить до системи незалежних ознак, спрощуючи подальший аналіз.

Основна ідея полягає в тому, що лише "суттєві" ознаки з великою різницею у значеннях залишаються для подальшого аналізу. PCA є популярним через його оптимальність у лінійному стисненні та відновленні множини векторів високої

розмірності. Також важливою перевагою є можливість обчислення параметрів моделі безпосередньо з даних[22].

Приклад використання PCA показано на рисунку 2.5.

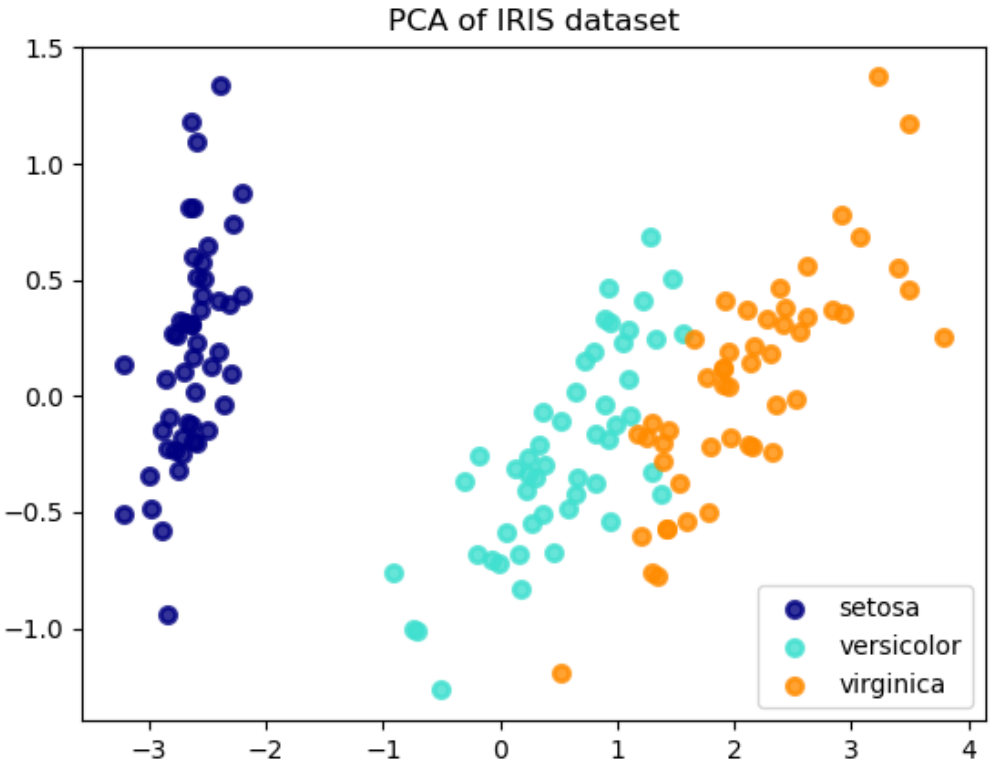


Рисунок 2.5 – Приклад використання методу PCA.

Незважаючи на привабливість PCA, він має свої недоліки. Зокрема, стандартні методи знаходження головних компонент можуть стикатися з проблемами при обробці великих обсягів даних високої розмірності. Також виникають труднощі у випадку неповних даних, коли неясно, як правильно враховувати пропущені значення[23].

Незважаючи на ці недоліки, PCA залишається популярним методом для зменшення розмірності часових рядів у вдосконалених алгоритмах, які враховують великий обсяг даних, а також працюють ефективно з пропущеними та зашумленими даними.

2.3. Міри відстані для послідовних даних

Однією з ключових складових процесу кластеризації є функція, що використовується для визначення ступеня схожості між двома наборами даних, що порівнюються. Ці дані можуть мати різні форми, такі як вихідні значення однакової або неоднакової довжини, вектори ознак, матриці переходів та інше.

Ця функція вимірює відстань або схожість між елементами даних та визначає, наскільки вони близькі одне до одного в просторі характеристик. Для векторів ознак це може включати в себе розрахунок евклідової відстані, косинусної схожості або інших метрик. Важливою особливістю є те, що ця функція повинна бути чутливою до властивостей конкретного типу даних та придатною для використання в контексті конкретного завдання кластеризації.

При використанні кластеризації, де мета полягає в групуванні схожих елементів, обрання відповідної функції вимірювання подібності є ключовим етапом для досягнення ефективних та змістовних результатів[24].

Евклідова відстань є метрикою, що визначає відстань між двома точками в просторі. Ця метрика широко використовується в задачах, де важлива абсолютна відстань між точками, таких як геометричні обчислення, машинне навчання та кластерний аналіз. Її використання дозволяє враховувати геометричні властивості даних і визначати їхню схожість чи віддаленість у просторі ознак.

Якщо xi та vj є P-вимірним вектором, то Евклідова відстань обчислюється як:

$$dE = \sqrt{\sum} (xik - vjk) \; 2 \; P \; k{=}1 \; . \qquad (2.1)$$

Середньоквадратична відстань, також відома як Манхеттенська відстань або відстань L1, представляє суму абсолютних різниць між відповідними координатами точок. Для двовимірного простору з точками (x_1, y_1) та (x_2, y_2), формула для обчислення середньоквадратичної відстані (d) виглядає наступним чином:

$$d = |x_2 - x_1| + |y_2 - y_1|. \qquad (2.2)$$

Ця метрика добре підходить для випадків, де важливо враховувати лише абсолютні різниці між координатами, і вона знаходить застосування в різних областях, включаючи аналіз даних та оптимізацію маршрутів.

Відстань Мінковського є узагальненням евклідової відстані, який визначається як:

$$dM = \sqrt{\sum} (xik - vjk) \; 2 \; P \; k{=}1 \; q \; . \qquad (2.3)$$

Параметр (p) дозволяє налаштовувати чутливість відстані до абсолютних чи квадратичних відхилень, роблячи її більш гнучкою та адаптованою до конкретних вимог аналізу даних чи моделювання.

2.4. Динамічна деформація часу

Динамічна деформація часу - це концепція, пов'язана з аналізом часових рядів, яка вказує на зміну темпу часового ряду в різні періоди. Ця ідея виникає з усвідомлення того, що швидкість або темп, з яким відбувається яка-небудь подія або явище, може змінюватися в часі.

В контексті часових рядів динамічна деформація часу може виявлятися в різноманітних способах. Наприклад, часовий ряд, який відображає температуру протягом року, може показувати динаміку, де швидкість зміни температури залежить від сезону. Це може бути виражено у тому, що влітку температура змінюється швидше, ніж взимку[25].

Також, динамічна деформація часу може бути пов'язана з тим, як швидкість зміни в часовому ряді може змінюватися через тривалі періоди. Наприклад, економічні показники можуть підпадати під впливом циклічних економічних процесів, що призводить до зміни темпу зростання або спаду.

Наприклад, нехай у нас є часовий ряд даних про температуру повітря в місті Києві за останні 10 років. Ми хочемо виявити наявність динамічної деформації часу в цьому часовому ряді.

Спочатку ми відфільтруємо часовий ряд, щоб усунути сезонні коливання. Для цього ми можемо використовувати метод скользящего середнього з періодом, що відповідає одному сезону.

Далі ми розділяємо часовий ряд на періоди по 12 місяців.

Для кожного періоду ми можемо використовувати метод середнього для визначення динамічної деформації часу.

У результаті ми отримаємо значення динамічної деформації часу для кожного періоду. Якщо значення динамічної деформації часу значні, це може вказувати на наявність динамічної деформації часу в часовому ряді[26].

Наприклад, якщо значення динамічної деформації часу збільшуються з часом, це може вказувати на те, що темп зміни температури повітря в Києві зростає.

На рисунку 2.6 показано приклад розрахування динамічної деформації часу.

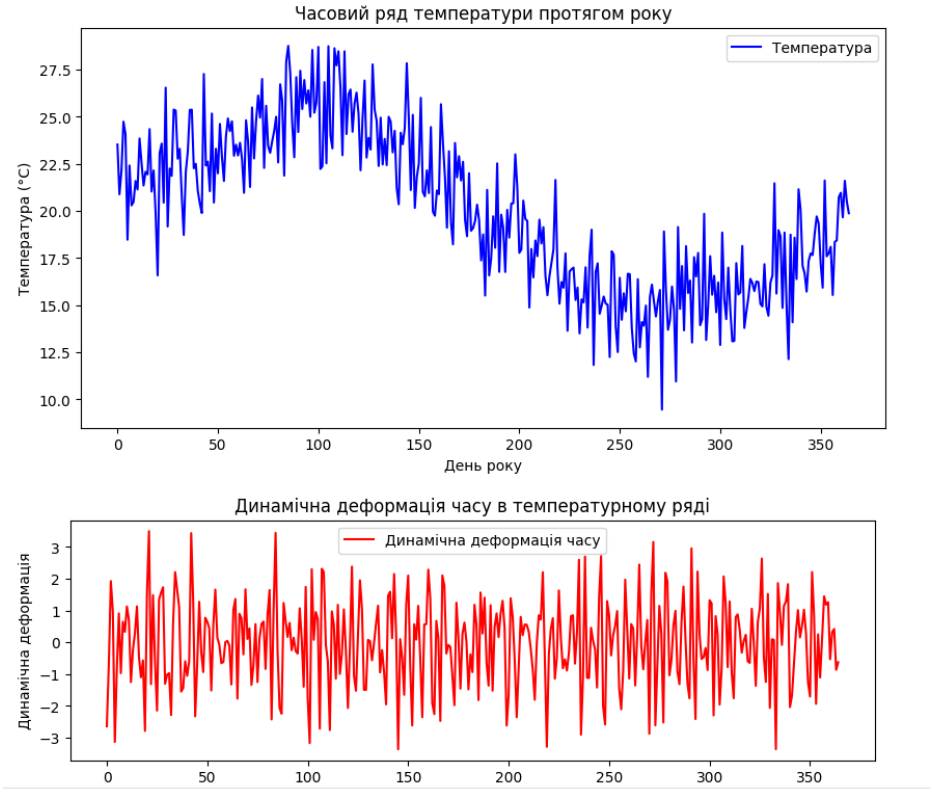


Рисунок 2.6 – Приклад розрахування динамічної деформації часу.

Одним із способів аналізу динамічної деформації часу є використання методів часового розгортання (time warping), коли змінюється швидкість "проходження" часу в різних періодах. Це може допомогти вирівнювати часові ряди і виявляти певні закономірності або циклічні зміни.

Використання методів часового розгортання в аналізі динамічної деформації часових рядів полягає в можливості адаптувати часові ряди до змін швидкості розвитку подій в різних частинах ряду. Це корисно для розкриття закономірностей, циклічних змін чи асинхронії у динаміці даних. Застосування цих методів може полегшити порівняння рядів, вирівнювання циклів та виділення ключових особливостей динаміки[27].

Вирівнювання часових рядів за допомогою методів часового розгортання є важливим етапом у розумінні динамічної деформації. Це дозволяє аналізувати сигнали, які можуть рухатися або змінюватися швидкістю в різних фрагментах часу.

Зміна швидкості часового ряду може виникати з різних причин, таких як зміни в темпі подій, циклічні зміни амплітуди або фреквенції, або нерегулярність у динаміці процесу. Часове розгортання дозволяє узгоджувати ці зміни та створює базу для подальшого аналізу та виявлення залежностей.

Застосування цих методів може бути важливим в багатьох областях, таких як обробка сигналів, медична діагностика, фінансова аналітика та багато інших, де важливо виявляти та розуміти динамічні зміни в часових рядах для кращого управління та прийняття рішень.

Враховуючи гнучкість методів часового розгортання, отримані в результаті вирівнювання часові ряди можна використовувати для подальшого аналізу. Це дозволяє виявляти циклічні тенденції, періодичні закономірності та інші динамічні особливості, які можуть залишатися невидимими при звичайному порівнянні рядів[28].

Застосування цих методів особливо корисне в сценаріях, де динаміка системи несталі або змінюється з часом. Вони дозволяють розкривати та

аналізувати відмінності у темпі розвитку подій, допомагаючи виявляти потенційно важливі періоди та події.

2.5. Висновки

У цьому розділі було проведено аналіз алгоритмів попередньої обробки часових рядів та алгоритмів онлайн кластеризації. Початковий етап аналізу включав перевірку статистичних гіпотез властивостей часового ряду в процесі попереднього аналізу вибірки.

Далі в розділі розглядалися методи зменшення розмірності, які допомагають зберегти важливу інформацію з часових рядів при зниженні їхньої складності. Аналізувалися також міри відстані для послідовних даних, які є ключовим елементом у визначенні схожості між рядами. Зокрема, досліджувалася динамічна деформація часу як один із методів врахування змін в темпі та часових зсувів у часових рядах.

3. ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

3.1. Обґрунтування вибору середовища програмної реалізації

Jupyter Notebook - це інтерактивне середовище для програмування, яке дозволяє об'єднувати код, текстові описи та графіки в одному документі. Воно широко використовується для аналізу даних, наукових досліджень та навчання. У Jupyter Notebook можна працювати з різними мовами програмування, але найбільш популярною є Python.

Кожен документ у Jupyter Notebook складається з комірок, які можуть бути виконані окремо. Комірки бувають двох типів: комірки з кодом і текстові комірки. Комірки з кодом використовуються для написання програмного коду, який можна виконати, а текстові комірки дозволяють додавати описи, відформатований текст та зображення.

Інтерактивність Jupyter Notebook полягає в тому, що ви можете виконувати окремі частини коду та спостерігати за результатами в реальному часі. Результати виводяться безпосередньо під коміркою коду, що дозволяє легко візуалізувати та розуміти результати обчислень[29].

На рисунку 3.1 показано офіційни веб-сайт Jupyter Notebook.

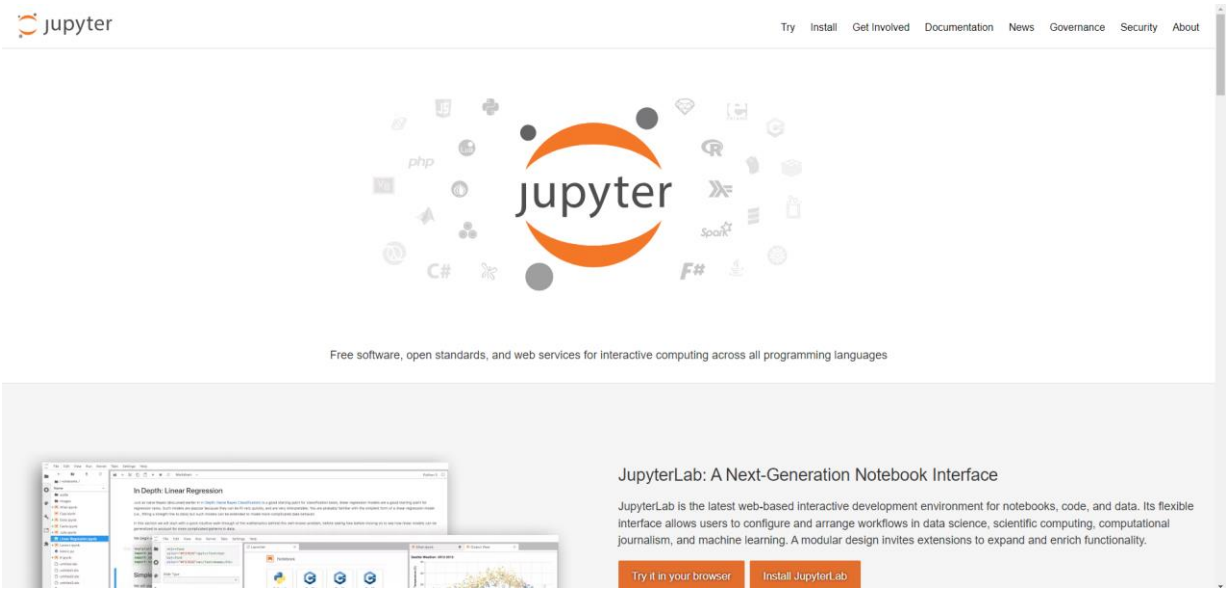


Рисунок 3.1 – Веб-сайт Jupyter Notebook.

Крім того, Jupyter Notebook може використовувати різноманітні бібліотеки для візуалізації даних, статистичного аналізу та машинного навчання, роблячи його потужним інструментом для дослідження та аналізу даних.

Jupyter Notebook надає можливість зберігати результати обчислень у самому документі, що робить його ідеальним для створення звітів, наукових робіт чи навчальних матеріалів. Ви можете легко експортувати ваші ноутбуки у різні формати, такі як HTML, PDF або презентації слайдів, що дозволяє поділитися своєю роботою з іншими.

Ще однією корисною особливістю є можливість використовувати команди шелу (командного рядка) прямо з комірки коду. Це відкриває широкі можливості для виконання системних команд, інтеграції з іншими інструментами та автоматизації задач.

Усе це робить Jupyter Notebook важливим інструментом для дослідження та співпраці в галузях, де важливо об'єднати код, дані та пояснення в єдиному документі. З його допомогою ви можете створювати, тестувати та демонструвати ваш код та аналіз даних в інтерактивному та легко зрозумілому форматі.

Jupyter Notebook також підтримує візуалізацію даних через різноманітні бібліотеки, такі як Matplotlib, Seaborn та Plotly. Це дозволяє створювати графіки, діаграми та інші візуальні елементи безпосередньо в середовищі ноутбука, щоб краще розуміти залежності в даних[30].

Завдяки можливості вставляти зображення, відео та аудіо файли, Jupyter Notebook дозволяє вам інтегрувати різноманітні медіаелементи безпосередньо в документ. Це робить його ефективним інструментом для створення інтерактивних навчальних матеріалів та презентацій.

Загальне користування Jupyter Notebook спрощує роботу з великими обсягами даних та обчисленнями, а його підтримка різних мов програмування розширює його застосування на різні галузі та проекти. Це інтерактивне середовище продовжує здобувати популярність як серед дослідників, вчених, так і вчителів, що робить його важливим інструментом в сфері обробки даних та програмування.

Jupyter Notebook також підтримує візуалізацію даних через різноманітні бібліотеки, такі як Matplotlib, Seaborn та Plotly. Це дозволяє створювати графіки, діаграми та інші візуальні елементи безпосередньо в середовищі ноутбука, щоб краще розуміти залежності в даних.

Завдяки можливості вставляти зображення, відео та аудіо файли, Jupyter Notebook дозволяє вам інтегрувати різноманітні медіаелементи безпосередньо в документ. Це робить його ефективним інструментом для створення інтерактивних навчальних матеріалів та презентацій.

Загальне користування Jupyter Notebook спрощує роботу з великими обсягами даних та обчисленнями, а його підтримка різних мов програмування розширює його застосування на різні галузі та проекти. Це інтерактивне середовище продовжує здобувати популярність як серед дослідників, вчених, так і вчителів, що робить його важливим інструментом в сфері обробки даних та програмування.

Завдяки вбудованому підтримці LaTeX, Jupyter Notebook дозволяє вам вставляти математичні формули та символи для точного відображення математичних концепцій. Це особливо корисно для наукових досліджень та освіти в галузі математики та статистики[31].

Ще однією перевагою Jupyter Notebook є можливість взаємодії з обчислювальними інструментами та сервісами в хмарі. За допомогою вбудованих розширень та додатків, ви можете працювати з обчислювальними ресурсами, такими як Google Colab або Microsoft Azure Notebooks, просто і легко.

Також слід відзначити, що Jupyter Notebook зберігає історію виконання коду, що дозволяє повертатися до попередніх результатів та аналізувати зміни в процесі розробки. Це робить ноутбук чудовим інструментом для дослідження та відлагодження коду.

Jupyter Notebook забезпечує інтеграцію з версійним контролем, таким як Git, що полегшує спільну роботу над проектами. Ви можете фіксувати зміни, створювати гілки та об'єднувати їх безпосередньо з Jupyter Notebook, що полегшує управління проектами та збереження консистентності коду.

Для великих обсягів даних Jupyter Notebook також надає можливості використання розподілених обчислень та паралельної обробки даних. Це особливо корисно при роботі з великими наборами даних чи виконанні обчислень, які можуть бути оптимізовані за допомогою паралельних обчислень.

Надалі, Jupyter Notebook забезпечує можливість використовувати інтерактивні віджети, які дозволяють вам створювати власні користувацькі інтерфейси для взаємодії з кодом та даними. Це робить можливим створення складних візуальних елементів для кращого зрозуміння та керування обчисленнями.

Python - це високорівнева мова програмування загального призначення, яка вперше була випущена у 1991 році Гвідо ван Россумом. Мова вирізняється читабельністю коду та простотою синтаксису, що робить її популярною серед початківців і досвідчених програмістів.

Python підтримує об'єктно-орієнтоване, імперативне та функціональне програмування. Вона має широкий спектр бібліотек і модулів, що полегшують розробку програм та сприяють їх використанню в різноманітних областях, таких як веб-розробка, аналіз даних, штучний інтелект і багато інших.

Однією з особливостей Python є його динамічна типізація, що дозволяє змінювати типи даних об'єктів під час виконання програми. Це полегшує розробку та зменшує кількість необхідного коду[32].

Python активно розвивається, і виходять нові версії з різними покращеннями та нововведеннями. Мова має велику спільноту користувачів, що сприяє обміну досвідом та підтримці новачків.

Ця мова також використовується для створення різноманітних застосунків, від невеликих скриптів до великих проектів. Python відомий своєю простотою та ефективністю, що робить його однією з найпопулярніших мов програмування в світі.

Python вражає своєю універсальністю і застосовується в широкому спектрі завдань. Завдяки простому синтаксису, вивчення мови стає доступним для новачків, але водночас вона пропонує потужність і гнучкість для досвідчених

розробників. Його велика активна спільнота забезпечує постійний розвиток мови та підтримку користувачів на всіх етапах їхнього досвіду. Python — це не просто мова програмування; це екосистема, що розвивається, в якій можна реалізувати практично будь-яку ідею від маленьких проектів до великих інноваційних розробок.

Python також славиться своєю здатністю швидко реагувати на зміни в технологічному середовищі. Багато сучасних технологій, таких як штучний інтелект, машинне навчання та аналіз даних, використовують Python як основну мову програмування. Більше того, існують різноманітні інструменти та фреймворки, які полегшують розробку та роботу з цими технологіями[30].

Завдяки своїй переносимості між різними операційними системами, Python є вибором для розробки крос-платформених додатків. Ця особливість робить його ідеальним вибором для розробників, які хочуть створювати програми, що працюють на різних платформах без значних змін у коді.

Python також відомий своєю активною спільнотою, яка взаємодіє через форуми, конференції та спільні проекти. Це сприяє обміну досвідом та ідеями, а також сприяє розвитку і вдосконаленню мови.

Однією з ключових особливостей Python є його "зен" — набір принципів, визначених в Довіднику по стилі програмування Python (PEP 8). Ці принципи спрямовані на створення читабельного та зрозумілого коду, що полегшує співпрацю між розробниками та сприяє обслуговуваності проектів[33].

Незалежно від того, чи ви новачок чи досвідчений програміст, Python відкриває широкі можливості для реалізації та вдосконалення програмних рішень у різних галузях. Його природна елегантність та гнучкість роблять його важливим інструментом для тих, хто бажає швидко та ефективно втілювати свої ідеї в життя.

3.2. Аналіз обраних бібліотек

Оскільки програмне рішення задачі потребує окремих інструментів під окремі задачі, першим кроком було імпортовано список необхідних бібліотек (рис. 3.2).

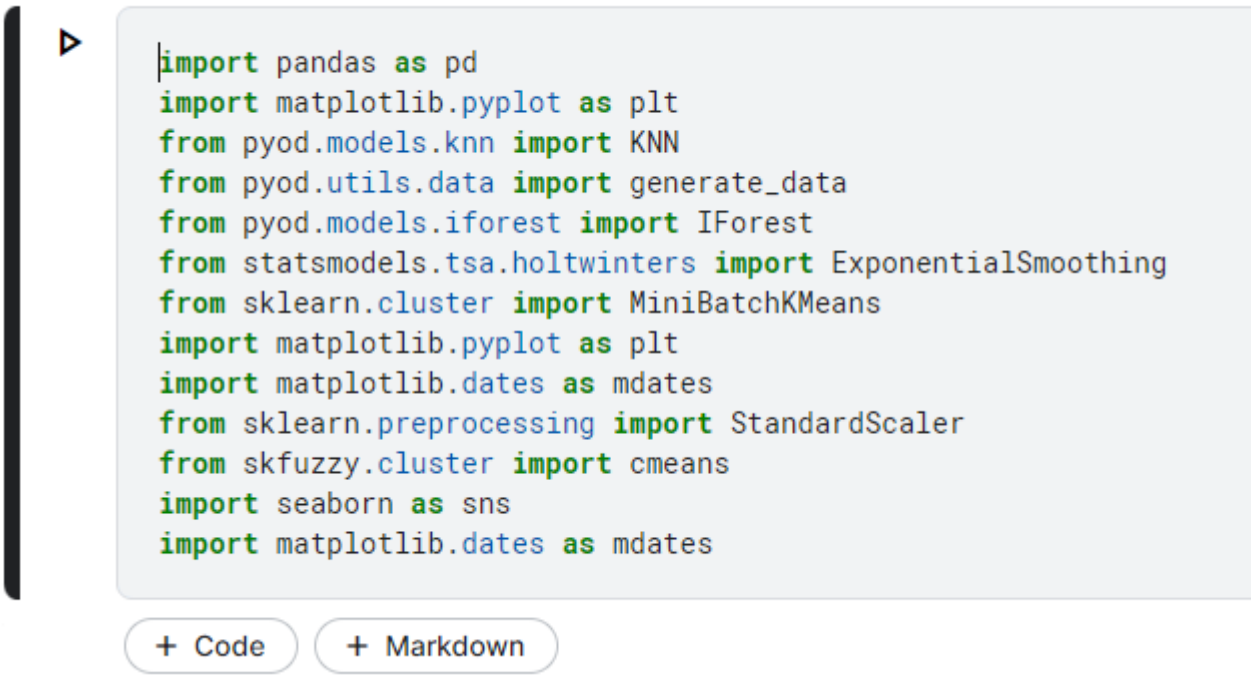


Рисунок 3.2 – Імпорт бібліотек у середовище розробки

– Бібліотека pandas використовується для роботи з даними у вигляді таблиць та рядків.

– Matplotlib – це бібліотека для візуалізації даних, де plt вказує на модуль для створення графіків.

– PyOD (Python Outlier Detection) - це бібліотека для виявлення викидів у даних.

– KNN (k-Nearest Neighbors) та IForest (Isolation Forest) - це моделі для виявлення аномалій.

– Statsmodels - бібліотека для статистичного аналізу даних, включаючи моделі прогнозування часових рядів, такі як ExponentialSmoothing.

– Scikit-learn - бібліотека для машинного навчання, MiniBatchKMeans - це алгоритм кластеризації методом міні-пакетів k-середніх.

– Seaborn - бібліотека для візуалізації даних на основі Matplotlib.

– Skfuzzy - це бібліотека для роботи з нечіткими множинами та нечіткими кластерними алгоритмами.

– Бібліотека mdates в Matplotlib використовується для роботи з форматами дат.

– Бібліотека StandardScaler з scikit-learn використовується для стандартизації даних перед їх використанням у моделях машинного навчання.

Пакет NumPy є невід'ємним інструментом для мови програмування Python. Він використовується для обробки даних, реалізації алгоритмів машинного навчання, наукових обчислень, а також спрощує роботу з векторами та матрицями. Декілька провідних бібліотек Python, таких як scikit-learn, SciPy, pandas і tensorflow, базують свою інфраструктуру на NumPy. Здатність аналізувати числові дані та вміння працювати з NumPy надає суттєву перевагу при вирішенні складних завдань бібліотек[33].

NumPy виник як нащадок Numeric, розробленого Джимом Хугуніном. Також був створений пакет Numarray з додатковою функціональністю. У 2005 році Тревіс Оліфант випустив пакет NumPy, об'єднавши особливості Numarray та Numeric. Це відкритий проект, у якому багато людей взяли участь у розвитку.

NumPy, або Numerical Python, є бібліотекою мови Python, яка надає потужні можливості:

– N-мірні масиви;

– високорівневі функції;

– інструменти для інтеграції коду C/C++ та Fortran;

– використання лінійної алгебри, перетворень Фур'є та можливостей генерації випадкових чисел.

Ця бібліотека також пропонує ефективний багатовимірний контейнер для зберігання різноманітних даних різних типів.

Для реалізації системи кластеризації було обрано бібліотеку Pandas. Pandas

- це бібліотека Python, спеціально призначена для обробки та аналізу структурованих даних, і назва її походить від "panel data" (панельні дані). Панельні дані визначаються як інформація, отримана в результаті досліджень та структурована у вигляді таблиць, і для роботи з такими масивами даних була створена бібліотека Pandas.

Pandas – це відкрита бібліотека, і її вихідний код у відкритому доступі розміщено на GitHub. Користувачі можуть активно взаємодіяти з бібліотекою, додаючи свій код, розширюючи методи роботи та оновлюючи її розділи. Для роботи з Pandas необхідний компілятор C/C++ та середовище розробки Python.

За допомогою Pandas фахівці можуть легко групувати та візуалізувати дані, створювати зведені таблиці та виконувати вибірку за певними ознаками. Для успішного аналізу даних з Pandas важливо розуміти структури даних всередині бібліотеки. Зокрема, важливі поняття такі, як Pandas Series (серія) - одномірний масив, схожий на пронумерований список, та Pandas DataFrame - двовимірний масив, подібний до таблиці або аркуша Excel, що складається зі стовпців і рядків.

Matplotlib є бібліотекою, створеною для візуалізації двовимірної (2D) та тривимірної (3D) графіки, і вона має відкритий вихідний код, призначений для будівництва графіків. Її автор, Джон Д. Хантер, почав розробку Matplotlib під час своїх досліджень у галузі нейробіології, де вона використовувалася для візуалізації активності в корі головного мозку пацієнтів з епілепсією. Бібліотека вийшла у світ у 2003 році з метою поліпшення візуалізації та вивчення загальних закономірностей[34].

Matplotlib став однією з найбільш широко використовуваних бібліотек для побудови графіків разом із мовою програмування Python. Ця бібліотека не обмежена платформою і працює на різних операційних системах, таких як Windows, Mac OS та Linux.

У даній роботі використовувався модуль Pyplot у бібліотеці Matplotlib. Pyplot - це інтерфейс прикладного програмування, що складається з функцій та методів, які сприяють обробці даних для їх візуалізації. Інтерфейс схожий на

Matlab, що полегшує роботу для тих, хто вже має досвід з MATLAB та навпаки. До його функцій входять сюжети, зображення, гістограми, таблиці, кругові діаграми та потокові графіки.

Основні переваги використання Matplotlib включають його простоту використання, можливість створення високоякісних графіків у різних форматах, таких як png і pdf, і можливість управління різними параметрами фігури, такими як DPI, колір фігури та її розмір[30].

Загальна ідея полягає в тому, що ці інструменти спільно дозволяють виявляти структури, аномалії та взаємозв'язки в часових рядах, що може бути корисним для подальшого кластерного аналізу та виведення узорів в даних.

Statsmodels - бібліотека для статистичного моделювання в Python. Вона дозволяє аналізувати та вивчати різноманітні статистичні моделі, включаючи лінійні, часові ряди та нелінійні. Statsmodels дозволяє проводити тестування гіпотез, отримувати статистичні висновки, будувати моделі регресії та аналізувати залежні вибірки. Бібліотека надає ефективні інструменти для вивчення та розуміння взаємозв'язків у ваших даних з точки зору статистики.

Statsmodels розширює можливості аналізу даних, надаючи інструменти для обробки часових рядів, включаючи побудову ARIMA та SARIMA моделей. Вона також дозволяє використовувати методи часового розгортання для дослідження динаміки деформацій у часі[31].

Окрім того, бібліотека має інтерфейс для використання в аналізі залежностей між різними змінними та для розв'язання завдань, пов'язаних із залежними вибірками. Вона надає інструменти для побудови та оцінки лінійних та нелінійних моделей, що дозволяє досліджувати складні взаємозв'язки в даних.

Завдяки можливостям Statsmodels, користувачі можуть проводити ретельний статистичний аналіз та використовувати результати для прийняття обґрунтованих рішень в різних областях, включаючи економіку, фінанси, медицину та соціальні науки.

Бібліотека Statsmodels також надає інструменти для візуалізації результатів аналізу даних. Вона спрощує створення графіків та візуалізацію

статистичних параметрів, що допомагає краще розуміти залежності в даних.

Додатково, важливо відзначити, що Statsmodels є активно розвиваючимся проектом, і нові версії можуть включати додаткові функції та покращення. Ця бібліотека є цінним інструментом для дослідників, аналітиків та науковців, які працюють з даними та хочуть проводити глибокий статистичний аналіз та моделювання.

3.3. Підготовка та аналіз датасету

Згідно із зазначеними методами аналізу часових рядів, перший етап включає в себе важливий процес – предобробку. Основна мета цього етапу полягає у створенні векторів-ознак, що відображають основні характеристики ряду за допомогою непрямих вимірювань. Серед цих вимірювань особливою увагою наділяються такі параметри, як середнє значення, дисперсія, експоненційне згладжування. Вони дозволяють отримати важливі відомості про тенденції, варіацію та взаємозв'язок подій у часовому ряді, що є ключовим етапом для подальшого узагальненого аналізу та моделювання.

Першим кроком був збір достовірних валютних даних з відкритих та достовірних джерел. В цьому випадку історичні дані валютної пари було отримано з веб-сайту «investing.com».

Investing.com — це веб-сайт, який надає безкоштовні котирування фондового ринку, фінансові новини, реальний час та багато іншого для різних ринків та активів. Тут ви можна знайти економічний календар, календар доходів, аналіз, думки, криптовалюти, індекси, товари та багато іншого.

Завантажені дані валютної пари NZD-USD було конвертовано в формат CSV для подальшої обробки. На рисунку 3.3 показано код імпорту датасету у програмне середовище.

```python
import pandas as pd

# Шляхи до датасетів
nzd_usd_path = "/kaggle/input/usd-chf/NZD_USD.csv"

# Завантаження датасетів за допомогою pandas
nzd_usd_data = pd.read_csv(nzd_usd_path)

nzd_usd_data.insert(0, 'id', range(1, 1 + len(nzd_usd_data)))

# Для кожного датасету
def preprocess_dataset(data):
    # Видалення стовпців 'Vol.' і 'Change %'
    data = data.drop(['Vol.', 'Change %', 'Vol', 'Diff', 'Change'], axis=1, errors='ignore'

    # Перейменування стовпців 'Max' і 'Min' у 'High' і 'Low'
    data = data.rename(columns={'Max': 'High', 'Min': 'Low'})

    return data

# Застосування до кожного датасету
nzd_usd_data = preprocess_dataset(nzd_usd_data)

# Виведення перших кількох рядків кожного датасету для перевірки
print("NZD_USD:")
nzd_usd_data
```

Рисунок 3.3 – Імпорт датасету в програмне середовище.

Оскільки отримані дані з джерела мали в собі деякі нечислові дані які не мають ніякої цінності в цьому досліджені, їх було видалено. Окрім того, для зручності подальшого маніпулювання даними деякі стовпці було перейменовано.

Завантажений та імпортований дата сет показано на рисунку 3.4.

	id	Date	Price	Open	High	Low
0	1	12/14/2023	0.6236	0.6191	0.6250	0.6190
1	2	12/13/2023	0.6173	0.6131	0.6215	0.6083
2	3	12/12/2023	0.6132	0.6123	0.6170	0.6105
3	4	12/11/2023	0.6121	0.6118	0.6138	0.6104
4	5	12/08/2023	0.6121	0.6169	0.6176	0.6104
...	...	...	...	...	...	...
504	505	01/07/2022	0.6780	0.6748	0.6787	0.6734
505	506	01/06/2022	0.6745	0.6794	0.6799	0.6733
506	507	01/05/2022	0.6792	0.6811	0.6839	0.6790
507	508	01/04/2022	0.6812	0.6794	0.6826	0.6764
508	509	01/03/2022	0.6784	0.6826	0.6858	0.6773

Рисунок 3.4 – Приклад даних з завантаженого датасету.

З отриманих даних було сформовано часовий ряд ціни валютної пари, який показано на рисунку 3.5.

Рисунок 3.5 – Валютний часовий ряд.

Наступним кроком було побудовано графіки тренду, сезонності та залишкової складової.

Графіки тренду залишкової складової в аналізі часових рядів є важливими інструментами для розуміння та прогнозування змін у даному наборі даних.

Тренд вказує на загальний напрямок руху даних протягом тривалого періоду, а залишкова складова представляє собою випадкові або непередбачувані відхилення від тренду та сезонності.

Ці компоненти дозволяють вченим і аналітикам отримувати глибше розуміння структури даних і виявляти ключові патерни, які можуть бути використані для прийняття управлінських рішень або прогнозування майбутніх значень. Аналіз тренду допомагає визначити загальну тенденцію зростання або зниження, а залишкова складова може допомогти ідентифікувати виняткові події або аномалії, які не вписуються в очікувані зміни.

Ці графіки важливі для ефективного моделювання та передбачення часових рядів, дозволяючи аналітикам розкривати суттєві патерни та фактори, що впливають на дані. Вони стають основою для розвитку прогнозних моделей та стратегій управління, спрямованих на оптимізацію різних бізнес-процесів.

На рисунку 3.6 показано графік часового ряду та графік тренду.

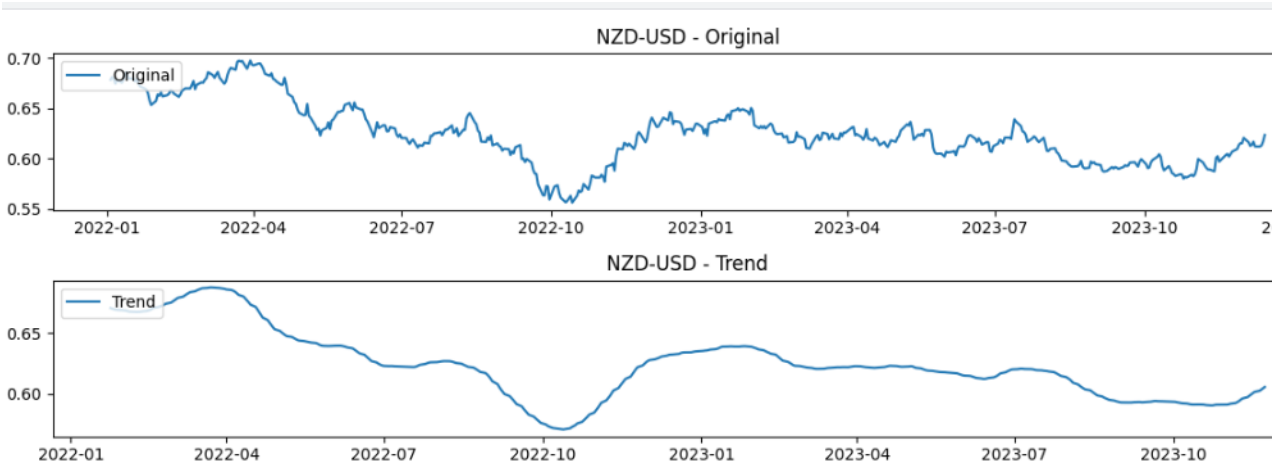


Рисунок 3.6 – Оригінальний графік часового ряду та графік тренду.

Цей графік показує курс новозеландського долара до долара США (NZD/USD) протягом останніх двох років. З початку 2022 року курс NZD/USD коливався в діапазоні від 0,55 до 0,70. У першому кварталі 2022 року курс зріс до 0,65, але потім почав падати. У другому кварталі курс продовжував падати і

досяг мінімуму в 0,55 у серпні 2022 року. У третьому кварталі курс почав відновлюватися і досяг 0,65 у листопаді 2022 року. У четвертому кварталі курс знову почав падати і досяг 0,60 у грудні 2022 року.

На графіку також показано тренд NZD/USD. Тренд - це загальний напрямок руху ціни. У цьому випадку тренд NZD/USD є знижувальним. Це означає, що в довгостроковій перспективі курс NZD/USD, ймовірно, буде продовжувати падати.

Існує кілька факторів, які можуть впливати на курс NZD/USD. До них відносяться:

- Економічна ситуація в Новій Зеландії та США
- Ставки відсотків у Новій Зеландії та США
- Інфляція в Новій Зеландії та США
- Попит і пропозиція на новозеландський долар і долар США

Далі, було побудовано графік залишкової складової які показано на рисунку 3.7.

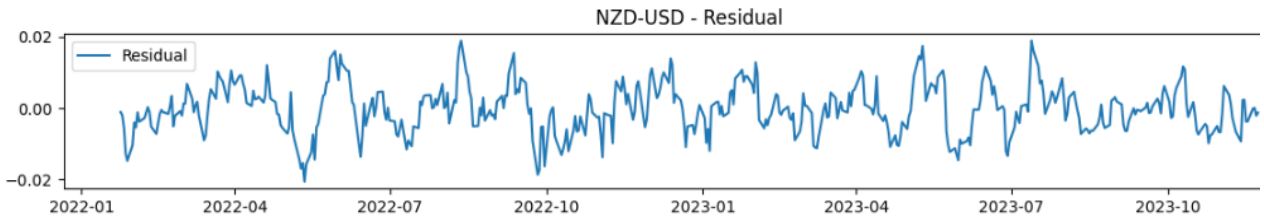


Рисунок 3.7 – Графік залишкової складової.

Графіки на рисунку 3.7 показують зміни у співвідношенні між новозеландським доларом (NZD) та доларом США (USD) протягом останніх 24 місяців.

Графік залишків показує відхилення фактичних значень співвідношення NZD/USD від середніх сезонних значень. Як видно з графіка, залишки були в основному позитивними в першій половині 2022 року, що означає, що співвідношення NZD/USD було вище, ніж очікувалося. Однак у другій половині

2022 року залишки стали в основному негативними, що означає, що співвідношення NZD/USD було нижче, ніж очікувалося.

Наступним кроком було розраховано динаміку дисперсії цін для цього часового ряду з використанням алгоритму Велфорда. Числовий ряд Велфорда утворюється так: починаючи з числа 1, на кожному кроці вибирається нове число, яке ще не з'являлося в послідовності, але є найменшим можливим для того, щоб робити середню значення послідовності цілими числами. Якщо числа вибрані з великої послідовності, то середні значення можуть бути цілими числами або наближеними до цілих чисел.

Код реалізації цього алгоритму показано на рисунку 3.8.

```
def welford_variance(data):
    n = 0  # кількість даних
    mean = 0.0  # середнє
    m2 = 0.0  # момент другого порядку (для обчислення дисперсії)

    for x in data:
        n += 1
        delta = x - mean
        mean += delta / n
        delta2 = x - mean
        m2 += delta * delta2

    if n < 2:
        return float('nan')

    variance = m2 / n
    return variance

def plot_price_variance(data, date_column='Date', price_column='Price', title='Динаміка дисперсії цін з часом'):
    # Перетворення стовпця 'Date' в об'єкт datetime
    data[date_column] = pd.to_datetime(data[date_column])

    # Застосовуємо алгоритм Велфорда для обчислення дисперсії
    data['Price_Variance'] = data[price_column].expanding().apply(welford_variance)

    # Побудова графіку
    plt.figure(figsize=(10, 6))
    plt.plot(data[date_column], data['Price_Variance'], label='Дисперсія цін', color='blue')
    plt.title(title)
    plt.xlabel('Дата')
    plt.ylabel('Дисперсія')
    plt.legend()
    plt.grid(True)

    # Встановлення інтервалу дат
    date_interval = pd.date_range(start=data[date_column].min(), end=data[date_column].max(), freq='M')
    plt.xticks(date_interval, rotation=45)
    plt.xlim(data[date_column].min(), data[date_column].max())
```

Рисунок 3.8 – Реалізація алгоритму Велфорда.

На рисунку 3.9 показано розраховану динаміку дисперсії цін для валютного часового ряду з використанням вищеописаного алгоритму.

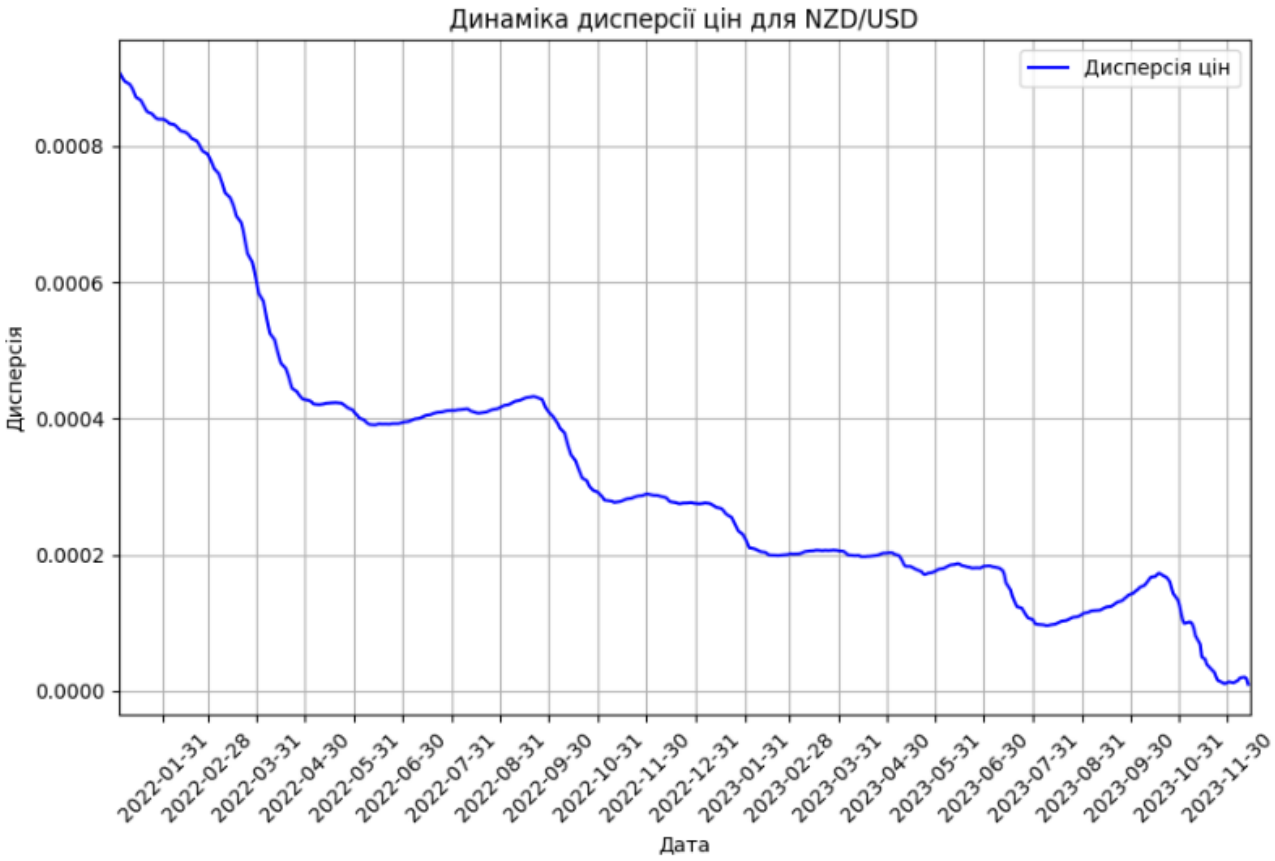


Рисунок 3.9 – Динаміка дисперсії часового ряду.

Графік на рисунку 3.9 показує динаміку дисперсії цін для NZD/USD. Дисперсія цін - це міра того, наскільки сильно ціни коливаються навколо середньої ціни. На графіку показано, що дисперсія цін для NZD/USD була відносно високою на початку 2022 року, але потім почала знижуватися.

Лінія на графіку представляє дисперсію цін. Лінія починається на високому рівні в 0,0008 у січні 2022 року і потім поступово знижується до 0,0000 у грудні 2023 року. Це означає, що ціни на NZD/USD стали менш волатильними протягом цього періоду.

Існує кілька можливих пояснень для цього зниження дисперсії. Одне пояснення полягає в тому, що економічні умови стали більш стабільними. Інше

пояснення полягає в тому, що інвестори стали більш впевненими в перспективах курсу NZD/USD.

Зниження дисперсії цін може бути позитивним сигналом для інвесторів, які торгують валютними парами. Це означає, що ціни стали менш волатильними і, отже, менш ризикованими.

Наступним кроком було розраховано середнє та експоненційне згладжування. Результати було відображено на графіку який показано на рисунку 3.10.

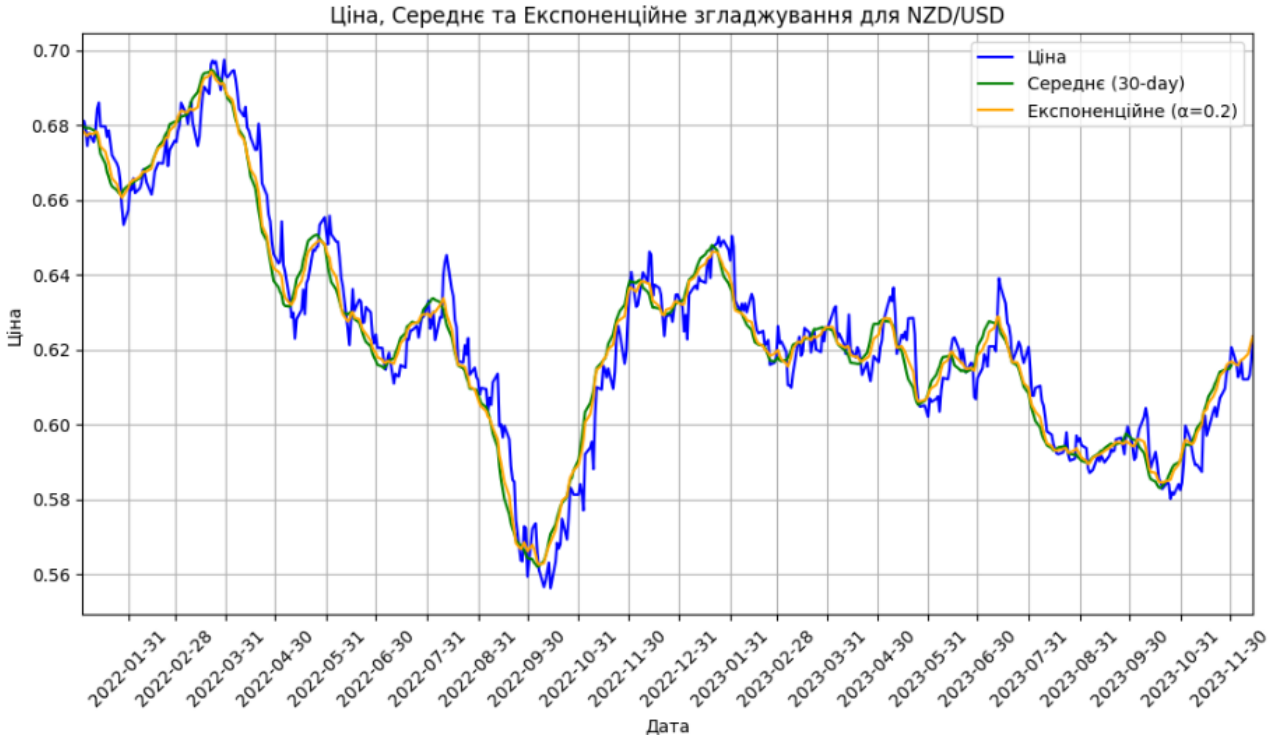


Рисунок 3.10 – Графік середнього та експоненційного згладжування для часового ряду.

Графік що було відображено на рисунку 3.10 показує ціну новозеландського долара (NZD) щодо долара США (USD) за період з 1 січня 2022 року по 16 грудня 2023 року. Графік відображає три типи ліній:

- Ціна: Ця лінія показує фактичну ціну NZD/USD за кожний день.
- Середнє (30-денне): Ця лінія показує середню ціну NZD/USD за останні 30 днів.

– Експоненційне (a=0.2): Ця лінія показує експоненційне згладжування ціни NZD/USD.

Середнє (30-денне) ціни NZD/USD також зростало з початку 2022 року. 31 січня 2022 року середнє значення становило 0,60 USD, а 16 грудня 2023 року – 0,63 USD. Це зростання становило близько 5%.

Експоненційне (a=0.2) ціни NZD/USD також зростало з початку 2022 року. 31 січня 2022 року експоненційне значення становило 0,60 USD, а 16 грудня 2023 року – 0,64 USD. Це зростання становило близько 7%.

Важливо зазначити, що ціна валюти може змінюватися під впливом різних факторів, таких як економічні новини, політичні події та інфляція. Тому важливо аналізувати такі фактори, перш ніж приймати будь-які інвестиційні рішення.

3.4. Програмна реалізація

Першим кроком в програмній реалізації було розраховано коефіцієнт ефективності. Було побудовано алгоритм аналізу фінансових даних, спрямований на виявлення можливих точок входу та виходу на ринок для торгівлі валютною парою NZD/USD. Програма використовує розраховані раніше показники та осцилятори для надання інформації про можливі зміни в ринковому тренді та його майбутній напрямок.

Алгоритм починається із завантаження фінансових даних та обчислення змін цін, що дозволяє виявити відмінності між сусідніми торговельними днями. Далі використовуються ковзні середні (MA) для створення індикаторів короткострокового та довгострокового тренду. Сигнали покупки або продажу генеруються на основі перетину цих середніх.

Осцилятор "стохастика" та осцилятор "MACD" додають додаткові шари аналізу. "Стохастик" вказує на ступінь перекупленості чи перепроданості ринку, визначаючи його відносну позицію відносно максимального та мінімального значень цін. "MACD" використовується для виявлення змін у ринковому імпульсі та генерації сигналів.

Поєднання цих показників та їхніх вагових коефіцієнтів дозволяє створити загальний вектор ефективності (K), який представляє комплексний огляд ринкових умов. Зсув значень деяких показників на один день вперед враховує природу торгівельних сигналів, які можуть з'являтися після розрахунку.

Завершальний етап алгоритму визначає найсильніший сигнал серед розглянутих, надаючи ключову інформацію для прийняття рішення щодо входу чи виходу з ринку. Вивід результатів у формі таблиці надає зручний інструмент для аналізу та прийняття торгівельних рішень. Таблицю результатів усіх описаних розрахунків показано на рисунку 3.11.

	Date	Price	MA_Crossover	Stochastic_Oscillator	K_mod
504	2022-01-07	0.6780	1	61.722488	19.009947
505	2022-01-06	0.6745	1	33.908046	18.816946
506	2022-01-05	0.6792	1	58.024691	10.472511
507	2022-01-04	0.6812	1	65.714286	17.707492
508	2022-01-03	0.6784	1	33.913043	20.014379

	S%K	S%D	kst	macd1	macd2	kmacd	K	\
504	67.129564	78.726618	-4.0	0.002889	0.001891	0.0	15.009947	
505	52.665375	66.015894	-1.0	0.002498	0.002013	0.0	17.816946	
506	51.218408	57.004449	-1.0	0.002539	0.002118	0.0	9.472511	
507	52.549008	52.144264	-1.0	0.002701	0.002234	0.0	16.707492	
508	52.550674	52.106030	0.0	0.002574	0.002302	0.0	20.014379	

Рисунок 3.11 – Результат розрахунку вектору ефективності.

Також, значення отримані під час проведення описаних розрахунків було виведено графічно у якості лінійного графіку. Графік ціни та ковзних середніх показано на рисунку 3.12.

Рисунок 3.12 – Графік ціни та ковзних середніх.

Показаний на рисунку 3.12 графік відображає ціну активу протягом певного періоду часу, а також дві ковзні середні, які розраховуються на основі даних про ціну.

У цьому випадку ковзні середні розраховуються на основі 10 та 50 точок даних. Це означає, що коротка ковзна середня (короткий MA) оновлюється кожні 10 днів, а довга ковзна середня (довгий MA) – кожні 50 днів.

Як видно на графіку, ціна активу була відносно стабільною протягом перших кількох місяців 2022 року. Однак у травні ціна почала зростати, і цей тренд продовжувався до жовтня. У листопаді ціна почала знижуватися, і цей тренд продовжувався до січня 2023 року.

Короткий MA також показує зростання ціни протягом перших кількох місяців 2022 року. Однак у травні короткий MA почав знижуватися, і цей тренд продовжувався до жовтня. У листопаді короткий MA почав зростати, і цей тренд продовжувався до січня 2023 року.

Довгий MA показує більш стабільну тенденцію, ніж короткий MA. Довгий MA був відносно стабільним протягом перших кількох місяців 2022 року, але він почав зростати у травні. Цей тренд продовжувався до жовтня, коли довгий MA

почав знижуватися. У листопаді довгий МА почав зростати, і цей тренд продовжувався до січня 2023 року.

На рисунку 3.13 показано графік осцилятору «MACD».

Рисунок 3.13 – Графік осцилятору «MACD».

MACD – це технічній індикатор, який використовується для виявлення трендів та можливих розворотів. Він складається з двох ліній: MACD, сигнальної лінії.

MACD – це основна лінія індикатора. Вона розраховується як різниця між довгою ковзною середньою (26 періодів) та короткою ковзною середньою (12 періодів). Сигнальна лінія – це проста ковзна середня MACD. Вона розраховується як проста ковзна середня MACD за 9 періодів.

Графік показує, що MACD була позитивною протягом перших кількох місяців 2022 року. Це означає, що ціна активу була в зростаючому тренді. Однак у травні MACD перетнула сигнальну лінію вниз, що є ознакою можливого розвороту тренду. Цей розворот підтвердився у жовтні, коли MACD перетнула нульову лінію вниз.

На рисунку 3.14 показано вектор ефективності «K_mod».

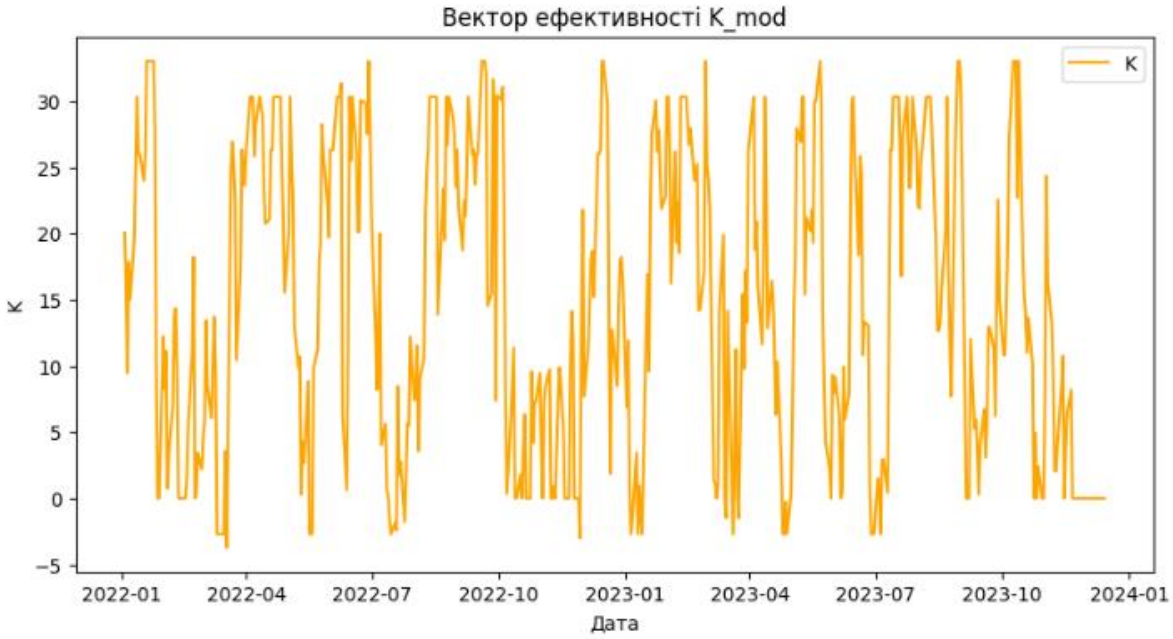


Рисунок 3.14 – Графік вектору ефективності «K_mod».

Графік показує, що вектор ефективності K_mod був позитивним протягом перших кількох місяців 2022 року. Це означає, що ціна активу була в зростаючому тренді. Однак у травні вектор ефективності K_mod перетнув нульову лінію вниз, що є ознакою можливого розвороту тренду. Цей розворот підтвердився у жовтні, коли вектор ефективності K_mod перетнув нульову лінію вниз.

Використовуючи результати попередніх досліджень, можна перейти до кластеризації даних.

На рисунку 3.15 показано код алгоритму кластеризації даних.

```
def create_observation_matrix(data):
    n = len(data)
    X = np.zeros((n, 3))

    max_K_mod = np.max(data['K_mod'])  # Знаходимо максимальне значення K_mod

    for i in range(n - 1):
        X[i, 0] = data['K_mod'].iloc[i]
        X[i, 1] = data['K_mod'].iloc[i]
        X[i, 2] = data['K_mod'].iloc[i + 1] - max_K_mod

    X[-1, 0] = data['K_mod'].iloc[-1]
    X[-1, 1] = data['K_mod'].iloc[-1]
    X[-1, 2] = data['K_mod'].iloc[-1]

    return X

# Виклик функції для створення матриці спостережень X
observation_matrix = create_observation_matrix(data)

observation_matrix[np.isnan(observation_matrix)] = 0

def perform_clustering(observation_matrix, c=2):
    cntr, u, _, _, _, _, _ = fuzz.cluster.cmeans(
        observation_matrix.T, c, 2, error=0.005, maxiter=1000)
    return cntr, u

# Кластеризація для покупок
c = 2
cntr_buy, u_buy = perform_clustering(observation_matrix, c)
F_buy = u_buy.T
F_sell = -F_buy  # Перетворення матриці -F
```

Рисунок 3.15 – Код алгоритму кластеризації даних.

У показаному на рисунку 3.15 коді реалізовано алгоритм для кластеризації даних за допомогою нечіткої кластеризації. Основна мета - визначити ступінь приналежності кожного спостереження до різних кластерів. Алгоритм використовує бібліотеку scikit-fuzzy (skfuzzy), яка забезпечує зручні інструменти для роботи з нечіткістю та нечіткою кластеризацією.

Спершу, створюється матриця спостережень (observation_matrix), яка включає у себе значення стовпця 'K_mod' з датасету. Ця матриця готується для подальшого використання у кластеризаційному алгоритмі.

Далі викликається функція perform_clustering, яка використовує нечітку кластеризацію (fuzzy c-means) для розподілу спостережень між кластерами. В результаті отримуємо центри кластерів (cntr_buy) і матрицю приналежності (u_buy) для кластера "покупок".

Зазначений код також передбачає можливість використання того ж самого алгоритму для іншого кластера, який вказаний як "продажі" (F_sell). Це робиться шляхом зміни знаків матриці приналежності для кластера "покупок" на протилежні.

На рисунку 3.16 показано графік результатів кластеризації.

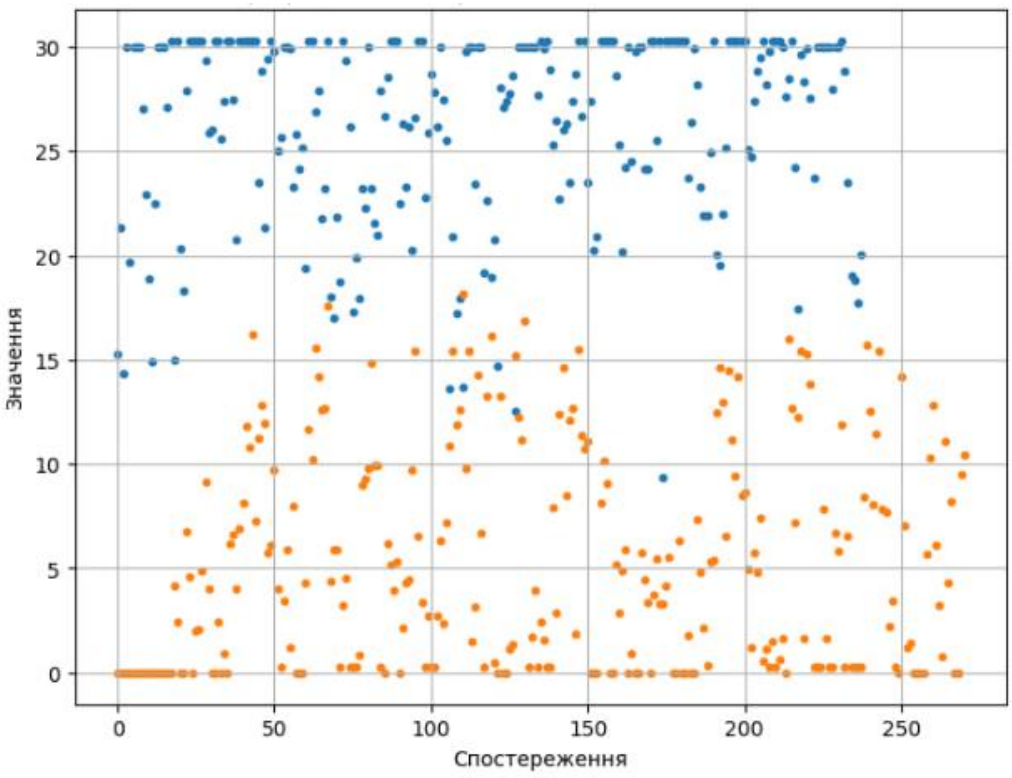


Рисунок 3.16 – Графік результатів кластеризації даних матриці спостереження.

Графік показує залежність між кількістю кластерів і кількістю спостережень у кожному кластері. Кількість спостережень у кожному кластері представлена точками на графіку.

Графік складається з двох кластерів, позначених синім і зеленим кольором. Кластер 1 (помаранчевий) містить більше спостережень, ніж кластер 2 (синій).

З використанням результатів кластеризації було сформовано новий вектор ефективності з нормуванням значень у проміжку [-1;1] який в подальшому буде використовуватись для системи підтримки прийняття рішень.

Код формування вектору ефективності та нормуванням значень показано на рисунку 3.17.

```python
# Функція для створення вектора ефективності K_mod
def create_efficiency_vector(F_buy, F_sell):
    n = min(F_buy.shape[0], F_sell.shape[0])
    K_mod = np.zeros(n)

    for i in range(n):
        K_mod[i] = np.dot(F_buy[i], np.arange(c_buy)) - np.dot(F_sell[i], np.arange(c_sell))

    # Нормалізація вектора K_mod до проміжку [-1, 1]
    K_mod = 2 * (K_mod - np.min(K_mod)) / (np.max(K_mod) - np.min(K_mod)) - 1

    return K_mod

# Виклик функції для створення вектора ефективності K_mod
K_mod = create_efficiency_vector(F_buy, F_sell)

# Додавання вектора K_mod до датафрейму
data['K_mod'] = K_mod

# Вивід датафрейму з оновленим стовпцем K_mod
print(data[['Date', 'K_mod']])
```

Рисунок 3.17 – Код формування вектору ефективності та нормування значень.

У цьому коді визначається функція create_efficiency_vector, яка призначена для створення вектора ефективності K_mod на основі купівельних та продажних кластерів.

Функція приймає матриці F_buy і F_sell, що, містять дані про купівельні та продажні фактори відповідно. Для кожного рядка обчислюється значення K_mod за допомогою вагового сумування, де ваги визначаються кількістю купівельних та продажних кластерів.

Отриманий вектор K_mod нормалізується до інтервалу [-1, 1], щоб забезпечити стандартизацію результатів.

Нарешті, створений вектор ефективності K_mod додається до датафрейму data в новий стовпець з назвою 'K_mod'. Весь датафрейм, що включає стовпці 'Date' і 'K_mod', виводиться на екран.

На рисунку 3.18 показано результат виконання описаного вище коду.

	Date	K_mod
0	2023-12-14	-0.385206
1	2023-12-13	-0.385206
2	2023-12-12	-0.385206
3	2023-12-11	-0.385206
4	2023-12-08	-0.385206
..	...	...
504	2022-01-07	0.545943
505	2022-01-06	0.061372
506	2022-01-05	-0.492353
507	2022-01-04	0.465567
508	2022-01-03	0.569681

Рисунок 3.18 – Сформований вектор що містить в собі нормовані значення.

Як показано на рисунку 3.19, значення було розраховано та нормалізовано успішно. Для перевірки отриманих значень було створено систему прийняття рішень яка базуючись на отриманих раніше даних буде визначати найкращі позиції для покупки та продажу валюти. Система працює за наступним алгоритмом:

– Додаються нові стовпці до датафрейму, які визначають сигнали для відкриття позиції на покупку або продаж. Сигнали формуються на підставі трендів та вектора ефективності K_mod.

– Алгоритм перевіряє умову, що тренд на попередній день був рівний або меншим за 2. Це може вказувати на тенденцію до позитивного руху. Далі, перевіряється поточний тренд, чи є він позитивним. Крім того, розглядається вектор ефективності K_mod, і сигнал покупки генерується, якщо значення K_mod більше 0.8. Об'єднання умов за допомогою операції І (&) гарантує, що всі три умови повинні виконуватись одночасно для генерації сигналу покупки.

– У випадку сигналу продажу алгоритм аналогічно перевіряє три умови. Перша умова перевіряє, чи тренд на попередньому день був рівний або

більшим за 0, що може вказувати на тенденцію до негативного руху. Друга умова перевіряє поточний негативний тренд, і, нарешті, вектор ефективності K_mod повинен бути менше або рівний -0.5 для генерації сигналу продажу.

– Код інтегрується по датасету, реагуючи на сигнали покупки та продажу. При спрацюванні сигналу покупки відкривається позиція, а при спрацюванні сигналу продажу вона закривається. За допомогою цих дій обчислюється прибуток чи збиток від кожної угоди. Також реєструється баланс і виводяться результати торгівлі.

– Початковий баланс складає 10 000, та на кожну покупку виділяється 10% від поточного балансу.

– На завершення код виводить графік цін з маркерами, що позначають моменти відкриття та закриття позицій. Також надає інформацію про угоди, загальний прибуток, початковий баланс та кількість успішних та невдалих угод.

Результат роботи системи прийняття рішень показано на рисунку 3.19 у якості графіку з відображенням сигналів покупки та продажу, та на рисунку 3.20 де відображена таблиця з детальною інформацією про кожну угоду.

Рисунок 3.19 – Графік з ідентифікованими моментами зміни тренду.

Деталі угод з різницею цін:

Покупка (дата)	Ціна покупки	Продаж (дата)	Ціна продажу	Прибуток	Різниця цін
2022-01-04	0.6812	2022-01-12	0.6844	4.697592	0.0032
2022-01-26	0.6651	2022-03-21	0.6884	35.032326	0.0233
2022-05-11	0.6299	2022-05-26	0.6477	28.258454	0.0178
2022-06-10	0.6370	2022-06-15	0.6284	-13.500785	-0.0086
2022-07-04	0.6205	2022-08-12	0.6453	39.967768	0.0248
2022-10-06	0.5656	2022-12-14	0.6454	141.089109	0.0798
2022-12-20	0.6347	2022-12-27	0.6275	-11.343942	-0.0072
2023-01-04	0.6293	2023-01-20	0.6472	28.444303	0.0179
2023-03-06	0.6194	2023-03-31	0.6257	10.171133	0.0063
2023-04-24	0.6166	2023-05-05	0.6293	20.596821	0.0127
2023-05-26	0.6048	2023-06-14	0.6204	25.793651	0.0156
2023-06-27	0.6162	2023-07-14	0.6368	33.430704	0.0206
2023-08-15	0.5949	2023-08-18	0.5921	-4.706673	-0.0028
2023-08-23	0.5979	2023-08-29	0.5971	-1.338016	-0.0008
2023-09-05	0.5884	2023-09-27	0.5921	6.288239	0.0037
2023-09-29	0.5995	2023-10-06	0.5987	-1.334445	-0.0008

Початковий баланс: 10000.0

Успішних угод: 11, Невдалих угод: 5

Баланс після всіх угод: 10341.546239141417

Рисунок 3.20 – Таблиця результатів.

На основі аналізу таблиці угод можна зробити висновок про успішну роботу системи прийняття рішень. Зафіксовано 16 угод, з яких 11 були успішними, а 5 - невдалими. Загальний прибуток від усіх угод склав 341.55 доларів США.

Для перевірки ефективності та адаптивності моделі, її також було використано для інших валютних пар. Для кожної з наступних валютних пар було зроблено описані в цьому розділі розрахунки та кластеризацію часового ряду, після чого отримані дані було використано в системі прийняття рішень.

На рисунках 3.21-3.22 показано графік з ідентифікованими моментами зміни тренду та таблицю результатів для валютної пари EUR_USD.

Рисунок 3.21 – Графік з ідентифікованими моментами зміни тренду для валютної пари EUR_USD.

Покупка (дата)	Ціна покупки	Продаж (дата)	Ціна продажу	Прибуток	Різниця цін
2022-01-26	1.1237	2022-02-02	1.1303	5.873454	0.0066
2022-03-03	1.1064	2022-03-16	1.1032	-2.892263	-0.0032
2022-03-22	1.1027	2022-03-31	1.1065	3.446087	0.0038
2022-04-27	1.0555	2022-04-29	1.0541	-1.326386	-0.0014
2022-05-11	1.0511	2022-05-26	1.0724	20.264485	0.0213
2022-06-10	1.0515	2022-06-23	1.0523	0.760818	0.0008
2022-07-12	1.0036	2022-07-18	1.0141	10.462336	0.0105
2022-07-25	1.0220	2022-08-09	1.0211	-0.880626	-0.0009
2022-08-19	1.0034	2022-09-09	1.0039	0.498306	0.0005
2022-09-22	0.9836	2022-11-10	1.0208	37.820252	0.0372
2022-11-11	1.0352	2023-01-23	1.0868	49.845440	0.0516
2023-02-22	1.0601	2023-02-28	1.0576	-2.358268	-0.0025
2023-03-06	1.0678	2023-04-11	1.0910	21.726915	0.0232
2023-04-14	1.1000	2023-04-25	1.0972	-2.545455	-0.0028
2023-05-26	1.0724	2023-06-14	1.0831	9.977620	0.0107
2023-06-27	1.0959	2023-07-17	1.1234	25.093530	0.0275
2023-09-29	1.0570	2023-11-16	1.0850	26.490066	0.0280

Початковий баланс: 10000.0

Успішних угод: 12, Невдалих угод: 5

Баланс після всіх угод: 10202.256311781697

Рисунок 3.22 – Таблиця результатів для валютної пари EUR_USD.

Таблиця містить дані про 22 угоди, які були здійснені з 26 січня 2022 року по 16 листопада 2023 року. Початковий баланс становив 10000 доларів США. Успішних угод було 12, а невдалих - 5. За результатами всіх угод баланс становив 10202,25 доларів США.

Максимальний прибуток від однієї угоди становить 49,84 доларів США, а максимальний збиток - 5,87 доларів США.

Узагальнюючи, можна сказати, що система прийняття рішень показала позитивний результат з даними цієї валютної пари.

На рисунках 3.23-3.24 показано графік з ідентифікованими моментами зміни тренду та таблицю результатів для валютної пари AUD_USD.

Рисунок 3.23 – Графік з ідентифікованими моментами зміни тренду для валютної пари AUD_USD.

Деталі угод з різницею цін:

Покупка (дата)	Ціна покупки	Продаж (дата)	Ціна продажу	Прибуток	Різниця цін
2022-01-04	0.7238	2022-01-11	0.7209	-4.006632	-0.0029
2022-01-26	0.7114	2022-04-04	0.7542	60.163059	0.0428
2022-05-11	0.6937	2022-06-06	0.7191	36.615252	0.0254
2022-07-04	0.6864	2022-08-10	0.7077	31.031469	0.0213
2022-08-31	0.6839	2022-09-09	0.6841	0.292440	0.0002
2022-10-14	0.6196	2022-11-11	0.6702	81.665591	0.0506
2022-11-17	0.6681	2022-11-29	0.6686	0.748391	0.0005
2022-12-02	0.6789	2022-12-09	0.6795	0.883783	0.0006
2022-12-21	0.6706	2023-01-24	0.7047	50.849985	0.0341
2023-03-06	0.6727	2023-04-12	0.6688	-5.797532	-0.0039
2023-04-25	0.6625	2023-05-05	0.6748	18.566038	0.0123
2023-05-30	0.6517	2023-06-14	0.6793	42.350775	0.0276
2023-06-27	0.6684	2023-07-12	0.6787	15.409934	0.0103
2023-08-15	0.6454	2023-08-28	0.6429	-3.873567	-0.0025
2023-09-04	0.6458	2023-09-15	0.6432	-4.026014	-0.0026

Початковий баланс: 10000.0

Успішних угод: 11, Невдалих угод: 4

Баланс після всіх угод: 10320.872970372999

Рисунок 3.24 – Таблиця результатів для валютної пари AUD_USD.

Як можна бачити на рисунку 3.24, загальна сума прибутку становить 320,87 доларів США, що є гарним показником. Крім того, кількість успішних угод більше, ніж невдалих, що також є позитивним знаком.

Результати системи є вдалими у різних часових інтервалах, здійснюючи угоди на протязі кількох днів до кількох місяців.

3.5. Висновки

У цьому розділі проведено експериментальні дослідження ефективності обраного підходу до кластеризації на часових рядах. Обґрунтовано вибір середовища програмної реалізації, проведено аналіз обраних бібліотек та підготовлено датасет для випробувань. В результаті проведених досліджень та успішної кластеризації даних було побудовано систему прийняття рішень, яка показала гарні результати у визначенні моментів зміни тенденцій, що дають змогу здійснювати операції купівлі або продажу фінансових інструментів.

4. ЕКОНОМІЧНИЙ РОЗДІЛ

Науково-технічна розробка має право на існування та впровадження, якщо вона відповідає вимогам часу, як в напрямку науково-технічного прогресу та і в плані економіки. Тому для науково-дослідної роботи необхідно оцінювати економічну ефективність результатів виконаної роботи.

4.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Як було зазначено раніше, головною метою трейдингу завжди є отримання стабільного прибутку та його підвищення в довгостроковій перспективі, необхідно враховувати те, що успіх в торгівлі цінними паперами залежить від того, наскільки краще та швидше трейдер розумітиме найменші коливання та зміни на ринку. Як наслідок, для досягнення найкращих результатів потрібно комбінувати існуючі методи аналізу ринків, що призводить до необхідності опрацювання обширного спектру ознак для ухвалення рішення.

Тому перед виконаною магістерською кваліфікаційною роботою було поставлено мету: підвищення ефективності прогнозування динаміки фінансового часового ряду шляхом ідентифікації його взаємозалежностей. Для цього було: детально вивчено модель для ідентифікації залежностей у часових рядах, розроблено математичну модель для кластеризації вибраних об'єктів.

В результаті було розроблено алгоритму та методики ідентифікації ринкових залежностей на основі впровадженої математичної моделі. Цей інноваційний підхід був конкретизований та впроваджений у формі автоматичної системи прийняття рішень. Підхід відзначається тим, що він дозволяє враховувати різноманітні фактори та показники, що можуть впливати на ринкові умови, забезпечуючи більш точну та комплексну ідентифікацію залежностей для подальшого автоматизованого прийняття рішень.

Для встановлення комерційного потенціалу розробленого алгоритму, було запрошено 3-х відомих експертів – кандидатів технічних наук, доцентів Софину О. Ю., Гармаша В.В. та Маслія Р. В.

Встановлення комерційного потенціалу розробленого алгоритму було здійснено за критеріями, наведеними в таблиці 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання технічного рівня та комерційного потенціалу будь-якої розробки і їх бальна оцінка

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту	Технічні та споживчі властивості продукту	Технічні та споживчі властивості продукту на	Технічні та споживчі властивості продукту	Технічні та споживчі властивості продукту значно

	значно гірші, ніж в аналогів	трохи гірші, ніж в аналогів	рівні аналогів	трохи кращі, ніж в аналогів	кращі, ніж в аналогів
Ринкові перспективи					
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Продовження таблиці 4.1

	Критерії оцінювання та бали (за 5-ти бальною шкалою)				
	0	1	2	3	4
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно найняти фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування.	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві

11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін Реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років.Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Запрошені експерти оцінили розроблений нами мобільний додаток та програмний інтерфейс таким чином (див. таблицю 4.2):

Таблиця 4.2 – Результати технологічного аудиту розробленого мобільного додатка (за шкалою оцінювання 0-1-2-3-4)

Критерії	Прізвище, ініціали експерта		
	Софина О. Ю.	Маслій Р. В.	Гармаш В.В.
	Бали, виставлені експертами:		
1	3	4	4
2	4	3	4
3	3	3	3
4	4	4	4
5	4	3	4
6	4	3	3
7	3	4	4
8	4	4	3
9	3	4	4
10	3	4	4
11	4	4	4
12	3	4	4
Сума балів	$СБ_1 = 42$	44	45

Середньоарифметична сума балів $\overline{СБ}$, що їх виставили експерти, становила:

$$\overline{СБ} = \frac{\sum_{1}^{3} Б_i}{3} = \frac{42 + 44 + 45}{3} = \frac{131}{3} = 43,67$$

Встановлення комерційного потенціалу розробленого нами мобільного додатка та програмного інтерфейсу будемо здійснювати на основі рекомендацій, наведених в таблиці 4.3 [1].

Таблиця 4.3 – Рівні комерційного потенціалу будь-якої наукової розробки

Середньоарифметична сума балів $\overline{СБ}$, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0 – 10	Низький
11 – 20	Нижче середнього
21 – 30	Середній
31 – 40	Вище середнього
41 – 48	Високий

Оскільки середньоарифметична сума балів, що їх виставили експерти, складає 43,67 балів, то це свідчить, що розроблений нами алгоритм має рівень комерційного потенціалу, який вважається «високим».

Це пояснюється тим, що розробка алгоритму та методики ідентифікації ринкових залежностей на основі впровадженої математичної моделі дозволяє враховувати різноманітні фактори та показники, що можуть впливати на ринкові умови, забезпечуючи більш точну та комплексну ідентифікацію залежностей для подальшого автоматизованого прийняття рішень.

4.2 **Розрахунок витрат на розроблення алгоритму та методики ідентифікації ринкових залежностей на основі впровадженої математичної моделі**

При виконанні роботи були зроблені певні витрати.

Зокрема:

А) Основна заробітна плата $З_о$ розробників, яка визначається за формулою:

$$З_о = \frac{М}{Т_р} \cdot t \text{ грн}, \quad (4.1)$$

де М – місячний посадовий оклад розробника, грн; приймемо, що

М = (6700…23000) грн/місяць;

$Т_р$ – число робочих днів в місяці; приймемо $Т_р$ = 20 день;

t – число днів роботи розробників.

Зроблені розрахунки зведемо до таблиці 4.4:

Таблиця 4.4 – Основна заробітна плата розробників

Найменування посади виконавця	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на оплату праці, грн
1. Науковий керівник магістерської роботи	20000	1000	20 годин	≈ 3333
2. Магістрант-студент-виконавець	2000 (беремо 6700)	335	80	≈ 26800
3. Консультант з економічної частини	18000	900	1,5 години	≈ 225 (при 6-годинному робочому дні)
Загалом				$З_о$ = 30 300 грн

Б) Додаткова заробітна плата $З_д$ розробників розраховується як (10…12)% від величини їх основної заробітної плати, тобто:

$$З_д = \alpha \cdot З_о = (0{,}1...0{,}12) \cdot З_о \qquad (4.2)$$

Приймемо, що α = 0,11. Тоді для випадку отримаємо:

$$З_д = 0{,}11 \times 30300 = 3333 \text{ грн.}$$

В) Нарахування на заробітну плату $НЗП_{зп}$ розробників (дослідників) розраховуються за формулою:

$$НЗП_{зп} = (З_о + З_д) \cdot \frac{\beta}{100}, \qquad (4.3)$$

де β – ставка обов'язкового єдиного внеску на державне соціальне страхування, %. β = 22%. Тоді:

$$НЗН_{зп} = (30300 + 3333) \times 0,22 = 7399,26 \approx 7400 \text{ грн.}$$

Г) Амортизація основних засобів А, які використовувались під час виконання цієї роботи:

$$А = \frac{Ц \cdot Н_а}{100} \cdot \frac{Т}{12} \text{ грн,} \qquad (4.4)$$

де Ц – загальна балансова вартість основних засобів, грн;

$Н_а$ – річна норма амортизаційних відрахувань. Для випадку можна прийняти, що $Н_а$ = (2,5...25)%;

Т – термін використання основних засобів, місяці.

Зроблені розрахунки зведено в таблицю 4.4.

Таблиця 4.5 – Розрахунок амортизаційних відрахувань

Найменування обладнання, приміщень тощо	Балансова вартість, грн.	Норма амортизації, %	Термін використання, міс.	Величина амортизаційних відрахувань, грн
1. Комп'ютерна техніка, обладнання тощо	48000	25	3,1 (при 85% використанні	2635
2. Приміщення університету, кафедри	22000	3,5	3,1 при 90% використанні	≈ 179
Всього				А = 2814 грн

Д) Витрати на матеріали М розраховуються за формулою:

$$М = \sum_{1}^{n} Н_i \cdot Ц_i \cdot К_i - \sum_{1}^{n} В_i \cdot Ц_в \text{ грн.} \quad (4.5)$$

де $Н_i$ – витрати матеріалу *і*-го найменування, кг; $Ц_i$ – вартість матеріалу *і*-го найменування; $К_i$ – коефіцієнт транспортних витрат, $К_i$ = (1,1…1,15); $В_i$ – маса відходів матеріалу *і*-го найменування; $Ц_в$ – ціна відходів матеріалу *і*-го найменування; n – кількість видів матеріалів.

Е) Витрати на комплектуючі К розраховуються за формулою:

$$К = \sum_{1}^{n} Н_i \cdot Ц_i \cdot К_i \text{ грн} \quad (4.6)$$

де $Н_i$ – кількість комплектуючих *і*-го виду, шт.; $Ц_i$ – ціна комплектуючих *і*-го виду; $К_i$ – коефіцієнт транспортних витрат, $К_i$ = (1,1…1,15); n – кількість видів комплектуючих.

Під час виконання роботи загальні витрати на матеріали та комплектуючі склали приблизно 1000 грн.

Ж) Витрати на силову електроенергію $В_е$ розраховуються за формулою:

$$В_е = \frac{В \cdot П \cdot Ф \cdot К_п}{К_д} \quad (4.7)$$

де В – вартість 1 кВт-год. електроенергії, в 2023 р. В ≈ 4,5 грн/кВт;

П – установлена потужність обладнання, кВт; П = 1,0 кВт;

Ф – фактична кількість годин роботи обладнання, годин.

Приймемо, що Ф = 200 годин;

$К_п$ – коефіцієнт використання потужності; $К_п$ < 1 = 0,85.

$К_д$ – коефіцієнт корисної дії, $К_д$ = 0,8.

Тоді витрати на силову електроенергію будуть дорівнювати:

$$В_е = \frac{В \cdot П \cdot Ф \cdot К_п}{К_д} = \frac{4{,}5 \cdot 1{,}0 \cdot 200 \cdot 0{,}85}{0{,}8} = 956{,}25 \approx 956 \text{ грн}$$

И) Інші витрати $В_{інш}$ можна прийняти як (50…300)% від основної заробітної плати розробників, тобто:

$$В_{інш} = (0,5....3) \times З_о \quad (4.8)$$

Для випадку отримаємо:

$$В_{інш} = 1,3 \times 30300 = 39390 \text{ грн.}$$

К) Сума всіх попередніх статей витрат складає витрати на виконання роботи безпосередньо розробником-магістрантом – В.

$$В = 30300+ 3333 + 7400 + 2814 + 1000+956 + 39390 = 85193 \text{ грн.}$$

Л) Загальні витрати на розроблення мобільного додатка та програмного інтерфейсу $В_{заг}$ становлять:

$$В_{заг} = \frac{В}{\beta} \quad (4.9)$$

де β – коефіцієнт, який характеризує етап (стадію) виконання цієї роботи. Можна прийняти, що, $\beta \approx 0,90$, оскільки робота практично завершена.

Тоді:

$$В_{заг} = \frac{85193}{0,9} = 94658,89 \approx 94660 \text{ грн}$$

Тобто прогнозовані загальні витрати на розробку мобільного додатка та програмного інтерфейсу становлять приблизно 94660 тисячі грн.

4.3 **Розрахунок економічного ефекту від можливої комерціалізації розробки**

Економічний ефект від впровадження та можливої комерціалізації розробленого алгоритму та методики ідентифікації ринкових залежностей на основі впровадженої математичної моделі пояснюється його значно кращими прикладними можливостями. Тому розробку можна реалізовувати на ринку дещо дорожче, ніж аналогічні методи.

Кластерний аналіз часових рядів може бути корисним із економічного погляду в різних контекстах. Відповіді на питання про економічний попит можуть залежати від конкретного застосування та сфери економіки.

Аналіз ринку також показав, що потенційна кількість замовників такого продукту з реалізованим алгоритмом ідентифікації ринкових залежностей становить 6-8 компаній на рік. Для розрахунків приймемо 8 клієнтів. Разом з тим, проведений аналіз показав, що кількість таких клієнтів буде зростати, тобто можна очікувати зростання попиту на розробку принаймні протягом 3-х років після її впровадження.

Тобто, якщо розробка буде впроваджена з 1 січня 2024 року, то її результати будуть виявлятися протягом 2024-го, 2025-го та 2026-го років.

Прогноз зростання попиту на розробку складає по роках:

а) 2024 р. – приблизно +5 шт. до базового року;

б) 2025 р. – +15 шт. до базового року;

в) 2026 р. – +10 шт. до базового року (оскільки можуть з’явитися ще кращі розробки, зроблені студентами-магістрантами ВНТУ).

Можливе збільшення чистого прибутку $\Delta П_i$, що його може отримати потенційний інвестор від виведення розробки на ринок, становитиме:

$$\Delta П_i = \sum_{1}^{n} (\Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\upsilon}{100}) \qquad (4.10)$$

де $\Delta Ц_о$ – покращення основного якісного показника від впровадження результатів розробки у цьому році. Для випадку це є збільшення ціни реалізації розробки $\Delta Ц_о$ = 70 - 50 = + 20 тисяч грн;

N – основний кількісний показник, який визначає обсяг діяльності у році до впровадження результатів розробки; N = 8 шт.;

ΔN – покращення основного кількісного показника від впровадження результатів розробки.

Таке покращення становитиме по роках, відповідно: у 2024 році – + 5 шт., у 2025 році + 15 шт., та у 2026 році + 10 шт.;

$Ц_о$ – основний якісний показник (тобто ціна), який визначає обсяг діяльності у році після впровадження результатів розробки, грн; $Ц_о$= 70 тисяч грн;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки; для випадку n = 3;

λ – коефіцієнт, який враховує сплату податку на додану вартість; $\lambda = 0{,}8333$;

ρ – коефіцієнт, який враховує рентабельність продукту. Рекомендується приймати ρ = (0,2...0,5); візьмемо ρ = 0,5;

υ – ставка податку на прибуток. У 2023-25 роках υ = 18%.

Тоді можливе зростання чистого прибутку $\Delta П_1$ для потенційного інвестора протягом першого року від можливого впровадження розробки (2024 р.) становитиме:

$$\Delta П_1 = [20 \cdot 8 + 70 \cdot 5] \cdot 0{,}8333 \cdot 0{,}5 \cdot (1 - \frac{18}{100}) \approx 174 \text{ тис. грн.}$$

Можливе зростання чистого прибутку $\Delta П_2$ для потенційного інвестора від можливого впровадження розробки протягом другого (2025) року складе:

$$\Delta П_2 = [20 \cdot 8 + 70 \cdot 15] \cdot 0{,}8333 \cdot 0{,}5 \cdot (1 - \frac{18}{100}) \approx 413 \text{ тис. грн.}$$

Можливе зростання чистого прибутку $\Delta П_3$ для потенційного інвестора від можливого впровадження розробки протягом третього (2026) року складе:

$$\Delta П_3 = [20 \cdot 8 + 70 \cdot 10] \cdot 0,8333 \cdot 0,5 \cdot (1 - \frac{18}{100}) \approx 294 \text{ тис. грн.}$$

Приведена вартість зростання всіх чистих прибутків від можливого впровадження розробки становитиме:

$$ПП = \sum_{1}^{т} \frac{\Delta П_i}{(1+\tau)^t} \quad (4.11)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої роботи, грн;

т – період часу, протягом якого виявляються результати впровадженої роботи, роки. Для випадку т = 3 роки;

τ – ставка дисконтування. Приймемо τ = 0,10 (10%);

t – період часу від моменту початку розроблення мобільного додатка до моменту отримання можливих чистих прибутків потенційним інвестором.

Тоді приведена вартість зростання всіх можливих чистих прибутків ПП, що їх може отримати потенційний інвестор від комерціалізації розробки, складе:

$$ПП = \frac{174}{(1+0,1)^2} + \frac{413}{(1+0,1)^3} + \frac{294}{(1+0,1)^4} \approx 144 + 310 + 201 = 655 \text{ тисяч грн.}$$

Теперішня вартість інвестицій PV (вартість стартапу), що повинні бути вкладені для реалізації розробки: PV = (1,0…5) × $В_{заг}$.

Для випадку PV = (1,0…5) × 94.6 = 2 × 94,6 = 189,2 тисяч грн.

Абсолютний ефект від можливих вкладених інвестицій $Е_{абс}$.

$$Е_{абс} = ПП - PV \quad (4.12)$$

де ПП – приведена вартість збільшення всіх чистих прибутків для інвестора від можливого впровадження розробки, грн;

PV – теперішня вартість інвестицій PV = 189,2 тисяч грн.

Абсолютний ефект від можливого впровадження розробки складе:

$$E_{абс} = 655 – 189,2 = 465,8 \text{ тисяч грн.}$$

Оскільки $E_{абс} > 0$, то комерціалізація розробки може бути доцільною.

Далі розрахуємо внутрішню дохідність $E_в$ вкладених інвестицій:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1 \qquad (4.13)$$

де $E_{абс}$ – абсолютний ефект вкладених інвестицій; $E_{абс}$ = 465,8 тис. грн;

PV –теперішня вартість початкових інвестицій PV = 189,2 тис. грн;

$T_ж$ – життєвий цикл розробки, роки.

$T_ж$ = 4 років (2023-й, 2024-й, 2025-й, 2026-й роки)

Для випадку отримаємо:

$$B_{заг} = \sqrt[4]{1 + \frac{465,8}{189,2}} - 1 = 1,373 - 1 = 0,373 = 37,3\%$$

Далі визначимо ту мінімальну дохідність, нижче за яку потенційному інвестору не вигідно буде займатися комерціалізацією розробки.

Мінімальна дохідність або мінімальна (бар'єрна) ставка дисконтування $\tau_{мін}$ визначається за формулою:

$$\tau_{мін} = d + f \qquad (4.14)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2022-2023 роках в Україні d = (0,10...0,12);

f – показник, що характеризує ризикованість вкладень;

f = (0,1...0,50). Приймемо f = 0,25.

Для випадку отримаємо:

$$\tau_{мін} = 0,12 + 0,25 = 0,37 \text{ або } \tau_{мін} = 37\%.$$

Оскільки величина $E_в = 37,3\% > \tau_{мін} = 37\%$, то потенційний інвестор у принципі може бути зацікавлений у фінансуванні та комерціалізації розробки.

Далі розраховуємо термін окупності коштів, вкладених у можливу комерціалізацію розробленого алгоритму та методики ідентифікації ринкових залежностей.

Термін окупності $T_{ок}$ розраховується за формулою:

$$T_{ок} = \frac{1}{E_в} \qquad (4.15)$$

Для випадку термін окупності $T_{ок}$ коштів становитиме:

$$T_{ок} = \frac{1}{0,373} = 2,68 \text{ років} < 3 \text{ років},$$

що свідчить про потенційну доцільність комерціалізації розробленого нами алгоритму та методики ідентифікації ринкових залежностей.

Далі проведено моделювання залежності величини внутрішньої дохідності вкладених потенційних інвестицій від рівня інфляції в країні.

Якщо рівень інфляції в країні зросте до 20%, то:

$$ПП = \frac{174}{(1+0,2)^2} + \frac{413}{(1+0,2)^3} + \frac{294}{(1+0,2)^4} \approx 121 + 239 + 142 = 502 \text{ тисяч грн.}$$

Тоді абсолютний ефект від можливого впровадження розробки за три роки складе:

$$E_{абс} = 502 – 189,2 = 312,8 \text{ тисяч грн.}$$

Внутрішня дохідність $E_в$ вкладених інвестицій становитиме:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1 \qquad (4.16)$$

де $E_{абс}$ – абсолютний ефект вкладених інвестицій; $E_{абс}$ = 312,8 тисяч грн;

PV –теперішня вартість початкових інвестицій PV = 189,2 тисяч грн.

Для випадку отримаємо:

$$B_{заг} = \sqrt[4]{1 + \frac{312,8}{189,2}} - 1 = 1,276 - 1 = 0,276 = 27,6\%$$

Оскільки величина $E_в$ = 28,5% < $\tau_{мін}$ = 37%, то потенційний інвестор може бути НЕ зацікавлений у фінансуванні та комерціалізації розробки.

Зроблені розрахунки у вигляді графіків наведено на рис. 4.1.

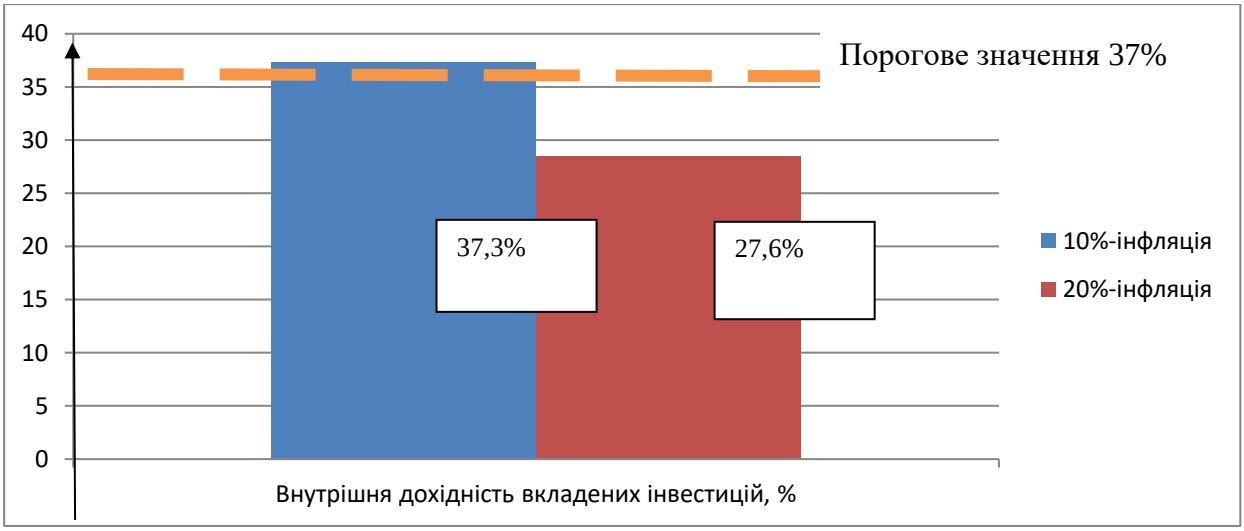


Рисунок 4.1 – Моделювання залежності величини внутрішньої дохідності потенційних інвестицій від рівня інфляції в країні

Аналіз діаграм на рис 41 показує, що при рівні інфляції в 10% величина внутрішньої дохідності інвестицій становить $E_в$ = 37,3%, що більше порогового значення $\tau_{мін}$ = 37% і тому комерціалізація розробки може бути доцільною. При рівні інфляції в 20% величина внутрішньої дохідності інвестицій, вкладених в комерціалізацію розробки, становить всього $E_в$ = 27,6%, що менше порогового

значення $\tau_{мін} = 37\%$, і тому комерціалізація розробки потенційним інвестором може бути проблематичною.

Остаточне рішення з цього питання потребує проведення додаткових розрахунків (можливо – зниження рівня ризикованості вкладень тощо).

Результати виконаної економічної частини магістерської кваліфікаційної роботи зведено у таблицю:

Показники	Задані у ТЗ	Досягнуті у магістерській кваліфікаційній роботі	Висновок
1. Витрати на розробку	Не більше 100 тис. грн	94660 грн.	Досягнуто
2. Абсолютний ефект від впровадження розробки, тисяч грн	Не менше 450 тисяч грн	465,8 тисяч грн (при 10%-інфляції)	Виконано
3. Внутрішня дохідність інвестицій, %	не менше 37%	37,3%	Досягнуто
4. Термін окупності інвестицій, роки	до 3-ти років	2,68 років	Виконано

Таким чином, основні техніко-економічні показники розробленого нами алгоритму та методики ідентифікації ринкових залежностей в часових рядах, були успішно виконані.

ВИСНОВКИ

У першому розділі проведено аналіз предметної області, що пов'язана з використанням часових рядів у завданнях кластеризації. Розглянуті основні аспекти використання часових рядів, визначено ключові терміни та методи, зокрема, кластеризацію часових рядів. Здійснений огляд методів кластеризації. На підставі проведеного аналізу сформульована постановка задачі дослідження, яка визначає цільові напрямки та завдання, що розв'язуються у наступних розділах роботи.

Другий розділ присвячено аналізу алгоритмів попередньої обробки часових рядів та алгоритмів кластеризації. Перший етап аналізу включав перевірку статистичних гіпотез щодо властивостей часового ряду під час попереднього аналізу вибірки, що надало глибше розуміння структури даних перед застосуванням алгоритмів кластеризації. Далі розглядалися методи зменшення розмірності та міри відстані для послідовних даних. Висновки розділу підкреслюють важливість вибору оптимальних методів попередньої обробки та переваги використання відповідних алгоритмів кластеризації у контексті аналізу часових рядів, встановлюючи базовий фреймворк для подальших досліджень та розробки ефективних аналітичних інструментів.

У третьому розділі проведено експериментальні дослідження, спрямовані на визначення ефективності обраного підходу до кластеризації часових рядів. Етапи включали вибір середовища програмної реалізації, аналіз обраних бібліотек, підготовку та аналіз датасету, визначення та обрахунок необхідних показників для створення вектора ефективності та на його основі було проведено кластеризацію даних та програмну реалізацію алгоритмів. В результаті роботи було отримано систему прийняття рішень, яка показала гарні результати у визначенні моментів зміни тенденцій, що дають змогу здійснювати операції купівлі або продажу фінансових інструментів.

У четвертому розділі, економічній частині, проведено розрахунки для визначення вартості розробки та потенційного доходу. Згідно з економічним аналізом, термін окупності інвестицій складає 2,68 років.

ПЕРЕЛІК ПОСИЛАНЬ

1. Volosyuk Y. (2020) Analysis of clustering algorithm for Data Mining tasks, pp. 112–119.

2. Aggarwal, C. C. (2015). Data mining: the textbook. Springer.

3. Birant, D. (2011). Data mining using RFM analysis. IntechOpen.

4. Юрченко М. Є. (2019). Прогнозування та аналіз часових рядів.

5. Kay S. M. (1993), Fundamentals of statistical signal processing.

6. Han, J., Pei, J., & Kamber, M. (2011). Data mining: concepts and techniques. Elsevier.

7. Liao, T. W. (2005). Clustering of time series data—a survey. Pattern recognition, 38(11), 1857-1874.

8. Кобилін, І. О. (2019). Нечітка кластеризація часових рядів в інтелектуальному аналізі потоків даних.

9. Mantula, E., & Mashtalir, V. (2013, June). An Adaptive Forecasting of Nonlinear Nonstationary Time Series under Short Learning Samples. In ICTERI (pp. 91-98).

10. Гороховатський, В. О., & Творошенко, І. С. (2021). Методи інтелектуального аналізу та оброблення даних: навч. посібник.

11. Mashtalir, S., & Mashtalir, V. (2020). Spatio-temporal video segmentation. In Advances in Spatio-Temporal Segmentation of Visual Data (pp. 161-210). Springer, Cham

12. Kinoshenko, D., Mashtalir, V., & Shlyakhov, V. (2007). A partition metric for clustering features analysis.

13. Mashtalir, V., Ruban, I., & Levashenko, V. (Eds.). (2020). Advances in Spatio-Temporal Segmentation of Visual Data. Springer.

14. Jolliffe I.T. Principal Component Analysis, Series: Springer Series in Statistics, 2nd ed., Springer, NY, 2002, XXIX, 487 p. 28 illus. ISBN 978-0-387-95442-4.

15. Chupikov, A., Kinoshenko, D., Mashtalir, V., & Shcherbinin, K. (2006,

July). Image retrieval with segmentation-based query. In International Workshop on Adaptive Multimedia Retrieval (pp. 207-221). Springer, Berlin, Heidelberg.

16. Mantula, E., & Mashtalir, V. (2013, June). An Adaptive Forecasting of Nonlinear Nonstationary Time Series under Short Learning Samples. In ICTERI (pp. 91-98).

17. Kinoshenko, D., Mashtalir, V., Shlyakhov, V., & Yegorova, E. (2012). nested Partitions Properties for spatial content Image retrieval. In Multimedia Storage and Retrieval Innovations for Digital Library Systems (pp. 240-269). IGI Global.

18. Точне обчислення середніх так коваріацій методом Велфорда. URL: https://habr.com/ru/post/333426/#3-vychislenie-srednih [Дата звернення: 01.10.2023].

19. Welford's online algorithm. URL: https://en.wikipedia.org/wiki/Algorithms_for_calculating_variance#Online_algorithm [Дата звернення: 02.10.2023].

20. Mashtalir V., Mashtalir S. (2020). Video shot boundary detection via sequential clustering.

21. Mashtalir, S., Mashtalir, V., & Stolbovyi, M. (2018, August). Representative based clustering of long multivariate sequences with different lengths. In 2018 IEEE second international conference on Data Stream Mining & Processing (DSMP) (pp. 545-548). IEEE.

22. Bogucharskiy, S., & Mashtalir, V. (2014). On matrix modification of clarans clustering method in large video surveillance databases.

23. Bogucharskiy, S., Mashtalir, V. (2015). Image segmentation via Xmeans under overlapping classes.

24. Bobrowski, L., Mashtalir, V., Topczewska, M. (2007) Pattern discovery through separable data projections.

25. Mashtalir, V., & Bogucharskiy, S. (2015). Covering Image Segmentation via Matrix X-means and J-means Clustering. Sensors & Transducers, 195(12), 56.

26. Mashtalir, V. P., Shlyakhov, V. V., & Yakovlev, S. V. (2014). Group structures on quotient sets in classification problems. Cybernetics and Systems

Analysis, 50(4), 507-518.

27. Kagramanyan, A., Mashtalir, V., & Shlyakhov, V. (2014). Metrical evaluation of spatial content for segmentationbased image retrieval.\

28. Maulik, U., & Bandyopadhyay, S. (2002). Performance evaluation of some clustering algorithms and validity indices. IEEE Transactions on pattern analysis and machine intelligence, 24(12), 1650-1654.

29. Albon, C., & VanderPlas, J. (2015). Learning Jupyter: Interactive computing for data science. O'Reilly Media.**

30. Schafer, C. (2023). Jupyter Notebooks: The Complete Guide [Udemy video course].

31. McKinney, W. (2017). Python for data analysis: A gentle introduction (2nd ed.). O'Reilly Media.**

32. Matplotlib in Python. URL: https://education–wiki.com/8072875–matplotlib–in–python [Дата звернення: 02.10.2023].

33. NumPy in Python. URL: https://numpy.org/ [Дата звернення: 17.10.2023].

34. Scikit-learn in Python. URL: https://scikit-learn.org/ [Дата звернення: 17.10.2023].

35. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт. / Укладачі В.О. Козловський, О.Й. Лесько, В.В.Кавецький. – Вінниця : ВНТУ, 2021. – 42 с

Додаток А (обов’язковий)
Технічне завдання

ВНТУ

ЗАТВЕРДЖЕНО
Завідувач кафедри АІІТ,
д.т.н., професор
_____ Олег БІСІКАЛО
“ __ ” _____ 2023 р.

ТЕХНІЧНЕ ЗАВДАННЯ
на виконання магістерської кваліфікаційної роботи

Використання кластерного аналізу для ідентифікації залежностей на часових рядах
(тема)

08-33.МКР.___.00.000 ТЗ

Студент групи 1ІСТ-22м
Радислав ГРІНЧАК
_____

Керівник: д.т.н., доцент,
Владислав КАБАЧІЙ
_____

Вінниця 2023

1. Назва та галузь застосування

1.1. Назва – Використання кластерного аналізу для ідентифікації залежностей на часових рядах.

1.2. Галузь застосування – Аналіз фінансових даних для ідентифікації спільних торговельних підходів серед інвесторів та розуміння їхнього впливу на ринкові динаміки.

2. Підстава для проведення розробки.

Тема магістерської кваліфікаційної роботи затверджена наказом по ВНТУ №247 від "18" вересня 2023р.

3. Мета та призначення розробки.

Метою роботи є підвищення ефективності прогнозування динаміки фінансового часового ряду шляхом ідентифікації його залежностей на основі використання кластерного аналізу, розробка системи прийняття рішень.

4. Джерела розробки.

Магістерська кваліфікаційна робота виконується вперше. В ході проведення розробки повинні використовуватись такі документи:

1. Ситник В. Ф. Інтелектуальний аналіз даних (дейтамайнінг):Навч. посібник. — К.: КНЕУ, 2007. — 376 с.
2. Чубукова И. А. Data Mining: учебное пособие. — М.: Интернет университет информационных технологий: БИНОМ: Лаборатория знаний, 2006. — 382 с. — ISBN 5-9556-0064-7.
3. Ralph Kimball, Joe Caserta. The Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data. — John Wiley & Sons, 2004. — 528 p. — ISBN 978-0-764-56757-5.
4. David Loshin. ETL (Extract, Transform, Load) // Business Intelligence. — 2nd. — Morgan Kaufmann, 2012. — 400 p. — ISBN 978-0-12-385890-0

5. Вимоги до розробки.

5.1. Перелік головних етапів виконання розробки:

- Ретроспективний аналіз часових рядів для визначення основних паттернів та тенденцій.
- Порівняльний огляд існуючих методів та алгоритмів кластерного аналізу часових рядів.
- Формулювання вимог до системи для ефективного виявлення та класифікації залежностей у часових рядах.
- Проектування та розробка системи кластерного аналізу для ідентифікації взаємозв'язків у динаміці часових рядів.
- Оцінка ефективності системи та аналіз її результативності на реальних часових рядах.
- Здійснення інтеграції системи у робочі процеси з прийняття рішень відповідно до визначених вимог та функціональності.
-

5.2. Основні технічні вимоги до розробки.

5.2.1. Вимоги до програмної платформи:

- WINDOWS 7\8\10.

5.2.2. Умови експлуатації системи:

- робота на стандартних ПЕОМ в приміщеннях зі стандартними умовами;
- можливість цілодобового функціонування системи;
- текст програмного забезпечення системи є цілком закритим.

6. Стадії та етапи розробки.

6.1 Пояснювальна записка:

- Дослідження актуальності поставленої задачі 04.09.2023р.
- Дослідження існуючих методів кластеризації 07.09.2023р.
- Збір даних та формування датасету 13.09.2023р.
- Аналіз даних та створення алгоритму класифікації 25.09.2023р.
- Апробація результатів дослідження 15.10.2023р.
- Оформлення пояснювальної записки, графічного матеріалу і презентації 29.10.2023р.

6.2 Графічні матеріали:

- Графік динаміки дисперсії часового ряду 05.11.2023р.
- Графік результатів кластеризації 14.11.2023р.
- Результуючий графік системи прийняття рішень 20.11.2023р.

7. Порядок контролю і приймання.

7.1. Хід виконання роботи контролюється керівником роботи. Рубіжний контроль провести до «28» листопада 2023 р.

7.2. Атестація проекту здійснюється на попередньому захисті. Попередній захист магістерської кваліфікаційної роботи провести до «10» грудня 2023р.

7.3. Підсумкове рішення щодо оцінки якості виконання роботи приймається на засіданні ЕК.

7.4. Захист магістерської кваліфікаційної роботи провести до «20» грудня 2023р.

Додаток Б (обов’язковий)
ІЛЮСТРАТИВНА ЧАСТИНА

Використання кластерного аналізу для ідентифікації залежностей на часових рядах.

Студент групи 1ІСТ-22м
Радислав ГРІНЧАК
_____

Керівник: д.т.н., доцент,
Владислав КАБАЧІЙ
_____

Вінниця 2023

Рисунок Б1. Валютний часовий ряд.

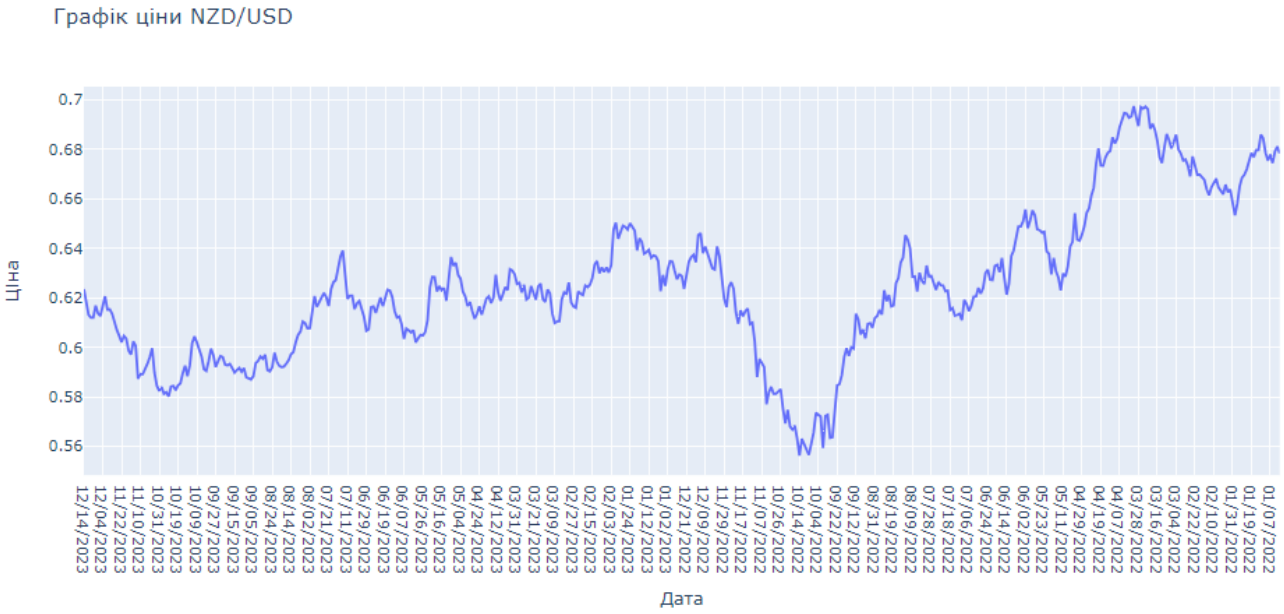


Рисунок Б2. Графік ціни та ковзних середніх.

Рисунок Б3. Графік осцилятору «MACD».

Рисунок Б4. Графік осцилятору «Стохастик».

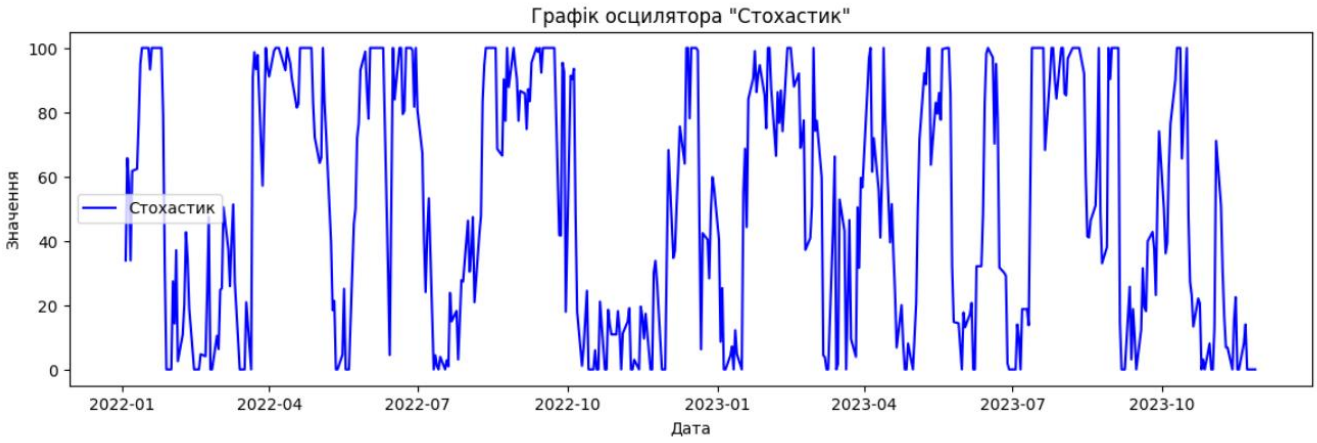


Рисунок Б5. Графік вектору ефективності «K_mod».

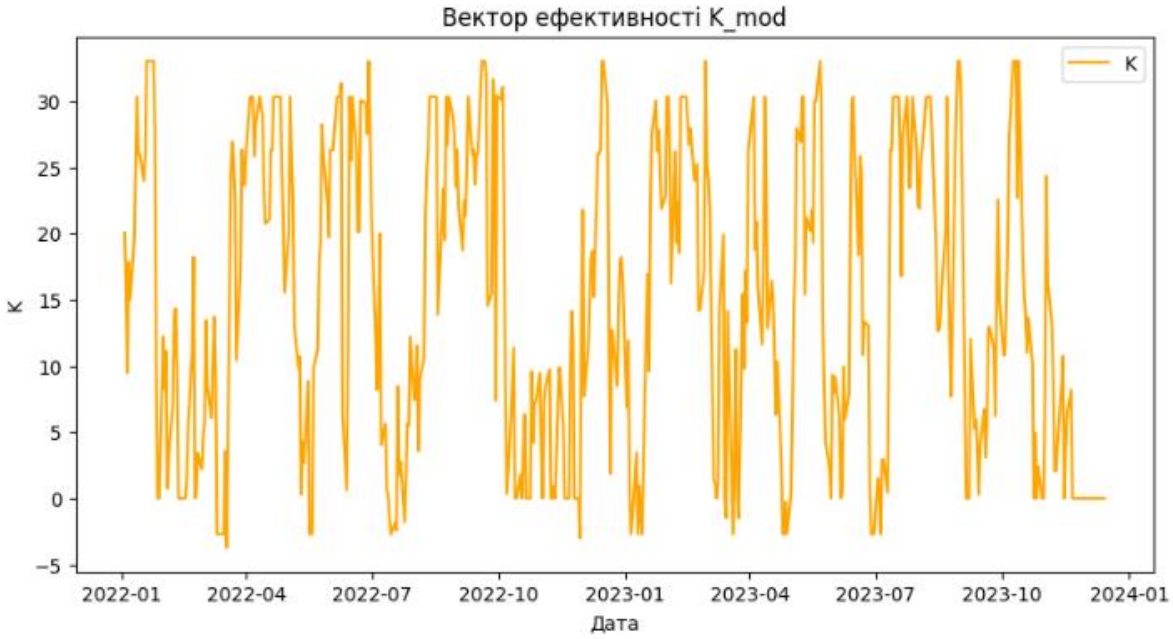


Рисунок Б6. Графік результатів кластеризації даних матриці спостереження.

Рисунок Б7. Графік з ідентифікованими моментами зміни тренду.

Рисунок Б8. Таблиця результатів угод.

Деталі угод з різницею цін:

Покупка (дата)	Ціна покупки	Продаж (дата)	Ціна продажу	Прибуток	Різниця цін
2022-01-04	0.6812	2022-01-12	0.6844	4.697592	0.0032
2022-01-26	0.6651	2022-03-21	0.6884	35.032326	0.0233
2022-05-11	0.6299	2022-05-26	0.6477	28.258454	0.0178
2022-06-10	0.6370	2022-06-15	0.6284	-13.500785	-0.0086
2022-07-04	0.6205	2022-08-12	0.6453	39.967768	0.0248
2022-10-06	0.5656	2022-12-14	0.6454	141.089109	0.0798
2022-12-20	0.6347	2022-12-27	0.6275	-11.343942	-0.0072
2023-01-04	0.6293	2023-01-20	0.6472	28.444303	0.0179
2023-03-06	0.6194	2023-03-31	0.6257	10.171133	0.0063
2023-04-24	0.6166	2023-05-05	0.6293	20.596821	0.0127
2023-05-26	0.6048	2023-06-14	0.6204	25.793651	0.0156
2023-06-27	0.6162	2023-07-14	0.6368	33.430704	0.0206
2023-08-15	0.5949	2023-08-18	0.5921	-4.706673	-0.0028
2023-08-23	0.5979	2023-08-29	0.5971	-1.338016	-0.0008
2023-09-05	0.5884	2023-09-27	0.5921	6.288239	0.0037
2023-09-29	0.5995	2023-10-06	0.5987	-1.334445	-0.0008

Початковий баланс: 10000.0

Успішних угод: 11, Невдалих угод: 5

Баланс після всіх угод: 10341.546239141417

Додаток В (обов'язковий)
Лістинг програмного забезпечення

```
# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:33.067007Z","iopub.execute_input":"2023-12-17T05:11:33.067480Z","iopub.status.idle":"2023-12-17T05:11:33.079534Z","shell.execute_reply.started":"2023-12-17T05:11:33.067443Z","shell.execute_reply":"2023-12-17T05:11:33.078120Z"}}
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:33.081662Z","iopub.execute_input":"2023-12-17T05:11:33.082062Z","iopub.status.idle":"2023-12-17T05:11:33.115201Z","shell.execute_reply.started":"2023-12-17T05:11:33.082029Z","shell.execute_reply":"2023-12-17T05:11:33.114045Z"}}
import pandas as pd

# Шляхи до датасетів
nzd_usd_path = "/kaggle/input/usd-chf/NZD_USD.csv"

# Завантаження датасетів за допомогою pandas
nzd_usd_data = pd.read_csv(nzd_usd_path)

nzd_usd_data.insert(0, 'id', range(1, 1 + len(nzd_usd_data)))

# Для кожного датасету
def preprocess_dataset(data):
    # Видалення стовпців 'Vol.' і 'Change %'
    data = data.drop(['Vol.', 'Change %', 'Vol', 'Diff', 'Change'], axis=1, errors='ignore')
```

```
    # Перейменування стовпців 'Max' і 'Min' у 'High' і 'Low'
    data = data.rename(columns={'Max': 'High', 'Min': 'Low'})

    return data

# Застосування до кожного датасету
nzd_usd_data = preprocess_dataset(nzd_usd_data)

# Виведення перших кількох рядків кожного датасету для перевірки
print("NZD_USD:")
nzd_usd_data

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:33.117565Z","iopub.execute_input":"2023-12-17T05:11:33.118149Z","iopub.status.idle":"2023-12-17T05:11:33.200316Z","shell.execute_reply.started":"2023-12-17T05:11:33.118106Z","shell.execute_reply":"2023-12-17T05:11:33.199551Z"}}
import plotly.express as px

# Створення графіку Price з датами на вісі x
fig = px.line(nzd_usd_data, x='Date', y='Price', title='Графік ціни NZD/USD', labels={'Price': 'Ціна', 'Date': 'Дата'})

# Відображення графіку
fig.show()


# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:33.201432Z","iopub.execute_input":"2023-12-17T05:11:33.202434Z","iopub.status.idle":"2023-12-17T05:11:34.614581Z","shell.execute_reply.started":"2023-12-17T05:11:33.202400Z","shell.execute_reply":"2023-12-17T05:11:34.613689Z"}}
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose

# Функція для відображення тренду, сезонності та залишкової складової
def plot_seasonal_decomposition(data, currency_name):
    # Встановлення 'Date' як індексу
    data['Date'] = pd.to_datetime(data['Date'])  # Забезпечення, що 'Date' є в форматі дати

    # Визначення тренду, сезонності та залишкової складової
    result = seasonal_decompose(data['Price'], model='additive', period=30)

    # Відображення графіків
    plt.figure(figsize=(12, 8))
```

```
    plt.subplot(4, 1, 1)
    plt.plot(result.observed, label='Original')
    plt.legend(loc='upper left')
    plt.title(f'{currency_name} - Original')

    plt.subplot(4, 1, 2)
    plt.plot(result.trend, label='Trend')
    plt.legend(loc='upper left')
    plt.title(f'{currency_name} - Trend')

    plt.subplot(4, 1, 3)
    plt.plot(result.seasonal, label='Seasonal')
    plt.legend(loc='upper left')
    plt.title(f'{currency_name} - Seasonal')

    plt.subplot(4, 1, 4)
    plt.plot(result.resid, label='Residual')
    plt.legend(loc='upper left')
    plt.title(f'{currency_name} - Residual')

    plt.tight_layout()
    plt.show()

# Виклик функції для кожного датасету
plot_seasonal_decomposition(nzd_usd_data, 'NZD-USD')



# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:34.616988Z","iopub.execute_input":"2023-12-17T05:11:34.617344Z","iopub.status.idle":"2023-12-17T05:11:48.876286Z","shell.execute_reply.started":"2023-12-17T05:11:34.617311Z","shell.execute_reply":"2023-12-17T05:11:48.874593Z"}}
pip install -U scikit-fuzzy

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:48.878590Z","iopub.execute_input":"2023-12-17T05:11:48.878951Z","iopub.status.idle":"2023-12-17T05:11:48.887203Z","shell.execute_reply.started":"2023-12-17T05:11:48.878918Z","shell.execute_reply":"2023-12-17T05:11:48.886145Z"}}
import pandas as pd

# Якщо відсутні дані, можна використовувати fillna(0) або інше значення за замовчуванням

def welford_variance(data):
    n = 0  # кількість даних
```

```
    mean = 0.0  # середнє
    m2 = 0.0  # момент другого порядку (для обчислення дисперсії)

    for x in data:
        n += 1
        delta = x - mean
        mean += delta / n
        delta2 = x - mean
        m2 += delta * delta2

    if n < 2:
        return float('nan')

    variance = m2 / n
    return variance

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:48.889135Z","iopub.execute_input":"2023-12-17T05:11:48.889640Z","iopub.status.idle":"2023-12-17T05:11:49.496027Z","shell.execute_reply.started":"2023-12-17T05:11:48.889598Z","shell.execute_reply":"2023-12-17T05:11:49.494838Z"}}
import pandas as pd
import matplotlib.pyplot as plt

def plot_price_variance(data, date_column='Date', price_column='Price', title='Динаміка дисперсії цін з часом'):
    # Перетворення стовпця 'Date' в об'єкт datetime
    data[date_column] = pd.to_datetime(data[date_column])

    # Застосовуємо алгоритм Велфорда для обчислення дисперсії
    data['Price_Variance'] = data[price_column].expanding().apply(welford_variance)

    # Побудова графіку
    plt.figure(figsize=(10, 6))
    plt.plot(data[date_column], data['Price_Variance'], label='Дисперсія цін', color='blue')
    plt.title(title)
    plt.xlabel('Дата')
    plt.ylabel('Дисперсія')
    plt.legend()
    plt.grid(True)

    # Встановлення інтервалу дат
    date_interval = pd.date_range(start=data[date_column].min(), end=data[date_column].max(), freq='M')
    plt.xticks(date_interval, rotation=45)
    plt.xlim(data[date_column].min(), data[date_column].max())
```

```
    plt.show()

# Застосування функції до різних наборів даних
plot_price_variance(nzd_usd_data, title='Динаміка дисперсії цін для NZD/USD')

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:49.498035Z","iopub.execute_input":"2023-12-17T05:11:49.498496Z","iopub.status.idle":"2023-12-17T05:11:50.065337Z","shell.execute_reply.started":"2023-12-17T05:11:49.498452Z","shell.execute_reply":"2023-12-17T05:11:50.064183Z"}}
import pandas as pd
import matplotlib.pyplot as plt

def plot_price_and_averages(data, date_column='Date', price_column='Price', window_size=30, alpha=0.2, title='Ціна, Середнє та Експоненційне згладжування'):
    # Перетворення стовпця 'Date' в об'єкт datetime
    data[date_column] = pd.to_datetime(data[date_column])

    # Функція для обчислення середнього значення
    def moving_average(series, window_size):
        return series.rolling(window=window_size).mean()

    # Функція для обчислення експоненційного згладжування
    def exponential_smoothing(series, alpha):
        return series.ewm(alpha=alpha, adjust=False).mean()

    # Застосовуємо функції до стовпця "Price"
    data['MA_{}'.format(window_size)] = moving_average(data[price_column], window_size)
    data['EMA_{}'.format(alpha)] = exponential_smoothing(data[price_column], alpha)

    # Побудова графіку
    plt.figure(figsize=(12, 6))
    plt.plot(data[date_column], data[price_column], label='Ціна', color='blue')
    plt.plot(data[date_column], data['MA_{}'.format(window_size)], label='Середнє ({}-day)'.format(window_size), color='green')
    plt.plot(data[date_column], data['EMA_{}'.format(alpha)], label='Експоненційне (α={})'.format(alpha), color='orange')

    plt.title(title)
    plt.xlabel('Дата')
    plt.ylabel('Ціна')
    plt.legend()
    plt.grid(True)
```

```
    # Встановлення інтервалу дат
    date_interval = pd.date_range(start=data[date_column].min(), end=data[date_column].max(), freq='M')
    plt.xticks(date_interval, rotation=45)
    plt.xlim(data[date_column].min(), data[date_column].max())

    plt.show()

# Застосування функції до різних наборів даних
plot_price_and_averages(nzd_usd_data, title='Ціна, Середнє та Експоненційне згладжування для NZD/USD')



# %% [code]



# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:50.067037Z","iopub.execute_input":"2023-12-17T05:11:50.067428Z","iopub.status.idle":"2023-12-17T05:11:50.222229Z","shell.execute_reply.started":"2023-12-17T05:11:50.067357Z","shell.execute_reply":"2023-12-17T05:11:50.221060Z"}}
import pandas as pd
import numpy as np

# Завантаження даних
data = nzd_usd_data

# Визначення вектора зміни ціни
price_vector = data['Price'].diff()

# Аналіз взаємного розміщення двох ковзних середніх (Moving Average)
short_window = 10
long_window = 50

data['Short_MA'] = data['Price'].rolling(window=short_window, min_periods=1).mean()
data['Long_MA'] = data['Price'].rolling(window=long_window, min_periods=1).mean()

data['MA_Crossover'] = np.where(data['Short_MA'] > data['Long_MA'], 1, 0)

# Аналіз осцилятора "стохастика" (stochastic)
k_period = 14
data['Stochastic_Oscillator'] = (data['Price'] - data['Price'].rolling(window=k_period).min()) / (
        data['Price'].rolling(window=k_period).max() - data['Price'].rolling(window=k_period).min()) * 100

# Аналіз осцилятора "MACD"
```

```
short_window_macd = 12
long_window_macd = 26
signal_window_macd = 9

data['Short_EMA'] = data['Price'].ewm(span=short_window_macd, adjust=False).mean()
data['Long_EMA'] = data['Price'].ewm(span=long_window_macd, adjust=False).mean()

data['MACD'] = data['Short_EMA'] - data['Long_EMA']
data['Signal_Line'] = data['MACD'].ewm(span=signal_window_macd, adjust=False).mean()

# Створення вектора ефективності K_mod
weight_MA_Crossover = 0.3
weight_Stochastic_Oscillator = 0.3
weight_MACD = 0.2
weight_Signal_Line = 0.2

data['K_mod'] = (
    weight_MA_Crossover * data['MA_Crossover'] +
    weight_Stochastic_Oscillator * data['Stochastic_Oscillator'] +
    weight_MACD * (data['MACD'] - data['Signal_Line'])
)

# Зсув значень K_mod лише для рядків, де вони були розраховані
data['K_mod'] = data['K_mod'].shift(1)

# Визначення векторів S%K та S%D
n = 3  # порядок лінії %K
m = 3  # порядок лінії %D

data['S%K'] = data['Stochastic_Oscillator'].rolling(window=n, min_periods=1).mean()
data['S%D'] = data['S%K'].rolling(window=m, min_periods=1).mean()

# Визначення точок перетину
data['Intersection'] = np.where(data['S%K'] > data['S%D'], 1, -1)

# Визначення напрямку перетину для визначення тренду
data['Trend_Direction'] = data['Intersection'].diff()

# Визначення коефіцієнта ефективності kst
data['kst'] = 0  # Ініціалізація коефіцієнту ефективності

# Розрахунок коефіцієнта ефективності kst
```

```
for i in range(1, len(data)):
    if data.at[i, 'Intersection'] == 1:
        if data.at[i, 'S%K'] < 25:
            data.at[i, 'kst'] = 4
        elif 25 <= data.at[i, 'S%K'] < 50:
            data.at[i, 'kst'] = 1
    elif data.at[i, 'Intersection'] == -1:
        if data.at[i, 'S%K'] > 75:
            data.at[i, 'kst'] = -4
        elif 50 < data.at[i, 'S%K'] <= 75:
            data.at[i, 'kst'] = -1

    # Врахування напрямку тренду
    if data.at[i, 'Trend_Direction'] > 0:
        data.at[i, 'kst'] = max(0, data.at[i, 'kst'])
    elif data.at[i, 'Trend_Direction'] < 0:
        data.at[i, 'kst'] = min(0, data.at[i, 'kst'])

# Зсув значень kst лише для рядків, де вони були розраховані
data['kst'] = data['kst'].shift(1)

# Аналіз взаємного розміщення двох ліній осцилятора MACD
data['macd1'] = data['Short_EMA'] - data['Long_EMA']
data['macd2'] = data['Signal_Line']

# Визначення точок перетину ліній macd1 та macd2
data['MACD_Intersection'] = np.where(data['macd1'] > data['macd2'], 1, -1)

# Визначення напрямку перетину для визначення тренду
data['MACD_Trend_Direction'] = data['MACD_Intersection'].diff()

# Визначення коефіцієнта ефективності kmacd
data['kmacd'] = 0  # Ініціалізація коефіцієнту ефективності

# Розрахунок коефіцієнта ефективності kmacd
for i in range(1, len(data)):
    if data.at[i, 'MACD_Intersection'] == 1:
        if data.at[i, 'macd1'] < 0:
            data.at[i, 'kmacd'] = 3
    elif data.at[i, 'MACD_Intersection'] == -1:
        if data.at[i, 'macd1'] > 0:
            data.at[i, 'kmacd'] = -3
```

```
# Зсув значень kmacd лише для рядків, де вони були розраховані
data['kmacd'] = data['kmacd'].shift(1)

# Об'єднання векторів ефективності в загальний вектор K
data['K'] = np.nansum([data['K_mod'], data['kst'], data['kmacd']], axis=0)

# Визначення найсильнішого сигналу для прийняття рішення
data['Strongest_Signal'] = np.nanmax(data[['K_mod', 'kst', 'kmacd']], axis=1)

# Вивід результатів
print(data[['Date', 'Price', 'MA_Crossover', 'Stochastic_Oscillator', 'K_mod', 'S%K', 'S%D', 'kst', 'macd1', 'macd2', 'kmacd', 'K', 'Strongest_Signal']])



# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:50.224128Z","iopub.execute_input":"2023-12-17T05:11:50.225158Z","iopub.status.idle":"2023-12-17T05:11:50.271659Z","shell.execute_reply.started":"2023-12-17T05:11:50.225112Z","shell.execute_reply":"2023-12-17T05:11:50.270465Z"}}
nzd_usd_data

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:50.277325Z","iopub.execute_input":"2023-12-17T05:11:50.278557Z","iopub.status.idle":"2023-12-17T05:11:52.014136Z","shell.execute_reply.started":"2023-12-17T05:11:50.278509Z","shell.execute_reply":"2023-12-17T05:11:52.012979Z"}}
import matplotlib.pyplot as plt

# Графік ціни та ковзних середніх
plt.figure(figsize=(14, 4))
plt.plot(data['Date'], data['Price'], label='Ціна')
plt.plot(data['Date'], data['Short_MA'], label=f'Короткий MA ({short_window})')
plt.plot(data['Date'], data['Long_MA'], label=f'Довгий MA ({long_window})')
plt.title('Графік ціни та ковзних середніх')
plt.xlabel('Дата')
plt.ylabel('Значення')
plt.legend()
plt.show()

# Графік осцилятора "стохастика"
plt.figure(figsize=(14, 4))
plt.plot(data['Date'], data['Stochastic_Oscillator'], label='Стохастик', color='blue')
plt.title('Графік осцилятора "Стохастик"')
plt.xlabel('Дата')
```

```
plt.ylabel('Значення')
plt.legend()
plt.show()

# Графік осцилятора "MACD" та сигнальної лінії
plt.figure(figsize=(14, 4))
plt.plot(data['Date'], data['MACD'], label='MACD', color='blue')
plt.plot(data['Date'], data['Signal_Line'], label='Сигнальна лінія', color='orange')
plt.title('Графік осцилятора "MACD" та сигнальної лінії')
plt.xlabel('Дата')
plt.ylabel('Значення')
plt.legend()
plt.show()

# Графік вектора ефективності K_mod
plt.figure(figsize=(14, 4))
plt.plot(data['Date'], data['K'], label='K', color='orange')
plt.title('Графік вектора ефективності K_mod')
plt.xlabel('Дата')
plt.ylabel('Значення')
plt.legend()
plt.show()


# %% [markdown]
#

# %% [code]
# Визначення вектора спостережень
data['Observations'] = np.where(data['K_mod'] > 0, 1, 0)

# Визначення рішень на основі порівняння значень вектора спостережень для двох послідовних моментів часу
data['Decision'] = np.where(data['Observations'] > data['Observations'].shift(1), 'Покупка активів', 'Утримання або продаж активів')

# Виведення результатів
print(data[['Date', 'Price', 'K_mod', 'Observations', 'Decision']])


# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:52.038695Z","iopub.execute_input":"2023-12-17T05:11:52.039197Z","iopub.status.idle":"2023-12-17T05:11:52.047573Z","shell.execute_reply.started":"2023-12-17T05:11:52.039150Z","shell.execute_reply":"2023-12-17T05:11:52.046453Z"}}
```

```
data.columns

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:52.049180Z","iopub.execute_input":"2023-12-17T05:11:52.049642Z","iopub.status.idle":"2023-12-17T05:11:52.098386Z","shell.execute_reply.started":"2023-12-17T05:11:52.049600Z","shell.execute_reply":"2023-12-17T05:11:52.097126Z"}}
import numpy as np
import pandas as pd
import skfuzzy as fuzz

# data = ...

def create_observation_matrix(data):
    n = len(data)
    X = np.zeros((n, 3))

    max_K_mod = np.max(data['K_mod'])  # Знаходимо максимальне значення K_mod

    for i in range(n - 1):
        X[i, 0] = data['K_mod'].iloc[i]
        X[i, 1] = data['K_mod'].iloc[i]
        X[i, 2] = data['K_mod'].iloc[i + 1] - max_K_mod

    X[-1, 0] = data['K_mod'].iloc[-1]
    X[-1, 1] = data['K_mod'].iloc[-1]
    X[-1, 2] = data['K_mod'].iloc[-1]

    return X

# Виклик функції для створення матриці спостережень X
observation_matrix = create_observation_matrix(data)

observation_matrix[np.isnan(observation_matrix)] = 0

def perform_clustering(observation_matrix, c=2):
    cntr, u, _, _, _, _, _ = fuzz.cluster.cmeans(
        observation_matrix.T, c, 2, error=0.005, maxiter=1000)
    return cntr, u

# Кластеризація для покупок
c = 2
cntr_buy, u_buy = perform_clustering(observation_matrix, c)
F_buy = u_buy.T
```

```
F_sell = -F_buy  # Перетворення матриці -F

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:52.100217Z","iopub.execute_input":"2023-12-17T05:11:52.101553Z","iopub.status.idle":"2023-12-17T05:11:52.109579Z","shell.execute_reply.started":"2023-12-17T05:11:52.101511Z","shell.execute_reply":"2023-12-17T05:11:52.108352Z"}}
observation_matrix

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:52.111436Z","iopub.execute_input":"2023-12-17T05:11:52.111767Z","iopub.status.idle":"2023-12-17T05:11:52.449029Z","shell.execute_reply.started":"2023-12-17T05:11:52.111739Z","shell.execute_reply":"2023-12-17T05:11:52.447884Z"}}
cluster_labels = np.argmax(u_buy, axis=0)

# Візуалізація
plt.figure(figsize=(8, 6))

# Виведення кластерів
for cluster in range(c):
    cluster_points = observation_matrix[cluster_labels == cluster]
    plt.scatter([i for i in range(len(cluster_points))], cluster_points[:, 1], s=10, label=f'Кластер {cluster + 1}')

plt.title('Точковий графік та кластеризація')
plt.xlabel('Спостереження')
plt.ylabel('Значення')
plt.grid(True)

# Перемістимо легенду за графік
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.show()

# %% [code]


# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:52.450641Z","iopub.execute_input":"2023-12-17T05:11:52.451075Z","iopub.status.idle":"2023-12-17T05:11:52.795298Z","shell.execute_reply.started":"2023-12-17T05:11:52.451031Z","shell.execute_reply":"2023-12-17T05:11:52.794327Z"}}
import matplotlib.pyplot as plt
import numpy as np

# Відобразимо матрицю приналежності u_buy
plt.figure(figsize=(8, 6))
for i in range(c):
```

```
    plt.plot(u_buy[i], label=f'Кластер {i + 1}')

plt.title('Матриця приналежності до кластерів (u_buy)')
plt.xlabel('Спостереження')
plt.ylabel('Приналежність')
plt.legend()
plt.grid(True)
plt.show()

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:52.796745Z","iopub.execute_input":"2023-12-17T05:11:52.797687Z","iopub.status.idle":"2023-12-17T05:11:52.806743Z","shell.execute_reply.started":"2023-12-17T05:11:52.797640Z","shell.execute_reply":"2023-12-17T05:11:52.805469Z"}}
F_buy

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:52.808165Z","iopub.execute_input":"2023-12-17T05:11:52.808596Z","iopub.status.idle":"2023-12-17T05:11:52.820560Z","shell.execute_reply.started":"2023-12-17T05:11:52.808555Z","shell.execute_reply":"2023-12-17T05:11:52.819380Z"}}
F_sell

# %% [markdown]
#
#
#

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:52.822440Z","iopub.execute_input":"2023-12-17T05:11:52.822823Z","iopub.status.idle":"2023-12-17T05:11:52.844882Z","shell.execute_reply.started":"2023-12-17T05:11:52.822791Z","shell.execute_reply":"2023-12-17T05:11:52.843758Z"}}
# Параметри кластеризації
c_buy = 2
c_sell = 2

# Функція для створення вектора ефективності K_mod
def create_efficiency_vector(F_buy, F_sell):
    n = min(F_buy.shape[0], F_sell.shape[0])
    K_mod = np.zeros(n)

    for i in range(n):
        K_mod[i] = np.dot(F_buy[i], np.arange(c_buy)) - np.dot(F_sell[i], np.arange(c_sell))

    # Нормалізація вектора K_mod до проміжку [-1, 1]
    K_mod = 2 * (K_mod - np.min(K_mod)) / (np.max(K_mod) - np.min(K_mod)) - 1
```

```
    return K_mod

# Виклик функції для створення вектора ефективності K_mod
K_mod = create_efficiency_vector(F_buy, F_sell)

# Додавання вектора K_mod до датафрейму
data['K_mod'] = K_mod

# Вивід датафрейму з оновленим стовпцем K_mod
print(data[['Date', 'K_mod']])

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:52.846175Z","iopub.execute_input":"2023-12-17T05:11:52.846529Z","iopub.status.idle":"2023-12-17T05:11:52.854106Z","shell.execute_reply.started":"2023-12-17T05:11:52.846498Z","shell.execute_reply":"2023-12-17T05:11:52.853054Z"}}
data = data.sort_values(by='Date')

# %% [code] {"execution":{"iopub.status.busy":"2023-12-17T05:11:52.855710Z","iopub.execute_input":"2023-12-17T05:11:52.856317Z","iopub.status.idle":"2023-12-17T05:11:53.029657Z","shell.execute_reply.started":"2023-12-17T05:11:52.856282Z","shell.execute_reply":"2023-12-17T05:11:53.028446Z"}}
import pandas as pd
import plotly.graph_objects as go

# Додамо нові стовпці для сигналів покупки та продажу
data['Buy_Signal'] = ((data['Trend_Direction'].shift(1) <= 2) & (data['Trend_Direction'] > 0) & (data['K_mod'] > -0.5)).astype(int)
data['Sell_Signal'] = ((data['Trend_Direction'].shift(1) >= 0) & (data['Trend_Direction'] < 0) & (data['K_mod'] <= 0.5)).astype(int)

# Створення окремого датафрейму для збереження дат покупок та продаж
signals_list = []

# Ініціалізація балансу та створення порожнього датафрейму для угод
balance = 10000  # Початковий баланс
deals_columns = ['Покупка (дата)', 'Ціна покупки', 'Продаж (дата)', 'Ціна продажу', 'Прибуток']
deals_list = []
successful_deals = 0  # Лічильник успішних угод
unsuccessful_deals = 0  # Лічильник невдалих угод
amount_to_buy = 0  # Кількість валюти для купівлі
buy_signal_triggered = False  # Прапорець для відстеження виконання сигналу покупки

# Проходження по датасету та прийняття рішень
for i in range(1, len(data)):
```

```
    if data['Buy_Signal'].iloc[i] == 1 and not buy_signal_triggered:
        # Відкриття позиції на покупку
        purchase_date = data['Date'].iloc[i]
        purchase_price = data['Price'].iloc[i]
        amount_to_buy = 0.1 * balance / purchase_price  # Кількість валюти для купівлі (10% від балансу)
        balance -= 0.1 * balance  # Зменшення балансу на суму покупки
        buy_signal_triggered = True  # Встановлення прапорця, що покупка виконана
        signals_list.append({'Date': purchase_date, 'Signal': 'Buy'})  # Збереження сигналу покупки

    elif data['Sell_Signal'].iloc[i] == 1 and buy_signal_triggered:
        # Відкриття позиції на продаж
        sell_date = data['Date'].iloc[i]
        sell_price = data['Price'].iloc[i]
        if amount_to_buy > 0:
            profit = amount_to_buy * (sell_price - purchase_price)  # Обчислення прибутку від угоди
            balance += amount_to_buy * purchase_price  # Збільшення балансу на вартість покупки
            # Запис угоди у список
            deals_list.append([purchase_date, purchase_price, sell_date, sell_price, profit])
            if profit > 0:
                successful_deals += 1  # Збільшення лічильника успішних угод
            else:
                unsuccessful_deals += 1  # Збільшення лічильника невдалих угод
            amount_to_buy = 0  # Скидання кількості валюти для купівлі
            buy_signal_triggered = False  # Скидання прапорця, що покупка виконана
            signals_list.append({'Date': sell_date, 'Signal': 'Sell'})  # Збереження сигналу продажу

# Створення DataFrame зі списку угод
deals = pd.DataFrame(deals_list, columns=deals_columns)
# Створення DataFrame для сигналів
signals_df = pd.DataFrame(signals_list)

# Виведення графіка ціни з індикаторами покупки-продажу
fig = go.Figure()

fig.add_trace(go.Scatter(x=data['Date'], y=data['Price'], mode='lines', name='Ціна', line=dict(color='grey')))

# Сигнали покупки та продажу
for index, row in signals_df.iterrows():
    color = 'green' if row['Signal'] == 'Buy' else 'red'
    marker_size = 8  # Задайте бажаний розмір маркерів
    fig.add_trace(go.Scatter(x=[row['Date']], y=[data[data['Date'] == row['Date']]['Price'].iloc[0]],
                             mode='markers', name=row['Signal'], marker=dict(color=color, size=marker_size)))
```

```
fig.update_layout(title='Графік ціни з індикаторами покупки-продажу', xaxis_title='Дата', yaxis_title='Ціна')
fig.show()
balance=balance+1000
# Виведення угод та загального прибутку

deals_sorted = deals.sort_values(by=['Покупка (дата)'], ignore_index=True)
deals_sorted['Різниця цін'] = deals_sorted['Ціна продажу'] - deals_sorted['Ціна покупки']

# Виведення датафрейму угод з різницею цін
print("\nДеталі угод з різницею цін:")
print(deals_sorted[['Покупка (дата)', 'Ціна покупки', 'Продаж (дата)', 'Ціна продажу', 'Прибуток', 'Різниця цін']].to_string(index=False))

# Виведення початкового балансу
print(f'Початковий баланс: {balance}')

# Виведення загальної кількості успішних та невдалих угод
print(f'Успішних угод: {successful_deals}, Невдалих угод: {unsuccessful_deals}')

# Розрахунок балансу після всіх угод
final_balance = balance + deals_sorted['Прибуток'].sum()
print(f'Баланс після всіх угод: {final_balance}')



# %% [markdown]
# ###

# %% [code]



# %% [code]
```

Додаток Г (обов'язковий)

Протокол перевірки на плагіат

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Розробка захищеного сховища даних із використанням технологій блокчейн

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ: кафедра Автоматизації та інтелектуальних інформаційних технологій, факультет інтелектуальних інформаційних технологій та автоматизації
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 99.5% Схожість 0.5%

Аналіз звіту подібності (відмітити потрібне):

☑ 1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

☐ 2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

☐ 3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Роман МАСЛІЙ
(підпис) (ім'я, ПРІЗВИЩЕ)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Радислав ГРІНЧАК
(підпис) (ім'я, ПРІЗВИЩЕ)

Керівник роботи _____ Владислав КАБАЧІЙ
(підпис) (ім'я, ПРІЗВИЩЕ)