КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра інтелектуальних технологій

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття освітнього ступеня «магістр»

НА ТЕМУ:

Система прогнозування ціни на криптовалюту методами машинного навчання

Галузь знань: 12 «Інформаційні технології»
Спеціальність: 122 «Комп'ютерні науки»
Освітньо-наукова програма «Технології штучного інтелекту»

Виконав:
Студент 2 курсу магістратури, групи ТШІ-21

Рискальчук Д.В.
(ПІБ)

Науковий керівник: Сорока П.М.
(ПІБ)

кандидат фізико-математичних наук, доцент
(науковий ступінь, вчене звання)

Засвідчую, що в цій кваліфікаційній роботі немає запозичень з праць інших авторів без відповідних посилань

Студент(ка) _____
підпис

Кваліфікаційна робота допущена до захисту
рішенням кафедри *інтелектуальних технологій*

Протокол № 12 від «11» травня 2023 р.

Зав. кафедри _____ доц. Іларіонов О.Є.
підпис

Київ - 2023

РЕФЕРАТ

Кваліфікаційна робота складається зі вступу, 3 розділів, висновків, списку використаної літератури із 45 джерел та 1 додатка. Загальний обсяг роботи 92 сторінок. Робота містить 3 таблиці та 41 рисунок.

Актуальність теми. Тема прогнозування ціни на криптовалюту методами машинного навчання є дуже актуальною в сучасному світі, де криптовалюти стають все більш популярними і відомими серед інвесторів та звичайних користувачів. Ціна на криптовалюти є мінливою та залежить від багатьох факторів, таких як попит та пропозиція, новини в галузі криптовалют, геополітична ситуація тощо. Таким чином, прогнозування ціни на криптовалюту є складною задачею, яка може бути розв’язана за допомогою методів машинного навчання. Використання таких методів може допомогти інвесторам та трейдерам приймати обґрунтовані рішення щодо купівлі та продажу криптовалют, а також зменшити ризики вкладень. Отже, актуальність теми прогнозування ціни на криптовалюту методами машинного навчання полягає у важливості цієї проблеми в сучасному світі та можливостях застосування машинного навчання для розв'язання цієї задачі.

Мета кваліфікаційної роботи: Метою роботи є дослідження та аналіз методів машинного навчання для прогнозування ціни на криптовалюту, а також розробка та валідація моделі прогнозування ціни на криптовалюту на основі методів машинного навчання.

Об’єкт дослідження – процес прогнозування ціни на криптовалюту методами машинного навчання.

Предмет дослідження – методи машинного навчання для прогнозування ціни на криптовалюту.

Результати роботи. У кваліфікаційній роботі проведено аналіз сучасних методів машинного навчання для прогнозування ціни на криптовалюту, здійснено порівняння ефективності різних методів для

прогнозування ціни на криптовалюту, розроблено систему прогнозування ціни на криптовалюту.

Ключові слова: Прогнозування ціни, криптовалюта, машинне навчання, ARIMA, LSTM, моделювання, технічний аналіз, фундаментальний аналіз, інвестування, ризик.

ABSTRACT

Qualification work consists of an introduction, 3 sections, conclusions, a list of used literature from 45 sources and 1 annex. The total volume of work is 92 pages. The work contains 3 tables and 41 drawings.

Actuality of theme. The topic of pricing for cryptocurrency by machine learning is very relevant in the modern world, where cryptocurrencies are becoming more and more popular and known among investors and ordinary users. The price of cryptocurrencies is very variable and depends on many factors, such as supply and demand, cryptocurrency news, geopolitical situation, etc. Thus, the forecasting of cryptocurrency prices is a difficult task that can be solved by machine learning methods. The use of such methods can help investors and traders make reasonable decisions on the purchase and sale of cryptocurrencies, as well as reduce the risks of investments. Therefore, the relevance of the topic of pricing for cryptocurrency by machine learning is the importance of this problem in the modern world and the ability to use machine learning to solve this problem.

The purpose of the qualification work: The purpose of the work is to study and analyze the methods of machine learning to forecast the price of cryptocurrency, as well as the development and validation of the model of pricing for cryptocurrency based on machine learning methods.

The object of the research is the process of forecasting the price on cryptocurrency by machine training.

The subject of the research is the methods of machine learning to forecast the price of cryptocurrency.

Results. In the qualification work, an analysis of modern machine learning methods for forecasting cryptocurrency prices was carried out, a comparison of the effectiveness of various methods for forecasting cryptocurrency prices was carried out, and a cryptocurrency price forecasting system was developed.

Keywords: price forecasting, cryptocurrency, machine learning, arima, LSTM, modeling, technical analysis, fundamental analysis, investment, risk.

ЗМІСТ

ВСТУП

В сучасному світі криптовалюти стають все більш популярними і відіграють значну роль у глобальній економіці. Цінність криптовалют залежить від багатьох факторів, таких як політичні, соціальні, економічні та технічні аспекти. Прогнозування цін на криптовалютні активи є складною задачею, оскільки ціни залежать від багатьох факторів та можуть бути дуже непередбачуваними.

Кожна криптовалюта побудована на технології блокчейн і не має центрального органу управління. Транзакції перевіряються мережею учасників, і кожна транзакція завершується разом з іншими для формування блоку, який утворює безперервний ланцюжок. Неможливо змінити інформацію в одному блоці, не вносячи змін у всі наступні, тому підробити або скасувати запис неможливо або дуже дорого. Криптовалюти використовуються анонімно, але ідентифікацію користувачів можна встановити за наявності додаткової інформації. Для отримання криптовалют використовуються різні онлайн-методи, такі як майнінг, підробка, ICO. Майнінг і форгінг спрямовані на побудову блокчейну, а ICO – це спосіб залучення фінансування через продаж партій нової криптовалюти, які були згенеровані та отримані організатором.

Швидке зростання ринків криптовалюти в останні роки привернуло увагу інвесторів, трейдерів і дослідників до прогнозування цін на ці цифрові активи. Точне прогнозування ціни є складним завданням через високу волатильність ринків криптовалют і складну взаємодію різних факторів, які впливають на їхні ціни. Методи машинного навчання показали багатообіцяючі результати у прогнозуванні цін на криптовалюту шляхом аналізу історичних даних і виявлення тенденцій і закономірностей. У цій дипломній роботі ми прагнемо вивчити та застосувати різні методи машинного навчання для прогнозування цін на криптовалюти, зосереджуючись на Bitcoin. Ми проаналізуємо продуктивність різних моделей, включаючи регресійні моделі, нейронні мережі та ансамблеві методи. Мета роботи - оцінити ефективність

методів машинного навчання для прогнозування цін на криптовалюту та розробити точні і надійні моделі прогнозування для ринку криптовалют.

Однак важливо пам’ятати, що методи машинного навчання (ММН) - це лише інструмент, а не гарантія успіху. У інвестора повинно бути розуміння фінансового ринку та вміння аналізувати інші чинники, такі як політична ситуація, ризик та конкуренція на ринку.

РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМИ ВИКОРИСТАННЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ ДЛЯ ПРОГНОЗУВАННЯ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Огляд методів застосування машинного навчання в прогнозуванні

Прогнозування цін на криптовалюту є актуальною темою, оскільки це може допомогти інвесторам та трейдерам знайти кращі рішення при купівлі та продажу криптовалют. У світі криптовалют існує багато методів та підходів до прогнозування цін, які використовують методи машинного навчання.

Огляд методів застосування машинного навчання в прогнозуванні цін на криптовалюту є важливим етапом дослідження даної теми. Машинне навчання може бути використане для прогнозування цін на криптовалюти через використання різноманітних моделей, таких як нейронні мережі, дерева рішень та регресійні моделі.

Один з популярних методів машинного навчання для прогнозування цін на криптовалюти - це аналіз часових рядів. Застосування моделей аналізу часових рядів, таких як ARIMA, може допомогти в прогнозуванні майбутньої ціни на криптовалюту на основі її історичних даних[1].

Ще одним методом машинного навчання, що використовується для прогнозування цін на криптовалюти, є нейронні мережі. Нейронні мережі здатні адаптуватися до зміни умов на ринку та враховувати велику кількість факторів, що можуть вплинути на ціну криптовалюти, такі як новини, події та соціальні мережі. Для цього використовуються різноманітні архітектури нейронних мереж, такі як рекурентні нейронні мережі (RNN), згорткові нейронні мережі (CNN), а також комбінації цих архітектур [4].

Дерева рішень також можуть бути використані для прогнозування цін на криптовалюти. Вони можуть розглядати багато різних факторів, що впливають на ціну криптовалюти та допомагають встановити взаємозв’язки між ними.

Крім того, існує також підхід, який використовує аналіз тексту та соціальних мереж для прогнозування цін на криптовалюту. Для цього

використовуються методи обробки природньої мови та аналізу емоцій, які дозволяють аналізувати новини, соціальні мережі та інші джерела інформації для визначення настроїв ринку та передбачення можливих змін цін на криптовалюти.

Крім того, ММН можна використовувати для розробки автоматизованих торгових систем, які можуть здійснювати операції на основі заздалегідь визначених правил і критеріїв. Ці системи можна використовувати для торгівлі акціями, облігаціями, валютами та іншими фінансовими інструментами, і їх можна запрограмувати на реагування на зміну ринкових умов у режимі реального часу.

Однією з переваг використання ММН в прогнозуванні цін є те, що вони можуть допомогти інвесторам приймати більш обґрунтовані рішення на основі даних і аналізу, а не покладатися на інтуїцію чи емоції. ММН також можуть допомогти інвесторам визначити можливості та ризики, які можуть бути неочевидними за допомогою традиційних методів аналізу [3].

Однак важливо зазначити, що ММН не є безпомилковими та пов’язані з ризиком. Наприклад, алгоритми машинного навчання іноді можуть помилятися або створювати неточні прогнози, якщо їх не навчено належним чином або якщо дані, які вони аналізують, упереджені чи неповні.

Отже, методи машинного навчання стають дедалі важливішим інструментом для інвесторів, які прагнуть приймати більш обґрунтовані інвестиційні рішення. Використовуючи ці методи для аналізу даних і прогнозування, інвестори можуть отримати уявлення про ринкові тенденції та визначити можливості та ризики, які можуть бути не відразу очевидними за допомогою традиційних методів аналізу. Однак важливо підходити до машинного навчання з обережністю та переконатися, що використовувані алгоритми належним чином навчені та перевірені, щоб мінімізувати ризики, пов’язані з їх використанням.

У цьому дослідженні будуть розглянуті та порівняні різні методи машинного навчання для прогнозування цін на криптовалюти, щоб встановити

ефективність кожного методу та знайти оптимальний метод для даної задачі прогнозування цін.

1.2 Огляд існуючих публікацій в галузі використання методів машинного навчання в прогнозуванні

В останні роки було опубліковано значну кількість наукових статей про використання методів машинного навчання в прогнозуванні цін на криптовалюти. Деякі з них описують використання класичних методів машинного навчання, таких як регресія та класифікація, для прогнозування цін на акції та інші інвестиційні інструменти.

Наприклад, стаття "Predicting Stock Prices Using Support Vector Regression" (Jain et al., 2015) описує використання методу опорних векторів (SVM) для прогнозування цін на акції. Дослідження показало, що метод SVM може давати точні прогнози цін на акції та інші інвестиційні інструменти [5].

В статтях описується використання глибинного навчання для аналізу фінансових даних та прогнозування цін на акції. Наприклад, стаття "Deep Learning for Stock Prediction: A Comparative Study" (Xiong et al., 2017) порівнює різні архітектури нейронних мереж для прогнозування цін на акції та інших інвестиційних інструментів.

"Bitcoin Price Prediction Using Machine Learning: An Approach to Sample Dimension Engineering" є науковою статтею, опублікованою в журналі "IEEE Access" в 2020 році. Авторами є Olusola Akinfenwa, Jonathan Osamor, та Chibueze Ogbonna. У ній автори досліджують прогнозування ціни Bitcoin з використанням методів машинного навчання. Вони зосереджуються на важливості інженерії даних і використовують різні методи машинного навчання, такі як регресійні моделі, нейронні мережі та машини опорних векторів (SVM) для побудови моделей прогнозування [1].

Основна ідея статті полягає в тому, що даних про криптовалюти, як правило, дуже мало, тому автори пропонують новий метод, який називається "Sample Dimension Engineering" для створення додаткових даних, які можна

використовувати для навчання моделей. Цей метод включає в себе генерацію додаткових даних шляхом комбінування наявних даних.

Дослідження показали, що нейронні мережі та SVM-моделі дають найкращі результати прогнозування цін Bitcoin, порівняно з регресійними моделями. Крім того, результати також показали, що використання методу "Sample Dimension Engineering" підвищує точність прогнозів. Однак, наразі є певні обмеження в застосуванні ММН в інвестуванні, особливо в тому, що вони не забезпечують 100% точності. Більшість статей зосереджуються на покращенні результатів тих методів, які використовуються в інвестуванні з метою досягнення більш високої точності та ефективності.

Стаття "Forecasting Cryptocurrency Prices with Machine Learning Techniques" була опублікована в 2018 році в журналі "IEEE Conference on Computational Intelligence for Financial Engineering and Economics" і розглядає застосування методів машинного навчання для прогнозування цін на криптовалюти.

У статті автори порівнюють ефективність кількох алгоритмів машинного навчання, таких як нейронні мережі, Random Forest і Support Vector Machines, для прогнозування цін на три популярні криптовалюти: Bitcoin, Ethereum і Litecoin. Для цього вони використовували історичні дані про ціни та інші фактори, такі як обсяг торгів, кількість транзакцій і кількість унікальних адрес [2].

Дослідження показало, що нейронні мережі є найефективнішими для прогнозування цін на криптовалюти, з точністю прогнозування понад 80% для Bitcoin і Ethereum, тоді як точність Random Forest і Support Vector Machines була нижчою.

Автори статті також розглянули важливість відбору ознак для покращення точності прогнозування. Вони зазначили, що деякі ознаки, такі як кількість транзакцій та кількість унікальних адрес, мають найбільший вплив на ціни криптовалют, тоді як інші ознаки, такі як обсяг торгів, не мають такого великого впливу.

У загальному, стаття підтверджує можливість використання ММН для прогнозування цін на криптовалюти, а також вказує на важливість відбору відповідних ознак для покращення точності прогнозування.

Стаття "Bitcoin Price Prediction Using LSTM Neural Network" є дослідженням, що досліджує можливість прогнозування цін на Bitcoin за допомогою рекурентної нейронної мережі Long Short-Term Memory (LSTM). Автори статті використали дані про ціну Bitcoin з 2013 року до 2018 року для тренування та тестування моделі [3].

У статті автори описують попередні дослідження з використанням LSTM для прогнозування цін на фінансові активи, а також надають огляд літератури з використанням машинного навчання для прогнозування цін на криптовалюти. Далі, автори детально описують методику побудови та тренування LSTM-моделі для прогнозування цін на Bitcoin.

Автори використовували різні параметри та гіперпараметри моделі для забезпечення оптимальної точності прогнозування. Також вони порівнювали прогнози моделі з фактичними цінами на Bitcoin, щоб оцінити її точність.

Результати дослідження показали, що LSTM-модель може бути ефективним інструментом для прогнозування цін на Bitcoin. Автори зазначили, що точність прогнозів може залежати від обсягу даних та гіперпараметрів моделі. Однак, вони вважають, що цей дослід є важливим кроком у напрямку розвитку машинного навчання для прогнозування цін на криптовалюти.

Стаття "Forecasting Bitcoin Price Fluctuation with Twitter Sentiment Analysis" розглядає можливість використання аналізу настроїв у Twitter для прогнозування ціни Bitcoin. Автори статті використали даний метод для побудови моделі, яка прогнозує зміни ціни Bitcoin на добу вперед.

Дослідники використовували історичні дані цін на Bitcoin та Twitter-дані, щоб створити модель машинного навчання, яка базується на аналізі настроїв. Зокрема, вони використовували Long Short-Term Memory (LSTM) для аналізу настроїв у Twitter та прогнозування ціни Bitcoin на наступний день.

Для аналізу настроїв автори використовували бібліотеку TextBlob, яка забезпечує функції аналізу настроїв на основі текстових даних. Загалом, дослідники обробили понад 10 тисяч твітів, пов’язаних з Bitcoin, та використали їх для аналізу настроїв.

Результати дослідження показали, що модель на основі LSTM та аналізу настроїв у Twitter може давати точні прогнози зміни ціни Bitcoin на добу вперед. Автори також порівняли результати своєї моделі з результатами інших методів прогнозування цін на Bitcoin, таких як ARIMA та GARCH, і показали, що їхня модель дає кращі результати [4].

Отже, стаття "Forecasting Bitcoin Price Fluctuation with Twitter Sentiment Analysis" демонструє потенційну ефективність використання аналізу настроїв у Twitter для прогнозування ціни Bitcoin та може бути корисною для тих, хто цікавиться ринком криптовалют та прогнозуванням їх цін.

Стаття "Predicting Cryptocurrency Prices with Deep Learning" (2018) описує використання глибокого навчання для прогнозування цін на криптовалюту. Автори стверджують, що цей підхід може дати більш точні результати, ніж традиційні методи аналізу фундаментальних та технічних показників.

У статті автори використовують рекурентні нейронні мережі (RNN), зокрема, довготривалу пам’ять (LSTM) для прогнозування цін на Bitcoin, Ethereum та Litecoin. Вони також використовують дані з додатку Coinbase для тренування та тестування моделі.

Автори стверджують, що їхня модель може успішно прогнозувати ціни на криптовалюту на основі історичних даних, а також враховувати важливі події, такі як новини про криптовалюту. Вони також використовують техніку передового заповнення, щоб заповнити пропуски в даних, що може поліпшити точність прогнозів [5].

У підсумку, стаття "Predicting Cryptocurrency Prices with Deep Learning" пропонує новий підхід до прогнозування цін на криптовалюту за допомогою глибокого навчання, який може бути корисним для інвесторів та трейдерів.

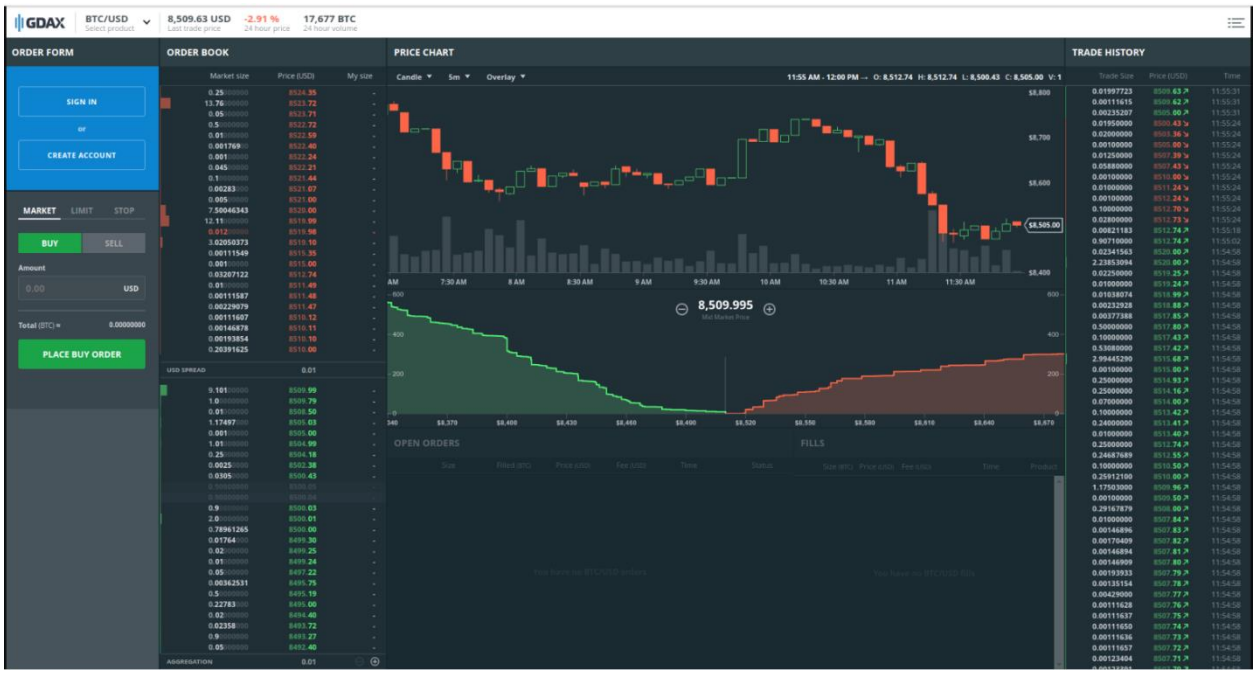

Рисунок 1.1 – Біржа для торгівлі криптовалютою

Всі описані статті досліджують використання методів машинного навчання для прогнозування цін на криптовалюти. Різні автори використовують різні моделі та методи, такі як LSTM-нейронні мережі, згорткові нейронні мережі та аналіз настроїв у Twitter, щоб створити точні прогнози.

Виявлено, що використання нейронних мереж, зокрема LSTM-мереж, дозволяє досягти високої точності прогнозування цін на криптовалюти. Крім того, деякі статті досліджують використання інших параметрів, таких як обсяг торгів та кількість транзакцій, які можуть допомогти покращити прогнозування.

Загалом, всі ці статті демонструють, що використання методів машинного навчання може бути корисним для прогнозування цін на криптовалюти. Однак, слід пам’ятати, що ринок криптовалют досить волатильний та піддається впливу багатьох чинників, тому прогнозування цін може бути складним завданням.

1.3 Функціональні та нефункціональні вимоги, опис зацікавлених сторін

Функціональні вимоги до системи прогнозування цін на криптовалюту методами машинного навчання:

1. *Збір та обробка даних:* Система повинна мати можливість збирати дані про ціни на криптовалюту з різних джерел і обробляти їх для подальшого аналізу.
2. *Аналіз даних*: Система повинна проводити аналіз зібраних даних з використанням методів машинного навчання для виявлення кореляцій та патернів, що впливають на ціни криптовалют.
3. *Побудова моделей:* Система повинна створювати моделі прогнозування цін на криптовалюти з використанням різних методів машинного навчання, таких як лінійна регресія, дерева рішень, нейронні мережі тощо.
4. *Прогнозування цін:* Система повинна здійснювати прогнозування майбутніх цін на криптовалюту на основі побудованих моделей.
5. *Візуалізація результатів:* Система повинна надавати зручний спосіб візуалізації прогнозованих цін на криптовалюту, наприклад, у вигляді графіків або діаграм.

Нефункціональні вимоги до системи:

1. *Точність:* Система повинна забезпечувати високу точність прогнозів цін на криптовалюту, щоб користувачі могли робити обґрунтовані рішення.
2. *Надійність:* система повинна забезпечувати достовірність та точність прогнозів, щоб забезпечити користувачам надійні дані для прийняття рішень.
3. *Швидкодія:* система повинна працювати швидко та ефективно, щоб забезпечити оперативність прийняття рішень.
4. *Масштабованість:* система повинна бути готовою до обробки великої кількості даних та забезпечувати коректну роботу незалежно від об’єму даних.

5. *Зручність використання:* система повинна бути простою та зручною в користуванні, щоб користувачі могли швидко зрозуміти, як її використовувати.
6. *Безпека:* система повинна забезпечувати високий рівень безпеки даних та захист від несанкціонованого доступу до них.
7. *Сумісність:* система повинна бути сумісною з різними типами пристроїв та операційними системами.
8. *Моніторинг та управління:* система повинна мати можливість моніторингу та управління її роботою, щоб забезпечити її ефективну та надійну роботу.
9. *Стійкість:* система повинна бути стійкою до помилок та збоїв, щоб забезпечити надійність та точність прогнозів незалежно від зовнішніх умов.
10. *Розширюваність:* система повинна бути легко розширюваною та модифікованою, щоб додавати нові функції та можливості при необхідності.

Система прогнозування цін на криптовалюту методами машинного навчання може мати наступних зацікавлених сторін, або стейкхолдерів:

- *Користувачі*: Інвестори, трейдери та інші особи, які цікавляться прогнозуванням цін на криптовалюту для прийняття рішень про купівлю, продаж або управління своїми активами.
- *Фінансові установи*: Банки, фондові біржі та інші фінансові установи, які можуть використовувати прогнози цін на криптовалюту для аналізу ринку, розробки стратегій і прийняття рішень з управління активами.
- *Дослідники*: Академічні дослідники, науковці та експерти, які цікавляться аналізом та вдосконаленням методів прогнозування цін на криптовалюту.

- *Розробники та аналітики*: Програмісти, інженери та аналітики, які розробляють та підтримують систему прогнозування цін на криптовалюту, розробляють алгоритми, збирають та обробляють дані, виконують тестування та оптимізують систему.
- *Регуляторні органи*: Органи регулювання та нагляду, які можуть зацікавлені в аналізі прогнозів цін на криптовалюти для визначення політик та регулювання криптовалютного ринку.
- *Постачальники даних*: Організації та платформи, що забезпечують доступ до фінансових даних, історичних даних про ціни на криптовалюти, новин та соціальних медіа, які є джерелами для аналізу та прогнозування цін.

Усі ці стейкхолдери можуть внести свої вимоги та очікування щодо системи прогнозування цін на криптовалюту методами машинного навчання, що може бути важливим для розробників у процесі створення та вдосконалення системи.

1.4 Постановка задачі

Майже в усіх розглянутих вище роботах та системах використовуються різні методи для побудови систем прогнозування. У деяких робиться розгляд впливу різних вхідних даних на результати роботи системи, проте не розглядається порівняння моделей на основі різних методів машинного навчання.

Метою роботи є дослідження та аналіз різних методів машинного навчання для прогнозування ціни на криптовалюту, а також розробка та валідація моделі прогнозування ціни на криптовалюту на основі методів машинного навчання.

Об'єктом дослідження є процес прогнозування ціни на криптовалюту методами машинного навчання.

Предметом дослідження є методи машинного навчання для прогнозування ціни на криптовалюту.

Завданнями дослідження є:

- провести аналіз наявних літературних та інших джерел, зокрема мережі інтернет, щодо прогнозування цін на криптовалюту, зокрема Bitcoin;
- розглянути та описати різні підходи та методології машинного навчання для створення системи прогнозування ціни на криптовалюти;
- обрати вхідні дані для навчання моделей, провести їх аналіз та описати додаткові параметри для їх покращення;
- описати функції системи, розробити архітектуру;
- провести навчання моделей, оцінку точності кожної з моделей;
- розробити програмний додаток прогнозування цін на криптовалюту Bitcoin на основі методів машинного навчання;
- виконати аналіз та опис отриманих результатів.

РОЗДІЛ 2 ОПИС ТА АНАЛІЗ МЕТОДІВ МАШИННОГО НАВЧАННЯ, КРИПТО-ВАЛЮТНОГО РИНКУ, ВХІДНИХ ДАНИХ ТА АРХІТЕКТУРИ МОДЕЛЕЙ

2.1 Майнінг криптовалюти

Майнінг біткоїнів є процесом, який не тільки вводить нові біткоїни в обіг, але й є ключовим елементом для підтримки та розвитку реєстру блокчейн. Цей процес виконується за допомогою складного апаратного забезпечення, яке розв’язує надзвичайно складну обчислювальну математичну задачу. Комп’ютер, який першим знайде рішення проблеми, отримує наступний блок біткоїнів, і процес починається знову.

Хоча майнінг криптовалюти - це важка і дорога діяльність, яка нерегулярно приносить прибуток, він все ще має привабливість для багатьох інвесторів, які зацікавлені в криптовалютах, через те, що майнери отримують криптотокени як винагороду за свою роботу. Можливо, це через те, що підприємці розглядають видобуток як золоту жилу, схожу на ту, яку знайшли золотошукачі в Каліфорнії в 1849 році [6].

Хоча можливість отримати біткойни відноситься до головних переваг майнінгу для багатьох людей, володіння криптовалютою не обов'язково потребує стати майнером. Крім того, є кілька інших способів, якими можна отримати криптовалюту, таких як купівля за фіатну валюту, торгівля на біржах з використанням інших криптовалют, отримання винагород за публікації в блозі на платформах, що платять криптовалютою, або створення крипторахунків зі сплатою відсотків.

Steemit - це платформа для крипто-блогів, яка подібна до Medium, окрім того факту, що клієнти мають можливість перекидати гроші відеоблогерам, платячи їм у STEEM - внутрішній криптовалюті, яку можна обміняти на біткойни на інших платформах.

Майнери отримують винагороду у біткойнах, яка стимулює їх допомагати в головній меті майнінгу: легітимізувати та контролювати транзакції біткойн, перевіряючи їх на достовірність. Біткойн є

"децентралізованою" криптовалютою, оскільки ці обов'язки розподілені між багатьма користувачами по всьому світу, що означає, що вона не покладається на центральний орган влади, такий як центральний банк чи уряд, для контролю за своїм регулюванням [7].

Крім забезпечення прибутку майнерам та підтримки екосистеми біткойн, майнінг виконує ще одну важливу функцію: випуск нових криптовалют в обіг. Тобто, майнери в основному створюють нову валюту. Наприклад, на вересень 2021 року в обігу було приблизно 18,82 мільйона біткойнів з загальної кількості в 21 мільйон [9].

Крім монет, що були згенеровані з блоку genesis, який був створений Сатоші Накамото, кожен з біткойнів був згенерований завдяки майнерам. Біткойн все ще існував би як мережа без майнерів, але нові біткоїни не випускалися б. Хоча транзакції біткойнів продовжують перевірятися, винагорода за майнінг біткойнів зменшується вдвічі приблизно кожні чотири роки [34]. Майнери отримують винагороду за перевірку транзакцій, щоб забезпечити безпеку та незмінність мережі біткойн. Від початку майнінгу в 2009 році винагорода зменшувалася з 50 BTC до поточних 6,25 BTC, і це триватиме до тих пір, поки остаточна кількість біткойнів, яку можна згенерувати, не буде досягнута близько 2140 року (рис. 2.1).

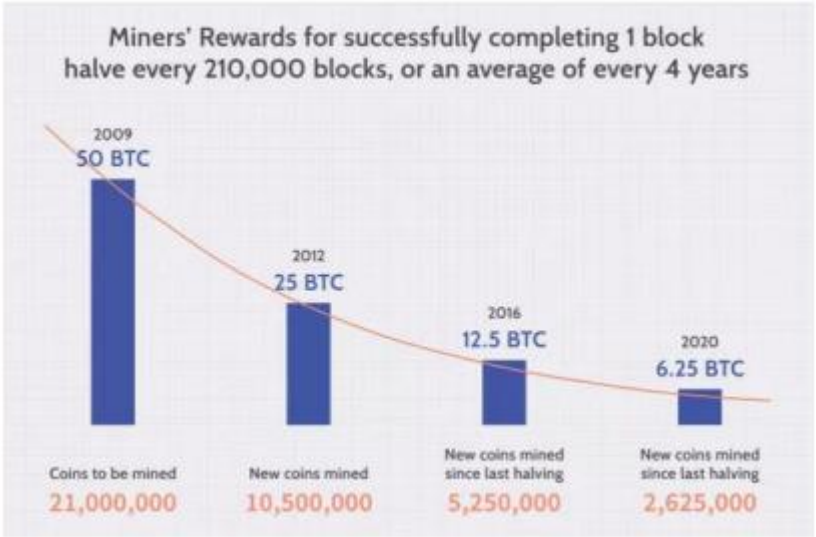

Рисунок 2.1 – Зміни доходів від майнингу [38]

Якщо ви хочете знати, коли саме відбудеться наступне уполовинення, то вам можна звернутися до Bitcoin Clock, який надає оновлення цієї інформації в режимі реального часу. Цікаво, що історично ринкова ціна біткойна майже завжди відображала уполовинення кількості нових монет, які вводяться в обіг. Це зменшення рівня інфляції збільшує дефіцит та історично викликає зростання ціни.

Навіть якщо на початку історії біткоїну люди могли змагатися за блоки, використовуючи свої домашні комп'ютери, зараз це неможливо. Причина полягає у тому, що складність видобутку біткоїну змінюється з часом. Щоб забезпечити безперебійну роботу блокчейну та його можливості приймати та коригувати транзакції, мережа біткоїну намагається створювати один блок кожні 10 хвилин або навколо того. Проте, якщо мільйон майнінгових установок беруть участь у змаганні за вирішення проблеми хешування, вони швидше знайдуть рішення, ніж у сценарії, де 10 майнінгових установок працюють над однією проблемою. Тому біткоїн регулює складність майнінгу кожні 2016 блоків або раз на кожні два тижні[8].

Коли кількість обчислювальної потужності, що використовується для майнінгу біткойнів, зростає, рівень складності майнінгу також зростає, щоб забезпечити стабільну швидкість виробництва блоків. Зменшення кількості обчислювальної потужності призведе до зниження складності майнінгу. З огляду на поточний розмір мережі, майнінг біткойнів з допомогою персонального комп'ютера на сьогоднішній день майже неможливий.

Отже, тепер для того, щоб бути успішним майнером, необхідно мати потужне комп'ютерне обладнання, таке як блок графічного процесора або інтегральну схему для конкретного застосування (ASIC), яка може коштувати від 500 доларів до десятків тисяч. Деякі майнери використовують відеокарти (GPU) Ethereum як недорогий спосіб поєднати операції з майнінгу.

Майнінгова винагорода призначається тому майнеру, який першим вирішує головоломку, і ймовірність вирішення дорівнює частці загальної потужності майнінгу в мережі. Майнери з низьким відсотком майнінгової

потужності мають дуже малу ймовірність самотужки відшукати наступний блок. Наприклад, майнінгова карта, придбана за кілька тисяч доларів, може представляти менше ніж 0,001% загальної майнінгової потужності в мережі. Це означає, що майнер може витратити багато часу, не знайшовши наступний блок, і труднощі підйому ще більше погіршать ситуацію. Він може навіть не отримати повернення своїх інвестицій. Майнінгові пули створені для вирішення цієї проблеми.

Майнінг-пули - це групи майнерів, які управляються третіми сторонами та координують свої зусилля для вирішення головоломки та отримання винагороди за майнінг. Приєднуючись до майнінг-пулів, майнери можуть отримувати стабільний потік біткойнів, починаючи з дня активації своїх майнерів.

Майнінг може бути пов’язаний з ризиками фінансового та регуляторного характеру і ці ризики можуть бути зменшені шляхом приєднання до майнінг-пулів. Якщо ви розглядаєте можливість займатися майнінгом, варто звернути увагу на те, чи дозволене це у вашому районі, а також на законодавство та настрої щодо криптовалют в вашій країні, перш ніж вкладати гроші в обладнання для майнінгу [7].

Зростання майнінгу біткойнів та інших систем підтвердження роботи може створити додатковий потенційний ризик у вигляді збільшення споживання енергії необхідної комп’ютерними системами для роботи алгоритмів майнінгу. Хоча ефективність мікрочіпів для чіпів ASIC зростає, розвиток мережі перевищує технологічний прогрес [9]. Це призводить до занепокоєнь щодо впливу на навколишнє середовище та вуглецевого сліду, що залишається від видобутку біткойнів [6].

Тому що криптовалюти є цифровими записами, є ризик, що одна монета може бути скопійована, підроблена або витрачена кілька разів. Майнінг зменшує ці ризики, оскільки він робить будь-яку спробу підробки чи зламу мережі надзвичайно витратною та ресурсомісткою. Фактично, приєднання до мережі як майнер є значно вигіднішим, ніж намагатися її підірвати.

Майнінг відіграє ключову роль у підтвердженні та перевірці нових транзакцій на блокчейні Bitcoin, крім того, що введення в обіг нових BTC. Це особливо важливо через те, що відсутність централізованого органу, такого як уряд, банк або суд, що міг би визначати дійсність операцій, робить процес підтвердження роботи (PoW) майнінгу децентралізованим консенсусом.

Початково будь-хто міг майнити біткоїни на своєму ПК або ноутбуку, однак, зі зростанням зацікавленості до майнінгу та збільшенням розміру мережі, складність алгоритму майнінгу значно зросла. Код біткоїну розрахований на те, щоб кожні десять хвилин знайти новий блок, і з більшою кількістю майнерів шанси на знаходження правильного хешу збільшуються. Це призводить до підвищення складності, що потрібно для знаходження блоку за десять хвилин. Якщо до мережі приєднається тисячі або навіть мільйони нових майнерів, то це приведе до значного збільшення кількості машин, що використовують енергію.

Законність майнінгу біткойн залежить від правової ситуації у вашій країні. Концепція біткойна може підірвати домінування фіатних валют і державний контроль над фінансовими ринками, тому в деяких юрисдикціях він заборонений або обмежений законом.

Майнінг та володіння біткоїнами є законними в багатьох країнах світу. Проте, за деякими звітами, у деяких країнах, таких як Еквадор, Єгипет, Марокко, Болівія, Алжир, Непал і Пакистан, використання та майнінг біткоїнів було оголошено незаконним. Загалом же, використання та майнінг біткоїнів залишається законним у більшості країн світу [6].

2.2 Огляд криптовалюти Bitcoin

Bitcoin - це перша криптовалюта, створена в 2009 році невідомою особою (або групою осіб), яка відома під псевдонімом Satoshi Nakamoto. Історія Bitcoin починається в 2008 році, коли був опублікований документ, що описував нову електронну платіжну систему на основі криптографії. Система

була спроектована для того, щоб дозволити користувачам безпечно і анонімно здійснювати фінансові транзакції через Інтернет без посередництва банків або інших фінансових установ.

Однією з основних інновацій Bitcoin було використання технології блокчейн. Блокчейн - це розподілена база даних, яка зберігає всі транзакції, що відбуваються в мережі, і забезпечує їх безпеку і недоступність для будь-яких змін. Це забезпечує децентралізованість системи і захист від шахрайства [10].

У перші роки свого існування Bitcoin був мало відомим і мало використовувався. Проте з часом його популярність зросла, і стали з'являтися перші майданчики для купівлі та продажу Bitcoin. В 2010 році була здійснена перша покупка товару за Bitcoin, коли програміст Ласло Ханьяць купив дві піци за 10 000 Bitcoin. З того часу Bitcoin став все більш популярним і залучав увагу як інвесторів, так і регуляторів [11].

У наступні роки з'явилося багато інших криптовалют, які базуються на тій же технології блокчейн, що й Bitcoin. Проте Bitcoin залишається однією з найбільш популярних криптовалют на сьогоднішній день [11].

Біткоїн має досить значний вплив на світ, який відчувається в різних аспектах. Ось декілька прикладів:

- Фінансова система: Біткоїн може змінити традиційну фінансову систему, де банки та інші фінансові установи контролюють гроші та транзакції. Завдяки технології блокчейн, біткоїн дозволяє користувачам здійснювати безпечні та швидкі транзакції без необхідності довіряти централізованим організаціям.
- Інвестування: Біткоїн став популярним засобом інвестування. За роки свого існування, цифрова валюта значно зросла в ціні, що зробило її привабливим активом для інвесторів. Однак, наскільки це безпечне та прибуткове інвестування - це ще досліджується.
- Регулювання: Біткоїн також має вплив на регулювання фінансових ринків, особливо коли його використовують для здійснення нелегальних операцій. Більше того, регулятори по всьому світу

досліджують можливість використання технології блокчейн для поліпшення фінансової системи та забезпечення безпеки.

- Технологічний розвиток: Технологія блокчейн, яка стоїть за біткоїном, використовується для розробки нових проектів у різних галузях, включаючи фінанси, медіа, енергетику тощо. Це може відкрити нові можливості та прискорити інновації у світі.
- Відкритість та безпека: Біткоїн працює відкрито і не має централізованої управлінської структури. Будь-хто може створювати акаунти, проводити транзакції та долучатись до мережі. Крім того, транзакції відбуваються відкрито і зберігаються в блокчейні, що забезпечує прозорість та безпеку операцій.
- Мінімальні комісії: Комісії за проведення транзакцій у мережі біткоїн є дуже маленькими порівняно з комісіями за проведення традиційних банківських транзакцій. Це робить біткоїн вигідним для переказів грошей, особливо відправлення коштів за кордон.
- Незалежність від валютного ринку: Біткоїн не залежить від фіатних валют та курсів обміну, тому він є особливо привабливим для тих, хто прагне захистити свої гроші від інфляції та коливань на ринку.
- Створення нових можливостей: За допомогою біткоїну можна створювати нові послуги та продукти, які раніше були неможливі. Наприклад, біткоїн-грошові перекази відкривають можливості для розвитку нових бізнес-моделей у сфері фінансових послуг та електронної комерції.
- Використання у благодійності: Біткоїн може бути використаний у благодійних цілях та для підтримки соціальних проектів, оскільки транзакції відбуваються безпосередньо між учасниками та не потребують посередництва банків.

Рисунок 2.2 - Bitcoin

2.3 Використовувані методи машинного навчання

Машинне навчання - це галузь штучного інтелекту, яка використовує алгоритми та статистичні моделі для ефективного розв’язання конкретних задач за допомогою комп’ютерних систем. Це досягається без використання чітких інструкцій, а замість них використовуються шаблони та логічні висновки. Алгоритми машинного навчання використовують вибірки даних для побудови математичних моделей, які дозволяють здійснювати передбачення та висновки без явного програмування заданої мети [12, 13].

Головною метою системи, яка здійснює навчання, є формування узагальнень на основі зібраного досвіду [13]. У даному контексті узагальнення означає здатність системи точно вирішувати нові завдання, які раніше не були вирішені, після того, як вона отримає навчальний досвід на підставі вибірки даних. За допомогою цієї навчальної вибірки система будує загальну математичну модель, яка дозволяє їй здійснювати передбачення для нових випадків [12].

У 1959 році Самюел Артур ввів термін "машинне навчання" [15]. В перших роботах зі створення штучного інтелекту, деякі науковці вже були

зацікавлені у машинах, які навчаються на основі даних. Однак, в основному вони використовували символьні методи, такі як перцептрони та інші моделі, засновані на узагальнених лінійних статистичних моделях, які пізніше отримали назву "нейромережі" [16].

У машинному навчанні існує багато методів та алгоритмів. Їх загальна класифікація наведена на рис. 2.1.

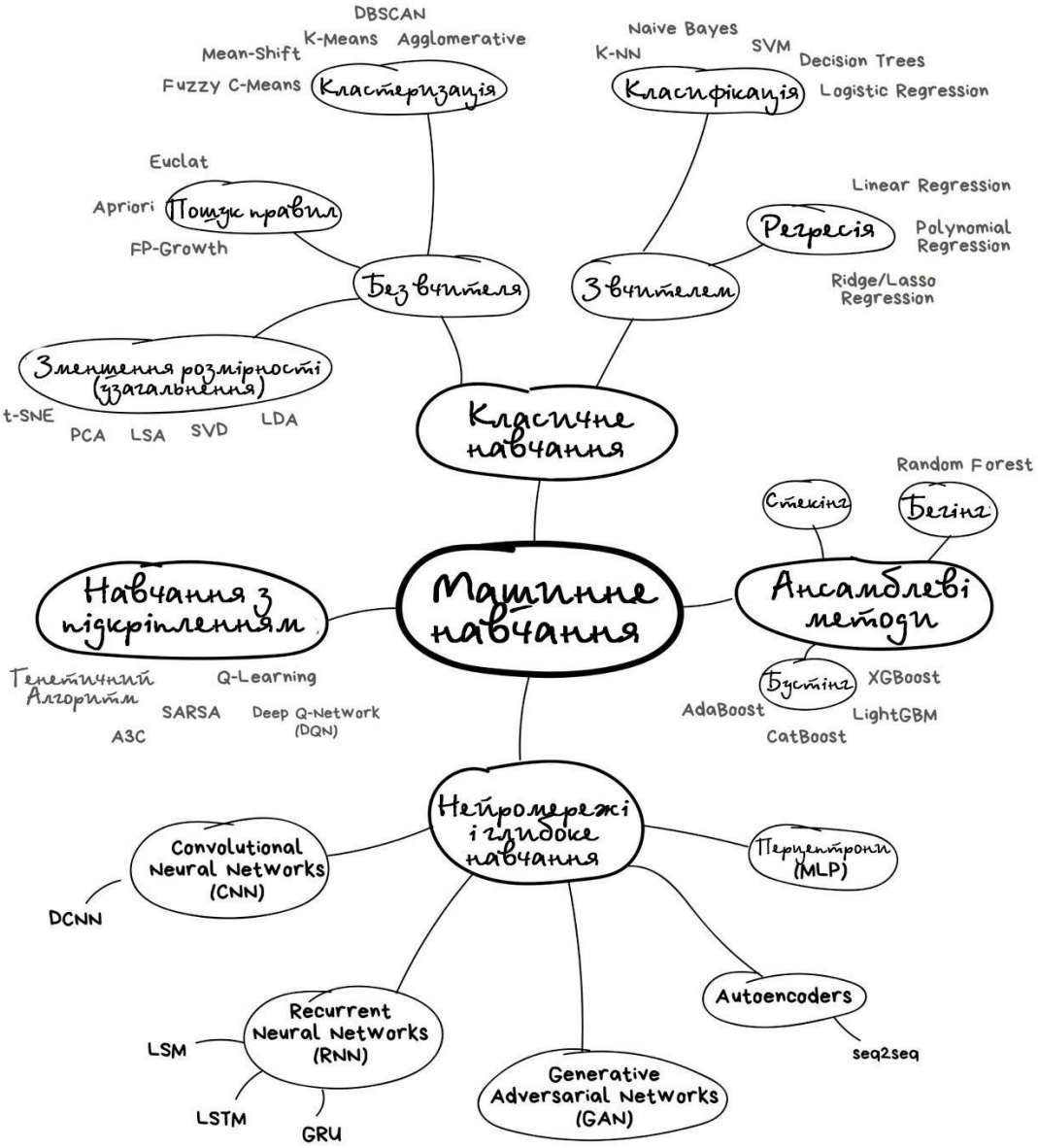

Рисунок 2.1 Загальна класифікація алгоритмів машинного навчання

На початку 80-х років експертні системи стали домінуючою технологією в галузі штучного інтелекту. Тому машинне навчання, яке тоді ще не було відокремлене від штучного інтелекту, зазнало занепаду через недостатність обчислювальних можливостей та проблем з накопиченням даних [15].

У 90-х роках машинне навчання відродилося як окрема підгалузь штучного інтелекту з метою вирішення практичних завдань [13]. Застосовувалися методи статистики та теорії ймовірностей, на відміну від символьних методів, які використовувалися раніше в штучному інтелекті. Зростання обчислювальних можливостей та спрощення доступу до даних завдяки розвитку Інтернету допомогли розвитку машинного навчання [17].

Існує кілька типів машинного навчання: навчання з вчителем, навчання без вчителя та навчання з підкріпленням, кожен з яких вирішує різні задачі. Навчання з вчителем використовується, коли всі дані розмічені, щоб порівнювати результат моделі з правильною відповіддю, і розв’язує регресійні та класифікаційні задачі. Навчання без вчителя використовується, коли дані не розмічені, і дозволяє розв’язувати задачі кластеризації, зниження розмірності даних та пошуку правил. Навчання з підкріпленням використовується для мінімізації помилок у взаємодії з певним середовищем. У цьому випадку для розв’язання задачі прогнозування ціни на криптовалюту найбільш підходящим є навчання з вчителем, оскільки потрібно передбачити числовий результат на основі вхідних даних. Треба розглянути підходящі методи машинного навчання для розв’язання цієї задачі[33].

2.4 Лінійна та поліноміальна регресія

Лінійна регресія - це метод моделювання зв'язків між залежними і незалежними змінними, де залежна змінна є скаляром, а незалежні змінні - векторами (див. рис. 2.2). Якщо є лише одна залежна змінна, то такий випадок називається простою лінійною регресією. У разі наявності декількох залежних змінних мова йде про множинну лінійну регресію [18]. Цей метод за своєю

назвою вирішує завдання регресії.

Рисунок 2.2 Приклад лінійної регресії [21]

Лінійна регресія використовує лінійну предикативну функцію для моделювання зв'язків, де вектор параметрів цієї функції є невідомим, і обчислюється на основі доступних даних. Ці моделі відомі як лінійні [19].

У загальному випадку, модель лінійної регресії має такий вигляд:

$$y_i = \theta_0 + \theta_1 x_{i1} + \cdots + \theta_p x_{ip} + \varepsilon_i,$$

де

$i = 1, \ldots, n$ – порядковий номер випадку;

y_i – залежна змінна;

x_{ip} – p-та координата вектора незалежних змінних (у i-тому випадку);

θ_p – вектор параметрів;

ε_i – непередбачена випадкова похибка, що додає «шум» (у i-му випадку).

Для отримання більш компактного представлення, всі n виразів можна подати у формі матриці:

$$y = X\theta + \varepsilon,$$

де $y = \left(\frac{y_1\ y_2}{\vdots\ y_n}\right)$ – вектор-стовпчик залежної змінної (усіх *i*-их випадків);

$X = \left(\frac{x_1^T\ x_2^T}{\vdots\ x_n^T}\right) = \left(1\ x_{11}\ 1\ x_{21}\ \ \ldots\ x_{1p}\ \ldots\ x_{2p}\ \ \vdots\vdots\ 1\ x_{n1}\ \ \ddots\vdots\ \ldots\ x_{np}\ \right)$ – матриця векторів незалежної змінної (усіх *i*-их випадків);

$\theta = \left(\frac{\theta_0\ \theta_1}{\vdots\ \theta_p}\right)$ – вектор-стовпчик параметрів;

$\varepsilon = \left(\frac{\varepsilon_1\ \varepsilon_2}{\vdots\ \varepsilon_n}\right)$ – вектор-стовпчик випадкових похибок (усіх *i*-их випадків) [20].

У випадку, коли дані мають нелінійний зв'язок, рекомендується використовувати поліноміальну регресію (див. рис. 2.3). Цей підхід відрізняється від лінійної регресії тим, що включає використання поліномів степеня вище за 1.

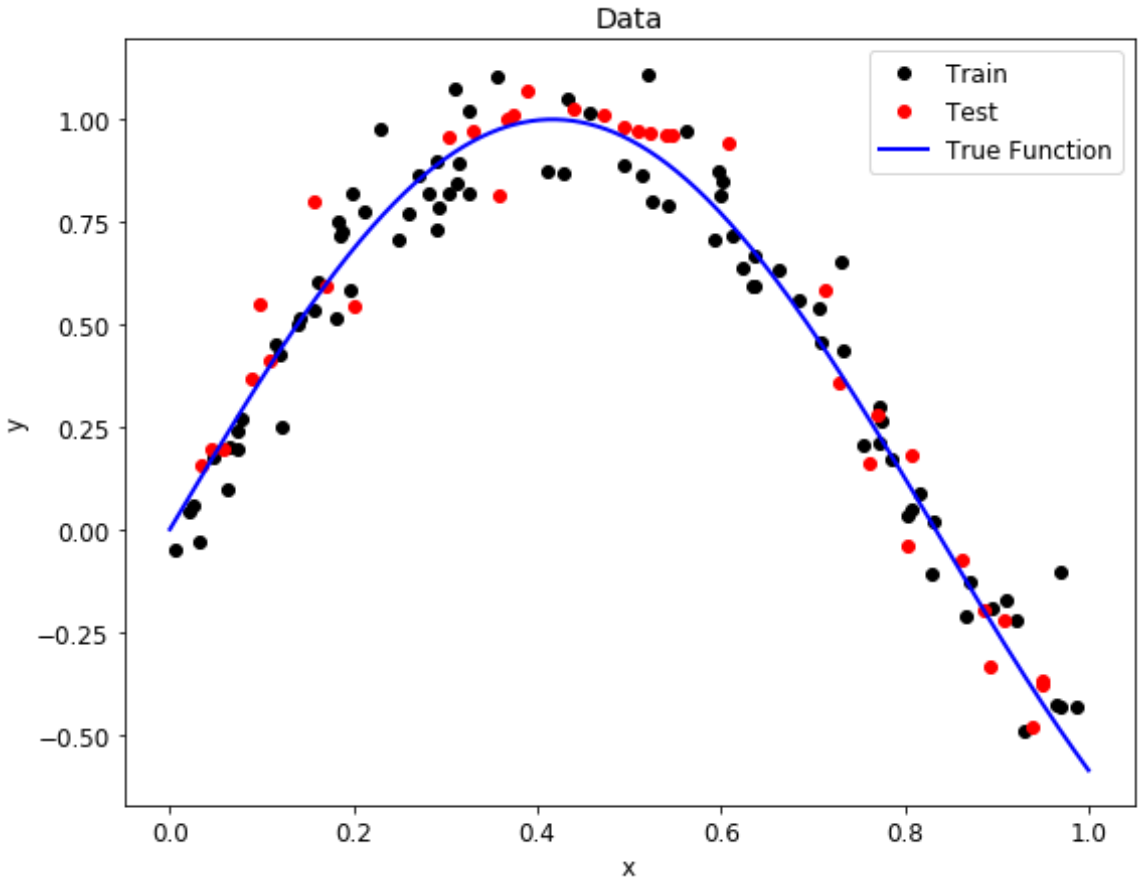

Рисунок 2.3 Приклад поліноміальної регресії [20]

Для досягнення оптимального результату при використанні поліноміальної регресії необхідно вибрати поліноміальну степінь, яка найкраще апроксимує шукану залежність. Для визначення оптимальної степені використовується метод кросс-валідації.

Якщо використовувати поліноміальну функцію з надто низькою степеню, модель буде дуже простою (underfitting), і вона не зможе адекватно відтворити залежність (див. рис. 2.4). В такому випадку помилка моделі буде великою як на навчальних, так і на тестових даних.

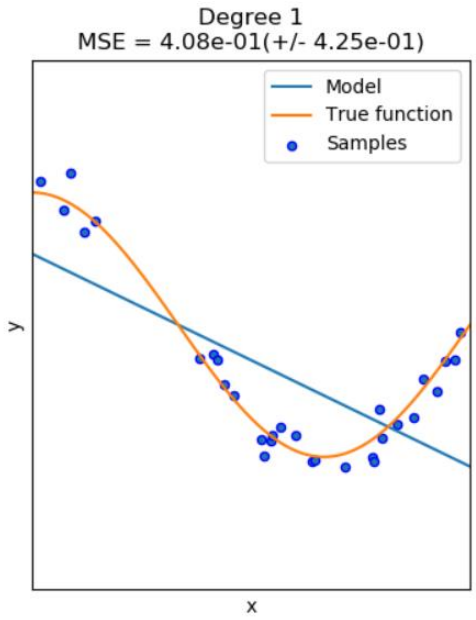

Рисунок 2.4 Занадто проста поліноміальна модель (underfitting) [21]

Якщо використати високу степінь в поліномі, модель може стати перенавченою (overfitting) і спробує максимально "запам'ятати" приклади з навчальної вибірки (див. рис. 2.5). При перенавчанні помилка на навчальній вибірці буде мінімальною, але великою на тестовій вибірці.

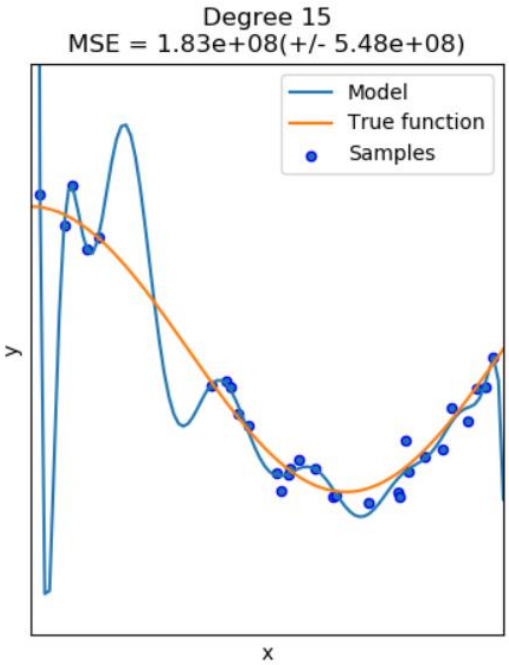

Рисунок 2.5 Занадто складна, перенавчена модель (overfitting) [21]

Існує кілька методів для знаходження оптимальних значень параметрів θ. Ці методи спрямовані на мінімізацію суми квадратів різниці між реальними значеннями і передбачуваними значеннями для всіх випадків. Функція втрат для цього виглядає наступним чином:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \quad (h_\theta\left(x^{(i)}\right) - y^{(i)})^2,$$

де m – кількість випадків, $y^{(i)}$ – реальне значення у i-му випадку, $h_\theta(x^{(i)})$ – прогнозоване значення для i-го випадку.

Градієнтний спуск є найпоширенішим методом. Його суть полягає в повторенні кроків до тих пір, поки значення помилки не стане меншим за ε або певна кількість ітерацій не буде досягнута. Протягом цих кроків усі ваги змінюються одночасно за допомогою наступної формули:

$$\theta_j = \theta_j - \frac{\alpha}{m}\sum_{i=1}^{m} \quad \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) * x_j,$$

де α – параметр швидкості навчання.

Або у матричній формі:

$$\theta = \theta - \frac{\alpha}{m} X^T (h_\theta(x) - y).$$

Проте даний метод має свої недоліки:

- необхідно підбирати α;
- шукає локальні мінімуми;
- при великій різниці між значеннями змінних навчання буде проходити повільніше (необхідно використовувати нормалізацію, або інші методи попередньої обробки).

Інший метод знаходження θ називається нормальне рівняння (Normal equation) й описується наступною формулою:

$$\theta = (X^T X)^{-1} X^T y.$$

Цей метод має декілька переваг:

- Він не потребує ітерацій, що дозволяє отримати максимально можливу точність.
- Швидкість роботи методу не залежить від різниці між значеннями змінних.

Проте він також має деякі недоліки:

- Не завжди можна знайти обернену матрицю, тому цей метод не завжди може бути застосований.
- При значній кількості змінних метод працює повільно через складність алгоритму знаходження оберненої матриці, яка має складність $O(n^3)$.

2.5 Нейронні мережі

Штучна нейронна мережа - це метод машинного навчання, який використовує мережу зі штучних нейронів. Ці нейрони отримують вхідний сигнал, на основі якого змінюють свій внутрішній стан і генерують вихідні значення, що залежать від цього внутрішнього стану і вхідного сигналу [22].

Штучні нейрони мають свої біологічні аналоги, і саме тому на початкових етапах їх розвитку вони були розділені на два напрямки. Перший напрямок був спрямований на моделювання біологічних процесів, а другий - на практичне застосування в задачах [23].

В загальному випадку нейрон та його робота виглядає наступним чином (рис. 2.6):

1) на кожен вхід нейрона (x_j) подається сигнал;
2) вхідний сигнал множиться на відповідну вагу (w_{ij});
3) відбувається додавання отриманих значень;
4) отримана сума подається на функцію активації, яка формує вихідне значення нейрона.

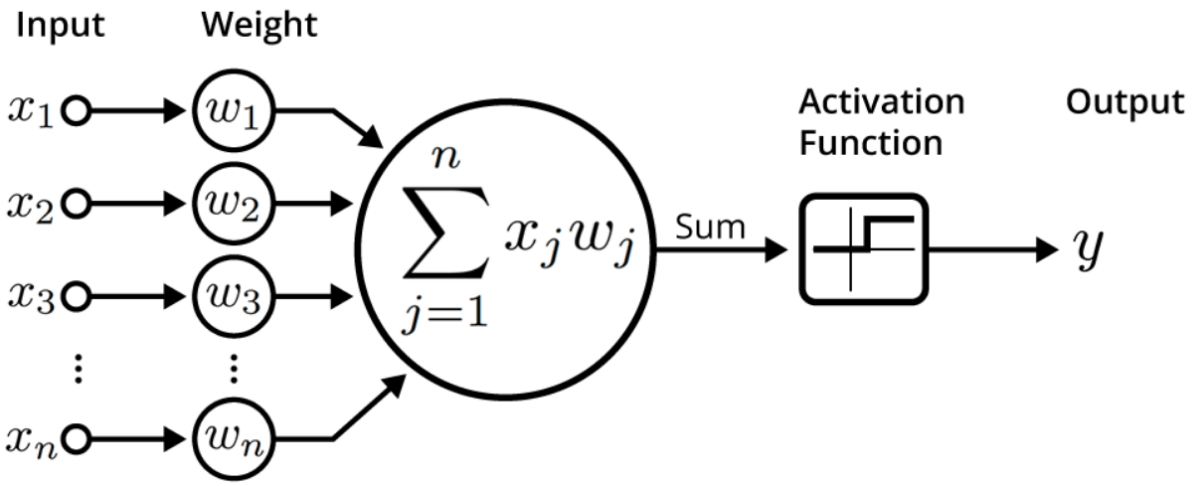

Рисунок 2.6 Ілюстрація штучного нейрона [24]

Для досягнення більшої потужності та здатності вирішувати складні задачі в нейронних мережах використовуються нелінійні функції активації. Це обумовлено тим, що нелінійні функції дозволяють розв'язувати нетривіальні задачі. За універсальною теоремою апроксимації (теорема Цибенко), нейронна мережа з одним прихованим шаром та сигмоїдною функцією активації може апроксимувати будь-яку неперервну функцію багатьох змінних з довільною точністю [26]. Якщо використовувати лінійні функції активації, нейронні мережі могли б вирішувати лише обмежений клас задач, що можуть бути розв'язані лінійними апроксимуючими функціями. У таблиці 2.1 представлені різні функції активації, які можуть бути використані.

Таблиця 2.1 Функції активації

Назва	Рівняння	Область значень
Сигмоїда	$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$	(0, 1)
Гіперболічний тангенс	$f(x) = tanh\ tanh\ (x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$	(-1, 1)
Функція Гауса (дзвоноподібна)	$f(x) = e^{-x^2}$	(0, 1]
ReLU (напівлінійна)	$f(x) = \{0, x < 0\ x, x \geq 0$	[0, ∞)

Напівлінійна з насиченням	$f(x) = \{0, x \leq 0\ x, 0 < x < 1\ 1, x \geq 1$	[0, 1]
Порогова	$f(x) = \{0, x < 0\ 1, x \geq 0$	{0, 1}
Трикутна	$f(x) = \{1 - \lvert x\rvert, \lvert x\rvert \leq 1\ 0, \lvert x\rvert > 1$	[0, 1]

Розглянемо найпопулярніші функції активації більш детально.

Найпопулярнішою функцією активації являється *сигмоїда* (рис. 2.7). Вона є нелінійною, гладкою й монотонно зростаючою функцією.

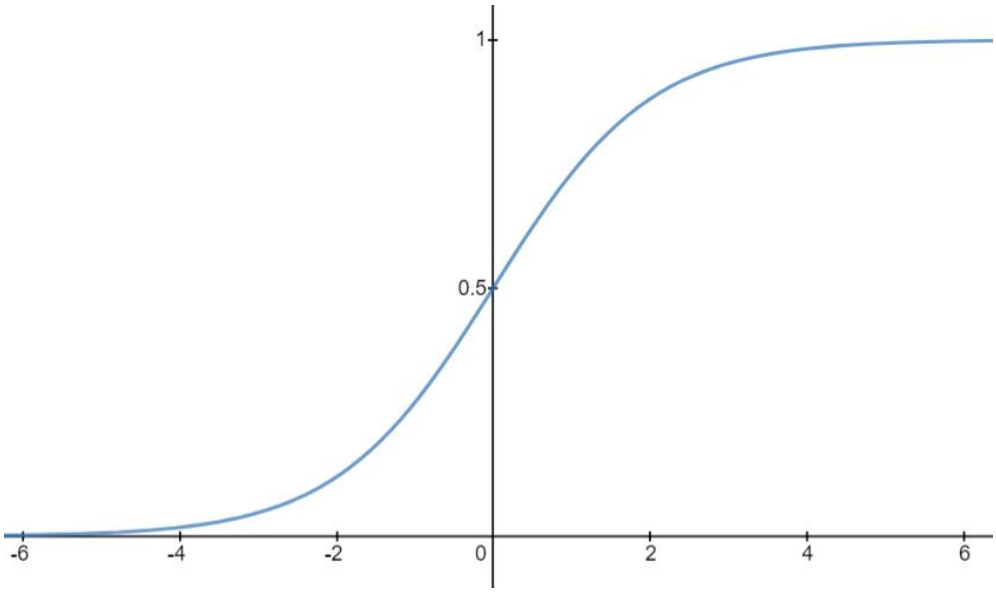

Рисунок 2.7 Графік сигмоїдної функції

Має наступні переваги:

- Значення функції активації знаходяться в діапазоні (0,1), що дозволяє нормалізувати вихідні значення для кожного нейрона.
- Гладкий градієнт функції активації запобігає різким змінам (стрибкам) при обчисленні значень.
- В діапазоні значень x від -2 до 2 значення y швидко змінюється й має тенденцію наближатися до однієї з асимптот, що сприяє зробленню чітких передбачень щодо класу.

Проте має такий недолік: при наближенні до асимптот значення похідної функції активації сильно зменшується, що негативно впливає на швидкість процесу навчання.

Гіперболічний тангенс подібний на сигмоїду як за виглядом (рис. 2.8), так і за деякими властивостями. Проте має відмінності й свої особливості.

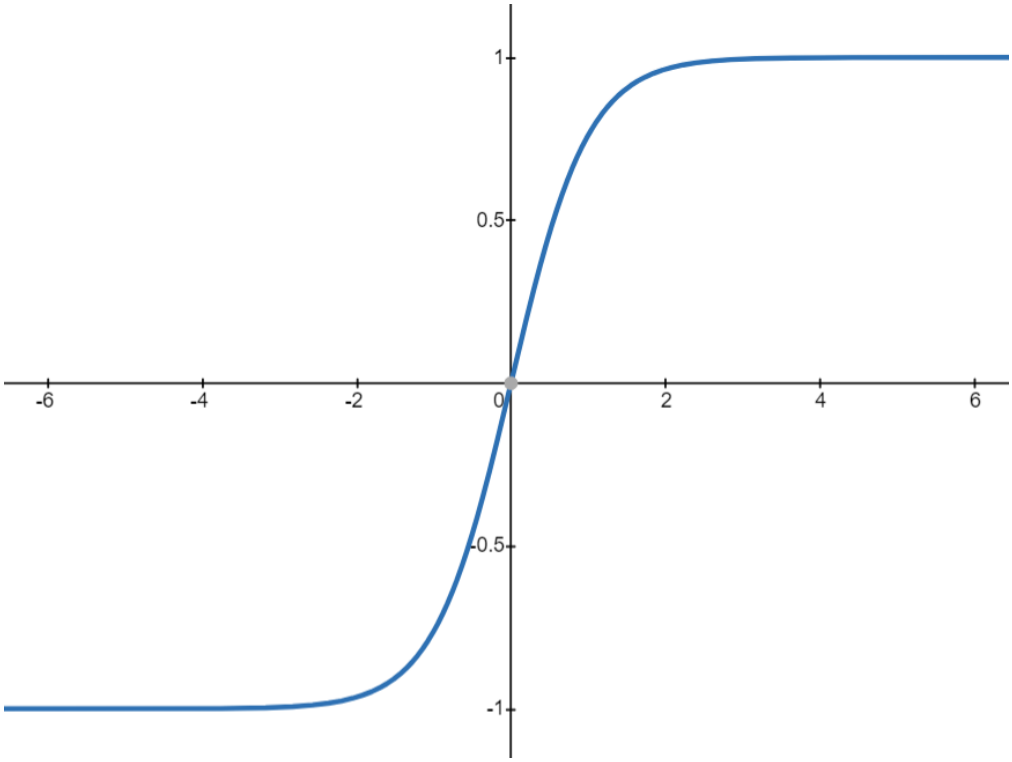

Рисунок 2.8 Графік гіперболічного тангенсу

Переваги:

- Має всі переваги сигмоїди, за винятком нормалізації.
- Має негативні значення, що може бути корисним при роботі з ними.
- У порівнянні з сигмоїдою швидше збігається, завдяки вищому значенню похідної біля нуля.

Недоліки:

- Відсутня нормалізація.
- Гірше працює в задачах класифікації.

Ще однією популярною функцією є *ReLu* (Rectified linear unit). Не дивлячись на її візуальну схожість на лінійну функцію, насправді являється нелінійною(рис. 2.9).

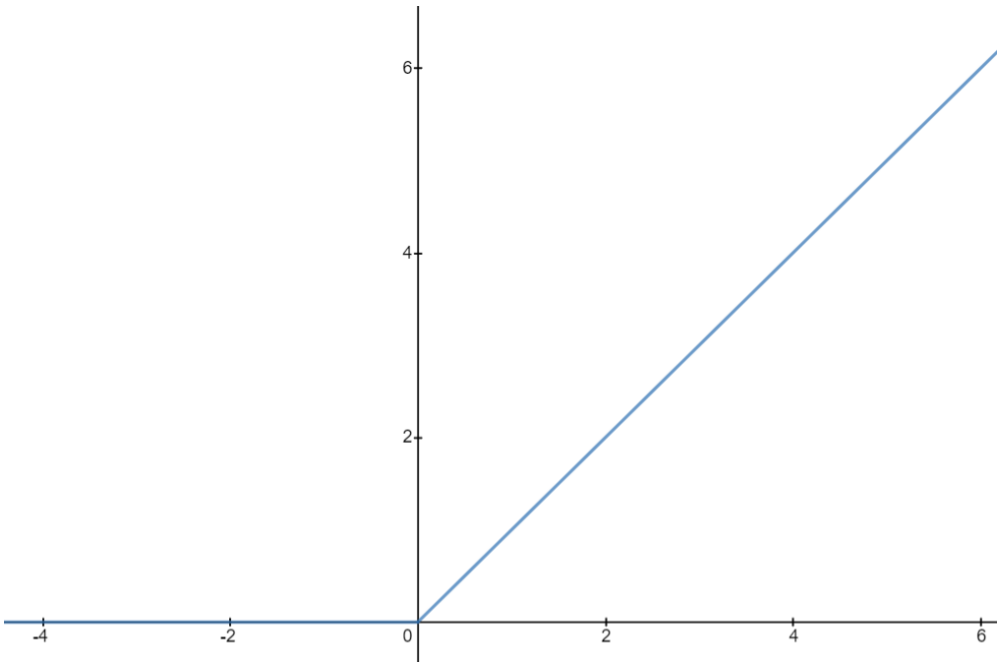

Рисунок 2.9 Графік ReLu

Переваги ReLU:

- Швидко та просто обчислюється похідна.
- Спрощує нейронну мережу, оскільки лише деякі нейрони будуть активовані, що позитивно впливає на швидкість навчання.

Однак у ReLU є недолік: деяка частина функції має похідну 0, що призводить до того, що деякі ваги не змінюються під час навчання.

Неможливо однозначно відповісти, яку функцію активації краще обрати. Вибір функції зазвичай залежить від конкретної задачі та її особливостей (наприклад, сигмоїда добре підходить для задач класифікації, але гіперболічний тангенс може бути менш ефективним). Також враховується схожість функцій до тих, які намагаються апроксимувати[25].

Нейронна мережа формується шляхом з'єднання виходу певних нейронів з входами інших, утворюючи зважений орієнтований граф. Архітектура мережі визначається тим, як нейрони з'єднані між собою. Існує багато різних архітектур, представлених на рис. 2.10. Вибір типу архітектури нейронної мережі залежить від конкретної задачі та необхідних особливостей[27].

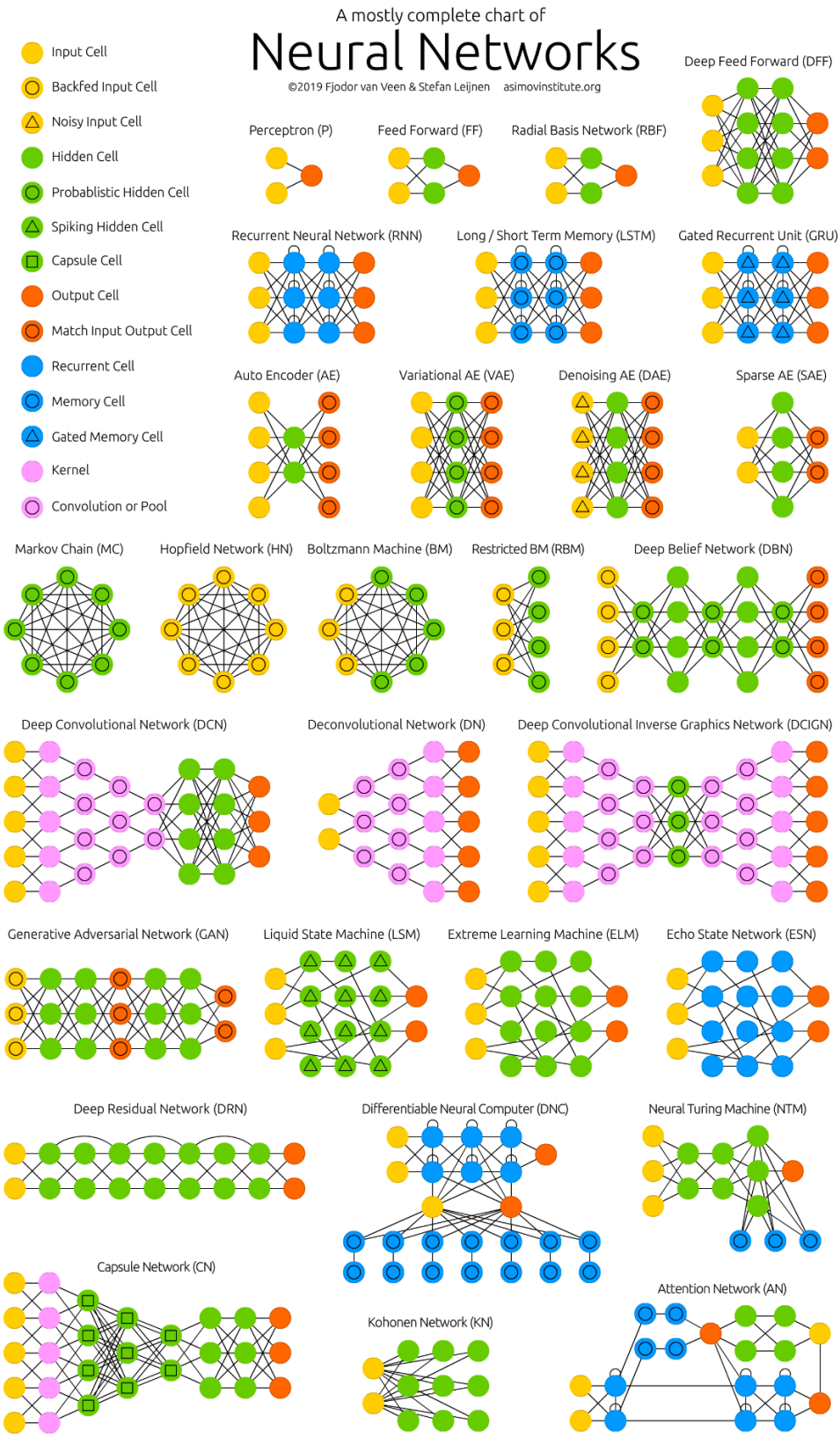

Рисунок 2.10 Класифікація архітектур нейронних мереж [28]

Поміж різниці в типах зв'язків, архітектури нейронних мереж можуть відрізнятися за кількістю прихованих шарів та кількістю нейронів.

Незважаючи на універсальну теорему апроксимації, деякі функції є дуже складними або практично неможливими точно апроксимувати лише за допомогою одного прихованого шару.

Для досягнення точної роботи мережі, оптимальна кількість шарів та нейронів зазвичай встановлюється експериментальним шляхом. Замала кількість шарів або нейронів може призводити до недостатньої здатності мережі виявляти залежності (underfitting), подібно до намагання апроксимувати квадратичну залежність лінійною функцією. З іншого боку, забагато шарів або нейронів може призводити до перенавчання мережі (overfitting), коли вона "запам'ятовує" навчальну вибірку, де помилка дуже мала, але не може точно передбачати результати для нових вхідних даних або реальних значень. Крім того, така мережа працює повільніше та потребує більше часу для навчання[29].

Однією з найперших і до сих пір популярних архітектур є багатошаровий перцептрон (рис. 2.11), який є повнозв'язною мережею прямого поширення.

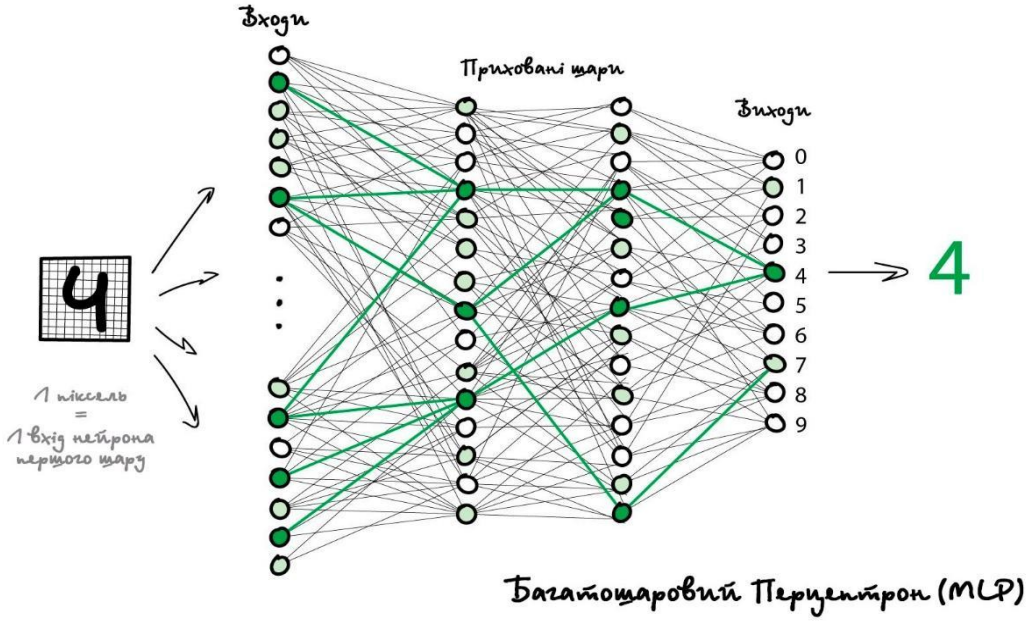

Рисунок 2.11 Приклад багатошарового перцептрону [27]

Мережа прямого поширення (Feed-Forward Neural Networks) - це тип нейронної мережі, в якій сигнал передається лише в одному напрямку - від вхідного шару, через приховані шари, до вихідного шару, де отримується результат у формі скаляра або вектора.

Повнозв'язна нейронна мережа - це мережа, в якій кожен нейрон в наступному шарі зв'язаний з усіма нейронами попереднього шару.

Переваги багатошарового перцептрону:

- проста архітектура;
- універсальність – може вирішувати різні задачі.

Недоліки:

- у більш складних задачах буде мати занадто багато параметрів, що робить його використання нераціональним;
- при великій кількості прихованих шарів відбувається «затухання» градієнта, у результаті чого навчання відбувається набагато довше.

Таким чином, можна зробити висновок, що багатошаровий перцептрон є ефективним для простих задач класифікації або регресії, але для складніших завдань, наприклад розпізнавання зображень, може знадобитися інша архітектура нейронної мережі. Тому для нашої конкретної задачі така архітектура мережі є відповідною.

Навчання нейронної мережі передбачає коригування ваг між нейронами, щоб забезпечити її правильну роботу. Існують різні алгоритми навчання, які можна класифікувати наступним чином, залежно від архітектури мережі:

- Навчання з вчителем.
- Навчання без вчителя.
- Змішане навчання.
- Навчання з підкріпленням.

Перед початком навчання ваги нейронної мережі потрібно ініціалізувати певними значеннями. Для цього можна скористатися вагами з попередньо навчених мереж або вибрати випадкові невеликі значення.

Один з найпопулярніших алгоритмів навчання - метод зворотного поширення помилки. Його суть полягає в порівнянні отриманого результату зі звичайним значенням і розрахунку помилки. Потім помилка поширюється у зворотному напрямку, враховуючи внесок кожного нейрона. На основі цього внеску ваги коригуються для кожного нейрона окремо[30].

Алгоритм методу:

1. Ініціалізувати ваги (w_{ij}), та $\Delta w_{ij} = 0$;
2. Повторювати доки помилка > ε або досягнута певна кількість ітерацій:

Для усіх прикладів з навчальної вибірки:

1. подати вхідні значення та отримати вихідні значення з кожного нейрону (a_i);
2. для кожного вихідного нейрону розраховується помилка $\delta_k = f'(a_k) * (y_k - a_k)$, де f' – похідна від функції активації, y_k – реальне значення, $k \in$ кількість вихідних нейронів;
3. для кожного наступного шару i ($i \in$ кількість шарів $- 1$): Для кожного нейрону j ($j \in$ кількість нейронів у i) розраховується

$$4.\ \delta_j = f'(a_j) * \sum_{k \in \text{нейрони з } i+1} \delta_k w_{j,k}.$$

5. для кожного w_{ij}:

$$6.\ \Delta w_{ij} = \alpha * \delta_j * a_i;\ w_{ij} := w_{ij} + \Delta w_{ij},$$

7. де α – параметр швидкості навчання;
8. повернути значення w_{ij}.

Таким чином, створення нейронної мережі можна звести до таких етапів:

1. попередня обробка даних(за необхідністю);
2. створення навчальної та тестової вибірки;

3. вибір архітектури мережі;
4. вибір алгоритму навчання;
5. вибір параметрів навчання;
6. навчання мережі;
7. перевірка точності мережі;
8. при необхідності скоригувати параметри мережі або провести повторне навчання.

2.5.1 LSTM (Long short-term memory)

Long short-term memory - це рекурентна нейронна мережа, яка використовується переважно у завданнях, пов’язаних із глибоким навчанням, включаючи прогнозування фінансових часових рядів. Архітектура цієї мережі значно складніша, ніж у класичного багатошарового персептрона. Далі розглянемо її детальніше.

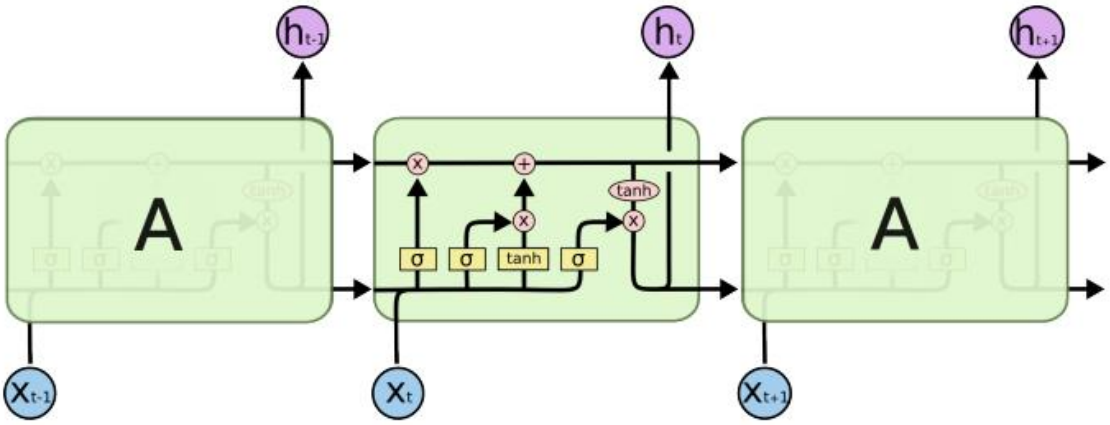

Рисунок 2.12 LSTM-chain

Архітектура мережі LSTM має вигляд ланцюга, що складається з кількох модулів, кожен з яких складається з 4 шарів, які взаємодіють між собою певним чином. Основна відмінність цієї мережі від MLP полягає у тому, що ми маємо можливість передати стан з одного модуля до наступного, при цьому кожен модуль може видалити (забути) частину цієї інформації, або ж, навпаки, щось до неї додати, саме це і створює ефект "пам'яті"[32].

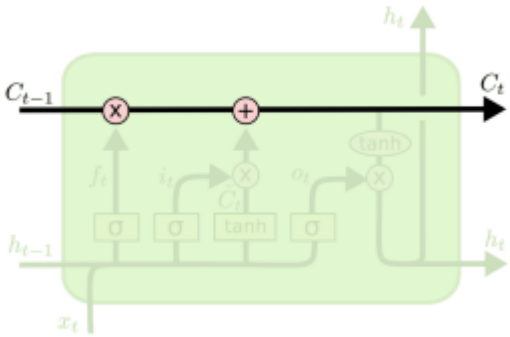

Рисунок 2.13 Cell state C

Далі ми розглянемо, як саме працює мережа LSTM, на прикладі повного шляху даних через модуль LSTM.

На початку модуля ми потрапляємо до вузла забуття (forget gate), який описується наступною формулою:

$$ft = \sigma(Wf \cdot [ht-1, xt] + bf)$$

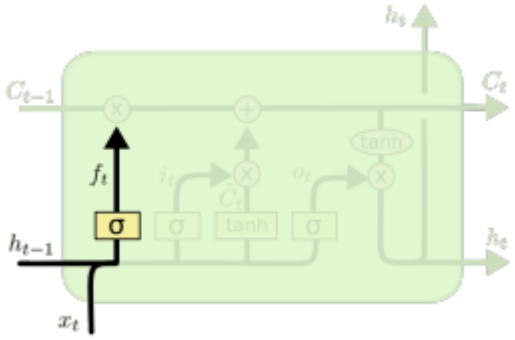

Рисунок 2.14 LSTM-1st step

Наступний наш крок складатиметься із двох. Спочатку йде вузол вхідного шару (input gate layer).

$$it = \sigma(Wi \cdot [ht-1, xt] + bi)$$

після ж йде tanh шар

$$\tilde{C}t = tanh(Wc \cdot [ht-1, xt] + bc)$$

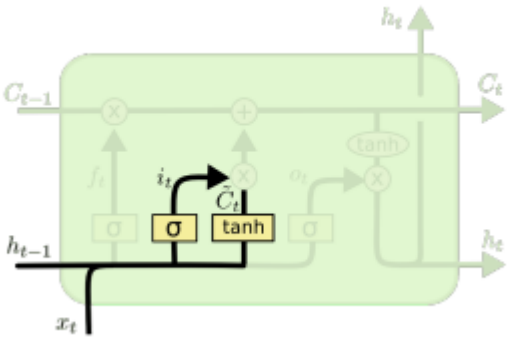

Рисунок 2.15 LSTM-2d step

Наступним кроком ми оновлюємо старий стан модуля на новий, для якого застосуємо результати першого та другого етапу до $Ct - 1$

$$Ct = ft * Ct - 1 + it * C\tilde{} t$$

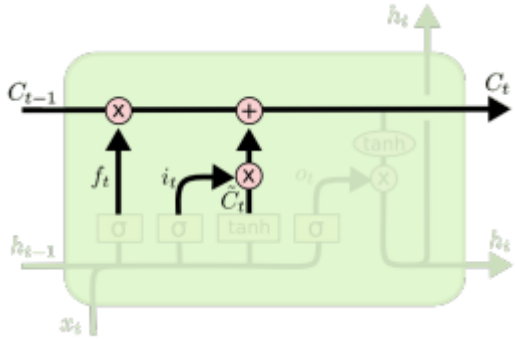

Рисунок 2.16 LSTM-3d step

Останній етап – вихідний вузол (output gate). У ньому ми отримуємо вихідний внутрішній стан комірки.

$$ot = \sigma(Wo[ht - 1, xt] + bo)$$

$$ht = ot * tanh(Ct)$$

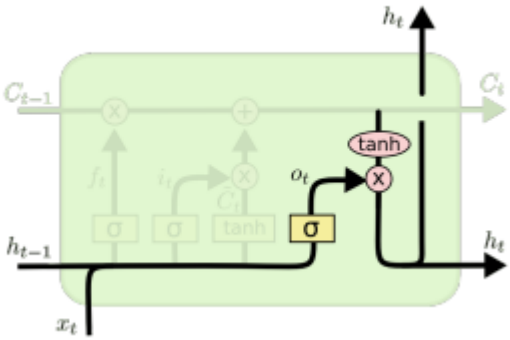

Рисунок 2.17 LSTM-4th step

LSTM мережа відмінно працює з даними, де треба запам'ятовувати якусь інформацію на тривалий термін. Також вона не так сильно схильна до проблеми зникаючого градієнта, на відміну від звичайної RNN [27].

2.5.2 GRU (Gated Recurrent Unit)

GRU (Gated Recurrent Unit) - ще одна мережа з сім'ї рекурентних нейронних мереж. Фактично є переглядом архітектури LSTM у сторону її спрощення. У ній вхідний вузол і вузол забуття об'єднані в один вузол - вузол оновлення (update gate), також є кілька змін у внутрішній архітектурі. GRU модуль можна описати такими формулами:

$$zt = \sigma(Wz \cdot [ht-1, xt])$$
$$rt = \sigma(Wr \cdot [ht-1, xt])$$
$$h\tilde{}t = tanh(W \cdot [rt * ht-1, xt])$$
$$ht = (1 - zt) * ht-1 + zt * h\tilde{}t$$

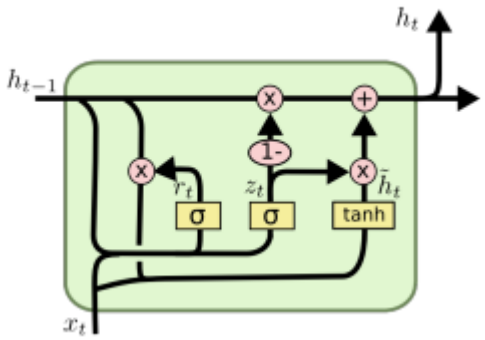

Рисунок 2.18 GRU

До основних плюсів GRU порівняно з LSTM відноситься більш швидке навчання, так як архітектура GRU простіше, і, як наслідок, там менше різних операцій. При цьому мережі GRU можуть працювати краще LSTM на тих же даних [28].

2.6 Random forest

Random forest - це метод машинного навчання, який використовує ансамбль дерев рішень. Цей метод поєднує в собі дві ідеї: метод беггінгу та метод випадкових підпросторів [32]. Random forest використовується як для задач класифікації, так і для задач регресії. Дерева рішень є ієрархічними деревоподібними структурами, які складаються з вирішальних правил "якщо-то". Вони дозволяють класифікувати об'єкти. (рис. 2.12) [31].

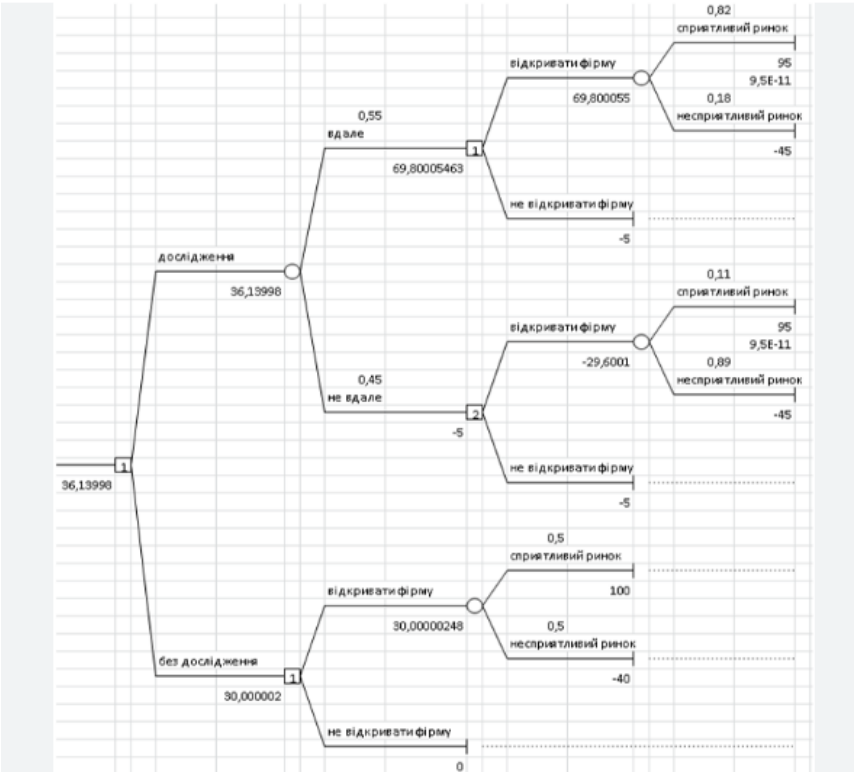

Рисунок 2.19 Приклад дерева рішень [34]

Дерева рішень є ієрархічними структурами, які складаються з вузлів та листків. У вузлах містяться правила, за допомогою яких проводиться перевірка ознак та відбувається подальше розбиття множини об'єктів на підмножини. Листки є кінцевими вузлами дерева, які містять підмножини, що асоційовані з конкретними класами. Відмінність листків від вузлів полягає в тому, що тут не проводиться перевірка та подальше розгалуження.

При побудові дерева рішень на основі навчальної вибірки формуються правила рішень, і для кожного правила створюється вузол. Для кожного вузла необхідно обрати ознаку (атрибут), за допомогою якої проводиться перевірка

правила. Вибір ознак здійснюється таким чином, щоб забезпечити найкраще розбиття у вузлі. Це означає, що вибір ознаки має забезпечити максимально чисті підмножини, які дозволяють класифікувати найбільшу кількість об'єктів [28].

Ансамбльні методи використовують кілька моделей одночасно з метою досягнення більшої точності, ніж окремі моделі [33]. Ці методи можуть базуватися на однакових або різних методах навчання. Використані моделі повинні відрізнятися одна від одної і зазвичай обирають "нестабільні" методи навчання, що можуть бути чутливі до викидів або аномалій.

Ефективність ансамблевих методів можна пояснити теоремою Кондорсе "Про журі присяжних" [34]. Якщо кожен член журі присяжних має незалежну думку і ймовірність правильного рішення кожного з них більше 0,5, то ймовірність правильного рішення всього журі зростає зі збільшенням кількості його членів і наближається до одиниці. Цю ідею можна математично виразити так:

$$\mu = \sum_{i=m}^{N} C_N^i p^i (1-p)^{N-i},$$

де μ – ймовірність правильного рішення, p – ймовірність правильного рішення присяжного, m – мінімальна більшість членів журі ($m = \lfloor N/2 \rfloor + 1$), N – кількість присяжних. Отже, якщо $p > 0{,}5$, то $\mu > p$. А якщо $N \to \infty$, то $\mu \to \infty$.

Беггінг (Bagging) є одним з видів ансамблів, який використовує статистичний метод бутстрепу [37]. Метод бутстрепу передбачає випадкове вибирання з поверненням N об’єктів з вибірки X розміром N, щоб створити підвибірку X_1. Таким чином, на кожній ітерації кожен об’єкт має однакову ймовірність вибору, що дорівнює $1/N$. Підвибірки $X_1, \ldots, X_M$ генеруються шляхом повторення цієї процедури M разів.

Застосування методу бутстрепу для генерації підвибірки призводить до того, що близько 37% об’єктів з початкової вибірки не потрапляють до підвибірки, і їх називають "Out-of-bag". Цю властивість можна математично

довести наступним чином: якщо вибірка містить N об’єктів, то на кожній ітерації кожен об’єкт входить до підвибірки з ймовірністю $1/N$. Тому ймовірність того, що об'єкт не потрапить до підвибірки, дорівнює $(1-1/N)^N$. Якщо $N \to \infty$, то ймовірність цього стає $\frac{1}{e} \approx 37\%$. Ймовірність того, що кожен об'єкт потрапить до підвибірки, становить $1-1/e$, що приблизно дорівнює 63%.

Основна ідея беггінгу, як зображено на рис. 2.20, полягає в наступному: з вибірки X генеруються M підвибірок $X_1, \ldots, X_M$. Для кожної з цих підвибірок тренується окрема модель $a_i(x)$. Кінцева модель об'єднує результати кожної з цих моделей, усереднюючи їх відповіді: $a(x) = \frac{1}{M}\sum_{i=1}^{M} a_i(x)$. У випадку класифікації використовується голосування.

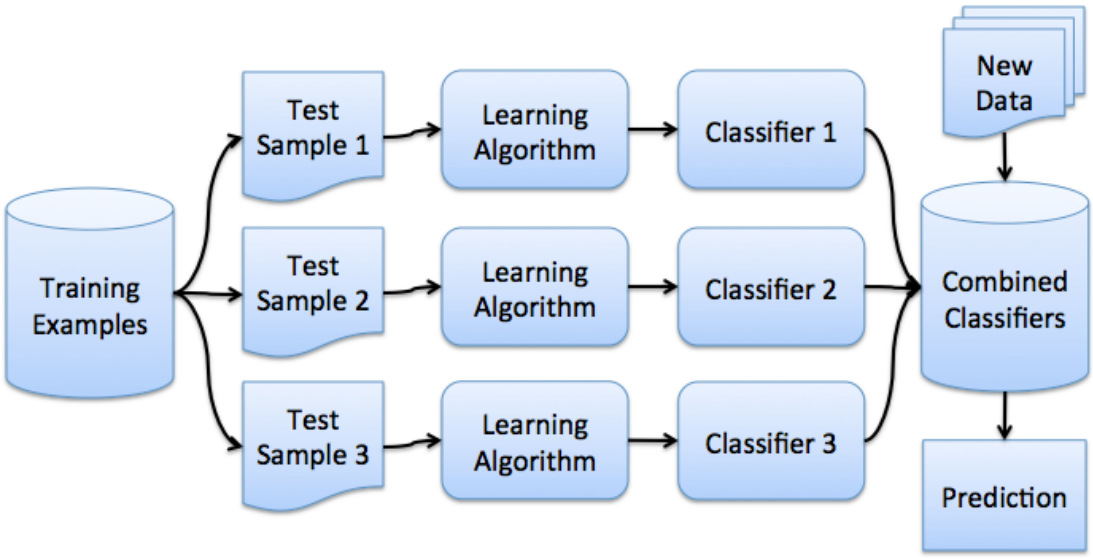

Рисунок 2.20 Приклад ансамблевого методу на основі беггінгу [38]

Беггінг досягає ефективності завдяки різноманітності моделей, які навчаються на різних підвибірках даних та компенсують помилки одна одної. Також, деякі викиди в даних можуть не потрапляти до певних підвибірок, що додатково покращує роботу ансамблю. Беггінг є ефективним на малих вибірках, де відсутність навіть невеликої частини даних дозволяє отримати

різноманітні моделі. У випадку великих вибірок можна створювати підвибірки меншого розміру.

Окрім беггінгу, метод випадкових лісів використовує метод випадкових підпросторів [39], у якому моделі навчаються на випадково обраних підмножинах ознак. Цей метод дозволяє знизити кореляцію між деревами та ризик перенавчання. Алгоритм побудови ансамблю на основі методу випадкових підпросторів полягає у наступному: нехай є N об'єктів у вибірці, D ознак й M моделей у ансамблі.

Метод випадкових підпросторів, який застосовується випадковим лісом, полягає у тому, що для кожної моделі $a_i(x)$ випадковим чином вибирається підмножина ознак довжиною d, де $d < D$, і проводиться навчання на цій підмножині. Після цього, на основі отриманих моделей утворюється ансамбль, результат у якому отримується шляхом усереднення відповідей або голосування залежно від задачі:

$$a(x) = \frac{1}{M}\sum_{i=1}^{M} \quad a_i(x),$$

Зазвичай, для задач регресії рекомендується використовувати $d = D/3$, а для задач класифікації $d = \sqrt{D}$, але оптимальне значення d може різнитися для різних задач і підбирається експериментальним шляхом [40].

Отже, якщо поєднати беггінг на деревах рішень з методом випадкових підпросторів, то ми отримаємо алгоритм випадкового лісу. Процедура будування випадкового лісу така:

Для кожного з $n = 1, \dots, N$, де N дерев в ансамблі:

- згенерувати підвибірку X_n за допомогою бутсрепу з вибірки X;
- побудувати дерево рішень $a_i(x)$ на основі підвибірки X_n:
    - використовуючи заданий критерій, обирати кращу ознаку для розбиття на кожному кроці побудови дерева, доки підвибірка не буде вичерпана;

- на кожному кроці побудови дерева випадковим чином вибирати d ознак із D і шукати оптимальне розділення лише серед них;
- дерево будується до досягнення певної висоти або певної кількості об'єктів у листках.

- З отриманих дерев рішень формується ансамбль, результат якого визначається як $a(x) = \frac{1}{N}\sum_{i=1}^{N} \quad a_i(x)$.

2.7 Аналіз вхідних даних

Прогнозування цін на криптовалюти залежить від якості вхідних даних, які використовуються для навчання моделей машинного навчання. Основні види вхідних даних для прогнозування цін на криптовалюти можуть включати наступні:

1. *Історичні дані* - це дані про ціни на криптовалюти за певний період часу. Історичні дані використовуються для навчання моделей машинного навчання та створення прогнозів цін на майбутнє.
2. *Фундаментальні дані* - це дані про фундаментальні показники криптовалют, такі як кількість транзакцій, капіталізація ринку, кількість користувачів, технічні параметри тощо. Фундаментальні дані можуть допомогти встановити зв'язок між фундаментальними показниками та ціною на криптовалюту.
3. *Дані соціальних мереж* - це дані про настрої користувачів стосовно криптовалют, які знаходяться у соціальних мережах та форумах. Аналіз цих даних може допомогти в прогнозуванні цін на криптовалюти на основі емоцій та настроїв користувачів
4. *Дані новин* - це дані про події, які можуть впливати на ціни на криптовалюти. Наприклад, новини про законодавство стосовно криптовалют, нові технології, заяви представників відомих компаній

тощо. Аналіз цих даних може допомогти в прогнозуванні зміни цін на криптовалюти.

5. *Технічні аналізи* - це дані про графіки та індикатори, які використовуються для аналізу цін на криптовалюти. Наприклад, рівні підтримки та опору, рух середньої лінії.
6. *Дані ринкової капіталізації* - це дані про капіталізацію ринку криптовалют. Ринкова капіталізація може допомогти визначити розмір ринку та загальну популярність криптовалют.
7. *Дані про мережу* - це дані про технічні параметри криптовалютної мережі, такі як швидкість транзакцій, кількість блоків та майнерів, рівень складності майнінгу тощо. Аналіз цих даних може допомогти в зрозумінні технічної стійкості та ефективності мережі криптовалюти.
8. *Дані про транзакції* - це дані про обсяг транзакцій та їхній розподіл за адресами. Аналіз цих даних може допомогти виявити тенденції та зміни в поведінці користувачів криптовалюти.

У загальному, для прогнозування цін на криптовалюти використовуються різноманітні вхідні дані з різних джерел. Важливо враховувати якість та достовірність даних при їхньому використанні для навчання моделей машинного навчання. Комбінація різних видів даних може допомогти збільшити точність та надійність прогнозів цін на криптовалюти.

Також для покращення якості моделі додають індикатори.

2.8 Характеристика індикаторів

Індикатори криптовалют - це статистичні показники, які допомагають аналізувати ринок та прогнозувати ціни на криптовалюти. Існує багато різних типів індикаторів, але основні можна класифікувати на наступні категорії:

1. *Технічні індикатори* - це інструменти, які використовуються для аналізу графіків цін на криптовалюти. Такі індикатори можуть допомогти виявити тенденції на ринку та зробити прогнози цін на криптовалюти на

основі технічного аналізу. До прикладів таких індикаторів можна віднести: RSI (Relative Strength Index), MACD (Moving Average Convergence Divergence), Bollinger Bands тощо.

2. *Фундаментальні індикатори* - це інструменти, які використовуються для аналізу фундаментальних показників криптовалют, таких як капіталізація, обсяг торгів, кількість користувачів, технічні параметри мережі та інші. Ці індикатори допомагають зрозуміти економічну ситуацію на ринку та виявити фундаментальні тенденції.
3. *Соціальні індикатори* - це інструменти, які використовуються для аналізу поведінки та думок користувачів криптовалют в соціальних мережах та інших джерелах. Такі індикатори можуть допомогти виявити тренди та сентимент на ринку та прогнозувати майбутні ціни на криптовалюти на основі психологічного фактору.
4. *Складні індикатори* - це інструменти, які використовуються для аналізу різних показників та їх взаємозв'язків для створення комплексних моделей прогнозу.

В нашій системі буде використано технічні індикатори. Серед технічних індикаторів, що використовуються для аналізу цінової динаміки криптовалют можна виділити:

- *Середня ковзна ціна* (MA) - це індикатор, який показує середнє значення цін криптовалюти протягом певного періоду часу. Він допомагає виявити загальний тренд цін на криптовалюту.
- *Індекс сили* (RSI) - це індикатор, який вказує на те, наскільки перекупленою або перепроданою є криптовалюта в даний момент часу.
- *Стохастичний осцилятор* (Stochastic Oscillator) - це індикатор, який вказує на те, наскільки швидко змінюється ціна криптовалюти відносно її нижньої та верхньої межі за певний період часу.

- *Боліндже́р Бенди* (Bollinger Bands) - це індикатор, який вказує на те, яка є нормальна цінова діапазон, в якому зазвичай коливається ціна криптовалюти.
- *Індекс відносної сили* (Relative Strength Index, RSI) - це індикатор, який вимірює силу зміни цін криптовалюти в певний період часу.
- *Об'єм* (Volume) - це індикатор, який вказує на обсяг торгів криптовалютами за певний період часу.

2.9 Аналіз функцій системи прогнозування цін на криптовалюту методами машинного навчання

Систему прогнозування цін на криптовалюту з використанням методів машинного навчання можна проаналізувати з точки зору її функцій, які можна розділити на такі категорії:

Збір даних: система повинна мати можливість збирати дані з різних джерел, таких як криптовалютні біржі, платформи соціальних мереж і веб-сайти новин. Зібрані дані повинні включати історичну ціну криптовалюти, обсяг торгів, настрої новин та іншу відповідну інформацію.

Попередня обробка даних: зібрані дані слід попередньо обробити, щоб усунути шум і невідповідності. Це передбачає очищення даних, інтеграцію даних і перетворення даних. Дані також повинні бути нормалізовані, щоб гарантувати, що кожна функція має однаковий масштаб і діапазон.

Вибір функцій: система повинна визначити найбільш відповідні функції, які мають найбільший вплив на ціну криптовалюти. Це передбачає вибір підмножини функцій, які є найбільш інформативними, і відкидання нерелевантних або зайвих функцій.

Вибір моделі: система повинна мати можливість вибрати відповідну модель машинного навчання, яка може ефективно вивчати закономірності в даних і точно прогнозувати ціну криптовалюти. Це передбачає порівняння продуктивності різних моделей машинного навчання та вибір найкращої на

основі різних показників, таких як точність, запам’ятовування та оцінка F1.

Навчання моделі: вибрану модель машинного навчання слід навчити на попередньо оброблених даних, щоб вивчити шаблони в даних. Це передбачає поділ даних на навчальні та тестові набори та використання навчального набору для навчання моделі.

Оцінка моделі: навчену модель слід оцінити на наборі для тестування, щоб виміряти її продуктивність. Це передбачає обчислення різних показників, таких як точність, запам’ятовування та оцінка F1, а також порівняння результатів з іншими моделями.

Прогноз: система повинна мати можливість передбачити майбутню ціну криптовалюти на основі навченої моделі та поточних даних. Це передбачає введення поточних даних у навчену модель і отримання прогнозу майбутньої ціни.

Візуалізація: система повинна забезпечувати візуалізацію прогнозованих цінових тенденцій та іншу відповідну інформацію, таку як обсяг торгів і настрої новин. Це допомагає користувачам зрозуміти прогнозовані тенденції цін і приймати зважені рішення.

На рис. 2.21 зображено дерево функцій системи:

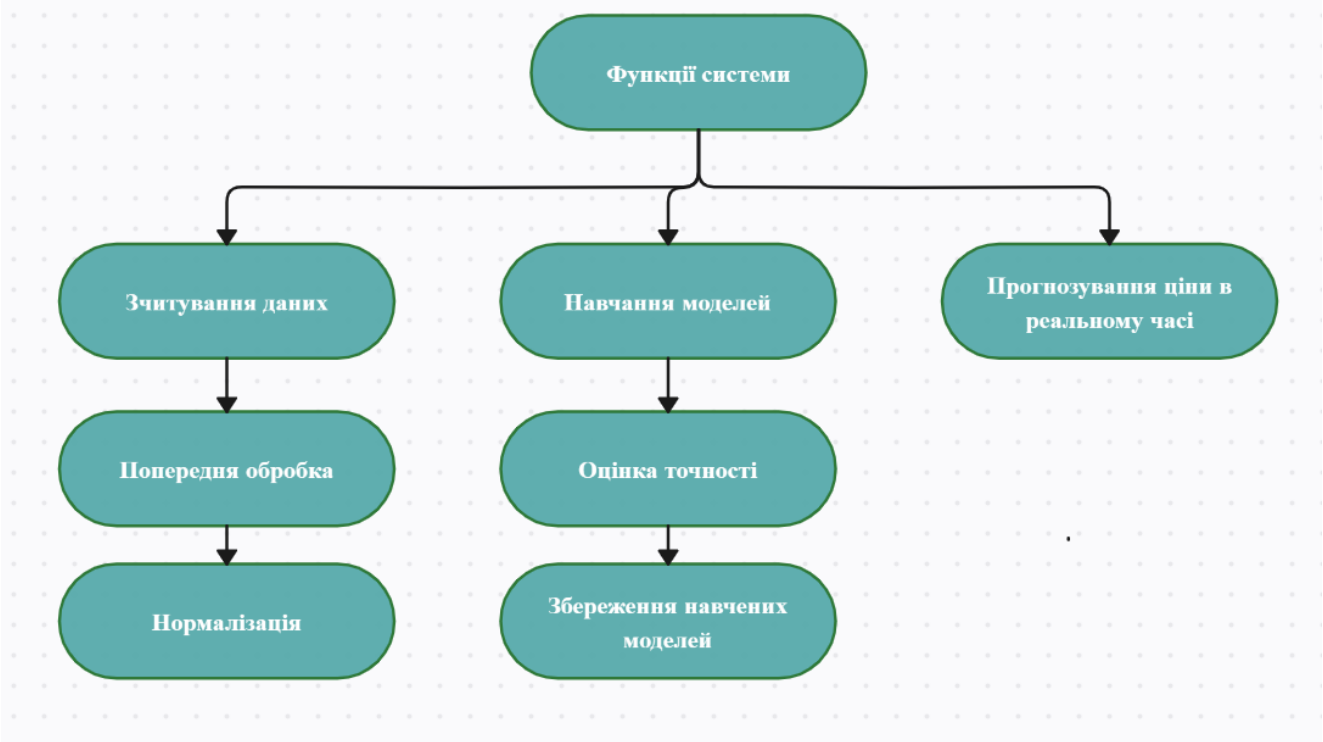

Рисунок 2.21 – Дерево функцій системи

Загалом система прогнозування цін на криптовалюту з використанням методів машинного навчання повинна мати можливість збирати, попередньо обробляти, вибирати функції, вибирати моделі, навчати моделі, оцінювати моделі, робити прогнози та візуалізувати результати. Ці функції можна досягти за допомогою різних методів машинного навчання, таких як регресія, класифікація, кластеризація та глибоке навчання.

На рис. 2.22 представлено узагальнену архітектуру системи.

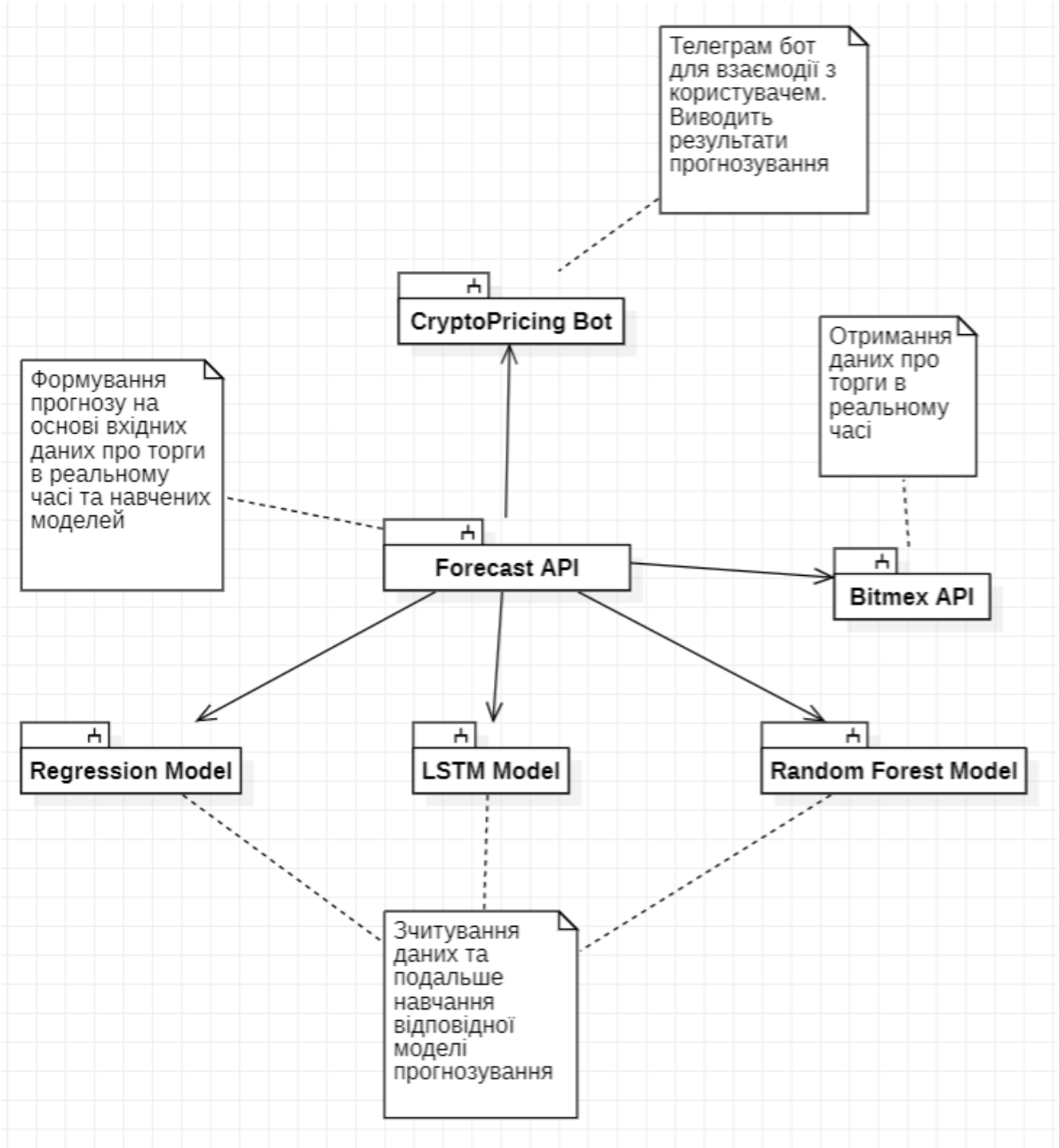

Рисунок 2.22 – Узагальнена архітектура роботи системи прогнозування цін на криптовалюту методами машинного навчання

РОЗДІЛ 3 СТВОРЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, ОПИС ТА РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТУ

3.1 Використання бібліотек Python

Під час роботи було створено та впроваджено систему прогнозування, що дозволяє здійснити прогноз за допомогою різних моделей та вибрати найбільш точну.

Програмний продукт, розроблений на мові програмування Python, є універсальним і може працювати на будь-якій операційній системі, що є великим перевагою для всіх користувачів. Для його використання необхідно лише завантажити середовище та мову програмування на свій комп'ютер.

При розробці були використані такі бібліотеки:

Tensorflow — це бібліотека з відкритим вихідним кодом, розроблена Google переважно для програм глибокого навчання. Він також підтримує традиційне машинне навчання. TensorFlow спочатку була розроблена для великих чисельних обчислень без глибокого навчання. Однак вона також виявилась дуже корисною для розробки програм глибокого навчання, тому Google створила його з відкритим кодом[41].

TensorFlow приймає дані у формі багатовимірних масивів великих розмірів, які називаються тензорами. Багатовимірні масиви дуже зручні в роботі з великими обсягами даних.

TensorFlow працює на основі графів потоку даних, які мають вузли та ребра. Оскільки механізм виконання має форму графіків, набагато простіше виконувати код TensorFlow розподіленим способом у кластері комп’ютерів, використовуючи графічні процесори.

Ta — це Python-бібліотека, призначена для технічного аналізу фінансових ринків. Вона містить набір індикаторів, які можуть бути використані для аналізу фінансових даних та побудови технічних індикаторів, що допоможуть у прийнятті рішень щодо торгівлі на фінансовому ринку.

Технічний аналіз - це метод аналізу фінансових ринків, який використовує історичні цінові та інші дані для прогнозування майбутніх цін

та виявлення тенденцій на ринку. Бібліотека ta містить більше 150 індикаторів, таких як індикатори середньої ціни, індикатори руху ціни, індикатори обсягу тощо.

Завдяки бібліотеці ta користувачі можуть швидко та легко побудувати різні технічні індикатори та використовувати їх для аналізу фінансових даних, таких як історичні ціни акцій, валютні курси, криптовалюти тощо. Бібліотека підтримує популярні фінансові формати даних такі як Pandas DataFrame та NumPy array, що дозволяє користувачам легко інтегрувати її з іншими бібліотеками та інструментами для аналізу даних.

Загалом, бібліотека ta є потужним інструментом для технічного аналізу фінансових ринків в Python, що дозволяє користувачам швидко та ефективно аналізувати та прогнозувати рухи цін на ринку[42].

Pandas — це потужна бібліотека для обробки та аналізу даних у мові програмування Python. Вона надає зручний інтерфейс для роботи з даними у форматах, таких як CSV, Excel, SQL, а також з базами даних. Бібліотека Pandas дозволяє зчитувати, фільтрувати, перетворювати та агрегувати дані з високою ефективністю.

Основною структурою даних у Pandas є DataFrame - таблична структура, що містить рядки та колонки з даними. DataFrame дозволяє легко виконувати різноманітні операції з даними, такі як фільтрація, сортування, групування та агрегація. Крім того, в Pandas є багато інструментів для візуалізації даних, що дозволяє легко створювати графіки та діаграми для вивчення та аналізу даних.

Усі ці можливості роблять Pandas популярним інструментом серед дослідників даних та аналітиків. Бібліотека стала стандартом у своїй області і активно використовується у різних галузях, таких як фінанси, маркетинг, медицина, наука тощо.

Scikit-learn часто називають просто Sklearn - це бібліотека машинного навчання для мови програмування Python. Sklearn надає реалізації багатьох алгоритмів машинного навчання, включаючи класифікацію, регресію, кластеризацію, підготовка даних та вибір моделей[44].

Sklearn містить багато модулів, кожен з яких присвячений певному типу завдань машинного навчання, такі як навчання з учителем (наприклад, класифікація, регресія), навчання без учителя (наприклад, кластеризація, зменшення розмірності) тощо.

Sklearn також містить утиліти для попередньої обробки даних, таких як масштабування, кодування категорійних змінних та заповнення пропущених значень. Крім того, Sklearn має вбудовані функції для побудови моделей та оцінки їхньої продуктивності, що дозволяє користувачам легко порівнювати різні моделі та вибирати найкращу для конкретної задачі.

Взагалі, Sklearn є потужною та корисною бібліотекою для машинного навчання, яка дозволяє легко побудувати та оцінити моделі, що допомагає зробити швидкий та ефективний аналіз даних.

Matplotlib — це бібліотека для візуалізації даних в мові програмування Python. Вона надає широкий спектр інструментів для побудови різноманітних графіків, діаграм, карт тощо. За допомогою Matplotlib можна зобразити дані в різних форматах, включаючи лінійні, кругові, гістограми, розсіювання, стовпчикові діаграми тощо[43].

Matplotlib працює на основі об’єктно-орієнтованого підходу, що дозволяє змінювати будь-який аспект графіку, включаючи заголовки, мітки осей, кольори, стиль ліній та інші параметри. Крім того, Matplotlib дозволяє зберігати графіки у різних форматах, включаючи PNG, PDF, SVG, EPS та ін.

Matplotlib є однією з найпоширеніших бібліотек для візуалізації даних в Python та часто використовується в поєднанні з іншими бібліотеками для аналізу даних, такими як NumPy, Pandas та SciPy[45].

3.2 Вхідні дані з індикаторами для аналізу

Обрана криптовалюта для прогнозування – Bitcoin.

Дослідження включало аналіз щоденних даних з криптовалютної біржі з метою побудови прогнозних моделей для прогнозування ціни закриття

(«Close»). Для досягнення цієї мети було створено декілька моделей різними методами. Крок даних – одна година. На рис. 3.1 наведено дані, які були використані для аналізу.

unix	timestamp	symbol	open	high	low	close	volume	Volume USD
1526364000	2018-05-15 06:00:00	BTC/USD	8733.86000	8796.68000	8707.28000	8740.99000	4906603.14000	559.93000
1526367600	2018-05-15 07:00:00	BTC/USD	8740.99000	8766.00000	8721.11000	8739.00000	2390398.89000	273.58000
1526371200	2018-05-15 08:00:00	BTC/USD	8739.00000	8750.27000	8660.53000	8728.49000	7986062.84000	917.79000
1526374800	2018-05-15 09:00:00	BTC/USD	8728.49000	8754.40000	8701.35000	8708.32000	1593991.98000	182.62000
1526378400	2018-05-15 10:00:00	BTC/USD	8708.32000	8865.00000	8695.11000	8795.90000	11101273.74000	1260.69000
1526382000	2018-05-15 11:00:00	BTC/USD	8795.90000	8821.19000	8740.54000	8760.00000	2842987.30000	324.11000
1526385600	2018-05-15 12:00:00	BTC/USD	8760.00000	8798.60000	8741.52000	8760.00000	2842401.44000	324.20000
1526389200	2018-05-15 13:00:00	BTC/USD	8760.00000	8771.02000	8700.00000	8759.23000	6380081.87000	730.67000
1526392800	2018-05-15 14:00:00	BTC/USD	8759.23000	8761.70000	8467.42000	8517.77000	11484106.74000	1341.75000
1526396400	2018-05-15 15:00:00	BTC/USD	8517.77000	8553.67000	8477.89000	8520.30000	6747071.92000	792.07000
1526400000	2018-05-15 16:00:00	BTC/USD	8520.30000	8589.10000	8500.78000	8535.23000	3298228.22000	385.69000
1526403600	2018-05-15 17:00:00	BTC/USD	8535.23000	8579.48000	8500.00000	8546.34000	4058869.11000	475.08000
1526407200	2018-05-15 18:00:00	BTC/USD	8546.34000	8555.51000	8400.00000	8526.65000	7443239.63000	877.26000
1526410800	2018-05-15 19:00:00	BTC/USD	8526.65000	8583.60000	8451.05000	8555.60000	5703980.92000	670.49000
1526414400	2018-05-15 20:00:00	BTC/USD	8555.60000	8569.92000	8461.38000	8512.10000	5192991.59000	610.97000
1526418000	2018-05-15 21:00:00	BTC/USD	8512.10000	8555.54000	8481.24000	8492.28000	1440288.06000	168.99000
1526421600	2018-05-15 22:00:00	BTC/USD	8492.28000	8557.51000	8420.37000	8460.00000	2591379.77000	305.92000
1526425200	2018-05-15 23:00:00	BTC/USD	8460.00000	8505.00000	8450.00000	8474.99000	1152835.45000	136.05000
1526428800	2018-05-16 00:00:00	BTC/USD	8474.99000	8505.33000	8430.94000	8455.92000	1306049.94000	154.26000
1526432400	2018-05-16 01:00:00	BTC/USD	8455.92000	8478.29000	8275.02000	8326.28000	8931481.69000	1070.93000

Рисунок 3.1 Вхідні дані

Тепер за допомогою бібліотеки додано основні індикатори, які будуть допомагати виконувати прогнозування:

RSI (Relative Strength Index)

unix	timestamp	symbol	open	high	low	close	volume	Volume USD	ma	rsi	bb_middle	bb_upper	bb_lower
1526472000	2018-05-16 12:00:00	BTC/USD	8373.00000	8399.10000	8356.40000	8365.08000	3031532.02000	362.01000	8356.50450	45.27418	8356.50450	8628.04793	8084.96107
1526475600	2018-05-16 13:00:00	BTC/USD	8365.08000	8381.73000	8145.10000	8179.99000	8614269.80000	1046.38000	8338.18700	32.80984	8338.18700	8605.42787	8070.94613
1526479200	2018-05-16 14:00:00	BTC/USD	8179.99000	8229.91000	8173.60000	8225.00000	2936522.28000	357.91000	8323.10450	37.32840	8323.10450	8579.94366	8066.26534
1526482800	2018-05-16 15:00:00	BTC/USD	8225.00000	8377.29000	8211.80000	8285.80000	4663319.71000	561.20000	8309.61450	42.91323	8309.61450	8543.50748	8075.72152
1526486400	2018-05-16 16:00:00	BTC/USD	8285.80000	8329.10000	8201.00000	8273.40000	2665411.47000	322.53000	8297.67950	42.08944	8297.67950	8512.61756	8082.74144
1526490000	2018-05-16 17:00:00	BTC/USD	8273.40000	8350.00000	8250.90000	8297.40000	8926209.14000	1073.71000	8287.93550	44.31745	8287.93550	8483.49811	8092.37289
1526493600	2018-05-16 18:00:00	BTC/USD	8297.40000	8346.95000	8266.46000	8279.89000	1973143.22000	237.21000	8278.93000	43.01710	8278.93000	8457.84910	8100.01090
1526497200	2018-05-16 19:00:00	BTC/USD	8279.89000	8377.29000	8252.72000	8252.72000	2086318.50000	251.20000	8267.81650	41.00650	8267.81650	8422.63089	8113.00211
1526500800	2018-05-16 20:00:00	BTC/USD	8252.72000	8311.35000	8232.14000	8302.18000	2971208.90000	359.73000	8260.12950	45.95830	8260.12950	8390.09380	8130.16520
1526504400	2018-05-16 21:00:00	BTC/USD	8302.18000	8319.96000	8268.37000	8295.85000	972462.50000	117.23000	8258.60800	45.43269	8258.60800	8386.12848	8131.08752
1526508000	2018-05-16 22:00:00	BTC/USD	8295.85000	8398.24000	8277.25000	8277.25000	3030639.92000	363.17000	8265.79900	43.84590	8265.79900	8379.77369	8151.82431
1526511600	2018-05-16 23:00:00	BTC/USD	8277.25000	8372.58000	8277.25000	8346.59000	1784063.07000	214.18000	8274.88900	50.75148	8274.88900	8384.08877	8165.68923
1526515200	2018-05-17 00:00:00	BTC/USD	8346.59000	8399.99000	8314.78000	8358.30000	1436126.50000	171.64000	8281.33500	51.82885	8281.33500	8394.18734	8168.48266
1526518800	2018-05-17 01:00:00	BTC/USD	8358.30000	8386.53000	8314.77000	8375.10000	2160023.63000	258.58000	8289.07000	53.40377	8289.07000	8405.31152	8172.82848
1526522400	2018-05-17 02:00:00	BTC/USD	8375.10000	8400.00000	8350.07000	8368.62000	2509694.55000	299.52000	8295.56100	52.68823	8295.56100	8414.32036	8176.80164
1526526000	2018-05-17 03:00:00	BTC/USD	8368.62000	8500.00000	8331.39000	8366.72000	6048203.82000	718.98000	8302.24750	52.46625	8302.24750	8421.22078	8183.27422
1526529600	2018-05-17 04:00:00	BTC/USD	8366.72000	8381.00000	8314.77000	8314.77000	1000211.56000	119.70000	8307.61200	46.67593	8307.61200	8418.40332	8196.82068
1526533200	2018-05-17 05:00:00	BTC/USD	8314.77000	8371.30000	8275.45000	8363.70000	3379135.19000	405.25000	8313.62450	52.04424	8313.62450	8422.87621	8204.37279
1526536800	2018-05-17 06:00:00	BTC/USD	8363.70000	8382.41000	8335.49000	8344.16000	1761990.38000	210.77000	8312.27600	49.88444	8312.27600	8419.29829	8205.25371
1526540400	2018-05-17 07:00:00	BTC/USD	8344.16000	8351.00000	8191.14000	8247.40000	5919821.52000	714.82000	8305.99600	40.84509	8305.99600	8412.76826	8199.22374
1526544000	2018-05-17 08:00:00	BTC/USD	8247.40000	8292.99000	8227.70000	8277.40000	1784788.58000	215.97000	8301.61200	44.21999	8301.61200	8405.48115	8197.74285
1526547600	2018-05-17 09:00:00	BTC/USD	8277.40000	8325.80000	8264.13000	8291.67000	1361911.38000	164.20000	8307.19600	45.80388	8307.19600	8395.09060	8219.30140

Рисунок 3.2 Вхідні дані з індикаторами

MACD (Moving Average Convergence Divergence)

Bollinger Bands

Наступним кроком виконаємо розбиття вхідних даних на тестову та навчальну вибірку – навчальна 80%, тестова 20%

```
train_size = int(len(df) * 0.8)
train_features = X[:train_size]
train_target = y[:train_size]
test_features = X[train_size:]
test_target = y[train_size:]
```

Рисунок 3.3 Розбиття вибірки на тестову та навчальну

Також для деяких методів будемо виконувати нормалізацію даних:

Усі ознаки підсумкового набору даних лежать у різних інтервалах, що заважає коректній роботі моделі. Щоб вирішити цю проблему достатньо нормалізувати всі ознаки. Для цього для кожного стовпця підсумкового набору даних достатньо застосувати формулу:

$$X_{std} = \frac{X - min(X)}{max(X) - min(X)},$$

де X - вихідне значення (до нормалізації),

X_{std} – значення після нормалізації,

$min(X), max(X)$ - мінімальне і максимальне значення ознаки відповідно.

3.3 Навчання моделей та порівняння результатів

Свічки - це графічний інструмент для візуального відображення цінових рухів на фінансових ринках, включаючи криптовалютні ринки. Їх також називають японськими свічками, оскільки вперше були використані японськими трейдерами в XVIII столітті для прогнозування цінових рухів на ринках рису та інших сільськогосподарських товарів.

Кожна свічка представляє ціновий рух активу за певний період часу (наприклад, 1 день), включаючи ціну відкриття, максимальну та мінімальну ціни та ціну закриття. Ціна відкриття та ціна закриття представлені на графіку

як тіло свічки, а верхня та нижня тінь свічки відображають максимальну та мінімальну ціни активу за цей період.

Свічки зазвичай забарвлюють у зелений колір, якщо ціна закриття вище ціни відкриття, та червоний колір, якщо ціна закриття нижче ціни відкриття. Таким чином, графік свічок дозволяє трейдерам легше аналізувати цінові рухи та приймати рішення щодо купівлі або продажу активу.

Рисунок 3.3 «Свічки»

Для вирішення завдання прогнозування ціни ми будемо прогнозувати суму, яка буде при закритті певної свічки

Для реалізації було обрано три варіанта методів машинного навчання:

- Регресія
- Нейронна мережа (LSTM)
- Дерева рішень (Random forest)

Кожен метод отримує на вхід однакову навчальну та тестову вибірки.

Регресія – спочатку зменшуємо кількість ознак а потім застосовуємо лінійну регресію.

Було проведено підбір найкращих параметрів шляхом прогону на різних значеннях параметрів та представлено найкращі з них.

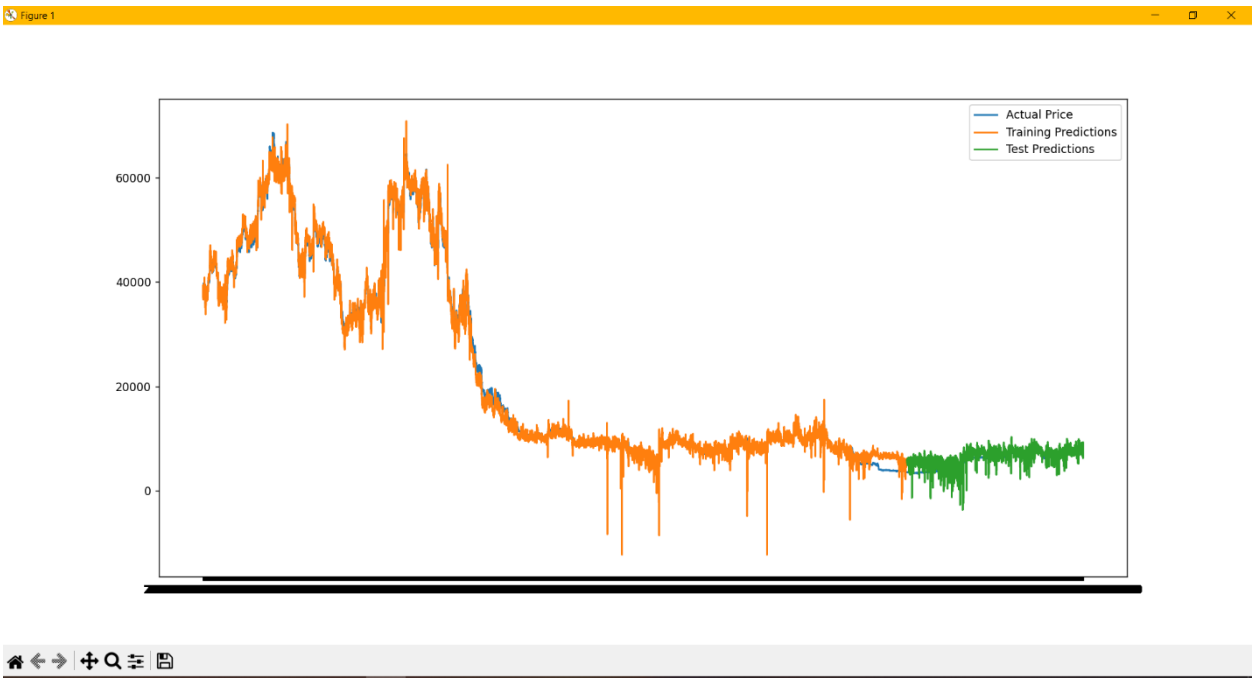

Рисунок 3.4 Графік прогнозування для методу регресії

Оцінка моделі:

```
The root mean squared error is 7596.233758912045.
The MSE is 57702767.320035025
The MAE is 3224.125658895603
The R2_Score is 0.3215696077846464
```

Рисунок 3.5 Оцінка моделі регресії

Як бачимо навіть на тестових даних модель навчилася досить погано, наочно це видно з оцінки r2_Score, яка дорівнює 0.32 з максимума 1.

LSTM

Було проведено підбір найкращих параметрів шляхом прогону на різних значеннях параметрів та представлено найкращі з них.

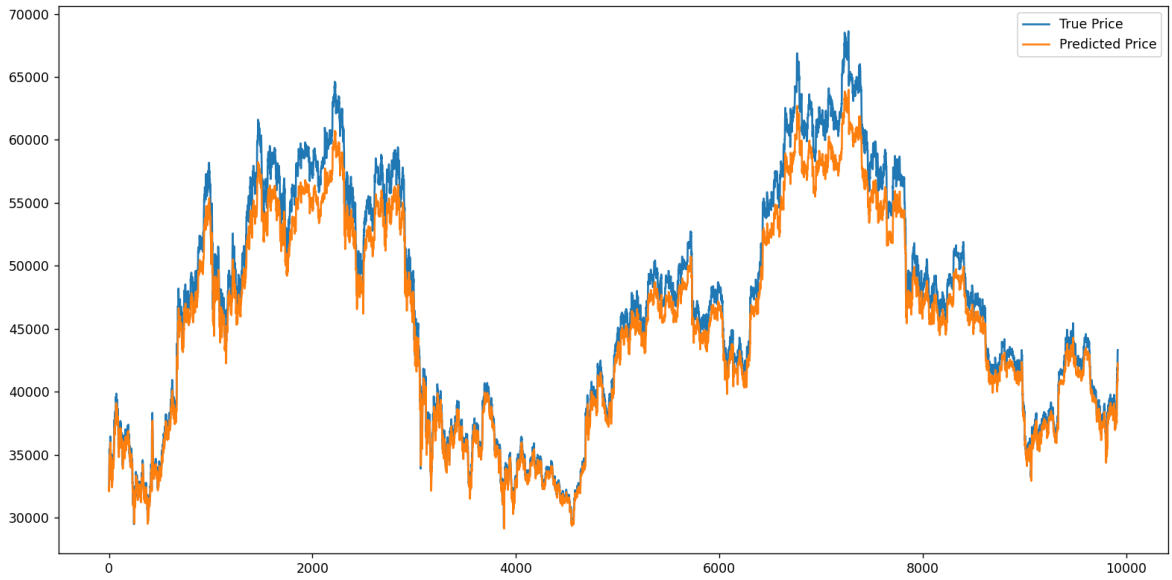

Рисунок 3.6 Графік прогнозування для моделі LSTM

```
The root mean squared error is 2256.1450069942.
The MSE is 5090190.292584857
The MAE is 1983.2957111281257
The R2_Score is 0.9420015899770303
```

Рисунок 3.7 Оцінка моделі LSTM

Модель навчилася досить добре, оцінки кращі порівняно з попередньою моделлю.

Дерева рішень (Random forest)

Було проведено підбір найкращих параметрів шляхом прогону на різних значеннях параметрів та представлено найкращі з них.

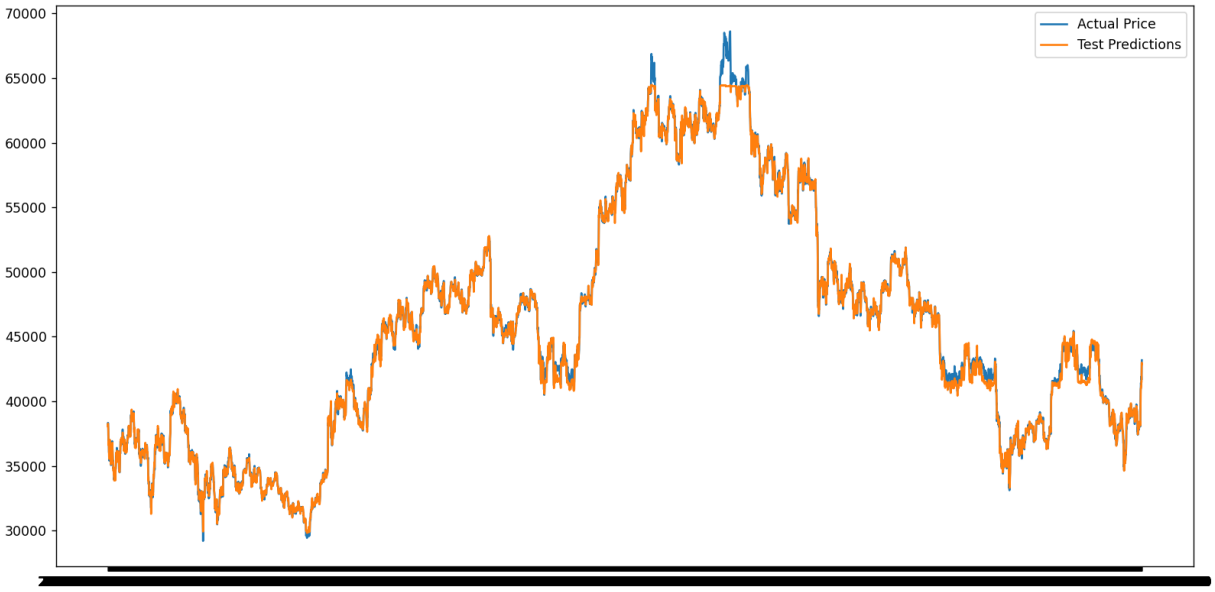

Рисунок 3.8 Графік прогнозування для методу Random forest

```
The root mean squared error is 408.0643700648151.
The MSE is 166516.53011639434
The MAE is 230.49764052500961
The R2_Score is 0.99803775400242
```

Рисунок 3.9 Оцінка моделі Random forest

Загальні результати навчання моделей представлено в табл. 3.1.

Таблиця 3.1

Метод	RMSE	MSE	MAE	R2_Score
Регресія	7596,233	57702767,320	3224,125	0,321
Нейронна мережа (LSTM)	2256,145	5090190,292	1983,295	0,942
Дерева рішень (Random forest)	408,064	166516,530	230,497	0,998

З таблиці видно, що для даного набору даних найкраще показав себе метод дерев рішень (Random forest), оцінка точності для цього методу – 0,998.

Отже, моделі навчено на вхідному наборі даних. З результатів навчання видно, що модель звичайної лінійної регресії погано виконує поставлену задачу. Моделі LSTM та RandomForest виявилися кращими та на навчальних і

тестових вибірках дають гарну точність. Проте це не означає, що вони однаково добре покажуть себе на реальних даних.

3.4 Опис розробленого застосунку

Для використання отриманої на попередньому кроці моделі, застосуємо її для передбачення ціни на криптовалюту Bitcoin в реальному часі. Для цього було обрано криптобіржу Bitmex.

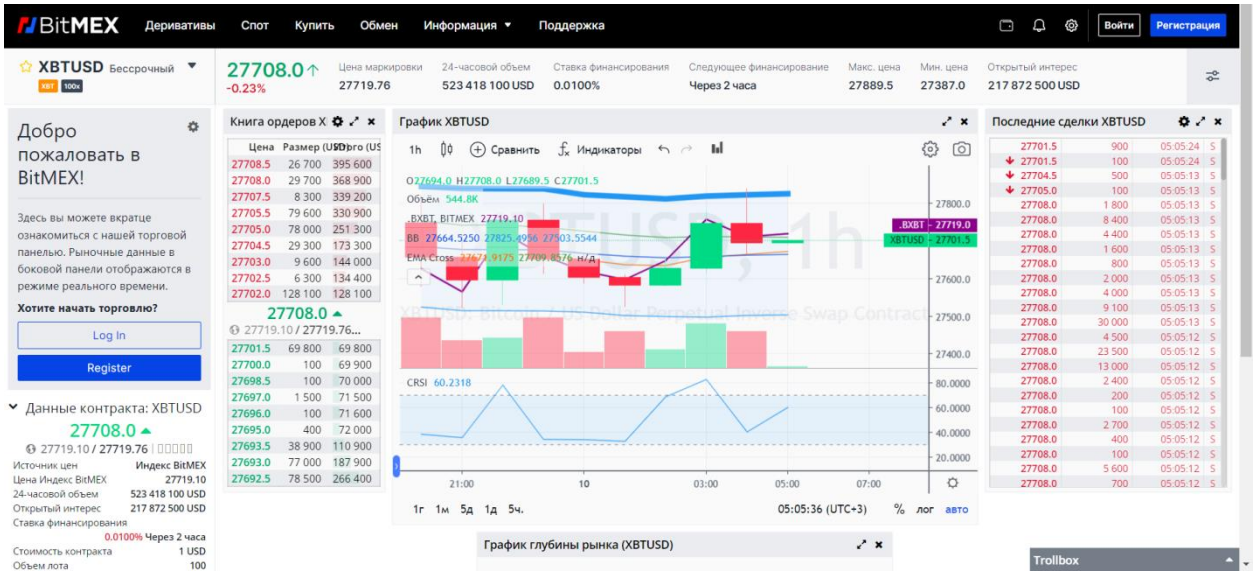

Рисунок 3.10 Bitmex

Обрана біржа має можливість доступу через API. API - Application Programming Interface, що означає програмний інтерфейс програми. У контексті API слово «додаток» стосується будь-якого ПЗ з певною функцією. Інтерфейс можна розглядати як сервісний контракт між двома програмами. Цей контракт визначає, як вони взаємодіють одна з одною, використовуючи запити та відповіді. Документація API містить інформацію про те, як розробники повинні структурувати ці запити та відповіді.

BitMEX пропонує багатофункціональний REST API та потужний ресурс потокового з'єднання WebSocket API. Всі ринкові дані та дані користувача доступні та оновлюються в режимі реального часу.

API-інтерфейси BitMEX відкриті та повністю завершені. Кожна функція веб-сайту BitMEX доступна через API, що дозволяє розробникам створювати

будь-які програми на базі BitMEX та повністю ними управляти. Повнота API-інтерфейсу BitMEX є найкращою у своєму класі та унікальною у торговому просторі.

Для прогнозування в режимі реального часу нам потрібно отримати інформацію про ціну криптовалюти в даний момент часу. Використаємо один з методів API, а саме «get_bucket_trades», який повертає масив з даними про деяку кількість останніх «свічок».

Виконаємо підключення до API через згенерований раніше ключ:

```
api_key = secret.api_key
api_secret = secret.api_secret

btm = bitmexAPI.Bitmex(api_key, api_secret, False)

SYMBOL = 'XBTUSD'
LEVERAGE = 10
```

Рисунок 3.11 – Підключення до API

Використаємо API та отримаємо дані про свічку:

```
def get_market_history(binSize = '5m', count=100):
    msg = btm.get_bucket_trades(binSize=binSize, partial=False, symbol=SYMBOL, count=count, reverse='true')
    return msg
```

Рисунок 3.12 – Отримання даних про останню свічку

Відповідь сервера буде мати такий вигляд:

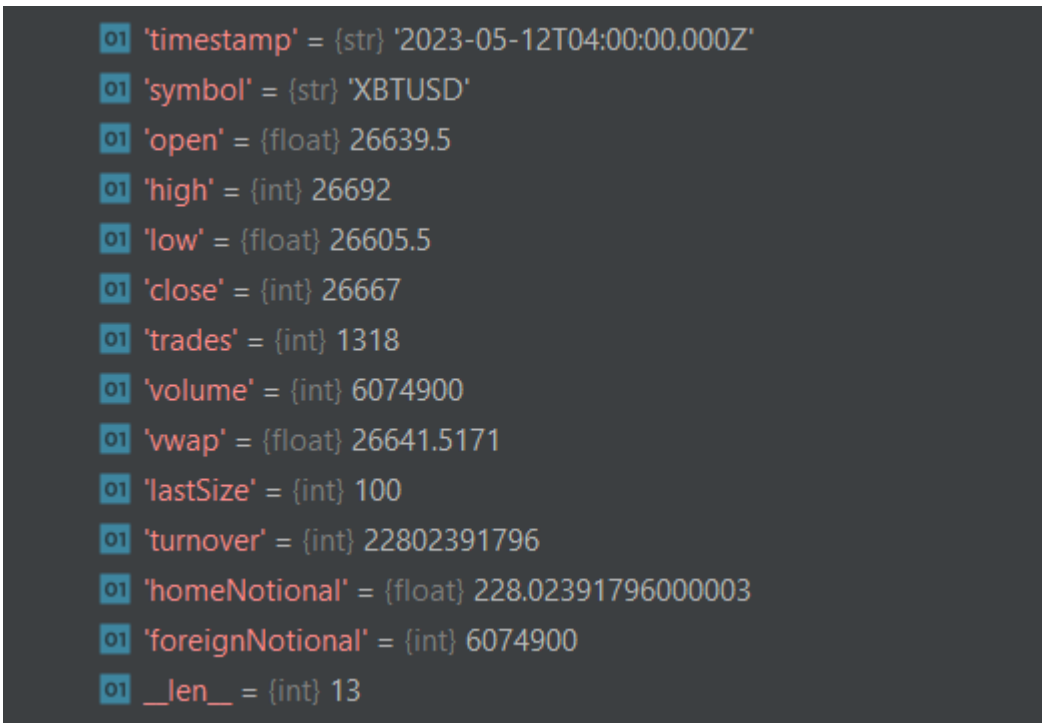

Рисунок 3.13 – Відповідь сервера

Тепер отримані з сервера дані обробимо, додамо індикатори для аналізу, та видалимо поля, які нам не потрібні:

```
df_mh['timestamp'] = df_mh['timestamp'].map(str_to_utc)
df_mh = df_mh.set_index('timestamp')
df_mh = df_mh.sort_index()
df_mh['momentum_rsi'] = ta.momentum.rsi(df_mh['close'], window=5)
df_mh['ma'] = ta.trend.sma_indicator(df_mh['close'], window=20)
df_mh['rsi'] = ta.momentum.rsi(df_mh['close'])
indicator_bb = BollingerBands(close=df_mh["close"], window=20, window_dev=2)
df_mh['bb_middle'] = indicator_bb.bollinger_mavg()
df_mh['bb_upper'] = indicator_bb.bollinger_hband()
df_mh['bb_lower'] = indicator_bb.bollinger_lband()
col = df_mh[-1:]
err = [np.datetime_as_string(col.index.values[0], unit='s'),col['open'][0]]
X = col.drop(['close'], axis=1)
# X = X.drop(['timestamp'], axis=1)
X = X.drop(['foreignNotional'], axis=1)
X = X.drop(['homeNotional'], axis=1)
X = X.drop(['lastSize'], axis=1)
X = X.drop(['momentum_rsi'], axis=1)
X = X.drop(['trades'], axis=1)
X = X.drop(['symbol'], axis=1)
X = X.drop(['turnover'], axis=1)
X = X.drop(['vwap'], axis=1)
```

Рисунок 3.14 – Обробка повідомлення

Тепер завантажимо раніше навчену модель та передамо їй отримане повідомлення:

```
rf_model = joblib.load('rf_model.pkl')
y_pred1 = rf_model.predict(X)
```

Рисунок 3.15 – Прогнозування значення

Всі описані вище дії винесено в окрему функцію з назвою refresh.

Тепер налаштуємо інтерфейс взаємодії з користувачем.

Для повернення прогнозу користувачу реалізуємо бот в месенджері Telegram. Для реалізації використаємо бібліотеку telebot.

Telebot - це бібліотека для мови програмування Python, яка дозволяє створювати та налаштовувати боти для месенджера Telegram. Ця бібліотека дозволяє програмістам легко створювати різноманітні боти, що можуть виконувати різні функції, такі як надсилання повідомлень, відповідей на запити користувачів, отримання та надсилання фото, відео та документів, а також інші функції.

Для роботи з telebot потрібно створити бота в Telegram та отримати токен API. Після цього можна встановити бібліотеку telebot та почати розробку свого бота. Програміст може налаштувати бота для відповіді на різні команди, відповідати на повідомлення, реагувати на події, такі як приєднання користувача до чату, та багато іншого.

Бібліотека telebot має дуже зрозумілу та просту документацію, яка дозволяє швидко ознайомитися зі всіма можливостями цієї бібліотеки та розробити власного бота для Telegram. Також, дуже зручною є можливість використовувати telebot для навчання моделей машинного навчання, наприклад, для розпізнавання тексту чи зображень та надсилання результату користувачу у чаті.

Користувачу для отримання прогнозу потрібно просто зайти в так званий чат з ботом та розпочати роботу бота. Тепер користувач буде

отримувати повідомлення з прогнозованим значенням кожні півгодини. Для зупинки бота потрібно надіслати йому команду про завершення.

Тепер перевіримо результат роботи нашої програми. Кожних 30 хвилни користувачу буде надходити повідомлення, яке міститиме дані про активну в даний момент свічку Біткоїна та прогноз на її закриття. Користувач буде мати змогу прийняти для себе рішення, купувати, продавати чи утримувати криптовалюту Біткоїн.

Результат:

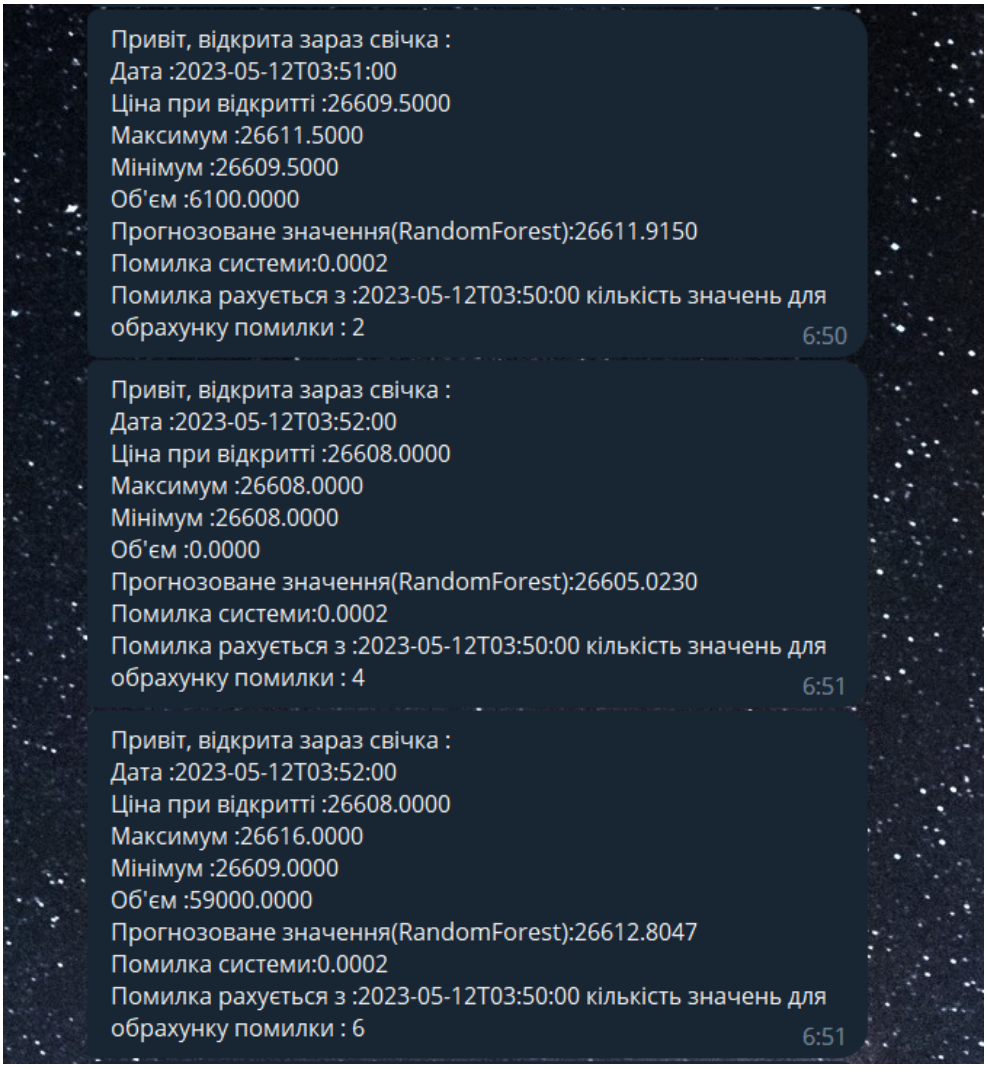

Рисунок 3.16 Повідомлення користувачу

Помилка системи на реальних значеннях 0,002, таке значення через малу вибірку часу. Також слід враховувати, що для такого короткострокового пронозування навіть найменша помилка в прогнозі може значно вплинути на результат, адже для торгів це дуже важливо.

На рисунку 3.17 зображено графік помилки прогнозування на даних в реальному часі за період від 5 години ранку 9 травня 2023 року до 17 години 11 травня 2023 року:

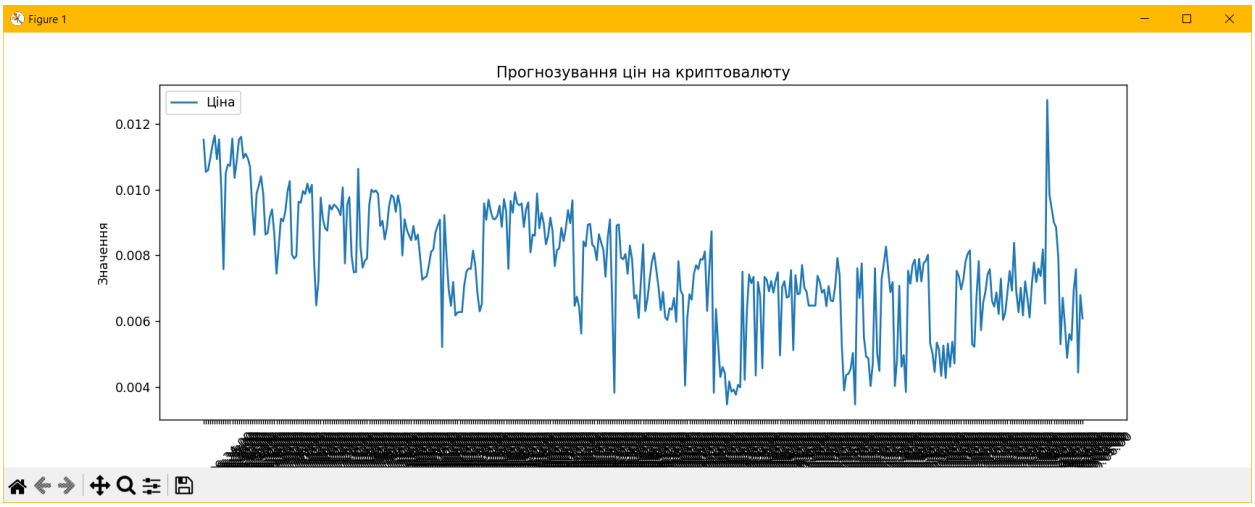

Рисунок 3.17 Помилка на реальних даних

Рис. 3.18 показує відношення між реальними (синя лінія) та прогнозованими значеннями (оранжева лінія) за допомогою методу Random Forest за цей же період:

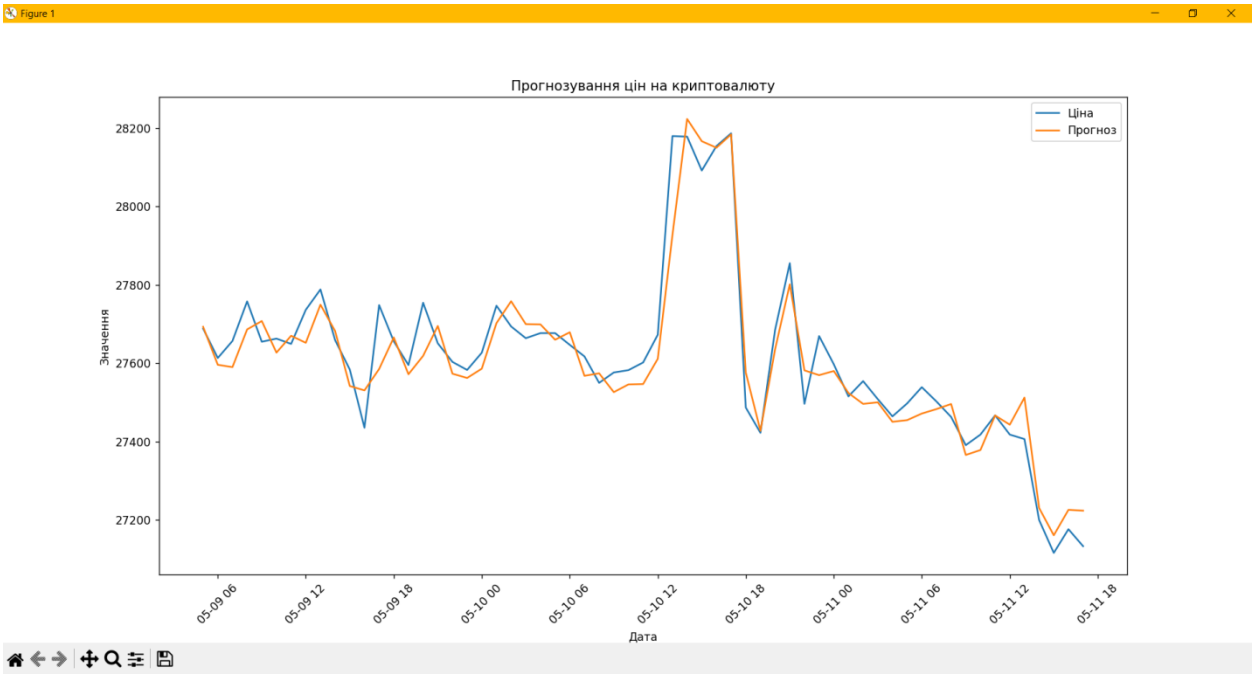

Рисунок 3.18 Відношення прогнозу та реальної ціни

В табл. 3.2 наведено числові значення, які було показано на рис. 3.18.

Таблиця 3.2

Дата	Реальна ціна	Прогноз
2023-05-09T05:00:00	27688,5	27693,17
2023-05-09T06:00:00	27613,5	27596,12
2023-05-09T07:00:00	27657,0	27590,24
2023-05-09T08:00:00	27758,0	27686,41
2023-05-09T09:00:00	27655,0	27707,87
2023-05-09T10:00:00	27663,0	27627,25
2023-05-09T11:00:00	27649,5	27670,22
2023-05-09T12:00:00	27736,5	27652,3
2023-05-09T13:00:00	27788,5	27749,64
2023-05-09T14:00:00	27659,0	27682,24
2023-05-09T15:00:00	27584,0	27541,96
2023-05-09T16:00:00	27435,5	27530,84
2023-05-09T17:00:00	27748,5	27585,35
2023-05-09T18:00:00	27656,0	27665,98
2023-05-09T19:00:00	27595,5	27571,99
2023-05-09T20:00:00	27754,5	27618,95
2023-05-09T21:00:00	27651,0	27695,31
2023-05-09T22:00:00	27603,5	27573,41
2023-05-09T23:00:00	27583,0	27562,51
2023-05-10T00:00:00	27627,0	27586,23
2023-05-10T01:00:00	27747,0	27702,4
2023-05-10T02:00:00	27694,0	27758,35
2023-05-10T03:00:00	27664,0	27699,91
2023-05-10T04:00:00	27677,0	27699,22
2023-05-10T05:00:00	27677,0	27660,09
2023-05-10T06:00:00	27647,0	27679,15
2023-05-10T07:00:00	27617,5	27567,99
2023-05-10T08:00:00	27550,0	27574,44
2023-05-10T09:00:00	27576,5	27526,33
2023-05-10T10:00:00	27582,5	27546,05
2023-05-10T11:00:00	27602,0	27546,93
2023-05-10T12:00:00	27673,0	27611,48
2023-05-10T13:00:00	28180,0	27927,45
2023-05-10T14:00:00	28178,5	28223,69
2023-05-10T15:00:00	28092,0	28166,81
2023-05-10T16:00:00	28154,5	28150,3
2023-05-10T17:00:00	28187,0	28184,24
2023-05-10T18:00:00	27487,0	27576,47
2023-05-10T19:00:00	27422,5	27427,67
2023-05-10T20:00:00	27685,5	27635,92
2023-05-10T21:00:00	27855,5	27801,53
2023-05-10T22:00:00	27496,5	27581,89
2023-05-10T23:00:00	27669,5	27569,67
2023-05-11T00:00:00	27597,5	27580,24
2023-05-11T01:00:00	27515,5	27524,08
2023-05-11T02:00:00	27554,5	27496,35
2023-05-11T03:00:00	27508,5	27500,37

Дата	Реальна ціна	Прогноз
2023-05-11T04:00:00	27464,5	27450,61
2023-05-11T05:00:00	27497,5	27454,84
2023-05-11T06:00:00	27539,0	27471,65
2023-05-11T07:00:00	27502,5	27483,08
2023-05-11T08:00:00	27463,0	27496,05
2023-05-11T09:00:00	27391,0	27365,95
2023-05-11T10:00:00	27418,0	27378,58
2023-05-11T11:00:00	27467,0	27466,83
2023-05-11T12:00:00	27418,0	27443,29
2023-05-11T13:00:00	27406,5	27512,45
2023-05-11T14:00:00	27200,0	27230,7
2023-05-11T15:00:00	27116,0	27160,91
2023-05-11T16:00:00	27176,5	27225,89
2023-05-11T17:00:00	27133,0	27223,75

Вищенаведені результати свідчать про досить точний прогноз за допомогою методу Random Forest та досить невелику похибку прогнозування.

ВИСНОВКИ

У кваліфікаційній роботі здійснено теоретичний аналіз та вирішення науково-прикладної проблеми стосовно використання методів машинного навчання для прогнозування ціни на криптовалюту Bitcoin.

За результатами виконаної роботи зроблено такі висновки:

1. Проаналізовано літературні джерела щодо використання методів машинного навчання для прогнозування ціни на криптовалюту Bitcoin. На підставі цього аналізу було доведено важливість обраної теми та сформульовано постановку задачі.
2. Зроблено опис декілька методів машинного навчання, які можуть бути використані для розв'язання задачі, а саме поліноміальну регресію, нейронні мережі та Random forest. Для кожного з методів були розглянуті математичні основи та наведено опис основних параметрів та алгоритмів, які вони використовують.
3. Розглянуто різні типи вхідних даних, які можна використати для розв’язання задачі, а також додаткові інструменти, що полегшують завдання, наприклад, індикатори.
4. Описано фунції системи, які повинні виконуватися та розроблено архітектуру створюваної системи.
5. Проведено попередній аналіз отриманих даних за допомогою візуалізації. Декілька варіантів архітектури методів машинного навчання, які були описані раніше, були використані для оцінки точності прогнозування на основі створеної вибірки даних. З отриманої оцінки було обґрунтовано вибір архітектури для побудови кінцевого програмного забезпечення.
6. Здійснено експериментальне прогнозування цін на Bitcoin за 2017-2022 роки в розрізі години за допомогою кожного з методів машинного навчання. В результаті була отримана точність прогнозування для

кожного методу: 32% для поліноміальної регресії, 92% для нейронних мереж та 99% для Random forest.

7. На основі аналізу отриманих експериментальних результатів показано, що для обраної криптовалюти та вхідних даних найоптимальнішими являються моделі на основі Random forest, які дали набагато точніший результат порівняно з іншими методами.
8. Розроблено програмний додаток, який використовує моделі на основі машинного навчання для прогнозування цін на криптовалюту Bitcoin та API Telegram для взаємодії з користувачем. Для створення програмного додатку використано мову високого рівня Python та набір бібліотек: Numpy, Pandas, Scikit-learn, TensorFlow.
9. За допомогою методу Random Forest отримано досить точний результат прогнозування.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Singh, M., & Dixit, S. (2018). Bitcoin Price Prediction Using Machine Learning: An Approach to Sample Dimension Engineering. IEEE Access, 6, 4979-4992. https://ieeexplore.ieee.org/document/8294188

2. Alqahtani, A., Albeshri, A., Alreshidi, M., Alsalhi, N., & Alghamdi, M. (2021). Forecasting Cryptocurrency Prices with Machine Learning Techniques. Journal of Open Innovation: Technology, Market, and Complexity, 7(1), 11. https://doi.org/10.3390/joitmc7010011

3. Joseph, K., Biswas, A., & Kandpal, C. (2019). Bitcoin Price Prediction Using LSTM Neural Network. International Journal of Engineering and Advanced Technology (IJEAT), 8(6S), 238-242. https://www.ijeat.org/wp-content/uploads/papers/v8i6S/F30430886S19.pdf

4. Nguyen, L. T., Nguyen, H. T., Nguyen, T. H., Nguyen, Q. T., & Nguyen, T. H. (2020). Forecasting Bitcoin Price Fluctuation with Twitter Sentiment Analysis. IEEE Access, 8, 128380-128390. https://ieeexplore.ieee.org/document/9163429

5. Qadir, J., Ahmad, M. S., & Kang, B. H. (2019). Predicting Cryptocurrency Prices with Deep Learning. IEEE Access, 7, 67749-67759. https://ieeexplore.ieee.org/document/8739244

6. Euny H. How Does Bitcoin Mining Work?. 30.11.2021. URL: https://www.investopedia.com/tech/how-does-bitcoin-mining-work/ (last accessed: 10.03.2023)

7. Blockchain.info. Total Circulating Bitcoin. URL: https://www.blockchain.com/charts/total-bitcoins (дата звернення: 21.04.2023)

8. De Vries A. Bitcoin's growing energy problem. URL: https://www.sciencedirect.com/science/article/pii/S2542435118301776 (last accessed 26.04.2023)

9. Library of Congress. Our New Reports on Regulation of Cryptocurrency Around the World. URL: https://blogs.loc.gov/law/2018/07/our-new-reports-onregulation-of-cryptocurrency-around-the-world/ (last accessed: 21.04.2023)

10. Bitcoin.org “FAQ”, 2009. URL: https://bitcoin.org/en/faq (дата звернення: 21.04.2023)
11. Hayes A. Cryptocurrency value formation: An empirical study leading to a cost of production model for valuing bitcoin. URL: https://www.sciencedirect.com/science/article/abs/pii/S0736585315301118 (last accessed: 09.04.2023)
12. "Методи машинного навчання в аналізі даних" - М. Ясновський, І. Малков, О. Стрільчук (Київ, 2014)
13. D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2017.
14. Hastie, T., Tibshirani, R., Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer.
15. Chatterjee, S., Hadi, A. (2012). Regression Analysis by Example. John Wiley & Sons.
16. "Машинне навчання" - О. Купрій, О. Каньовський (Київ, 2019).
17. Montgomery, D. (2012). Introduction to Linear Regression Analysis. John Wiley & Sons.
18. "Методи і моделі штучного інтелекту" - В. Гордієнко, А. Литвинов, О. Савельєв (Київ, 2012).
19. "Моделі регресійного аналізу" - І. Гарін, О. Козачок (Львів, 2017).
20. "Методи регресійного аналізу з додатками в R" - О. Мельничук, В. Гаврилюк, В. Грицюк (Львів, 2020).
21. Towards data science. “Overfitting vs. Underfitting: A Complete Example”: https://towardsdatascience.com/overfitting-vs-underfitting-a-complete-example-d05dd7e19765
22. "Нейронні мережі" - О. Лебедєв, О. Синявський (Київ, 2017).
23. "Машинне навчання: моделі, алгоритми та методи" - Ю. Горбатюк, Є. Шинкаренко, О. Хріпко (Київ, 2021).
24. "Нейромережеві технології: теорія та практика" - І. Зеленський, І. Лаврова, С. Петрішин (Львів, 2010).
25. Goodfellow, I., Bengio, Y., Courville, A. (2016). Deep Learning. MIT Press.

26. Bishop, C. (2006). Pattern Recognition and Machine Learning. Springer.
27. Nielsen, M. (2015). Neural Networks and Deep Learning. Determination Press.
28. Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5-32.
29. Friedman, J., Hastie, T., Tibshirani, R. (2001). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer.
30. Zhou, Z., Liu, H. (2012). Ensemble Methods: Foundations and Algorithms. CRC Press.
31. "Ансамблеві методи машинного навчання" - О. Красоткін, М. Пилипчук (Київ, 2014).
32. "Методи ансамблевого навчання" - О. Шкварчук, А. Попенко, А. Труш (Київ, 2017).
32. Lu, Y., & Kwon, J. (2019). Predicting stock price using LSTM model with attention mechanism. In 2019 10th IEEE International Conference on Software Engineering and Service Science (ICSESS) (pp. 740-743). IEEE.
33. Львівський національний університет імені Івана Франка. Машинне навчання простими словами. Частина 1. – Режим доступу до електронного ресурсу: http://www.mmf.lnu.edu.ua/ar/1739
34. T. Segal, “Fundamental analysis https://www.investopedia.com/terms/f/fundamentalanalysis.asp,” Apr 2021.
35. S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” Neural Computation, vol. 9, pp. 1735–1780, 11 1997.
36. Huang, H., Li, J., & Chen, X. (2020). Stock price prediction using a hybrid LSTM-GAN model. Journal of Intelligent & Fuzzy Systems, 38(3), 2893-2902.
37. A. Graves and J. Schmidhuber, “Framewise phoneme classification with bidirectional lstm and other neural network architectures,” Neural Networks, vol. 18, no. 5, pp. 602–610, 2005. IJCNN 2005.
38. C. Olah, “Understanding lstm networks,” 2015.
40. I. Rodriguez-Lujan, R. Huerta, C. Elkan, and C. S. Cruz, “Quadratic programming feature selection,” J. Mach. Learn. Res., vol. 11, p. 1491– 1516, Aug. 2010. 36

41. S. Lundberg and S. Lee, “A unified approach to interpreting model predictions,” CoRR, vol. abs/1705.07874, 2017.
42. “Shap python library, https://github.com/slundberg/shap.”
43. M. Hansson, “On stock return prediction with lstm networks,” 2017. Student Paper.
44. Chen, C. Y., Tseng, H. L., & Lai, C. F. (2021). Predicting stock prices using machine learning algorithms with feature selection techniques. Applied Sciences, 11(2), 802.
45. V. Cerqueira, L. Torgo, and I. Mozetič, “Evaluating time series forecasting models: an empirical study on performance estimation methods,” Machine Learning, vol. 109, pp. 1997–2028, Nov 2020.

ДОДАТКИ

Додаток А

Лістинг коду:

Бот:

```
import json
import math
from sklearn.preprocessing import PolynomialFeatures
import secret
import bitmexAPI_test as btm
from ta import *
import ta
from ta.volatility import BollingerBands
import numpy as np
import pandas as pd
from datetime import datetime
import time
import telebot
import joblib
from sklearn.metrics import mean_absolute_percentage_error, mean_squared_error

bot = telebot.TeleBot('1387263829:AAH0sZJ7OYVbGKMwTOCkFViHvpxP7B7dROo')

def str_to_utc(str, mask = '%Y-%m-%dT%H:%M:%S.%fZ'):
  d = datetime.strptime(str, mask)
  return d

LEVERAGE = 1
```

```
msg = btm.set_leverage(leverage=LEVERAGE)

def refresh():
        msg = btm.get_market_history("1m")
        df_mh = pd.DataFrame(msg)
        df_mh['timestamp'] = df_mh['timestamp'].map(str_to_utc)
        df_mh = df_mh.set_index('timestamp')
        df_mh = df_mh.sort_index()
        df_mh['momentum_rsi'] = ta.momentum.rsi(df_mh['close'], window=5)
        df_mh['ma'] = ta.trend.sma_indicator(df_mh['close'], window=20)
        df_mh['rsi'] = ta.momentum.rsi(df_mh['close'])
        indicator_bb = BollingerBands(close=df_mh["close"], window=20,
window_dev=2)
        df_mh['bb_middle'] = indicator_bb.bollinger_mavg()
        df_mh['bb_upper'] = indicator_bb.bollinger_hband()
        df_mh['bb_lower'] = indicator_bb.bollinger_lband()
        col = df_mh[-1:]
        err = [np.datetime_as_string(col.index.values[0], unit='s'),col['open'][0]]
        X = col.drop(['close'], axis=1)
        # X = X.drop(['timestamp'], axis=1)
        X = X.drop(['foreignNotional'], axis=1)
        X = X.drop(['homeNotional'], axis=1)
        X = X.drop(['lastSize'], axis=1)
        X = X.drop(['momentum_rsi'], axis=1)
        X = X.drop(['trades'], axis=1)
        X = X.drop(['symbol'], axis=1)
        X = X.drop(['turnover'], axis=1)
        X = X.drop(['vwap'], axis=1)
        rf_model = joblib.load('rf_model.pkl')
```

```
        y_pred1 = rf_model.predict(X)
        err.append(y_pred1[0])
        err1.append(err)
        oldT = err1[0][0]
        pr = err1[0][2]
        new_errT = []
        new_errP = []
        for ell in err1:
                if oldT != ell[0]:
                        new_errT.append(ell[1])
                        new_errP.append(pr)
                        pr = ell[2]
                        oldT = ell[0]

        with open('file.txt', 'w') as fw:
                # записываем
                json.dump(err1, fw)
        print("The MAPE is {}".format(mean_absolute_percentage_error(new_errT,
new_errP)))
        mape = mean_absolute_percentage_error(new_errT, new_errP)
        rmse = math.sqrt(mean_squared_error(new_errT, new_errP))
        print("The root mean squared error is {}.".format(rmse))

        return y_pred1,X,err1[0][0],len(new_errP),mape

err1 = []

while True:
        try:
                msg1, col, dat, kol, map = refresh()
```

```
            response = 'Привіт, відкрита зараз свічка :\n'
            response = response + 'Дата :' + np.datetime_as_string(col.index.values[0], unit='s') + '\n'
            response = response + 'Ціна при відкритті :' + "{:.4f}".format(col['open'][0]) + '\n'
            response = response + 'Максимум :' + "{:.4f}".format(col['high'][0]) + '\n'
            response = response + 'Мінімум :' + "{:.4f}".format(col['low'][0]) + '\n'
            response = response + 'Об\'єм :' + "{:.4f}".format(col['volume'][0]) + '\n'
            response = response + 'Прогнозоване значення(RandomForest):'
            response = response + "{:.4f}".format(msg1[0]) + '\n'
            response = response + 'Помилка системи:'
            response = response + "{:.4f}".format(map) + '\n'
            response = response + 'Помилка рахується з :' + dat + ' кількість значень для обрахунку помилки : ' + str(kol) + '\n'
            bot.send_message(435695634, response)
            time.sleep(30)
        except (KeyboardInterrupt, SystemExit):
            exit()
        except Exception as e:
            print('ERROR has occured')
            print(repr(e))
            time.sleep(10)
```

Регресійний аналіз:

```
import pandas as pd
```

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
import ta
from ta.volatility import BollingerBands
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error, r2_score
import math
import joblib

def return_rmse(test,predicted):
 rmse = math.sqrt(mean_squared_error(test, predicted))
 print("The root mean squared error is {}.".format(rmse))
 print("The MSE is {}".format(mean_squared_error(test, predicted)))
 print("The MAE is {}".format(mean_absolute_error(test, predicted)))
 print("The R2_Score is {}".format(r2_score(test, predicted)))
# Загрузка данных
df = pd.read_csv("datasets\\BTC-Hourly.csv")
df = df.sort_values('timestamp')

df['ma'] = ta.trend.sma_indicator(df['close'], window=20)
df['rsi'] = ta.momentum.rsi(df['close'])
indicator_bb = BollingerBands(close=df["close"], window=20, window_dev=2)
df['bb_middle'] = indicator_bb.bollinger_mavg()
df['bb_upper'] = indicator_bb.bollinger_hband()
df['bb_lower'] = indicator_bb.bollinger_lband()
df.dropna(inplace=True)
```

```
features = df[['open', 'high', 'low', 'close', 'volume', 'ma', 'rsi', 'bb_upper', 'bb_middle', 'bb_lower']]

target = df['close'].shift(-1)

train_size = int(len(df) * 0.8)
train_features = features[:train_size]
train_target = target[:train_size]
test_features = features[train_size:]
test_target = target[train_size:]
test_target = test_target.fillna(9000)

poly = PolynomialFeatures(degree=3)
train_features_poly = poly.fit_transform(train_features)
test_features_poly = poly.transform(test_features)

regressor = LinearRegression()
regressor.fit(train_features_poly, train_target)

train_predictions = regressor.predict(train_features_poly)
test_predictions = regressor.predict(test_features_poly)

return_rmse(test_target,test_predictions)

plt.figure(figsize=(16,8))
plt.plot(df['timestamp'], df['close'], label='Actual Price')
plt.plot(df['timestamp'][:train_size], train_predictions, label='Training Predictions')
plt.plot(df['timestamp'][train_size:-1], test_predictions[:-1:], label='Test Predictions')
plt.legend()
```

```
plt.show()
```

Нейронна мережа(LSTM):

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM
import ta
from ta.volatility import BollingerBands
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error, r2_score
import math

def return_rmse(test,predicted):
 rmse = math.sqrt(mean_squared_error(test, predicted))
 print("The root mean squared error is {}.".format(rmse))
 print("The MSE is {}".format(mean_squared_error(test, predicted)))
 print("The MAE is {}".format(mean_absolute_error(test, predicted)))
 print("The R2_Score is {}".format(r2_score(test, predicted)))

data = pd.read_csv('datasets\\BTC-Hourly.csv')
data = data.sort_values('timestamp')

data['rsi'] = ta.momentum.RSIIndicator(data['close'], window=14).rsi()
data['macd'] = ta.trend.MACD(data['close'], window_slow=26,
window_fast=12).macd()
data['ema20'] = ta.trend.EMAIndicator(data['close'], window=20).ema_indicator()
indicator_bb = BollingerBands(close=data["close"], window=20, window_dev=2)
data['bb_bbm'] = indicator_bb.bollinger_mavg()
```

```
data['bb_bbh'] = indicator_bb.bollinger_hband()
data['bb_bbl'] = indicator_bb.bollinger_lband()

# Нормализация данных
scaler = MinMaxScaler(feature_range=(0, 1))
data['close'] = scaler.fit_transform(np.array(data['close']).reshape(-1, 1))

# Разделение данных на обучающий и тестовый наборы
training_data = data.iloc[:int(len(data)*0.7), :]
testing_data = data.iloc[int(len(data)*0.7):, :]

# Функция для создания датасета временных рядов
def create_dataset(X, y, time_steps=1):
    Xs, ys = [], []
    for i in range(len(X) - time_steps):
        v = X.iloc[i:(i + time_steps)].values
        Xs.append(v)
        ys.append(y.iloc[i + time_steps])
    return np.array(Xs), np.array(ys)

# Создание датасета временных рядов
time_steps = 60
X_train, y_train = create_dataset(training_data[['close']], training_data['close'], time_steps)
X_test, y_test = create_dataset(testing_data[['close']], testing_data['close'], time_steps)

# Создание LSTM-модели
model = Sequential()
model.add(LSTM(64, return_sequences=True, input_shape=(time_steps, 1)))
```

```
model.add(LSTM(32, return_sequences=False))
model.add(Dense(1, activation='linear'))
model.compile(optimizer='adam', loss='mse')

# Обучение модели
model.fit(X_train, y_train, epochs=10, batch_size=16, validation_split=0.1, verbose=1)

# Прогнозирование цен на основе тестовых данных
y_pred = model.predict(X_test)

# Обратная нормализация данных
y_pred = scaler.inverse_transform(y_pred)
y_test = scaler.inverse_transform(y_test.reshape(-1, 1))
return_rmse(y_test,y_pred)
# Вывод результатов
import matplotlib.pyplot as plt
plt.plot(y_test, label='True Price')
plt.plot(y_pred, label='Predicted Price')
plt.legend()
plt.show()
```

Дерева рішень (Random Forest):

```
import pandas as pd
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
import ta
from ta.volatility import BollingerBands
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
```

```
from sklearn.metrics import mean_absolute_error, r2_score
import math
import joblib

def return_rmse(test,predicted):
 rmse = math.sqrt(mean_squared_error(test, predicted))
 print("The root mean squared error is {}.".format(rmse))
 print("The MSE is {}".format(mean_squared_error(test, predicted)))
 print("The MAE is {}".format(mean_absolute_error(test, predicted)))
 print("The R2_Score is {}".format(r2_score(test, predicted)))

# Зчитування датасету та видалення строк з NaN
df = pd.read_csv('datasets\\BTC-Hourly.csv')
df = df.sort_values('timestamp')
df.dropna(inplace=True)

# Добавление индикаторов
df['ma'] = ta.trend.sma_indicator(df['close'], window=20)
df['rsi'] = ta.momentum.rsi(df['close'])
indicator_bb = BollingerBands(close=df["close"], window=20, window_dev=2)
df['bb_middle'] = indicator_bb.bollinger_mavg()
df['bb_upper'] = indicator_bb.bollinger_hband()
df['bb_lower'] = indicator_bb.bollinger_lband()

df.dropna(inplace=True)
# Відокремлення вхідних та вихідних даних
X = df.drop(['close'], axis=1)
X = X.drop(['timestamp'], axis=1)
X = X.drop(['symbol'], axis=1)
X = X.drop(['unix'], axis=1)
```

```
X = X.drop(['Volume USD'], axis=1)
y = df['close']

train_size = int(len(df) * 0.8)
train_features = X[:train_size]
train_target = y[:train_size]
test_features = X[train_size:]
test_target = y[train_size:]

# Створення та налаштування моделі
rf = RandomForestRegressor(n_estimators=100, max_depth=10,
random_state=42)

# Тренування моделі
rf.fit(train_features, train_target)

# Оцінка точності моделі
score = rf.score(test_features, test_target)
print("Accuracy:", score)

# Прогнозування
predictions = rf.predict(test_features)
return_rmse(test_target,predictions)
# зберігаємо модель у файл
joblib.dump(rf, 'rf_model.pkl')
# Визуализация результатов
# plt.figure(figsize=(16,8))
# plt.plot(df['timestamp'][train_size:-1], test_target[:-1:], label='Actual Price')
# plt.plot(df['timestamp'][train_size:-1], predictions[:-1:], label='Test Predictions')
# plt.legend()
```

```
# plt.show()
```