Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Порівняння ефективності різних алгоритмів класифікації у прогнозуванні фінансових ринків
(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-21-3
Анатолій Ксьонз
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект
(повна назва освітньої програми)
Керівник ст. викл. Олександр Стьопін
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Олег ЗОЛОТУХІН
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Ксьонзу Анатолію Анатолійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Порівняння ефективності різних алгоритмів класифікації у прогнозуванні фінансових ринків

затверджена наказом університету від 19 травня 20 25 р. № 378Ст

2. Термін подання студентом роботи до екзаменаційної комісії 18 червня 20 25 р.

3. Вихідні дані до роботи Сучасні наукові дослідження з фінансового прогнозування та алгоритмічної торгівлі, документацію до бібліотек Python (Freqtrade, scikit-learn, TensorFlow, pandas, matplotlib), відкриті історичні записи про біржові операції (BTC/USDT, ETH/USDT)

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та постановка задачі
2) Методологічна база дослідження
3) Розробка класифікаційних моделей для фінансового прогнозування

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	19.05.2025	виконано
2	Аналіз предметної галузі	21.05.2025	виконано
3	Формалізація мети, постановка задач	26.05.2025	виконано
4	Підбір та аналіз алгоритмів класифікації	29.05.2025	виконано
5	Підготовка та обробка біржових даних (OHLCV)	01.06.2025	виконано
6	Побудова та навчання моделей машинного	04.06.2025	виконано
	навчання		
7	Розробка єдиного тестового середовища для	07.06.2025	виконано
	backtesting		
8	Аналіз результатів та візуалізація ефективності	09.06.2025	виконано
	моделей		
9	Оформлення пояснювальної записки	13.06.2025	виконано
10	Підготовка презентації та доповіді	16.06.2025	виконано
11	Захист кваліфікаційної роботи перед ЕК	18.06.2025	виконано

Дата видачі завдання 19 травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ ст. викл. Олександр Стьопін
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 95 с., 19 рис., 3 табл., 3 дод., 31 джерело.

КЛАСИФІКАЦІЯ, МАШИННЕ НАВЧАННЯ, ОНЛАЙН-ТРЕЙДИНГ, ПРОГНОЗУВАННЯ, ПРОГРАМНІ ПАТЕРНИ, ТЕХНІЧНИЙ АНАЛІЗ, ФІНАНСОВИЙ РИНОК, ШТУЧНИЙ ІНТЕЛЕКТ, BACKTESTING, OHLCV, PYTHON.

Об’єкт дослідження – процеси автоматизованого прийняття рішень у сфері фінансового прогнозування.

Предмет дослідження – алгоритми класифікації, які застосовуються для передбачення торгових рішень на основі технічних індикаторів.

Мета дослідження – провести експериментальне порівняння ефективності різних класифікаційних моделей у контексті прогнозування ринкових сигналів (Buy/Hold/Sell) та оцінити їх практичну придатність у трейдингових стратегіях.

Методи дослідження – аналіз наукових джерел, формалізація фінансових індикаторів як ознак для навчання, реалізація моделей у Python, навчання на OHLCV-даних, тестування моделей через backtesting, порівняння точності та стабільності результатів.

У межах роботи реалізовано кілька моделей класифікації, натренованих на технічних індикаторах, з метою визначення торгових дій. Vоделі було протестовано в однакових умовах, що дозволило об’єктивно порівняти їх точність, узагальнювальну здатність і вплив на прибутковість торгової стратегії. Проведено серію експериментів, результати яких візуалізовано та систематизовано.

ABSTRACT

Bachelor’s thesis contains: 95 pp., 19 fig., 3 tabl., 3 ann., 31 references.

ARTIFICIAL INTELLIGENCE, BACKTESTING, CLASSIFICATION, FINANCIAL MARKET, FORECASTING, MACHINE LEARNING, OHLCV, ONLINE TRADING, PROGRAMMING PATTERNS, PYTHON, TECHNICAL ANALYSIS.

The object of the study is the process of automated decision-making in financial forecasting.

The subject of the study is classification algorithms used to predict trading decisions based on technical indicators.

The aim of the research is to perform an experimental comparison of the effectiveness of various classification models in forecasting market signals (Buy/Hold/Sell) and to evaluate their practical applicability in trading strategies.

The research methods include literature review, feature engineering of financial indicators, implementation of classification models in Python, training on OHLCV market data, backtesting, and comparative analysis of model accuracy and stability.

As part of the study, several machine learning models were developed and trained on technical indicators to predict trading actions. All models were evaluated under identical conditions, enabling an objective comparison of their predictive accuracy, generalization capabilities, and influence on trading strategy profitability. A series of experiments was conducted, and the results were visualized and systematically analyzed.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AI – Artificial Intelligence – штучний інтелект;

BTC/USDT, ETH/USDT – торгові пари криптовалют (Bitcoin/Tether, Ethereum/Tether);

Classifier – алгоритм, який визначає належність об'єкта до одного з класів;

CSV – Comma-Separated Values – формат табличних даних, розділених комами;

EMA – Exponential Moving Average – експоненціальне ковзне середнє;

F1-score – метрика точності класифікації, що враховує precision і recall;

Freqtrade – фреймворк на Python для розробки та тестування торгових стратегій;

JSON – JavaScript Object Notation – формат обміну структурованими даними;

ML – Machine Learning – машинне навчання;

OHLCV – Open, High, Low, Close, Volume – структура фінансових даних з біржових торгів;

PDF – Portable Document Format – формат електронних документів;

ROC – Rate of Change – індикатор швидкості зміни ціни;

RSI – Relative Strength Index – індекс відносної сили, технічний індикатор;

SMA – Simple Moving Average – просте ковзне середнє;

SVM – Support Vector Machine – алгоритм машинного навчання для класифікації;

Trading signal – прогнозована модель поведінки на ринку.

ВСТУП

У сучасному світі фінансові ринки перетворилися на глобальний і надзвичайно динамічний механізм, який функціонує у режимі реального часу та охоплює мільйони учасників по всьому світу. З огляду на високу мінливість ринкових умов, все більше компаній та приватних трейдерів звертаються до автоматизованих інтелектуальних систем для прийняття рішень. Особливої ваги набуває завдання прогнозування ринкових змін, яке потребує не лише опрацювання великих обсягів історичних даних, але й глибокого аналізу взаємозв'язків між різними макро- та мікроекономічними чинниками. Це включає новинні потоки, обсяги торгів, поведінкові патерни учасників ринку, технічні індикатори, а також загальний інформаційний фон, що впливає на очікування інвесторів.

У цьому контексті дедалі більшого поширення набувають методи машинного навчання, зокрема алгоритми класифікації, які дозволяють виявляти приховані залежності у складних багатовимірних даних та формувати прогнози щодо майбутніх дій на ринку. Такі алгоритми можуть ефективно вирішувати задачу визначення торгових сигналів – наприклад, коли доцільно купувати актив, утримувати його або продавати. Серед них найбільш відомими є логістична регресія, метод опорних векторів (SVM), дерева рішень, випадкові ліси (Random Forest), багатошарові персептрони (MLP), а також ансамблеві моделі, які поєднують декілька підходів для досягнення більшої точності та стійкості до шуму.

Проте вибір найефективнішого алгоритму не є очевидним: кожна модель має свої переваги й обмеження залежно від структури вхідних даних, вибраних ознак, параметрів оптимізації та цільової мети аналізу. У фінансовій сфері, де кожне неправильне рішення може спричинити значні збитки, критично важливо обґрунтувати вибір методів і довести їхню доцільність на практиці. Саме тому порівняння різних класифікаційних

підходів у рамках єдиного експериментального середовища є актуальною й практично цінною задачею.

Особливої уваги надано ринку криптовалют – одному з найбільш волатильних і чутливих до новин сегментів сучасних фінансів. На відміну від традиційних активів, криптовалюти торгуються цілодобово, мають низький поріг входу для учасників, а їхня ціна формується значною мірою під впливом спекулятивних очікувань. Завдяки відкритим біржовим API та доступу до історичних даних у форматі OHLCV (Open, High, Low, Close, Volume), крипторинок є зручною платформою для експериментального тестування фінансових моделей.

Метою даної кваліфікаційної роботи є створення експериментального середовища для навчання та тестування декількох класифікаційних моделей з подальшим порівнянням їх ефективності в задачі прогнозування торгових сигналів (Buy/Hold/Sell). Особливий акцент зроблено на оцінці точності, стабільності, узагальнювальної здатності та впливу моделі на симульований фінансовий результат. Результати дослідження дозволяють виявити найбільш перспективні алгоритми та запропонувати практичні рекомендації щодо їх використання у реальних умовах.

Актуальність обраної теми визначається сукупністю факторів: розвитком фінансових технологій, поширенням алгоритмічного трейдингу, зростанням попиту на персоналізовані інструменти аналізу даних, а також потребою в адаптивних, надійних та прозорих рішеннях у сфері фінансів. Об’єднання інструментів штучного інтелекту з методами технічного аналізу формує потужний підхід до прогнозування, здатний покращити якість інвестиційних рішень не лише для професіоналів, а й для широкого кола користувачів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

Фінансові ринки характеризуються складною багатофакторною природою та високою мінливістю, що значно ускладнює процес прийняття інвестиційних рішень. З розвитком цифрових технологій усе більшого поширення набуває використання алгоритмічних систем, які здатні автоматизувати аналіз даних та оптимізувати стратегії торгівлі. У межах цього дослідження розглядається питання підвищення ефективності прогнозування торгових рішень за допомогою моделей класифікації, що мають здатність адаптуватися до ринкових умов.

1.1 Актуальність прогнозування на фінансових ринках

Алгоритмічна торгівля (автоматизований трейдинг) – це підхід до здійснення фінансових операцій, при якому рішення про купівлю або продаж активів приймаються програмним кодом на основі заздалегідь заданих правил. Основна мета такого підходу – виключити суб’єктивний фактор, підвищити швидкість реагування на ринкові зміни та забезпечити дисципліноване дотримання торгових стратегій.

Перші системи автоматизованої торгівлі з’явилися ще у 1970–80-х роках, однак справжній прорив стався на початку 2000-х із розвитком електронних платформ і доступу до ринку через API. Сучасні трейдинг-системи використовують десятки показників технічного аналізу, складні формули та математичні моделі, що реалізовані в коді на мовах Python, C++, JavaScript тощо. Важливим етапом у цьому процесі є перехід від rule-based логіки до моделей, навчених на історичних даних – тобто систем машинного навчання, здатних виявляти закономірності, які складно описати вручну.

Особливий інтерес становлять трейдінг-боти – автономні програмні агенти, які здатні здійснювати торгові операції на біржах без безпосереднього втручання людини. Такі боти зазвичай взаємодіють із

торговими платформами за допомогою API (Binance API, FTX API), аналізують ринкові дані в реальному часі та приймають рішення на основі заданих алгоритмів або результатів моделі машинного навчання.

Архітектура типового бота включає кілька ключових модулів: модуль збору даних, модуль аналітики (розрахунок технічних індикаторів), модуль прийняття рішень (класифікатор або rule-based логіка) та модуль взаємодії з біржею (відправка наказів на купівлю або продаж). Залежно від складності реалізації, такі системи можуть працювати як у режимі симуляції (backtesting), так і в реальному часі (live-trading) [1].

На сьогоднішній день існує безліч відкритих платформ для створення трейдинг-ботів. Однією з найпопулярніших є Freqtrade – Python-фреймворк, що дозволяє створювати, тестувати та розгортати торгові стратегії з можливістю інтеграції моделей машинного навчання. Саме на основі цієї платформи реалізовано частину експериментальної частини дослідження.

Важливою особливістю ринку є висока волатильність і динамічність, тому навіть незначні затримки в прийнятті рішень або некоректне реагування на зміну тренду можуть призвести до суттєвих збитків. У цьому контексті зростає роль систем, що здатні не просто дотримуватись фіксованих правил, а навчатися з даних – тобто працювати в режимі класифікації ринкових станів із прогнозом майбутньої поведінки.

1.2 Проблематика класифікації торгових сигналів

1.2.1 Особливості фінансових часових рядів як вхідних даних

У задачах машинного навчання важливу роль відіграє якість і структура вхідних даних. Фінансові часові ряди, які використовуються для побудови моделей прогнозування в алгоритмічній торгівлі, мають низку специфічних характеристик, що суттєво ускладнюють процес моделювання та впливають на точність класифікаційних алгоритмів.

Перш за все, фінансові дані є нестаціонарними, тобто їхні статистичні властивості (середнє значення, дисперсія, кореляція) змінюються з часом. Модель, навчена на одному часовому відрізку, може бути неактуальною або малоефективною на іншому. Тренд на ринку, що спостерігається протягом певного періоду, може змінитися на флет або обернений тренд через зовнішні чинники – економічні новини, політичні події або загальний стан ринку. Тому перед обробкою часових рядів необхідно враховувати адаптивність моделей і використовувати методи, які зменшують вплив нестаціонарності, наприклад ковзні середні або нормалізацію даних.

Іншою характерною рисою фінансових рядів є високий рівень шуму. Ринок часто змінюється через спекулятивні дії, непередбачувані новини або навіть психологічні чинники, які не мають фундаментального підґрунтя. У результаті сигнали, які справді мають прогностичну цінність, можуть бути «приховані» за шумом. Це створює значні труднощі для алгоритмів машинного навчання, які можуть сприймати шум як закономірність і переобучатися. У таких випадках важливо застосовувати фільтри або методи регуляризації, щоб зменшити ризик перенавчання та покращити узагальнювальну здатність моделі [2].

Фінансові часові ряди також демонструють високу волатильність, схильність до різких змін значень за короткий проміжок часу. На відміну від більш «спокійних» даних (погодних або виробничих), ціни активів можуть змінюватися на десятки відсотків протягом декількох хвилин або годин. Це вимагає від моделей здатності швидко реагувати на зміни та адаптувати свої рішення в умовах високої невизначеності. Стандартні методи аналізу, які припускають рівномірність змін, у таких умовах не дають задовільних результатів.

Фінансові ряди часто мають сезонні або циклічні коливання. Наприклад, в окремі години дня (на відкритті чи закритті біржі) активність торгів зростає, а у вихідні практично зникає. Також є глобальні патерни: наприкінці місяця інвестори можуть переоцінювати позиції, а в період

оголошення фінансових звітів – активно торгувати відповідними акціями. Такі регулярні зміни необхідно враховувати в моделі, додаючи відповідні календарні або тимчасові ознаки (день тижня, година доби), інакше точність класифікації торгових сигналів значно знизиться.

Ще однією складністю є автокореляція – залежність поточних значень від попередніх. У коротких часових інтервалах це явище може бути суттєвим: якщо ціна різко зростає, існує висока ймовірність, що зростання продовжиться ще деякий час. Водночас, у довших часових відрізках така залежність може зникати або змінюватися. Для правильної обробки автокореляції часто використовують ковзні вікна, лаги або методи згладжування, однак надмірне врахування таких ефектів також може призвести до перенавчання.

Окремо слід виділити вплив зовнішніх факторів, які не завжди присутні у самих вхідних даних, але можуть суттєво впливати на ринок: новини, геополітичні події, публікації великих інвесторів у соціальних мережах тощо. Включення таких джерел до моделі (у вигляді тональності новин або індикаторів новинного фону) може суттєво підвищити точність класифікації, однак вимагає складнішої системи збору, обробки та синхронізації додаткової інформації.

1.2.2 Труднощі формалізації цільових класів (Buy/Hold/Sell)

Побудова моделей машинного навчання передбачає наявність не лише вхідних даних, але й чітко визначених вихідних міток, за якими алгоритм навчається. У випадку класифікації торгових сигналів це означає, що кожному часовому відрізку або моменту часу необхідно приписати одне з рішень: купувати актив, утримувати позицію або продавати. І саме на етапі формалізації цих цільових класів виникає низка концептуальних і практичних труднощів.

Найперша проблема полягає у визначенні самого поняття «правильного моменту» для купівлі чи продажу. У фінансах не існує об’єктивної істини: одна й та сама ситуація на ринку може трактуватися по-різному залежно від стратегії, горизонту інвестування, рівня ризику й навіть особистих переконань трейдера. Те, що для одного є сигналом до купівлі, для іншого – привід залишитися осторонь або зафіксувати прибуток. У контексті навчання моделей це означає, що спочатку аналітик повинен самостійно сформулювати правила, за якими будуть виставлені мітки – наприклад, якщо після поточного моменту ціна зросте більш ніж на 1,5% за наступні 10 хвилин, то вважати цю точку «Buy». І навпаки, якщо відбудеться падіння, позначити «Sell». Але така формалізація завжди є дещо штучною, бо базується на ретроспективному знанні, яке в реальному часі недоступне.

Ще однією складністю є часовий горизонт, на якому будується розмітка. Різні трейдери оперують різними масштабами, хтось оцінює зміни щосекунди, хтось орієнтується на денні або тижневі тренди. Вибір довжини «вікна майбутнього», у межах якого оцінюється результат дії (чи дійсно купівля принесла прибуток), суттєво впливає на формування навчальної вибірки. Невдалий вибір цього параметра може як приховати корисну інформацію, так і спровокувати появу хибних патернів [3].

Окрема проблема – дисбаланс класів. На практиці найбільшу частину ринку становлять періоди, коли нічого радикального не відбувається: ціна коливається у вузькому діапазоні, відсутній чіткий тренд. Це природно призводить до того, що більшість прикладів у навчальній вибірці отримують мітку «Hold». У результаті модель починає переважно прогнозувати саме цей клас, оскільки він дозволяє досягти високої формальної точності навіть без реальної корисності. Такий ефект часто вводить в оману, адже з формального боку точність виглядає високою, але стратегія, заснована на таких прогнозах, може бути абсолютно непридатною для застосування.

Дисбаланс класів також ускладнює процес оптимізації моделі: стандартні функції втрат не враховують вагу помилок залежно від класу. Помилковий сигнал «Buy» у момент, коли слід було «Sell», має зовсім інші наслідки, ніж помилка «Hold» у нейтральний момент. Щоб вирішити цю проблему, застосовують зважування класів або змінюють метрики оцінки, однак це потребує додаткового налаштування, яке залежить від специфіки даних.

1.2.3 Вплив якості класифікації на прибутковість стратегії

Одним з найбільш хибних уявлень при розробці моделей прогнозування на фінансових даних є ототожнення високої точності класифікації з високою ефективністю в торгівлі. На перший погляд, здається очевидним: якщо модель правильно визначає торгові сигнали, вона має приносити прибуток. Проте на практиці між формальними метриками класифікації та реальною доходністю існує складна і далеко не лінійна залежність.

Фінансові ринки відрізняються високою чутливістю до кожного рішення. Один невчасний сигнал «Buy», особливо в момент, коли ринок іде вниз, може призвести до значних втрат. Навпаки, навіть одиничний пропущений «Sell» у критичній точці може «з'їсти» прибуток, сформований багатьма попередніми правильними рішеннями. У такій ситуації загальна точність – це занадто абстрактна величина, яка не враховує асиметрії наслідків різних типів помилок.

Ще складнішою ситуація стає, якщо враховувати величину руху ринку. Модель може «вгадати» 90% незначних коливань, але при цьому пропустити кілька ключових моментів із сильними трендами. У результаті фінансовий результат буде значно гіршим, ніж у менш точної, але стратегічно чутливішої моделі. Це вказує на необхідність використовувати не лише класифікаційні метрики (точність, recall, F1-score), а й фінансово-

орієнтовані: загальний прибуток (cumulative return), коефіцієнт Sharpe, максимальне просідання (max drawdown), співвідношення прибутку до ризику тощо.

Додатковим ускладненням є взаємодія рішень у часі. Кожне торгове рішення впливає на наступне: відкривши позицію, трейдер обмежений у своїх наступних діях. Наприклад, помилковий сигнал на купівлю змушує утримувати збиткову позицію, з якої не так просто вийти, а кожна наступна дія стає залежною від попередньої. У машинному навчанні це означає, що навіть при високій точності на окремих точках модель може формувати нестабільні або непридатні для використання ланцюжки рішень [4].

Водночас, інші моделі можуть мати меншу формальну точність, але забезпечувати стабільнішу та безпечнішу торгівлю. Це ще раз доводить, що при оцінці моделей класифікації у фінансовому контексті ключовим є не лише аналіз їхніх передбачень, а й моделювання їх впливу на реальну торгову стратегію. Тут доречним є впровадження backtesting-середовищ, у яких можна симулювати поведінку моделі на історичних даних і зіставити її прогнози з реальними фінансовими результатами [5].

Цінність моделі в трейдингу визначається не кількістю вгаданих рішень, а балансом між ризиком і прибутком. Тільки врахування цього балансу на всіх етапах – від розмітки даних до метрик оцінки, дозволяє говорити про справжню ефективність класифікаційного алгоритму в контексті фінансових ринків.

1.3 Огляд підходів до фінансового прогнозування

Історично прогнозування фінансових ринків формувалося на перетині досвіду, інтуїції та спостережень. Протягом десятиліть трейдери намагалися вивести закономірності з поведінки ціни, щоб визначити вигідні моменти для входу та виходу з ринку. З появою комп’ютерних систем ці інтуїтивні спостереження перетворилися на формалізовані стратегії. Водночас, поява

великого обсягу історичних даних та зростання обчислювальних можливостей створили умови для застосування більш точних і гнучких математичних моделей. Сьогодні в торговельних системах співіснують класичні rule-based методики, статистичні моделі та методи машинного навчання.

1.3.1 Технічний аналіз та rule-based підходи

Технічний аналіз посідає одне з центральних місць у структурі класичних підходів до фінансового прогнозування. Його методологічна основа полягає у припущенні, що всі значущі для ринку фактори економічні, політичні, емоційні, вже відображені у динаміці цін. Таким чином, замість аналізу причин змін варто зосередити увагу на самій поведінці ціни, намагаючись виявити характерні закономірності, що повторюються в історії. Такий підхід передбачає побудову прогнозу на основі інтерпретації графічних структур (патернів) і поведінки індикаторів технічного аналізу.

Основним операційним принципом технічного аналізу є правило, згідно з яким «ринок має пам’ять». Інакше кажучи, вважається, що ринкові учасники часто діють схожим чином в однакових ситуаціях, створюючи повторювані шаблони поведінки ціни. Ці шаблони зазвичай ідентифікуються через так звані «технічні сигнали» – тригери, які можуть бути виражені у вигляді відомих формацій або через значення обчислюваних індикаторів, таких як ковзні середні (SMA, EMA), осцилятори (RSI, Stochastic), об’ємні та волатильні показники [6].

Rule-based трейдинг, що базується на технічному аналізі, передбачає формалізацію цих сигналів у вигляді чітко заданих правил прийняття рішень. Такі правила визначають, за яких умов потрібно відкрити або закрити позицію, збільшити обсяг чи змінити рівень стоп-лосу. Перетин короткої та довгої ковзної середньої може інтерпретуватися як початок нового тренду, і відповідно запускати торгівельну операцію. Ці правила

реалізуються у вигляді скриптів або стратегій, які автоматично перевіряють умови на ринку і реагують у заздалегідь визначений спосіб.

Перевагою такого підходу є його передбачуваність і прозорість. Кожне рішення може бути повністю пояснене, що є цінним з погляду ризик-менеджменту та нормативної відповідності. Крім того, rule-based моделі зручні для тестування на історичних даних, що дозволяє швидко оцінити їх ефективність до застосування в реальній торгівлі. Проте формальна простота таких систем приховує кілька суттєвих обмежень, які виявляються особливо критичними в умовах сучасного волатильного ринку.

Найбільш очевидною проблемою є обмежена гнучкість. Технічні правила не здатні змінюватися у відповідь на нові ринкові умови. Ринок є адаптивним середовищем, де ефективність одного й того самого підходу може зменшуватись із часом через зміну поведінки учасників або зростання алгоритмічної конкуренції. Як наслідок, стратегії, які давали прибуток протягом певного періоду, можуть втратити релевантність, залишаючись при цьому структурно незмінними.

Крім того, rule-based системи не мають здатності до самонавчання. Усі параметри та умови мають бути зафіксовані вручну, що знижує здатність до адаптації. Більше того, оптимізація таких стратегій часто базується на переборі параметрів (grid search), який підвищує ризик перенавчання до історичних даних, явища, відомого як overfitting. Така стратегія може показувати ідеальні результати на тренувальному періоді, але повністю втрачати ефективність на нових даних через неспроможність узагальнювати закономірності.

Слід зазначити, що більшість rule-based стратегій побудовані на фундаментально лінійних залежностях: вони реагують на порогові значення одного або двох індикаторів без урахування глибших або нелінійних взаємозв’язків. Це різко обмежує їхню здатність працювати у складних ринкових умовах, де рішення залежить не лише від рівня ціни, а й від темпу зміни, контексту, часової структури та синхронних коливань інших активів.

Технічний аналіз залишається важливою частиною фінансової культури трейдингу. Його візуальні та логічні принципи формують базу для початкового розуміння ринку та розробки простих стратегій. Але саме обмеженість rule-based систем змусила сучасну фінансову індустрію звернутися до методів, які здатні навчатися з даних, реагувати на зміну контексту і виявляти приховані закономірності. У зв’язку з цим технічний аналіз дедалі частіше використовується як частина гібридних систем, де фіксовані правила доповнюються алгоритмами машинного навчання, що дозволяє зберегти інтерпретованість і водночас підвищити гнучкість та стійкість стратегії.

1.3.2 Прогнозування на основі статистичних моделей

На відміну від фіксованих логічних конструкцій, побудованих на правилах технічного аналізу, статистичні методи прогнозування орієнтовані на математичну формалізацію ринкових процесів. Їх суть полягає не в евристичному описі поведінки ціни, а в побудові моделі, що відображає статистичні залежності між послідовними спостереженнями у часовому ряді. Застосування таких підходів у фінансах дозволяє не лише створювати прогнози, а й розуміти внутрішню структуру даних, що є основою для формування рішень із чітко визначеним рівнем довіри.

У статистичних моделях прогнозування базується на концепції автокореляції, що передбачає наявність часових залежностей між попередніми й поточними спостереженнями. Серед найбільш поширених моделей цього класу особливе місце займає ARIMA – авторегресивна інтегрована модель ковзної середньої. Вона поєднує три ключові компоненти: авторегресію (AR), яка враховує вплив попередніх значень; інтегрування (I), що відповідає за обробку нестаціонарності через диференціювання; та компонент ковзної середньої (MA), який моделює взаємозв’язок залишкових похибок [7].

У складніших фінансових умовах, де волатильність ринку не є сталою, широкого поширення набула модель GARCH (Generalized Autoregressive Conditional Heteroskedasticity). Вона дозволяє динамічно оцінювати дисперсію (тобто рівень ризику) на кожному кроці прогнозу, що особливо цінно в умовах ринку, схильного до сплесків нестабільності. GARCH-моделі здатні ефективно вловлювати кластери волатильності, коли періоди високої турбулентності змінюються фазами відносної стабільності, що є типовою рисою поведінки більшості фінансових активів.

Значною перевагою статистичних моделей є інтерпретованість результатів. Вони забезпечують можливість не лише передбачати майбутнє значення, але й обчислювати довірчі інтервали, перевіряти гіпотези про структуру ряду, оцінювати ступінь значущості змінних. Така прозорість дозволяє фінансовим аналітикам не просто використовувати модель як чорний ящик, а критично осмислювати її висновки і приймати виважені рішення з урахуванням кількісної невизначеності [8].

Однак, застосування статистичних моделей у фінансовому контексті супроводжується низкою важливих обмежень. По-перше, більшість класичних моделей передбачають стаціонарність ряду – умову, за якої середнє, дисперсія та інші характеристики не змінюються з часом. У практиці реальних фінансових ринків таку умову дотримати практично неможливо. Внаслідок постійних змін економічного фону, новинного середовища та поведінки учасників ринку, властивості даних змінюються динамічно, а самі ряди набувають складної структури, далекої від стаціонарності. Тому перед моделюванням часто застосовуються різноманітні трансформації (логарифмування, диференціювання), які не завжди зберігають первинний економічний зміст.

Крім того, класичні моделі погано масштабуються на високовимірні вхідні простори. У випадку, коли до моделі додаються додаткові технічні індикатори, макроекономічні змінні або фактори поведінкової економіки, виникає проблема мультиколінеарності та зростання дисперсії оцінок. Це

різко знижує прогностичну здатність моделі та ускладнює її використання у складному багатофакторному середовищі.

Ще одним викликом є слабка здатність обробляти нелінійні залежності. Більшість статистичних моделей базуються на лінійних припущеннях, тоді як фінансові ринки характеризуються високою складністю і наявністю взаємодій, які не можна звести до простих адитивних ефектів. У таких випадках навіть добре параметризована модель може не вловити приховані патерни, що формуються під впливом зовнішніх або латентних факторів.

Статистичні методи є чутливими до вибору параметрів: кількість лагів, порядок інтегрування, форма розподілу залишків, усі ці елементи можуть істотно вплинути на стабільність та точність прогнозу. Відсутність автоматичних механізмів адаптації до нових даних обмежує здатність моделі пристосовуватися до змін середовища в режимі реального часу.

1.3.3 Машинне навчання та класифікаційні моделі

У контексті прогнозування фінансових ринків машинне навчання становить собою альтернативу традиційним аналітичним підходам, яка здатна виявляти приховані, складні та часто нелінійні взаємозв’язки у даних, що залишаються поза досяжністю класичних rule-based або статистичних моделей. Його застосування базується на здатності алгоритмів адаптуватися до нових ситуацій шляхом узагальнення знань, отриманих з історичних прикладів. У трейдингу це означає можливість автоматично вчитися на основі ринкових закономірностей і розвивати моделі, які не просто імітують минуле, а будують гнучкі стратегії реагування на майбутнє.

Замість ручного кодування фіксованих правил, класифікаційна модель отримує на вхід ознаки, сформовані з ринкових даних, і навчається на історичному досвіді, щоб передбачити стан ринку в певний момент часу. В якості ознак можуть виступати значення технічних індикаторів, зміни

обсягу торгів, відносні позиції ціни щодо ковзних середніх, а також метаінформація про сезонність, волатильність або контекст останніх подій. Цільова змінна в таких задачах формулюється як клас: рішення купити актив, утримувати позицію або продати.

Ключова перевага класифікаційного підходу полягає у здатності виявляти складні шаблони, які важко або неможливо описати аналітично. Алгоритм може навчитись розпізнавати сигнали, сформовані комбінацією ознак, поведінкових аномалій або лагових впливів, що не очевидні при побіжному спостереженні. Це дозволяє формувати прогнози не лише на основі поточного значення ринку, а й з урахуванням контексту динаміки змін. Саме така контекстуальна глибина створює основу для розробки адаптивних систем, які відчутно перевершують традиційні методи в умовах нестійких, волатильних ринків [9].

Алгоритми машинного навчання умовно поділяються на два великі підходи: прості класифікатори, які засновані на лінійній або розгалуженій логіці прийняття рішень, та нейромережеві структури, здатні моделювати складні трансформації ознак. До першої групи належать логістична регресія, метод опорних векторів, дерева рішень та ансамблеві моделі, які дозволяють легко інтерпретувати результат і забезпечують високу швидкість роботи. У той час як глибокі нейронні мережі, зокрема рекурентні та трансформерні архітектури – демонструють здатність до моделювання часової динаміки та виявлення зв’язків у довгих залежностях, які не враховуються у звичайних підходах.

Втім, попри потенціал, машинне навчання не позбавлене ускладнень. Найбільшою загрозою є перенавчання на історичних даних, що призводить до гарних результатів у минулому, але провалу в майбутньому. Це трапляється тоді, коли модель запам’ятовує шум, а не закономірність, і тому втрачає здатність до узагальнення. Щоб цього уникнути, застосовують регуляризацію, крос-валідацію, зважування класів та інші техніки стабілізації навчання [10].

Високі показники точності класифікації самі по собі не є гарантією ефективності у фінансовому сенсі. Модель може демонструвати ідеальні метрики на рівні класифікатора, однак генерувати торгові сигнали, які не приносять прибутку або є надто ризикованими. Тому успішне впровадження машинного навчання у трейдинг вимагає поєднання обчислювальної оптимізації з доменним розумінням ринку, а також обов'язкового тестування моделі в середовищі backtesting, що імітує реальні умови торгівлі.

Сучасні дослідження у галузі фінансового ML свідчать про доцільність побудови гібридних архітектур, у яких машинне навчання працює в парі з фільтрами технічного аналізу або статистичними моделями. Такі системи здатні поєднувати адаптивність з інтерпретованістю, що особливо важливо у високоризикованих фінансових середовищах.

1.3.4 Гібридні підходи та моделі з підкріпленням

У міру розвитку фінансових ринків та ускладнення їхньої структури, зростає потреба у методах прогнозування, які можуть одночасно забезпечувати високу точність, адаптивність, прозорість і стратегічну гнучкість. Саме в цій точці з'являється ідея гібридизації, коли різні аналітичні підходи: технічний аналіз, статистичні моделі, методи машинного навчання – поєднуються в єдину архітектуру, здатну вловлювати багаторівневі залежності між змінними, враховувати часовий контекст і пристосовуватись до динамічних змін середовища. Гібридні системи прагнуть об'єднати структуровану логіку rule-based підходів із гнучкістю самонавчання, властивою класифікаторам і нейронним мережам, формуючи баланс між контрольованістю рішень та їх ефективністю.

У таких системах статистичні моделі часто виконують роль фільтрів чи попередньої обробки, стабілізуючи часовий ряд, зменшуючи шум або виділяючи трендову складову. Потім результат передається у блоки

машинного навчання, які вивчають більш тонкі закономірності або адаптуються до короткострокових змін ринку.

Певні моделі також використовують технічні індикатори як вхідні ознаки, комбінуючи їх із даними глибокого контексту або новинним тлом. Така багаторівнева обробка дозволяє не лише підвищити якість прогнозу, а й сформувати більш стійкі сигнали до дії, здатні функціонувати у режимі невизначеності [11].

Окремий напрям у гібридних стратегіях становить застосування методів з підкріпленням (reinforcement learning, RL), які імітують логіку навчання через досвід. У класичному машинному навчанні модель навчається на фіксованому наборі даних з готовими відповідями, тоді як у RL система вчиться самостійно, взаємодіючи з середовищем та отримуючи зворотний зв’язок у вигляді нагороди або покарання за вчинені дії. У контексті фінансів це дозволяє моделювати стратегічну поведінку трейдера або агента, який приймає послідовність рішень: від відкриття позиції до її закриття, з урахуванням прибутку, ризику та динаміки ринку.

Особливістю підходів з підкріпленням є те, що вони не просто прогнозують, а оптимізують політику поведінки. Метою стає не точне передбачення наступного кроку ціни, а знаходження стратегії, яка забезпечує максимальну сумарну винагороду в довгостроковій перспективі.

Це радикально змінює логіку побудови торгових систем. RL-моделі починають діяти автономно, самостійно тестуючи різні дії в різних ринкових сценаріях, накопичуючи досвід і поступово вдосконалюючи свою політику. Такий підхід добре вписується в ідеологію автоматизованого трейдингу, де ринкове середовище можна сприймати як динамічну гру з невизначеними правилами, змінною інформацією та обмеженим часом на ухвалення рішення [12].

У більшості сучасних реалізацій RL у трейдингу використовуються варіації Q-learning, SARSA, Deep Q-Networks (DQN) або Policy Gradient методів. Вони дозволяють створювати агентів, які оперують не лише

поточними ознаками, а й станом ринку у ширшому часовому контексті. Це важливо, оскільки в багатьох випадках прибутковість або збитковість стратегії визначається не моментом входу, а сукупністю рішень, прийнятих у серії. Навчання таких агентів потребує симуляційного середовища, тобто системи, яка здатна емулювати ринок і дозволяти агенту проводити експерименти над ним без ризику втрат.

Попри свої переваги, методи з підкріпленням стикаються з низкою обмежень. Висока чутливість до структури нагород, складність налаштування параметрів, великий обсяг необхідних симуляцій, нестабільність політики в нестійких умовах, усе це ускладнює використання RL у реальних фінансових продуктах.

Водночас, комбінація таких моделей із іншими алгоритмами, як-от класифікатори чи статистичні моделі, дозволяє пом’якшити слабкі місця кожного окремого методу та отримати систему з більш високим рівнем адаптивності.

Гібридні підходи також відкривають шлях до інтеграції людської експертизи. У деяких архітектурах частина рішень передається на розгляд аналітику, а система виступає як радник. В інших – людські патерни використовуються як сигнали для попереднього навчання агента, а модель адаптується до індивідуального стилю трейдера. Усе це створює можливість побудови персоналізованих систем підтримки фінансових рішень, що комбінують обчислювальну потужність з інтуїтивним досвідом.

Гібридні моделі та методи з підкріпленням формують нову парадигму фінансового прогнозування, у якій технічна точність поєднується з адаптивною стратегією. Вони не лише підвищують ефективність, а й розширюють функціональність трейдингових систем, наближаючи їх до інтелектуальних агентів, здатних взаємодіяти з ринком як складною, але керованою системою.

1.4 Постановка задачі дослідження

Задача побудови класифікаційної моделі для фінансового прогнозування постає в точці перетину кількох фундаментальних викликів: складності поведінки ринку, наявності шуму в даних, багатовимірності ознак і вимоги до високої точності в умовах обмеженого ризику. У фінансовому контексті класифікація набуває особливого значення, оскільки її результати безпосередньо впливають на ухвалення рішень, пов’язаних із реальними фінансовими втратами або прибутками. У межах цього дослідження розглядається задача визначення доцільного торгового рішення – купівлі, утримання або продажу активу, на основі аналізу поточного та історичного ринкового контексту.

Формулювання задачі класифікації вимагає чіткого визначення як вхідного простору, так і цільової змінної. Вхідний простір описується сукупністю ознак, отриманих із відкритих фінансових даних. Такі ознаки повинні не просто відображати числову динаміку, а бути інформативними з точки зору ринкової поведінки. Це передбачає побудову трансформованих рядів, похідних від технічних індикаторів, нормалізованих обсягів торгів, трендових характеристик, індексів волатильності тощо. Кожна ознака має бути змістовно обґрунтована і повинна підвищувати здатність моделі до відокремлення класів у багатовимірному просторі.

Цільова змінна, відповідно до обраної парадигми, є трьохкласовою категоріальною величиною: Buy / Hold / Sell. Її формування базується на відкладеному ринковому результаті, зафіксованому через визначений часовий інтервал після моменту прийняття рішення. Такий підхід дозволяє верифікувати, чи було в поточному стані ринку достатнє підґрунтя для формування активної дії. При цьому важливо уникати накладання інформації з майбутнього на процес формування ознак, щоб зберегти коректність моделі в умовах практичного використання.

Особливістю задачі є те, що формальна точність класифікації не є самодостатньою метою. Модель має бути не лише здатною правильно класифікувати стани, а й забезпечувати результат, що має фінансову цінність. Це означає, що оцінювання ефективності повинно включати як класичні метрики класифікації (точність, повнота, F1-міра), так і фінансово орієнтовані показники, що характеризують поведінку стратегії на основі прогнозів: накопичений прибуток, коефіцієнт Шарпа, рівень просідання, співвідношення прибутку до ризику. Такі багатовекторні оцінки дозволяють повноцінно оцінити практичну придатність кожного з алгоритмів [13].

Алгоритми, що підлягають порівнянню, включають класичні методи (логістичну регресію, дерева рішень, метод опорних векторів), ансамблеві моделі (випадкові ліси, градієнтний бустинг), а також багатошарові персептрони. Вибір цих моделей зумовлений їхньою репрезентативністю у фінансових задачах та здатністю до узагальнення складної структури вхідного простору.

Порівняння проводиться не лише за точністю прогнозу, але й за стабільністю результатів у часі, чутливістю до класового дисбалансу та здатністю витримувати зміну ринкової динаміки. Особливу увагу буде приділено дослідженню поведінки моделей у періоди високої волатильності, коли помилки класифікації є найбільш критичними. Такий підхід дозволить сформувати вичерпне уявлення про переваги й обмеження кожного з алгоритмів у контексті задачі прогнозування ринкових рішень.

2 МЕТОДОЛОГІЧНА БАЗА ДОСЛІДЖЕННЯ

Практичне дослідження у сфері фінансового прогнозування потребує чітко структурованої методології, яка поєднує наукову обґрунтованість, технічну реалізовуваність і можливість подальшої перевірки результатів. Методологічна конструкція включає визначення характеристик вхідних даних, способів їх трансформації, обґрунтування вибору моделей, критеріїв оцінки якості прогнозу, а також процедури експериментального порівняння.

Актуальність розробки надійної методичної основи зумовлена складністю досліджуваної галузі. Фінансові часові ряди характеризуються високою волатильністю, структурною нестабільністю та зміщенням у статистичних властивостях. Це викликає потребу в побудові системи, яка не лише навчається на історичних даних, але й здатна узагальнювати знання, зберігаючи ефективність у змінних ринкових умовах. Виходячи з цього, одним із ключових викликів стає створення репрезентативного простору ознак, що поєднує в собі як базову інформацію про ціну й обсяги, так і похідні індикатори з галузі технічного аналізу, які можуть виступати узагальненими маркерами ринкової поведінки.

Методологія побудована з урахуванням того, що фінансова класифікація вимагає спеціальних підходів до вибору цільової змінної. На відміну від класичних задач з фіксованою відповіддю, у цій роботі клас визначається відкладеним ефектом від прийнятого рішення, тобто прибутком або збитком, який міг би бути отриманий у разі виконання прогнозованої дії. Таким чином, задачі класифікації поєднуються з елементами оцінювання стратегічної поведінки, що ускладнює як етап підготовки даних, так і формування підходів до валідації [14].

Для того щоб уникнути методологічних спотворень при порівнянні моделей, усі алгоритми навчаються та тестуються в єдиному середовищі, з використанням одного й того ж набору ознак і однаково структурованого набору цільових класів. Це дозволяє виключити вплив сторонніх факторів і

зосередитись на реальних відмінностях у структурі моделей. Такий підхід забезпечує внутрішню валідність дослідження, дозволяючи інтерпретувати відмінності в результатах як наслідок характеристик самих моделей, а не методичних похибок у підготовці даних (рисунок 2.1).

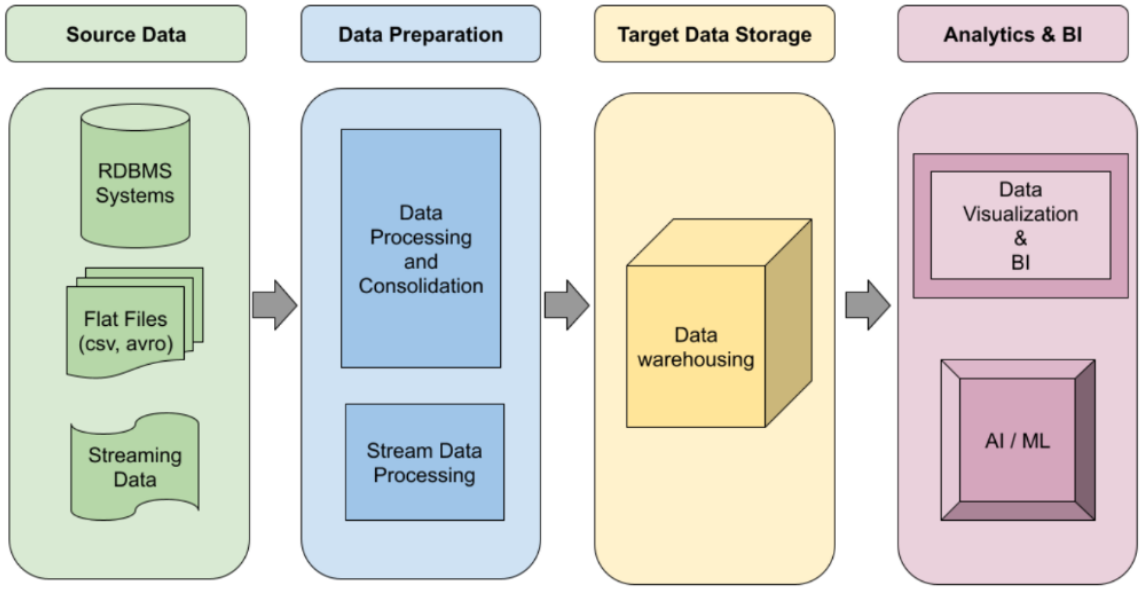

Рисунок 2.1 – Обробка даних і моделювання конвеєра

Особливе місце в методології займає вибір метрик оцінки якості моделей. Оскільки у фінансовому прогнозуванні важливе не лише теоретичне передбачення класу, а й його практичний вплив на результат, то оцінювання базується як на традиційних показниках точності, так і на фінансових індикаторах, які імітують результат торгівлі на основі прогнозу. Це дозволяє визначити, наскільки модель не лише класифікує правильно, але й приносить реальний результат у симульованих або наближених до реальності умовах [15].

Методологія передбачає побудову уніфікованого середовища backtesting, у якому кожна модель може бути підключена до однакової торгової логіки, що імітує просту стратегію з однаковими умовами:

стартовим капіталом, комісією, без кредитного плеча. Це дозволяє отримати порівнювані показники прибутковості, волатильності, просідання капіталу.

2.1 Вхідні дані та їх специфіка

2.1.1 Структура фінансових часових рядів (OHLCV)

Фінансові часові ряди є базовим джерелом даних для будь-якого типу аналітики, пов’язаної з динамікою ринку. Саме з них формуються уявлення про тенденції, зміни попиту і пропозиції, ринкову волатильність, поведінку трейдерів та інституційних гравців. Одним із найпоширеніших форматів подання фінансових даних є структура OHLCV, яка охоплює ключові параметри ринкової активності за фіксований часовий інтервал: Open, High, Low, Close та Volume. Такий формат не лише надає повну картину руху ціни в межах певного періоду, а й є критично важливим для побудови ознак у задачах машинного навчання [16].

Компонент Open означає ціну активу на початку обраного таймфрейму. Вона фіксує точку входу в період і слугує відправною точкою для аналізу зміни ринкового стану. High і Low описують відповідно максимальне і мінімальне значення ціни протягом інтервалу, відображаючи межі коливань ціни. Вони є основою для оцінки волатильності та сили імпульсів у межах обраного періоду. Close – найважливіший параметр, який фіксує ціну активу на завершення таймфрейму. Саме цей показник найчастіше використовується для побудови графіків, обчислення технічних індикаторів і формування трендових ознак. Volume, у свою чергу, відображає кількість або обсяг угод, укладених протягом інтервалу. Він дозволяє оцінити інтенсивність торгів та визначити фази ринку з підвищеною або зниженою активністю.

Наведена структура має перевагу над простими tick-даними, оскільки дозволяє агрегувати інформацію в зручному для аналізу вигляді. На основі

OHLCV-рядків будуються свічкові графіки, які є стандартом візуалізації динаміки ринку у більшості торгових платформ. Крім того, ця структура дозволяє легко адаптувати дані до будь-якої частоти, від хвилинної до тижневої, залежно від цілей дослідження.

Таймфрейм визначає, який обсяг інформації агрегується у межах одного запису. При зміні таймфрейму змінюється не тільки кількість рядків у датасеті, а й логіка поведінки самих показників. У коротких таймфреймах (1 хвилина, 5 хвилин) переважає локальний шум і випадкові коливання, а довгі (1 година, 1 день, 1 тиждень) дозволяють виявити макрорухи, тренди, зони накопичення або розподілу. Вибір таймфрейму тісно пов’язаний із типом моделі, що застосовується, та зі стилем торгівлі, на який орієнтується дослідник.

У задачах класифікації фінансових станів таймфрейм має особливе значення, оскільки впливає на інтерпретацію цільової змінної. В одному контексті «Buy»-сигнал, згенерований на основі п’ятихвилинного графіку, може мати короткотерміновий характер і втратити актуальність вже за кілька хвилин. У іншому випадку «Buy», побудований на щоденних спостереженнях, може відповідати середньостроковій або навіть довгостроковій позиції. Це означає, що вибір таймфрейму безпосередньо впливає на те, що саме модель вважає ринковим сигналом і як формується відповідна мітка у навчальному наборі.

Важливо враховувати проблему часової агрегації. При переході до більших таймфреймів частина локальних сигналів втрачається, а при роботі з надто дрібними модель ризикує навчитись на шумі замість закономірностей. Це ставить перед дослідником завдання підібрати такий рівень деталізації, який дозволить зберегти необхідний баланс між стабільністю й чутливістю до подій. З методологічної точки зору, таймфрейм є частиною гіперпараметрів, які потребують узгодження з цілями експерименту, глибиною моделі та частотою торгівлі.

Структура OHLCV також створює можливості для побудови похідних індикаторів, що базуються на співвідношенні між її компонентами. Такі індикатори, як зміна між Open і Close, діапазон High-Low, середня свічка, об’ємна волатильність, співвідношення ціни закриття до попередньої, є основою для генерації навчальних ознак. Їх використання дозволяє моделі не просто запам’ятовувати абсолютні значення, а навчатися на динаміці, тобто розуміти, у якому стані перебував ринок і до чого це могло призвести. Це робить OHLCV не лише джерелом «сирих» чисел, а повноцінною структурою, яка містить усю необхідну інформацію для побудови навчального процесу в задачах фінансової класифікації [17].

Структура OHLCV виконує роль базового блока у процесі аналізу фінансових ринків. Вона забезпечує гнучкість у виборі частоти, стабільність при обробці, змістовну повноту та придатність для побудови ознак. Саме тому вона стала де-факто стандартом не лише у трейдингу, а й у машинному навчанні, орієнтованому на фінансові дані.

2.1.2 Проблеми, пов’язані з фінансовими даними

Фінансові часові ряди, попри свою структурну уніфікованість, є одними з найскладніших типів даних для аналізу. Це зумовлено низкою властивостей, які не лише ускладнюють їх обробку, а й впливають на здатність моделей машинного навчання будувати точні й узагальнені прогнози. На відміну від природничих або технічних спостережень, дані фінансових ринків мають динамічний, адаптивний і нестійкий характер.

Однією з фундаментальних властивостей фінансових рядів є їхня нестаціонарність. У класичному розумінні, стаціонарний ряд повинен зберігати постійні характеристики, такі як середнє, дисперсія і кореляційна структура, незалежно від моменту спостереження. Проте в реальних фінансових даних ці характеристики змінюються у часі. Середнє значення ціни, волатильність, частота імпульсів і навіть базові взаємозв’язки між

змінними не залишаються стабільними. Це ускладнює побудову моделей, які ґрунтуються на припущенні про фіксовану статистичну структуру даних, і часто призводить до погіршення точності на тестових або майбутніх вибірках [18].

Нестаціонарність тісно пов’язана з явищем автокореляції, яке широко спостерігається у фінансових часових рядах. Поточне значення фінансового інструменту часто має залежність від попередніх спостережень. Хоч така залежність може бути короткостроковою, вона все ж порушує припущення незалежності спостережень, яке лежить в основі багатьох алгоритмів машинного навчання. У зв’язку з цим, моделі, що не враховують структуру автокореляції, можуть неправильно оцінювати значущість ознак і схильні до перенавчання.

Існують часові вікна, у межах яких ринки демонструють схожу поведінку протягом різних періодів. Це може стосуватися окремих днів тижня, годин доби або фаз місяця. Проте сезонність у фінансових даних часто носить нестабільний характер і може змінюватися під впливом глобальних факторів. Моделі, які не враховують періодичність або неправильно її інтерпретують, мають ризик переоцінити або недооцінити вплив часових патернів.

Особливу складність становить волатильність. У фінансовому контексті цей термін означає ступінь розкиду значень у ряді або амплітуду коливань цін. Волатильність не лише змінюється у часі, а й має властивість кластеризації. Це означає, що періоди високої активності змінюються фазами відносного затишшя, але розподілені вони нерівномірно. Моделі машинного навчання часто погано реагують на ці перепади, оскільки гіпотеза стабільності входів і розподілу шуму виявляється порушеною. У такі моменти поведінка ознак змінюється, і модель, навчена на стабільному сегменті, втрачає здатність до коректної генералізації.

Зовнішні події, психологічні фактори, непередбачувані інформаційні вкиди або навіть поведінка алгоритмічних систем можуть призвести до

локальних аномалій у часовому ряді, які не мають фундаментального підґрунтя. У таких умовах навіть найсильніші сигнали можуть бути приховані під впливом випадкових флуктуацій, а модель починає «вивчати» структуру, яка насправді не повторюється. Результатом стає зниження стійкості та збільшення ризику помилкових рішень [19].

Окрім цього, у фінансових даних часто присутні сплески активності, які важко передбачити на основі історичної інформації. Такі події, як раптове падіння курсу, злиття компаній, регуляторні оголошення або зміна монетарної політики, здатні миттєво змінити логіку ринку. Їх складно формалізувати в межах традиційного набору ознак, а їхній вплив не можна пояснити лише математичними параметрами часового ряду. Вони часто формують різкі злами у поведінці інструментів, що робить неможливою побудову стабільної моделі без включення зовнішнього контексту.

Сукупність цих факторів створює середовище, у якому застосування алгоритмів машинного навчання потребує спеціального підходу. Просте перенесення традиційних моделей без урахування структури ринку призводить до неточностей, надмірної впевненості у прогнозах та серйозних фінансових наслідків. Для підвищення точності й стабільності моделей необхідно не лише враховувати специфіку фінансових даних, а й будувати навчальний процес так, щоб він був стійким до змін динаміки, нестаціонарності та локальних аномалій.

2.1.3 Підготовка та трансформація вхідних даних

Процес підготовки даних, або preprocessing, відіграє ключову роль у створенні надійного класифікатора, оскільки помилки на цьому етапі неминуче вплинуть на стабільність і точність моделі.

Першим завданням у трансформації ринкових даних є очищення. У фінансових часових рядах часто трапляються пропущені значення, нечислові значення, дублі записів, неузгоджені часові мітки, а також

відхилення у форматах. У разі аналізу криптовалют або ринків, що працюють безперервно, особливу увагу слід приділяти часовим розривам через технічні збої або зміну API. Для забезпечення цілісності ряду відновлюють повну часову послідовність, інтерполюють або усувають пропуски, приводять усі дані до одного масштабу і синхронізують таймстемпи.

Після очищення даних здійснюється їх згладжування. У високочастотних рядах спостерігається значна кількість шуму, який перешкоджає моделі виявляти корисні закономірності. Щоб зменшити вплив випадкових коливань, застосовують ковзні середні, експоненціальне згладжування або фільтрацію, яка зберігає трендову складову. При цьому важливо зберігати баланс між пригніченням шуму і збереженням ключових коливань, що несуть інформацію для майбутнього прогнозу [20].

Одним із критичних етапів обробки є нормалізація даних. У фінансових наборах різні ознаки можуть мати різні масштаби, одиниці виміру або розподіли. Це створює труднощі для моделей, які базуються на відстанях або градієнтах. Для усунення цієї проблеми використовуються методи масштабування, такі як мінімакс-нормалізація або стандартизація. Вибір методу залежить від типу моделі, стабільності вибірки та наявності викидів. Ці трансформації дають змогу моделі оперувати даними на уніфікованому рівні, що підвищує точність і швидкість навчання.

Наступним кроком у формуванні навчальної вибірки є побудова лагів. Машинне навчання вимагає незалежних спостережень, але у фінансових часових рядах послідовні дані тісно пов’язані між собою. Щоб навчити модель враховувати цю залежність, створюються зміщені версії ознак, які містять значення за попередні кроки. Лагові змінні дозволяють моделі бачити контекст у часі й будувати прогнози на основі попередніх станів ринку. Їх кількість визначається експериментально з урахуванням горизонту прогнозу, інформативності та стабільності моделі.

Для підвищення інформативності вхідного простору додаються похідні ознаки. До них належать відносні зміни цін, об’ємні імпульси, кут нахилу тренду, розмах свічки та інші комбіновані характеристики. Вони будуються на базі первинних компонентів OHLCV та дозволяють моделі фіксувати не лише значення, а й темп, напрям і силу змін. За правильної побудови такі ознаки покращують здатність моделі відокремлювати класи, знижують вплив шуму й покращують здатність до узагальнення [21].

Останнім етапом у трансформації є формування вікон для прогнозу. Щоб змоделювати реальну ситуацію ухвалення рішення, навчальні приклади будуються з певного обсягу попередніх спостережень, на основі яких формується прогноз на наступний крок або певний горизонт у майбутньому. Такий підхід дає змогу моделі працювати з послідовностями, не порушуючи структури часових рядів. Вікна можуть бути фіксованими або змінними за довжиною, залежно від завдання і типу обраного класифікатора.

2.2 Формування ознак

Після завершення етапу попередньої обробки фінансових даних важливо перейти до побудови ознак, які виконують роль інформаційного підґрунтя для класифікаційних алгоритмів. Цей етап безпосередньо впливає на здатність моделі виокремлювати закономірності, розрізняти ринкові стани та ухвалювати рішення на основі історичної поведінки. Ознаковий простір повинен бути не лише репрезентативним, а й стабільним, адаптивним до змін ринкових умов і достатньо гнучким, щоб забезпечити можливість узагальнення. У фінансовому прогнозуванні важливо не лише надати моделі числові значення, а й трансформувати сировинні дані у форму, яка є семантично осмисленою, структурно зваженою і статистично значущою.

Формування ознак у задачах класифікації фінансових рішень базується на припущенні, що майбутня поведінка ринку певною мірою залежить від історичної динаміки цін, обсягів, волатильності та структури попередніх змін. У реальних ринкових умовах більшість сигналів формується не як реакція на одиничне значення, а як результат взаємодії кількох чинників у часовому просторі. Саме тому однією з центральних задач під час створення ознак є виявлення таких патернів, які мають стабільний причинно-наслідковий зв’язок із майбутніми рухами ринку [22].

Найчастіше для формування ознак застосовують технічні індикатори. Вони становлять собою формалізовані обчислювані структури, які інтерпретують взаємозв’язки між елементами OHLCV-даних та виявляють приховані властивості ринку. В основі більшості індикаторів лежать принципи статистичної згладженості, оцінювання трендових напрямів, аналіз темпів зміни ціни та вимірювання ступеня ринкової активності. Кожен індикатор відображає певний аспект ринкової поведінки, і їх використання у поєднанні дозволяє моделі бачити ринок під різними кутами (таблиця 2.1).

До однієї з найбільш поширених категорій належать трендові індикатори. Вони відображають напрям руху ціни на заданому часовому відрізку, дозволяючи визначити чи ринок перебуває в стані зростання, зниження або коливання без чітко вираженого тренду. Найбільш відомими представниками цієї категорії є ковзні середні. Вони дозволяють згладити локальні коливання ціни та виділити основну тенденцію. Використання двох ковзних середніх з різними періодами дозволяє визначити зміну фази ринку. Ще одним важливим трендовим індикатором є MACD, який вимірює різницю між коротко- та довгостроковими експоненціальними середніми. Він дозволяє виявити зміну імпульсу в ринку та може виступати як предиктор розвороту.

Таблиця 2.1 – Основні технічні індикатори, використані для формування ознак

Назва індикатора	Короткий опис	Інтерпретація
SMA (Simple Moving Average)	Проста ковзна середня ціни	Згладжує ціновий тренд
EMA (Exponential Moving Average)	Експоненціальна ковзна середня	Швидше реагує на зміни
RSI (Relative Strength Index)	Індекс відносної сили	Показує перекупленість або перепроданість ринку
BBWidth (Bollinger Bands Width)	Ширина смуг Болліджера	Оцінює волатильність
ROC (Rate of Change)	Темп зміни ціни	Визначає швидкість руху ринку
Volume	Об’єм торгів за період	Вказує на активність
MACD (Moving Average Convergence Divergence)	Різниця між короткою і довгою EMA	Сигнал розвороту тренду

У побудові ознак часто використовуються осцилятори. Ці індикатори дозволяють оцінити ступінь перекупленості або перепроданості ринку. Вони коливаються в межах певного діапазону і дозволяють визначити моменти, коли ціна активу відхиляється від своєї умовної рівноваги. RSI оцінює відносну силу недавніх прибутків і втрат. Стохастичний осцилятор порівнює поточну ціну закриття з діапазоном цін за визначений період часу. Включення осциляторів у ознаковий простір дозволяє моделі краще реагувати на імпульсні аномалії, реверсні сигнали та короткострокові збої у ринку [23].

Важливе значення мають індикатори волатильності. Вони характеризують амплітуду коливань ціни та рівень невизначеності, властивий поточному ринковому стану. Найбільш поширеними є ATR, який вимірює середній істинний діапазон коливань, та BBWidth, що відображає ширину смуг Болліджера і використовується для визначення періодів

стиснення або розширення волатильності. Наявність таких індикаторів у моделі дозволяє точніше інтерпретувати сигнали в умовах змінного ризику та уникати хибних рішень, спричинених надмірною нестабільністю.

До окремого класу можна віднести часові ознаки. Незважаючи на те, що вони не є технічними індикаторами в прямому сенсі, вони надають важливу інформацію про структурні ритми ринку. Серед таких ознак можна виокремити порядковий номер години або дня тижня, фазу торгової сесії, момент публікації новин чи макроекономічних даних. Ринок демонструє підвищену активність у визначені часові інтервали, і включення таких ознак дозволяє моделі враховувати сезонні закономірності без необхідності використовувати складні рекурентні структури.

При побудові ознак важливо дотримуватись принципу несуперечності, коли жодна ознака не повинна містити інформації про майбутнє. Це особливо важливо у фінансовому контексті, де навіть незначне інформаційне зміщення вперед може зробити модель непридатною для практичного використання. Усі обчислення виконуються на основі значень, доступних на момент прийняття рішення, що гарантує коректність симуляцій і справедливість навчального процесу.

Надмірна кількість похідних індикаторів може призвести до мультиколінеарності, коли ознаки дублюють одна одну і знижують ефективність навчання. З іншого боку, правильне поєднання ознак із різних категорій дозволяє моделі фіксувати складні патерни, які базуються на взаємодії різних ринкових сигналів. Для цього часто використовують процедури відбору ознак, а також аналіз кореляційних матриць на етапі валідації [24].

У результаті формування ознак є не просто технічним етапом обробки даних, а складовою стратегічного проєктування моделі. Саме від вибору ознакового простору залежить те, що саме модель сприйматиме як сигнал і які патерни будуть для неї доступними.

У складному середовищі фінансового ринку ознаки мають відігравати роль фільтрів, що відсікають шум і фокусуються на інформативних аспектах динаміки ціни. Лише за цієї умови класифікаційна модель зможе працювати стабільно, узагальнювати закономірності та генерувати рішення, що мають практичну цінність.

2.3 Формалізація задачі та обрані алгоритми

2.3.1 Формалізація задачі класифікації в фінансовому контексті

Ключовою умовою коректної побудови моделі класифікації є точна формалізація задачі в контексті ринкової логіки. Метою дослідження є навчання моделі, яка здатна на основі поточного ринкового стану передбачити доцільність одного з трьох рішень: відкриття довгої позиції, утримання існуючої або закриття позиції. Такий підхід зумовлює необхідність багатокласової постановки задачі, де кожна мітка відповідає одному з можливих торгових сценаріїв. На відміну від бінарної класифікації, у якій прогноз обмежується лише купівлею або продажем, багатокласовий підхід дає змогу моделі інтерпретувати і нейтральні ринкові ситуації, коли жодна дія не є оптимальною. Це наближає поведінку моделі до реальної логіки прийняття рішень у трейдингу.

Формування цільової змінної здійснюється на основі майбутньої зміни ціни після певного фіксованого проміжку часу. Для цього використовують концепцію відкладеного прибутку, що означає різницю між ціною активу на момент прийняття рішення та ціною через n періодів у майбутньому. Якщо така різниця перевищує визначене позитивне порогове значення, спостереженню приписується мітка купівлі. Якщо зміна ціни негативна і перевищує за модулем аналогічний поріг, міткою є продаж. У випадку, коли зміна незначна або лежить у межах нульової зони, спостереженню приписується клас утримання. Таким чином, формування

класів відбувається на основі спрямованості і сили очікуваного руху ринку, а сам алгоритм прогнозує не факт зміни ціни, а доцільність певної торгової дії.

Часовий горизонт, на який здійснюється прогноз, визначає глибину відкладеного прибутку і має критичний вплив на розмітку. Якщо горизонт надто короткий, мітки можуть відповідати випадковим коливанням або шуму, і модель буде навчатися на даних, що не мають стійкої закономірності. Якщо горизонт надто довгий, кількість однозначних сигналів зменшується, і класифікація стає надмірно загальною.

Застосування трьох класів пояснюється не лише ринковою логікою, а й необхідністю моделі навчитись розрізняти ситуації, у яких активна дія є недоцільною. На реальних фінансових даних більшість моментів часу не супроводжується чіткими сигналами до купівлі чи продажу. Включення класу утримання дозволяє моделі формувати більш обережну політику поведінки, уникаючи надлишкової торгівлі та зниження прибутковості через комісії або хибні входи [25].

Окремою проблемою є суттєвий дисбаланс між кількістю спостережень у класах. Через природу фінансового ринку класи утримання зазвичай мають значно більшу кількість прикладів, ніж класи купівлі або продажу. Це призводить до зміщення моделі в бік переоцінки нейтральних сценаріїв і зниження її здатності вчасно виявляти сигнали до дії. Щоб нейтралізувати цей ефект, застосовуються вагові коефіцієнти до функції втрат або методи ресемплінгу, які дозволяють збалансувати внесок кожного класу в процес навчання. Завдяки цьому модель не лише зберігає чутливість до рідкісних подій, а й краще узагальнює залежності між ринковим контекстом і прогнозованим рішенням.

У результаті постановка задачі класифікації базується на ринково обґрунтованій логіці та дозволяє моделі працювати у форматі, наближеному до реального торгового середовища.

2.3.2 Критерії вибору алгоритмів

Вибір моделей для порівняння у фінансовій класифікації ґрунтується на необхідності охопити широкий спектр підходів, що репрезентують різні класи алгоритмів машинного навчання. Такий підхід дозволяє не лише порівняти абсолютну ефективність кожної моделі, а й оцінити здатність різних типів архітектур до узагальнення, адаптації до ринкової нестійкості та реагування на складні нелінійні патерни у вхідних даних.

До аналізу включено чотири моделі: логістичну регресію, метод опорних векторів, випадковий ліс і багатошаровий персептрон. Вони репрезентують відповідно лінійні, ядрові, ансамблеві та нейромережеві підходи. Такий вибір забезпечує як мінімальну необхідну різноманітність алгоритмів, так і збереження контрольованості експерименту, оскільки кожна з моделей має усталене теоретичне підґрунтя та підтверджену ефективність у задачах фінансового прогнозування.

Логістична регресія є лінійною моделлю, яка встановлює ймовірнісну залежність між входами і класами за допомогою логістичної функції. Її включення в експеримент обґрунтоване тим, що вона виступає як базова модель з високим рівнем інтерпретованості. Вона дозволяє оцінити здатність до лінійного розділення класів і є відправною точкою для порівняння з більш складними алгоритмами. Модель не має глибини, але демонструє високу швидкість обчислень і є стійкою до переобучення за умови правильного нормування даних.

Метод опорних векторів представляє клас моделей, які формують розділення класів через побудову гіперплощини у просторі ознак. Завдяки використанню ядрових функцій модель здатна перетворити вхідний простір у вищий вимір, у якому класи стають лінійно роздільними. SVM демонструє високу ефективність на невеликих і середніх за розміром вибірках, особливо у випадках, коли кордон між класами складний або нечіткий. Її здатність до

регуляризації дає змогу уникати переобучення, а контроль параметрів дозволяє налаштовувати баланс між точністю і узагальнювальністю.

Випадковий ліс, як представник ансамблевих методів, ґрунтується на комбінації великої кількості слабких моделей, а саме дерев рішень, кожне з яких навчається на випадковому підмноженні даних. Така архітектура забезпечує високу стійкість до шуму, підвищену стабільність результатів і добру здатність до роботи з гетерогенними ознаками. Модель є інтерпретованою на рівні окремих дерев, що дозволяє досліджувати важливість ознак, і є придатною для задач із помірною нелінійністю.

Багатошаровий персептрон входить до класу глибоких моделей, що мають здатність моделювати складні взаємозв’язки між ознаками. На відміну від традиційних нейромереж, він застосовується у задачах, де потрібне точне розпізнавання патернів на обмеженому обсязі вхідних даних. MLP дозволяє моделювати складні нелінійні залежності між ринковими станами та відповідними діями.

При правильному налаштуванні архітектури, виборі функцій активації та розмірі прихованих шарів модель здатна відтворювати логіку трейдера на рівні ухвалення рішення, навіть за відсутності явних ознак у первинному сигналі. Вибір моделей ґрунтується на прагненні створити систему, в якій будуть представлені як прості, так і високопотужні моделі. Це дозволяє перевірити гіпотезу про те, чи складність архітектури дійсно покращує прогноз, чи достатньо простих моделей за умови якісної підготовки даних [26]. Такий підхід також дає змогу оцінити, наскільки ефективно кожен клас алгоритмів справляється з особливостями фінансових часових рядів.

2.3.3 Узгодження навчального середовища

Порівняльне дослідження ефективності класифікаційних алгоритмів потребує створення уніфікованого навчального середовища, яке виключає

вплив сторонніх чинників на результати моделювання. Таке середовище має забезпечувати рівні умови для кожної моделі, включно з однаковим набором ознак, ідентичними процедурами попередньої обробки та розмітки, а також єдиними правилами розподілу даних на тренувальну і тестову вибірки. Лише за дотримання цих умов можливо провести об’єктивне порівняння продуктивності моделей з мінімальним впливом методичних похибок.

Базовою архітектурою реалізації є pipeline, який включає чіткі етапи: попередня обробка даних, формування ознак, генерація цільової змінної, масштабування та нормалізація, навчання моделі, валідація, тестування і фінальна оцінка результатів. Кожен із цих етапів реалізується у вигляді модульної структури, що дозволяє легко підключати нові моделі без зміни логіки підготовки даних. Усі алгоритми навчаються на одних і тих самих прикладах, що унеможливлює ситуацію, коли кращі результати зумовлені кращою вибіркою, а не внутрішньою якістю моделі.

Окрему увагу приділено контролю за ефектом перенавчання, що особливо критично у випадку роботи з високочутливими до шуму фінансовими даними. Для оцінки стабільності моделей застосовується крос-валідація, яка забезпечує перевірку моделі на різних фрагментах часових рядів. Оскільки у фінансових задачах дані мають часову залежність, класичну k-fold крос-валідацію замінено на покрокову валідацію за принципом expanding window. Такий підхід дозволяє моделі навчатися на зростаючій кількості даних і тестуватися на подальших спостереженнях без порушення хронології [27].

Регуляризація моделей здійснюється через налаштування відповідних гіперпараметрів. Для логістичної регресії та SVM використовуються параметри, що обмежують складність моделі. У випадку Random Forest обмежується глибина дерев та кількість вибраних ознак на вузол. Для MLP реалізовано механізм dropout та обмеження розміру шарів, що дозволяє уникнути перенавчання на тренувальних даних.

Усі розрахунки ознак та цільових міток відбуваються в межах доступного на момент прогнозу історичного інтервалу. Це гарантує відсутність time leakage і дозволяє симулювати реальні умови, у яких система працюватиме під час застосування. Перевірка на витік здійснюється шляхом аналізу структури датафрейму, перевірки часових індексів та розмітки міток на основі зміщених спостережень.

Для фінальної оцінки результатів використовується спільний набір метрик, який включає як класифікаційні показники (точність, F1-міра, recall), так і фінансові (накопичений прибуток, коефіцієнт Шарпа, просідання капіталу). Всі результати оцінюються на одній тестовій вибірці, яка залишається незмінною для кожного експерименту. Такий підхід забезпечує не лише чесність порівняння, а й дає змогу отримати повну картину про практичну доцільність використання кожної з моделей у реальному середовищі алгоритмічного трейдингу.

2.4 Методи оцінювання якості моделей

Стандартні метрики точності, які застосовуються в задачах класифікації, не завжди відображають реальну практичну ефективність прогнозу в умовах фінансового середовища. Саме тому доцільно поєднувати класичні показники якості класифікації з фінансово-орієнтованими метриками, що дозволяють визначити вплив прогнозу на результативність торгової стратегії.

Класичні метрики якості класифікації становлять основу оцінки кожної моделі в системі машинного навчання. Вони використовуються для кількісного визначення точності прогнозів та ідентифікації проблем, пов’язаних з дисбалансом класів, недостатньою чутливістю або помилковими позитивними сигналами. Одним із найбільш базових показників є точність класифікації, яка визначається як частка правильно класифікованих прикладів серед усіх спостережень. Вона зручна для першої

оцінки якості моделі, однак є недостатньо інформативною у випадках, коли розподіл класів є нерівномірним. У фінансових задачах, де більшість прикладів можуть бути віднесені до класу утримання, модель із високою точністю може виявитися абсолютно непридатною, якщо вона ігнорує рідкісні, але ключові сигнали до дії [28].

Саме тому до системи оцінювання включаються інші метрики, що дозволяють більш глибоко аналізувати поведінку моделі. Precision або точність позитивного прогнозу визначає, яка частка передбачених позитивних спостережень дійсно належить до позитивного класу. У контексті фінансового трейдингу це дає змогу оцінити ризик хибних входів, які призводять до втрат. Recall або повнота характеризує здатність моделі виявляти всі релевантні сигнали до дії. Високий показник recall означає, що модель не пропускає прибуткові можливості, навіть якщо водночас зростає кількість хибнопозитивних сигналів. Для досягнення балансу між точністю і повнотою застосовується F1-міра, яка є гармонійним середнім цих двох показників і дозволяє оцінити загальну ефективність моделі при прийнятті рішень.

Окрім числових показників, важливим інструментом оцінювання є матриця помилок, або confusion matrix. Вона дозволяє візуально відобразити кількість правильних і неправильних класифікацій по кожному класу. З її допомогою можна виявити, чи існує системне зміщення в бік певного класу, які саме переходи між класами є найбільш проблемними, і де саме модель припускається помилок. У задачах з трьома класами така матриця дозволяє відстежити, чи часто модель плутає сигнали купівлі з утриманням або продажу з нейтральним станом, що має критичне значення при формуванні торгових стратегій [29].

Разом із класифікаційними метриками у фінансовому контексті обов’язковим є використання оцінок, що характеризують економічну ефективність моделі. Основним показником у цій групі є накопичений прибуток, який визначається як сума всіх змін капіталу, зумовлених діями,

що були прийняті на основі прогнозів моделі. Цей показник дозволяє оцінити, чи трансформуються прогнозовані сигнали у реальні позитивні результати. Однак прибуток сам по собі не є достатнім критерієм, оскільки не враховує рівень ризику, необхідного для його досягнення.

Для глибшої інтерпретації використовується коефіцієнт Шарпа, який визначає співвідношення між середнім доходом і стандартним відхиленням прибутковості. Він дозволяє оцінити ефективність стратегії з урахуванням волатильності. Високе значення цього коефіцієнта вказує на стабільну прибутковість при відносно низькому ризику, що є бажаним у реальному трейдингу. Ще одним важливим показником є максимальне просідання капіталу, яке вимірює найбільшу відносну втрату за обраний період. Ця метрика дозволяє оцінити потенційні ризики стратегії і визначити її стійкість до несприятливих умов ринку.

Доповненням до цього є показники стабільності, які характеризують рівномірність зростання капіталу в часі. Стабільність у фінансовій моделі означає, що прибутковість формується не випадковими сплесками, а як результат послідовної генерації якісних сигналів. Це дозволяє моделі бути надійною у довгостроковій перспективі, що особливо важливо для інституційного застосування. Такі показники дозволяють уникнути помилкових висновків про ефективність моделі, які можуть виникати при оцінці за короткий період або в умовах випадкових трендів.

2.5 Структура експерименту та обґрунтування вибору

Навчання всіх моделей здійснюється на одному і тому ж датасеті, який сформовано з історичних фінансових даних, попередньо оброблених і збагачених технічними індикаторами та часовими ознаками. Поділ даних на тренувальну та тестову вибірки виконується з дотриманням часової послідовності, що виключає можливість потрапляння майбутньої інформації у процес навчання. Цей підхід дозволяє уникнути викривлень,

пов’язаних з перемішуванням даних, і моделює ситуацію, в якій модель приймає рішення на основі знань, сформованих до моменту прогнозу [30].

Тестування моделей проводиться в умовах симульованої торгівлі, де результати класифікації інтерпретуються як торгові дії. Для кожного прогнозу класу Buy, Hold або Sell визначається відповідна дія з портфелем: відкриття довгої позиції, утримання або закриття активу. Умови симуляції є фіксованими для всіх моделей: не використовуються кредитні плечі, комісії встановлено на однаковому рівні, обсяги операцій нормалізовані. Такий підхід дозволяє зіставляти не лише точність прогнозів, а й їх фінансовий ефект – прибуток, просідання, стабільність капіталу.

Оцінка кожної моделі відбувається за двома групами метрик: класифікаційними та фінансовими. Така багатовимірна оцінка дозволяє виявити моделі, які хоч і демонструють високу точність на тестових даних, але не забезпечують реального прибутку або є нестабільними в динаміці. Система оцінювання формує загальний підхід до вимірювання практичної цінності моделі, орієнтуючи аналіз не лише на математичну точність, а й на економічну доцільність.

Вибір моделей, включених до експерименту, є обґрунтованим як з наукової, так і з прикладної точки зору. Кожна з них представляє окремий клас алгоритмів, що мають різну складність, інтерпретованість і гнучкість. Логістична регресія є базовим еталоном, на який орієнтується порівняння інших підходів. SVM дозволяє моделювати складні межі між класами в обмеженому ознаковому просторі. Random Forest додає елемент ансамблевості, що підвищує стійкість до шуму. MLP виступає як представник нейромережевих структур, які здатні моделювати складні нелінійні патерни у поведінці ринку.

Завдяки цьому структура експерименту надає можливість сформулювати висновки про межі ефективності різних класів підходів, їх практичну цінність і потенціал для застосування у реальних умовах автоматизованої торгівлі.

3 РОЗРОБКА КЛАСИФІКАЦІЙНИХ МОДЕЛЕЙ ДЛЯ ФІНАНСОВОГО ПРОГНОЗУВАННЯ

3.1 Вибір та підготовка історичних біржових даних (OHLCV)

Для побудови ефективної системи класифікації торгових сигналів у цій роботі було використано великий відкритий датасет «Bitcoin Historical Datasets 2018–2024» [31], доступний на платформі Kaggle. Даний ресурс містить детальні історичні біржові котирування криптовалюти Bitcoin за п’ятирічний період, отримані через офіційне API Binance. Такі дані є ідеальним середовищем для апробації алгоритмів фінансового прогнозування, оскільки вони охоплюють різні ринкові фази – від стадій стрімкого зростання до періодів тривалих спадів та консолідацій.

Початковий масив містить класичні для фінансових часових рядів поля OHLCV (Open, High, Low, Close, Volume), а також точний часовий штамп (timestamp) для кожного інтервалу. Така структура дає змогу не лише відтворити історичний рух ринку в деталях, а й генерувати додаткові аналітичні ознаки (технічні індикатори, лаги, ознаки волатильності тощо), що істотно розширює інформаційний зміст даних. Особливо важливою є наявність об'єму (Volume), оскільки це дає змогу аналізувати ринкову активність і виявляти потенційні фази маніпуляцій або накопичення.

Перед початком моделювання було проведено низку обов’язкових процедур з попередньої обробки даних. По-перше, всі часові значення (Open time) були переведені у формат datetime для подальшої агрегації та побудови часових ознак. Було впорядковано записи за хронологією, щоб уникнути потенційного data leakage при формуванні тренувальної та тестової вибірок. Далі здійснено очистку від пропусків – всі відсутні значення заповнювалися шляхом прямої інтерполяції (forward fill), після чого були видалені будь-які залишкові пропуски, що могли залишитися у датасеті. Для забезпечення репрезентативності дослідження фокус було зроблено на інтервалі з 2018 по

2024 рік, що дало можливість охопити ключові макротренди та екстремальні ринкові рухи.

3.2 Розрахунок технічних індикаторів та формування ознак

Важливою складовою підготовки фінансових часових рядів до моделювання є розрахунок додаткових аналітичних ознак на основі базових даних OHLCV. Саме грамотне формування ознак дозволяє надати моделі більше інформації про ринкові патерни та динаміку, що у підсумку підвищує якість прогнозу та стабільність класифікації торгових сигналів.

На першому етапі до кожного запису у вихідному датафреймі додавались так звані технічні індикатори. Вони традиційно використовуються у фінансовому аналізі та трейдингу, оскільки узагальнюють складну динаміку цін у вигляді компактних числових показників.

Були розраховані найпопулярніші технічні індикатори, зокрема проста (SMA) та експоненціальна (EMA) ковзні середні, індекс відносної сили (RSI), осцилятор MACD, смуги Болліндже́ра, ширина смуг Болліндже́ра, Rate of Change (ROC), а також ковзна середня для об'єму. На рисунку 3.1 наведено приклад вигляду датафрейму після додавання технічних індикаторів: видно, що на початкових кроках частина значень є пропущеними (NaN), що пояснюється особливостями розрахунку індикаторів із певним вікном згладжування.

Для подальшої роботи всі рядки з пропущеними значеннями було видалено (рисунок 3.2), що дозволило забезпечити цілісність датасету для машинного навчання й уникнути викривлення результатів. Додаткові ознаки значно розширюють інформативність даних: наприклад, комбінація ковзних середніх та RSI дозволяє моделі відстежувати як довгостроковий тренд, так і короткострокову перекупленість чи перепроданість активу.

```
Перші рядки обробленого датафрейму:
            Open time      Open      High       Low     Close      Volume
0 2018-01-01 00:00:00  13715.65  13715.65  13400.01  13529.01  443.356199
1 2018-01-01 01:00:00  13528.99  13595.89  13155.38  13203.06  383.697006
2 2018-01-01 02:00:00  13203.00  13418.43  13200.00  13330.18  429.064572
3 2018-01-01 03:00:00  13330.26  13611.27  13290.00  13410.03  420.087030
4 2018-01-01 04:00:00  13434.98  13623.29  13322.15  13601.01  340.807329

Інформація про датафрейм:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 64967 entries, 0 to 64966
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Open time  64967 non-null  datetime64[ns]
 1   Open       64967 non-null  float64
 2   High       64967 non-null  float64
 3   Low        64967 non-null  float64
 4   Close      64967 non-null  float64
 5   Volume     64967 non-null  float64
dtypes: datetime64[ns](1), float64(5)
memory usage: 3.0 MB
```

Рисунок 3.1 – Датафрейм після додавання технічних індикаторів

```
3 індикаторами:
       Close        sma_14        ema_14  ...  bb_width       roc  volume_sma
0   13529.01           NaN           NaN  ...       NaN       NaN         NaN
1   13203.06           NaN           NaN  ...       NaN       NaN         NaN
2   13330.18           NaN           NaN  ...       NaN       NaN         NaN
3   13410.03           NaN           NaN  ...       NaN       NaN         NaN
4   13601.01           NaN           NaN  ...       NaN       NaN         NaN
5   13558.99           NaN           NaN  ...       NaN       NaN         NaN
6   13780.41           NaN           NaN  ...       NaN       NaN         NaN
7   13570.35           NaN           NaN  ...       NaN       NaN         NaN
8   13499.99           NaN           NaN  ...       NaN       NaN         NaN
9   13616.99           NaN           NaN  ...       NaN       NaN         NaN
10  13570.01           NaN           NaN  ...       NaN  0.303052         NaN
11  13220.56           NaN           NaN  ...       NaN  0.132545         NaN
12  13172.42           NaN           NaN  ...       NaN -1.183480         NaN
13  13017.00  13434.286429  13398.407246  ...       NaN -2.930866  357.897815
14  13211.39  13411.599286  13373.471614  ...       NaN -2.864640  375.691469
15  13247.00  13414.737857  13356.608732  ...       NaN -2.300983  378.789680
16  13018.00  13392.439286  13311.460901  ...       NaN -5.532564  374.384608
17  13022.00  13364.722857  13272.866114  ...       NaN -4.040795  360.704965
18  13135.00  13331.436429  13254.483966  ...       NaN -2.703632  353.816353
19  13240.37  13308.677857  13252.602103  ...  0.067186 -2.765809  342.943199
```

Рисунок 3.2 – Видалення рядків з пропущеними значеннями

На наступному етапі було реалізовано автоматизоване формування лагових та часових ознак. Лагові ознаки це попередні значення цін або індикаторів, які дозволяють моделі аналізувати динаміку зміни ринку не тільки в поточний момент, а й з урахуванням кількох попередніх періодів.

У лістингу 3.1 наведено фрагмент коду, що додає три лаги для ціни закриття та RSI, а також автоматично формує часові ознаки: годину, день тижня, місяць та ознаку вихідного дня. Такі ознаки дають змогу моделі враховувати циклічність ринку, можливі сезонні ефекти чи зміни волатильності в різний час доби.

Лістинг 3.1 – Функція для додавання лагових та часових ознак

```
def add_lags_and_time_features(df, n_lags=3):
    for lag in range(1, n_lags+1):
        df[f'close_lag_{lag}'] = df['Close'].shift(lag)
        df[f'rsi_lag_{lag}'] = df['rsi_14'].shift(lag)
    df['hour'] = df['Open time'].dt.hour
    df['day_of_week'] = df['Open time'].dt.dayofweek
    df['month'] = df['Open time'].dt.month
    df['is_weekend']                                    =
df['day_of_week'].isin([5,6]).astype(int)
    return df
```

Особливої уваги у задачах класифікації потребує формування цільової змінної. Класи Buy, Hold та Sell визначалися на основі майбутнього відсоткового приросту ціни через певний горизонт. Якщо приріст ціни перевищував встановлений поріг, такий випадок позначався як Buy, якщо був менше –1%, як Sell, інакше – як Hold. Це дозволяє адаптувати модель до реальних сценаріїв трейдингу, де важливе не просто спрямування руху, а його сила та значущість. На рисунку 3.3 наведено приклад сформованої цільової змінної у кінці препроцесингу.

```
Після видалення NaN:
            Open time      Open      High  ...  bb_width       roc  volume_sma
25 2018-01-02 01:00:00  13769.98  13800.00  ...  0.069754  2.664754  378.271308
26 2018-01-02 02:00:00  13611.93  13678.33  ...  0.064857  3.871562  410.702884
27 2018-01-02 03:00:00  13520.32  13539.98  ...  0.063221  2.547842  426.062812
28 2018-01-02 04:00:00  13353.78  13480.84  ...  0.063903 -0.058546  447.487923
29 2018-01-02 05:00:00  13129.03  13244.99  ...  0.061535 -0.554516  459.840120
```

Рисунок 3.3 – Сформована цільова змінна

Лістинг 3.2 – Функція для створення цільової змінної (Buy/Hold/Sell)

```
    def add_target_label(df, horizon=10, threshold=0.01):
        df['future_return'] = (df['Close'].shift(-horizon) - df['Close']) / df['Close']
        def label(row):
            if row['future_return'] > threshold:
                return 'Buy'
            elif row['future_return'] < -threshold:
                return 'Sell'
            else:
                return 'Hold'
        df['target'] = df.apply(label, axis=1)
        df = df.drop(columns=['future_return'])
        df = df.iloc[:-horizon]
        return df
```

Розрахунок технічних індикаторів, побудова лагових, часових та цільових ознак створюють основу для глибокого аналізу ринку з використанням сучасних методів машинного навчання. Саме багатий набір інформативних ознак дозволяє навчати складні моделі (нейронні мережі та ансамблеві алгоритми), що враховують як класичні технічні патерни, так і нетривіальні зв’язки у часових рядах. Цей підхід забезпечує високу адаптивність та стійкість моделей до різних сценаріїв ринкової динаміки. Повний код передобробки даних наведено у додатку Б.

3.3 Побудова класифікаційних моделей у Python

Після завершення підготовки даних наступним ключовим етапом став процес побудови класифікаційних моделей із використанням сучасних інструментів машинного навчання в середовищі Python.

Для реалізації цього завдання було створено гнучкий пайплайн, який поєднує всі необхідні кроки обробки, інженерії ознак і навчання моделей із подальшим тестуванням.

На першому етапі було організовано формування ознак та цільової змінної відповідно до результатів попередніх підрозділів. Було використано датасет, що містить усі необхідні технічні та часові індикатори.

Перед початком навчання модель, відповідно до загальноприйнятої практики роботи з часовими рядами, було розбито на тренувальну та тестову вибірки. Для цього застосовувався функціонал train_test_split з параметром stratify, що дозволяє забезпечити рівномірний розподіл класів у вибірках і підвищує достовірність оцінки якості моделі.

Розмір навчальної та тестової підмножин було визначено у співвідношенні приблизно 70/30. На рисунку 3.4 наведено приклад діапазонів train- та test-вибірок, які охоплюють періоди з 2018 по 2022 роки (навчання) та з 2022 по 2025 (тестування). Такий підхід гарантує, що алгоритми тестуються на ще «невідомих» майбутніх даних, максимально наближених до реальних умов експлуатації моделі.

```
Name: count, dtype: int64
Train size: 38959, Test size: 25973
Train period: 2018-01-02 01:00:00 → 2022-06-18 08:00:00
Test period: 2022-06-18 09:00:00 → 2025-06-04 14:00:00
```

Рисунок 3.4 – Приклад діапазонів train- та test-вибірок

Наступним етапом стала нормалізація ознак, оскільки фінансові ряди зазвичай містять значення з різними масштабами та одиницями виміру, що може негативно вплинути на стабільність і швидкість збіжності моделей. Для вирівнювання масштабів застосовувався стандартний підхід масштабування (StandardScaler), який центрує ознаки відносно нуля та нормалізує їх до стандартного відхилення.

З метою зменшення розмірності простору ознак та виділення найбільш інформативних змінних у пайплайні застосовувався крок відбору ознак (feature selection) за допомогою методу SelectKBest із критерієм дисперсійного аналізу (f_classif). Це дозволило залишити у моделі лише ті ознаки, які мають найбільший вплив на результат класифікації, уникнути мультиколінеарності та прискорити навчання без втрати якості.

Для вирівнюванню дисбалансу класів застосовано алгоритм SMOTE, який синтетично збільшує кількість зразків для менш представлених класів. Це суттєво підвищує здатність моделі розпізнавати рідкісні сигнали, не жертвуючи якістю для домінуючого класу Hold.

Після формування збалансованих та нормалізованих підмножин даних було реалізовано тренування моделей. У якості базового класифікатора застосовувалася логістична регресія з підбором гіперпараметрів через GridSearchCV, що дозволяє оптимізувати конфігурацію моделі за рахунок автоматичного перебору різних значень параметрів та багатошарові нейронні мережі (MLP), застосовувався єдиний пайплайн з аналогічними кроками підготовки та оптимізації. Це забезпечує коректність та об’єктивність порівняння результатів усіх підходів.

Оцінка ефективності кожної моделі здійснювалася на відкладеній тестовій вибірці з використанням метрик точності (accuracy), середньої зваженої F1-міри (macro F1) та аналізу матриці помилок. Для кожного класу окремо оцінювалися precision, recall та F1-score, що дозволяє комплексно характеризувати здатність моделі до розпізнавання різних торгових сигналів.

3.4 Проведення навчання та backtesting у єдиному середовищі

Після реалізації етапів попередньої обробки, формування ознак та балансування класів було створено інтегроване програмне середовище, що дозволяє автоматизовано здійснювати повний цикл навчання, валідації та тестування моделей. Такий підхід забезпечує не лише зручність у проведенні численних експериментів, а й гарантує відтворюваність результатів і об’єктивність порівняння різних алгоритмів.

Усі основні кроки, від завантаження сирих біржових даних, розрахунку технічних індикаторів, масштабування ознак і балансування вибірки до навчання моделей і подальшої оцінки їхньої якості, були реалізовані у межах одного скрипта на Python. Це дозволило ефективно керувати експериментом і легко варіювати як гіперпараметри, так і підходи до побудови ознак чи вибору моделей.

Ключовою складовою стала реалізація backtesting, перевірки моделі на історичних даних, відмінних від навчальної вибірки. На відміну від класичного розділення на train/test, саме послідовне тестування на «майбутньому», яке не було доступне під час навчання, дозволяє максимально наблизити оцінку якості моделі до реальних умов експлуатації у торгових стратегіях. Backtesting дає змогу імітувати процес прийняття рішень в режимі «онлайн» і таким чином виявити, як поведінка моделі змінюється на різних фазах ринку.

У створеному середовищі весь експериментальний процес було організовано як єдиний цикл: спочатку проводиться підготовка даних і навчання моделей на train-вибірці, далі виконується прогнозування та детальний аналіз якості на відкладених тестових даних. Результати моделювання автоматично виводяться у вигляді табличних звітів (classification report), матриць помилок (confusion matrix) та графічних інтерпретацій, що дозволяє всебічно оцінити переваги й обмеження кожного підходу.

Всі обрані моделі (Logistic Regression, Random Forest, XGBoost, MLP) піддавались backtesting у ідентичних умовах, що забезпечує їхню порівнюваність і дає змогу зробити коректні висновки щодо відносної ефективності кожної, вихідний код програми наведено у додатку А.

3.5 Порівняльний аналіз ефективності моделей

Після реалізації повного циклу підготовки історичних фінансових даних, побудови ознак, балансування вибірок і налаштування єдиного середовища для навчання та тестування, наступним ключовим етапом дослідження стало проведення порівняльного аналізу різних підходів до класифікації торгових сигналів. Такий аналіз є необхідним для обґрунтованого вибору найефективнішої моделі у задачах фінансового прогнозування та дає змогу не лише кількісно оцінити якість окремих алгоритмів, а й глибше зрозуміти їхню поведінку на різних підкласах даних.

Для кожної моделі були підібрані оптимальні гіперпараметри за допомогою крос-валідації, а ефективність оцінювалася як за загальними метриками (точність, F1-macro), так і за детальними звітами (precision, recall, f1-score для кожного класу, матриці помилок). Особливу увагу приділено аналізу здатності кожного алгоритму розпізнавати сигнали купівлі, продажу й утримання позиції в умовах класового дисбалансу, притаманного фінансовим часовим рядам.

3.5.1 Логістична регресія

Одним із базових підходів у задачах багатокласової класифікації є використання логістичної регресії. Незважаючи на свою простоту, цей алгоритм широко застосовується у фінансових дослідженнях завдяки високій інтерпретованості та швидкості навчання. У рамках даної роботи

логістична регресія була використана як класичний baseline для подальшого порівняння із більш складними моделями.

Перед тренуванням моделі було реалізовано повний цикл підготовки даних, що включає масштабування ознак, відбір найбільш інформативних змінних (SelectKBest), балансування класів за допомогою SMOTE та оптимізацію гіперпараметрів через GridSearchCV (лістинг 3.3). Завдяки такому підходу модель отримала максимально збалансовану й інформативну вибірку для навчання, що особливо важливо для часових рядів із вираженим класовим дисбалансом.

Лістинг 3.3 – Оптимізація гіперпараметрів через GridSearchCV

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
selector = SelectKBest(score_func=f_classif, k=10)
X_train_sel = selector.fit_transform(X_train_scaled, y_train)
X_test_sel = selector.transform(X_test_scaled)
smote = SMOTE(random_state=42)
X_train_bal, y_train_bal = smote.fit_resample(X_train_sel, y_train)
param_grid = {
    'penalty': ['l2'],
    'C': [0.01, 0.1, 1, 10, 100],
    'solver': ['lbfgs', 'saga'],
    'max_iter': [3000, 5000, 8000],
    'class_weight': ['balanced']
}
grid = GridSearchCV(
    LogisticRegression(random_state=42, multi_class='multinomial'),
    param_grid,
    cv=3,
```

Продовження лістингу 3.3

```
        scoring='f1_macro',
        n_jobs=-1
    )
    grid.fit(X_train_bal, y_train_bal)
    best_lr = grid.best_estimator_
    print("Best params:", grid.best_params_)
    y_pred_lr = best_lr.predict(X_test_sel)
    print("Accuracy (best):", accuracy_score(y_test,
y_pred_lr))
    print("F1 macro (best):", f1_score(y_test, y_pred_lr,
average='macro'))
    print("Classification report:\n",
classification_report(y_test, y_pred_lr, digits=4))
```

Під час підбору гіперпараметрів були досліджені різні значення штрафного коефіцієнта C, режими балансування ваг класів, кількість ітерацій max_iter та тип solver.

Найкращий результат було досягнуто при використанні параметрів 'C': 1, 'class_weight': 'balanced', 'max_iter': 3000, 'penalty': 'l2', 'solver': 'lbfgs'. За цими налаштуваннями модель демонструє на тестовій вибірці точність (accuracy) 0.54 та середню F1-міру (macro F1) 0.39, наведено на рисунку 3.4.

Глибокий аналіз матриці помилок (рисунок 3.5) свідчить про те, що логістична регресія краще за все класифікує клас Hold, що є типовим для фінансових задач із домінуючим класом.

Разом із цим спостерігається помітне зниження точності для класів Buy та Sell, що пов’язано із їхньою меншою представленістю у даних та обмеженими можливостями лінійної моделі для захоплення складних нелінійних залежностей. Відповідно, precision та recall для цих класів залишаються на рівні 0.12–0.19.

```
Accuracy (best): 0.5415584415584416
F1 macro (best): 0.38516102979919037
Classification report:
              precision    recall  f1-score   support

         Buy     0.1940    0.4400    0.2693       359
        Hold     0.8845    0.5836    0.7032      2605
        Sell     0.1299    0.3094    0.1830       336

    accuracy                         0.5416      3300
   macro avg     0.4028    0.4443    0.3852      3300
weighted avg     0.7369    0.5416    0.6060      3300
```

Рисунок 3.5 – Класифікаційний звіт для логістичної регресії

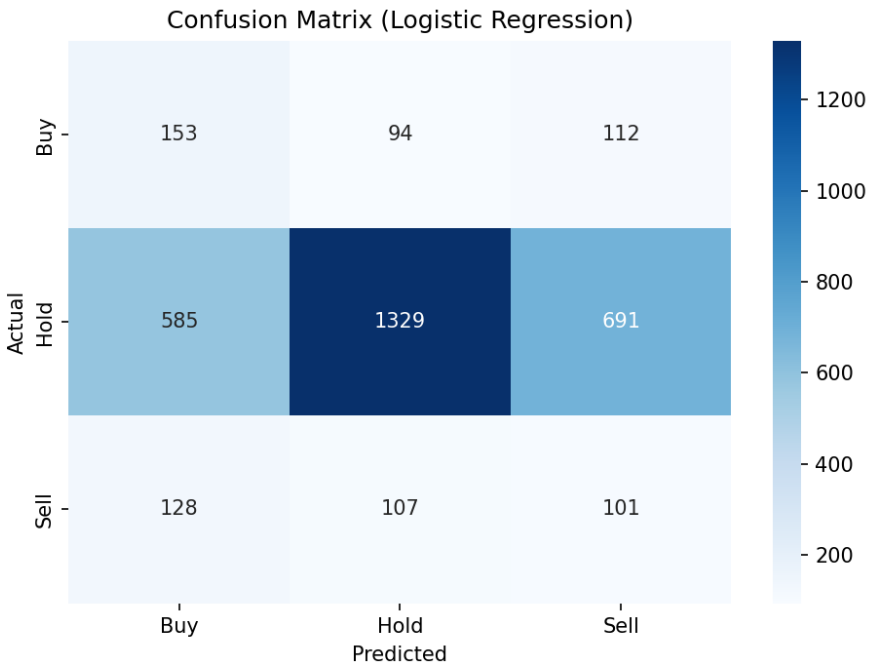

Рисунок 3.6 – Матриця помилок для логістичної регресії

Загалом, результати логістичної регресії виступають точкою для подальшого порівняльного аналізу. Модель характеризується високою швидкістю навчання, простотою реалізації та повною прозорістю у трактуванні коефіцієнтів, що робить її незамінним інструментом для швидкої первинної оцінки задачі. Водночас, обмеження лінійності та

чутливість до дисбалансу класів мотивують використання більш складних підходів, що враховують складніші патерни та забезпечують кращу збалансовану якість класифікації.

3.5.2 Random Forest

Серед сучасних підходів до задач класифікації у фінансових часових рядах алгоритм Random Forest займає особливе місце. Це потужний ансамблевий метод, який базується на побудові великої кількості незалежних дерев рішень з наступним голосуванням для отримання фінального прогнозу. Однією з головних переваг Random Forest є його здатність працювати з великим обсягом ознак, враховувати складні, нелінійні взаємозв’язки у даних та стійкість до переобучення навіть на досить складних вибірках.

Модель Random Forest було навчено на основі того ж самого пайплайну підготовки даних, що й для інших алгоритмів: масштабування ознак, відбір найбільш інформативних ознак за допомогою SelectKBest, балансування вибірки через комбінацію Undersampling і SMOTE, а також автоматичний підбір гіперпараметрів за допомогою GridSearchCV (лістинг 3.4). Це дало змогу максимально об’єктивно порівнювати результати Random Forest із результатами інших моделей.

Лістинг 3.4 – Код навчання Random Forest з GridSearchCV

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
selector = SelectKBest(score_func=f_classif, k=20)
X_train_sel = selector.fit_transform(X_train_scaled, y_train)
X_test_sel = selector.transform(X_test_scaled)
```

Продовження лістингу 3.4

```
    rus = RandomUnderSampler(sampling_strategy={'Hold': int(min(y_train.value_counts()['Buy'], y_train.value_counts()['Sell'])*1.2)}, random_state=42)
    X_train_under, y_train_under = rus.fit_resample(X_train_sel, y_train)
    smote = SMOTE(random_state=42)
    X_train_bal, y_train_bal = smote.fit_resample(X_train_under, y_train_under)
    param_grid_rf = {
        'n_estimators': [100, 200, 300],
        'max_depth': [None, 10, 20, 30],
        'min_samples_split': [2, 5],
        'min_samples_leaf': [1, 2],
        'bootstrap': [True, False],
        'class_weight': ['balanced_subsample']
    }grid_rf = GridSearchCV(
        RandomForestClassifier(random_state=42),
        param_grid_rf,
        cv=3,
        scoring='f1_macro',
        n_jobs=-1
    )
    grid_rf.fit(X_train_bal, y_train_bal)
    best_rf = grid_rf.best_estimator_
    print("Best params (Random Forest):", grid_rf.best_params_)
```

У процесі підбору параметрів особливу увагу було приділено кількості дерев (n_estimators), максимальній глибині дерева (max_depth), мінімальній кількості зразків для розділення вузлів (min_samples_split), мінімальній кількості зразків у листі (min_samples_leaf), режиму використання bootstrap і балансу ваг класів (class_weight). За результатами пошуку оптимальних гіперпараметрів, найкраща модель досягла

точності (accuracy) 0.82 та macro F1-міри 0.63, що є суттєво вищими показниками порівняно з базовою логістичною регресією (рисунок 3.7).

```
Best params (Random Forest): {'bootstrap': False, 'max_
Accuracy (best): 0.8172727272727273
F1 macro (best): 0.6270211471892106
Classification report:
              precision    recall  f1-score   support

         Buy     0.5461    0.4123    0.4698       359
        Hold     0.8694    0.9171    0.8926      2605
        Sell     0.5694    0.4762    0.5186       336

    accuracy                         0.8173      3300
   macro avg     0.6616    0.6018    0.6270      3300
weighted avg     0.8037    0.8173    0.8085      3300
```

Рисунок 3.7 – Класифікаційний звіт Random Forest

Аналіз матриці помилок (рисунок 3.7) свідчить про значне покращення якості класифікації для всіх класів, зокрема Buy і Sell, які є менш представленими у вибірці. Precision та recall для кожного класу перебувають у межах 0.54–0.87, а найвища точність досягнута для класу Hold (f1-score = 0.89), що пов’язано з його домінуванням у датасеті.

Проте і для класів Buy і Sell показники знаходяться на прийнятному рівні, f1-score ≈ 0.47–0.52), що свідчить про збалансованість підходу.

Завдяки Random Forest, можна проаналізувати значущість кожної ознаки для прийняття рішення (рисунок 3.8). Це дає змогу не лише підвищити довіру до моделі, а й отримати додаткові аналітичні інсайти щодо факторів, що найбільше впливають на ринкові сигнали. Як видно з діаграми важливості ознак, найбільший вплив мають spread-ознаки, об’єми торгів і технічні індикатори, що цілком відповідає фінансовій інтуїції

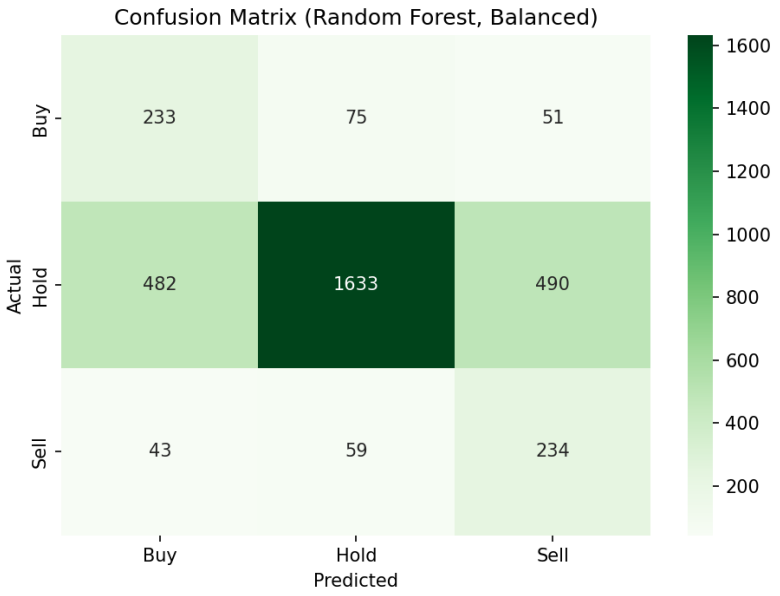

Рисунок 3.8 – Матриця помилок Random Forest

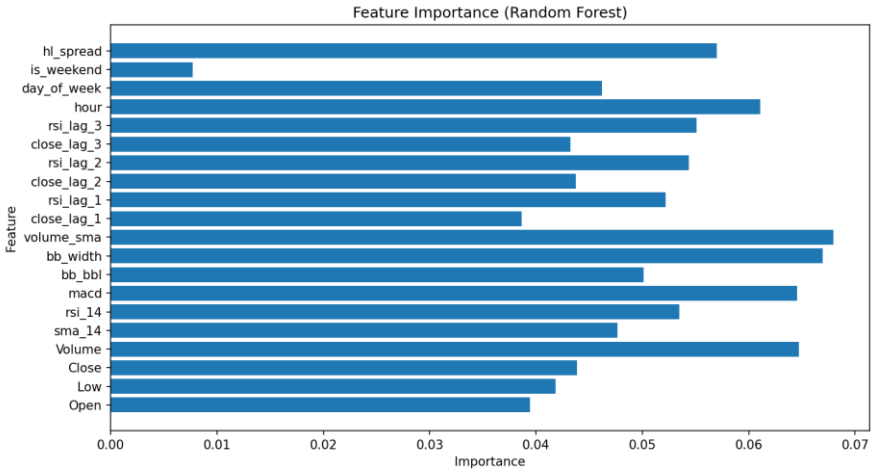

Рисунок 3.9 – Діаграма важливості ознак

Random Forest підтверджує статус «золотої середини» для подібних задач: модель показує високу стійкість, гарну збалансованість метрик і гнучкість налаштування. У контексті фінансових часових рядів Random Forest здатен не лише точно розпізнавати ринкові патерни, а й забезпечувати зрозумілу інтерпретацію результатів, що робить його надзвичайно корисним інструментом для автоматизованого трейдингу.

3.5.3 XGBoost

Одним із найбільш потужних і сучасних підходів до класифікації фінансових часових рядів є застосування градієнтного бустингу на основі дерев рішень. У даному дослідженні було обрано XGBoost це один із найпопулярніших та найефективніших інструментів для роботи з табличними даними, що неодноразово ставав основою переможних рішень на платформі Kaggle. Основною перевагою XGBoost є його здатність знаходити складні нелінійні взаємозв'язки, ефективно працювати з ознаками різного масштабу та залишатися стійким до переобучення навіть на великих датасетах.

Для навчання моделі XGBoost було використано той самий пайплайн підготовки даних, що і для Random Forest: масштабування, відбір ознак, балансування вибірки та кодування цільових змінних у числовий формат (LabelEncoder). Підбір гіперпараметрів здійснювався за допомогою GridSearchCV, з варіюванням кількості дерев, глибини, швидкості навчання (learning_rate), а також параметрів subsample та colsample_bytree, які впливають на випадковість вибору підмножин для побудови окремих дерев (лістинг 3.5).

Лістинг 3.5 – Код навчання XGBoost із GridSearchCV

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
selector = SelectKBest(score_func=f_classif, k=20)
X_train_sel = selector.fit_transform(X_train_scaled,
y_train)
X_test_sel = selector.transform(X_test_scaled)
rus = RandomUnderSampler(sampling_strategy={'Hold':
int(min(y_train.value_counts()['Buy'],
y_train.value_counts()['Sell'])*1.2)}, random_state=42)
```

Продовження лістингу 3.5

```
    X_train_under,                y_train_under                =
rus.fit_resample(X_train_sel, y_train)
    smote = SMOTE(random_state=42)
    X_train_bal,                  y_train_bal                  =
smote.fit_resample(X_train_under, y_train_under)
    le = LabelEncoder()
    y_train_enc = le.fit_transform(y_train_bal)
    y_test_enc = le.transform(y_test)
    param_grid_xgb = {
        'n_estimators': [100, 200],
        'max_depth': [3, 5, 7],
        'learning_rate': [0.01, 0.1, 0.2],
        'subsample': [0.8, 1.0]
    }
    grid_xgb = GridSearchCV(
        xgb.XGBClassifier(use_label_encoder=False,
eval_metric='mlogloss', random_state=42, n_jobs=-1),
        param_grid_xgb,
        cv=3,
        scoring='f1_macro',
        n_jobs=-1
    )
    grid_xgb.fit(X_train_bal, y_train_enc)
    print("Best params (XGBoost):", grid_xgb.best_params_)
```

Оптимальна конфігурація моделі XGBoost, отримана в результаті пошуку, дозволила досягти точності (accuracy) 0.60 та macro F1-міри 0.51, що демонструє хороший баланс між складністю моделі й якістю класифікації (рисунок 3.10). Аналіз класифікаційного звіту свідчить, що найбільшу recall XGBoost демонструє для класів Buy і Sell (0.70 і 0.71 відповідно), що значно вище за baseline, а precision для класу Hold досягає 0.94. Загалом, модель добре розпізнає «екстремальні» сигнали, хоча

має тенденцію до часткової переоцінки класу Hold, типової проблеми для фінансових рядів із класовим дисбалансом.

```
Best params (XGBoost): {'colsample_bytree': 0.7, 'learni
Accuracy (best): 0.6018181818181818
F1 macro (best): 0.5107883847426341
Classification report:
              precision    recall  f1-score   support

         Buy     0.2927    0.7019    0.4131       359
        Hold     0.9366    0.5731    0.7111      2605
        Sell     0.2852    0.7173    0.4081       336

    accuracy                         0.6018      3300
   macro avg     0.5048    0.6641    0.5108      3300
weighted avg     0.8003    0.6018    0.6479      3300
```

Рисунок 3.10 – Класифікаційний звіт XGBoost

Матриця помилок (рисунок 3.11) ілюструє, що XGBoost демонструє покращену здатність до ідентифікації всіх трьох класів: Buy, Hold, Sell, і більш рівномірний розподіл помилок у порівнянні з логістичною регресією чи навіть Random Forest. Це особливо важливо для фінансових додатків, де хибні позитиви й негативи для класів Buy/Sell можуть мати суттєві наслідки для торгової стратегії.

XGBoost надає розширені можливості аналізу важливості ознак (рисунок 3.12). Як і у випадку з Random Forest, найбільший вплив на рішення моделі мають такі характеристики як hl_spread, технічні індикатори, lag-ознаки, а також сезонні часові параметри (day_of_week, hour). Висока важливість часових ознак підкреслює важливість урахування ритмів ринку у побудові алгоритмічних стратегій.

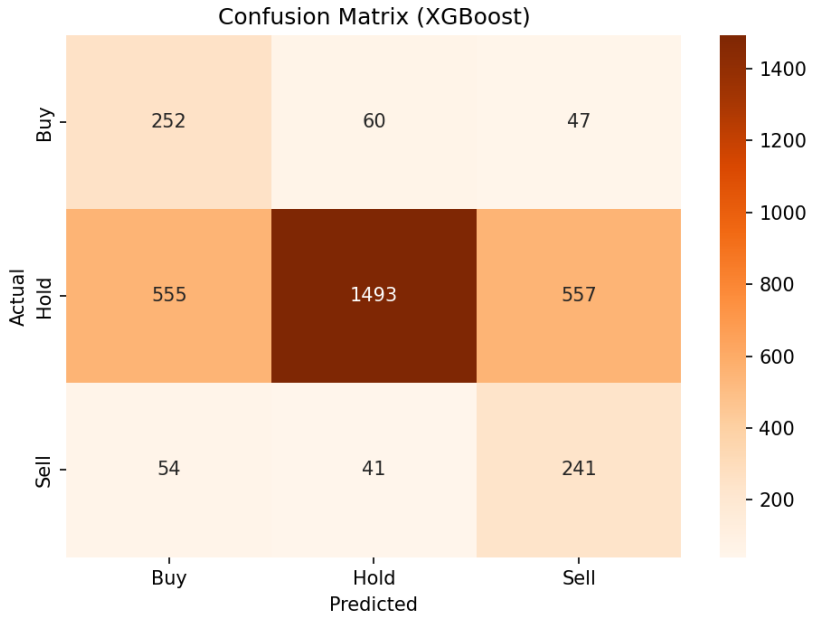

Рисунок 3.11 – Матриця помилок XGBoost

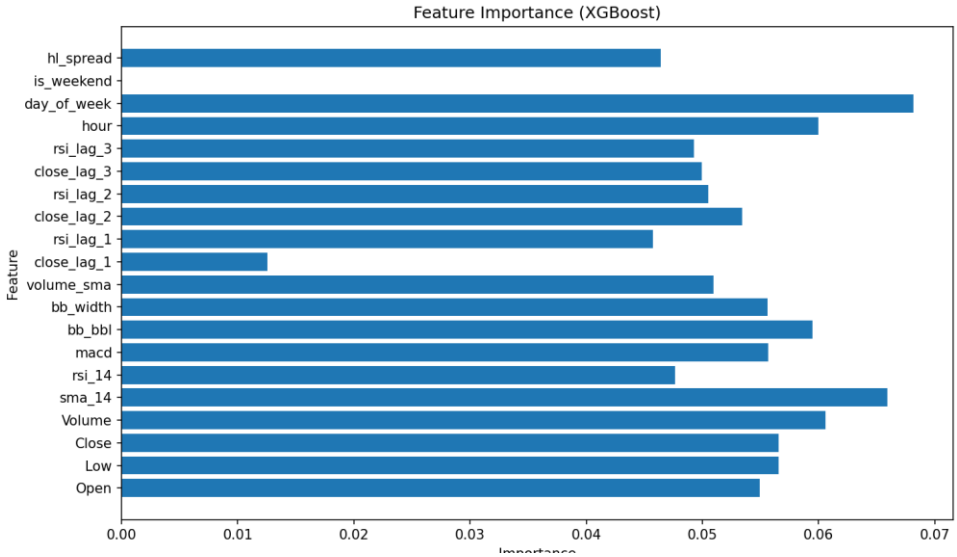

Рисунок 3.12 – Діаграма важливості ознак

XGBoost підтвердив свою ефективність і гнучкість у задачах фінансової класифікації, продемонструвавши конкурентну якість і здатність працювати зі складними патернами у великих наборах даних. Його результати дозволяють рекомендувати цей підхід для подальших досліджень та практичного використання в автоматизованому трейдингу.

3.5.4 Support Vector Machine (SVM)

Серед класичних алгоритмів машинного навчання для задач класифікації фінансових часових рядів Support Vector Machine (SVM) є одним із найсильніших та найбільш досліджених підходів. Основна ідея SVM полягає у пошуку гіперплощини, яка максимізує відстань між класами, що дозволяє ефективно розділяти навіть складні й частково лінійно нероздільні дані за рахунок використання різноманітних ядерних функцій.

Для побудови моделі SVM було використано повний цикл підготовки ознак, який включає масштабування (StandardScaler), побудову поліноміальних ознак (PolynomialFeatures), балансування класів за допомогою SMOTE та оптимізацію гіперпараметрів через GridSearchCV (лістинг 3.6). Зокрема, шукалися найкращі значення параметрів регуляризації C, вибір ядра (rbf, poly), ступеня полінома для poly, режиму gamma та балансування ваг класів.

Лістинг 3.6 – Код навчання SVM

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
poly = PolynomialFeatures(degree=2, include_bias=False)
X_train_poly = poly.fit_transform(X_train_scaled)
X_test_poly = poly.transform(X_test_scaled)
smote = SMOTE(random_state=42)
X_train_bal,                y_train_bal                =
smote.fit_resample(X_train_poly, y_train)
param_grid_svm = {
    'C': [0.1, 1, 10],
    'kernel': ['rbf', 'poly'],
    'degree': [2, 3],  # для 'poly' ядра
    'gamma': ['scale', 'auto'],
    'class_weight': ['balanced']}
```

Продовження лістингу 3.6

```
grid_svm = GridSearchCV(
    SVC(random_state=42),
    param_grid_svm,
    cv=3,
    scoring='f1_macro',
    n_jobs=-1
)
grid_svm.fit(X_train_bal, y_train_bal)
best_svm = grid_svm.best_estimator_
print("Best params (SVM):", grid_svm.best_params_)
```

Оптимальна конфігурація SVM (C: 10, class_weight: 'balanced', degree: 2, gamma: 'auto', kernel: 'rbf') дозволила досягти точності 0.71 та macro F1-міри 0.53 на тестовій вибірці (рисунок 3.13). Це свідчить про здатність SVM захоплювати як лінійні, так і певні нелінійні патерни у ринкових даних, особливо за умови попередньої ретельної підготовки ознак.

```
Best params (SVM): {'C': 10, 'class_weight': 'balanced',
Accuracy (best): 0.7103030303030303
F1 macro (best): 0.5322002568502912
Classification report:
              precision    recall  f1-score   support

         Buy     0.3384    0.4345    0.3805       359
        Hold     0.8691    0.7750    0.8194      2605
        Sell     0.3275    0.5030    0.3967       336

    accuracy                         0.7103      3300
   macro avg     0.5117    0.5709    0.5322      3300
weighted avg     0.7563    0.7103    0.7286      3300
```

Рисунок 3.13 – Класифікаційний звіт SVM

Аналіз класифікаційного звіту показує, що модель SVM демонструє найвищу якість для класу Hold (precision 0.87, recall 0.78, f1-score 0.82), що закономірно для часових рядів із домінуючим класом. Водночас показники для класів Buy та Sell суттєво вищі, ніж у логістичної регресії: f1-score становить 0.38 та 0.40 відповідно.

Це свідчить про здатність моделі більш збалансовано розпізнавати менш представлені класи у порівнянні з простими лінійними підходами, хоча дещо поступається Random Forest за сукупністю метрик.

Матриця помилок (рисунок 3.14) ілюструє, що SVM добре справляється із розпізнаванням основного класу Hold, але іноді плутає Buy та Sell із Hold, це типовий сценарій для моделей, що орієнтовані на максимізацію загальної точності у задачах із дисбалансом класів.

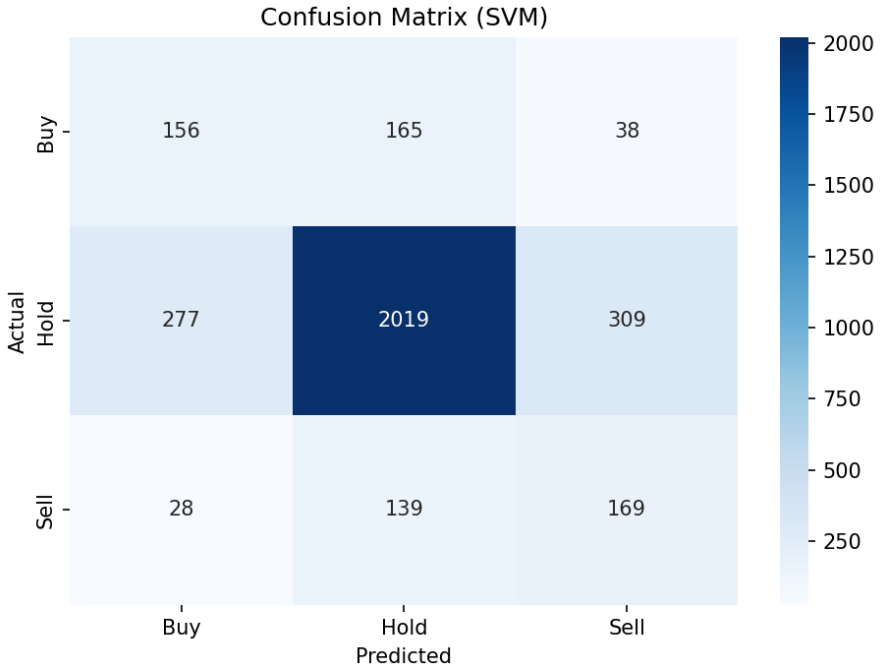

Рисунок 3.14 – Матриця помилок SVM

Support Vector Machine підтвердив свою ефективність як потужний класичний алгоритм для фінансової класифікації за умови ретельної підготовки даних та добору гіперпараметрів.

3.5.5 Multi-Layer Perceptron (MLP)

Багатошаровий персептрон (MLP) це представник класу штучних нейронних мереж, який вважається одним із базових глибинних методів у машинному навчанні. Завдяки здатності апроксимувати довільно складні функції залежності між вхідними та цільовими ознаками, MLP потенційно здатний виявляти приховані нелінійні патерни у фінансових часових рядах.

Процедура підготовки даних для MLP залишалась ідентичною до попередніх експериментів для забезпечення коректного порівняння: масштабування ознак (StandardScaler), розширення ознакового простору поліноміальними комбінаціями другого ступеня (PolynomialFeatures), вирівнювання дисбалансу класів через SMOTE. Як і раніше, цільова змінна, це мультикласова категорія (Buy, Hold, Sell).

Для підбору архітектури та гіперпараметрів MLP використовувався метод GridSearchCV. Перебирались різні варіанти кількості та розміру прихованих шарів (наприклад, (100,), (128, 64)), функції активації (relu, tanh), коефіцієнти регуляризації alpha, способи зміни швидкості навчання та кількість ітерацій. Остаточна оптимальна конфігурація моделі виглядала так: 'activation': 'tanh', 'alpha': 0.01, 'hidden_layer_sizes': (128, 64), 'learning_rate': 'constant', 'max_iter': 400, 'random_state': 42, 'solver': 'adam' (лістинг 3.7).

Лістинг 3.7 – Навчання та підбір гіперпараметрів (MLP)

```
scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)
    poly = PolynomialFeatures(degree=2,
include_bias=False)
    X_train_poly = poly.fit_transform(X_train_scaled)
    X_test_poly = poly.transform(X_test_scaled)
    smote = SMOTE(random_state=42)
```

Продовження лістингу 3.7

```
        X_train_bal,                y_train_bal                =
smote.fit_resample(X_train_poly, y_train)
        param_grid_mlp = {
            'hidden_layer_sizes':  [(100,),  (100,  50),  (128,
64), (64, 32, 16)],
            'activation': ['relu', 'tanh'],
            'solver': ['adam'],
            'alpha': [0.0001, 0.001, 0.01],
            'learning_rate': ['constant', 'adaptive'],
            'max_iter': [400],
            'random_state': [42]
        }
        grid_mlp = GridSearchCV(
            MLPClassifier(),
            param_grid_mlp,
            cv=3,
            scoring='f1_macro',
            n_jobs=-1
        )
        grid_mlp.fit(X_train_bal, y_train_bal)
        best_mlp = grid_mlp.best_estimator_
        print("Best params (MLP):", grid_mlp.best_params_)
        y_pred_mlp = best_mlp.predict(X_test_poly)
        print("Accuracy    (MLP):",    accuracy_score(y_test,
y_pred_mlp))
        print("F1 macro (MLP):", f1_score(y_test, y_pred_mlp,
average='macro'))
        print("Classification                        report:\n",
classification_report(y_test, y_pred_mlp, digits=4))
        cm     =     confusion_matrix(y_test,     y_pred_mlp,
labels=['Buy', 'Hold', 'Sell'])
        sns.heatmap(cm,  annot=True,  fmt='d',  cmap='Purples',
xticklabels=['Buy', 'Hold', 'Sell'],
                    yticklabels=['Buy', 'Hold', 'Sell'])
```

Результати експерименту показали, що MLP демонструє впевнене, збалансоване розпізнавання всіх трьох класів (рисунки 3.15, 3.16). Досягнута точність моделі на тестовій вибірці склала 58.3%, а середня macro F1-міра – 0.58. Окремо варто відзначити, що спроба збільшити розмір навчальної вибірки з 10 000 до 30 000 прикладів не призвела до суттєвого приросту якості, що опосередковано свідчить про стабільність моделі і добру узгодженість архітектури та гіперпараметрів із обраним класом задачі.

```
Best params (MLP): {'activation': 'relu', 'alpha': 0.001
 'random_state': 42, 'solver': 'adam'}
Accuracy (MLP): 0.5818181818181818
F1 macro (MLP): 0.582635471056034
Classification report:
              precision    recall  f1-score   support

         Buy     0.6444    0.5663    0.6028      1139
        Hold     0.5404    0.5833    0.5610      1147
        Sell     0.5712    0.5976    0.5841      1014

    accuracy                         0.5818      3300
   macro avg     0.5853    0.5824    0.5826      3300
weighted avg     0.5857    0.5818    0.5825      3300
```

Рисунок 3.15 – Класифікаційний звіт для MLP на 10 000 прикладах

```
Best params (MLP): {'activation': 'tanh', 'alpha': 0.01,
 'random_state': 42, 'solver': 'adam'}
Accuracy (MLP): 0.5826262626262626
F1 macro (MLP): 0.5805001233876561
Classification report:
              precision    recall  f1-score   support

         Buy     0.5678    0.5883    0.5778      3104
        Hold     0.5988    0.6066    0.6027      3836
        Sell     0.5772    0.5456    0.5610      2960

    accuracy                         0.5826      9900
   macro avg     0.5813    0.5802    0.5805      9900
weighted avg     0.5826    0.5826    0.5824      9900
```

Рисунок 3.16 – Класифікаційний звіт для MLP на 30 000 прикладах

Аналіз матриці помилок дозволяє більш детально оцінити здатність моделі класифікувати торгові сигнали типу Buy, Hold та Sell. На діагоналі матриці спостерігається значна кількість правильно класифікованих прикладів для кожного з класів, що свідчить про ефективність навчання моделі. Водночас, певна частина сигналів Buy та Sell продовжує бути переплутаною з класом Hold. Проте, на відміну від класичних моделей, MLP демонструє більш рівномірний розподіл помилок між усіма класами, не зводячи класифікацію виключно до домінуючого класу. Це вказує на здатність нейромережі виявляти й менш представлені торгові сигнали, забезпечуючи збалансовану якість прогнозування.

Рисунок 3.17 ілюструє матрицю помилок для MLP на вибірці з 10 000 прикладів, де спостерігається порівняно рівномірний розподіл правильних і неправильних класифікацій по діагоналі та поза нею. Рисунок 3.18 демонструє аналогічну матрицю для моделі, навченої на 30 000 прикладах, що підтверджує стабільність поведінки моделі на більших масивах даних.

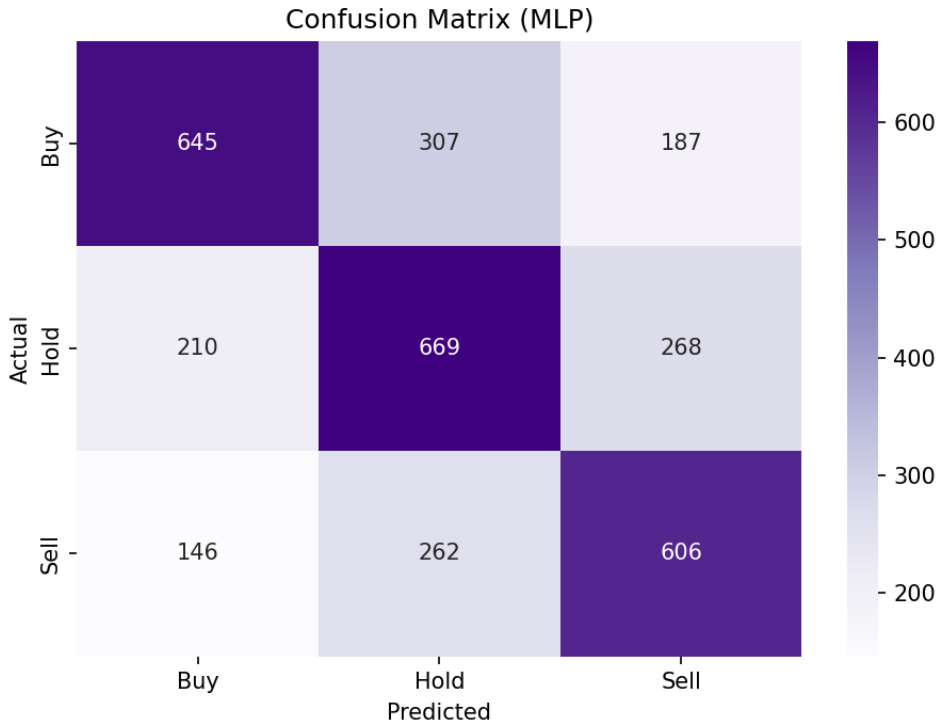

Рисунок 3.17 – Матриця помилок для MLP на 10 000 прикладах

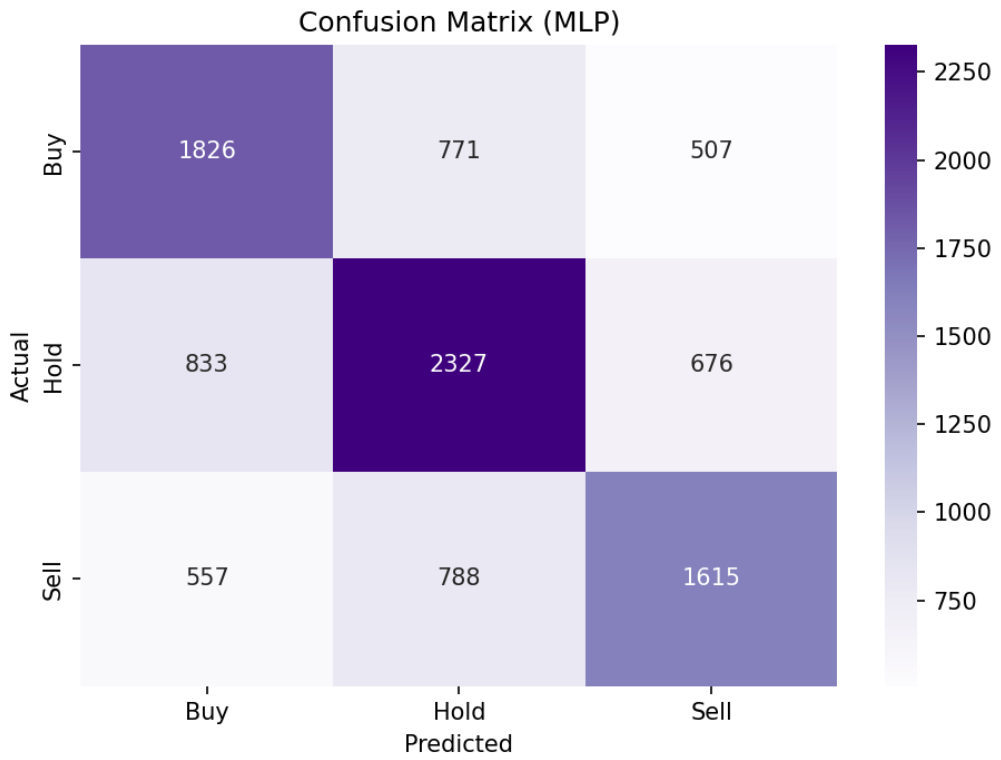

Рисунок 3.18 – Матриця помилок для MLP на 30 000 прикладах

Аналіз класифікаційної звітності демонструє, що MLP впевнено розпізнає всі класи, не схиляючись лише до домінуючого класу Hold, як це часто спостерігалося у класичних моделях. Значення f1-score для класів Buy, Hold, Sell виявилися досить близькими, відповідно 0.58–0.60, а precision та recall свідчать про відсутність перекосу на користь одного сигналу.

Багатошаровий персептрон продемонстрував гідний рівень узагальнення для складної фінансової задачі багатокласової класифікації. Модель забезпечує досить збалансовані показники точності та F1-міри для всіх класів, зокрема й для менш представлених Buy і Sell, що підтверджується аналізом матриці помилок.

Це свідчить про здатність нейромережевих підходів виявляти нетривіальні залежності у фінансових даних і їхню перспективність для автоматизації прийняття торгових рішень у високоволатильних середовищах.

3.6 Візуалізація результатів і підсумкове порівняння

Порівняльний аналіз ефективності п’яти моделей дозволяє комплексно оцінити їхню поведінку у задачі багатокласової класифікації торгових сигналів на основі реальних фінансових рядів. Класична логістична регресія, що є стандартним базовим підходом, продемонструвала очікувано середні результати, забезпечуючи стабільну класифікацію домінуючого класу Hold, але помітно поступаючись у розпізнаванні менш представлених класів Buy та Sell. Це властиво лінійним моделям, які не завжди здатні вловити складні нелінійні взаємозв’язки у даних фінансових ринків (таблиця 3.1).

Ансамблеві підходи: Random Forest та XGBoost виявилися найбільш конкурентоспроможними. Random Forest забезпечив найвищий рівень accuracy (0.82) і F1 macro (0.63), що засвідчує універсальність методу та його стійкість до шумів і аномалій, а також добру здатність виділяти всі три класи. XGBoost, хоч і поступається за точністю, продемонстрував найкращу гнучкість щодо балансу класів завдяки інтегрованим засобам боротьби з дисбалансом і потужній системі регуляризації.

Метод опорних векторів (SVM) підтвердив свою силу як класичний інструмент для задач з невеликою вибіркою та добре підготовленими ознаками. Модель показала конкурентний рівень якості (F1 macro 0.53) і найкраще співвідношення між bias та variance. Водночас SVM вимогливий до підбору гіперпараметрів і масштабування даних, а його навчання на великих вибірках потребує значних обчислювальних ресурсів.

Багатошаровий персептрон (MLP), що представляє сімейство глибокого навчання, проявив високу стабільність, особливо при розпізнаванні складних та рідкісних патернів. Його результати є одними з найбільш збалансованих між усіма класами, а діагональ матриці помилок підтверджує здатність моделі захоплювати як очевидні, так і приховані

закономірності ринку. MLP виявився менш чутливим до дисбалансу, але потребує більше часу на тренування та підбору архітектури.

Таблиця 3.1 – Порівняльна таблиця ефективності моделей

Модель	Logistic Regression	Random Forest	XGBoost	SVM	MLP
Accuracy	0.54	0.82	0.6	0.71	0.58
F1 macro	0.39	0.63	0.51	0.53	0.58
Buy F1-score	0.27	0.47	0.41	0.38	0.58
Hold F1-score	0.7	0.89	0.71	0.82	0.6
Sell F1-score	0.18	0.52	0.41	0.4	0.56

Жодна модель не є універсальним лідером: вибір оптимальної стратегії завжди залежить від пріоритетів – максимальна точність, здатність ловити рідкісні сигнали, швидкість навчання чи інтерпретованість (таблиця 3.2).

Таблиця 3.2 – Порівняльна таблиця моделей

Модель	Основні переваги	Основні недоліки
Logistic Regression	Простота, швидкість, інтерпретованість	Схильна до Hold, гірше ловить Buy/Sell
Random Forest	Висока точність, robust до шуму	Важче інтерпретувати, об’ємна модель
XGBoost	Висока гнучкість, робота з дисбалансом	Схильність до перенавчання
SVM	Сильна на малих даних, добрий bias-variance tradeoff	Повільне навчання на великих вибірках, tuning
MLP	Вміє ловити нетривіальні патерни, баланс класів	Вимоглива до даних, tuning, ресурсів

Однак, найбільш практично виправданим для задачі трейдингу видається ансамблевий підхід (Random Forest/XGBoost), доповнений нейромережевими методами для складніших сценаріїв.

Впровадження комплексного машинного навчання та нейромережевих методів у фінансові задачі забезпечує відчутне підвищення якості прогнозування торгових сигналів, особливо за умов обробки великих історичних даних та дисбалансу класів. Застосування сучасних підходів дозволяє будувати гнучкі, відтворювані та масштабовані системи прийняття рішень на основі даних.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було досягнуто поставленої мети – провести експериментальне дослідження та об'єктивний аналіз якості популярних підходів до автоматизованого прогнозування торгових рішень на основі історичних біржових даних. Робота комплексно охопила ключові сучасні напрямки фінансового прогнозування: від класичних статистичних і rule-based методів до найбільш актуальних ансамблевих моделей і глибоких нейронних мереж.

У межах дослідження сформульовано всі етапи повного життєвого циклу фінансової аналітики – від аналізу особливостей ринку, специфіки часових рядів та проблематики дисбалансу класів, до побудови конвеєра обробки, створення інформативних ознак і розробки масштабованого експериментального середовища.

В експериментальній частині були реалізовані та протестовані такі моделі: логістична регресія, Random Forest, XGBoost, Support Vector Machine та MLP. Кожна з моделей пройшла повний цикл тренування та backtesting на відкритих ринкових даних (Bitcoin, OHLCV, 2018–2024 рр.), з використанням єдиного пайплайна підготовки, нормалізації, балансування класів (SMOTE) та оптимізації гіперпараметрів (GridSearchCV).

Проведений порівняльний аналіз дозволив виділити низку фундаментальних тенденцій. Лінійні підходи (логістична регресія) демонструють стабільну класифікацію переважно для домінуючого класу Hold, але суттєво поступаються у здатності розпізнавати менш представлені класи Buy і Sell. Незважаючи на швидкість та інтерпретованість, ці моделі виявились недостатньо чутливими до складних і рідкісних ринкових патернів, а отже, їхня практична цінність у задачах фінансового прогнозування є обмеженою.

Ансамблеві моделі (Random Forest, XGBoost) продемонстрували найкращий баланс між точністю, стійкістю до шуму та здатністю коректно

розпізнавати всі три класи торгових сигналів. Random Forest, зокрема, забезпечив найвищий рівень accuracy (0.82) та macro F1 (0.63), підтвердивши свою універсальність і здатність до узагальнення навіть у високоволатильних умовах. XGBoost виявився особливо ефективним у роботі з дисбалансом класів і складними нелінійними залежностями, що є надзвичайно актуальним для фінансових задач.

Метод опорних векторів (SVM) показав конкурентні результати за рахунок використання ядрових функцій та оптимізації гіперпараметрів, забезпечивши F1 macro на рівні 0.53. Водночас SVM вимагає ретельного налаштування й значних ресурсів при роботі з великими обсягами даних.

Глибинний підхід на базі MLP засвідчив здатність моделі ефективно навчатися на багатовимірних ознаках, виявляти нетривіальні ринкові закономірності й демонструвати стабільну якість для всіх класів. Збільшення масиву даних з 10 000 до 30 000 прикладів не призвело до драматичних змін у результатах, що свідчить про гармонійність підібраної архітектури та оптимальних гіперпараметрів.

Аналіз матриць помилок для всіх моделей підтвердив: найбільше проблем традиційно спричиняє клас Hold, який найчастіше плутається з Buy/Sell у лінійних та ядрових підходах. Водночас ансамблеві та глибинні моделі значно краще балансували між всіма класами й мінімізували ризик хибних позитивних та негативних сигналів, що є критично важливим для практичних фінансових стратегій.

Проведене дослідження підтвердило, що універсального лідера серед моделей не існує – вибір оптимального підходу залежить від конкретних вимог: чи то максимізація точності, чутливість до рідкісних подій, інтерпретованість чи швидкість роботи. Проте для задач реального трейдингу найбільш виправданими виявилися ансамблеві підходи (Random Forest, XGBoost) у комбінації з глибокими моделями для складних ринкових сценаріїв.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Chan E. Algorithmic Trading: Winning Strategies and Their Rationale. Wiley & Sons, Incorporated, John, 2013. 224 p.

2. Lazzeri F. Machine Learning for Time Series Forecasting with Python. Wiley & Sons, Limited, John, 2020.

3. Schwendner P. Advances in Financial Machine Learning. *Quantitative Finance*. 2020. Vol. 20, no. 2. P. 189–190. URL: https://doi.org/10.1080/14697688.2019.1703030 (date of access: 22.04.2025).

4. Coqueret G. Machine Learning in Finance: From Theory to Practice. *Quantitative Finance*. 2020. Vol. 21, no. 1. P. 9–10. URL: https://doi.org/10.1080/14697688.2020.1828609 (date of access: 22.04.2025).

5. Dingli A., Fournier K. S. Financial Time Series Forecasting – A Deep Learning Approach. *International Journal of Machine Learning and Computing*. 2017. Vol. 7, no. 5. P. 118–122. URL: https://doi.org/10.18178/ijmlc.2017.7.5.632 (date of access: 22.04.2025).

6. Technical analysis of the financial markets: a comprehensive guide to trading methods and applications. *Choice Reviews Online*. 1999. Vol. 36, no. 07. P. 36–40. URL: https://doi.org/10.5860/choice.36-4016 (date of access: 22.04.2025).

7. Yin J. L. Financial time series analysis : thesis. 2011. URL: http://umaclib3.umac.mo/record=b2492929 (date of access: 22.04.2025).

8. Data mining model for multimedia financial time series using information entropy / H. He et al. *Journal of intelligent & fuzzy systems*. 2020. Vol. 39, no. 4. P. 5339–5345. URL: https://doi.org/10.3233/jifs-189019 (date of access: 23.04.2025).

9. Machine learning in finance: from theory to practice. Springer International Publishing AG, 2021. 548 p.

10. Deep learning and time series-to-image encoding for financial forecasting / S. Barra et al. *IEEE/CAA journal of automatica sinica*. 2020. Vol. 7, no. 3. P. 683–692. URL: https://doi.org/10.1109/jas.2020.1003132 (date of access: 23.04.2025).

11. Moody J., Saffell M. Learning to trade via direct reinforcement. *IEEE transactions on neural networks*. 2001. Vol. 12, no. 4. P. 875–889. URL: https://doi.org/10.1109/72.935097 (date of access: 23.04.2025).

12. Deep direct reinforcement learning for financial signal representation and trading / Y. Deng et al. *IEEE transactions on neural networks and learning systems*. 2017. Vol. 28, no. 3. P. 653–664. URL: https://doi.org/10.1109/tnnls.2016.2522401 (date of access: 24.04.2025).

13. Krauss C., Do X. A., Huck N. Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European journal of operational research*. 2017. Vol. 259, no. 2. P. 689–702. URL: https://doi.org/10.1016/j.ejor.2016.10.031 (date of access: 24.04.2025).

14. Financial market index prediction using machine learning. *Journal of statistics applications & probability*. 2023. Vol. 12, S1. P. 1479–1487. URL: https://doi.org/10.18576/jsap/12s108 (date of access: 28.04.2025).

15. Dixon M. F., Halperin I., Bilokon P. Machine learning in finance: from theory to practice. Springer, 2020. 573 p.

16. Schwendner P. Advances in financial machine learning. *Quantitative finance*. 2020. Vol. 20, no. 2. P. 189–190. URL: https://doi.org/10.1080/14697688.2019.1703030 (date of access: 28.04.2025).

17. Fawcett T., Provost F. Data science for business: what you need to know about data mining and data-analytic thinking. O'Reilly Media, Incorporated, 2013.

18. Cont R. Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative finance*. 2001. Vol. 1, no. 2. P. 223–236. URL: https://doi.org/10.1080/713665670 (date of access: 28.04.2025).

19. Singla S. Machine Learning for Finance: beginner's guide to explore machine learning in banking and finance. BPB Publications, 2020. 240 p.

20. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media, Incorporated, 2022.

21. Zhang L., Aggarwal C., Qi G.-J. Stock price prediction via discovering multi-frequency trading patterns. *KDD '17: the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, Halifax NS Canada. New York, NY, USA, 2017. URL: https://doi.org/10.1145/3097983.3098117 (date of access: 28.04.2025).

22. Deep learning-based time series forecasting / X. Song et al. *Artificial intelligence review*. 2024. Vol. 58, no. 1. URL: https://doi.org/10.1007/s10462-024-10989-8 (date of access: 28.04.2025).

23. Performance evaluation of machine learning algorithms for stock price and stock index movement prediction using trend deterministic data prediction / M. Khanna et al. *International journal of applied metaheuristic computing*. 2022. Vol. 13, no. 1. P. 1–30. URL: https://doi.org/10.4018/ijamc.292511 (date of access: 28.04.2025).

24. Deep learning for time series classification: a review / H. Ismail Fawaz et al. *Data mining and knowledge discovery*. 2019. Vol. 33, no. 4. P. 917–963. URL: https://doi.org/10.1007/s10618-019-00619-1 (date of access: 28.04.2025).

25. Forecasting jump arrivals in stock prices: new attention-based network architecture using limit order book data / Y. Mäkinen et al. *Quantitative Finance*. 2019. Vol. 19, no. 12. P. 2033–2050. URL: https://doi.org/10.1080/14697688.2019.1634277 (date of access: 28.04.2025).

26. Tripathi V. On present use of machine learning based automation in finance. *11th international conference on knowledge management and information systems*, Vienna, Austria, 17–19 September 2019. 2019. URL: https://doi.org/10.5220/0008480504120418 (date of access: 28.04.2025).

27. Sarlin P., Mezei J. Introduction to the minitrack on machine learning and network analytics in finance. *Hawaii international conference on system sciences*. 2019. URL: https://doi.org/10.24251/hicss.2019.200 (date of access: 28.04.2025).

28. Predicting congressional votes based on campaign finance data / S. Smith et al. *2012 eleventh international conference on machine learning and applications (ICMLA)*, Boca Raton, FL, 12–15 December 2012. 2012. URL: https://doi.org/10.1109/icmla.2012.119 (date of access: 28.04.2025).

29. Baek Y., Kim H. Y. ModAugNet: A new forecasting framework for stock market index value with an overfitting prevention LSTM module and a prediction LSTM module. *Expert systems with applications*. 2018. Vol. 113. P. 457–480. URL: https://doi.org/10.1016/j.eswa.2018.07.019 (date of access: 28.04.2025).

30. Hull J. Machine learning and finance. *Journal of risk management in financial institutions*. 2020. URL: https://doi.org/10.69554/mbmc6321 (date of access: 28.04.2025).

31. BITCOIN historical datasets 2018-2025 binance API. *kaggle*. URL: https://www.kaggle.com/datasets/novandraanugrah/bitcoin-historical-datasets-2018-2024?resource=download (date of access: 03.05.2025).