МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ

Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА на тему : «ДОСЛІДЖЕННЯ ТЕХНОЛОГІЙ РЕАЛІЗАЦІЇ ТОРГОВИХ БОТІВ»

Виконав: студент 2 курсу, групи 8.1224
спеціальності 122 комп'ютерні науки
(шифр і назва спеціальності)
освітньої програми комп'ютерні науки
(назва освітньої програми)
Русін В. О.
(ініціали та прізвище)
Керівник доцент кафедри комп'ютерних наук, к.т.н. Бюро С.Ю.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент завідувач кафедри програмної інженерії, к.ф.-м.н. Лісняк А.О.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра комп'ютерних наук

Рівень вищої освіти магістр

Спеціальність 122 комп'ютерні науки
(шифр і назва)

Освітня програма комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук,
д.т.н., професор

_____ Шило Г.М.
(підпис)

" ____ _____ 2025 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Русіну В'ячеславу Олександровичу
(прізвище, ім'я та по-батькові)

1. Тема роботи Дослідження технологій реалізації торгових ботів

керівник роботи Бюро Сергій Юрійович, к.т.н., доцент
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 28 » лютого 2025 року № 326-с

2. Строк подання студентом роботи 02.12.2025

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

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Експериментальне дослідження та аналіз результатів

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

6. Консультанти розділів роботи

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

7. Дата видачі завдання 03.03.2025

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	04.04.2025	
2.	Збір вихідних даних.	10.05.2025	
3.	Обробка методичних та теоретичних	05.06.2025	
	джерел.		
4.	Розробка першого та другого розділу.	20.10.2025	
5.	Розробка третього розділу.	10.11.2025	
6.	Оформлення та нормоконтроль	25.11.2025	
	кваліфікаційної роботи магістра.		
7.	Захист кваліфікаційної роботи.	17.12.2025	

Студент _____ В.О.Русін
(підпис) (ініціали та прізвище)

Керівник роботи _____ С.Ю.Бюро
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.Г.Спиця
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Дослідження технологій реалізації торгових ботів»: 36 с., 6 рис., 15 джерел, 3 додатки.

АЛГОРИТМІЧНА ТОРГІВЛЯ, АНСАМБЛЕВІ МОДЕЛІ, МАШИННЕ НАВЧАННЯ, ПРОГНОЗУВАННЯ ЧАСОВИХ РЯДІВ, РЕКУРЕНТНІ НЕЙРОННІ МЕРЕЖІ, СТЕКУВАННЯ МОДЕЛЕЙ, ТЕХНІЧНІ ІНДИКАТОРИ, ФІНАНСОВІ РИНКИ.

Об’єкт дослідження – процес прогнозування поведінки фінансового ринку на основі історичних ринкових даних та зовнішньої інформації.

Предмет дослідження – методи та алгоритми машинного навчання і нейронних мереж, що застосовуються для прогнозування майбутніх цінових рухів фінансових інструментів та формування торгових рішень.

Мета роботи – розробка та експериментальне дослідження системи прогнозування фінансового ринку, заснованої на використанні стекованої моделі машинного навчання, з метою підвищення ефективності алгоритмічної торгової стратегії.

Методи дослідження – методи аналізу часових рядів, машинного навчання, рекурентні нейронні мережі типу LSTM, ансамблеві та стековані моделі, статистичні методи оцінювання якості прогнозів, методи зворотного тестування торгових стратегій.

У кваліфікаційній роботі розглянуто підхід до прогнозування фінансового ринку, що поєднує технічні індикатори та інформаційні характеристики новинного фону. Запропоновано стековану модель прогнозування, яка використовує кілька базових алгоритмів машинного навчання та рекурентну нейронну мережу для оцінки майбутнього локального максимуму ціни. На основі отриманих прогнозів сформовано торгову стратегію та проведено

експериментальне дослідження її ефективності з використанням стандартних фінансових показників. Отримані результати підтверджують практичну доцільність застосування запропонованого підходу для задач алгоритмічної торгівлі.

SUMMARY

Master's Thesis "Research of Technologies for the Implementation of Trading Bots": 36 pages, 6 figures, 15 references, 3 annexes.

ALGORITHMIC TRADING, ENSEMBLE MODELS, MACHINE LEARNING, TIME SERIES FORECASTING, RECURRENT NEURAL NETWORKS, MODEL STACKING, TECHNICAL INDICATORS, FINANCIAL MARKETS.

Object of the study – process of prediction of financial market behavior based on historical market data and external information.

Subject of the study - machine learning methods and neural network algorithms used to predict future price movements of financial instruments and to generate trading decisions.

Aim of the study - creation and experimental research of a stacked machine learning model based system for financial market forecasting as an attempt to make the algorithmic trading strategy more efficient.

Methods of research - time series analysis methods, machine learning techniques, recurrent neural networks of the LSTM type, ensemble and stacked models, statistical methods for evaluating forecasting accuracy, and backtesting methods for trading strategies.

The master's thesis proposes a financial market forecast that would combine technical indicators with informational characteristics reflected from the news background. A stacked forecast model is suggested in which several base machine learning algorithms are aggregated together with a recurrent neural network estimating future local maximum values of prices. Based on these forecasts, a trading strategy was built and experimentally assessed for its efficiency using standard

financial performance metrics. Results achieved ensure practical implementability of the proposed approach to an algorithmic trading task.

ЗМІСТ

Завдання на кваліфікаційну роботу……..…..2
Реферат…… 4
Summary……6
Вступ…….9
1. ТЕХНОЛОГІЇ РЕАЛІЗАЦІЇ ТОРГОВИХ БОТІВ ……...……12
1.1. Особливості алгоритмічної торгівлі…….….. 12
1.2. Особливості використання торгових ботів…….14
2. РОЗРОБКА МАТЕМАТИЧНИХ ТА ІНФОРМАЦІЙНИХ МОДЕЛЕЙ…….16
2.1. Формулювання задачі прогнозування…….…16
2.2. Формування простору ознак……...…17
2.3. Архітектура стекової моделі прогнозування…….....18
2.4. Алгоритм системи прогнозування…….…..19
2.5. Інформаційна модель системи…….20
3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТОРГОВОГО БОТА….20
3.1. Вибір середовища та інструментів реалізації……20
3.2. Архітектура програмної системи……...….20
4. ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ….24
4.1. Дані та умови проведення експерименту……...24
4.2. Налаштування моделі та дизайн експерименту……...….25
4.3. Результати тестування торгової стратегії на попередніх дани…….26
4.4. Аналіз фінансових показників…….....28
ВИСНОВКИ……29
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ……..30
ДОДАТОК А……....30
ДОДАТОК Б……....32
ДОДАТОК В……....34

ВСТУП

Алгоритмічна торгівля займає чільне місце серед сучасних фінансових технологій, оскільки вона дозволяє автоматизувати прийняття рішень у торгівлі та усунути людський фактор. Передумовами, що закладають основу для переходу від традиційних стратегій до інтелектуальних систем, є зростання обсягів ринкових даних, вища швидкість обробки та вдосконалення методів машинного навчання. Прикладом інтелектуальної системи є бот; програмні засоби можуть аналізувати ринкову ситуацію, генерувати сигнали, прогнозувати рух цін, автоматично виконувати операції купівлі-продажу.

Навіть з величезним розвитком інструментів автоматизації, питання технологій для реалізації торгових ботів залишається невирішеним. Різні підходи – від класичних індикаторних стратегій до моделей машинного навчання та гібридних систем, що містять більше одного методу одночасно – у більшості опублікованих робіт [1, 5] або зосереджуються лише на торгових стратегіях, або на використанні окремих алгоритмів прогнозування, рідко – на цілісних техно-дослідженнях, що включають математичне моделювання, архітектуру та реалізацію ботів.

Об'єктом дослідження є процес побудови та функціонування торгових ботів в алгоритмічних торгових системах.

Предметом дослідження є технологія реалізації торгових ботів, включаючи математичні моделі, інформаційні структури, алгоритми та архітектуру програмного забезпечення.

Метою цієї роботи є дослідження сучасних технологій реалізації торгових ботів та розробка власного бота, який би поєднував технічний аналіз з методами машинного навчання, що використовуються для прогнозування ринкових процесів.

Для досягнення мети необхідно виконати такі завдання:

– проаналізувати особливості алгоритмічної торгівлі та використання торгових ботів;

– вивчити сучасні технології створення та впровадження торгових ботів;

– встановити вимоги та розробити математичну модель бота;

– розробити інформаційну модель, а також узагальнений алгоритм для системи;

– виконати розробку архітектури програмного забезпечення;

– реалізувати програмний модуль торгового бота на Python.

– провести експериментальне дослідження та тестування розробленої системи;

– проаналізувати метрики Sharpe, Sortino, Calmar, Max Drawdown, ROI на предмет значень, які прийматимуть ці торговельні стратегії.

Наукова новизна полягає в комбінованій системі автоматичного прийняття рішень на фінансових ринках, що включає аналіз індикаторів та методи машинного навчання. У роботі досліджується та реалізується математичне моделювання торгового бота аж до його програмної реалізації з практичним тестуванням на практиці. У цій новині також розглядається повний цикл розробки – від створення на рівні математичних моделей до їх втілення у вигляді програмних продуктів, які зрештою можуть бути навіть практично протестовані.

Практичне значення має створення програмного забезпечення, яке можна використовувати як для освітніх та наукових цілей, так і для вирішення практичних задач у сфері алгоритмічної торгівлі. Цей бот стане основою, на якій можна буде проводити подальше вдосконалення, інтегрувати його з торговими терміналами або використовувати для розробки складніших інтелектуальних систем.

На захист будуть представлені:

– результати аналізу технологій впровадження торгових ботів;

– математичні та інформаційні моделі торгової системи;

– архітектура програмного забезпечення.

– розроблений програмний модуль торгового бота.

– результати експериментальних досліджень та тестування на попередніх даних.

РОЗДІЛ 1. ТЕХНОЛОГІЇ РЕАЛІЗАЦІЇ ТОРГОВИХ БОТІВ

1.1. Особливості алгоритмічної торгівлі

Торгівля доступна на різних торгових майданчиках після реєстрації у брокера, який має ліцензію на доступ до цих ринків. Трейдер встановлює спеціальне клієнтське програмне забезпечення (термінал), яке дозволяє здійснювати угоди з візуалізацією та оновленням котирувань акцій у режимі реального часу за допомогою каналів передачі даних, що працюють через Інтернет між серверами, встановленими на відповідній біржі, інфраструктурою брокера та віддаленим робочим місцем користувача.

Більшість сучасних терміналів також мають вбудовані мови сценаріїв, що підтримують написання торгових стратегій як алгоритмів на основі правил, що виконуються автоматично без втручання людини щоразу, коли приймається рішення щодо купівлі або продажу певного цінного паперу.[3]

Алгоритмічна торгівля (АТ) стосується процесу здійснення фінансових транзакцій на основі певних математичних правил, програмних алгоритмів та моделей аналізу даних. У дискреційній торгівлі рішення приймає людина, але в алгоритмічній торгівлі процес аналізу, генерації торгових сигналів та виконання ордерів повністю або частково автоматизований комп'ютерною програмою. Таким чином, АТ допомагає усунути емоційний вплив у торгівлі, одночасно підвищуючи швидкість реакції, а також обробляючи великі обсяги ринкової інформації.

Частка алгоритмічної торгівлі автоматично зростає зі збільшенням пропускної здатності та доступністю високочастотних API на біржі. Значна частка торгів на провідних біржах виконується автоматизованими системами, що працюють на основі попередньо визначених стратегій. Автоматизація

забезпечує точність, масштабованість, швидку адаптацію до ринкових умов – у разі будь-яких змін ціни, обсягу чи волатильності; таким чином, підтримуючи всі форми та варіації, які можуть знадобитися на ринку.

Ключовими особливостями алгоритмічної торгівлі є формалізація правил, автоматичне виконання рішень, висока швидкість обробки інформації, стабільність та повторюваність поведінки. Вона також значною мірою включає результати, отримані в результаті оптимізації за допомогою методів машинного навчання, які мають особливі вимоги до інструментів, технологій та моделей, що використовуються для створення ефективних торгових ботів.

Торговий бот – це прикладна програма, яка проводить аналіз ринку та торгує на основі певної логіки або моделі прогнозування. Технологічний аспект його реалізації включає інфраструктурні рішення, архітектуру програмного забезпечення та алгоритмічні методи.

Що стосується інфраструктури, торговий бот зазвичай включає сервер або хмарний екземпляр для безперервної роботи, джерела ринкових даних (біржові API, новинні та аналітичні сервіси), системи зберігання та обробки (реляційні та нереляційні СУБД), а також інструменти розгортання та моніторингу. Чітке домінування Python як основної мови програмування забезпечується бібліотеками, доступними для аналізу часових рядів (pandas), машинного навчання (scikit-learn, xgboost, keras), візуалізації (matplotlib), розрахунку технічних індикаторів (TA-Lib) та інтеграції з фондовими API.

Технологічні алгоритми у створенні торгових систем завжди варіювалися від простих стратегій на основі правил до складніших ансамблевих моделей. Класичні індикаторні підходи включають перетин ковзних середніх, осцилятори, такі як RSI, MACD, стохастичний осцилятор та інші. Статистичні моделі дозволяють враховувати кореляційні властивості рядів; особливо популярними є моделі ARIMA або GARCH. Зі сучасним бумом машинного навчання з'явилися дерева рішень та випадкові ліси разом із градієнтним

бустингом, а також нейронні мережі, зокрема LSTM, для моделювання часових залежностей.

Поточна тенденція полягає в гібридних системах. Технічні індикатори як базові характеристики, статистичні індикатори волатильності, характеристики новинних настроїв та складні моделі машинного навчання. Така система може враховувати нелінійні залежності, новинний контекст та мінливість ринкових режимів.

1.2. Особливості використання торгових ботів

Використання торгових ботів охоплює алгоритмічну торгівлю на фондовому, валютному та криптовалютному ринках, високочастотну торгівлю, маркетмейкерство, арбітражні стратегії та оптимізацію портфеля. Основними перевагами торгових ботів є стабільність виконання стратегій, усунення емоційного фактора, здатність обробляти великі обсяги ринкових та новинних даних, а також висока швидкість прийняття рішень.

Однак, використання торгових ботів пов'язане з високими вимогами до якості даних та ризиком перенавчання моделей. Окрім цього, існують проблеми, пов'язані із залежністю від параметрів стратегії, інфраструктурними обмеженнями та ризиками технічних збоїв або зміни ринкового режиму. Це підкреслює важливість не лише розробки торгової системи, але й належного тестування її стабільності, оцінки та подальшої адаптації.

З огляду на проведений аналіз, у цій кваліфікаційній роботі поставлено проблему дослідження та розробки такого торгового бота, який би генерував торгові сигнали на основі комплексної оцінки ринкової ситуації, що включає технічний аналіз, математичні моделі часових рядів, методи машинного навчання та аналіз настроїв, а також дозволяє проводити тестування стратегії на історичних даних.

Для досягнення цієї мети, необхідно:

– дослідити та систематизувати технології впровадження торгових ботів.

– визначити вимоги до торгового бота, його архітектури та математичних моделей,

– розробити інформаційну модель системи та алгоритму,

– побудувати стек моделей машинного навчання та впровадити їх,

– інтегрувати аналіз настроїв як функцію у формування характеристик процесу,

– провести тестування на зворотному шляху з оцінкою результатів за основними фінансовими показниками.

РОЗДІЛ 2. РОЗРОБКА МАТЕМАТИЧНИХ ТА ІНФОРМАЦІЙНИХ МОДЕЛЕЙ

2.1. Формулювання задачі прогнозування

У цій кваліфікаційній роботі сформульовано задачу прогнозування поведінки фінансового ринку, представлену як задача регресійного аналізу за часовими рядами. Метою є розробка та навчання моделі, здатної визначати майбутню локальну максимальну оцінку ціни для заданого фінансового інструменту на певному часовому горизонті, використовуючи технічні та інформаційні характеристики.

Нехай часовий ряд цін на деякий фінансовий актив задано наступним чином:

$$P = \{p1, p2, \ldots, pt\}$$

Де pt – значення ціни в момент часу t. Для кожного моменту часу будується вектор ознак Xt, який збирає та узагальнює інформацію про поточний стан ринку.

Визначимо залежну змінну або цільову змінну як майбутній локальний максимум ціни:

$$yt = \max\{pt+1, pt+2, \ldots, pt+k\}$$

k тут визначається як горизонт прогнозування.

Така постановка задачі дозволяє оцінити потенціал руху цін у найближчому майбутньому, а також є придатною для подальшої розробки торгових рішень.

2.2. Формування простору ознак

Для побудови моделі прогнозування використовується комплексний простір ознак, сформований на основі ринкових і новинних даних. У кожен момент часу,

Xt = [T It, Lagt, V olt, Sentt]

де TI – значення технічних індикаторів,

Lag – ознаки затримки, що відображають часову динаміку параметрів,

Vol – показники волатильності та похідні характеристики свічок,

Sent – числова оцінка настрою новинного фону.

Новинні дані попередньо очищаються від екстремальних значень та масштабуються до обмеженого діапазону. Зокрема, нормалізація використовує редукцію значень до інтервалу [−1; 1], що забезпечує сумісність з іншими ознаками та стабільність навчання моделі.

Усі ознаки очищуються, масштабуються та нормалізуються перед використанням у моделюванні, що зменшує вплив шуму та покращує узагальнюючу здатність прогностичної моделі.

2.3. Архітектура стекової моделі прогнозування

Рішенням проблеми є стекована модель прогнозування, що містить алгоритми машинного навчання з рекурентною нейронною мережею LSTM. Кожна базова модель надає свій прогноз цільової змінної на ідентичному просторі ознак.

Базові моделі, що генерують часткові прогнози, тоді остаточний прогноз визначається метамоделлю наступним чином:

$$yt = gf1(Xt), f2(Xt), \ldots, fM\ (Xt)$$

Де g – це регресійна метамодель, яка об'єднує результати базових алгоритмів в одному прогнозі.

Використання стекованої архітектури дозволяє зменшити вплив випадкового шуму, підвищити стабільність результатів та покращити якість прогнозування на нових даних за рахунок комбінування різних підходів до аналізу ринку.

2.4. Алгоритм системи прогнозування

Розроблена система прогнозування функціонує як послідовний конвеєр обробки даних. Етапи включають:

- завантаження історичних ринкових та новинних даних;
- очищення та попередня обробка даних;
- формування вектора ознак, що включає технічні індикатори, параметри затримки та оцінки настроїв;
- навчання стекована модель прогнозування;
- отримання числового прогнозу майбутнього локального цінового максимуму;
- перетворення прогнозу в торгові сигнали з урахуванням порогових умов та механізмів управління ризиками.

Зв'язок між прогнозованими значеннями та торговими сигналами можна виразити таким чином:

$$Signal = \{\text{купувати, якщо } \hat{y}\,t - p\,t > \theta,$$

продавати, якщо $p\,t - \hat{y}\,t > \theta,\}$

де θ – це пороговий параметр, який визначає, наскільки агресивно або чутливо визначає чутливість торгової стратегії.

2.5. Інформаційна модель системи

Інформаційна модель системи прогнозування містить такі основні компоненти: ринкові дані, новинну інформацію, простір ознак, модель прогнозу, торгові сигнали та результати зворотного тестування. Взаємодія між компонентами відбувається послідовно відповідно до алгоритму роботи системи.

Дані ринку та новин формують вхідну інформацію, на основі якої будується вектор ознак для використання стекованою моделлю в прогнозуванні. Отримані прогнози перетворюються на торговельні рішення, а результати їх застосування оцінюються за допомогою фінансових показників ефективності, детально проаналізованих у четвертому розділі цієї роботи.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТОРГОВОГО БОТА

3.1. Вибір середовища та інструментів реалізації

Для реалізації торгового бота було обрано мову програмування Python, оскільки вона має розвинену екосистему бібліотек для роботи з часовими рядами, машинним навчанням, обробкою текстів та інтеграцією з біржовими API. У роботі використовувалися бібліотеки pandas і numpy для обробки даних, scikit-learn, xgboost і lightgbm для побудови моделей машинного навчання, tensorflow або keras для реалізації нейронних мереж, TA-Lib для обчислення технічних індикаторів, а також matplotlib для візуалізації результатів.

Проект організовано за модульною структурою з каталогами вихідного коду, даних, проміжних результатів, моделей та звітів. Це спрощує розробку, тестування та розгортання системи.

3.2. Архітектура програмної системи

Архітектура програмної системи торгового бота розроблена на основі принципів модульності та розподілу відповідальності. Основними компонентами є:

– модулі для завантаження та оновлення ринкових даних;

– модуль розробки функцій;

– модуль аналізу настроїв новин;

– модуль для стекування та навчання моделей машинного навчання,

– модуль для генерації торгових сигналів,

– модуль тестування на попередніх даних,

– модуль візуалізації результатів.

Класи та функції. Окремі класи та функції з чітко визначеними інтерфейсами використовуються для взаємодії між різними модулями. Наприклад, всередині класу конвеєра TrainPreprocessPipeline конвеєри послідовних процесів завантажують дані, об'єднують їх, очищують, розраховують індикатори та готують матриці ознак для моделей. RunBacktest розміщує угоди на основі згенерованих сигналів на історичних даних для обчислення кривої капіталу.

У реалізації торгового бота особливо важливими стають модулі попередньої обробки та побудови ознак, машинного навчання, генерації сигналів та зворотного тестування. Модуль TrainPreprocessPipeline інтегрує як обробку ринкових даних, так і новинних даних, розрахунок технічних індикаторів, підготовку зразків для моделей. Модуль train_model навчає базові моделі та за потреби створює стекову модель. Модуль зворотного тестування (RunBacktest) приймає історичні дані та послідовність сигналів, моделює угоди для генерації показників ефективності.

Система також містить додаткові компоненти для налаштування параметрів, ведення журналу та організації експерименту. Прозорість розробки, що дозволяє повторення експерименту та подальше масштабування системи, забезпечується допоміжними компонентами, що входять до основної структури.

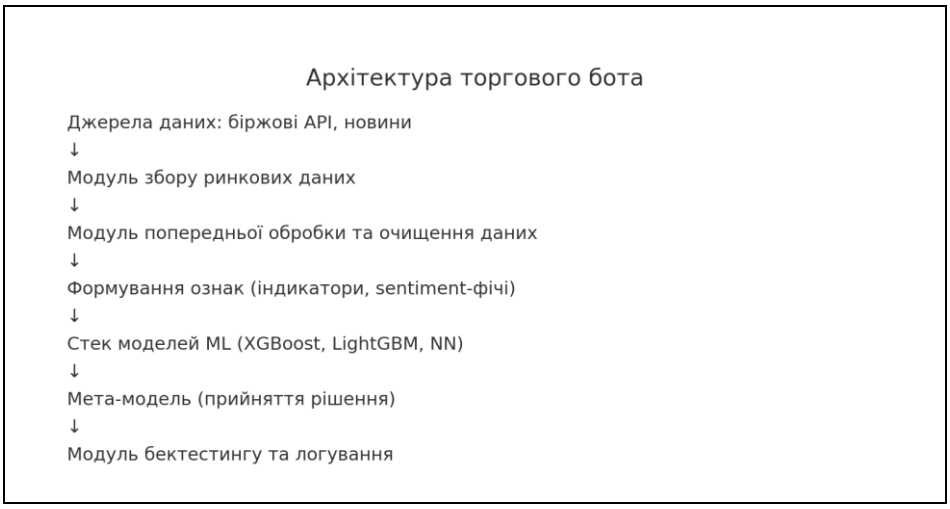


Рисунок 3.1 – Загальна архітектура торгового бота

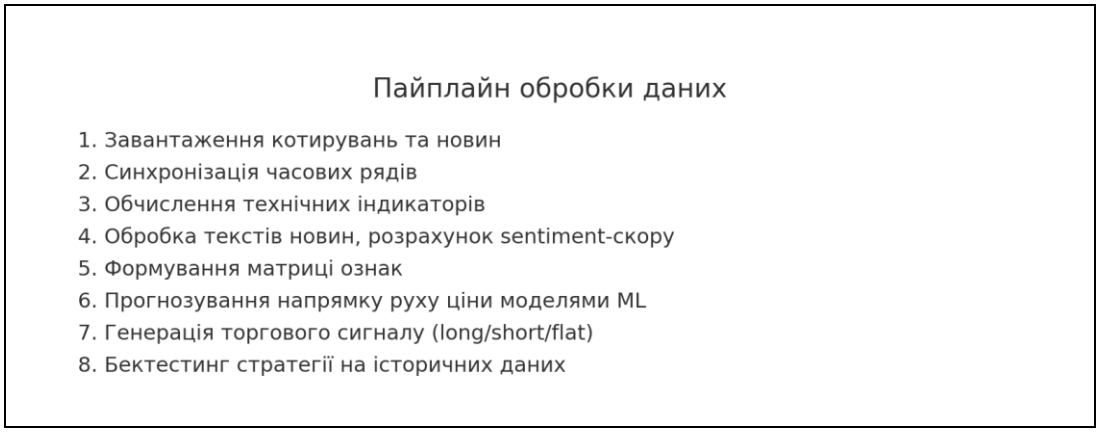


Рисунок 3.2 – Пайплайн обробки даних та прийняття рішень

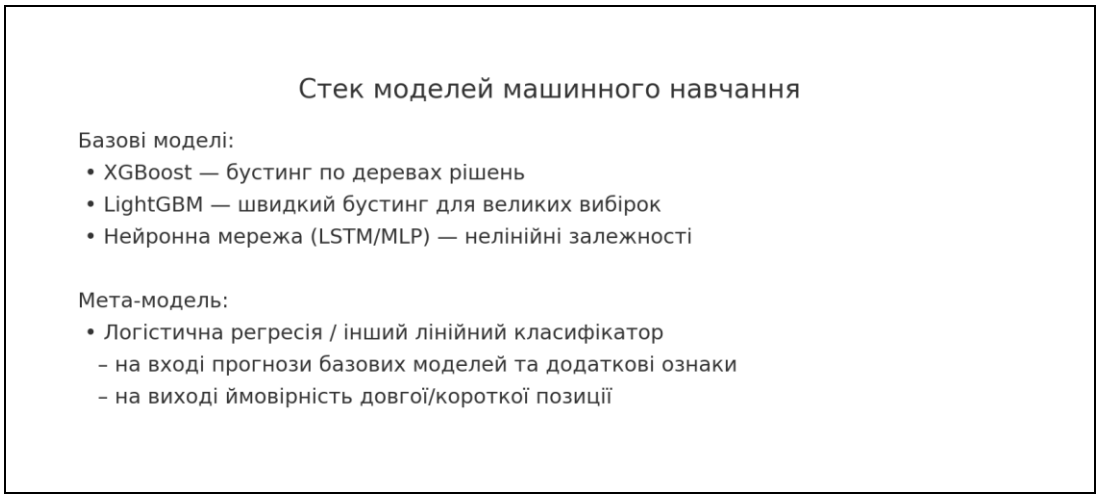


Рисунок 3.3 – Стек моделей машинного навчання

РОЗДІЛ 4. ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

4.1. Дані та умови проведення експерименту

Експериментальне дослідження було проведено для оцінки ефективності розробленої стекової моделі прогнозування та торгової стратегії, побудованої на її основі. В експерименті використовувалися історичні ринкові дані валютного інструменту, отримані з торгового терміналу, а також новинна інформація, що використовувалася для формування індикаторів настроїв.

Були сформовані часові ряди цін на основі історичних котирувань та інших параметрів, необхідних для розрахунку технічних індикаторів. Дані новин були оброблені для отримання числових оцінок настроїв, які пізніше були додані до вектора ознак. Отримані дані пройшли етапи очищення, нормалізації та об'єднання в єдиний набір для подальшого моделювання.

Для правильної оцінки якості моделі дані були розділені на навчальну, валідаційну та тестову вибірки відповідно до часової структури ряду. Це дозволяє уникнути будь-якого можливого витоку інформації з майбутніх періодів та робить експериментальні результати повністю об'єктивними.

4.2. Налаштування моделі та дизайн експерименту

В експерименті використовувалася стекована модель прогнозування, що поєднує кілька алгоритмів машинного навчання та одну рекурентну нейронну мережу типу LSTM. Модель мала завдання передбачити наступний майбутній

локальний максимум ціни на заданому часовому горизонті, що дозволило б визначити, наскільки потенційно може рухатися ринок найближчим часом.

Модель навчалася на сформованій навчальній вибірці, а контроль якості проводився на тестових даних. Отримані прогнози використовувалися для генерації торгових сигналів на основі порогових значень. Для управління ризиками використовувалися механізми обмеження збитків та фіксації прибутку, що забезпечувало контроль рівня просідання капіталу.

Торгова стратегія була протестована за допомогою процедури зворотнього тестування на історичних даних, який імітував порядок торгових операцій на історичних даних. Під час цього тесту були проаналізовані показники прибутковості та ризику, щоб дати уявлення про її стабільність та продемонструвати, наскільки вона застосовна на практиці.

4.3. Результати тестування торгової стратегії на попередніх даних

Результати тестування на попередніх даних констатують, що торгова стратегія поводиться стабільно протягом тестового інтервалу. Крива капіталу має загалом зростаючу тенденцію з деякими періодами корекції, після яких прибутковість відновлюється.

Динамічний аналіз результатів чітко показує відсутність раптових, різких або неконтрольованих спадів, що підтверджує ефективність застосованих механізмів управління ризиками. Отримані торгові сигнали формувалися відповідно до прогнозів моделі та не призводили до надмірної кількості збиткових угод.

Порівняння з базовими підходами показало, що вдосконалена система є кращою з точки зору стабільності результатів та співвідношення прибутковості

до ризику. Таким чином, це застосування стекованої моделі є обгрунтованим в алгоритмічній торгівлі.

4.4. Аналіз фінансових показників

Для кількісної оцінки ефективності торговельної стратегії було використано стандартні фінансові показники, зокрема, коефіцієнти Шарпа та Сортіно, максимальну просадку та показник рентабельності інвестицій (ROI). Значення, отримані за цими показниками, вказують на виправданий рівень прибутковості за помірного рівня ризику.

Коефіцієнт Шарпа більше одиниці, що свідчить про ефективне співвідношення дохідності та ризику від загальної волатильності портфеля. Коефіцієнт Сортіно підтверджує контроль ризику зниження, а максимальна просадка знаходиться в межах прийнятного рівня. Загальна рентабельність інвестицій є позитивною, що підтверджує практичну застосовність розробленої стратегії.

Отримані результати дають підстави для висновків щодо ефективності застосованого комбінованого підходу до прогнозування ринку та його динаміки, а також можливості подальшого розвитку запропонованої системи.

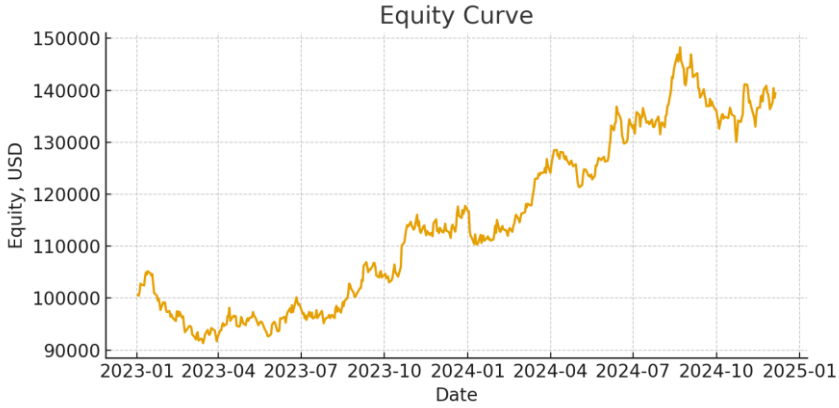


Рисунок 4.1 – Динаміка портфеля (equity curve)

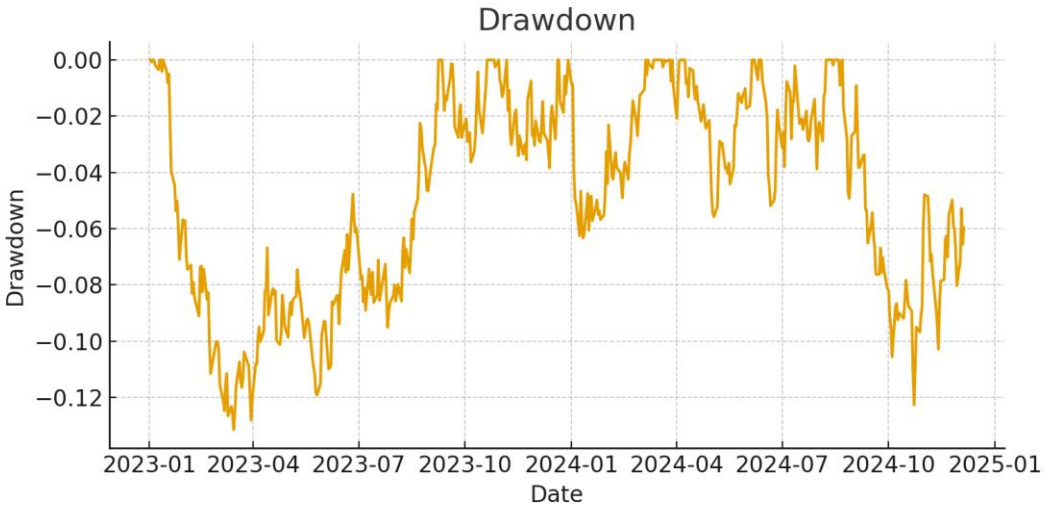


Рисунок 4.2 – Графік просадки портфеля

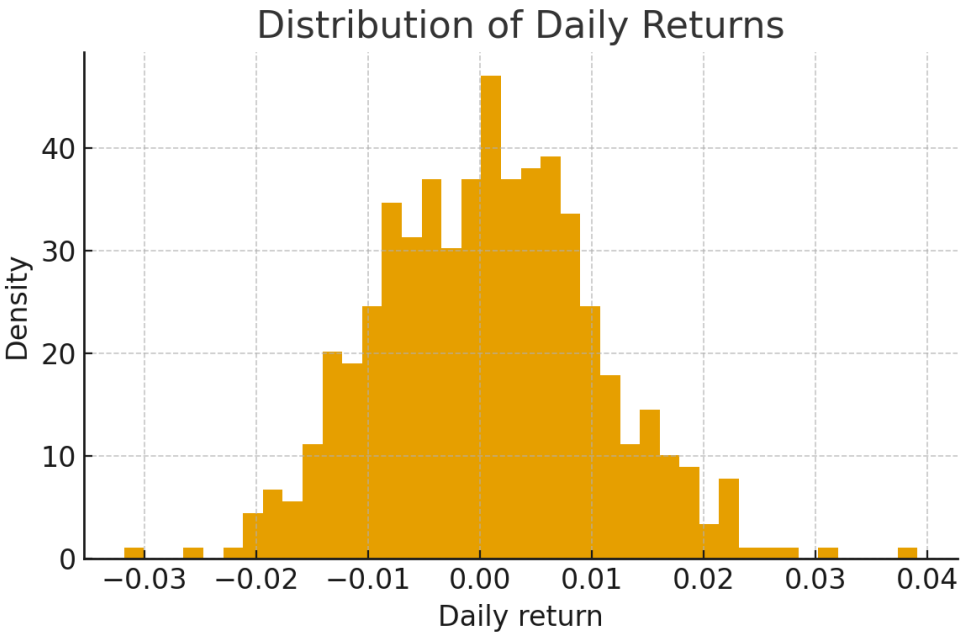


Рисунок 4.3 – Розподіл денних ретернів стратегії

ВИСНОВКИ

Під час кваліфікаційної роботи було проведено дослідження потенційного застосування комбінованих методів прогнозування для підтримки процесів прийняття торгових рішень на фінансових ринках. На основі аналізу сучасних підходів до алгоритмічної торгівлі було сформульовано теоретичні передумови для побудови системи, яка б об'єднала технічні індикатори та характеристики затримки з оцінкою новинних настроїв.

У роботі сформульовано математичну модель для задачі прогнозування. Модель була розроблена в стекової архітектурі, що поєднує кілька алгоритмів машинного навчання з нейронною мережею LSTM. Разом із самою моделлю прогнозування було реалізовано конвеєр, що містить усі етапи обробки даних – від завантаження до очищення та побудови ознак. Отриманий результат дозволив сформувати торговельну стратегію, яку можна було оцінити за допомогою інструментів зворотного тестування.

Отримані результати чітко відповідають цілям та завданням, поставленим у вступі. Модель довела свою здатність генерувати узгоджені прогнози, і була запроваджена стратегія, спрямована на стабільність результатів у межах тестового інтервалу. Отримані значення ефективності самі по собі говорять про практичну застосовність запропонованого підходу.

Підсумовуючи виконану роботу, створена система є виправданою та перспективною для подальшого розвитку. Використання комбінованих моделей машинного навчання в поєднанні з технічними та новинними особливостями підтвердило її ефективність та відкриває можливості для подальшої оптимізації торгових стратегій та розширення функціональності автоматизованих рішень.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Geron A. *Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow*. Sebastopol : O'Reilly Media, 2019. 856 p.
2. Davey K. *Building Winning Algorithmic Trading Systems*. Hoboken : Wiley Trading, 2014. 256 p.
3. Kaufman P. J. *Trading Systems and Methods*. Hoboken : Wiley, 2013. 1200 p.
4. Tsay R. S. *Analysis of Financial Time Series*. Hoboken : Wiley, 2010. 712 p.
5. Hastie T., Tibshirani R., Friedman J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York : Springer, 2009. 745 p.
6. Paul W., Baschnagel J. *Stochastic Processes: From Physics to Finance*. Cham : Springer, 2020. 420 p.
7. Jenkins G. M., Reinsel G. C., Ljung G. M. *Time Series Analysis: Forecasting and Control*. Hoboken : Wiley, 2008. 704 p.
8. Kaabar S. *New Technical Indicators: A Complete Statistical Guide*. [S. l.] : Independently Published, 2020. 320 p.
9. Kaabar S. *The New Book of Backtests: Trading Objectively*. [S. l.] : Independently Published, 2020. 280 p.
10. Kaabar S. *The Handbook of Exotic Trading Strategies*. [S. l.] : Independently Published, 2020. 350 p.
11. Chan E. *Algorithmic Trading: Winning Strategies and Their Rationale*. Hoboken : Wiley, 2013. 304 p.
12. Narang R. *Inside the Black Box: A Simple Guide to Quantitative and High-Frequency Trading*. Hoboken : Wiley, 2013. 352 p.
13. López de Prado M. *Advances in Financial Machine Learning*. Hoboken : Wiley, 2018. 552 p.

14. Patel J., Shah S., Thakkar P. Predicting stock market movement using machine learning techniques // *International Journal of Computer Applications*. 2015. Vol. 111, No. 1. P. 1–6.

15. Zhang Y., Aggarwal C. Neural approaches to financial forecasting // *Journal of Machine Learning Research*. 2018. Vol. 19. P. 1–30.

ДОДАТОК А

Модуль передобробки даних TrainPreprocessPipeline

Нижче наведено фрагмент коду, що реалізує конвеєр передобробки даних, обчислення технічних індикаторів та підготовку навчальної вибірки для моделей машинного навчання.

```
from tqdm import tqdm
import pandas as pd
from src.features.preprocess_utils import (
LLMSentimentAnalysis,
TAIndicators,
NumericalPreprocess,
SplitData
)
from src.utils.data_preprocess_log_config import logger

class TrainPreprocessPipeline:
"""
#! Use `"./data/interim/merged_scrapped_n_price.csv"`
if setting `with_llm_sentiment`=`True` in the `run` method.

Set `mock` to `True` to use a mock sentiment analysis model (always returns neutral sentiment).
"""
def __init__(
self,
data_path: str = "./data/interim/parsed_scraped_data_clipped.csv",
with_llm_sentiment: bool = False,
mock: bool = False,
) -> None:
logger.info("Initializing TrainPreprocessPipeline")
self.__source_file_path = data_path
self.with_llm_sentiment = with_llm_sentiment
```

```
    self.mock = mock
    self.llm_sentiment_analysis = LLMSentimentAnalysis(mock=self.mock)
    self.ta_indicators = TAIndicators()
    self.numerical_preprocess = NumericalPreprocess()
    self.split_data = SplitData()

    def __load_data(self) -> pd.DataFrame:
    """
    Loads data from the specified file path.

    Args:
    None

    Returns:
    pd.DataFrame: The loaded data from the CSV file.

    Raises:
    FileNotFoundError: If the file specified in the source file path does not exist.
    """
    logger.info("Loading data from %s", self.__source_file_path)
    try:
    return pd.read_csv(self.__source_file_path)
    except FileNotFoundError as e:
    logger.error("Error loading data from %s:\n %s", self.__source_file_path, e)

    def run(
    self,
    lookback: int = 6,
    target_column: str = "target",
    save_path: str = "./data/processed/"
    ) -> tuple[pd.DataFrame, pd.DataFrame, pd.DataFrame]:
    """
    Runs the preprocessing pipeline to process raw data, apply sentiment analysis (optional),
    add technical analysis indicators, and prepare time series data for training, testing, and validation.
```

ДОДАТОК Б

Модуль навчання моделей train_model

Фрагмент коду, що відповідає за навчання моделей машинного навчання та збереження навчених ваг.

```
from datetime import datetime
from time import perf_counter
import json
from math import sqrt
import pandas as pd
from pandas import Index
import numpy as np
import joblib
from sklearn.ensemble import RandomForestRegressor
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import root_mean_squared_error
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from xgboost import XGBRegressor, Booster
import lightgbm as lgb
from src.utils.model_train_log_config import logger

# Tensorflow libraries
import tensorflow as tf
from tensorflow.keras.models import Sequential
import tensorflow.keras.layers as L

class SelectFeatures():
def __init__(
self,
top_k: int = 20,
train_data_path: str = "./data/processed/series_train_preprocessed.csv",
test_data_path: str = "./data/processed/series_test_preprocessed.csv",
val_data_path: str = "./data/processed/series_validate_preprocessed.csv",
```

```
save_path: str = "./artefacts/",
report_path: str = "./reports/",
) -> None:
self.top_k = top_k

self.__save_path = save_path
self.__report_path = report_path

self.train_data_path = train_data_path
self.test_data_path = test_data_path
self.val_data_path = val_data_path

self.train_data: pd.DataFrame = None
self.test_data: pd.DataFrame = None
self.val_data: pd.DataFrame = None

self.__load_data()
self.X_train, self.y_train = self.__get_X_y(self.train_data)
self.X_test, self.y_test = self.__get_X_y(self.test_data)
self.X_val, self.y_val = self.__get_X_y(self.val_data)

self.rf_rmse: float = None
self.xgb_rmse: float = None
self.lgb_rmse: float = None

self.rf_lstm_rmse: float = None
self.xgb_lstm_rmse: float = None
self.lgb_lstm_rmse: float = None

self.rf_time: float = None
self.xgb_time: float = None
```

ДОДАТОК В

Модуль бектестингу RunBacktest

Фрагмент коду модуля бектестингу, що виконує симуляцію угод на історичних даних.

```
from time import perf_counter, strftime, gmtime
import datetime
import pandas as pd
from backtesting import Backtest, Strategy
from src.backtest.backtest_utils import PREDICTIONS

#! Toggle this to switch between hardcoded backtest length of 9000
#! or dynamic based on the pipeline configuration
MANUAL_GAME = True
if MANUAL_GAME:
PREDICTIONS = PREDICTIONS[-9000:]


class RunBacktest:
def __init__(
self,
raw_data_path: str = "./data/interim/merged_scrapped_n_price.csv",
predictions: list = None,
strategy: Strategy = None,
cash: float = 300,
margin: float = 1/5,
commission: float = 0.0002,
report_path: str = "./reports/",
):
self.raw_data = pd.read_csv(raw_data_path)
self.predictions = predictions
self.strategy = strategy
self.cash = cash
self.margin = margin
```

```
    self.commission = commission
    self.report_path = report_path

    self.backtest_data = self.__prepare_data()

    def run(self) -> None:
    """
    Executes the backtest and optimization for the specified strategy.

    This method performs a backtest using the provided strategy, optimizes key parameters,
    saves the results to CSV files, and outputs statistics and a heatmap.

    Args:
    None

    Returns:
    None

    Outputs:
    - Saves backtest statistics to `{self.report_path}backtest_stats.csv`.
    - Saves optimization heatmap to `{self.report_path}backtest_heatmap.csv`.
    - Prints backtest runtime, statistics, and heatmap.

    Optimization Parameters:
    - `close_days`: Range of days to close overdue trades (1 to 2).
    - `sl_perc`: Stop loss percentages (0.001 to 0.008 in steps of 0.001).
    - `tp_perc`: Take profit percentages (0.001 to 0.008 in steps of 0.001).
    - `buy_threshold`: Buy thresholds (0.001 to 0.005 in steps of 0.001).
    - `sell_threshold`: Sell thresholds (0.001 to 0.005 in steps of 0.001).

    Example:
    RunBacktest().run()
    """
    print("Running backtest for {} predictions...".format(len(self.predictions)))

    start = perf_counter()
    backtest = Backtest(
```

```
data=self.backtest_data,
strategy=self.strategy,
cash=self.cash,
margin=self.margin,
commission=self.commission
)

stats, heatmap = backtest.optimize(
close_days=[i for i in range(1, 3)],
sl_perc=[i/1000 for i in range(1, 9)],  # Expanding the range for stop loss
tp_perc=[i/1000 for i in range(1, 9)],  # Expanding the range for take profit
buy_threshold=[i/1000 for i in range(1, 6)],  # Testing different buy thresholds
sell_threshold=[i/1000 for i in range(1, 6)],  # Testing different sell thresholds
maximize='Return [%]', max_tries=5000,
```