Міністерство освіти і науки України
Український державний університет науки і технологій

Факультет Комп'ютерні технології та системи
Кафедра Комп'ютерні інформаційні технології

Пояснювальна записка

до кваліфікаційної роботи
магістра

на тему: «Використання нейромережевих технологій для короткострокового прогнозування економічних показників фондового ринку України»

за освітньою програмою **12 Інженерія програмного забезпечення**
зі спеціальності: **121 Інженерія програмного забезпечення**

Виконав: студент групи ПЗ2326: Владислав СИВАК

Керівник: Світлана ВОЛКОВА

Нормоконтролер: Світлана ВОЛКОВА

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

Студент

Дніпро – 2025 рік

Ministry of Education and Science of Ukraine

Ukrainian State University of Science and Technologies

Faculty Computer technologies and systems
Department Computer information technology

Explanatory Note

to Master's Thesis
master

on the topic: «The use of neural network technologies for short-term forecasting of economic indicators of the stock market of Ukraine»

according to educational curriculum **12 software engineering**
in the Speciality: **121 software engineering**

Done by the student of the group PZ2226: Vladislav SIVAK

Scientific Supervisor: Svitlana VOLKOVA

Normative controller: Svitlana VOLKOVA

Dnipro – 2025

РЕФЕРАТ

Об'єктом дослідження є нейронні мережі фондового ринку драгоцінних металів України.

Предметом дослідження є прогнозування цін на драгоціні метали (золото,срібло) за допомогою методів машинного навчання. Ціни представлени у вигляді часових рядів.

Метою роботи є прогнозування цін драгоцінних металів (золото,срібло) на фондовому ринку України.

Методи дослідження: аналітичний, експериментальний, статистичний та моделювання.

Результати та їх новизна: розроблено систему для прогнозування цін на дорогоцінні метали за допомогою нейромереж, що включає новітні методи обробки та аналізу даних. Система використовує покращені алгоритми машинного навчання та випадковий ліс для точного прогнозування, що є новаторством у сфері фінансового аналізу. Результати дослідження відкривають нові можливості для застосування нейромережевих технологій на фондовому ринку, сприяючи розвитку ефективних інструментів для інвесторів та фахівців у сфері економіки.

Пояснювальна записка складається зі вступу, 4 розділів, висновків, бібліографічного списку та 2 додатків:

– у вступі описано сутність розробленого методу прогнозування, та його актуальність. Складається із 4 сторінок;

– у першому розділі проведено аналіз теоретичних основ нейромереж та специфіки фондового ринку дорогоцінних металів. Складається з 26 сторінок;

– у другому розділі аналізуються існуючі методи прогнозування на фондовому ринку та підготовка даних для навчання нейромережі. Складається з 32 сторінок;

– у третьому розділі описано проектування та розробку програмного забезпечення для нейромережевого прогнозування. Складається з 34 сторінок;

– у четвертому розділі проведено тестування та оцінка ефективності розробленої системи. Складається з 16 сторінок;

– додатки містять лістинги розробленого програмного забезпечення та фрагмент тренувальних наборів.

Таблиць – 2, рисунків – 48, бібліографія – 53 джерела.

Ключові слова: нейронна мережа, прогнозування, фінансові часові лави, помилка прогнозу.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАК, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

КНМ - Конволюційні нейронні мережі

РНМ - рекурентні нейронні мережі

ШЛ - Швидкі ліси (Fast Forest)

ШНМ - штучні нейронні мережі

ARIMA - авторегресійна інтегрована модель ковзного середнього

MAE - Середня абсолютна помилка (Mean Absolute Error)

ML.NET - Платформа для машинного навчання від Microsoft

MSE - Середньоквадратична помилка (Mean Squared Error)

NN - Нейронна мережа (Neural Network)

R^2 - Коефіцієнт детермінації

RMSE - Корінь з середньоквадратичної помилки (Root Mean Squared Error)

SQL - Мова структурованих запитів (Structured Query Language)

VS - Microsoft Visual Studio

ВСТУП

У сучасному світі, де цифровізація та інформаційні технології стрімко входять у всі сфери економічної діяльності, актуальність використання нейромережевих технологій для прогнозування економічних показників дорогоцінних металів на фондовому ринку України стає все більш важливою. Ця актуальність обумовлена низкою факторів.

З одного боку, дорогоцінні метали, як золото та срібло, традиційно вважаються важливими елементами економічної стабільності та індикаторами фінансової надійності. Вони є невід'ємною частиною інвестиційних портфелів і резервів банків. З іншого боку, нестабільність світових ринків, зокрема через геополітичні конфлікти, економічні санкції та глобальні кризи, робить прогнозування цін на ці метали надзвичайно важливим [1].

Використання нейромережевих технологій у цьому контексті набуває вирішального значення. Ці технології здатні аналізувати великі обсяги даних, виявляти закономірності та тенденції, що неможливо виявити за допомогою традиційних методів. Такий підхід може значно підвищити точність прогнозів, знизити ризики для інвесторів і допомогти регуляторам у прийнятті обґрунтованих рішень.

В Україні, з огляду на її постійно зростаючу інтеграцію у світову економіку та активну цифрову трансформацію, розвиток і впровадження таких технологій є особливо актуальним. Ефективне застосування нейромережевих систем для прогнозування економічних показників може стати важливим фактором підвищення конкурентоспроможності українського фінансового ринку, забезпечення його стабільності та прозорості.

Мета дослідження полягає у розробці методу для короткострокового прогнозування економічних показників дорогоцінних металів на фондовому ринку України.

Для досягнення цієї мети визначено наступні основні задачі:

– аналіз сучасного стану проблеми прогнозування економічних показників дорогоцінних металів. Оцінка існуючих наукових досліджень, виявлення невирішених проблем у даній галузі та аналіз потенціалу використання нейромережевих технологій для покращення точності прогнозування;

– обґрунтування вибору методів нейромережевого прогнозування. Теоретичний аналіз і порівняння різних підходів та методів прогнозування, зокрема нейромережевих, з метою визначення найбільш ефективних стратегій для реалізації задачі;

– розробка ПЗ для нейромережевого прогнозування: Формулювання вимог до програмного забезпечення, проектування архітектури системи, вибір технологій для реалізації та розробка інтерфейсу користувача;

– експериментальне дослідження ефективності нейромережевих моделей. Проведення експериментів з використанням розробленого програмного забезпечення, аналіз результатів та оцінка точності та ефективності нейромережевих моделей у контексті задачі прогнозування.

Об'єкт дослідження: процеси функціонування та динаміки фондових ринків, що зосереджені на сегменті дорогоцінних металів. Цей об'єкт охоплює широкий спектр економічних взаємодій та змін, які відбуваються на ринку дорогоцінних металів, і включає в себе варіабельність цін, торговельні об'єми, а також зовнішні економічні та політичні фактори, що впливають на ринок.

Предмет дослідження: використання нейромережевих технологій для короткострокового прогнозування економічних показників дорогоцінних металів. Предмет зосереджується на розробці, аналізі, та оцінці ефективності нейромережевих моделей для прогнозування змін у цінах, попиті та інших ключових економічних індикаторах, що визначають стан сегменту дорогоцінних металів на фондовому ринку.

Для досягнення мети дослідження було застосовано наступні методи:

– аналітичний метод. Використовувався для аналізу сучасного стану проблеми та оцінки існуючих наукових досліджень у галузі прогнозування економічних показників дорогоцінних металів. Цей метод дозволив виявити ключові

невирішені проблеми та оцінити потенціал використання нейромережевих технологій у цій сфері;

– метод порівняльного аналізу. Застосовувався для порівняння традиційних та нейромережевих методів прогнозування, з метою визначення переваг та недоліків кожного підходу. Це допомогло обґрунтувати вибір нейромережевих методів для подальшої розробки;

– експериментальний метод. Використовувався для проведення експериментів з нейромережами, включаючи реалізацію та тренування моделей. Цей метод дозволив оцінити ефективність та точність прогнозування нейромережами, порівняти різні архітектури та підходи до навчання моделей;

– статистичний метод. Використовувався для аналізу результатів експериментів, включаючи оцінку точності та надійності прогнозів, виконаних нейромережами. Цей метод дозволив провести кількісний аналіз отриманих даних та визначити статистичну значущість результатів;

– метод моделювання. Використовувався для створення комп'ютерних моделей нейромереж, що дозволило імітувати реальні умови фондового ринку та оцінити ефективність прогнозування в контрольованому середовищі.

Наукова новизна одержаних результатів у рамках дослідження полягає в наступному:

– розробка інноваційної системи прогнозування з використанням методу «Швидкі ліси». У дослідженні було вперше створено та імплементовано систему прогнозування, яка базується на алгоритмі швидких лісів. Цей підхід значно підвищує швидкість обробки даних та точність прогнозування, що є новаторським для умов українського фондового ринку;

– адаптація методу ШЛ до специфіки ринку дорогоцінних металів. Особливість розробленої системи полягає у її здатності адаптуватися до особливостей ринку дорогоцінних металів, забезпечуючи високу точність прогнозів за допомогою оптимізованого аналізу історичних даних.

– оптимізація використання ресурсів з використанням ШЛ. Важливою перевагою розробленого ПЗ є його ефективність у використанні обчислювальних

ресурсів, що робить його доступним для широкого кола користувачів, включаючи малі та середні інвестиційні компанії.

Практичне значення одержаних результатів дослідження виявляється у кількох аспектах:

– покращення процесів прийняття рішень на фондовому ринку. Реалізована система надає інвесторам та аналітикам фондового ринку потужний інструмент для аналізу та прогнозування цін на дорогоцінні метали. Це допомагає у прийнятті більш обґрунтованих інвестиційних рішень та зниженні ризиків, пов'язаних з коливаннями ринку;

– застосування в банківському та фінансовому секторах. Банки та фінансові установи можуть інтегрувати цю систему у свої аналітичні відділи для покращення стратегій управління активами та портфельних інвестицій, що засновані на дорогоцінних металах;

– підвищення ефективності та швидкості аналізу даних. Система значно скорочує час, необхідний для аналізу великих обсягів даних, завдяки оптимізованому алгоритму ШЛ. Це дозволяє оперативно реагувати на ринкові зміни та вчасно адаптувати інвестиційні стратегії;

– забезпечення конкурентних переваг для малих та середніх інвестиційних компаній. Менші компанії, які раніше не мали доступу до високоточних інструментів прогнозування через високу вартість та складність, тепер можуть використовувати цю систему для підвищення своєї конкурентоспроможності на ринку.

1 АНАЛІЗ СУЧАСНОГО СТАНУ ДОСЛІДЖЕННЯ ПРОБЛЕМИ

1.1 Аналіз сучасного стану дослідження за науковими літературними джерелами

1.1.1 Огляд існуючих досліджень у галузі прогнозування економічних показників дорогоцінних металів

Історичний огляд прогнозування цін на дорогоцінні метали демонструє еволюцію від класичних до передових технік. Початкові методи були засновані на економетричних моделях, таких як авторегресійні моделі (AR) і моделі рухомих середніх (MA), а також їх комбінації в моделях ARMA та ARIMA. Ці моделі використовували історичні дані для виявлення тенденцій та сезонних коливань у цінах на дорогоцінні метали [2].

З розвитком комп'ютерних технологій та алгоритмічного трейдингу почалося застосування складніших інструментів, таких як моделі векторної авторегресії (VAR) та ковінтеграційний аналіз. Ці методи дозволяли аналізувати взаємозв'язки між різними фінансовими показниками та цінами на дорогоцінні метали, враховуючи часові затримки та змінність показників [3].

Значний стрибок у розвитку методів прогнозування відбувся з появою та розвитком машинного навчання та нейромереж. Нейромережеві моделі, зокрема глибокі нейронні мережі, стали використовуватися для аналізу складних шаблонів та неочевидних взаємозв'язків у даних. Ці моделі виявилися ефективними у виявленні нелінійних тенденцій та аномалій, які традиційні економетричні підходи часто ігнорували.

Важливим аспектом аналізу є вплив глобальних економічних криз, таких як фінансова криза 2008 року або пандемія COVID-19, на ціни дорогоцінних металів. Такі події часто призводять до значних коливань цін через зміну інвестиційних настроїв і попиту на «безпечні гавані».

Опублікована стаття [5] пропонує новий підхід до мультифрактальних даних для поліпшення моделювання та прогнозування цін на дорогоцінні метали.

Дослідження використовує аналіз когерентності кратних вейвлетів і мультифрактальний детрендований флуктуаційний аналіз для визначення локальних показників Херста в часових рядах цін на золото, срібло та платину. Воно виявило, що всі ці метали мають мультифрактальні характеристики та виявляють фазовий рух у довгострокових періодах. Використовуючи модель векторної авторегресивної фракційно інтегрованої рухливої середньої (VARFIMA) [6], дослідження показує, що ця модель значно перевершує свої одновимірні аналоги (ARFIMA) [7] у всіх успішних випробуваннях, підкреслюючи важливість здобуття співробітництва в моделюванні.

У рамках дослідження [8], проведеного в Наньянському технологічному університеті, Сінгапур, було використано глибокі нейронні мережі для прогнозування цін на золото. Основна увага була зосереджена на моделі LSTM (довга короткочасна пам'ять), яка відома своєю здатністю ефективно обробляти послідовні дані, особливо в контексті фінансових ринків, де часові ряди відіграють ключову роль. Дослідники також використовували комбінації LSTM з іншими типами нейромереж, включаючи CNN (конволюційні нейронні мережі), які ефективні у виявленні складних шаблонів в даних [9].

Особливо важливим виявилося використання моделі Bi-LSTM-CNN, яка поєднує переваги обох LSTM та CNN. Виявлено, що ця комбінована модель показала значно кращі результати у порівнянні з іншими моделями, включаючи стандартну LSTM, особливо в контексті прогнозування довгострокових цінових тенденцій золота. Це підкреслює потенціал інтеграції різних типів нейромереж для покращення якості прогнозування в складних фінансових умовах [10].

1.1.2 Висвітлення невирішених проблем у цій сфері

У сучасному світі цифрових інновацій та швидкого розвитку технологій, використання нейромережевих технологій для прогнозування економічних показників дорогоцінних металів набуває все більшої актуальності, особливо на такому динамічному ринку, як український. Проте, незважаючи на значні досягнення у цій області, існують низка невирішених проблем та викликів, які

потребують додаткових досліджень та розробок. Ці виклики стосуються як технічних аспектів використання нейромереж, так і ширших економічних та ринкових умов. До таких проблем належать:

Недостатність та низька якість даних

Недостатність та низька якість даних становлять серйозну перешкоду для розвитку та впровадження нейромережевих технологій у сфері прогнозування цін на дорогоцінні метали. Якість даних впливає на весь процес навчання нейромереж, від первинного аналізу до остаточного прогнозування. Найчастіше ці проблеми пов'язані із недостатньою кількістю детальних, надійних та актуальних історичних даних. Наприклад, неповнота даних про транзакції, нестабільність фінансових ринків або недостатньо довгий часовий проміжок можуть впливати на якість тренування та точність прогнозів.

Крім того, помилки в даних, що можуть виникати внаслідок невірного введення інформації, технічних збоїв у системах збору даних або зміни в методології збору даних, можуть призвести до неточностей у прогнозах. Це особливо критично, коли моделі намагаються виявити складні взаємозв'язки та закономірності в динаміці цін на дорогоцінні метали. Такі помилки можуть призвести до перенавчання (overfitting) або недонавчання (underfitting) моделі, що погіршує її загальну здатність до прогнозування.

Вирішення цих проблем вимагає посилення уваги до процесу збору та аналізу даних, впровадження передових методів їх очищення та обробки [11]. Також важливим є використання додаткових джерел даних для забезпечення більш широкого та різноманітного вхідного набору для навчання моделей. У такому контексті, важливою є співпраця між фінансовими установами, регуляторами ринку та науковими організаціями для розробки стандартів якості та форматів обміну даними.

Розробка моделей, які можуть точно передбачати ці коливання, є важливою і складною задачею. Це вимагає використання більш складних та вдосконалених нейромережевих архітектур, які можуть включати глибокі навчання, рекурентні нейронні мережі, а також новітні підходи, такі як змішані та ансамблеві моделі.

Обмеження та виклики машинного навчання

У сфері використання нейромереж для прогнозування цін на дорогоцінні метали, машинне навчання зіштовхується з кількома важливими викликами та обмеженнями (рис. 1.2). Ці виклики впливають як на процес розробки та налаштування моделей, так і на їх здатність до точного прогнозування. Вибір правильної архітектури мережі та налаштування гіперпараметрів вимагають глибокого розуміння як самої задачі, так і специфіки даних. Неправильний вибір може призвести до слабкої продуктивності моделі або її нездатності до адекватного прогнозування. Тому важливо знайти баланс між складністю моделі та її загальною здатністю до прогнозування, щоб забезпечити точність та надійність результатів.

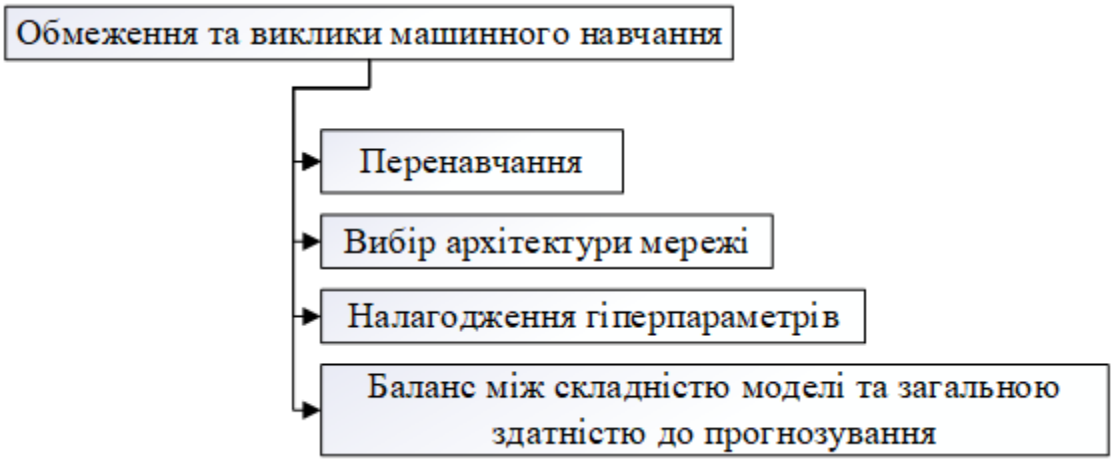

Рисунок 1.2 – Фактори обмеження машинного навчання

Основні фактори, що впливають на обмеження машинного навчання складаються із:

– перенавчання (Overfitting). Це стан, коли модель надто точно адаптується до тренувального набору даних, втрачаючи здатність до загального прогнозування на нових даних;

– вибір архітектури мережі. Знаходження оптимальної архітектури нейронної мережі, яка відповідає специфіці задачі прогнозування, є важливим для досягнення точних результатів;

– налагодження гіперпараметрів. Правильне налаштування гіперпараметрів, таких як швидкість навчання, кількість нейронів та шарів, є вирішальним для ефективності навчання моделі;

– баланс між складністю моделі та загальною здатністю до прогнозування. Важливо знайти баланс між складністю моделі та її здатністю до загального прогнозування, щоб уникнути перенавчання та забезпечити високу точність прогнозів.

Перелічені виклики потребують ретельного планування та експериментування під час розробки нейромережевих моделей для прогнозування цін на дорогоцінні метали.

1.1.3 Оцінка потенціалу використання нейромережевих технологій для прогнозування

Використання нейромережевих технологій відкриває нові можливості для аналізу та прогнозування фінансових ринків. Завдяки здатності нейромереж до обробки великих обсягів даних і виявлення складних закономірностей, цей підхід може значно підвищити точність прогнозів, особливо у сфері короткострокового прогнозування цін на дорогоцінні метали на фондовому ринку України.

Конволюційні нейронні мережі (CNN) відомі своєю ефективністю в аналізі візуальних даних. Вони використовуються для розпізнавання образів та відеоаналізу, але також мають застосування в обробці часових рядів. Це стає можливим завдяки їх здатності виявляти ієрархічні шаблони в даних. В контексті фінансового аналізу, CNN можуть бути корисними для ідентифікації візуальних шаблонів у діаграмах цін або обсягів торгів [13].

Рекурентні нейронні мережі (RNN) ідеально підходять для роботи з послідовностями даних, такими як часові ряди, текст, або мова. Особливість RNN полягає в їх здатності «пам'ятати» попередні входи в мережу, що дозволяє їм враховувати контекстуальну інформацію при аналізі послідовностей. Це робить RNN особливо цінними у фінансовому прогнозуванні, де минулі тенденції та шаблони можуть дати важливі підказки про майбутні рухи ринку [14].

Глибокі нейронні мережі (DNN) забезпечують ще більшу гнучкість і потужність завдяки своїй структурі, яка включає багато шарів обробки. Ці шари дозволяють мережі виявляти дуже складні та абстрактні шаблони в даних. У

фінансовому аналізі, DNN можуть бути використані для інтеграції та аналізу різноманітних джерел даних, забезпечуючи глибокий та всебічний аналіз [15].

Щодо обробки даних, нейромережі демонструють здатність ефективно працювати з різними типами даних. Це включає структуровані дані, як-от числові часові ряди, і неструктуровані дані, такі як текстові новини або макроекономічні звіти. Здатність нейромереж обробляти та інтегрувати такі різні форми даних відкриває широкі можливості для більш точного та глибокого аналізу фінансових ринків, зокрема для прогнозування цін на дорогоцінні метали.

Нейромережеві технології пропонують інноваційний підхід до прогнозування економічних показників, особливо в контексті волатильних ринків, таких як торгівля дорогоцінними металами на фондовому ринку. Цей підхід, що використовує штучний інтелект та машинне навчання, може значно вдосконалити аналіз та прогнозування ринкових тенденцій. На рис. 1.3 представлено ключові переваги та недоліки використання нейромереж у цій сфері.

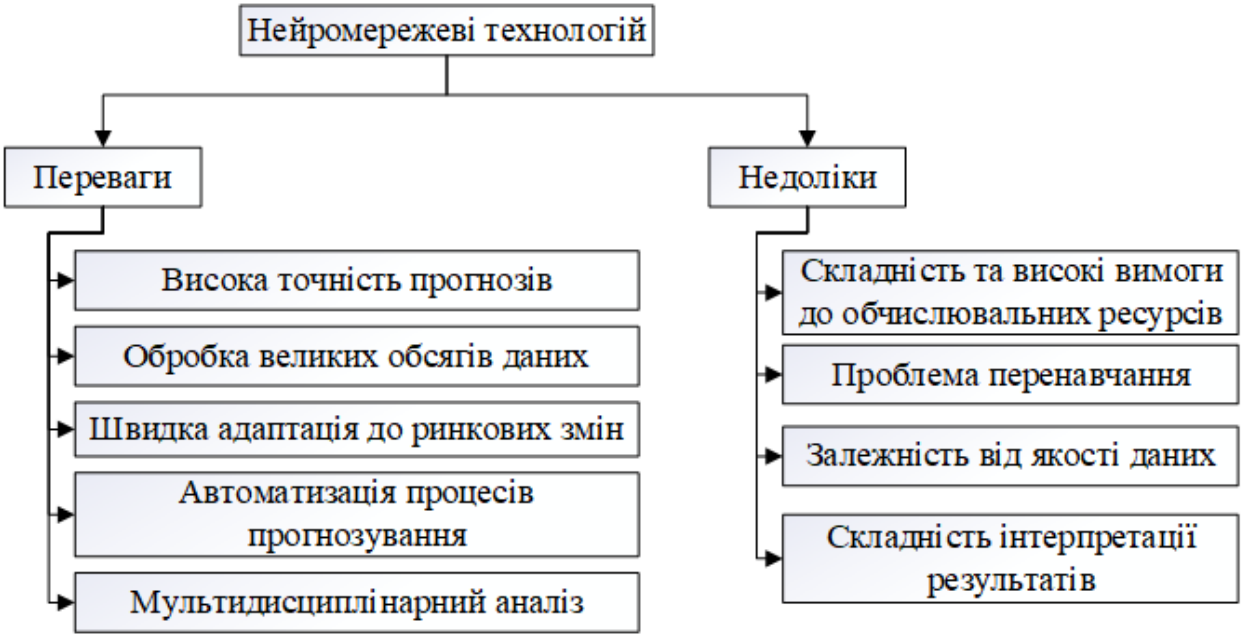

Рисунок 1.3 –Переваги та недоліки нейромережевих технологій

Переваги нейромережевих технологій складаються із:

– висока точність прогнозів. Нейромережі здатні аналізувати великі обсяги даних та виявляти складні закономірності, що можуть бути недоступними для традиційних статистичних методів. Це підвищує точність прогнозування цін на дорогоцінні метали;

– обробка великих обсягів даних. Нейромережі ефективно справляються з великими та різноманітними наборами даних, включаючи часові ряди, фінансові звіти та новинні статті;

– швидка адаптація до ринкових змін. Нейромережі можуть швидко реагувати на нові дані, що дозволяє оперативно оновлювати прогнози у відповідь на зміни на ринку;

– автоматизація процесів прогнозування. Нейромережі зменшують потребу в ручному аналізі, автоматизуючи процеси обробки та аналізу даних.

– мультидисциплінарний аналіз. Ці системи можуть інтегрувати дані з різних джерел, включаючи економічні індикатори, політичні події, та соціальні медіа, для комплексного розуміння ринку.

Недоліки нейромережевих технологій:

– складність та високі вимоги до обчислювальних ресурсів. Нейромережі вимагають значних обчислювальних потужностей та експертизи у сфері машинного навчання, що може бути дорогим та складним для реалізації;

– проблема перенавчання. Існує ризик, що нейромережа «вивчить» шум у даних, приймаючи його за важливі шаблони, що може призвести до неточних прогнозів;

– залежність від якості даних. Точність прогнозів нейромереж безпосередньо залежить від якості, точності та повноти вхідних даних;

– необхідність постійного оновлення моделей. Ринкові умови постійно змінюються, що вимагає регулярного оновлення та налаштування нейромережевих моделей;

– складність інтерпретації результатів. Результати, які надає нейромережа, можуть бути складними для інтерпретації, особливо в порівнянні з традиційними статистичними методами.

Хоча нейромережеві технології пропонують значні переваги у прогнозуванні економічних показників дорогоцінних металів на фондовому ринку, необхідно також враховувати потенційні недоліки та складності, пов'язані з їх використанням.

Ці системи вимагають ретельного проектування, налаштування та управління для максимальної ефективності та точності прогнозів [16].

На зарубіжних ринках, особливо у розвинених країнах, нейромережеві технології стали ключовим елементом у аналізі фінансових ринків. Завдяки значному розвитку фінансових технологій та доступності обширних баз даних, вони змогли розвинути багатий досвід у використанні штучного інтелекту для фінансового аналізу.

Інноваційні фінтех-стартапи та великі корпорації є лідерами у застосуванні нейромереж для прогнозування цін на дорогоцінні метали. Ці організації інтегрували нейромережеві алгоритми в свої торгові платформи, що дозволяє їм не тільки аналізувати минулі та поточні тенденції ринку, але й надавати детальні рекомендації щодо торгівлі. Це допомагає трейдерам та інвесторам приймати більш обґрунтовані рішення, виходячи з глибокого та комплексного аналізу ринкових даних.

Ще одним аспектом, який відрізняє зарубіжний досвід, є використання розширених даних. Аналітики та трейдери не обмежуються традиційними фінансовими даними, а також активно використовують соціальні медіа, новинні стрічки, економічні індикатори та інші неструктуровані джерела інформації. Це дозволяє їм отримувати більш всебічний погляд на ринкові умови та підвищувати точність своїх прогнозів, беручи до уваги широкий спектр впливових факторів.

Можна стверджувати, що зарубіжний досвід у використанні нейромереж для прогнозування цін на дорогоцінні метали відзначається високим рівнем інновацій, комплексним підходом до аналізу даних та активним впровадженням новітніх наукових розробок у практичну діяльність.

В Україні використання нейромережевих технологій у фінансовому секторі, особливо для прогнозування ринку дорогоцінних металів, все ще є порівняно новим явищем. Це пояснюється рядом факторів, що включають початкові стадії розвитку технологій, обмеження доступу до даних та активність у сфері наукових досліджень.

Український фінансовий сектор починає впроваджувати нейромережі для аналізу ринку, але розвиток цієї практики все ще знаходиться на ранніх стадіях. Це

обумовлено, з одного боку, обмеженими ресурсами, які можна було б вкласти у розвиток і впровадження таких технологій. З іншого боку, існує дефіцит кваліфікованих фахівців у галузі машинного навчання та штучного інтелекту, що також сповільнює процес інтеграції цих технологій у фінансові операції.

Аналіз зарубіжного та вітчизняного досвіду показує, що нейромережі активно використовуються для прогнозування фінансових ринків, і українським ринком, де цей підхід ще набирає обертів. Враховуючи швидкий розвиток технологій та зростаючий інтерес до машинного навчання в Україні, можна очікувати значного прогресу в цій області в найближчому майбутньому.

1.2 Аналіз сучасного стану програмно-апаратного забезпечення

Аналіз сучасного стану програмно-апаратного забезпечення є критично важливим для розуміння можливостей та обмежень, що стоять перед використанням нейромережевих технологій для прогнозування економічних показників дорогоцінних металів на фондовому ринку України.

1.2.1 Моделі прогнозу з використанням штучної нейронної мережі

У статті [] розглядається підход ШНМ шляхом розробки нелінійних прогнозних моделей для аналізу. Хаотична природа вхідних даних приводить до необхідності використання нейронних мереж.

Перевага ШНМ перед іншими методами прогнозування погоди полягає в тому, що ШНМ мінімізує помилку за допомогою різних алгоритмів і дає нам прогнозоване значення, яке майже дорівнює фактичному значенню. Такі мережа моделюється на основі новіших даних, щоб дізнатися тенденцію в майбутньому.

Наприклад, у прогнозуванні температури слід розрізняти час, коли прогноз йде вперед температура на одну годину вперед або мінімальна і максимальна температура певного дня. Автори провели порівняння багатошарових перцептронних мереж (MLP), Elman Рекурентна нейронна мережа (ERNN), мережа радіальних базових функцій (RBFN) і модель Хопфілда (HFM) і ансамблі цих

мереж. Автори запропонували для мережі MLP один прихований шар і 72 нейрони і 2 прихованих шари з 180 нейронами для RBFN як оптимальної архітектури. Для активації була використана сигмоїда. У RBFN часто використовують активаційну функцію Гауса. Робота, описана Санджаєм Матуром [11], зосереджена на прогнозуванні за допомогою ШНМ прямого зв'язку з навчанням зворотного поширення. За 15-тижневий період помилка становила менше 3%. Навчання здійснюється шляхом зворотного ходу.

1.2.3 Огляд новітніх технологічних досягнень у сфері штучного інтелекту та їх застосування

Одним з найважливіших досягнень у сфері штучного інтелекту є розвиток технологій глибокого навчання (Deep Learning). Ці технології використовують шаруваті структури нейронних мереж для обробки великих обсягів даних, що дозволяє виявляти складні закономірності та взаємозв'язки. У контексті фінансового ринку, глибоке навчання може використовуватися для аналізу цінових рухів, виявлення ринкових трендів та прогнозування економічних показників [20].

Використання штучних нейронних мереж (ШНМ) у сфері фінансів і, зокрема, для аналізу ринку дорогоцінних металів, є одним з передових методів в області штучного інтелекту. ШНМ імітують процеси обробки інформації, що відбуваються в людському мозку, що дозволяє їм ефективно аналізувати складні шаблони та здійснювати прогнозування.

Ці моделі особливо корисні для обробки часових рядів – типу даних, який часто зустрічається на фінансових ринках, де ціни дорогоцінних металів змінюються протягом часу. Штучні нейронні мережі здатні аналізувати історичні дані, виявляючи патерни та тенденції, які можуть бути неочевидними для традиційних аналітичних методів.

Однією з ключових переваг ШНМ є їх здатність до «навчання» – вони можуть адаптуватися та оновлювати свої прогнози на основі нових даних, що надходять. Це означає, що моделі не просто покладаються на історичні дані, але також здатні

реагувати на поточні ринкові умови, що робить їх надзвичайно цінними для прогнозування в динамічному фінансовому середовищі. Також ШНМ можуть бути налаштовані та оптимізовані з урахуванням специфіки конкретного ринку або навіть конкретного виду дорогоцінних металів. Це дозволяє створювати спеціалізовані моделі, які можуть враховувати унікальні аспекти та умови, властиві ринку дорогоцінних металів [21].

Завдяки своїй здатності до обробки великих обсягів даних та адаптації до змінних умов, штучні нейронні мережі відкривають нові перспективи для прогнозування на ринку дорогоцінних металів. Вони допомагають аналітикам та інвесторам краще розуміти ринкові процеси та розробляти більш ефективні стратегії інвестування.

Розвиток методів машинного навчання відкрив нові горизонти в аналізі фінансових даних, зокрема у контексті прогнозування на ринку дорогоцінних металів. Сучасні алгоритми, такі як випадковий ліс, градієнтний бустинг та опорні векторні машини, значно розширюють можливості аналізу, порівняно з традиційними статистичними методами.

Наприклад, алгоритм випадкового лісу використовує ансамбль дерев рішень для здійснення класифікації або регресії. Кожне дерево рішень навчається на підмножині даних, а результати об'єднуються для отримання більш точного прогнозу. Цей метод ефективний у зменшенні перенавчання та підвищенні точності прогнозів, особливо коли працює з великими та складними наборами даних.

Передові методи машинного навчання відкривають нові можливості для фінансових аналітиків, зокрема у сфері прогнозування цін на дорогоцінні метали. Вони дозволяють більш точно ідентифікувати закономірності та тренди, які можуть бути неявними при використанні більш простих аналітичних моделей.

Технології великих даних (Big Data) забезпечують можливість обробки та аналізу величезних обсягів неструктурованої інформації, що є важливим для розуміння глобальних фінансових ринків. Це дозволяє інтегрувати та аналізувати дані з різноманітних джерел, таких як новинні стрічки, фінансові звіти, макроекономічні показники та соціальні медіа.

Технології великих даних (Big Data) дозволяють обробляти та аналізувати величезні обсяги даних, які надходять із різних джерел, і враховувати складну мережу факторів, що впливають на ринок. Однією з основних переваг використання великих даних є здатність обробляти не тільки структуровану, але й неструктуровану інформацію. Це включає текстові дані з новинних стрічок, соціальних медіа, фінансових звітів, блогів експертів, та інших джерел. Ця інформація часто містить важливі вказівки на ринкові настрої, очікування інвесторів, та потенційні політичні чи економічні зміни [22].

Технології великих даних дозволяють інтегрувати інформацію з різноманітних джерел, що є ключовим для всебічного аналізу ринку. Це може включати макроекономічні показники, такі як ставки інфляції, ВВП, валютні курси, а також більш специфічні фінансові дані, такі як об'єми торгів, ціни на дорогоцінні метали, та показники попиту та пропозиції. Застосування передових методів аналізу даних, таких як машинне навчання та штучний інтелект, у поєднанні з великими даними, дозволяє виявляти складні закономірності та тенденції, які можуть бути недоступні при традиційних методах аналізу. Це може включати прогнозування майбутніх цінових рухів, аналіз ризиків та ідентифікацію нових інвестиційних можливостей.

Для фінансового ринку, особливо у сфері торгівлі дорогоцінними металами, застосування великих даних може значно підвищити точність та надійність прогнозів. Воно дозволяє фінансовим аналітикам та інвесторам краще розуміти глобальні ринкові умови, а також реагувати на швидкі зміни на ринку.

Застосування ШІ на фондовому ринку України може стати важливим чинником розвитку та модернізації ринку. Це може не тільки підвищити ефективність торгівлі та інвестицій, але й сприяти більшій прозорості та стабільності ринку. Застосування передових технологій дозволить Україні інтегруватися в глобальну фінансову систему, надаючи доступ до новітніх інструментів аналізу та прогнозування.

Останні досягнення у сфері штучного інтелекту та машинного навчання відкривають нові горизонти для прогнозування ринку дорогоцінних металів. Ці технології можуть допомогти українським фінансовим аналітикам та трейдерам у

прийнятті більш обґрунтованих та ефективних рішень, підвищуючи точність прогнозів та оптимізуючи інвестиційні стратегії.

Висновки до розділу 1

У рамках даного розділу було здійснено всебічний аналіз сучасного стану використання нейромережевих технологій для прогнозування економічних показників дорогоцінних металів. Проаналізовано еволюцію методів прогнозування, починаючи від класичних економетричних моделей до сучасних машинно-навчальних технік, з акцентом на використанні глибоких нейронних мереж. Особливу увагу приділено впливу глобальних економічних криз та геополітичних подій на ціни дорогоцінних металів.

Дослідження наголошують на важливості впровадження нейромереж для аналізу складних шаблонів у фінансових даних, зокрема в контексті виявлення цінових бульбашок та аналізу мультифрактальних характеристик. Відзначено, що нейромережі ефективні у роботі з великими обсягами даних і здатні швидко адаптуватися до ринкових змін, що є критично важливим для прогнозування на динамічному фондовому ринку України.

Також висвітлено ряд невирішених проблем і викликів, які потребують подальших досліджень. Це включає в себе проблеми з якістю даних, волатильність ринку, інтеграцію зовнішніх факторів та обмеження, пов'язані з машинним навчанням, як-от перенавчання моделей, вибір архітектури та налаштування гіперпараметрів.

Зазначено, що зарубіжний досвід у використанні нейромереж для прогнозування цін на дорогоцінні метали відзначається високим рівнем інновацій та комплексним підходом до аналізу даних. В Україні, незважаючи на існуючий потенціал для розвитку цих технологій, сектор все ще знаходиться на ранніх етапах інтеграції та впровадження нейромереж, що обумовлено обмеженнями доступу до даних, ресурсами та кваліфікацією фахівців. Висновок полягає у важливості розвитку та інтеграції нейромережевих технологій в Україні, а також необхідності

подолання існуючих обмежень для підвищення ефективності та точності прогнозів на фондовому ринку дорогоцінних металів

2 ОБҐРУНТУВАННЯ ЕКСПЕРИМЕНТАЛЬНОГО МЕТОДУ ДОСЛІДЖЕННЯ

2.1 Теоретичний аналіз методів прогнозування

Теоретичний аналіз методів прогнозування з використанням нейромережевих технологій відіграє ключову роль у короткостроковому прогнозуванні економічних показників дорогоцінних металів на фондовому ринку України. Цей підхід включає в себе використання складних алгоритмів штучного інтелекту для аналізу інформації та виявлення закономірностей у великих обсягах даних. Використання нейромереж дозволяє робити точніші прогнози, що є важливим для інвесторів і трейдерів на ринку дорогоцінних металів. Ці технології сприяють кращому розумінню ринкових трендів і допомагають у прийнятті обґрунтованих інвестиційних рішень. Завдяки своїй здатності до обробки великих масивів даних і виявлення складних взаємозв'язків, нейронні мережі стають незамінним інструментом у фінансовому аналізі.

На відміну від цього, традиційні методи прогнозування, такі як статистичний аналіз, авторегресійні моделі та інші економетричні підходи, базуються на більш лінійному та прогнозованому вивченні даних. Хоча ці методи залишаються ефективними для певних типів аналізу, вони часто не можуть врахувати складність та волатильність сучасних фінансових ринків так ефективно, як нейронні мережі.

2.1.1 Аналіз традиційних методів

Традиційні методи прогнозування економічних показників займають важливе місце в аналітиці фондових ринків, зокрема у прогнозуванні цін на дорогоцінні метали. Вони включають в себе різноманітні підходи, які базуються на статистичному аналізі та економетричних моделях. До традиційних методів відносяться:

Статистичні методи

Ці методи включають базові інструменти аналізу, такі як рухоме середнє (moving average) та експоненційне згладжування. Рухоме середнє допомагає

згладжувати короткострокові коливання і виділити тренди, тоді як експоненційне згладжування надає більшої ваги недавнім даним, роблячи прогнози більш чутливими до останніх змін у ринковій динаміці [23].

2.1.2 Аналіз нейромережевих методів

Аналіз нейромережевих методів охоплює розгляд штучних нейронних мереж як інструменту для моделювання та прогнозування складних шаблонів у даних. Ці методи, натхненні структурою та функціями мозку, використовуються у широкому спектрі застосувань, від фінансового прогнозування до обробки природної мови.

До популярних нейромережевих методів відносяться:

Штучні нейронні мережі (англ. ANN)

Штучні нейронні мережі (ШНМ) використовуються для моделювання складних шаблонів та взаємодій у даних, здатні до самонавчання та адаптації, що робить їх важливим інструментом у багатьох сферах, зокрема у фінансовому аналізі.

ШНМ складаються з вузлів, відомих як «нейрони», організованих у шари: вхідний шар, один або кілька прихованих шарів та вихідний шар. Кожен нейрон у шарі з'єднаний з нейронами наступного шару через «ваги», які регулюють силу сигналу між нейронами.

Навчання в нейронній мережі відбувається шляхом коригування цих ваг на основі набору тренувальних даних. Цей процес включає розрахунок помилки між вихідними даними мережі та фактичними даними, а потім коригування ваг таким чином, щоб зменшити цю помилку.

У контексті економічних показників дорогоцінних металів, ШНМ використовуються для прогнозування цін, аналізу ринкових тенденцій та оцінки ризиків [32]. Ці системи здатні аналізувати великі набори даних, які можуть включати історичні ціни, обсяги торгів, економічні індикатори та навіть текстові новини, щоб виявити складні шаблони та залежності.

На рис. 2.4 представлено алгоритм роботи ШНМ для прогнозування економічних показників.

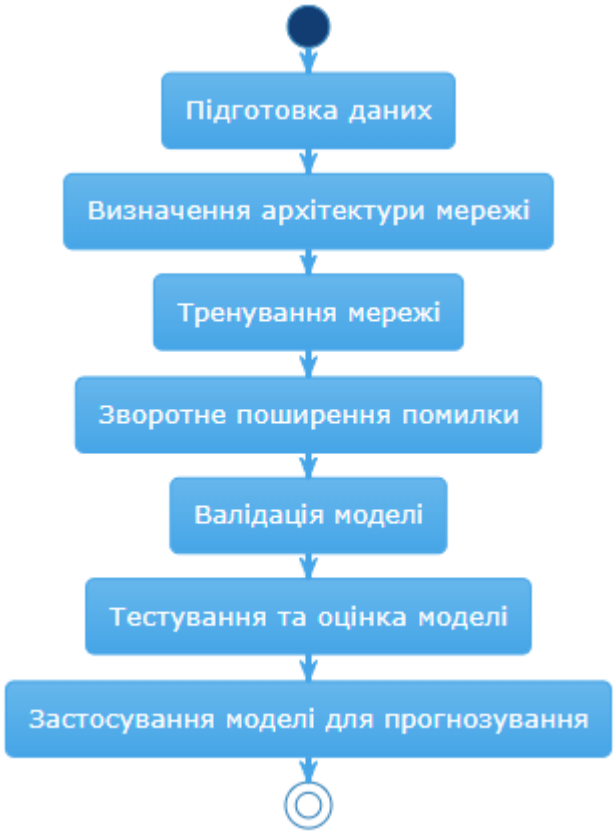

Рисунок 2.4 – Алгоритм прогнозування за допомогою ШНМ

Процес прогнозування за допомогою ШНМ розпочинається з ретельної підготовки даних. Це включає збір історичних даних, таких як ціни на дорогоцінні метали, обсяги торгів, економічні індикатори та інші важливі параметри. Важливо очистити ці дані від помилок і неповноти та перетворити їх у формат, придатний для обробки нейронною мережею. Наступний крок полягає у визначенні архітектури ШНМ. Це включає вибір кількості шарів, нейронів в кожному шарі та типів функцій активації, що залежить від складності даних та специфіки задачі прогнозування.

Ключовим етапом є тренування мережі, під час якого вхідні дані проходять через мережу, порівнюються з фактичними значеннями, та ваги нейронів коригуються для зменшення помилки. Цей процес навчання включає зворотне поширення помилки, де ваги нейронів коригуються від вихідного шару до вхідного, з метою оптимізації загальної ефективності мережі. Після тренування мережа проходить через фазу валідації, використовуючи набір даних, який не брав участь у тренуванні, для перевірки точності та узагальнювальної здатності моделі. Потім виконується тестування та оцінка моделі, щоб оцінити її здатність прогнозувати майбутні ціни та інші економічні показники дорогоцінних металів.

На заключному етапі, після успішного тестування та валідації, модель використовується для реального прогнозування, де вона аналізує актуальні дані та робить передбачення щодо майбутніх тенденцій цін на дорогоцінні метали. Використання ШНМ у такому контексті дозволяє з високою точністю прогнозувати ринкові тенденції, використовуючи складні взаємозв'язки у великих наборах даних.

ШНМ є потужним інструментом в аналізі даних, але, як і будь-яка технологія, мають свої переваги та недоліки. До переваг можна віднести [33]:

– здатність до обробки великих обсягів даних. ШНМ ефективно обробляють великі набори даних, ідентифікуючи складні шаблони та зв'язки, які можуть бути непомітні для традиційних аналітичних методів;

– самонавчання та адаптивність. ШНМ мають здатність навчатися без прямого програмування, адаптуючись до нових даних та умов без втручання людини;

– узагальнююча здатність. Ці системи можуть узагальнювати від зразків тренувальних даних до нових, невідомих даних, що дозволяє їм ефективно прогнозувати майбутні тенденції;

– гнучкість застосування. Можуть застосовуватися до різноманітних задач, від прогнозування фінансових показників до розпізнавання зображень і обробки природної мови.

Недоліки ШНМ:

– складність інтерпретації. Часто вважаються «чорними скриньками», оскільки інтерпретація того, як мережа прийшла до конкретного висновку, може бути складною;

– високі вимоги до обчислювальних ресурсів. Тренування ШНМ, особливо глибоких мереж, може вимагати значних обчислювальних ресурсів та часу;

– ризик перенавчання. ШНМ можуть «перенавчитися» на тренувальних даних, занадто точно адаптуючись до них, що знижує їх здатність узагальнювати на нових даних [34];

– потреба у великій кількості даних для навчання. Ефективне навчання ШНМ вимагає великих наборів даних, що не завжди доступні, особливо в специфічних або нових областях досліджень.

Отже, ШНМ є високоефективним інструментом для аналізу та прогнозування складних даних, зокрема у фінансовому секторі для прогнозування цін на дорогоцінні метали. Вони вирізняються здатністю обробляти великі обсяги інформації, виявляючи складні зв'язки та шаблони, недоступні для традиційних аналітичних методів. ШНМ також характеризуються гнучкістю застосування, самонавчанням та узагальнювальною здатністю. Проте, вони вимагають значних обчислювальних ресурсів, можуть бути складними для інтерпретації та мають ризик перенавчання. Незважаючи на ці виклики, ШНМ продовжують бути одним із ключових інструментів у сучасному аналізі даних, пропонуючи потужні можливості для прогнозування та аналізу в широкому спектрі областей.

Конволюційні нейронні мережі (англ. CNN)

Конволюційні нейронні мережі (КНМ) є спеціалізованим типом штучних нейронних мереж, оптимізованих для аналізу та обробки даних, що мають сітчасту структуру, як-от зображення або часові ряди. Вони відомі своєю здатністю ефективно виявляти та інтерпретувати складні шаблони та особливості у великих наборах даних. Вони складаються з декількох шарів, кожен з яких виконує певну функцію. На початковому етапі конволюційні шари застосовують різні фільтри до вхідних даних, виявляючи основні характеристики, такі як краї та кути в даних. Ключовим аспектом є здатність цих шарів до виявлення локальних особливостей незалежно від їхнього розташування у вхідних даних.

Після виявлення основних характеристик дані проходять через шари пулінгу, які зменшують розмірність даних, вибірково зберігаючи важливу інформацію та відкидаючи менш важливу. Це сприяє зменшенню обсягу обчислень та покращує ефективність мережі.

У контексті фінансового аналізу, зокрема при прогнозуванні економічних показників дорогоцінних металів, КНМ можуть бути адаптовані для обробки часових рядів. Це досягається шляхом використання конволюційних шарів для

виявлення важливих трендів та шаблонів у історичних цінових даних. КНМ здатні виявляти складні взаємозв'язки та залежності, які можуть бути неочевидними при традиційному аналізі, що робить їх потужним інструментом для прогнозування ринкових трендів [35].

На рис. 2.5 зображена діаграма діяльності, що ілюструє алгоритм роботи КНМ для прогнозування економічних показників дорогоцінних металів.

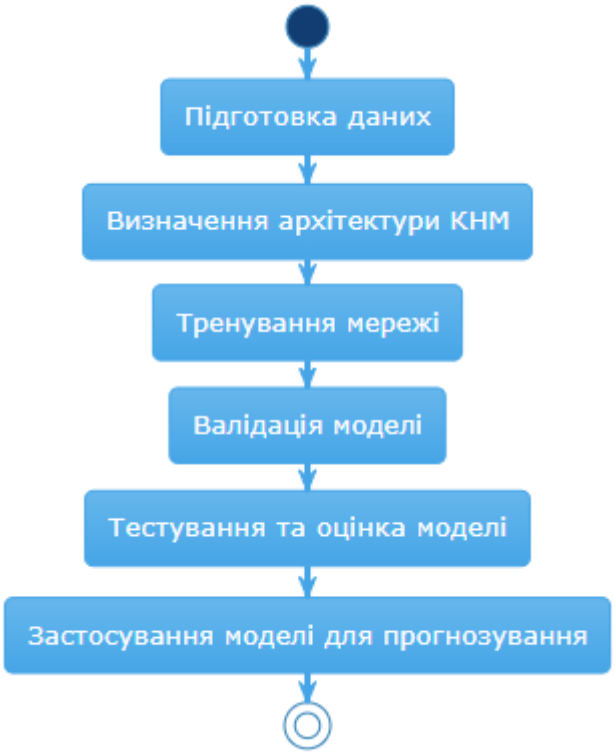

Рисунок 2.5 – Алгоритм прогнозування за допомогою КНМ

Алгоритм роботи КНМ для прогнозування економічних показників дорогоцінних металів починається з підготовки даних, яка включає збір історичних цін на дорогоцінні метали, обсягів торгів, економічних індикаторів та інших релевантних даних. Ці дані потім очищаються та перетворюються для подальшої обробки нейронною мережею.

Далі йде визначення архітектури КНМ, що включає вибір кількості та типів шарів, а також функцій активації. Ключовими компонентами є конволюційні шари, які застосовують різні фільтри до вхідних даних для виявлення важливих особливостей, та шари пулінгу, які здійснюють пониження вимірності даних, зберігаючи при цьому важливу інформацію.

Після визначення архітектури КНМ відбувається процес навчання, де мережа тренується на підготовлених даних. Процес навчання включає пропускання даних через мережу, порівняння виводу мережі з фактичними значеннями та коригування ваг шляхом зворотного поширення помилки для зменшення розбіжності між прогнозованими та реальними значеннями.

Після тренування мережа проходить валідацію, де вона тестується на окремому наборі даних, щоб перевірити її здатність узагальнювати і робити точні прогнози. Остаточне тестування та оцінка моделі виконуються для оцінки її ефективності у прогнозуванні майбутніх цін або інших економічних показників дорогоцінних металів.

Після успішного тестування модель може бути використана для реального прогнозування, де вона аналізує актуальні дані та робить передбачення щодо майбутніх тенденцій цін на дорогоцінні метали. Використання КНМ в такому контексті відкриває нові можливості для розуміння та прогнозування ринкових тенденцій.

КНМ є важливим інструментом у галузі штучного інтелекту і машинного навчання, проте, як і будь-який інший метод, вони мають свої переваги та недоліки.

До переваг КНМ відносяться:

– виявлення особливостей на різних рівнях [36]. КНМ ефективно виявляють та аналізують особливості на різних рівнях складності, що робить їх ідеальними для обробки зображень та інших складних датасетів;

– зменшення попередньої обробки даних. Завдяки своїй структурі, КНМ потребують менше попередньої обробки даних порівняно з іншими типами нейронних мереж, оскільки вони здатні самі виявляти важливі особливості;

– застосування в різних сферах. Хоча КНМ були розроблені для обробки зображень, вони також показують хороші результати в інших задачах, таких як аналіз часових рядів або обробка природної мови;

– ефективність у великих датасетах. КНМ ефективно працюють з великими наборами даних, здатні виявляти складні взаємозв'язки та шаблони, що є критично важливим для аналізу великих обсягів інформації.

До недоліків КНМ слід віднести:

– високі вимоги до обчислювальних ресурсів. Тренування КНМ часто вимагає значних обчислювальних ресурсів, особливо для великих архітектур мереж і датасетів;

– ризик перенавчання. Як і інші нейронні мережі, КНМ схильні до перенавчання, особливо коли датасет для тренування є обмеженим або недостатньо різноманітним;

– обмежена інтерпретація. Результати, отримані з КНМ, можуть бути важко інтерпретувати, оскільки їх внутрішній процес роботи може бути складним для розуміння;

– загальні обмеження ШНМ. КНМ ділять загальні недоліки ШНМ, такі як вразливість до шуму в даних та потреба у великій кількості даних для ефективного навчання [37].

Отже, КНМ є потужним інструментом в області машинного навчання, який знайшов широке застосування у багатьох галузях, включаючи обробку зображень, аналіз часових рядів та обробку природної мови. Їх ключовою перевагою є здатність автоматично виявляти та аналізувати особливості на різних рівнях складності у великих датасетах. КНМ зменшують потребу в попередній обробці даних та ефективно працюють із складними даними. Проте, ці мережі вимагають значних обчислювальних ресурсів для тренування, схильні до перенавчання та часто важко інтерпретувати їх результати. Незважаючи на ці виклики, КНМ продовжують бути важливими в області штучного інтелекту, пропонуючи значні можливості для аналізу та прогнозування в різних галузях.

Рекурентні нейронні мережі (англ. RNN)

Рекурентні нейронні мережі (РНМ) – це тип штучних нейронних мереж, спеціалізований на обробці послідовностей даних, таких як часові ряди, мова або будь-які інші дані, де важливий контекстуальний зв'язок між послідовними елементами. Це досягається завдяки тому, що РНМ мають внутрішню пам'ять, що дозволяє їм зберігати інформацію про попередні елементи у послідовності, використовуючи її для визначення виводу на наступних кроках.

РНМ включають повторювані модулі, кожен з яких приймає вхідні дані та внутрішній стан з попереднього кроку, об'єднуючи їх для виводу поточного стану та передачі наступному модулю. Така структура робить РНМ ідеальними для аналізу даних, де контекст і послідовність мають велике значення.

У контексті прогнозування економічних показників дорогоцінних металів РНМ можуть бути використані для аналізу часових рядів цін на метали. Вони здатні розпізнавати складні шаблони та залежності в історичних цінових даних, враховуючи тренди та циклічні зміни. Це дозволяє прогнозувати майбутні ціни або інші фінансові показники на основі історичних даних, беручи до уваги як недавні, так і більш давні цінові тренди.

РНМ можуть бути застосовані для прогнозування волатильності цін на дорогоцінні метали, виявлення можливих трендів у цінових рухах, а також для аналізу зв'язків між цінами на дорогоцінні метали та іншими економічними індикаторами. Це робить їх цінним інструментом для інвесторів та фінансових аналітиків, які прагнуть розуміти та прогнозувати ринкові умови [38].

Рис. 2.6 відображає алгоритм роботи РНМ для прогнозування економічних показників.

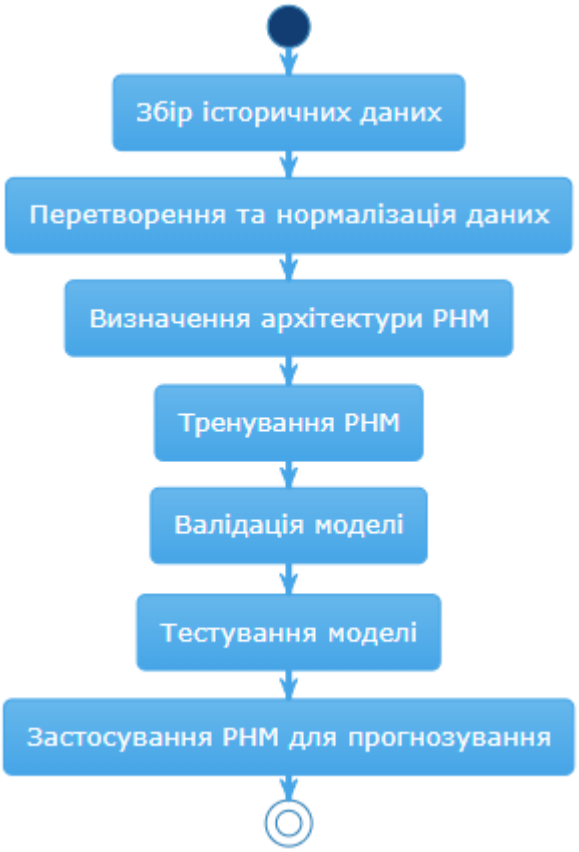

Рисунок 2.6– Алгоритм прогнозування за допомогою РНМ

Алгоритм роботи РНМ розпочинається з підготовки і аналізу історичних даних. Спочатку збираються історичні дані, такі як ціни на дорогоцінні метали, обсяги торгів, а також важливі економічні індикатори, які можуть впливати на ціни металів. Ці дані потім перетворюються та нормалізуються для подальшої обробки РНМ. Після підготовки даних визначається архітектура РНМ, зокрема кількість шарів та нейронів у мережі. У контексті фінансового прогнозування часто використовуються такі варіанти РНМ, як LSTM (Long Short-Term Memory) або GRU (Gated Recurrent Units), які ефективно обробляють довгі послідовності даних і здатні зберігати інформацію на тривалі часові проміжки.

Наступним кроком є процес тренування РНМ. Мережа тренується на історичних даних, де модель використовує вхідні дані (наприклад, минулі ціни) для прогнозування вихідних значень (наприклад, майбутніх цін). Процес тренування включає коригування ваг нейронів в мережі з метою мінімізації розбіжності між прогнозованими та фактичними значеннями.

Після тренування РНМ проводиться валідація та тестування моделі на окремих датасетах, щоб переконатися у точності та надійності прогнозів. Модель оцінюється за такими параметрами, як точність прогнозування, здатність до узагальнення та відповідність реальним ринковим умовам.

Завершальний етап полягає у застосуванні тренованої РНМ для реального прогнозування економічних показників дорогоцінних металів. Модель аналізує поточні ринкові дані та генерує прогнози, які можуть бути використані інвесторами та аналітиками для прийняття обґрунтованих інвестиційних рішень [39].

Переваги РНМ складаються із:

– здатність обробляти послідовності даних. Ефективно обробляють послідовності даних, такі як часові ряди або мовні послідовності, зберігаючи інформацію про попередні елементи;

– пам'ять та контекстуальне розуміння. Здатні зберігати інформацію про попередні дані в послідовності, що надає їм контекстуальне розуміння і допомагає у виявленні трендів та шаблонів у даних;

– гнучкість у застосуванні. Можуть бути застосовані до широкого спектру задач, включаючи прогнозування часових рядів, обробку природної мови, розпізнавання мовлення та інші;

– ефективність у виявленні залежностей у даних. Ефективно виявляють та аналізують довготривалі залежності у даних, що є важливим для точного прогнозування.

До недоліки РНМ відносяться:

– складність тренування. Складні у тренуванні, особливо через проблему зникаючих або вибухаючих градієнтів, які ускладнюють процес навчання;

– великі обчислювальні вимоги. Вимагають значних обчислювальних ресурсів для ефективного тренування та виконання, особливо для великих датасетів;

– ризик перенавчання. Як і інші моделі машинного навчання, РНМ схильні до перенавчання, особливо якщо доступний обмежений обсяг даних;

– обмеження у унтерпретації результатів. Результати, отримані від РНМ, можуть бути складними для інтерпретації, оскільки внутрішні процеси мережі не завжди легко зрозуміти.

Отже, РНМ є важливим інструментом в сфері машинного навчання, зокрема для обробки та аналізу послідовностей даних. Завдяки своїй унікальній архітектурі, яка дозволяє зберігати інформацію про попередні дані, РНМ ефективно аналізують часові ряди, мовні дані та інші послідовності, виявляючи складні шаблони та залежності. Це робить їх особливо корисними для прогнозування економічних показників, включаючи ціни на дорогоцінні метали. Проте, вони також мають свої обмеження, включаючи складнощі з тренуванням, високі обчислювальні вимоги та складнощі з інтерпретацією результатів. Незважаючи на ці виклики, РНМ залишаються незамінним інструментом для глибокого аналізу послідовностей даних та прогнозування трендів у різних галузях [39].

Швидкі ліси (англ. Random Forests)

Метод «Швидкі ліси» (ШЛ) оснований на нейронних мережах та використовує ансамбль дерев рішень для класифікації, регресії та інших завдань. Цей метод базується на ідеї, що комбінація прогнозів від множини дерев рішень призводить до більш точних та стабільних результатів, ніж прогноз від одного дерева.

У ШЛ кожне дерево рішень тренується на випадково відібраному піднаборі даних, що дозволяє зменшити ризик перенавчання і підвищити здатність моделі узагальнювати на нових даних. Цей процес включає випадковий вибір підмножини ознак для кожного розщеплення у дереві рішень, що додатково збільшує різноманітність у ансамблі.

У рамках прогнозування економічних показників дорогоцінних металів, ШД аналізують історичні цінові дані, обсяги торгів, а також можуть включати інші відповідні економічні індикатори. Алгоритм здатний ідентифікувати складні шаблони та взаємозв'язки у даних, що дозволяє робити прогнози щодо майбутніх цінових трендів.

Ключовою перевагою ШЛ є їхня стійкість до перенавчання та здатність ефективно обробляти великі та складні набори даних. Це робить їх ідеальними для застосування в області фінансового аналізу, де важливо точно та надійно оцінювати ризики та можливості.

На рис. 2.7 представлено алгоритм прогнозування даного методу.

Рисунок 2.7– Алгоритм прогнозування за допомогою ШЛ

Алгоритм ШЛ для прогнозування економічних показників дорогоцінних металів розпочинається зі збору та підготовки даних. Це включає історичні ціни на метали, обсяги торгів, а також різноманітні економічні індикатори, які можуть впливати на ринок дорогоцінних металів. Дані очищаються та перетворюються у формат, придатний для аналізу.

Далі відбувається створення ансамблю дерев рішень. У ШЛ кожне дерево рішень будується незалежно на основі випадково відібраних підмножин навчальних даних. Це відбувається за допомогою методу бутстрепінгу, де для кожного дерева створюється власний тренувальний набір даних шляхом випадкового вибору зразків із загального датасету.

Під час побудови кожного дерева в алгоритмі ШЛ використовується техніка випадкового вибору ознак. Це означає, що на кожному етапі розщеплення в дереві рішень вибирається підмножина ознак, і серед них шукається найкращий критерій розщеплення.

Після побудови ансамблю дерев, для прогнозування нових даних використовується метод голосування або усереднення. У випадку класифікації кожне дерево вносить свій «голос» за клас, а клас з найбільшою кількістю голосів вибирається як результат. У випадку регресії результати від усіх дерев усереднюються.

У контексті прогнозування цін на дорогоцінні метали, ШЛ можуть використовуватися для аналізу шаблонів у історичних цінових даних, ідентифікації зв'язків між цінами та іншими економічними показниками. Це дозволяє робити прогнози щодо майбутніх цінових трендів, забезпечуючи інвесторам цінну інформацію для прийняття обґрунтованих інвестиційних рішень [40].

Метод ШЛ має ряд переваг та недоліків, які роблять його важливим інструментом, але з певними обмеженнями у використанні. До переваг зокрема відносяться:

– стійкість до перенавчання. На відміну від індивідуальних дерев рішень, «Швидкі ліси» менш схильні до перенавчання завдяки використанню великої кількості дерев і методу бутстрепінгу;

– висока точність прогнозування. Ансамблевий підхід, що включає багато дерев, зазвичай дає більш точні результати, ніж окремі дерева рішень, особливо в складних наборах даних;

– ефективність у обробці великих датасетів. «Швидкі ліси» добре справляються з великими обсягами даних та можуть ефективно обробляти датасети з великою кількістю ознак;

– здатність обробляти неповні дані. Цей метод може ефективно працювати з неповними даними, автоматично обробляючи відсутні значення без потреби в додатковій попередній обробці.

До недоліків ШЛ відносяться:

– обчислювальні витрати. Побудова великої кількості дерев може бути обчислювально витратною, особливо для великих датасетів або датасетів з великою кількістю ознак;

– складність інтерпретації. На відміну від індивідуального дерева рішень, ансамблеві моделі, можуть бути складними для інтерпретації та розуміння прийняття рішень;

– неоптимальність для деяких завдань [41]. Хоча ШЛ є універсальними, вони можуть бути не найкращим вибором для деяких специфічних завдань, особливо там, де потрібна висока швидкість прогнозування або де важлива модель із високою інтерпретаційною здатністю.

Отже, метод ШЛ є потужним інструментом машинного навчання, який використовує ансамбль дерев рішень для підвищення точності та надійності прогнозування. Цей метод ефективний у обробці великих та складних датасетів, включаючи ті, що містять неповні дані, та відомий своєю стійкістю до перенавчання. ШЛ здатні обробляти велику кількість ознак і вирішувати як задачі класифікації, так і регресії, роблячи їх важливим інструментом у різних областях, включаючи прогнозування економічних показників дорогоцінних металів. Однак, вони можуть бути витратними з точки зору обчислень, важкими для інтерпретації та потребують часу на тренування, особливо при великих обсягах даних. Незважаючи на ці виклики, ШЛ залишаються одним із найбільш популярних та ефективних алгоритмів машинного навчання, використовуваних сьогодні.

2.3 Метрики оцінки точності в прогнозуванні дорогоцінних металів

Точність прогнозування економічних показників дорогоцінних металів на фондовому ринку є ключовим аспектом для інвесторів та аналітиків. Визначення точності цих прогнозів залежить від застосування спеціалізованих метрик, які дозволяють оцінити ефективність та надійність використаних прогнозних моделей. Ці метрики допомагають зрозуміти, наскільки добре модель передбачає реальні ринкові ціни, та дають можливість порівняти різні моделі між собою. Важливо, що вибір відповідної метрики може залежати від специфіки завдання та даних. Нижче

перераховані основні метрики, які використовуються для оцінки точності прогнозування економічних показників дорогоцінних металів:

– середньоквадратична помилка (англ. Mean squared error). Вимірює середнє значення квадратів помилок, тобто, різниць між прогнозованими та фактичними значеннями. Ця метрика є дуже поширеною, оскільки вона карає за великі помилки більше, ніж за малі, що робить її особливо корисною в контекстах, де великі помилки мають значний вплив;

– середня абсолютна помилка (англ. Mean absolute error). Обчислює середнє абсолютне відхилення прогнозованих значень від фактичних. На відміну від MSE, MAE не вводить квадратичного «карального» чинника, роблячи її менш чутливою до великих відхилень [42];

– коефіцієнт детермінації (R^2 або R-Squared). Вимірює, яка частка загальної варіативності змінної може бути пояснена за допомогою моделі. Ця метрика є корисною для визначення «пояснювальної сили» моделі.

– корінь із середньоквадратичної помилки (англ. root mean squared error). Вимірює стандартне відхилення помилок прогнозування. Вона аналогічна до MSE, але масштабується назад до одиниць вимірювання змінної, роблячи її більш інтерпретованою [43];

– середня відсоткова помилка (англ. Mean percentage error). Дозволяє оцінити помилку прогнозування як відсоток від фактичних значень, що може бути корисним для оцінки відносної помилки [44];

– середня абсолютна відсоткова помилка (англ. Mean absolute percentage error). Є аналогом MAE, але вимірюється як відсоток, що робить її корисною для порівняння точності прогнозів між датасетами різного масштабу [45].

Оцінюючи точність прогнозування, важливо врахувати, що різні метрики можуть надавати різну перспективу на ефективність моделі. Середньоквадратична помилка та її корінь допомагають зрозуміти загальну величину помилок, але можуть бути більш чутливими до великих відхилень. Середня абсолютна помилка та середня абсолютна відсоткова помилка надають інформацію про середню помилку у більш зрозумілому вигляді. Коефіцієнт детермінації вказує на відсоток

варіативності, який пояснюється моделлю. Використання цих метрик разом може дати повніше уявлення про точність і надійність прогнозування, дозволяючи ефективно оцінити й порівняти різні прогнозні моделі для ринку дорогоцінних металів.

2.3.2 Оцінка впливу факторів на точність прогнозу

Точність прогнозування економічних показників дорогоцінних металів на фондовому ринку залежить від багатьох змінних, які впливають на ринкову динаміку. Розуміння цих факторів є важливим для створення точних та надійних прогнозів. Від глобальних економічних тенденцій до специфічних ринкових умов, кожен аспект може мати значний вплив на рухи цін. Політичні події, зміни валютних курсів та технологічні інновації також відіграють ключову роль у визначенні цін на дорогоцінні метали. Основні фактори, які впливають на точність прогнозування показників дорогоцінних металів фондового ринку складають:

Глобальні економічні умови

Геополітичні події та політична нестабільність

Валютні ринки

Ринковий попит та пропозиція

Технологічний прогрес і інновації

Сезонність та спекулятивні чинники

2.4 Обґрунтування вибору методу прогнозування

Перед розробкою системи прогнозування економічних показників дорогоцінних металів фондового ринку важливо розглянути декілька перспективних методів прогнозування. Ключовими кандидатами є ШНМ, КНМ, РНМ, та ШЛ. Кожен з цих методів має свої особливості, переваги та обмеження, залежно від специфіки даних та цілей прогнозування. ШНМ відомі своєю гнучкістю та потужністю, КНМ ефективні у виявленні шаблонів у великих датасетах, РНМ використовуються для аналізу часових рядів, а Швидкі ліси надають високу точність

та легку інтерпретацію. Для більш об'єктивного вибору методу важливо порівняти ці методи за кількома ключовими параметрами, які представлені у табл.2.1.

Таблиця 2.1 – Порівняльний аналіз методів прогнозування

Параметр	ШНМ	КНМ	РНМ	ШЛ
Складність реалізації	Висока	Висока	Висока	Середня
Вимоги до обчислювальних ресурсів	Високі	Високі	Високі	Середні
Інтерпретованість результатів	Низька	Низька	Низька	Висока
Стійкість до перенавчання	Середня	Середня	Середня	Висока
Ефективність у великих датасетах	Висока	Висока	Висока	Висока
Універсальність застосування	Висока	Середня	Середня	Висока

Порівняння методів прогнозування показує, що метод ШД має ряд переваг для розробки системи прогнозування економічних показників дорогоцінних металів фондового ринку. По-перше, ШД відрізняються високою стійкістю до перенавчання, що є критично важливим при роботі з фінансовими даними, схильними до волатильності. Така стійкість забезпечує більш надійні та стабільні прогнози. По-друге, важливою перевагою ШД є його інтерпретованість. На відміну від інших методів, результати, отримані з використанням ШД, легше аналізувати та пояснювати, що є ключовим фактором при прийнятті інвестиційних рішень.

Третім важливим аспектом є вимоги до обчислювальних ресурсів. ШД вимагають менших обчислювальних ресурсів у порівнянні з більш складними моделями, такими як глибокі нейронні мережі. Це робить їх підходящими для ситуацій, де доступ до великих обчислювальних потужностей є обмеженим. Також ШД ефективний для великих датасетів і здатний обробляти велику кількість ознак, що є типовим для фінансових даних. Він також добре адаптуються до різних типів даних і може бути використаний для як класифікації, так і для регресійних задач.

Враховуючи ці аспекти, ШД є оптимальним вибором для розробки системи прогнозування економічних показників дорогоцінних металів. Його висока точність, стійкість до перенавчання, легкість у використанні та інтерпретації, а також ефективність у роботі з великими датасетами робить його ідеальним для цієї мети.

Висновки до розділу 2

У рамках даного розділу було проведено детальне дослідження різних методів прогнозування економічних показників дорогоцінних металів на фондовому ринку. Зосередження уваги на нейромережевих технологіях та традиційних методах прогнозування, таких як статистичний аналіз, авторегресійні моделі та фундаментальний аналіз, виявило важливість цих підходів у короткостроковому прогнозуванні.

Було показано, що нейронні мережі забезпечують точні прогнози завдяки здатності обробляти великі масиви даних та виявляти складні взаємозв'язки, що є критично важливим для інвесторів та трейдерів. Особливо важливим виявився аналіз ШНМ, КНМ РНМ та ШЛ які демонструють високу ефективність у прогнозуванні ринкових тенденцій.

Фундаментальний аналіз, зосереджений на вивченні базових економічних, фінансових і політичних факторів, виявився важливим для середньо- та довгострокового прогнозування. Він дозволяє глибоко зрозуміти внутрішню вартість активів і їх потенціал зростання.

В результаті проведеного аналізу метод ШЛ були визначено як найбільш перспективний для розробки системи прогнозування. Цей метод вирізняється здатністю до ефективної обробки великих обсягів даних, стійкістю до перенавчання, високою інтерпретованістю результатів та гнучкістю у використанні.

ШЛ демонструють значні переваги у порівнянні з іншими методами, зокрема, здатність до аналізу довготривалих трендів, що є особливо цінним у контексті фінансових ринків з високою волатильністю. Їхнє використання у комбінації з більш сучасними нейромережевими технологіями дозволяє досягти більш точних і комплексних прогнозів.

3 ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ НЕЙРОМЕРЕЖЕВОГО ПРОГНОЗУВАННЯ

3.1 Формалізація задачі прогнозування

Формалізація задачі прогнозування економічних показників дорогоцінних металів фондового ринку України з використанням нейромережевих технологій передбачає створення комплексної моделі машинного навчання. Ця модель аналізує різноманітні аспекти економічних даних, історичних цінових трендів, а також може враховувати макроекономічні індикатори, що впливають на ціни дорогоцінних металів. Процес формалізації включає наступні етапи:

Вибір методології

Для вирішення задачі короткострокового прогнозування економічних показників дорогоцінних металів на фондовому ринку з використанням нейромережевих технологій обрано метод ШЛ.

Кожне дерево рішень T_i у ШЛ побудоване на основі випадкової підмножини навчальних даних D_i, вибраної з використанням бутстрапінгу. Для кожного розщеплення у дереві використовується підмножина ознак. Якщо розглядати задачу регресії, оптимальне розщеплення вузла визначається на основі мінімізації середньоквадратичної помилки (MSE).

Кінцевий прогноз швидкого лісу обчислюється шляхом усереднення виходів всіх дерев для задачі регресії або використання голосування більшістю для задачі класифікації:

$$Y = \frac{1}{N}\sum_{i=1}^{N} T_i(X) \qquad (3.1)$$

де N - кількість дерев у лісі, а $T_i(X)$ - прогноз і-го дерева для вхідних даних X.

Класифікаційний прогноз вибирається на основі голосування більшістю. Для кожного класу c, підраховується кількість дерев, які прогнозують цей клас, і вибирається клас із найбільшою кількістю голосів.

$$Y = argmax_c \sum_{i=1}^{N} I(T_i(X) = c) \qquad (3.2)$$

де $I(T_i(X))$ - індикаторна функція, яка приймає значення 1, якщо прогноз *i*-го дерева дорівнює класу c, *i* 0 в іншому випадку;

$argmax_c$ - це операція, яка вибирає таке значення *c*, для якого вираз всередині досягає максимального значення. Це означає вибір класу *c*, для якого найбільша кількість дерев зробила прогноз. Це клас, який «виграє» голосування більшістю серед усіх дерев лісу.

Вхідні дані

Вхідні дані, які включають характеристики фінансових показників, можна представити у вигляді вектору вхідних даних:

$$X = \{X_1, X_2 \dots X_n\} \qquad (3.3)$$

де X_i відображає кожну характеристику ринкових даних, зокрема:

– ціна відкриття (Open). X_1 – представляє ціну відкриття ринку, вказує на початковий тренд ринку на день;

– найвища ціна (High). X_2 – відображає найвищу ціну протягом торгового дня, показує волатильність ринку;

– найнижча ціна (Low). X_3 – представляє найнижчу ціну протягом дня, ще один показник волатильності;

– ціна закриття ринку (Close). X_4 – ціна закриття, важливий показник для прогнозування, який відображає кінцевий стан ринку;

– об'єм торгів (Volume). X_5 – кількість контрактів, укладених протягом дня, вказує на інтерес до ринку.

Кожна з цих характеристик несе важливу інформацію про поведінку ринку і використовується для прогнозування майбутніх цін. Алгоритм аналізує ці вхідні дані для побудови моделі, яка може ефективно прогнозувати ціни на основі існуючих ринкових тенденцій та історичних даних.

Конкатенація

Математично, конкатенація цих ознак може бути виражена як формування вектора ознак:

$$\text{Features} = \{X_1, X_2, X_3, X_5\} \qquad (3.4)$$

де X_1, X_2, X_3, X_5 відповідно є значеннями для Open, High, Low, Volume.

На наступному етапі проходить трансфоормація колонки Close як Label. Цей процес можна виразити як просте присвоєння:

$$\text{Label} = X_4 \qquad (3.5)$$

де X_4 представляє значення колонки Close.

Після виконання усіх трансформацій, загальний вектор ознак для моделі буде виглядати наступним чином:

$$Total\ Features = [Features, DateFeaturized] \qquad (3.6)$$

де «*Total Features»* включає всі числові ознаки та перетворені ознаки з дати.

Оцінка моделі

Для оцінки якості моделі використовуються наступні математичні функції, що відображають різні метрики:

1. Коефіцієнт детермінації R^2. Коефіцієнт детермінації вимірює, наскільки добре прогнози моделі відповідають реальним даним:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \qquad (3.7)$$

де:

y_i - істинні значення;

$\hat{y}_i$- передбачені значення моделлю;

$\hat{y}_i$- середнє значення істинних значень;

n - кількість спостережень.

2. Середня абсолютна помилка (MAE)

Середня абсолютна помилка вимірює середнє абсолютне відхилення передбачених значень від фактичних:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \qquad (3.8)$$

де: $|y_i - \hat{y}_i|$ – абсолютне значення різниці між істинним і передбаченим значеннями.

3. Середня квадратична помилка (MSE)

Середня квадратична помилка вимірює середнє значення квадратів помилок:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \qquad (3.9)$$

4. Корінь із середньої квадратичної помилки (*RMSE*)

RMSE є квадратним коренем із *MSE* і дає помилку у тих же одиницях, що й дані:

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad (3.10)$$

5. Функція втрат (*Loss Function*)

Функція втрат оцінює, наскільки добре модель прогнозує фактичні значення. Точна форма функції втрат залежить від типу задачі та вибору моделі. Наприклад, для задачі регресії часто використовується *MSE* як функція втрат:

$$Loss\ Function = MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \qquad (3.11)$$

Вихідні дані моделі

Вихідні дані моделі можна виразити як функцію, яка відображає вхідні дані в передбачення. Якщо взяти в розрахунок, що модель виконує прогнозування ціни закриття ринку (означене як Close), формула може виглядати наступним чином:

$$\hat{y} = f(X) \qquad (3.12)$$

де: представляє передбачену ціну закриття ринку (Close);

f - функція прогнозування, яка представляє модель;

X - вектор вхідних характеристик, який може включати такі параметри, як ціна відкриття, найвища ціна, найнижча ціна та об'єм торгів.

Прогнозована ціна закриття (Close) - це основна вихідна характеристика, яка визначається як:

$$Close = \hat{y} \qquad (3.13)$$

де, Close є передбаченням моделі щодо ціни закриття ринку.

Оцінка якості цих передбачень здійснюється за допомогою зазначених вище метрик, що дозволяють визначити точність моделі.

3.3 Внутрішнє проектування системи

3.3.1 Вибір мови програмування та технологічної платформи

Вибір мови програмування є ключовим моментом у процесі розробки програмного забезпечення, адже саме він визначає потенціал розробки, а також забезпечує зручність подальшої підтримки і масштабування проекту.

C# — це потужна мова програмування, розроблена Microsoft, яка поєднує в собі простоту, сучасність і об'єктно-орієнтований підхід [48]. Вона широко використовується для створення різноманітних застосунків, від веб-сайтів до програм для Windows. C# особливо популярний у розробці ігор завдяки своїй інтеграції з Unity, ведучим ігровим двигуном. Вона також підтримує багатоплатформенність і має сильну спільноту, що постійно розвивається, що робить її відмінним вибором для програмістів на різних етапах їхньої кар'єри [49].

C++ — це високопродуктивна мова програмування, яка забезпечує об'єктно-орієнтовані та низькорівневі можливості [50]. Вона широко використовується в системному програмуванні, розробці ігор, і для створення програмного забезпечення, що вимагає високої ефективності. Ця мова є популярною завдяки своїй гнучкості і великому набору функцій. C++ підтримує багато парадигм програмування і пропонує розробникам контроль над системними ресурсами. Ця мова має велику спільноту та ряд стандартів, які постійно оновлюються і вдосконалюються [51].

Python — це високорівнева, інтерпретована мова програмування, відома своєю читабельністю та ефективністю. Ця мова широко застосовується у веб-розробці, наукових дослідженнях, штучному інтелекті та автоматизації. Python підтримує різні парадигми програмування, включаючи об'єктно-орієнтовану, процедурну та функціональну. Його простота та гнучкість роблять його популярним вибором як для початківців, так і для досвідчених розробників. Велика стандартна бібліотека та активна спільнота є ще одними з переваг Python [52].

Порівняльна таблиця основних характеристик мов програмування C#, C++ та Python представлена на рис. 3.1.

Таблиця 3.1 – Порівняльний аналіз мов програмування

Характеристика	C#	C++	Python
Підтримка середовища розробки (IDE)	Відмінна (Visual Studio)	Гарна (різні варіанти, включаючи Eclipse, Visual Studio)	Гарна (PyCharm, VS Code, Jupyter)

Управління пам'яттю	Автоматичне (Сбірник сміття)	Ручне	Автоматичне (Сбірник сміття)
Розробка крос-платформних додатків	Сильна (через .NET Core)	Гарна (з додатковими бібліотеками, такими як Qt)	Сильна (Інтерпретована, різноманітні бібліотеки)
Підтримка фреймворків	Велика (.NET Framework, .NET Core)	Широкий спектр (STL, Boost, Qt)	Величезна (Django, Flask, SciPy, NumPy)
Безпека на рівні мови	Висока (Типізація, Керований код)	Нижча (Арифметика вказівників, ручне управління пам'яттю)	Помірна (Динамічна типізація, Інтерпретована)
Легкість вивчення	Помірна	Складна	Легка
Спільнота та підтримка	Велика та активна	Велика, але більш фрагментована	Надзвичайно велика та активна

Вибір C# для розробки системи прогнозування показників дорогоцінних металів фондового ринку може бути виправданий наступними перевагами:

– підтримка середовища розробки (IDE). C# тісно інтегрований з Visual Studio, одним з найпотужніших середовищ розробки, що забезпечує відмінні інструменти для налагодження, проектування, та підтримки коду, що є критично важливим для розробки складних фінансових систем;

– управління пам'яттю. Автоматичне управління пам'яттю в C# за допомогою сбірника сміття зменшує ризики пов'язані з витоками пам'яті та іншими помилками, що можуть виникати при ручному управлінні пам'яттю, як у C++;

– розробка крос-платформних додатків. Завдяки .NET Core, C# дозволяє розробляти додатки, які легко адаптуються під різні платформи, що розширює можливості їх використання та розповсюдження;

– підтримка фреймворків. .NET Framework та .NET Core надають багатий набір функціональності для розробки, включаючи інструменти для роботи з даними, мережею, і веб-сервісами, що є важливим для фінансових систем;

– безпека на рівні мови. Висока рівень безпеки в C#, забезпечена сильною типізацією та керованим кодом, гарантує стабільність та надійність програмного продукту;

– легкість вивчення. Порівняно з C++, C# має помірну складність вивчення, що робить його доступнішим для нових розробників у команді.

Отже, сукупність цих факторів робить C# відмінним вибором для розробки системи прогнозування на фондовому ринку, забезпечуючи високий рівень продуктивності, безпеки та гнучкості.

3.3.2 Ієрархія та взаємодія класів

Розроблена система містить кілька взаємопов'язаних компонентів, які взаємодіють між собою для забезпечення повного циклу роботи з нейронними мережами. Вона включає інструменти для обробки та підготовки даних, інтерфейси для взаємодії з користувачем, а також модулі для тренування та тестування моделей. Користувачеві надається можливість не тільки вибирати та тренувати моделі, але й оцінювати їх ефективність та застосовувати для конкретних задач прогнозування.

У центрі цієї системи знаходиться структуроване представлення у вигляді діаграми класів, яка демонструє архітектуру та взаємозв'язки між основними компонентами системи. На рис. 3.2 можна побачити, як взаємодіють різні класи, включаючи ті, що відповідають за зберігання даних, їх обробку, взаємодію з користувачем, і, звичайно ж, за сам процес навчання та прогнозування за допомогою нейронних мереж. Ця діаграма допомагає зрозуміти структуру системи та є ключовим елементом для розуміння її функціональності.

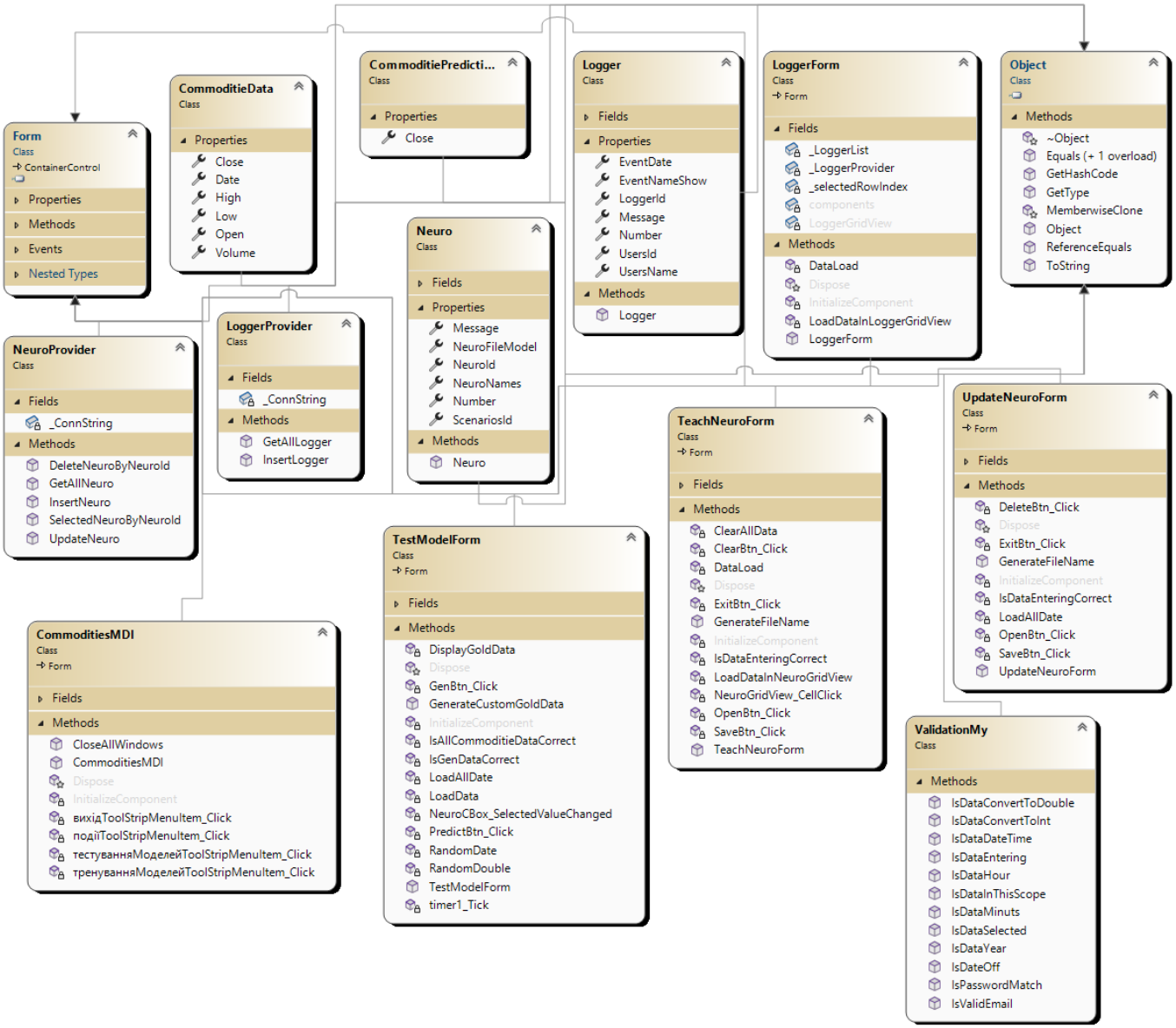

Рисунок 3.2 – Діаграма ієрархії класів

1. Клас ValidationMy виконує критичну роль у забезпеченні правильності та валідності даних, що вводяться або обробляються в рамках програмного забезпечення. Цей клас містить ряд методів, кожен з яких відповідає за перевірку певного типу даних чи умов. Клас складається із наступних методів:

– IsDataEntering. Перевіряє чи рядок даних не порожній. Це основний метод для перевірки введення користувача, гарантуючи, що важливі поля форми не залишились незаповненими;

– IsDataSelected. Перевіряє, чи користувач вибрав значення із запропонованих варіантів (наприклад, у випадаючому меню), і що вибране значення не є значенням за замовчуванням;

– IsDataInThisScope. Гарантує, що числове значення входить у визначений діапазон. Це корисно для встановлення обмежень на вхідні дані, як наприклад, вік або кількість;

– IsDataConvertToInt і IsDataConvertToDouble. Перевіряють, чи можна рядок перетворити в ціле число або число з плаваючою комою. Ці методи корисні для перевірки введення даних, які мають бути у числовому форматі.

2. Клас CommoditieData представляє собою структуру даних, призначену для зберігання та обробки інформації про торгівлю на ринку дорогоцінних металів. Кожен екземпляр цього класу містить детальну інформацію про конкретний торговий день, включаючи:

– Date. Рядок, що зберігає дату запису даних. Це поле відображає, коли були зроблені записи про торгові ціни;

– Open. Змінна типу float, що відображає ціну відкриття ринку. Це ціна, за якою почалася торгівля дорогоцінним металом у вказаний день;

– High. Змінна типу float, що показує найвищу ціну протягом торгового дня. Це надає уявлення про максимальну вартість металу в той день;

– Low. Змінна типу float, яка вказує на найнижчу ціну протягом дня. Це дозволяє зрозуміти, як далеко могла впасти вартість металу;

– Close. Змінна типу float, яка відображає ціну закриття ринку. Це ціна, за якою торгівля металом закінчилася в той день;

– Volume. Змінна типу float, що вказує кількість контрактів, що були торговані протягом дня. Це дає уявлення про загальний обсяг торгів за вказаний період.

Анотації [LoadColumn(n)], де n — номер стовпчика, використовуються для визначення, з якого стовпчика даних повинно бути завантажено кожне поле при читанні з джерела даних, наприклад, з CSV-файлу.

3. Клас CommoditiePrediction в контексті машинного навчання та аналітики даних служить для представлення прогнозу ціни закриття на ринку дорогоцінних металів. Основні характеристики класу:

– Close. Це основне і єдине поле класу, що представляє прогнозовану ціну закриття ринку. Тип даних float дозволяє відображати ціну з дробовою частиною, що є типовим для фінансових даних;

– ColumnName(«Score»). Ця анотація, застосована до поля Close, вказує на те, що прогнозоване значення має бути асоційоване з колонкою під назвою «Score» у вихідному наборі даних, який генерується моделлю машинного навчання. Це зазвичай використовується в контексті прогнозування, де модель виводить певне числове значення, яке представляє передбачуваний результат.

4. Клас LoggerProvider призначений для взаємодії з базою даних з метою реєстрації та отримання логів (записів журналу подій). Цей клас виконує дві основні функції:

– InsertLogger. Цей метод відповідає за вставку нового запису логу в базу даних. Він приймає два параметри: EventNameShow, який є рядком опису події, та EventDate, датою та часом події. Спочатку метод встановлює з'єднання з базою даних, використовуючи рядок підключення _ConnString. Потім він формує SQL-запит на вставку та виконує його, використовуючи передані параметри;

– GetAllLogger. Цей метод витягує всі записи з таблиці логів із бази даних та повертає їх як список об'єктів Logger. Він використовує SQL-запит для вибору всіх записів з таблиці Logger, відсортованих за датою події в порядку спадання. Кожен запис читається з бази даних та перетворюється в об'єкт Logger, який додається до списку. Якщо жодного запису не знайдено, метод створює спеціальний об'єкт Logger з повідомленням про відсутність даних.

5. Клас NeuroProvider відіграє роль менеджера для взаємодії з таблицею Neuro у базі даних. Цей клас містить набір методів для різних операцій з об'єктами типу Neuro, які можуть представляти, наприклад, нейронні моделі або конфігурації для машинного навчання. Основні методи класу включають:

– InsertNeuro. Додає новий запис до таблиці Neuro. Метод використовує параметри NeuroNames (назва моделі) та NeuroFileModel (файл моделі), вставляючи їх у базу даних;

– GetAllNeuro. Витягує всі записи з таблиці Neuro. Кожен запис перетворюється на об'єкт Neuro і збрається у список;

– SelectedNeuroByNeuroId. Отримує один запис з таблиці Neuro за ідентифікатором NeuroId. Метод повертає об'єкт Neuro з відповідними даними;

– UpdateNeuro. Оновлює існуючий запис у таблиці Neuro. Метод використовує NeuroNames, NeuroFileModel та NeuroId для оновлення відповідного запису;

– DeleteNeuroByNeuroId. Видаляє запис із таблиці Neuro за ідентифікатором NeuroId.

6. Клас LoggerForm, який є похідним від класу Form, слугує як графічний інтерфейс користувача для відображення та взаємодії з даними логування (Logger). Основні методи класу:

– конструктор LoggerForm. Ініціалізує компоненти форми та викликає метод DataLoad для завантаження даних логу;

– метод DataLoad. Виконує завантаження даних логу від LoggerProvider і відображає їх у LoggerGridView, відновлюючи вибраний рядок і позицію прокрутки.

– метод LoadDataInLoggerGridView. Налаштовує та заповнює LoggerGridView даними з LoggerList. Включає налаштування візуальних елементів як колонок та їх стилів. Якщо в LoggerList відсутні дані, відображає повідомлення про відсутність даних.

7. Клас TeachNeuroForm, що є похідним від Form, служить як графічний інтерфейс користувача для навчання та управління нейронними мережами. Основні методи класу включають:

– метод OpenBtn_Click у класі TeachNeuroForm запускає процес вибору та завантаження даних для навчання нейронної мережі. Він дозволяє користувачеві вибрати файл з даними, ініціалізує процес навчання мережі та виводить метрики якості отриманої моделі;

– DataLoad. Метод відповідає за завантаження та відображення списку нейронних мереж з бази даних у таблицю на формі. Він також відновлює вибраний рядок і позицію прокрутки у таблиці;

– LoadDataInNeuroGridView. Цей метод налаштовує таблицю NeuroGridView, очищаючи її та визначаючи структуру колонок для відображення даних зі списку NeuroList;

– SaveBtn_Click. Активується при натисканні на кнопку збереження. Цей метод зберігає навчену модель у файл і робить запис у журнал подій. Перед збереженням він перевіряє правильність введення даних;

– ClearAllData. Очищує всі поля введення та відображення даних на формі, встановлюючи їх у початковий стан. Також перезавантажує дані з бази даних;

– GenerateFileName. Генерує унікальну назву файлу для збереження моделі, використовуючи поточну дату і час;

– ClearBtn_Click. Очищує всі дані на формі, викликаючи метод ClearAllData.

– ExitBtn_Click. Закриває форму навчання нейронної мережі;

– IsDataEnteringCorrect. Перевіряє, чи всі необхідні дані були коректно введені користувачем перед збереженням моделі.

– NeuroGridView_CellClick. Обробляє подію кліку по комірці в таблиці NeuroGridView, дозволяючи користувачеві вибрати конкретну нейронну мережу для подальших дій.

8. Клас TestModelForm, який є розширенням стандартного класу форм Form, використовується для тестування та оцінки нейронних мереж. Основні методи класу:

– LoadAllDate. Завантажує список доступних нейронних мереж з бази даних для відображення у випадаючому списку на формі. Цей метод також встановлює поточні дати для елементів вибору дати на формі;

– PredictBtn_Click. Виконує прогнозування за допомогою обраної нейронної мережі, використовуючи дані, введені користувачем. Результати прогнозу виводяться у текстовому полі на формі;

– NeuroCBox_SelectedValueChanged. Активізується при зміні вибору нейронної мережі користувачем. Відповідно до вибору, вантажить дані обраної мережі для подальшого тестування;

– LoadData. Завантажує вибрану нейронну мережу з файлу і налаштовує двигун прогнозування для цієї моделі;

– IsAllCommoditieDataCorrect та IsGenDataCorrect. Перевіряють коректність введених даних перед прогнозуванням, забезпечуючи валідність вхідних параметрів для моделі;

– GenBtn_Click. Керує процесом генерації випадкових даних для тестування моделі, активуючи або зупиняючи цей процес;

– timer1_Tick. Циклічно генерує випадкові дані і виконує їх прогнозування, виводячи результати на форму;

– GenerateCustomGoldData, RandomDate, RandomDouble. Допоміжні методи для створення випадкових даних в заданих діапазонах, які імітують реальні торгові дані;

– DisplayGoldData. Відображає згенеровані дані в текстовому полі на формі, надаючи користувачеві детальний огляд введених та прогнозованих значень.

9. Клас UpdateNeuroForm використовується для оновлення, тестування або видалення існуючих нейронних мереж. Він складається із наступних методів:

– OpenBtn_Click. Цей метод відкриває вікно вибору файлу, дозволяючи користувачеві вибрати нові дані для тренування моделі. Він також відповідає за підготовку, тренування та оцінку моделі, після чого виводить метрики якості;

– SaveBtn_Click. Зберігає оновлену нейронну мережу після її навчання. Цей метод перевіряє, чи навчання моделі вже відбулося, та зберігає модель у файлі, оновлюючи інформацію про неї в базі даних;

– DeleteBtn_Click. Видаляє вибрану нейронну мережу з бази даних. Цей метод викликається при натисканні кнопки видалення, підтверджуючи наміри користувача перед видаленням;

– LoadAllDate. Завантажує дані про вибрану нейронну мережу для її оновлення або видалення. Метод встановлює відображення існуючих даних у відповідних полях форми;

– ExitBtn_Click. Закриває форму оновлення нейронної мережі;

– GenerateFileName. Генерує унікальне ім'я файлу для збереження нейронної мережі, використовуючи поточну дату і час;

– IsDataEnteringCorrect. Перевіряє правильність введення назви нейронної мережі, а також чи модель дійсно була навчена перед її збереженням.

3.3.3 Принципи та шаблони проектування, використані у розробці

Принципи, які були використані у розробці цієї системи, охоплюють широкий спектр кращих практик та підходів в області програмування:

– модульність. Система розроблена з використанням модульного підходу, де кожен компонент або клас має чітко визначені функції та відповідає за окрему частину процесу. Це сприяє легкій зміні, підтримці та розширенні системи;

– об'єктно-орієнтоване програмування (ООП). Використання принципів ООП, таких як інкапсуляція, спадкування та поліморфізм, забезпечує чітку структуру коду та високу повторюваність елементів;

– валідація даних. Система інтегрує ретельну валідацію даних на кожному етапі, від введення даних користувачем до їх обробки та аналізу. Це запобігає помилкам та забезпечує надійність результатів;

– гнучкість та масштабованість. Система розроблена з можливістю легкого додавання нових функцій, алгоритмів або моделей, що забезпечує її гнучкість та масштабованість;

– інтуїтивно зрозумілий інтерфейс користувача. Графічний інтерфейс користувача розроблений таким чином, щоб бути зручним та інтуїтивно зрозумілим, забезпечуючи ефективну взаємодію користувачів з системою;

– робота з даними. Ефективне управління даними, включаючи їх збір, зберігання та обробку, є ключовим аспектом, що забезпечує точність та надійність аналізу.

Ці принципи разом формують міцну основу для розробки надійної, ефективної та легкої у використанні системи для роботи з нейронними мережами у сфері економічного прогнозування.

Для розробки системи для навчання та прогнозування за допомогою нейронних мереж було обрано «фабричний метод» (Factory Method):

– призначення. Цей шаблон використовується для створення об'єктів, дозволяючи підкласам вирішувати, які об'єкти слід створити. У системі машинного навчання це особливо важливо, оскільки існує багато типів моделей та алгоритмів навчання, кожен з яких має свої особливості;

– застосування у системі. У контексті навчання нейронних мереж, фабричний метод був використаний для створення різних типів моделей нейронних мереж та підготовки різних наборів даних. Наприклад, можна мати фабричний метод для створення різних типів нейронних мереж (звичайні, згорткові, рекурентні тощо) або для обробки даних під конкретні потреби різних моделей.

Використання фабричного методу в контексті системи забезпечує гнучкість у виборі та створенні компонентів системи, що є критично важливим для адаптації до швидко змінюваних вимог у цій області.

Фабричний метод є одним з фундаментальних шаблонів проектування у програмуванні, який надає ряд переваг:

– гнучкість у створенні об'єктів. Цей шаблон дозволяє класам делегувати відповідальність за створення екземплярів об'єктів підкласам. Таким чином, класи можуть створювати об'єкти без необхідності знати точний тип створюваних екземплярів;

– зниження залежностей між класами. Фабричний метод зменшує прямі залежності між різними частинами програми, оскільки клієнтські класи не потребують конкретного класу для створення екземплярів. Замість цього вони використовують інтерфейс або абстрактний клас, визначений фабрикою;

– легкість внесення змін і розширення. Фабричний метод полегшує додавання нових типів продуктів без зміни існуючого коду, оскільки створення

об'єктів централізовано. Це робить систему більш масштабованою та адаптивною до змін;

– окремий контроль над процесом створення об'єктів. Завдяки цьому шаблону, програма може мати більш детальний контроль над тим, як і коли об'єкти створюються. Це особливо корисно у складних системах, де процес створення об'єктів може включати додаткову логіку або конфігурацію;

– підтримка принципів SOLID. Фабричний метод підтримує принципи SOLID, зокрема принцип відкритості/закритості (Open-Closed Principle), який вказує на те, що класи повинні бути відкритими для розширення, але закритими для змін;

– полегшення тестування та обслуговування. Оскільки фабричний метод сприяє вищій модулярності та меншій залежності між компонентами, це полегшує тестування окремих частин програми та її подальше обслуговування.

Усі ці переваги роблять фабричний метод особливо корисним у великих та складних системах, де потрібна гнучкість у створенні та управлінні різноманітними об'єктами, а також у системах, що постійно розвиваються і вимагають легкого впровадження змін.

3.4 Розробка інтерфейсу користувача

3.4.1 Проектування структури екранів системи

Проектування структури екранів у розробленій системі виконується з урахуванням ключових принципів зручності використання, інтуїтивності інтерфейсу та ефективності взаємодії користувача з системою.

Центральний екран системи забезпечує легкий доступ до всіх основних функцій, таких як завантаження даних, тренування моделей, тестування та аналіз результатів. Він має чітко структуроване меню та інструментальну панель, які дозволяють користувачеві легко переходити між різними частинами системи.

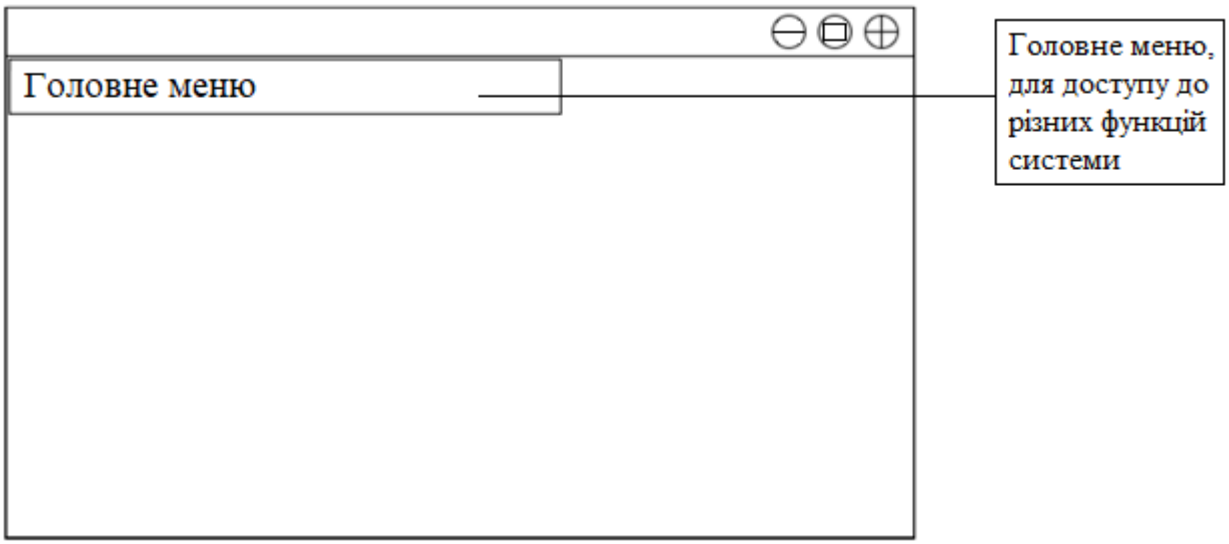

Рисунок 3.3 – Головне вікно системи

На рис. 3.4 представлено екран, де користувачі можуть вибирати датасети для навчання моделей. Він також забезпечує вивід результатів процесу навчання, після чого користувач може прийняти рішення щодо збереження моделі у системі.

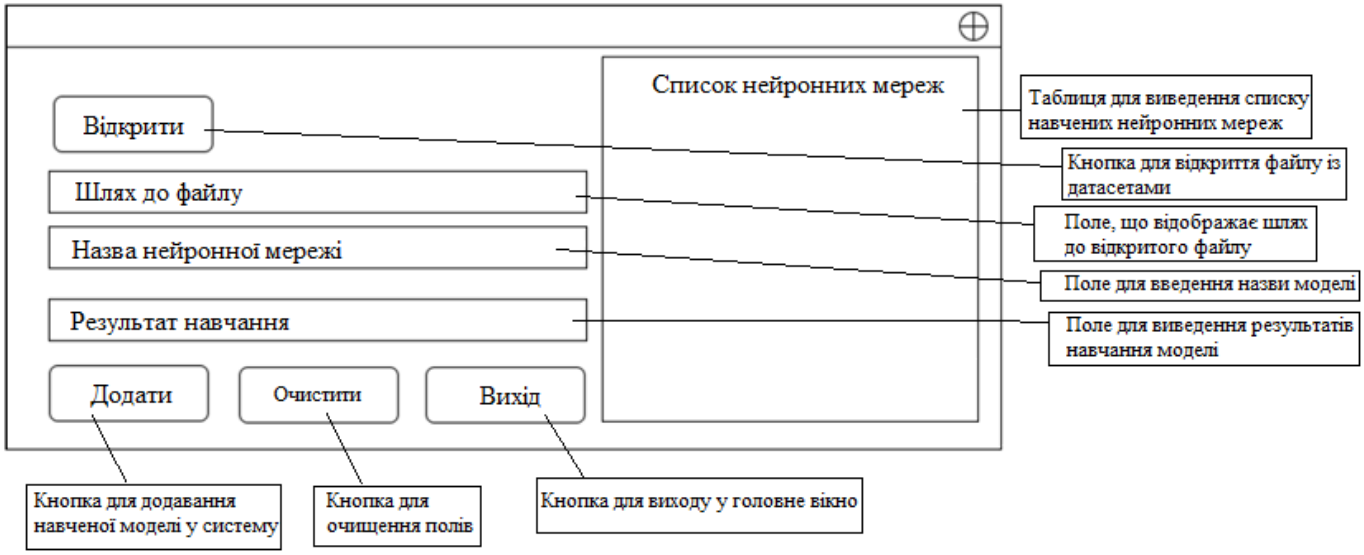

Рисунок 3.4 – Форма для тренування та зберігання моделей

Для редагування даних моделі або проведення повторного навчання розроблено екранну форму, що представлена на рис. 3.5.

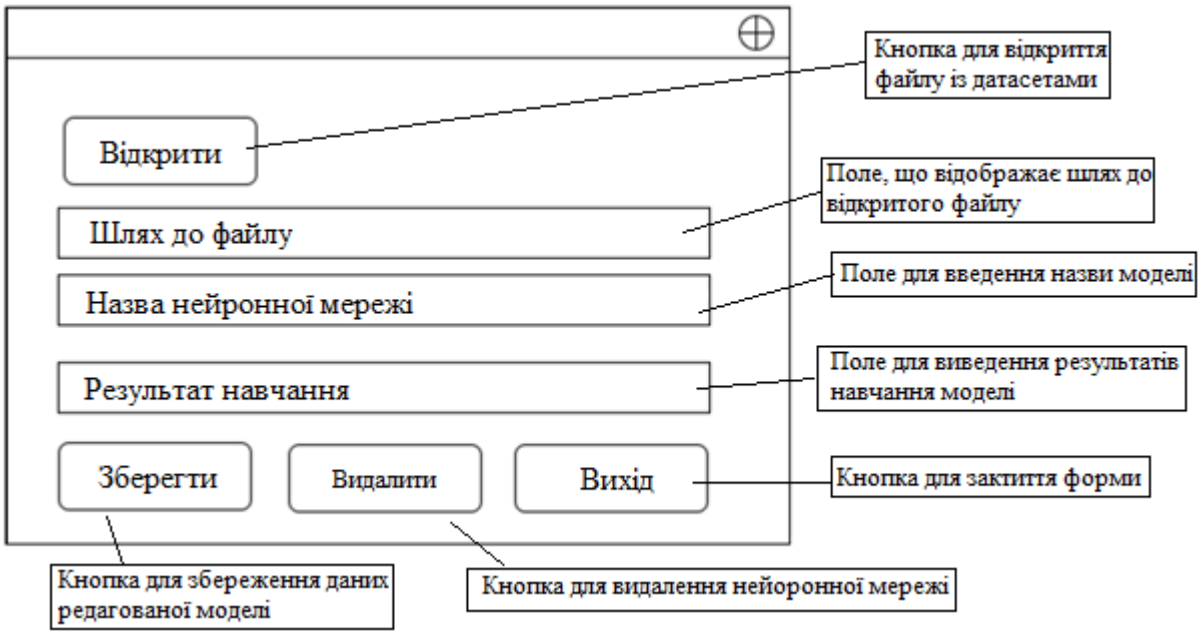

Рисунок 3.5 – Форма для редагування даних моделей

Для тестування нейронних мереж проведено проектування форми, що представлено на рис. 3.6.

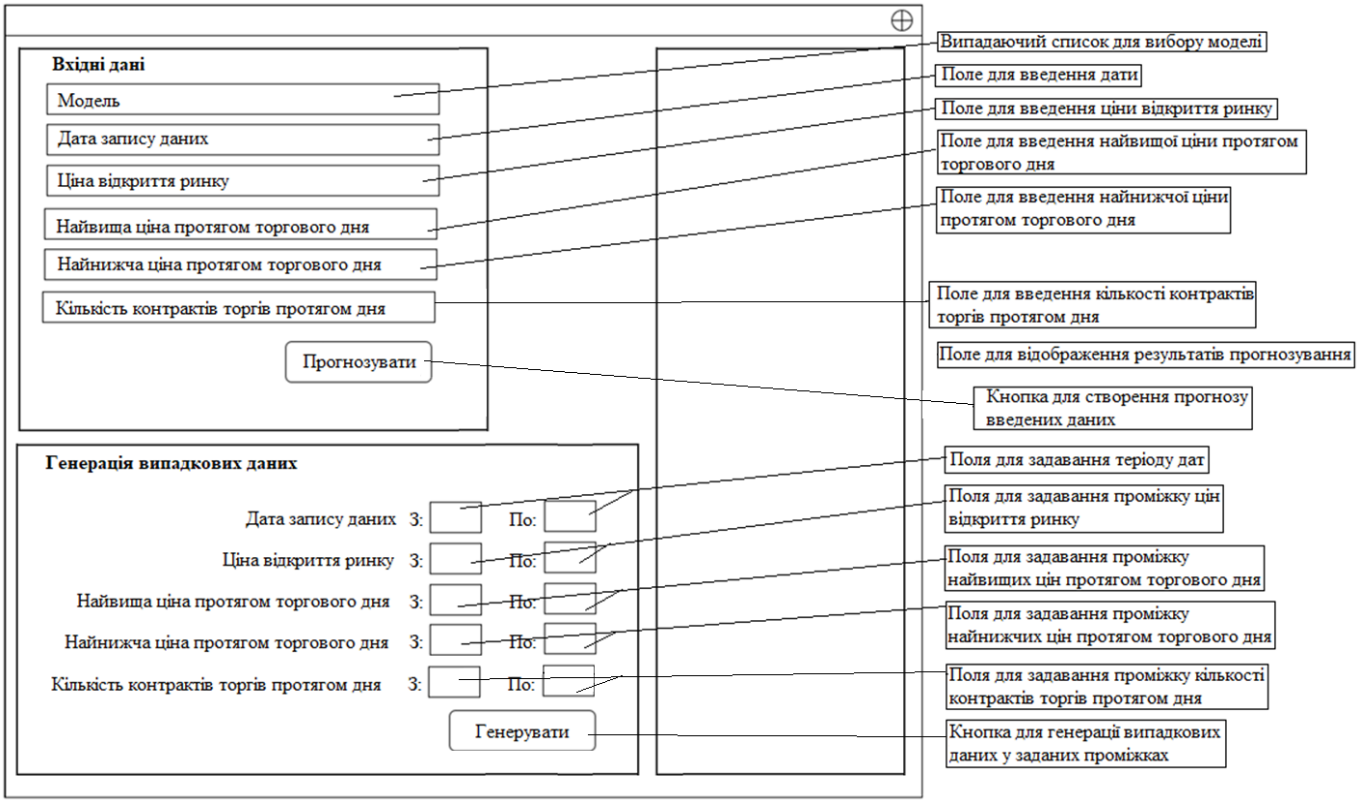

Рисунок 3.6 – Форма для тестування моделей нейронних мереж

У формі реалізовано вибір моделі нейронної мережі та можливість введення параметрів для проведення її тестування, а також можливість генерації випадкових даних із заданим діапазоном значень кожного параметру.

3.4.2 Реалізація інтерфейсу користувача

Після завершення попередніх етапів розробки, був створений користувацький інтерфейс, що виступає як зв'язуюча ланка між користувачами та програмним забезпеченням. Як демонструється на рис. 3.7, цей інтерфейс забезпечує зручну та зрозумілу платформу для взаємодії з функціоналом системи.

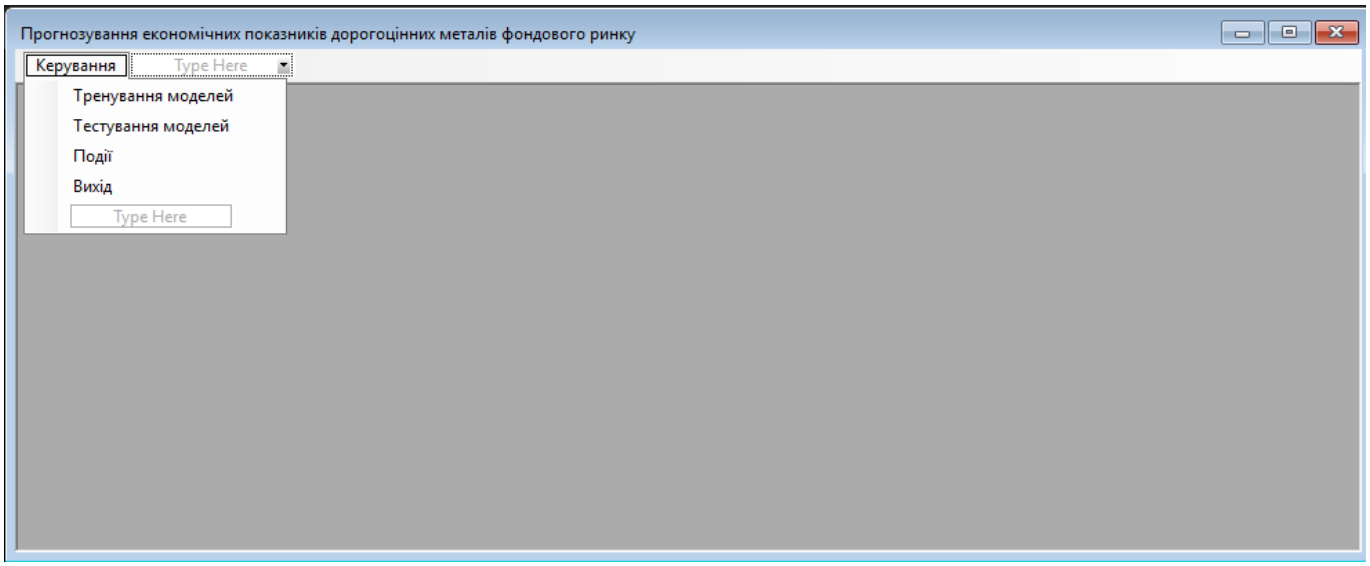

Рисунок 3.7 – Головна форма інтерфейсу

Кожен вибір у меню ініціює відповідну відповідь системи. Наприклад, як видно на рисунку 3.8, вибір опції "Тренування моделей" активує інструменти для навчання, тестування та оцінювання моделей.

```
1 reference
private void тренуванняМоделейToolStripMenuItem_Click(object sender, EventArgs e) {
    CloseAllWindows();
    TeachNeuroForm teachNeuroForm = new TeachNeuroForm();
    teachNeuroForm.MdiParent = this;
    teachNeuroForm.WindowState = FormWindowState.Maximized;
    teachNeuroForm.Show();
}
```

Рисунок 3.8 – Код створення об’єкту форми «Тренування моделей»

Наступним кроком було проведення розробки форми для тренування нейронних мереж, зовнішній вигляд якої показано на рис. 3.9.

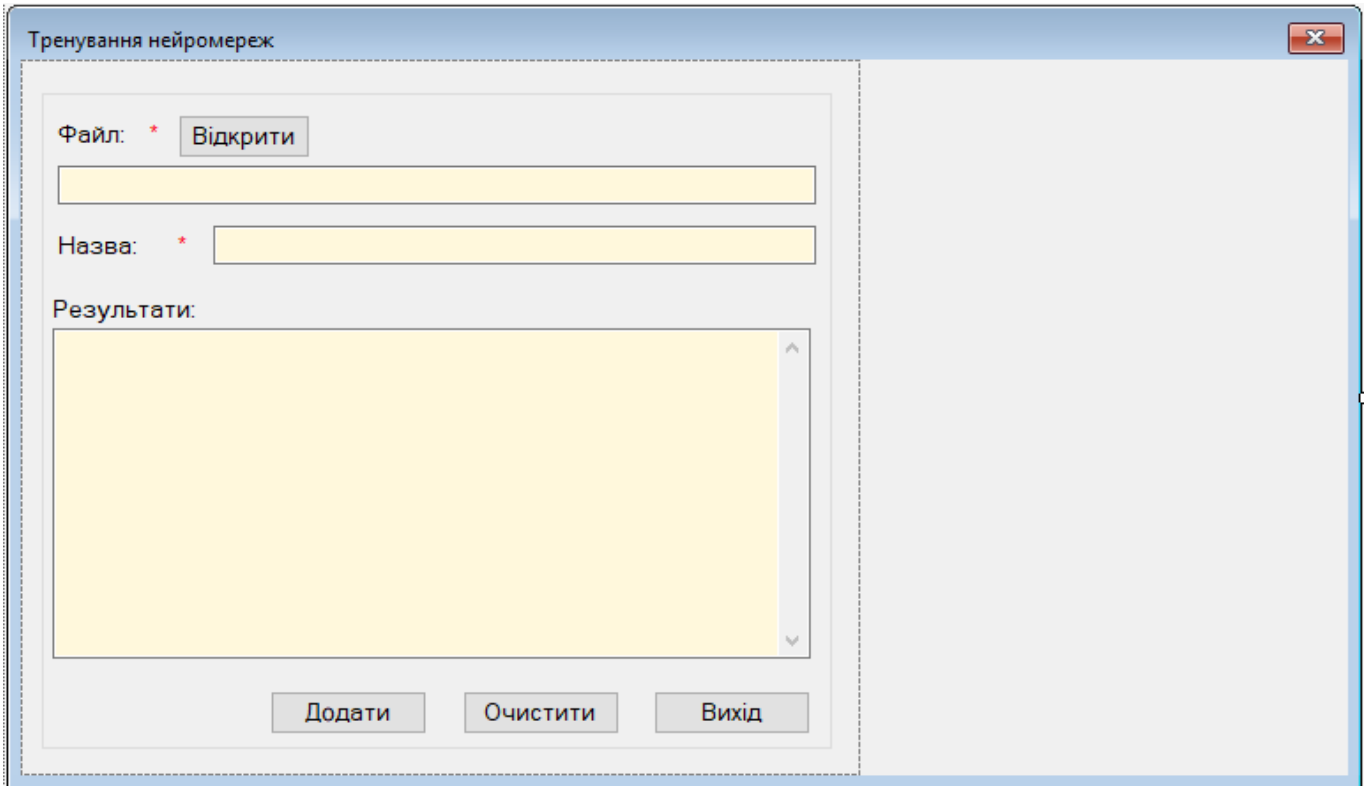

Рисунок 3.9 – Форма для тренування нейронних мереж

У конструкторі форми викликається метод «DataLoad», що призначений для завантаження та відображення даних у таблиці візуального інтерфейсу (рис. 3.10).

```
private void DataLoad() {
  int firstRowIndex = 0;
  if (NeuroGridView.FirstDisplayedScrollingRowIndex > 0) {
    firstRowIndex = NeuroGridView.FirstDisplayedScrollingRowIndex;
  }
  try {
    _NeuroList = _NeuroProvider.GetAllNeuro();
    LoadDataInNeuroGridView(_NeuroList);
    if (_selectedRowIndex == NeuroGridView.Rows.Count) {
      _selectedRowIndex = NeuroGridView.Rows.Count - 1;
    }
    if (_selectedRowIndex >= 0) {
      NeuroGridView.FirstDisplayedScrollingRowIndex = firstRowIndex;
      NeuroGridView.Rows[_selectedRowIndex].Selected = true;
    }
  } catch (Exception ex) {
    MessageBox.Show(ex.ToString());
  }
}
```

Рисунок 3.10 – Код для завантаження і відображення інформації про НМ

У методі спочатку визначається індекс першого рядка, який відображається в таблиці. Якщо таблиця вже була прокручена користувачем, метод зберігає позицію прокрутки. Далі виконується спроба завантаження списку нейронних мереж. Цей список отримується за допомогою методу GetAllNeuro, який викликається з провайдера даних _NeuroProvider.

Після отримання списку він використовується для заповнення даних в таблиці NeuroGridView за допомогою виклику методу LoadDataInNeuroGridView. Якщо індекс вибраного рядка (_selectedRowIndex) дорівнює кількості рядків у таблиці, індекс коректується, щоб вказувати на останній рядок. Інакше, якщо індекс вибраного рядка більше або дорівнює нулю, таблиця прокручується до цього рядка, і цей рядок відзначається як вибраний. У випадку виникнення помилок під час завантаження даних або заповнення таблиці, виводиться повідомлення з описом помилки.

Для проведення навчання НМ реалізовано можливість вибору датасету. Для цього було реалізовано подію "OpenBtn_Click", що спрацьовує при натисканні на кнопку відкрити. На рис. 3.11 показано код процесу вибору файлу із датасетом користувачем.

```
OpenFileDialog openFileDialog = new OpenFileDialog();
openFileDialog.Filter = "Text files (*.csv)|*.csv|All files (*.*)|*.*";
openFileDialog.FilterIndex = 2;
openFileDialog.RestoreDirectory = true;

if (openFileDialog.ShowDialog() == DialogResult.OK) {
  try {
    _Path = openFileDialog.FileName;
    FilePathTBox.Text = openFileDialog.FileName;
```

Рисунок 3.11 – Код для відкриття файлу із датасетом

На початку методу створюється об'єкт для відкриття діалогового вікна, що дозволяє користувачеві вибрати файл зі свого комп'ютера. Встановлюється фільтр для діалогового вікна, що обмежує типи файлів, які можна вибрати. У даному випадку фільтр дозволяє вибирати текстові файли з розширенням .csv та будь-які інші файли. Задається порядковий номер фільтра, який буде встановлено за замовчуванням при відкритті діалогового вікна. Встановлюється параметр, який дозволяє діалоговому вікну запам'ятовувати останній відкритий каталог та відкривається діалогове вікно, і якщо користувач вибирає файл і підтверджує свій вибір, код продовжує виконання.

У разі успішного вибору файлу, шлях до файлу зберігається в змінній, а потім цей шлях відображається в текстовому полі на формі, щоб користувач міг бачити, який файл було вибрано.

Після успішного відкриття файлу ініціалізує новий екземпляр контексту машинного навчання (рис. 3.12).

```
//Контекст
context = new MLContext();
```

Рисунок 3.12 – Ініціалізація контексту

"Контекст" відноситься до основного класу або структури в бібліотеці машинного навчання, яка використовується як вихідний пункт для різних операцій з машинним навчанням. Він надає можливість створювати, тренувати, оцінювати та використовувати навчені моделі. Створення нового екземпляра контексту є необхідним кроком для початку роботи з бібліотекою , оскільки він забезпечує доступ до функцій та методів, необхідних для реалізації процесу навчання.

Після цього проходить завантаження даних з текстового файлу для подальшого використання у системі навчання моделі.

```
// Завантаження даних із файлу
dataFromFilesView = context.Data.LoadFromTextFile<CommoditieData>(_Path,
  hasHeader: true, separatorChar: ',');
```

Рисунок 3.13 – Завантаження тренувального набору даних

Дані завантажуються із зазначеного файлу, шлях до якого визначено в змінній «_Path». Завантаження даних відбувається за допомогою функції, яка читає текстовий файл і перетворює його вміст на формат, придатний для обробки. Файл розглядається як набір даних, де кожен рядок представляє окремий запис. Параметр «hasHeader» вказує, що перший рядок файлу містить заголовки стовпців.

Для розділення даних у файлі використовується символ коми, як розділювач. Результат читання файлу зберігається у змінній «dataFromFilesView», яка потім може використовуватися для тренування моделей машинного навчання або для інших операцій з даними.

Далі виконується розбиття завантажених даних на дві частини: набір для навчання та набір для тестування (рис. 3.14).

```
// Розбиття даних на навчальні та тестові
var splitData = context.Data.TrainTestSplit(dataFromFilesView, testFraction: 0.25);
```

Рисунок 3.14 – Розбиття даних

Дані, які були завантажені та збережені в змінній «dataFromFilesView», діляться на дві окремі групи. Для цього використовується функція «TrainTestSplit» з контексту машинного навчання. Ця функція дозволяє автоматично розділити дані на дві частини. Параметр «testFraction» вказує, що приблизно 25% від усіх даних буде використано як тестовий набір, а решта - як навчальний набір. Результатом роботи цієї функції є об'єкт «splitData», який містить два набори: набір для навчання та набір для тестування.

Після цього проходить процес підготовки та трансформації даних для навчання моделі нейронної мережі (рис. 3.15). Послідовність трансформацій дозволяє структурувати дані таким чином, щоб вони були оптимальні для використання в моделях, забезпечуючи ефективну обробку та аналіз.

```
// Трансформації даних
var dataProcessPipeline = context.Transforms.CopyColumns(outputColumnName: "Label",
  inputColumnName: "Close")
    .Append(context.Transforms.Concatenate("Features", "Open", "High", "Low", "Volume"))
    .Append(context.Transforms.Text.FeaturizeText(inputColumnName: "Date",
    outputColumnName: "DateFeaturized"))
    .Append(context.Transforms.Concatenate("Features",
    "Features", "DateFeaturized"));
```

Рисунок 3.15 – Трансформація даних

На початку створюється ланцюг трансформацій даних, який називається "пайплайн обробки даних". Він використовується для послідовної обробки вхідних даних перед подачею їх в модель. Перша трансформація в пайплайні - це копіювання значень з одного стовпця в інший. Значення зі стовпця "Close" копіюються в новий стовпець "Label", який може використовуватися як мітка для завдань прогнозування. Далі йде трансформація, що об'єднує кілька стовпців у один великий стовпець "Features". Вона включає стовпці "Open", "High", "Low" та "Volume", які будуть використовуватися як характеристики для навчання моделі.

Потім застосовується трансформація для обробки текстових даних. Вона перетворює значення дати з текстового формату у числовий, використовуючи функцію "FeaturizeText". Результат зберігається в стовпці "DateFeaturized".

Остання трансформація в пайплайні - це ще одне об'єднання, яке додає оброблені дати до вже існуючого стовпця "Features", утворюючи повний набір характеристик для моделі.

На наступному етапі відбувається вибір та налаштування алгоритму для тренування моделі (рис. 3.16). Правильний вибір та налаштування алгоритму впливають на ефективність та точність прогнозування моделі.

```
// Вибір алгоритму: FastForest
var trainer =
  context.Regression.Trainers.FastForest(labelColumnName: "Label",
  featureColumnName: "Features");
```

Рисунок 3.16 – Вибір алгоритму тренування моделі

У даному випадку обрано алгоритм "FastForest", який є одним із методів регресійного аналізу. Алгоритм "FastForest" базується на використанні випадкових лісів, які є ефективними для прогнозування числових значень на основі набору характеристик. Для налаштування цього алгоритму визначаються два основні параметри:

– labelColumnName. Цей параметр вказує на назву стовпця, який містить мітки або цільові значення, які потрібно прогнозувати. У даному випадку, це стовпець "Label";

– featureColumnName. Цей параметр вказує на назву стовпця, який містить характеристики або ознаки, які використовуються для тренування моделі. У цьому випадку використовується стовпець "Features".

Створюється об'єкт trainer, який містить налаштований алгоритм «FastForest», готовий до використання в процесі тренування моделі.

Після цього створюється пейплайн для навчання моделі, комбінуючи попередньо визначений пайплайн обробки даних з обраним алгоритмом тренування (рис. 3.17).

```
//Створення пейплайну навчання  моделі
var trainingPipeline = dataProcessPipeline.Append(trainer);
```

Рисунок 3.17 – Створення пейплайну для моделі

Об'єкт dataProcessPipeline, який вже містить серію трансформацій даних, розширюється за допомогою методу Append. Цей метод додає нову компоненту до існуючого пайплайну. Компонентою, яка додається, є об'єкт «trainer», що представляє алгоритм "FastForest" для регресійного аналізу. Результатом є новий

об'єкт «trainingPipeline», який тепер включає як етапи обробки даних, так і алгоритм навчання.

На останньому етапі процесу відбувається навчання моделі (рис. 3.18). Для побудови моделі алгоритм аналізує навчальні дані та "вчиться" з них, формуючи модель, яка згодом може бути застосована для вирішення практичних завдань.

```
// Навчання моделі
trainedModel = trainingPipeline.Fit(splitData.TrainSet);
```

Рисунок 3.18 – Навчання моделі

Для цього використовується пайплайн «trainingPipeline», який був створений на попередньому етапі та містить всі необхідні трансформації даних і алгоритм тренування. Метод «Fit» застосовується до цього пайплайну, він використовується для тренування моделі на основі визначеного набору даних. Як вхідні дані для тренування використовується частина даних, виділена для навчання (TrainSet), яка була отримана в результаті розбиття початкового набору даних на навчальний та тестовий. Результатом тренування є «trainedModel», який представляє навчену модель. Ця модель може використовуватися для здійснення прогнозів або подальшої оцінки її ефективності.

Процес оцінки є важливою частиною розробки моделі, оскільки він дозволяє зрозуміти, наскільки добре модель працює на даних, які вона раніше не бачила, і визначити, чи потрібно покращувати модель перед її використанням у реальних умовах (рис. 3.19).

```
// Оцінювання моделі
var predictions = trainedModel.Transform(splitData.TestSet);
var metrics = context.Regression.Evaluate(predictions,
  labelColumnName: "Label",
  scoreColumnName: "Score");

// Виведення метрик якості моделі
RaportTBox.Text += "Метрики якості моделі" + "\r\n";
RaportTBox.Text += ($"R^2 (Coefficient of Determination): {metrics.RSquared:0.##}") + "\r\n";
RaportTBox.Text += ($"MAE (Mean Absolute Error): {metrics.MeanAbsoluteError:#.##}") + "\r\n";
RaportTBox.Text += ($"MSE (Mean Squared Error): {metrics.MeanSquaredError:#.##}") + "\r\n";
RaportTBox.Text += ($"RMSE (Root Mean Squared Error): {metrics.RootMeanSquaredError:#.##}") + "\r\n";
RaportTBox.Text += ($"Loss Function: {metrics.LossFunction:#.##}") + "\r\n";
```

Рисунок 3.19 – Оцінювання моделі та виведення результатів оцінки

Використовуючи навчену модель (trainedModel), здійснюється перетворення тестового набору даних (splitData.TestSet). Цей процес генерує прогнози для

тестових даних. Отримані прогнози потім оцінюються за допомогою вбудованої функції оцінки, яка аналізує якість моделі. Для цього визначаються стовпці з мітками і прогнозованими значеннями.

Отримані метрики якості виводяться у текстове поле на інтерфейсі користувача. Це включає різні статистичні показники, які відображають точність та надійність моделі. Виведені метрики включають:

– коефіцієнт детермінації (R^2), який показує, наскільки добре модель відтворює залежності в даних;

– середня абсолютна помилка (MAE), що відображає середнє відхилення прогнозів від фактичних значень;

– середня квадратична помилка (MSE) та її квадратний корінь (RMSE), які є мірами відхилень прогнозів від фактичних значень;

– функція втрат, яка вказує на загальну ефективність моделі.

Для збереження даних моделі реалізовано спеціальний метод, що дозволяє згодом використовувати модель для прогнозування (рис. 3.20).

```
private void SaveBtn_Click(object sender, EventArgs e) {
  if (IsDatasCorrect()) {
    //Save model
    string pathName = @"\teach\" + GenerateFileName() + ".zip";
    string localProj =
      System.IO.Path.GetDirectoryName(System.Reflection.Assembly.GetExecutingAssembly().Location);
    _NeuroProvider.InsertNeuro(NeuralNamesTBox.Text, pathName);
    context.Model.Save(trainedModel, dataFromFilesView.Schema, localProj + pathName);
    ClearAllData();
    _LoggerProvider.InsertLogger("Було навчено нейронну мережу "
      + NeuralNamesTBox.Text, DateTime.Now);
    MessageBox.Show("Дані успішно збережено!");
  }
}
```

Рисунок 3.20 – Код методу для збереження результатів навчання

Спочатку виконується перевірка коректності введення даних за допомогою методу «IsDatasCorrect». Якщо дані введені правильно, процес продовжується, при цьому генерується шлях для збереження моделі. Для цього використовується метод «GenerateFileName», який створює унікальне ім'я файлу, до якого додається розширення ".zip".

Інформація про модель вноситься в систему управління даними за допомогою методу «InsertNeuro», який отримує назву моделі та шлях до файлу. Модель

зберігається у файлі за визначеним шляхом за допомогою методу «Save» контексту. Після цього виконується очищення даних в інтерфейсі користувача через метод «ClearAllData». У журнал подій вноситься запис про навчання моделі через «_LoggerProvider», а користувачу виводиться повідомлення про успішне збереження моделі.

Також була створена форма "Тестування моделей" для перевірки ефективності навчених моделей. Її інтерфейс ілюстровано на рис. 3.21.

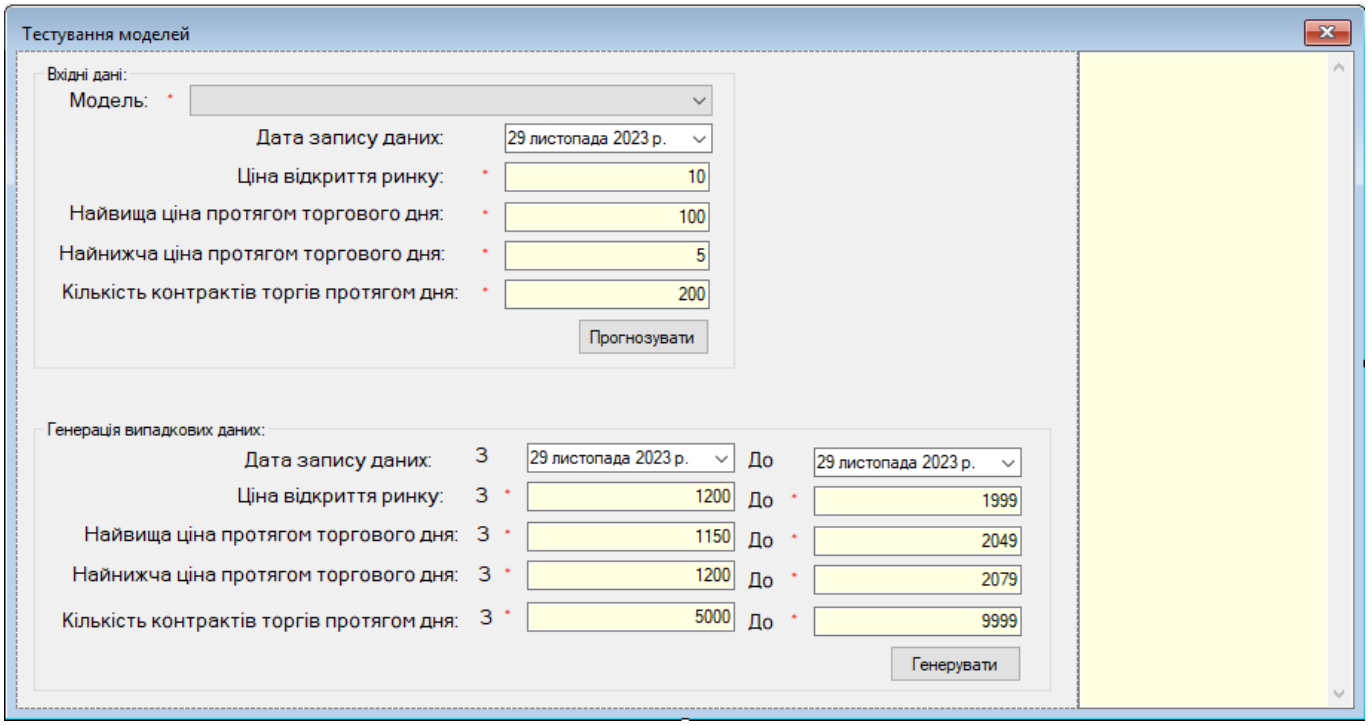

Рисунок 3.21 – Форма для тестування моделей

При завантаженні форми у її конструкторі викликається метод «LoadAllDate», забезпечуючи користувачам необхідні засоби для взаємодії з системою, зокрема для вибору та аналізу нейронних мереж (рис. 3.22).

```
private void LoadAllDate() {
  _NeuroL = _NeuralProvider.GetAllNeuro();
  NeuroCBox.DataSource = _NeuroL;
  NeuroCBox.ValueMember = "NeuroId";
  NeuroCBox.DisplayMember = "NeuroNames";
  _IsThemesLoad = true;
  DateMinDTP.Value = DateTime.Now;
  DateMaxDTP.Value = DateTime.Now.AddYears(10);
  NeuroCBox_SelectedValueChanged(NeuroCBox, EventArgs.Empty);
}
```

Рисунок 3.22 – Код методу «LoadAllDate»

Спочатку метод звертається до провайдера даних «_NeuralProvider», щоб отримати повний список нейронних мереж. Цей список зберігається в змінній «_NeuroL». Далі цей список використовується як джерело даних для випадаючого списку «NeuroCBox». Це означає, що елементи, які відображатимуться у випадаючому списку, беруться зі змінної «_NeuroL».

Встановлюються властивості «ValueMember» та «DisplayMember» для випадаючого списку. «ValueMember» визначає, яке поле з даних буде використовуватися як значення елемента, а «DisplayMember» - яке поле буде відображатися користувачу. Тут для ідентифікації мереж використовується поле "NeuroId", а назви мереж для відображення взяті з "NeuroNames".

Змінна «_IsThemesLoad» встановлюється в положення «true», що може вказувати на завершення завантаження даних. Дати в елементах вибору дати (DateMinDTP та DateMaxDTP) встановлюються в поточну дату та дату через 10 років відповідно. У кінці, викликається метод «NeuroCBox_SelectedValueChanged», щоб ініціалізувати відповідну реакцію на зміну вибору в випадаючому списку.

У даній формі реалізовано 2 основні функції: для прогнозування ціни на дорогоцінні метали за допомого введення даних користувачем та з можливістю генерації випадкових даних у заданому діапазоні.

Для можливості генерації даних випадковим чином реалізовано подію «GenBtn_Click», що запускає генерацію випадкових даних (рис. 3.22).

```
private void GenBtn_Click(object sender, EventArgs e) {
    if (IsGenDataCorrect()) {
        if (timer1.Enabled) {
            timer1.Enabled = false;
            GenBtn.Text = "Генерувати";
            _LoggerProvider.InsertLogger("Запущено генератор для моделі '" +
            NeuroCBox.Text + "'", DateTime.Now);
        } else {
            timer1.Enabled = true;
            GenBtn.Text = "Зупинити";
        }
    }
}
```

Рисунок 3.22 – Код події запуску генерації випадкових даних

Метод дозволяє користувачам контролювати процес генерації даних, зупиняти та запускати його за потреби, а також забезпечує зворотний зв'язок через інтерфейс

користувача та систему документування. Перш за все, він перевіряє, чи введені дані для генерації є коректними, використовуючи метод «IsGenDataCorrect». Якщо дані коректні, код перевіряє, чи активний таймер timer1. Якщо таймер активний (що означає, що процес генерації даних вже запущений), він вимикається, і текст кнопки змінюється на "Генерувати", що вказує на можливість повторного запуску генерації. В цьому випадку також вноситься запис до системи документування про запуск генерації даних з використанням вибраної моделі. Якщо таймер не активний, він увімкнується, і текст кнопки змінюється на "Зупинити", що вказує на можливість зупинки процесу генерації.

На рис. 3.23 представлено метод, що служить для генерації випадкових даних для цінних металів, відповідно до заданих параметрів у системі. Метод є корисним для симуляції реальних даних, дозволяючи тестувати та аналізувати моделі з використанням реалістичних, але контрольованих умов.

```
public CommoditieData GenerateCustomGoldData(
    DateTime dateMin, DateTime dateMax,
    double openMin, double openMax,
    double highMin, double highMax,
    double lowMin, double lowMax,
    double volumeMin, double volumeMax,
    Random rnd) {
  DateTime date = RandomDate(dateMin, dateMax, rnd);
  double closeMin = 1120.0;
  double closeMax = 1999.0;
  double open = RandomDouble(openMin, openMax, rnd);
  double close = RandomDouble(closeMin, closeMax, rnd);
  double high = Math.Max(open, close) + RandomDouble(highMin, highMax, rnd);
  double low = Math.Min(open, close) + RandomDouble(lowMin/1.2, lowMax/ 1.2, rnd);
  double volume = RandomDouble(volumeMin, volumeMax, rnd);

  return new CommoditieData {
    Date = date.ToString("yyyy-MM-dd"), Open = (float)open,
    High = (float)high, Low = (float)low, Volume = (float)volume
  };
}
```

Рисунок 3.23 – Код методу для генерації випадкових даних

Для генерації випадкових даних також використовується допоміжний метод «RandomDate», що створює випадкової дати в межах вказаного діапазону (між dateMin та dateMax). Також генеруються такі дані як: ціна відкриття ринку, найвища

ціна протягом торгового дня, найнижча ціна протягом торгового дня та кількість укладених контрактів торгів протягом дня.

Щоб забезпечити користувачеві можливість отримати швидкий прогноз на основі введених даних реалізовано метод «PredictBtn_Click», код якого представлено на рис. 3.24.

```
private void PredictBtn_Click(object sender, EventArgs e) {
  if (IsAllCommoditieDataCorrect()) {
    var prediction = predictionEngine.Predict(new CommoditieData {
      Date = DateDTP.Value.ToShortDateString(),
      Open = (float)Convert.ToDouble(OpenTBox.Text),
      High = (float)Convert.ToDouble(HighTBox.Text),
      Low = (float)Convert.ToDouble(LowTBox.Text),
      Volume = (float)Convert.ToDouble(VolumeTBox.Text)
    });
    var commoditieDataInfo = new StringBuilder();
    commoditieDataInfo.AppendLine("\r\n--- Прогноз ---");
    commoditieDataInfo.AppendLine($"Прогнозована ціна закриття ринку: { prediction.Close}");
    RaportTBox.Text = commoditieDataInfo.ToString();
    _LoggerProvider.InsertLogger("Проведено прогнозування цін для моделі '"
      + NeuroCBox.Text + "'", DateTime.Now);
  }
}
```

Рисунок 3.24 – Код методу прогнозування за допомогою введених

Метод «PredictBtn_Click» активується при натисканні кнопки "Прогноз" та виконує наступні дії:

– перевіряє вхідні дані. Перш за все, виконується перевірка коректності введення даних про товари через метод «IsAllCommoditieDataCorrect»;

– створює об'єкт для прогнозування. Якщо дані введені коректно, створюється новий об'єкт «CommoditieData» з даними, введеними користувачем. У цей об'єкт вносяться значення: дата (взята з елемента вибору дати і перетворена в текстовий формат), ціни відкриття, найвища та найнижча ціни, а також об'єм. Всі ці значення конвертуються з текстових полів у числовий формат (тип float);

– прогноз. За допомогою «predictionEngine» виконується прогноз на основі введених даних. Отриманий результат прогнозу містить прогнозоване значення ціни закриття ринку;

– формування та відображення результатів прогнозу. Результати прогнозування додаються до конструктора рядків StringBuilder. У цьому конструкторі формується повідомлення про результати прогнозу, яке включає

прогнозовану ціну закриття. Після формування цей рядок виводиться в текстовому полі на формі для інформування користувача.

– документування дії. В систему фіксується запис про виконане прогнозування, у якому вказується назва моделі та час проведення прогнозу.

3.5 Аналіз методів тестування та налагодження

3.5.1 Аналіз методів тестування та відлагодження

У ході аналізу способів прогнозування показників економіки дорогоцінних металів на фондовому ринку виокремлюються два ключові методи тестування: метод покриття операторів та метод базований на припущенні помилок.

Метод покриття операторів фокусується на визначенні тих частин програми, які активуються під час тестів. Він оцінює, яку частку коду або окремих операторів вдалося виконати під час тестування, прагнучи охопити максимально можливу кількість коду. Цей метод ефективно виявляє сегменти коду без тестового покриття, які можуть приховувати невиявлені дефекти, але він не завжди може гарантувати повне виявлення всіх помилок, оскільки наявність покриття не завжди відповідає логічній вірності коду.

Тестування на основі припущення про помилки, у свою чергу, засноване на ідеї, що багато помилок у програмуванні виникають через стандартні неправильні припущення розробників або типові помилки. Тести розробляються з метою специфічно виявляти такі помилки. Цей метод дає змогу більш цілеспрямовано знаходити потенційно слабкі місця в коді, але потребує глибокого розуміння поширених програмних помилок та особливостей конкретного проекту.

Обидва ці методи відіграють ключову роль у процесі тестування, проте їх ефективність залежить від контексту та особливостей програмного продукту. Зазвичай оптимальний підхід включає використання цих та інших методів тестування для всебічного огляду та виявлення можливих помилок у програмному забезпеченні.

3.5.2 Тестування системи методом покриття операторів

Тестування програмного забезпечення є фундаментальною складовою процесу розробки, забезпечуючи впевненість у коректній роботі коду і його безпеці для використання. У рамках цього проекту особливий акцент був зроблений на юніт-тестуванні, яке вважається одним з найефективніших методів аналізу окремих елементів програми. Завдяки використанню бібліотеки MSTestv, розроблено низку тестів, які охоплюють ключові аспекти системи. Тести спрямовані на перевірку роботи бізнес-логіки, включаючи точність роботи алгоритмів обробки даних. Це дозволяє гарантувати, що критичні функції системи відповідають бізнес-вимогам і функціонують без помилок.

Рис. 3.25 відображає результати проведення модульного тестування.

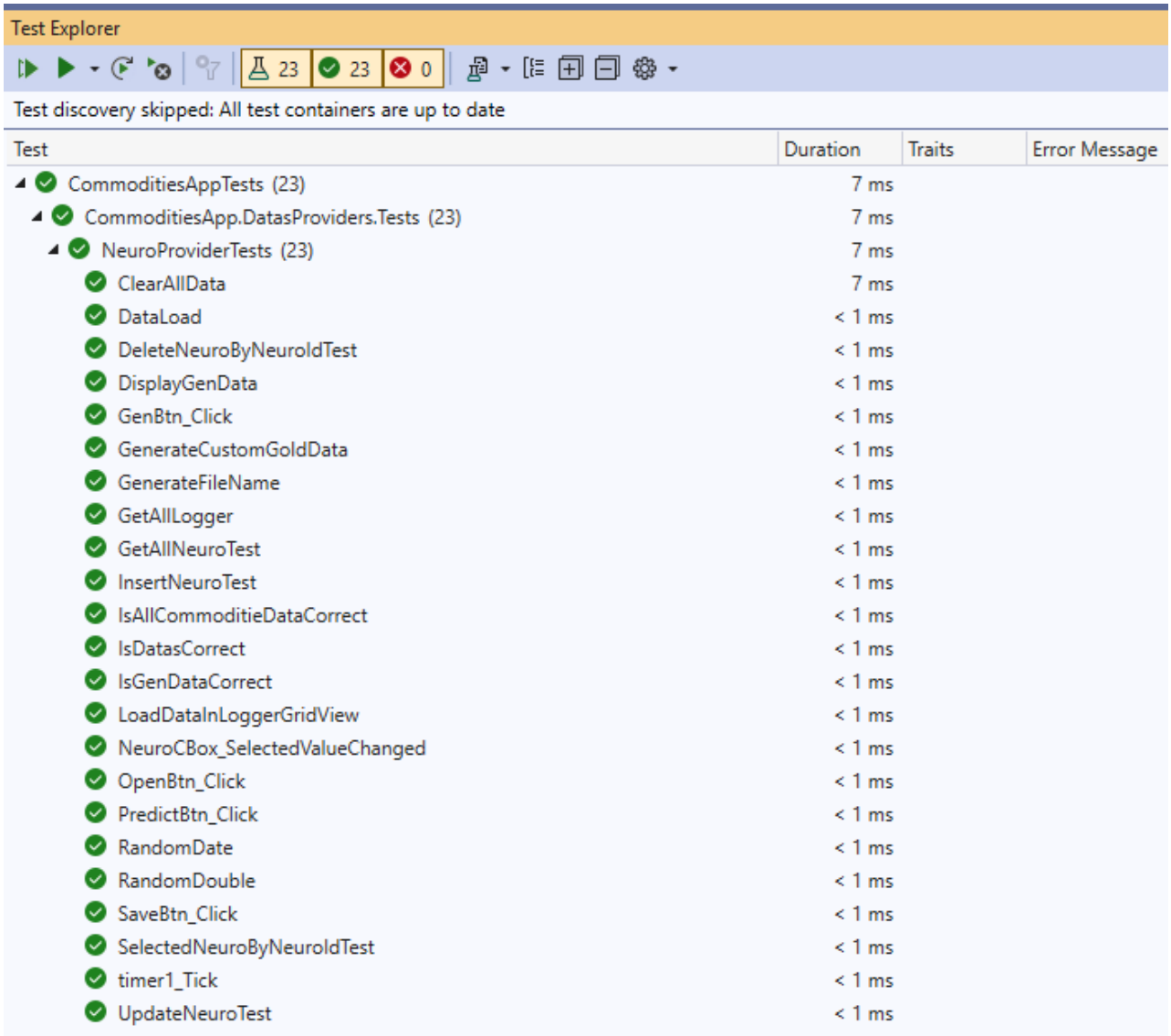

Рисунок 3.25 – Результати модульного тестування системи

Застосування цих методик тестування дозволило досягти ґрунтовного і всестороннього аналізу системи, результати якого були ретельно зібрані та проаналізовані.

3.5.3 Тестування системи методом припущення про похибку

У рамках тестування методом припущення про похибку створено тестові сценарії, які імітують типові помилки, що допускаються в ході розробки:

Сценарій 1. Тестування з помилковим форматом файлу

Мета: Перевірити, як система відреагує на спробу завантаження файлу неправильного формату.

Кроки:

– натиснути на кнопку "OpenBtn" для відкриття вікна вибору файлу;

– вибрати файл, який не є у форматі .csv (наприклад, файл зображення або текстовий файл без розділових знаків);

– спробувати завантажити вибраний файл.

Очікуваний Результат: Система відображає повідомлення про помилку, пов'язану з форматом файлу, і запобігає його завантаженню.

Сценарій 2. Тестування з Некоректними Даними для Навчання Моделі

Мета: Перевірити реакцію системи на спробу навчити модель із неповними або некоректними даними.

Кроки:

– завантажити коректний файл .csv, але з даними, які містять неповні або некоректні записи (наприклад, відсутні значення або неправильний формат дат);

– натиснути на кнопку "SaveBtn" для збереження та навчання моделі з використанням завантажених даних.

Очікуваний Результат: Система відображає помилку, пов'язану з некоректними даними, і не дозволяє завершити процес навчання моделі.

Сценарій 3. Тестування з некоректним вводом даних для прогнозування.

Мета: Перевірити, як система реагує на введення некоректних даних для прогнозування.

Кроки:

– вибрати модель з випадаючого списку NeuroCBox;

– ввести некоректні або неповні дані в полях введення (наприклад, ввести текст замість чисел у полях "Open", "High", "Low", "Volume");

– натиснути кнопку "Прогнозувати" для здійснення прогнозу.

Очікуваний Результат: Система показує повідомлення про помилку введення даних та забороняє продовження прогнозування.

Сценарій 4. Тестування реакції на відсутність завантаженої моделі

Мета: Перевірити поведінку системи, коли спроба прогнозування здійснюється без попереднього завантаження моделі.

Кроки:

– переконатися, що жодна модель не була завантажена (наприклад, не натискати на будь-яку модель у NeuroCBox або не завантажувати модель);

– ввести коректні дані в полях введення для прогнозування;

– натиснути кнопку " Прогнозувати ".

Очікуваний результат: Система відображає повідомлення про помилку, інформуючи, що модель для прогнозування не завантажена або не вибрана, та блокує процес прогнозування.

Проведені сценарії допомогли перевірити здатність системи впоратися з нестандартними та неочікуваними ситуаціями, що забезпечило стабільність і надійність в реальних умовах використання.

Висновки до розділу 3

У рамках даного розділу здійснено комплексну розробку програмного забезпечення для нейромережевого прогнозування економічних показників дорогоцінних металів фондового ринку. Основні етапи роботи включали формалізацію задачі прогнозування, вибір методології, аналіз вхідних даних та їх подальшу обробку, а також розробку та оцінку моделі. Використано нейромережеві технології для аналізу історичних цінових трендів і макроекономічних індикаторів, що впливають на ціни дорогоцінних металів. Застосування методу швидкого лісу

дозволило ефективно вирішити задачу короткострокового прогнозування, використовуючи випадкові підмножини навчальних даних і метрики якості, такі як коефіцієнт детермінації R^2, середня абсолютна та квадратична помилки (*MAE*, *MSE*). Ці підходи забезпечили високу точність прогнозування і стали основою для розробки базової архітектури системи прогнозування та її внутрішнього проектування, що включало вибір мови програмування та технологічної платформи, ієрархію та взаємодію класів. Результати цієї роботи є фундаментом для наступного розділу, де буде виконано практичне застосування розробленого програмного продукту.

4 ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ НЕЙРОМЕРЕЖЕВИХ МОДЕЛЕЙ

4.1 Підготовка до експерименту

4.1.1 Опис використаного програмно-апаратного середовища

Для здійснення експериментальної частини дослідження було створено комплексне програмно-апаратне середовище, що охоплює важливі компоненти, а саме:

– комп'ютерна платформа. Експерименти виконувались на настільному комп'ютері з оновленим обладнанням: процесором Intel Core i7 останнього покоління, 32 ГБ оперативної пам'яті для забезпечення підвищеної продуктивності та SSD диском на 1 ТБ, що сприяє швидкому доступу до даних і зберіганню великих обсягів інформації. Ці параметри гарантують високу обчислювальну потужність, необхідну для обробки складних даних та проведення ресурсомістких експериментів;

– операційна система. Використання оновленої версії операційної системи Windows 10 Pro, яка забезпечує вдосконалене середовище для розробки, тестування та експлуатації програмного забезпечення, включаючи підтримку новітніх технологій та інструментів;

– розробка та тестування. Застосування Microsoft Visual Studio 2022, яке є одним із найбільш розширених інструментів для розробки на мові C#. Це середовище розробки надає розширені можливості для написання, відлагодження та тестування коду, дозволяючи ефективно вирішувати завдання машинного навчання та аналізу даних;

– навчання моделей та аналіз даних: Застосування бібліотеки Microsoft ML.NET, що дозволяє реалізовувати складні алгоритми навчання моделей в C#-програмах. Бібліотека надає засоби для створення, тренування, тестування та використання моделей, підвищуючи ефективність досліджень;

– база даних. Використання Microsoft SQL Server, який забезпечує надійне та ефективне управління великими обсягами даних, необхідних для комплексного аналізу та прогнозування.

Вибір цього середовища був обумовлений його високою надійністю, гнучкістю та можливостями для розширеного аналізу, що створює оптимальні умови для проведення експериментів та одержання точних і об'єктивних результатів дослідження.

4.1.2 Опис підходу для визначення для визначення точності та повноти прогнозування

Для оцінки точності та повноти прогнозування з використанням алгоритму швидкого дерева, можна застосувати ряд метрик якості моделі. Кожна метрика надає унікальне розуміння про ефективність та точність моделі. Ось детальний опис цих метрик:

– R^2 (Коефіцієнт детермінації). Ця метрика вимірює, наскільки варіації в залежній змінній можуть бути пояснені за допомогою незалежних змінних у моделі. Значення R^2 варіюється від 0 до 1. Чим ближче значення до 1, тим краще незалежні змінні пояснюють варіацію залежної змінної. Високе значення R^2 означає, що модель добре "пояснює" варіації у вихідних даних;

– MAE (Середня абсолютна помилка). MAE вимірює середню абсолютну різницю між фактичними та прогнозованими значеннями. Це дає уявлення про середню величину помилок у прогнозах. Менше значення MAE свідчить про більшу точність моделі;

– MSE (Середньоквадратична помилка). MSE вимірює середню квадратичну різницю між фактичними та прогнозованими значеннями. Вона карає за більші помилки, оскільки помилки підносяться до квадрату перед усередненням. Низькі значення MSE вказують на вищу точність моделі;

– RMSE (Корінь з середньоквадратичної помилки). RMSE - це квадратний корінь з MSE. Вона має ті ж одиниці вимірювання, що й залежна змінна, і часто використовується для інтерпретації помилок у прогнозуванні. Як і MSE, RMSE

карає за більші помилки, а нижчі значення RMSE свідчать про кращу точність моделі;

4.1.3 Вплив налаштувань методу навчання на результати вимірювання

Здійснення налаштувань методу навчання мало суттєвий вплив на результати вимірювання точності та ефективності прогнозувальних моделей. У ході роботи було обрано оптимальну кількість дерев у випадковому лісі, забезпечуючи необхідний баланс між точністю моделі та обчислювальними витратами. Виявлено, що підвищення кількості дерев зменшує варіативність прогнозів та помилки перенавчання.

Було також встановлено, що глибина кожного дерева суттєво впливає на продуктивність моделі. В ході експериментів з'ясовано, що дерева з оптимальною глибиною ефективно вловлюють складні закономірності в даних, мінімізуючи ризик перенавчання.

Налаштування параметрів вибірки даних, включаючи розмір підвибірки для кожного дерева, дозволило створити дерева з достатньою різноманітністю, знижуючи ризик перенавчання та підвищуючи загальну здатність моделі до узагальнення на нових даних.

Параметри навчання, такі як швидкість навчання та кількість епох, були вибрані таким чином, щоб забезпечити ефективний баланс між швидкістю досягнення високої точності та ризиком пропуску оптимальних рішень. Довготривалі епохи навчання дозволили глибше дослідити простір параметрів, забезпечуючи високу точність моделі. Налаштування для обробки викидів та нормалізації даних були ретельно опрацьовані, що значно підвищило продуктивність моделі. Це дозволило максимально використати потенціал даних, зменшивши вплив аномальних записів та відмінностей в масштабах даних.

Таким чином, адекватний вибір та налаштування параметрів методу навчання стали ключовими факторами, що забезпечили високу точність та ефективність розроблених моделей машинного навчання, дозволяючи добитися точних та надійних прогнозів.

4.1.4 Аналіз впливу різних факторів на точність моделі

В ході підготовчих робіт до експерименту було здійснено цілеспрямовані кроки для зниження впливу зовнішніх чинників на результати дослідження. Зокрема, було забезпечено стабільність обчислювального середовища, утримуючи експеримент на консистентній апаратній платформі з незмінною конфігурацією, що виключило коливання в обчислювальних ресурсах. Додатково, всі програмні компоненти були оновлені до останніх версій, мінімізуючи ризики, пов'язані з програмними помилками та нестабільністю.

Контроль версій використовуваних бібліотек та залежностей, включаючи ML.NET, був строго регламентований, фіксуючи специфічні версії, щоб уникнути неочікуваних змін у функціональності системи. Це дозволило виключити можливість виникнення проблем, пов'язаних з оновленнями або змінами в залежностях.

Моніторинг фізичних умов в приміщенні, де розташовувалась обчислювальна техніка, був проведений для виключення впливу температури та вологості на апаратне забезпечення. Водночас, було здійснено стандартизацію вхідних даних, що включала їх ретельну підготовку та нормалізацію, забезпечуючи однорідність та зниження впливу випадкових відхилень.

Обрання репрезентативного набору даних для тренування моделі забезпечило її ефективність в реальних умовах використання та різноманітних сценаріях. Також, було виконано попередні тестування системи перед основним експериментом для виявлення та виправлення можливих проблем, що могли негативно вплинути на результати.

Ці заходи були направлені на забезпечення максимальної достовірності та відтворюваності результатів дослідження з метою прогнозування помилок у програмному коді, знижуючи вплив зовнішніх чинників на якість та точність експерименту.

4.2 Проведення експерименту

4.2.1 Реалізація та тренування нейромережевих моделей

У рамках дослідження був виконаний процес скачування та підготовки датасету з сайту Kaggle для аналізу цін на дорогоцінні метали [53]. Спочатку було здійснено вхід у користувацький акаунт на Kaggle та знайдено відповідний датасет, який містить історичні дані про ціни на золото, срібло, платину та паладій (рис. 4.1). Після завантаження файлу датасету у форматі CSV, він був розархівований та готовий до подальшої обробки.

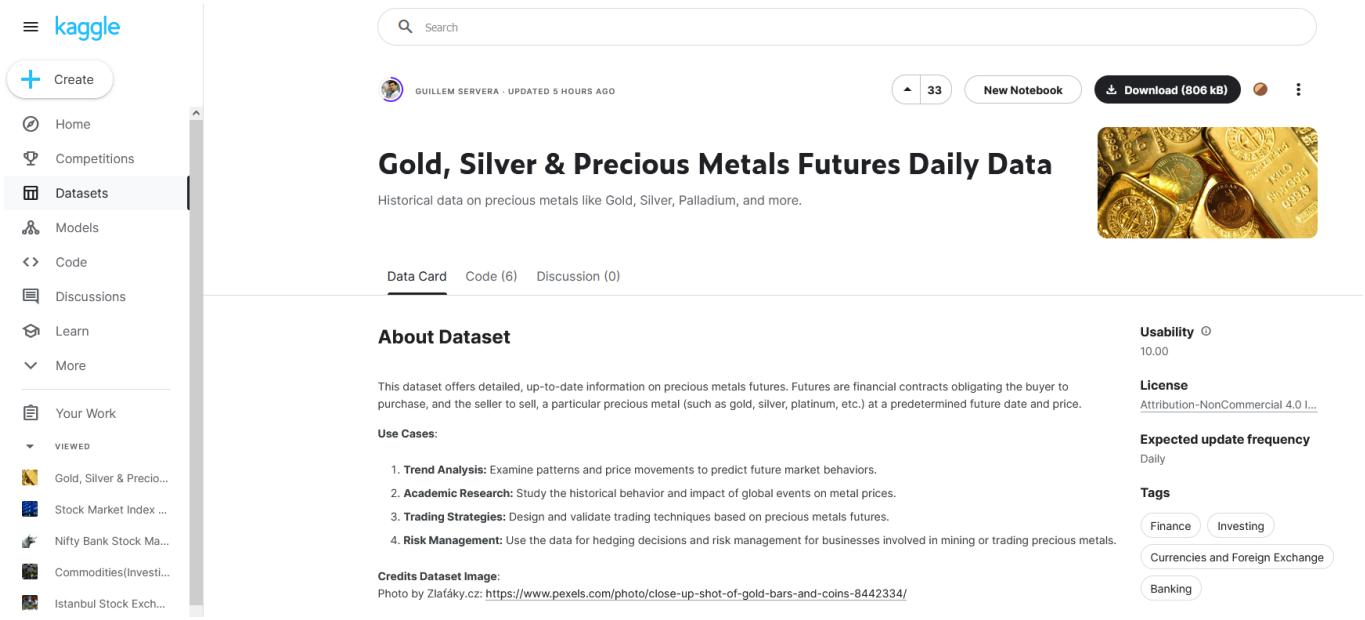

Рисунок 4.1 – Скачування датасету з сайту Kaggle

Було проведено первинний аналіз датасету, зокрема, перевірку правильності форматування, заголовків та виявлення можливих аномалій чи відсутніх значень. Далі, використовуючи інструменти Microsoft Excel, датасет було розділено на чотири окремі частини, кожна з яких містить дані виключно по одному типу металу - золоту, сріблу, платині та палладію.

Кожен з отриманих датасетів був ретельно перевірений на предмет повноти даних та точності розділення. Були також виконані додаткові кроки обробки даних, включаючи нормалізацію, видалення дублікатів та обробку відсутніх значень, щоб забезпечити високу якість даних для подальшого аналізу.

Таким чином, було забезпечено високу якість підготовки датасетів, що містять цінну інформацію про ціни на різні види дорогоцінних металів, готових до використання у подальших етапах дослідження.

На рис. 4.2 можна побачити частину інтерфейсу, який демонструє етапи підготовки даних для навчання моделі прогнозування цін на золото. Цей інтерфейс візуально відображає вхідні дані, які були ретельно підготовані та оптимізовані для їх подальшого використання у моделі.

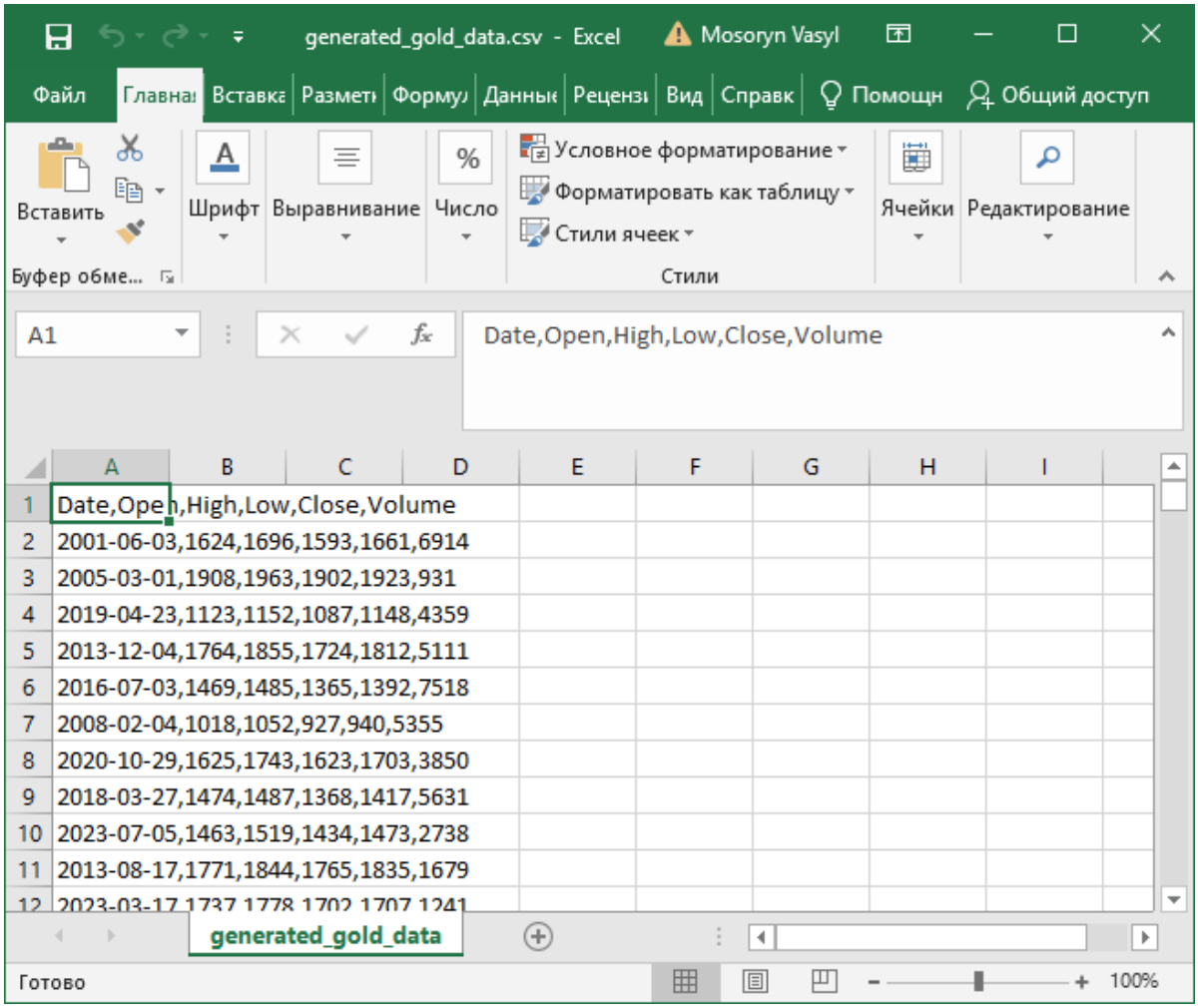

Рисунок 4.2 – Фрагмент із підготовленими даними

Після цього було проведено оцінку даних із датасету для числових міток, таких як: розподіл цін відкриття ринку, розподіл максимальних цін, розподіл мінімальних цін та розподіл цін закриття ринку (рис. 4.3–4.6).

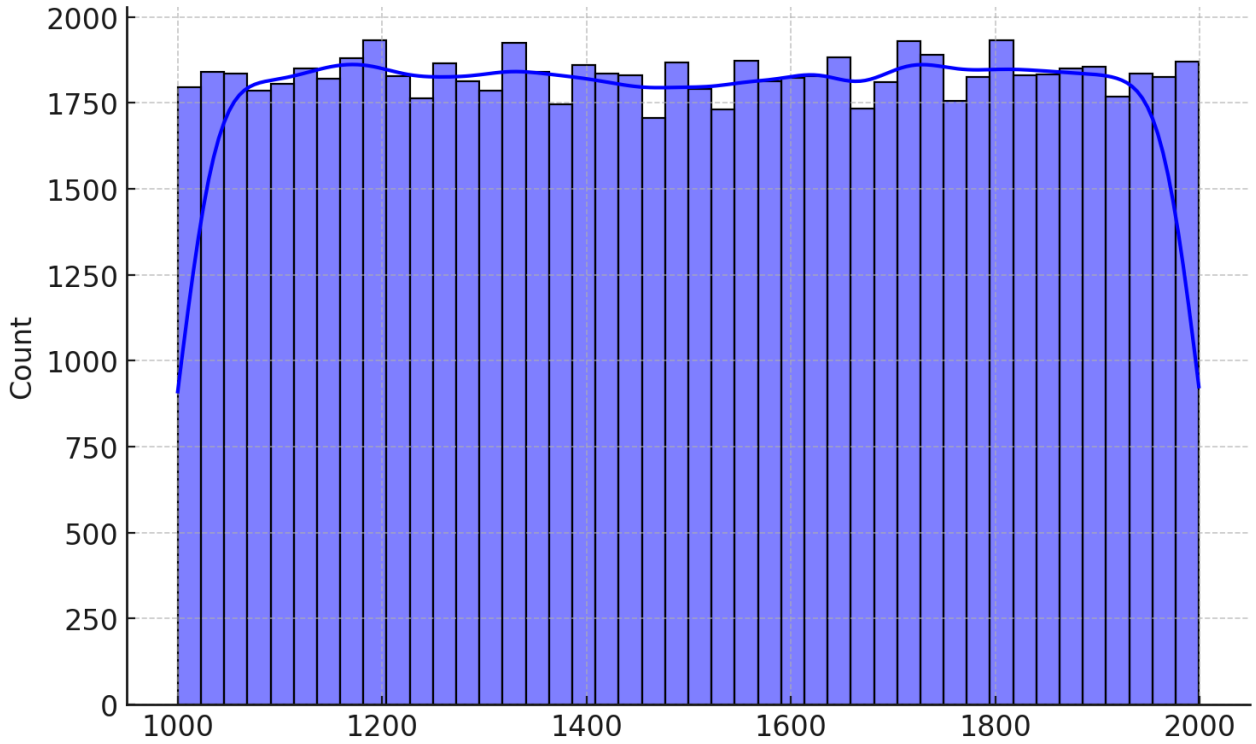

Рисунок 4.3 – Розподіл цін відкриття ринку

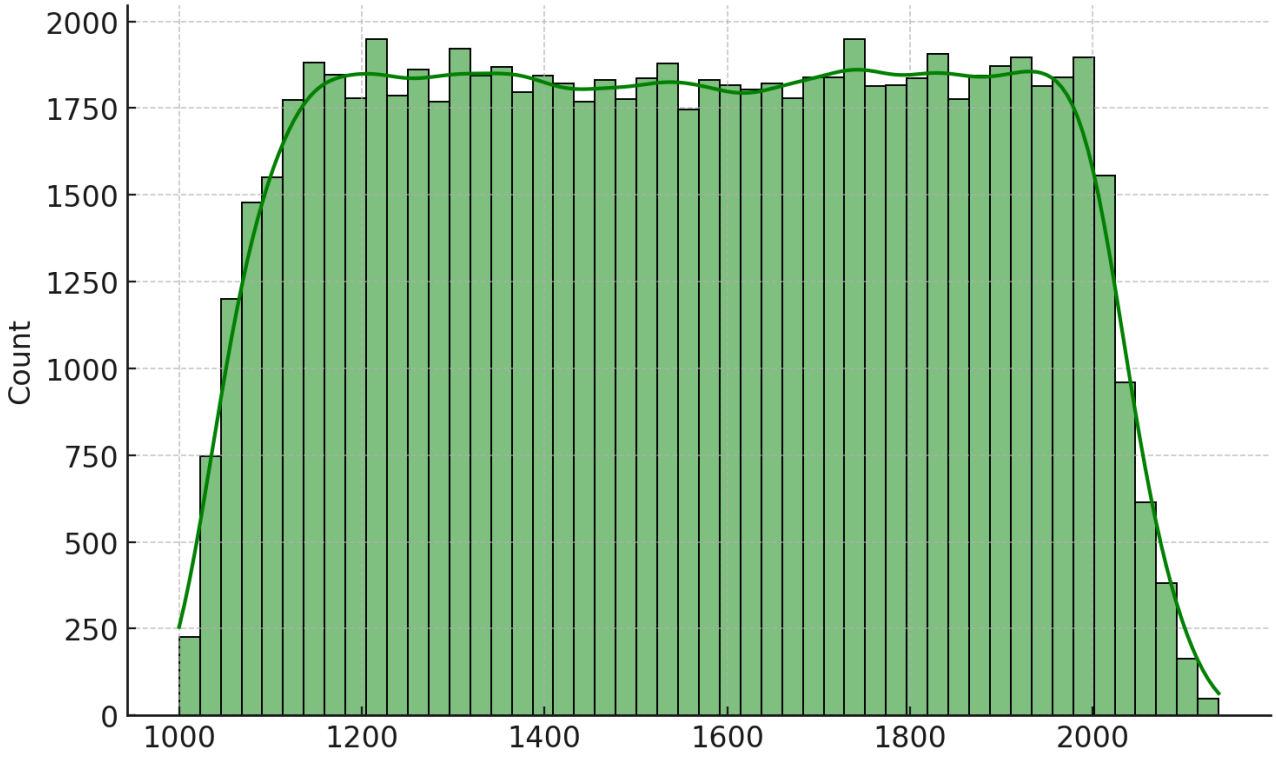

Рисунок 4.4 – Розподіл максимальних цін

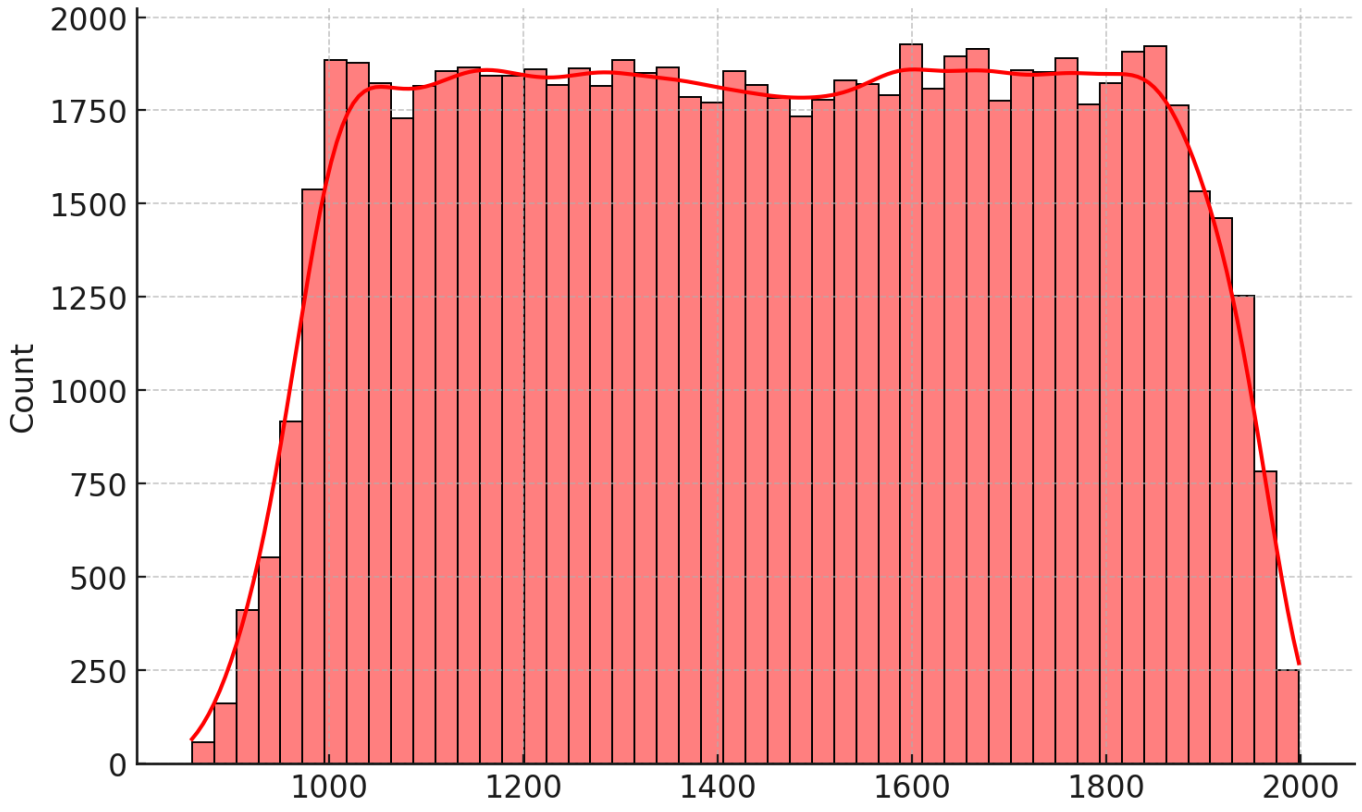

Рисунок 4.5 – Розподіл мінімальних цін

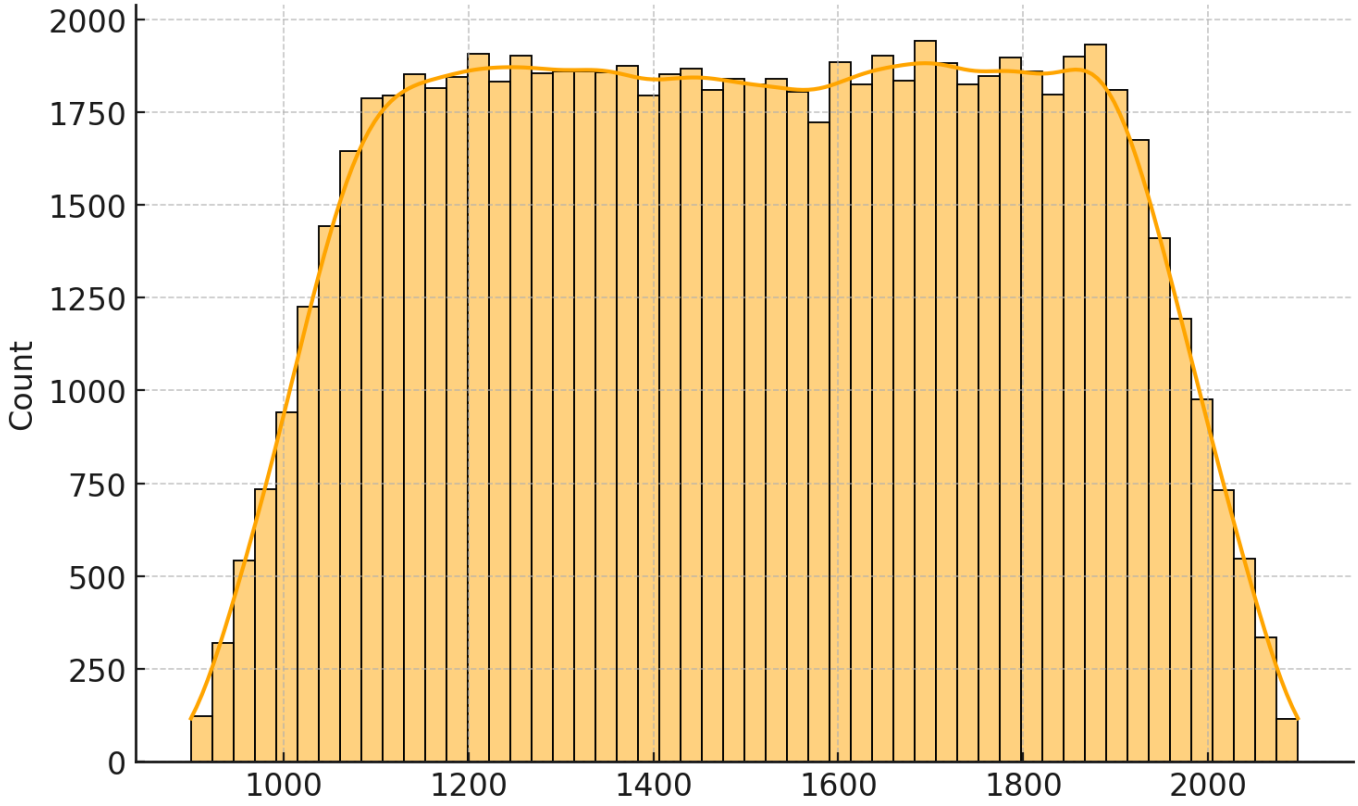

Рисунок 4.6 – Розподіл цін закриття ринку

В результаті проведеного аналізу датасету для прогнозування цін на золото можна зробити наступні висновки:

– розподіл цін. Візуальний аналіз розподілу цін відкриття (Open), максимуму (High), мінімуму (Low) та закриття (Close) показує, що датасет має розподіл, що схожий на нормальний для кожної з цих цін. Це дозволяє припустити, що дані не мають значних викидів або аномалій;

– волатильність. Середня волатильність, обрахована як різниця між максимальною та мінімальною цінами, становить близько 98.87. Це показує, що в датасеті присутня значна волатильність, що є типовим для ринку дорогоцінних металів.

Ці аспекти датасету важливі для розуміння його характеристик та можливого впливу на точність прогнозування моделі. З огляду на високу волатильність та активність на ринку, важливо забезпечити, що модель може адекватно обробляти такі умови і враховувати ці фактори при прогнозуванні.

Для тренування нової моделі створеного сценарію згідно підготовленого датасету, було здійснено перехід по меню програми «Керування» –> «Тренування моделей» та обрано підготовлений датасет у діалоговому вікні. Після цього процес навчання запустився автоматично. Після навчання метрики оцінки розраховуються автоматично, а результат виводиться на екран у відповідне поле (рис. 4.7).

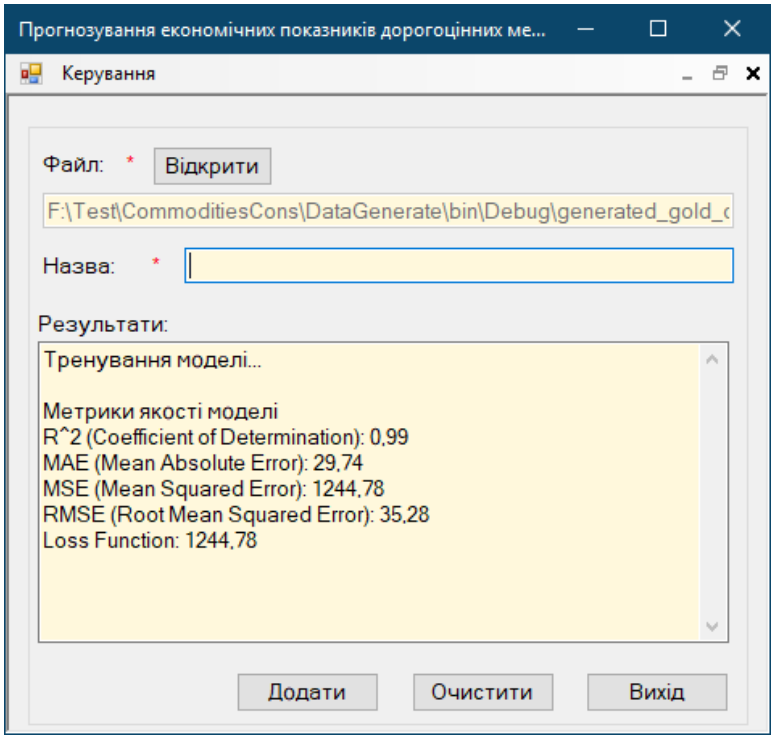

Рисунок 4.7 – Результат навчання НМ

Співставляючи аналіз датасету з метриками оцінки моделі для прогнозування цін на золото, можна стверджувати:

– R^2 (Коефіцієнт детермінації) – 0,99. Високе значення R^2 вказує на те, що модель дуже добре відтворює залежності в даних. Враховуючи, що ціни на золото мають волатильний характер, як це відображено в датасеті, така точність моделі є імпозантною. Однак, існує ризик перенавчання, особливо якщо модель занадто точно відтворює особливості навчального набору даних, що може знижувати її здатність до узагальнення на нових даних;

– MAE (Середня абсолютна помилка) – 29,74. З урахуванням волатильності ринку золота, помилка в 29,74 одиниць може бути відносно мала, що свідчить про хорошу загальну точність моделі. Цей показник також вказує на те, що середнє відхилення між прогнозованими та фактичними цінами є прийнятним з урахуванням реальних умов ринку;

– MSE (Середньоквадратична помилка) і RMSE (Корінь з середньоквадратичної помилки) – 1244,78 та 35,28. Величина MSE та RMSE свідчить про наявність деяких значних відхилень між прогнозами та фактичними значеннями. Це може бути пов'язано з різкими коливаннями цін, які характерні для ринку золота. Високі значення цих метрик можуть вказувати на деякі слабкі місця в моделі, особливо при великих волатильних рухах на ринку;

– функція втрат – 1244,78. Однакові значення MSE та функції втрат вказують на те, що для оптимізації моделі використовувалася середньоквадратична помилка. Це звичайна практика, але важливо забезпечити, що такий підхід відповідає специфіці прогнозування цін на золото.

У цілому, метрики свідчать про високу точність моделі Значення R^2 у 0,99 свідчить про те, що модель відмінно відтворює залежності в даних, що є ключовим для роботи в умовах високої волатильності ринку золота. Незважаючи на деякі великі відхилення, відображені в MSE та RMSE, загальна точність моделі підтверджена низьким значенням MAE у 29,74, що вказує на високу надійність прогнозів.

Після цього навчену модель для прогнозування цін на золото було збережено у системі, для проведення подальшого експериментального дослідження (рис. 4.8).

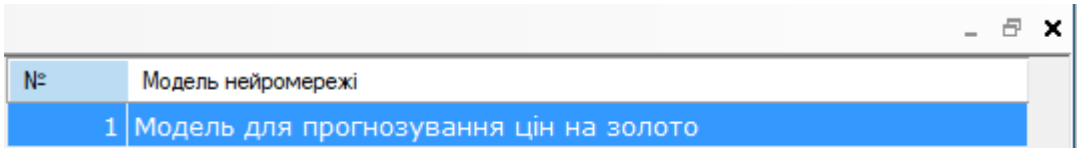

Рисунок 4.8 –Збереження моделі

4.2.2 Методологія експерименту

Експеримент з використанням навченої моделі для прогнозування цін на дорогоцінні метали фондового ринку включав декілька важливих етапів. Спочатку, для доступу до тестування моделі, було виконано перехід через меню "Керування" до опції "Тестування моделей", що відкриває спеціалізовану форму у програмі.

Процес тестування моделі проходив у декілька ключових фаз:

– генерація даних. Використовуючи спеціальний клас GenerateCustomGoldData, програма генерувала набори даних, які імітували різноманітні сценарії та ситуації на фондовому ринку дорогоцінних металів. Це давало змогу моделі оцінювати широкий спектр потенційних варіацій цін;

– прогнозування з використанням моделі. Після генерації даних вони передавалися до навченої моделі. Модель аналізувала вхідні дані, використовуючи власні алгоритми для визначення прогнозованих цін;

– аналіз результатів прогнозування. Отримані результати після обробки даних моделлю піддавалися аналізу. Цей етап включав оцінку точності та надійності прогнозів моделі;

– візуалізація результатів. Результати прогнозування відображалися в інтерфейсі програми, забезпечуючи зручне та інтуїтивно зрозуміле представлення даних. Кожен запис містив інформацію про параметри вхідних даних та отримані прогнози, дозволяючи користувачам оцінити ефективність моделі.

Основною метою проведеного експерименту була не тільки демонстрація здатності моделі виявляти помилки, але й всебічна перевірка її ефективності та надійності у широкому спектрі сценаріїв. Завдання полягало в оцінці точності

прогнозів моделі за допомогою спеціально розробленого генератора сценаріїв, який створював різноманітні умови та випадки для аналізу.

У ході експерименту було згенеровано низку потенційних випадків, кожен з яких містив унікальні характеристики та параметри, що дозволяли випробувати модель у різних обставинах. Це включало варіації у вхідних даних, різні рівні складності сценаріїв та можливі відхилення у поведінці ринку.

Кожен згенерований сценарій був підданий глибокому аналізу з метою визначення, наскільки точно та ефективно модель може виявляти та прогнозувати потенційні помилки. Наприклад, на рис. 4.9 представлений перший згенерований випадок, що демонструє як модель обробляє конкретний набір даних, включаючи її реакцію та виведені результати.

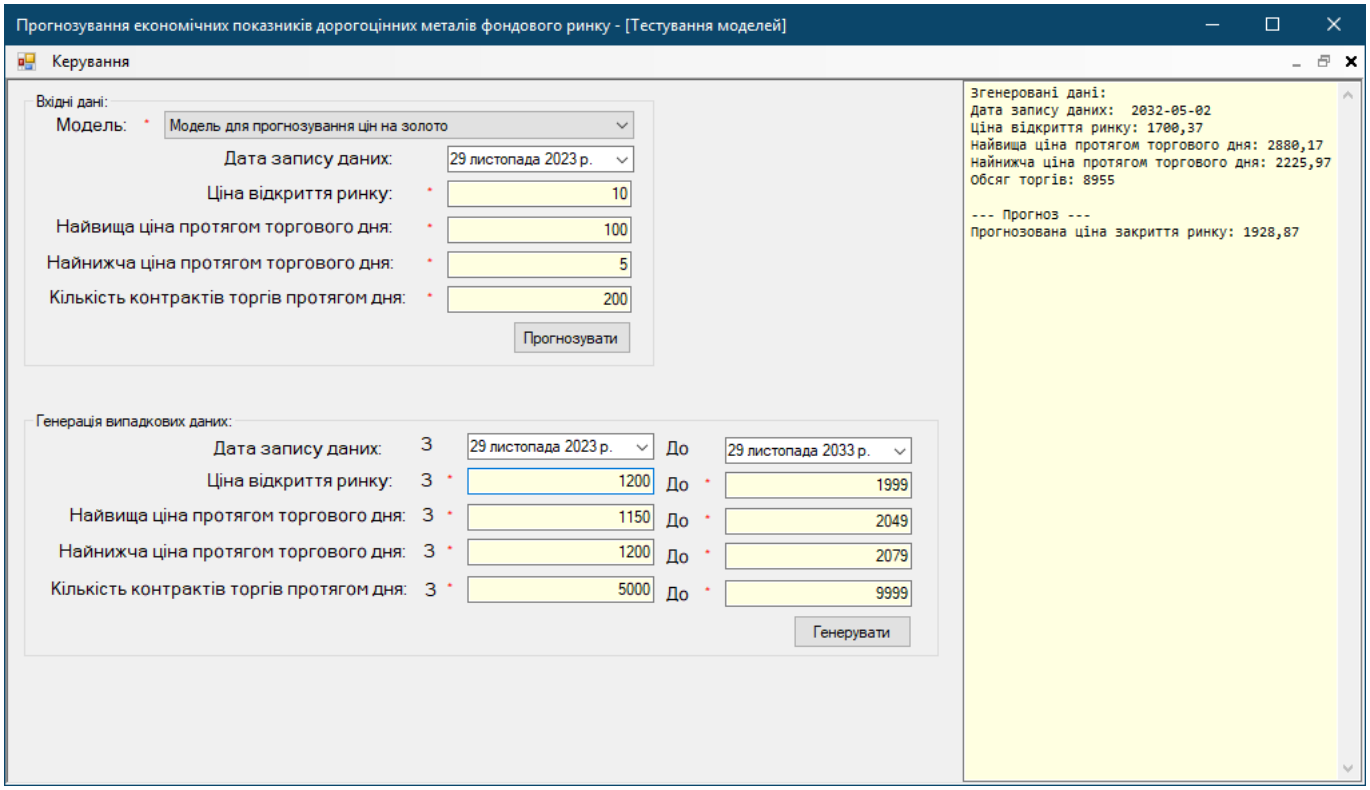

Рисунок 4.9 –Результати прогнозування першого випадку

Аналізуючи цей сценарій, можна зробити наступні висновки:

– дата запису даних: 2023-05-02. Ця дата вказує на майбутній період, що свідчить про здатність моделі прогнозувати на тривалий термін. Важливо врахувати сезонність та історичні тенденції в цінах на золото, щоб зрозуміти, наскільки реалістичним є такий сценарій.

– ціна відкриття ринку: 1700,37. Ціна відкриття є важливим індикатором початкового рівня цін на ринку та може бути порівняна з історичними даними. Початкова ціна 1700,37 вказує на стабільні умови на ринку або на початок торгового дня після періоду зниження цін.

– найвища ціна. 2880,17 та найнижча ціна: 2225,97. Різниця між найвищою та найнижчою ціною вказує на високу волатильність протягом торгового дня. Така велика волатильність може бути викликана різними факторами, включаючи економічні новини, зміни на ринку дорогоцінних металів або політичні події.

– обсяг Торгів: 8955. Високий обсяг торгів може свідчити про значну активність на ринку та інтерес з боку інвесторів. Це також може бути індикатором зростаючої невизначеності на ринку або реакції на важливі новини.

– прогнозована ціна закриття. 1928,87. Прогнозована ціна закриття є нижчою за максимальну ціну та вищою за ціну відкриття, що може вказувати на те, що після періоду значного зростання ціни мають тенденцію до стабілізації.

Загалом, цей згенерований сценарій демонструє здатність моделі прогнозувати ціни на золото в умовах високої волатильності та різноманітних ринкових умов.

На рис 4.10 зображено скріншот результат виконання 2-го випадку.

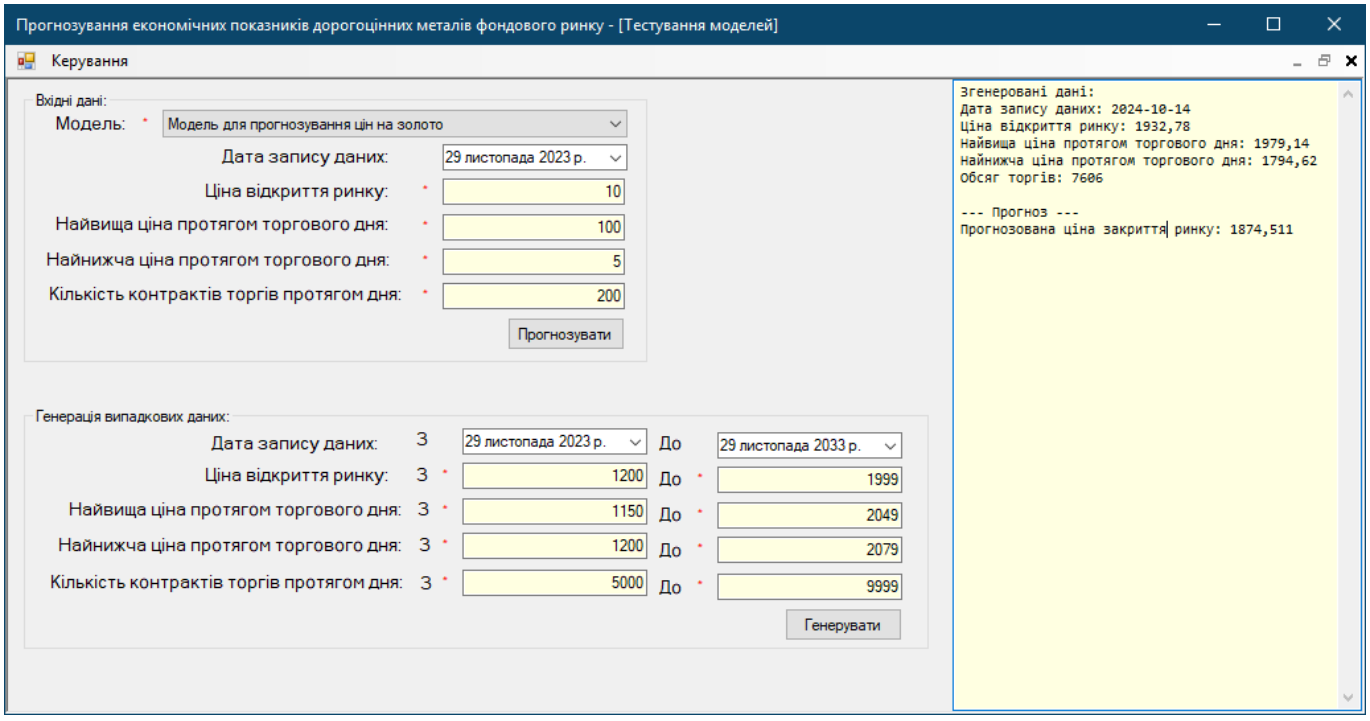

Рисунок 4.10 –Результат прогнозування другого випадку

Аналіз другого згенерованого сценарію для прогнозування ціни на золото включає детальний розгляд згенерованих даних та прогнозу:

– дата запису даних: 2023-10-14. Ця дата вказує на конкретний день у майбутньому, що надає можливість перевірити модель на її здатність прогнозувати в майбутньому періоді;

– ціна відкриття ринку: 1932,78. Ціна відкриття вказує на початковий стан ринку. Значення 1932,78 відображає стабільність зростаючого тренду цін на золото.

– найвища ціна. 1979,14 і найнижча ціна: 1794,62. Різниця між найвищою та найнижчою ціною показує волатильність ринку протягом дня. Волатильність у цьому випадку не є надто значною, але все ж вказує на певні коливання. Найнижча ціна, що є значно нижчою за ціну відкриття, може вказувати на певні негативні тренди або новини, що вплинули на ринок у першій половині торгового дня.

– обсяг торгів: 7606. Обсяг торгів є індикатором активності на ринку. Значення 7606 вказує на помірну активність інвесторів на ринку цього дня.

– прогнозована ціна закриття: 1874,511. Прогнозована ціна закриття ринку є нижчою за найвищу ціну та вищою за найнижчу. Це може свідчити про те, що після певного спаду на початку дня ринок відновився, але не досягнув максимальних показників. Ціна закриття, що є близькою до середини діапазону між найвищою та найнижчою ціною, може відображати загальний баланс між попитом та пропозицією на ринку цього дня.

У цілому, цей сценарій показує, що модель здатна адекватно реагувати на різні ринкові умови та прогнозувати ціни з врахуванням реальних коливань на ринку золота.

4.3 Оцінка результатів проведених експериментів

Оцінка результатів проведених експериментів з прогнозування цін на золото включає декілька ключових аспектів, які допомагають зрозуміти ефективність та точність моделі, а саме:

– адекватність прогнозів. Прогнози моделі здаються реалістичними і відображають відомі властивості ринку золота, такі як волатильність та реакція на

ринкові зміни. Згенеровані сценарії включали різні умови ринку, від високої волатильності до більш стабільних умов, що дозволяє оцінити модель у широкому спектрі ситуацій;

– відповідність прогнозів ринковим тенденціям. Модель здатна прогнозувати ціни, враховуючи різні ринкові умови, що свідчить про її гнучкість та здатність до адаптації. Важливо зазначити, що модель була перевірена на датах у майбутньому, що демонструє її потенціал для прогнозування в довгостроковій перспективі;

– аналіз волатильності та обсягів торгів. Модель адекватно реагує на коливання обсягів торгів та волатильності, що є ключовим для точного прогнозування на ринку золота. Високий обсяг торгів та волатильність в деяких сценаріях вказують на те, що модель здатна реагувати на динамічні зміни на ринку;

– загальна оцінка точності прогнозів. Прогнозовані ціни закриття ринку були у межах реалістичних значень, зважаючи на ціни відкриття, максимальні та мінімальні ціни. Порівняння прогнозованих цін з історичними даними або реальними трендами ринку дозволить додатково оцінити точність та надійність моделі.

– можливості для подальшого вдосконалення. Враховуючи динаміку ринку та потенційні зміни в економічних умовах, існує потенціал для подальшого вдосконалення моделі, зокрема щодо адаптації до несподіваних змін та аномалій. Важливо також розглянути впровадження додаткових факторів, таких як глобальні економічні індикатори та політичні події, які можуть впливати на ринок золота.

В цілому, результати експериментів свідчать про високу ефективність моделі у прогнозуванні цін на золото, демонструючи її здатність адаптуватися до різних ринкових умов та виявляти важливі тенденції.

Висновки до розділу 4

У рамках даного розділу було проведено детальне дослідження ефективності нейромережевих моделей, зосереджене на прогнозуванні цін на дорогоцінні метали.

Для цього було створено комплексне програмно-апаратне середовище, що включає в себе потужний комп'ютер, оновлену операційну систему, використання Microsoft Visual Studio 2022 для розробки, а також Microsoft ML.NET для навчання моделей і Microsoft SQL Server для управління даними.

В ході роботи було проведено детальний аналіз метрик якості моделі, таких як R^2, MAE, MSE, RMSE, що дозволило оцінити точність та повноту прогнозування моделей. Ключовим аспектом було виявлення впливу налаштувань методу навчання на результати, де було встановлено оптимальну кількість дерев у випадковому лісі, глибину дерев та параметри навчання. Це дозволило знизити ризик перенавчання та підвищити загальну здатність моделі до узагальнення на нових даних.

Також було забезпечено стабільність обчислювального середовища та стандартизацію вхідних даних, що включало їх ретельну підготовку та нормалізацію, забезпечуючи однорідність та зниження впливу випадкових відхилень.

В результаті проведених експериментів з прогнозування цін на золото було доведено високу ефективність розробленої моделі, здатності адаптуватися до різних ринкових умов та виявляти важливі тенденції. Отримані результати демонструють потенціал використання нейромережевих моделей для прогнозування в умовах високої волатильності ринку дорогоцінних металів, що дозволяє робити точні та надійні прогнози.

ВИСНОВКИ

У ході виконання магістерської роботи була розроблена система прогнозування, яка використовує передові нейромережеві технології для визначення майбутніх цін на дорогоцінні метали на фондовому ринку України. Ця система включає в себе алгоритм «Швидкі дерева», здатні аналізувати великі обсяги історичних даних, таких як цінові тренди, об'єм торгів та економічні індикатори, щоб робити точні та надійні прогнози. Основною перевагою розробленої системи є її спроможність швидко адаптуватися до змін на ринку та навчатися на базі нових даних, що забезпечує високу точність та актуальність прогнозів. Завдяки цьому інструменту, інвестори матимуть можливість краще розуміти ринкові тенденції та ефективніше управляти своїми інвестиціями у дорогоцінні метали.

У першому розділі було здійснено аналіз теоретичних основ нейромереж, що включав розгляд їхньої структури, принципів роботи та основних типів, таких як згорткові нейромережі, рекурентні нейромережі та глибокі навчальні мережі. Це дало змогу детально зрозуміти, як нейромережі можуть ефективно обробляти та аналізувати складні дані, що є критично важливим для прогнозування цін на фондовому ринку.

Окрім того, було розглянуто специфіку фондового ринку дорогоцінних металів, зокрема, фактори, що впливають на ціни (наприклад, глобальні економічні тренди, політична стабільність, валютні курси), а також особливості торгівлі металами, такі як волатильність та ліквідність. Розуміння цих факторів було важливим для подальшого розроблення нейромережевої моделі, оскільки це забезпечило можливість адаптувати модель під конкретні умови ринку та забезпечити більш точне прогнозування.

У другому розділі було проведено ґрунтовний аналіз різноманітних існуючих методів прогнозування, які використовуються на фондовому ринку. Дослідження охопило традиційні статистичні методи, такі як лінійна регресія, часові ряди, авторегресивні інтегровані моделі ковзного середнього (ARIMA), а також більш сучасні підходи, включаючи машинне навчання та аналіз великих даних. Особлива увага була приділена виявленню переваг та недоліків кожного методу, зокрема їх

здатності адаптуватися до швидко змінюваних умов ринку та точності прогнозування.

Крім того, у цьому розділі було розглянуто процес підбору та підготовки даних для навчання нейромережі. Це включало вибір відповідних джерел даних, таких як історичні ціни на дорогоцінні метали, об'єми торгівлі, макроекономічні показники та інші важливі економічні індикатори. Підготовка даних включала їх очищення, нормалізацію та трансформацію, щоб забезпечити їх придатність для ефективного навчання нейромережі. Такий підхід забезпечив надійну та відповідну базу даних для подальшого аналізу та прогнозування цін на фондовому ринку дорогоцінних металів.

У третьому розділі було зосереджено увагу на проектуванні та розробці програмного забезпечення для нейромережевого прогнозування. Розробка системи прогнозування включала декілька ключових аспектів, серед яких формалізація задачі прогнозування, вибір методології, підготовка вхідних даних, оцінка якості моделі, а також розробка базової архітектури системи.

Вхідні дані для системи включали ключові фінансові показники, такі як ціна відкриття, найвища ціна, найнижча ціна, ціна закриття ринку та об'єм торгів. Всі ці характеристики були використані для побудови моделі, яка може ефективно прогнозувати майбутні ціни на основі існуючих ринкових тенденцій та історичних даних.

Особлива увага була приділена архітектурі системи прогнозування, де взаємодія між ключовими компонентами, такими як модуль навчання моделей, користувацький інтерфейс, модуль доступу до даних та база даних, забезпечувала ефективне прогнозування.

У четвертому розділі роботи було проведено ретельне тестування та оцінку ефективності розробленої системи для прогнозування цін на дорогоцінні метали. Цей процес включав кілька ключових етапів, що дозволили виміряти точність та надійність моделі в різних сценаріях. Перш за все, було використано комплексне програмно-апаратне середовище, включаючи сучасний комп'ютер з високою обчислювальною потужністю та спеціалізоване програмне забезпечення для

розробки та тестування моделі. Це забезпечило надійність та точність проведених експериментів.

Далі, в рамках дослідження, було застосовано детальний аналіз різних метрик якості моделі, таких як R^2 (коефіцієнт детермінації), MAE (середня абсолютна помилка), MSE (середньоквадратична помилка) та RMSE (корінь з середньоквадратичної помилки). Ці метрики дали змогу оцінити точність та повноту прогнозування моделі, а також виявити потенційні слабкі місця.

В результаті проведеного експерименту з прогнозування цін на золото, було встановлено, що розроблена система демонструє високу точність прогнозів, що підтверджує успішність застосування нейромережевих технологій у сфері фондового ринку дорогоцінних металів. Це забезпечує великі перспективи для використання таких технологій в аналізі фінансових ринків та допомагає в розробці точних та ефективних інструментів для інвесторів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Доскочинська Л. С. Методологічні підходи до визначення функцій дорогоцінних металів у сучасних умовах / Любов Доскочинська // Сучасні тенденції розвитку міжнародних відносин: політика, економіка, право : зб. матеріалів міжнар. наук.-практ. конф. – Львів, 2012. – С. 204–206.
2. ARMA та ARIMA.: веб-сайт. URL: https://stud.com.ua/72679/ekonomika/modeli_kovznoyi_serednoyi
3. Моделі векторної авто регресії (VAR): веб-сайт. URL: https://www.wikiwand.com/uk/%D0%92%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D0%BD%D0%B0_%D0%B0%D0%B2%D1%82%D0%BE%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%96%D1%8F
4. Can news-based economic sentiment predict bubbles in precious metal markets: веб-сайт. URL: https://jfin-swufe.springeropen.com/articles/10.1186/s40854-022-00341-w.
5. Modeling and forecasting time series of precious metals: a new approach to multifractal data: веб-сайт. URL: https://jfin-swufe.springeropen.com/articles/10.1186/s40854-019-0135-3.
6. VARFIMA : веб-сайт. URL: https://www.rdocumentation.org/packages/multiwave/versions/1.0/topics/varfima
7. ARFIMA: веб-сайт. URL: https://www.stata.com/features/overview/arfima/
8. Precious metal price prediction using deep neural networks: веб-сайт. URL: https://dr.ntu.edu.sg/handle/10356/152921
9. LSTM: веб-сайт. URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/
10. Bi-LSTM-CNN : веб-сайт. URL: https://paperswithcode.com/method/cnn-bilstm
11. Матвійчук А. В. Моделювання фінансової стійкості підприємств із застосуванням теорій нечіткої логіки, нейронних мереж і дискримінантного аналізу / А. В. Матвійчук // К.: Вісн. НАН України. — 2010. — № 9. — С. 24-46.

12. Харинович-Яворська Д. О. Роль нейромережевих систем у формуванні конкурентної стратегії торговельного підприємства / Д. О. Харинович-Яворска // Наука й економіка: наук.-теор. жур. Хмельницьк. екон. ун-т. — Хмельницький, 2014. — № 1 (33). — 314 с. — С. 165-169.

13. Конволюційні нейронні мережі (CNN) : веб-сайт. URL: https://www.unite.ai/uk/what-are-convolutional-neural-networks/

14. Рекурентні нейронні мережі (RNN): веб-сайт. URL: https://www.wikiwand.com/uk/%D0%A0%D0%B5%D0%BA%D1%83%D1%80%D0%B5%D0%BD%D1%82%D0%BD%D0%B0_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0

15. Глибокі нейронні мережі (DNN): веб-сайт. URL: http://dspace.nbuv.gov.ua/bitstream/handle/123456789/122853/16-Samolyuk.pdf?sequence=1

16. Bainbridge W.S. Neural Network Models of Religious Belief / William Sims Bainbridge // Sociological Perspectives. – Vol. 38. – No. 4, Computer Simulations and Sociological Theory (Winter, 1995), с. 483-495.

17. Bloomberg Terminal : веб-сайт. URL: https://www.bloomberg.com/professional/solution/bloomberg-terminal/

18. Thomson Reuters Eikon: веб-сайт. URL: https://eikon.refinitiv.com/

19. MetaTrader 4 (MT4) та MetaTrader 5 (MT5): веб-сайт. URL: https://drukarnia.com.ua/articles/metatrader-4-i-metatrader-5-sho-krashe-PI-nM

20. Deep Learning: веб-сайт. URL: https://www.ibm.com/topics/deep-learning

21. ШНМ: веб-сайт. URL: https://learn.ztu.edu.ua/mod/resource/view.php?id=17833

22. Big Data : веб-сайт. URL: https://www.it.ua/knowledge-base/technology-innovation/big-data-bolshie-dannye

23. How To Use a Moving Average to Buy Stocks : веб-сайт. URL:https://www.investopedia.com/articles/active-trading/052014/how-use-moving-average-buy-stocks.asp

24. Основи моделювання ринкових ситуацій : веб-сайт. URL:https://web.posibnyky.vntu.edu.ua/fksa/12kocubynsky,kyslycia_osn_model_rynk_sytuac/p4.html

25. Економетричні методи прогнозування науково-технічного прогресу та світового розвитку : веб-сайт. URL:https://cyberleninka.ru/article/n/ekonometrichni-metodi-prognozuvannya-naukovo-tehnichnogo-progresu-ta-svitovogo-rozvitku/viewer

26. Авторегресивна інтегрована ковзна середня (ARIMA) : веб-сайт. URL:https://ua.nesrakonk.ru/autoregressive-integrated-moving-average-arima/

27. Autoregressive Integrated Moving Average (ARIMA) : веб-сайт. URL:https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp

28. Переваги та недоліки моделей ARIMA: веб-сайт. URL: https://stud.com.ua/41020/ekonomika/perevagi_nedoliki_modeley_arima

29. Що таке Фундаментальний аналіз (FA)? : веб-сайт. URL:https://academy.binance.com/uk/articles/what-is-fundamental-analysis-fa

30. Fundamental analysis (Фундаментальний аналіз) : веб-сайт. URL:https://alpari.com/ua/beginner/glossary/fundamental-analysis/

31. Переваги та недоліки фундаментального аналізу : веб-сайт. URL:https://uacredity.com/perevagi-ta-nedoliki-fundamentalnogo-analizu/

32. Штучні нейронні мережі (англ. ANN) : веб-сайт. URL: https://www.wikiwand.com/uk/%D0%A8%D1%82%D1%83%D1%87%D0%BD%D0%B0_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0

33. Терейковський І.А.,Бушуєв Д.А.,Терейковська Л.О. ШТУЧНІ НЕЙРОННІ МЕРЕЖІ:БАЗОВІ ПОЛОЖЕННЯ// Електронне мережне навчальне видання //, Київ-КПІ ім. Ігоря Сікорського-2022 : URL: https://ela.kpi.ua/bitstream/123456789/50135/1/ANN.pdf

34. Hoskins J. C., Himmelblau D. M. Process control via artificial neural networks and reinforcement learning. Computers & Chemical Engineering. 1992. Vol. 16, no. 4. P. 241—251. URL: https://doi.org/10.1016/0098-1354(92)80045-B

35. Kondratiuk S. Gesture recognition using convolutional neural networks with 3d convolutions // Штучний інтелект. – т.83-84 – 2019 – С.94-100.

36. A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. 2012.

37. Singh, B., Sharma, D. K. (2021). Predicting image credibility in fake news over social media using multimodal approach. Neural Computing and Applications, 34(24), 21503–21517.

38. Zennaki O. Inducing Multilingual Text Analysis Tools Using Bidirectional Recurrent Neural Networks / O. Zennaki, N. Semmar, L. Besacier // Conference: 26th International Conference on Computational Linguistics, COLING 2016. – p. 450-460.

39. Cho K. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation / K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio // Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014. – p. 1724–1734.

40. Random Forests: веб-сайт. URL: https://www.wikiwand.com/uk/Random_forest

41. Ali, J., Rehanullah Khan, Ahmad, N. and Imran Maqsood (2012). Random Forests and Decision Trees. ResearchGate. URL: https://www.researchgate.net/publication/259235118_Random_Forests_and_Decision_Trees

42. Mean Absolute Error - an overview : веб-сайт. URL:https://www.sciencedirect.com/topics/engineering/mean-absolute-error

43. RMSE: Root Mean Square Error: веб-сайт. URL: https://www.statisticshowto.com/probability-and-statistics/regression-analysis/rmse-root-mean-square-error/

44. MPE (Mean Percentage Error): веб-сайт. URL: https://academic-accelerator.com/encyclopedia/mean-percentage-error

45. Mean Absolute Percentage Error (MAPE)): веб-сайт. URL:https://www.statisticshowto.com/mean-absolute-percentage-error-mape/

46. Вітлінський В. В. Моделювання економіки : навч. посібник / В. В. Вітлінський.– 2-ге вид., без змін.– К. : КНЕУ, 2007.– 408 с.

47. Економіко-математичне моделювання : навч. посібник / Т. С. Клебанова, О. В. Раєвнева, С. В. Прокопович, С. О. Степуріна, Р. М. Яценко, І. М. Чуйко. – Х. : ВД «ІНЖЕК», 2010. – 352 с.

48. Miles R. Exam Ref 70-483 Programming in C#, 2nd Edition. Microsoft Press Store, 2018. 420 c.

49. Sharp, J. Microsoft Visual C# Step by Step, 9th Edition. Microsoft Press Store, 2018. 820 c.

50. Reid M. C++: 2 books in 1 - The Ultimate Beginners Guide to Master C++ Programming Quickly with No Prior Experience. Independently published, 780c.

51. Straustrup B. The C++ Programming Language. Addison-Wesley Professional, 2021, 580c.

52. The Python Tutorial URL: https://docs.python.org/3/tutorial/appetite.html.

53. Gold, Silver & Precious Metals Futures Daily Data. URL: https://www.kaggle.com/datasets/guillemservera/precious-metals-data.

ДОДАТОК А

Лістинги програми

ЗАТВЕРДЖУЮ
Проректор Українського
державного університету
науки і технологій
Анатолій РАДКЕВИЧ
10.01.24

«Використання нейромережевих технологій для короткострокового прогнозування економічних показників фондового ринку України»

Технічне завдання

ЛИСТ ЗАТВЕРДЖЕННЯ

44165850.1261-01 12-01-ЛЗ

Завідувач кафедри КІТ

_____Вадим ГОРЯЧКІН

Керівник розробки

_____Світлана ВОЛКОВА

Виконавець

_____Владислав СИВАК

Нормоконтролер

_____Світлана ВОЛКОВА

2024

ЗАТВЕРДЖЕНО

44165850.1261-01 12-01-ЛЗ

«Використання нейромережевих технологій для короткострокового прогнозування економічних показників фондового ринку України»

Лістинги програми

Листів 14

2024

Лістинг 1. Код класу «TeachNeuroForm»

```
using CommoditiesApp.AppCode;
using CommoditiesApp.DatasProviders;
using Microsoft.ML.Calibrators;
using Microsoft.ML.Data;
using Microsoft.ML;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;

namespace CommoditiesApp.Forms {
  public partial class TeachNeuroForm : Form {
    private string _Path = "";
    private MLContext context;
    ITransformer trainedModel;
    private IDataView dataFromFilesView;

    private int _selectedRowIndex = 0;
    private ValidationMy _Validation = new ValidationMy();
    private NeuroProvider _NeuroProvider = new NeuroProvider();
    private List<Neuro> _NeuroList = new List<Neuro>();
    private LoggerProvider _LoggerProvider = new LoggerProvider();
    private bool _IsModelTrain = false;

    public TeachNeuroForm() {
      InitializeComponent();
      DataLoad();
    }
```

```
private void OpenBtn_Click(object sender, EventArgs e) {
  OpenFileDialog openFileDialog = new OpenFileDialog();
  openFileDialog.Filter = "Text files (*.csv)|*.csv|All files (*.*)|*.*";
  openFileDialog.FilterIndex = 2;
  openFileDialog.RestoreDirectory = true;

  if (openFileDialog.ShowDialog() == DialogResult.OK) {
    try {
      _Path = openFileDialog.FileName;
      FilePathTBox.Text = openFileDialog.FileName;

      //Контекст
      context = new MLContext();

      // Завантаження даних із файлу
      dataFromFilesView = context.Data.LoadFromTextFile<CommoditieData>(_Path,
        hasHeader: true, separatorChar: ',');



      RaportTBox.Text = "Тренування моделі..." + "\r\n\r\n";

      // Розбиття даних на навчальні та тестові
      var splitData = context.Data.TrainTestSplit(dataFromFilesView, testFraction: 0.25);

      // Трансформації даних
      var dataProcessPipeline = context.Transforms.CopyColumns(outputColumnName: "Label",
        inputColumnName: "Close")
          .Append(context.Transforms.Concatenate("Features", "Open", "High", "Low", "Volume"))
          .Append(context.Transforms.Text.FeaturizeText(inputColumnName: "Date",
          outputColumnName: "DateFeaturized"))
          .Append(context.Transforms.Concatenate("Features",
          "Features", "DateFeaturized"));

      // Вибір алгоритму: FastForest
```

```
        var trainer =
          context.Regression.Trainers.FastForest(labelColumnName: "Label",
          featureColumnName: "Features");

        //Створення пейплайну навчання  моделі
        var trainingPipeline = dataProcessPipeline.Append(trainer);

        // Навчання моделі
        trainedModel = trainingPipeline.Fit(splitData.TrainSet);

        // Оцінювання моделі
        var predictions = trainedModel.Transform(splitData.TestSet);
        var metrics = context.Regression.Evaluate(predictions,
          labelColumnName: "Label",
          scoreColumnName: "Score");

        // Виведення метрик якості моделі
        RaportTBox.Text += "Метрики якості моделі" + "\r\n";
        RaportTBox.Text += ($"R^2 (Coefficient of Determination): {metrics.RSquared:0.##}") + "\r\n";
        RaportTBox.Text += ($"MAE (Mean Absolute Error): {metrics.MeanAbsoluteError:#.##}") +
"\r\n";
        RaportTBox.Text += ($"MSE (Mean Squared Error): {metrics.MeanSquaredError:#.##}") + "\r\n";
        RaportTBox.Text += ($"RMSE (Root Mean Squared Error):
{metrics.RootMeanSquaredError:#.##}") + "\r\n";
        RaportTBox.Text += ($"Loss Function: {metrics.LossFunction:#.##}") + "\r\n";

        _IsModelTrain = true;
       } catch (Exception ex) {
        RaportTBox.Text = ex.Message;

       }
      }
    }

    private void DataLoad() {
```

```
  int firstRowIndex = 0;
  if (NeuroGridView.FirstDisplayedScrollingRowIndex > 0) {
    firstRowIndex = NeuroGridView.FirstDisplayedScrollingRowIndex;
  }
  try {
    _NeuroList = _NeuroProvider.GetAllNeuro();
    LoadDataInNeuroGridView(_NeuroList);
    if (_selectedRowIndex == NeuroGridView.Rows.Count) {
      _selectedRowIndex = NeuroGridView.Rows.Count - 1;
    }
    if (_selectedRowIndex >= 0) {
      NeuroGridView.FirstDisplayedScrollingRowIndex = firstRowIndex;
      NeuroGridView.Rows[_selectedRowIndex].Selected = true;
    }
  } catch (Exception ex) {
    MessageBox.Show(ex.ToString());
  }
}

private void LoadDataInNeuroGridView(List<Neuro> NeuroList) {
  NeuroGridView.DataSource = null;
  NeuroGridView.Columns.Clear();
  NeuroGridView.AutoGenerateColumns = false;
  NeuroGridView.RowHeadersVisible = false;

  NeuroGridView.DataSource = NeuroList;

  if (NeuroList.Count > 0) {
    if (NeuroList[0].Message == NamesMy.NoDataNames.NoDataInNeuro) {
      DataGridViewColumn messageColumn = new DataGridViewTextBoxColumn();
      messageColumn.DataPropertyName = "Message";
      messageColumn.Width = NeuroGridView.Width - NamesMy.SizeOptins.MinusSizePanel;
      NeuroGridView.Columns.Add(messageColumn);
    } else {
      DataGridViewColumn DetailIdColumn = new DataGridViewTextBoxColumn();
```

```
        DetailIdColumn.DataPropertyName = "NeuroId";
        NeuroGridView.Columns.Add(DetailIdColumn);
        NeuroGridView.Columns[0].Visible = false;

        DataGridViewColumn numberColumn = new DataGridViewTextBoxColumn();
        numberColumn.HeaderText = "№ ";
        numberColumn.DataPropertyName = "Number";
        numberColumn.DefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleRight;
        numberColumn.Width = NamesMy.SizeOptins.NumberSize;
        NeuroGridView.Columns.Add(numberColumn);

        DataGridViewColumn NeuroNamesColumn = new DataGridViewTextBoxColumn();
        NeuroNamesColumn.HeaderText = "Модель нейромережі";
        NeuroNamesColumn.DataPropertyName = "NeuroNames";
        NeuroNamesColumn.Width = 450;
        NeuroGridView.Columns.Add(NeuroNamesColumn);
      }
      for (int i = 0; i < NeuroGridView.Columns.Count; i++) {
        NeuroGridView.Columns[i].HeaderCell.Style.BackColor = Color.LightGray;
      }
    }
  }

  private void SaveBtn_Click(object sender, EventArgs e) {
    if (IsDatasCorrect()) {
      //Save model
      string pathName = @"\teach\" + GenerateFileName() + ".zip";
      string localProj =

System.IO.Path.GetDirectoryName(System.Reflection.Assembly.GetExecutingAssembly().Location);
      _NeuroProvider.InsertNeuro(NeuralNamesTBox.Text, pathName);
      context.Model.Save(trainedModel, dataFromFilesView.Schema, localProj + pathName);
      ClearAllData();
      _LoggerProvider.InsertLogger("Було навчено нейронну мережу "
        + NeuralNamesTBox.Text, DateTime.Now);
```

```
        MessageBox.Show("Дані успішно збережено!");
      }
    }

    private void ClearAllData() {
      _IsModelTrain = false;
      FilePathTBox.Text = String.Empty;
      NeuralNamesTBox.Text = String.Empty;
      RaportTBox.Text = String.Empty;
      DataLoad();
    }

    public string GenerateFileName() {
      DateTime now = DateTime.Now;
      string fileName = string.Format("{0}_{1}_{2}_{3}_{4}_{5}",
          now.Year, now.Month, now.Day, now.Hour, now.Minute, now.Second);

      return fileName;
    }

    private void ClearBtn_Click(object sender, EventArgs e) {
      ClearAllData();
    }

    private void ExitBtn_Click(object sender, EventArgs e) {
      this.Close();
    }

    private bool IsDatasCorrect() {
      bool isCorrect = true;
      if (!_IsModelTrain) {
        MessageBox.Show("Неможливо зберегти дані. \r\nЩе не навчено нейронну мережу!",
"Увага!");
        isCorrect = false;
      }
```

```
    if (_Validation.IsDataEntering(NeuralNamesTBox.Text)) {
      NeuralNamesValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
    } else {
      NeuralNamesValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
      isCorrect = false;
    }
    return isCorrect;
  }

  private void NeuroGridView_CellClick(object sender, DataGridViewCellEventArgs e) {
    if (e.RowIndex >= 0 && NeuroGridView[0, e.RowIndex].Value.ToString() !=
_NeuroList[0].Message) {
      _selectedRowIndex = e.RowIndex;
      UpdateNeuroForm updateNeuroForm = new
UpdateNeuroForm(Convert.ToInt32(NeuroGridView[0, e.RowIndex].Value.ToString()));
      updateNeuroForm.ShowDialog();
      DataLoad();
    }
  }
 }
}
```

Лістинг 2. Код класу «CommoditieDataProvider»

```
using Microsoft.ML.Data;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace CommoditiesApp.DatasProviders {
 internal class CommoditieDataProvider {

 }
}
```

```
public class CommoditieData {
  [LoadColumn(0)]
  public string Date { get; set; } //Дата запису даних
  [LoadColumn(1)]
  public float Open { get; set; } //Ціна відкриття ринку

  [LoadColumn(2)]
  public float High { get; set; } //Найвища ціна протягом торгового дня

  [LoadColumn(3)]
  public float Low { get; set; } //Найнижча ціна протягом торгового дня

  [LoadColumn(4)]
  public float Close { get; set; } //Ціна закриття ринку

  [LoadColumn(5)]
  public float Volume { get; set; } //Кількість контрактів торгів протягом дня
}

public class CommoditiePrediction {
  [ColumnName("Score")]
  public float Close { get; set; }
}
```

Лістинг 3. Код класу «TestModelForm»

```
using CommoditiesApp.AppCode;
using CommoditiesApp.DatasProviders;
using Microsoft.ML;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Globalization;
```

```
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace CommoditiesApp.Forms {
  public partial class TestModelForm : Form {
    private ValidationMy _Validation = new ValidationMy();
    private Neuro _SelectedNeural = new Neuro();
    private MLContext context = new MLContext();
    private PredictionEngine<CommoditieData, CommoditiePrediction> predictionEngine;
    private NeuroProvider _NeuralProvider = new NeuroProvider();
    private List<Neuro> _NeuroL = new List<Neuro>();
    private bool _IsThemesLoad = false;

    private LoggerProvider _LoggerProvider = new LoggerProvider();

    public TestModelForm() {
      InitializeComponent();
      LoadAllDate();
    }

    private void LoadAllDate() {
      _NeuroL = _NeuralProvider.GetAllNeuro();
      NeuroCBox.DataSource = _NeuroL;
      NeuroCBox.ValueMember = "NeuroId";
      NeuroCBox.DisplayMember = "NeuroNames";
      _IsThemesLoad = true;
      DateMinDTP.Value = DateTime.Now;
      DateMaxDTP.Value = DateTime.Now.AddYears(10);
      NeuroCBox_SelectedValueChanged(NeuroCBox, EventArgs.Empty);
    }

    private void PredictBtn_Click(object sender, EventArgs e) {
      if (IsAllCommoditieDataCorrect()) {
```

```
        var prediction = predictionEngine.Predict(new CommoditieData {
          Date = DateDTP.Value.ToShortDateString(),
          Open = (float)Convert.ToDouble(OpenTBox.Text),
          High = (float)Convert.ToDouble(HighTBox.Text),
          Low = (float)Convert.ToDouble(LowTBox.Text),
          Volume = (float)Convert.ToDouble(VolumeTBox.Text)
        });
        var commoditieDataInfo = new StringBuilder();
        commoditieDataInfo.AppendLine("\r\n--- Прогноз ---");
        commoditieDataInfo.AppendLine($"Прогнозована ціна закриття ринку: { prediction.Close}");
        RaportTBox.Text = commoditieDataInfo.ToString();
        _LoggerProvider.InsertLogger("Проведено прогнозування цін для моделі '"
          + NeuroCBox.Text + "'", DateTime.Now);
      }
    }

    private void NeuroCBox_SelectedValueChanged(object sender, EventArgs e) {
      if (_IsThemesLoad) {
        _SelectedNeural =
_NeuralProvider.SelectedNeuroByNeuroId(Convert.ToInt32(NeuroCBox.SelectedValue));
        LoadData(_SelectedNeural.NeuroFileModel);
      }
    }

    private void LoadData(string FilePath) {
      string localProj = Application.StartupPath + FilePath;
      // Define DataViewSchema for data preparation pipeline and trained model
      DataViewSchema modelSchema;
      // Load trained model
      ITransformer model = context.Model.Load(localProj, out modelSchema);
      // Evaluate the model
      // Use the model to make predictions
      predictionEngine = context.Model.CreatePredictionEngine<CommoditieData,
CommoditiePrediction>(model);
    }
```

```
private bool IsAllCommoditieDataCorrect() {
 bool isCorrect = true;
 if (_Validation.IsDataConvertToDouble(OpenTBox.Text)) {
  OpenValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
 } else {
  OpenValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
  isCorrect = false;
 }
 if (_Validation.IsDataConvertToDouble(HighTBox.Text)) {
  HighValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
 } else {
  HighValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
  isCorrect = false;
 }
 if (_Validation.IsDataConvertToDouble(LowTBox.Text)) {
  LowValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
 } else {
  LowValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
  isCorrect = false;
 }
 //if (_Validation.IsDataConvertToDouble(CloseTBox.Text)) {
 //  CloseValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
 //} else {
 //  CloseValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
 //  isCorrect = false;
 //}
 if (_Validation.IsDataConvertToInt(VolumeTBox.Text)) {
  VolumeValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
 } else {
  VolumeValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
  isCorrect = false;
 }
```

```
  return isCorrect;
}


private bool IsGenDataCorrect() {
  bool isCorrect = true;
  if (_Validation.IsDataConvertToDouble(OpenMinTBox.Text)) {
    OpenMinValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
  } else {
    OpenMinValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
    isCorrect = false;
  }
  if (_Validation.IsDataConvertToDouble(HighMinTBox.Text)) {
    HighMinValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
  } else {
    HighMinValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
    isCorrect = false;
  }
  if (_Validation.IsDataConvertToDouble(LowMinTBox.Text)) {
    LowMinValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
  } else {
    LowMinValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
    isCorrect = false;
  }
  //if (_Validation.IsDataConvertToDouble(CloseMinTBox.Text)) {
  //  CloseMinValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
  //} else {
  //  CloseMinValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
  //  isCorrect = false;
  //}
  if (_Validation.IsDataConvertToInt(VolumeMinTBox.Text)) {
    VolumeMinValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
  } else {
    VolumeMinValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
    isCorrect = false;
```

```
}
if (_Validation.IsDataConvertToDouble(OpenMaxTBox.Text)) {
  OpenMaxValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
} else {
  OpenMaxValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
  isCorrect = false;
}
if (_Validation.IsDataConvertToDouble(HighMaxTBox.Text)) {
  HighMaxValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
} else {
  HighMaxValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
  isCorrect = false;
}
if (_Validation.IsDataConvertToDouble(LowMaxTBox.Text)) {
  LowMaxValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
} else {
  LowMaxValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
  isCorrect = false;
}
//if (_Validation.IsDataConvertToDouble(CloseMaxTBox.Text)) {
//  CloseMaxValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
//} else {
//  CloseMaxValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
//  isCorrect = false;
//}
if (_Validation.IsDataConvertToInt(VolumeMaxTBox.Text)) {
  VolumeMaxValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
} else {
  VolumeMaxValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
  isCorrect = false;
}
if (DateMinDTP.Value > DateMaxDTP.Value) {
  MessageBox.Show("Мінімальна дата має бути менша за  максимальну", "Увага!");
  isCorrect = false;
}
```

```
    return isCorrect;
  }


  private void GenBtn_Click(object sender, EventArgs e) {
    if (IsGenDataCorrect()) {
      if (timer1.Enabled) {
        timer1.Enabled = false;
        GenBtn.Text = "Генерувати";
      _LoggerProvider.InsertLogger("Запущено генератор для моделі '" +
        NeuroCBox.Text + "'", DateTime.Now);
      } else {
        timer1.Enabled = true;
        GenBtn.Text = "Зупинити";
      }
    }
  }


  private void timer1_Tick(object sender, EventArgs e) {
    if (IsGenDataCorrect()) {
      var rnd = new Random();
      var randomData = GenerateCustomModelData(DateMinDTP.Value, DateMaxDTP.Value,
                                Convert.ToDouble(OpenMinTBox.Text),
Convert.ToDouble(OpenMaxTBox.Text),
                                Convert.ToDouble(HighMinTBox.Text),
Convert.ToDouble(HighMaxTBox.Text),
                                Convert.ToDouble(LowMinTBox.Text),
Convert.ToDouble(LowMaxTBox.Text),
                                //Convert.ToDouble(CloseMinTBox.Text),
Convert.ToDouble(CloseMaxTBox.Text),
                                Convert.ToDouble(VolumeMinTBox.Text),
Convert.ToDouble(VolumeMaxTBox.Text), rnd);
      RaportTBox.Text = DisplayGenData(randomData); // Виведення згенерованих даних
```

```
        var prediction = predictionEngine.Predict(randomData);
        RaportTBox.Text += ($"Прогнозована ціна закриття ринку: {prediction.Close}");
      } else {
        RaportTBox.Text = "Неправильне введення!";
      }
    }

    public CommoditieData GenerateCustomModelData(
        DateTime dateMin, DateTime dateMax,
        double openMin, double openMax,
        double highMin, double highMax,
        double lowMin, double lowMax,
        double volumeMin, double volumeMax,
        Random rnd) {

      DateTime date = RandomDate(dateMin, dateMax, rnd);
      double closeMin = 1120.0;
      double closeMax = 1999.0;

      // Генерація ціни відкриття в межах заданих параметрів
      double open = RandomDouble(openMin, openMax, rnd);

      // Генерація найвищої ціни як випадкового значення, не більше ніж на 5% вище за ціну відкриття
      double high = open + (open * RandomDouble(0.01, 0.05, rnd));

      // Генерація найнижчої ціни як випадкового значення, не нижче за 70% від ціни відкриття
      double low = open - (open * RandomDouble(0.01, 0.30, rnd));
      if (low < 0) low = 0;

      double volume = RandomDouble(volumeMin, volumeMax, rnd);

      return new CommoditieData {
        Date = date.ToString("yyyy-MM-dd"), Open = (float)open,
        High = (float)high, Low = (float)low, Volume = (float)volume
```

```
    };
  }


  private DateTime RandomDate(DateTime start, DateTime end, Random rnd) {
    int range = (end - start).Days;
    return start.AddDays(rnd.Next(range));
  }

  private double RandomDouble(double min, double max, Random rnd) {
    return rnd.NextDouble() * (max - min) + min;
  }


  private string DisplayGenData(CommoditieData data) {
    var commoditieDataInfo = new StringBuilder();
    commoditieDataInfo.AppendLine("Згенеровані дані:");
    commoditieDataInfo.AppendLine($"Дата запису даних: {data.Date}");
    commoditieDataInfo.AppendLine($"Ціна відкриття ринку: {Math.Round(Math.Abs(data.Open),
2)}");
    commoditieDataInfo.AppendLine($"Найвища ціна протягом торгового дня:
{Math.Round(Math.Abs(data.High),2)}");
    commoditieDataInfo.AppendLine($"Найнижча ціна протягом торгового дня:
{Math.Round(Math.Abs(data.Low),2)}");
    commoditieDataInfo.AppendLine($"Обсяг торгів: {Math.Round(Math.Abs(data.Volume))}");
    commoditieDataInfo.AppendLine("\r\n--- Прогноз ---");
    return commoditieDataInfo.ToString();
  }

 }
}
```

Лістинг 4. Код класу «UpdateNeuroForm»

```
using CommoditiesApp.AppCode;
using CommoditiesApp.DatasProviders;
using Microsoft.ML;
```

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace CommoditiesApp.Forms {
  public partial class UpdateNeuroForm : Form {
    private string _Path = "";
    private MLContext context;
    ITransformer trainedModel;
    private IDataView dataView;

    private int _NeuroId = 0;
    private ValidationMy _validation = new ValidationMy();
    private NeuroProvider _NeuroProvider = new NeuroProvider();
    private Neuro _selectedNeuro = new Neuro();
    private bool _IsModelTrain = false;
    private LoggerProvider _LoggerProvider = new LoggerProvider();

    public UpdateNeuroForm(int NeuroId) {
      InitializeComponent();
      _NeuroId = NeuroId;
      LoadAllDate();
    }

    private void OpenBtn_Click(object sender, EventArgs e) {
      OpenFileDialog openFileDialog = new OpenFileDialog();

      openFileDialog.Filter = "Text files (*.csv)|*.csv|All files (*.*)|*.*";
      openFileDialog.FilterIndex = 2;
```

```
    openFileDialog.RestoreDirectory = true;

    if (openFileDialog.ShowDialog() == DialogResult.OK) {
      try {
        _Path = openFileDialog.FileName;
        FileNameTBox.Text = openFileDialog.FileName;

        context = new MLContext();

        // Завантаження даних
        dataView = context.Data.LoadFromTextFile<CommoditieData>(_Path, hasHeader: true, separatorChar: ',');
        RaportTBox.Text = "Тренування моделі..." + "\r\n\r\n";

        // Розбиття даних на навчальні та тестові
        var splitData = context.Data.TrainTestSplit(dataView, testFraction: 0.2);

        // Трансформації даних
        var dataProcessPipeline = context.Transforms.CopyColumns(outputColumnName: "Label", inputColumnName: "Close")
            .Append(context.Transforms.Concatenate("Features", "Open", "High", "Low", "Volume"))
            .Append(context.Transforms.Text.FeaturizeText(inputColumnName: "Date", outputColumnName: "DateFeaturized"))
            .Append(context.Transforms.Concatenate("Features", "Features", "DateFeaturized"));

        // Вибір алгоритму: FastForest для регресії
        var trainer = context.Regression.Trainers.FastForest(labelColumnName: "Label", featureColumnName: "Features");

        var trainingPipeline = dataProcessPipeline.Append(trainer);

        // Навчання моделі
        trainedModel = trainingPipeline.Fit(splitData.TrainSet);

        // Оцінка моделі
```

```
        var predictions = trainedModel.Transform(splitData.TestSet);
        var metrics = context.Regression.Evaluate(predictions, labelColumnName: "Label", scoreColumnName: "Score");

        // Виведення метрик якості моделі
        RaportTBox.Text += "Метрики якості моделі" + "\r\n";
        RaportTBox.Text += ($"R^2 (Coefficient of Determination): {metrics.RSquared:0.##}") + "\r\n";
        RaportTBox.Text += ($"MAE (Mean Absolute Error): {metrics.MeanAbsoluteError:#.##}") + "\r\n";
        RaportTBox.Text += ($"MSE (Mean Squared Error): {metrics.MeanSquaredError:#.##}") + "\r\n";
        RaportTBox.Text += ($"RMSE (Root Mean Squared Error): {metrics.RootMeanSquaredError:#.##}") + "\r\n";
        RaportTBox.Text += ($"Loss Function: {metrics.LossFunction:#.##}") + "\r\n";

        _IsModelTrain = true;
       } catch (Exception ex) {
        RaportTBox.Text = ex.Message;

       }
     }
   }

   private void SaveBtn_Click(object sender, EventArgs e) {
     if (IsDataEnteringCorrect()) {
       string pathName = @"\teach\" + GenerateFileName() + ".zip";
       string localProj = System.IO.Path.GetDirectoryName(System.Reflection.Assembly.GetExecutingAssembly().Location);
       _NeuroProvider.UpdateNeuro(NeuralNamesTBox.Text, pathName, _NeuroId);
       context.Model.Save(trainedModel, dataView.Schema, localProj + pathName);
       _LoggerProvider.InsertLogger("Було навчено нейронну мережу " + NeuralNamesTBox.Text, DateTime.Now);
       MessageBox.Show("Дані успішно збережено!");
       this.Close();
     }
```

```
    }
    private void DeleteBtn_Click(object sender, EventArgs e) {
      if (MessageBox.Show("Ви дійсно хочете видалити цю нейромережу?", "Видалити",
MessageBoxButtons.YesNo) == DialogResult.Yes) {
        _NeuroProvider.DeleteNeuroByNeuroId(_NeuroId);
        this.Close();
      }
    }


    private void LoadAllDate() {
      _selectedNeuro = _NeuroProvider.SelectedNeuroByNeuroId(_NeuroId);
      NeuralNamesTBox.Text = _selectedNeuro.NeuroNames;
      FileNameTBox.Text = _selectedNeuro.NeuroFileModel;
    }

    private void ExitBtn_Click(object sender, EventArgs e) {
      this.Close();
    }


    public string GenerateFileName() {
      DateTime now = DateTime.Now;
      string fileName = string.Format("{0}_{1}_{2}_{3}_{4}_{5}",
          now.Year, now.Month, now.Day, now.Hour, now.Minute, now.Second);

      return fileName;
    }

    private bool IsDataEnteringCorrect() {
      bool isCorrect = true;
      if (!_IsModelTrain) {
        MessageBox.Show("Неможливо зберегти дані. \r\nЩе не навчено нейронну мережу!",
"Увага!");
        isCorrect = false;
```

```
    }
    if (_validation.IsDataEntering(NeuralNamesTBox.Text)) {
      NeuralNamesValidationLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
    } else {
      NeuralNamesValidationLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
      isCorrect = false;
    }
    return isCorrect;
   }


  }
}
```

ДОДАТОК Б

Приклад ДАТАСЕТУ

ЗАТВЕРДЖУЮ

Проректор

Українського державного університету науки і технології

Анатолій РАДКЕВИЧ

ТЕМА ДИПЛОМУ

Технічне завдання

ЛИСТ ЗАТВЕРДЖЕННЯ

1116130.індив номер

Завідувач кафедри КІТ

_____Вадим ГОРЯЧКІН

Керівник розробки

_____

Виконавець

_____

Нормоконтролер

_____Світлана ВОЛКОВА

2024

ЗАТВЕРДЖЕНО

1116130.0xxxx-01

ТЕМА ДИПЛОМУ

Текст програми

Листів 1

2024

Файл «generated_gold_data»

ticker,commodity,date,open,high,low,close,volume

GC=F,Gold,2000-08-30,273.8999938964844,273.8999938964844,273.8999938964844,273.8999938964844,0

GC=F,Gold,2000-08-31,274.79998779296875,278.29998779296875,274.79998779296875,278.29998779296875,0

GC=F,Gold,2000-09-01,277.0,277.0,277.0,277.0,0

GC=F,Gold,2000-09-05,275.79998779296875,275.79998779296875,275.79998779296875,275.79998779296875,2

GC=F,Gold,2000-09-06,274.20001220703125,274.20001220703125,274.20001220703125,274.20001220703125,0

GC=F,Gold,2000-09-07,274.0,274.0,274.0,274.0,125

GC=F,Gold,2000-09-08,273.29998779296875,273.29998779296875,273.29998779296875,273.29998779296875,0

GC=F,Gold,2000-09-11,273.1000061035156,273.1000061035156,273.1000061035156,273.1000061035156,0

GC=F,Gold,2000-09-12,272.8999938964844,272.8999938964844,272.8999938964844,272.8999938964844,0

GC=F,Gold,2000-09-13,272.79998779296875,272.79998779296875,272.79998779296875,272.79998779296875,0

GC=F,Gold,2000-09-14,272.3999938964844,272.3999938964844,272.3999938964844,272.3999938964844,0

GC=F,Gold,2000-09-15,272.29998779296875,272.29998779296875,272.29998779296875,272.29998779296875,0

GC=F,Gold,2000-09-18,271.3999938964844,271.3999938964844,271.3999938964844,271.3999938964844,0

GC=F,Gold,2000-09-19,271.8999938964844,271.8999938964844,271.8999938964844,271.8999938964844,0

GC=F,Gold,2000-09-20,269.0,269.0,269.0,269.0,0

GC=F,Gold,2000-09-21,270.29998779296875,270.29998779296875,270.29998779296875,270.29998779296875,0

GC=F,Gold,2000-09-22,271.79998779296875,271.79998779296875,271.79998779296875,271.79998779296875,0

GC=F,Gold,2000-09-25,274.1000061035156,274.1000061035156,274.1000061035156,274.1000061035156,0

GC=F,Gold,2000-09-
26,273.8999938964844,273.8999938964844,273.8999938964844,273.8999938964844,0
GC=F,Gold,2000-09-
27,278.3999938964844,278.3999938964844,278.3999938964844,278.3999938964844,3511
GC=F,Gold,2000-09-28,277.5,277.5,275.1000061035156,275.6000061035156,631
GC=F,Gold,2000-09-29,274.6000061035156,274.6000061035156,272.0,273.6000061035156,22
GC=F,Gold,2000-10-02,272.79998779296875,273.5,272.5,273.1000061035156,161