МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕПАРТАМЕНТ ОСВІТИ І НАУКИ ВИКОНАВЧОГО ОРГАНУ
КИЇВСЬКОЇ МІСЬКОЇ РАДИ
(КИЇВСЬКОЇ МІСЬКОЇ ДЕРЖАВНОЇ АДМІНІСТРАЦІЇ)
КИЇВСЬКЕ ТЕРИТОРІАЛЬНЕ ВІДДІЛЕННЯ МАЛОЇ АКАДЕМІЇ НАУК УКРАЇНИ
(КИЇВСЬКА МАЛА АКАДЕМІЯ НАУК)

Відділення комп’ютерних наук

Секція: системи та технології штучного інтелекту

ЗАСТОСУВАННЯ НЕЙРОННИХ МЕРЕЖ ДЛЯ ПРОГНОЗУВАННЯ КУРСУ КРИПТОВАЛЮТ

Роботу виконав:
Макаренко Максим Олегович,
студент 1-го курсу
Київського фахового коледжу туризму та готельного господарства.

Науковий керівник:
Дем’яненко Валентина Борисівна,
завідувач відділу інформаційно-дидактичного моделювання Національного центру «Мала академія наук України», кандидат педагогічних наук

КИЇВ – 2022

Анотація

Застосування нейронних мереж для прогнозування курсу криптовалют

Макаренко Максим Олегович, Київське територіальне відділення Малої академії наук України, Комунальний позашкільний навчальний заклад «Київська мала академія наук учнівської молоді», студент 1-го курсу Київського фахового коледжу туризму та готельного господарства. Науковий керівник: Дем'яненко Валентина Борисівна, завідувачка відділу інформаційно-дидактичного моделювання Національного центру «Мала академія наук України», кандидат педагогічних наук.

Останнім часом все більшої популярності набувають інвестиції та торгівля криптовалютами. На динаміку курсу впливає одночасно значна кількість об'єктивних та суб'єктивних чинників. Застосування нейронних мереж для таких цілей є перспективним, малодослідженим теоретичним та прикладним напрямом досліджень. В роботі проаналізовано теоретичні засади розробки нейронних мереж та типи торгових угод. Описано архітектуру розробленого програмного забезпечення, структуру нейронної мережі, її налаштування та параметри навчання. Апробовано розробку та її ефективність в процесі торгівлі. Мета: розробити програмне забезпечення на основі нейронних мереж для прогнозування курсу криптовалют та здійснення на основі цих прогнозів торгових операцій. Для досягнення мети було поставлено такі завдання: проаналізувати теоретичні засади розробки нейронних мереж; визначити особливості різних типів біржових торгових угод; розробити програмне забезпечення для прогнозування курсу криптовалют; провести апробацію розробленого програмного забезпечення, з метою визначення його ефективності.

Як результат, розроблено програмне забезпечення для прогнозування зміни курсів криптовалют та подальшого використання цих прогнозів в торгівлі.

Ключові слова: машинне навчання, нейронні мережі, тренування нейронних мереж, оптимізатор, функція втрат, криптовалюти.

ЗМІСТ

ВСТУП

Актуальність. Економічна нестабільність, світові та національні кризи, які серед іншого, призводять до інфляції і значних втрат заощаджень, спонукають все більшу кількість людей шукати нові способи збереження та примноження своїх коштів. Останнім часом все більшої популярності набувають інвестиції та торгівля криптовалютами. Однак прогнозування їх курсу є на багато складнішим процесом, ніж прогнозування курсу національних валют, ціни дорогоцінних металів, чи акцій різноманітних компаній. На динаміку курсу впливає одночасно велика кількість об’єктивних та суб’єктивних чинників, врахувати які людському мозку просто не під силу. Застосування нейронних мереж для таких цілей є перспективним, малодослідженим і маловипробуваним теоретичним та прикладним напрямом досліджень.

Мета: розробити програмне забезпечення на основі нейронних мереж для прогнозування курсу криптовалют та здійснення на основі цих прогнозів торгових операцій.

Для досягнення мети необхідно виконати такі завдання:

1. Проаналізувати теоретичні засади розробки нейронних мереж;
2. Визначити особливості різних типів біржових торгових угод;
3. Розробити програмне забезпечення для прогнозування курсу криптовалют.
4. Провести апробацію розробленого програмного забезпечення, з метою визначення його ефективності.
5. Виправити недоліки, виявлені в процесі практичного використання програми для здійснення торгових операцій.

Об’єкт дослідження: процес розробки систем штучного інтелекту.

Предмет дослідження: використання нейронних мереж для прогнозування курсу криптовалют.

Практична значущість отриманих результатів полягає в можливості використовувати програмне забезпечення для здійснення малоризикованих торгових операцій з криптовалютами.

Нейронні мережі не часто, але використовуються для подібних цілей, ми спробували реалізувати такі можливості в нашій розробці через використання широкого набору параметрів для аналізу, особливі налаштування, гнучкість, масштабованість та адаптивність. Було продемонстровано високу точність короткострокових прогнозів. Результативність прогнозів складає 90 %.

РОЗДІЛ 1

Теоретичні засади розробки нейронних мереж та здійснення біржових торгових операцій

1.1. Нейронні мережі та їх місце в технологіях машинного навчання

Технології машинного навчання дуже багатогранні. Існує значна кількість моделей та алгоритмів, пов’язаних з цим напрямом. Головне призначення технологій машинного навчання – знайти розв’язування задачі, коли не існує лінійного вирішення або воно занадто складне для програмної реалізації. Моделі машинного навчання – це математичний шаблон, зазначивши налаштування якого, можливо розв’язати задачу певного типу. Окрім моделі, технологія машинного навчання включає алгоритм навчання. Алгоритм навчання – це набір певних дій, що допомагає пристосувати шаблонну модель до розв’язування конкретної задачі. Особливе місце в технологіях машинного навчання займають нейронні мережі.

Нейронна мережа – це математична модель, яка побудована подібно до біологічних нейронних мереж в структурі людського мозку і призначена для знаходження закономірностей в наборі даних.

Якщо значна кількість технології машинного навчання призначені для розв’язування певного типу задач та обмежені специфікою свого влаштування, нейронні мережі є універсальним інструментом машинного навчання, що не мають жодних обмежень. Нейронні мережі були обрані в цьому дослідженні, як основа алгоритму прийняття торгових рішень тому, що застосування інших моделей машинного навчання для досягнення поставленої мети дослідження є менш ефективними, адже моделей, що розроблялися для розв’язування цієї задачі не розглядались. Отже, запропонована тема дослідницької роботи є доцільною для наукового пошуку знаходження розв’язків, поставлених завдань.

Для того, щоб досягти мети дослідницької роботи необхідно проаналізувати та узагальнити теоретичні джерела, пов’язані з предметом дослідження: визначити, якими бувають нейронні мережі, які технології та алгоритми покладені в їх основу. Це необхідно для глибокого розуміння процесів, без яких неможливо правильно визначити тип нейронних мереж, методи опрацювання даних, принцип

формування навчальної вибірки, параметри навчання та налаштування нейронної мережі, що знадобиться при розробці програмного забезпечення.

1.2. Нейронні мережі прямого поширення

Узагальнюючи матеріали подані в опрацьованій нами статті можемо зазначити [1]: нейронні мережі прямого поширення – це архітектура нейронних мереж в якій нейрони розташовані прошарками, таким чином, що кожен нейрон вхідного та прихованих прошарків пов'язаний з кожним нейроном наступного прошарку. Отже, дані постійно передаються вперед. Нейронні мережі прямого поширення не мають рекурентних прошарків, нейрони не мають зв'язків між нейронами свого, або попередніх прошарків, зв'язки між нейронами не утворюють циклу. Нейронні мережі прямого поширення були першим і найпростішим типом штучних нейронних мереж. Такі нейронні мережі в основному використовуються для контрольованого навчання в тих випадках, коли дані, що підлягають опрацюванню, не є ані послідовними, ані часовими.

Найпростіший тип нейронної мережі прямого поширення – це персептрон (SLP), нейронна мережа прямого поширення без прихованих одиниць (рис. 1.1.). Таким чином, персептрон має лише вхідний і вихідний рівень. Вихідні одиниці обчислюються безпосередньо з суми добутку їх ваги з відповідними вхідними одиницями, плюс деяке зміщення.

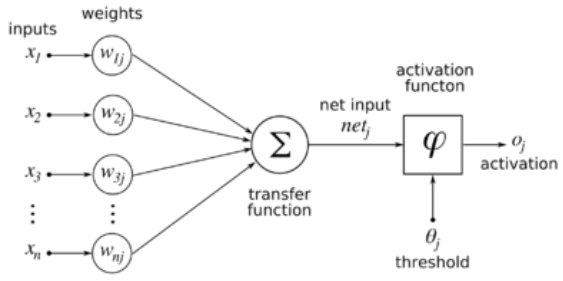


Рис. 1.1. Модель персептрона [1].

Багатошаровий персептрон (MLP) – це штучна нейронна мережа, що складається з багатьох персептронів. На відміну від одношарових персептронів, MLP здатні навчитися обчислювати нелінійно-відокремлювані функції. Оскільки вони можуть вивчати нелінійні функції, вони є однією з основних технік машинного навчання як для регресії, так і для класифікації при контрольованому

навчанні. MLP зазвичай організовані у вигляді послідовних прошарків (рис. 1.2.). Штучна нейронна мережа прямого поширення складається з вхідного шару, деякої кількості (можливо, нуля) прихованих шарів та вихідного шару. У випадку одношарового персептрона прихованих шарів немає, тому загальна кількість шарів – два. MLP, навпаки, мають принаймні один прихований шар, кожен з яких складається з колекції персептронів.

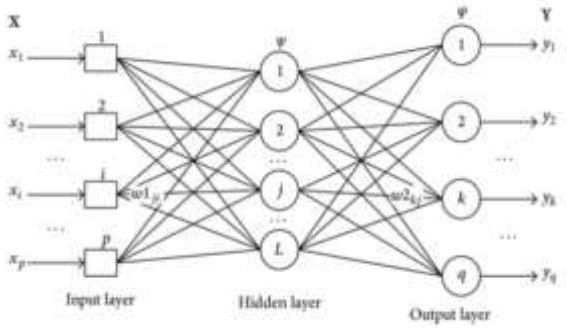


Рис. 1.2. Багатошаровий персептрон [1].

1.3. Згорткові нейронні мережі

Проаналізувавши наукові джерела можемо запропонувати таке розуміння згорткових нейронних мереж [2]: Згорткові нейронні мережі (CNN) – це архітектура нейронних мереж глибокого навчання, призначена для опрацювання структурованих масивів даних, таких як зображення. Згорткові нейронні мережі широко використовуються в комп'ютерному зорі – технологія опрацювання графічної інформації (розпізнавання, класифікація, кластеризація зображень). Також використання нейромережі такої архітектури досягли успіху в опрацюванні природних мов, в класифікації тексту.

Згорткові нейронні мережі – це нейронні мережі прямого поширення, які мають хоча б один згортковий прошарок призначений для розпізнавання дрібних образів, таких як лінія, гострі кути, градієнти, тощо. Згорткові нейронні мережі базуються на архітектурі нейронних мереж прямого поширення і влаштовані подібним чином. Вони зазвичай мають 20-30 згорткових прошарків, які накладаються один на одного. При цьому кожен наступний прошарок здатний розпізнавати все більш складніші та витонченіші форми. Таким чином згорткові

нейронні мережі можуть досліджувати ієрархічні особливості образу. Після ряду згорткових прошарків, як правило, присутні декілька (один і більше) повнозв'зних прошарків прямого поширення (MLP) (Рис. 1.3.).

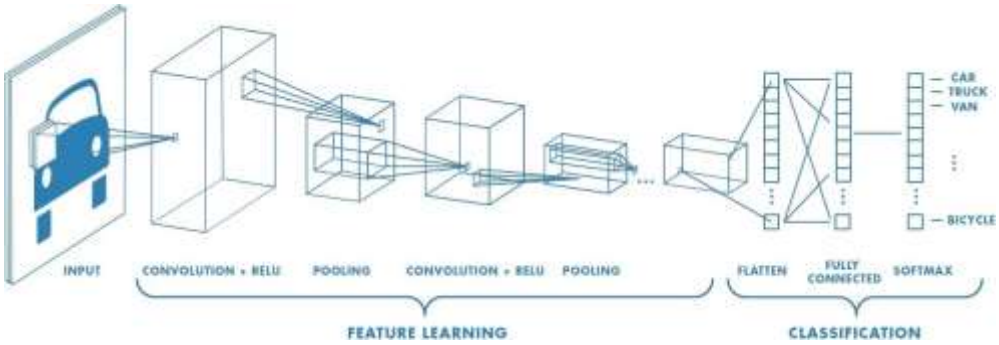


Рис. 1.3. Згорткова нейронна мережа [2].

Розберемо влаштування загорткової нейронної мережі на прикладі LeNet-5 (рис. 1.4.) для класифікації зображень розміром 32×32 пікселі за 10 категоріям, результати наведено у таблиці 1.1.

Таблиця 1.1.

Структура нейронної мережі LeNet-5

Прошарок	Опис
C_1	Перший згортковий прошарок, складається з шести згорткових ядер (5×5), передає до наступного прошарку шість зображень (28×28), призначений для розпізнавання найголовніших деталей зображення.
S_2	Перший підвибірковий прошарок, також відомий як середній шар об'єднання, передає до наступного прошарку шість зображень (14×14), призначений для усереднення блоків (2×2) до одного пікселя.
C_3	Другий згортковий прошарок, складається з шістнадцяти згорткових ядер (5×5), передає до наступного прошарку шістнадцять зображень (10×10).

S_4	Другий підвибірковий прошарок, передає до наступного прошарку шістнадцять зображень(5×5), призначений для усереднення блоків (2×2) до одного пікселя.
C_5	Третій згортковий прошарок, передає до наступного прошарку одномірний масив з 120 елементів, призначений для перетворення масиву (5×5×16) в одномірний.
F_6	Перший та єдиний повнозвя'зний прошарок, передає до наступного прошарку масив з 10 елементів.
Вихідний прошарок	Softmax прошарок, повертає масив з 10 елементів, призначений для інтерпретації результатів в категорії ймовірностей (сума елементів масива дорівнює одиниці – 100%)

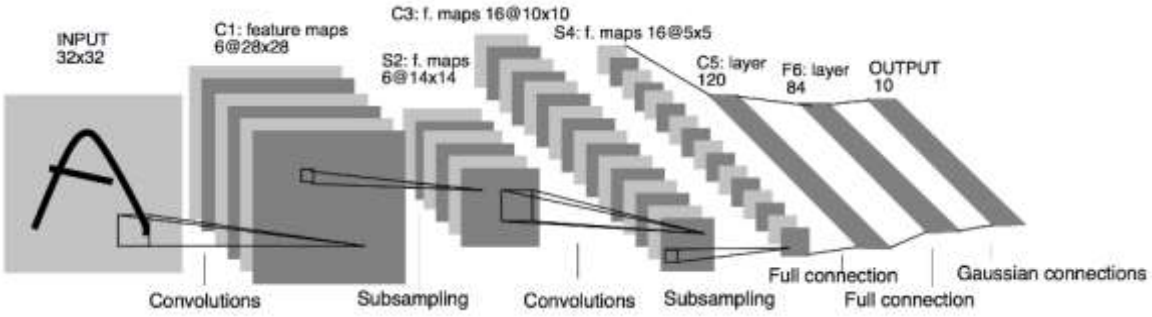


Рис. 1.4. Модель LeNet-5 [2].

Згортковий прошарок (рис. 1.5.) – це основний структурний елемент згорткових нейронних мереж. Головна частина такого прошарку – це згорткове ядро. Після того як вхідний тензор був розбитий на ряд тензорів розміру аналогічного до розміру згорткового ядра, знаходиться тензор скалярних добутків отриманих тензорів на саме згорткове ядро.

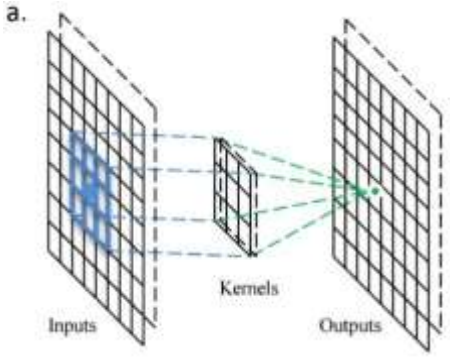


Рис. 1.5. Принцип роботи згорткового ядра [2].

1.4. Рекурентні нейронні мережі

Узагальнюючи дані подані в статті пропонуємо таке визначення [3]: Рекурентні нейронні мережі (RNN) – це архітектура нейронних мереж, призначена для опрацювання інформації типу «серія», послідовностей, або колекцій з не лімітованим розміром (рис. 1.6.).

Рекурентні нейронні мережі обробляють кожну одиницю вхідної інформації враховуючі результати опрацювання попередньої одиниці. RNN не мають лімітованих входів і виходів, вони можуть приймати та повертати будь яку кількість структурованих масивів даних. На вихід RNN впливає не тільки вагові коефіцієнт, як у звичайної нейронної мережі прямого поширення, але і прихований вектор стану - контекст попередньо обробленої інформації. Отже, один і той же вхід може давати різний результат залежно від попередніх входів у серії.

Підводячи підсумок, в MLP розмір вхідного вектору фіксований, як і розмір вихідного вектора. Така мережа стає рекурентною коли до неї неодноразово застосовується перетворення до ряду заданих входів і створюється серія вихідних векторів. Не існує попередньо встановленого обмеження на розмір вхідного і вихідного вектора. Окрім генерування вихідних даних, RNN оновлює прихований стан мережі на основі вхідних даних і використовує його при обробці наступного вхідного сигналу.

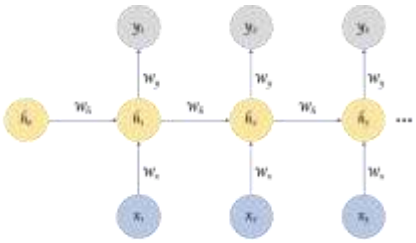


Рис. 1.6. Рекурентна нейронна мережа [3].

1.5. Алгоритм навчання нейронних мереж

1.5.1. Градієнтний спуск

Будь-який алгоритм навчання нейронних мереж націлений на мінімізацію помилки, оціненої за деяким критерієм якості. Найпопулярнішим алгоритмом

навчання нейронних мереж є алгоритм «Backpropagation», оснований на алгоритмі «градієнтний спуск». «градієнтний спуск – це ітераційний алгоритм оптимізації першого порядку, в якому для знаходження локального мінімуму функції здійснюються кроки, пропорційні протилежному значенню градієнту функції в поточній точці» [4].

Розглянемо принцип роботи градієнтного спуску (рис. 1.7.) для мінімізації функції $F(x)$, де $F(x)$ – визначена, диференційована. Початкова точка a. Для мінімізації вищеописаної функції потрібно зробити крок в напрямку $-\nabla F(a)$. Для визначення довжини кроку використовується коефіцієнт λ при $-\nabla F(a)$. Отже для мінімізації функції будемо використовувати правило: $a_{t+1} = a_t - \lambda \cdot \nabla F(a_t)$. При цьому виконується нерівність $F(a_{t+1}) \leq F(a_t)$.

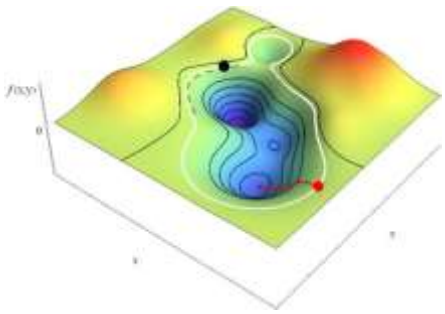


Рис. 1.7. Градієнтний спуск [4].

1.5.2. Опис алгоритму «Backpropagation»

Backpropagation – це узагальнена назва групи алгоритмів навчання нейронних мереж, які основані на алгоритмі градієнтного списку (рис. 1.8.). До цієї групи входять такі класи оптимізаторів: Adadelta, Adagrad, Adam, Adamax, Ftrl, Nadam, RMSprop, SGD.

Розглянемо принцип роботи алгоритму Backpropagation. Помилку нейронної мережі будемо визначати, як $E = C(t, y)$, де $C(t, y)$ – функція втрат, t – цільовий результат, y – фактичний результат. Для кожного нейрона j, його вихід v_j будем визначати як зазначено у формулі (1.1.)

$$v_j = \varphi(net_j) = \varphi\left(\sum_{k=1}^{n} \omega_{kj} \cdot v_k\right) \quad (1.1.)$$

де, net_j – вхідна інформація, k – нейрони попереднього прошарку, ω – ваговий коефіцієнт, φ – функція активації. Визначати локальні градієнти кожного

нейрона l вихідного прошарку будемо, як $\delta_l = E \cdot \varphi'(v_k)$, де v_k – сума вхідних сигналів. Визначати локальний градієнт для кожного нейрона j вхідного та прихованих прошарків будемо як $\delta_j = \delta_n \cdot \omega_{jn} \cdot \varphi'(v_k)$, де v_k – сума вхідних сигналів, n – нейрон наступного прошарку. Отже ваги будемо корегувати за формулою (1.2.):

$$\omega_{kj} = \omega_{kj} - \lambda \cdot \delta_j \cdot v_k \, (1.2.)$$

де, j – нейрон ваговий коефіцієнт якого корегується, k – нейрон попереднього прошарку, λ – крок збіжності (коефіцієнт корекції) (рис. 1.8.).

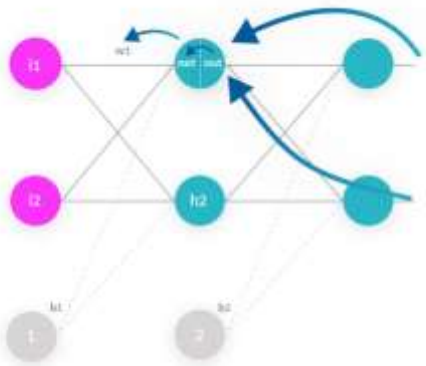


Рис. 1.8. Алгоритм Backpropagation [4].

1.5.3. Критерій якості та функція втрат

Важливою частиною процесу машинного навчання є оцінка втрат моделі (помилки). Для оцінки втрат використовують критерії якості. Критерій якості – це функція, яка оцінює розбіжність між практичними передбаченнями моделі та цільовими (істинними). Критерій якості який використовується безпосередньо в процесі навчання називається функцією втрат. При моделюванні нейронних мереж всі дані діляться на дві або три підвибірки: навчальна, валідаційна, тестова. Навчальна вибірка – це масив спостережень, який використовується при навчанні моделі. Валідаційна вибірка – це масив спостережень, який використовується для об'єктивної оцінки якості навчання нейронної мережі. Якщо валідаційна підвибірка якимось чином бере участь у процесі навчання (наприклад, зростання втрат на валідаційній вибірці може стати приводом для зупинки процесу навчання), то вводиться тестова підвибірка. На основі тестової вибірки оцінюються втрати моделі вже після завершення процесу навчання. Валідаційна та тестова підвибірки,

як правило становлять, 20-25% від навчальної (кожна). Розглянемо основні критерії якості які використовуються для моделювання нейронних мереж при розв'язуванні задач регресії:

$$MAE\ (mean\ absolute\ error) = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n};$$

$$MAPE\ (mean\ absolute\ percentage\ error) = \frac{100\%}{n} \cdot \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|;$$

$$MSE\ (mean\ squared\ error) = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n};$$

$$LC\ (log\ cosh) = \sum_{i=1}^{n} (\log(\cosh(y_i - \hat{y}_i)));$$

$$CS\ (cosine\ similarity) = \frac{\sum_{i=1}^{n} (y_i \cdot \hat{y}_i)}{\sqrt{\sum_{i=1}^{n} y_i^2} \cdot \sqrt{\sum_{i=1}^{n} \hat{y}_i^2}}.$$

В наведених формулах n – це кількість спостережень у вибірці, $\hat{y}$ – практичні передбачення моделі, y – цільові передбачення.

1.6. Біржова торгівля та типи торгових угод

Для того, щоб розробити якісно продуктивне програмне забезпечення для торгівлі криптовалютою необхідно з'ясувати, які типи торгових угод існують та як відбувається відкриття, фіксація позиції, яка комісія береться з відкриття/закриття угоди, які особливості та додаткові умови позиції характерні для кожного типу торгових угод.

Найпростіший варіант торгівлі – це спотові угоди. Охарактеризуємо даний тип угод. Спотовий ринок – це ринок де можна торгувати фінансовими інструментами напряму. Це означає, що на спотових ринках можна придбати криптовалюту (або акції, дорогоцінні метали, матеріали, енергоресурси тощо), якою можна вільно розпоряджатися поза біржою, тобто мати у власності. Це найменш ризикований тип торгівлі, головним недоліком якого є високі комісії (наприклад, на біржі binance: 1% від об'єму позиції, сплачується як при відкритті, так і при закритті позиції) [5]. Проте, існують інші види угод, які не постачають базового активу, вони називаються деривативами (похідні угоди). Найрозповсюдженішим видом деривативів є ф'ючерсні угоди. Охарактеризуємо ф'ючерси. Ф'ючерс – це угода, яка встановлює зобов'язання купити, або продати фінансовий актив в майбутньому за заздалегідь встановленою ціною. Ф'ючерси є

термінові та безтермінові. У випадку з терміновими ф’ючерсами угода повинна бути виконана в конкретний момент часу, що встановлений в договорі. Безтермінові ф’ючерси напроти, можуть бути виконані в будь-який момент часу [6]. Проте, за підтримку безтермінової ф’ючерсної позиції необхідно регулярно платити ставку фінансування (наприклад, на біржі binace кожні 8 годин), що може зробити довгострокову позицію заздалегідь збитковою. Ставка фінансування динамічно змінюється, тому розрахувати точну суму, що буде сплачена за підтримку позиції неможливо. Головною перевагою ф’ючерсів є низька комісію. Залежно від типу заявки, по якій буде відкрита, або закрита позиція комісія на binance складає 0.2% (лімітний ордер), або 0.4% (ринковий ордер). Також розглянемо такий тип торгових угод, як опціони [7]. Опціон – це угода, яка передбачає можливість купівлі (продажу) фінансового активу по мінімальній (максимальній) ціні за певний період часу, що називається терміном експірації. Характерною особливістю опціонів є термін експірації та страйк. Страйк це рівень, який повинна перетнути ціна, для того, щоб угода увійшла в силу. Від страйку залежить ціна опціону.

Висновки до розділу 1

Було детально проаналізовано теоретичні засади дослідження, знання та розуміння яких необхідне для досягнення мети дослідження. Дано визначення терміну «нейронна мережа». Досліджено основні типи нейронних мереж: нейронні мережі прямого поширення, згорткові нейронні мережі, рекурентні нейронні мережі. Ми проаналізували найрозповсюдженіші алгоритми навчання нейронних мереж, їх складові та математичний опис алгоритму. Було описано види торгових угод, що необхідно для того, щоб визначити під яку задачу буде написано програмне забезпечення. Підсумовуючи особливості торгових умов, було зроблено висновок, що необхідно розробляти програмне забезпечення для торгівлі ф’ючерсними контрактами. Це означає, що нейронна мережа повинна буде прогнозувати зміни в курсі в межах дня, а це встановлює особливі вимоги до навчальної вибірки.

РОЗДІЛ 2

Особливості розробленого програмного забезпечення на основі нейронних мереж для прогнозування курсу криптовалют

2.1. Практичні аспекти використання нейронних мереж в дослідницькій роботі та архітектура розробленого програмного забезпечення

Для того, щоб визначитися з типом нейронної мережі, спочатку треба проаналізувати дані, які можна зібрати під навчальну вибірку. З відкритих джерел інформації можна дістати дуже обмежений набір даних, при чому історичних даних, які можна було б використати в навчальній вибірці немає зовсім. Тому, навчальну вибірку довелося збирати в реальному часі, а складається вона з восьми фундаментальних показників і більш ніж сотні технічних. Виходячи з того, що даних для навчання не вистачає, а структура вхідних даних складна, було вирішено використовувати нейронні мережі прямого розповсюдження.

Після того, як було визначено тип нейронних мереж, під які буде написано програмне забезпечення та визначено дані, які можливо зібрати, необхідно визначитися з архітектурою програмного забезпечення. Раціональна та продумана архітектура програми – це основна передумова її результативності. Адже, якщо програма буде неправильно побудована, то вносити будь-які правки, додавати новий функціонал, видаляти існуючий, адаптувати програмний код під іншу задачу буде надто тяжко. Архітектуру було розбито на два рівні: функціональний та скриптовий. Весь функціонал програми записаний в трьох класах, що відповідають за збір інформації, аналіз інформації та весь функціонал пов’язаний з нейронними мережами. Кожен клас не пов’язаний з іншим та не залежить від нього. Клас зберігає всі дані та весь функціонал, що необхідний для його роботи. Весь функціонал максимально гнучкий. В скриптах використовується функціонал цих трьох класів, прописується як саме треба використати цей функціонал. Нижче наведено спрощене схематичне влаштування програмного забезпечення та скриптів parser (підготовка навчальної вибірки) і model (надання сигналу на купівлю, або продаж) (рис. 2.1, 2.2, 2.3.).

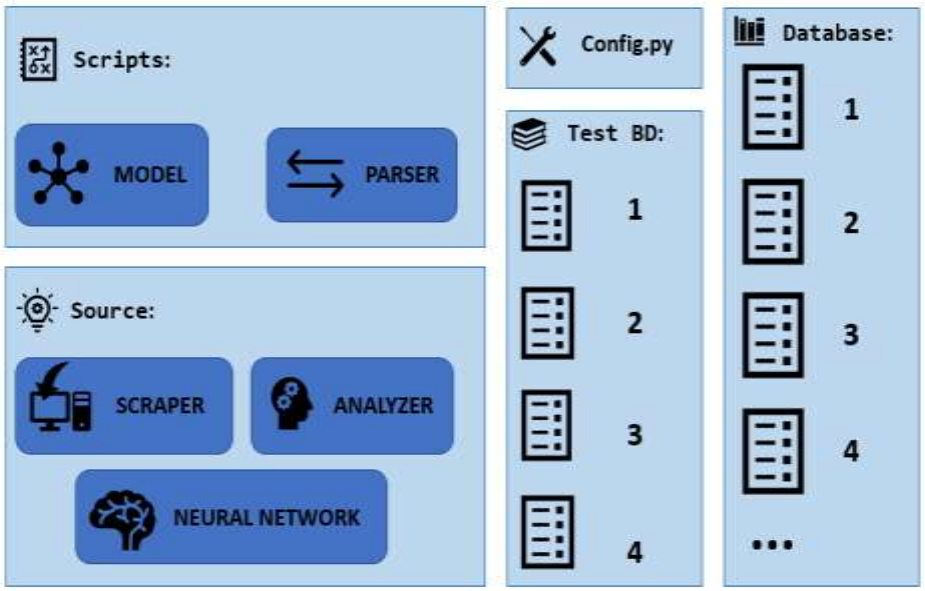


Рис. 2.1. Архітектура програмного забезпечення.

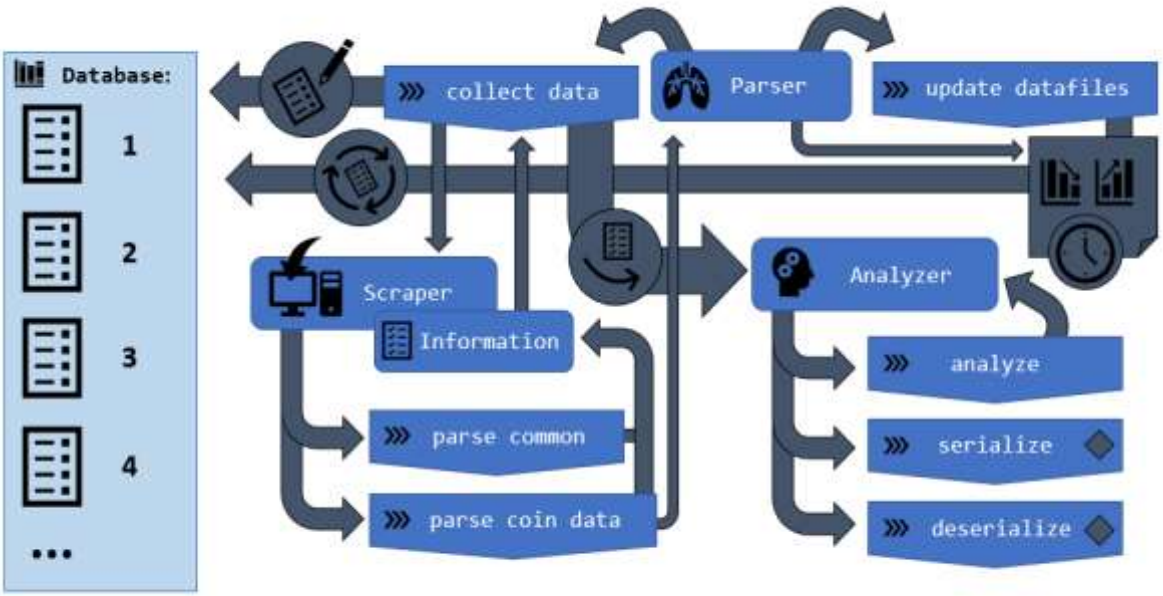


Рис. 2.2. Влаштування скрипта Parser.

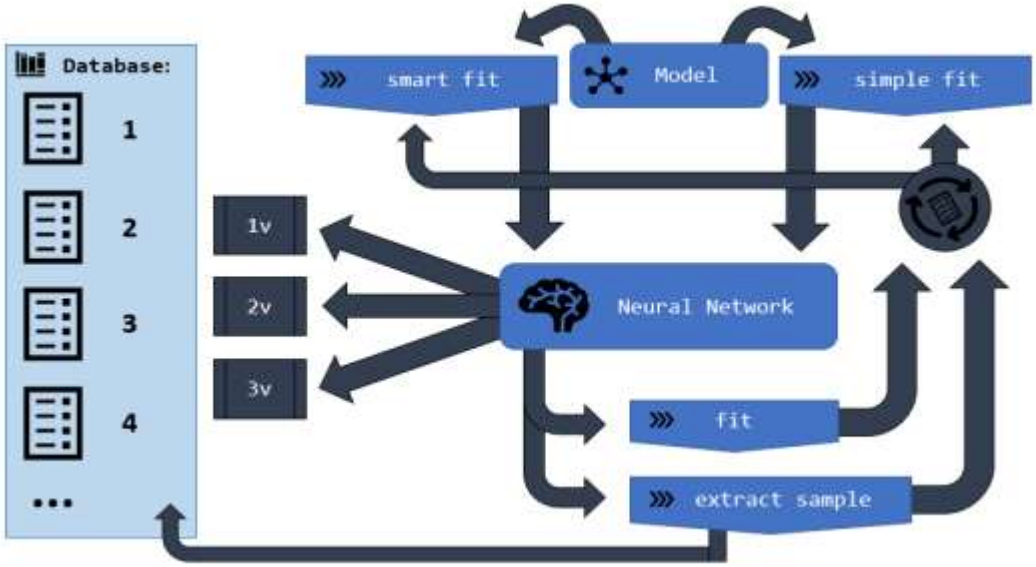


Рис. 2.3. Влаштування скрипта Model.

На схемах відображено лише невелика частина скриптів, методів та функцій, які реалізовані в програмному забезпеченні. Для максимальної гнучкості розробленого програмного забезпечення було передбачено три можливі конфігурації нейронної мережі (під різні методи навчання та різний характер навчальної вибірки) та два кардинально різні методи навчання нейронних мереж. Перший метод розроблений, щоб взяти від навчальної вибірки максимум (найкраще підходить в умовах обмеженої навчальної вибірки), у другий метод вписана складна логіка, яка перешкоджає перенавчанню нейронної мережі (після навчання таким чином, нейронна мережа повинна знайти загальну закономірність і не підлаштовуватись під окремі випадки).

2.2. Аналіз технічних рішень, зроблених під час розробки програмного забезпечення

Весь код написаний на мові програмування Python, зібрані дані зберігаються в теці, кожне спостереження (одиниця даних з навчальної вибірки) зберігається в окремому файлі формату JSON. На знімках екрана знаходяться методи, що відповідають за сереалізацію та десеріалізацію спостережень з бази даних (рис. 2.4, 2.5, 2.6.).

```
96      def serialize(self):
97
98          info = {...}
113
114         output_dict = {"Information": info,
115                        "Conclusion": self.conclusion,
116                        "Model_input_full": self.model_input_full,
117                        "Model_input_base": self.model_input_base,
118                        "Result": self.result}
119
120         with open(self.directory + self.info.datafile_index, "w") as outfile:
121             json.dump(output_dict, outfile)
```

Рис. 2.4. Метод serialize.

```
192     def deserialize(self):
193         datafiles = next(walk(self.directory), (None, None, []))[2]
194         data = []
195         for file_name in datafiles:
196             if "json" in file_name:
197                 with open(self.directory + file_name, 'r') as input_file:
198                     data.append(json.loads(input_file.read()))
199         return data
200
```

Рис. 2.5. Метод deserialize.

```python
def collect_data(test_mode=True):
    scraper_obj = Scraper(print_opt=False, test_mode=test_mode)
    scraper_obj.parse_common()

    for s in SYMBOLS:
        print("-" * 30, s, "-" * 30)
        scraper_obj.set_symbol(s)
        scraper_obj.parse_coin_data()

        info = scraper_obj.info
        print(f"(finished scraping)")

        analyzer_obj = Analyzer(info, test_mode=test_mode)
        analyzer_obj.analyse()
        analyzer_obj.serialize()
        print(f"(finished analysis)")
        print("-" * 61, "\n")


def update_datafiles(test_mode=True):
    scraper_obj = Scraper(print_opt=False, test_mode=test_mode)
    analyzer_obj = Analyzer(info=scraper_obj.info, test_mode=test_mode)

    data = analyzer_obj.deserialize()
    for item in data:
        analyzer_obj.update_datafile(item, scraper_obj)
```

Рис. 2.6. Функціонал відповідальний за збір та оновлення навчальної вибірки.

2.3. Використання нейронних мереж в розробці

В програмі використано бібліотеки: os, tensorflow, keras, pandas, matplotlib, requests, lxml, numpy, json та ще 9 інших бібліотек.

В програмному забезпеченні прописаний весь необхідний функціонал для генерації, компіляції, навчання нейронної мережі, отримання та опрацювання вихідних сигналів. Замість однієї налаштованої нейронної мережі є можливість створювати та навчати 72 кардинально різні моделі з різними налаштуваннями, причому, кожна з 72-ох конфігурацій підлаштовуються під вирішення задачі прогнозування курсу.

Як приклад, наведем одну із структур нейронної мережі, разом з налаштуваннями моделі та параметрами навчання, що використано в програмному

забезпеченні. Нейронна мережа має 10 прошарків з такою послідовністю нейронів: 120, 220, 440, 360, 240, 180, 90, 40, 10, 1. Активатор нейронів: sigmoid. Функція втрат: MSE. Оптимізатор: SGD, з методом моментів. Коефіцієнт збіжності алгоритму: 0.0006. Коефіцієнт momentum: 0.1. Валідаційна вибірка складає 20% від загальної. Максимальна кількість епох навчання: 1600. Навчання зупиняється, якщо фіксується дивергенція втрат по навчальній та валідаційній вибірці протягом 40 епох навчання. Нейронна мережа аналізує 106 параметрів, серед яких: index S&P500, DXY, ціна золота та срібла, alt-season index, fear and gear index, індикатори RSI, CCI, ADX&DI, MACD, Stochastic, Momentum та багато інших параметрів.

На знімках екрану знаходиться скрипт model, що відповідає за надання сигналів на купівлю, або продаж (рис. 2.7, 2.8.).

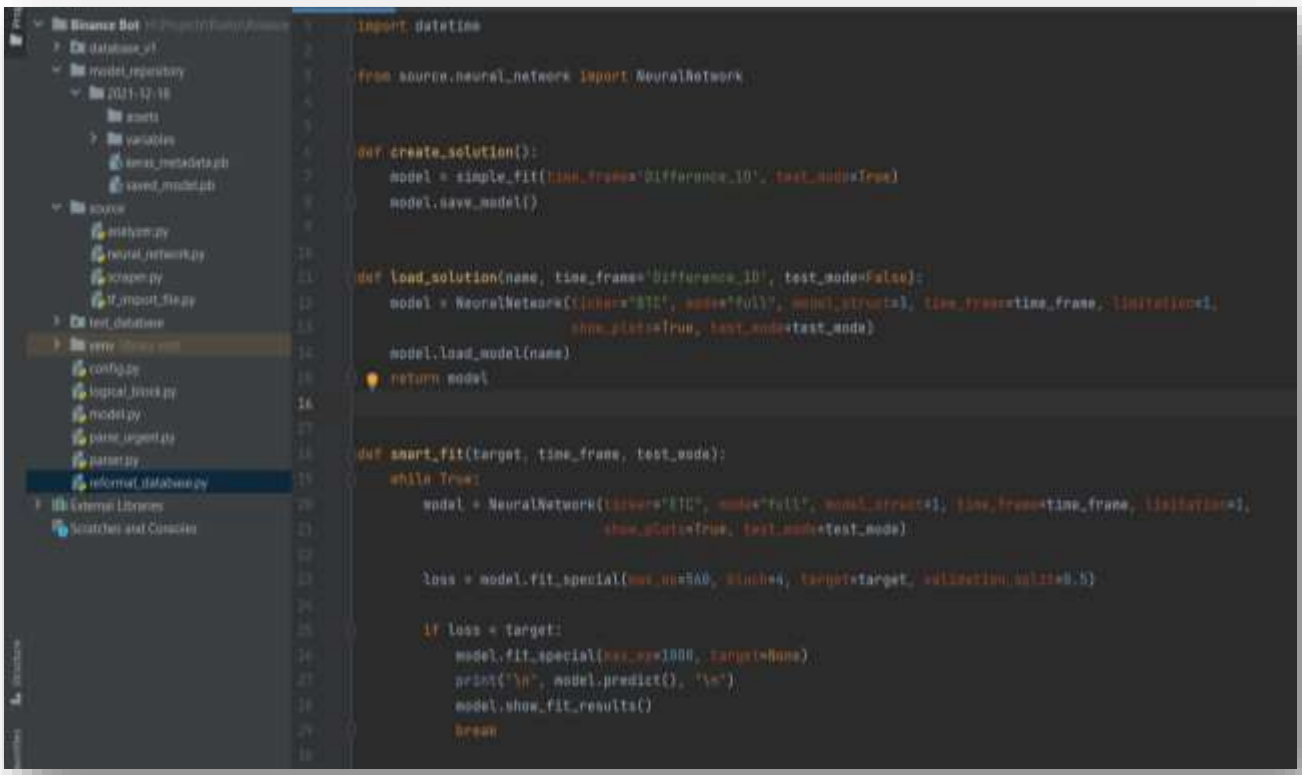


Рис. 2.7. Скрипт model.

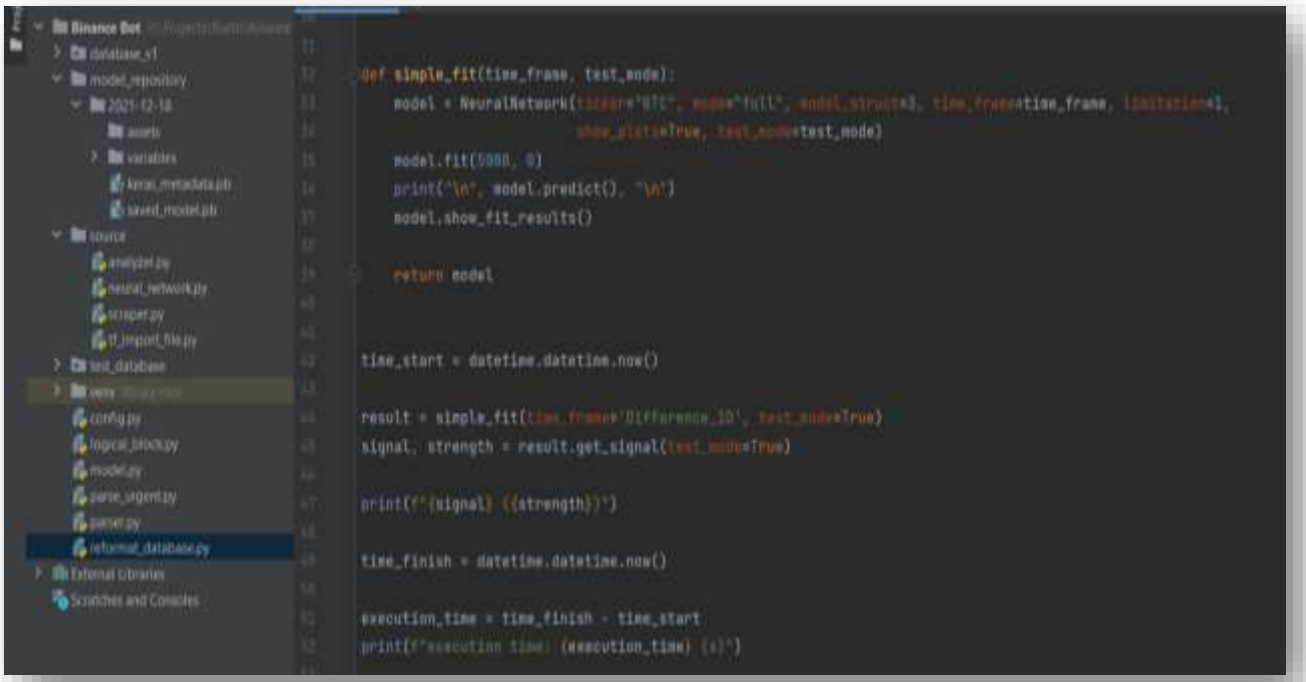


Рис. 2.8. Скрипт model (продовження).

Всі необхідні дані збираються за допомогою бібліотеки «requests», це найпростіший та найшвидший спосіб збору даних з веб-сайтів. Для вирішення задачі, де треба збирати величезні об’єми даних, вирішено не використовувати бібліотеки, що забезпечують автоматизацію, адже це не є найраціональнішим варіантом як по складності коду, так і по швидкості виконання.

Було підібрано максимально ефективні конфігурації нейронних мереж та методи навчання, що дало можливість алгоритму навчання точно та однозначно визначати зв'язок між вхідними та вихідними параметрами. В результаті маємо програму, що дає точні прогнози курсу криптовалют. Під час першого етапу тестування програма продемонструвала теоретичну точність в 98% (за навчальною вибіркою) і практичну точність прогнозування в 71% (на реальних торгових угодах). Було виявлено недоліки, помилки, нераціональні рішення. Серед них: не оптимізований алгоритм збору даних, недосконалий метод обробки навчальної вибірки, не раціональний метод ініціалізації класу Analyzer, помилки в послідовності спостережень у вибірці, хибні дані про історію котирувань тощо. Всі виявлені недоліки були виправлені, було доопрацьовано алгоритм прийняття

торгових рішень. Другий етап тестування показав практичну точність прогнозування в 90% (9 правильних прогнозів з 10 отриманих).

В додатках (додатки A, B, C, D) наведено приклади прогнозів (за чотири дні), побудованих на основі сигналів нейронної мережі під час випробування розробленого програмного забезпечення. На першому зображенні, в кожному з додатків накреслено прогноз в середовищі TradingView. На кожному зі знімків екрану видно дату та час, коли знімок був зроблений (на панелі задач). На знімках екрану видно назву фінансового інструменту та джерело даних про котирування інструменту по якому будується прогноз (легенда в лівій верхній частині графіка з TradingView): “Bitcoin / TetherUS • BINANCE • TradingView”. Горизонтальні лінії нижче ціни – найвірогідніші відмітки, яких може досягти ціна при падінні, там необхідно добирати (фіксувати) частину позиції, залежно від типу позиції (LONG/SHORT). Горизонтальні лінії вище ціни – найвірогідніші відмітки, яких може досягти ціна при зростанні, необхідно фіксувати (добирати) частину позиції, залежно від типу позиції. Друге зображення в Додатку A (аналогічні є в Додатках B, C) демонструє візуалізацію прогнозу нейронної мережі. Друге зображення в Додатку B (аналогічні є в Додатках C, D) відображає прогноз нейронної мережі та результати опрацювання цього прогнозу в консолі середовища PyCharm. Відповідно до логіки, закладеної в алгоритм програми сигнали нейронної мережі можуть бути відфільтровані як: занадто слабкий (або не підтверджений), мало підтверджений, достатньо підтверджений. Від того, як був відфільтрований сигнал нейронної мережі необхідно відштовхуватися при аналізі інших метрик та індикаторів про побудові прогнозу.

Висновки до розділу 2

Було спроектовано архітектуру програмного забезпечення та визначено основні практичні аспекти використання нейронних мереж в даній дослідницькій роботі. Описано технічні рішення, які були реалізовані для досягнення кінцевої мети дослідження. Кінцева мета дослідницької роботи була досягнута. Програмне забезпечення для прогнозування курсу криптовалют було розроблене та

протестоване, недоліки розробки, що були виявлені під час використання, були виправлені.

ВИСНОВКИ

Паніка на ринку криптовалют, що нещодавно призвела до падіння ринкової капіталізації на 35% продемонструвала кардинальні зміни в ринкових закономірностях. Індикатори та стратегії, які демонстрували високі результати роками, виявилися застарілими та неефективними. Коли фінансові ринки росли роками (через програму стимулювання ФРС США), ріс і криптовалютний ринок. Ріст був тривалим та непохитним і це не дало змогу вчасно розпізнати застарілість існуючих індикаторів, торгових стратегій та торгових систем. Зараз, коли зміни на ринку вже очевидні, людям доводиться переглядати існуючі торгові стратегії та системи, шукати нові закономірності та писати нові індикатори. Стосовно індикаторів, вони не здатні адаптуватися під нові умови, що відбуваються постійно. В основі індикатора лежить математична формула, яку в найкращому випадку доведеться переглядати, в найгіршому – забути про неї через деякий час. Натомість індикатор написаний на основі технологій машинного навчання (не обов'язково на основі нейронної мережі) може адаптуватися під нові умови, після кожного перенавчання моделі. Замість того, щоб індикатор з кожним днем все більше втрачав свою ефективність, індикатор на основі технологій машинного навчання буде ставати краще та краще щодня. Саме тому, тема дослідження є актуальною, а розробка ефективною та результативною. Всі завдання дослідницької роботи були виконані, мету досягнуто.

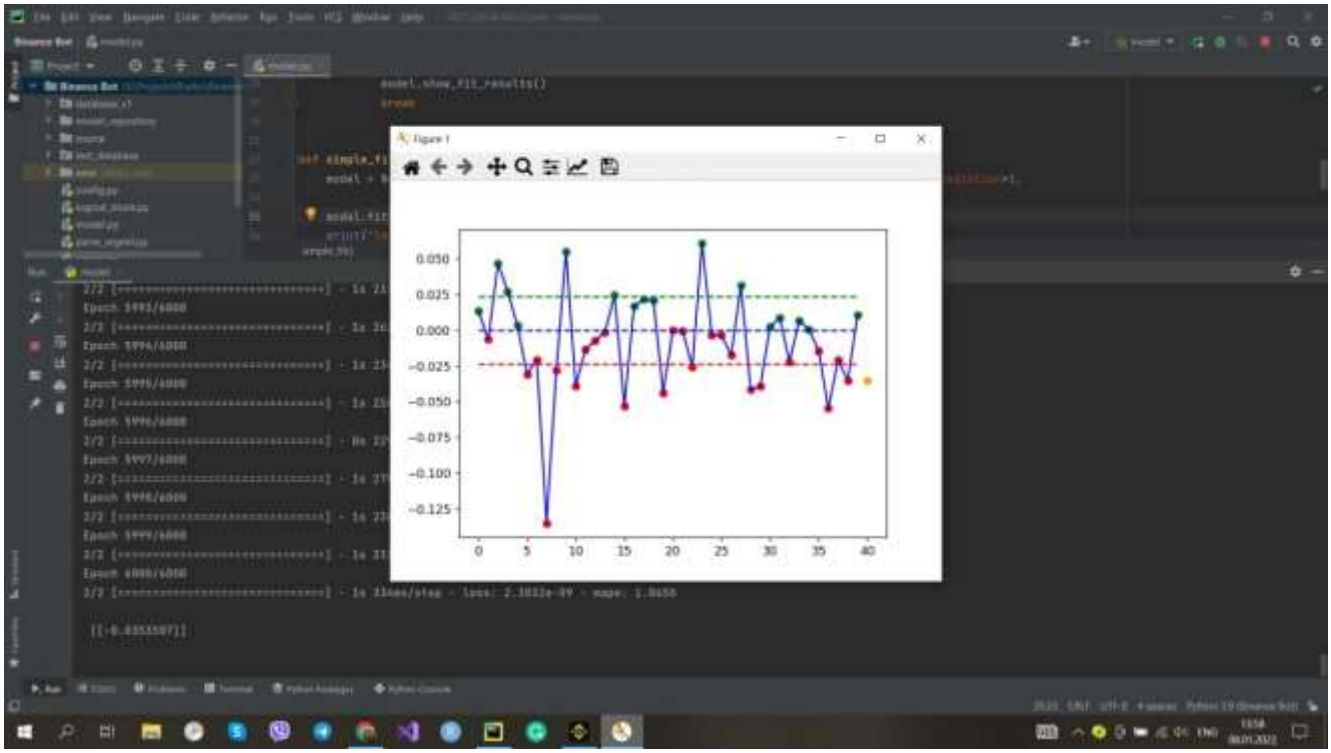
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Feed-Forward networks. URL : https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Architecture/feedforward.html.
2. Wood T. Convolutional Neural Network. URL: https://deepai.org/machine-learning-glossary-and-terms/convolutional-neural-network.
3. Venkatachalam M. Recurrent Neural Networks. URL: https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce.
4. Gradient descent, Wikipedia. URL: https://en.wikipedia.org/wiki/Gradient_descent.
5. What Is a Spot Market and How to do Spot Trading. URL: https://academy.binance.com/en/articles/what-is-a-spot-market-and-how-to-do-spot-trading.
6. What is futures market. URL: https://www.investopedia.com/terms/f/futuresmarket.asp.
7. What Is an Option. URL: https://www.investopedia.com/terms/o/option.asp.

ДОДАТКИ

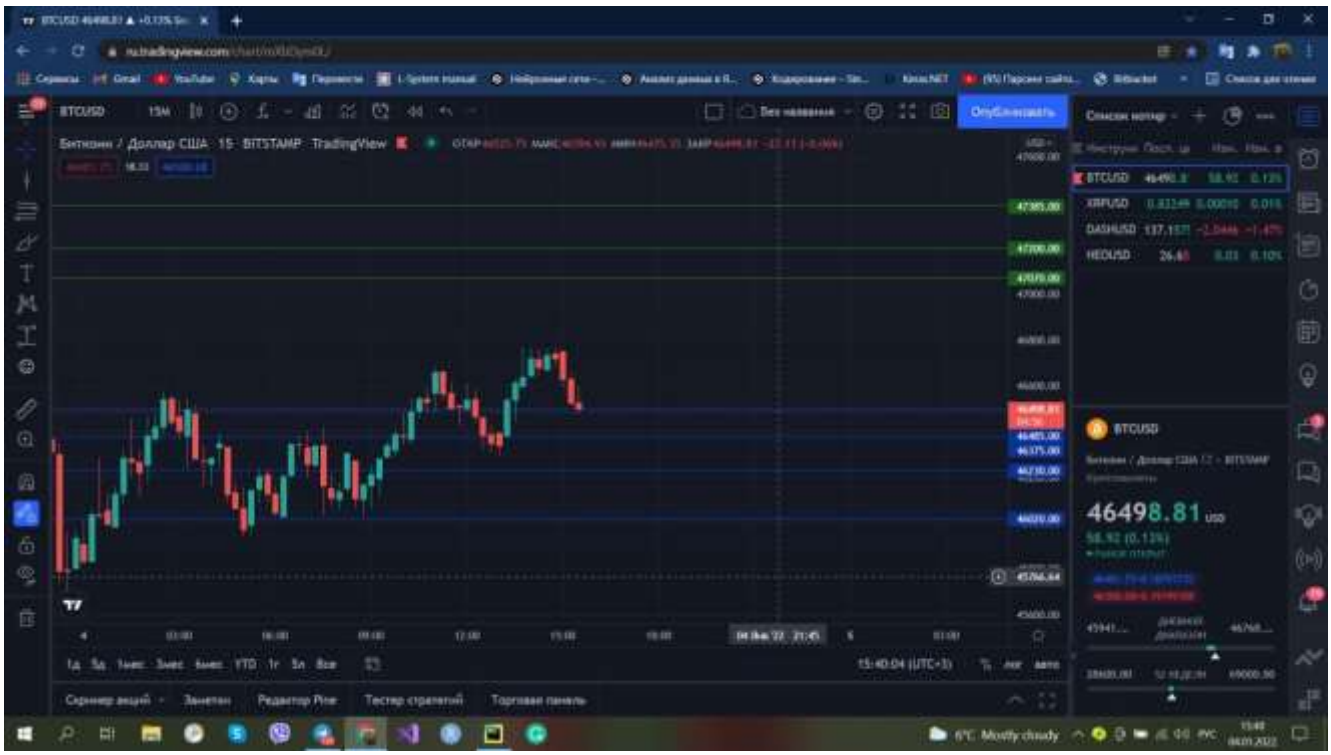
Додаток А

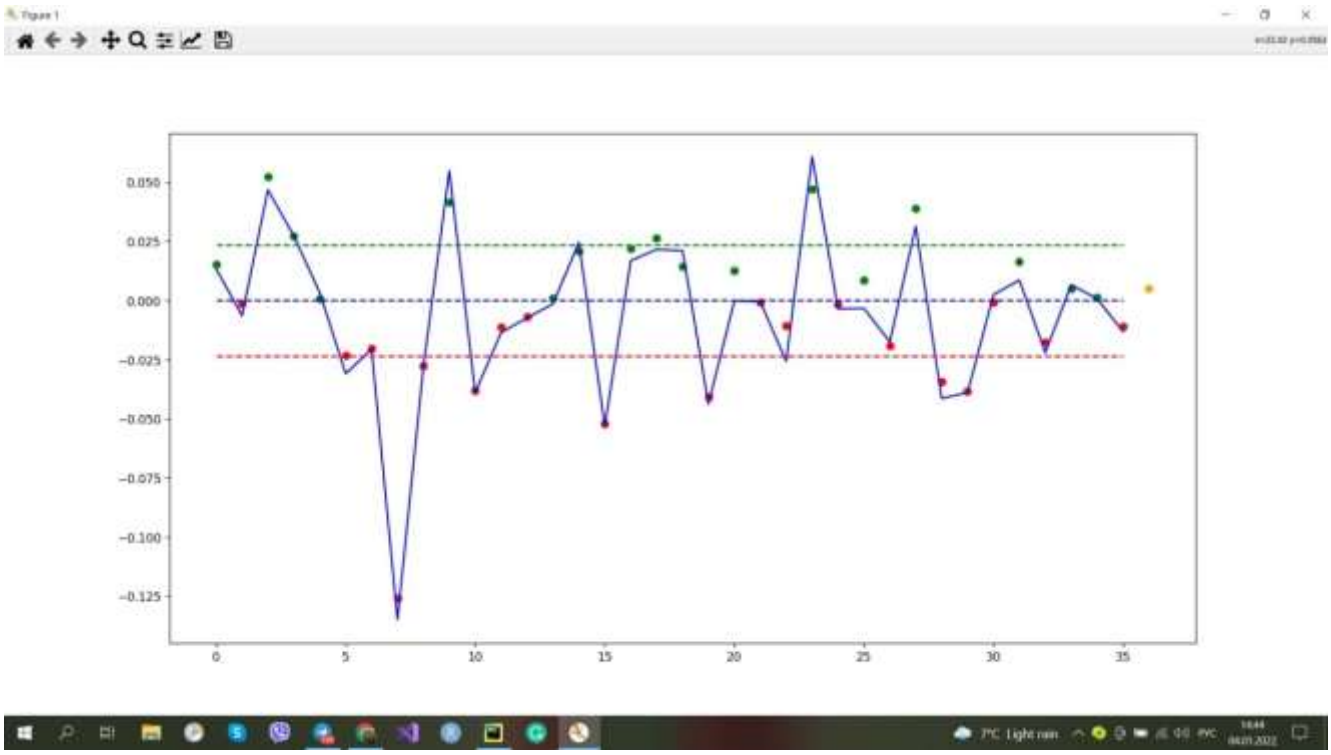
Побудова прогнозу від 08.01.2022 р.



Додаток Б

Побудова прогнозу від 04.01.2022 р.





Додаток В

Побудова прогнозу від 02.01.2022 р.