НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

«На правах рукопису»
УДК 004.94

«До захисту допущено»
Завідувач кафедри
_____ О.Л. Тимощук
«___»_____20__ р.

Магістерська дисертація
на здобуття ступеня магістра
за освітньо – науковою програмою
зі спеціальності 122 Комп'ютерні науки
на тему: «Методи інтелектуального аналізу даних для прогнозування цін на нафту»

Виконав:
студент ІІ курсу, групи КА – 03мп
Сніжко Андрій Сергійович

_____

Керівник:
доцент кафедри ММСА, д.т.н,
доц., Недашківська Н.І.

_____

Рецензент:
доцент кафедри системного проектування
КПІ ім. Ігоря Сікорського, к.т.н, доц.,
Гіоргізова-Гай В.Ш.

_____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

Київ
2021

Національний технічний університет України
«Київський політехнічний інститут
імені Ігоря Сікорського»
Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

Рівень вищої освіти – другий (магістерський)

Спеціальність – 122 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ О.Л.Тимощук
«___»_____20__ р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Сніжку Андрію Сергійовичу

1. **Тема дисертації:** «Методи інтелектуального аналізу даних для прогнозування цін на нафту», науковий керівник дисертації Недашківська Надія Іванівна, д.т.н, доцент, затверджені наказом по університету від «02» листопада 2021 р. № 3651-с.

2. Термін подання студентом: 13 грудня 2021 р.

3. Об'єкт дослідження: Ринок нафти.

4. Предмет дослідження: Прогнозування ціни на нафту на основі історичних даних.

5. Перелік завдань, які потрібно розробити:

1. Здійснити огляд технічної літератури за темою роботи;
2. Дослідити актуальність обраної теми;

3. Ознайомитись з існуючими підходами до прогнозування нестаціонарних часових рядів;
4. Здійснити порівняльний аналіз авторегресійних моделей, ковзного середнього та гетероскедастичних моделей, ансамблевих моделей на основі дерев рішень, методів рекурентних нейронних мереж для регресії, проаналізувати їх переваги та недоліки;
5. Розробити програмний продукт для побудови і оцінювання моделей регресії. Провести експерименти щодо підбору параметрів та вибору найкращої моделі.
6. Провести аналіз результатів;
7. Провести аналіз ринкових можливостей запуску стартап-проекту;
8. Розробити концептуальні висновки;
9. Підготувати ілюстративний матеріал;
10. Оформити пояснювальну записку.

6. Орієнтовний перелік ілюстративного матеріалу:

1. Постановка завдання дослідження;
2. Практичний приклад роботи програми, розробленої в ході дослідження;
3. Структура програмного продукту;
4. Розрахункові таблиці з результатами.

7. Дата видачі завдання: 1 вересня 2021 року.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Отримання завдання на магістерську дисертацію	01.09.2021-07.09.2021	
2.	Огляд технічної літератури за темою	08.09.2021-14.09.2021	
3.	Концептуальний вступ дисертації. Формулювання об’єкта, предмета, цілі, завдань, новизни, практичної значущості результатів. Дослідження актуальності обраної теми	15.09.2021-21.09.2021	
4.	Перший розділ. Аналіз існуючих підходів до обробки природної мови	22.09.2021-28.09.2021	
5.	Другий розділ. Математичне обґрунтування та структура моделі мови	29.09.2021-05.10.2021	
6.	Розробка початкового програмного забезпечення	06.10.2021-12.10.2021	
7.	Третій розділ. Покращення якості програмного забезпечення	13.10.2021-19.10.2021	
8.	Аналіз результатів	20.10.2021-26.10.2021	
9.	Проведення аналізу ринкових можливостей стартап-проекту	27.10.2021-02.11.2021	
10.	Підготовка ілюстративного матеріалу	03.11.2021-09.11.2021	
11.	Оформлення пояснювальної записки	10.11.2021-16.11.2021	

Студент _____ А.С. Сніжко

Науковий керівник дисертації _____ Н.І. Недашківська

РЕФЕРАТ

Магістерська дисертація містить: 140 с., 3 табл., 62 рис., 1 дод. та 30 джерел.

РЕГРЕСІЯ, ДЕРЕВА РІШЕНЬ, ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ, МАШИННЕ НАВЧАННЯ, ПРОГНОЗУВАННЯ ЦІНИ, РИНОК НАФТИ.

Об’єктом дослідження є вибірка цін на нафту з 1986 року.

Предметом дослідження є методи інтелектуального аналізу даних на основі регресії з використанням дерев рішень.

Програмною мовою була обрана Python.

В даній роботі проведено дослідження ринку нафти. Для побудови моделей були використані дерева рішень та методи машинного навчання, нейронні мережі. Прогнозування було виконано на основі даних про зміну цін на нафту.

При виконанні роботи було встановлено два методи, що дають найкращі результати які достатньо близькі до реальних. Напрямок розвитку роботи є в розширенні функціоналу та зменшенні похибки прогнозування ціни. Планується додати аналіз ринку палива та встановити залежність між цінами на паливо та нафту.

ABSTRACT

Master`s thesis: 140 p., 3 tabl., 62 fig., 1 add. And 30 references.

REGRESSION, DECISION TREES, INTELLIGENT DATA ANALYSIS, MACHINE LEARNING, PRICE FORECAST, OIL MARKET.

The object of research is the selection of oil prices since 1986.

The subject of research covers methods of intelligent data analysis based on regression using decision trees.

Programming language Python.

This work is dedicated to the analysis of the oil market. Decision trees and machine learning methods were used to build the models. The forecast was conducted using the oil prices data. The prospects of development of the work depend on the expansion of the functionality and reduction the drawbacks in the price forecasting. It is planned to add the fuel market analysis to it and monitor the dependence between the fuel and oil prices.

ЗМІСТ

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

ОПЕК – організація країн-експортерів нафти;

ФВА – функціонально-вартісний аналіз;

ПП – програмний продукт;

МН – машинне навчання;

СПМ – середньоквадратична помилка моделі.

НМ – нейронна мережа

РНМ – рекурентна нейронна мережа

SVM – метод опорних векторів

SVR – регресія методом опорних векторів

GWO – grey wolf optimization

ВСТУП

19 століття було періодом великих змін і швидкої індустріалізації. Залізна та металургійна промисловість породила нові будівельні матеріали, залізниці з'єднали країну, а відкриття нафти стало новим джерелом палива. Відкриття гейзера Spindletop в 1901 році призвело до величезного зростання нафтової промисловості. Протягом року було створено понад 1500 нафтових компаній, і нафта стала домінуючим паливом 20 століття і невід'ємною частиною американської економіки.

Багато перших дослідників Америки стикалися з родовищами нафти в тій чи іншій формі. Вони відзначили нафтові плями біля узбережжя Каліфорнії в шістнадцятому столітті. Луї Еванс розташував родовища вздовж східного узбережжя на карті англійських середніх колоній 1775 року.

Сира нафта є одним з найбільш економічно зрілих товарних ринків у світі. Незважаючи на те, що більшість сирої нафти виробляється відносно невеликою кількістю компаній і часто у віддалених місцях, які дуже віддалені від точки споживання, торгівля сирою нафтою є надійною та глобальною за своєю природою. Майже 80% міжнародних трансакцій з сирою нафтою передбачають доставку по водному транспорту в супертанкерах. Нафтотрейдери можуть швидко перенаправляти операції на ринки, де ціни вищі.

Нафта та вугілля є глобальними товарами, які поставляються по всьому світу. Таким чином, глобальний попит і пропозиція визначає ціни на ці джерела енергії. Події в усьому світі можуть вплинути на наші домашні ціни на енергію на основі нафти, як-от бензин і мазут. Ціни на нафту зараз високі через швидке зростання попиту в країнах, що розвиваються (насамперед, в Азії). Оскільки попит у цих місцях зростає, все більше нафтових вантажів спрямовується до цих країн. Ціни в інших країнах повинні зрости в результаті. Політичні заворушення в деяких нафтовидобувних країнах також

сприяють високим цінам – по суті, існує побоювання, що політична нестабільність може припинити видобуток нафти в цих країнах. ОПЕК, великий нафтовидобувний картель, дійсно має певну здатність впливати на світові ціни, але вплив ОПЕК на світовому ринку нафти швидко зменшується, оскільки відкриваються та розвиваються нові поставки в країни, що не входять до ОПЕК.

Основна концепція ринку є примітивною, вона складається з двох речей – попиту та пропозиції. З першого погляду може здатися, що ріст попиту – веде до росту ціни, ріст пропозиції – веде до зменшення ціни. В реальному житті все відбувається зовсім другим чином. Реальна ціна на нафту формується на так званому ф'ючерсному ринку. Ф'ючерсний контракт – вид контракту між покупцем та продавцем, в якому чітко прописана ціна та об'єм продажу у заздалегідь визначену дату. Подібний контракт накладає на обидві сторони угоди певні зобов'язання, що повинні бути виконані вчасно. Є два типи ф'ючерсних трейдерів:

- спекулянти;
- хеджери.

Спекулянт – людина, що не збирається купувати нафту. Її ціль – заробити гроші на вгадуванні майбутнього напрямку ціни на ринку. В свою чергу прикладом хеджера може бути якась авіакомпанія, що хоче захистити себе від потенційного зростання цін.

В сьогоденних реаліях можливість прогнозування ціни на ринку нафти дає людині певні переваги та змогу заробляти кошти. Саме тому ця задача є корисною та цікавою. Я вважаю, що ця задача повинна бути розглянути з двох боків, а саме – прогнозування за допомогою дерев рішень та прогнозування з використанням нейронних мереж. Ця магістерська дисертація присвячена вивчену ринка нафти та прогнозування поведінки цін.

РОЗДІЛ 1 АНАЛІТИЧНИЙ ОГЛЯД РИНКУ НАФТИ

1.1 Історія формування ринку нафти

Перші поселенці використовували нафту для ліків, а також як мастило для возів та інструментів. Кам’яна нафта, видобута зі сланцю, стала доступною як гас ще до початку промислової революції. Подорожуючи по Австрії, Джон Остін, нью-йоркський купець, помітив ефективну дешеву масляну лампу і зробив модель, яка модернізувала гасові лампи. Незабаром американська промисловість кам’яної нафти почала процвітати, оскільки китовий жир зріс у ціні через зростаючий дефіцит цього ссавця. Семюел Даунер-молодший, ранній підприємець, запатентував «Гас» як торгову назву в 1859 році та ліцензував його використання. У міру збільшення видобутку та переробки нафти ціни обвалилися, що стало характерним для галузі.

Першою нафтовою корпорацією, яка була створена для розробки нафти, знайденої на воді поблизу Тітусвілля, штат Пенсільванія, була Pennsylvania Rock Oil Company з Коннектикуту (пізніше Seneca Oil Company). Джордж Х. Біссел, нью-йоркський юрист, і Джеймс Таунсенд, бізнесмен з Нью-Хейвена, зацікавилися, коли доктор Бенджамін Силліман з Єльського університету проаналізував пляшку олії і сказав, що вона стане чудовим світлом. Біссел і кілька друзів купили землю поблизу Тітусвілля і заручили Едвіна Л. Дрейка знайти там нафту. Дрейк найняв Вільяма Сміта, досвідченого бурильника солі, для нагляду за буровими операціями, і 27 серпня 1859 року вони видобули нафту на глибині шістдесят дев'яти футів. Наскільки відомо, це був перший випадок, коли нафта була видобута на її джерелі за допомогою бура.

Тітусвілль та інші міста в цьому районі процвітали. Одним із тих, хто чув про відкриття, був Джон Д. Рокфеллер. Завдяки своїм підприємницьким інстинктам і геніальності в організації компаній Рокфеллер став провідною фігурою в нафтовій промисловості США. У 1859 році він разом із партнером

керував комісійною фірмою в Клівленді. Незабаром вони його продали і побудували невеликий нафтопереробний завод. Рокфеллер викупив свого партнера і в 1866 році відкрив експортний офіс у Нью-Йорку. Наступного року він, його брат Вільям, С. В. Харкнесс і Генрі М. Флаглер створили те, що мало стати Standard Oil Company. Багато хто вважає Флаглера майже такою ж важливою фігурою в нафтовому бізнесі, як і сам Джон Д..

Додаткові відкриття поблизу колодязя Дрейка привели до створення численних фірм, і компанія Рокфеллера швидко почала викуповувати або об’єднуватися зі своїми конкурентами. Як сказав це Джон Д., їхня мета полягала в тому, щоб «об’єднати нашу майстерність і капітал». До 1870 року Standard став домінуючою нафтопереробною компанією в Пенсільванії.

Трубопроводи стали головним фактором у прагненні Standard отримати бізнес і прибуток. Семюель Ван Сікель побудував чотиримильний трубопровід від Пітоула, штат Пенсільванія, до найближчої залізниці. Коли Рокфеллер помітив це, він почав купувати трубопроводи для Standard. Незабаром компанія володіла більшістю ліній, які забезпечували дешеве й ефективне транспортування нафти. Клівленд став центром нафтопереробної промисловості в основному завдяки своїм транспортним системам.

Коли ціни на продукцію знизилися, паніка, що виникла, привела до початку альянсу Standard Oil в 1871 році. Протягом одинадцяти років компанія стала частково інтегрованою по горизонталі та вертикалі і стала однією з найбільших корпорацій світу. Альянс найняв промислового хіміка Германа Фраша II для видалення сірки з нафти, знайденої в Лімі, штат Огайо. Сірка дуже ускладнювала перегонку гасу, і навіть тоді він мав мерзенний запах — ще одну проблему, яку вирішив Фраш. Після цього Standard найняв науковців як для покращення свого продукту, так і для чистих досліджень. Незабаром гас замінив інші освітлювачі; він був надійнішим, ефективнішим та економнішим, ніж інші види палива.

Східні міста, пов'язані з нафтовими родовищами, залізницею та човнами, також набули популярності. Експортна торгівля з Філадельфії,

Нью-Йорка та Балтімора стала настільки важливою, що Standard та інші компанії розмістили нафтопереробні заводи в цих містах. Вже в 1866 р. вартість нафтопродуктів, експортованих до Європи, забезпечувала торговий баланс, достатній для сплати відсотків за американськими облігаціями, що зберігалися за кордоном.

Коли Громадянська війна перервала регулярний потік гасу та інших нафтопродуктів до західних штатів, посилився тиск, щоб знайти кращий метод використання нафти, знайдений у таких штатах, як Каліфорнія. Але Standard мало цікавився нафтовою промисловістю на Західному узбережжі до 1900 року. Того року вона купила Pacific Coast Oil Company і в 1906 році включила всі свої західні операції в Pacific Oil, тепер Chevron.

У 1892 році Едвард Л. Доені виявив першу свердловину Лос-Анджелеса, а через п’ять років у цьому районі було двадцять п’ять сотен свердловин і двісті нафтових компаній. Коли Standard увійшов у Каліфорнію в 1900 році, там уже процвітали сім інтегрованих нафтових компаній. Найважливішою з них була Union Oil Company[31].

Труднощі з функціонуванням плюс загроза оподаткування його власності за межами штату привели до створення в 1882 році Standard Oil Trust. У 1899 році траст створив Standard Oil Company (Нью-Джерсі), який став материнською компанією. Траст контролював корпорації-члени в основному через володіння акціями, що мало чим відрізняється від сучасної холдингової компанії.

Величезне зростання Standard не відбулося без конкуренції. У 1895 році виробники Пенсільванії задумали створення важливого конкурента, Pure Oil Company, Ltd.

У 1901 році біля Бомонта, штат Техас, на кургані під назвою Веретена стався один з найбільших і значущих нафтових ударів в історії. Цей страйк припинив будь-яку можливу монополію Standard Oil. Через рік після відкриття Spindletop було створено понад півтори сотні нафтових компаній. З них вижило менше десятка, в основному Gulf Oil Corporation, Magnolia

Petroleum Company і Texas Company. Sun Oil Company, концерн Огайо-Індіана, також переїхав до району Бомонт, як і інші фірми. Інші нафтові страйки відбулися в Оклахомі, Луїзіані, Арканзасі, Колорадо та Канзасі. Видобуток нафти в Сполучених Штатах до 1909 року більше ніж дорівнював видобутку решти світу разом узятих.

Багато менших компаній розвинулися за межами Північного Сходу та Середнього Заходу, де працювали Рокфеллер та його соратники. Нафта, знайдена на Корсикані, штат Техас, у 1890-х роках привернула чудового пенсільванця Джозефа С. (“Buckskin Joe”) Куллінана, який організував кілька невеликих компаній. Пізніше він переїхав до Spindletop, де він став значущою в організації Texas Company, яка незабаром стала головним конкурентом Standard. Анрі Детердінг, творець Royal Dutch-Shell Group в Голландії та Великобританії, переїхав до Каліфорнії в 1912 році зі своєю American Gasoline Company (Shell Company of California після 1914 року).

Оскільки «Стандард Ойл» зростав у багатстві та владі, він зіткнувся з великою ворожістю не лише з боку своїх конкурентів, а й з боку значної частини громадськості. «Стандарт» боровся з конкуренцією, забезпечуючи пільгові тарифи на залізничні перевезення та знижки на свої відправлення. Це також вплинуло на законодавчі органи та Конгрес за допомогою тактики, яка, хоча й була поширена в ту епоху, була неетичною. Також не було кращим поводження компанії з робочою силою.

У 1911 році Верховний суд оголосив, що Standard Trust діяв з метою монополізації та стримування торгівлі, і наказав розпустити трест на тридцять чотири компанії. Суд вважав, що частка трасту в галузі скоротилася з 33 до 13 відсотків. Відокремлення філій Standard виявилося складним. Деякі з них продавали, інші виробляли, деякі доопрацьовували, і ці проблеми швидко перейшли до вертикальної інтеграції їх бізнесу. Але рішення 1911 року гарантувало, що хоча в галузі можуть бути гіганти, вони принаймні конкурують один з одним.

Збільшення продажів бензину спочатку для автомобілів, а потім і для літаків на початку 1900-х відбулося в міру того, як у Сполучених Штатах зростали відкриття нафти. У нафтової промисловості з'явився величезний новий ринок для того, що багато років було марним побічним продуктом процесу перегонки. Як тільки двигуни внутрішнього згоряння створили попит, нафтопереробники шукали кращі методи виробництва та вдосконалення бензинів.

До початку Першої світової війни Сполучені Штати постачали нафту союзникам, а в 1917 році нафтові компанії співпрацювали з Управлінням палива. Наприкінці війни керівники, які працювали в цьому агентстві, створили Американський інститут нафти (1919), який з часом став головною силою в економіці та бізнесі.

Незважаючи на те, що нафтова промисловість США до війни активно здійснювала продажі за кордоном, їй належало небагато іноземної нерухомості. Судячи з урядових опитувань, багато виробників вважали, що незабаром виникне серйозний дефіцит нафти. І міністр торгівлі Герберт Гувер, і держсекретар Чарльз Еванс Хьюз почали тиснути на американські компанії, щоб вони шукали нафту за кордоном. Ці фірми інвестували на Близькому Сході, Південно-Східній Азії та Південній Америці та шукали нафту скрізь, продовжуючи експортувати нафту зі Сполучених Штатів.

Особою, яка звернула увагу на Сполучені Штати, був Колумбус Меріон («тато») Столяр. Столяр переконався, що деякі рівнини в структурі басейну Східного Техасу містять нафту. Він отримав оренду поблизу Тайлера, штат Техас, і 5 жовтня 1930 року, пробуривши дві сухі свердловини, вибив, можливо, найбільший нафтовий басейн, коли-небудь знайдений в Америці. Він лежав під 140 000 акрів і містив 5 мільярдів барелів. Х. Л. Хант, нафтовий підприємець, купив оренду у Столяра, а потім продав їх нафтовим компаніям з прибутком у 100 мільйонів доларів, тим самим додавши йому і без того значні статки.

У певному сенсі страйк Столярів відбувся в невідповідний час; це був початок Великої депресії. Ціна на нафту впала до десяти центів за барель у 1931 році, створивши хаос у промисловості. Але деякі заходи Нового курсу відновили трохи процвітання, а потім Друга світова війна надзвичайно стимулювала нафтовий бізнес[2].

Різні нафтові страйки зосередили увагу на правовій ситуації, унікальній для Сполучених Штатів. Право власності на землю принесло з собою права на всі корисні копалини, які в загальному праві називають «правом вилучення». Нафтові компанії, як і інші мінеральні компанії, вели переговори з кожним землевласником про права на буріння. Це право на захоплення продовжувалося роками, незважаючи на зусилля таких гігантів галузі, як Генрі Л. Доерті з нафтової компанії Cities Service, який прагнув запровадити об'єднання нафтових родовищ. Право захоплення забезпечувало дострокове вичерпання нафтових родовищ і трагічну розтрату цінного джерела енергії. Уоллес Е. Пратт, геолог і багаторічний лідер Jersey Standard, підрахував, що, вивільняючи природний газ, який часто лежить в основі нафтових басейнів, і використовуючи погані технології виробництва, виробники нафти витратили щонайменше 75 відсотків нафти та природного газу, знайдених на сьогодні. у Сполучених Штатах[31].

Друга світова війна зробила нафтову промисловість ключовим американським ресурсом. Головну роль у конфлікті відіграли дослідження нафтової компанії та керівне керівництво. Дослідження збільшили кількість продуктів, виготовлених з нафти та природного газу, у тому числі вибухонебезпечного троту та штучного каучуку. Продукт спільної власності Джерсі-Дюпон, тетраетилсвинець, модернізував бензин для підвищення швидкості літака. Нафтові танкери постачали бензин для союзників, які відчували великий ризик атак підводних човнів. Під час війни уряд нормував бензин і контролював ціни. Зрештою, війна покінчила з помилкою, що американські поставки нафти були необмеженими, тому промисловість і

забезпечення нафти стали головним пріоритетом як зовнішньої, так і внутрішньої політики.

Коли війна закінчилася, Сполучені Штати зіткнулися з проблемою стабілізації миру. Протягом наступних сорока п'яти років відбулися численні серйозні кризи, у багатьох з яких нафта відігравала ключову роль. Відразу після війни Європа зазнала дефіциту вугілля, першої енергетичної кризи. План Маршалла, створений для вирішення цієї та інших проблем, завадила перша іранська криза 1950-1954 років. Від Суецької кризи 1956 року до іракського вторгнення в Кувейт у 1990 році нафта виявилася найважливішим фактором близькосхідної політики Америки. Сполучені Штати намагалися збалансувати підтримку нової держави Ізраїль з тиском виробників нафти, переважно арабських, об’єднаних у 1960 році як Організацію країн-експортерів нафти (ОПЕК). Це виявилося дедалі складнішим, оскільки Сполучені Штати стали все більш залежними від імпортованої нафти. У Сполучених Штатах рівень життя, заснований на дешевій нафті, постійно зростав, і населення, яке звикло до такого способу життя, чинило опір усім заходам щодо збереження. Сполучені Штати продовжують споживати близько двох третин світового видобутку нафти. Нафту слід вважати ключовим фактором рівня життя в Сполучених Штатах і значною мірою її рейтингом як світової держави[5].

Частина енергетичної проблеми після 1940 року була результатом виснаження внутрішніх запасів нафти під час Другої світової війни — близько 6 мільярдів барелів. Експерти стверджують, що в боротьбі з В’єтнамом Сполучені Штати постачали близько 5 мільярдів барелів нафти, хоча велика кількість її надходила з близькосхідної власності, що належить американським компаніям. Після 1960-х років, коли внутрішнє виробництво скоротилося, а попит зріс, нафтова промисловість змушена була імпортувати величезну кількість з Близького Сходу та Венесуели. Ключове джерело енергії країни все більше залежало від збалансування дипломатичних відносин з арабськими нафтовидобувними країнами.

У той час як Сполучені Штати були наділені багатими запасами нафти, їх зростання до рангу великої держави прискорилося. У сучасному світі, як держава, що залежить від нафти, вона повинна шукати альтернативні джерела енергії або влаштовувати кардинальні зміни у своєму способі життя та позиції у світі.

1.2 Вплив ОПЕК на ринок нафти

Поява сланцевої нафти в США суттєво змінила структуру світового видобутку нафти. Поява США як домінуючого гравця на ринку мала глибокі наслідки для стратегічної поведінка інших виробників нафти, зокрема тих, хто об'єднався в Організацію країн-експортерів нафти (ОПЕК).

Як картель, країни-члени ОПЕК+ колективно домовляються про те, скільки нафти виробляти, що безпосередньо впливає на готові поставки сирої нафти на світовому ринку в будь-який момент часу. Згодом ОПЕК+ справляє значний вплив на світову ринкову ціну на нафту і, зрозуміло, прагне утримувати її відносно високою, щоб максимізувати прибутковість.

Якщо країни ОПЕК+ незадоволені ціною на нафту, то в їхніх інтересах скоротити пропозицію нафти, щоб ціни зростали. Однак жодна окрема країна насправді не хоче скорочувати пропозицію, оскільки це означало б зменшення доходів. В ідеалі вони хочуть, щоб ціна на нафту зростала, одночасно збільшуючи пропозицію, щоб зростали і доходи. Але це не ринкова динаміка. Зобов’язання ОПЕК+ скоротити постачання спричиняє негайний стрибок ціни на нафту. З часом ціна повертається до рівня, зазвичай нижчого, коли пропозиція істотно не скорочується або попит коригується.

І навпаки, ОПЕК+ може вирішити збільшити пропозицію. Наприклад, 22 червня 2018 року картель зібрався у Відні і оголосив, що вони будуть

збільшувати пропозицію.3 Великою причиною для цього було компенсувати надзвичайно низький видобуток у Венесуели, який є членом ОПЕК+.

ОПЕК, до складу якої входять 14 членів у тому числі великі виробники, такі як Саудівська Аравія, Ірак, Іран та Об'єднані Арабські Емірати, становили близько 43 відсотків світового видобутку нафти у 2008 році. У 2019 році її частка впала до 39, оскільки США збільшили свою частку на нафтовому ринку з 8 до 15 відсотків. Не тільки США зараз є найбільшими виробники нафти, вони також стали експортерами нафти після заборони, яка заважала виробникам продавати нафту за кордон (було знято у грудні 2015 р.).

У 2016 р., намагаючись відновити певний контроль над ціни на нафту, ОПЕК та ряд інших неамериканських виробників нафти уклали альянс, відомий як ОПЕК+. Серед не членів ОПЕК ця коаліція включала великих виробників, як Росія (другий за величиною виробник нафти), Мексика та Казахстан. Загалом на долю ОПЕК+ припадає близько 45 відсотків світового видобутку нафти. Стратегія, прийнята ОПЕК+, полягала в встановлення чітких виробничих цілей для кожного члена з метою забезпечення запасів нафти до середнього за 2010-2014 роки.

Саудівська Аравія та Росія, два з найбільших експортерів нафти в світі, які мають можливість збільшити видобуток, є великими прихильниками збільшення пропозиції, оскільки це збільшило б їхні доходи. Проте інші країни, які не можуть нарощувати виробництво через те, що вони працюють на повну потужність, або їм це заборонено, будуть проти цього[31].

Зрештою, сили попиту та пропозиції визначають цінову рівновагу, хоча оголошення ОПЕК+ можуть тимчасово вплинути на ціну нафти, змінивши очікування. Показовим випадком, коли очікування ОПЕК+ будуть змінені, є коли її частка у світовому видобутку нафти зменшується, а нове видобуток надходить із інших країн, таких як США та Канада.

У березні 2020 року коаліція ОПЕК+ розпалася. Після загального ризику, пов’язаного з поширенням коронавірусу, ціни на нафту обвалився.

Коли ОПЕК запропонувала здійснити подальше скорочення видобутку, щоб підняти ціни на нафту, Росія відмовилася співпрацювати, аргументуючи це тим, що американські виробники виграють найбільше від нових зусиль підтримати ціни. Зіткнення між Саудівською Аравією і Росією означало закінчення коаліції ОПЕК+[5].

1.3 Формування ціни на нафту

Нафта все ще відіграє важливу роль у світовій економіці, незважаючи на постійні зусилля щодо скорочення її використання та пошуку альтернативних джерел зеленої енергії. На перших порах пошук нафти вважався певною неприємністю, оскільки передбачуваними скарбами зазвичай були вода або сіль. Лише в 1847 році на Апшеронському півострові в Азербайджані була пробурена перша комерційна нафтова свердловина. Нафтова промисловість США зародилася 12 років потому, у 1859 році, з навмисним бурінням поблизу Тітусвілля, штат Пенсільванія. (Буріння в Сполучених Штатах почалося на початку 1800-х років, але вони бурили для розсолу, тому будь-яке відкриття нафти було випадковим.)

Хоча більша частина раннього попиту на нафту була на гас і масляні лампи, лише в 1901 році перша комерційна свердловина, здатна до масового видобутку, була пробурена на місці, відомому як Spindletop в південно-східному Техасі. На цьому місці було видобуто понад 100 000 барелів нафти за один день, що більше, ніж усі інші нафтові свердловини в Сполучених Штатах разом узяті. Багато хто стверджує, що сучасна нафтова ера зародилася того дня в 1901 році, оскільки нафта незабаром мала замінити вугілля як основне джерело палива у світі.

Використання нафти в якості палива продовжує залишатися основним фактором, що робить її товаром високого попиту в усьому світі, але як визначаються ціни?

Оскільки нафта є світовим товаром із високим попитом, існує ймовірність того, що великі коливання цін можуть мати значний економічний вплив. На ціну нафти впливають два основні фактори:

- пропозиція і попит;
- ринкові настрої.

Концепція попиту та пропозиції досить проста. У міру збільшення попиту (або зменшення пропозиції) ціна повинна зростати. У міру зменшення попиту (або збільшення пропозиції) ціна повинна знизитися. Звучить просто?

Іншим ключовим фактором у визначенні цін на нафту є настрої. Одне лише переконання в тому, що попит на нафту різко зросте в якийсь момент у майбутньому, може призвести до різкого зростання цін на нафту в даний час, оскільки спекулянти та хеджери скуповують ф'ючерсні контракти на нафту. Звісно, вірно й навпаки. Віра в те, що попит на нафту знизиться в якийсь момент у майбутньому, може призвести до різкого зниження цін у сьогоденні, оскільки ф'ючерсні контракти на нафту будуть продані (можливо, також продані в короткі терміни), що означає, що ціни можуть залежати лише від ринкової психології.

Основна теорія попиту та пропозиції стверджує, що чим більше виробляється продукт, тим дешевше він повинен продаватись за всіх рівних умов. Це симбіотичний танець. Причина, чому вироблялося більше товару, в першу чергу, полягає в тому, що це стало економічно ефективнішим (або не менш економічно ефективним). Якби хтось винайшов техніку стимулювання свердловини, яка могла б подвоїти видобуток нафтового родовища лише за невелику додаткову вартість, тоді, якщо попит залишиться статичним, ціни повинні впасти.

Насправді, були періоди часу, коли пропозиція збільшувалася. У 2019 році видобуток нафти в Північній Америці був на найвищому рівні, а родовища в Північній Дакоті та Альберті були такими плідними, як ніколи.

Тут теорія протистоїть практикі. Виробництво було високим, але розподіл і доопрацювання не встигали за ним. Сполучені Штати будують в середньому один нафтопереробний завод за десятиліття (будівництво сповільнилося з 1970-х років).

Запитайте у нафтопереробника, і вони скажуть вам, що надлишкові потужності існують для задоволення майбутнього попиту. Крім того, з історичної точки зору, здається, можливий 29-річний (плюс-мінус один-два роки) цикл, який регулює поведінку цін на сировину в цілому. З початку зростання нафти як товару з високим попитом на початку 1900-х років основні піки індексу сировини припали на 1920, 1958 та 1980 роки. Пік нафти прийшовся разом із індексом товарів у 1920 та 1980 роках. Важливо зазначити, що пропозиція, попит і настрої мають перевагу над циклами, оскільки цикли є лише орієнтирами, а не правилами.

Тоді виникає проблема картелів. Ймовірно, найбільшим впливом на ціни на нафту є ОПЕК. Хоча статут організації прямо не говорить про це, ОПЕК була заснована в 1960-х роках, щоб — грубо кажучи — встановлювати ціни на нафту і газ. Обмежуючи видобуток, ОПЕК могла б змусити ціни зростати і, таким чином, теоретично отримувати більший прибуток, ніж якби кожна з її країн-членів продавала на світовому ринку за постійним курсом. Протягом 1970-х і більшої частини 1980-х років він дотримувався цієї, хоча і дещо неетичної, стратегії.

Як цитує П. Дж. О'Рурка, «деякі люди вступають у картелі через жадібність, а потім через жадібність намагаються вийти з картелів». За даними Управління енергетичної інформації США, країни-члени ОПЕК часто перевищують свої квоти, продаючи кілька мільйонів додаткових барелів, знаючи, що правоохоронці не можуть зупинити їх у цьому. Оскільки

Канада, Китай, Росія та Сполучені Штати не є членами — і збільшують власний видобуток — ОПЕК стає обмеженою у своїх можливостях.

Хоча консорціум пообіцяв утримувати ціну на нафту вище 100 доларів за барель у доступному для огляду майбутньому, у середині 2014 року він відмовився скорочувати видобуток нафти, навіть коли ціни почали падати. В результаті вартість нафти впала з пікових від 100 доларів за барель до нижче ніж 50 доларів за барель. Станом на січень 2021 року ціни на нафту коливаються вище 52 доларів[34].

На відміну від більшості продуктів, ціни на нафту не визначаються виключно пропозицією, попитом і настроями ринку щодо фізичного продукту. Швидше, пропозиція, попит і настрої щодо ф'ючерсних контрактів на нафту, якими активно торгують спекулянти, відіграють домінуючу роль у визначенні ціни. Також певну роль можуть зіграти циклічні тенденції на ринку товарів. Незалежно від того, як остаточно визначається ціна, виходячи з її використання в паливі та незліченних споживчих товарах, схоже, що нафта й надалі матиме високий попит у доступному для огляду майбутньому.

Існує декілька основних факторів, що сприяють ціноутворенню на ринку нафти[6], а саме:

- попит та пропозиція;
- комерційні запаси нафти;
- кількість бурових вишок;
- непередбачувані події.

Нижче, на рис. 1.1 відображено ціни на нафту за останні 40 років.

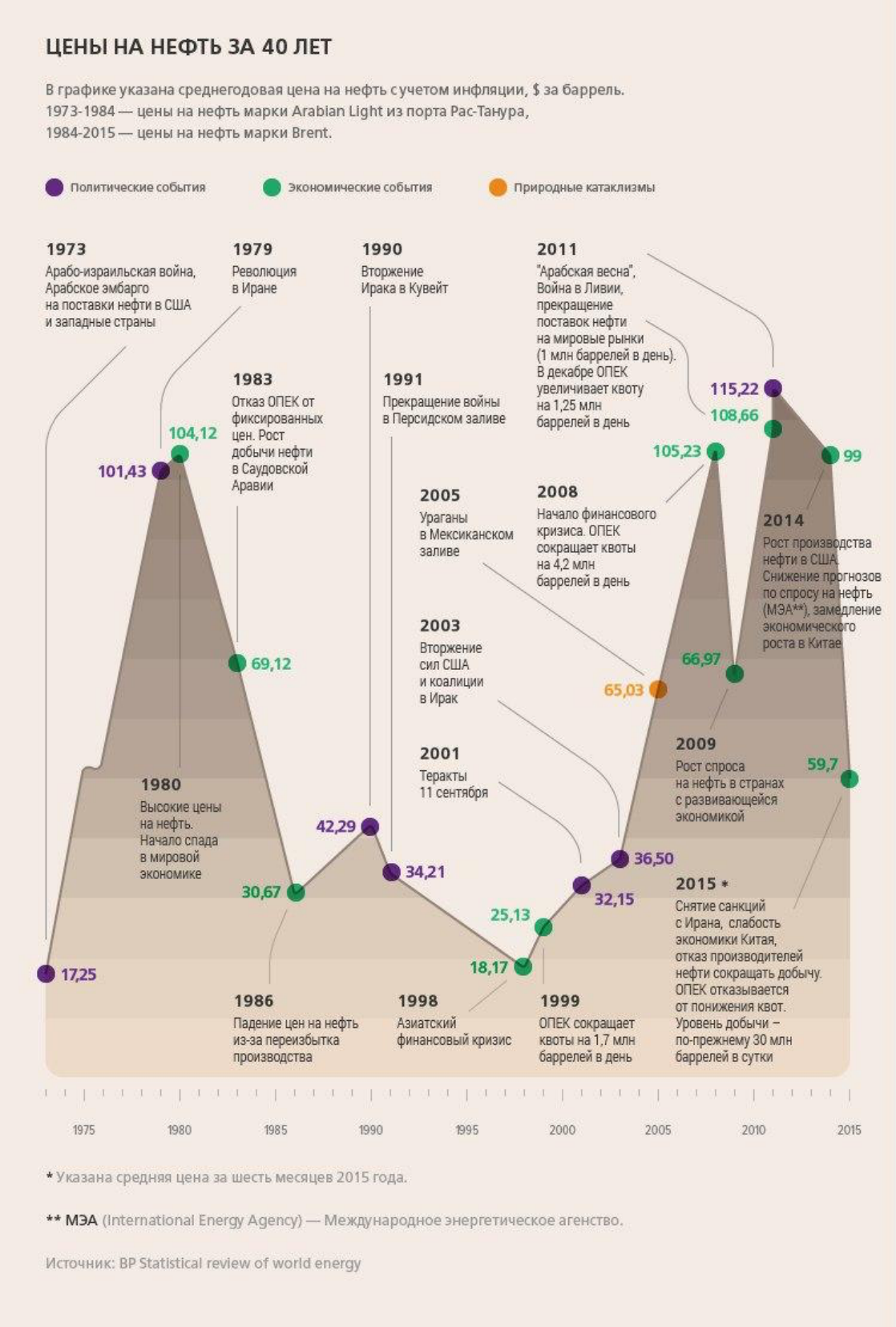

Рисунок 1.1 – Ціни на нафту за 40 років [2]

На спотовому ринку нафта купується і продається за готівку і негайно доставляється. На додаток до всіх фактичних барелів нафти, якими торгують, існує другий ринок – ф’ючерсний ринок, який торгує «паперовими»

барелями. Це просто вказує на те, що нафта торгується на «папері» на основі передбачуваної грошової вартості нафти і зазвичай не відбувається фізичного обміну продуктом. Паперові контракти на нафту купують і продають на основі очікуваних ринкових умов у найближчі місяці або навіть роки. На ф'ючерсному ринку є два типи покупців і продавців, що будуть описані далі.

Фактичні виробники або користувачі сирої нафти (наприклад, нафтопереробні заводи) і ті, хто купує ф'ючерсні контракти як інвестиції, не маючи наміру коли-небудь заволодіти фактичною сирою нафтою. Перша група використовує ф'ючерсний ринок, щоб захистити себе від волатильності цін, зафіксувавши дохід для покриття витрат і гарантування прибутку від майбутнього видобутку нафти. Нафтопереробні заводи купують ф'ючерсні контракти, щоб зафіксувати витрати на закупівлю нафти.

Друга група (інвестори, які не виробляють і не споживають нафту) купують ф’ючерсні контракти на нафту з інвестиційними цілями і можуть заробляти гроші, правильно вгадуючи, зростуть чи зменшаться ціни в майбутньому.

ЗМІ найчастіше цитують ринкову ціну ф’ючерсного ринку в найближчий місяць як репрезентацію поточної ціни на нафту. Поточна спотова ціна на нафту залежить від ринкової ціни ф'ючерсу, оскільки ф'ючерсна ціна представляє сукупний погляд ринку на певний момент часу щодо того, куди можуть рухатися ціни.

Показник комерційних запасів нафти відображає, кількість реальних запасів нафти по відношенню до прогнозованої заздалегідь кількості. Щосереди оприлюднюється офіційна статистика від AAEI. Збільшення запасів щодо прогнозних даних означає, що або надлишкова кількість нафти вироблялася, або споживання було слабкішим за очікуване(рис. 1.2).

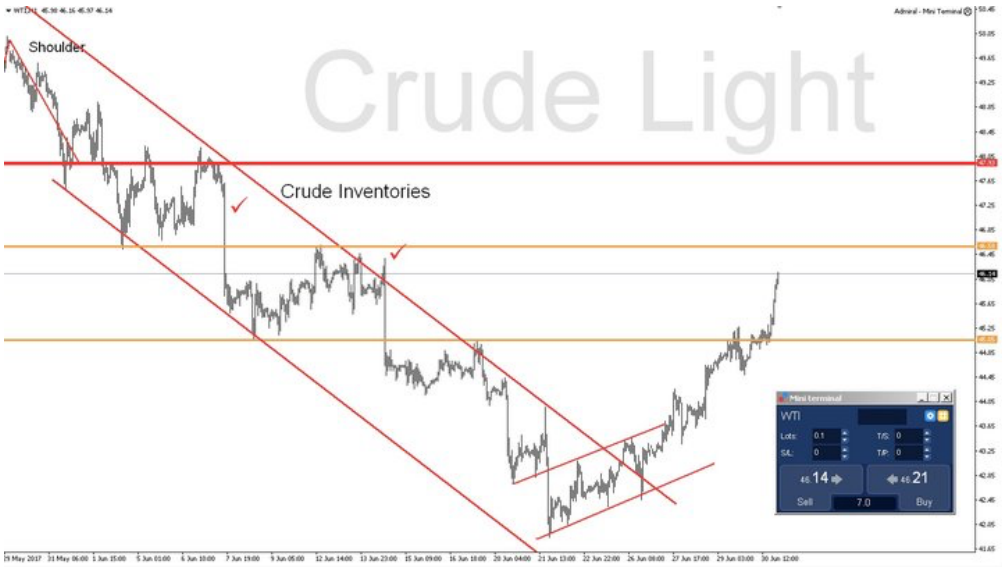

Рисунок 1.2 - Статистика комерційних запасів нафти [5]

Кількість бурових вишок показана на рис. 1.3.

Rotary Rig Count
6/30/2017

BAKER HUGHES

Location	Week	+/-	Week Ago	+/-	Year Ago
Land	915	0	915	507	408
Inland Waters	4	0	4	0	4
Offshore	21	-1	22	2	19
United States Total	940	-1	941	509	431
Gulf Of Mexico	21	0	21	3	18
Canada	189	19	170	113	76
North America	1129	18	1111	622	507
U.S. Breakout Information	**This Week**	**+/-**	**Last Week**	**+/-**	**Year Ago**
Oil	756	-2	758	415	341
Gas	184	1	183	95	89
Miscellaneous	0	0	0	-1	1
Directional	71	-1	72	33	38
Horizontal	792	0	792	460	332
Vertical	77	0	77	16	61
Canada Breakout Information	**This Week**	**+/-**	**Last Week**	**+/-**	**Year Ago**
Oil	112	14	98	77	35
Gas	77	5	72	37	40
Miscellaneous	0	0	0	-1	1

Рисунок 1.3 - Кількість бурових вишок [4]

Природні і техногенні катастрофи можуть вплинути на ціни на нафту, якщо вони матимуть глобальні наслідки.

COVID-19. У січні 2020 року багато уряди почали обмежувати пересування по країні і закривати підприємства, щоб зупинити пандемію коронавіруса. Попит на нафту почав падати. У першому кварталі 2020 року

споживання нафти в середньому становило 94,4 млн барелів на добу, що на 5,6 млн барелів на добу більше, ніж у попередньому році. Падіння попиту ускладнилося через надлишок пропозиції. 6 березня Росія оголосила, що збільшить видобуток в квітні. Щоб зберегти частку ринку, ОПЕК оголосила, що також збільшить видобуток. Коли складські приміщення стали заповнені, ціни впали в мінус. Торговці були готові платити гроші за доставку нафти, так як місця для її зберігання просто не було. 12 квітня 2020 року ОПЕК і Росія домовилися про зниження видобутку, щоб підтримати ціни. Цього було недостатньо для переконання трейдерів, що пропозиція не перевищить попит. На рис. 1.4 відображена від'ємна ціна на нафту. Станом на 20 квітня 2020 року ціна за барель нафти впала до -$ 36,98.

Рисунок 1.4 - Графік цін на нафту [8]

«Катріна» - ураган 5 категорії, який обрушився на Луїзіану 29 серпня 2005 року. У період з 29 серпня по 5 вересня середня ціна на звичайний бензин в США зросла на 46 центів до 3,07 долара за галон. Це був

найбільший щотижневий стрибок цін за всю історію. Ураган «Катріна» торкнувся 25% видобутку сирої нафти в США. Це відбулося слідом за ураганом «Рита». Сукупний вплив двох ураганів скоротив надходження сирої нафти на нафтопереробні заводи до 11,7 млн барелів на день протягом тижня. Це був найнижчий середній показник з березня 1987 року[8]. Ціни на нафту у 2005 році відображені на рис. 1.5.

Рисунок 1.5 - Графік цін на нафту 2005 рік [8]

1.4 Модель для прогнозування ціни на нафту

На сьогодні існує незліченна кількість методів, що дозволять провести аналіз ціни на нафту та спробувати спрогнозувати її. Існують всім відомі моделі економетрії, підходи з використанням дерев рішень, методи машинного навчання та нейронні мережі. Підходи з використанням машинного навчання та нейронних мереж мають значну перевагу через свою точність та об’єктивність результатів, але в той же час, економічне інтерпретація цих результатів є досить важким процесом. З кожним днем можливості машинного навчання ростуть, з’являються нові методи з

використанням дерев рішень, моделі дерев рішень інтерпретується в теорії (Loh, 2014 року). Метою цього дослідження є побудова коректних моделей та порівняння роботи та коректності моделей машинного навчання та дерев рішень у задачі прогнозування ціни на нафту[9].

1.5 Висновки

Така задача як побудова моделей для прогнозування цін на нафту є досить цікавою, але в той же час важкою, бо ринок нафти є досить волотильним. В ході роботи над цією магістерською дисертацією я ознайомився з безліччю літератури, що допомогла мені набути теоретичних та практичних відомостей, а саме: історична література, економічна та технічна. Вивчивши усі можливі відомості про ринок нафти, як він працює та як проходять торгівельні процеси на цьому ринку, я прийняв рішення про використання сучасних методів регресійного аналізу та нейронних мереж.

РОЗДІЛ 2 РЕГРЕСІЙНИЙ АНАЛІЗ. НЕЙРОННІ МЕРЕЖІ. МЕТОДИ ТА МЕТОДОЛОГІЇ ПРОГНОЗУВАННЯ НА ОСНОВІ РЕКУРЕНТНИХ НЕЙРОННИХ МЕРЕЖ. ТЕОРІЯ ТА ВІДОМОСТІ

2.1 Вступ

У статистичному моделюванні регресійний аналіз являє собою набір статистичних процесів для оцінки взаємозв'язків між залежною змінною (часто позначається «вихідної змінної») і однієї або декількома незалежними змінними (часто позначаються «предикторами», «коваріатамі» або «ознаками»).

Регресія – одностороння стохастична залежність, що встановлює відповідність між випадковими змінними. На відміну від суто функціональної залежності $y = f(x)$, коли кожне значення незалежної змінної x відповідає одному визначеному значенню y, у випадку регресії одне і те ж значення x може відповідати різним значенням y[10]. Приклад показано на рисунку 2.1.

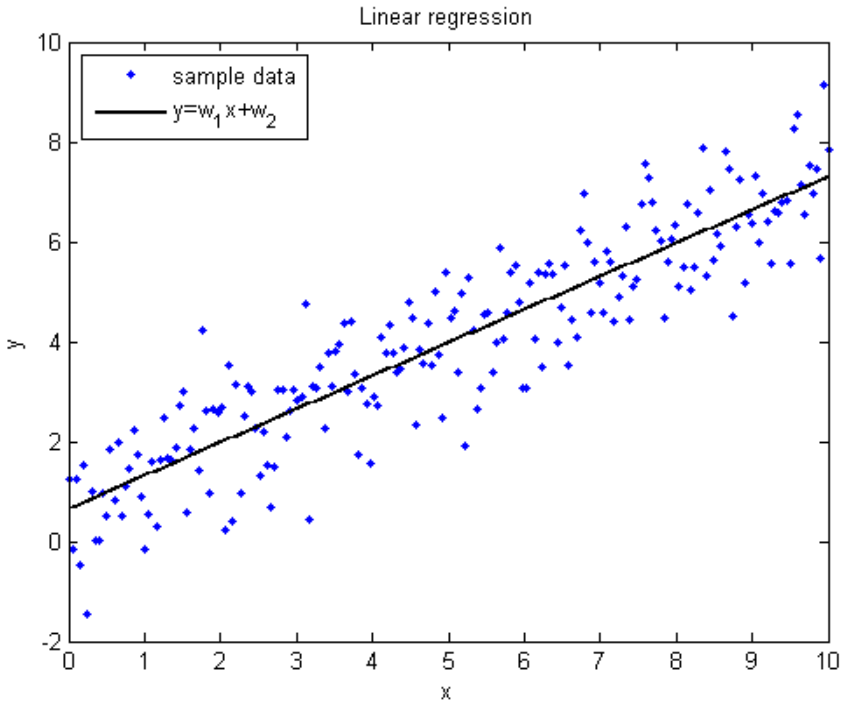

Рисунок 2.1 - Лінійна регресія

Штучні нейронні мережі (ШНМ), які зазвичай називають просто нейронними мережами (НН), є обчислювальними системами, натхненними біологічними нейронними мережами, які утворюють мозок тварин.

ШНМ заснована на сукупності пов’язаних одиниць або вузлів, які називаються штучними нейронами, які вільно моделюють нейрони в біологічному мозку. Кожне з’єднання, як і синапси в біологічному мозку, може передавати сигнал іншим нейронам. Штучний нейрон отримує сигнал, потім обробляє його і може сигналізувати підключеним до нього нейронам. «Сигнал» при з’єднанні є дійсним числом, а вихід кожного нейрона обчислюється за допомогою деякої нелінійної функції суми його вхідних даних. З’єднання називаються ребрами. Нейрони та ребра зазвичай мають вагу, який коригується в міру навчання. Вага збільшує або зменшує силу сигналу при з'єднанні. Нейрони можуть мати такий поріг, що сигнал надсилається лише в тому випадку, якщо сукупний сигнал перетинає цей поріг. Як правило, нейрони об’єднуються в шари. Різні шари можуть виконувати різні перетворення на своїх входах. Сигнали поширюються від першого шару (вхідного шару) до останнього шару (вихідного шару).

За винятком рекурентних нейронних мереж (РНМ), більшість нейронних мереж (НМ) не мають пам’яті для обробки окремо відображених вхідних даних, і між ними не підтримується стан (Chollet and Allaire, 2018). Тришарова нейронна мережа (НМ) складається з вхідного шару, прихованого шару і, нарешті, вихідного шару. РНМ є потужними та надійними типами нейронних мереж. Вони мають внутрішню пам’ять, яка може запам’ятати важливу інформацію про отримані вхідні дані, щоб адекватно прогнозувати, що станеться далі. РНМ мають структуру, подібну до інших типів нейронних мереж з модифікаціями, щоб врахувати приховані шари попередніх кроків часу на поточному кроці часу. Двома варіантами РНМ є довготривала короткочасна пам'ять (LSTM) і рекурентний блок із замкнутою системою (GRU). Дані в цих мережах (тобто РНМ та його варіанти) надходять у будь-

якому напрямку всередині груп і між групами. Одна з основних проблем із простим РНМ полягає в тому, що він повинен мати можливість зберігати інформацію про вхідні дані, які спостерігалися за багато кроків часу раніше, але проблема зникаючого градієнта ускладнює вивчення таких довгострокових залежностей. Шар LSTM, розроблений Hochreiter і Schmidhuber (1997), і шари GRU, розроблені Chung et al. (2015) у 2014 р. були покликані вирішити цю проблему. Модель LSTM і модель GRU широко використовуються, оскільки обидва вони кращі за інші методи і можуть розумно викликати періоди пікового потоку (Apaydin et al., 2020).

У минулому для прогнозування цін на нафту застосовувалося багато методів, але жоден з них не реалізував гібридний підхід, порівнюючи алгоритм AdaBoost і мережу довготривалої короткострокової пам’яті (LSTM), а також алгоритм AdaBoost і мережу GRU (Gated Recurrent Unit). Ми прагнемо прогнозувати ціну на нафту, порівнюючи AdaBoost-LSTM з AdaBoost-GRU для підвищення точності прогнозування. Sun et al. (2018) раніше спрогнозував фінансові часові ряди, що складалися з двох типових фондових індексів і двох основних валютних курсів, і показав, що підхід ансамблевого навчання AdaBoost-LSTM перевершує інші моделі єдиного прогнозування та підходи ансамблевого навчання. Тому одна з можливих актуальних задач – це порівняти моделі AdaBoost-LSTM та AdaBoost-GRU, та перевірити їхню продуктивність з результатами двох варіантів РНМ: LSTM та GRU.

2.2 Дерева Рішень «Decision Trees»

Дерева рішень («Decision Trees») - це статистичний метод, що дозволяє передбачати приналежність спостережень або об'єктів до того чи іншого класу категоріальної залежною змінною або середнє значення кількісної

змінної в залежності від відповідних значень однієї або декількох незалежних змінних.

Дерево має багато аналогій в реальному житті і виявляється, що воно вплинуло на широку область машинного навчання, що охоплює як класифікацію, так і регресію. При аналізі рішень дерево рішень може використовуватися для візуального і явного уявлення рішень і їх прийняття. Як випливає з назви, метод використовує деревоподібну модель рішень. Незважаючи на те, що цей інструмент широко використовується для дослідження стратегії для досягнення певної мети, він також широко використовується в машинному навчанні.

Мета методу дерев рішень – спрогнозувати значення цільової змінної в залежності від відповідних значень незалежних змінних (предикторів, атрибутів). В свою чергу, дерева рішень поділяються на дерева регресії і дерева класифікації.

Якщо мова йде про дерева регресії, прогнозується значення цільової змінної, що залежить від відповідних значень предикторів.

Наприклад, прогнозується ймовірність регістрації на певному сайті в залежності від статі і віку відвідувача. Дерева регресії працюють з кількісною цільовою змінною.

В деревах класифікації все виглядає інакше. Робиться прогноз приналежність об'єкта до тієї чи іншої категорії цільової змінної в залежності від відповідних значень предикторів.

Наприклад, класифікуються добрі і погані учні в залежності від їх оцінок. Дерева класифікації працюють з категоріальної цільової змінної.

Залежність значення цільової змінної від значень предикторів, представляється у вигляді ієрархічної структури - «дерева». Якщо залежна змінна є категоріальної, будується дерево класифікації. Якщо залежна змінна є кількісної, будується дерево регресії.

Також дерева рішень допомагають вирішувати завдання опису об’єктів. Тобто за допомогою дерева, можна замінити велику структуру на більш зручну, маленьку.

2.2.1 Основні терміни та поняття

Для розуміння теорії дерев рішень треба розібратися з основними термінами[19]:

- об’єкт – приклад, шаблон, дослідження;
- атрибут – ознака, незалежна змінна, властивість;
- цільова змінна – залежна змінна, мітка класу;
- вузол – внутрішній вузол дерева, вузол перевірки;
- кореневий вузол – початковий вузол дерева рішень;
- лист – кінцевий вузол дерева, вузол рішення;
- вирішальне правило – умова у вузлі, перевірка;
- ентропія – міра хаотичності інформації.

2.2.2 Представлення алгоритму у вигляді дерева

У прикладі дерева рішень для класифікації, яке зображено на рис. 2.2, використовуються 3 атрибута з набору даних, такі як стать людини, її вік і кількість дітей. У дерева є внутрішній вузол, позначений жирним текстом чорного кольору, який представляє умову, на основі якої це дерево розділяється на гілки або ребра. Листами дерева слугують рішення, тобто задані класи в задачі класифікації. В даному прикладі класами є загинув

пасажир (клас 1) або вижив (клас 2). Класи представлені на рис.2.2 червоним та зеленим кольором, відповідно.

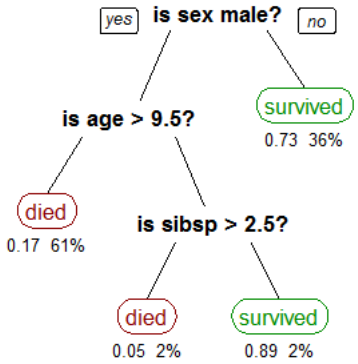

Рисунок 2.2 - Дерево рішень [25]

Аналогічно представлені дерева регресії (рис. 2.3). Відмінність від дерев класифікації полягає у тому, що на їх основі прогнозуються неперервні значення, наприклад ціна на нерухомість.

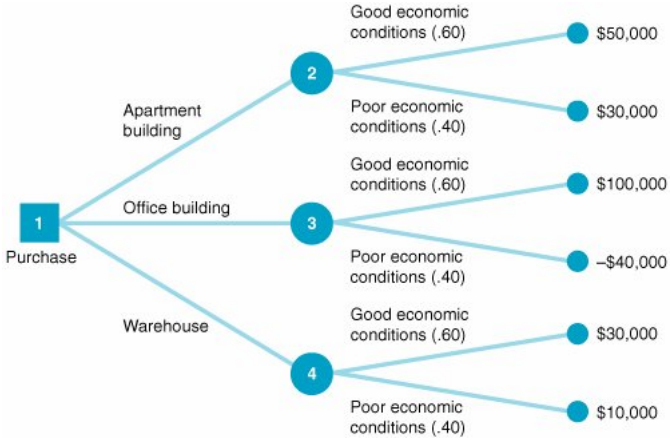

Рисунок 2.3 - Дерево регресії [25]

Розглянемо суть алгоритму розбиття «CART» дерев рішень для класифікації та регресії[20]. В цьому алгоритмі у випадку задачі регресії мінімізується наступна функція вартості:

$$J(T, x_h) = \frac{|T_{left}|}{|T|} MSE(T_{left}) + \frac{|T_{right}|}{|T|} MSE(T_{right}) \quad (2.1)$$

$$MSE(T_{left}) = \sum_{k \in T_{left}} \left(\hat{y}_{T_{left}} - y_k \right)^2 \quad (2.2)$$

$$\hat{y}_{T_{left}} = \frac{1}{|T_{left}|}\sum_{k \in T_{left}} y_k \quad (2.3)$$

$$X_h^* = argmin_h J(T, x_h), \quad (2.4)$$

де $X_h^* -$ корінь дерева(піддерева), $MSE -$ середньоквадратична похибка.

Алгоритм «CART» виконує рекурсивне розбиття множини на підмножини, що відповідають мінімально можливим значенням MSE, з ваговими коефіцієнтами, що позначають відносні розміри цих підмножин[25].

MSE(mean squared error) – середньоквадратична помилка моделі.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (2.5)$$

Коефіцієнт детермінації — це міра залежності варіацій залежної змінної від варіацій незалежних змінних. Він відображає наскільки спостереження побудованої моделі підтверджують реальні спостереження.

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2} \quad (2.6)$$

Алгоритм розбиття є жадібним. Це означає, що на кожному кроці він робить локально оптимальний вибір, допускаючи, що результат буде глобально оптимальним. Дерево рішень, побудоване алгоритмом розбиття, зазвичай розростається довільно великим. Тому потрібні спеціальні методи редукції цих дерев.

2.2.3 Метод побудови дерева рішень. Алгоритм розбиття

Даний підрозділ присвячено огляду загального підходу до побудови дерева рішень з використанням алгоритму розбиття.

Ідея:

- побудова дерева зверху-вниз;

- рекурсивне розбиття навчальної вибірки на максимально більш «чисті» підмножини;

- розбиття має бути значущим – класифікувати найбільшу кількість елементів навчальної вибірки.

Якщо множина T містить елементи, які відносяться до різних класів, тоді:

$$T = \bigcup_{i=1}^{q_h} T_i$$

$$T_i \subseteq T: x_h = c_{hi} \tag{2.7}$$

Множини T_i більш «чисті» в порівнянні з T.

$$T \coloneqq T_i, \forall_i = 1, \dots, q_h \tag{2.8}$$

Далі, на рис. 2.4 показано приклад розбиття на множини.

Де грає	Суперник	Температура	Дощ	Перемога
Вдома	Вище	Висока	Так	Ні
Вдома	Вище	Висока	Ні	Так
Вдома	Нижче	Норма	Ні	Так
Вдома	Нижче	Висока	Так	Ні
В гостях	Нижче	Норма	Так	Так
В гостях	Нижче	Висока	Так	Ні
В гостях	Нижче	Висока	Ні	Ні
В гостях	Вище	Норма	Ні	Так

Рисунок 2.4 - Розбиття на множини [25]

Критерії вибору змінної розбиття:

- ентропійні;

- джині.

Нехай множина T складається з n об'єктів, k з яких мають властивість S. Тоді ентропія множини T по відношенню до властивості S:

$$H(T,S) = -\frac{k}{n}\log_2\frac{k}{n} - \frac{n-k}{n}\log_2\frac{n-k}{n} \quad (2.9)$$

де, $H(T,S)$=1, коли $k = \frac{n}{2}$.

Якщо S може приймати s різних значень, кожне з яких – в k_i, тоді:

$$H(T,S) = -\sum_{i=1}^{s}\frac{k_i}{n}\log_2\frac{k_i}{n} \quad (2.10)$$

Далі, на рис. 2.5 наведено приклад розрахунку ентропії.

Де грає	Суперник	Температура	Дощ	Перемога
Вдома	Вище	Висока	Так	Ні
Вдома	Нижче	Норма	Ні	Так
В гостях	Нижче	Норма	Так	Так
В гостях	Нижче	Висока	Так	Ні
Вдома	Вище	Висока	Ні	Так
Вдома	Нижче	Висока	Так	Ні
В гостях	Нижче	Висока	Ні	Ні
В гостях	Вище	Норма	Ні	Так

$$H(T,Victory) = -\frac{4}{8}\log_2\frac{4}{8} - \frac{4}{8}\log_2\frac{4}{8} = 1$$

Рисунок 2.5 - Розрахунок ентропії [25]

Міра забрудненості Джині формула (2.11):

$$G(T_i, V) = 1 - \sum_{k=1}^{S}(p_{i,k})^2, \quad (2.11)$$

де $p_{i,k}$ – доля прикладів класу k серед навчальних прикладів в i – му вузлі.

$G(T_i, V) = 0$, якщо T_i містить приклади одного класу[25].

2.2.4 Метод побудови дерева рішень. Критерій зупинки.

Якщо дотримуватися суто теоретичного погляду, то алгоритм зупиниться в той момент, коли усі параметри будуть розподілені по підмножинах T_i. На більш менш великому наборі даних відбудеться проблема перенавчання. Перенавчання – ознака моделі при котрій дерево рішень гарно справляється з навчальною вибіркою, але абсолютно неспроможне працювати на нових даних. Для уникнення цієї проблеми були розроблені наступні підходи:

- рання зупинка;

-зупинка в разі, якщо всі об'єкти в вершині відносяться до одного класу;

- обмеження максимальної глибини дерева;

- завдання мінімально допустимого числа прикладів у вузлі.

Рання зупинка – зупинка алгоритму за критерієм. Тобто, задається певний критерій для навчання, при досягненні якого алгоритм зупиняється, наприклад відсоток правильно розпізнаних даних. В Scikit-Learn клас Decision TreeRegression має різні гіперпараметри.

Обмеження максимальної глибини дерева(max_depth) – зупинка алгоритму в момент досягнення заданої кількості розбиття гілок дерева.

Завдання мінімально допустимого числа прикладів у вузлі(min_samples_split) – обмеження на мінімальну кількість прикладів у вузлі перш ній його можна буде розщепити. Це використовується для уникнення простих розбиттів в яких правило є невагомими.

Min_samples_leaf – мінімальна кількість прикладів у листковому вузлі.

Max_leaf_nodes – максимальна кількість листкових вузлів.

Max_features – максимальна кількість ознак, які оцінюються при розщепленні кожного вузла[28].

Параметри max_... слід зменшувати, а параметри min_... слід збільшувати для зменшення ризику перенавчання дерева. Приклад наведено на рис. 2.6.

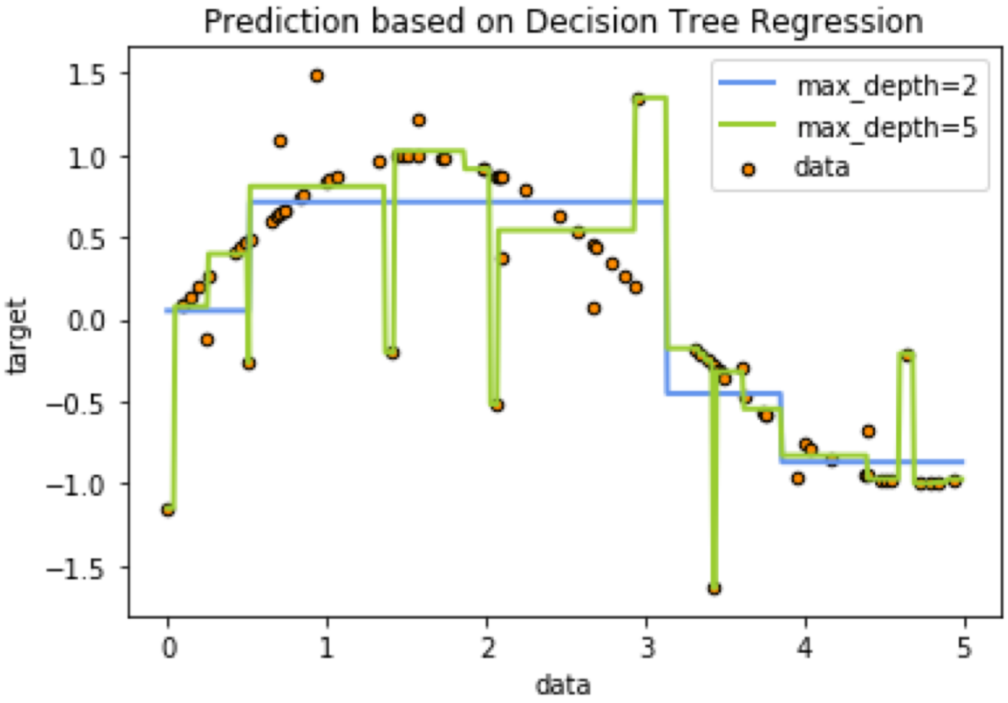

Рисунок 2.6 – Застосування гіперпараметру[24]

2.3 Ансамбль дерев рішень

Давайте уявимо, що ви проводите соціальне опитування серед великої кількості населення, а потім агрегуєте їх відповіді. З великою ймовірністю виявиться, що ця агрегована відповідь буде краща за відповідь експерта у галузі. Це називається колективним розумом. Аналогічно якщо ви агрегуєте прогнози групи прогнозаторів (таких як класифікатори або регресори), то в більшості випадків отримаєте кращі прогнози, ніж прогноз від найкращого індивідуального прогнозатора. Група прогнозаторів називається ансамблем. Відповідно, прийом носить назву ансамблеве навчання, а алгоритм – ансамблевий метод. На рисунку 2.7 зображено приклад ансамблю дерев рішень.

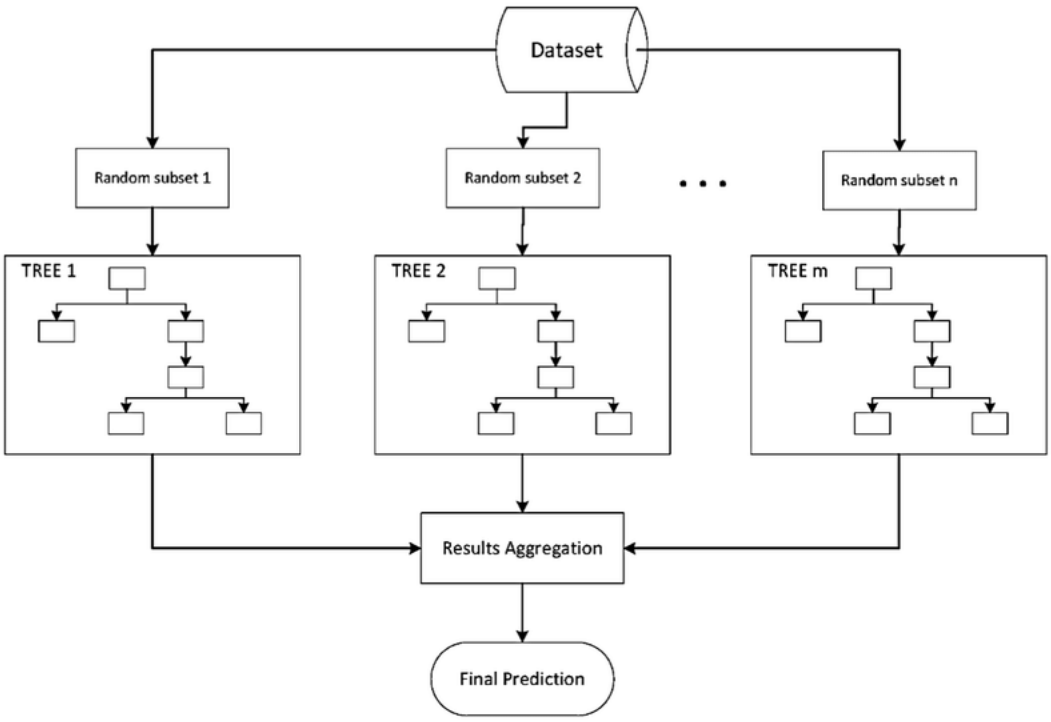

Рисунок 2.7 - Ансамбль дерев рішень[25]

Наприклад, ви можете навчати групу класифікаторів на основі дерев прийняття рішень, використавши для кожного різні випадкові піднабори навчального набору. Для формування прогнозів ви лише отримуєте прогнози всіх індивідуальних дерев і прогнозуєте клас, який став володарем більшості голосів дерев. Такий ансамбль дерев прийняття рішень називається випадковим лісом і, незважаючи на свою простоту, є одним з самих потужних алгоритмів МН, доступних на сьогоднішній день[21]. Самими популярними методами є бегінг(bagging), бустінг(boosting).

Bagging (Bootstrap Aggregation) використовується, коли наша мета - зменшити дисперсію дерева рішень. Тут ідея полягає в тому, щоб створити кілька підмножин даних з навчальної вибірки, обраної випадковим чином. Тепер кожна колекція даних підмножини використовується для навчання їх дерев рішень. В результаті ми отримуємо ансамбль різних моделей. Використовується середнє значення всіх прогнозів для різних дерев, яке є більш надійним, ніж одне дерево рішень.

Boosting це ще один метод ансамблю для створення колекції предикторов. У цьому методі моделі формують ознаки і передають їх

наступним моделям для покращення результату. Іншими словами, ми підбираємо послідовні дерева (випадкова вибірка), і на кожному кроці мета полягає в тому, щоб знайти чисту помилку з попереднього дерева[22].

2.3.1 Алгоритм Random ForestRegressor

Алгоритм RandomForestRegressor базується на технології беггінгу. Кожне дерево в ансамблі навчається на випадковій підмножині навчальних даних. Також алгоритм обирає випадковим чином набір ознак, тобто не використовує всі ознаки для створення кожного дерева. Приклад випадкового лісу наведено на рис. 2.8.

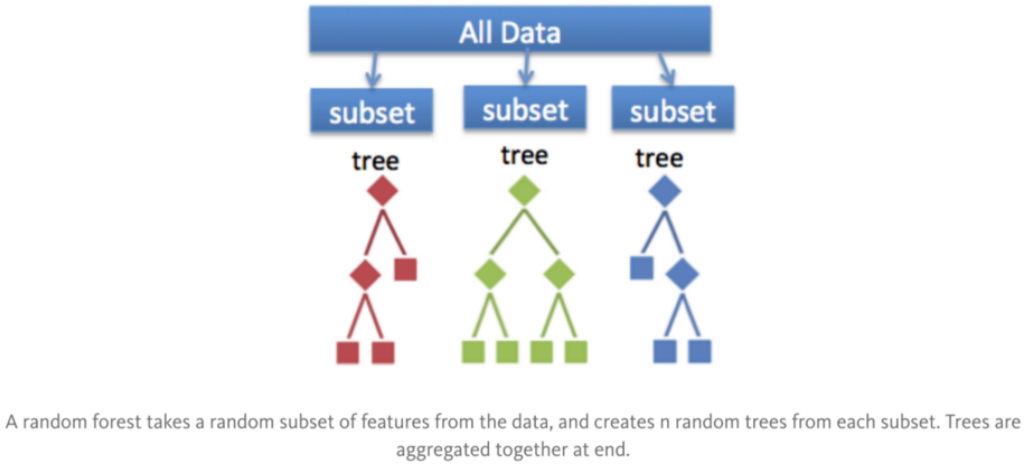

Рисунок 2.8 – Приклад випадкового лісу

Нехай початковий набір даних має N записів (точок) та M ознак (атрибутів). Алгоритм RandomForestRegressor складається з наступних загальних етапів [23]:

1. Випадковим чином із повторенням обирається підвибірка з початкового набору навчальних даних.

2. Випадковим чином обирається підмножина з M ознак. З неї обирається найкраща ознака і використовується для ітеративного поділу вузла.
3. Етапи 1 і 2 повторюються до вичерпання підвибірки.
4. Агрегуються прогнози індивідуальних дерев і отримується остаточний прогноз.

2.3.2 Extremely Randomized Trees

Extremely Randomized Trees додають ще один крок рандомізації до алгоритму випадкового лісу. Random Forest обчислює оптимальний поділ, щоб забезпечити можливість охоплення ознаки в випадково обраній підмножині, і потім обирає кращу ознаку для поділу. Замість цього ExtraTrees вибере випадковий поділ для кожної ознаки в випадковій підмножині, а потім вибере кращу ознаку для поділу, і порівняє ці випадково вибрані ознаки для поділу. Надзвичайно рандомізовані дерева набагато ефективніше в обчислювальному відношенні, ніж метод випадкового лісу, і їх продуктивність майже завжди на одному рівні. У деяких випадках вони можуть навіть працювати краще[24].

2.3.3 Gradient Boosting

Gradient Boosting – один із різновидів методу бустингу. Ансамбль дерев за технологією градієнтного бустингу будується шляхом послідовного навчання окремих дерев рішень. Кожне наступне дерево у ансамблі намагається зменшити помилку як різницю між фактичними і прогнозними

значеннями цільової змінної [22]. Основна ідея градієнтного бустингу у тому, що увага приділяється випадкам, які трудно передбачити, і модель навчається на попередніх помилках. На кожному етапі виявляється слабкий учень і його прогноз порівнюється з правильною відповіддю, яка нам відома відповідно до методу навчання з учителем. Навчання моделі градієнтного бустингу здійснюється алгоритмом градієнтного спуску, і виконується мінімізація обраної диференційованої функції втрат. Градієнт як частинна похідна від обраної функції втрат, характеризує крутизну функції помилок.

Переваги методу градієнтного спуску:

- Можна використовувати різні функції втрат.

Недоліки методу градієнтного спуску:

- Схильний до небажаного перенавчання.
- Потребує ретельного підбору гіперпараметрів.

2.4 Методи та методології прогнозування на основі рекурентних нейронних мереж

2.4.1 Long Short-Term Memory

Однією з варіацій архітектури РНМ є довготривала короткочасна пам'ять (LSTM), яка явно розроблена для уникнення проблем з довгостроковою залежністю. Хоча LSTM має подібну структуру до РНМ, але звичайний LSTM має три вузли (тобто введення, забуття та вихід), блоковий вхід, одну комірку, функцію активації виводу та з’єднання (Greff et al. , 2016). LSTM була першою рекурентною мережевою архітектурою, яка подолала проблему зникаючого та вибухового градієнта. Вузол забуття в LSTM вирішує, яку інформацію пропускати або викинути зі стану комірки, вхідний вузол визначає, яка нова інформація повинна зберігатися в стані комірки, тоді як вихідний вентиль регулює те, що виробляє кожна клітинка. Більше

того, це буде залежати від стану комірки, відфільтрованих та нещодавно доданих даних. На рис. 2.9 показана схема комірки довгострокової пам'яті. Кожен вентиль відповідає ваговій матриці, і його можна дізнатися під час навчання мережі

Схема ванільного LSTM має три вузла (тобто введення, забуття та виведення), блочний вхід, одну комірку, функцію активації виводу та з'єднання[26].

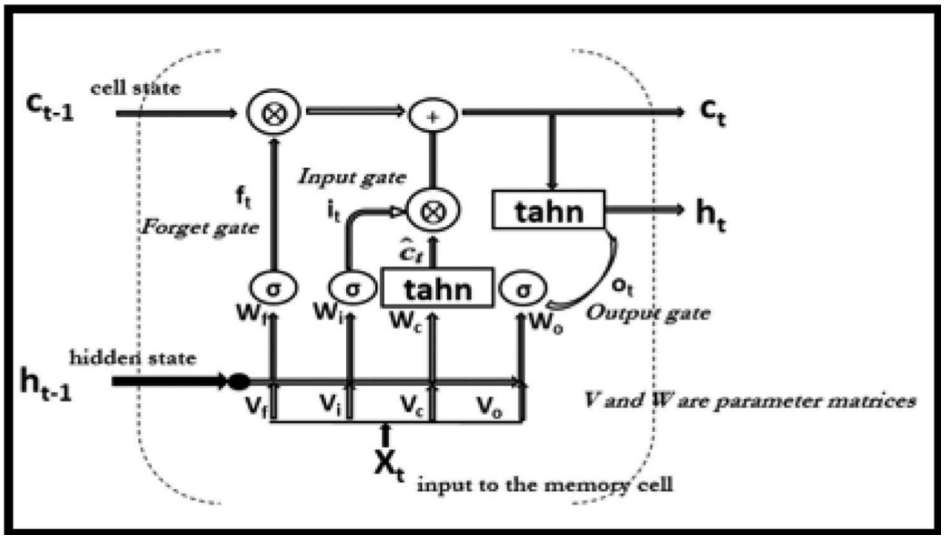

Рисунок 2.9 - Схема комірки довгострокової пам'яті [26]

$$i_t(input\ gate) = \sigma(W_i h_{t-1} v_i x_t + b_i) \quad (2.12)$$

Формула нижче показує очікуване значення поточного стану і вихід з вхідного гейту.

$$\hat{C}_t = tanh(W_c h_{t-1} + v_c x_t + b_c) \quad (2.13)$$

$$f_t(forget\ gate) = \sigma(W_f h_{t-1} + v_f x_t + b_f) \quad (2.14)$$

$$O_t(output\ gate) = \sigma(W_0 h_{t-1} + v_0 x_t + b_0) \quad (2.15)$$

У наведених вище рівняннях W_i , W_c , W_f і W_o представляють вагові матриці вхідних даних, тоді як V_i, V_c, V_f і V_o є ваговими матрицями рекурентних зв'язків. Індекси i, c, f та o означають вхідний вузел, комірку пам’яті, вузел забуття та вихідний вузел відповідно. I, b_i, b_c, b_f і b_o є відповідними зміщеннями кожного гейту, тоді як x_t означає вхід до комірки пам’яті.

Рівняння (2.12), (2.13) і (2.14) представляють вхідний вузел, вузел забуття та вихідний вузел відповідно. Рівняння (2.13) використовується для оновлення стану забуття, який наведено в рівнянні (2.16) нижче.

$$C_t = (f_t C_{t-1} + i_t \hat{C}_t) \tag{2.16}$$

Нарешті, рівняння (2.17) нижче представляє прихований стан, обчислений шляхом фільтрації осередку C_t з вихідним гейтом після застосування функції активації гіперболічного тангенса (tanh).

$$h_t = O_t * \tanh(C_t) \tag{2.17}$$

2.4.2 Вентильний рекурентний вузол

Рівні GRU працюють за тим же принципом, що й LSTM, але він об’єднує вхідний вузол і вузол забуття, а також об’єднує стан комірки та прихований стан. Таким чином, GRU можна описати як простіший механізм стробування, який вимагає менше параметрів, ніж LSTM (Chung et al., 2015). Це робить GRU дешевшим у експлуатації. Рівень GRU краще запам’ятовує інформацію про недавнє минуле, ніж про далеке минуле, щоб виконати поточне завдання, а новіші точки даних, природно, є більш передбачливими,

ніж старі точки даних (Chollet and Allaire, 2018). GRU складається з вузла скидання та вузла оновлення, як показано на рис. 2.10. Вузол скидання поєднує новий вхід із попередньою пам'яттю. Тобто він вирішує, скільки попередньої інформації слід забути. Вузол оновлення, з іншого боку, визначає обсяг попередньої пам’яті для збереження. Він працює аналогічно вузлу забуття та входу в LSTM[27].

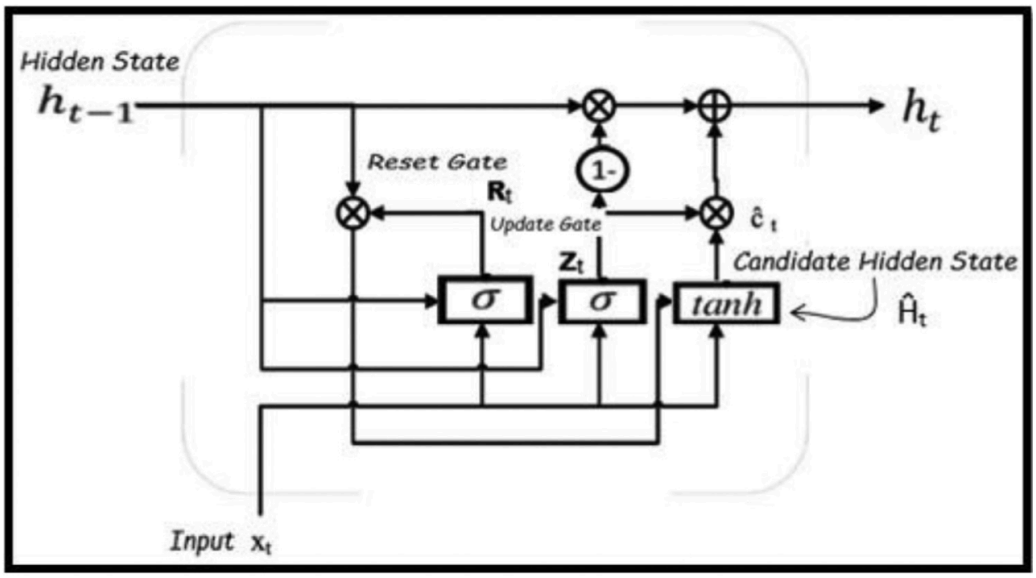

Рисунок 2.10 – Схема GRU [26]

$$Z_t(Update\ Gate) = \sigma\,(W_z[h_{t-1}, x_t] + b_z) \quad (2.18)$$

$$R_t(Update\ Gate) = \sigma\,(W_R[h_{t-1}, x_t] + b_R) \quad (2.19)$$

$$\widehat{H} = tanh(W[R_t * + h_{t-1}, x_t] + b_h) \quad (2.20)$$

$$h_t = (1 - Z_t) * h_{t-1} + Z_t * \widehat{H}_t \quad (2.21)$$

2.4.3 Різниця між LSTM та GRU

Існують відмінності між GRU та LSTM, і вони полягають у тому, що, по-перше, LSTM має три шлюзи, які складаються з шлюза входу, виходу та забуття, а GRU має два вентилі, які скидаються та оновлюються. По-друге,

GRU має внутрішню пам'ять. По-третє, GRU відкриває всю пам'ять і приховані шари, тоді як LSTM — ні. По-четверте, GRU добре застосовується, якщо набір даних невеликий, тоді як LSTM відносно добре працює для більшого набору даних. Схематично різницю відображено на рис. 2.11.[28]

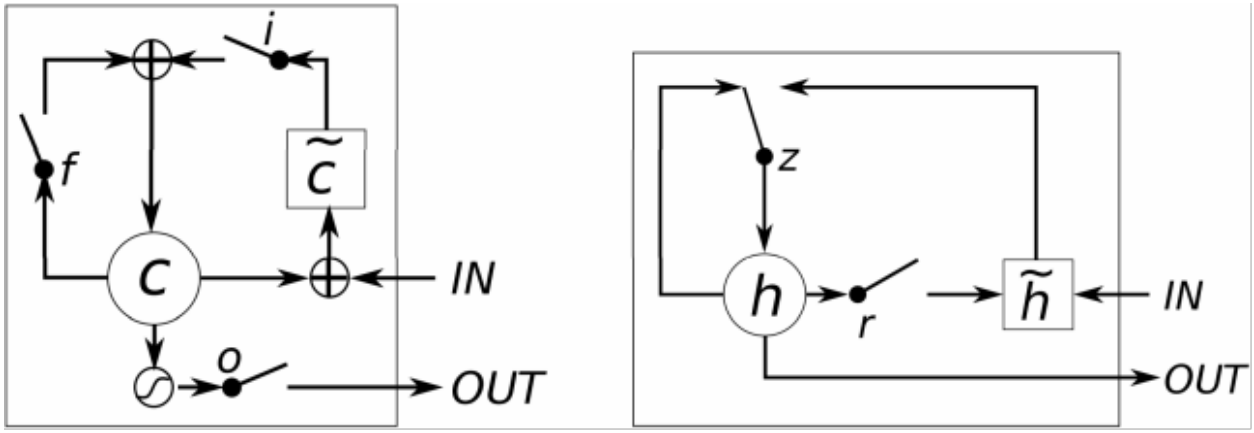

Рисунок 2.11 – Різниця LSTM(ліворуч) та GRU(праворуч) [26]

2.4.4 Алгоритм AdaBoost

Алгоритм AdaBoost є короткою формою Adaptive Boosting, введеною Freund and Schapire (1997). Цей метод зменшує упередження, об’єднуючи різні методи машинного навчання в єдину модель прогнозування. Під час посилення спостереження зважуються так, що деякі з них неодноразово беруть участь у нових комбінаціях. Кожен класифікатор також навчається на даних з урахуванням успіху попередніх класифікаторів. Після кожного кроку тренування ваги перерозподіляються. Неправильно класифіковані дані мають більшу вагу, щоб виділити найскладніші випадки. Таким чином, наступні учні будуть зосереджуватись на них під час навчання. У таблиці 2.1 показано застосування алгоритму AdaBoost.

Таблиця 2.1 – Алгоритм AdaBoost

Підвищення продуктивності	Досягнення цілі	Зона застосування	Функція
За рахунок надання більшої ваги для зразків неправильно класифікованих класів	Розробка потужного класифікатора з надзвичайною точністю збільшує силу прогнозування	Градієнтний спуск	Використання базового слабкого учня для формування сильного учня шляхом зваженого додавання слабких учнів

Для кращої продуктивності алгоритмам ШНМ потрібно більше точок даних. З цієї причини привласнюються внутрішньодобові або добові дані, де можна отримати велику кількість спостережень за менший період. Оскільки економічні умови змінюються з часом, навчання мережі застарілою інформацією може призвести до поганого узагальнення моделі, яке може не відповідати нинішній економічній ситуації (Elshendy et al., 2018). У цьому дослідженні були використані добові дані за останні одинадцять років і вісім місяців, тобто з 23 жовтня 2009 року по 23 червня 2021 року, для прогнозування експортної ціни на нафту. Це графічно показано на рис. 2.12.

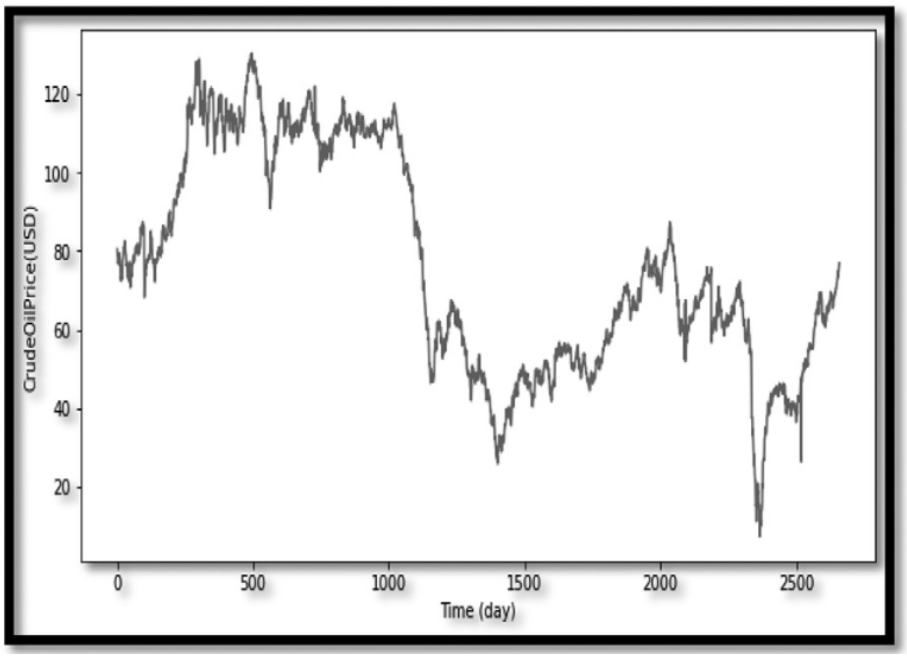

Рисунок 2.12 – Ціна на нафту

В таблиці 2.2 наведено описову статистику даних цього дослідження.

Таблиця 2.2 – Статистика експорту нафти

COUNT	MEAN	STD	MIN	25%	50%	75%	MAX
2660.0	76.38	28.09	7.15	52.95	70.04	108.54	130.43

З таблиці 2.2 ми бачимо, що мінімальна ціна для цих періодів становить 7,15 доларів США, а максимальна експортна ціна – 130,43 доларів США.

Попередня обробка даних в ANN має важливе значення для підвищення точності прогнозування (Ahmed et al., 2010; McAdam and McNelis, 2005; Zhang et al., 1998) і має перевагу зменшення ризику обчислювальних проблем для проведення навчання. процес більш ефективний (Zhang et al., 1998). Перший крок обробки полягає в нормалізації даних за допомогою міні-макс масштабера. Ми розглянули [−1, 1] для виконання нашого аналізу. У вихідному рівні AdaBoost-LSTM і AdaBoost-GRU необхідно встановити діапазон масштабу від −1 до +1 для вхідних і вихідних змінних, щоб вони відповідали масштабу функції активації (тобто гіперболічного tan).

Використовуючи вікно, що рухається вперед розміром 5, ми використали перші 5 точок даних як вхідні дані (провісник, тобто від x1 до x5), щоб передбачити шосту точку даних (ціль, тобто y1), і ми використали вікно, яке- від другої до шостої точки даних як вхідних даних (тобто від x2 до x6) для прогнозування сьомої точки даних тощо. Щоб зафіксувати розмір вікна для переміщення до п’яти, ми використовуємо функцію зсуву pandas, щоб перемістити весь стовпець на вказане число. Дані складаються з 2655 рядків із шістьма стовпцями. Вплив навчальних даних і даних тестування досліджували шляхом їх поділу на 75% і 25% відповідно, як показано в таблиці 2.3 і рис. 2.13.

Таблиця 2.3 – таблиця впливу даних

Розбивка даних	Розмірність даних	Стовпці прогнозування	Цільовий стовпець
Тренувальні дані	1991.6	1-5	6
Тестові дані	664.6	1-5	6

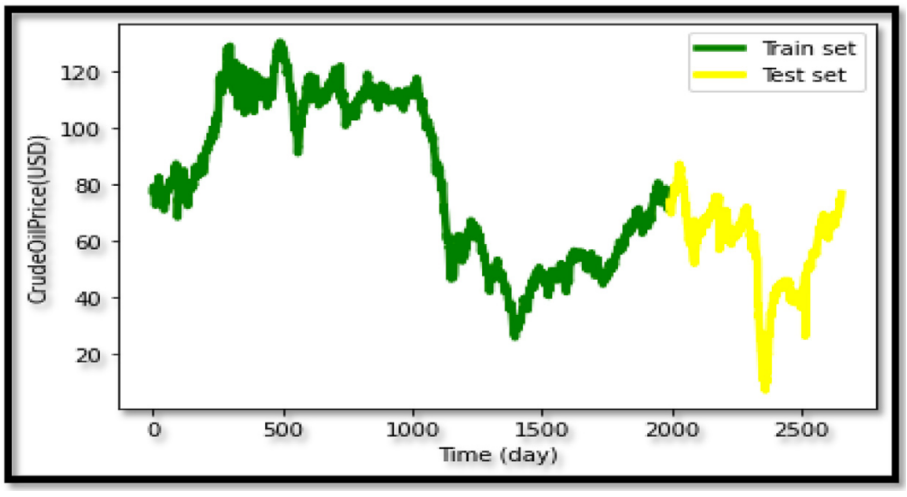

Рисунок 2.13 – Розподіл тестової та тренувальної вибірок

Таким чином, фіксація вікна розміру 5 є першими спостереженнями 1991 року, які враховують 75% загальної кількості даних, як навчальний набір, і решта 664 спостережень призначені для тестового набору.

Очікується, що вхідні та цільові дані Keras LSTM, GRU, AdaBoost-LSTM і AdaBoost-GRU будуть мати фіксовану тривимірну форму. Це

кількість спостережень, які потрібно надати моделі, довжина прихованого стану та кількість предикторів. Коли моделі були створені для LSTM і GRU, було два прихованих шари, які включали 256 нейронів, один нейрон у вихідному шарі, а функції випадання були встановлені на 0,5. Функції вартості були розраховані як середньоквадратична помилка, тоді як оптимізатором була Adaptive Moment Estimation (Adam). Крім того, під час підгонки моделей для епох було встановлено значення 100 з розмірами пакетів 10. Оскільки порядок даних важливий для нашого аналізу, для випадкового (тобто, перемішування) було встановлено значення "false", а детальне значення було встановлено на один . Коли були створені моделі LSTM і GRU, алгоритм AdaBoost був застосований, помістивши їх у внутрішню обгортку sklearn і, нарешті, розширений. На рис. 2.14, що знаходиться нижче, показано етапи обробки даних, які включають застосування алгоритму AdaBoost. Реалізація AdaBoost на будь-якій моделі включає обгортку визначеної моделі та покращує її за допомогою класифікатора AdaBoost або AdaBoost Regressor залежно від випадку.

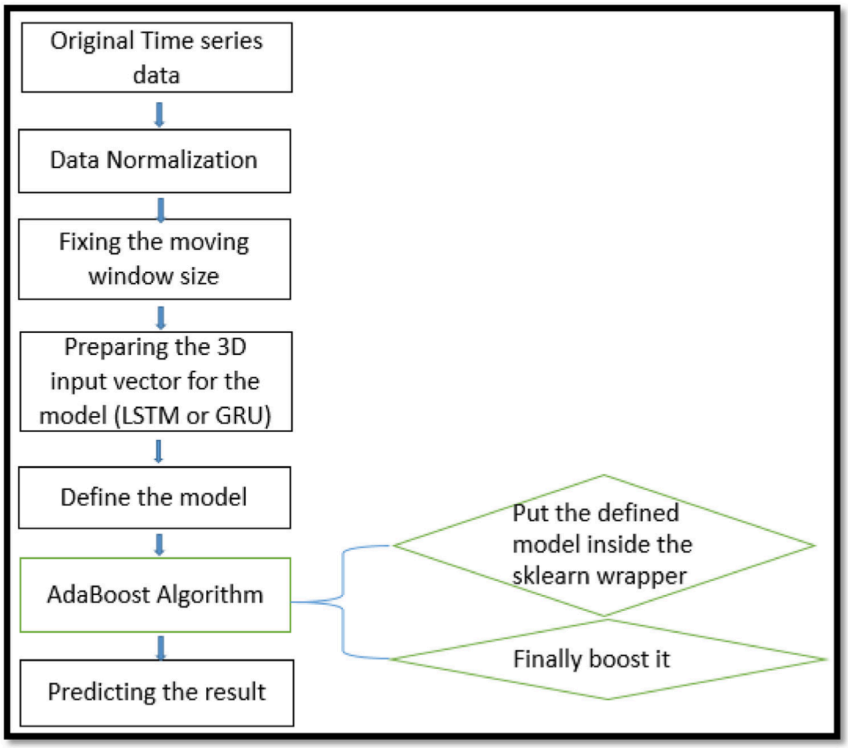

Рисунок 2.14 - Блок-схема етапів обробки даних, включаючи те, як застосовується алгоритм AdaBoost [25]

Існує кілька методів оцінки ефективності прогнозування з різною спрямованістю та характеристиками. Для оцінки ефективності прогнозування запропонованого AdaBoost-GRU порівняно з AdaBoost-LSTM та іншими моделями порівняльного аналізу було використано п’ять різних статистичних даних, а саме: середня абсолютна помилка (MAE), середньоквадратична помилка (RMSE), індекс розсіювання (SI) були використані середня абсолютна відсоткова помилка (MAPE) та середня зважена абсолютна відсоткова помилка (WMAPE).

Середня абсолютна помилка (MAE) і середньоквадратична помилка (RMSE) є найбільш широко використовуваними показниками прогнозування статистиками. Хоча MAE і RMSE легко обчислити, значення в RMSE важко інтерпретувати порівняно з MAE. Обидва ці показники є помилками, що залежать від масштабу, що означає, що ми не можемо використовувати ці показники для порівняння результатів двох різних прогнозів часових рядів з різними одиницями вимірювання Chatfield, 1988). Оскільки значення RMSE не вказують нам, хороші оцінки чи ні, також використовується термін, який називається індексом розсіювання (SI), який визначається для визначення того, чи є RMSE хорошим чи ні. SI, який є нормованим RMSE, розраховується шляхом ділення RMSE на середнє значення даних, і оцінки є прийнятними, якщо значення SI менше одиниці. Отримані тут значення переводять у відсотки. Рівняння 2.22 - 2.24 - математичне представлення MAE та RMSE та SI відповідно.

$$MAE = \frac{\sum_{i=1}^{N} |F_i - A_i|}{N} \tag{2.22}$$

$$RMSE = \frac{\sqrt{\sum_{i=1}^{N} (A_i - F_i)^2}}{N} \tag{2.23}$$

$$SI = \frac{RMSE}{dm} \tag{2.24}$$

Показники оцінки є дуже важливим аспектом роботи, і їх слід брати до уваги, оскільки вони фактично говорять про ступінь ефективності використаних методів. З цієї причини в роботі також використовувалися середня абсолютна відсоткова помилка (MAPE) і зважена середня абсолютна відсоткова помилка (WMAPE), які не залежать від масштабу. Таким чином, їх безпечно використовувати для порівняння продуктивності значень прогнозу часових рядів з різними одиницями. Однак MAPE не є відповідним методом, де є поєднання швидкої та повільної зміни значень, оскільки він цього не розуміє. Тому було застосовано середню зважену абсолютну відсоткову помилку (WMAPE), яка є варіантом MAPE для подолання проблем нескінченних помилок MAPE. Рівняння 2.25 та 2.26 є математичним виразом метрик MAPE та WMAPE відповідно.

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{A_i - F_i}{Ai}\right| \tag{2.25}$$

$$WMPE = \frac{\sum_{i=1}^{N}|A_i - F_i|}{\sum_{i=1}^{N}|A_i|} \tag{2.26}$$

У наведених вище метричних рівняннях N — це кількість точок даних, F_i — прогнозоване значення, A_i — фактичне спостереження, а «dm» — середнє (або середнє) даних[28].

2.4.5 Навчання с частковим залученням вчителя

Зовсім нещодавно з’явилася категорія алгоритмів машинного навчання, відома як напів-контрольоване навчання (SSL), головною перевагою якого є те, що воно дозволяє використовувати переваги як навчання з учителем, так і навчання без нагляду. Основною метою навчання під керівництвом є створення точних класифікаторів або регресорів з використанням розмічених

даних. З іншого боку, навчання без нагляду зазвичай використовується для виявлення структури даних із немаркованих даних. У напів-керованому навчанні значуще представлення складно структурованих даних ідентифікується з немаркованих даних, а потім функція рішення або регресія досягається як для позначених, так і для немічених, що є гладким щодо базової геометрії. SSL вважається більш прагматичною схемою навчання, оскільки багато практичних областей знаходяться в такій ситуації, що існує велика кількість немаркованих даних, але обмежені мічені дані, створення яких може бути дорогим, складним і тривалим. Багато пов’язаних досліджень продемонстрували дійсність SSL у ряді областей застосування, таких як фільтрація спаму, категоризація документів, відеоспостереження, класифікація тексту, фрагментація тексту, експресія генів. Класифікація даних, класифікація веб-сторінок тощо. У цих літературах SSL часто порівнюють з репрезентативними моделями навчання з наглядом і показує свою перевагу над ними завдяки його здатності навчатися лише від кількох позначених дані з використанням великої кількості немаркованих даних. Існує цілий спектр цікавих ідей щодо того, як вчитися як на позначених, так і на немаркованих даних, наприклад, підхід на основі максимізації очікувань, самонавчання, спільне навчання, трансдуктивні опорні векторні машини, а також підходи, засновані на графах, такі як скорочення графіка, гармонічний підхід, локальна та глобальна узгодженість тощо. У SSL на основі графів об’єкти з’єднуються через схожість між ними, і передбачення сутності здійснюється шляхом оцінки поширеного впливу сусідніх сутностей через зв’язки, які існують у них.

Представлення складних відносин між сутностями є природним у структурі навчання SSL, і поширення зміни в сутності чітко пояснюється через структуру мережі. Розглядаючи економічні фактори, включаючи ціну на нафту, як суб'єкти мережі, ці особливості SSL сприятимуть вирішенню складності та нерегулярності проблеми прогнозування ціни на нафту. SSL певною мірою використовувався для оцінки типів об’єктів, які не є часовими

рядами, але не для типів часових рядів. Таким чином, намір тут полягає в тому, щоб використати цей метод представлення зв'язку між об'єктами типу часових рядів, а потім застосувати SSL для прогнозування зростання і зниження цін на нафту. Запропонований підхід SSL буде застосовуватися до прогнозування ціни на нафту West Texas Intermediate (WTI) з січня 1992 року по червень 2008 року і буде перевірено шляхом порівняння з моделлю авторегресії, моделлю логістичної регресії, моделлю ANN та Модель SVM[27].

2.5 Метод опорних векторів

Метод опорних векторів (SVM) добре відомий в задачах класифікації. Однак використання SVM в регресії не так добре задокументовано. Ці типи моделей відомі як опорна векторна регресія (SVR).

Потрібно розуміти переваги SVR у порівнянні з іншими моделями регресії, глибоко зануритися в математику, що лежить в основі алгоритму, і зрозуміти роботу методу на простому прикладі.

У більшості моделей лінійної регресії метою є мінімізувати суму квадратів помилок. Візьмемо, наприклад, звичайні найменші квадрати (OLS). Цільова функція для OLS з одним предиктором (функцією) виглядає так:

$$MIN = \sum_{i=1}^{N}(y_i - w_i x_i)^2, \tag{2.27}$$

де y_i – тестові значення, w_i – коефіцієнт, x_i – предиктор.

Цей алгоритм визнає наявність нелінійності в даних і забезпечує точну модель прогнозування.

Алгоритм регресії SVM називається регресією опорного вектора або SVR. Перш ніж розпочати роботу з алгоритмом, необхідно мати розуміння про те, що насправді таке метод опорних векторів.

У машинному навчанні метод опорних векторів — це контрольовані моделі навчання з відповідними алгоритмами навчання, які аналізують дані, що використовуються для класифікації та регресійного аналізу. У регресії опорного вектора пряма лінія, яка необхідна для розміщення даних, називається гіперплощиною.

Метою алгоритму опорних векторів є знайти гіперплощину в n-вимірному просторі, яка чітко класифікує точки даних. Точки даних по обидва боки від гіперплощини, які є найближчими до гіперплощини, називаються опорними векторами. Вони впливають на положення та орієнтацію гіперплощини і, таким чином, допомагають побудувати SVM.

2.5.1 Гіперпараметри для SVR

Тепер, коли у нас є розуміння про те, що таке метод опорних векторів, ми розглянемо різні гіперпараметри, які використовуються в регресії опорних векторів. Нижче наведено пояснення для ключових параметрів SVR.

Гіперплощинаце – межі рішення, які використовуються для прогнозування безперервного результату. Точки даних по обидва боки від гіперплощини, які є найближчими до гіперплощини, називаються опорними векторами. Вони використовуються для побудови потрібної лінії, яка показує прогнозований результат алгоритму.

Ядро – це набір математичних функцій, який приймає дані як вхідні та перетворює їх у необхідну форму. Зазвичай вони використовуються для пошуку гіперплощини у просторі вищих вимірів. Найбільш широко використовувані такі ядра: лінійне, нелінійне, поліноміальне, радіально-

базисну функцію (RBF) і сигмовидну. За замовчуванням в якості ядра використовується RBF. Кожне з цих ядер використовується залежно від набору даних.

Граничні лінії – це дві лінії, які проведені навколо гіперплощини на відстані ε (епсилон). Вони використовується для створення межі між точками даних.

2.5.2 Регресія методом опорних векторів

На відміну від інших моделей регресії, які намагаються мінімізувати похибку між реальним і прогнозованим значенням, SVR намагається вмістити найкращу лінію в межах порогового значення. SVR – це контрольований алгоритм навчання, який використовується для прогнозування дискретних значень. Регресія опорного вектора використовує той же принцип, що й SVM. Основна ідея SVR полягає в тому, щоб знайти найбільш підходящу лінію. У SVR найкраще підходить лінія — гіперплощина, яка має максимальну кількість точок. Порогове значення – це відстань між гіперплощиною та граничною лінією. Складність SVR за часом більш ніж квадратична з кількістю вибірок, що ускладнює масштабування до наборів даних з більш ніж 10000 вибірок.

Для великих наборів даних використовується лінійний SVR або SGD Regressor. Лінійний SVR забезпечує швидшу реалізацію, ніж SVR, але враховує лише лінійне ядро. Модель, створена за допомогою регресії опорного вектора, залежить лише від підмножини навчальних даних, оскільки функція вартості ігнорує вибірки, прогноз яких близький до цільового значення[29]. Приклад прогнозування методом регресії проілюстровано нижче на рисунку 2.15.

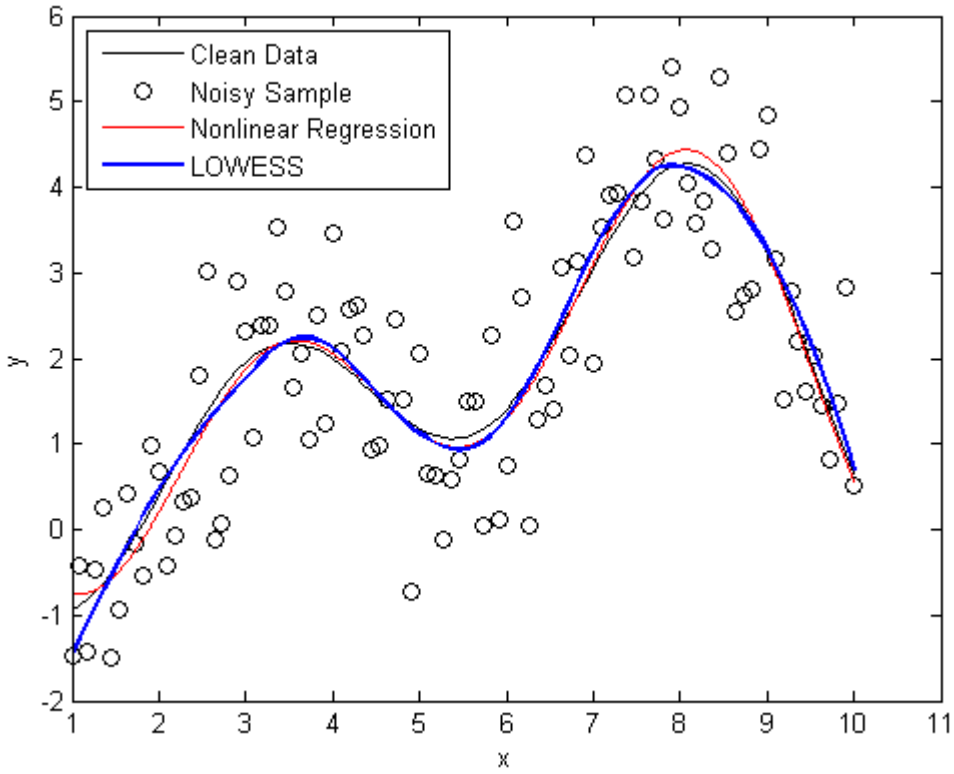

Рисунок 2.15 – Прогнозування методом SVR [29]

2.6 Grey Wolf Optimization

Алгоритм GWO імітує ієрархію лідерства та механізм полювання сірих вовків у природі. Для моделювання ієрархії лідерства використовуються чотири типи сірих вовків, такі як альфа, бета, дельта та омега. Крім того, для оптимізації реалізовано три основні етапи полювання: пошук жертви, оточування здобичі та напад на здобич.

2.6.1 Grey Wolf Optimization

Сірий вовк (Canis lupus) належить до сімейства Canidae. Сірі вовки вважаються хижаками вершини, тобто вони знаходяться на вершині харчового ланцюга. Сірі вовки переважно воліють жити в зграї. Середня кількість груп – 5-12 осіб. Особливо цікаво, що вони мають дуже сувору соціальну домінантну ієрархію.

Лідерами є самець і самка, які називаються альфами. Альфа в основному відповідає за прийняття рішень щодо полювання, місця сну, часу пробудження тощо. Рішення альфи диктуються зграї. Проте також спостерігається певна демократична поведінка, коли альфа слідує за іншими вовками в зграї. Під час зборів вся зграя визнає альфу, притискаючи хвости. Альфа-вовка також називають домінуючим вовком, оскільки його/її накази повинні виконуватися зграєю. Альфа-вовкам дозволено спаровуватися лише в зграї. Цікаво, що альфа не обов'язково є найсильнішим членом зграї, але найкращим з точки зору управління зграєю. Це показує, що організованість і дисципліна зграї набагато важливіші за її силу.

Другим рівнем в ієрархії сірих вовків є бета. Бета — це підлеглі вовки, які допомагають альфі приймати рішення чи інші види діяльності зграї. Бета-вовк може бути як чоловіком, так і жінкою, і він/вона, ймовірно, є найкращим кандидатом на роль альфою, якщо один з альфа-вовків помре або стане дуже старим. Бета-вовк повинен поважати альфу, але також командує іншими вовками нижчого рівня. Він грає роль порадника альфи та капітана зграї. Бета-версія підсилює команди альфи в усьому пакеті та дає зворотний зв'язок альфа-версії.

Сірий вовк найнижчого рангу - омега. Вовки Омега завжди повинні підкорятися всім іншим домінуючим вовкам. Це останні вовки, яким дозволено їсти. Може здатися, що омега не є важливою особистістю в зграї, але було помічено, що вся зграя стикається з внутрішньою боротьбою та

проблемами у разі втрати омеги. Це допомагає задовольнити всю зграю та підтримувати структуру домінування. У деяких випадках омега також є нянькою в зграї.

Якщо вовк не є альфа, бета чи омега, його називають підлеглим (або дельтою в деяких посиланнях). Дельта-вовки повинні підкорятися альфам і бета, але вони домінують над омегою. До цієї категорії належать розвідники, дозорі, старійшини, мисливці та доглядачі. Розвідники несуть відповідальність за спостереження за межами території та попередження зграї у разі будь-якої небезпеки. Стражі захищають і гарантують безпеку зграї. Старійшини — це досвідчені вовки, які раніше були альфа- або бета-версією. Мисливці допомагають альфам і бетам, коли полюють на здобич і забезпечують їжу зграї. Нарешті, доглядачі відповідають за догляд за слабкими, хворими та пораненими вовками в зграї. На рисунку 2.16 показано ієрархію в сімействі вовків.

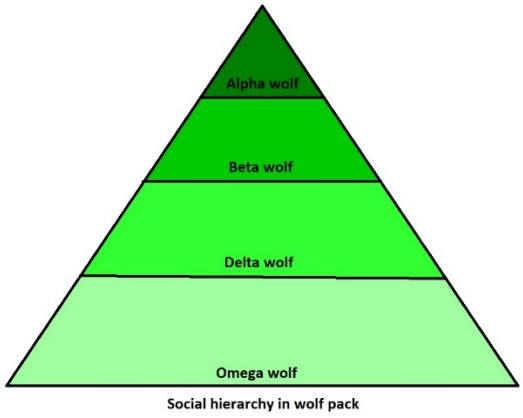

Рисунок 2.16 – Ієрархія в сімействі вовків [30]

2.6.2 Структура алгоритму GWO

Алгоритм GWO має таку структуру:

- ініціалізувати популяцію вовків X_i(i=1…n);
- ініціалізуйте α, A і C;
- обчисліть придатність кожного пошукового агента;

- X_α це найкращий пошуковий агент;
- X_β це другий найкращий пошуковий агент;
- X_δ це третій найкращий пошуковий агент;
- while (t < Max number of intereations);
- for each пошуковий агент;
- оновіти позицію поточного агента пошуку за наведеними вище рівняннями;
- end for ;
- оновити α, A і C;
- розрахувати придатність усіх пошукових агентів;
- оновити X_α, X_β, X_δ;
- t = t+1;
- end while;
- return X_α.

Щоб побачити, як GWO теоретично здатний вирішувати проблеми оптимізації, можна звернути увагу на деякі моменти:

- Запропонована соціальна ієрархія допомагає GWO зберігати найкращі рішення, отримані на даний момент під час ітерації.
- Запропонований окружаючий механізм визначає коло у формі кола навколо рішень, яке може бути розширено до більших розмірів як гіперсфера.
- Випадкові параметри A і C допомагають рішенням-кандидатам мати гіперсфери з різними випадковими радіусами.
- Запропонований метод полювання дозволяє за допомогою потенційних рішень визначити ймовірне положення здобичі.
- Розвідка та експлуатація гарантуються адаптивними значеннями α та A.
- Адаптивні значення параметрів a та A дозволяють GWO плавно переходити між розвідкою та експлуатацією.

- При зменшенні A половина ітерацій присвячена дослідженню (|A|≥1), а інша половина присвячена експлуатації (|A|<1).
- GWO має лише два основних параметри, які потрібно налаштувати (α і C).

Щоб краще зрозуміти цей алгоритм, нижче буде наведено один конкретний приклад поведінки вовчої зграї. На рисунку 2.17 (ліворуч) ми бачимо початковий стан агентів, де видобуток або оптимальний розчин має червоний колір. Найближчі до здобичі вовки (альфа, бета і дельта) мають зелений, синій і фіолетовий колір відповідно. Чорні точки представляють інших вовків (омег), рисунок 2.17.

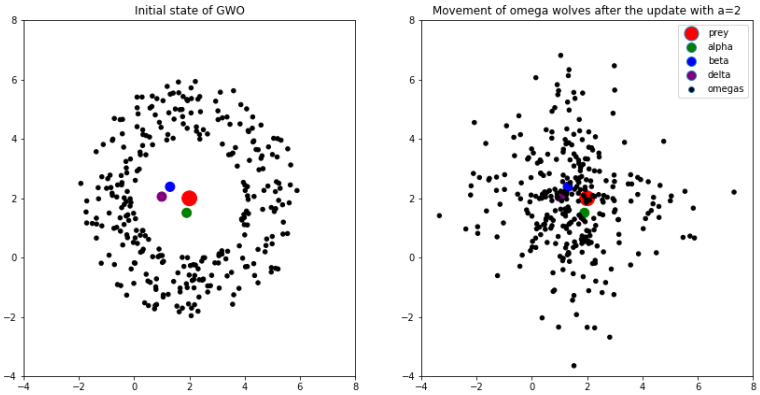

Рисунок 2.17 – початковий стан агентів [30]

Далі, якщо ми оновимо положення омега-вовків відповідно до формул, описаних раніше, ми зможемо спостерігати поведінку агентів на рисунку 2.17 (праворуч).

По-друге, наступний рисунок 2.18 показує поведінку вовчої зграї, коли для змінної α встановлено значення 1. Зазвичай це означає, що ми знаходимося в середині процесу оптимізації, і процес експлуатації все частіше з’являється рисунок 2.18.

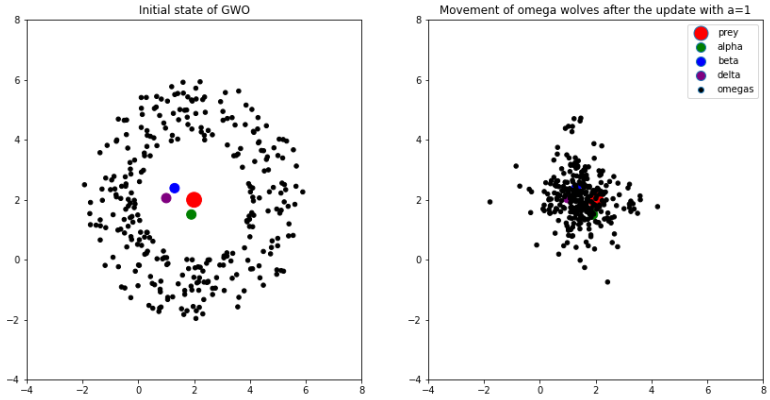

Рисунок 2.18 – Зміна поведінки вовчої зграї [30]

Зверніть увагу, що змінна α, яка контролює компроміс між дослідженням і експлуатацією, має значення 2. Це означає, що ми віддаємо перевагу надмірній експлуатації дослідження.

Нарешті, на рисунку 2.19 змінна α дорівнює 0. Це означає, що процес оптимізації завершується, і ми сподіваємося, що ми зближуємося до оптимального рішення після трьох найкращих вовків. Ми бачимо, як омега-вовки збираються між трьома найкращими вовками, навколо точки, яка геометрично представляє центроїд між альфа-, бета- та дельта-агентами, що показано на рисунку 2.19.

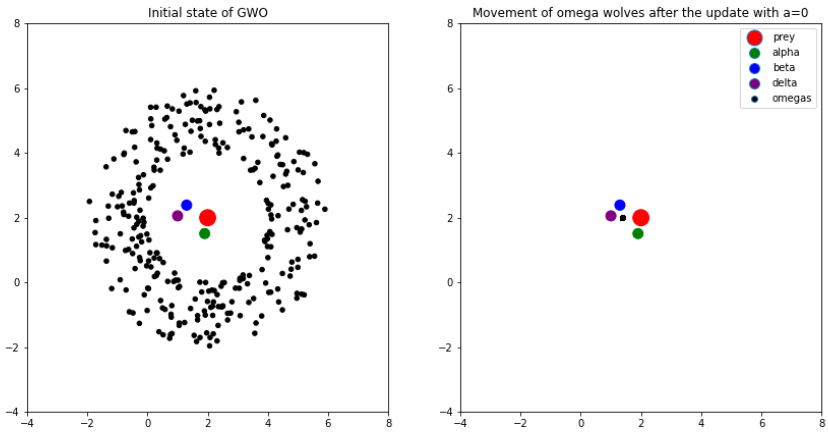

Рисунок 2.19 – Процес оптимізації алгоритму [30]

Зверніть увагу, що ми не оновлювали положення альфа-, бета- та дельта-агентів лише через порівняння з омегами.

2.7 Висновки

Даний розділ присвячено регресійному аналізу та методам машинного навчання, що його використовують. Окремо приділено увагу прогнозуванню на основі рекуретних нейронних мереж. Порівняння результатів прогнозування за допомогою різних моделей та обрання найкрощої. Сама тема регресійного аналізу є досить великою та застосовується у різних напрямках аналізу. У конкретній роботі РА використовується для прогнозування цін на нафту. Оскільки формування ціни є досить складним, то лінійні методи відпадають одразу. Я вирішив працювати у двох напрямках, перший - підхід з використанням ансамблю дерев рішень і класичних методів, другий – використання рекурентних нейронних мереж. Цей підхід є досить оптимальним, адже завдяки своїм особливостям гарантує досить гарний результат прогнозу та погляд на задачу з різних сторін. Мною були розглянуті методі беггінг та бустінг. Для реалізації програмного продукту я обрав методи Random Forest, Decision Trees, GradientBoosting, ExtremelyRandomizedTrees, LSTM, GRU, AdaBoost, SVM. З моєї точки зору вони показали досить гарні результати.

РОЗДІЛ 3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНІ РЕЗУЛЬТАТИ

3.1 Вступ

Модель для прогнозування цін на нафту була розроблена з використанням теоретичних відомостей, історичних фактів, методів інтелектуального аналізу даних та машинного навчання.

Для створення програмного продукту була обрана мова програмування Python. Вона є досить зручною у використанні, має велику швидкодію, широкий функціонал та спектр можливостей. Завдяки різноманітним бібліотекам та інструментам кінцевий продукт є дуже високої якості, а час на його розробку зменшується у порівнянні з іншими мовами програмування.

Програмний продукт є гнучким, кросплатформеним і може використовуватися на будь-якому комп’ютері з встановленим Python і відповідними бібліотеками.

3.2 Бібліотеки

Використані бібліотеки:

1. pandas – використовується для роботи з даними.
2. numpy – використовується для роботи з багатовимірними масивами та матрицями.
3. sklearn.ensemble – бібліотека для робити з ансамблями дерев рішень.
4. sklearn.tree – бібліотека для роботи з деревами рішень.
5. sklearn.linear_model – бібліотека з лінійними моделями.
6. sklearn.model_slection – бібліотека для роботи з даними.
7. sklearn.metrics – бібліотека функцій оцінки, продуктивності.

8. matplotlib.pyplot – використовується для побудови графіків.

3.3 Експерименти

Нижче, у таблиці 3.1, наведено основні метрики, за якими оцінювалися моделі.

Таблиця 3.1 – Метрики моделей

Назва моделі	R-square Value	Mean Squared Error	Explained Variance Score	Mean Absolute Error	Median Absolute Error
Random Forest	0.9600078	46.14370	0.960588	4.770672	2.866699
Extremely Trees	0.98745845	14.4706820	0.987468	2.583831	1.5214999
Gradient Boosting	0.9833593	19.200260	0.983360	3.01084194	2.1799603
Decision Tree Regression	0.8864948	130.964509	0.891784	7.203219	3.6531451
Linear Regression	0.7018832	343.97296	0.7412638	14.553832	12.377887
Hist Gradient Boosting	0.892661	123.84887	0.90026	7.302708	3.703953

На рис. 3.1 – 3.39 наведено результати роботи програми.

```
SCORERANDOMFOREST 0.986150697933221 1
SCORE_HISTGRADIENT 0.8383279532148777
SCORE_EXTRATREE 0.9917463829871511
SCOREGRADIENTBOOSTING 0.985413407951858
SCOREDECISIONTREE 0.8819225968027677
SCORELINEAR 0.5906587746235229
```

Рисунок 3.1 – Точність алгоритмів

```
 ***Random Forest Regressor***
R-Square Value: 0.9891189476534826
Mean Squared Error 9.894086251625314
Explained Variance Score 0.9892686149475304
Mean Absolute Error 2.353508888888908 4
Median Absolute Error 1.747900000000005
```

Рисунок 3.2 – Оцінки методу Random Forest

```
 ***Extra Tree Model***
R-Square Value: 0.9913393285609078
Mean Squared Error 7.875105043750019
Explained Variance Score 0.9915671008708611
Mean Absolute Error 1.850298611111133
Median Absolute Error 1.187499999999993
```

Рисунок 3.3 – Оцінки методу ExtremelyTrees

```
 ***Gradient Tree Boosting***
R-Square Value: 0.979191550572670 7
Mean Squared Error 18.921018559613387
Explained Variance Score 0.9792249108229091
Mean Absolute Error 2.938590306751457
Median Absolute Error 1.865376525542853 2
```

Рисунок 3.4 – Оцінки методу GradientBoosting

```
 ***Decision Tree Regressor***
R-Square Value: 0.950143075615325 3
Mean Squared Error 45.334650950431154
Explained Variance Score 0.9502101907251657
Mean Absolute Error 4.057910642489286
Median Absolute Error 2.677285714285716 3
```

Рисунок 3.5 – Оцінки методу Decision Tree Regression

```
 ***Linear Regression Model***
R-Square Value: 0.741934053053640 5
Mean Squared Error 234.65806949379123
Explained Variance Score 0.7419486804198145
Mean Absolute Error 13.008459979142405
Median Absolute Error 11.885118740179676
```

Рисунок 3.6 – Оцінки методу Linear Regression

```
 ***HIST_Gradient Tree Boosting***
R-Square Value: 0.8383279532148777
Mean Squared Error 119.58043644576031
Explained Variance Score 0.8537347440867262
Mean Absolute Error 6.347666459293221
Median Absolute Error 3.1393096111846583
```

Рисунок 3.7 – Оцінка методу HistGradientBoosting

```
 ***Ada Bossting***
R-Square Value: 0.9292777480917822
Mean Squared Error 62.36267440623768
Explained Variance Score 0.9305024105829655
Mean Absolute Error 5.530599831330918
Median Absolute Error 3.7211805555555815
```

Рисунок 3.8 – Оцінка методу Ada Boosting

Рисунок 3.9 – Реальні ціни на нафту 2017 рік

Month,Year,Predicted Price
,1,2017,48.525225
,2,2017,49.82722500000016
,3,2017,49.36537499999995
,4,2017,53.39965000000026
,5,2017,57.67912500000028
,6,2017,58.42012499999998
,7,2017,52.463800000000084
,8,2017,46.77847500000013
,9,2017,46.77847500000013
,10,2017,46.77847500000013
0,11,2017,46.77847500000013
1,12,2017,46.77847500000013

Рисунок 3.10 – Прогнозування методом Random Forest

Month,Year,Predicted Price
),1,2017,52.78199999999995
.,2,2017,50.57999999999995
,3,2017,47.82000000000002
,4,2017,54.4499999999999
,5,2017,54.16209999999994
,6,2017,52.371899999999954
,7,2017,50.899999999999984
,8,2017,42.86999999999992
,9,2017,45.38699999999992
),10,2017,49.09949999999993
.0,11,2017,49.657499999999935
.1,12,2017,47.13919999999994

Рисунок 3.11 – Прогнозування ціни методом ExtraTrees

Month,Year,Predicted Price
),1,2017,47.2239317967904
l,2,2017,50.6162531301886
?,3,2017,47.80064194602698
3,4,2017,54.417009060197174
4,5,2017,51.31893791354574
5,6,2017,59.73729084562934
6,7,2017,60.99356580056546
7,8,2017,42.83952020280306
3,9,2017,40.4460626094592
9,10,2017,33.11633756620499
l0,11,2017,28.164322303529495
l1,12,2017,13.916635471499267

Рисунок 3.12 – Прогнозування ціни методом GradientBoosting

Month,Year,Predicted Price
),1,2017,52.556666666666665
.,2,2017,52.556666666666665
,3,2017,52.556666666666665
,4,2017,52.556666666666665
,5,2017,52.556666666666665
,6,2017,52.556666666666665
,7,2017,46.885
',8,2017,46.885
,9,2017,46.885
),10,2017,46.885
.0,11,2017,46.885
.1,12,2017,46.885

Рисунок 3.13 – Прогнозування ціни методом Decision Tree

Month,Year,Predicted Price
),1,2017,91.64301934085688
L,2,2017,92.15656812415
2,3,2017,92.67011690744312
3,4,2017,93.18366569073714
4,5,2017,93.69721447403026
5,6,2017,94.21076325732338
5,7,2017,94.7243120406165
7,8,2017,95.23786082390961
3,9,2017,95.75140960720273
9,10,2017,96.26495839049585
L0,11,2017,96.77850717378897
L1,12,2017,97.292055957083

Рисунок 3.14 – Прогнозування методом лінійної регресії

Month,Year,Predicted Price
),1,2017,81.83569642906033
.,2,2017,85.68668664253124
2,3,2017,85.68668664253124
3,4,2017,85.68668664253124
4,5,2017,86.1968314888789
5,6,2017,84.29408485933438
5,7,2017,84.43837810866194
7,8,2017,90.19647662405247
3,9,2017,87.85429839346364
9,10,2017,87.03905068334353
.0,11,2017,86.37080142223523
.1,12,2017,84.69146707880039

Рисунок 3.15 – Прогнозування методом HistGradient

Month	Mean Price of Crude Oil
1	40,622
2	40,66566667
3	42,19333333
4	43,36633333
5	43,72133333
6	44,15
7	44,67733333
8	44,51666667
9	44,24758621
10	43,30724138
11	42,1662069
12	40,77551724

Рисунок 3.16 – Прогнозування методом AdaBoosting

```
[{'normalize': True}
 {'criterion': 'mse', 'max_depth': None, 'max_features': 'auto', 'min_samples_split': 2, 'n_estimators': 450}
 {'learning_rate': 0.2, 'loss': 'ls', 'max_features': 'auto'}
 {'learning_rate': 0.2, 'loss': 'least_squares'}
 {'criterion': 'friedman_mse', 'splitter': 'best'}
 {'criterion': 'mse', 'max_depth': None, 'n_estimators': 150}]
```

Рисунок 3.17 – Використання GridSearch для пошуку оптимальних параметрів

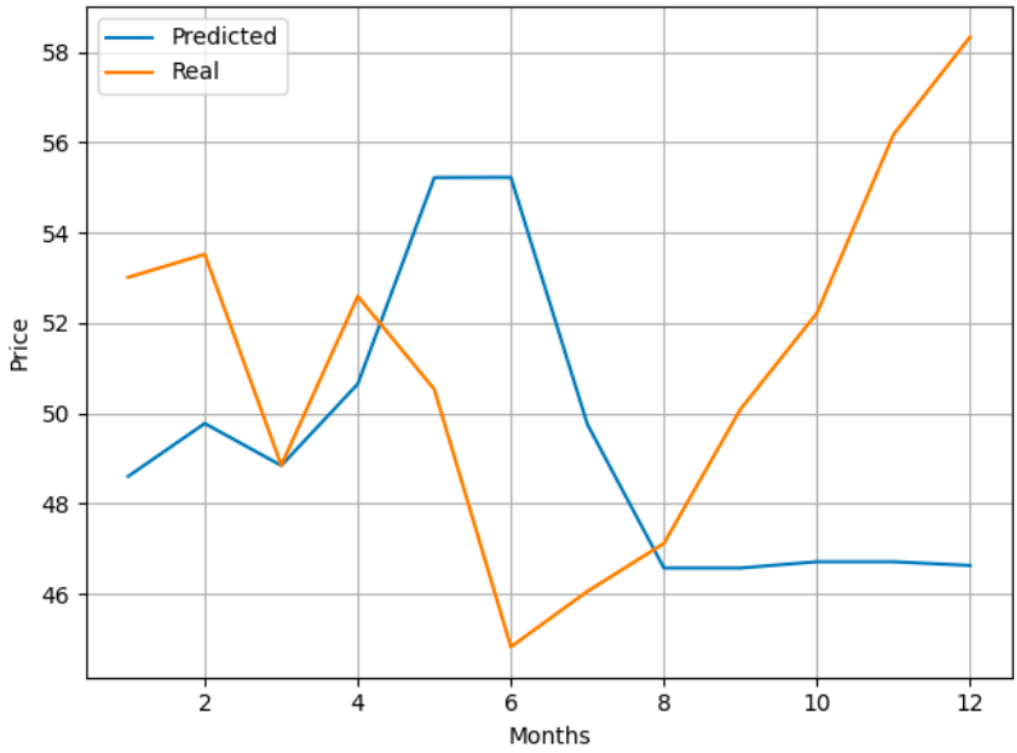

Рисунок 3.18 – Порівняння цін Random Forest та реальними

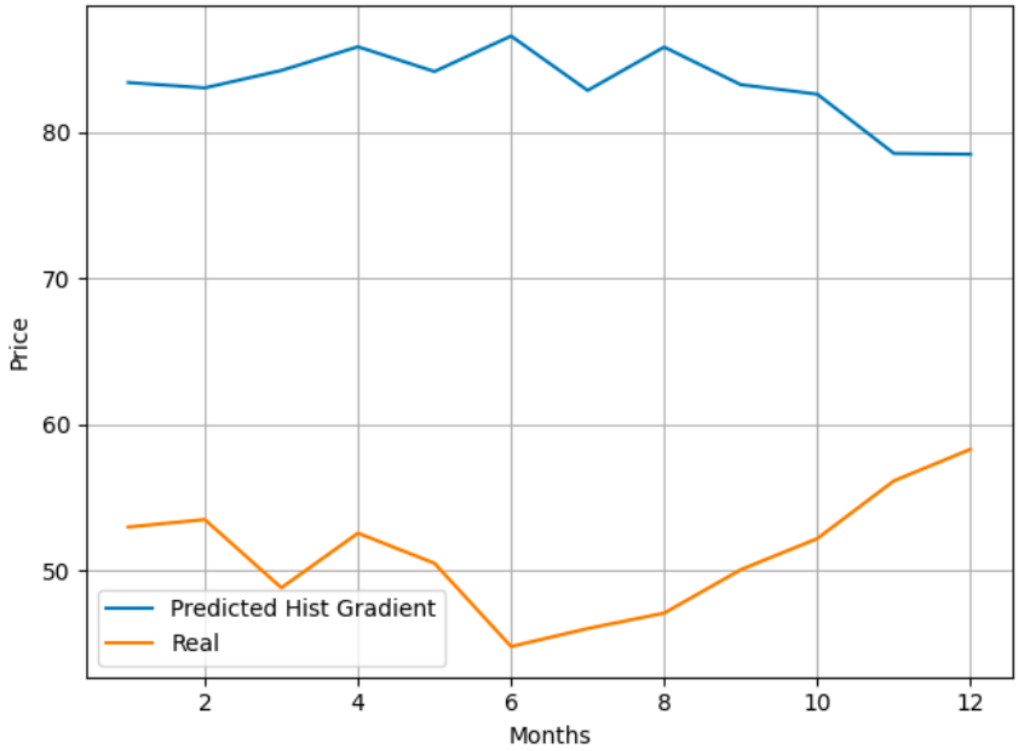

Рисунок 3.19 – Порівняння цін

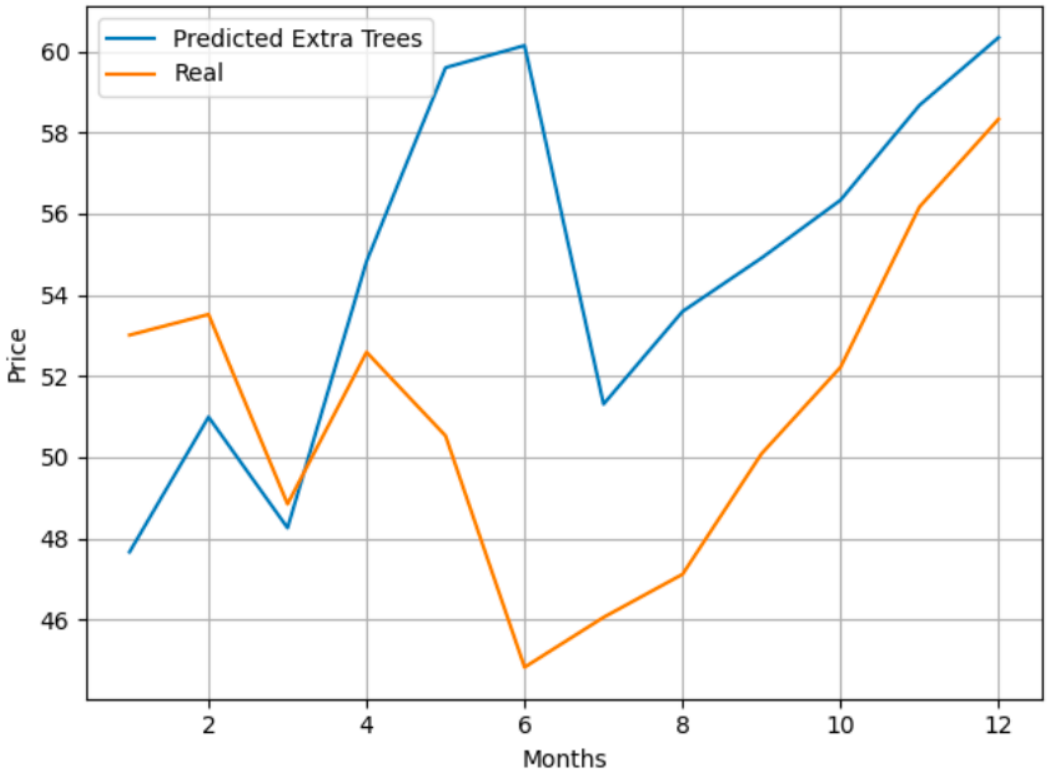

Рисунок 3.20 – Порівняння цін

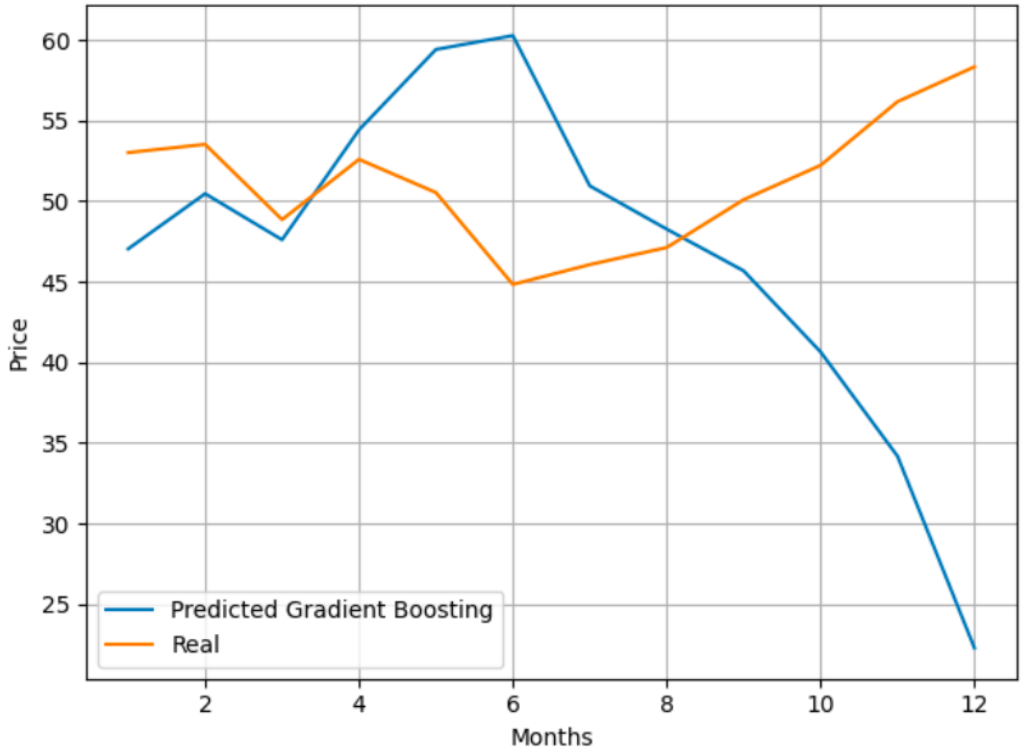

Рисунок 3.21 – Порівняння цін

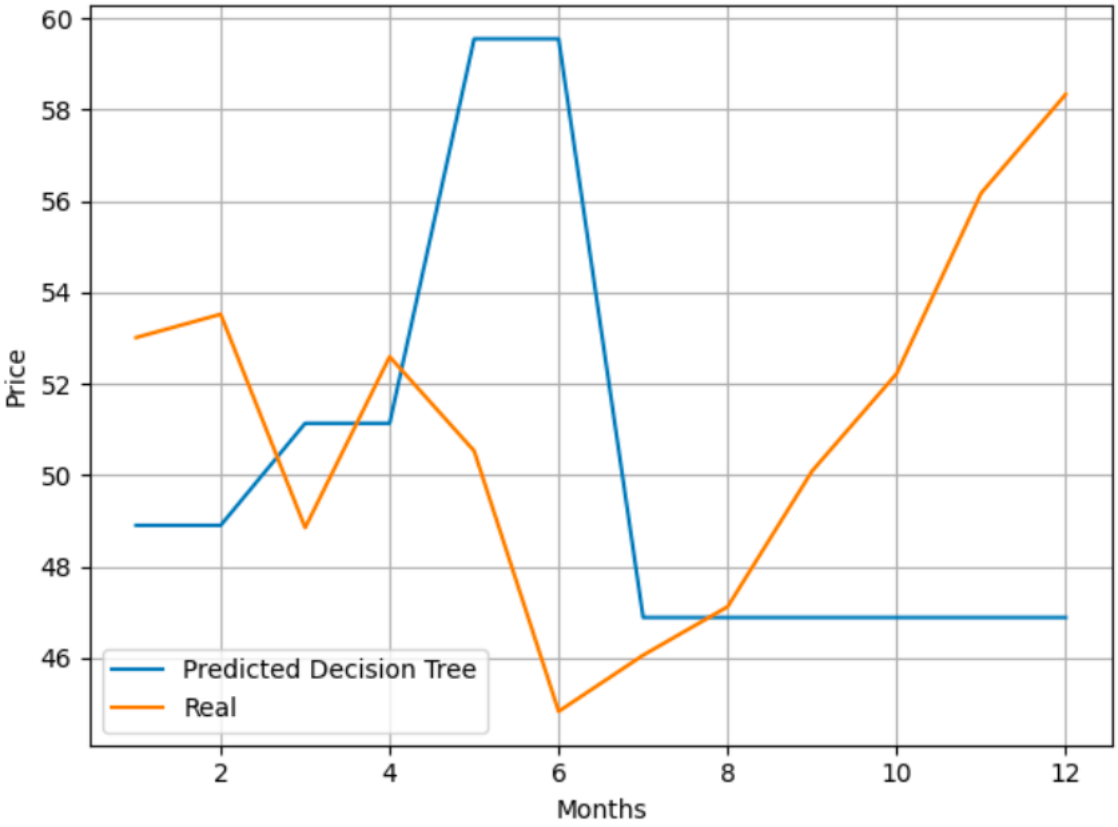

Рисунок 3.22 – Порівняння цін

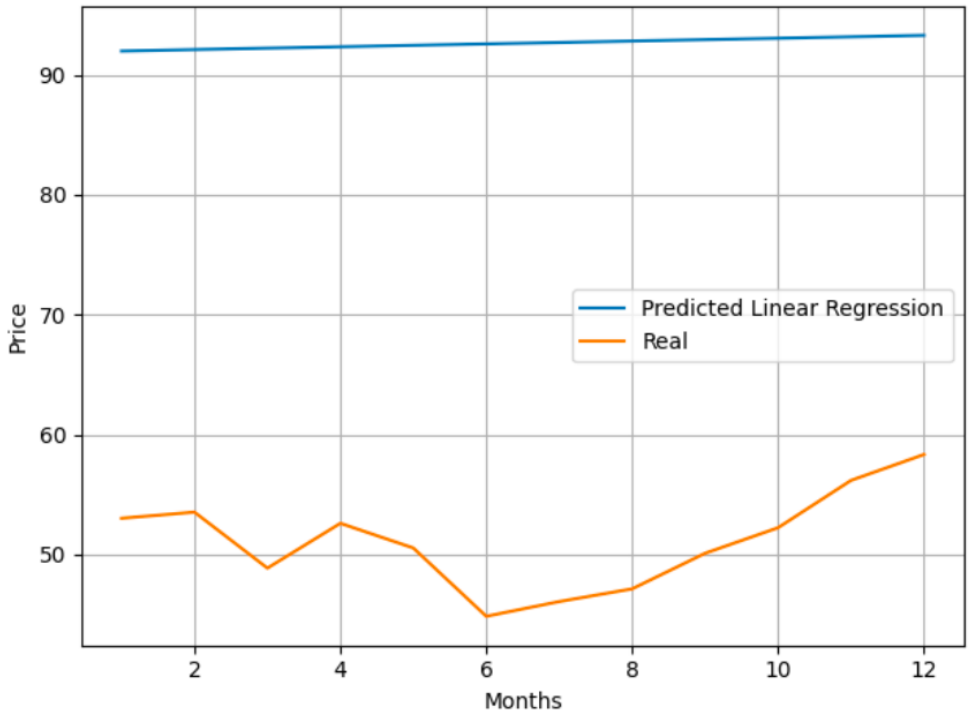

Рисунок 3.23 – Порівняння цін

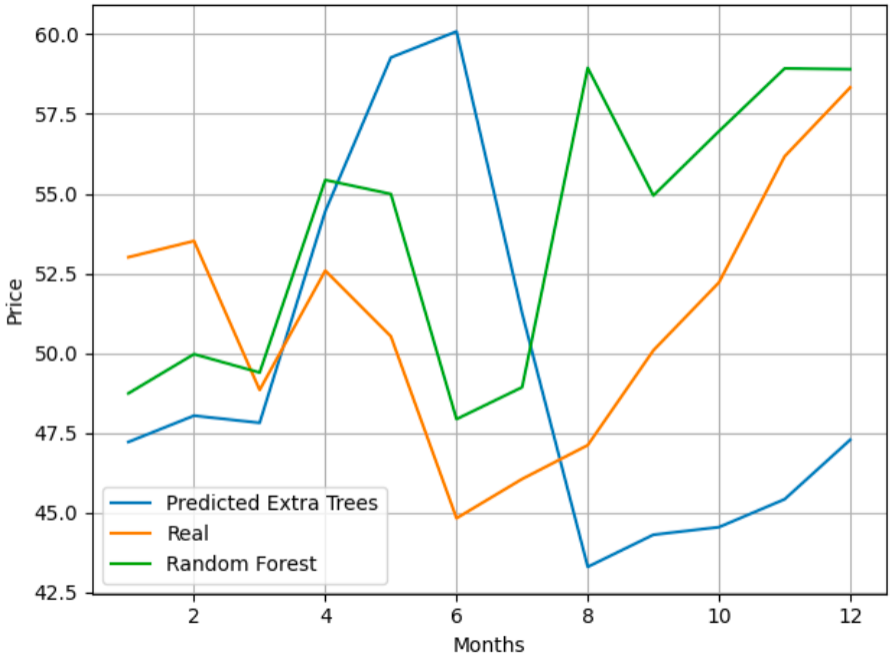

Рисунок 3.24 – Порівняння цін

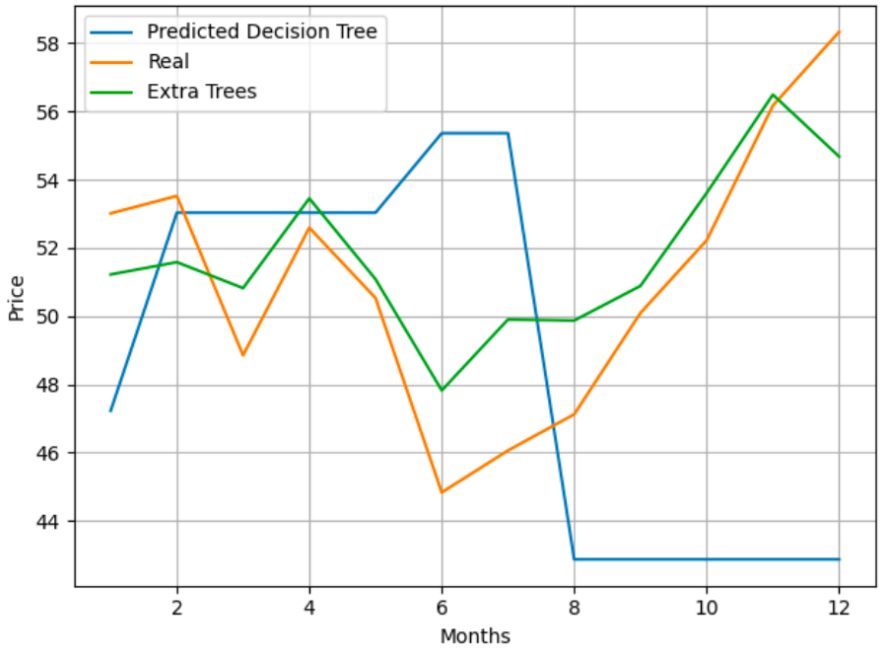

Рисунок 3.25 – Порівняння цін

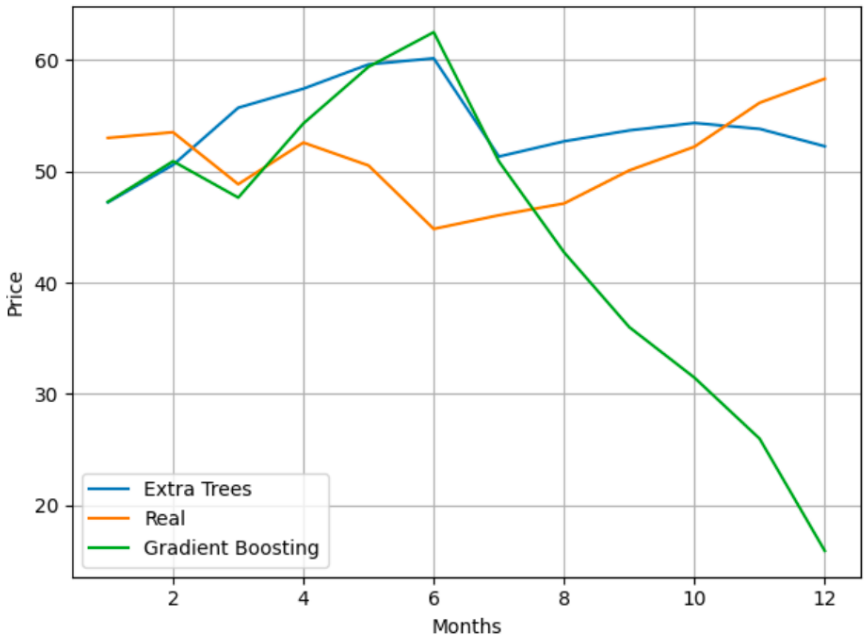

Рисунок 3.26 – Порівняння цін

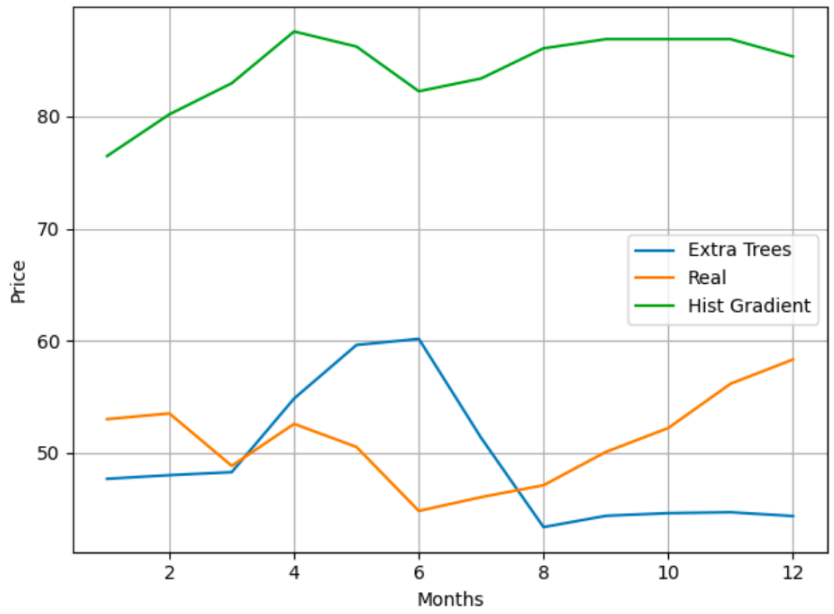

Рисунок 3.27 – Порівняння цін

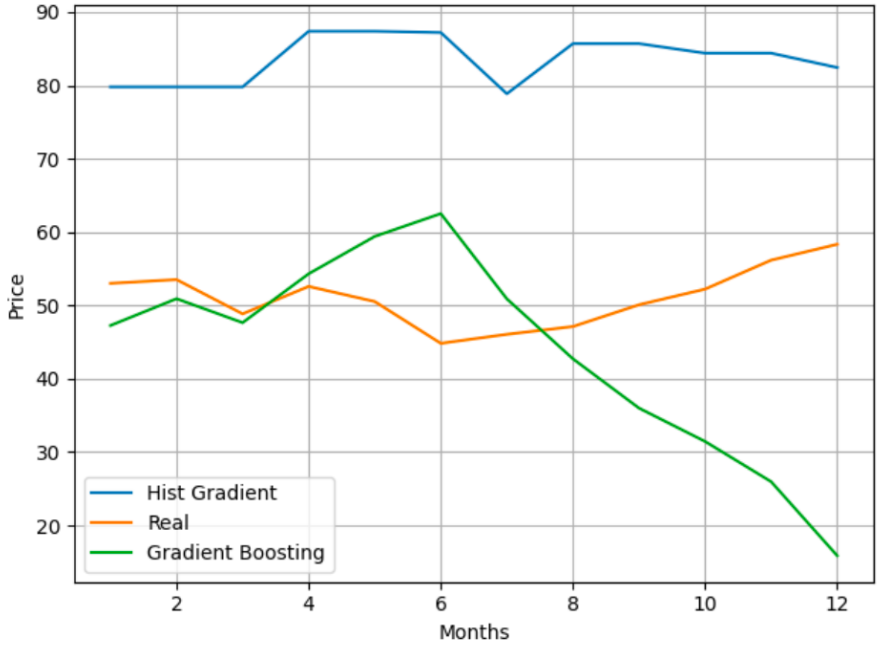

Рисунок 3.28 – Порівняння цін

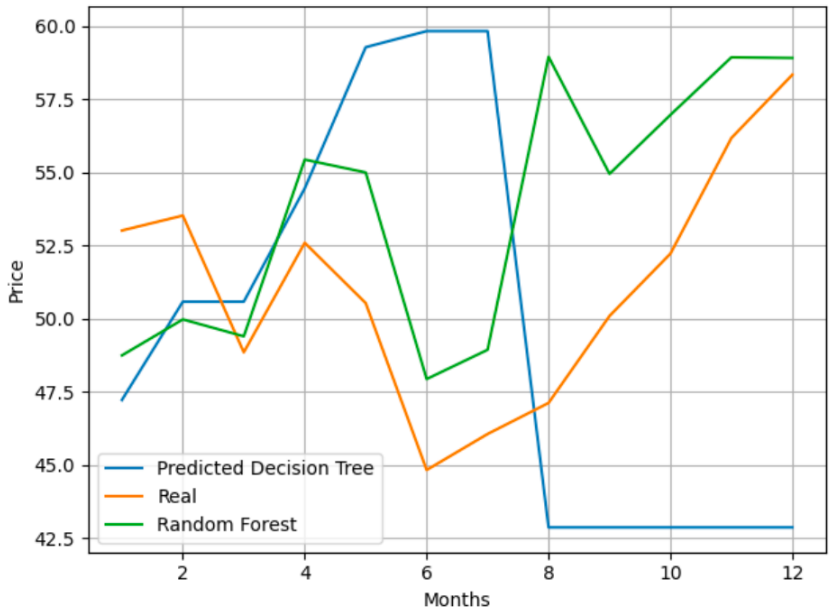

Рисунок 3.29 – Порівняння цін

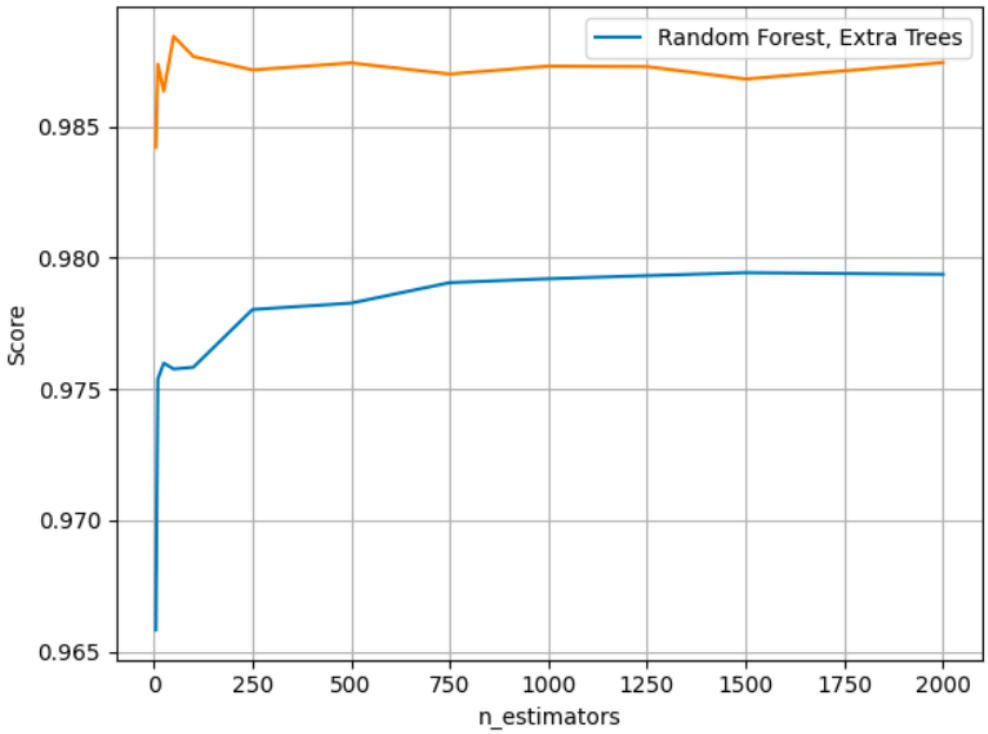

Рисунок 3.30 – Порівняння точності для Random Forest та Extra Trees

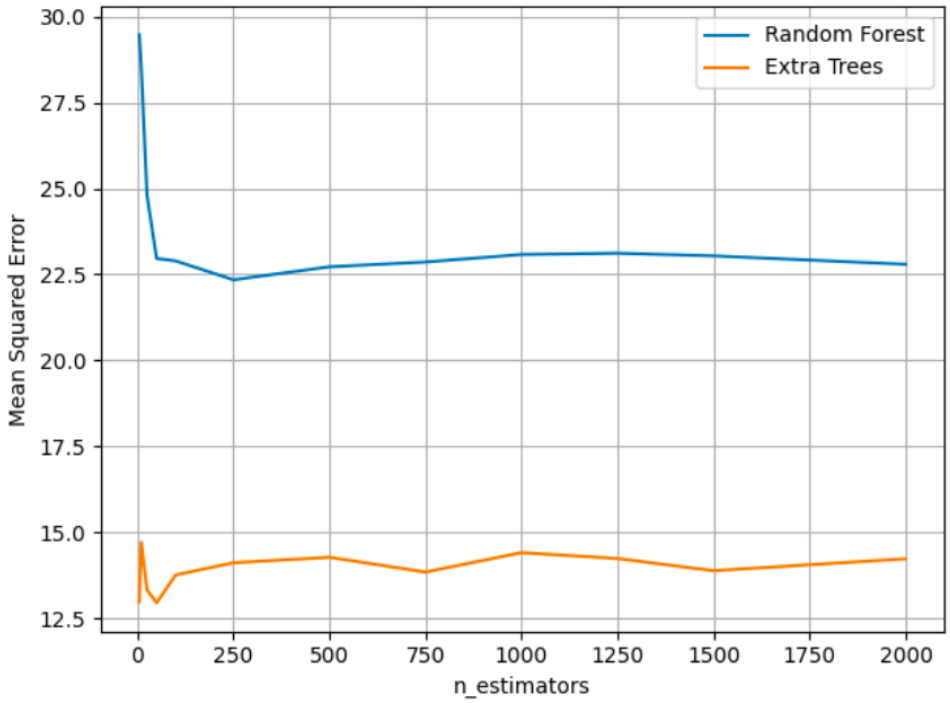

Рисунок 3.31 – Порівняння Mean Squared Error

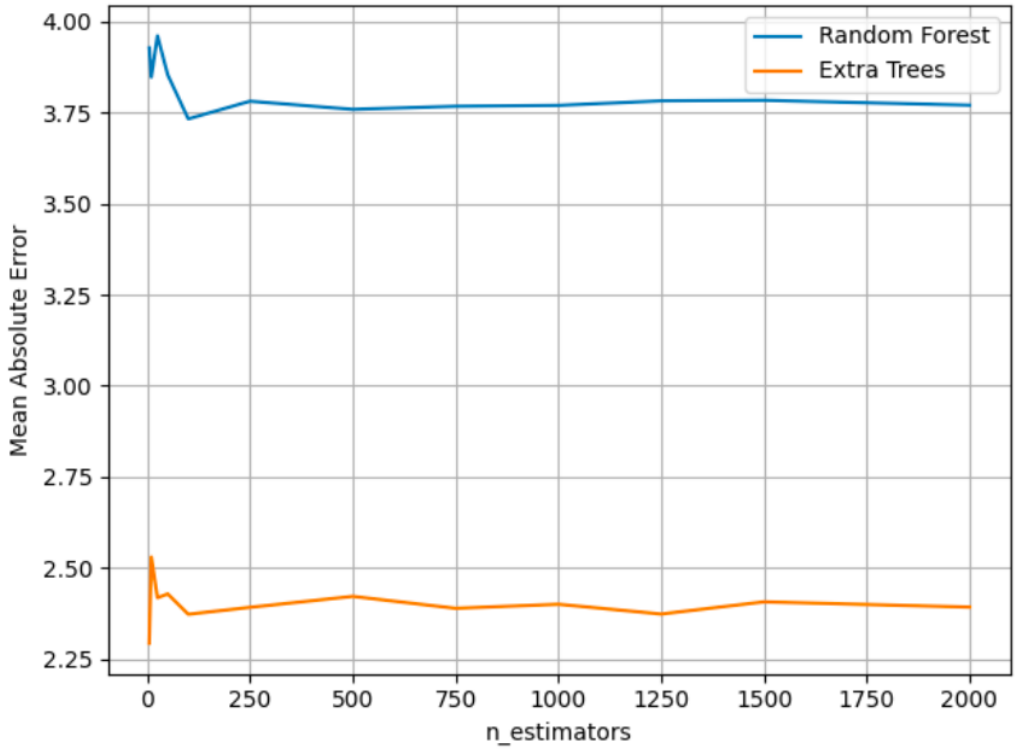

Рисунок 3.32 – Порівняння Mean Absolute Error

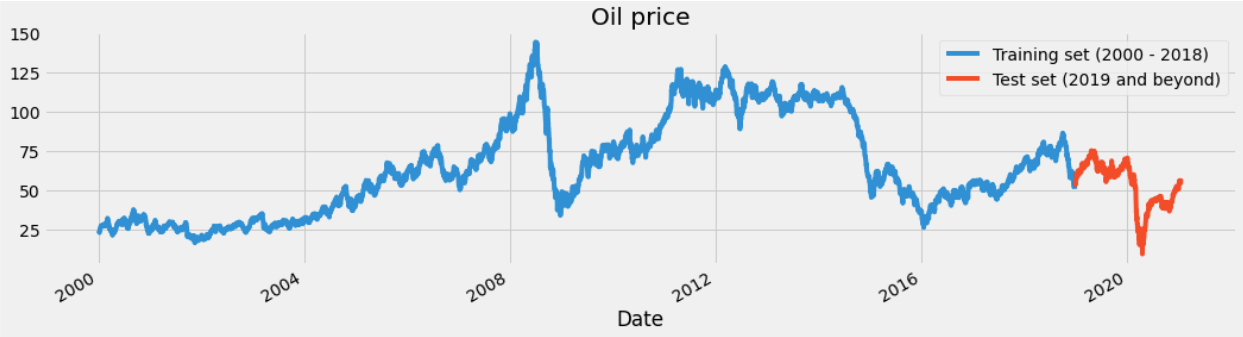

Рисунок 3.33 – Розподіл даних на тренувальні та тестові

```
epoch,loss
0,0.02267362737045236
1,0.008194060441760282
2,0.0074813486038575044
3,0.006162006081113987
4,0.005915606532427291
5,0.005248722180061036
6,0.0047943154158341245
7,0.004201134749493498
8,0.003947253733494895
9,0.003422287640362778
10,0.003730346815657468
11,0.003160880409583049
12,0.0033304524285590557
13,0.00291169582069308
14,0.0028541620670134355
15,0.002775330025226497
16,0.0025610451912624708
17,0.0026100535196151293
18,0.0023016475767012935
19,0.002250734626532139
```

Рисунок 3.34 – Помилка MSE на кожній з епох для LSTM

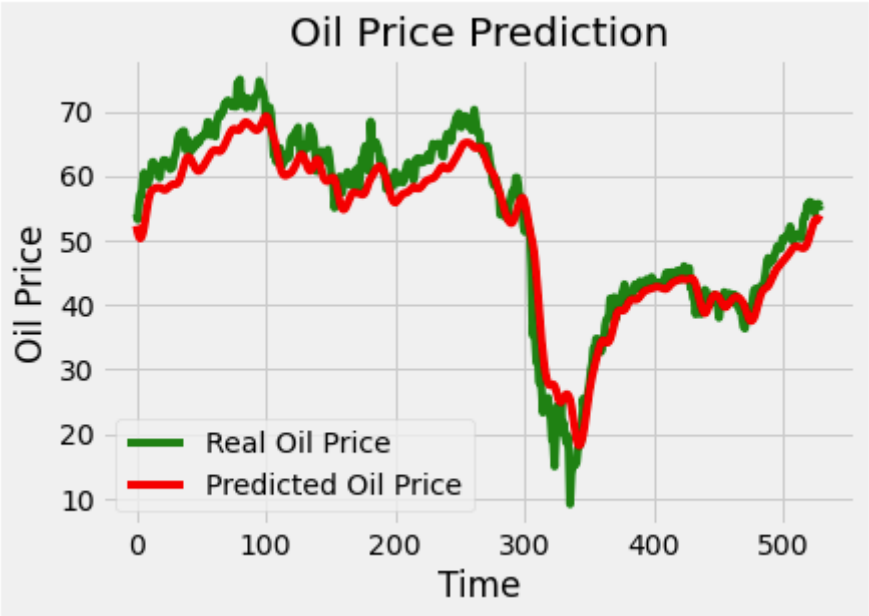

Рисунок 3.35 – Порівняння прогнозу LSTM з реальними цінами

```
The root mean squared error is 3.8779516004580055.
The MSE is 15.038508615494806
The MAE is 3.150442341354658
The R2_Score is 0.9243094294847897
The MAPE is 7.087570667266846
```

Рисунок 3.36 – Оцінка мережі LSTM

```
epoch,accuracy,loss
0,0.0,0.049618569900626575
1,0.00041981527,0.0047635820429474205
2,0.00041981527,0.0019527342293929938
3,0.00041981527,0.0017912850971216387
4,0.00041981527,0.0015861129516357984
5,0.00041981527,0.00153078063085009
6,0.00041981527,0.0015347732410775567
7,0.00041981527,0.0014374539740231141
8,0.00041981527,0.0013612109953824957
9,0.00041981527,0.0014041610604905118
```

Рисунок 3.37 – Помилка MSE та точність на кожній з епох для GRU

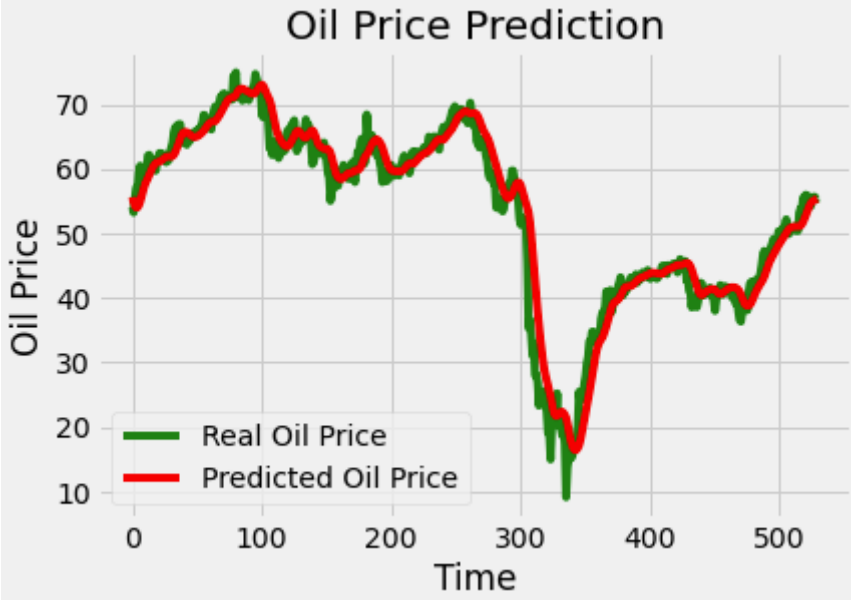

Рисунок 3.38 – Порівняння прогнозу цін GRU з реальними

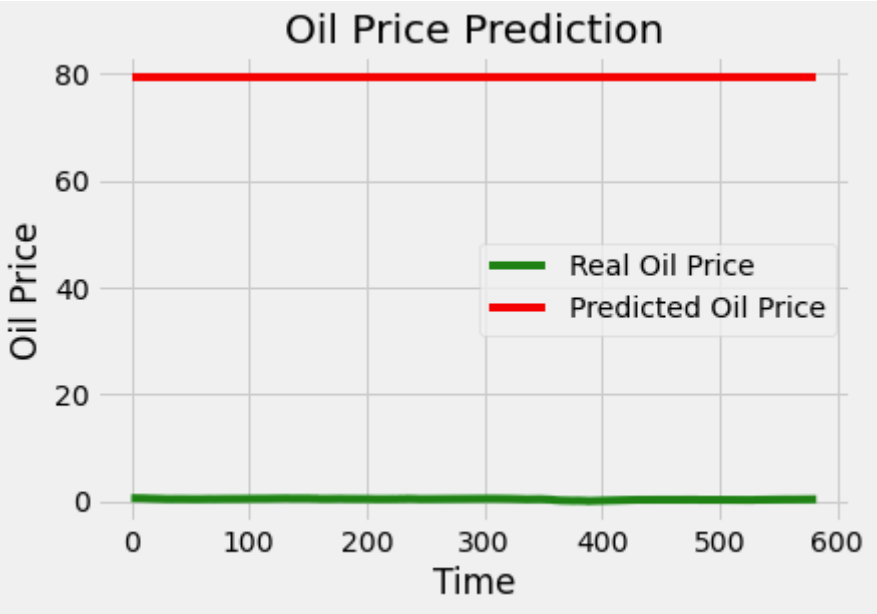

Рисунок 3.39 – Результати алогиртму SVR

3.4 Висновки

В даному розділі показані експерименти при роботі з програмою. На мою думку, реалізовані алгоритми показали себе досить непогано. Результати досить різні і залежать від методу та його параметрів. Використовуючи GridSearch вдалося підібрати вдалі параметри, що позитивно вплинули на результат роботи програмного продукту. Краще всього показали себе такі методи як ExtraTrees та Random Forest, в окремих випадках різниця спрогнозованих результатів з реальними не перевищує 5%. З іншого боку, алгоритм SVR показав погані результати, з чого можна зробити висновок, що він не підходить для розв’язку такого роду задач.

Що до нейронних мереж, то LSTM та GRU показали гарні результати. Точність для обох мереж вийшла більше 95%. Отже можна стверджувати, що ціль роботи, а саме побудова та реалізація нейронних мереж виконана.

На мою думку поставлена задача була виконана, програмний продукт є закінченим та максимально оптимізованим.

РОЗДІЛ 4 РОЗРОБКА СТАРТАП-ПРОЕКТУ

4.1 Вступ та постановка задачі стартап проекту

На сьогодні великої популярності набуває такий вид підприємництва як стартап. Стартап-проект — є комерційним проектом, який знаходиться в стані розробки, або нещодавно вийшов на ринок. Характерною особливістю стартапу, що відрізняє його від малого бізнесу, є оригінальність та інновації, він не може бути копією вже реалізованих ідей. При цьому проект не обов'язково повинен бути масштабного характеру, головне, щоби він був креативним, а його завдання — спрощувати людям будь-які дії в їх повсякденному житті.

Наразі, з появою Інтернету та сучасних технологій, стало простіше заходити на ринок, знаходити інвесторів та споживачів. З'явилося набагато більше можливостей для розвитку свого проекту за кордоном, ніж раніше. Проте розробка стартапу є досить ризикованим завданням. Не всім вдається довести свій стартап-проект до ринкового впровадження. За статистикою успіху досягає лише 10–20 % від усіх стартап-проектів.

Запуск стартапу передбачає цілий ряд обов'язкових дій, у межах яких визначають ринкові перспективи стартапу, графік розробки, принципи організації виробництва, заходи з залучення інвесторів та аналіз ризиків.

Різко постає питання у створенні диверсифікованих інвестиційних портфелів. Одним зі способів інвестування можна розглядати нафту та бензин. Робота дозволяє побудувати відповідний портфель, взявши за основу описану систему прогнозування.

Для розробки стартап проекту та виведення його на ринок необхідно провести детальне дослідження, яке передбачає виконання наведених нижче чотирьох кроків.

Здійснити маркетинговий аналіз стартап-проекту, в рамках якого:

1) розробити опис ідеї проекту, визначити основні напрямки використання товару чи послуги та сформулювати основні відмінності від

товарів/послуг конкурентів;

2) проаналізувати ринкові можливості для його реалізації;

3) розробити стратегію виведення товару на ринок базуючись на аналізі ринкового середовища.

Організація стартап-проекту, яка включає такі кроки:

1) скласти календарний план реалізації та запуску стартап-проекту;

2) визначити плановий обсяг виробництва потенційного товару та н його основі розрахувати потребу у матеріальних ресурсах і персоналі;

3) розрахувати витрати, необхідні для реалізації проекту, та витрати на запуск проекту.

Виконати фінансово-економічний аналіз та оцінити ризики стартап проекту, в межах якого:

1) визначити обсяг інвестиційних витрат;

2) розрахувати основні фінансово-економічні показники проекту (собівартість, ціну продукту/послуги, податковий збір та чистий прибуток) та визначити показники інвестиційної привабливості проекту (рентабельність продажів, період окупності проекту);

3) визначити основні ризики проекту та способи для їх запобігання. Розробити заходи з комерціалізації проекту. Цей етап націлений на пошук фінансування проекту та просування інвестиційної пропозиції.

Для його досягнення необхідно:

1) визначити цільову групу інвесторів та описати їх бізнес інтереси;

2) скласти інвестиційну пропозицію: стислий опис проекту для ознайомлення інвестора із стартап-проектом;

3) визначити основні канали та заходи для просування оферти інвесторам.

4.2 Карта стартап проекту

Стартап проект полягає у створенні системи прийняття рішень для автоматичного проведення торговельних операцій на ринку нафти та палива. Така програма аналізує масиви даних, після чого формує біржовий приказ оптимальний, обгрунтований історичними даними. На основі даної системи пропонується створити інвестиційний фонд у нафто або паливні активи.

В таблиці 4.1 представлено основна інформація проекту, розкрито основну ідею та рішення поставленої проблеми.

Таблиця 4.1 – Інформаційна карта проекту

1. Назва проекту	Система інвестування на ринку нафти та палива на основі методів машинного навчання
2. Коротка анотація	Система дозволяє керувати активами користувача і приймає статистично обгрунтоване рішення спрямована на збільшення об'єму капіталу інвестиційного портфеля користувача.
3.Термін реалізації проекту	6 місяців

Продовження таблиці 4.1

4. Необхідні ресурси	Обладнання – компьютер. Програмне забезпечення, операційна система, антивірусне обладнання. Електрика, газ, водопостачання, Інтернет. Фінансові ресурси – заробітна плата працівникам на 6 місяців роботи, гроші на на оплату комунальних послуг, оренди, реклами тощо. Приміщення з усіма необхідними комунікаціями.
5. Опис проблеми, які вирішує проект	Дана комплексна система дозволяє забезпечити прийняття інвестиційних рішень. Система позбавлена суб'єктивності та забезпечує спосіб прийняття рішень аналогічний до поглядів інвестора. Система працює безпосередньо з акаунтом користувача і не має повного доступу до фінансів, що забезпечує захист даних користувача.
6. Головні цілі та завдання проекту	Основна мета проекту – доведення ефективності обраної системи та розробленого продукту. Додаткові завдання – новий досвід, розробка комплексної системи, робота із реальними даними та створення комерційно успішного продукту.

Продовження таблиці 4.1

7. Очікувані результати	Автоматична система прийняття рішень, здатна працювати на необмеженій кількості користувачів в автономному режимі з допоміжними інструментами побудованих на статистиці, для спрощення аналізу та прийняття рішень.

4.3 Технологічний аудит ідеї проекту

Далі буде виділено основу ідею стартапу та зведено її в наступній таблиці 4.2.

Таблиця 4.2 – Опис ідеї стартап-проекту

Зміст ідеї	**Напрямки застосування**	**Вигоди для користувача**
Дана комплексна система дозволяє побудувати власну інвестиційну стратегію що є легальним шляхом бля будь-якого українця забезпечити стабільні накопичення та їх зростання.	1. Робота в якості компанії по управлінню активами, що надаються користувачем	Система дозволяє повністю автоматизувати дохід користувача, компанія несе відповідальність за кошти користувачів.
	2. Робота в якості управляючої компанії за обмеженим доступом до акаунтів користувачів.	Система дозволяє повністю автоматизувати дохід користувача. Моніторинг дохідності покладається на користувача. Користувач в будь-який момент може вивести кошти або відмовитись від послуг компанії.

Далі було визначено конкурентів на ринку і зроблено порівняльний аналіз програмних продуктів конкурентів, виявлено їх переваги та недоліки. Також було представлено перелік переваг над існуючими програмними

рішеннями (таблиця 4.3, 4.4). Варто відмітити відсутність конкурентів на ринку України тому порівняння було зроблене з американськими конкурентами.

Таблиця 4.3 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	**Техніко Економічні характеристики ідеї**	**(потенційні) товари/концепції конкурентів**				**W (слабка сторона)**	**N (нейтральна)**	**S(сильна)**
		Мій проект	**En Helix**	**Allegro**	**Trafigura**			
1.	Точність прогнозування	Застосування моделі, що безпосередньо оцінює ймовірність його потенційної привабливості	Власний алгоритм	Власний алгоритм	Власний алгоритм			+
2.	Простота управління капіталом інвестора (швидкість зняття грошей інвестором тощо)	Все зводиться до отримання сигналу на укладання угоди та аналізу її доцільності з допомогою інструментів	Процедура складна	Спрощена процедура залучення капіталу	Процедура складна			+

Продовження таблиці 4.3

3.	Ризики невірного прогнозу	Існуют ь, через велику кількість факторів.	Невідом о (комерційна таємниця)	Невідом о (комерційна таємниця)	Невідом о (комерційна таємниця)		+	
4.	Доступність/Зручність	Зручний веб-застосунок, який потребує обчислювальних мінімум ресурсів	Власний інтерфейс	Власний інтерфейс	Власний інтерфейс		+	

Таблиця 4.4 – Технологічна здійсненність ідеї проекту

№ п/п	**Ідея проекту**	**Технології її реалізації**	**Наявність технологій**	**Доступність технологій**
1.	Створення комплексної системи керування активами користувача	Використання мови програмування C#	Потрібні доопрацювання	Не доступні
2.		Використання мови програмування Python	Наявна	Не доступні

Продовження таблиці 4.4

3.		Використання мови програмування Pine	Наявна	Доступні
Обрана технологія реалізації ідеї проекту: мова програмування Pine				

4.4 Аналіз ринкових можливостей запуску стартап-проекту

У таблиці 4.5 зображено попередню характеристику потенційного ринку стартап-проекту.

Таблиця 4.5 – Попередня характеристика потенційного ринку стартап-проекту.

№ п/п	**Показники стану ринку (найменування)**	**Характеристика**
1	Кількість головних гравців, од	0
2	Загальний обсяг продаж	30 млн $
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності по ринку	12%

Після аналізу ринку можна зробити висновок, що він є сприятливим

для створення програмного продукту, оскільки динаміка ринку позитивна, а конкуренти відсутні.

Наступним кроком необхідно охарактеризувати основні групи потенційних користувачів продукту і скласти опис вимог кожної такої групи (таблиця 4.6).

Таблиця 4.6 – Характеристика потенційних клієнтів стартап-проекту.

№ п/п	**Потреба, що формує ринок**	**Цільова аудиторія (цільові сегменти ринку)**	**Відмінності у поведінці різних потенційних цільових груп клієнтів**	**Вимоги споживачів до товару**
1	Необхідність пенсійних накопичень	громадяни України в віці від 18 до 50 років	Незначна кількість початкового капіталу.	Простота використання, простота моніторингу.
2	Збереження бюджету установи та його примноження	Малі підприємства, селищні ради, громадські бюджети	Цікавить короткострокое інвестування.	Великий объем, гибкість капіталу.
3	Довгострокове інвестування.	Великі інвестори	Цікавить передусім відсутність ризику	Високі вимоги до контролю капіталу.

Далі необхідно проаналізувати можливі загрози, що можуть виникнути на етапі виведення продукту на споживацький ринок і перешкодити успішному запуску проекту. Результати представлені у таблиці 4.7.

Таблиця 4.7 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1.	Конкуренція	Доволі незначний рівень виходу на ринок, можливий вихід нових компаній протягом 3 наступних років	Пришвидшити вихід нових програмних продуктів
2.	Збут	Ускладнення збуту через недовіру користувачів	Розміщення додаткових рекламних банерів в інтернеті, розширена наукова база.

Також необхідно розглянути можливі фактори, що навпаки сприятимуть запуску проекту. Результати представлені у таблиці 4.8.

Таблиця 4.8 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Відсутність конкуренції	Бути першим на ринку	Розширення Можливостей продукту, концентрація на якості
2	Створення позитивного іміджу компанії.	Надання послуг на найвищому рівні, забезпечення задоволення клієнтів.	Створення якісної рекламної кампанії, Підтримка клієнтів

Далі необхідно охарактеризувати конкурентне середовище, а саме визначити тип та рівень конкуренції. Результати аналізу наведено у

таблиці 4.9.

Таблиця 4.9 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	**В чому проявляється дана характеристика**	**Вплив на діяльність підприємства**
1.Вказати тип конкуренції - Досконала конкуренція	Багато систем/команд аналітиків	Розробити впізнаваний продукт, якість, що вирізнятиме нас від конкурентів
2. За рівнем конкурентної боротьби: міжнародний	На ринку присутні системи, розроблені за кордоном.	Розширення аудиторії, розширення списку мов, які підтримуються системою
3. За галузевою ознакою - внутрішньогалузева	____	____
4. Конкуренція за видами товарів: товарно-родова	Конкуренція між прогнозами інформаційних систем та команд аналітиків.	Збільшення точності та швидкості відправки біржових приказів
5.За характером Конкурентних переваг: Нецінова	Різні способи прогнозування дають різну точність	Розробка кращих(точних) алгоритмів
6. За інтенсивністю: марочна	Впізнаваний бренд надає великих переваг	Велику увагу приділити розвитку бренду

Далі необхідно виконати детальний аналіз конкуренції за моделлю 5 сил конкуренції Майкла Портера, яка використовується для розуміння структури галузі, аналізу її привабливості з точки зору отримання прибутку,

оцінки конкуренції і розробки стратегії бізнесу. Результати аналізу зведено в таблицю 4.10.

Таблиця 4.10 – Аналіз конкуренції в галузі за М. Портером.

Складові аналізу	**Прямі конкуренти в галузі**	**Потенційні конкуренти**	**Постачальник и**	**Клієнти**	**Товари-замінники**
	Інші комплексні системи	**Гнучкі ціни, розмір капіталовкладень**	**Фактори сили постачальників**	**Контроль якості, система інформації**	**Ціна, лояльність ь споживачів**
Висновки:	Інтенсивна конкуренція можлива в майбутньому	Є як можливості входження на ринок, так і нові потенційні конкуренти	Постачальники відсутні	Клієнти не диктують ь умови роботи на ринку	Товари Замінники відсутні

Результати аналізу конкурентного середовища підтверджують, що на ринку сприятлива ситуація для створення і запуску даного стартап-проекту. Ґрунтуючись на проведеному аналізі конкуренції (таблиця 4.10), а також враховуючи характеристики ідеї стартап-проекту (таблиця 4.5), характеристики потенційних клієнтів і їх вимоги до продукту (таблиця 4.6) та фактори ринкового середовища (таблиці 4.7 та 4.8) було сформульовано та обґрунтовано перелік факторів конкурентоспроможності. Аналіз оформлено

в таблицю 4.11.

Таблиця 4.11 – Обґрунтування факторів конкурентоспроможності

Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
Низька конкуренція	В нашій країні на час розробки стартапу не було виявлено конкурентів
Доступність програмного продукту	Розроблений продукт є загальнодоступним і кросплатформенним. Для доступу необхідне підключення до мережі Інтернет.
Зручність використання	Інтерфейс надається власне біржею користувача, який вже знайомий користувачу.

Після проведення аналізу можна виділити сильні та слабкі (які потребують вдосконалення) сторони продукту (таблиця 4.12).

Таблиця 4.12 – Порівняльний аналіз сильних та слабких сторін системи

№ п/п	**Фактор конкурентоспроможності**	**Бали 1-20**	**Рейтинг товарів-конкурентів**						
			–3	*–2*	*–1*	*0*	*+1*	*+2*	*+3*
1	Низька конкуренція	20				+			
2	Доступність програмного продукту	20				+			
3	Зручність використання	18		+					

Завершальним етапом аналізу ринкових можливостей для запуску проекту є складання SWOT-аналізу. Він дозволяє оцінити можливості та загрози бізнесу, а також сильні і слабкі сторони продукту. Результати

наведені у таблиці 4.13.

Таблиця 4.13 – SWOT- аналіз стартап-проекту

Сильні сторони: відсутність конкурентів; дружній інтерфейс; не вимагає спеціальних навичок/знань для використання.	Слабкі сторони: немає налагодженої клієнтської бази.
Можливості: розширення списку мов для аналізу; додавання нових предметних областей; інтеграція з іншими програмними системами.	Загрози: поява конкурентів; збут .

На основі SWOT-аналізу було спроектовано альтернативну ринкову поведінку для інтеграції стартап-проекту на ринок та приблизний час реалізації системного комплексу, з урахуванням потенційних проектів, що можуть бути виведені на ринок (таблиця 4.14).

Таблиця 4.14 – Альтернативи ринкового впровадження стартап проекту

№ п/п	**Альтернатива (орієнтовний комплекс заходів) ринкової поведінки**	**Ймовірність отримання ресурсів**	**Строки реалізації**
1	Швидкий вихід на ринок із «сирим» продуктом, можливі проблеми із точністю прогнозу та універсальністю	10%	3 місяці
2	Поступовий вихід з готовим продуктом. Висока якість та конкурентоспроможна ціна.	25%	6 місяців

Отже, в результаті детального аналізу ринкового та конкурентного середовища, факторів загроз та можливостей, сильних та слабких сторін продукту можна зробити висновок, що на ринку склалися сприятливі умови для впровадження товару і, що даний товар відповідає вимогам користувачів.

4.5 Розроблення ринкової стратегії проекту

Розроблення ринкової стратегії перш за все передбачає визначення стратегії охоплення ринку. Для цього було охарактеризовано цільові групи

потенційних споживачів (таблиця 4.15).

Таблиця 4.15 – Вибір цільових груп потенційних споживачів

№ п/п	**Опис профілю цільової групи потенційних клієнтів**	**Готовність споживачів сприйняти продукт**	**Орієнтовний попит в межах цільової групи**	**Інтенсивність конкуренції в сегменті**	**Простота входу у сегмент**
1	Громадяни, що прагнуть створити накопичення	Низька готовність	5%	Низька	Середня
2	Малі та середні підприємства	Висока	15%	Низька	Висока
3	Великі компанії з власними інвесторами	Висока	30%	Висока	Висока
Які цільові групи обрано:1,2					

Стратегією охоплення ринку було обрано недиференційований (масовий) маркетинг — компанія концентрує свої зусилля одразу на всіх сегментах споживачів. Для роботи в обраних сегментах ринку сформовано базову стратегію розвитку (таблиці 4.16, 4.17, 4.18).

Таблиця 4.16 – Визначення базової стратегії розвитку

п/п	**Альтернатива розвитку проекту**	**Стратегія охоплення ринку**	**Ключові конкурентоспроможні позиції відповідно до обраної альтернативи**	**Базова стратегія розвитку***
	1 та 2	Стратегія недиференційований маркетингу	Висока універсальність, багатогалузевість, висока якість, ціна.	Масовий маркетинг

Таблиця 4.17 – Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку?	**Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?**	**Чи буде компанія копіювати основні характеристики товару конкурента, і які?**	**Стратегія конкурентної поведінки***
Так	Так	Ні	Стратегія виклику лідера

Таблиця 4.18 – Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	**Базова стратегія розвитку**	**Ключові конкурентоспроможні позиції власного стартап-проекту**	**Вибір асоціацій, які мають формувати комплексну позицію власного проекту (три ключових)**
Легкість у використанні, висока точність аналізу	Позиціювання як продукту, що використовує передові технології для аналізу	Якість прогнозу, універсальність.	По іміджу Позиціювання на низькій ціні Легкість у використанні

4.6 Висновки до 4 розділу

В даному розділі було повністю виконано перший етап розроблення стартап проекту, а саме, виконано маркетинговий аналіз стартап проекту. За допомогою нього можна сказати, що існує можливість ринкової комерціалізації проекту, адже на ринку відсутні конкуренти, до того ж рентабельність роботи є досить високою. З огляду на потенційну групу клієнтів, а саме, громадян, малий бізнес, та інноваційність технології є великі перспективи впровадження даного програмного забезпечення у вигляді колючого продукту фонду заощаджень.

Отже, робота може бути розглянута як стартап що спрямований на стратегії голубих океанів (не зайнятих ринків). Про це свідчить відсутність конкурентів та величезна потенційна аудиторія. На цій основі вважаю необхідним продовжувати досліджувати ринок нафти та палива з позиції інвестиційної привабливості та прогнозування.

ВИСНОВКИ

В даній роботи були побудовані різні моделі для прогнозування цін на нафту. Були досліджені методи регресійного аналізу, а саме дерева рішень. Також були побудовані моделі на основі нейронних мереж. Була опрацьована література, розглянуті історичні відомості про формування ринку нафти. На основі теоретичних відомостей був обраний метод регресійного аналізу , адже він якнайкраще підходить для вирішення задачі магістерської дисертації. Було обрано метод дерев рішень у його різних варіаціях. Та два види нейронних мереж. Побудовані моделі були порівняні з реальними показниками на ринку. У процесі виконання роботи, було досліджено обрані моделі, виявлено залежність точності моделі від її гіперпараметрів. Завдяки обраним моделям було створено програмний продукт який дає точні результати при тестуванні, в окремих випадках похибка між результатами програмного продукту та реальними не перевищує декількох відсотків. Після аналізу результатів програмного продукту були обрані дві моделі, що дають найкращий результат у порівнянні з іншими.

ПЕРЕЛІК ПОСИЛАНЬ

1.Reng L., History of Oil, 14.02.2020. URL: https://www.ektinteractive.com/history-of-oil/ (Last accessed: 02.04.2020).

2.Buisseret D., et. al., Historic Texas from the Air. Chicago: The University of Chicago Press, 2009. 163p.

3.Lamar K. International Directory of Company Histories. 1.11.2020. URL: https://books.google.com.ua/books/about/International_Directory_of_Company_Histo.html?id=FQi7AAAAIAAJ&redir_esc=y (Last accessed 10.10.2021)

4.Lioudis N., OPEC's Influence on Global Oil Prices, 21.04.2020. URL: https://www.investopedia.com/ask/answers/060415/how-much-influence-does-opec-have-global-price-oil.asp (Last accessed: 17.04.2020).

5.Dr. Edmund M. Daukoru, Oil market stability: the role of OPEC 8 September, 14.02.2006. URL: https://www.opec.org/opec_web/en/press_room/994.htm (Last acessed: 23.04.2020).

6.Amadeo K., What Affects Oil Prices? Three Critical Factors, 23.04.2020. URL: https://www.thebalance.com/how-are-oil-prices-determined-3305650 (Last accessed: 02.05.2020).

7.Bajpai P., Top Factors That Affect the Price of Oil 21.04.2020. URL: https://www.investopedia.com/articles/investing/072515/top-factors-reports-affect-price-oil.asp (Last accessed 27.04.2020).

8.Kosakowski P., What determines oil prices?, 21.04.2020. URL: https://www.investopedia.com/articles/economics/08/determining-oil-prices.asp (Last accessed: 09.05.2020).

9.Chen E., He J., Crude Oil Price Prediction with Decision Tree Based Regression Approach, 5.01.2019. URL: https://scholarworks.lib.csusb.edu/jitim/ (Last accessed: 05.05.2020).

10.Foley B., What is Regression Analysis and Why Should I Use It?, 14.02.2018. URL: https://www.surveygizmo.com/resources/blog/regression-analysis/ (Last accessed: 29.04.2020).

11.Bhalla D., 15 Types of regression in data science, 25.03.2018. URL: https://www.listendata.com/2018/03/regression-analysis.html (Last accessed: 14.05.2020).

12.Pant A., Introduction to Linear Regression and Polynomial Regression, 13.01.2019. URL: https://towardsdatascience.com/introduction-to-linear-regression-and-polynomial-regression-f8adc96f31cb (Last accessed: 12.05.2020).

13.Верморель Ж., Квантильная регрессия, 03.02.2012. URL: https://www.lokad.com/ru/%D0%BA%D0%B2%D0%B0%D0%BD%D1%82%D0%B8%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F-%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%81%D0%B8%D1%8F-(%D0%B2%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5-%D1%80%D1%8F%D0%B4%D1%8B)-%D0%BE%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5 (дата звернення: 14.05.2020).

14.Stephaniem K., Ridge Regression: Simple Definition, 29.07.2017. URL: https://www.statisticshowto.com/ridge-regression/ (Last accessed: 14.05.2020).

15.Padmanabha A., Ridge Regression, 18.05.2016. URL: https://brilliant.org/wiki/ridge-regression/ (Last accessed: 16.05.2020).

16. Stephaniem K., Lasso Regression: Simple Definition, 24.09.2015. URL: https://www.statisticshowto.com/lasso-regression/ (Last accessed: 16.05.2016).

17. Sunil R., 7 Regression Techniques you should know!, 14.08.2015. URL: https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/ (Last accessed: 17.05.2020).

18.I. Ruczinski, C. Kooperberg, M. Leblanc, Logic Regression. Baltimore: Johns Hopkins University Pres, 2005. 623p.

19.Шахиди А., Деревья решений: общие принципы, 4.12.2019. URL: https://loginom.ru/blog/decision-tree-p1 (дата звернення: 18.05.2020).

20.Gupta P., Decision Trees in Machine Learning, 17.05.2017. URL: https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052 (Last accessed: 19.05.2020).

21. Geron O. Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow. Boston: Oreilly, 2018. 239p.

22.Nagpal A., Decision Tree Ensembles- Bagging and Boosting, 17.10.2017. URL: https://towardsdatascience.com/decision-tree-ensembles-bagging-and-boosting-266a8ba60fd9 (Last accessed: 19.05.2020).

23.Lutins E., Ensemble Methods in Machine Learning: What are They and Why Use Them?, 02.08.2017. URL: https://towardsdatascience.com/ensemble-methods-in-machine-learning-what-are-they-and-why-use-them-68ec3f9fef5f (Last accessed: 20.05.2020).

24.Russel M., Decision Tree Ensemble Methods, 15.08.2017. URL: https://medium.com/@rnbrown/decision-tree-ensemble-methods-6a89181b7083 (Last accessed: 20.05.2020)

25.Бутко М.П. та інші, Системний підхід і моделювання в наукових дослідженнях. Київ: Центр учбової літератури, 2014. 345с.

26.Lean Yu, Mengyao Ma. A memory-trait-driven decomposition–reconstruction–ensemble learning paradigm for oil price forecasting. *Apply Soft Computing Journal*. Vol. 2, No. 1. 2021. P.1-7.

27.Shuang Gao, Yalin Lei. A new approach for crude oil price prediction based on stream learning. *Geoscience Frontiers*. Vol. 13, No. 1.2017. P.183-187.

28.Ganiyu Adewale Busaria, Dong Hoon Lim. Crude oil price prediction: A comparison between AdaBoost-LSTM and AdaBoost-GRU for improving forecasting performance. 04.09.2019. URL: https://www.sciencedirect.com/science/article/abs/pii/S009813542100291X (Last accessed: 6.10.2021).

29.Ashwin Raj, Unlocking the True Power of Support Vector Regression, 3.10.2020. URL: https://towardsdatascience.com/unlocking-the-true-power-of-support-vector-regression-847fd123a4a0 (дата звернення 11.11.2021).

30.Enes Zvornicanin, Grey Wolf Optimizaion Algrotihm, 13.10.2021. URL: https://www.baeldung.com/cs/grey-wolf-optimization (дата звернення 29.10.2021).

31.Mattew Korn, Oil Industry, 21.08.2021. URL: https://www.history.com/topics/industrial-revolution/oil-industry (Last accessed: 10.09.2021).

32.Nick Lioudis, OPEC's Influence on Global Oil Prices, 09.09.2021. URL: https://www.investopedia.com/ask/answers/060415/how-much-influence-does-opec-have-global-price-oil.asp (Last accessed: 10.10.2021).

33.Paul Kosakowski, What Determines Oil Prices?, 14.09.202. URL : https://www.investopedia.com/articles/economics/08/determining-oil-prices.asp (Last accessed: 20.09.2021).

ДОДАТОК А ЛІСТИНГ ПРОГРАМИ

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-


import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split, ShuffleSplit

from sklearn.metrics import r2_score
from sklearn.metrics import explained_variance_score
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import median_absolute_error
from sklearn.metrics import mean_squared_error

from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.ensemble import GradientBoostingRegressor, AdaBoostRegressor
# Setup paths for the input data
path = "/kaggle/input/pet-pri-spt-s1-m/PET_PRI_SPT_S1_M.xls"
# Path to save the test results of predicted  prices
test_path = "testResults.csv"
```

```
# Array of Dataframes input
data = []

for i in range(2):
    if (i != 0):
        data.append(pd.read_excel(path, sheet_name=i, skiprows=[0, 1]))

# result storing average value and price change per year
resultdf = []

# result storing average value and price change per year
resultdf_monthly = []

# Types of Data Available
# types = ['Crude Oil', 'Conventional Gasoline', 'RBOB Regular Gasoline', 'Heating Oil', 'Diesel Fuel',
#          'Kerosene Type Jet Fuel', 'Propane']
types = ['Çrude Oil', 'Conventional Gasoline']

#### This method applies the initial analysis of Finding Monthly and Yearly Change in Price
def analyze(data2, ind):
    monthList = []
    yearList = []
    monthDiff = []
    nextData = []
    row_iterator = data2.iterrows()

    str_cols = data2.columns[data2.dtypes == object]
    data2[str_cols] = data2[str_cols].fillna('.')
```

```
data2.fillna(0, inplace=True)


for index, row in row_iterator:
    current_date = row['Date']
    monthList.append(current_date.month)
    yearList.append(current_date.year)
    if (index == len(data2) - 1):
        nextData.append(data2.iloc[index][1])
        monthDiff.append(0)
    else:
        nextData.append(data2.iloc[index + 1][1])
        monthDiff.append(data2.iloc[index + 1][1] - row[1])


data2['Month'] = monthList
data2['Year'] = yearList
data2['Monthly Change in Price'] = monthDiff


# generate map from keys
map_month = dict.fromkeys(data2.Month.unique())


for month in data2.Month.unique():
    df1 = data2.loc[data2['Month'] == month]
    print('*******')
    print('For month', month)
    print(df1.iloc[:, 1])
    mean_mnth = np.asarray(df1.iloc[:, 1]).mean()
    print('mean:', mean_mnth)
    map_month[month] = mean_mnth
```

```
yearls_m = list(map_month.keys())
print(map_month)

meanls_m = list(map_month.values())

res = pd.DataFrame(np.column_stack([yearls_m, meanls_m]),
                   columns=['Month', 'Mean Price of ' + str(types[ind])])

resultdf_monthly.append(res)

# generate map from keys
map_ = dict.fromkeys(data2.Year.unique())

for yr in data2.Year.unique():
    df = data2.loc[data2['Year'] == yr]
    mean = np.asarray(df.iloc[:, 1]).mean()
    map_[yr] = mean

yearls = list(map_.keys())

meanls = list(map_.values())

result = pd.DataFrame(np.column_stack([yearls, meanls]),
                      columns=['Year', 'Mean Price of ' + str(types[ind])])

yearDiff = []
for index, row in result.iterrows():
```

```
        if (index == len(result) - 1):
            yearDiff.append(0)
        else:
            yearDiff.append(result.iloc[index + 1][1] - row[1])

    result['Yearly Change in Price'] = yearDiff

    resultdf.append(result)


### Method where different machine learning models are trained and comparitive study is done
def trainModels(file, imp_attr):
    # split into train and test
    train, test = train_test_split(file, test_size=0.2)

    # Removing the target/predictor from the train data
    targ = train[list(target_col)]

    ### Random Forest Model
    rand_forest_model = RandomForestRegressor(n_estimators=1000, max_features=2,
oob_score=True, random_state=115)
    rand_forest_model.fit(train[list(imp_attr)], targ.values.ravel())
    print('SCORERANDOMFOREST', rand_forest_model.score(test[list(imp_attr)],
test[list(target_col)]))
    ###EXtraTreeRedression Model
    extra_tree_model = ExtraTreesRegressor(criterion='mse', max_depth=None,
max_features='auto', min_samples_split=2)
    extra_tree_model.fit(train[list(imp_attr)], targ.values.ravel())
    print('SCORE_EXTRATREE', extra_tree_model.score(test[list(imp_attr)],
test[list(target_col)]))
```

```
    ### Gradient Boosting

    gr_boosting_model = GradientBoostingRegressor(n_estimators=500, learning_rate=0.1,
subsample=1, max_features=2, loss='ls')

    gr_boosting_model.fit(train[list(imp_attr)], targ.values.ravel())

    print('SCOREGRADIENTBOOSTING', gr_boosting_model.score(test[list(imp_attr)],
test[list(target_col)]))

    ###AdaBoosting

    a_boosting_model = AdaBoostRegressor(n_estimators=500, learning_rate=0.1,loss='square')

    a_boosting_model.fit(train[list(imp_attr)], targ.values.ravel())

    print('SCOREADABOOSTING', a_boosting_model.score(test[list(imp_attr)],
test[list(target_col)]))

    ### Decision Tree Model

    decision_tree_model = DecisionTreeRegressor(max_depth=4)

    decision_tree_model.fit(train[list(imp_attr)], targ)

    print('SCOREDECISIONTREE', decision_tree_model.score(test[list(imp_attr)],
test[list(target_col)]))


    ### Linear Regression Model

    linear_model = LinearRegression()

    linear_model.fit(train[list(imp_attr)], targ)

    print('SCORELINEAR', linear_model.score(test[list(imp_attr)], test[list(target_col)]))

    return a_boosting_model, rand_forest_model, extra_tree_model, gr_boosting_model,
decision_tree_model, linear_model, test




#### Method which tests the given model and Prints out the statistics regarding each of them

def testNEvalModels(test, ab_model, rf_model, et_model, gd_model,  dt_model, lm_model,
imp_attr):

    print('\n Evaluation Staistics:')

    ### Evaluating Random Forest
```

```
print("\n ***Random Forest Regressor***")
# Evaluation metric: r square
r2 = r2_score(test[list(target_col)], rf_model.predict(test[list(imp_attr)]))
print("R-Square Value:", r2)

# extracting the test target values and convert to float
true_vals = test[list(target_col)].values
true_vals_flt = true_vals.astype(np.float)

prediction = rf_model.predict(test[list(imp_attr)])

# reshaping the array is required to convert it into numpy array
aa = prediction.reshape(-1, 1)

mean_squared_error(true_vals_flt, prediction)

mse = np.mean((true_vals_flt - aa) ** 2)
print("Mean Squared Error", mse)

print("Explained Variance Score", explained_variance_score(true_vals_flt, prediction))
print("Mean Absolute Error", mean_absolute_error(true_vals_flt, prediction))
print("Median Absolute Error", median_absolute_error(true_vals_flt, prediction))
### Evaluating Extra Tree Model
print("\n ***Extra Tree Model***")
# Evaluation metric: r square
r2_et = r2_score(test[list(target_col)], et_model.predict(test[list(imp_attr)]))
print("R-Square Value:", r2_et)
```

```
# extracting the test target values and convert to float
true_vals = test[list(target_col)].values
true_vals_flt = true_vals.astype(np.float)

prediction = et_model.predict(test[list(imp_attr)])

# reshaping the array is required to convert it into numpy array
aa = prediction.reshape(-1, 1)

mean_squared_error(true_vals_flt, prediction)

mse = np.mean((true_vals_flt - aa) ** 2)
print("Mean Squared Error", mse)

print("Explained Variance Score", explained_variance_score(true_vals_flt, prediction))
print("Mean Absolute Error", mean_absolute_error(true_vals_flt, prediction))
print("Median Absolute Error", median_absolute_error(true_vals_flt, prediction))
### Evaluating Gradient Method
print("\n ***Gradient Tree Boosting***")
# Evaluation metric: r square
r2_gd = r2_score(test[list(target_col)], gd_model.predict(test[list(imp_attr)]))
print("R-Square Value:", r2_gd)

# extracting the test target values and convert to float
true_vals = test[list(target_col)].values
true_vals_flt = true_vals.astype(np.float)

prediction = gd_model.predict(test[list(imp_attr)])
```

```
###ADABOOSTING NEW
# reshaping the array is required to convert it into numpy array
aa = prediction.reshape(-1, 1)

mean_squared_error(true_vals_flt, prediction)

mse = np.mean((true_vals_flt - aa) ** 2)
print("Mean Squared Error", mse)

print("Explained Variance Score", explained_variance_score(true_vals_flt, prediction))
print("Mean Absolute Error", mean_absolute_error(true_vals_flt, prediction))
print("Median Absolute Error", median_absolute_error(true_vals_flt, prediction))
### Evaluating Gradient Method
print("\n ***Ada Bossting***")
# Evaluation metric: r square
r2_ab = r2_score(test[list(target_col)], ab_model.predict(test[list(imp_attr)]))
print("R-Square Value:", r2_ab)

# extracting the test target values and convert to float
true_vals = test[list(target_col)].values
true_vals_flt = true_vals.astype(np.float)

prediction = ab_model.predict(test[list(imp_attr)])

# reshaping the array is required to convert it into numpy array
aa = prediction.reshape(-1, 1)

mean_squared_error(true_vals_flt, prediction)
```

```
mse = np.mean((true_vals_flt - aa) ** 2)
print("Mean Squared Error", mse)

print("Explained Variance Score", explained_variance_score(true_vals_flt, prediction))
print("Mean Absolute Error", mean_absolute_error(true_vals_flt, prediction))
print("Median Absolute Error", median_absolute_error(true_vals_flt, prediction))
### Evaluating Decision tree
print("\n ***Decision Tree Regressor***")
y_2 = dt_model.predict(test[list(imp_attr)])
r2_dt = r2_score(true_vals_flt, y_2)
print("R-Square Value:", r2_dt)

print("Mean Squared Error", mean_squared_error(true_vals_flt, y_2))

print("Explained Variance Score", explained_variance_score(true_vals_flt, y_2))
print("Mean Absolute Error", mean_absolute_error(true_vals_flt, y_2))
print("Median Absolute Error", median_absolute_error(true_vals_flt, y_2))

### Evaluating Linear Model
print("\n ***Linear Regression Model***")
pred_lm = lm_model.predict(test[list(imp_attr)])

r2_lm = r2_score(true_vals_flt, pred_lm)

print("R-Square Value:", r2_lm)

print("Mean Squared Error", mean_squared_error(true_vals_flt, pred_lm))
```

```
    print("Explained Variance Score", explained_variance_score(true_vals_flt, pred_lm))
    print("Mean Absolute Error", mean_absolute_error(true_vals_flt, pred_lm))
    print("Median Absolute Error", median_absolute_error(true_vals_flt, pred_lm))



#### The chosen model is then applied to predict the future crude oil prices
def applyModel(model, test):
    # Apply selected model to test data
    pred = model.predict(test[list(imp_attr)])

    # save predicted into target column in test
    test_dtm['Predicted Price'] = pred

    # save df to file
    test_dtm.to_csv(test_path)



# Create a Excel Writer Object to Write Yearly analysis Price Change
writer = pd.ExcelWriter('output_y.xlsx')
writer_m = pd.ExcelWriter('output_m.xlsx')

for i in range(len(data)):
    analyze(data[i], i)

# Storing resultant yearly analysis
for i in range(len(resultdf)):
    resultdf[i].to_excel(writer, 'data' + str(i))
```

```
writer.save()

# Storing resultant yearly analysis
for i in range(len(resultdf_monthly)):
    resultdf_monthly[i].to_excel(writer_m, 'data' + str(i))
writer_m.save()

##### Code to predict the next 6 months Prices

# Crude oil data being taken as input
train_data = data[0]
target_col = ["Cushing, OK WTI Spot Price FOB (Dollars per Barrel)"]

# features selected on which the model would be based on
imp_attr = ['Month', 'Year']

# Training different models and choosing the best one for prediction
# using_gridsearchcv(train_data, imp_attr)
ab, rf, et, gd, dt, lm, test = trainModels(train_data, imp_attr)
testNEvalModels(test, ab, rf, et, gd, dt, lm, imp_attr)

# create test data for next 12 months
test_dtm = {}
test_dtm['Month'] = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
test_dtm['Year'] = [2016, 2016, 2016, 2016, 2016, 2016, 2016, 2016, 2016, 2016, 2016, 2016]
test_dtm = pd.DataFrame.from_dict(test_dtm)

# As per the Evaluation Statistics we get to know that Random Forest
```

```
# is performing better than other considered models, so we apply rf to test data
applyModel(et, test_dtm)

print("Yearly Change in Price is stored in file named: output_y.xlsx")
print("Monthly Change in Price is stored in file named: output_m.xlsx")
print("The resultant Predicted Values are stored in a File named: testResults.csv")
def using_gridsearchcv(file,imp_attr):

  train, test = train_test_split(file, test_size=0.2)
  targ = train[list(target_col)]
  algoritmy = {
    'linear_regression' : {
      'model': LinearRegression(),
      'parameters': {
        'normalize': [True, False]
      }
    },
    'extra_tree' : {
      'model': ExtraTreesRegressor(),
      'parameters': {
        'n_estimators': [i for i in range(50, 1500, 50)],
        'min_samples_split': [2, 3, 4, 5],
        'criterion': ['mse','mae'],
        'max_features': ['auto', 'sqrt', 'log2'],
        'max_depth': [None, 5, 8]
      }
    },
    'gradient_boosting': {
      'model': GradientBoostingRegressor(),
```

```
        'parameters': {
            'learning_rate': [0.1, 0.2, 0.3],
            ## 'subsample': [1, 2, 3],
            'max_features': ['auto', 'sqrt', 'log2'],
            'loss': ['ls', 'lad']
        }
    },
    'decision_tree': {
        'model': DecisionTreeRegressor(),
        'parameters': {
            'criterion': ['mse', 'friedman_mse'],
            'splitter': ['best', 'random']
        },
    },
    'random_forest': {
        'model': RandomForestRegressor(),
        'parameters': {
            'n_estimators': [i for i in range(50, 1500, 50)],
            'max_depth': [5, 8, None],
            'criterion': ['mse']
        }
    }
}
scores = []
cv = ShuffleSplit(n_splits=5, test_size=0.2, random_state=0)
for model_name, config in algoritmy.items():
    gs = GridSearchCV(config['model'], config['parameters'], cv=cv, return_train_score=False)
    print("Fit...", config['model'])
    gs.fit(train[list(imp_attr)], targ.values.ravel())
```

```
        scores.append({
            'model': model_name,
            'best_score': gs.best_score_,
            'best_parameters': gs.best_params_
        })

    return pd.DataFrame(scores, columns=['model', 'best_score', 'best_parameters'])

A = using_gridsearchcv(train_data, imp_attr)
print(A)
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout, GRU, Bidirectional
from keras.optimizers import SGD
import math
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error, r2_score
from keras.callbacks import *
from keras.losses import MAPE
from numpy import mean
def plot_predictions(test,predicted):
    plt.plot(test, color='green',label='Real Oil Price')
    plt.plot(predicted, color='red',label='Predicted Oil Price')
    plt.title('Oil Price Prediction')
    plt.xlabel('Time')
```

```
    plt.ylabel('Oil Price')
    plt.legend()
    plt.figure(figsize=(16,4))
    plt.show()


def return_rmse(test,predicted):
    rmse = math.sqrt(mean_squared_error(test, predicted))
    print("The root mean squared error is {}.".format(rmse))
    print("The MSE is {}".format(mean_squared_error(test, predicted)))
    print("The MAE is {}".format(mean_absolute_error(test, predicted)))
    print("The R2_Score is {}".format(r2_score(test, predicted)))
    print("The MAPE is {}".format(mean(MAPE(test, predicted))))
dataset = pd.read_csv('../input/brentoilprices123/BrentOilPrices.csv', index_col='Date',
parse_dates=['Date'])
dataset.head()
# Checking for missing values
training_set = dataset['2000':'2018'].iloc[:,0:1].values
test_set = dataset['2019':].iloc[:,0:1].values
print(training_set.shape)
print(test_set.shape)
# We have chosen 'High' attribute for prices. Let's see what it looks like
dataset["Price"]['2000':'2018'].plot(figsize=(16,4),legend=True)
dataset["Price"]['2019':].plot(figsize=(16,4),legend=True)
plt.legend(['Training set (2000 - 2018)','Test set (2019 and beyond)'])
plt.title('Oil price')
plt.show()
# Scaling the training set
sc = MinMaxScaler(feature_range=(0,1))
training_set_scaled = sc.fit_transform(training_set)
```

```
# print(training_set_scaled)

print(training_set.shape)

print(test_set.shape)

window = 60

# So for each element of training set, we have 60 previous training set elements

X_train = []

y_train = []

for i in range(window,training_set.shape[0]):

   X_train.append(training_set_scaled[i-window:i,0])

   y_train.append(training_set_scaled[i,0])

X_train, y_train = np.array(X_train), np.array(y_train)

# Reshaping X_train for efficient modelling

X_train = np.reshape(X_train, (X_train.shape[0],X_train.shape[1],1))

X_train.shape

# Now to get the test set ready in a similar way as the training set.

# The following has been done so first 60 entires of test set have 60 previous values which is impossible to get unless we take the whole

# 'High' attribute data for processing

dataset_total = pd.concat((dataset["Price"]['2000':'2018'],dataset["Price"]['2019':]),axis=0)

inputs = dataset_total[len(dataset_total)-len(test_set) - window:].values

# print(inputs.shape)

inputs = inputs.reshape(-1,1)

print(inputs.shape)


# print(inputs.shape)

inputs  = sc.transform(inputs)

# print(inputs)

# print(inputs.shape)

print(test_set.shape)
```

```
# print(test_set)

# Preparing X_test and predicting the prices

X_test = []

y_test = []

for i in range(window,test_set.shape[0]+window):

   X_test.append(inputs[i-window:i,0])

   y_test.append(inputs[i,0])

X_test = np.array(X_test)


X_test = np.reshape(X_test, (X_test.shape[0],X_test.shape[1],1))

# The LSTM architecture

regressor = Sequential()

# First LSTM layer with Dropout regularisation

regressor.add(LSTM(units=50, return_sequences=True, input_shape=(X_train.shape[1],1)))

regressor.add(Dropout(0.2))

# Second LSTM layer

regressor.add(LSTM(units=50, return_sequences=True))

regressor.add(Dropout(0.2))

# Third LSTM layer

regressor.add(LSTM(units=50, return_sequences=True))

regressor.add(Dropout(0.2))

# Fourth LSTM layer

regressor.add(LSTM(units=50))

regressor.add(Dropout(0.2))

# The output layer

regressor.add(Dense(units=1))


# Compiling the RNN

regressor.compile(optimizer='rmsprop',loss='mean_squared_error')
```

```
# Fitting to the training set

csv_logger = CSVLogger('trainingLSTM.log')

regressor.fit(X_train,y_train,epochs=20,batch_size=128,callbacks=[csv_logger])

predicted_stock_price = regressor.predict(X_test)

predicted_stock_price = sc.inverse_transform(predicted_stock_price)

# Visualizing the results for LSTM

plot_predictions(test_set,predicted_stock_price)

# Evaluating our model

return_rmse(test_set,predicted_stock_price)

# The GRU architecture

regressorGRU = Sequential()

# First GRU layer with Dropout regularisation

regressorGRU.add(GRU(units=120, return_sequences=True, input_shape=(X_train.shape[1],1),
activation='tanh'))

regressorGRU.add(Dropout(0.2))

# Second GRU layer

regressorGRU.add(GRU(units=120, return_sequences=True, input_shape=(X_train.shape[1],1),
activation='tanh'))

regressorGRU.add(Dropout(0.2))

# Third GRU layer

regressorGRU.add(GRU(units=120, return_sequences=True, input_shape=(X_train.shape[1],1),
activation='tanh'))

regressorGRU.add(Dropout(0.2))

# Fourth GRU layer

regressorGRU.add(GRU(units=120, activation='tanh'))

regressorGRU.add(Dropout(0.2))

# The output layer

regressorGRU.add(Dense(units=1))

# Compiling the RNN
```

```
regressorGRU.compile(optimizer=SGD(lr=0.01, decay=1e-7, momentum=0.9,
nesterov=False),loss='mean_squared_error',metrics=['accuracy'])

# Fitting to the training set

csv_logger = CSVLogger('trainingGRU.log')

regressorGRU.fit(X_train,y_train,epochs=10,batch_size=150,callbacks=[csv_logger])

predicted_stock_price = regressorGRU.predict(X_test)

predicted_stock_price = sc.inverse_transform(predicted_stock_price)


# Visualizing the results for GRU

plot_predictions(test_set,predicted_stock_price)


# Evaluating our model

return_rmse(test_set,predicted_stock_price)

# Checking for missing values

training_set = dataset['2000':'2018'].iloc[:,0:1].values

test_set = dataset['2019':].iloc[:,0:1].values

print(training_set.shape)

print(test_set.shape)

X_train2 = []

y_train2 = []

for i in range(window,training_set.shape[0]):

   X_train2.append(training_set[i-window:i,0])

   y_train2.append(training_set[i,0])

X_train2, y_train2 = np.array(X_train2), np.array(y_train2)

print(X_train2.shape)

print(y_train2.shape)

# Now to get the test set ready in a similar way as the training set.

# The following has been done so first 60 entires of test set have 60 previous values which is impossible
to get unless we take the whole
```

```
# 'High' attribute data for processing
dataset_total2 = pd.concat((dataset["Price"]['2000':'2018'],dataset["Price"]['2019':]),axis=0)
inputs2 = dataset_total2[len(dataset_total)-len(test_set) - window:].values
# print(inputs.shape)
inputs2 = inputs.reshape(-1,1)
print(inputs2.shape)


# print(inputs.shape)
# inputs  = sc.transform(inputs)
# print(inputs.shape)
print(test_set.shape)
# Preparing X_test and predicting the prices
X_test2 = []
y_test2 = []
for i in range(window,inputs.shape[0]):
   X_test2.append(inputs2[i-window:i,0])
   y_test2.append(inputs2[i,0])
X_test2, y_test2 = np.array(X_test2), np.array(y_test2)
print(X_test2.shape)
print(y_test2.shape)
import numpy as np
from sklearn import svm
from sklearn.svm import SVR
import sklearn.model_selection
import numpy.random as rd
import matplotlib.pyplot as plt
from sklearn.model_selection import KFold
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import cross_val_score
```

```
from sklearn.model_selection import train_test_split
import warnings, pandas as pd,numpy as np, time, math, configparser,random



## 1. GWO optimization algorithm
def sanitized_gwo(X_train2,X_test2,y_train2,y_test2,SearchAgents_no,T,dim,lb,ub):
    Alpha_position=[0,0] # Initialize the position of Alpha Wolf
    Beta_position=[0,0]
    Delta_position=[0,0]

    Alpha_score = float("inf") # Initialize the value of Alpha Wolf's objective function
    Beta_score = float("inf")
    Delta_score = float("inf")

    Positions = np.dot(rd.rand(SearchAgents_no,dim),(ub-lb))+lb # initialize the first search position

    Convergence_curve=np.zeros((1,T))# initialization fusion curve

    iterations = []
    accuracy = []

    # Main Loop
    t = 0
    while t < T:

        # Iterate over each wolf
        for i in range(0,(Positions.shape[0])):
            #If the search position exceeds the search space, you need to return to the search space
            for j in range(0,(Positions.shape[1])):
```

```
            Flag4ub=Positions[i,j]>ub

            Flag4lb=Positions[i,j]<lb

            #If the wolf's position is between the maximum and minimum, the position does not need to be adjusted, if it exceeds the maximum, the maximum returns to the maximum value boundary



            if Flag4ub:

                Positions[i,j] = ub

            if Flag4lb:

                Positions[i,j] = lb

        '''SVM MODEL TRAINING - FOR CLASSIFICATION PROBLEM DATASET'''

        #rbf_svm = svm.SVC(kernel = 'rbf', C = Positions[i][0], gamma = Positions[i][1]).fit(X_train, y_train) #svm

        #cv_accuracies = cross_val_score(rbf_svm,X_test,y_test,cv =3,scoring = 'accuracy')



        '''SVR MODEL TRAINING - FOR REGRESSION PROBLEM DATASET'''

        rbf_regressor = svm.SVR(kernel = 'rbf', C = Positions[i][0], gamma = Positions[i][1]).fit(X_train2, y_train2)  #svm

        cv_accuracies = cross_val_score(rbf_regressor,X_test2,y_test2,cv =3,scoring = 'neg_mean_squared_error') # Taking negated value of MSE




        #To minimize the error rate

        accuracies = cv_accuracies.mean()

        fitness_value = (1 - accuracies)*100

        if fitness_value<Alpha_score: # If the objective function value is less than the objective function value of Alpha Wolf

            Alpha_score=fitness_value # Then update the target function value of Alpha Wolf to the optimal target function value

            Alpha_position=Positions[i] # At the same time update the position of the Alpha wolf to the optimal position

        if fitness_value>Alpha_score and fitness_value<Beta_score:  # If the objective function value is between the objective function value of Alpha Wolf and Beta Wolf
```

```
        Beta_score=fitness_value # Then update the target function value of Beta Wolf to the optimal target function value

        Beta_position=Positions[i]

      if fitness_value>Alpha_score and fitness_value>Beta_score and fitness_value<Delta_score: #If the target function value is between the target function value of Beta Wolf and Delta Wolf

        Delta_score=fitness_value  # Then update the target function value of Delta Wolf to the optimal target function value

        Delta_position=Positions[i]



    a=2-t*(2/T)


    # Iterate over each wolf
    for i in range(0,(Positions.shape[0])):
       #Traverse through each dimension
       for j in range(0,(Positions.shape[1])):
          #Surround prey, location update
          r1=rd.random(1)#Generate a random number between 0 ~ 1
          r2=rd.random(1)
          A1=2*a*r1-a # calculation factor A
          #C1=2*r2 # calculation factor C
          C1 = 0.5 + (0.5*math.exp(-j/500)) + (1.4*(math.sin(j)/30)) # Time varying Acceleration constant


          #Alphawolf location update


          D_alpha=abs(C1*Alpha_position[j]-Positions[i,j])
          X1=Alpha_position[j]-A1*D_alpha


          r1=rd.random(1)
          r2=rd.random(1)
```

```
            A2=2*a*r1-a

            #C2=2*r2

            C2  = 1 + (1.4*(1 - math.exp(-j/500)) ) + (1.4*(math.sin(j)/30)) #Difference Mean based Perturbation time varying parameter


            # Beta wolf location update

            D_beta=abs(C2*Beta_position[j]-Positions[i,j])

            X2=Beta_position[j]-A2*D_beta

            r1=rd.random(1)

            r2=rd.random(1)


            A3=2*a*r1-a

            #C3=2*r2

            C3=(1/(1+ math.exp(-0.0001*j/T) )) + ((0.5 - 2.5) * ((j/T) **2)) #sigmoid-based acceleration coefficient


            # Delta Wolf Location Update

            D_delta=abs(C3*Delta_position[j]-Positions[i,j])

            X3=Delta_position[j]-A3*D_delta


            # Location update

            Positions[i,j]=(X1+X2+X3)/3



        t = t + 1

        iterations.append(t)

        accuracy.append((100-Alpha_score)/100)

        print('----------------Count of iterations----------------' + str(t))
```

```
        print(Positions)
        print('C and gamma:' + str(Alpha_position))
        print('accuracy:' + str((100-Alpha_score)/100))

    best_C=Alpha_position[0]
    best_gamma=Alpha_position[1]

    return best_C,best_gamma,iterations,accuracy

# Plot Convergence Curve
def plot(iterations,accuracy):
    plt.plot(iterations,accuracy)
    plt.xlabel('Count of iterations',size = 20)
    plt.ylabel('Accuracy',size = 20)
    plt.title('Sanitized GWO-SVM parameter optimization (SGWO_SVM)')
    plt.show()

if __name__ == '__main__':


    print('----------------2. Parameter setting-----------')
    SearchAgents_no=20 #Number of Wolfs
    T=2 # maximum number of iterations
    dim=2 #Need to optimize two variables - Cost and Gamma
    lb=0.01 #lower bound Parameter
    ub=10 #upper bound Parameter

    print('----------------3.LARGE-----------------')
```

```
    best_C,best_gamma,iterations,accuracy =
sanitized_gwo(X_train2,X_test2,y_train2,y_test2,SearchAgents_no,T,dim,lb,ub)


    print('----------------4. The result shows----------------')
    print("The best C is " + str(best_C))
    print("The best gamma is " + str(best_gamma))
    plot(iterations,accuracy)


#Apply Optimal Parameters to SVR
svr_regressor= SVR(kernel='rbf', C = best_C,gamma=best_gamma )
svr_regressor.fit(X_train2,y_train2)
y_pred = svr_regressor.predict(X_test2)
# APPLYING K-FOLD CROSS VALIDATION on RF model
accuracies = cross_val_score(svr_regressor, X = X_train2, y = y_train2, cv = 10)
accuracy_mean= accuracies.mean()
accuracies.std()*100


mse = mean_squared_error(y_test2, y_pred)
rmse = np.sqrt(mse)
r2=r2_score(y_test2, y_pred)
nrmse=rmse/(y_test2.max() - y_test2.min())
print("SVR RESULTS - C AND GAMMA PARAMETERS OPTIMIZED BY GRAY WOLF OPTIMIZATION")
print("RMSE =", rmse)
print("MSE =", mse)
print("Normalized RMSE=",nrmse)
print("R Square =",r2)
print("K-fold accuracy mean",accuracy_mean)
# Visualizing the results for SVR
plot_predictions(y_test2,y_pred)
```

```
X_train2.shape,X_test2.shape,y_train2.shape,y_test2.shape
```