МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра інтелектуальних технологій

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему:

Система прогнозування курсу криптовалют

Галузь знань **12 «Інформаційні технології»**
Спеціальність **122 «Комп'ютерні науки»**
Освітня програма «**Комп'ютерні науки»**
Освітній рівень: бакалавр

Виконав: студент 4 курсу, групи КН- 41

Міхалкін В.В.
(прізвище та ініціали)

Керівник Гайна Г.А.
(прізвище та ініціали)
к.т.н. професор
(науковий ступінь, звання)

Випускна кваліфікаційна робота бакалавра допущена до захисту
рішенням кафедри *інтелектуальних технологій*
Протокол № 11 від 06.06.2022 р.
зав. кафедри_____ доц. Іларіонов О.Є.

Київ - 2022

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інтелектуальних технологій

Спеціальність 122 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ

Завідувач кафедри
інтелектуальних технологій

Іларіонов О.Є.

_____

"___"_____2022 р.

ЗАВДАННЯ

НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

__Міхалкіну_____ Вячеслав Вікторовичу_____

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи)

Розробка інформаційної системи прогнозування курсу криптовалют

затверджена протоколом засідання кафедри від «23» грудня 2021 р. № 4

2. Термін здачі студентом закінченого проекту (роботи) 29 травня 2022 року

3. Вихідні дані до проекту (роботи)
Інформаційна система прогнозування курсу криптовалют. Прогноз криптовалюти біткойн на 3 дні вперед,історичний курс

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)

Аналітичний огляд теми роботи, аналіз існуючих та вибір математичного забезпечення, вибір стеку технологій та розробка програмного забезпечення системи прогнозування.

5. Перелік презентаційного матеріалу (з точним зазначенням обов'язкових презентацій)

Тема та мета роботи, що пропонується, етапи досліджень, результат та аналіз отриманих даних прогнозування

6. Консультанти з випускної кваліфікаційної роботи із зазначенням розділів випускної кваліфікаційної роботи, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

1	Гайна Георгій Анатолійович		
2	Гайна Георгій Анатолійович		
3	Гайна Георгій Анатолійович		

7. Дата видачі завдання 15 лютого 2022 року

Керівник _____ / Гайна Г.А. /
(підпис) (ПІБ)

Завдання прийняв до виконання _____ ____ / Міхалкін В.В. /
(підпис) (ПІБ)

КАЛЕНДАРНИЙ ПЛАН

Студент-дипломник _____ _____ / Міхалкін В.В. /
(підпис) (ПІБ)

Керівник випускної кваліфікаційної роботи _____ / Гайна Г.А. /
(підпис) (ПІБ)

Анотація

В даній роботі вивчаються математичні методи прогнозування курсу криптовалют, застосування цих методів у програмному забезпеченні для отримання прогнозування. Також аналізуються отримані дані та проводяться розрахунки для оцінки точності.

***Ключові слова*:** криптовалюти, модель агресивного ковзного середнього ARIMA,паритет купівельної спроможності, часові ряди, принцип відносної економічної стабільності, побудова економетричної моделі.

Abstract

This paper studies the mathematical methods of cryptocurrency exchange rate forecasting, the application of these methods in software for forecasting. The obtained data are also analyzed and calculations are performed to assess the **accuracy.**

Key words: cryptocurrencies, ARIMA aggressive moving average model, purchasing power parity, time series, principle of relative economic stability, construction of econometric model.

ЗМІСТ

ВСТУП

Глобалізація, зростання фінансових ринків, тотальна комп'ютеризація та розвиток інформаційних технологій призвели до появи величезної кількості додаткових інституцій, фінансових інструментів та нових форм взаємодії між людьми в нашому суспільстві. Один із таких інститутів є електронні гроші або криптовалюти.

Криптовалюти дозволяють обмінювати ресурси в електронному вигляді та виключають будь-яку долю регулюючої сторони, яка б підтримувала інфраструктуру (наприклад, банк) і при цьому дозволяла безпечно обмінюватися ресурсами. Роль бухгалтерських операцій передано з централізованих інститутів у мережу автономних комп'ютерів, утворюючи децентралізовану структуру, що працює поза впливом будь-якої управлінської установи. Найчастіше курс криптовалют не є стабільним як щодо національних валют, так і між один одним. На нього впливає спит та пропозиція споживачів, інвесторів, трейдерів, а також економіко-політична ситуація, інформаційне поле та інші події, що характерні не лише для сфери криптовалюту, а й для економіки в цілому. В результаті цих факторів наближаються значні колівання курсу і його прогноз стає дуже складним завданням. Проте, розв'язання цього завдання має значення для підприємницької діяльності. Іноді це дає колосальні можливості для заробітку чи мінімізації фінансових втрат. Передбачати поведінку валютних курсів прагнути багато: банки, брокери, державні та приватні інвестиційні організації та прості трейдери. Таким чином, аналіз та розробка методів прогнозування курсу криптовалют є актуальним завданням та потребує проведення дослідження.

Об'єктом дослідження є курс криптовалюти.

Предметом дослідження є програма прогнозування курсу криптовалюти.

Результатом даного дослідження має стати програмне забезпечення, що дозволяє передбачити курс криптовалюти на прикладі криптовалюти Bitcoin.

1. АНАЛІТИЧНИЙ ОГЛЯД

1.1 Огляд алгоритмів прогнозування курсів криптовалют

Для початку розглянемо основні методи, які можуть бути використані для передбачення курсів валют у цілому. До таких методів відносять:

- Теорія паритету купівельної спроможності
- Аналіз часових рядів
- принцип відносної економічної стабільності
- Побудова економетричної моделі
- Теорія паритету купівельної спроможності

Сучасний розвиток світової економіки зробило найважливішими дослідження, що ґрунтуються на порівнянні потенціалів окремих держав та різних регіонів світу. Оскільки макроекономічні показники будь-якої країни виражені у національній валюті, при дослідженні виникає проблема їх перетворення на однакові грошові одиниці.

Одним із найважливіших економічних показників держави є його валовий внутрішній продукт. Оптимальний варіант для підрахунку світового валового внутрішнього продукту виходить з коефіцієнтів порівняння купівельної спроможності валют. Значення цих коефіцієнтів визначаються ставленням цін купівельних кошиків різних країн. На цьому засновано теорію паритету купівельної спроможності (ПКС). Іншими словами, розраховується кількість валюти однієї країни (виражене у грошових одиницях іншої держави), необхідної для купівлі одного й того самого товару (послуги) в обох країнах.

Основою паритету купівельної спроможності є закон однієї ціни, з якого випливає, що вартість однакових товарів (послуг) в різних державах має бути однаковою. Як приклад можна навести ціну на шафу в Китаї та Австралії. З урахуванням обмінного курсу та не зважаючи на транспортні витрати, ціна на цей виріб має бути практично однаковою.

Відповідно до теорії паритету купівельної спроможності, курс валют варіюється і компенсує зростання цін за інфляції.

Аналіз тимчасових рядів

Даний метод є комплексом математико-статистичних способів аналізу. Він служить для виявлення структури ряду послідовних значень, які характеризують зміна у часі певних показників. Метою є їхнє прогнозування. Тимчасові ряди можуть набувати різних форм. Прийнято розрізняти кілька класів моделей часових рядів.

Серед них модель агресивного ковзного середнього (ARMA) користується найбільшою популярністю. Саме її використовують для складання прогнозу динаміки руху валютних пар. Вирахування проводяться з використанням спеціальної комп'ютерної програми. Параметри годинного ряду вводяться заздалегідь. Результат – цінова модель будь-якого валютного активу.

Принцип відносної економічної стабільності

Використовуючи цей спосіб, базуються на динаміці економічного зростання різних країн. Це багато в чому визначає рух валютного курсу. Досвідчені економісти вміло використовують такий інструмент прогнозування ситуації на фінансовому ринку. Логічне обґрунтування принципу — наявність стабільного економічного зростання країни завжди залучає іноземні інвестиції. Для розвитку бізнесу в такій «здоровій» державі потрібна купівля національної валюти, що незмінно призводить до зростання на неї та її подальшого зміцнення.

Принцип відносної економічної стабільності допомагає визначати інтенсивність інвестиційних потоків. Бізнесменів завжди цікавлять високі відсоткові ставки. Вони дають можливість отримати максимальну вигоду з вкладених коштів. При цьому знову підвищується спрос на національну валюту і як наслідок відбувається її зміцнення. Низький рівень процентних ставок не залучає іноземний капітал, а стимулює внутрішнє кредитування.

За всі свої переваги принцип відносної економічної стабільності не дає можливості прогнозувати розмір курсу валюти. Тому його використовують у комплексі з іншими методами.

Побудова економетричної моделі

Економетрична модель, у разі, полягає у реальній залежності курсу обміну валюти від низки значних економічних факторів. Ці фактори інвестор обирає сам.

Провівши детальний аналіз, він має визначити:

- які саме показники слід задіяти в розрахунках;
- який вид повинен мати залежність між відібраними показниками.

Наприклад, для побудови економетричної моделі прогнозування курсу американського долара в нашій країні дослідники можуть використати такі параметри, як індекс Доу-Джонса, ціни на нафту, вартість золота. На думку експертів, ці фактори прирівнюються до індикаторів стану балансу та ситуації на ринку Forex. Само собою зрозуміло, що в цій моделі не враховані інші параметри, які можна виділити у зв'язку із тісною зв'язкою Forex з іншими ринками.

Вищеописаний спосіб є досить складним, але за успішно складеною економетричною моделлю можна легко проводити розрахунки певних валютних активів, підставляючи у формулу вихідні дані.

Слід зазначити, що прогнозування курсу криптовалют цін принципово відрізняється від прогнозування інших фінансових активів, зокрема звичайних (фіатних) валют, що мають велику кількість теоретичних та емпіричних досліджень, які були зосереджені на вивченні їхньої моделі динаміки.

На сьогоднішній день у вчених немає єдиної думки щодо фундаментальної цінності криптовалют. Домінує теза про те, що обмінний курс більшості криптовалют визначається лише співвідношенням споживання та пропозиції [1-3].

Наприклад, емпіричний аналіз трьох найбільш капіталізованих криптовалют (Біткойн, Ріппл, та Ethereum) не виявив статичного зв'язку між їх прибутковістю складністю їх добутку.

У той же час макроекономічні фактори, які зазвичай визначають динаміку валютних, фондових і товарних ринків, не мають істотного впливу на динаміку ринку криптовалют.

У проведених дослідженнях [4] було виявлено, що вплив Фондового ринку США та світового фондового ринку на волатильність біткойнів не був значним.

Крім того, дослідження, опубликовані в [5-7], показують, що цінова динаміка криптовалют описується формулою класичних моделей цінових міхурів Сорнетта та їх модифікацій.

Ряд недавніх досліджень ринку криптовалют показують, що, на відміну від інших фінансових активів, ціна криптовалюти залежить від низки специфічних факторів, які формують їх попит, наприклад, кількість трендів Google, пошукові запити, кількість постів у соціальних мережах та інших засобах масової інформації [8]. Ці дослідження обґрунтували можливість використання нетипових факторів як предиктори.

Всі ці фактори ускладнюють розвиток економетричних моделей динаміки цін криптовалюти.

Останнім часом непараметричні методи, засновані на Machine Learning та глибокому навчанні, набули популярності для завдань аналізу та прогнозування фінансово-економічної тимчасової послідовності.

Моделі машинного навчання засновані на спеціальних штучних мережах, що дозволяють вирішити проблему передбачення та класифікація з використанням навчання послідовності даних. Ефективність таких моделей залежить від швидкості навчання та ступеня універсальності апроксимуючих функцій.

Ці моделі поєднують у собі арсенал потужних методів, таких як штучна нейронна мережа (ІНС), Методи опорних векторів (SVM), рішення та Дерево класифікації (DT, CT), нечітка логіка, генетика алгоритми (ГА), лінійні та нелінійні статистичні моделі, і т.д.

Приклади їх ефективного використання в прогнозуванні курсів валют та фондових індексів наведені в роботі [9]. У кількох інших дослідженнях [10-12] повідомлялося про результати прогнозування курсу біткойна з використанням класичної моделі ARIMA та використання різних методів машинного навчання, наприклад випадковий ліс (RF), логістична регресія (LR), лінійний дискримінантний аналіз (LDA) та довготривала короткочасна пам'ять (LSTM).

Результати цього аналізу показали, що моделі, які покладалися на навчання, виявилися більш придатними для прогнозування цін на криптовалюті, так і їх волатильності.

У дослідженні [13] провели порівняльний аналіз можливості прогнозування моделей ARIMA з рекурентними нейронними мережами (RNN) для таких криптовалют, як DASH, Ethereum (ETH), Litecoin (LTC), Siacoin (SC), Stellar (STR), NEM (XEM) , Монеро (XMR) та Ріпл (XRP). Результати показали, що нейронні мережі мають гірші властивості прогнозування, ніж моделі ARIMA.

Таким чином, другий підхід є більш придатним для прогнозування цінових тенденцій у криптовалюті. Розглянемо цей метод докладніше.

Авторегресійне інтегроване ковзне середнє, або ARIMA, є модель статистичного аналізу, яка використовує дані часових рядів або для кращого розуміння набору даних, або для прогнозування майбутніх тенденцій.

Авторагресійна інтегрована модель ковзного середнього є формою регресійного аналізу, яка вимірює вплив однієї залежної змінної по відношенню до інших змінних, що змінюються. Мета моделі полягає у тому, щоб передбачити майбутні рухи цінних паперів або фінансового ринку, досліджуючи різницю між значеннями у ряду, а не через фактичні значення.

Модель ARIMA можна зрозуміти, описав кожен із її компонентів таким чином:

- Авторегресія (AR): відноситься до моделі, яка показує змінну змінну, яка регресує за своїми власними запізнілими або попередніми значеннями.

- Інтегрований (I): представляє різницю необроблених спостережень, що дозволяє тимчасовим рядам стати стаціонарними (тобто значення даних замінюються різницею між значеннями даних та попередніми значеннями).
- Ковзне середнє (MA): включає залежність між спостереженням та залишковою помилкою з моделі ковзного середнього, що застосовується до спостережень, що запізнюються.

Кожен компонент ARIMA функціонує як параметр зі стандартним записом. Для моделей ARIMA стандартним записом будуть p, d і q, де цілі значення замінюють параметри, щоб вказати тип моделі ARIMA, що використовується. Параметри можуть бути визначені як:

- p: кількість спостережень, що запізнюються у моделі; також відомий як порядок відставання.
- d: кількість разів, коли необроблені спостереження різняться; також відомий як ступінь розходження.
- q: розмір вікна ковзної середньої; також відомий як порядок ковзної середньої.

Наприклад, у моделі лінійної регресії враховуються кількість та тип компонентів. Значення 0, яке можна використовувати як параметр, означає, що конкретний компонент не повинен використовуватись у моделі. Таким чином, модель ARIMA може бути побудована як для виконання функції моделі ARMA, так і для простих моделей AR, I або MA.

Моделі ARIMA засновані на припущенні, що попередні значення мають деякий залишковий ефект на поточні чи майбутні значення. Наприклад, інвестор, який використовує модель ARIMA для прогнозування цін на акції, виходитиме з того, що нові покупці та продавці цих акцій залежать від недавніх ринкових угод при прийнятті рішення про те, скільки запропонувати або прийняти за цінну папір.

Хоча це припущення справедливе у багатьох випадках, це завжди так. Наприклад, в роки, що передували фінансовій кризі 2008 року, більшість

інвесторів не знали про ризики, пов'язані з великими портфелями цінних паперів з іпотечним покриттям (MBS), що належать багатьом фінансовим фірмам.

У той час у інвестора, який використовує авторегресійну модель для прогнозування динаміки фінансових акцій США, були вагомі підстави передбачати тенденцію, що триває, до стабільних або зростаючих цін на акції в цьому секторі. Проте, як тільки стало відомо, що багатьом фінансовим установам загрожує неминучий крах, інвестори раптово стали менше турбуватися про нещодавні ціни на ці акції і набагато більше стурбовані їхньою схильністю до ризику. Таким чином, ринок швидко переоцінив акції фінансових компаній до набагато нижчого рівня, крок, який повністю спростував би авторегресійну модель.

1.2 Огляд існуючих систем прогнозування курсу

Розглянемо кілька програм, які дозволяють провести аналіз курсу криптовалют та побудувати його прогноз.

SAS Forecast Studio.

SAS Forecast Studio – це програма, призначена для ускорення процесу прогнозування за рахунок автоматизації (рис. 1.1). Програмне забезпечення забезпечує автоматичний вибір моделей тимчасових рядів для використання в прогнозуванні даних із тимчасовими мітками. SAS Forecast Studio постачається як частина SAS Forecast Server корпоративного рішення.

За допомогою цієї програми можна виконувати такі завдання:

- Створювати прогнози автоматично, використовуючи моделі, що постачаються із SAS Forecast Studio. Прогнози можна створювати за допомогою переліку вибору моделі.
- Створювати власні моделі прогнозування чи комбінувати згенеровані моделі.

- Виконувати ієрархічне прогнозування «згори вниз», «знизу вгору» і «від середини до виходу».
- Проводити визуальний аналіз та діагностику даних часових рядів
- Перевизначати прогнози та вказувати, як слід вирішувати конфлікти.
- Додавати події у модель або весь проект.
- Аналізувати вплив на прогнози різних майбутніх значень для вхідного ряду
- Виконувати моделювання ковзання, щоб визначити найкращу модель для даних.
- Створювати звіти, щоб поділитись результатами прогнозування з іншими людьми.

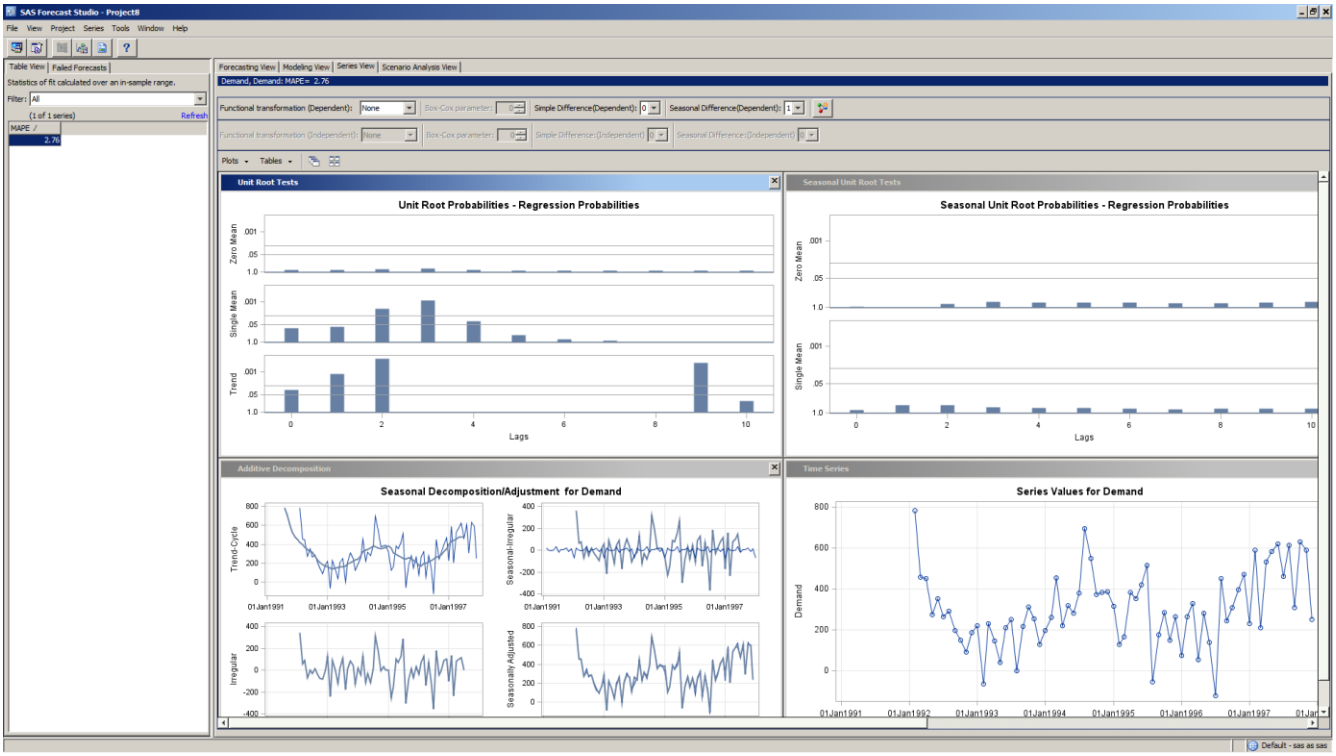


Рисунок 1.1 – Інтерфейс програми SAS Forecast Studio

Forecast Pro.

Автоматичний алгоритм «кращого вибору» Forecast Pro дозволяє створювати точні прогнози для багатьох параметрів та враховувати сезонні закономірності,

тенденції, бізнес-цикли, рекламну діяльність та багато іншого. Forecast Pro надає параметри моделювання на основі меню та повного набору діагностичних інструментів (рисунок 1.2).

Елемент управління табличними даними, подібний до Excel, дозволяє легко налаштовувати відсотки, збільшення або просто вводити нові значення. Коригування прогнозу можна вносити на будь-якому рівні ієрархії та всі зміни автоматично узгоджуються. Зручні поля для коментарів полегшують документування змін, а звіти про перевизначення забезпечують повну прозорість.

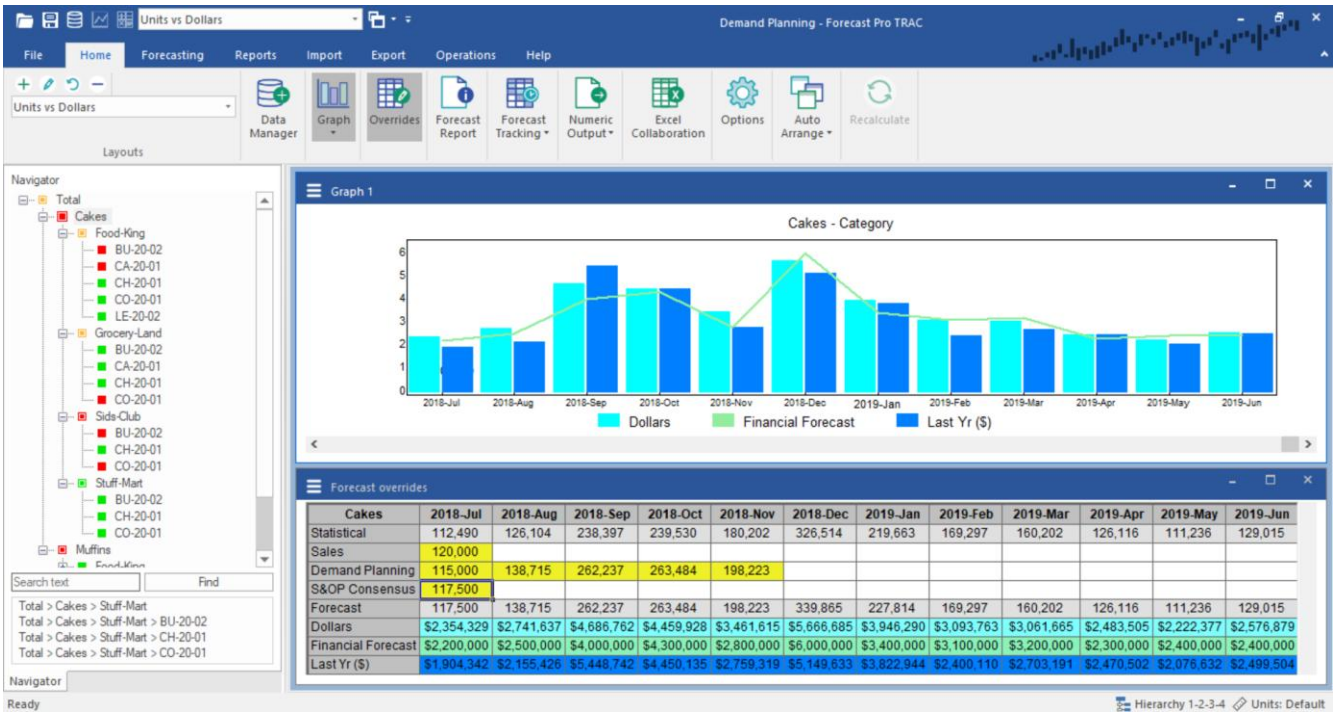
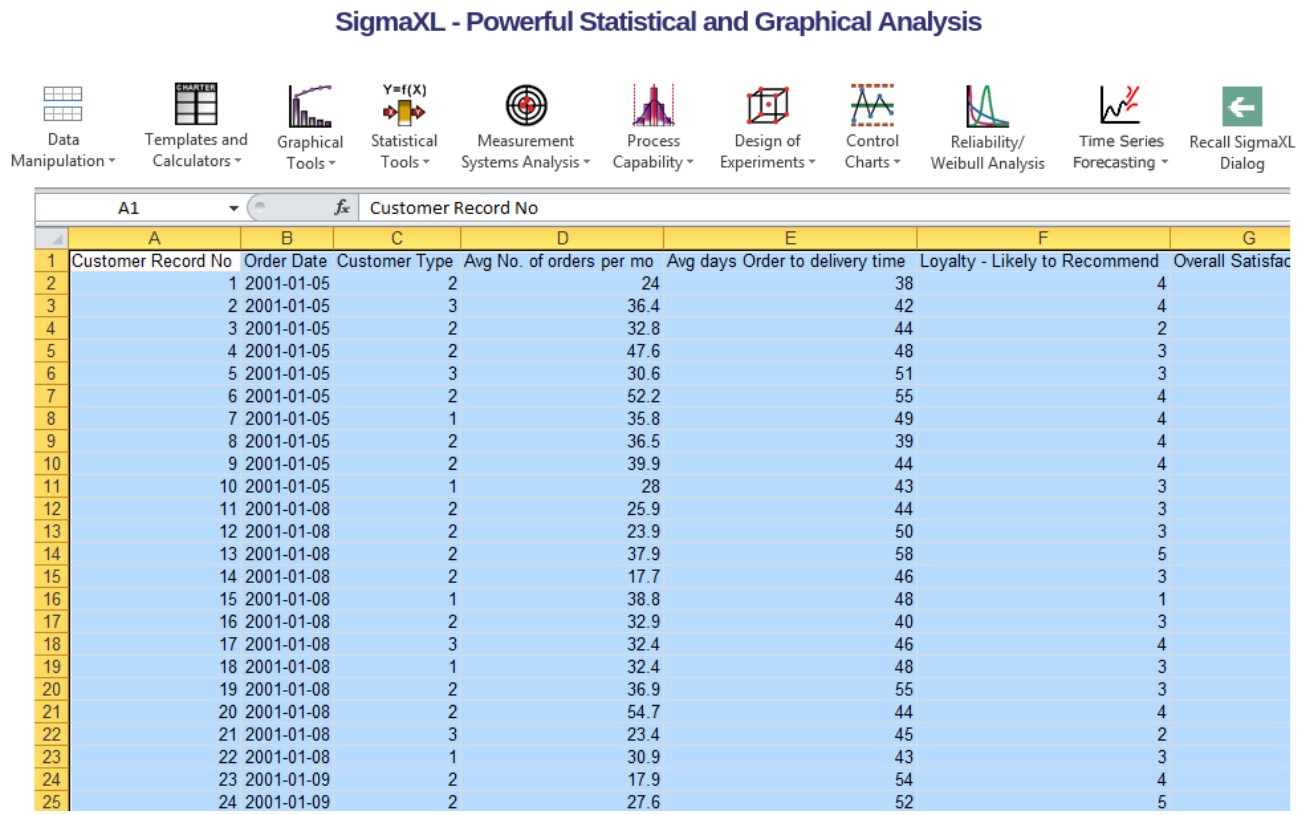


Рисунок 1.2 – Інтерфейс програми Forecast Pro

SigmaXL (рисунок 1.3).

SigmaXL був розроблений з нуля, щоб бути потужним, але простим у використанні інструментом, який дозволяє користувачам вимірювати, аналізувати, покращувати та контролювати свої сервісні, транзакційні та виробничі процеси. Як надбудова до вже знайомого Microsoft Excel SigmaXL ідеально підходить для навчання методу «ощадливе виробництво + шість сигм» або для використання в

курсі статистики в коледжі. Версія 9 додає прогнозування тимчасових рядів та розширені контрольні діаграми.



	A	B	C	D	E	F	G
1	Customer Record No	Order Date	Customer Type	Avg No. of orders per mo	Avg days Order to delivery time	Loyalty - Likely to Recommend	Overall Satisfac
2	1	2001-01-05	2	24	38	4	
3	2	2001-01-05	3	36.4	42	4	
4	3	2001-01-05	2	32.8	44	2	
5	4	2001-01-05	2	47.6	48	3	
6	5	2001-01-05	3	30.6	51	3	
7	6	2001-01-05	2	52.2	55	4	
8	7	2001-01-05	1	35.8	49	4	
9	8	2001-01-05	2	36.5	39	4	
10	9	2001-01-05	2	39.9	44	4	
11	10	2001-01-05	1	28	43	3	
12	11	2001-01-08	2	25.9	44	3	
13	12	2001-01-08	2	23.9	50	3	
14	13	2001-01-08	2	37.9	58	5	
15	14	2001-01-08	2	17.7	46	3	
16	15	2001-01-08	1	38.8	48	1	
17	16	2001-01-08	2	32.9	40	3	
18	17	2001-01-08	3	32.4	46	4	
19	18	2001-01-08	1	32.4	48	3	
20	19	2001-01-08	2	36.9	55	3	
21	20	2001-01-08	2	54.7	44	4	
22	21	2001-01-08	3	23.4	45	2	
23	22	2001-01-08	1	30.9	43	3	
24	23	2001-01-09	2	17.9	54	4	
25	24	2001-01-09	2	27.6	52	5	

Рисунок 1.3 – Інтерфейс програми SigmaXL

Розглянуті вище програми, як і аналогічні їм, можуть проводити прогнозування курсу криптовалют. Для цього потрібне створення моделей та її подальший аналіз. Проте, подібне завдання складне для користувача, який не має відповідної кваліфікації. Крім цього, щодо аналізу потрібно заздалегідь зібрані дані. Тому подібні програми, як і багато інших, які не потрапили до огляду, не підходять для використання користувачем, який не має суттєвих навичок у галузі обробки даних.

1.3 Постановка задачі проектування

В результаті проведеного огляду сформулюємо завдання проектування.

Метою проектування є розробка програми прогнозування курсу криптовалют на прикладі Bitcoin. Для досягнення цієї мети необхідно вирішити такі завдання:

- розробити математичне забезпечення проекту
- розробити інформаційне забезпечення проекту
- розробити програмне забезпечення проекту
- провести тестування розробленої програми
- створити інструкцію користувача

Програма повинна являти собою веб-додаток, який дозволяє користувачеві отримати прогноз курсу криптовалюти Bitcoin на 3 дні вперед, починаючи з поточної дати. Також у користувача має бути можливість вибору іншої дати не пізніше поточної.

Вхідними даними роботи програми є архів з курсом криптовалюти, який має оновлюватися щодня. Вихідними даними є прогноз курсу криптовалюти на три наступні дні.

Алгоритм роботи програми має бути побудований за допомогою моделі ARIMA.

2. ПРОЕКТНІ РІШЕННЯ

2.1 Математичне забезпечення проекту

Нехай заданий часовий ряд даних X_t, де t - цілісний індекс та X_t - дійсні числа, тоді модель ARMA задається як:

$$X_t - \alpha_1 X_{t-1} - \ldots - \alpha_{p'} X_{t-p'} = \epsilon_t + \Theta_1 \epsilon_{t-1} + \ldots + \Theta_q \epsilon_{t-q}$$

або

$$\left(1 - \sum_{i=1}^{p'} \alpha_i L^i\right) X_t = \left(1 + \sum_{i=1}^{q} \Theta_i L^i\right) \epsilon_t$$

де L – оператор запізнення, α_i – параметри авторегресійної частини моделі, $\square_q$- параметри ковзного середнього та $\square_t$- елементи помилки. Елементи помилки зазвичай передбачаються незалежними, однак розподіленими змінними, вибраними з нормального розподілу зі середнім нульовим значенням [14].

Припустимо, що багаточлен $\left(1 - \sum_{i=1}^{p'} \alpha_i L^i\right)$має одиничний корінь кратності d. Тоді його можна переписати як:

$$\left(1 - \sum_{i=1}^{p'} \alpha_i L^i\right) = \left(1 - \sum_{i=1}^{p'-d} \phi_i L^i\right)(1 - L)^d$$

Процес ARIMA(p,d,q) виражає цю властивість поліноміальної факторизації з p=p'−d і визначається як [15]:

$$\left(1 - \sum_{i=1}^{p'-d} \phi_i L^i\right)(1 - L)^d X_t = \left(1 + \sum_{i=1}^{q} \Theta_i L^i\right) \epsilon_t$$

і тому його можна розглядати як окремий випадок процесу ARMA (p + d, q), що має авторегресійний багаточлен з d одиничним корінням.

Модель ARIMA можна як "каскад" двох моделей [16]. Перший нестаціонарний:

$$Y_t = (1 - L)^d X_t$$

У той час як другий є стаціонарним у широкому значенні:

$$\left(1 - \sum_{i=1}^{p} \phi_i L^i\right) Y_t = \left(1 + \sum_{i=1}^{q} \Theta_i L^i\right) \epsilon_t$$

Тепер можна робити прогнози процесу Yt , використовуючи узагальнення методом авторегресійного прогнозування.

2.2 Інформаційне забезпечення проекту

Розглянемо дані, що використовуються для створення моделі.

Дані завантажуються із файлу CSV. На рисунку 2.1 показано, як виглядають перші п'ять рядків файлу.

	Date	Open	High	Low	Close	Volume	Market Cap
0	2017-07-31	2763.24	2889.62	2720.61	2875.34	860,575,000	45,535,800,000
1	2017-07-30	2724.39	2758.53	2644.85	2757.18	705,943,000	44,890,700,000
2	2017-07-29	2807.02	2808.76	2692.80	2726.45	803,746,000	46,246,700,000
3	2017-07-28	2679.73	2897.45	2679.73	2809.01	1,380,100,000	44,144,400,000
4	2017-07-27	2538.71	2693.32	2529.34	2671.78	789,104,000	41,816,500,000

Рисунок 2.1 – Дані про курс біткоїну

Як видно з малюнка, у файлі представлені такі стовпці:

- Date - дата

- Open – Курс під час відкриття торгів

- High – Найбільший курс за день

- Low – Найменший курс за день

- Close – Курс при закритті торгів

- Volume – Обсяг торгів

- Market Cap – Капіталізація ринку.

Файл завантажений з web-ресурсу kuggle і є історичними даними про курс біткоїну.

Використовуючи дату як індекс, побудований графік курсу біткоїну з датою по осі x і ціною закриття по осі y (рис. 2.2, 2.3). Обробка даних провадилася за допомогою Python.

```
data = train['Close']
Date1 = train['Date']
train1 = train[['Date','Close']]
# Setting the Date as Index
train2 = train1.set_index('Date')
train2.sort_index(inplace=True)
print (type(train2))
print (train2.head())
plot.plot(train2)
plot.xlabel('Date', fontsize=12)
plot.ylabel('Price in USD', fontsize=12)
plot.title("Closing price distribution of bitcoin", fontsize=15)
plot.show()
```

Рисунок 2.2 – Вихідний код побудови графіка курсу біткоїну

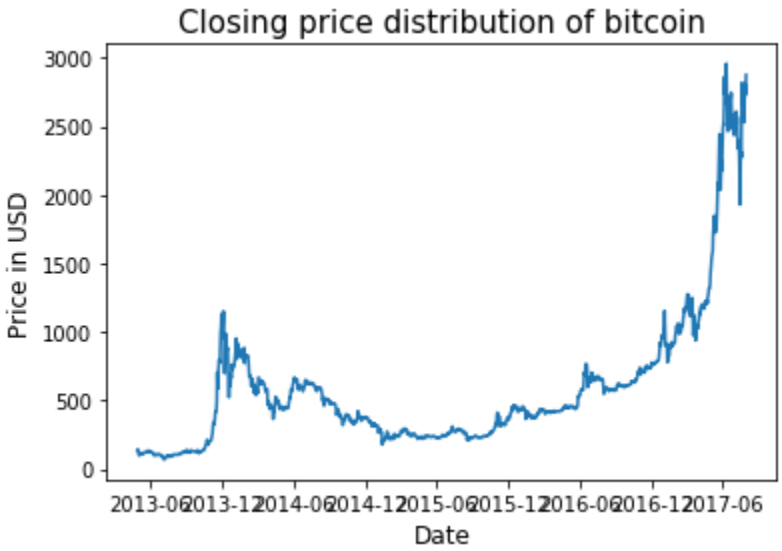


Рисунок 2.3 – Графік курсу біткоїну

Наступним етапом є приведення даних до стаційного вигляду. Щоб зробити годинний ряд стаційарним, поточне значення віднімається від попередніх значень. Завдяки цьому стабілізується середнє значення і, отже, зростають шанси на стаціонарність часових рядів. Процес віднімання показань на рисунку 2.4.

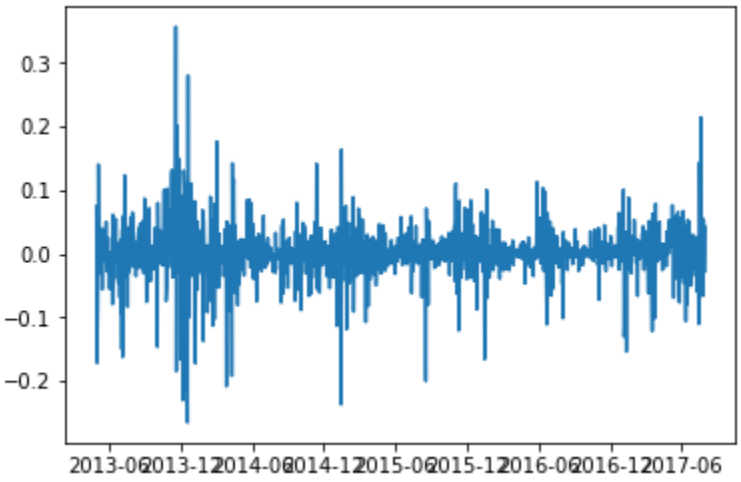


Рисунок 2.4 – Результат обчислення значень

Після цього перевіримо стаціонарність даних (рис. 2.5)

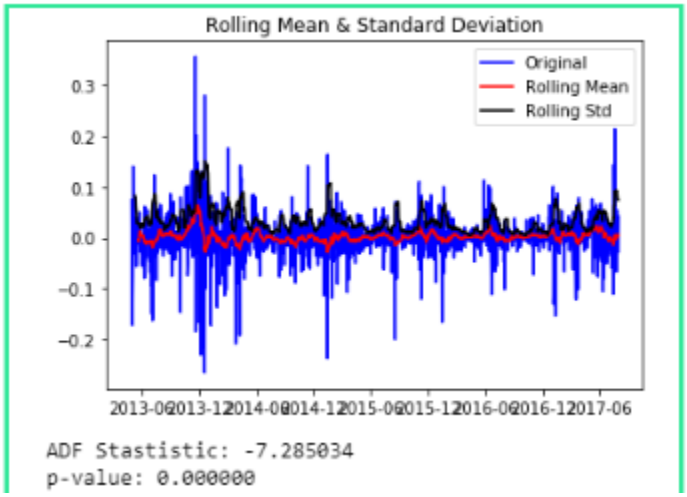


Рисунок 2.5 – Перевірка стаціонарності даних

Використовуваний нами часовий ряд тепер стаціонарний, оскільки значення p менше 0,05 тому може застосовуватися у моделі прогнозування тимчасових рядів.

2.3 Програмне забезпечення проекту

Далі проведемо аналіз стеків технологій, які підходять для реалізації програмного забезпечення, описаного в першому розділі.

Стек розробки означає комбінацію технологій, таких як мови програмування, їх модулі, бібліотеки та фреймворки. Ці компоненти доповнюють один одного для ефективного створення програмного проекту, в даному випадку веб-додатки.

Хоча існують самодостатні та універсальні технології веб-додатків, які за певних умов можуть власними силами забезпечувати бажаний результат, сфера їх застосування обмежена. Найчастіше проект виграє від різноманітності технологій веб-розробки, які слід вибирати окремо для зовнішнього та внутрішнього програмування, щоб оптимізувати продуктивність.

Однак, широкий вибір доступних технологій веб-застосунків ставить проблему правильного вибору. Більш того, недостатньо просто вибрати найкращу технологію для розробки веб-додатків, необхідно проаналізувати різні аспекти конкретного проекту та підібрати оптимальне поєднання інструментів.

Ключем до успішної розробки є пошук групи технологій, що спеціально підходять як для клієнтських, так і для серверних аспектів веб-додатку. Розглянемо ці рівні розробки і дізнаємося, які стеки додатків застосовні для кожного з них.

Клієнтська технологія.

Клієнтська частина програми - це частина, яка видна користувачам і є інтерфейсом програми. Тому його зазвичай називають "front-end". Технології веб-стеку, пов'язані з цим рівнем розробки, нечисленні:

- Мова гіпертекстової розмітки (HTML) керує структурою інформації, що відображається користувачем у веб-браузері.
- Каскадні таблиці стилів (CSS) визначають стиль відображених даних, керуючи такими параметрами, як шрифти, макети та кольори тексту та фону.
- JavaScript керує інтерактивними веб-функціями програми.

Усі ці технології – потужні інструменти в руках кваліфікованих спеціалістів. Проте їхня комбінація відкриває перед розробниками веб-додатків абсолютно новий рівень можливостей. Хоча JavaScript, HTML і CSS вважаються традиційними стандартами в цій галузі, їх можна замінити на такі інструменти розробки, як Apache Flex та інші.

Використання фреймворків — поширена практика, що полегшує процес розробки веб-додатків. Популярні приклади інтерфейсних фреймворків та бібліотек включають:

- Angular – фреймворк, розроблений Google для проектування динамічної структури даних для веб-застосунків.
- React — бібліотека Facebook Inc., орієнтована на проектування інтерфейсів односторінкових додатків, що означає, що програма може відображати

різні дані на одній сторінці без її перезавантаження. Використання JSX – синтаксичного розширення React для Javascript – розробники можуть легко створювати масштабовані інтерактивні рішення.

- Vue.js - полегшення JavaScript-фреймворку для створення адаптивних інтерфейсів односторінкових веб-додатків, що швидко набирає популярності серед розробників з моменту своєї появи в 2014 році.

Щодо веб-додатків сучасна фронтенд-розробка вимагає поєднання кількох компонентів: бібліотек та фреймворків, оскільки одного вже недостатньо для задоволення вимог клієнта. З цієї причини набирають популярності спеціалізовані набори інструментів для веб-розробки. Найбільш популярні їх приклади:

- Bootstrap – колекція шаблонів CSS та JavaScript для дизайну компонентів інтерфейсу. Його основна мета - підвищити чуйність і підтримати принцип веб-розробки "спочатку мобільні". Варто відзначити, що основна увага в цій структурі приділяється веб-сторінкам, а не веб-додаткам.
- Foundation — набір з трьох спеціалізованих фреймворків для веб-сайтів, програм та електронної пошти. Це заощаджує час для розробників інтерфейсу завдяки включеному набору ефективних компонентів для HTML, JS та CSS, орієнтованих на чуйність та підхід «спочатку мобільні».

Кожен із вищезазначених інструментів має свій набір переваг та недоліків, спільноту учасників та низку найбільш підходящих сценаріїв впровадження.

Серверна технологія

Серверна частина веб-програми прихована від користувачів та включає всі компоненти, необхідні для запуску програми.

Компоненти серверної частини включають:

- Мова програмування для написання коду веб-програми. Як і в разі розробки на стороні клієнта, існує кілька мов та безліч фреймворків для спрощення та покращення процесу написання коду. Серед найбільш популярних інструментів для серверного програмування можна назвати такі мови, як Python,

PHP, Ruby, Java та Scala, а також їх фреймворки: Django, Laravel, Ruby on Rails, Spring та Play відповідно.

- База даних для зберігання даних програми. Ці об'єкти можуть бути реляційними або нереляційними залежно від моделі управління даними. Реляційні бази даних використовують у своїй роботі мову структурованих запитів (SQL), тоді як нереляційні (NoSQL) бази даних використовують інші моделі для зберігання та пошуку даних. Відповідно до ресурсу DB-Engines, топ-5 баз даних: Oracle, MySQL, Microsoft SQL Server, PostgreSQL та MongoDB. Примітно, що вони, крім останньої, є реляційними.
- Сервер для обробки запитів, що надходять із боку клієнта, тобто від користувачів програми. Тут вибір досить обмежень: багато веб-програм розробляються з використанням Nginx або Apache як сервери додатків.

Операційна система, використана розробниками, може проводити вибір компонентів у стеку.

Наведемо два найбільш популярні стеки веб-розробки та їх короткий опис нижче.

Python + Django — це потужний стек, який дає безліч переваг розробникам веб-застосунків завдяки чудовій взаємодії, оскільки Django написаний на Python. Крім того, ці дві технології відносно прості для розуміння та вивчення та пов'язані з чистим та коротким кодом. Ці функції дозволяють упростити та прискорити процес розробки, що в поєднанні з вбудованою функціональністю обох технологій призводить до швидкого та ефективного створення веб-додатків.

Залежно від проекту стек «Python + Django» часто включає MySQL як компонент бази даних та/або може додатково поєднуватися з Javascript, Apache та іншими технологіями.

LAMP – це стек, що включає операційну систему Linux, HTTP-сервер Apache, програмне забезпечення бази даних MySQL та PHP як мову

програмування. Цей стек розробки є дуже популярним завдяки своїй продуктивності та гнучкості, оскільки допускає безліч варіацій та модифікацій.

У цьому конкретному контексті операційна система є обмеженням: можна встановити сервер LAMP у Windows за допомогою підсистеми Windows для Linux (WSL). Як альтернативу можна замінити Linux на MacOS (MAMP), BSD (BAMP) чи Solaris (SAMP), якщо хочете. Більш того, можна використовувати в цьому стеку Windows та Microsoft SQL Server (WASP) замість Linux та MySQL відповідно.

MEAN складається з бази даних MongoDB, веб-фреймворку Express.js, що використовується для серверної частини, Angular.js як каркас для зовнішнього інтерфейсу та Node.js — середовища виконання, яке функціонує як веб-сервер. Ця концепція використовує JavaScript як для серверної, так і для клієнтської розробки.

Крім того, існує тенденція, що швидко розвивається, так званих «хмарних» додатків, яка переглядає концепцію використання технологічних стеків. Такий підхід дозволяє побудувати додаток у вигляді окремих сервісів, і кожен із цих сервісів може бути розроблений власною мовою та фреймворком. Це дозволяє вибрати оптимальну технологію для кожного конкретного сервісу, яку можна розробляти, тестувати та оновлювати незалежно та/або одночасно.

Хмарна своя програма розгортається і розміщується на одній із платформ хмарних обчислень, таких як Amazon Web Services, які пропонують повний спектр функцій, доступних через різні API. Загалом AWS включає понад 90 сервісів, від бази даних та сховища до мультимедіа та машинного навчання.

Таким чином, виходячи з аналізу існуючих технологій та прикладів сайту, можна сформувати стек технологій для розробки сайту.

Для створення статичного наповнення сайту необхідно використовувати мову розмітки HTML 5 та каскадні таблиці стилів CSS . Ці технології на сьогоднішній день є, по суті, стандартом де-факто при розробці статичних веб-сторінок.

Для надання динамічності сторінкам необхідно використовувати JavaScript та бібліотеку функцій jquery , що дозволить за допомогою готових рішень реалізувати багато динамічних функцій сторінок

Для серверної мови програмування необхідно використовувати Python та його фреймворк для веб-додатків – DJANGO. Це рішення дозволить швидко створити сайт із готових шаблонів із достатніми показниками швидкості роботи, стійкості до навантаження та безпеки з використанням вільного програмного забезпечення. За промовчанням у проекті використовується база даних SQLite .

SQLite – це система управління реляційними базами даних (RDBMS). На відміну від багатьох інших систем управління базами даних SQLite не використовує клієнт-серверний механізм.

SQLite зазвичай слідує синтаксису PostgreSQL, але не вимагає перевірки типів. Це означає, що, наприклад, можна вставити рядок у стовпець, визначений як ціле число.

SQLite є популярним вибором як вбудоване програмне забезпечення бази даних для локального/клієнтського сховища в прикладному програмному забезпеченні, такому як веб-браузер. На сьогоднішній день є широко розповсюдженим двигуном бази даних, оскільки використовують у декількох широко поширених браузерах, операційних системах та вбудовуваних системах (таких як мобільні телефони). SQLite має прив'язки до багатьох мов програмування.

На відміну від клієнт-серверних систем управління базами даних механізм SQLite не має автономних процесів, з якими взаємодіє прикладна програма. Натомість підключається бібліотека SQLite, яка стає невід'ємною частиною прикладної програми. Зв'язування може бути статичним чи динамічним. ПЗ використовує функціональні можливості SQLite за допомогою простих функцій, які зменшують затримку при доступі до бази даних: виклики функцій в рамках одного процесу більш ефективні, ніж взаємодія між процесами.

SQLite зберігає всю базу даних (визначення, таблиці, індекси та самі дані) у вигляді одного кроссплатформенного файлу на хост-комп'ютері. Він продає цей простий дизайн, блокуючи весь файл бази даних під час запису. Операції читання SQLite можуть виконуватись у багатозадачному режимі, хоча записи можуть виконуватись лише послідовно.

Через безсерверну конструкцію SQLite вимагають меншої настройки, ніж клієнт-серверні бази даних. SQLite називається zero-conf, тому що він не вимагає управління службами (такими як сценарії запуску) або контролю доступу на основі GRANT та паролів. Керування доступом здійснюється за допомогою дозволів файлової системи, що надаються самому файлу бази даних. Бази даних у клієнт-серверних системах використовують дозволи файлової системи, які надають доступ до файлів бази даних лише процесу демону.

Ще одним наслідком безсерверної архітектури є те, що кілька процесів може мати можливості запису файлу бази даних. У серверних базах даних кілька модулів запису підключатимуться до одного і того ж демона, який може самостійно обробляти свої блокування. SQLite, з іншого боку, має покладатися на блокування файлової системи. Він менше знає про інші процеси, що одночасно звертаються до бази даних. Тому SQLite не є найкращим вибором для розгортання з інтенсивним записом. Однак для простих запитів із невеликим паралелізмом продуктивність SQLite виграє від відсутності накладних витрат на передачу своїх даних іншому процесу.

Основні причини вибору СУБД:

- висока надійність;
- малі вимоги до дискового простору та пам'яті;
- вільне поширення СУБД у відкритих кодах;
- кроссплатформенність.

Програму організовано у вигляді веб-сайту, розробленого за допомогою фреймворку MVC. MVC - це платформа веб-застосунків, яка реалізує шаблон моделі-представлення-контролера (MVC).

До появи фреймворків MVC веб-програмування змішувало код бази даних із серверним кодом сторінки.

Шаблон MVC був створений для відокремлення бізнес-логіки від представлення. MVC – найпопулярніша архітектура, яка використовується сьогодні. Багато популярних фреймворків, таких як Ruby on Rails, Laravel, CodeIgniter та Django, використовують його. Архітектура MVC ділити додаток на наступні три рівні:

- Модель.
- Подання
- Контролер.

Обговоримо кожен із них окремо.

Моделі представляють, як дані організовані у базі даних. Іншими словами, у шаблоні MVC ми використовуємо моделі визначення таблиць нашої бази даних, а також відносин між ними.

Подання – це те, що бачити користувач, коли відвідує сайт. Наприклад, повідомлення в блогі, контактна форма і т. д. - це приклади уявлень. Подання містить всю інформацію, яка зрештою буде надіслана клієнту, тобто веб-браузеру. Зазвичай уявлення є HTML-документами.

Контролер контролює потік інформації. При запиті сторінки цей запит передається контролеру, він використовує запрограмовану логіку, щоб вирішити, яку інформацію необхідно витягти з бази даних і яку інформацію слід передати уявленню. Контролер – це серце архітектури MVC, тому що він діє як клей між моделями та уявленнями [17].

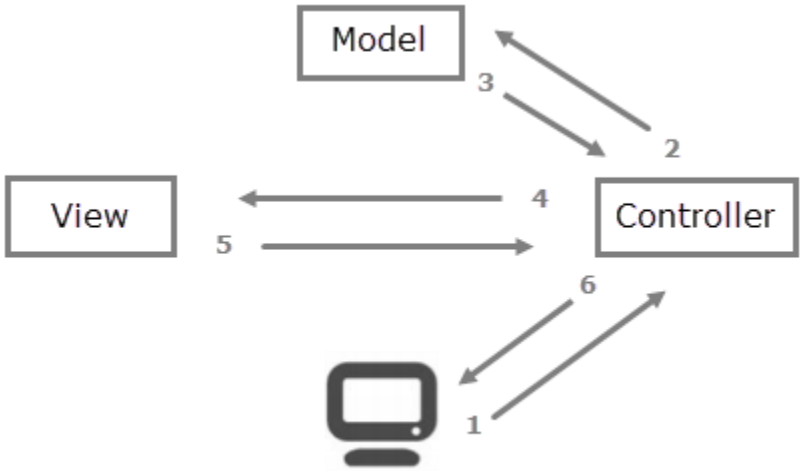


Рисунок 2.8 - Схема передачі у шаблоні MVC

Розглянемо короткий опис кроків, що виникають при надсиланні запиту на MVC:

- Веб-браузер або клієнт надсилає запит на веб-сервер, прохаючи сервер відобразити дані.
- Запит, отриманий сервером, передається контролеру програми.
- Контролер проситиме модель отримати запитувані.
- Модель надсилає дані контролеру.
- Потім контролер передає дані у виставу.
- Подання використовує дані для створення HTML-сторінки.
- Нарешті, контролер повертає клієнту вміст HTML.

Слід зазначити, що Django слідує шаблону MVC, але використовує іншу термінологію. Django використовує термін шаблони для представлення та подання для контролера. Іншими словами, уявлення в Django називаються шаблонами, а контролери – уявленнями. Отже, наш HTML-код буде у шаблонах, а код Python – у моделях.

Для створення нового проекту з віртуального оточення, що містить Django, виконаємо команду django-admin startproject bitcoin_prediction. Потім додамо до

проекту нове додаток, виконавши команду python manage.py startapp core. На малюнку 2.9 показано сформоване у проекті дерево файлів.

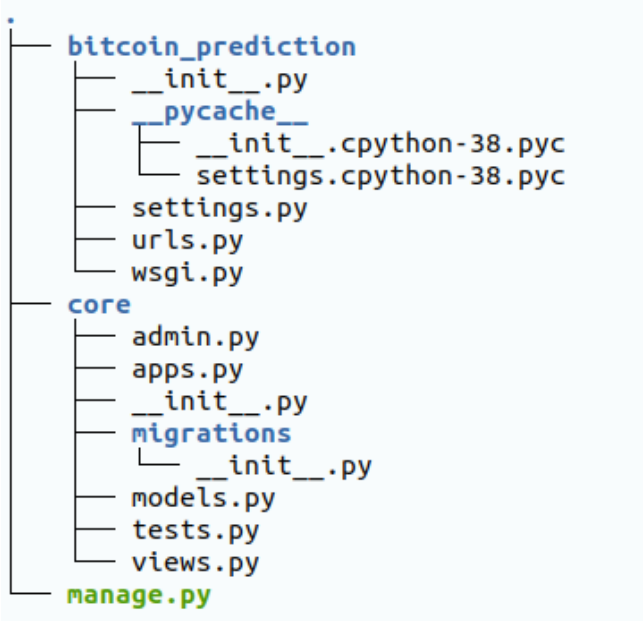


Рисунок 2.9 – Дерево файлів проекту

Наступним кроком є створення моделі Arima. Модель реалізована як окремий модуль arima.py.

Роботу моделі можна представити як чорну скриньку, в яку передається інформацію про курс біткоїну і на виході повертається прогноз курсу (рис. 2.10).

Рисунок 2.10 – Схема роботи моделі

Основні функції програми показано малюнку 2.11.

Як видно з малюнка, функції можна поділити на основні та службові.

До основних функцій відносяться:

- Get_rate – повертає прогноз курсу криптовалюти на основі моделі ARIMA

- Fetch_data – отримує дані про поточний курс із зазначеного як аргумент ресурсу та зберігає інформацію в базі даних

- Get_help – виводить довідку про програму

- Get_history – виводить історію курсу на основі збережених даних

До службових функцій відносяться:

- Parse_file – аналізує файл з даними про курс криптовалюти та повертає інформацію про курс у вигляді масиву

- Load_data - завантажує файл із даними

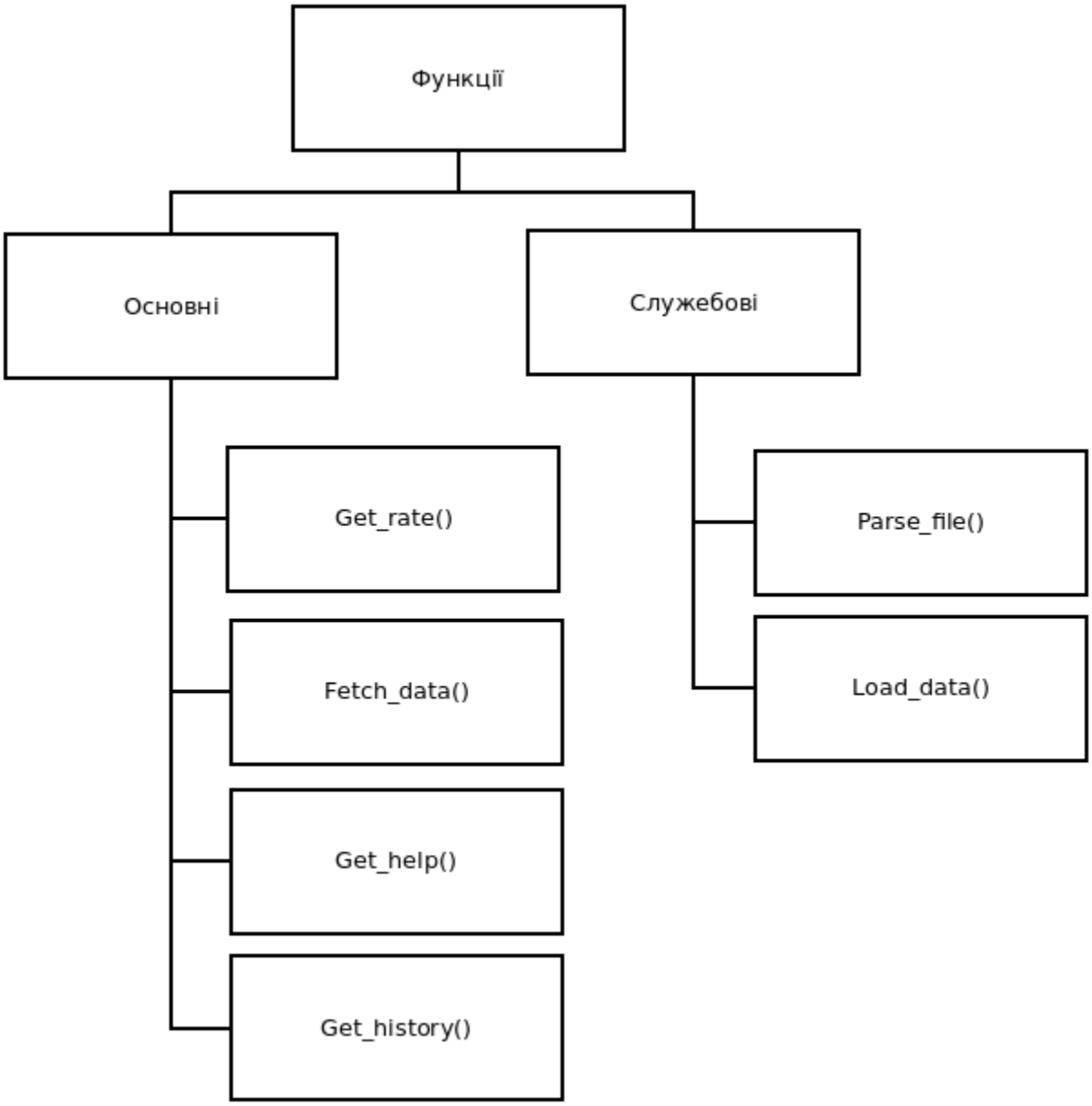


Рисунок 2.11 – Функції програми

Розробка елементів інтерфейсу здійснюється за допомогою шаблонів веб-сторінок, які розташовані у папках templates та core/templates. У першій міститься базовий шаблон, який використовується в кожній сторінці інформаційної системи і включає підключення стилів і бібліотеки jquery. Дерево вікон програми представлене малюнку 2.12

Рисунок 2.12 – Дерево вікон програми

Далі представимо логіку роботи клієнтської частини ІВ та наведемо опис сторінок.

Після запуску проекту відображається вікно авторизації (рис. 2.13).

Рисунок 2.13 – Вікно авторизації

Якщо авторизаційні дані відповідають аккаунту користувача, відкривається головна сторінка, в якому користувач може побачити поточний курс біткоїну, а також має можливість дізнатися про прогноз курсу криптовалюти, а також переглянути збіг прогнозу з історичними даними (рис. 2.14).

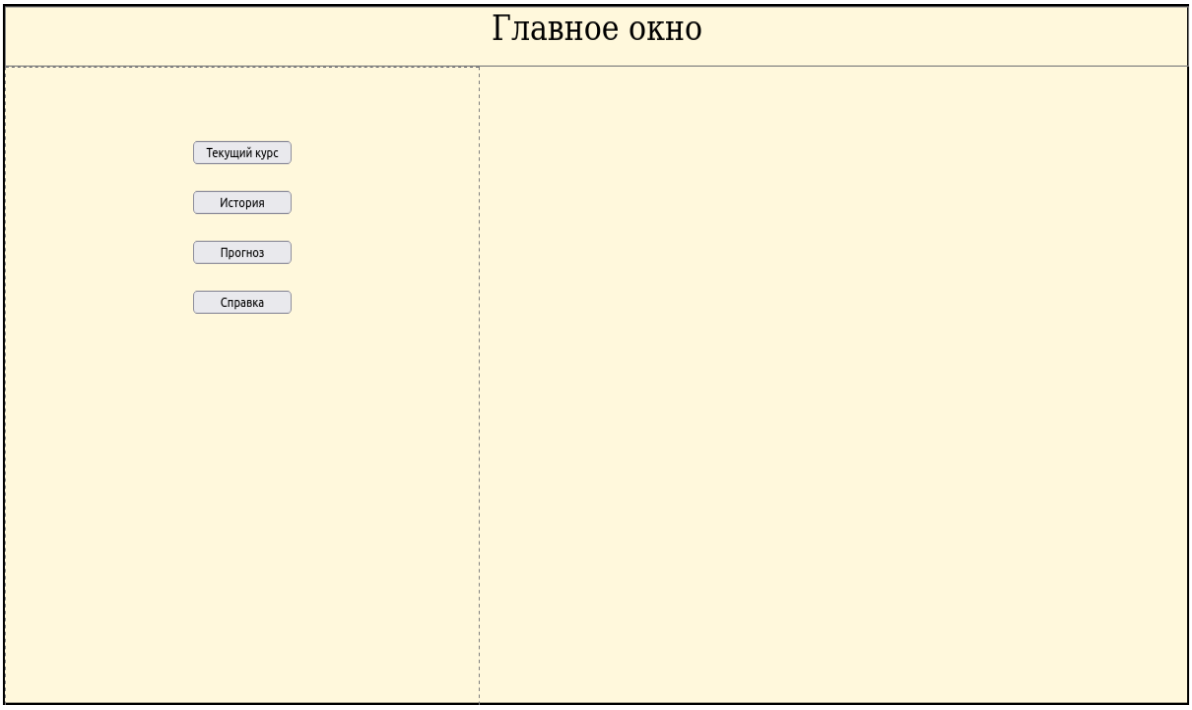


Рисунок 2.14 – Головна сторінка

3. РЕАЛІЗАЦІЯ, ТЕСТОВИЙ ПРИКЛАД

3.1 Тестування роботи програми

На основі тестових даних описаних раніше проведемо моделювання курсу біткоїну та порівняємо дані моделі з реальними даними (рис. 3.1). Тестові дані в даному випадку прийняті як історичні та доступні у відповідному вікні програми. Детальніше інтерфейс програми описів у наступному розділі.

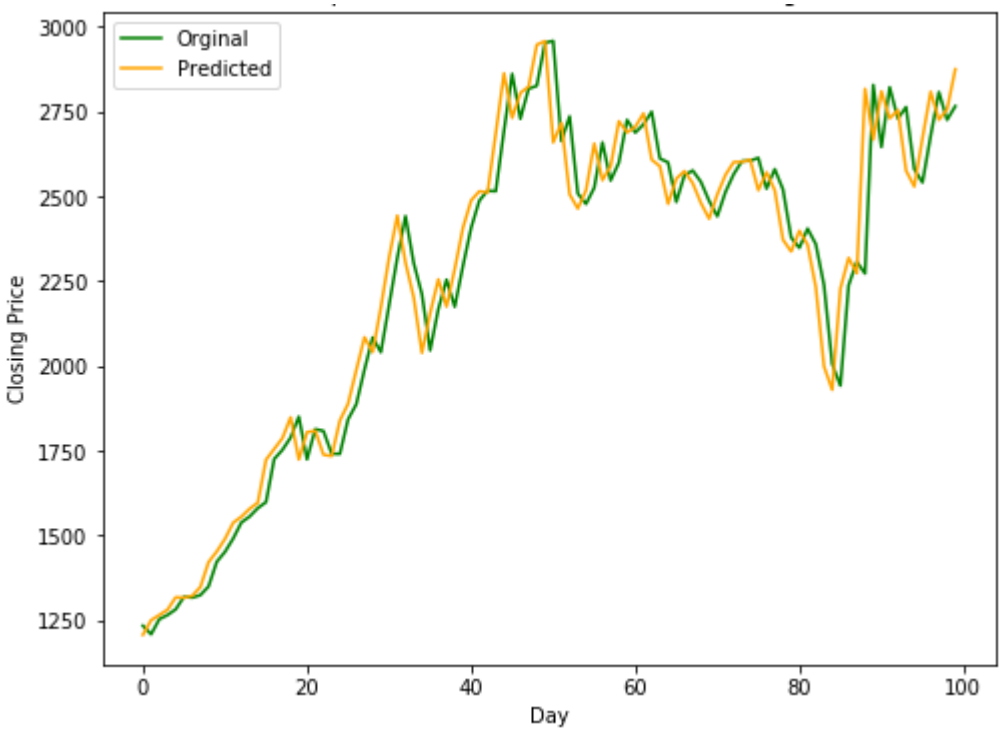


Рисунок 3.1 – Графік фактичних та передбачених даних

Як очевидно з аналізу, передбачені дані досить точно повторюють реальні.

Також проведемо аналогічні вимірювання з даними, взятими на більшому діапазоні дат, починаючи з 2013 до 2018 року. Зразок цих даних уявлень малюнку 3.2.

	open	high	low	close
date				
2013-04-28	135.30	135.98	132.10	134.21
2013-04-29	134.44	147.49	134.00	144.54
2013-04-30	144.00	146.93	134.05	139.00
2013-05-01	139.00	139.89	107.72	116.99
2013-05-02	116.38	125.60	92.28	105.21

Рисунок 3.2 – Набір тестових даних

Графік реальних та історичних значень цього набору уявлень на рисунку 3.3. У програмі ця функціональність закладена у вікні "Прогноз".

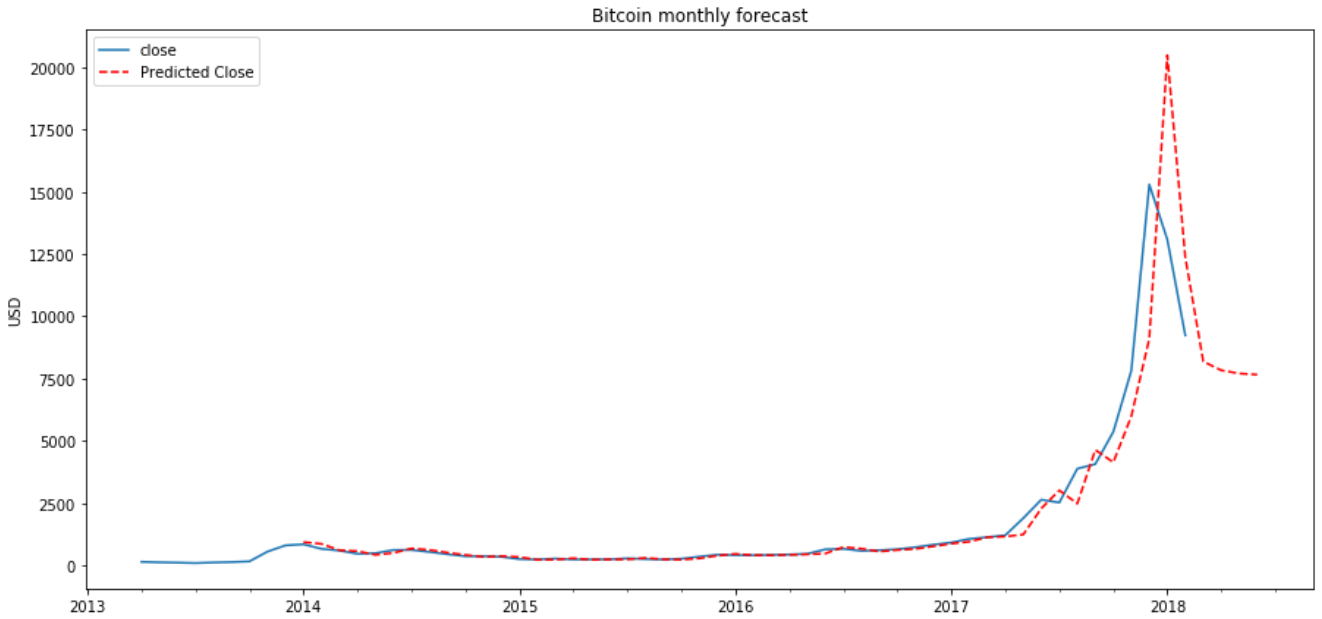


Рисунок 3.3 – Графік фактичних та передбачених значень

Для оцінки правильності моделі побудуємо такі залежності, як стандартизований залишок, гістограму та щільність розподілу, графік QQ значень та корелограму.

Стандартизований залишок є мірою різниці між спостережуваними та очікуваними значеннями. Як очевидно з рисунку 3.4, ця величина досить мала протягом усіх спостережень.

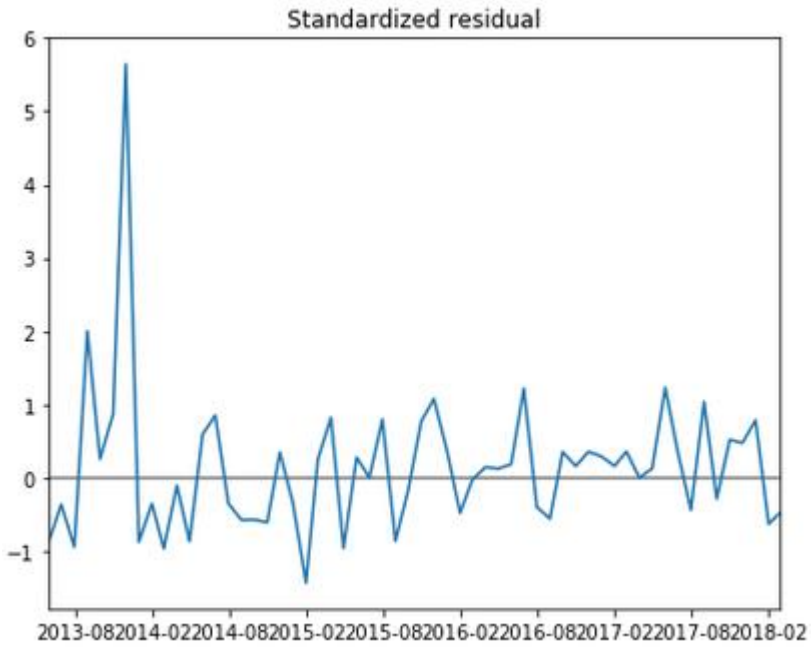


Рисунок 3.4 – Стандартизований залишок

Як показано на гістограмі (рис. 3.5), викиди, які присутні на початку вимірювань, не вносять великого вкладу в загальний результат.

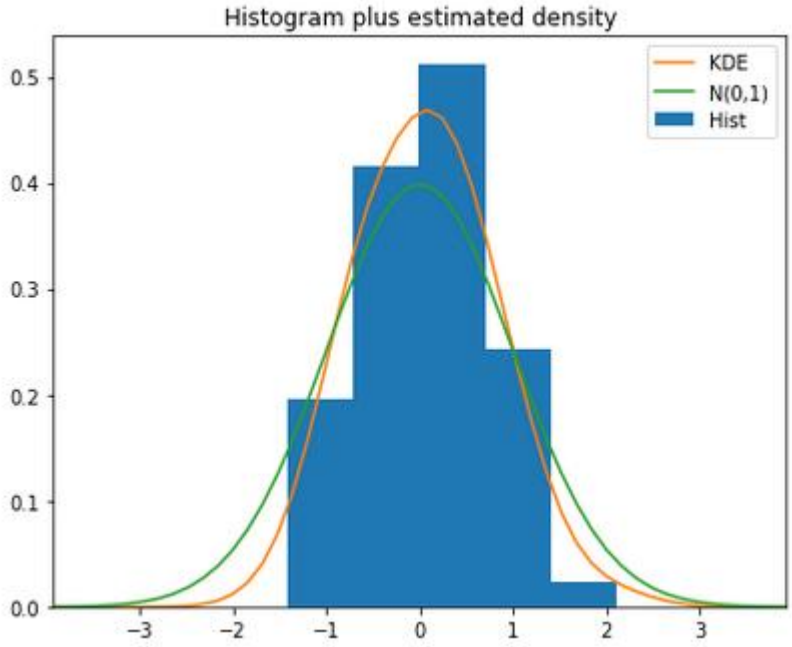


Рисунок 3.5 – Гістограма та щільність розподілу

Також на графіку 3.6 показані реальні та передбачені значення у вигляді QQ графіка, крім цього побудована королелограма цих значень (рис. 3.7).

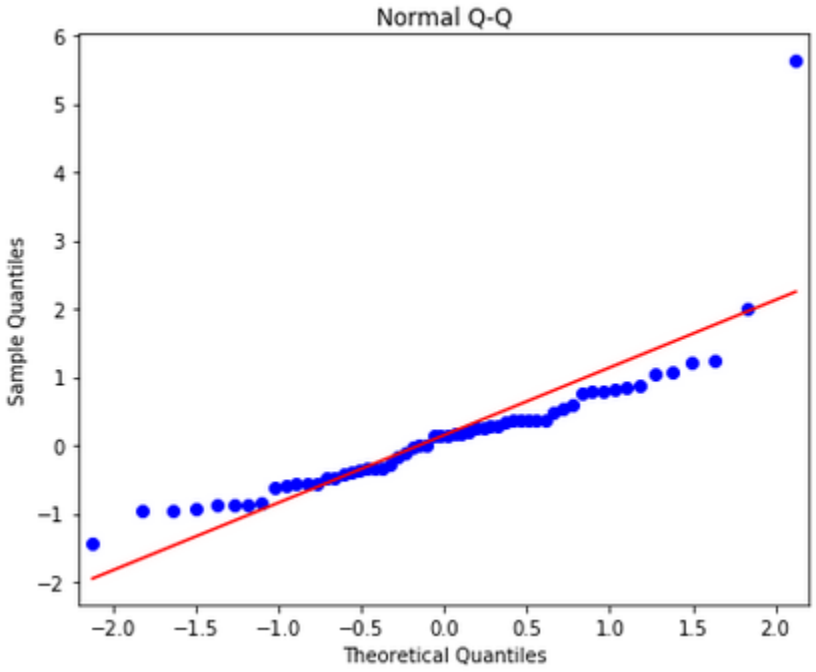


Рисунок 3.6 – Графік QQ значень

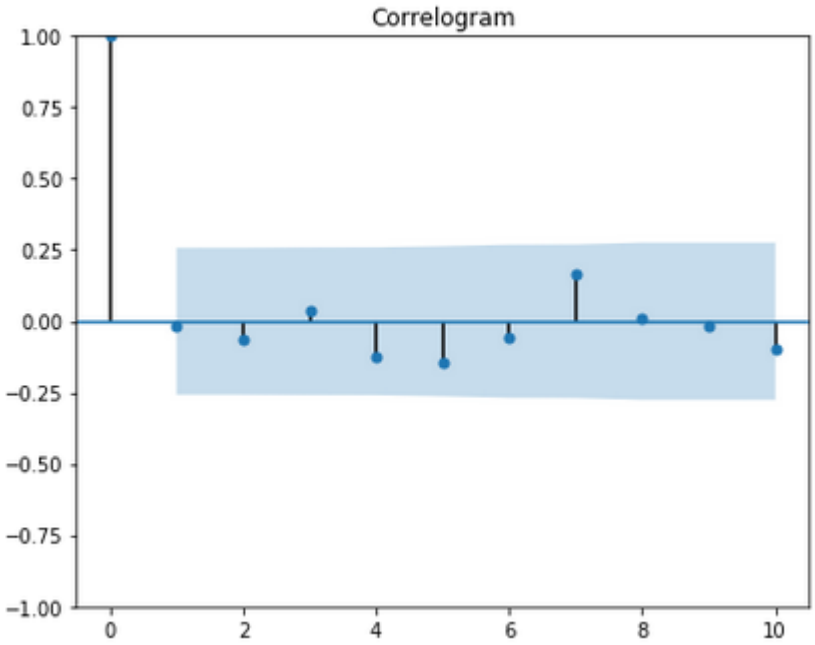


Рисунок 3.7 – Корелограма моделі

Нарешті, знайдемо значення середньоквадратичної помилки моделі. Для цього скористаємося кодом, наведеним нижче.

```
y_forecasted = btc_month2.forecast
y_truth = btc_month2['2015-01-01':'2017-01-01'].close

# Compute the root mean square error
rmse = np.sqrt(((y_forecasted - y_truth) ** 2).mean())
print('Mean Squared Error: {}'.format(round(rmse, 2)))
```

Рисунок 3.8 – Код для обчислення середньої квадратичної помилки

В результаті набуваємо значення середньоквадратичної помилки, що дорівнює 85.18 (рис. 3.9)

```
Mean Squared Error: 85.18
```

Рисунок 3.9 – Значення середньої квадратичної помилки для моделі

Таким чином, дані значення показують, що розроблена модель досить точно описує реальні дані і може бути використана для прогнозування курсу біткоїну. Далі розглянемо інтерфейс програми.

3.2 Інструкція користувача

Програма розроблена у вигляді web-програми, тому доступна з будь-якого комп'ютера, на якому встановлений браузер і який має доступ до Інтернету. Для розгортання програми на хостингу потрібна підтримка мови програмування Python та фреймворку Django.

Розглянемо інтерфейс програми. Після переходу на посилання проекту, у браузері з'являється вікно авторизації (рис . 3.10), в якому потрібно ввести логін та пароль користувача. Якщо облікові дані неправильні, з'являється повідомлення про помилку. Якщо облікові дані вірні, відбувається перехід на головну сторінку.

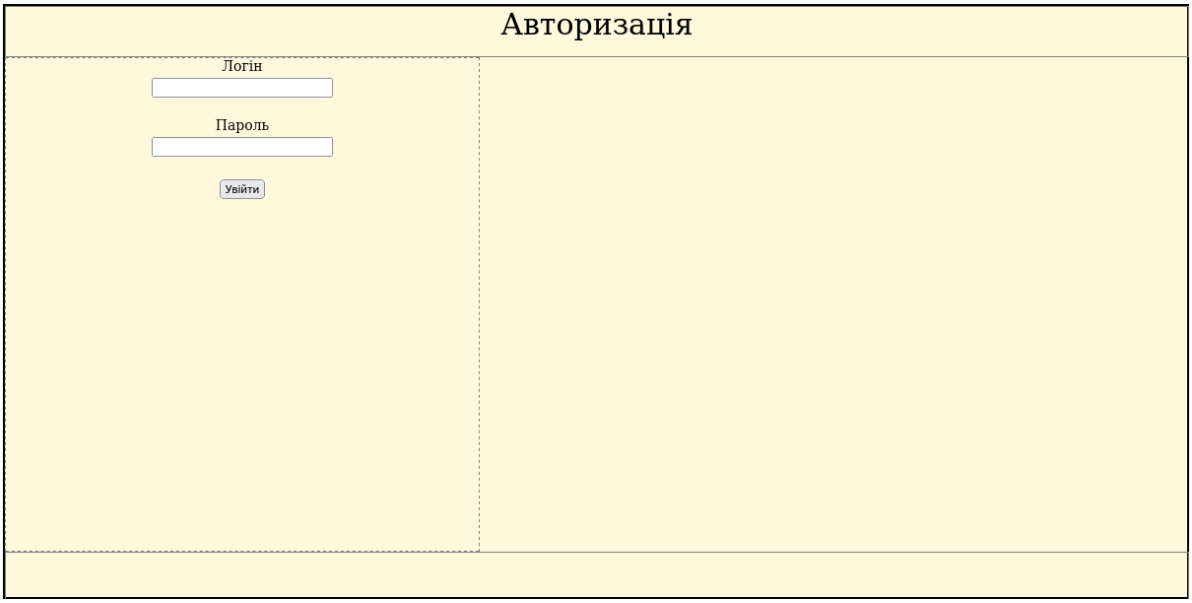


Рисунок 3.10 – Сторінка авторизації

Головне вікно дозволяє виконувати такі операції (рис. 3.11):

- переглядати інформацію про поточний курс криптовалюти Біткоїн та орієнтовний прогноз на наступний день;
- відображати динаміку курсу за останні 100 днів
- завантажувати та обробляти за допомогою моделі ARIMA дані про курс
- Переглядати довідкову інформацію.

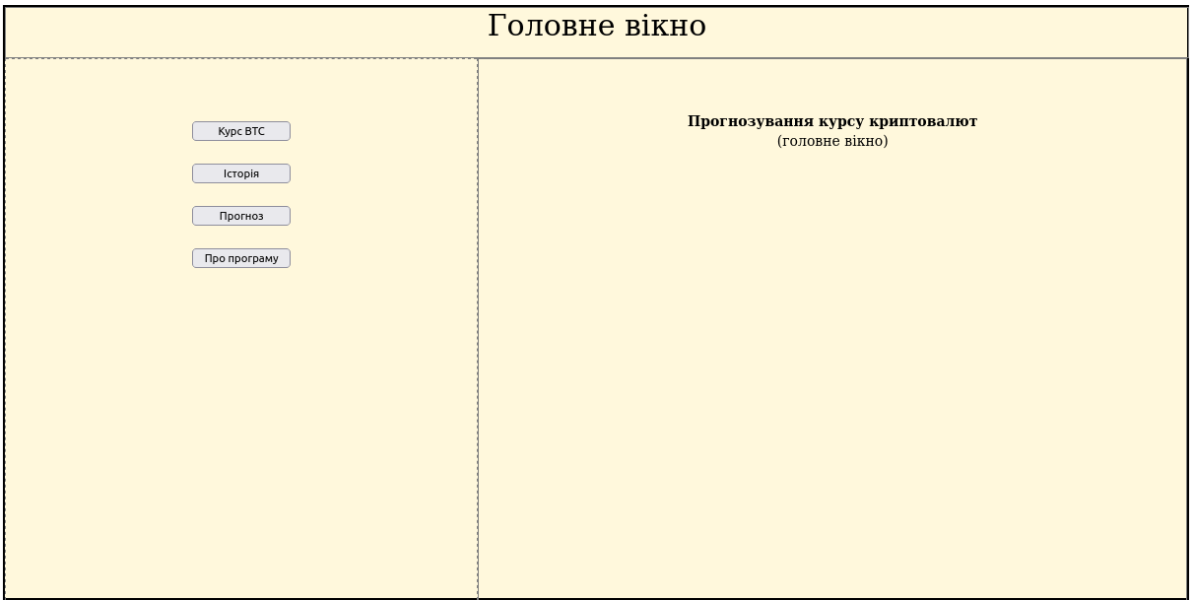


Рисунок 3.11 – Головне вікно програми

При натисканні кнопки “Курс BTC” відображається інформація про поточний курс біткоїну (рис. 3.12). Курс оновлюється щороку. Також можна побачити прогноз курсу на наступний день, що базується на створеній моделі.

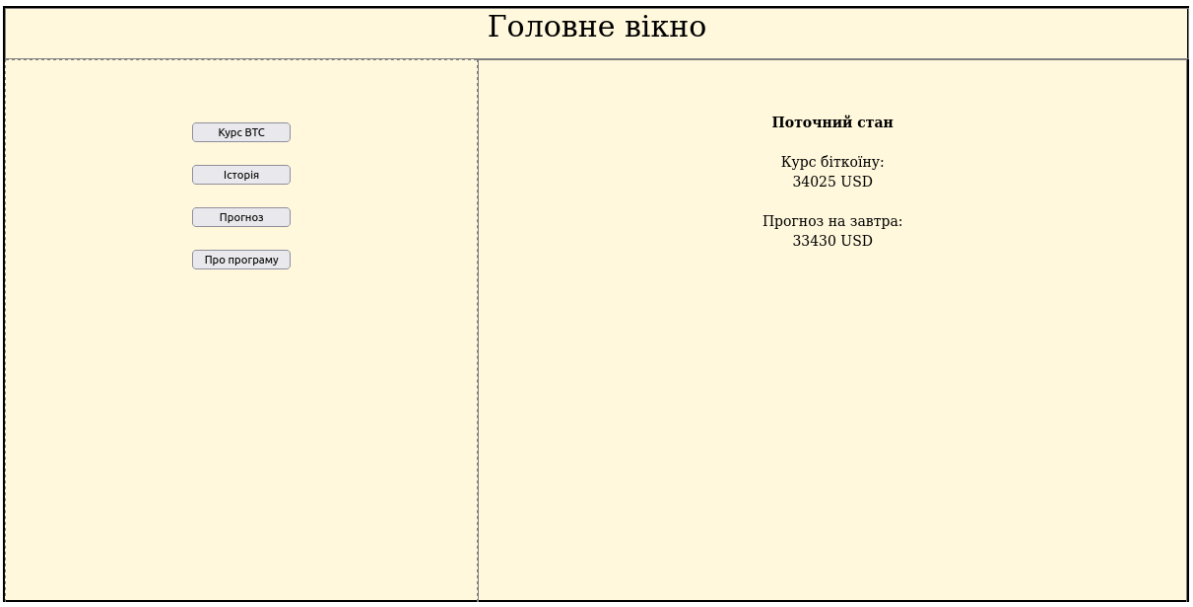


Рисунок 3.12 – Відображення поточного курсу Біткоїна

Для перегляду історії зміни курсу необхідно натиснути кнопку “Історія” (рис. 3.13). Програма на основі даних про курс біткоїну будує графік, що відображає курс останніх 100 днів. Також до цих даних додається прогноз, який дозволяє побачити наскільки точно модель передбачає динаміку даних.

Рисунок 3.13 – Відображення історичних даних та прогнозу щодо них

Програма дозволяє завантажувати та обробляти дані, для цього треба натиснути кнопку “Прогноз”. На рисунку 3.14 показані дані та прогноз для курсу біткоїну з 2013 по 2018 роки, які описані раніше.

Рисунок 3.14 – Моделювання архівних даних

Натискання кнопки “Про програму” відображає вікно з короткою справкою про можливості програми (рис. 3.15).

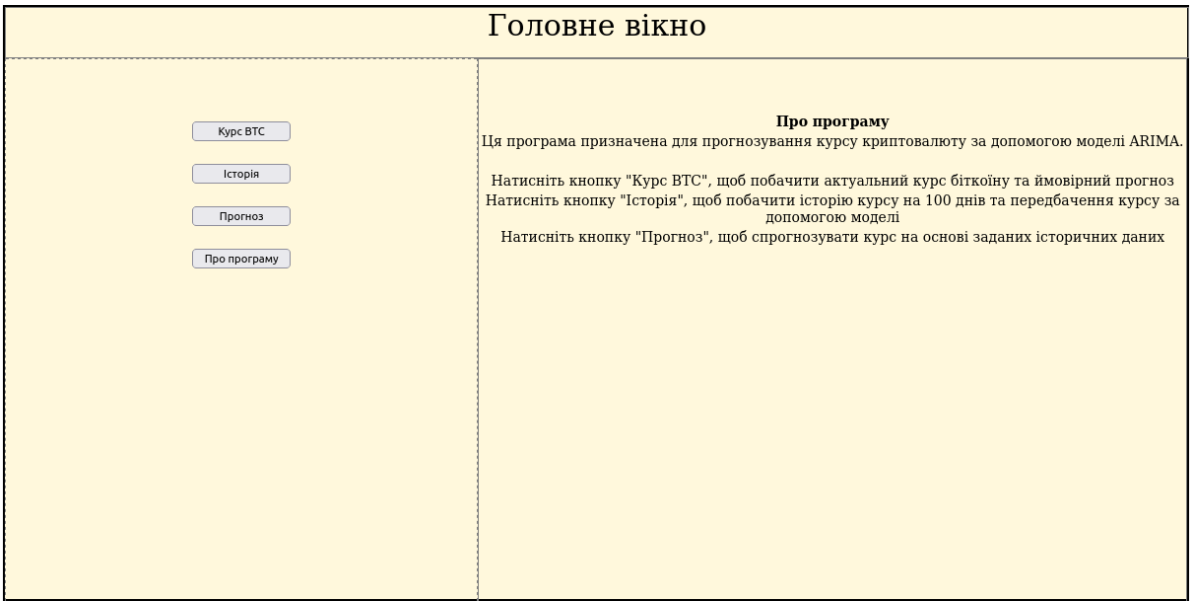


Рисунок 3.15 – Довідкова інформація

Таким чином, у цьому розділі описано роботу програми, включаючи аналіз результатів роботи та опис інтерфейсу користувача.

ВИСНОВКИ

У цій роботі досліджувалися можливості прогнозування курсу криптовалют. Результатом дослідження стала програма, яка прогнозує курс криптовалюти Біткоїн за допомогою математичного апарату ARIMA. Програма розроблена за допомогою мови програмування Python і є web-додатку, яке виконує кілька завдань:

- відображення поточного курсу біткоїну та його прогноз
- відображення графіків курсу криптовалюти біткоїну та прогнозу курсу протягом більше 3 місяців
- можливість обробки історичних даних про курс криптовалюті.

Пояснювальна записка складається із трьох розділів.

У першому розділі проведено аналіз сучасних методів прогнозування курсу валют в цілому та криптовалют зокрема. Показано, що курс криптовалют зараз можна прогнозувати шляхом аналізу часових рядів. Серед методів такого аналізу вибрано модель ARIMA.

В іншому розділі наведено математичне забезпечення проекту, показано математичну сутність моделі ARIMA. Також проведено підготовку тестового набору даних про курс криптовалюти, показано, що тестовий набір даних може використовуватись для прогнозування курсу. Показано розробку програми.

У третьому розділі проаналізовано результати роботи програми, які показують, що модель ARIMA може використовуватись для прогнозування курсу криптовалюти. Розроблено посібник користувача, в якому описано можливості програми та наведено приклади її роботи.

Таким чином, у цьому дослідженні вирішено всі поставлені завдання та досягнуто мети дослідження.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ ТА ІНТЕРНЕТ-ДЖЕРЕЛ

1. S tepanenko, O.: Problems and Prospects of Financial Provision Modeling of Ecological, Economics and Production Systems in the Digital Economy. B: Answers at 7th International youth conference on Perspectives of science and education, pp. 421-430 (2019)

2. Bouoiyour, J., Selmi, R.: What does Bitcoin look like? Ann. Econom. Financ. 16(2), 449–492 (2015)

3. Cheah, E., Fry, J.: Speculative bubbles y Bitcoin markets? An empirical investigation y the fundamental value of bitcoin. Економ. Lett. 130, 32-36 (2015)

4. Casey, MJ (2015). The Age of Cryptocurrency: How Bitcoin and Blockchain Are Challenging the Global Economic Order. London: St. Martin's Press, 368.

5. Antonopoulos, AM (2017). Mastering Bitcoin: Unlocking Digital Crypto-Currencies. London: O'Reilly Media, 416.

6. Ivasenko, AG (2015). Rynok tsennykh bumag: instruments and mekhanizmy funktsionirovanieia. Москва: Kno-Rus, 272.

7. Wafi, AS, Hassan, H., Mabrouk, A. (2015). Fundamental Analysis Models y Financial Markets – Review Study. Procedia Economics and Finance, 30, 939-947. doi: http://doi.org/10.1016/s2212-5671(15)01344-1

8. Мерфі, Дж. Дж. (2011). Технічні аналізи фючерсних ринків. Teoriia та практика. Москва: Alpina Pablisher, 616.

9. Luther, WJ (2015). Cryptocurrencies, network effects, і switching costs. Contemporary Economic Policy, 34(3), 553-571. doi: http://doi.org/10.1111/coep.12151

10. Dale, O. (2018). Cryptocurrency Trading: Momentum Indicators and Moving Averages. Available at: https://blockonomi.com/moving-averages/

11. Podkopytnova, DV, Atanova, EV (2017) Застосування методом skolziashchikh srednikh на фондовому ринку. Научно-практическое дослідження, 9 (9), 105-107.

12. Grafik aktsii APPLE INC. Trendové indikatory. Available at: https://ua.tradingview.com/chart/

13. Зіновік, О. (2017). Порівняння різних tipov skolziashchikh srednikh v torgovle ». Available at: https://www.mql5.com/ru/articles/3791

14. Gheyas IA, Smith LS A Neural Network Approach до Time Series Forecasting // Proceedings of World Congress on Engineering, London, 2009, Vol 2 [електронний ресурс]. P. 1292 – 1296. URL: www.iaeng.org/publication/WCE2009/WCE2009_pp1292-1296.pdf

15. Mazengia DH Forecasting Spot Electricity Market Prices Using Time Series Models: Thesis for the degree of Master of Science in Electric Power Engineering . Gothenburg, Chalmers University of Technology, 2008. 89 p.

16. Norizan M., Maizah Hura A., Zuhaimy I. Short Term Load Forecasting Using Double Seasonal ARIMA Model // Regional Conference on Statistical Sciences, Malaysia, Kelantan, 2010. P. 57 - 73.

17. Collantes-Duarte J., Rivas-Echeverriat F. Time Series Forecasting using ARIMA, Neural Networks and Neo Fuzzy Neurons // WSEAS International Conference on Neural Networks and Applications, Switzerland, 2002 [електронний ресурс]. 6 p. URL: www.wseas.us/e-library/conferences/switzerland2002/papers/464.pdf

ДОДАТКИ

Вихідний код модуля прогнозування

```
# Necessary Packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import statsmodels.formula.api as smf
з statsmodels.tsa.stattools import adfuller
from sklearn.metrics import mean_squared_error
from pandas import read_csv
from pandas import datetime
from pandas import DataFrame
з statsmodels.tsa.arima_model import ARIMA
from matplotlib import pyplot
# Bitcoin data
data1 = pd.read_csv('bitcoin.csv', header=0)
# visualize the data frame, first 10 elements
print(data1.head(4150))
#graphical representation
data1.plot()
plt.show()
#read data again
series = read_csv('bitcoin.csv', header=3370, index_col=0)
```

```
від statsmodels.graphics.tsaplots import acf, pacf
з statsmodels.graphics.tsaplots import plot_acf, plot_pacf
pyplot.figure()
pyplot.subplot(211)
plot_acf(series, ax=pyplot.gca(),lags=30)
pyplot.subplot(212)
plot_pacf(series, ax=pyplot.gca(),lags=30)
pyplot.show()
# returns or first difference of the time series
returns=series.diff()
returns.plot()
plt.show()
з statsmodels.tsa.arima_model import ARIMA
import pmdarima as pm
model = pm.auto_arima(series, start_p=1, start_q=1, test='adf', max_p=6, max_q=6, m=1, d=None, seasonal=False, start_P=0, D=0,
trace=True, error_action='ignore', suppress_warnings=True, stepwise=True)
print(model.summary())
# fit an ARIMA model for the series
model = ARIMA (series, order=(1,1,1))
model_fit = model.fit(disp=0)
print(model_fit.summary())
# plot residual errors
residuals = DataFrame(model_fit.resid)
fig, ax = plt.subplots(1,2)
residuals.plot(title="Residuals", ax=ax[0])
residuals.plot(kind='kde', title='Density', ax=ax[1])
```

```
plt.show()
print(residuals.describe())
print(model_fit.aic)
# Actual vs Fitted
def forecast_accuracy(forecast, actual):
mape = np.mean(np.abs(forecast - actual)/np.abs(actual)) # MAPE
me = np.mean(forecast - actual) # ME
mae = np.mean(np.abs(forecast - actual)) # MAE
mpe = np.mean((forecast - actual)/actual) # MPE
rmse = np.mean((forecast - actual)**2)**.5 # RMSE
corr = np.corrcoef(forecast, actual)[0,1] # corr
mins = np.amin(np.hstack([forecast[:,None],actual[:,None]]), axis=1)
maxs = np.amax(np.hstack([forecast[:,None],actual[:,None]]), axis=1)
minmax = 1 - np.mean(mins/maxs) # minmax
# acf1 = acf (fc-test) [1] # ACF1
return({'mape':mape, 'me':me, 'mae': mae,'mpe': mpe, 'rmse':rmse,'corr':corr, 'minmax':minmax})
yy=model_fit.predict(652)
xx=series.iloc[652:781,0]
tt=forecast_accuracy(yy, xx)
print('forecast accuracy',tt)
pyplot.figure()
plt.scatter(xx,yy, marker='o')
pyplot.show()
pyplot.figure()
model_fit.plot_predict(652,781)
model_fit.plot_predict(dynamic=False)
```

```
pyplot.show()
# separate data in training and test data (95% i 5%)
forecast3, stderr, conf_int = model_fit.forecast(10)
print(forecast3)
pyplot.plot(forecast3,color='green')
pyplot.show()
pyplot.figure()
model_fit.plot_predict(752,791)
pyplot.show()
from math import sqrt
from sklearn.metrics import mean_squared_error
rms = sqrt(mean_squared_error(yy, xx))
print('RMSE', rms)
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import statsmodels.formula.api as smf
ɜ statsmodels.tsa.stattools import adfuller
from sklearn.metrics import mean_squared_error
from pandas import read_csv
from pandas import datetime
from pandas import DataFrame
ɜ statsmodels.tsa.arima_model import ARIMA
from matplotlib import pyplot
ɜ statsmodels.tsa.arima_model import ARIMA
```

```
з statsmodels.tsa.arima_model import ARMA
від statsmodels.tsa.seasonal import seasonal_decompose
# Bitcoin data
data = pd.read_csv('bitcoin.csv')
#divide into train and validation set
train = data[:int(0.835*(len(data)))]
valid = data[int(0.835*(len(data))):]
#preprocessing (since arima є univariate series as input)
train.drop('Date',axis=1,inplace=True)
valid.drop('Date',axis=1,inplace=True)
#plotting the data
train['bprice'].plot()
valid['bprice'].plot()
#building the model
від pmdarima import auto_arima
model = auto_arima (train, suppress_warnings = True)
model.fit(train)
trace=True,
error_action='ignore',
forecast = model.predict(n_periods=len(valid))
forecast = pd.DataFrame(forecast,index = valid.index,columns=['Prediction'])
#plot the predictions for validation set
pyplot.figure()
plt.figure(figsize=(10, 6))
plt.plot (train, label = 'Train')
plt.plot(valid, label='Valid')
```

```
plt.plot(forecast, label='Prediction')
plt.show()
#calculate rmse
from math import sqrt
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
rms1 = sqrt(mean_squared_error(valid,forecast))
print('RMSE',rms1)
```

print('MAE', mean_absolute_error(valid,