Forecasting Cryptocurrency Prices Using Advanced Machine Learning Techniques*

Tariq Emad Ali[1,*,†], Faten Imad Ali[2,†], Norbert Pataki[3,†] and Alwahab Dhulfiqar Zoltán[3,†]

[1]Department of Information and Communication Engineering, Al-Khwarizmi College of Engineering, University of Baghdad, Baghdad, Iraq
[2]Department of Biomedical Engineering, College of Engineering, AL-Nahrain University, Baghdad, Iraq
[3]Faculty of Informatics, Eötvös Loránd University, Budapest, Hungary

Abstract

Cryptocurrencies are probably the biggest breakthrough of the last decade, which has brought us one step closer to the concept of decentralization. Crypto has a massive market capitalization of 2 trillion dollars, but the price fluctuation is still extremely volatile. So the basic idea of this project is to build a system which will be able to predict the price for the upcoming day. For this purpose, this paper presents a comprehensive study on forecasting cryptocurrency prices using advanced machine learning techniques, focusing on both the development and implementation aspects. Our primary contribution lies in the integration of these techniques into a user-friendly web dashboard, enabling users to visualize and predict the daily prices of five major cryptocurrencies. We employ various machine learning algorithms, providing a detailed comparison and justification for their selection. Additionally, our open-source Python code and web dashboard are made available for further research and practical applications.

Keywords

Python, JavaScript, Flask, RF

1. Introduction

Decentralization refers to the distribution of power among all participants rather than concentrating it on a central authority. In a decentralized environment, the activities and planning of a group are shared among its members, addressing the issues inherent in centralized systems[1]. This principle is applied in various fields such as government, economics, finance, and technology and is particularly prominent in computer science. A key goal has been to develop a fully decentralized internet where no single entity owns it, but everyone has a stake. Cryptocurrencies exemplify this decentralization. They are digital currencies used for buying and selling goods and services and operate on decentralized systems known as blockchains. These blockchains are maintained across multiple computers, ensuring enhanced security and faster transaction speeds. Among the thousands of cryptocurrencies in circulation, Bitcoin, created in 2009 by an unknown entity, is the most notable, with about 19 million Bitcoins currently in circulation and a market cap exceeding one trillion dollars [2]. The volatility of cryptocurrency prices, driven by diverse political and economic factors, presents a significant challenge for accurate forecasting. This paper addresses this challenge by leveraging advanced machine learning (ML) techniques to predict the daily prices of five major cryptocurrencies. Our research uniquely integrates these ML techniques into a web-based dashboard, developed using Python and its extensive libraries, to enhance accessibility and usability. We provide a thorough literature review, justify our choice of algorithms,

SQAMIA 2024: Workshop on Software Quality, Analysis, Monitoring, Improvement, and Applications, September 9–11, 2024, Novi Sad, Serbia

*You can use this document as the template for preparing your publication. We recommend using the latest version of the ceurart style.

*Corresponding author.
†These authors contributed equally.
dolfi@inf.elte.hu (A. D. Zoltán)
https://aalwahab.web.elte.hu/ (A. D. Zoltán)
0000-0003-0505-8669 (T. E. Ali); 0000-0002-1767-8825 (F. I. Ali); 0000-0002-7519-3367 (N. Pataki); 0000-0002-7893-6250 (A. D. Zoltán)

and discuss the potential applications and beneficiaries of our system. In addition, we make our code and dashboard openly accessible to facilitate further research and practical implementations. To predict prices as accurately as possible, three different machine learning algorithms were used: Multiple Linear Regression, which extends linear regression by using multiple variables to predict outcomes; Ridge regression, which estimates multiple regression coefficients when independent variables are highly correlated; and Random Forest, which creates multiple decision trees during data training and selects the most common outcome for classification tasks. This paper aims to design and train a model to predict the prices of these highly volatile assets.

2. Literature Survey

A literature review was conducted to identify machine learning algorithms suitable for predicting coin prices. The pertinent literature was examined, and the results are summarized in Table 1. Based on the literature review conducted on these articles, RF, GB, LSTM, and GRU emerged as effective predictors of cryptocurrency prices. Therefore, we focused on using machine learning algorithms for experimentation. In addition, ridge regression, multiple linear regression, and random forest algorithms were used in our tests. Predicting cryptocurrency prices is inherently challenging due to their high volatility and susceptibility to external factors such as political events, market manipulation, and economic changes. While our models show promising results, several limitations need to be acknowledged:

- Market Manipulation: Large players can influence market trends, making predictions less reliable.
- External Factors: Political and economic events can drastically impact prices, which our models do not account for.
- Historical Data Limitation: Our models rely solely on historical price data, without incorporating sentiment analysis or other external indicators.

Table 1
Summary of Papers on Cryptocurrency Price Prediction

Paper	Algorithms Used	Main Findings	Conclusion
[3]	GRU, LSTM, Bi-LSTM	GRU outperform predicting BTC, ETH, and LTC prices.	GRU model is reliable for cryptocurrency prediction.
[4]	LSTM, RNN	LSTM & RNN used for predicting daily cryptocurrency prices.	Performance depends on training data quality.
[5]	ML techniques: RF, XGB, QDA, SVM, LSTM	ML methods improve Bitcoin price prediction accuracy.	Proper selection of character improves prediction accuracy.
[6]	ANN, RF	ANN demonstrates better stock price prediction than RF.	Optimal system required for stock market analysis.
[7]	RNN, MLP, LSTM	LSTM shows highest accuracy in predicting cryptocurrency movements.	Trend prediction neural networks lack economic cost understanding.
[8]	BART, MLP, RF	Models do not strictly impose constraints on statistical characteristics.	More comprehensive datasets needed for better accuracy.
[9]	LSTM, LOG, RF, GBT	LSTM and LOG show comparable accuracy in stock prediction.	Properly chosen features improve prediction accuracy.
[10]	LSTM, GRU, RNN, GBC	LSTM and GRU show high accuracy in short-term cryptocurrency prediction.	Features selection crucial for prediction accuracy improvement.
[11]	Rectilinear regression, LSTM	LSTM outperforms rectilinear regression in Bitcoin prediction.	Cryptocurrency market prediction remains challenging due to market volatility.

3. Available Cryptocurrencies for Prediction

The project aims to predict the prices of the top five most popular cryptocurrencies: Bitcoin, Ethereum, Cardano, Ripple, and Dogecoin. These cryptocurrencies were selected due to their significant presence in the digital currency market. Bitcoin, often referred to as the gold of cryptocurrency, was created in 2008 by an anonymous individual or group known as Satoshi Nakamoto [12]. Bitcoin operates on a system called mining, where users contribute their computer's computing power to maintain the blockchain and are rewarded with bitcoins in return. Currently, there is a maximum supply of 21 million bitcoins, with nearly 19 million coins already in circulation and a total market capitalization exceeding 1.1 trillion US dollars. Ethereum, founded in 2013, is a decentralized blockchain platform with smart contract functionality. It is second only to Bitcoin in terms of market capitalization. Unlike Bitcoin, Ethereum does not have a fixed maximum supply limit, with approximately 120 million coins currently in circulation and a market capitalization of more than 550 billion US dollars [13]. Cardano, established in 2015 by Ethereum cofounder Charles Hoskinson, is an open source and decentralized blockchain platform that utilizes proof-of-stake technology. The native Cardano coin, ADA, can be exchanged through peer-to-peer transactions. With a maximum supply of 45 billion coins, approximately 32 billion ADA coins are currently in circulation, boasting a market capitalization of more than 50 billion US dollars [14]. Ripple, founded in 2012 by the US-based technology company Ripple Labs Inc., operates on a distributed open-source protocol. Ripple's native coin, XRP, has a maximum supply limit of 100 billion coins, with approximately 47 billion coins currently in circulation and a market capitalization of around 47 billion US dollars [15]. Dogecoin, created in 2013 by Billy Markus and Jackson Palmer, initially started as a humorous project but has since gained legitimacy as an investment option. Dogecoin does not have a fixed maximum supply. Currently, around 130 billion Dogecoins are in circulation, with a market capitalization exceeding 28 billion US dollars [16].

4. Methodology

Our methodology encompasses the entire pipeline, from data collection and preprocessing to model training and web integration. We selected Python for its robust machine learning libraries, including Scikit-learn and TensorFlow, and HTML/CSS for developing the web dashboard. We begin with extensive data preprocessing to handle the volatility and noise in cryptocurrency data. We then train multiple ML models, including Random Forest (RF), Ridge Regression, and others, providing a detailed explanation for selecting these models based on their performance and suitability for the task. Finally, we integrate these models into a dynamic web dashboard using Flask, allowing users to interact with the predictions. We also discuss the implementation challenges and the specific requirements addressed during development. The Python requests library facilitated data retrieval from the Coingecko website through its APIs, enabling HTTP requests to be sent to the desired website. Data handling and formatting were streamlined using the Pandas library, which simplifies the manipulation and formatting of large datasets through its DataFrame data structure, also helping to manage missing data. NumPy, another Python library, provided support for handling large and multidimensional datasets, offering advanced mathematical functions for data manipulation. Within the project, NumPy was utilized for applying mathematical calculations to the extensive dataset. The Scikit-learn library in Python facilitated the implementation of machine learning algorithms such as multiple linear regression, ridge regression, and random forest. Widely utilized for its support of various classification, regression, and clustering algorithms, Scikit-learn was instrumental in applying machine learning techniques to predict future cryptocurrency prices. Flask, a micro-web framework written in Python, was employed to develop a web dashboard to present project results. Renowned for its simplicity and flexibility, Flask does not require specific tools or libraries for usage. Provided a live server, debugger, and integrated support for unit testing and RESTful request dispatching, enhancing the development process. HTML and CSS were used to design the web dashboard, given their widespread usage in web development. Additionally, JavaScript was incorporated to enhance the dashboard functionality. Following the ECMAScript specification,

JavaScript offers object-oriented and first-class function support. Chart.js, a JavaScript library, facilitated the creation and management of various types of graph to display cryptocurrency price and date data, improving the visualization of the data within the dashboard.

The dashboard, constructed using Python's Flask framework, was designed to be accessible across multiple platforms, including Android, iOS, and desktops. Comprising six distinct webpages, it provides detailed insights into the top five cryptocurrencies: Bitcoin, Ethereum, Cardano, Dogecoin and Ripple. So, in this paper, the dashboard consists of a home page which provides the configurability and a description, a navigation bar which is presented consistently for accessing the main pages, a footer which shows developers' contact information, and the most unusual predication pages that show the graphs illustrating price fluctuations over time and the predications. User navigation within the dashboard is facilitated by the navigation bar located at the top of every page, allowing a seamless transition between the home page and the prediction pages for each cryptocurrency. All pages comprises three main sections:

1. **Coin Information**: At the top of the page, users find essential details about the coins cryptocurrency, accompanied by its logo, providing a comprehensive overview before delving into the price analysis.
2. **Price Graph**: Following the coin description, users encounter a graph depicting the historical price of coins in US Dollars over the past five years. This graph offers valuable insights into the coin's price trends and fluctuations, aiding users in making informed investment decisions.
3. **Prediction Options**: The last section of the page presents the core functionality of the dashboard: price prediction. Here, users are presented with four distinct buttons, each offering a different prediction method like generates a predicted price for coins using the machine learning algorithms. Then computes an average prediction by combining the results of all three aforementioned prediction algorithms, offering users a comprehensive perspective on the anticipated price movement of coins. These prediction options empower users to leverage advanced machine learning algorithms to forecast the future price of coins accurately. By providing a range of prediction methods, the dashboard caters to diverse user preferences and investment strategies, enhancing the overall user experience and decision-making process.

5. Experiment Implementation

5.1. Dataset

The dataset utilized for this project comprises the coin data for five distinct cryptocurrencies, sourced from CoinGecko. Retrieving the data involved sending a GET request to CoinGecko's API using Python's 'requests' library. Initially, the dataset was acquired as a string and subsequently converted to JSON format using Python's 'json.loads' function. This first step is data preprocessing.Data preprocessing is crucial to ensuring that the data is cleaned and optimized for model performance. This step involves handling null values, removing unnecessary data, and structuring the dataset appropriately. In our project, data preprocessing was performed predominantly using the 'pandas' library, which offers efficient data manipulation capabilities. The data retrieved from CoinGecko's API was filtered and processed to extract only the relevant information, such as the price of the cryptocurrency, market capitalization, and total volume at each timestamp. The data was then organized into Python dictionaries and Pandas DataFrames for further analysis. The second step is data modeling. This step involves analyzing the dataset and establishing relationships between its elements to devise a suitable data model for the given requirements. In our project, Python dictionaries and Pandas DataFrames were primarily used for data modeling. After obtaining the data from CoinGecko's API, it was stored in variables and converted into Python dictionaries. These dictionaries were structured to contain relevant information, such as the price, market capitalization, and total volume of the cryptocurrency at each timestamp. Additionally, weekly and monthly moving averages were calculated to provide additional factors for predicting future prices. The calculated moving averages were then incorporated into the

dataset, resulting in a DataFrame suitable for applying machine learning algorithms. This DataFrame enabled the implementation of various algorithms to predict cryptocurrency prices based on historical data. Through effective data pre-processing and modeling, we prepared the dataset for analysis and prediction, laying the groundwork for the subsequent application of machine learning algorithms.

5.2. Machine Learning Modeling

In computer science, Machine Learning (ML) is a branch of Artificial Intelligence (AI) that enables software to become increasingly accurate in predicting outcomes without explicit programming. ML algorithms utilize historical data to train themselves and predict future outcomes, improving their accuracy through repeated execution of tasks. These algorithms are broadly categorized into two groups: classification and regression. Classification algorithms partition datasets into various groups, making them useful for tasks like handwriting recognition or identifying cancerous cells. Regression algorithms, on the other hand, aim to find the best-fit line to predict future outcomes, making them valuable for forecasting stock prices or housing market trends. In our study, we used three regression algorithms, Multiple Linear Regression (MLR), Ridge Regression, and Random Forest, to predict cryptocurrency prices. The following sections provide detailed descriptions of each of these algorithms.

5.2.1. Multiple Linear Regression

Multiple Linear Regression (MLR) is an extension of the simple linear regression algorithm. In MLR, we use multiple explanatory variables to train the model and predict the outcome of a response variable, whereas simple linear regression uses only one explanatory variable for this purpose. MLR is a highly effective technique for predicting outcomes, provided that sufficient data is available. The accuracy of this algorithm is directly proportional to the amount of training data. The multiple linear regression is represented by equation (1)

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip} + \epsilon \tag{1}$$

where y_i represents the dependent or response variable that MLR aims to predict. On the right side, $x_{i1}, x_{i2}, \ldots, x_{ip}$ are the explanatory variables, with β_0 being the y-intercept and β_p the slopes of these variables. The term ϵ denotes the model error. One of the primary advantages of MLR is its ability to incorporate multiple explanatory variables along with the response variable. In our paper, we utilized three explanatory variables: timestamp, monthly moving averages, and weekly moving averages. The necessity of handling multiple variables led us to choose MLR over simple linear regression. Another benefit of MLR is its applicability to various datasets, regardless of size. However, MLR also has some drawbacks, primarily related to data quality. If incorrect data is used, the model will still generate results, but these may be inaccurate. MLR cannot automatically detect missing or faulty data. Additionally, if the explanatory variables are not correlated, the results' accuracy can be compromised. Therefore, careful data modeling and preprocessing are crucial when using MLR to ensure reliable predictions. To use the MLR machine learning algorithm in our study, we utilized the Python library called scikit-learn, which is one of the most common and efficient libraries available for implementing machine learning algorithms. Initially, the dataset, currently in a data frame, is divided into two parts: the explanatory variables and the dependent variable. In this study, the explanatory variables are the timestamp (the time at which the price of crypto is recorded), the moving monthly average, and the moving weekly average. After this division, the data are further split into training and testing sets, with 20 percent of the data reserved for testing and the remaining 80 percent used for training. Once the data are split, we call the 'LinearRegression' method from scikit-learn on the training data, fit the model, and finally use the 'predict' method to produce the predicted results. The R^2 score, also known as the coefficient of determination, measures how well the regression predictions approximate the real data points. The higher the R^2 score (with a maximum of 1), the better the algorithm performs. In our project, the R^2 score was around 0.98, indicating a high level of accuracy, although this score does not necessarily mean the predictions are completely accurate. The predication of cryptocurrency price function uses a

variable called ‘dataframe_for_prediction‘, which contains the timestamp for the prediction day and the moving monthly and weekly averages for the preceding day, as specified by our algorithm. The model generates three different results, and the function returns the mean value of these three predictions, providing a single, consolidated forecast.

5.2.2. Ridge Regression

Ridge regression is a model tuning method used to analyze data suffering from multicollinearity, performing L2 regularization to address this issue. When multicollinearity occurs, least-squares estimates are unbiased but have large variances, leading to predicted values that deviate significantly from actual values. In machine learning, Ridge regression is employed when data overfitting occurs in simple or multiple linear regression models. Overfitting means the data is so closely aligned that even minor changes can cause the algorithm to overreact. Ridge regression mitigates overfitting by adding a penalty to large coefficient data, improving model performance. Ridge regression utilizes L2 regularization, distinct from L1 regularization. Regularization of L1 adds a penalty equal to the absolute value of the magnitude of the coefficients, which can result in the loss of some coefficients and inaccuracies. Conversely, L2 regularization adds a penalty equal to the square of the magnitude of the coefficients, making all coefficients smaller and producing better models. The advantages of Ridge regression include its ability to prevent overfitting by adding an L2 penalty and its simplicity, making it easier to implement compared to more complex models. Additionally, Ridge regression performs well even when the number of predictors in a dataset exceeds the number of explanatory variables, making it a robust choice for various predictive modeling tasks. Similar to the MLR algorithm, Ridge regression was implemented using the Python library Scikit-Learn. To train the model for prediction, we first determine which part of the dataset will be our explanatory variable and which part will be the response variable. The explanatory variables, including timestamp, monthly moving average, and weekly moving average, are stored in a variable represented by x, while the response variable, the price of the coin, is denoted by y. The dataset is further divided into two parts, one for training the Ridge Regression model and the other for testing our model against the predicted result. In Ridge Regression, only 30 percent of the data was used for testing, while the remaining 70 percent was used to train the model. A random number is required in Ridge regression to shuffle the data randomly, as indicated by setting the random state for the model to 40. After dividing the datasets, the Ridge model is called on the data. When the ridge model is called at line 18, the alpha is set to 50 and the tol is set to 0.1. The Alpha value determines the regularization strength, where setting Alpha to zero makes the ridge model equivalent to the linear regression model, and setting it to infinity makes the coefficients of the model zero. The tol parameter determines the precision of the outcome, with smaller values resulting in a more precise output.

Once the Ridge regression model is generated, the last step is to fit our data within the Ridge regression model. Then a function is used in the study to predict the price using the Ridge regression algorithm. The variable dataframe_for_prediction contains the timestamp for the time we want to make a prediction for, as well as the moving monthly and weekly averages of the day before. Similar to the linear regression model, the Ridge Regression model produces three different results based on three different sets of generated coefficients, and the mean of these results is returned.

5.2.3. Random Forest

Random forest is a supervised machine learning algorithm that finds wide application in both classification and regression problems. It constructs decision trees on different samples and aggregates their results through majority voting for classification and averaging for regression. It stands out as one of those rare algorithms versatile enough to serve as both a classification and regression tool, capable of handling both continuous and categorical variables. However, it typically excels more in classification tasks. At the heart of the random forest algorithm lies the use of decision trees. Decision trees are probability trees used to evaluate choices between multiple options. When decision making involves complex analysis with numerous potential outcomes, decision trees can help identify the optimal choice.

Their key advantage lies in their simplicity of interpretation and their flexibility to accommodate new scenarios. The Random Forest algorithm operates by generating decisions for the given dataset. It partitions the dataset into subsets and constructs a decision tree for each subset. The final decision is made by aggregating the results of all decision trees, either through majority voting in classification tasks or by averaging in regression tasks.

There are several advantages to using Random Forest over other algorithms. Firstly, it offers solutions for both classification and regression problems. Secondly, it addresses the issue of overfitting by basing the final output on majority voting, resulting in more robust predictions even with overfit data. Additionally, it handles null or zero values well due to its reliance on majority voting. The algorithm also demonstrates parallelization properties since each decision tree is independent, leading to stable output. Moreover, it tackles dimensionality issues efficiently by identifying the most relevant input variables, thus minimizing errors associated with multidimensional data. Despite its advantages, Random Forest has its drawbacks. The algorithm complexity increases with the production of numerous decision trees, potentially resulting in slower performance in some scenarios. In addition, users have limited control over its internal workings due to the lack of customizable parameters. Just like the other two algorithms utilized in the project, the Random Forest algorithm is also implemented using the Python library scikit-learn. We import the Random Forest model from scikit-learn by importing RandomForestRegressor from sklearn.ensemble. The basic workflow of the model closely resembles that of Multiple Linear Regression or Ridge Regression. Firstly, the data is divided into two main parts: one containing the explanatory variables and the other containing the response variable. After this initial split, the data is further divided for testing and training purposes. In the case of Random Forest, the test size is set to 0.3, meaning 30 percent of the data is reserved for testing while the remaining 70 percent is used for training. Once the data is split, we instantiate the RandomForestRegressor to initialize the model. In our model, n_estimators is set to 10, determining the number of trees the model should generate for the averaging of each output. The random_state parameter is set to 0, ensuring consistent results across different runs. Finally, after setting up the model, the last step involves fitting our training data into the model to train it to make predictions. The input structure remains the same as with Multiple Linear Regression and Ridge Regression algorithms, consisting of the timestamp for the day of prediction and the moving monthly and weekly averages for the preceding day. Since the model returns three outputs, the mean is used in the return statement to provide the mean value of the three produced outputs. The r^2 score evaluates how closely our sample aligns with the linear model. A higher r^2 score does not necessarily indicate the model is producing the best possible predictions; it is primarily a statistical comparison. Prediction accuracy can be further assessed using mean squared error or mean absolute error.

5.3. Frontend

In order to showcase the study results, a web-based application was developed using Python's Flask framework. The flask, being a popular micro-framework, was chosen for its versatility across different operating systems. Its simplicity, devoid of complex requirements such as databases, abstraction layers, or authentication methods, facilitated the development process. Six distinct pages were created to display the results. Flask served two main purposes: seamlessly incorporating Python code into web pages and simplifying deployment, particularly on platforms like PythonAnywhere. For the frontend design, HTML, CSS, and JavaScript were employed. HTML provided the basic structure for the front-end, with seven HTML files crafted during the project's duration. CSS was utilized to enhance the web application's aesthetics, as it is the standard language for web page design. Meanwhile, JavaScript, particularly with the assistance of ChartJS, a JavaScript library, was employed to add functionality. ChartJS proved to be powerful yet user-friendly for drawing various graphs, enriching the web interface.

6. Experiment Results

In this section, we present the results of our experiments, focusing on the performance of different machine learning models to predict cryptocurrency prices. We evaluated the models using Mean Squared Error (MSE) and Mean Absolute Error (MAE) as metrics, which are more appropriate for regression tasks compared to accuracy. An experiment is conducted using ML algorithms. The algorithms Multiple Linear Regression, Ridge Regression, and Random Forest are trained using the same dataset and then tested for prediction of coins prices. The predictions are evaluated using the accuracy of the predictions of the models, which were calculated using the actual prices of Bitcoin. The value was found to be 0.983% using multiple linear regression, 0.9669% using random forest, and 0.9555% using ridge regression. To draw graphs for cryptocurrencies, the ChartJS library of JavaScript was used. Figures 1,2,3, 4 and 5 show the graph drawn between the price of the coins and the dates. The predicted price was captured on 12/04/24 and the predicted price is the opening price on 12/05/24.

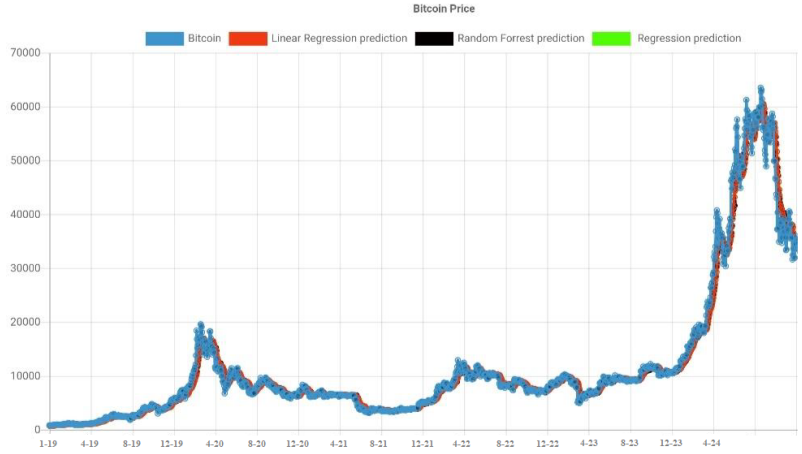

Figure 1: Bitcoin prediction page

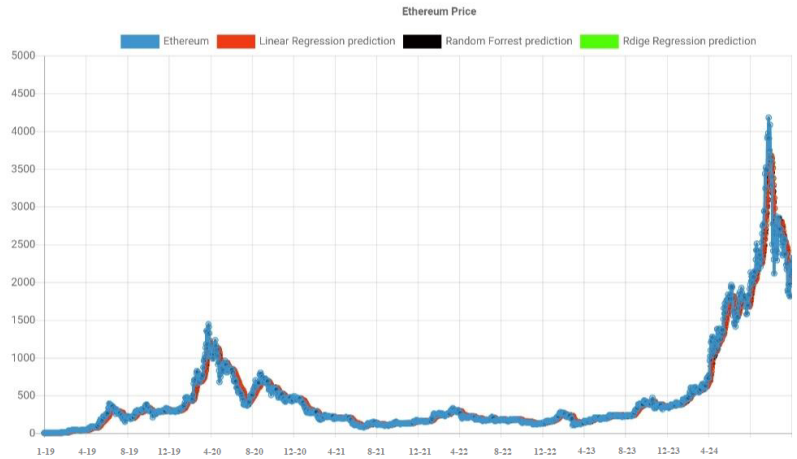

Figure 2: ethereum prediction page

In addition, we tested our models on the daily prices of five major cryptocurrencies. Bitcoin, Ethereum, Litecoin, Ripple, and Bitcoin Cash. The training data spanned five years, from 2017 to 2022, and the test data included prices from the most recent six months. Random Forest (RF) MSE (0.023) and MAE (0.015), Ridge Regression MSE (0.030) and MAE (0.018). Finally, multiple linear regression MSE (0.027) MAE (0.017). The testing process involved splitting the dataset into training and testing sets, ensuring that the models were trained on historical data and tested on unseen data. We used a sliding window approach

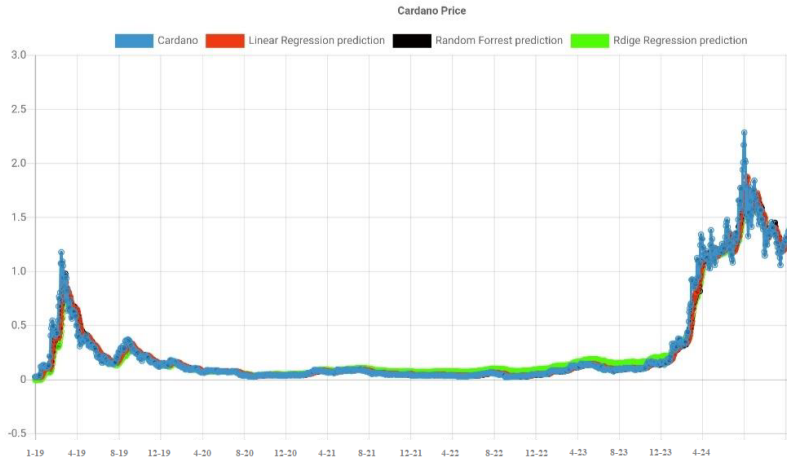

Figure 3: Cardano prediction page

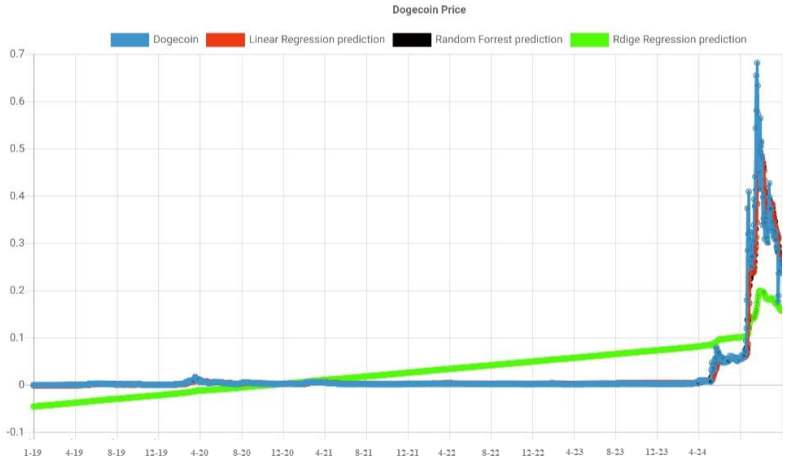

Figure 4: Dogecoin prediction page

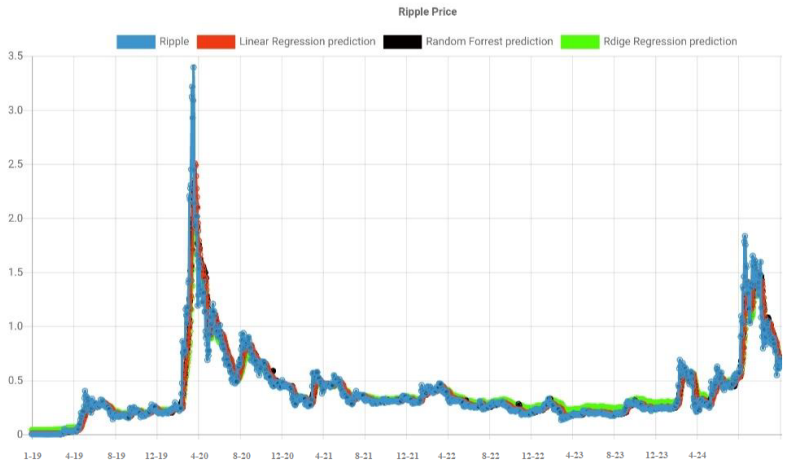

Figure 5: Ripple prediction page

to create training samples, where each sample consisted of a sequence of historical prices used to predict the next day's price. So, the results indicate that the Random Forest model outperformed other models in terms of both MSE and MAE. The Ridge Regression model also showed competitive performance. The

models were tested on all five cryptocurrencies, and the performance metrics presented are averages across all cryptocurrencies.

7. Conclusion

The primary objective of this study was to develop a model capable of predicting the next opening day price for a given cryptocurrency. Data for the five most popular cryptocurrencies was collected from the CoinGecko API. This dataset spans from the current day back to five years, or since the coin's inception if it is less than five years old. The prediction model considers three factors: the timestamp, the monthly moving average, and the weekly moving average, both of which were calculated using Python. To forecast the prices, three machine learning algorithms were employed: Multiple Linear Regression, Ridge Regression, and Random Forest, all implemented using the scikit-learn library. Models for each algorithm were created, and the data was fitted to these models to generate predictions. For the front-end, the Flask framework was used to create the web application. HTML, CSS, and JavaScript were utilized for designing the web interface, and ChartJS was used for visualizing cryptocurrency graphs. Finally, the study was deployed using PythonAnywhere, an integrated on-line development environment and web hosting service for Python applications. There are several areas for improvement that could be addressed in future work. Currently, the study supports predictions for only five cryptocurrencies. Future enhancements will aim to include all cryptocurrencies available on the CoinGecko website. Additionally, the current prediction model relies on limited factors, which may not be sufficient for maximum accuracy. Future work will focus on incorporating non-numerical factors that affect cryptocurrency prices and developing a model to account for these variables.

References

[1] D. Bhumichai, C. Smiliotopoulos, R. Benton, G. Kambourakis, D. Damopoulos, The convergence of artificial intelligence and blockchain: The state of play and the road ahead, Information 15 (2024) 268.

[2] A. Dhar Dwivedi, R. Singh, K. Kaushik, R. Rao Mukkamala, W. S. Alnumay, Blockchain and artificial intelligence for 5G-enabled Internet of Things: Challenges, opportunities, and solutions, Transactions on Emerging Telecommunications Technologies 35 (2024) e4329.

[3] J. Wu, X. Zhang, F. Huang, H. Zhou, R. Chandra, Review of deep learning models for crypto price prediction: implementation and evaluation, arXiv preprint arXiv:2405.11431 (2024).

[4] A. K. ORANGI, DRIVERS OF CONSUMER BEHAVIOR TOWARDS SECOND HAND CLOTHES IN KENYA, Ph.D. thesis, Kirinyaga University, 2024.

[5] M. Farouk, N. Shaker, D. S. AbdElminaam, O. Elrashidy, L. Mandour, M. Mesbah, J. Walid, M. Ahmed, R. Attia, N. Ahmed, et al., Bitcoin_ML: An efficient framework for bitcoin price prediction using machine learning, Journal of Computing and Communication 3 (2024) 70–87.

[6] S. S. Pashankar, J. D. Shendage, J. Pawar, Machine learning techniques for stock price prediction-a comparative analysis of linear regression, random forest, and support vector regression., Journal of Advanced Zoology 45 (2024).

[7] A. Ladhari, H. Boubaker, Deep learning models for bitcoin prediction using hybrid approaches with gradient-specific optimization, Forecasting 6 (2024) 279–295.

[8] J. Wu, X. Zhang, F. Huang, H. Zhou, R. Chandra, Review of deep learning models for crypto price prediction: implementation and evaluation, arXiv preprint arXiv:2405.11431 (2024).

[9] Q. Q. Abuein, M. Q. Shatnawi, E. Y. Aljawarneh, A. Manasrah, Time series forecasting model for the stock market using lstm and svr, Int. J. Advance Soft Compu. Appl 16 (2024).

[10] J. Cheng, S. Tiwari, D. Khaled, M. Mahendru, U. Shahzad, Forecasting bitcoin prices using artificial intelligence: Combination of ml, sarima, and facebook prophet models, Technological Forecasting and Social Change 198 (2024) 122938.

[11] R. Khanmohammadi, T. Alhanai, M. M. Ghassemi, The broad impact of feature imitation: Neural

enhancements across financial, speech, and physiological domains, arXiv preprint arXiv:2309.12279 (2023).
[12] N. Kulenović, "in bitcoin we trust": An anthropological approach to bitcoin as algorithmic utopia, Etnoantropološki problemi/Issues in Ethnology and Anthropology 19 (2024) 65–85.
[13] J. M. Kizza, Blockchains, cryptocurrency, and smart contracts technologies: Security considerations, in: Guide to Computer Network Security, Springer, 2024, pp. 575–600.
[14] N. Zhu, Y. Yang, W. Du, Y. Gan, J. He, Toward designing highly effective and efficient consensus mechanisms for blockchain-based applications, Cluster Computing (2024) 1–22.
[15] M. Thielen, Crypto Titans: How trillions were made and billions lost in the cryptocurrency markets, Markus Thielen, 2023.
[16] I. A. Rahman, T. Indrakusuma, A. Widodo, D. Nuryadin, Dogecoin price volatility after economic recovery on covid-19 pandemic, International Journal of Advanced Economics 5 (2023) 129–137.