The Relationship between Bitcoin, Nasdaq and U.S. Dollar Index

HO QUANG LAM[1], LE THI LE TRUC[2], AND NGUYEN THANH DAT[3]

[1]Faculty of Information Systems, University of Information Technology, (e-mail: 21521049@gm.uit.edu.vn)
[2]Faculty of Information Systems, University of Information Technology, (e-mail: 21521586@gm.uit.edu.vn)
[3]Faculty of Information Systems, University of Information Technology, (e-mail: 21521938@gm.uit.edu.vn)

ABSTRACT This paper investigates the long-run interaction between Bitcoin, Nasdaq and U.S. Dollar Index by applying weekly data from a January 3, 2013 until March 1, 2024.This study uses Linear Regression (LR), Long Short-Term Memory (LSTM), Autoregressive Integrated Moving Average (ARIMA), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), Gauss Newton Method Non-Linear (GNM), Bagging model, RESCNN methods to examine the long-run association between the variables.

INDEX TERMS Keywords: Bitcoin, Nasdaq, U.S. Dollar Index

I. INTRODUCTION

The development of money has been influenced by the evolving demands of human cultures and technological advancements. Over time, paper cash emerged to address the requirements of growing economies, and the transition from physical goods to plastic cards promoted faster transactions. With the advent of the electronic era, electronic cash systems were developed, enabling seamless and rapid transactions. However, the decentralized nature of Bitcoin, based on blockchain technology, is currently challenging well-established financial institutions. In recent years, the number of cryptocurrencies has exponentially increased, with Bitcoin being the dominant player. Understanding the relationship between Bitcoin and traditional financial indicators, such as the U.S. Dollar Index and the Nasdaq stock market index, is crucial for grasping its valuation and integration into the global financial system.

In the course of this research, we use Linear Regression (LR), Long Short-Term Memory (LSTM), Autoregressive Integrated Moving Average (ARIMA), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), Gauss Newton Method Non-Linear (GNM), Bagging model, RESCNN methods to examine the long-run association between the variables.

II. RELATED WORKS

Dwyer (2015) [1] delivers an influential paper demonstrating that BTC has higher average monthly volatility than gold or a group of international currencies. Urquhart (2016) [2], Nadarajah and Chu (2017) [3], and Bariviera (2017) [4] all corroborate this result by demonstrating BTC's inefficient returns.

Several studies have explored the relationship between Bitcoin, gold, and traditional currencies, such as the US dollar. Dyhrberg (2016) [5] suggests that Bitcoin can be a useful tool for risk management, especially for risk-averse investors who anticipate negative market shocks. Baur et al. (2018) [6] argue that Bitcoin exhibits different return, volatility, and correlation characteristics compared to gold and the US dollar, indicating its unique nature as an asset.

Dirican and Canoz (2017) [7] employed the ARDL boundary test approach to find a cointegration relationship between Bitcoin prices and the NASDAQ index, revealing hidden links underneath apparent discrepancies.

Studies have also examined the relationship between Bitcoin and equity markets, particularly during periods of uncertainty. The COVID-19 pandemic has acted as a catalyst for further research in this area. Quantile regression analysis conducted by Nguyen (2022) [8] revealed that during periods of high uncertainty, such as the COVID-19 crisis, the returns of the S&P 500 had a significant impact on Bitcoin returns. Additionally, stock market shocks had an effect on Bitcoin volatility during these years. This indicates that during times of heightened uncertainty, there is a stronger connection between the stock market and Bitcoin.

Several significant findings have emerged from studies examining the relationship between Bitcoin and the stock market. Wang et al. (2019) [9] found that the S&P 500 and Dow Jones indexes have a positive influence on Bitcoin, suggesting a favorable association between the cryptocurrency and the stock market. Maghyereh and Abdoh (2021) [10] discovered that Bitcoin and the US stock market exhibit positive co-movement at specific frequencies and time periods, indicating a potential interdependence between the two.

Additionally, Bouri et al. (2022) [11] demonstrated that the co-movement between US equities and Bitcoin changes over time and frequency, highlighting the dynamic nature of their interaction.

III. MATERIALS

A. DATASET

We get data on cryptocurrency prices from the Investing.com website with three datasets contains historical price data for three popular cryptocurrencies: Bitcoin, Nasdaq, U.S. Dollar Index and covers the time period from January 03, 2013 to March 1, 2024. Each dataset consists of 2023 rows and 7 columns include Date, Price, Open, High, Low, Vol., Change

Date: This column represents the date of the recorded data point. It provides the chronological information for each observation in the dataset.

Price: This column represents the closing price of the asset or security being analyzed (e.g., Bitcoin, stock, commodity) on a specific date. It indicates the value of the asset at the end of the trading day.

Open: This column represents the opening price of the asset on a specific date. It indicates the value of the asset at the beginning of the trading day.

High: This column represents the highest price reached by the asset during the trading day on a specific date. It provides insight into the peak value of the asset during that period.

Low: This column represents the lowest price reached by the asset during the trading day on a specific date. It provides insight into the lowest value of the asset during that period.

Vol. (Volume): This column represents the trading volume of the asset on a specific date. It indicates the total number of shares, contracts, or units of the asset that were traded during the trading day.

Change: This column represents the percentage change in the price of the asset compared to the previous trading day's closing price. It indicates the percentage increase or decrease in value between consecutive trading days.

B. DESCRIPTIVE STATISTICS

TABLE 1. US DOLLAR, NASDAQ, BITCOIN's Descriptive Statistics

	US DOLLAR	NASDAQ	BITCOIN
Mean	94.77986986	8291.193794	13062.80386
Standard Error	0.133999242	71.12929725	255.8327202
Median	95.754	7408.085	6636
Mode	104.182	13721.03	238.9
Standard Deviation	7.144811411	3743.591069	16218.66456
Sample Variance	51.0483301	14014474.09	263045080
Kurtosis	-0.037847286	-1.122949701	0.696153059
Skewness	-0.421644463	0.501032222	1.323476527
Range	34.921	13108.58	67493.6
Minimum	79.126	3166.36	34.3
Maximum	114.047	16274.94	67527.9
Sum	269459.17	22966606.81	52499408.7
Count	2843	2770	4019

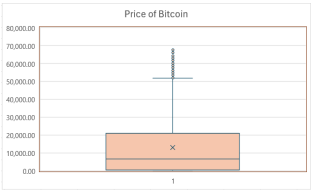

FIGURE 1. Bitcoin stock price's boxplot

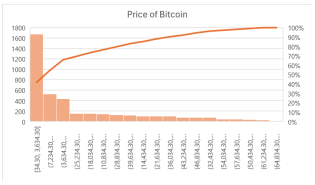

FIGURE 2. Bitcoin stock price's histogram

FIGURE 3. NASDAQ stock price's boxplot

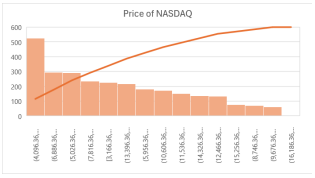

FIGURE 4. NASDAQ stock price's histogram

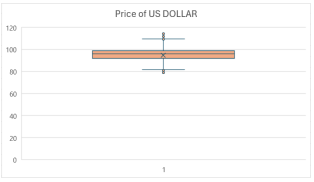

FIGURE 5. US DOLLAR stock price's boxplot

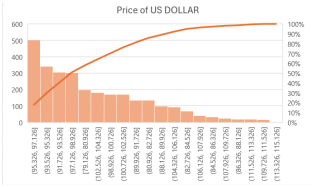

FIGURE 6. US DOLLAR stock price's histogram

IV. METHODOLOGY

A. LINEAR REGRESSION

Simple linear regression [12] estimates how much Y will change when X changes by a certain amount. With the correlation coefficient, the variables X and Y are inter- changeable. With regression, we are trying to predict the Y variable from X using a linear relationship (i.e., a line): A simple linear regression model has the form:

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

Where:

- Y is the dependent variable (Target Variable).
- X_1 is the independent (explanatory) variable.
- β_0 is the intercept term.
- β_1 is the regression coefficient for the independent variable.
- ε is the error term.

When there are multiple predictors, the equation is simply extended to accommodate them: A multiple linear regression model has the form:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \varepsilon$$

Where:

- Y is the dependent variable (Target Variable).
- $X_1, X_2, \ldots, X_k$ are the independent (explanatory) variables.

• β_0 is the intercept term.
• $\beta_1, ..., \beta_k$ are the regression coefficients for the independent variables.
• ε is the error term.

B. ARIMA

ARIMA [13] stands for AutoRegressive (AR) Integrated (I) Moving Average (MA) and represents a cornerstone in time series forecasting. It is a statistical method that has gained immense popularity due to its efficacy in handling various standard temporal structures present in time series data.

AR(p): Autoregression - is the process of finding the relationship between current data and p previous data (lag)

$$Y = \beta_0 + \beta_1 X_{t-1} + \beta_2 X_{t-2} + \cdots + \beta_k X_{t-k} + \varepsilon_t$$

Where:
• Y is current observed value.
• $X_{t-1}, X_{t-2}, \ldots, X_{t-k}$ are past observed values.
• β_0 is the intercept term.
• $\beta_1, ..., \beta_k$ are regression analysis parameters.
• ε_t random forecasting error of the current period. The expected mean value is 0.

I(d): Integrated - Compare the difference between d observations (difference between the current value and d previous values)
• First Difference I(1): z(t) = y(t) - y(t - 1)
• Second Difference I(2): h(t) = z(t) - z(t - 1)

MA(q): Moving Average: is the process of finding a relationship between current data and q past errors

$$y_t = \beta_0 + \varepsilon_t + \beta_1 \varepsilon_{t-1} + \ldots + \beta_q \varepsilon_{t-q}$$

Where:
• y(t) is current observed value.
• $\varepsilon_(t-1), \varepsilon_(t-2), \ldots, \varepsilon_(t-k)$ are forecast error.
• β_0 is the intercept term.
• $\beta_1, ..., \beta_k$ mean values of y(t) and moving average coefficients.
• $\varepsilon_(t)$ random forecasting error of the current period. The expected mean value is 0.
• q is the number of past errors used in the moving average.

C. RNN

Recurrent Neural Networks (RNNs) are a type of neural network that can be used to model sequential data. RNNs, which are built upon feedforward networks, exhibit behavior similar to the human brain. In simple terms, recurrent neural networks have the ability to anticipate sequential data in ways that other algorithms cannot.

In standard neural networks, all inputs and outputs are independent of one another. However, in certain scenarios, such as predicting the next word in a sentence, the previous words are necessary and must be remembered. As a result, RNNs were developed, utilizing a Hidden Layer to address this challenge. The most crucial component of an RNN is the Hidden State, which retains specific information about a sequence.

RNNs have a Memory that stores all information about the computations. It employs the same settings for each input, as it performs the same task on all inputs or hidden layers, producing the same outcome. [14]

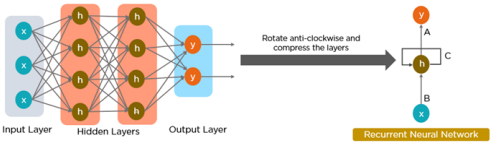

FIGURE 7. Recurrent Neural Network (RNN)

D. GRU

The Gated Recurrent Unit (GRU) [15] is a type of Recurrent Neural Network (RNN) that, in certain cases, has advantages over Long Short-Term Memory (LSTM). GRU uses less memory and is faster than LSTM, however, LSTM is more accurate when using datasets with longer sequences.

Additionally, GRUs address the vanishing gradient problem (the values used to update network weights) from which vanilla recurrent neural networks suffer. If the gradient shrinks over time as it back propagates, it may become too small to affect learning, thus making the neural network untrainable.

If a layer in a neural network cannot learn, RNNs can essentially "forget" longer sequences.

GRUs solve this problem through the use of two gates, the update gate and reset gate. These gates decide what information is allowed through to the output and can be trained to retain information from farther back. This allows it to pass relevant information down a chain of events to make better predictions.

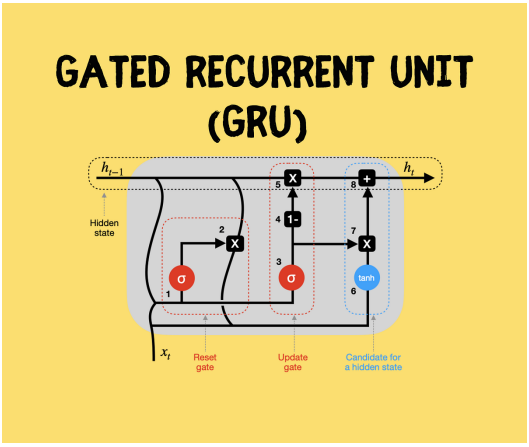

FIGURE 8. The Gated Recurrent Unit (GRU)

E. LSTM

Long Short-Term Memory (LSTM) [16] is a type of Recurrent Neural Network (RNN) that can retain long-term dependencies in sequential data. LSTMs are able to process and analyze sequential data, such as time series, text, and speech. They use a memory cell and gates to control the flow of information, allowing them to selectively retain or discard information as needed, and thus avoid the vanishing gradient problem that plagues traditional RNNs.

There are three types of gates in an LSTM: the input gate, the forget gate, and the output gate.

The input gate controls the flow of information into the memory cell. The forget gate controls the flow of information out of the memory cell. The output gate controls the flow of information out of the LSTM and into the output.

These three gates - the input gate, forget gate, and output gate - are all implemented using sigmoid functions, which produce an output between 0 and 1. These gates are trained using a backpropagation algorithm through the network.

The input gate decides which information to store in the memory cell. It is trained to open when the input is important and close when it is not.

The forget gate decides which information to discard from the memory cell. It is trained to open when the information is no longer important and close when it is.

The output gate is responsible for deciding which information to use for the output of the LSTM. It is trained to open when the information is important and close when it is not.

The gates in an LSTM are trained to open and close based on the input and the previous hidden state. This allows the LSTM to selectively retain or discard information, making it more effective at capturing long-term dependencies.

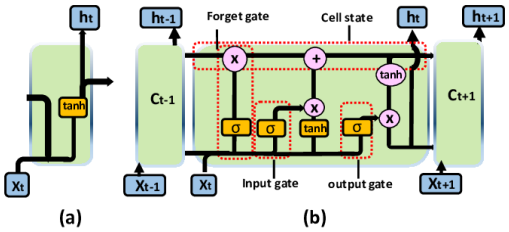

FIGURE 9. Long Short-Term Memory (LSTM)

F. GAUSS-NEWTON

The Gauss-Newton method is an optimization technique commonly used to evaluate parameters in nonlinear functions. It works by gradually decrementing functions and updating parameters in each loop to get closer to the optimal value.

$$\begin{pmatrix} \beta_{\text{new}} \\ \beta_1\text{new} \end{pmatrix} = \begin{pmatrix} \beta_{\text{bold}} \\ \beta_1\text{bold} \end{pmatrix} - (\mathbf{J}^T \cdot \mathbf{J})^{-1} \cdot \mathbf{J}^T \cdot \mathbf{r} \begin{pmatrix} \beta_{\text{bold}} \\ \beta_1\text{bold} \end{pmatrix} \quad (1)$$

Where:

- β_{0new}, β_{1new}, β_{0old}, β_{1old} is the value vector experience.
- J is mean partial derivative matrix (Jacobian matrix)
- r is risk function (residual function)

G. RANDOM FOREST

Random Forest [16] algorithm is a powerful tree learning technique in Machine Learning. It works by creating a number of Decision Trees during the training phase. Each tree is constructed using a random subset of the data set to measure a random subset of features in each partition. This randomness introduces variability among individual trees, reducing the risk of overfitting and improving overall prediction performance. In prediction, the algorithm aggregates the results of all trees, either by voting (for classification tasks) or by averaging (for regression tasks) This collaborative decision-making process, supported by multiple trees with their insights, provides an example stable and precise results. Random forests are widely used for classification and regression functions, which are known for their ability to handle complex data, reduce overfitting, and provide reliable forecasts in different environments.

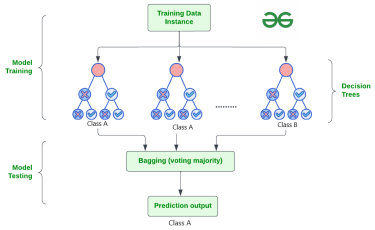

FIGURE 10. Random Forest Algortihm

H. RESCNN

Residual convolutional neural network (Res-CNN) is an architectural convolutional neural network designed to solve problems that arise when the network is too deep. Different from traditional Convolutional Neural Networks (CNN), Res-CNN adds collaborative blocks (residual blocks) to enhance the ability to learn deep privileges.

Specifically, Res-CNN starts with the first layer to receive data images. Next, the network uses convolutional layers to extract special input image words. However, unlike CNN, Res-CNN uses block communities consisting of 2 or 3 fast active layers, accompanied by hopping connections (skip connections). This connection helps transmit information from the input directly to the output of the block, avoiding information loss when the network is too deep.

Next, the network uses pooling layers to reduce the feature map's size, which reduces the number of parameters to learn. Finally, the fully connected layers will be used to perform classification based on the extracted features. The training process of Res-CNN uses a backpropagation algorithm to update the network parameters.

By combining slow distribution layers with residual blocks, Res-CNN can better learn special depths, thereby improving the performance of distributed CNN systems.

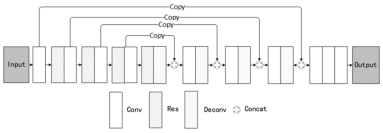

FIGURE 11. Residual convolutional neural network (Res-CNN)

V. RESULT

A. EVALUATION METHODS

Mean Percentage Absolute Error (MAPE): is the average percentage error in a set of predicted values.

$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} |y_i - \hat{y_i}| = 1$$

Root Mean Squared Error (RMSE): is the square root of average value of squared error in a set of predicted values.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y_i} - y_i)^2}{n}}$$

Mean Absolute Error (MSLE):is the relative difference between the log-transformed actual and predicted values.

$$MSLE = \frac{1}{n} \sum_{i=1}^{n} (log(1 + \hat{y_i}) - log(log(1 + y_i))^2$$

Where:

- n is the number of observations in the dataset.
- y_i is the true value.
- $\hat{y_i}$ is the predicted value.

B. BITCOIN DATASET

BITCOIN Dataset's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MSLE
LN	7:3	10508.77	10.71	0.015
	8:2	11729.2	10.825	0.019
	9:1	**7933.49**	**7.47**	**0.007**
ARIMA	7:3	11864.3	7.52	0.021
	8:2	8521.33	5.01	0.009
	9:1	**7006.54**	**3.73**	**0.006**
GRU	**7:3**	**1545.676**	**1.262**	**0.00033**
	8:2	1616.817	1.267	0.00035
	9:1	1699.655	1.052	0.00032
RNN	7:3	8620.284	8.559	0.01
	8:2	11729.2	10.825	0.019
	9:1	**7644.773**	**7.287**	**0.007**
LSTM	**7:3**	**7971.644**	**7.755**	**0.009**
	8:2	11711.484	10.809	0.019
	9:1	8629.708	8.253	0.009
GAUSS NEWTON	7:3	13156.831	13.336	0.021
	8:2	**7209.84**	**7.093**	**0.007**
	9:1	11945.338	11.444	0.016
RF	7:3	10949.0750	9.4738	0.0169
	8:2	11717.8586	10.8142	0.0189
	9:1	**6000.7953**	**5.2412**	**0.004**
RESCNN	7:3	941.7588	1.7384	0.0005
	8:2	939.7588	1.6546	0.0005
	9:1	**936.8374**	**1.6273**	**0.0005**

TABLE 2. BITCOIN Dataset's Evaluation

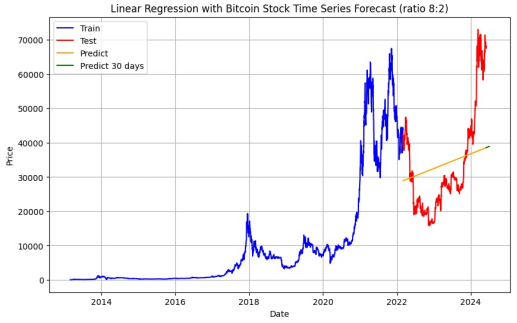

FIGURE 12. Linear model's result with 8:2 splitting proportion

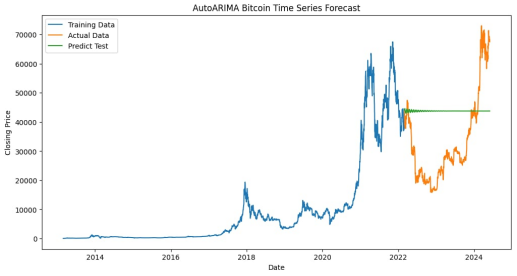

FIGURE 13. ARIMA model's result with 8:2 splitting proportion

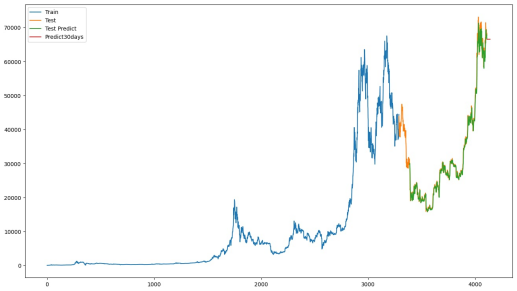

FIGURE 14. GRU model's result with 8:2 splitting proportion

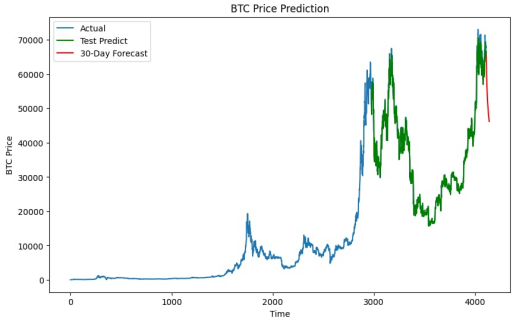

FIGURE 15. RNN model's result with 8:2 splitting proportion

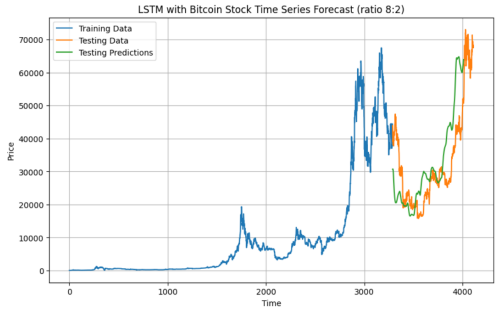

FIGURE 16. LSTM model's result with 8:2 splitting proportion

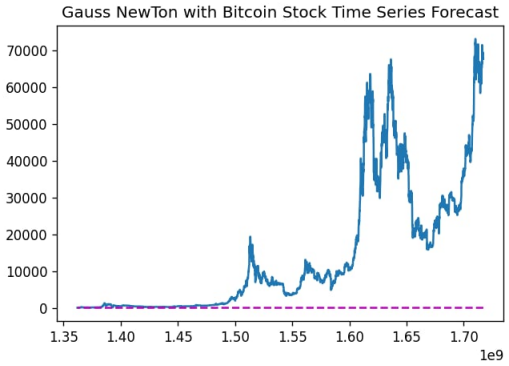

FIGURE 17. Gauss Newton model's result with 8:2 splitting proportion

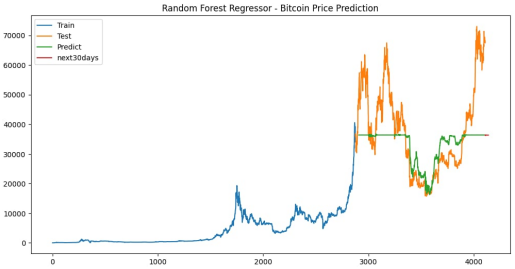

FIGURE 18. Random Forest model's result with 8:2 splitting proportion

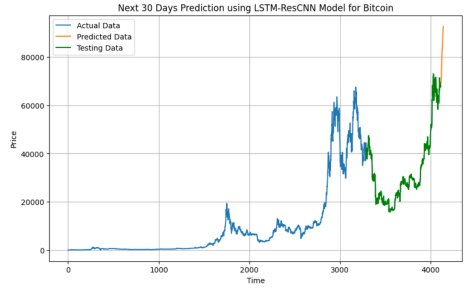

FIGURE 19. ResCNN model's result with 8:2 splitting proportion

C. USD DATASET

USD Dataset's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MSLE
LN	**7:3**	**4983.47**	**17.44**	**0.058**
	8:2	5293.6	26.28	0.063
	9:1	4894.46	25.85	0.055
ARIMA	7:3	977.55	1.76	0.002
	8:2	242.75	0.89	0.0002
	9:1	**162.85**	**0.75**	**0.00008**
GRU	7:3	454.9923	1.54	0.0005
	8:2	388.5658	1.406	0.0005
	9:1	**373.744**	**1.36**	**0.00038**
RNN	7:3	9682.514	43.586	0.161
	8:2	7136.268	36.166	0.106
	9:1	**1139.476**	**4.57**	**0.004**
LSTM	7:3	9693.439	43.648	0.162
	8:2	4564.211	23.154	0.05
	9:1	**1137.416**	**4.564**	**0.004**
GAUSS NEWTON	7:3	9428.531	41.483	0.154
	8:2	7054.485	34.819	0.102
	9:1	**1297.301**	**5.744**	**0.005**
RF	7:3	4988.1456	22.7511	0.0546
	8:2	4659.5801	23.6876	0.0516
	9:1	**1137.4155**	**4.5635**	**0.0036**
RESCNN	7:3	941.7588	1.7384	0.0005
	8:2	939.7588	1.6546	0.0005
	9:1	**936.8374**	**1.6273**	**0.0005**

TABLE 3. USD Dataset's Evaluation

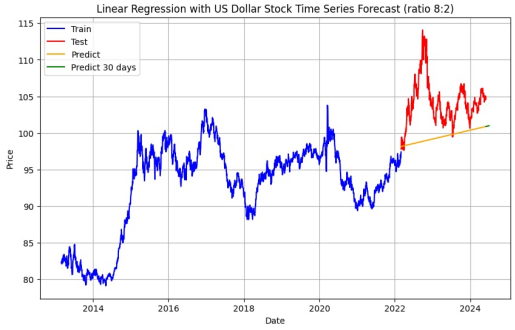

FIGURE 20. Linear model's result with 8:2 splitting proportion

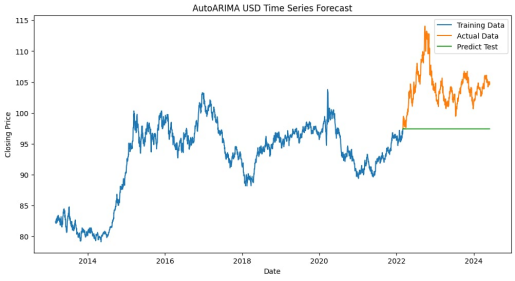

FIGURE 21. ARIMA model's result with 8:2 splitting proportion

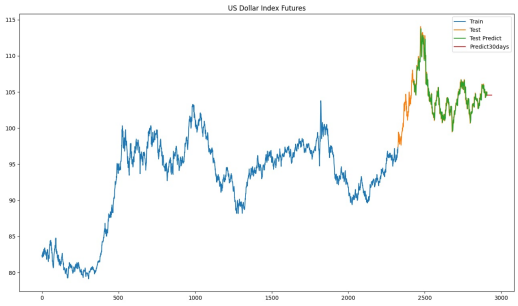

FIGURE 22. GRU model's result with 8:2 splitting proportion

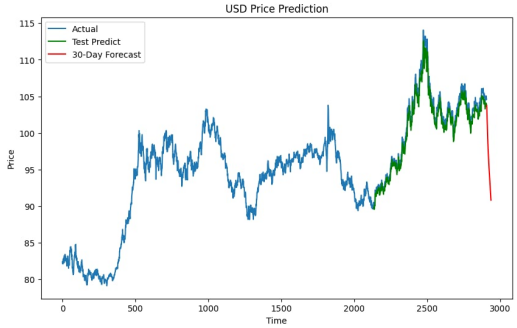

FIGURE 23. RNN model's result with 8:2 splitting proportion

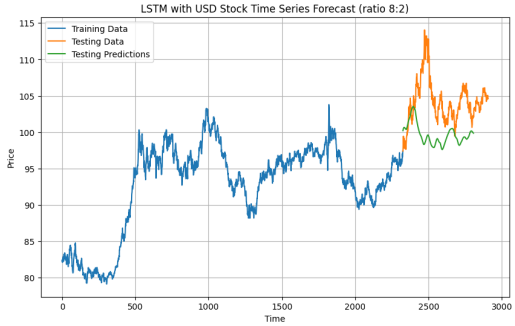

FIGURE 24. LSTM model's result with 8:2 splitting proportion

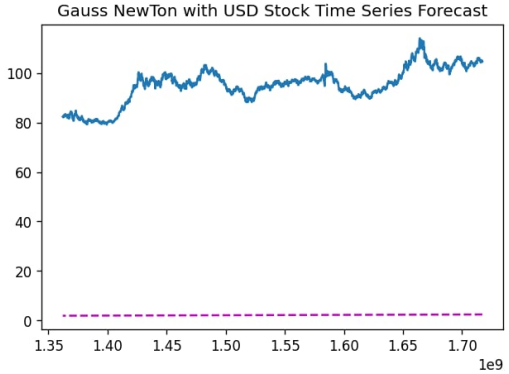

FIGURE 25. Gauss Newton model's result with 8:2 splitting proportion

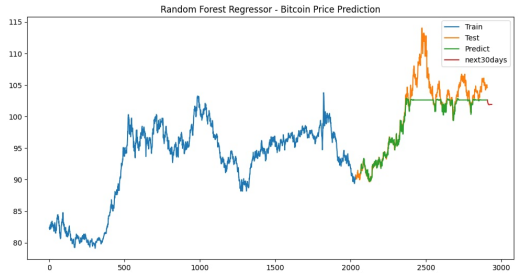

FIGURE 26. RF model's result with 8:2 splitting proportion

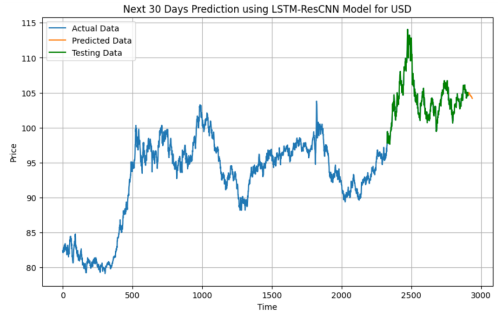

FIGURE 27. ResCNN model's result with 8:2 splitting proportion

D. NASDAQ DATASET

NASDAQ's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MSLE
LN	7:3	5690.9	13.03	0.021
	8:2	4904.44	10.28	0.016
	9:1	**2859.97**	**5.49**	**0.004**
ARIMA	7:3	5212.21	7.55	0.016
	8:2	1014.97	1.62	0.0005
	9:1	**822.63**	**1.26**	**0.0003**
GRU	7:3	916.692	1.67	0.00055
	8:2	948.341	1.74	0.00057
	9:1	**. 761.754**	**1.21**	**0.0003**
RNN	7:3	7847.594	15.278	0.041
	8:2	7501.223	15.14	0.036
	9:1	**3371.058**	**6.414**	**0.006**
LSTM	7:3	7849.75	15.29	0.04
	8:2	7501.73	15.15	0.04
	9:1	**3373.34**	**6.43**	**0.006**
GAUSS NEWTON	7:3	4288.68	8.641	0.012
	8:2	3771.703	7.756	0.009
	9:1	**3617.388**	**6.446**	**0.007**
RF	7:3	7849.6833	15.2872	0.0407
	8:2	7502.4992	15.1483	0.0357
	9:1	**3342.8102**	**6.3561**	**0.0057**
ResCNN	7:3	941.7588	1.7384	0.0005
	8:2	939.7588	1.6546	0.0005
	9:1	**936.8374**	**1.6273**	**0.0005**

TABLE 4. NASDAQ Dataset's Evaluation

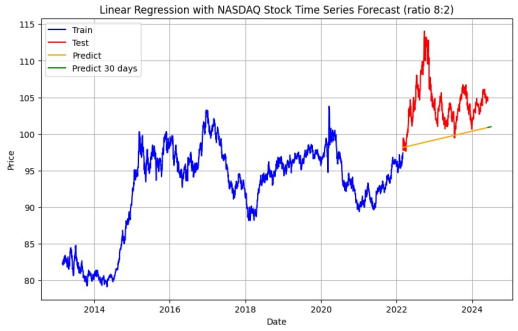

FIGURE 28. Linear model's result with 8:2 splitting proportion

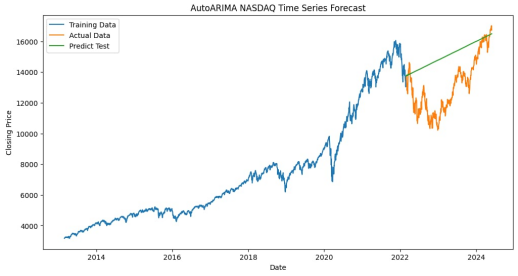

FIGURE 29. ARIMA model's result with 8:2 splitting proportion

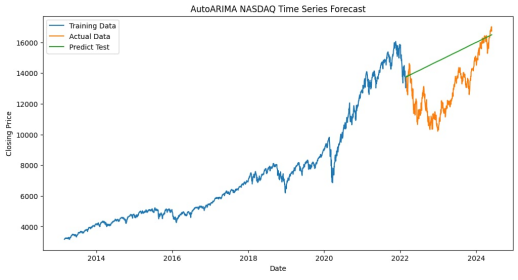

FIGURE 30. GRU model's result with 8:2 splitting proportion

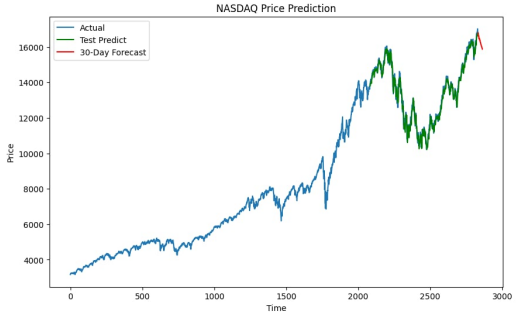

FIGURE 31. RNN model's result with 8:2 splitting proportion

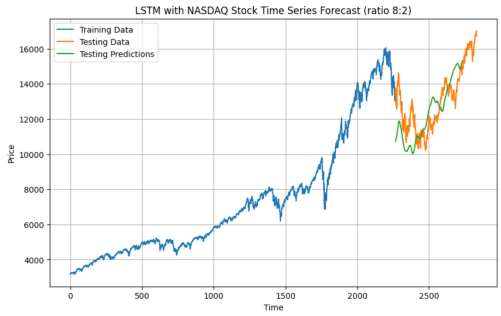

FIGURE 32. LSTM model's result with 8:2 splitting proportion

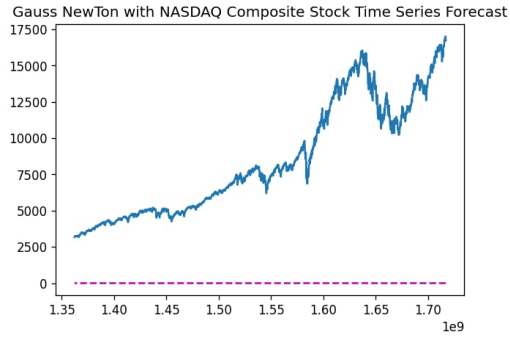

FIGURE 33. Gauss Newton model's result with 8:2 splitting proportion

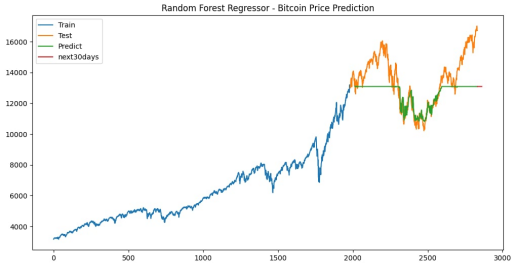

FIGURE 34. Random Forest model's result with 8:2 splitting proportion

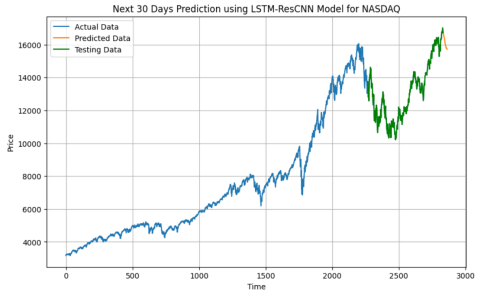

FIGURE 35. ResCNN model's result with 8:2 splitting proportion

VI. CONCLUSION

Kt lun mu —- Xóa dòng này

A. SUMMARY

In the achievement of forecasting stock prices, the exploration of diverse methodologies, ranging from traditional statistical models to advanced machine learning algorithms, has been aimed. Among the performed models, Linear Regression (LR), Auto Regressive Integrated Moving Average (ARIMA), Support Vector Regression (SVR), Seasonal Auto Regression Integrated Moving Average (SARIMA), Dynamic Linear Model (DLM), Bagging – GRU, and Simple Exponential Smoothing (SES), it becomes evident that Support Vector Regression (SVR), Gated Recurrent Unit (GRU), and Bagging GRU emerge as the most promising and effective models for predicting stock prices.

The intricacies of stock price forecasting, rooted in the complexity and unpredictability of financial markets, demand models that can capture nuanced patterns and relationships within the data. Support Vector Regression (SVR) showcases its efficacy in handling intricate relationships, providing robust predictions. Gated Recurrent Unit (GRU) models, with their ability to capture sequential dependencies, exhibit notable performance in forecasting stock prices. The introduction of ensemble learning through Bagging GRU further refines the predictive capabilities, offering a collective insight that surpasses individual models.

As evidenced by the evaluation metrics, including RMSE, MAPE, and MSLE, the SVR, GRU, and Bagging GRU models consistently demonstrate superior performance across various aspects of forecasting accuracy. Their adaptability to handle the inherent uncertainties of stock markets positions them as formidable tools for investors and analysts seeking reliable predictions.

B. FUTURE CONSIDERATIONS

In our future research, it is crucial to prioritize further optimization of the previously mentioned models. This optimization effort should specifically focus on:

• Enhancing the accuracy of the model. While the above algorithms have demonstrated promising results in predicting stock prices, there is a need to further improve the model's accuracy to ensure more precise forecasting outcomes.

• Exploring alternative machine learning algorithms or ensemble techniques. Ensemble techniques, such as combining multiple models or using various ensemble learning methods, can also improve the robustness and accuracy of the forecasts.

• Researching new forecasting models. The field of forecasting continuously evolves, with new algorithms and models being researched and developed. It is crucial to stay updated with these approaches and explore new forecasting models that offer improved accuracy and performance.

By continuously exploring and incorporating new features, data sources, and modeling techniques, we can strive for ongoing optimization of the forecasting models and enhance their ability to predict stock prices with greater precision and reliability.

ACKNOWLEDGMENT

First and foremost, we would like to express our sincere gratitude to **Assoc. Prof. Dr. Nguyen Dinh Thuan** and **Mr. Nguyen Minh Nhut** for their exceptional guidance, expertise, and invaluable feedback throughout the research process. Their mentorship and unwavering support have been instrumental in shaping the direction and quality of this study. Their profound knowledge, critical insights, and attention to detail have significantly contributed to the success of this research.

This research would not have been possible without the support and contributions of our mentors. We would like to extend our heartfelt thanks to everyone involved for their invaluable assistance, encouragement, and belief in our research. Thank you all for your invaluable assistance and encouragement.

REFERENCES

[1] Dwyer, G.P. (2015), The economics of Bitcoin and similar private digital currencies. Journal of Financial Stability, 17, 81-91.

[2] Urquhart, A. (2016), The inefficiency of Bitcoin. Economics Letters, 148, 80-82

[3] Nadarajah, S., Chu, J. (2017), On the inefficiency of Bitcoin. Economics Letters, 150, 6-9.

[4] Bariviera, A.F. (2017), The inefficiency of Bitcoin revisited: A dynamic approach. Economics Letters, 161, 1-4 Available:https://ieeexplore.ieee.org/document/7046047..

[5] Dyhrberg, A.H. (2016), Bitcoin, gold and the dollar-a GARCH volatility analysis. Finance Research Letters, 16, 85-92.

[6] Baur, D.G., Dimpfl, T., Kuck, K. (2018), Bitcoin, gold and the US dollar-a replication and extension. Finance Research Letters, 25, 103-110.

[7] Dirican, C., Canoz, I. (2017), The cointegration relationship between Bitcoin prices and major world stock indices: An analysis with ARDL model approach. Journal of Economics Finance and Accounting, 4(4), 377-392

[8] Nguyen, K.Q. (2022), The correlation between the stock market and Bitcoin during COVID-19 and other uncertainty periods. Finance Research Letters, 46, 102284.

[9] Shen, D., Urquhart, A., Wang, P. (2019), Does twitter predict Bitcoin? Economics Letters, 174, 118-122.

[10] Maghyereh, A., Abdoh, H. (2021), Time-frequency quantile dependence between Bitcoin and global equity markets. The North American Journal of Economics and Finance, 56, 101355

[11] Bouri, E., Kristoufek, L., Azoury, N. (2022), Bitcoin and SP500: Comovements of high-order moments in the time-frequency domain. PLoS One, 17(11), e0277924

[12] Peter Bruce, Andrew Bruce Peter Gedeck, "Practical Statistics for Data Scientists", 141-149

[13] Jason Brownlee (2023) , "How to Create an ARIMA Model for Time Series Forecasting in Python", Retrieved from $https : //machinelearningmastery.com/arima - for - time - series - forecasting - with - python/$

[14] Debasish Kalita (2024) , "A Brief Overview of Recurrent Neural Networks (RNN)", Retrieved from $https : //www.analyticsvidhya.com/blog/2022/03/a - brief - overview - of - recurrent - neural - networks - rnn/$

[15] (2023) , "Gated Recurrent Unit (GRU)". Retrieved from $https : //blog.marketmuse.com/glossary/gated - recurrent - unit - gru - definition/ : : text = TheMayankBanoula(2023), "IntroductiontoLongShort - TermMemory(LSTM)", Retrievedfromhttps : //www.simplilearn.com/tutorials/artificial - intelligence - tutorial/lstm : : text = LSTMs$

[16] "Random Forest Algorithm in Machine Learning". Retrieved from $https : //www.geeksforgeeks.org/random - forest - algorithm - in - machine - learning/$