**VIET NAM NATIONAL UNIVERSITY HO CHI MINH CITY**

**UNIVERSITY OF INFORMATION TECHNOLOGY**

**INFORMATION SYSTEM FACULTY**



**FINAL REPORT**

**BUSINESS ANALYSIS**

Instructor: Assoc.Prof. Nguyen Dinh Thuan

Nguyen Minh Nhut

Nguyen Thi Viet Huong

Student performance: Team 2

Nguyen Tan Phat 20521736

Lam Tra My 20521622

Nguyen Thi Thao Trang 20522040

**🙡🙢 Ho Chi Minh City, June 2023 🙠🙣**

# TEACHER’S COMMENTS

*……., date……...month……year 20…*

**Lecturer**

*(Sign and write full name****)***

TABLE OF CONTENTS

[TEACHER’S COMMENTS 2](#_Toc138186850)

[LIST OF TABLES AND FINGURES 4](#_Toc138186851)

[ABSTRACT 6](#_Toc138186852)

[I. INTRODUCTION 7](#_Toc138186853)

[II. RELATED WORK 8](#_Toc138186854)

[III. MATERIAL AND METHOD 10](#_Toc138186855)

[1. Dataset 10](#_Toc138186856)

[2. Tool used 13](#_Toc138186857)

[3. Evaluation forecasting models 14](#_Toc138186858)

[4. Artificial intelligence Models 15](#_Toc138186859)

[4.1. Linear Regression (LN) 15](#_Toc138186860)

[4.2. Autogressive Integrated Moving Average (ARIMA) 18](#_Toc138186861)

[4.3. Recurrent Neural Netwwork (RNN) 21](#_Toc138186862)

[4.4. Gated Recurrent Unit (GRU) 24](#_Toc138186863)

[4.5. Exponential Smoothing (ETS) 27](#_Toc138186864)

[4.6. Gradient Boosted Trees (GBT) 31](#_Toc138186865)

[4.7. Dynamic Linear Model (DLM) 34](#_Toc138186866)

[4.8. Extreme Gradient Boosting (XGBOOST) 37](#_Toc138186867)

[4.9. Deep Neural Network (DNN) 39](#_Toc138186868)

[IV. RESULT 42](#_Toc138186869)

[V. CONCLUSION 48](#_Toc138186870)

[JOB DISTRIBUTION 49](#_Toc138186871)

[REFERENCE 51](#_Toc138186872)

# LIST OF TABLES AND FINGURES

***List of tables***

[Table 1. Descriptive statistics of BTC 11](#_Toc138186803)

[Table 2. Descriptive statistics of ETH 12](#_Toc138186804)

[Table 3. Descriptive statistics of BNB 13](#_Toc138186805)

[Table 4. Evaluation of BTC 42](#_Toc138186806)

[Table 5. Evaluation of ETH 43](#_Toc138186807)

[Table 6. Evaluation of BNB 44](#_Toc138186808)

***List of Figures***

[Figure 1. Visualize data of BTC 10](#_Toc138186768)

[Figure 2. Visualize data of ETH 11](#_Toc138186769)

[Figure 3. Visualize data of BNB 12](#_Toc138186770)

[Figure 4. Result of LN model with the rate of 7-2-1 (%) 15](#_Toc138186771)

[Figure 5. Result of LN model with the rate of 6-2-2(%) 16](#_Toc138186772)

[Figure 6. Result of LN model with the rate of 5-3-2(%) 16](#_Toc138186773)

[Figure 7. Result of ARIMA model with the rate of 7-2-1(%) 18](#_Toc138186774)

[Figure 8. Result of ARIMA model with the rate of 6-2-2(%) 19](#_Toc138186775)

[Figure 9. Result of ARIMA model with the rate of 5-3-2(%) 19](#_Toc138186776)

[Figure 10. Operation diagram of RNN 20](#_Toc138186777)

[Figure 11. Construction of a node in a RNN 21](#_Toc138186778)

[Figure 12. Result of RNN model with the rate of 7-2-1(%) 22](#_Toc138186779)

[Figure 13. Result of RNN model with the rate of 6-2-2(%) 22](#_Toc138186780)

[Figure 14. Result of RNN model with the rate of 5-3-2(%) 23](#_Toc138186781)

[Figure 15. Construction of a node in a GRU 24](#_Toc138186782)

[Figure 16. Result of GRU model with the rate of 7-2-1(%) 25](#_Toc138186783)

[Figure 17. Result of GRU model with the rate of 5-3-2(%) 25](#_Toc138186784)

[Figure 18. Result of GRU model with the rate of 5-3-2(%) 26](#_Toc138186785)

[Figure 19. Result of ETS model with the rate of 7-2-1(%) 28](#_Toc138186786)

[Figure 20. Result of ETS model with the rate of 6-2-2(%) 29](#_Toc138186787)

[Figure 21. Result of ETS model with the rate of 5-3-2(%) 29](#_Toc138186788)

[Figure 22.Result of GBT model with the rate of 7-2-1(%) 32](#_Toc138186789)

[Figure 23. Result of GBT model with the rate of 6-2-2(%) 32](#_Toc138186790)

[Figure 24. Result of GBT model with the rate of 5-3-2(%) 33](#_Toc138186791)

[Figure 25. Result of DLM model with the rate of 7-2-1(%) 34](#_Toc138186792)

[Figure 26. Result of DLM model with the rate of 6-2-2(%) 35](#_Toc138186793)

[Figure 27. Result of DLM model with the rate of 5-3-2(%) 35](#_Toc138186794)

[Figure 28. A general architecture of XGBoost 36](#_Toc138186795)

[Figure 29. Result of XGBOOST model with the rate of 7-2-1(%) 37](#_Toc138186796)

[Figure 30. Result of XGBOOST model with the rate of 6-2-2(%) 37](#_Toc138186797)

[Figure 31. Result of XGBOOST model with the rate of 5-3-2(%) 38](#_Toc138186798)

[Figure 32. Architecture of Artificial Neural Network 39](#_Toc138186799)

[Figure 33. Result of DNN model with the rate of 7-2-1(%) 39](#_Toc138186800)

[Figure 34. Result of DNN model with the rate of 6-2-2(%) 40](#_Toc138186801)

[Figure 35. Result of DNN model with the rate of 5-3-2(%) 40](#_Toc138186802)

# ABSTRACT

Virtual currencies, also known as cryptocurrencies, have gained significant attention in recent years due to their potential for high returns and technological advancements. However, the volatility and unpredictability of these digital assets have presented challenges for investors and traders. To address these issues, the application of predictive methods has become essential in forecasting cryptocurrency prices. This scientific report focuses on predicting cryptocurrency prices using nine different forecasting models: Exponential Smoothing (ETS), Gradient Boosted Tree (GBT), Dynamic Linear Model (DLM), Deep Feedforward Neural Network (DNN), Extreme Gradient Boosting (XGBOOST), ARIMA, Linear Regression (LR), Recurrent Neural Network (RNN), and Gated Recurrent Unit (GRU). By comparing the performance of these nine models, this research aims to identify the most accurate and reliable approach for predicting cryptocurrency prices.

***Keywords*** *— cryptocurrency, virtual currency, Exponential Smoothing, Gradient Boosted Tree, Dynamic Linear Model, Deep Feedforward Neural Network, XGBOOST, ARIMA, Linear Regression, Recurrent Neural Network, Gated Recurrent Unit.*

# I. INTRODUCTION

Virtual currency, or cryptocurrency, has gained significant popularity in the era of technology 4.0, including in Vietnam. Its potential for high returns and technological innovation attracts individuals and financial institutions alike. However, the volatility and unpredictability of virtual currencies make it challenging for investors to make informed decisions and manage risks.

To address this issue, many researchers and projects worldwide, including in Vietnam, are focusing on virtual currency price prediction methods. These methods aim to provide accurate forecasts and information about the future price trends of virtual currencies. By utilizing models such as Exponential Smoothing, Gradient Boosted Tree, Dynamic Linear Model, Deep Feedforward Neural Network, Extreme Gradient Boosting, ARIMA, Linear Regression, Recurrent Neural Network, and Gated Recurrent Unit, researchers aim to develop powerful prediction tools to support investment decisions in the virtual currency market.

The objective of these projects is to provide investors and traders with reliable tools for decision-making based on accurate price forecasts. By analyzing data and evaluating the performance of different models using metrics like MAE, RMSE, and MAPE, researchers aim to identify the most accurate prediction model. Ultimately, the goal is to gain insights into virtual currency trends and price movements, enabling users to make smarter financial decisions and gain a competitive advantage.

# II. RELATED WORK

The following is a comprehensive summary of some of the related studies that we have reviewed, focusing on the topic of price prediction.

Reaz Chowdhury et al. conducted forecasts of the Cryptocurrencies Index 30 (cci30) from January 1st, 2015, to January 1st, 2017, using two variants of the Gradient Boosted Trees, Neural Network, Ensemble Learning Method, and K-NN models. In this case, the K-NN model did not work effectively compared to the other models [1].

V.Derbentsev et al. conducted forecasts for Bitcoin (BTC), Ethereum (ETH), and Ripple (XRP) from August 19th, 2013, until July 19th, 2016, using RNN (Recurrent Neural Network) and LSTM (Long Short-Term Memory) models. This study made adjustments to the units and epochs and employed MAPE (Mean Absolute Percentage Error) as the evaluation metric. The results indicate that the out-of-sample accuracy for short-term daily price forecasting using RNN and LSTM, in terms of MAPE, for three highly capitalized cryptocurrencies (BTC, ETH, and XRP), ranged from 0.92% to 2.61% [2].

LSTM neural networks were proposed by Hochreiter and Schmidhuber in 1997 to address the issue of learning and retaining information over long sequences. They introduced an efficient gradient-based method called LSTM, which incorporates memory cells and gating mechanisms. In a study by Krauss et al. (2017), different deep learning methods were compared for predicting stock performance. Random forests achieved the highest return of 0.43% per day, while LSTM networks in a subsequent study by Fischer & Krauss in 2018 outperformed all memory-free methods with a return of 0.46% per day. [3]

Taran Rishi conducted a "Stock Market Analysis Using Linear Regression" [4]. In the analysis, the residuals and p-value were calculated to assess the model's performance. The results indicated that the entire variation in the closing price of a stock could be accounted for by changes in the opening price, high price, low price, and volume of the stock. Hence, these variables were considered important in predicting closing prices. Specifically, the variables of opening, high, and low were found to be statistically significant, while volume was not deemed statistically significant within this model. To address multicollinearity, the author decided to exclude the high and low variables from the analysis. The reduced model demonstrated an R2 value of 0.9997, indicating that approximately 99.97% of the changes in a stock's closing price could be explained by alterations in the stock's opening price and volume. Consequently, both opening price and volume were identified as statistically significant variables in this model.

Time Series Analysis and Forecasting of Gold Price using ARIMA and LSTM Model; Research Dhruvi Sarvaiya and Disha Ramchandani, Department of Computer Engineering; Thadomal Shahani Engineering College, Mumbai, India; In this paper the authors have made use of various ARIMA models of permutations of p, d, q values to conclude that ARIMA model of order (1,1,2) as was deemed fit by the Augmented DickeyFuller test. and use LSTM model to improve accuracy by RNN with four layers of interaction structure [5].

Forecasting Gold Prices Using Multiple Linear Regression Method; Research Z. Ismail 1, A. Yahya 2 and A. Shabri 3; 1, 3 Department of Mathematics, Faculty of Science, 2 Department of Basic Education, Faculty of Education; University Technology Malaysia, 81310 Skudai, Johor Malaysia; In this paper using Linear Regression model to study gold price from London with some factors affecting gold price and factors used as independent variable [6].

# III. MATERIAL AND METHOD

## 1. Dataset

We get data on cryptocurrency prices from the Finance.yahoo website with three datasets contains historical price data for three popular cryptocurrencies: Bitcoin (BTC), Ethereum (ETH), Binance Coin (BNB) and covers the time period from December 01, 2017 to June 15, 2023. Each dataset consists of 2023 rows and 7 columns include Date, Open, High, Low, Close, Adj Close, Volume.

* Date: This column represents the date of the recorded cryptocurrency price data
* Open: The opening price of the cryptocurrency on that day
* High: The highest price reached by the cryptocurrency during the day
* Low: The lowest price reached by the cryptocurrency during the day
* Close: The closing price of the cryptocurrency on that day
* Adj Close: The adjusted closing price of the cryptocurrency on that day
* Volume: The trading volume of the cryptocurrency during the day

After collecting the data, we proceed to perform several preprocessing steps including data cleaning, feature selection, data reduction and data transformation to enhance the quality of the dataset. Divide the data into train set, test set and validate set with the ratio 7:2:1, 6:2:2 and 5:3:2. In this study, we visualized the 'Close' price data from three datasets (BTC, ETH, BNB) using the Matplotlib library in Python.

*A. Bitcon (BTC)*

Visualize data of BTC dataset:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Visualize data of BTC

Descriptive statistics of BTC dataset:

Table . Descriptive statistics of BTC

A picture containing text, screenshot, font, number

Description automatically generated

*B. Ethereum (ETH)*

Visualize data of BTC dataset:

A picture containing screenshot, text, plot, diagram

Description automatically generated

Figure . Visualize data of ETH

Descriptive statistics of BTC dataset:

Table . Descriptive statistics of ETH

A picture containing text, screenshot, font, number

Description automatically generated

*C. Binance Coin (BNB)*

Visualize data of BTC dataset:

A picture containing text, screenshot, plot, line

Description automatically generated

Figure . Visualize data of BNB

Descriptive statistics of BTC dataset:

Table . Descriptive statistics of BNB

A picture containing text, screenshot, font, number

Description automatically generated

## 2. Tool used

In this study, we use Python language, and the support tool is Google Colab. Besides, we use Python’s built-in libraries like Pandas to process data in the form of data frames. Matplotlib plots to visualize data illustrates the cryptocurrency's closing price data. Numpy helps with math and matrix operations in this experiment. And finally, the Scikit-learn library supports machine learning and regression models.

## 3. Evaluation forecasting models

To assess the accuracy of the models, we utilized three parameters: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE).

The smaller the three parameters, the better the performance of the algorithm.

* Mean Absolute Error (MAE)

Measures the average absolute difference between the predicted values and the actual values [7].

MAE =

With n is represents the number of observations,

is represents the actual value, is represents the predicted value.

* Root Mean Square Error (RMSE)

Is another commonly used metric to evaluate the accuracy of the models. It calculates the square root of the average of the squared differences between the predicted and actual values [8].

RMSE =

With n is represents the number of observations,

is represents the actual value, is represents the predicted value.

* Mean Absolute Percentage Error (MAPE)

MAPE measures the average percentage difference between the predicted and actual values [9].

MAPE =

With n is represents the number of observations,

is represents the actual value, is represents the predicted value.

## 4. Artificial intelligence Models

### 4.1. Linear Regression (LN)

Linear Regression is an algorithm of machine learning, based on supervised learning. Linear regression analysis is used to predict the value of a variable based on the value of another variable. The variable want to predict is called the dependent variable. The variable are using to predict the other variable's value is called the independent variable [10].

* Independent variables are the variables used to explain or predict the changes in the dependent variable. They are factors, characteristics, or conditions that are believed to have a relationship with the dependent variable.
* The dependent variable is the variable that depends on one or more other variables and is used to measure the changes in the independent variables. It is the outcome or phenomenon that we want to explain using the independent variables.

Formula:

y = β0 + β1 X1+ β2 X2 +….+ βn Xn + Ɛ [11]

Where:

* y is the dependent variable, representing the value to be predicted or explained.
* β0 is the intercept, representing the value of y when all the independent variables are zero.
* β1, β2, ..., βn are the coefficients corresponding to the independent variables X1, X2, ..., Xn. They represent the extent of the influence of each independent variable on the dependent variable.
* X1, X2, ..., Xn are the independent variables, which are the factors, characteristics, or conditions used to explain the changes in the dependent variable.
* ε is the error term, representing the unexplained variability of the linear regression model due to factors other than the independent variables.

Result of Linear Regression model with ratio 7-2-1:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of LN model with the rate of 7-2-1 (%)

Result of Linear Regression model with ratio 6-2-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of LN model with the rate of 6-2-2(%)

Result of Linear Regression model with ratio 5-3-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of LN model with the rate of 5-3-2(%)

### 4.2. Autogressive Integrated Moving Average (ARIMA)

ARIMA (Autoregressive Integrated Moving Average) model is one of the most popular time series analysis methods, used to predict future values based on past values of time series. ARIMA is used to model and predict non-stationary time series, i.e. trends, divergences, or trends in variance [12].

ARIMA consists of three main components:

* AR (Autoregression): The autoregressive part, which models the relationship between present and past values of a time series. The p-order delay is the backward value p time step of the sequence. Long or short delay in AR process depends on delay parameter p.

AR(p)=

* I (Integrated): Is the process of co-integration or taking the difference.

The general requirement of the algorithms in the time series is that the sequence must be stationary. To form a stationary series, one of the simplest methods is to take the difference.

I(1) = = -

I(d) = = ( ( …))

* MA (Moving Average): Moving average, which is the process of shifting or changing the mean of a series over time.

MA(q)=

Regression equations for ARIMA(p,d,q):

= + ... +

Conclude: ARIMA is a combined model of two autoregressive processes and moving average. Past data will be used to forecast future data. Before training the model, it is necessary to convert the series to a stationary series by taking the first difference or logarithm.

Result of ARIMA model with ratio 7-2-1:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of ARIMA model with the rate of 7-2-1(%)

Result of ARIMA model with ratio 6-2-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of ARIMA model with the rate of 6-2-2(%)

Result of ARIMA model with ratio 5-3-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of ARIMA model with the rate of 5-3-2(%)

### 4.3. Recurrent Neural Netwwork (RNN)

Recurrent Neural Network (RNN) is a type of artificial neural network designed for sequential data processing by maintaining internal memory, mainly used in speech recognition and natural language processing. (NLP). Designed to recognize patterns in data series, such as text, handwriting, speech and digital time series data emitted by sensors, stock markets and government agencies [13].

The basic building block of an RNN is a neuron with a self-loop connection, allowing it to receive its own output as an input for the next time step. The hidden state serves as the memory of the network and is updated at each time step as new inputs are processed.

A picture containing text

Description automatically generated

Figure . Operation diagram of RNN

A Neural Network usually includes 3 specific layers as follows:

* Input layer (x): The first layer contains input information.
* Hidden layer (ht): This is the most important layer in an RNN, where the network stores information internally. Each node in the hidden layer represents a hidden state at a specific time step and receives input from the input layer along with the hidden state of the previous node. This allows the RNN to propagate information in a time-dependent manner and retain information from previous time steps.
* Output layer: This is the final layer of the RNN where the prediction results are generated.

A picture containing diagram, text, screenshot

Description automatically generated

Figure . Construction of a node in a RNN

ht = g1(W \* ht-1 + U \* xt ) [14]

yt = g2(V \* ht )

* **xt**  represents the input at time step **t**.
* **ht** is the hidden state at step **t**. This is the memory of the network. **ht** is a combination of the previous hidden state **ht-1**and the input at time step **t** (**xt**). The activation function used is typically the hyperbolic tangent (**tanh**) or Rectified Linear Unit (**ReLU**).
* The output of each time step is **yt** . In one block of the RNN, there are two outputs.  **ht** represents the aggregated information from previous states to be passed along the network sequence, and we also have **yt** as the output for each time step.

Result of Recurrent Neural Network (RNN) model with ratio 7-2-1:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of RNN model with the rate of 7-2-1(%)

Result of Recurrent Neural Network (RNN) model with ratio 6-2-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of RNN model with the rate of 6-2-2(%)

Result of Recurrent Neural Network (RNN) model with ratio 5-3-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of RNN model with the rate of 5-3-2(%)

### 4.4. Gated Recurrent Unit (GRU)

GRU (Gated Recurrent Unit) is a neural network architecture used in natural language processing and time series prediction, aimed at solving the issue of vanishing information in traditional RNNs. It does so through the use of two gates: the reset gate and the update gate. Additionally, GRU is considered a variant of LSTM because both architectures are designed similarly and solve the same issue of vanishing information in RNNs. In general, a GRU cell consists of 4 components: [15]

* Reset gate: helps the model determine the amount of information in the past (information in step t-1) to be transferred to the future (step t).

* Update gate: This gate is used from the model to decide how much past information is to be transferred

* Current memory content: the current state of the network, calculated based on the input and the previous state of the network. It contains information about all the elements that have been processed in the input string so far.

* The final memory content represents the updated information output by the unit

Where:

* : input data
* : hidden state of the neural at time t-1
* : sigmod
* , , , : the weights of data
* : convolution multiplication

A picture containing diagram, rectangle, screenshot, line

Description automatically generated

Figure . Construction of a node in a GRU

Result of GRU model with ratio 7-2-1:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of GRU model with the rate of 7-2-1(%)

Result of GRU model with ratio 6-2-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of GRU model with the rate of 5-3-2(%)

Result of GRU model with ratio 5-3-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of GRU model with the rate of 5-3-2(%)

### 4.5. Exponential Smoothing (ETS)

ETS (Exponential Smoothing) is a method for forecasting univariate time series data. It works by assigning exponentially increasing weights to past observations. That is, the most recent past data will be given more weight than the more distant past data. And the model assumes that the future will somewhat resemble the recent past. Often used to make forecasts of time series data based on previous assumptions made by the user, such as seasonality (cyclical repetition) or propensity (rising/decreasing over time)[16]. Divided into 3 types: [17]

* Simple Exponential Smoothing

It is a time series forecasting method used with univariate data with no trends and no seasonal patterns

It takes a single parameter called alpha(), also known as the smoothing factor.

Alpha controls the rate at which the influence of past observations decreases exponentially. The parameter is usually set to a value between 0 and 1.

Where:

* : smoothed statistic, it is the simple weighted average of the current observation xt
* : actual value of the previous month
* : previous smoothed statistics (previous month results)
* α : data smoothing coefficient ( 0 < α < 1)
* t : time period
* Double Exponential Smoothing

Used for time series forecasting when the data tends to be linear and there is no seasonal pattern.

The coefficient is called beta(), which controls the fading effect of the change in trend.

For t > 1

Where:

* : best estimate of the trend at time t
* β : trend smoothing coefficient; 0 < β < 1
* Triple Exponential Smoothing

In this method, exponential smoothing is applied three times. This method is used for time series forecasting when the data has both linear trends and seasonal patterns. This method is also known as Holt-Winters exponential smoothing.

There are two types depending on the nature: Additive Seasonality and Multiplicative Seasonality

Where:

: sequence of seasonal correction factor at time t

γ : the smoothing coefficient varies by season; 0 < γ < 1

Result of ETS model with ratio 7-2-1:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of ETS model with the rate of 7-2-1(%)

Result of ETS model with ratio 6-2-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of ETS model with the rate of 6-2-2(%)

Result of ETS model with ratio 5-3-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of ETS model with the rate of 5-3-2(%)

### 4.6. Gradient Boosted Trees (GBT)

Boosting Techniques:

* Instead of trying to build a single good model, we will build a set of slightly weaker models, but combining the models will result in a superior model [18].
* The basic idea is that Boosting will generate a series of weak models, learning from each other, models that are trained sequentially. In other words, in Boosting, the following models will try to learn to limit the mistakes of the previous models .

Gradient boosted trees: Is a machine learning model used to solve regression and classification problems. It is a form of Ensemble Learning in which a large number of simple decision trees are combined to create a more powerful predictive model [19].

Algorithm idea:

* Combine multiple simple decision trees together to create a more complex and more predictive model. During training, the model generates new decision trees sequentially and focuses on the data points that the previous trees predicted wrong. This is done by computing the gradient of the loss function at those data points and using it to generate a new decision tree. Each new decision tree is designed to minimize the loss function for those data points.
* Once decision trees have been generated, GBT uses them to generate predictions by computing a weighted average of the predictions of those decision trees. Weights are assigned to each decision tree based on its performance in predicting the training set.

Steps to implement the algorithm [20].

* Input: traning set , a differentiable loss function L(y,F(x)), number of iterations M.
* Step 1: Build model with a constant value

= arg min

With is our predicted value, argmin means we have to find a predicted value/gamma for which the loss function is minimum.

* Step 2: Compute so-called pseudo-residuals

For m = 1 to M:

=for i = 1,...n

With is the previous model

* Step 3: Fit a base learner (or weak learner, e.g. tree) closed under scaling to pseudo-residuals, i.e. train it using the training set
* Step 4: Compute multiplier by solving the following one-dimensional optimization problem

= arg min

* Step 5: Update the model

+

With is the prediction of the base model (previous prediction), m is the number of decision trees made.

Result of GBT model with ratio 7-2-1:

A picture containing text, screenshot, diagram, plot

Description automatically generated

Figure .Result of GBT model with the rate of 7-2-1(%)

Result of GBT model with ratio 6-2-2:

A picture containing text, screenshot, diagram, plot

Description automatically generated

Figure . Result of GBT model with the rate of 6-2-2(%)

Result of GBT model with ratio 5-3-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of GBT model with the rate of 5-3-2(%)

### 4.7. Dynamic Linear Model (DLM)

Dynamic Linear Models (DLMs) or state space models define a very general class of non-stationary time series models. May include terms to model trends, seasonality, covariates and autoregressive components. Other time series models like ARMA models are particular DLMs. The main goals are short-term forecasting, intervention analysis and monitoring. [21]

t = 1, 2, . . . , T

is the observation at time t. We assume this is to be a scalar but could also be a vector.

is the vector of parameters at time t and of dimension p × 1

is the row vector (dimension 1 × p) of covariates at time t

is a matrix of dimension p × p known as evolution or transition matrix.

Result of DLM model with ratio 7-2-1:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of DLM model with the rate of 7-2-1(%)

Result of DLM model with ratio 6-2-2:

A picture containing screenshot, text, plot, diagram

Description automatically generated

Figure . Result of DLM model with the rate of 6-2-2(%)

Result of DLM model with ratio 5-3-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of DLM model with the rate of 5-3-2(%)

### 4.8. Extreme Gradient Boosting (XGBOOST)

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. The same code runs on major distributed environment (Hadoop, SGE, MPI) and can solve problems beyond billions of examples. [22]

Supervised machine learning uses algorithms to train a model to find patterns in a dataset with labels and features and then uses the trained model to predict the labels on a new dataset’s features.

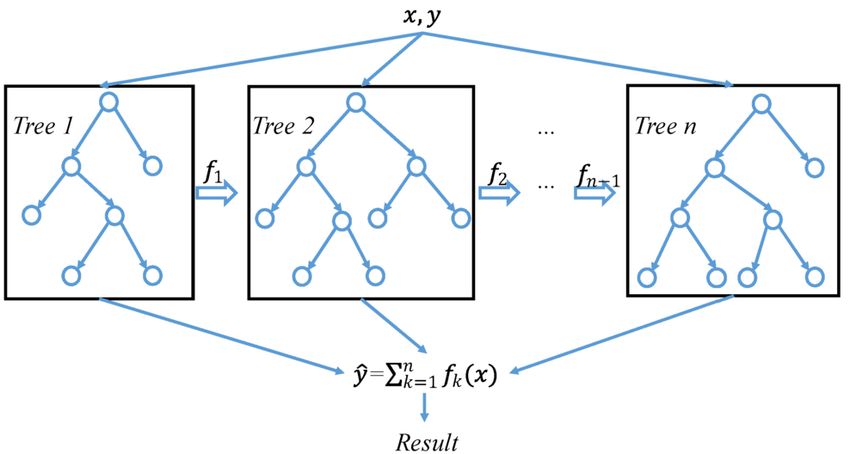


Figure . A general architecture of XGBoost

XGBoost is a scalable and highly accurate implementation of gradient boosting that pushes the limits of computing power for boosted tree algorithms, being built largely for energizing machine learning model performance and computational speed. With XGBoost, trees are built in parallel, instead of sequentially like GBDT. It follows a level-wise strategy, scanning across gradient values and using these partial sums to evaluate the quality of splits at every possible split in the training set. [23]

Result of XGBOOST model with ratio 7-2-1:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of XGBOOST model with the rate of 7-2-1(%)

Result of XGBOOST model with ratio 6-2-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of XGBOOST model with the rate of 6-2-2(%)

Result of XGBOOST model with ratio 5-3-2:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of XGBOOST model with the rate of 5-3-2(%)

### 4.9. Deep Neural Network (DNN)

Neural network is a mathematical model inspired by the way neurons in the human nervous system work. This model uses computational units called artificial neurons to process input and generate output based on weights and activation functions.

Neural networks in deep learning are often built with many interconnected neural layers. The layers of neurons include:

* Input layer (x) : Receives input data and forwards them to the next layer of neurons.
* Hidden layers: The layers of neurons between the input layer and the output layer. Hidden layers are responsible for processing information and learning complex representations from input data.
* Output layer (ŷ): Make predictions or model results based on the processing of previous layers of neurons.

DNN is a neural network with at least one hidden layer. It provides modelling for complicated nonlinear functions and has a high-level abstraction ability, which means that the fitting power of the model is significantly improved. Meanwhile, it is a kind of discriminant model which could be trained through the backpropagation algorithm.

The model architecture determines the complexity and expressivity of the model. By adding hidden layers and non-linear activation functions (for example, relu), the model can capture more complex relationships in the data.

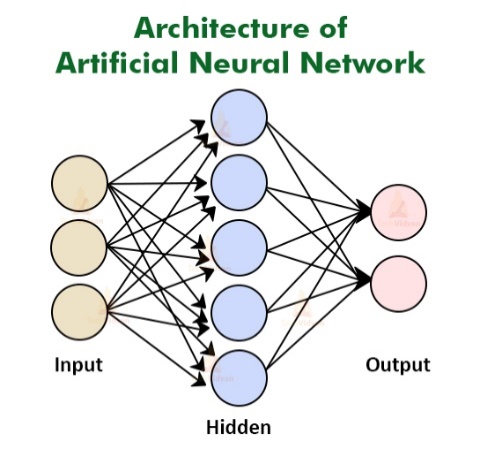


Figure . Architecture of Artificial Neural Network

Result of DNN model with ratio 7-2-1:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure . Result of DNN model with the rate of 7-2-1(%)

Result of DNN model with ratio 6-2-2:

A picture containing text, screenshot, diagram, plot

Description automatically generated

Figure . Result of DNN model with the rate of 6-2-2(%)

Result of DNN model with ratio 5-3-2:

A picture containing screenshot, text, diagram, plot

Description automatically generated

Figure . Result of DNN model with the rate of 5-3-2(%)

# IV. RESULT

In this study, we use 3 parameters RMSE, MAPE, MAE to evaluate the accuracy and suitability of the algorithms on each data set. The following table are the results of nine models for train, test, validate with ratio 7:2:1, 6:2:2, 5:3:2 on each dataset we got it:

*a) With BTC dataset:*

Table . Evaluation of BTC

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Proportion** | **RMSE** | **MAPE** | **MAE** |
| **LN** | 7:2:1 | 18199.80 | 60.32% | 15330.43 |
| 6:2:2 | 24380.44 | 46.91% | 22519.83 |
| 5:3:2 | 34883.84 | 73.58% | 31713.88 |
| **ARIMA** | 7:2:1 | 32595.75 | 118.86% | 29024.89 |
| 6:2:2 | 104868.07 | 212.94% | 92179.23 |
| 5:3:2 | 33137.42 | 67.34% | 29769.88 |
| **RNN** | 7:2:1 | 30642.19 | 7538484.22% | 29069.74 |
| 6:2:2 | 48275.47 | 7250867.61% | 47775.01 |
| 5:3:2 | 42279.77 | 6457462.13% | 41597.23 |
| **GRU** | 7:2:1 | 30367.53 | 7489570.53% | 28840.27 |
| 6:2:2 | 46649.42 | 69345.19% | 45949.36 |
| 5:3:2 | 45165.85 | 68107.95% | 44198.49 |
| **ETS** | 7:2:1 | 305953.67 | 1107.04% | 266019.92 |
| 6:2:2 | 263080.61 | 525.51% | 229681.86 |
| 5:3:2 | 69950.81 | 148.08% | 63320.91 |
| **GBT** | 7:2:1 | **908.43** | **2.05%** | **659.15** |
| 6:2:2 | 1682.52 | 2.77% | 1294.74 |
| 5:3:2 | 25580.44 | 47.19% | 22007.09 |
| **DLM** | 7:2:1 | 8059.21 | 29.37% | 7194.89 |
| 6:2:2 | 9575.23 | 20.62% | 8463.72 |
| 5:3:2 | 3077.5 | 5.92% | 2620.04 |
| **XGBOOST** | 7:2:1 | 1075.29 | 2.63% | 807.39 |
| 6:2:2 | 1671.57 | 2.64% | 1256.22 |
| 5:3:2 | 25857.36 | 47.91% | 22293.19 |
| **DNN** | 7:2:1 | **1031.05** | **2.23%** | **754.13** |
| 6:2:2 | 2377.01 | 4.51% | 2098.72 |
| 5:3:2 | 2426.28 | 4.65% | 2015.89 |

After building the models and evaluating the results on the test set with three ratios, 7-2-1, 6-2-2 and 5-3-2, for the nine main models: ETS, GBT, DLM, DNN, XGBOOST, ARIMA, Linear, RNN, and GRU.

For the parameter MAE evaluation of the closing price of the BTC set. Based on the results of Table 2, we can see that the GBT model has the lowest MAE value with 659.15 for the 7-2-1 ratio and 1294.74 for the 6-2-2 ratio and 22007.09 for the 5-3-2. Next is the DNN model with 754.13 for 7-2-1 and 2098.72 for 6-2-2 and 2015.89 for 5-3-2. The remaining models all have MAE values larger than GBT and DNN models.

For the parameter RMSE evaluation of the closing price of the BTC set. Based on the results of Table 2, we can see that the GBT model has the lowest RMSE value with 908.43 for the 7-2-1 ratio and 1682.52 for the 6-2-2 ratio and 25580.44 for the 5-3-2 ratio. Next is the DNN model with 1031.05 for the 7-2-1 ratio and 2377.01 for the 6-2-2 ratio and 2426.28 for the 5-3-2 ratio. The remaining models all have larger RMSE values.

The last parameter is the MAPE evaluation parameter of the closing price of the BTC set. Based on the results of Table 2, we can see that the GBT model has the lowest MAPE value with 2.05%, followed by the DNN model with 2.23% for the 7-2-1 ratio. With a ratio of 6-2-2, the XGBOOST model has the lowest MAPE value with 2.64%, followed by the GBT model with 2.77%. With a ratio of 5-3-2, the DNN model has the lowest MAPE value with 4.65%, followed by the DLM model with 5.92%. The remaining models all have larger MAPE values.

In general, through model evaluation parameters such as MAE, RMSE and MAPE, the GBT model almost gives the results with the smallest value, followed by the DNN model, then the XGBOOST model. From the above conclusion, we can see that the DNN and GBT models are the two most accurate models with the data set of BTC and are suitable for predicting.

*b) With ETH dataset:*

Table . Evaluation of ETH

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Proportion** | **RMSE** | **MAPE** | **MAE** |
| **LN** | 7:2:1 | 1334.19 | 45.22 | 1107.83 |
| 6:2:2 | 2653.78 | 81.32 | 2553.7 |
| 5:3:2 | 2788.64 | 101.54 | 2464.26 |
| **ARIMA** | 7:2:1 | 1745.79 | 92.91 | 1505.22 |
| 6:2:2 | 1034.9 | 27.47 | 818.75 |
| 5:3:2 | 2368.28 | 73.37 | 2022.53 |
| **RNN** | 7:2:1 | 2153.98 | 509173.92 | 2028.46 |
| 6:2:2 | 3231.78 | 47.15112.89 | 3176.12 |
| 5:3:2 | 2730.95 | 466239.63 | 2594.67 |
| **GRU** | 7:2:1 | 2126.75 | 5037.37 | 2004.48 |
| 6:2:2 | 3293.22 | 477850.02 | 3228.31 |
| 5:3:2 | 2861.25 | 480939.59 | 2700.23 |
| **ETS** | 7:2:1 | 15231.8 | 797.63 | 13114.76 |
| 6:2:2 | 5131.85 | 138.39 | 4214.01 |
| 5:3:2 | 8243.17 | 289.48 | 7188.45 |
| **GBT** | 7:2:1 | **191.49** | **3.32** | **99.35** |
| 6:2:2 | 1357.33 | 33.56 | 1151.42 |
| 5:3:2 | 1546.73 | 38.31 | 1200.86 |
| **DLM** | 7:2:1 | 637.4 | 33.59 | 550.25 |
| 6:2:2 | **448.44** | **10.82** | **371.26** |
| 5:3:2 | **203.29** | **6.26** | **165.08** |
| **XGBOOST** | 7:2:1 | 208.48 | 3.59 | 108.7 |
| 6:2:2 | 1384.27 | 34.53 | 1180.95 |
| 5:3:2 | 1565.49 | 38.79 | 1216.5 |
| **DNN** | 7:2:1 | **78.86** | **2.48** | **57.7** |
| 6:2:2 | **134.19** | **3.39** | **108.03** |
| 5:3:2 | **84.81** | **2.47** | **58.53** |

Based on the evaluation parameters MAE, RMSE, and MAPE for the closing price of the ETH dataset, we can analyze the results presented in Table.

For the MAE evaluation, the DNN model shows the lowest MAE values of 57.7 for the 7-2-1 ratio, 108.03 for the 6-2-2 ratio and 58.53 for the 5-3-2 ratio. The GBT model follows with MAE values of 99.35 for the 7-2-1 ratio and the DNN model follows with MAE values of 371.26 for 6-2-2 and 165.08 for 5-3-2. Other models in the evaluation have higher MAE values.

For the RMSE evaluation, the DNN model again exhibits the lowest values with 78.86 for the 7-2-1 ratio, 134.19 for the 6-2-2 ratio and 84.81 for the 5-3-2 ratio. The GBT model follows with RMSE values of 191.49 for the 7-2-1 ratio and the DNN model follows with RMSE values of 448.44 for 6-2-2 and 203.29 for the 5-3-2 ratio. Other models have higher RMSE values.

Regarding the MAPE evaluation, the DNN model shows the lowest MAPE values with 2.48% for the 7-2-1 ratio, while the GBT model follows with 3.32%. For the 6-2-2 ratio, the DNN model continues to have the lowest MAPE value of 3.39%, followed by the DLM model with 10.82%. For the 5-3-2 ratio, the DNN model continues to have the lowest MAPE value of 2.47%, followed by the DLM model with 6.26%. Other models have larger MAPE values.

In general, based on the evaluation parameters MAE, RMSE, and MAPE, the DLM and DNN models consistently provide lower values compared to the other models. Therefore, it can be concluded that the DLM and DNN models are suitable with the ETH dataset.

*c) With BNB dataset:*

Table . Evaluation of BNB

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Proportion** | **RMSE** | **MAPE** | **MAE** |
| **LN** | 7:2:1 | 158.81 | 28.14 | 119.8 |
| 6:2:2 | 379.22 | 82.97 | 366.64 |
| 5:3:2 | 350.45 | 76.84 | 293.26 |
| **ARIMA** | 7:2:1 | 144.7 | 41.71 | 128.35 |
| 6:2:2 | 269.37 | 52.72 | 219.01 |
| 5:3:2 | 352.92 | 76.76 | 295.2 |
| **RNN** | 7:2:1 | 328.04 | 68263.42 | 321.97 |
| 6:2:2 | 417.12 | 63641.99 | 411.85 |
| 5:3:2 | 315.73 | 55580.05 | 295.2 |
| **GRU** | 7:2:1 | 333.98 | 69498.33 | 327.84 |
| 6:2:2 | 429.36 | 65081.3 | 422.59 |
| 5:3:2 | 400.66 | 66930.28 | 369.19 |
| **ETS** | 7:2:1 | 576.4 | 164.91 | 491.48 |
| 6:2:2 | 2007.06 | 403.5 | 1702.57 |
| 5:3:2 | 155.21 | 54.33 | 113.64 |
| **GBT** | 7:2:1 | 10.44 | 2.11 | 7.63 |
| 6:2:2 | 193.99 | 35.2 | 167.62 |
| 5:3:2 | 340.5 | 68.19 | 282.07 |
| **DLM** | 7:2:1 | 20.007 | 5.33 | 16.67 |
| 6:2:2 | 15.22 | 3.97 | 11.49 |
| 5:3:2 | 43.58 | 9.53 | 35.35 |
| **XGBOOST** | 7:2:1 | 11.14 | 2.406 | 8.65 |
| 6:2:2 | 186.71 | 33.52 | 160.13 |
| 5:3:2 | 340.91 | 68.36 | 282.47 |
| **DNN** | 7:2:1 | 10.14 | 2.03 | 7.54 |
| 6:2:2 | 23.09 | 4.48 | 19.71 |
| 5:3:2 | 41.68 | 7.97 | 32.5 |

After building the models and evaluating the results on the test set with three ratios, 7-2-1, 6-2-2 and 5-3-2 for the nine main models: ETS, GBT, DLM, DNN, XGBOOST, ARIMA, Linear, RNN, and GRU.

For the parameter RMSE evaluation of the closing price of the BNB set. Based on the results of Table 2, we can see that the DNN model has the lowest RMSE value with 10.14 and followed by the GBT model with value 10.44 for the 7-2-1 The remaining models all have RMSE values larger than DNN and GBT models.

For the parameter MAE evaluation of the closing price of the BNB set. Based on the results of Table 2, we can see that the GBT model has the lowest MAE value with 7.63 and followed by the DNN model with value 7.54 for the 7-2-1 ratio.The remaining models all have larger MAE values.

The last parameter is the MAPE evaluation parameter of the closing price of the BNB set. Based on the results of Table 2, we can see that the DNN model has the lowest MAPE value with 2.03 and followed by the DNN model with value 2.11 for the 7-2-1 ratio. The remaining models all have larger MAPE values.

In general, through model evaluation parameters such as MAE, RMSE and MAPE, the DNN model almost gives the results with the smallest value, followed by the GBT model, then the DLM model. From the above conclusion, we can see that the GBT and DNN models are the two most accurate models and suitable with the data set of BNB

# V. CONCLUSION

With advancements in complex machine learning techniques, machine learning methods have become important tools for predicting. In this study, we used different machine learning models to analyze and forecast the prices of three popular cryptocurrencies: BTC, ETH, and BNB. Based on metrics such as Mean Absolute Percentage Error, Root Mean Square Error, and Mean Absolute Error, we evaluated the performance of the models and found that DLM, DNN, and GBT were suitable models for cryptocurrency price prediction. However, applying machine learning in this field also faces various challenges. The cryptocurrency market is highly volatile, making price prediction difficult. Additionally, the factors influencing cryptocurrency prices are decentralized and dependent on various factors such as adoption, news, regulations, and market sentiment.

In the future, we will strive to apply methods that can help increase the accuracy of prediction models. One approach would be to combine different machine learning methods to leverage the strengths of each method and minimize their weaknesses. For example, combining Deep Neural Networks (DNN) with Gradient Boosted Trees (GBT) can merge the deep learning capabilities of neural networks with the non-linear data processing abilities of boosted decision trees. This fusion can enhance the accuracy of cryptocurrency price predictions and shape market volatility. Furthermore, combining technical indicators such as Moving Averages, MACD (Moving Average Convergence Divergence), and RSI (Relative Strength Index) can be utilized alongside machine learning models to provide additional insights and improve the precision of price forecasts.

# JOB DISTRIBUTION

|  |  |  |  |
| --- | --- | --- | --- |
| Member  Work | **Nguyen Tan Phat**  **(Leader)** | **Lam Tra My** | **Nguyen Thi Thao Trang** |
| Problem statement | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated |
| Data collection | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated |
| Visualizing data | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated |
| Data Analysis | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated |
| LN model |  | Shape, arrow  Description automatically generated |  |
| ARIMA model | Shape, arrow  Description automatically generated |  |  |
| RNN model |  | Shape, arrow  Description automatically generated |  |
| GRU model |  |  | Shape, arrow  Description automatically generated |
| ETS model |  |  | Shape, arrow  Description automatically generated |
| GBT model | Shape, arrow  Description automatically generated |  |  |
| DLM model |  |  | Shape, arrow  Description automatically generated |
| XGBOOST model |  | Shape, arrow  Description automatically generated |  |
| DNN model | Shape, arrow  Description automatically generated |  |  |
| Visualizing results | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated |
| Applying goodness-of-fit measures | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated | Shape, arrow  Description automatically generated |
| Summarize and edit reports |  | Shape, arrow  Description automatically generated |  |
| Completion (%) | 100% | 100% | 100% |

# REFERENCE

1. Reaz Chowdhury, M.Arifur Rahman, M.Sohel Rahman, M.R.C. Mahdy Predicting and Forecasting the Price of Constituents and Index of Cryptocurrency Using Machine Learning, Physica A: Statistical Mechanics and its Applications, 1 August 2020.
2. V.Derbentseva,V.Babenko,K.Khrustalevc,H.Obruchd,S.Khrustalovac, Comparative Performance of Machine Learning Ensemble Algorithms for Forecasting Cryptocurrency Prices, *International Journal of Engineering*, 01 January 2021.
3. Reaz Chowdhury, M.Arifur Rahman, M.Sohel Rahman, M.R.C. Mahdy Predicting and Forecasting the Price of Constituents and Index of Cryptocurrency Using Machine Learning, Physica A: Statistical Mechanics and its Applications, 1 August 2020.
4. Taran Rishi, Stock Market Analysis Using Linear Regression, Proceedings of the Jepson Undergraduate Conference on International Economics, 7-2022
5. Time Series Analysis and Forecasting of Gold Price using ARIMA and LSTM Model; Dhruvi Sarvaiya - Disha Ramchandani; Ijraset International Journal For Research in Applied Science and Engineering Technology, 2022
6. Forecasting Gold Prices Using Multiple Linear Regression Method; Z. Ismail, A. Yahya and A. Shabri; American Journal of Applied Sciences 6, 2009.
7. C.Prof, “3 Ways to Calculate the Mean Absolute Error (MAE) in R [Examples]”, January 3, 2022. Available: <https://www.codingprof.com/3-ways-to-calculate-the-mean-absolute-error-mae-in-r-examples/> [Accessed June 15, 2023]
8. D.Christie, S.P.Neill, “Root-Mean-Squared Error”, 2022. Available: <https://www.sciencedirect.com/topics/engineering/root-mean-squared-error> [Accessed Jnue 15, 2023]
9. A.Roberts, “Mean Absolute Percentage Error (MAPE): What You Need To Know”, February 02, 2023. Available: <https://arize.com/blog-course/mean-absolute-percentage-error-mape-what-you-need-to-know/> [Accessed June 15, 2023]
10. M.Gupta, “Linear Regression in Machine Learning”. Available: <https://www.geeksforgeeks.org/ml-linear-regression/> [Accessed June 15, 2023]
11. R.Bevans, “Multiple Linear Regression | A Quick Guide (Examples)”, February 20, 2020. Available: <https://www.scribbr.com/statistics/multiple-linear-regression/#:~:text=What%20is%20multiple%20linear%20regression,variables%20using%20a%20straight%20line> [Accessed June 15, 2023]
12. S.Prabhakaran, “ARIMA Model – Complete Guide to Time Series Forecasting in Python”, August 22, 2021. Available: <https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/> [Accessed June 16, 2023]
13. D.Đ.Trình, “Recurrent Neural Network(RNN) trong TensorFlow”, April 1, 2022. Available: <https://websitehcm.com/recurrent-neural-network-rnn-trong-tensorflow/> [Accessed June 16, 2023]
14. N.T.Huyen, “Recurrent Neural Network: Từ RNN đến LSTM”, June 24, 2021. Available: <https://viblo.asia/p/recurrent-neural-network-tu-rnn-den-lstm-gGJ597z1ZX2> [Accessed June 16, 2023]
15. D.Kharazi, “Gated Recurrent Units”. Available: <https://dkharazi.github.io/notes/ml/nlp/gru> [Accessed June 16, 2023]
16. “Exponential Smoothing”, June 17, 2023. Available: <https://www.vedantu.com/maths/exponential-smoothing> [Accessed June 17, 2023]
17. Simplilearn, “An Introduction to Exponential Smoothing for Time Series Forecasting in Python”, May 25, 2023. Available: <https://www.simplilearn.com/exponential-smoothing-for-time-series-forecasting-in-python-article#how_to_configure_exponential_smoothing> [Accessed Jun 16, 2023]
18. B.T.Tung, “Gradient Boosting - Tất tần tật về thuật toán mạnh mẽ nhất trong Machine Learning”, May 28, 2021. Available: <https://viblo.asia/p/gradient-boosting-tat-tan-tat-ve-thuat-toan-manh-me-nhat-trong-machine-learning-YWOZrN7vZQ0#_2-bo-qua-bagging-chung-ta-den-voi-boosting-3> [Accessed June 16, 2023]
19. Gaurav, “An Introduction to Gradient Boosting Decision Trees”, June 12, 2021. Available: <https://www.machinelearningplus.com/machine-learning/an-introduction-to-gradient-boosting-decision-trees/#Gradient-Boosting-Decision-Trees> [Accessed June 16, 2023]
20. A.Saini, “Gradient Boosting Algorithm: A Complete Guide for Beginners”, September 20, 2021. Available: <https://www.analyticsvidhya.com/blog/2021/09/gradient-boosting-algorithm-a-complete-guide-for-beginners/> [Accessed June 15, 2023]
21. “Introduction to Dynamic Linear Models”, Department of Mathematics & Statistics. Available: <https://math.unm.edu/~ghuerta/tseries/dlmch2.pdf?fbclid=IwAR3fgE1ps6DJIX-B_HZ0dVzmOKy9raNcuidlCMuete0FiM3h3hBkv4byNIs> [Accessed June 17, 2023]
22. “XGBoost Documentation”. Available: <https://xgboost.readthedocs.io/en/stable/index.html> [Accessed June 16, 2023]
23. “XGBoost – What Is It and Why Does It Matter”, Nvidia. Available: <https://www.nvidia.com/en-us/glossary/data-science/xgboost/> [Accessed June 16, 2023]