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

**UNIVERSITY OF INFORMATION TECHNOLOGY**

**INFORMATION SYSTEM FACULTY**



Instructor:

Assoc. Prof. Nguyen Dinh Thuan

Nguyen Minh Nhut

Nguyen Thi Viet Huong

Student performance: Team 7

Nguyen Le Thai Hien 20521308

Tran Duy Khanh 20521457

Dinh Vo Xuan Hoan 20521337

**FINAL REPORT**

**BUSINESS ANALYTICS**

**Ho Chi Minh City, June 2023**

INSTRUCTOR'S COMMENTS

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

# **TABLE OF CONTENTS**

[**TABLE OF CONTENTS** 1](#_Toc138235856)

[I. INTRODUCTION 1](#_Toc138235857)

[II. RELATED WORK 2](#_Toc138235858)

[III. MODELING 3](#_Toc138235859)

[IV. METHOD 16](#_Toc138235860)

[V. EXPERIMENT 18](#_Toc138235861)

[VI. CONCLUSION 25](#_Toc138235862)

[VII. TABLE ASSIGNMENT 25](#_Toc138235863)

[VIII. REFERENCES 26](#_Toc138235864)

# I. INTRODUCTION

Cryptocurrencies have completely transformed the global financial landscape, offering a digital alternative for transactions. Bitcoin, being the frontrunner, has garnered immense attention and become synonymous with the concept of digital currencies. However, other cryptocurrencies like Ethereum and Binancecoin (BNB) have also gained significant traction and recognition in recent years.

The ability to accurately predict cryptocurrency price movements has become a topic of great interest for researchers and investors. Such predictions can provide valuable insights, aiding in informed investment decision-making.

This study aims to address this need by employing various statistical and machine-learning models to forecast the closing prices of three prominent cryptocurrencies: Bitcoin (BTC), Ethereum (ETH), and Binancecoin (BNB). The selected models encompass a range of approaches, including Autoregressive Integrated Moving Average (ARIMA), Linear Regression (LN), Exponential Smoothing (ETS), Gaussian Process Regression (GPR), Random Forest (RF), Extreme Gradient Boosting (XGBoost), as well as advanced artificial intelligence techniques like Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN), and Gated Recurrent Unit (GRU).

By comparing the performance of these models, this study aims to identify the most accurate approach for predicting the future closing prices of the selected cryptocurrencies. The accuracy of the forecasted prices will be assessed using crucial evaluation parameters, such as Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and Mean Absolute Error (MAE).

Gaining an understanding of the strengths and limitations of each forecasting model is paramount for investors navigating the highly volatile cryptocurrency market. The findings of this study will contribute to the existing body of knowledge in cryptocurrency price prediction, offering valuable insights to researchers, investors, and market participants. Leveraging the potential of statistical models, machine learning techniques, and artificial intelligence, this research seeks to enhance our comprehension of cryptocurrency price dynamics, facilitating more informed investment decisions.

# II. RELATED WORK

Over the past few years, Bitcoin has been a topic of interest of many, from academic researchers to trade investors. Bitcoin is the first as well as the most popular cryptocurrency till date. Since its launch in 2009, it has become widely popular amongst various kinds of people for its trading system without the need of a third party and also due to high volatility of Bitcoin price. Therefore, Shaily Roy, Samiha Nanjiba and Amitabha Chakrabartypropose a suitable model that can predict the market price of Bitcoin best by applying a few statistical analysis. Their work is done on four year's bitcoin data from 2013 to 2017 based on time series approaches especially autoregressive integrated moving average (ARIMA) model and the work finally could acquire an accuracy of 90% for deciding volatility in weighted costs of bitcoin in the short run.[1]

In another article, Nicola Uras, Lodovica Marchesi, Michele Marchesi, Roberto Tonelli working on forecast daily closing price series of Bitcoin, Litecoin and Ethereum cryptocurrencies, using data on prices and volumes of prior days. They used the Simple Linear Regression (SLR) model for uni-variate series forecast using only closing prices, and the Multiple Linear Regression (MLR) model for multivariate series using both price and volume data. They used two artificial neural networks as well: Multilayer Perceptron (MLP) and long short-term memory (LSTM). In this case the best results are obtained using more than one previous price, thus confirming the existence of time regimes different from random walks. Their models perform well also in terms of time complexity, and provide overall results better than those obtained in the benchmark studies, improving the state-of-the-art.[2]

Another study is Andrés Oviedo-Gómez, Juan Manuel Candelo-Viáfara & Diego Fernando Manotas-Duque evaluate different crypto market variables through a quantile regression model and thus identify the best predictors for Bitcoin price forecasting by machine learning models. The main finding was that the Gaussian Process Regression models allowed the best performance metrics through the following predictors: high and low Bitcoin price, ask-sum, and Bitcoin price lagged. Likewise, the Bitcoin price was predicted for the next seven days, and it was observed a significant approximation by the confidence intervals of Gaussian Process Regression.[3]

Siripurapu Mounika, Podila Anjali Yadav, Tulluru Yashaswi, Chalimadugu Yamini Krishna, Dr. Vuyyuru Krishna Reddy also give us a view through the deep learning models such as Convolutional Neural Networks (CNN). Their aim of the work is to give accurate predictions and forecast and bring the daily trend for crypto currency market. Experimental results show that the proposed system given better accuracy on predictions.[4]

Zheshi Chen, Chunhong Li, Wenjun Sun compared with benchmark results for daily price prediction, they achieve a better performance, with the highest accuracies of the statistical methods and machine learning algorithms of 66% and 65.3%, respectively. Machine learning models including Random Forest, XGBoost, Quadratic Discriminant Analysis, Support Vector Machine and Long Short-term Memory for Bitcoin 5-minute interval price prediction are superior to statistical methods, with accuracy reaching 67.2%. Their investigation of Bitcoin price prediction can be considered a pilot study of the importance of the sample dimension in machine learning techniques.[5]

V. Derbentsev, V. Babenko, K. Khrustalev, H. Obruch, S. Khrustalova give another approach of using Random Forests (RF) and Stochastic Gradient Boosting Machine (SGBM) Their results verify the applicability of the ML ensembles approach for the forecasting of cryptocurrency prices. The out of sample accuracy of short-term prediction daily close prices obtained by the SGBM and RF in terms of Mean Absolut Percentage Error (MAPE) for the three most capitalized cryptocurrencies (BTC, ETH, and XRP) were within 0.92-2.61 %.[6]

# III. MODELING

a) ARIMA

ARIMA is a statistical method for analyzing and forecasting time series data which is based on the Box and Jenkins method. This is a model used for forecasting that involves quantitative analysis over a period. The future value of the predictor variable will be influenced by the trend of that object in the past.

It is combining autoregressive (AR) with moving average (MA) method. Then, this hybrid method is integrated with data of difference process. The difference process is important to make sure data involved in this analysis can be represented as data with stationary characteristics. Therefore, the combination of this method is called as autoregressive integrated moving average.

ARIMA model consists of three combinations:

* Autoregressive – AR(p):

The notation AR (p) indicates an autoregressive model of order p. In an autoregression model, we forecast the variable of interest using a linear combination of past values of the variable. The term autoregression indicates that it is a regression of the variable against itself.

The AR (p) model is defined as:

Where:

is the value of the time series at time t.

is the ith value of the time series at time t.

is a constant term.

is the error term.

is the ith coeffficient of the AR model.

is the order of the AR model.

* Intergration – I(d):

It involves taking differences between the time series values to make the series stationary. Differencing is a method for removing trends and seasonality from a time series, which can make it easier to model.

Where:

is the value of the time series at time t.

d is the degree of differencing required to make the time series stationery.

Δ is the differencing operator, which is applied d times to the time series.

Δ = (1 - ) where is the lag operator.

* Moving Average – MA(q):

It involves regressing the current value on the past forecast errors.

The MA (q) model is defined as:

Where:

is the value of the time series at time t.

is the ith error term at the time t.

is the mean of the series.

is the error term.

is the ith coeffficient of the MA model.

is the order of the MA model.

Combine the AR(p), I(d), MA(q) we have ARIMA (p, d, q) is defined as:

b) LN

Regression analysis is a tool for building statistical models that characterize relationships among a dependent variable and one or more independent variables, all of which are numerical.

There are Simple Linear Regression and Multiple Linear Regression:

+ Simple Linear Regression estimates the relationship between a scalar response y and a single explanatory variable x (also called dependent variable y and independent variable 𝑥), given a set of data that includes observations for both variables for a particular sample.

+ Multiple Linear Regression is a generalization of simple linear regression in which is used to estimate the relationship between two or more independent variables and one dependent variable.

+ The Multivariable Linear Regression formula:

​*yi*​ = *β*0​ + *β*1​*xi*​+ *β*2​*xi*2​+...+ *βp*​*xip*​+ *ϵ*[8]

where, for i=n observations:

* *yi*​ is the dependent or predicted variable.
* *β*0 is the y-intercept, i.e., the value of y when both *xi* and *x*2 are 0.
* *β*1 and *β*2​ are the regression coefficients representing the change in *y* relative to a one-unit change in ​ *xi*1 and ​ *xi*2, respectively.
* *βp*​ is the slope coefficient for each independent variable.
* **ϵ** is the model’s random error (residual) term.

c) LSTM

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) that is specifically designed to handle sequential data, such as time series, speech, and text. LSTM networks are capable of learning long-term dependencies in sequential data, which makes them well suited for tasks such as language translation, speech recognition, and time series forecasting.[9]

A picture containing screenshot, diagram

Description automatically generated

*Figure 1. Structure of an LSTM*[10]

The structure of an LSTM is shown in the figure above. It consists of three interacting gates (input gate, output gate and forget gate) and numerous memory blocks known as cells, which store important information during the processing of sequential data.

The input gate controls what information is added to the memory cell. The forget gate controls what information is removed from the memory cell. And the output gate controls what information is output from the memory cell. This allows LSTM networks to selectively retain or discard information as it flows through the network, which allows them to learn long-term dependencies.

* Forget gate:

[10]

* Input gate:

[10]

* Output gate:

[10]

Formula explanation:

: sigmoid function, values range from [0, 1]

, , : weight matrix linking inputs and LSTM gates.

: hidden state at the previous time step

: input at the current time step

: hidden state at the current step

, , , : bias weight vector

: activation function

d) GRU

Gated recurrent units (GRUs) are a gating mechanism in recurrent neural networks, introduced in 2014 by Kyunghyun Cho et al. The GRU is like a long short-term memory (LSTM) with a forget gate, but has fewer parameters than LSTM, as it lacks an output gate.

The GRU model is a modified version of the LSTM model, it not only merges the forget gate and the input gate into an update gate but also drops the cell state, achieved reduction of amount of parameters. A GRU unit is composed of reset gate and update gate, due to the simpler architecture, it is contributing to train faster and search optimal solution easily

In the first step, reset gate is calculated using both the hidden state from the previous time step and the input data at the current time step, it be reserved by applying a sigmoid function

[11]

Where:

: is input data at the current time step.

: is the hidden state from the previous time step.

, : are the weighting vectors respectively.

Next step, decide the information which will be kept from the previous time steps together with the new inputs following.

[11]

Second, the update gate is computed using the previous hidden state and current input data using the same formula, like the reset gate. But each weight multiplied with the input and hidden state is independent and unique to each gate, which means the final vectors for the update gate are different from the reset gate.

[11]

Finally, summed with the output.

[11]

e) ETS

The Exponential Smoothing (ETS) is a time series forecasting technique that makes future predictions based on historical data. Exponential smoothing was proposed in the late 1950s (Brown, 1959; Holt, 1957; Winters, 1960), and has motivated some of the most successful forecasting methods.

The simpliest model is called the simple exponential smoothing (SES) model. This forecasting method is the most widely used of all forecasting techniques. This method is suitable for forecasting data with no clear trend or seasonal pattern.

*:* forecast value at time t + 1.

*:* forecast value at time t.

*:* actual value at the time t.

*:* smoothing parameter, between 0 and 1

For any α between 0 and 1, the weights attached to the observations decrease exponentially as we go back in time.

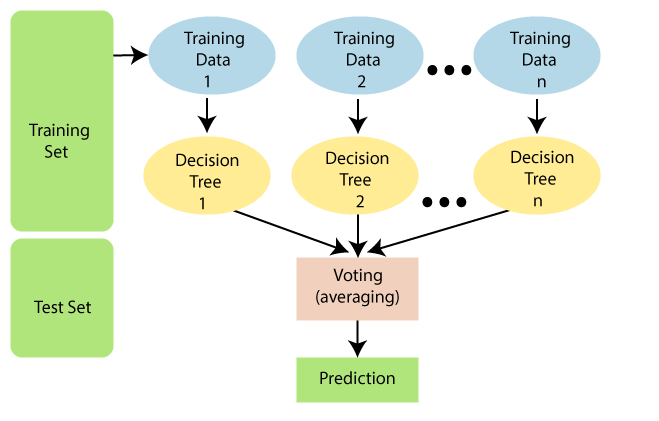
Small α value (close to 0): results in more smooth and slower response to changes in the time series.

Large α value (close to 1): results less smoothing and a faster response to changes in the time series.

f) RF

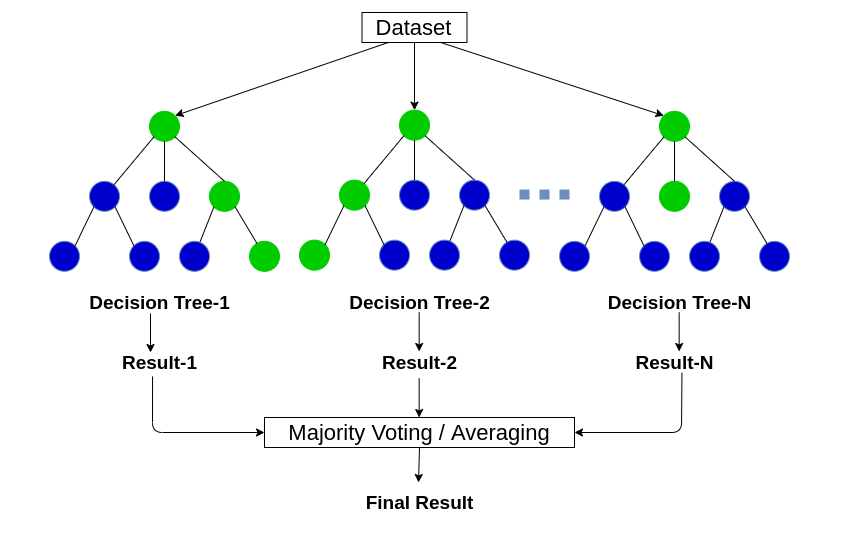
Random Forest is one of the most popular and commonly used algorithms by Data Scientists. Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

Random Forest is a classifier that contains several decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.



*Figure 2. How the Random Forest Algorithm work*[13]

Random Forest uses an ensemble technique called Boostrap Aggregation or also known as Bagging. Bagging, it chooses a random sample and creates a different training subset from training data, the final ouput is based on majority voting.



*Figure 3. Majority Voting/ Averaging*[14]

g) GPR

Gaussian Process Regression (GPR) has its origins in the “theory of Gaussian processes” and “Bayesian regression methods”. Although there is no specific date identified for the emergence of GPR, it has been developed and become a popular method in the fields of machine learning and statistics.

One significant contribution in the field of GPR is "Gaussian Processes for Machine Learning" by Carl Edward Rasmussen and Christopher K. I. Williams, published in 2006. This book provides a clear and detailed foundation for GPR and has significantly contributed to the development and application of the algorithm.

Gaussian process regression (GPR) is a nonparametric, Bayesian approach to regression that is making waves in machine learning. GPR has several benefits, working well on small datasets and having the ability to provide uncertainty measurements on predictions.

Fomula:

Rational quadratic kernel:

RationalQuadratic: This kernel combines components of the RBF (Radial Basis Function) kernel and includes a linear component as well. It is suitable for modeling patterns that exhibit complex non-uniform or non-periodic behavior in time series data.

[15]

With:

* the variance.
* represents the Euclidean distance between two input data points.
* the lengthscale.
* the scale-mixture ( > 0).

The parameter values (such as , , ) will be adjusted iteratively to find the maximum point of the Maximum Likelihood Estimation (MLE).

The log-likelihood function:

Where:

* is the value of the log likelihood function.
* is the vector of training data values.
* is the vector of predicted values corresponding to the training data.
* is the covariance matrix computed based on the kernel function and the training data.
* is the determinant of the matrix .
* is the number of training data points*.*

The Gaussian Process Regression (GPR) model can be represented by the following formulas:

The regression function modeled by a multivariate Gaussian is given as:

where:

* = [, . . ., ]: It represents the input data points, where , is the i-th data point.
* = [, . . ., ]: It represents the output values corresponding to the input data points. Each is the value of the function at .
* µ = [, . . ., ]: It represents the mean values corresponding to the input data points. Each mi is the mean value of the function at .
* = : It represents the covariance (or similarity) between the input data points and . is the covariance matrix, where each element represents the covariance between and .

The joint distribution of f and is expressed as:

Where:

* = , , and .
* The mean , = 0. (Centering the Mean: Setting the mean to zero helps in centering the predictions around zero, assuming that the function being modeled has a mean value of zero. This simplifies the analysis and interpretation of the GPR model.)

In more realistic situations, we don’t have access to true function values but noisy versions thereof . Assuming there is an additive independent and identically distributed (i.i.d.) Gaussian noise with variance , the prior on the noisy observations becomes. The joint distribution of the observed values and the function values at new testing points becom.

By deriving the conditional distribution, we get the predictive equations for Gaussian processes regression:

Where:

* Prediction for test data, given the joint distribution, the prediction for the test data can be calculated as:

[16]

* Prediction variance of the predictions can be calculated as:

[16]

.

*“”* is identity matrix, In the variance function , it can be noted that the variance does not depend on the observed output but only on the inputs and . This is a property of the Gaussian distribution.

Overall, GPR's ability to capture complex relationships, estimate variance, and handle different temporal patterns makes it a valuable tool for time series prediction tasks.

h) CNN

Convolutional Neural Network.

Lecun et al. proposed the CNN network concept in 1998. CNN performs well in both natural language processing and image processing (NLP). It can be used to forecast time series with succes.

A convolutional neural network (CNN) is essentially a neural network that employs the convolution operation (instead of a fully connected layer) as one of its layers. CNNs are an incredibly successful technology that has been applied to problems where in the input data on which predictions are to be made has a known grid-like topology, like a time series (a 1-D grid) or an image (a 2-D grid). CNNs ushered deep learning into modern times, solving one of the most crucial computational problems in the digital era of computer vision. With the popularity of CNNs, a surge in the research for deep learning was witnessed that continues today.

CNN has achieved significant success in the field of image processing, including facial recognition, object recognition, autonomous driving, and image classification. It has also been extended to apply to other domains such as natural language processing.

CNN is used to analyze and extract features from unstructured data such as images, audio, and time series data. There are different types of CNNs used for different tasks. Here are some popular types:

* One-Dimensional Convolutional Neural Network (CNN 1D): Used for time series data. It applies convolution along the time dimension to extract features from patterns and trends in the sequence.
* Two-Dimensional Convolutional Neural Network (CNN 2D): Used for image processing and other structured two-dimensional data. It applies convolution along both the vertical and horizontal dimensions of the image or two-dimensional data.
* Three-Dimensional Convolutional Neural Network (CNN 3D): Used for video processing and other structured three-dimensional data. It applies convolution along the vertical, horizontal, and temporal dimensions of the video or three-dimensional data.

In the case of predicting time series data, CNN 1D is commonly used. This is because CNN 1D can identify patterns and trends in the time series data without requiring specific positional information. By using convolutional layers, CNN 1D can automatically learn and extract important features from the time series. This allows it to recognize patterns and trends in the data while reducing the number of parameters to be learned compared to traditional neural networks.

CNN is made up of three primary layers a convolution layer, a pooling layer, and a fully connected layer. The convolution layer makes an effort to retrieve the best features from the 1-D matrix and perform calculation to provide a convoluted output, as shown in equation below. [rf]

[17]

Where is the convolution output, activation function is , is input value, is the weight, and is the bias.

A diagram of convolution structure

Description automatically generated with medium confidence

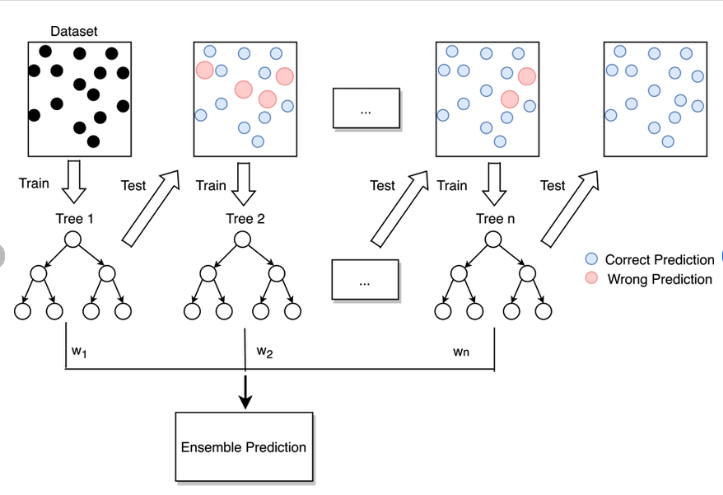
Figure 4. CNN one-dimensional structure.[17]

i) XGBoost

XGBoost, which stands for Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. It provides parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems.

Gradient Boosting Decision Trees (GBDT) is a decision tree ensemble learning algorithm similar to random forest, for classification and regression. Ensemble learning algorithms combine multiple machine learning algorithms to obtain a better model.

Random forest “bagging” minimizes the variance and overfitting, while GBDT “boosting” minimizes the bias and underfitting.



*Figure 5. Boosting*[18]

XGBoost is used for supervised learning problems, where we use the training data (with multiple features) to predict a target variable .

Objective: [19]

Where:

: training loss measures well model fit on training data.

: Regularization, measures complexity of trees, K is the number of trees.

Following additive training to learn tree ensembles, add a new function each time at a result, we will have:

[19]

: model at training round t

: new function at round t

Atfer all we will have the prediction at round t:

[19]

# IV. METHOD

a) ETS

Step-by-step explanation of the ETS model:

Prepare the data: Collect the time series data you want to analyze. This data should include timestamps and corresponding values, such as stock prices, temperature readings, or sales figures. Ensure that the data is properly formatted and available in a suitable data structure, such as a pandas DataFrame in Python.

Set initial values: Initialize the values for and can be the initial value of the time series or an initial estimate. can be the difference between the initial value and .

Implement the ETS algorithm: Apply the formula to calculate forecasts for future time points. In this formula, α is the coefficient that you need to determine to adjust the importance between and . The value of α typically ranges from 0 to 1.

Iterate step 3 for each time point: Apply the ETS formula to each subsequent time point in the time series to calculate forecasts. Start with , then compute , , and so on, using the same formula until the last time point.

Evaluate the results: Assess and compare the ETS forecasts with the actual values. Use evaluation metrics such as mean squared error (MSE) or mean absolute error (MAE) to measure the accuracy of the forecasts.

Iterate and adjust: If the forecast results are not satisfactory, you can adjust the value of α or modify other parameters in the ETS formula. Repeat the ETS implementation process and adjustments until you achieve the best possible results.

Please note that the implementation of ETS may depend on the programming language and libraries being used. The steps provided above offer a general guideline for implementing ETS using the formula .

b) Random Forest

Step-by-step explanation of RF model

Step 1: Data Preparation

Collect time data series you want, ensure your data is in a tabular format, with rows representing samples and columns representing features.

Step 2: Feature Selection

Select a large number of features to improve the model and prevent overfitting.

Step 3: Bagging

This step creates multiple decision trees by resampling the training data with replacement. So, you can avoid overfitting because decision trees is built independently.

Step 4: Voting/ Averaging

When you make predictions, RF uses voting. Each decision tree predicts the outcome.

c) GPR

Step-by-step explanation of the GPR model:

Step 1: Data Preparation.

Prepare the training data, consisting of input-output pairs (X, y), where X is the input variable and y is the corresponding target variable.

Ensure that the data is processed and normalized if necessary.

Step 2: Select Kernel Function.

Choose a suitable kernel function that captures the similarity between data points. Common choices include the Gaussian (RBF) kernel, RationalQuadratic kernel, and various other kernel functions.

Step 3: Determine Hyperparameters.

Determine the hyperparameters of the model, including the variance parameter ,the scale-mixture and length scale parameter of the kernel function.

Maximum Likelihood Estimation (MLE) or cross-validation can be used to estimate these hyperparameters.

Step 4: Build Kernel Matrix.

Use the selected kernel function and training data to compute the kernel matrix of size (n x n), where n is the number of training data points.

Step 5: Compute Covariance Matrix.

Compute the covariance matrix for prediction on both the training data and the test data, including the kernel matrices

Step 6: Prediction and variance Calculation.

Using the computed kernel matrices and covariance matrix, perform prediction for the test data ,using the formula:

Where y is the target vector of the training data.

Calculate the uncertainty (variance).

Step 7: Model Evaluation and Refinement

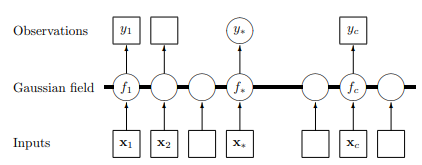
Evaluate the GPR model using evaluation metrics such as error measures (RMSE, MAPE, MAE).

Adjust the hyperparameters and consider selecting different kernel functions to improve the performance of the model if necessary.

Step 8: Using the trained model.

Once the model has been trained, evaluated, and refined, it can be used to predict new values on unseen data.

This is a general overview of the step-by-step process involved in building a GPR model. However, there can be variations and specific adjustments depending on the problem and requirements.



*Figure 6. Graphical model (chain graph) for a GP for regression*

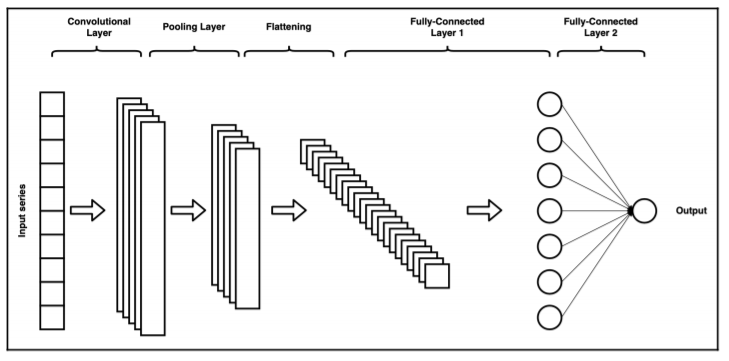
d) CNN

A The pooling layer takes the output of the convolutions as an input.

The max pooling function is used to choose the heavily weighted features in the pooling layer. The pooling layer’s output is passed to the flattened layer.

The flatten layer’s primary function is to convert the data into a single array form.

The fully connected layer receives the flatten layer’s output and processes it to obtain the results.



*Figure 7. 1D array for time series forecast.*

e) XGBoost.

Step-by-step explanation of XGBoost model

Step 1: Data Preparation

Collect time data series you want, ensure your data is in a tabular format, with rows representing samples and columns representing features

Step 2: Training data

A random subset of features is selected to determine the best split to improve generalization and prevent overfitting. Use this training data (with multiple features) to predict a target variable

Step 3: Train weak model

Following objective, each sample in the training dataset is assigned equal weights. After that train weak model, are typically simple models with low complexity.

Step 4: Evaluate, update and combine weak models

The weak model’s performance is evaluated on the training dataset, depend on mean squared error. After evaluating the weak model, weights of the training samples are updated. And then, the weak models once all trained, their predictions are combined to make a final prediction. The combination can be done in various ways, depending on the boosting algorithm.

[19]

Step 5: Final prediction

The prediction is often based on majority voting. While for regression problems, the predictions are averaged.

[19]

# V. EXPERIMENT

a) Dataset

This study selects daily historical data of three popular cryptos which are collected from the Investing.com Website.

The website link: <https://vn.investing.com/crypto/currencies>

This data includes information about Bitcoin, Ethereum, and BNB from May 18, 2018, to May 18, 2023, such as High, Low, Open, Price, Volume, and Change.

• High: This represents the highest price that a cryptocurrency reaches during the day.

• Low: This represents the lowest price that a cryptocurrency reaches during the day.

• Open: This represents the price of a cryptocurrency at the beginning of the day

• Price: This represents the current price of a cryptocurrency. By the end of the day, this will be known as the closing price.

• Volume: This represents the total number of units of a cryptocurrency that were traded during the day.

• Change: This represents the percentage change in the price of a cryptocurrency compared to its previous price. A positive change shows a price increase, while a negative change shows a price decrease.

Before performing some methods, we perform descriptive statistics on the data of the three cryptos using the Matplotlib tool in Python, including creating graphs that display the data and calculating some statistical values to better understand its characteristics.

A picture containing text, screenshot, font, number

Description automatically generated

*Table 1 Descriptive statistics of BTC.*

A picture containing text, screenshot, font, number

Description automatically generated

*Table 2 Descriptive statistics of ETH.*

A picture containing text, screenshot, font, number

Description automatically generated

*Table 3 Descriptive statistics of BNB.*

-Visualization.

1) BTC

A graph showing the price of bitcoin

Description automatically generated with medium confidence

*Figure 8. Visualization of BTC close price.*

2) ETH

A picture containing text, screenshot, plot, line

Description automatically generated

*Figure 9. Visualization of ETH close price.*

3) BNB

A picture containing screenshot, plot, text, line

Description automatically generated

*Figure 10. Visualization of BNB close price.*

b) Tool used

In this study, we used Python programming language and Python library on Jupyter Notebook software to build training models, calculate the evaluation, forecasting and visualize the results.

c) Split dataset

The dataset is divided into training, test, and validate sets based on three ratios include 70-20-10%, 60-30-10%, and 50-30-20%. The training set is used to create the model, and its performance is evaluated using the test set and validate set. To improve the dataset quality, certain preprocessing techniques are applied, including data cleansing, feature selection, data reduction, and data transformation.

d) Evaluation

In this research, predictive models are evaluated according to three criteria: MAPE, RMSE, and MAE.

In the following formulas:

n is the number of observations.

*Xi* element is the predicted *ith* value.

*Yi* element is the actual *ith* value.

- Mean Absolute Percentage Error – MAPE

[20]

(Best value = 0; worst value = +∞)

- Root Mean Squared Error - RMSE

[20]

(Best value = 0; worst value = +∞)

- Mean Absolute Error - MAE

[20]

(Best value = 0; worst value = +*∞*)

e) Result

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model | 7:2:1 | | | 6:3:1 | | | 5:3:2 | | |
| RMSE | MAPE | MAE | RMSE | MAPE | MAE | RMSE | MAPE | MAE |
| ARIMA | 33455.97 | 124.35% | 31265.1 | 14242.86 | 43.55% | 11506.54 | 29620.21 | 59.91% | 27435.14 |
| LN | 22897.28 | 81.99% | 18932.26 | 16692.12 | 48.81% | 13681.81 | 32634.15 | 68.53% | 30731.53 |
| LSTM | 15542.96 | 38.14% | 12226.16 | 18477.78 | 49.09% | 15037.39 | 14374.15 | 31.38% | 11509.19 |
| GRU | 15771.27 | 37.88% | 12358.92 | 18573.53 | 53.76% | 14843.68 | 20271.5 | 69.02% | 16796.31 |
| ETS | 57411.04 | 212.6% | 52285.84 | 12365.06 | 36.91% | 10215.43 | 16025.81 | 27.82% | 12871.72 |
| RF | 13225.67 | 39.28% | 11294.10 | 17393.72 | 52.70% | 14232.47 | 30060.92 | 61.10% | 27910.38 |
| GPR | 21679.50 | 53.85% | 18753.44 | 12065.66 | 22.83% | 9463.12 | 35914.3 | 75.3% | 33888.61 |
| CNN | 16549.7 | 44.77% | 13159.92 | 18791.54 | 47.92% | 15113.7 | 15535.77 | 33.36% | 12402.7 |
| XGBoost | 12911.91 | 42.27% | 11293.64 | 16995.69 | 52.83% | 13845.5 | 29838.67 | 60.5% | 27670.86 |

*Table 4. Training results of BTC dataset.*

Based on the table we conclude that the best model for forecasting the next 30 days BTC closing price is GPR model with the proportion 6:3:1 because it has the lowest RMSE, MAPE, MAE value.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model | 7:2:1 | | | 6:3:1 | | | 5:3:2 | | |
| RMSE | MAPE | MAE | RMSE | MAPE | MAE | RMSE | MAPE | MAE |
| ARIMA | 4461.28 | 225.95% | 3673 | 6041.72 | 284.09% | 5115.96 | 3490.67 | 112.06% | 3151.55 |
| LN | 1237.47 | 60.33% | 1089.10 | 1628.63 | 44.14% | 1318.19 | 2621.62 | 88.29% | 2398.75 |
| LSTM | 1262.48 | 43.47% | 992.68 | 1594.2 | 60.62% | 1290.33 | 1594.14 | 86.73 % | 1273.48 |
| GRU | 1273.83 | 42.98% | 998.50% | 1454.17 | 54.58 % | 1181.39 | 1356.16 | 72.27 % | 1087.92 |
| ETS | 5027.52 | 270.74% | 4501.69 | 6711.44 | 315.20% | 5702.24 | 2098.94 | 66.13% | 1874.47 |
| RF | 1218.92 | 42.55% | 959.04 | 1287.35 | 53.64% | 1064.96 | 2411.31 | 76.58% | 2168.04 |
| GPR | 608.29 | 29.34% | 538.93 | 1545.42 | 40.69% | 1223 | 2564.53 | 82.57% | 2319.88 |
| CNN | 1369.90 | 53.22% | 1091.27 | 1464.30 | 54.68% | 1185.80 | 1481.38 | 71.42% | 1194.30 |
| XGBoost | 1220.98 | 43.05% | 965.22 | 1458.15 | 68.27% | 1229.26 | 2421.84 | 77.18% | 2179.75 |

*Table 5. Training results of ETH dataset.*

Based on the table we conclude that the best model for forecasting the next 30 days ETH closing price is GPR model with the proportion 7-2-1 because it has the lowest RMSE, MAPE, MAE value.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model | 7:2:1 | | | 6:3:1 | | | 5:3:2 | | |
| RMSE | MAPE | MAE | RMSE | MAPE | MAE | RMSE | MAPE | MAE |
| ARIMA | 288.83 | 88,22% | 269.59 | 942.54 | 258.59% | 828.23 | 369.66 | 81.82% | 327.65 |
| LN | 133.00 | 33.12% | 114.95 | 211.59 | 42.16% | 176.07 | 368.89 | 83.07% | 327.50 |
| LSTM | 126.20 | 25.74% | 97.06 | 150.63 | 37.10% | 123.34 | 186.34 | 160.02% | 149.22 |
| GRU | 128.05 | 25.74% | 98.03 | 128.05 | 25.74% | 98.03 | 208.01 | 192.51% | 161.53 |
| ETS | 512.50 | 153.89% | 466.83 | 979.72 | 268.22% | 857.20 | 334.26 | 73.26% | 293.85 |
| RF | 116.61 | 23.94% | 89.60 | 229.90 | 66.22% | 208.26 | 361.85 | 79.23% | 319.76 |
| GPR | 67.45 | 16.20% | 52.35 | 240.50 | 56.78% | 214.19 | 377.58 | 84.88% | 335.53 |
| CNN | 145.21 | 32.80% | 114.71 | 149.71 | 33.21% | 118.39 | 240.95 | 178.60% | 188.49 |
| XGBoost | 115.87 | 23.80% | 89.11 | 239.82 | 69.61% | 219.34 | 362.13 | 79.28% | 320.03 |

*Table 6. Training results of BNB dataset.*

Based on the table we conclude that the best model for forecasting the next 30 days BNB closing price is GPR model with the proportion 7-2-1 because it has the lowest RMSE, MAPE, MAE value

f) Visualize  
Visualizing the predicted values and the actual values of the GPR model and the next 30 days forecasting values.

A picture containing text, screenshot, plot, diagram

Description automatically generated

*Figure 9. Predictions of the GPR model with BTC dataset and rate of 6:3:1.*

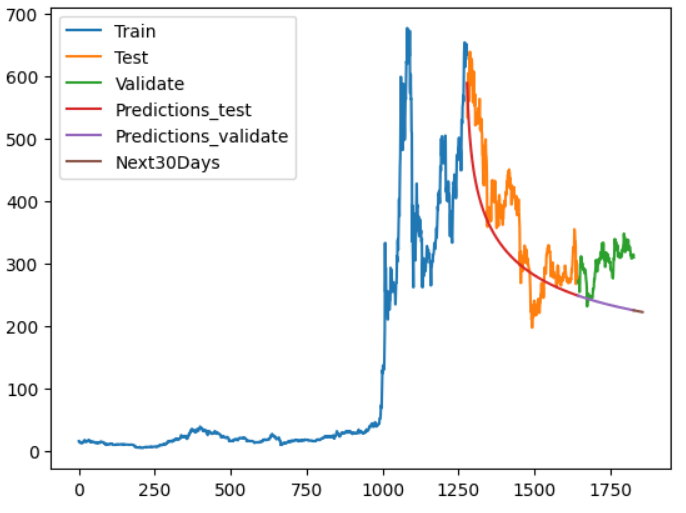
Visualizing the predicted values and the actual values of GPR model on ETH dataset and the next 30 days forecasting values.

A graph with blue and orange lines

Description automatically generated with low confidence

*Figure 10. Predictions of the GPR model with ETH dataset and rate of 7:2:1.*

Visualizing the predicted values and the actual values of GPR model on BNB dataset and the next 30 days forecasting values.



*Figure 11. Predictions of the GPR model with BNB dataset and rate of 7:2:1.*

# VI. CONCLUSION

In conclusion, this study compared the performance of various models in predicting the future prices of BTC, ETH, and BNB cryptocurrencies based on the resulting time series. Among the models tested, the Gaussian Process Regression (GPR) model emerged as the most suitable for this task. Conversely, the Linear Regression, ARIMA, LSTM, GRU, Random Forest, ETS, and XGBOOT models did not exhibit comparable performance. These findings underscore the significance of employing diverse modeling approaches in financial analysis and highlight the potential value of utilizing the GPR model for forecasting cryptocurrency prices in the future. Further research is warranted to validate these results and explore the performance of alternative models in different types of cryptocurrency price prediction tasks.

# VII. TABLE ASSIGNMENT

|  |  |  |  |
| --- | --- | --- | --- |
| **Member**  **Tasks** | **Nguyen Le Thai Hien** | **Tran Duy Khanh** | **Dinh Vo Xuan Hoan** |
| Problem statement | X | X | X |
| Build a report template | X |  |  |
| Work on ARIMA | X |  |  |
| Work on LN |  | X |  |
| Work on LSTM |  |  | X |
| Work on GRU | X |  | X |
| Work on ETS | X |  |  |
| Work on RF |  |  | X |
| Work on GPR |  | X |  |
| Work on CNN |  | X |  |
| Work on XGBoost | X |  | X |
| Summarize and edit reports | X | X | X |

# VIII. REFERENCES

[1] S. Roy, S. Nanjiba, and A. Chakrabarty, “Bitcoin Price Forecasting Using Time Series Analysis,” in *2018 21st International Conference of Computer and Information Technology (ICCIT)*, Dec. 2018, pp. 1–5. doi: 10.1109/ICCITECHN.2018.8631923.

[2] N. Uras, L. Marchesi, M. Marchesi, and R. Tonelli, “Forecasting Bitcoin closing price series using linear regression and neural networks models,” *PeerJ Comput. Sci.*, vol. 6, p. e279, Jul. 2020, doi: 10.7717/peerj-cs.279.

[3] A. Oviedo-Gómez, J. M. Candelo-Viáfara, and D. F. Manotas-Duque, “Bitcoin Price Forecasting Through Crypto Market Variables: Quantile Regression and Machine Learning Approaches,” in *Handbook on Decision Making: Volume 3: Trends and Challenges in Intelligent Decision Support Systems*, J. A. Zapata-Cortes, C. Sánchez-Ramírez, G. Alor-Hernández, and J. L. García-Alcaraz, Eds., in Intelligent Systems Reference Library. Cham: Springer International Publishing, 2023, pp. 253–271. doi: 10.1007/978-3-031-08246-7\_11.

[4] S. Mounika, “Crypto-Currency Price Prediction using CNN and LSTM Models,” *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 9, no. 3, pp. 107–114, Mar. 2021, doi: 10.22214/ijraset.2021.33191.

[5] Z. Chen, C. Li, and W. Sun, “Bitcoin price prediction using machine learning: An approach to sample dimension engineering,” *J. Comput. Appl. Math.*, vol. 365, p. 112395, Feb. 2020, doi: 10.1016/j.cam.2019.112395.

[6] V. Derbentsev, V. Babenko, K. Khrustalev, H. Obruch, and S. Khrustalova, “Comparative Performance of Machine Learning Ensemble Algorithms for Forecasting Cryptocurrency Prices,” *Int. J. Eng.*, vol. 34, no. 1, pp. 140–148, Jan. 2021, doi: 10.5829/ije.2021.34.01a.16.

[7] N. Abu Bakar and S. Rosbi, “Autoregressive Integrated Moving Average (ARIMA) Model for Forecasting Cryptocurrency Exchange Rate in High Volatility Environment: A New Insight of Bitcoin Transaction,” *Int. J. Adv. Eng. Res. Scinece*, vol. 4, Nov. 2017, doi: 10.22161/ijaers.4.11.20.

[8] “Multiple Linear Regression,” *Corporate Finance Institute*. https://corporatefinanceinstitute.com/resources/data-science/multiple-linear-regression/ (accessed Jun. 21, 2023).

[9] “Deep Learning | Introduction to Long Short Term Memory,” *GeeksforGeeks*, Jan. 16, 2019. https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/ (accessed Jun. 21, 2023).

[10] D. Kalita, “An Overview on Long Short Term Memory (LSTM),” *Analytics Vidhya*, Mar. 11, 2022. https://www.analyticsvidhya.com/blog/2022/03/an-overview-on-long-short-term-memory-lstm/ (accessed Jun. 21, 2023).

[11] C.-J. Chen, F.-I. Chou, and J.-H. Chou, “Temperature Prediction for Reheating Furnace by Gated Recurrent Unit Approach,” *IEEE Access*, vol. 10, pp. 33362–33369, 2022, doi: 10.1109/ACCESS.2022.3162424.

[12] *Chapter 8 Exponential smoothing | Forecasting: Principles and Practice (3rd ed)*. Accessed: Jun. 19, 2023. [Online]. Available: https://otexts.com/fpp3/expsmooth.html

[13] “Machine Learning Random Forest Algorithm - Javatpoint.” https://www.javatpoint.com/machine-learning-random-forest-algorithm (accessed Jun. 21, 2023).

[14] C. GOYAL, “Bagging- 25 Questions to Test Your Skills on Random Forest Algorithm,” *Analytics Vidhya*, May 31, 2021. https://www.analyticsvidhya.com/blog/2021/05/bagging-25-questions-to-test-your-skills-on-random-forest-algorithm/ (accessed Jun. 21, 2023).

[15] “Gaussian processes (3/3) - exploring kernels.” https://peterroelants.github.io/posts/gaussian-process-kernels/ (accessed Jun. 21, 2023).

[16] J. Wang, “An Intuitive Tutorial to Gaussian Processes Regression.” arXiv, Apr. 18, 2022. Accessed: Jun. 21, 2023. [Online]. Available: http://arxiv.org/abs/2009.10862

[17] S. Zaheer, N. Anjum, A. Algarni, J. Iqbal, S. Bourouis, and S. S. Ullah, “A Multi Parameter Forecasting for Stock Time Series Data Using LSTM and Deep Learning Model,” *Mathematics*, vol. 11, p. 590, Jan. 2023, doi: 10.3390/math11030590.

[18] “Flow diagram of gradient boosting machine learning method. The ensemble...,” *ResearchGate*. https://www.researchgate.net/figure/Flow-diagram-of-gradient-boosting-machine-learning-method-The-ensemble-classifiers\_fig1\_351542039 (accessed Jun. 21, 2023).

[19] “Introduction to Boosted Trees — xgboost 1.7.6 documentation.” https://xgboost.readthedocs.io/en/stable/tutorials/model.html (accessed Jun. 21, 2023).

[20] D. Chicco, M. J. Warrens, and G. Jurman, “The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation,” *PeerJ Comput. Sci.*, vol. 7, p. e623, Jul. 2021, doi: 10.7717/peerj-cs.623.