MACHINE LEARNING PROJECT REPORT

Course: CS 584

CRYPTO CURRENCY PRICE PREDICTION

Submitted by

DHRUVAL PATEL

CWID:A20549909

JASHESH MEHTA

CWID:A20552899

ZEEL PATEL

CWID:A20556822

Table of Contents

Introduction

In recent years, cryptocurrencies have transformed the way people think about financial transactions and reshaped the traditional financial industry. The emergence of cryptocurrencies has given investors new opportunities to make successful investments. Cryptocurrencies are virtual currencies that rely on blockchain technology, a distributed ledger system that enables safe and open exchanges without the need for a central authority. The decentralized nature of cryptocurrencies makes them less vulnerable to manipulation by financial institutions than traditional currencies. Investors trying to precisely forecast price fluctuations and make profitable investments face problems due to the cryptocurrency market's extreme volatility and unpredictability [43]. To overcome this investors have employed different techniques like performing time-series, fundamental, sentimental, and technical analysis using ML algorithms.

ML is a domain that deals with the development of algorithms and statistical models that enable computer systems to learn from data and improve their performance on a specific task. Some of the tasks of ML include classification, regression, anomaly detection, and Natural Language Processing (NLP). Making informed investment decisions can be facilitated by the application of ML in price prediction, which can offer useful insights into market dynamics . With the use of historical data analysis, ML algorithms may spot patterns, trends, and correlations that human analysts would miss. ML models can capture trend and momentum information and forecast future price movements by utilizing a variety of techniques, including time series analysis and incorporating technical indicators like the SMA, EMA, and RSI, among others .

The accuracy of price prediction in the cryptocurrency market can be improved by using technical indicators such as the SMA, EMA, and RSI as input. Both the SMA and EMA are trend-following indicators that are used to amplify price data and pinpoint the trend's direction. The momentum oscillator RSI, on the other hand, gauges how strongly prices fluctuate

Many researchers have used ML algorithms for price prediction of cryptocurrency. As per our knowledge, there lacks a detailed comparative analysis of machine learning algorithms for long-term cryptocurrency price prediction where technical indicators like RSI, EMA, SMA are used as input features which is a significant gap in the current research. Utilizing technical indicators derived from historical data and forecasting future cryptocurrency prices, could assist investors in making informed investment decisions in this unpredictable market.

This thesis aims to explore the application of ML algorithms, including time series analysis, in predicting the future prices of cryptocurrencies, with a special focus on Bitcoin. In specific this thesis aims to explore and find out which ML algorithms are better suited for making long-term predictions of Bitcoin prices while using technical indicators as input features and conduct an experiment to identify the most accurate model among the identified ML algorithms. Bitcoin being the oldest cryptocurrency to ever exist has a large amount of historic price data which would help in training the ML models to gain better performance.It also aspires to find out the impact of each input features on the ML models' performance.

Abstract

Due to its decentralized nature and opportunity for substantial gains, cryptocurrency has become a popular investment opportunity. However, the highly unpredictable and volatile nature of the cryptocurrency market poses a challenge for investors looking to predict price movements and make profitable investments. Time series analysis, which recognizes trends and patterns in previous price data to create forecasts about future price movements, is one of the prominent and effective techniques for price prediction. Integrating Machine learning (ML) techniques and technical indicators along with time series analysis, can enhance the prediction accuracy significantly.

Cryptocurrency price prediction is a complex task that involves analyzing a variety of factors, including historical price data, market capitalization, trading volumes, and social media sentiment. Several methods can be used to predict cryptocurrency prices, including technical analysis, fundamental analysis, and machine learning.Technical analysis involves analyzing historical price data to identify patterns and trends that may indicate future price movements. Fundamental analysis involves assessing the underlying value of a cryptocurrency by considering factors such as its technology, team, and use cases. Machine learning involves using algorithms to analyze large amounts of data to identify patterns and make predictions.

Aim and Objective

Aim :

The aim of this thesis is to compare the machine learning algorithms for the price prediction of Bitcoin while using technical indicators as inputs. The comparison of the algorithms will be done based on chosen evaluation metrics to find out which features in the dataset has the most significant effect on the predictions of each model.

Objectives :

1. To find out the ML algorithms that can be used for the prediction of Bitcoin prices.

2. To compare the predictions of each ML algorithm using various evaluation metric scores.

3. To find out which among the technical indicators taken as input has more effect on the prediction of each model.

DESCRIPTION OF DATASET

In order to accomplish the goal of the current study, cryptocurrency price prediction was acquired via Kaggle. This dataset encompasses historical information on cryptocurrency prices, market capitalization, trading volumes, and other pertinent factors

The dataset which we used for cryptocurrency price prediction has more than 44,000 entries which could help the model in predicting the future prices of cryptocurrency. The dataset includes information on exchanges, volume, high, low, close, and several other columns. A total of ten cryptocurrencies, such as ADA, BTC, and XRP, are included in the dataset

Overall, using a large dataset for cryptocurrency price prediction can provide several benefits, including increased accuracy, reduced overfitting, improved generalizability, and more insightful predictions.

List Of Algorithms

Back ground

Concepts linked to the research for this thesis are covered in this section. It provides a quick introduction to the idea of machine learning and talks about the pertinent algorithms utilized. The evaluation metrics and technical indicators are also highlighted.

Machine Learning

ML, a sub-field of Artificial Intelligence (AI), is an interdisciplinary field that focuses on the development of computational algorithms and models that can autonomously learn and improve from data. It encompasses a set of statistical techniques and algorithms that enable computer systems to analyze and interpret complex patterns and relationships within data, without being explicitly programmed.
ML has a wide range of applications, some of the included fields are: Computer vision- tasks like object detection, and object recognition. Prediction- tasks like the prediction of future trends, classification, analysis, and recommendation based on historic data. NLP involves the analysis and understanding of human language.
ML techniques are used for tasks such as sentiment analysis, text classification, language translation, chatbots, voice assistants, and text generation.
ML has made considerable strides in recent years thanks to the availability of big datasets, high computing power, and innovations in algorithmic techniques like Deep Learning (DL), AI, etc.

Hyperparameters

The ML models have certain variables that determine the learning process, they are known as Hyperparameters. The selection of these variables has a significant effect on the model's performance. Therefore, the hyperparameters for the models must be selected carefully. Hyperparameter tuning is the process where the model is tested with varied values of hyperparameters to find out the most optimal parameters for the model

Deep Learning

DL is a subfield of machine learning. It mainly focuses on training neural networks with multiple levels to identify and represent complex patterns and relationships in the data. Since deep learning deals with a range of neural network architectures, it is also known as Deep Neural Networks. DL models are designed to automatically extract hierarchical features from input data, enabling them to learn intricate patterns and relationships.

DL is effective for deciphering and obtaining knowledge from both massive amounts of data and data gathered from many sources . Some popular DL architectures include Convolutional Neural Networks (CNN)s for image and video processing, Recurrent Neural Networks (RNNs) for sequential data processing, and Generative Adversarial Networks (GANs) for generating new data sample

Recurrent Neural Networks

Recurrent Neural Network (RNN) is one of the most popular deep learning architectures and is used for a variety of tasks, including speech recognition, time series forecasting, creating image descriptions, video tagging, and many more.
RNN as the name suggests has cycles where the information is transmitted back into itself, thereby taking into account the previous input along with the current input. This enables RNNs to handle sequential data. When training an RNN, the network's parameters (weights and biases) are adjusted through a process called back-propagation, where the error signal is propagated backward from the output to the input layers. During this process, gradients are calculated, representing the rate of change of the error with respect to the network's parameters. These gradients are then used to update the parameters and improve the network's predictions.
Even though RNNs have successfully overcome the limitations of feed-forward neural networks, they suffer from the vanishing gradient problem where the gradients calculated during back-propagation diminish or "vanish" as they propagate backward through the layers of the network.

LSTM (Long Short-Term Memory)

Long Short-Term Memory (LSTM) is a recurrent neural network (RNN) design utilized extensively for analyzing and forecasting time series tasks. It is particularly useful for handling the vanishing gradient problem in traditional RNNs, which can make them less effective at capturing long-term dependencies in sequential data. The LSTM networks are composed of memory cells arranged in a sequence and connected through gates. These gates regulate the inflow and outflow of information in the cells. These gates include the input gate, forget gate, and output gate, which allow the network to selectively update and output information from each cell. A range of applications have employed LSTM networks, such as speech recognition, natural language processing, and image captioning. One area where they have shown promise is in the prediction of cryptocurrency prices. To use LSTM for cryptocurrency price prediction, the network is trained on historical price data, with the goal of learning to predict future prices based on past trends. The input to the network consists of a sequence of past prices, and the output is the predicted price at some point in the future.

LSTM Algorithm

1.Initialize the input sequence Y = [y1, y2, ..., yn] of length n

2.Initialize the hidden state hs and the cell state c0 to zero vectors of dimensionality m

3. For each timestep t from 1 to n, do the following:

a) Forget gate fg = σ(Uf * [h-1, yt] + b1), where Uf and b1 are learnable parameters and σ is the sigmoid function

b) Input gate ig = σ(Ui * [h-1, yt] + b2).2

c) Candidate cell state ĉs = tanh(Uc *[h-1, yt]+ b3).

d) Cell state cs = fg * cs-1 + ig * ĉs.

e) Output gate og = σ(Uo * [h-1, yt] + b4).

f) Hidden state hid = og * tanh(cs).

4. Output the final hidden state hid

Evaluation Metrics:

MSE:

MSE is a commonly used metric to measure the difference between the predicted and actual values of a regression problem [24]. It is calculated by taking the average of the squared differences between the predicted and actual values. A lower MSE value represents better performance.

The formula to calculate MSE is:

M SE = 1/n n∑i=1 (Yi − ˆYi)2

Where,

Yi is the actual value,

ˆYi is the predicted value,

n is the number of observation

R-Squared:

R2 (pronounced as "R-squared") is a statistical measure that represents the proportion of the variance in a dependent variable that can be explained by the independent variables in a regression model [24]. R2 is a measure of how well the regression model fits the data.

The value of R2 ranges from 0 to 1, with higher values indicating a better fit of the model to the data. An R2 value of 0 means that the model explains none of the variability of the response data around its mean, while an R2 value of 1 indicates that the model perfectly predicts the response data.

The formula to calculate R^2 is:

R^2 = 1 – SSR/SST

SSR (Sum of Squared Residuals) is the sum of the squared differences between the predicted values and the actual values.

SST (Total Sum of Squares) is the sum of the squared differences between the actual values and the mean of the target variable.

Training The Models

LSTM:

The LSTM neural network is built using Keras, a well-known DL library. The model is initially set up as an empty sequential model, allowing for the sequential addition of layers. A dense output layer with one unit and two LSTM layers, each with 50 lstm_units, is then added to the model. The loss function utilized is MSE, and the model is optimized using the Adam optimizer. Three technical indicators—RSI, EMA, and SMA—that are employed to forecast the price of Bitcoin make up the input data. After that, the input data is reconfigured to have three time steps and one feature per time step (3,1).

The target variable or Bitcoin price is represented by the output data, which is a 1D tensor with the same length as the input data. The fit method uses input data, target data, epochs (50) which tell how many times the model iterates through the complete training dataset, and batch size (32) which refers to the number of samples utilized in each training iteration, as inputs to train the model. Based on the difference between its predictions and the actual target values, the model modifies its weights and biases during training

Algorithm 3:

LSTM model training and evaluation

1: Input: Training data (Xtrain, ytrain), testing data (Xtest, ytest), hyperparameters

(nunits, nepochs, nbatch)

2: Output: LSTM model performance metrics

3: Initialize LSTM model lstm

4: Add LSTM layer with nunits units, return sequences, and input shape (3, 1)

5: Add LSTM layer with nunits units

6: Add Dense layer with 1 unit

7: Compile model with optimizer Adam and loss function mean squared error

8: Train model with nepochs epochs and batch size nbatch on training data

(Xtrain, ytrain)

9: Evaluate model on testing data (Xtest, ytest)

10: Predict prices on testing data using model lstm

11: Compute LSTM model RMSE, MSE, MAE, and R2 performance metrics

12: Compute LSTM model TWAP and VWAP performance metrics

13: return LSTM model performance metric

GRU:

Similar to the LSTM algorithm, GRU is also built using the Keras library. The training process of GRU is similar to that of LSTM, an empty sequential model is first created, and then 2 GRU layers with 50 units each and a dense output layer with 1 unit are added. The model is then fitted using input and target data. The input data which contains RSI, EMA, and SMA values, is reshaped to have 3 time steps and 1 feature per time step, yielding the 3D tensor of shape (number of samples, 1) as the input data. A 1D tensor of price values makes up the target data. The training is carried out with a batch size of 32 over 50 epochs

Algorithm 4 :GRU model training and evaluation.

1: Input: Training data (Xtrain, ytrain), testing data (Xtest, ytest), hyperparameters

(nunits, nepochs, nbatch)

2: Output: GRU model performance metrics

3: Initialize GRU model gru

4: Add EMA layer with nunits units, return sequences, and input shape (3, 1)

5: Add GRU layer with nunits units

6: Add Dense layer with 1 unit

7: Compile model with optimizer Adam and loss function mean squared error

8: Train model with nepochs epochs and batch size nbatch on training data

(Xtrain, ytrain)

9: Evaluate model on testing data (Xtest, ytest)

10: Predict prices on testing data using model gru

11: Compute GRU model RMSE, MSE, MAE, and R2 performance metrics

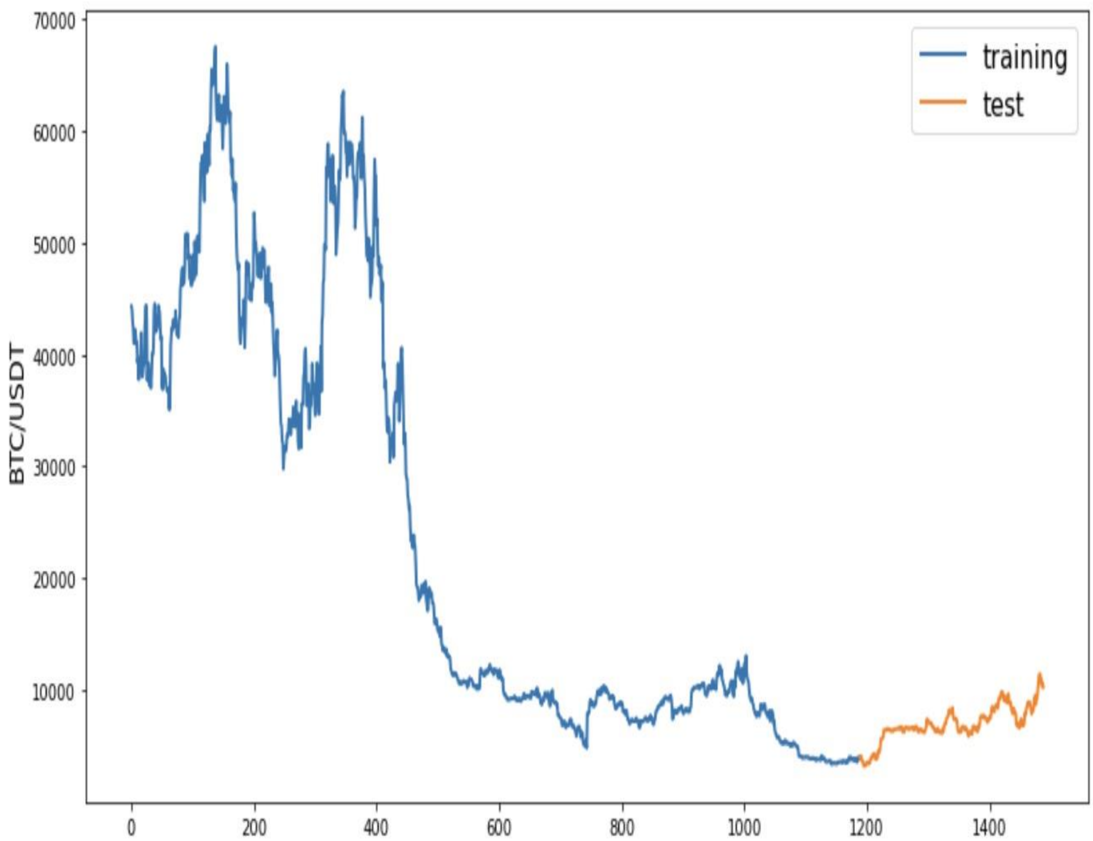
12: Compute GRU model TWAP and VWAP performance metrics

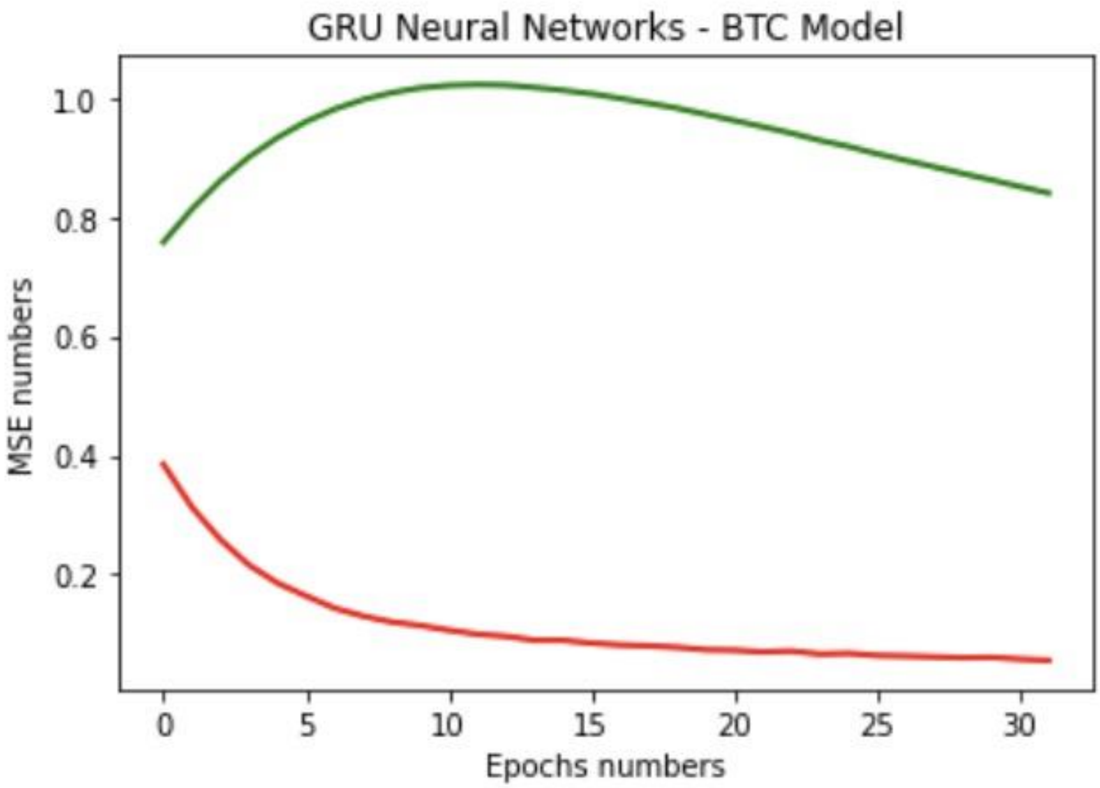
13: return GRU model performance metric

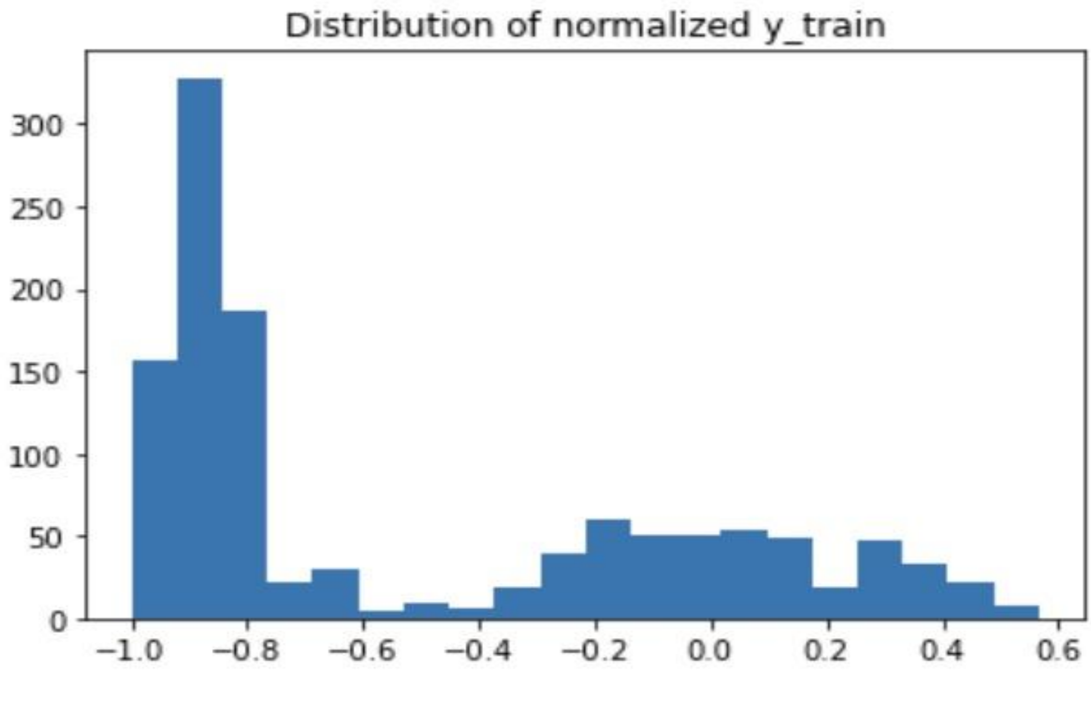
Project Link

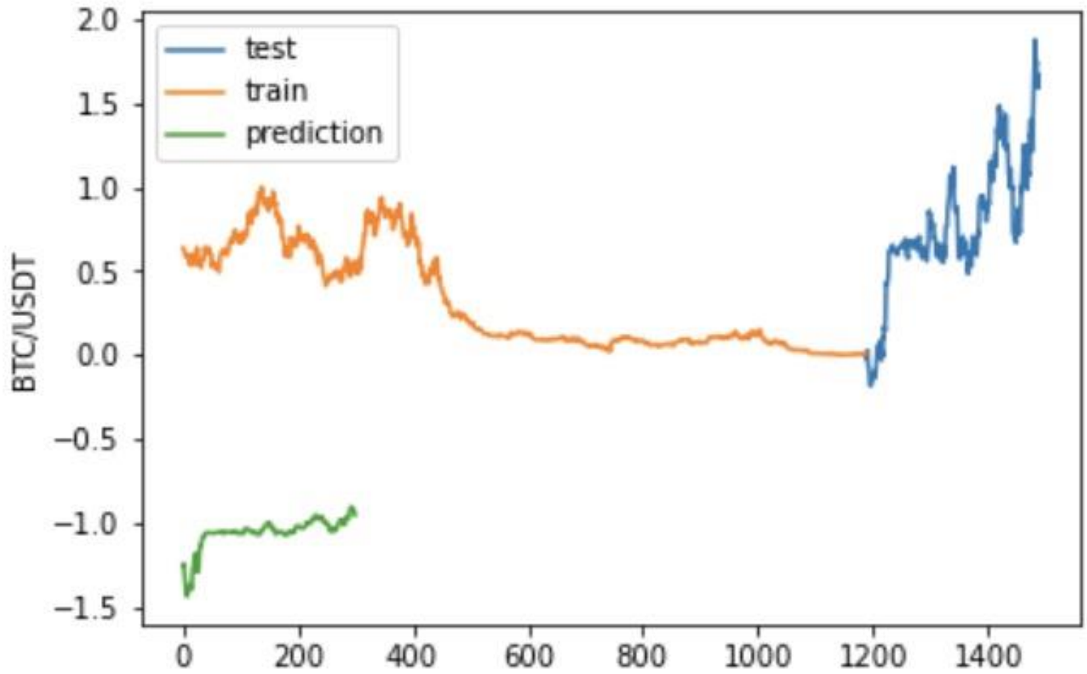
(https://github.com/Jackspa1160/Cryptocurrency_price_prediction)

Results:



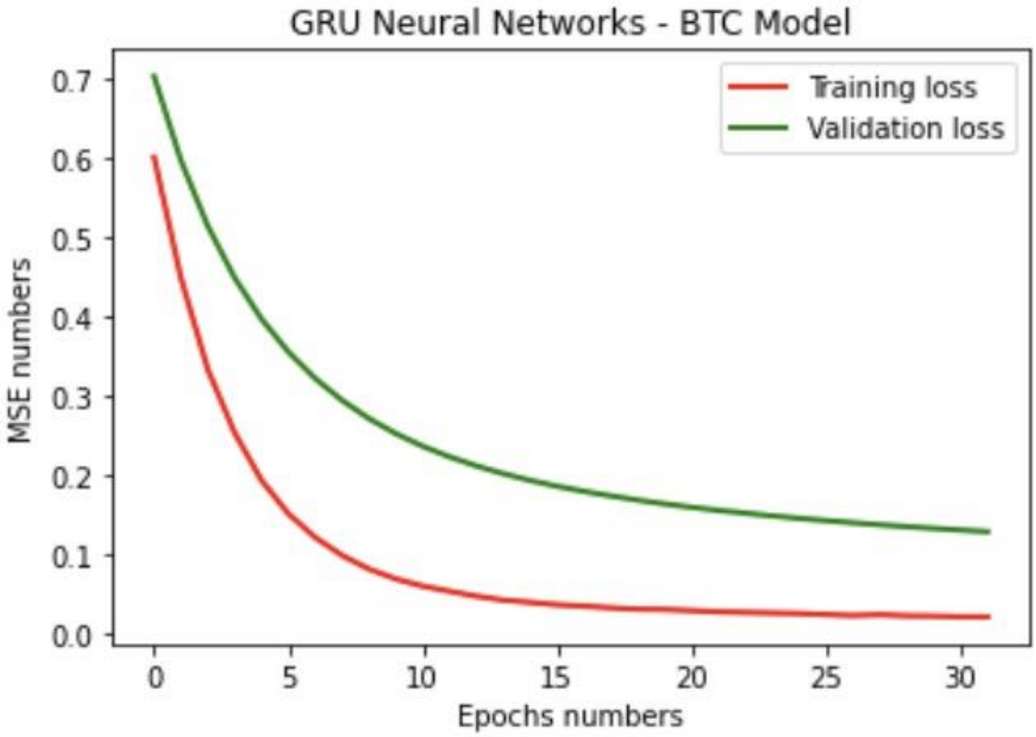

GRU Neural Networks - BTC Model
MSE numbers
1.0
0.8
0.6
0.4
0.2
0
5
10
15
20
25
30
Epochs numbers


Distribution of normalized y_train
300
250
200
150
100
50
0
−1.0
−0.8
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6



27 March Prediction = **46351.24609375 BTC/USDT**

27 March 02.22AM (Istanbul Time) = **46564.0000 BTC/USDT**




```
10/10 [==============================] - 0s 2ms/step
Mean Squared Error (MSE): 0.12936864640616208
R-squared (R2) Score: 19.239333619458233%
```