Note: This was originally a project for my final university assignment. I handled the code while my friends were responsible for this report.

Many thanks to Huynh Kim Ngan and Le Pham Nhat Vy for helping me. This project would not be possible without your effort.

**TABLE OF CONTENT**

[**INTRODUCTION** 3](#_Toc155050959)

[**QUESTION 1** 4](#_Toc155050960)

[1.1 Overview of the dataset 4](#_Toc155050961)

[1.2 Brief analysis of the dataset in the period of 02/12/2022 to 02/12/2023 5](#_Toc155050962)

[**QUESTION 2** 10](#_Toc155050963)

[2.1 Linear regression model 11](#_Toc155050964)

[2.2 SGD Regressor model 12](#_Toc155050965)

[2.3 Lasso model 13](#_Toc155050966)

[2.4 ElasticNet model 13](#_Toc155050967)

[2.5 Ridge model 14](#_Toc155050968)

[2.6 SVR model 15](#_Toc155050969)

[2.7 NuSVR model 15](#_Toc155050970)

[**QUESTION 3** 17](#_Toc155050971)

[3.1 Unidirectional LSTM 19](#_Toc155050972)

[3.2 Bidirectional LSTM 20](#_Toc155050973)

[3.3 Two-path LSTM 21](#_Toc155050974)

[3.4 Results of the LSTM models 22](#_Toc155050975)

[3.5 Hyperparameter tuning 24](#_Toc155050976)

[**QUESTION 4** 27](#_Toc155050977)

[**CONCLUSION** 29](#_Toc155050978)

[**REFERENCE** 30](#_Toc155050979)

# **INTRODUCTION**

The market for cryptocurrencies, led by Bitcoin, has grown significantly in popularity in recent years. With its use of encryption, cryptocurrency is a digital or virtual money that has the power to completely transform the financial sector by offering a decentralised, safe method of conducting business. Additionally, it may be used to speed up and reduce the cost of cross-border transactions. Therefore, cryptocurrencies have the potential to foster financial inclusion, providing access to banking services for the unbanked and underbanked populations worldwide.

However, due to its speculative nature and sharp price volatility, the cryptocurrency market has attracted the attention of traders and investors and emerged as speculative asset rather than a stable means of exchange.

Bitcoin is the first and most well-known cryptocurrency. Given that Bitcoin's price tends to be volatile and sensitive to a wide variety of circumstances, predicting its changes in value presents a unique difficulty. Our project aims to forecast its values by using Linear Regression and Long Short-Term Memory (LSTM) including Bidirectional LSTM and 2path LSTM. Also, The project will compare the performance of the Linear Regression model against the LSTM models. By evaluating the accuracy and capabilities of these models, we seek to contribute valuable insights into the effectiveness of traditional and advanced machine learning techniques in forecasting the Bitcoin prices.

The report of the project includes 4 parts:

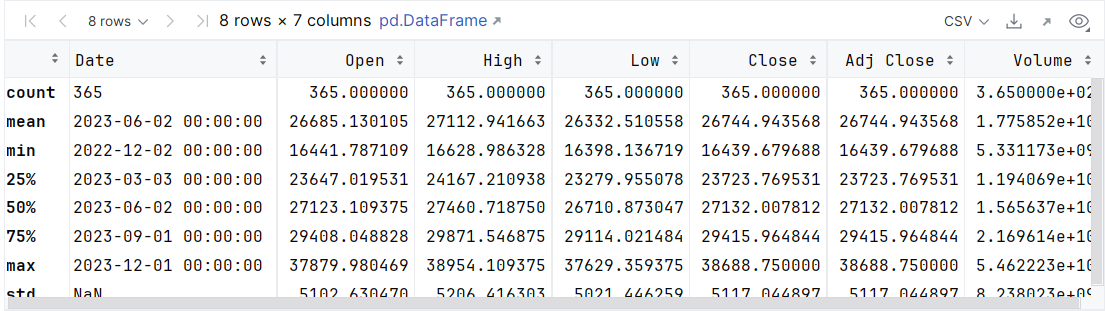
1. Question 1: Brief description about the data used in the project from Yahoo! Finance with descriptive statistics and plots
2. Question 2: Traditional data analysis using Linear Regression method on the data set and remarks on the model
3. Question 3: Analysis using LSTM Unidirectional, LSTM Bidirectional and LSTM 2path on the data set and remarks on the model
4. Question 4: Comparison, assessment of the models used and conclusions from these models

# **QUESTION 1**

## 1.1 Overview of the dataset

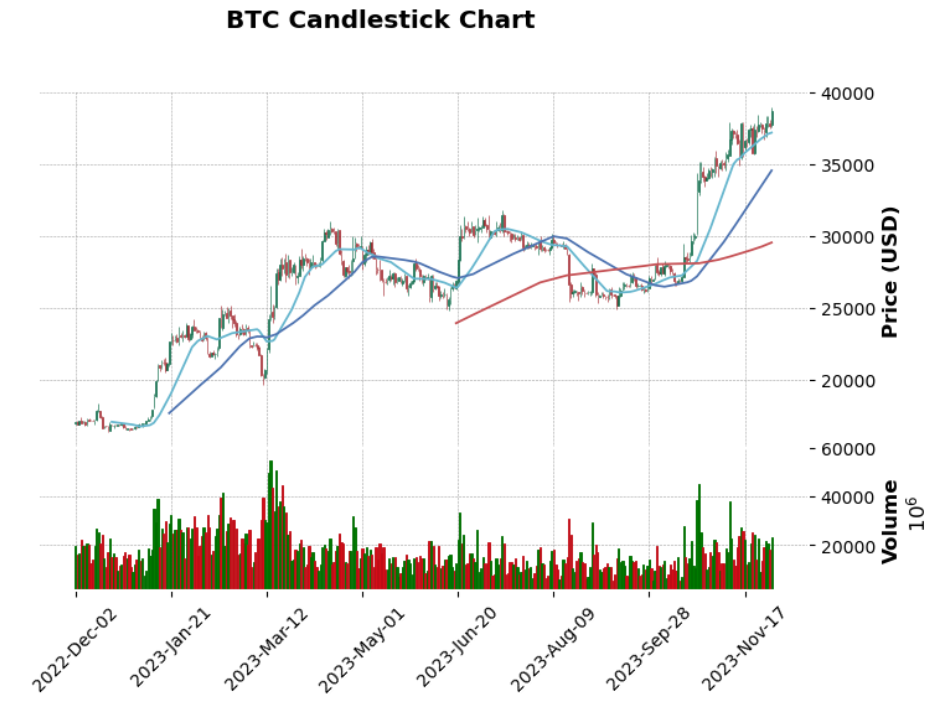
Unlike normal stocks, Bitcoin are traded 24/7, this means that there is no real “Open” or “Close” price of Bitcoin. However, to track the price of Bitcoin, Yahoo Finance and many other crypto trading platforms decided to take a snapshot of Bitcoin price at a certain time of the day. The opening prices are recorded at 12:00 AM (00:00) and closing prices are recorded at 11:59 PM (23:59) UTC time.

The dataset of our project includes Bitcoin opening and closing prices with volume traded in the period of 02/12/2022 to 02/12/2023, presented as following.



*Figure 1.1: Description of the dataset*

## 1.2 Brief analysis of the dataset in the period of 02/12/2022 to 02/12/2023



*Figure 1.2: BTC Candlestick Chart*

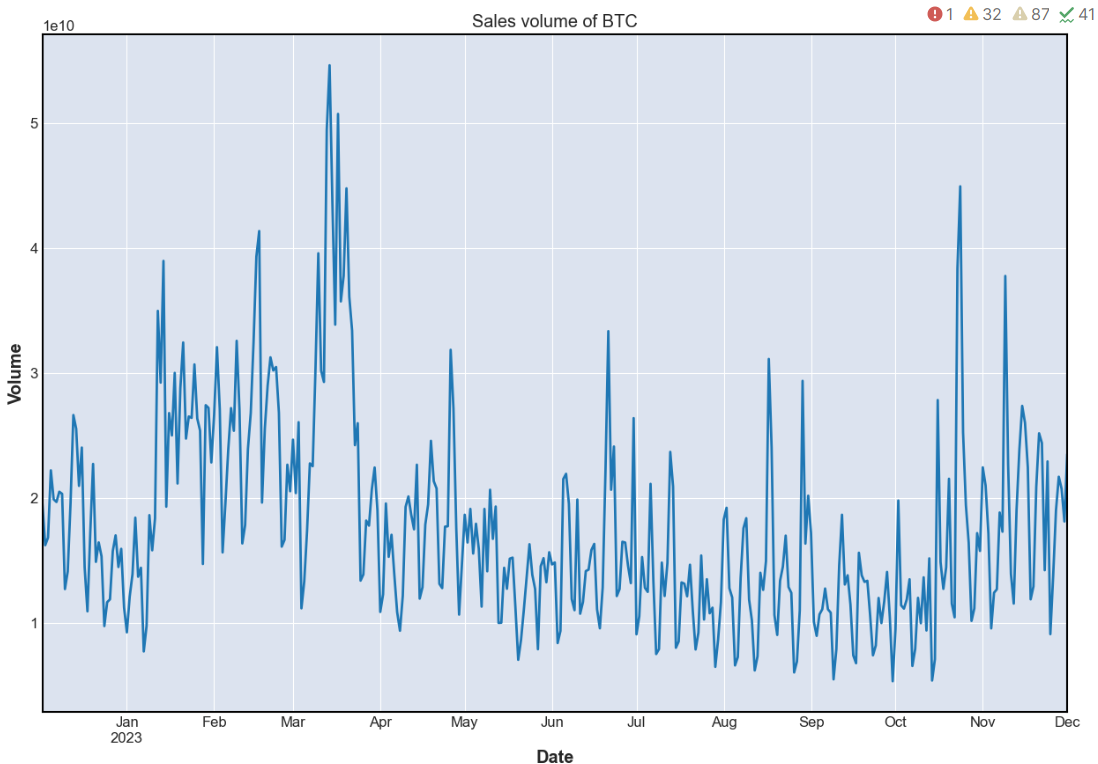
The data demonstrates a noteworthy degree of volatility, as prices undergo both positive and negative fluctuations. There is a discernible long-term growth tendency in the dataset. The values of Bitcoin typically exhibit an upward trend, suggesting that it has the potential to be a valuable store of wealth. The steady rise could be attributed to growing acceptance, institutional curiosity, and acknowledgment of Bitcoin as a digital asset.



*Figure 1.3: Closing price of BTC*

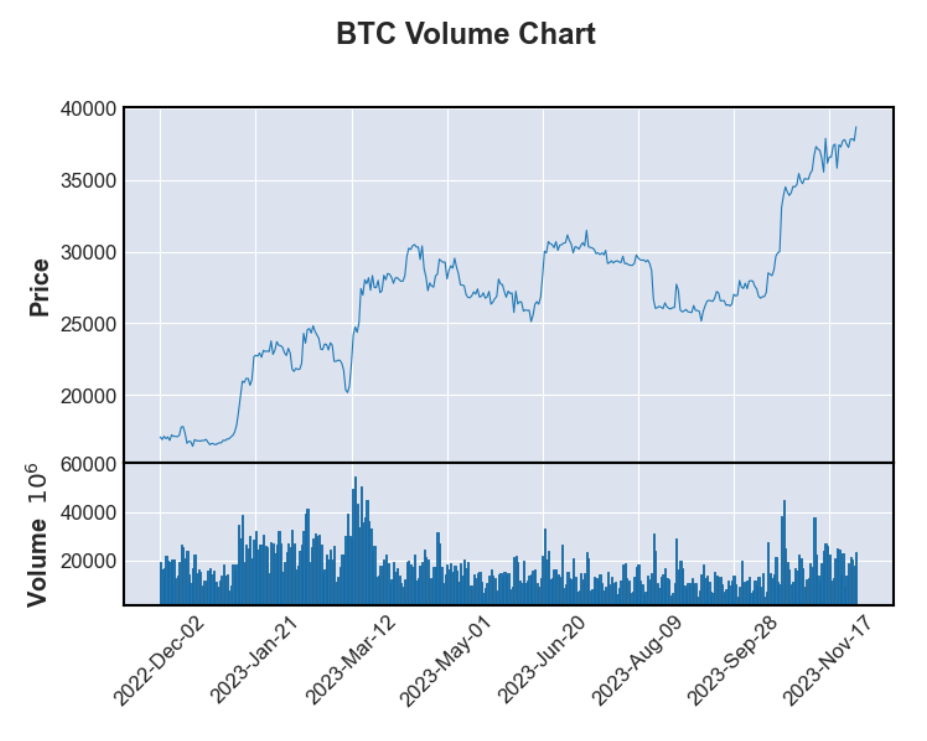
The closing values of Bitcoin appear to be generally trending upward, notwithstanding brief oscillations. The prices typically rise from USD17,088 to USD39,476 USD from 02/12/2022 to 02/12/2023. This general upward trend may be a sign of rising interest and money invested in Bitcoin.

The latter pattern of the dataset shows a significant spike in the closing values of Bitcoin, around November and December 2023. Numerous factors, such as growing institutional adoption, optimistic market mood, or macroeconomic conditions, could have an impact on this.



*Figure 1.4: Sales volume of BTC*

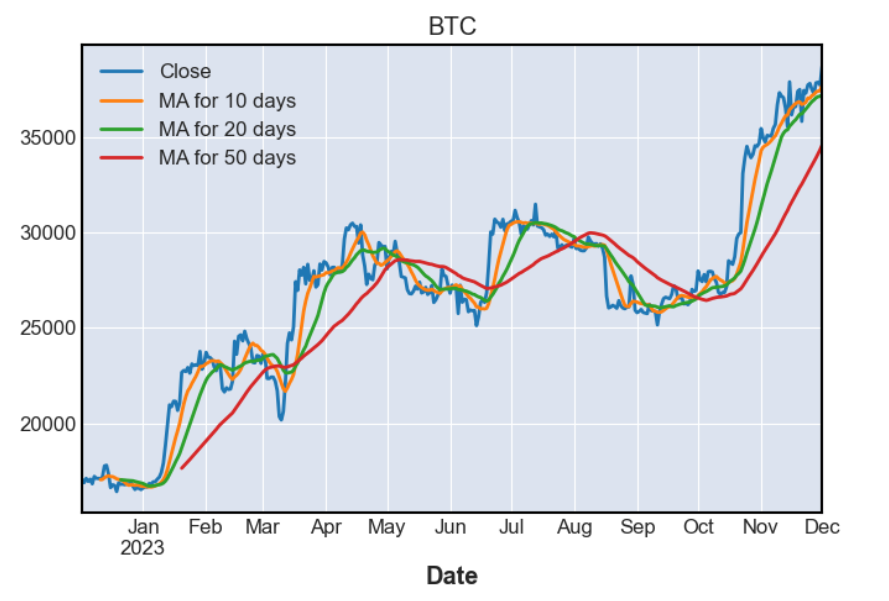
Bitcoin trading volume varies greatly day-to-day, from 5,331,172,801 USD to 54,622,230,164 USD. This suggests a vibrant and bustling market. Periods with continuously high trading volumes, like mid-March to mid-April 2023, may indicate a period of considerable market activity or increased interest. For example, on January 14, February 16 and March 18, the trading volume peaks, these increases may be linked to noteworthy news, events in the market, or shifts in investor mood.



*Figure 1.5: BTC Volume Chart*

The dataset shows high trade volumes and a period of significant volatility with widely varied closing prices. Notably, there are days like October 23, 2023, and October 25, 2023, when both the closing price and traded volume are high. These times may be indicative of substantial buying activity and considerable market interest, which would support price increases.

On the other hand, there have been occasions, such as May 11, 2023, and June 11, 2023, where the closing prices have peaked but the trading volumes have not been very high. This would point to a possible lack of consistent buying interest at those particular price peaks.



*Figure 1.6: Moving average of BTC price*

Comparing between the closing price line and the moving average lines, we see that the closing price line is mostly above the moving average lines. This shows that the trend is trending upward. Besides, MA-10 days is higher than MA-20 days and MA-20 days is higher than MA-50 days, further reinforcing the upward trend in both short-term and long-term.

# 

# **QUESTION 2**

Linear Regression is one of the traditional data analysis using a standard econometric method to analyze and predict the Closing Price of Bitcoin, we have used several regression models, such as Linear regression, SGDRegressor, Lasso, etc… to see how well they can perform.

The general form of each type of regression model is:

Where:

* is the dependent variable (target),
* is the independent variable (feature),
* is the intercept (y-intercept),
* is the slope (coefficient) of the independent variable,
* is the error term, representing unobserved factors affecting Y.

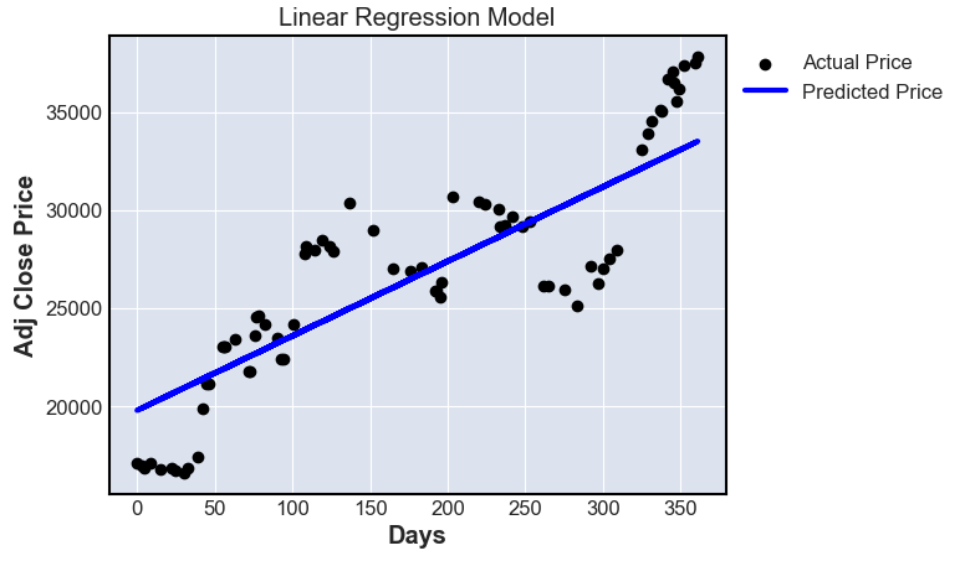
Before running the regression analysis, it is noted that these models are linear, while the price of the Bitcoin fluctuates with a huge volatility. Therefore, it can be expected that these models cannot perform well. To assess the effectiveness of the models used, we will use accuracy as a criterion. The Accuracy is calculated by subtracting RMSPE from 1, with RMSPE calculated as follows:

RMSPE=

where:

* is the total number of observations,
* is the actual value for the ()th observation,
* is the predicted value for the (i)th observation

## 2.1 Linear regression model



*Figure 2.1: Linear regression model*

The first model, linear regression, is used to estimate 'Adj Close' prices based on the number of days since the start of the dataset. We can see that the predict prices are not on the actual prices line, indicating that the accuracy of prediction by this model is not high. Indeed, the accuracy of the linear regression model is lower than 90%, specifically about 88.24%

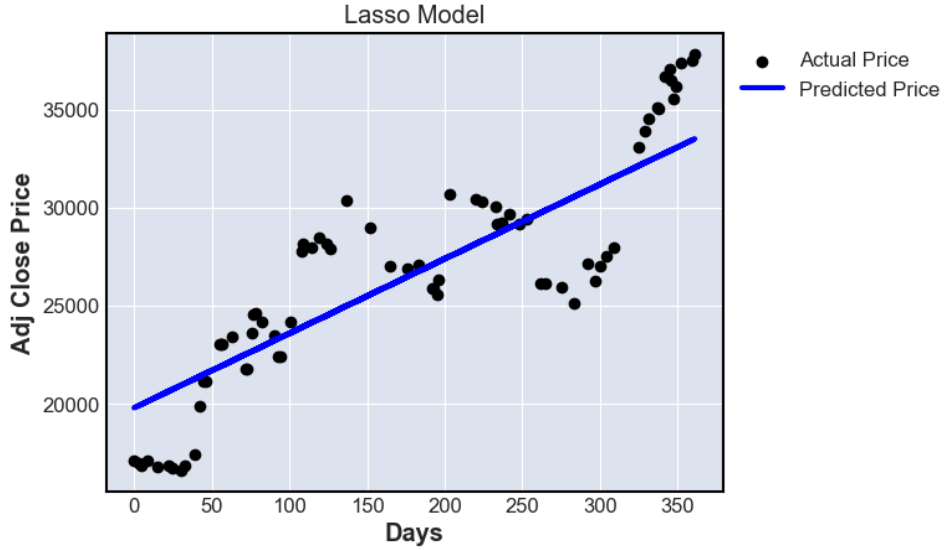
## 2.2 SGD Regressor model



*Figure 2.2: SGDRegressor model*

Coming to the second model, SGDRegressor is a versatile linear regression model that is particularly useful in scenarios where large datasets. It allows for efficient optimization using stochastic gradient descent, and its parameters can be tuned for regularization and other hyperparameters. From observation, we can see that SGDRegressor is similar with the forecast points being mostly outside the actual line. The accuracy of SGDRegressor is about 88.26%, only slightly better than linear regression.

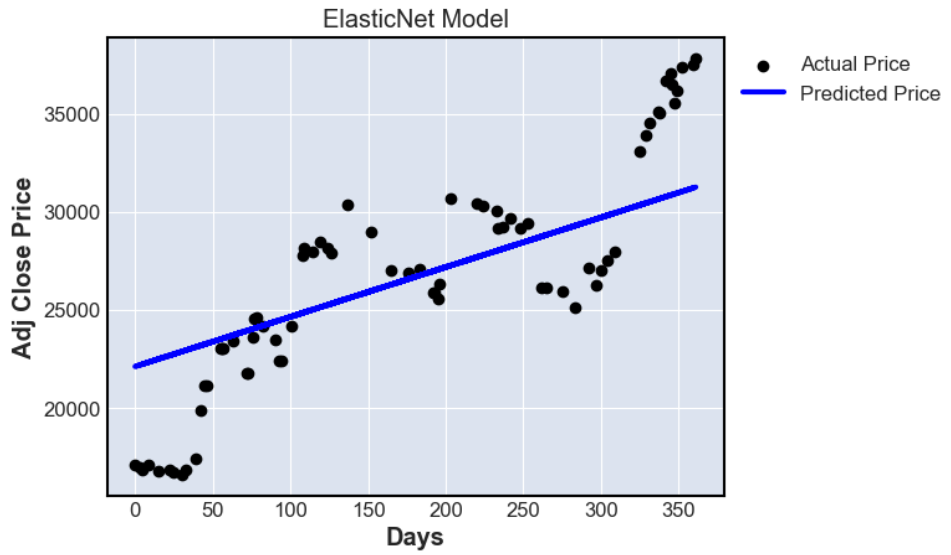
## 2.3 Lasso model



*Figure 2.3: Lasso model*

Lasso regression is a linear regression technique that incorporates L1 regularization into the cost function. When applying this model to the dataset, like the two models above, the predicted points are outside the actual line. Lasso regression only has about 88.24% accuracy.

## 2.4 ElasticNet model

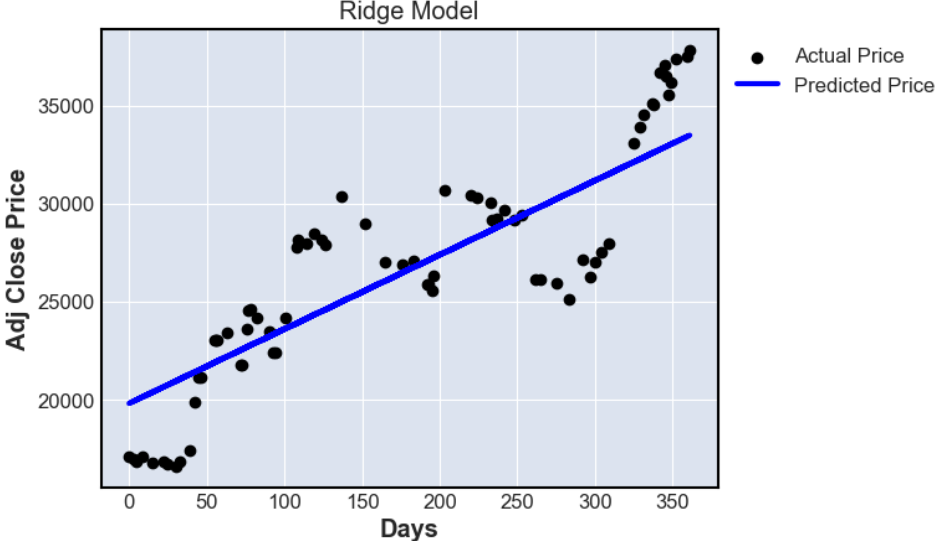


*Figure 2.4: ElasticNet model*

ElasticNet is a regularization technique that combines both L1 (Lasso) and L2 (Ridge) regularization terms in the linear regression cost function. The accuracy of

Elastic Net model is also not too high, about 84.58%

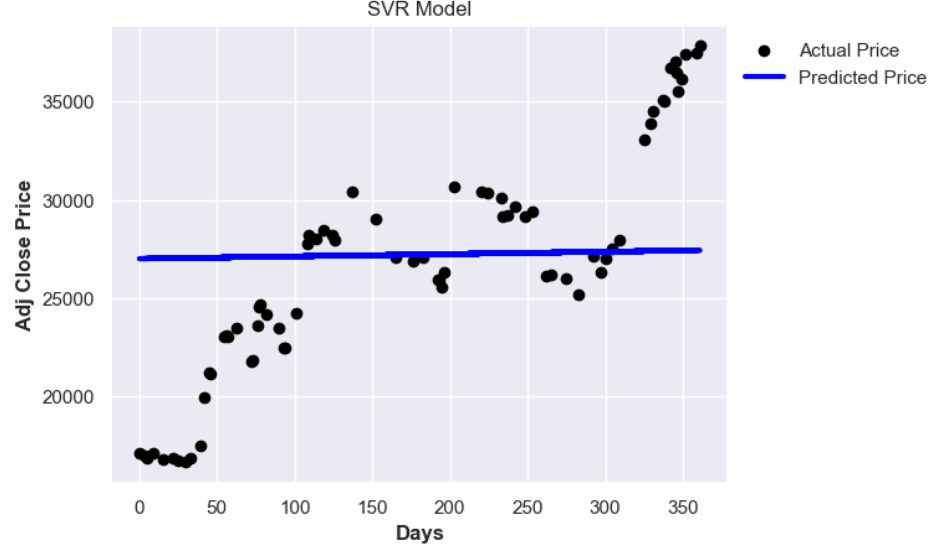
## 2.5 Ridge model



*Figure 2.5: Ridge model*

Ridge regression, also known as L2 regularization, is a linear regression technique that extends ordinary least squares (OLS) regression by adding a regularization term to the cost function. The observation shows that the prediction is relatively positive but the accuracy is only 88.21%.

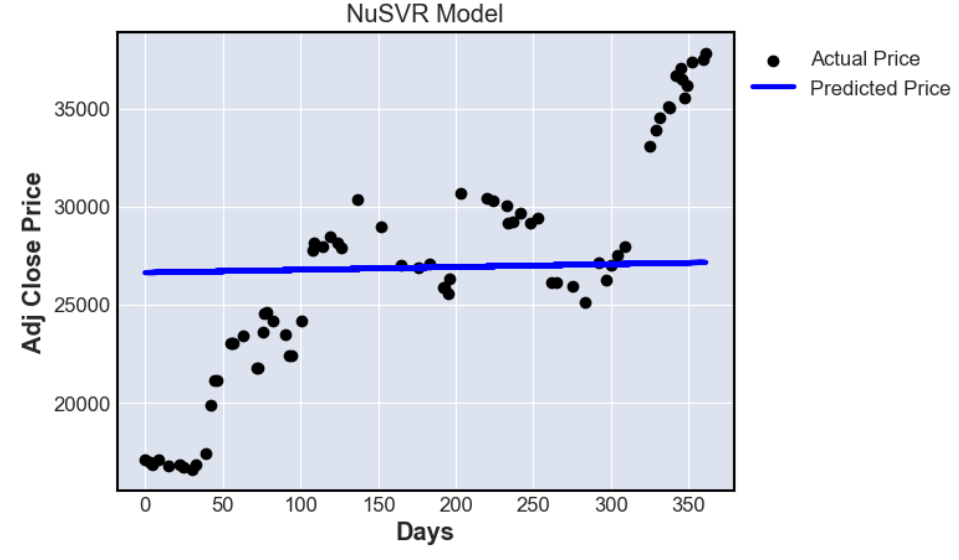
## 2.6 SVR model



*Figure 2.6: SVR model*

Support Vector Regression (SVR) is a regression technique that is an extension of Support Vector Machines (SVM) for classification. SVR is used for predicting a continuous target variable rather than class labels. When applied to this dataset, we see that the fit of the model with the dataset is not high, specifically, the predicted points are almost not on the actual line and the prediction accuracy is also low, only 73.56%.

## 2.7 NuSVR model



*Figure 2.7: NuSVR model*

NuSVR model is a variant of Support Vector Regression (SVR) that introduces an additional parameter ν(nu), to control the number of support vectors and the training error tolerance. Like the SVR model, NuSVR is also not suitable for this dataset and only produces a fairly low accuracy of about 74.29%.

Summarizing the above linear regression models, we see that the most suitable model in Linear regression with this dataset is the SGDRegressor model with the highest accuracy of about 88.26%.

# 

# **QUESTION 3**

To analyze the dataset, we decided to use Long Short-Term Memory as the AI models. Long Short-Term Memory (LSTM) models are a type of recurrent neural network (RNN) architecture that are designed to remember information for long periods of time. They are particularly useful for processing sequential data, where the order and context of the data points matter, such as time series data, sentences, or audio.

An LSTM cell consists of three main components: the input gate, the forget gate, and the output gate. Each of these gates plays a role in the updating and maintaining of the cell's internal state.

* **Input Gate**: This gate decides what new information will be stored in the cell state. It uses a sigmoid function to decide which values to let through (0 means "let nothing through", 1 means "let everything through"). A tanh layer creates new candidate values that could be added to the state.
* **Forget Gate**: This gate decides what information should be discarded from the cell state. It also uses a sigmoid function to output values between 0 and 1, with 0 meaning "completely forget" and 1 meaning "completely remember".
* **Cell State**: The cell state is updated based on the decisions from the input and forget gates. It forgets the information that the forget gate decided to forget, and then adds the new candidate values scaled by how much was decided to be updated at the input gate.
* **Output Gate**: This gate decides what the next hidden state should be. The hidden state contains information on previous inputs. The hidden state is also used for predictions.

The LSTM cell processes data sequentially, maintaining an internal state that encapsulates information about the temporal dependencies between the data points. This allows it to effectively capture and utilize the context from earlier in the sequence, which can be crucial for understanding the overall sequence.

In the context of time series forecasting, an LSTM model would take in a sequence of data points from the past to predict the future value. The LSTM cells would process these data points one at a time, updating their internal state at each step to capture the temporal dependencies in the data. This makes them well-suited for tasks like predicting stock prices or weather patterns, where the future value is often dependent on the trends and patterns in the past data.

The models used in this analysis are Unidirectional LSTM, Bidirectional LSTM, and Two-path LSTM. These LSTMs are quite similar in the working mechanism, with some notable differences in how they handle the data.

Long Short-Term Memory (LSTM), Bidirectional LSTM, and Two-Path LSTM are all variants of Recurrent Neural Networks (RNNs) that are designed to handle sequence prediction problems. They share similarities but also have distinct differences.

**Similarities**

* **Sequence Learning**: All three models are capable of learning from sequences of data. This makes them suitable for tasks such as time series prediction, natural language processing, and speech recognition.
* **Memory of Past Information**: They all have a form of memory that allows them to remember past information, which is crucial for sequence prediction tasks. This is achieved through the use of gates in their architecture.
* **Handling of Long Sequences**: They are all designed to handle the vanishing gradient problem, a common issue in traditional RNNs that makes it difficult to learn and tune the model parameters. This is particularly important when dealing with long sequences of data.

**Differences**

* **Direction of Information Flow**: In a standard LSTM, information flows in one direction, from the past to the future. In contrast, a Bidirectional LSTM has two layers of LSTMs, one processing the data from past to future (forward layer) and the other from future to past (backward layer). This allows the model to have access to information from both past and future states at any point in time. A Two-Path LSTM, on the other hand, has two separate paths for the input and the recurrent transformations, which can help in learning different types of dependencies in the data.
* **Complexity and Computational Requirements**: Bidirectional LSTMs and Two-Path LSTMs are generally more complex and computationally intensive than standard LSTMs due to their additional layers and paths.
* **Performance**: While all three models can perform well on sequence prediction tasks, their performance can vary depending on the specific task. For example, Bidirectional LSTMs can perform better on tasks where context from both past and future is important, such as in language translation. Two-Path LSTMs can be beneficial in tasks where different types of dependencies need to be captured separately.

## 3.1 Unidirectional LSTM

First, we define a Long Short-Term Memory (LSTM) model for sequential data processing. The Model\_LSTM class is initialized with various parameters, including learning rate, number of layers, input size, layer size, output size, and forget bias.

Inside the class, a function named lstm\_cell is defined to create an LSTM cell with a specified size. Subsequently, a multi-layer Recurrent Neural Network (RNN) is constructed using LSTM cells, with the number of layers determined by the num\_layers parameter. This RNN is encapsulated in a DropoutWrapper to mitigate overfitting, applying dropout to the RNN cells with a dropout rate specified by the forget\_bias parameter.

Placeholders are declared for input (self.X) and output (self.Y) data, representing a 3D tensor for input sequences and a 2D tensor for target outputs, respectively. Additionally, a placeholder for the hidden layer (self.hidden\_layer) is defined, representing the initial state of the hidden layers.

The dynamic RNN is established using tf.nn.dynamic\_rnn, taking into account the dropout-wrapped RNN cells (drop), input data (self.X), and initial hidden layer state (self.hidden\_layer). This RNN processes the input sequence and produces both the output sequence (self.outputs) and the final state of the hidden layers (self.last\_state).

A dense output layer is applied to the last output of the RNN (self.outputs[-1]) using tf.layers.dense, resulting in the model's predictions (self.logits). The cost function, calculated as the mean squared error between predictions and target outputs, is defined using tf.reduce\_mean(tf.square(self.Y - self.logits)).

The Adam optimizer is employed to minimize the cost during training, and the optimization operation is stored in self.optimizer. The model is now capable of processing sequential data, and it can be trained and evaluated for tasks such as time series prediction or sequence-to-sequence learning.

Additionally, the script includes auxiliary functions. calculate\_accuracy computes the accuracy of model predictions against actual values. anchor implements exponential smoothing on a signal using a weighted average, which can be useful for smoothing noisy data. These functions contribute to the overall utility of the script, enabling a comprehensive approach to handling and analyzing sequential data.

In the forecasting phase, the number of future predictions (future\_day) is set, and an array (output\_predict) is initialized to store the predictions. The initial hidden state (init\_value) is set to zeros, and predictions are generated for the training data in batches. If there are remaining data points, predictions are generated for them as well. Subsequently, predictions are made for the future data, and the hidden state is updated at each step.

The resulting predictions are transformed back to the original scale using inverse scaling. To enhance the forecast, exponential smoothing is applied using the anchor function. Finally, the script returns the smoothed predictions (deep\_future). In essence, the script defines, trains, and utilizes an LSTM model for time series forecasting, leveraging sequential information in the data to make accurate predictions.

## 3.2 Bidirectional LSTM

Firstly, the model is encapsulated in the Model\_LSTM\_Bidirectional class, which is initialized with various parameters like learning rate, number of layers, input size, layer size, output size, and forget bias. The class contains a method \_\_init\_\_ that sets up the architecture of the Bi-LSTM model using TensorFlow. It includes placeholders for input (self.X) and output (self.Y) data, representing the input sequence and target output, respectively.

Two sets of LSTM cells, one for the forward pass and another for the backward pass, are created using the lstm\_cell function. These cells are then encapsulated in DropoutWrapper to apply dropout regularization, and placeholders (self.forward\_hidden\_layer and self.backward\_hidden\_layer) are defined for the initial hidden states of the forward and backward cells.

The bidirectional dynamic RNN is constructed using tf.nn.bidirectional\_dynamic\_rnn, taking into account the forward and backward cells, the input sequence (self.X), and the initial hidden states. The outputs from both directions are concatenated, forming the bidirectional output sequence (self.outputs).

A dense layer is applied to the last element of the output sequence to obtain the final predictions (self.logits). The mean squared error is used as the cost function (self.cost), and the Adam optimizer is employed to minimize this cost during training (self.optimizer).

The forecast\_LSTM\_Bi function is then defined to train and predict future values using the Bi-LSTM model. It starts by resetting the default TensorFlow graph and initializing the Bi-LSTM model with specified parameters. A TensorFlow session is created, and all variables are initialized.

The training loop begins, iterating over epochs and processing the training data in batches. The forward and backward hidden states are updated at each step, and the loss and accuracy metrics are tracked.

After training, the script proceeds to forecasting. The future data points are predicted iteratively, and the hidden states are updated accordingly. The predictions are then transformed back to the original scale, and an exponential smoothing function (anchor) is applied to the forecasted values.

The script concludes with returning the forecasted values (deep\_future) for further analysis or visualization.

## 3.3 Two-path LSTM

This model is designed with two separate paths: forward and backward. The purpose of this dual-path architecture is to capture bidirectional temporal dependencies in the input time series data.

The class initialization defines the model's architecture. Two sets of LSTM cells are created within separate variable scopes, 'forward' and 'backward'. These cells constitute the forward and backward paths, each consisting of multiple layers specified by the num\_layers parameter. Dropout is applied to these cells to prevent overfitting. Placeholders (self.X\_forward and self.X\_backward) are declared for input sequences along each path, and placeholders (self.hidden\_layer\_forward and self.hidden\_layer\_backward) are set for the initial hidden states.

The forward and backward paths are executed independently using the dynamic\_rnn function, taking into account the corresponding input sequences and initial hidden states. Importantly, the output sequences from the forward path are subtracted from those of the backward path (self.outputs\_backward - self.outputs\_forward). This subtraction is designed to capture the bidirectional nature of temporal dependencies in the data.

The rest of the class initialization defines the output layer, cost function, and optimizer for training the model. The final output (self.logits) is obtained by applying a dense layer to the subtracted output sequences.

The training loop in the forecast\_LSTM\_2path function follows a similar structure to the previously explained LSTM model. The model is initialized, and a new TensorFlow session is started. The training loop iterates through epochs, and within each epoch, it processes the training data in batches. For each batch, the model is run on both forward and backward paths, updating the hidden states and calculating loss and accuracy.

The forecasting phase is then executed. The script initializes the prediction array and iterates through the training data to make predictions. If there are remaining data points, predictions are generated for them as well. Finally, predictions are made for future data points, and the hidden states are updated at each step.

The resulting predictions are transformed back to the original scale using inverse scaling. Additionally, the anchor function is applied for exponential smoothing. The script concludes by returning the smoothed predictions (deep\_future).

## 3.4 Results of the LSTM models

All models are tested using the last 30 days of the dataset. The accuracy of each simulation is calculated against the true trend of the data of the last 30 days:

Model LSTM, Simulation 1, Accuracy: 94.05%

Model LSTM\_Bi, Simulation 1, Accuracy: 92.51%

Model LSTM\_2path, Simulation 1, Accuracy: 93.64%

Model LSTM, Simulation 2, Accuracy: 91.82%

Model LSTM\_Bi, Simulation 2, Accuracy: 94.16%

Model LSTM\_2path, Simulation 2, Accuracy: 94.13%

Model LSTM, Simulation 3, Accuracy: 95.36%

Model LSTM\_Bi, Simulation 3, Accuracy: 94.38%

Model LSTM\_2path, Simulation 3, Accuracy: 92.39%

Model LSTM, Simulation 4, Accuracy: 95.42%

Model LSTM\_Bi, Simulation 4, Accuracy: 94.10%

Model LSTM\_2path, Simulation 4, Accuracy: 94.17%

Model LSTM, Simulation 5, Accuracy: 93.22%

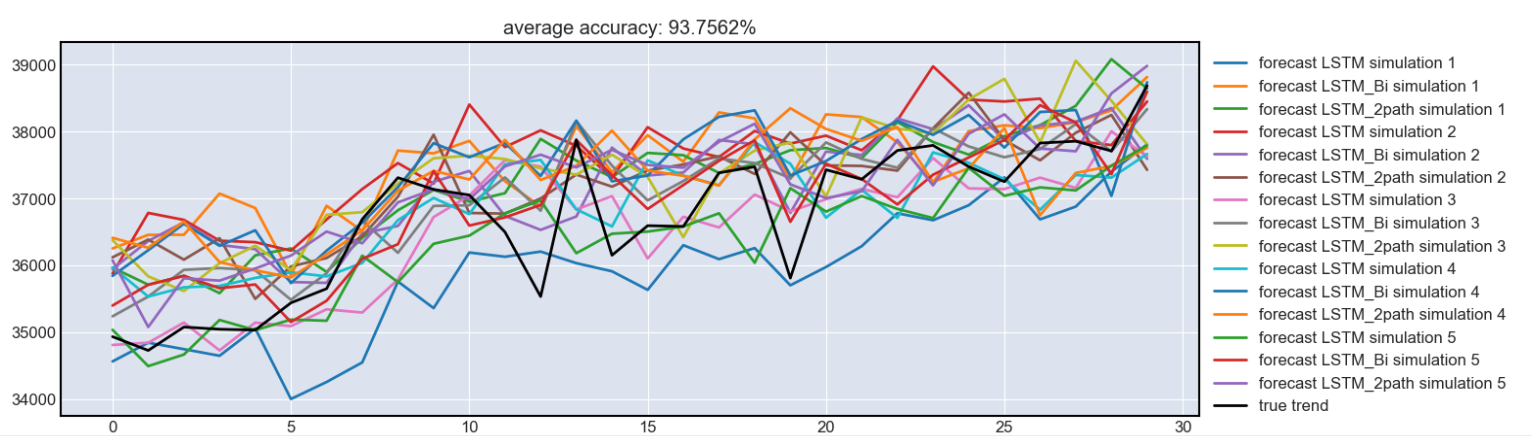
Model LSTM\_Bi, Simulation 5, Accuracy: 94.59%

Model LSTM\_2path, Simulation 5, Accuracy: 92.40%

Model LSTM average accuracy: 93.97%

Model LSTM\_Bi average accuracy: 93.95%

Model LSTM\_2path average accuracy: 93.35%



*Figure 3.1: Test result*

Based on the chart we can see:

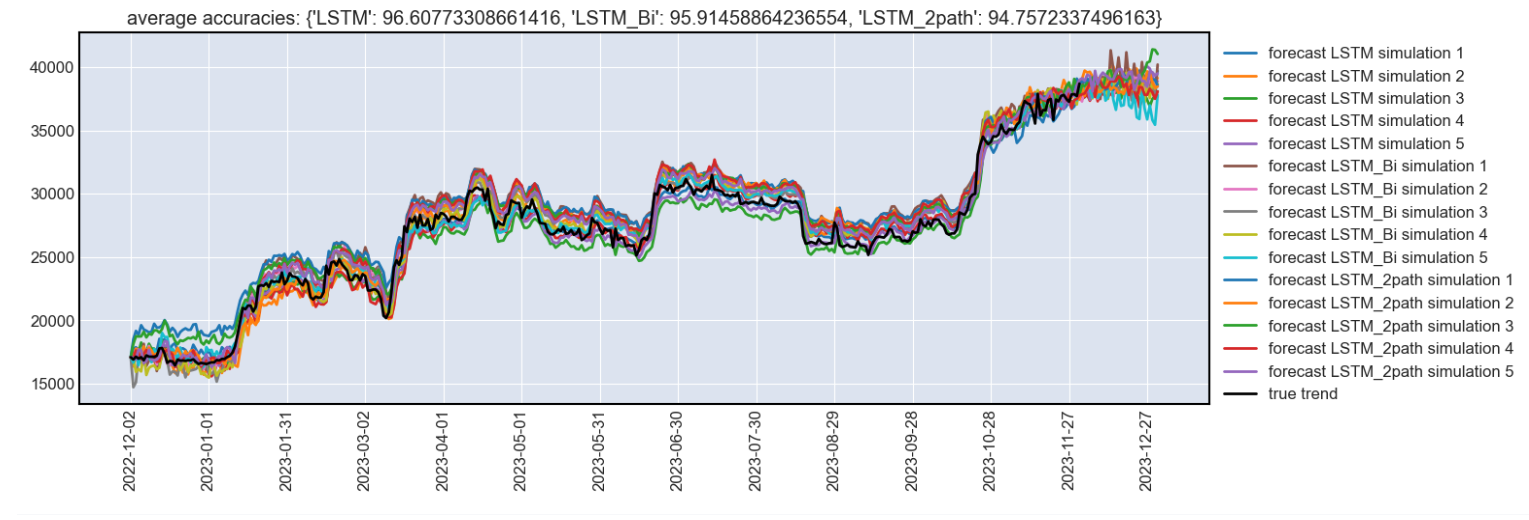
- Model LSTM average accuracy is 93.97% with the highest being 95.42% in simulation 4 and the lowest being 91.82% in simulation 2

- Model LSTM Bidirectional has an average accuracy of 93.95%. The highest is 94.59% in simulation 5 and the lowest is 92.51% in simulation 1

- Model LSTM 2\_path has an average accuracy of 93.35% with the highest accuracy belonging to simulation 4 at 94.17%, the lowest accuracy is simulation 3 at 92.39%

Concluding:

* The LSTM model has the highest average accuracy, however, there is quite large variation between series of simulation data.
* Model LSTM\_Bi has the second highest average accuracy and peak accuracy.
* Model LSTM\_2path has the lowest average accuracy among the three models, and the variation between series of simulation data is also quite large.
* All models are not good at capturing the trend of the true data, although they have a quite good accuracy compared to the linear models.

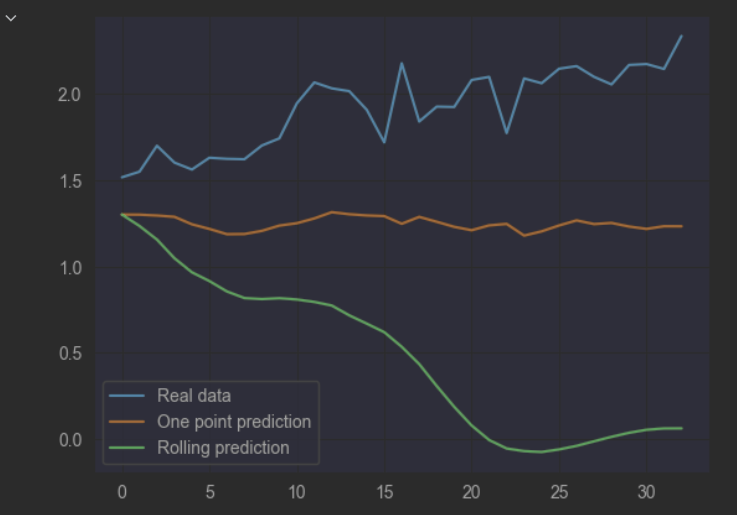


*Figure 3.2: Forecast for the next 30 days*

* After applying three models and five simulations over a period of 365 days, we still see that the LSTM model produces the highest accuracy of about 96.06%, following by LSTM\_Bi with 95.91% and LSTM\_2path with 94.76%
* Overall, it can be seen that all 3 models fit relatively well with the true trend. However, there is a slight shift in the trend of the true data, which would be further analyzed in the latter part.

## 3.5 Hyperparameter tuning

We have tried to tune the hyperparameters using the keras-tuner library. However, the results are significantly worse than the original parameters, therefore we chose to keep the original hyperparameters of the LSTM models.



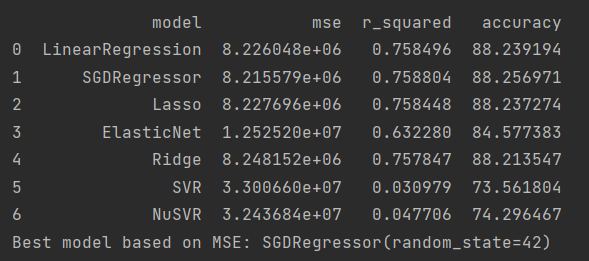
*Figure 3.3: Hyperparameter tuning*

However, we also tried to predict the movement of the price for the next day in percentage change. After running the models, we found out that the LSTM models have great difficulties in predicting the movement of the price for the next day.



It can be seen from the graph that there are only a few days the predictions can correctly predict the movement of the price. They also fail to predict the percentage change in price properly. However, in the previous models run with the actual price prediction instead of the price change, the reason why the price predicted by the models seems so accurate is due to the fact that we have feed the model the true price of that day, and the models only change that value by a small percentage for the prediction of the next day, which explains why the prediction made by the models always “lag” behind the true trend.

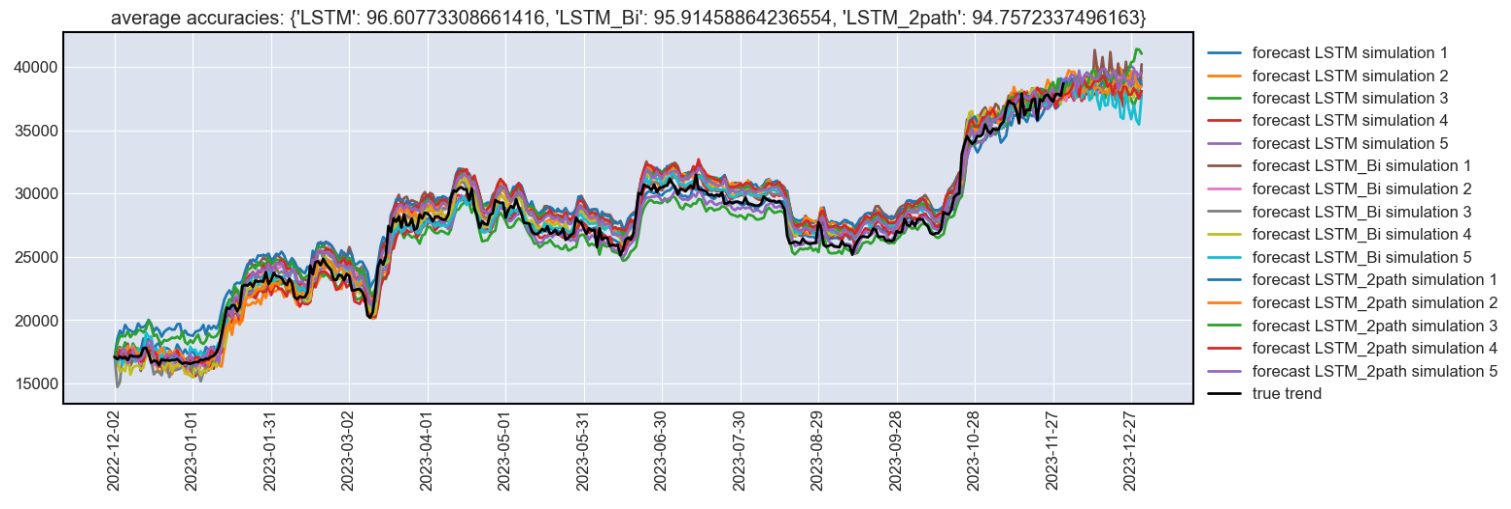
# **QUESTION 4**



*Figure 4.1: Linear regression models accuracy*

With the Linear Regression model, the average accuracy of the 7 models is 83.62%. Among them, SGDRegressor has the best accuracy, 88.25%. Five in seven models have performed better than the average, including Linear Regression, SGDRegressor, Lasso, ElasticNet and Ridge. Meanwhile, the remaining two models, SVR and NuSVR have the accuracy that are much lower than the others, at 73.56% and 74.296%, respectively.

Although the Linear Regression model have relatively good accuracy, it still seems to be lower than LSTM.



*Figure 4.2: Forcast for the next 30 days*

LSTM model have the overall average accuracy of 95.496%. To get this results, we have carried 5 simulations for each type of LSTM model, which are LSTM, LSTM Bidirectional and LSTM 2path. It can be seen that the average accuracy of LSTM outweighs the Linear Regression, higher than 11% accuracy. This is exceptionally significant as Bitcoin prices is really volatile and hard to predict precisely.

Among the LSTM models, the LSTM Unidirectional has the highest accuracy of 96.6%. Although the two remaining LSTM have the lower accuracy, the difference is small and insignificant. To be more precise, LSTM Bidirectional and LSTM 2path have the accuracy of 95.91% and 94.75% respectively, only approximately 1% difference. Therefore, it can be said that the Bitcoin prices are hard to predict accurately on the ladder of 100% accuracy.

In comparison, the accuracy difference between LSTM models is not as varied as Linear Regression. Despite the fact that LSTM is predicted with 5 simulations, the difference is not notable, therefore, it can be implied that LSTM models have a stable performance in predicting Bitcoin prices.

# 

# 

# **CONCLUSION**

Bitcoin price has significant volatility, making it difficult to predict accurately. In our project we use Linear Regression including SGDRegressor, Lasso, ElasticNet, Ridge, SVR and NuSVR, also LSTM models including basic LSTM Unidirectional, LSTM Bidirectional and LSTM 2path, to make the Bitcoin price forecast.

With the Linear Regressions and LSTMs, we can predict them with relatively good accuracy. Between these two models, the simpler one is, the more accuracy it can predict. In fact, Linear Regression is simpler than LSTM, easy to build and use. However, its accuracy is much lower than LSTM. By contrast, LSTM is a more stable and accurate tool to predict Bitcoin with an accuracy rate higher than 94%. The best models of Linear Regression and LSTM which have the highest accuracy are SGD Regressor and LSTM Unidirectional, and can provide an accuracy of 88.25% and 96.6% accuracy respectively.

It is also noticeable that with 5 simulations for the LSTM model, there are not much differences between the LSTM models, as the accuracy rate only ranges from 94.75% to 96.6%. While the accuracy range of Linear Regression is much wider, from 74.29% to 88.25%.

The discovery during the hyperparameter turing reveals that LSTM is actually not very useful in crypto trading, as the prediction always lags behind the true data. In real life, Bitcoin prices depend on many other factors such as interest rates (when interest rates are low, people will invest more in bitcoin as a way to make profit), or government policy (allow crypto trading or mutual funds). Using historical price to predict bitcoin prices in the future is an unrealistic way to invest and this method does not guarantee a good return on investment.

# **REFERENCE**

<https://www.mdpi.com/2504-3110/7/2/203>

<https://www.kaggle.com/code/meetnagadia/bitcoin-price-prediction-using-lstm#5.-Building-LSTM-Model>

1. Melitz, J. DP178 Monetary Discipline, Germany, and the European Monetary System; National Bureau of Economic Research (NBER) Working Paper No. 2319; National Bureau of Economic Research (NBER): Cambridge, MA, USA, 1987; Available online: https://ssrn.com/abstract=884539 (accessed on 24 September 2022).
2. Bulíř, A. Income inequality: Does inflation matter? IMF Staff. Pap. 2001, 48, 139–159. [Google Scholar]
3. Basco, S. Globalization and financial development: A model of the Dot-Com and the Housing Bubbles. J. Int. Econ. 2014, 92, 78–94. [Google Scholar] [CrossRef]
4. Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. Decentralized Bus. Rev. 2008, 21260. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 19 October 2022).
5. Sureshbhai, P.N.; Bhattacharya, P.; Tanwar, S. KaRuNa: A blockchain-based sentiment analysis framework for fraud cryptocurrency schemes. In Proceedings of the 2020 IEEE International Conference on Communications Workshops (ICC Workshops), Dublin, Ireland, 7–11 June 2020; pp. 1–6. [Google Scholar]
6. Rose, C. The evolution of digital currencies: Bitcoin, a cryptocurrency causing a monetary revolution. Int. Bus. Econ. Res. J. (IBER) 2015, 14, 617–622. [Google Scholar] [CrossRef]
7. Badea, L.; Mungiu-Pupăzan, M.C. The economic and environmental impact of bitcoin. IEEE Access 2021, 9, 48091–48104. [Google Scholar] [CrossRef]
8. Vranken, H. Sustainability of bitcoin and blockchains. Curr. Opin. Environ. Sustain. 2017, 28, 1–9. [Google Scholar] [CrossRef][Green Version]
9. Iwamura, M.; Kitamura, Y.; Matsumoto, T. Is bitcoin the only cryptocurrency in the town? economics of cryptocurrency and friedrich a. hayek. SSRN Electron. J. 2014. [Google Scholar] [CrossRef][Green Version]
10. Hassani, H.; Huang, X.; Silva, E. Big-crypto: Big data, blockchain and cryptocurrency. Big Data Cogn. Comput. 2018, 2, 34. [Google Scholar] [CrossRef][Green Version]
11. Hwang, K.; Chen, M. Big-Data Analytics for Cloud, IoT and Cognitive Computing; John Wiley & Sons: Hoboken, NJ, USA, 2017. [Google Scholar]
12. Hitam, N.A.; Ismail, A.R.; Samsudin, R.; Alkhammash, E.H. The Effect of Kernel Functions on Cryptocurrency Prediction Using Support Vector Machines. In Proceedings of the International Conference of Reliable Information and Communication Technology; Springer: Cham, Switzerland, 2022; pp. 319–332. [Google Scholar]
13. Andrianto, Y.; Diputra, Y. The effect of cryptocurrency on investment portfolio effectiveness. J. Financ. Account. 2017, 5, 229–238. [Google Scholar] [CrossRef][Green Version]
14. Saad, M.; Choi, J.; Nyang, D.; Kim, J.; Mohaisen, A. Toward characterizing blockchain-based cryptocurrencies for highly accurate predictions. IEEE Syst. J. 2019, 14, 321–332. [Google Scholar] [CrossRef]
15. Chowdhury, R.; Rahman, M.A.; Rahman, M.S.; Mahdy, M. An approach to predict and forecast the price of constituents and index of cryptocurrency using machine learning. Phys. A Stat. Mech. Its Appl. 2020, 551, 124569. [Google Scholar] [CrossRef]
16. Derbentsev, V.; Babenko, V.; Khrustalev, K.; Obruch, H.; Khrustalova, S. Comparative performance of machine learning ensemble algorithms for forecasting cryptocurrency prices. Int. J. Eng. 2021, 34, 140–148. [Google Scholar]
17. Chen, W.; Xu, H.; Jia, L.; Gao, Y. Machine learning model for Bitcoin exchange rate prediction using economic and technology determinants. Int. J. Forecast. 2021, 37, 28–43. [Google Scholar] [CrossRef]
18. Patel, M.M.; Tanwar, S.; Gupta, R.; Kumar, N. A deep learning-based cryptocurrency price prediction scheme for financial institutions. J. Inf. Secur. Appl. 2020, 55, 102583. [Google Scholar] [CrossRef]
19. Wu, C.H.; Lu, C.C.; Ma, Y.F.; Lu, R.S. A new forecasting framework for bitcoin price with LSTM. In Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW), Singapore, 17–20 November 2018; pp. 168–175. [Google Scholar]
20. Livieris, I.E.; Pintelas, E.; Stavroyiannis, S.; Pintelas, P. Ensemble deep learning models for forecasting cryptocurrency time-series. Algorithms 2020, 13, 121. [Google Scholar] [CrossRef]
21. Derbentsev, V.; Datsenko, N.; Babenko, V.; Pushko, O.; Pursky, O. Forecasting Cryptocurrency Prices Using Ensembles-Based Machine Learning Approach. In Proceedings of the 2020 IEEE International Conference on Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, Ukraine, 6–9 October 2020; pp. 707–712. [Google Scholar]
22. Zhang, Z.; Dai, H.N.; Zhou, J.; Mondal, S.K.; García, M.M.; Wang, H. Forecasting cryptocurrency price using convolutional neural networks with weighted and attentive memory channels. Expert Syst. Appl. 2021, 183, 115378. [Google Scholar] [CrossRef]
23. Ahsan, M.M.; Mahmud, M.P.; Saha, P.K.; Gupta, K.D.; Siddique, Z. Effect of data scaling methods on machine learning algorithms and model performance. Technologies 2021, 9, 52. [Google Scholar] [CrossRef]
24. Hochreiter, S.; Schmidhuber, J. Long short-term memory. Neural Comput. 1997, 9, 1735–1780. [Google Scholar] [CrossRef]
25. Ayoobi, N.; Sharifrazi, D.; Alizadehsani, R.; Shoeibi, A.; Gorriz, J.M.; Moosaei, H.; Khosravi, A.; Nahavandi, S.; Chofreh, A.G.; Goni, F.A.; et al. Time series forecasting of new cases and new deaths rate for COVID-19 using deep learning methods. Results Phys. 2021, 27, 104495. [Google Scholar] [CrossRef]
26. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv 2014, arXiv:1412.3555. [Google Scholar]
27. Yang, S.; Yu, X.; Zhou, Y. Lstm and gru neural network performance comparison study: Taking yelp review dataset as an example. In Proceedings of the 2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI), Shanghai, China, 12–14 June 2020; pp. 98–101. [Google Scholar]
28. Wang, X.; Jiang, W.; Luo, Z. Combination of convolutional and recurrent neural network for sentiment analysis of short texts. In Proceedings of the Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, Japan, 11–16 December 2016; pp. 2428–2437. [Google Scholar]
29. Dey, R.; Salem, F.M. Gate-variants of gated recurrent unit (GRU) neural networks. In Proceedings of the 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), Boston, MA, USA, 6–9 August 2017; pp. 1597–1600. [Google Scholar]
30. Schuster, M.; Paliwal, K.K. Bidirectional recurrent neural networks. IEEE Trans. Signal Process. 1997, 45, 2673–2681. [Google Scholar] [CrossRef][Green Version]
31. Lai, S.; Ye, C.; Zhou, H.J.H. Chinese stock trend prediction based on multi-feature learning and model fusion. In Proceedings of the 2021 IEEE International Conference on Smart Data Services (SMDS), Chicago, IL, USA, 5–10 September 2021; pp. 18–23. [Google Scholar]
32. Singh, A.; Kumar, A.; Akhtar, Z. Bitcoin Price Prediction: A Deep Learning Approach. In Proceedings of the 2021 8th International Conference on Signal Processing and Integrated Networks (SPIN), Noida, India, 26–27 August 2021; pp. 1053–1058. [Google Scholar]
33. Althelaya, K.A.; El-Alfy, E.S.M.; Mohammed, S. Stock market forecast using multivariate analysis with bidirectional and stacked (LSTM, GRU). In Proceedings of the 2018 21st Saudi Computer Society National Computer Conference (NCC), Riyadh, Saudi Arabia, 25–26 April 2018; pp. 1–7. [Google Scholar]
34. Na, Z.; Wang, Y.; Li, X.; Xia, J.; Liu, X.; Xiong, M.; Lu, W. Subcarrier allocation based simultaneous wireless information and power transfer algorithm in 5G cooperative OFDM communication systems. Phys. Commun. 2018, 29, 164–170. [Google Scholar] [CrossRef]
35. Hansun, S.; Wicaksana, A.; Khaliq, A.Q. Multivariate cryptocurrency prediction: Comparative analysis of three recurrent neural networks approaches. J. Big Data 2022, 9, 1–15. [Google Scholar] [CrossRef]
36. Ozturk Birim, S. An Analysis for Cryptocurrency Price Prediction Using Lstm, Gru, and the Bi- Directional Implications. In Developments in Financial and Economic Fields at the National and Global Scale; Cömert, M., Şimşek, A.E., Eds.; Gazi Kitabevi: Ankara, Türkiye, 2022; pp. 377–392. [Google Scholar]