**UNIVERSITY OF ECONOMICS AND LAW**

**FACULTY OF INFORMATION SYSTEM**

****

**Logo

Description automatically generated**

**FINAL PROJECT REPORT**

**SUBJECT: TIME SERIES AND FORECAST**

**LECTURER: Mr. LE HOANH SU**

**GROUP: NOMO**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Student’s Name** | **Student’s ID** | **Contribution** |
| **1** | **Nguyễn Ngọc Khôi Nguyên** | **K204161994** | **100%** |
| **2** | **Trương Công Vinh** | **K204162008** | **100%** |
| **3** | **Lê Minh Trí** | **K204162004** | **100%** |

Contents

[**1.** **Theoretical basis** 3](#_Toc135996940)

[**1.1** **Theory of XGBoost Algorithm** 3](#_Toc135996941)

[**1.2** **Theory of CNN Algorithm** 3](#_Toc135996942)

[**1.3** **Theory of LSTM Algorithm** 4](#_Toc135996943)

[**1.4** **Combination of CNN and LSTM** 5](#_Toc135996944)

[**2.** **Related research about Feature selections** 6](#_Toc135996945)

[**2.1** **Accroding to “Stock Price Prediction Method Based on XGboost Algorithm”** 6](#_Toc135996946)

[**2.2** **According to “Stock Price Prediction Methods based on FCM and DNN Algorithms”** 8](#_Toc135996947)

[**3.** **Time Series forcating with Machine Learning (XGBoost algorithm)** 9](#_Toc135996948)

[**3.1** **Features selection** 9](#_Toc135996949)

[**3.2** **Features calculation** 11](#_Toc135996950)

[**3.2.1** **MA** 11](#_Toc135996951)

[**3.2.2** **RSI** 11](#_Toc135996952)

[**3.2.3** **MACD** 12](#_Toc135996953)

[**3.2.4** **Features Overview** 13](#_Toc135996954)

[**3.3** **Features processing** 14](#_Toc135996955)

[**3.4** **Walk-forward validation** 16](#_Toc135996956)

[**3.5** **Testing model** 19](#_Toc135996957)

[**3.6** **Comparison** 30](#_Toc135996958)

[**4.** **Time Series forcating with Deep Learning** 33](#_Toc135996959)

[**4.1** **Preprocessing** 33](#_Toc135996960)

[**4.2** **Traning model with CNN** 33](#_Toc135996961)

[**4.3** **Traning model with LSTM** 34](#_Toc135996962)

[**4.4** **Traning model with CNN-LSTM** 35](#_Toc135996963)

[**5.** **Conclusion** 40](#_Toc135996964)

[Reference: 41](#_Toc135996965)

1. **Theoretical basis**
   1. **Theory of XGBoost Algorithm**

XGBoost (Extreme Gradient Boosting) is a machine learning algorithm based on decision trees that is capable of producing accurate and powerful predictions in many applications, including stock price prediction. It is an improved version of the Gradient Boosting algorithm.

In the stock price prediction problem, XGB can be used to create a prediction model based on features such as EMA, SMA, RSI, MACD, and MACD\_signal. The XGB algorithm will learn from the given data and find ways to map these features to the predicted values of stock prices in the future. We will use XGB via the XGBoost library.

* 1. **Theory of CNN Algorithm**

In order to handle picture and video data, the CNN (Convolutional Neural Network) method is utilized. Additionally, the convolutional and pooling layers of this technique aid in the learning of the characteristics of the input data. In this study, the time series data of stock prices will be processed using the CNN algorithm.

A picture containing text, diagram, font, line

Description automatically generated

Figure1 : The structure of Convolutional Neural Network

According to the study by Suresh and colleagues (2020) on "Forecasting Solar PV Output Using Convolutional Neural Networks with a Sliding Window Algorithm",a typical CNN model, as shown in Figure X, includes several layers: convolutional layer, pooling layer, flattening layer, and fully connected layer.

The primary element of the CNN network is the convolutional layer, which uses weight sharing and the sliding window concept to simplify processing. The kernel approach is applied at this layer to separate out various characteristics from the input data.

The pooling layer comes next. By decreasing the connections between layers and operating each object map separately, this layer is intended to lower the size of the pertinent object map. Reducing size and extracting better features for effective model training is the fundamental objective of the pooling procedure.

Before moving on to the last layer, the fully connected layer, which comprises weights and biases along with neurons to link neurons between various layers, comes the flattening layer, which is utilized to construct the one-dimensional vector required.

* 1. **Theory of LSTM Algorithm**

The memory units that make up the LSTM network are called cells. Each cell can be in either its cell state or its concealed state. The LSTM network's cells are utilized to make critical judgments by storing or ignoring data concerning crucial elements. These parts, which are referred to as gates, are arranged into forget gates, input gates, and output gates.

A picture containing text, diagram, screenshot, plan

Description automatically generated

Figure 2: The architecture of the LSTM network.

According to the structures displayed in Figure X, the LSTM model operates in three main stages:

In the first stage, The network uses the forget gate to decide what kind of information should be saved in the cell state or disregarded. The sigmoid function (S) is used to examine the input at the current time step and the prior value of the hidden state to start this process.

In the second stage, The old cell state is changed into a new cell state as the network processing progresses. This procedure chooses which fresh data needs to be stored in cells or long-term memory. The processing will use reference values from the forget gate, input gate, and cell update gate to determine the value for the new cell state.

After completing the update of the cell state, the final step establishes the concealed state's value. With the help of this procedure, the hidden state will serve as the network's memory, storing details about prior input and serving as a tool for prediction. We must zztake the value of the new cell state and the output gate for processing in order to ascertain the value of the hidden state.

* 1. **Combination of CNN and LSTM**

Convolutional Neural Network-Long Short-Term Memory, or CNN-LSTM (Convolutional Neural Network-Long Short-Term Memory), is a strong combination model that may be used to combine the high learning potential of LSTM algorithm in a time series with the efficacy of CNN to extract priceless knowledge. In particular, temporal aspects of the data will be extracted using CNN, and the data will be predicted using LSTM.

A picture containing diagram, text, line, font

Description automatically generated

Figure 3: The structure of the CNN-LSTM model.

CNN layers are used to build the CNN-LSTM model's architecture. The goal is to take the input data and extract characteristics from it. In order to facilitate sequence prediction, the output of the CNN layers is then sent to the LSTM layers and a dense layer at the output. To understand the sequence of each step in the model, we will briefly explain as follows:

First, the input layer will receive the corresponding input data.

The first convolutional layer, before displaying the result on the feature map, will review the input data entered in the preceding phase.

The second convolutional layer will search the feature map made from the preceding layer for more significant features, improving the model's accuracy. In order to interpret the input sequence, this layer employs 32 feature mappings per convolution layer and a kernel size of three steps. To recognize characteristics at a more abstract and complicated level, this layer will scan the input data more thoroughly than the previous convolutional layer.

The max pooling layer will reduce the feature maps and produce a smaller matrix in the previous stage by removing particular and difficult features.

The dropout layer improves the learning network to prevent overfitting in the model.

The flatten layer flattens the filtered feature maps into a solitary, lengthy vector that may be utilized as the LSTM model's decoding input.

The LSTM Vector layer represents the inside of the input sequence being repeated once for each time step in the output sequence; it is used to build an LSTM layer state vector whose dimensions are equal to the number of time steps in the input sequence, then repeat this vector to match the length of the input sequence.

The LSTM decoder features a hidden layer with 100 units that can output the full arrangement, and each layer has 100 units that supply daily values. These values are used to anticipate what will happen in the output order during the next 100 days.

The fully connected layer shows that the LSTM decoder may work at any moment similarly to both the output layer and the fully connected layer by understanding each step in the output sequence to eventually have comparable layers to anticipate a single output sequence.

The output layer provides the output value for the outcome of the forecast.

1. **Related research about Feature selections**
   1. [**Accroding to “Stock Price Prediction Method Based on XGboost Algorithm”**](#_heading=h.30j0zll)

Feature selection is the process of selecting a subset of relevant features or variables from a larger set of features that are used in building a model. The goal of feature selection is to reduce the dimensionality of the feature space, eliminate irrelevant features, and retain only the most important features that are relevant for the prediction task

1. Accroding to “Stock Price Prediction Method Based on XGboost Algorithm” [1] by Yifan Zhang. The authors used XGboost algorithm for stock price prediction, and they performed parameter selection and feature selection to optimize the model's performance.

a. Algorithm Parameters: The authors selected the following parameters for the XGboost algorithm: n\_estimators, max\_depth, min\_child\_weight, gamma, and learning\_rate. These parameters control the number of weak learners, depth of the tree, weight threshold of the smallest node, minimum threshold for node splitting, and weight reduction coefficient of each weak learner, respectively.

b. Predictive Index Selection: The authors selected daily opening price, closing price, high price, low price, and the total number of daily stock indicators of the stock as the data parameters for the XGboost algorithm. They also included the relative strength indicator (RSI) as a technical curve based on the ratio of the sum of up points and down points in a certain period of time. RSI is effective for stock price prediction based on investment strategies.

|  |  |
| --- | --- |
| n\_estimators | [100, 200, 300, 400] |
| learning\_rate | [0.001, 0.005, 0.01, 0.05] |
| max\_depth | [8, 10, 12, 15] |
| Gamma | [0.001, 0.005, 0.01, 0.02] |
| random\_state | [42, 43, 44, 45] |

Table 1. Parameter seek optimization settings

|  |  |
| --- | --- |
| EMA\_9 | Stock price index weighted moving average 9-day data |
| SMA\_5 | Stock short-term average moving line 5-day data |
| SMA\_15 | Short-term average stock price moving line 15-day data |
| SMA-30 | Short-term average stock price moving line 30-day data |
| RSI | SMA(MAX(Close-LastClose,0),N,1)/SMA(ABS(Close-LastClose),N,1)\*100 |
| MACD | EMA12-EMA26 |
| EMA12-EMA26 | MACD SIGNAL Moving average of ewm index weights for MACD with a span of 9 days |

The authors explained that the selection of these parameters and features was based on the characteristics of stock price data and the short-term nature of stock price prediction. The use of time series data and technical indicators such as RSI can help improve the accuracy of the model. It's worth noting that parameter and feature selection can be an iterative process, and it's important to evaluate the performance of the model on a validation set to avoid overfitting.

* 1. [**According to “Stock Price Prediction Methods based on FCM and DNN Algorithms”**](#_heading=h.1fob9te)

The authors use of Fuzzy Clustering Algorithm for Stock Value Feature Selection. The goal is to select the most representative features that reflect the stock value of listed companies. The five types of financial indicators that mainly reflect the stock value are profitability, development ability, shareholder profitability, solvency, and operating ability. Fuzzy clustering technology is used to cluster these five types of indicators, and the correlation index method is used to screen the indicators, compressing the stock value investment and finally selecting the stock value. The investment closely related to indicators constitutes the stock value feature set.

Authors also mentions attribute reduction, which is the process of analyzing the effectiveness of various features and selecting the most representative features. There are two methods for attribute reduction, feature selection, and feature extraction.

Steps of Fuzzy Clustering:

According to the definition of cluster analysis

- One is clustering based on sample similarity, called sample clustering

- The other is clustering based on indicator similarity, called index clustering.

Cluster analysis method can be subdivided into many methods according to the different mathematical tools used. The cluster analysis method using fuzzy mathematical tools is called fuzzy clustering.

Fuzzy clustering method can be used to cluster the multiple features of the sample to first realize their classification and then select the most representative feature composition among similar features. rough the selection of similar indicators, the purpose of feature screening can be achieved.

(1) Data standardization: the specific algorithm for data standardization includes two steps:

1. Translation and standard deviation transformation

2. Translation and range transformation

(2) Fuzzy similarity matrix establishment: suppose X and Y are two nonempty sets;

(3) Clustering: in this article, the transitive closure method is used for clustering. First, the fuzzy equivalence matrix is obtained by seeking the transfer closure using the square method. Second, the fuzzy equivalence matrix of the five aspects of profitability, development capability, shareholder profitability, solvency, and operating capability is calculated.

(4) Feature screening: the aim is to select the most representative indicators from each category of similar indicators, based on their correlation with other indicators in the same category. The correlation index method is used to select the best indicator.

The process involves calculating the correlation coefficient among the indicators in each category, and then computing the mean value of the square of the correlation coefficient between each indicator and other indicators. The indicator with the largest correlation index is then selected as the typical indicator for that category.

If there is only one indicator in a category, it is directly included in the indicator set. If there are two indicators in a category, one of them is chosen based on its higher correlation with the other indicators in the category. This ensures that the selected features cover the most comprehensive information for the classification task.

1. **Time Series forcating with Machine Learning (XGBoost algorithm)**
   1. **Features selection**

The target variable is the value that needs to be predicted, while features are the information that may affect the value of the target variable.

In this XGB problem, the target variable is the closing price (closeADC) of ADC stock. The dataset includes 1251 rows from 2017 to 2021, allowing us to analyze trends and predict the stock price of ADC in the future.

A black screen with white text

Description automatically generated with low confidence

Figure 4: Target variable

Specifically, technical indicators such as EMA\_9, SMA\_5, SMA\_10, SMA\_15, SMA\_30, RSI, MACD, and MACD\_signal will be used as features in the stock price prediction problem with XGB because they contain information about the trend and volatility of the stock price in the past:

* EMA\_9: This is the exponential moving average of the stock price over the last 9 days, calculated using the EMA formula. It helps the model predict the future trend of the stock price based on recent moving average values.
* SMA\_5, SMA\_10, SMA\_15, SMA\_30: These are the simple moving averages of the stock price over the last 5, 10, 15, and 30 days, respectively. They help the model predict the future trend of the stock price.
* RSI: This is an indicator that measures the level of overbought or oversold of the stock. It can be used to determine whether the stock price will likely increase or decrease in the future.
* MACD: This is the difference between the 12-day and 26-day exponential moving averages of the stock price. It can help the model predict the future changes in the stock price.

MACD\_signal: This is the 9-day exponential moving average of the MACD value. It can help the model determine the future trend of the stock price based on recent MACD values.

* 1. **Features calculation**

Firstly, the following code snippet will be used to add smoothing values to the features to prevent negative values and adjust the smoothness of the moving averages. The smaller the value of the smoothing\_value, the more sensitive the moving averages will be to new price values.



Figure 5: Smoothing value

* + 1. **MA**

Calculate EMA values using the pandas' ewm function and SMA using the pandas' rolling function as previously instructed in previous weeks.

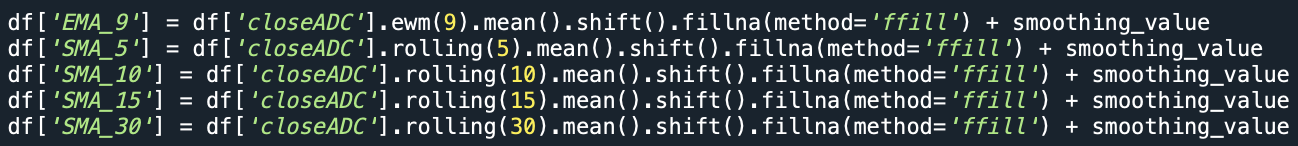


Figure 6: MA

Chart, line chart, histogram

Description automatically generated

Figure 7: Ma visualization

* + 1. **RSI**

Similarly, we will calculate the RSI index

Text

Description automatically generated

Figure 8: RSI

Chart

Description automatically generated

Figure 9: RSI visualization

* + 1. **MACD**

Similarly, we will calculate the MACD và MACD\_signal index

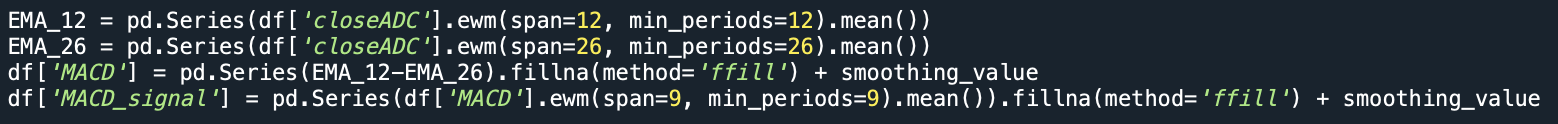


Figure 10: MACD

Chart, line chart, histogram

Description automatically generated

Figure 11: MACD visualization

* + 1. **Features Overview**

A screenshot of a computer

Description automatically generated with low confidence

Figure 12: Features overview

When computing features, there may be some NaN (Not a Number) values that appear. This happens because the dataset does not have enough data to compute them.

NaN values cannot be used to train an XGB model because they are illegal values and can cause errors during training. If we do not remove NaN values, the XGB model will be unable to learn information from these features, and the model's results may not be accurate.

* 1. **Features processing**

Firstly, the code below will create a new column "closeADC" shifted up by one row from the original "closeADC" column. By doing this, each row will contain the value of the "closeADC" column from the next row, meaning that the value of the next day will be used to predict the value of the current day.

This is very useful in the problem of price prediction using XGB because it allows the model to predict the value of the current day based on the value of the next day. This helps the model learn the correlation relationships between the values of consecutive days and improves the model's prediction ability.



Figure 13: Shifted dataset

We have conducted some research and found 2 ways to remove NaN values:

* Method 1: Use dropna() in Pandas to remove rows containing NaN values in the DataFrame. However, this method is prone to losing data, so we will move on to method 2 below, which is a way to fill in missing values instead of dropping them.
* Method 2: Use fillna() to replace NaN values with the mean value of the column.



Figure 14: Remove NaN values

The dataset has filled in the missing values

Table

Description automatically generated with medium confidence

Figure 15: The dataset

Next, because adding missing values can easily lead to outliers, we will remove outliers using the IQR method to determine and remove outliers.

Text

Description automatically generated with medium confidence

Figure 16: IQR method

The complete dataset for training has 967 rows, down from 1251, after processing and removing outliers.

A picture containing table

Description automatically generated

Figure 17: Completed dataset

* 1. **Walk-forward validation**

a. Splitting the dataset

The values of the test and validation sets account for a total of 30% of the dataset, while the rest is used for training.

This data split method ensures that the model is trained on a sufficiently large dataset and evaluated on independent datasets to ensure the feasibility and accuracy of the prediction model.

Text

Description automatically generated

Figure 18: Walk-forward validation

Chart, histogram

Description automatically generated

Figure 19: Walk-forward visualization

1. Remove unnecessary data (year) from test, valid and train

Text

Description automatically generated

Figure 20: Remove unnecessary data

1. Splitting target variables and features

In the problem of predicting closeADC price, according to previous XGB research, the data needs to be split into two components: the target variable and the features to train the prediction model.

Text

Description automatically generated

Figure 21: Splitting target variables and features

y\_train, y\_valid, and y\_test, are sets containing the target variable closeADC

Table

Description automatically generated with medium confidence

Figure 22: Target variable

X\_train, X\_valid, and X\_test are sets containing features EMA, SMA, RSI, MACD

A picture containing table

Description automatically generated

Figure 23: Features

Splitting the data in this way makes it easier to train the prediction model because the target variable is the value we want to predict and the features are the information we want to use to predict that value.

Text

Description automatically generated with medium confidence

Figure 24: Splitting completion

* 1. **Testing model**

XGB uses a method that focuses on the weaknesses of previous models and strengthens them by adding new decision trees and adjusting parameters to reduce overfitting and improve model accuracy.

Applying too many rules such as fillna and IQR can lead to overfitting. XGB provides parameters to adjust the model. The parameters in XGB are divided into three main groups:

* General parameters (1): Adjust the basic parameters of the algorithm
* Booster parameters (2): Adjust parameters related to decision trees
* Task parameters (3): Automatically adjust parameters to optimize the performance of the model.

Combining these three groups of parameters results in many different parameters. However, some parameters are considered important and need to be set to reasonable values to achieve maximum effectiveness in XGB model training:

* learning\_rate: Learning rate.
* max\_depth: Maximum depth of the tree, helps adjust the complexity of the model because when the depth is large, the model will be too complex and prone to overfitting.
* subsample and colsample\_bytree: Adjust the ratio of data and features used to train each tree. Helps reduce overfitting and increase training speed.
* gamma: Set the threshold to reduce the depth of the tree, helping to reduce overfitting.

We will conduct experiments on 3 XGB models from 3 different research papers to see which model learns the most accurately when combining the parameters. The parameters are all applied to the research papers.

1. Model 1

Search for the best parameters for the XGB model using GridSearchCV. It will iterate over all combinations of n\_estimators, learning\_rate, max\_depth, and gamma parameters to find the best parameter set for the model.

Text

Description automatically generated

Figure 25: Model by Wang, W., Liu, W., Zhu, L., Luo, R., Li, G., & Dai, S. (2021). *Stock price prediction methods based on FCM and DNN Algorithms*

Best params: *{'gamma': 0.02, 'learning\_rate': 0.05, 'max\_depth': 15, 'n\_estimators': 100, 'random\_state': 42}*

y\_true = [13.6 14.7 14.7 14.7 13.3]

y\_pred = [14.550083 14.569026 14.71867 14.769511 14.809822]

|  |  |
| --- | --- |
| Best validation score | 0.1679939229254836 |
| MAE | 0.8259512856427362 |
| MSE | 1.1574657673599908 |

Table X: Model 1 score

Comment:

The closer the score is to 1, the more accurate the model predicts on the evaluation dataset. In this case, the Best validation score has a value of 0.168, indicating that the model has a relatively acceptable ability to predict accurately on the evaluation dataset.

However, when comparing the predicted values y\_pred with the actual values y\_true, we see that the mean\_squared\_error value is 1.1574657673599908, which is quite high, indicating that the model still does not predict accurately as expected.

Evaluate the important features that affect the target variable in the model:

Chart

Description automatically generated with medium confidence

Table 26: Feature importance

Visualize the closeADC stock price prediction of model 1:

Chart, histogram

Description automatically generated

Chart, line chart, histogram

Description automatically generated

Graphical user interface, histogram

Description automatically generated

Figure 27: Model 1 visualization

1. Model 2

Similar to model 1, model 2 was referenced by us from another research study, however, the parameters in this study are quite numerous compared to the models we have previously studied.

Text

Description automatically generated

Figure 28: Aarshay Jain. (2023). *Mastering XGBoost Parameter Tuning: A Complete Guide with Python Codes*

Best params: *{'colsample\_bytree': 0.8262033389291895, 'gamma': 1.0742686120994283, 'learning\_rate': 0.06090293647773265, 'max\_depth': 8, 'min\_child\_weight': 8, 'n\_estimators': 649, 'reg\_alpha': 0.011367549151499352, 'reg\_lambda': 21.237483434782096, 'subsample': 0.7151718465902904}*

y\_true = [13.6 14.7 14.7 14.7 13.3]

y\_pred = [14.539252 14.656349 14.77622 14.858872 14.873167]

|  |  |
| --- | --- |
| Best validation score | 0.3790562998877088 |
| MAE | 0.8238584406235638 |
| MSE | 1.0739755140524165 |

Table X: Model 2 score

Comment:

Based on the mean squared error and best validation score values, it can be observed that model 2 is better than model 1.

The lower MSE value of model 2 indicates a lower prediction error.

The higher Best validation score shows that the accuracy of model 2 is better.

In summary, although they predict on the same dataset, model 2 performs better than model 1.

Evaluation of important features that affect the target variable in the model:

**Chart

Description automatically generated**

Figure 29: Feature importance

Visualize the closeADC stock price prediction of model 2:

Chart, histogram

Description automatically generated

Chart, line chart, histogram

Description automatically generated

Graphical user interface, histogram

Description automatically generated

Figure 30: Model 2 visualization

1. Model 3

Text

Description automatically generated

Figure 31: Aarshay Jain. (2023). *Mastering XGBoost Parameter Tuning: A Complete Guide with Python Codes*

Best params: *{'alpha': 0.2, 'colsample\_bytree': 0.7, 'gamma': 0.2, 'lambda': 0.2, 'learning\_rate': 0.1, 'max\_depth': 3, 'min\_child\_weight': 3, 'n\_estimators': 50, 'subsample': 0.7}*

y\_true = [13.6 14.7 14.7 14.7 13.3]

y\_pred = [14.577321 14.588515 14.793188 15.082795 15.101603]

| Best validation score | 0.4125857021887013 |
| --- | --- |
| MAE | 0.8740427050871009 |
| MSE | 1.1546627646976086 |

Table X: Model 3 score

Comment: The results of model 3 show a relatively good level of accuracy, with metrics that are quite similar to model 1 but not better than the second model.

Evaluation of the important features affecting the target variable in the model:

Chart

Description automatically generated with medium confidence

Figure 32: Feature importance

Visualize the closeADC stock price prediction of model 3:

Chart, histogram

Description automatically generated

Chart, histogram

Description automatically generated

Graphical user interface, chart, histogram

Description automatically generated

Figure 33: Model 3 visualization

* 1. **Comparison**

Actual results compared to predicted values ​​learned by the model

Text

Description automatically generated

Figure 34: Results compared

Looking at the validation score results, we can conclude that:

* Result 3 achieved the highest validation score, with a value of 0.4125857021887013.
* Result 2 achieved an average validation score, with a value of 0.3790562998877088.
* Result 1 achieved the lowest validation score, with a value of 0.1679939229254836.

However, to make more reliable conclusions, we need to consider other metrics that have been computed in the problem.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | **Model 1** | **Model 2** | **Model 3** |
| Best validation score | 0.168 | 0.380 | 0.412 |

Table X: Best validation score

From the MAE results of the three models, we can see that:

The model with the lowest MAE result is model 2, with a value of MAE = 0.8238584406235638. This indicates that this model is more accurate than the other two models in predicting the output values.

The model with the highest MAE result is model 3, with a value of MAE = 0.8740427050871009. This shows that this model is less accurate than the other two models in predicting the output values.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | **Model 1** | **Model 2** | **Model 3** |
| MAE | 0.826 | 0.823 | 0.874 |

Table : MAE

Based on MSE values, we can compare the effectiveness of the prediction models:

* Model 1 and Model 3 have MSE values of 1.1574657673599908 and 1.1546627646976086, respectively, which are relatively close to each other.
* Model 2 has the lowest MSE of 1.0739755140524165 among the three results, indicating that the model has produced more accurate and promising results than the other three models in predicting the time series values.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | **Model 1** | **Model 2** | **Model 3** |
| Mean\_squared\_error | 1.1574 | 1.0740 | 1.1546 |

Table : MSE

Based on the experimental results, Model 2 can be considered as the best-performing model among the three evaluated models. This can be explained by some metrics of Model 2. However, this result is not the best yet, and further research is still needed in the future if we want to fully understand the nature of XGB, through the following directions:

* Further research on improved methods for reducing the deviation between predicted values and actual values. One of the commonly used methods is to use optimization algorithms such as Gradient Descent to adjust the parameters of the model.
* Increasing the dataset to enable the model to learn more about the factors that affect the output value. One way to do this is to collect more data from different sources or augment existing data using other techniques such as data augmentation.

1. **Time Series forcating with Deep Learning**
   1. **Preprocessing**

A picture containing text, screenshot, font, line

Description automatically generated

Figure X: Calculate MA

The three MA values—"MA for 10 days," "MA for 50 days," and "MA for 100 days"—are computed and added to the dataset. The goal is to lessen short-term random volatility and establish a stock's long-term average price trend. As a result, investors are able to forecast stock prices and make wiser investing choices.A picture containing line, text, plot, screenshot

Description automatically generated

Figure 35: Show MA results

The closing price of the stock for the last few days is displayed on the MA indicator chart. The average closing price of the stock over the previous 10 days is the basis for the "MA for 10 days" index, as is the case for the "MA for 50 days" and "MA for 100 days" indices.

The stock may be priced cheaper than the average price over a specific period of time if it trades below the MA index, which might be a buy signal. In contrast, if the stock price is above the MA index, it can be a sell signal since it means the stock is priced higher than the average price over a specific time period. In this study, the stock price is calculated using the closing price.

* 1. **Traning model with CNN**

We just add 1 layer CNN for model

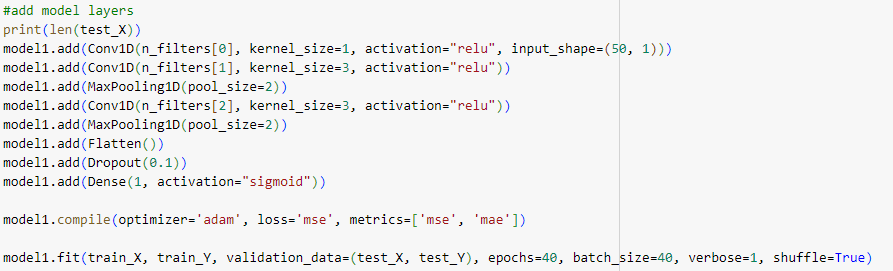
****

Figure X: Add layers CNN for model

Result:

A picture containing text, screenshot, diagram, plot

Description automatically generated

Figure 36: CNN Model to Forecast Stock Prices

|  |  |  |
| --- | --- | --- |
| Model | MSE | MAE |
| CNN | 0.0497 | 0.1615 |

Table : Evaluation results using the CNN-LSTM model

The findings indicate that the CNN model has shown accuracy in forecasting stock prices based on the comparison chart between predicted and real stock prices. Real stock prices is black line, Predicted stock prices is green line

There are still some difference between. The gap between them so far, However the indicators as MSE and MAE are considerate

* 1. **Traning model with LSTM**

For LSTM model, we pass CNN layer and add LSTM layer

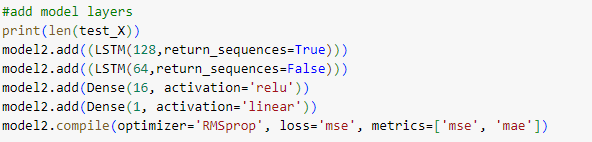
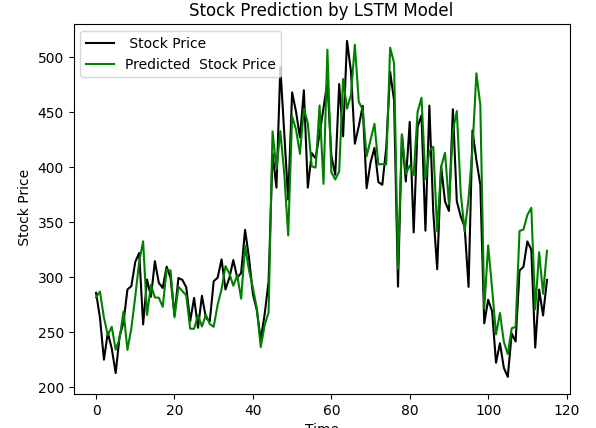


Figure X: Add layers CNN for model

Result:



|  |  |  |
| --- | --- | --- |
| Model | MSE | MAE |
| LSTM | 2.6643 | 1.3056 |

The result show that The predict value has same trend with actual value but the result indexs as - mse: 2.6643 - mae: 1.3056 are much higher than result of CNN model. By adding more data and changing the model's parameters, the model can be improved in the future

* 1. **Traning model with CNN-LSTM**

First, we will prepare the data for the training and testing sets by randomly selecting from the dataset and splitting it into a ratio.



Figure X: Set up train / test

In this study, we divided the test set to account for 20% of the total data and the remaining 80% of the data for the training set. Then, the training and test sets were transformed into NumPy arrays with a size of (number of samples, 1, 100, 1) to fit the input of the CNN model.

Next, we will proceed with training the model.

A picture containing text, receipt, document, font

Description automatically generated

Figure X: The model structure of CNN-LSTM.

We can view the structure of the CNN-LSTM model after adding layers in 3 main stages as follows:

* Phase 1: Add CNN layers

Here, we use the TimeDistributed() layer of TensorFlow to apply CNN layers to each input time series.

The first layer in our model is the Conv1D() layer with 64 filters, kernel size of 3, and relu activation function.

The first layer uses input\_shape to determine the input size. Here, None represents the batch size, 100 represents the length of the time series, and 1 represents the depth of each data point.

The next layer is the MaxPooling1D() layer with a pool size of 2. This layer is used to reduce the output matrix size by retaining the maximum value in each pool block.

We continue to add two more Conv1D() and MaxPooling1D() layers with different numbers of filters and kernel sizes to create a more complex model.

Finally, we add the Flatten() layer to reduce the output size to a one-dimensional vector for use with fully connected neural network layers afterwards.

* Phase 2: Add LSTM layers

Here, we start with the Bidirectional() layer of TensorFlow to combine processing of time-series data in both forward and backward directions.

We use the LSTM() layer with 100 memory units and return\_sequences=True to return output sequences instead of just the last output value.

Next, we add a Dropout() layer with a rate of 0.5 to prevent overfitting of the model.

Then, we add another LSTM() layer with 100 memory units and return\_sequences=False to return only the last output value of the sequence.

Finally, we add another Dropout() layer with a rate of 0.5 to further reduce overfitting.

With the use of Bidirectional() and LSTM() in this neural network, we can handle features of time-series data, such as the dependence of current values on previous values in the sequence, which helps improve the accuracy of the model predictions. The use of Dropout() layers helps the model avoid overfitting and achieve better performance on new data.

* Phase 3: Add Final layer - Dense

Here, we add a final Dense() layer with 1 output unit and 'linear' activation function. This layer will make predictions for the time series values based on the output from the LSTM layer in the previous part. With activation='linear', the model will make continuous predictions instead of classification.

After successfully creating the model, we will train the model based on the training dataset.



Figure X: Training model

We use the fit() function to train the model with the training data train\_X, train\_Y. Here, we also provide test data (test\_X, test\_Y) to evaluate the performance of the model on new data. We train the model with 40 epochs and a batch size of 40. With verbose=1, the model will display its training progress on the console. Finally, we use shuffle=True to shuffle the training data before training the model, which helps the model learn better from the input data.

Result:

A picture containing text, screenshot, plot, diagram

Description automatically generated

Figure X: CNN-LSTM-Based Model to Forecast Stock Prices

|  |  |  |
| --- | --- | --- |
| Model | MSE | MAE |
| CNN-LSTM | 0.0131 | 0.0804 |

Table X: Evaluation results using the CNN-LSTM model

The findings indicate that the LSTM-CNN model has shown accuracy in forecasting stock prices based on the comparison chart between predicted and real stock prices. But this result is the best of other model we have train before, it base on the combination of LSTM and CNN. The fact that the anticipated and actual values frequently coincide shows that the model is effective in forecasting future stock prices. However, there is still some difference between the anticipated and real prices for some data points. This can be the result of insufficient model optimization or unpredictability of external events. By adding more data and changing the model's parameters, the model can be improved in the future. Overall, the LSTM-CNN model has demonstrated strong predictive power and represents a substantial advancement over conventional models in terms of stock price prediction.

1. **Conclusion**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Type | Model | MSE | MAE | Best validation score |
| Deep Learning | LSTM | 2.6643 | 1.3056 |  |
| CNN | 0.0497 | 0.1615 |  |
| CNN-LSTM | 0.0131 | 0.0804 |  |
| Machine Learning | XGBoost Model 1 | 1.1574 | 0.8260 | 0.168 |
| XGBoost Model 2 | 1.0740 | 0.8238 | 0.380 |
| XGBoost Model 3 | 1.1546 | 0.8740 | 0.412 |

Compare between 3 ML model using XGBoost, it easy to find that XGBoost Model 2 can be considered as the best-performing model with MSE = 1.0740, Best validation score = 0.380

Compare between 3 DL model using LSTM and CNN-LSTM, we can realize that CNN-LSTM give result more optimal with MSE = 0.0131, MAE= 0.0804. Compare between ML and DL model depending on MSE, we can figure out DL give us a better result.

# Reference:

[1] Zhang, Y. (2022, December). Stock Price Prediction Method Based on XGBoost Algorithm. In 2022 International Conference on Bigdata Blockchain and Economy Management (ICBBEM 2022) (pp. 595-603). Atlantis Press.

[2] Wang, W., Liu, W., Zhu, L., Luo, R., Li, G., & Dai, S. (2021). Stock price prediction methods based on FCM and DNN Algorithms. Mobile Information Systems, 2021, 1-13.

[3] Eapen, J., Bein, D., & Verma, A. (2019, January). Novel deep learning model with CNN and bi-directional LSTM for improved stock market index prediction. In 2019 IEEE 9th annual computing and communication workshop and conference (CCWC) (pp. 0264-0270). IEEE.

[4] Lu, W., Li, J., Li, Y., Sun, A., & Wang, J. (2020). A CNN-LSTM-based model to forecast stock prices. Complexity, 2020, 1-10.

[5] Muhammad, L. J., Haruna, A. A., Sharif, U. S., & Mohammed, M. B. (2022). CNN-LSTM deep learning based forecasting model for COVID-19 infection cases in Nigeria, South Africa and Botswana. Health and Technology, 1-18.

[6] Mehtab, S., Sen, J., & Dasgupta, S. (2020, November). Robust analysis of stock price time series using CNN and LSTM-based deep learning models. In 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA) (pp. 1481-1486). IEEE.

[7] Suresh, V., Janik, P., Rezmer, J., & Leonowicz, Z. (2020). Forecasting solar PV output using convolutional neural networks with a sliding window algorithm. Energies, 13(3), 723.

[8] Aksan, F., Li, Y., Suresh, V., & Janik, P. (2023). CNN-LSTM vs. LSTM-CNN to Predict Power Flow Direction: A Case Study of the High-Voltage Subnet of Northeast Germany. Sensors, 23(2), 901.

[9] Aarshay Jain (2023). Mastering XGBoost Parameter Tuning: A Complete Guide with Python Codes