STOCK PRICE PREDICTION USING MACHINE LEARNING

1st Tran Phuong Thao
University of Information Technology
20521938@gm.uit.edu.vn

2nd Dang Thi Tuong Vy
University of Information Technology
20522176@gm.uit.edu.vn

3rd Nguyen Thanh Luan
University of Information Technology
20521582@gm.uit.edu.vn

***Abstract*—Predicting changes in stock prices is of great importance to investors, publicly traded companies, and governments. The question of whether the market can be accurately forecasted has been the subject of long-standing debates.This study explores the use of machine learning algorithms to accurately predict stock prices. By analyzing historical price data and other relevant features, the study trains and tests various machine learning models such as Linear Regression, ARIMA, LSTM, GRU, K-Nearest Neighbors (KNN), Support vector regression (SVR), ARIMAX, CNN(Convolutional Neural Network) , HMM(Hidden Markov Model). The results show that these algorithms can effectively capture patterns and relationships in stock price, with deep neural networks performing particularly well. The study concluded that machine learning algorithms hold promise as valuable tools for traders and investors in the volatile stock market.**

***Index Terms*—stock price, machine learning, ARIMA, time series.**

I. Introduction

Forecasting fluctuations in stock prices hold significant significance for investors, publicly listed corporations, and governmental bodies. The issue of whether the market can be reliably predicted has been a topic of enduring discussions. According to Malkiel's (1973) Random Walk Theory, stock prices are arbitrarily set, making it impractical to outperform the market. However, the emergence of artificial intelligence has demonstrated empirical proof that forecasting stock price movements is indeed achievable.

Additionally, the primary aim of this problem is to analyze the patterns and variations in stock prices to facilitate well-informed investment choices. The ability to predict stock prices can be instrumental in asset valuation, uncovering potential investment prospects, and effectively mitigating risks.

To predict stock prices, various methods, and tools are commonly used, including technical analysis, fundamental analysis, and data modeling. Technical analysis involves studying historical patterns and trends in stock prices to make predictions about the future. Fundamental analysis involves researching economic, financial, and company factors to evaluate the fundamental value of a company and thereby predict stock prices. Data modeling utilizes machine learning and artificial intelligence methods to analyze historical data and build predictive models for future stock prices.

Linear regression is a widely used algorithm in financial analysis, predicting stock and providing a simple yet effective approach to modeling the relationship between independent and dependent variables. In addition, machine learning algorithms that are often used in stock price prediction problems such as KNN, SVR also give quite good and reliable prediction results. With ARIMA, ARIMAX models, on the other hand, are specifically designed for time series forecasting and have been successfully applied in various domains. LSTM and GRU are deep learning architectures that excel at capturing long-term dependencies. To evaluate the problem as well as the performance of the machine learning model used to predict in the above data set, we will conduct an analysis of how, data, accuracy, and efficiency in the stock prediction process. 9 models have been introduced.

In addition to evaluating the performance and comparing results of machine learning algorithms using stock price data, this research paper aims to delve deeper into understanding the strengths and weaknesses of each algorithm. By analyzing their accuracy, robustness, and computational efficiency, we seek to gain insights into their applicability and limitations in the context of stock price prediction.

By conducting a thorough analysis and comparison of these machine learning algorithms, we strive to provide valuable insights into their performance in the dynamic and unpredictable stock market. This research can potentially uncover new perspectives and approaches for forecasting stock prices, benefiting both individual investors and institutional traders in optimizing their investment strategies.

II. Related research

The use of ARIMA, LSTM, GRU, K-Nearest Neighbors (KNN), Support Vector Regression (SVR), ARIMAX, Convolutional Neural Network (CNN), and Hidden Markov Model (HMM) models for stock price prediction is entirely feasible and scientific, as evidenced by several papers published in international conferences on this topic.

Here are some notable works for ARIMA, LSTM, and GRU models:

- " Predict stock prices with ARIMA and LSTM" by Ruochen Xiao, Yingying Feng, Lei Yan, Yihan Ma - This paper compares the performance of ARIMA and LSTM models in stock price prediction. It discusses the application of ARIMA and LSTM to time series data of stock prices and compares the prediction results.
- "Stock Price Correlation Coefficient Prediction with ARIMA-LSTM Hybrid Model" by Hyeong Kyu Choi

- ”Stock Price Prediction Using LSTM, RNN and CNN Models” (2017) - This paper focuses on comparing the performance of LSTM, RNN, and GRU models in stock price prediction. It provides a detailed analysis of these models and compares their prediction results on a dataset of stock prices.

With Machine Learning and other models, we can also mention some research works as follows:

- ”Stock Market Trend Prediction Using K-Nearest Neighbor (KNN) Algorithm” by V.SARALA , G.N.V.PHANI BHUSHAN (2022) - This paper proposes using the KNN model to predict stock prices. It discusses how to use KNN to find similar points in the feature space and predict stock prices based on these similar points.
- ”Support Vector Regression for Financial Time Series Forecasting” by Wei Hao, Songnian Yu - This paper introduces the use of Support Vector Regression (SVR) for predicting stock prices. It describes the application of SVR to time series data and compares the performance of SVR with other prediction methods.
- Shahzad Zaheer et al. (2023) [1] used LSTM and Deep Learning Model like CNN to take the input stock data and forecasts two stock parameters close price and high price for the next day, and result shows which model perform the best.
- ”Stock Market Trend Analysis Using Hidden Markov Models” Kavitha G, *Udhayakumar A , Nagarajan D This paper proposes using HMM to predict the movement direction of stock prices (increase/decrease). It describes the construction of the HMM model for stock price data and utilizes it to predict price changes in the future.

III. Method

Input is time series historical data of stock price. After the data goes through the preparation step: the selection of required features and preprocessing splits the data according to the training, validation, and testing dataset accordingly. Then will use Linear Regression, ARIMA, ARIMAX, LSTM, GRU, CNN, HMM, SVR, KNN to train. The selected model is trained using the prepared data set. During training, the model learns the underlying patterns, relationships, and structures in the data. The training process involves tuning the model's parameters to minimize the difference between the predicted output and the real output from the training data. After the model is trained, it needs to be evaluated to evaluate its performance and generalizability, this step involves using a separate evaluation dataset or using techniques such as cross-validate to estimate the model's performance on unseen data. Different metrics are used to measure performance depending on the specific problem, if after evaluating the model performance is not effective then go back to adjust the hyperparameters and then do the training again. Finally, the results are presented in the form of graphs.

The research process goes through the following steps:

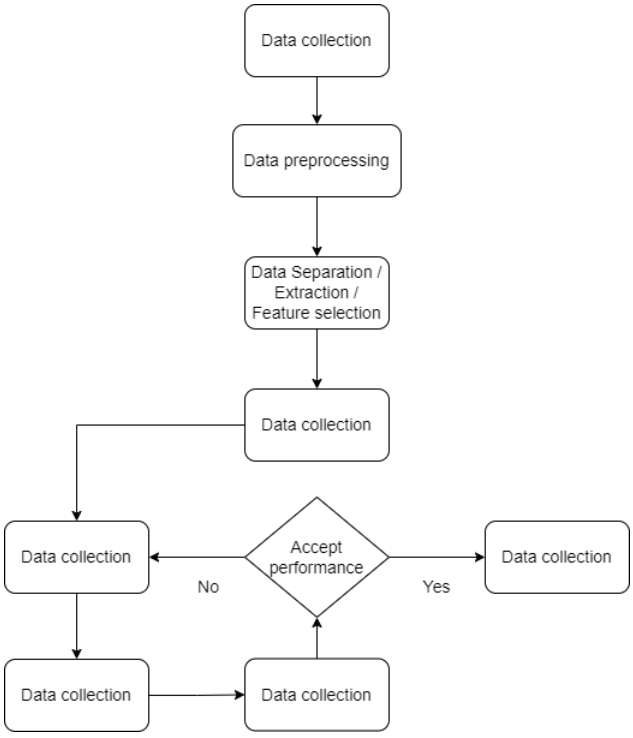

Fig. 1. Implementation process

- Step 1: Data collection - collect a representative and relevant dataset to train the model. The dataset must include input data (features) and corresponding output data (labels or targets) in this study we choose time series historical data of stock price.
- Step 2: Data preprocessing - After the data set is collected, it usually requires preprocessing. This step involves cleaning the data, dealing with missing values, removing outliers, and converting the data to a format suitable for training.
- Step 3: Data Separation / Extraction / Feature selection. In this step, we separate the data at different rates including train, test, validation data, and then select features to conduct training.
- Step 4: Model selection - in this study, we will perform on models including Linear Regression, ARIMA, ARIMAX, LSTM, GRU, CNN, HMM, SVR, KNN.
- Step 5: Tune the hyperparameters - set the appropriate hyperparameters for several models.
- Step 6: Model Training - The selected model is trained using the prepared dataset.
- Step 7: Model evaluation: After the model is trained, it needs to be evaluated to evaluate its performance and generalizability. If the performance is not acceptable, go back to step 5.
- Step 8: Show result.

A. Dataset

The dataset is a trio of stock prices including Amazon (AMZN), Nvidia (NVDA), Tesla (TSLA). The data is taken

from Yahoo! Financeii, a media website that is part of Yahoo! network. It provides financial news, data, and commentary including stock quotes, press releases, financial reports, and original content. It also offers a number of online tools for managing personal finances. In this paper, the time period represented starts from 2017-12-01 to 2023-06-09, and the attributes of the dataset are described in table I.

TABLE I
ATTRIBUTES IN THE STOCK PRICE DATASET

Attribute	Describe
Date	Stock price trading day
Open	The initial opening/price at a certain time
High	Highest opening price
Low	Lowest price of opening price
Close	The closing/final price of the stock at a certain time
Adj Close	Closing price after adjustment
Volume	Number of transactions during the day

Table II describes the measure of features selected for training in the dataset. In this paper, we choose the closing price of the stock price to perform.

TABLE II
DESCRIBES THE MEASURE OF FEATURES SELECTED FOR TRAINING IN THE DATASET

Measure	Amazon (AMZN)	Nvidia (NVDA)	Tesla (TSLA))
Count	1389	1389	1389
Mean	117.56	126.35	133.88
Standard deviation	1227.99	6621.81	12986.89
Min	56.70	31.77	11.93
Max	186.57	401.11	409.97

B. Tood used

During the experiment, we use Python language and the support tool is Google Colab. In addition, we use Python's built-in libraries such as Pandas to process data in the form of data frames. Matplotlib plots to visualize data. The figure below illustrates the result after completing a model. Numpy helps with math and matrix operations in this experiment. And finally, the Scikit-learn library supports machine learning and regression models.

C. Models

1) Linear Regression

Linear regression [1] is a type of statistical analysis used to predict the relationship between two variables. It assumes a linear relationship between the independent variable and the dependent variable and aims to find the best-fitting line that describes the relationship. Linear Regression is a simple yet powerful and mostly used algorithm in data science[1].

The formula for Multiple Linear Regression is [1]:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p + \varepsilon \quad (1)$$

In this formula:

- Y represents the dependent variable (output) that we want to predict or explain.
- $X_1, X_2, ... X_p$ are the independent variables (inputs) used to predict Y.
- $\beta_0, \ \beta_1, \ \beta_2, ..., \beta_p$: are the regression coefficients corresponding to the independent variables. They represent the extent of influence of each independent variable on the dependent variable.
- ε is the error term, representing the difference between the actual value of Y and the predicted value by the model.

2) Support vector regression (SVR)

Support Vector Regression (SVR) [2] is a powerful machine learning algorithm utilized for regression analysis tasks. It belongs to the family of Support Vector Machines (SVM) and is specifically designed to handle continuous target variables. The primary objective of SVR is to identify and construct a mathematical function that approximates the underlying relationship between the input variables and the target variable. SVR based on finding a linear regression line in the feature space that minimizes the error between the data points and the regression line. However, SVR can handle situations where the data is not clearly divided along a linear regression line.

Unlike traditional regression models, SVR focuses on minimizing the prediction error rather than fitting the data precisely. It accomplishes this by finding an optimal hyperplane that maximizes the margin, or distance, between the predicted values and the actual data points. SVR achieves a balance between simplicity and flexibility by allowing a specified tolerance, or margin of error, around the predicted values. Let us now define a different type of loss function termed an E-insensitive loss in SVR.

$$L(y, F(x_i, \hat{w})) = max(0, |y - F(x_i, \hat{w}) - \varepsilon)$$

- $L(y, F(x_i, \hat{w}))$: Loss function.
- y: The actual value of the data point.
- $F(x_i, \hat{w})$: Value predicted by the model.
- ε: Adjustment parameter allows deviation.

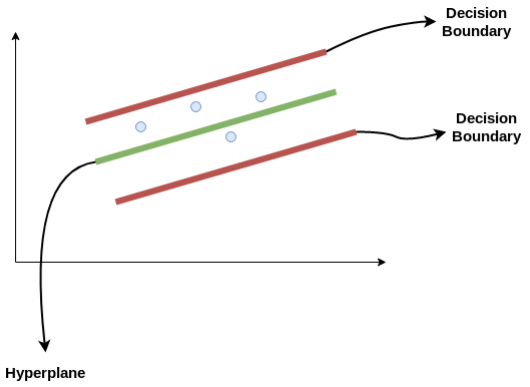

Fig. 2. Decision Boundary of SVR

In Support Vector Regression (SVR), can use various types of kernels to create different classification functions.

- Linear: This is the simplest and most basic kernel in SVR. It performs a linear projection of the input data into a linear space.

TABLE III
KERNELS USED IN THE SUPPORT VECTOR MACHINE

Kernel	Function
Linear	$k(x,z) = x^T z$
Polynomial	$k(x,z) = (r + \gamma x^T z)^d$
RBF	$k(x,z) = exp(-\gamma \|\|x-z\|\|_2^2), \gamma > 0$
Sigmoid	$k(x,z) = tanh(\gamma x^T z + r)$

- Polynomial: This kernel transforms the input data into a polynomial space, determined by a degree parameter and a bias coefficient.
- RBF: This kernel uses the RBF function to transform the input data into a non-linear space. The RBF function is defined by a slope parameter.
- Sigmoid: This kernel transforms the input data into a non-linear space using the sigmoid function. It can be used for problems with binary data.

3) K-Nearest Neighbors(KNN)

K-Nearest Neighbors (KNN) [4] is a popular machine learning algorithm used for both classification and regression tasks. While commonly associated with classification, KNN can also be adapted for regression analysis, where the goal is to predict continuous target variables.

In KNN regression, the algorithm works by finding the K nearest neighbors to a given data point in the feature space. Instead of assigning a class label, as in KNN classification, KNN regression calculates the average or weighted average of the target variable values of these K neighbors. This aggregated value serves as the predicted value for the target variable of the query data point. For a given input x of training data, K observations with xi in the proximity are considered and the average of the response of those K independent variables gives y:

$$\hat{y} = \frac{1}{k} \sum_{i}^{k} y_i \tag{2}$$

Several distance formulas are commonly used to calculate the k nearest neighbors for 2 data points, x and y, with k attributes. [4]

- Euclidean

$$\sqrt{\sum_{i=1}^{k} (x_i - y_i)^2} \tag{3}$$

- Manhattan

$$\sum_{i=1}^{k} |x_i - y_i| \tag{4}$$

- Minkowski

$$\left(\sum_{i=1}^{k} (|x_i - y_i|)^q \right)^{\frac{1}{q}} \tag{5}$$

4) Convolutional Neural Network(CNN)

CNN stands for Convolutional Neural Network, which is an advanced model widely used in the field of Deep Learning [10]. The CNN allows users to build classification and prediction systems with extremely high accuracy.

A convolutional neural network consists of an input layer, hidden layers and an output layer. In a convolutional neural network, the hidden layers include one or more layers that perform convolutions. Typically this includes a layer that performs a dot product of the convolution kernel with the layer's input matrix. This product is usually the Frobenius inner product, and its activation function is commonly ReLU [11]. As the convolution kernel slides along the input matrix for the layer, the convolution operation generates a feature map, which in turn contributes to the input of the next layer. This is followed by other layers such as pooling layers,flatten layer, fully connected layers, and normalization layers [10].

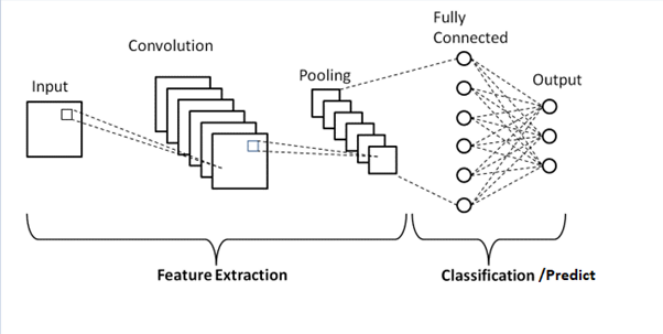

Fig. 3. Convolutional Neural Network(CNN) architecture diagram.

- Convolution layer: When using Convolutional layer in a CSV dataset, we perform convolution between the columns in the data table and the filters (kernels) to generate feature maps. These feature maps are used to extract features from the data.
- Max pooling layer: This layer is used to reduce the size of the feature maps and the number of parameters in the model. The Max pooling layer works by dividing the feature map into non-overlapping pooling regions and selecting the maximum value from each region to include in the output. This selection of the maximum value preserves the important features of the data while ignoring small irrelevant details.
- Flatten layer: After using the Convolutional layer and Max pooling layer to extract features from the CSV data, we obtain feature maps. These feature maps can be processed by the Flatten layer to convert them into a vector that can be input to the Fully connected layer.
- Fully connected layer (FCL): When using FCL, the features of the data are represented as a vector and are input into FCL layers to perform classification or prediction. These FCL layers learn the complex relationships between the features and provide predictions about the outcome.

CNN uses non-linear activation functions after each layer to enable the model to learn complex and non-linear features in the data. The commonly used activation functions are as follows [11]:

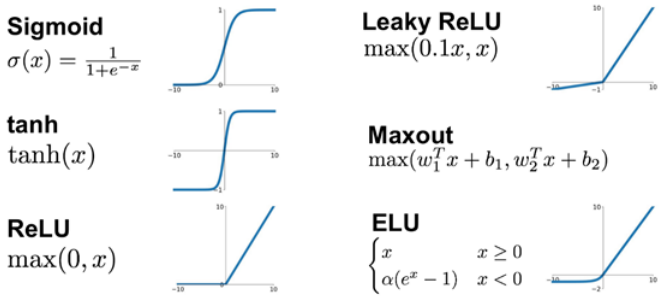

Fig. 4. Activation of Convolutional Neural Network

During the training process, CNN uses the backpropagation algorithm to adjust the weights of the network layers in order to minimize the loss function between the predicted output and the actual output. Optimization algorithms such as Gradient Descent and its variants are commonly applied to update the network weights.

Compute the gradient of the loss function with respect to the predicted output

$$\frac{\partial L}{\partial \mathbf{a}^L} \tag{6}$$

5) Hidden Markov Model (HMM)

Out of the several models available for modeling discrete sequential data, hidden Markov models are one of the more simple but powerful ones— they have been applied successfully to a wide variety of fields such as statistical mechanics, DNA analysis, speech recognition, and stock market predictions.[66]

The first-order Markov process makes a very important simplification to observed sequential data—the current system state depends only on the previous system state.

Additionally, hidden Markov models make one more important modification to the Markov process — the actual system states are assumed to be unobservable and are hidden.

For a sequence of hidden states Z, the hidden Markov process emits a corresponding sequence of observable processes X.

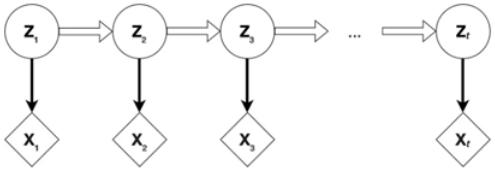

Fig. 5. Graph of a hidden Markov process. Unable to observe the actually hidden states of the system Z, and can only observe the observable processes X

Hidden Markov models are used to model a hidden Markov process. Hidden Markov models are defined by the following 3 model parameters:

- Initial hidden state probabilities. This vector describes the initial probabilities $\pi = [\pi_0, \pi_1, \pi_2, \ ...]$ of the system being in a particular hidden state
- Hidden state transition matrix A. Each row in A corresponds to a particular hidden state, and the columns for each row contain the transition probabilities from the current hidden state to a new hidden state.
- Observable emission probabilities $\theta = [\theta_0, \theta_1, \theta_2, \ ...]$. This vector describes the emission probabilities for the observable process X_i gave some hidden state Z_i.

[7]

6) ARIMA

ARIMA is an autoregressive integrated moving average, or ARIMA is a statistical analysis model that uses time series data to better understand the data set or predict future trends.

An ARIMA model can be understood by outlining each of its components as follows:

- Autoregression (AR): refers to a model that shows a changing variable that regresses on its own lagged, or prior, values.
- Integrated (I): represents the differencing of raw observations to allow the time series to become stationary (i.e., data values are replaced by the difference between the data values and the previous values).
- Moving average (MA): incorporates the dependency between an observation and a residual error from a moving average model applied to lagged observations.

Each component in ARIMA functions as a parameter with a standard notation. For ARIMA models, a standard notation would be ARIMA with p, d, and q, where integer values substitute for the parameters to indicate the type of ARIMA model used. The parameters can be defined as:

- p: the number of lag observations in the model, also known as the lag order.
- d: the number of times the raw observations are differenced; also known as the degree of differencing.
- q: the size of the moving average window, also known as the order of the moving average. [8]

The formula for a simple ARIMA(p,d,q) is:

$$Y_t = c + \phi_1 Y_{t-1} + \ ... + \ \phi_p Y_{t-p} + \theta_1 \varepsilon_{t-1} + ... + \theta_q \varepsilon_{t-q} + \varepsilon_t \tag{7}$$

In this formula:

- Y_t is the value of the time series at time t.
- c is a constant term representing the mean value of the time series.
- $\phi_1, ..., \phi_p$ are the autoregressive (AR) coefficients for the lagged values in the model.
- $\theta_1, ..., \theta_q$ are the moving average (MA) coefficients for the lagged error terms in the model.
- ε_t is the error term or residual at time t.

7) ARIMAX

ARIMAX (Autoregressive Integrated Moving Average with Exogenous Variables) is an extension of the ARIMA model that incorporates exogenous or external variables into the time series analysis. The X added to the end stands for

"exogenous". In other words, it suggests adding a separate different outside variable to help measure our endogenous variable.

X is the exogenous variable and it can be any variable. It can be a time-varying measurement like the inflation rate or the price of a different index. Or a categorical variable separating the different days of the week. It can also be a Boolean accounting for the special festive periods. Finally, it can stand for a combination of several different external factors.

ARIMAX models are used when we want to analyze and forecast time series data, taking into account not only past values but also the impact of exogenous variables. [9]

The formula for a simple ARIMAX(p,d,q) is:

$$Y_t = c + \phi_1 Y_{t-1} + \ldots + \phi_p Y_{t-p} + \beta_1 X_1(t) + \\ \ldots + \beta_q X_q(t) + \theta_1 \varepsilon_{t-1} + \ldots + \theta_q \varepsilon_{t-q} + \varepsilon_t \quad (8)$$

The ingredients in the formula are the same as ARIMA but with more:

- $X_1(t), \ldots, X_q(t)$ is the value of the time series at time t.
- $\beta_1, \ldots, \beta_q$ are the coefficients associated with the exogenous variables, representing their impact on the dependent variable.

8) Long Short-Term Memory (LSTM)

Long Short-Term Memory networks (LSTM) [12], often referred to as LSTM networks, are a special type of recurrent neural network (RNN) that are capable of learning long-term dependencies. LSTMs were introduced by Hochreiter Schmidhuber in 1997.

LSTM incorporates feedback connections, making it a recurrent neural network (RNN) [12]. This enables LSTM to not only process individual data points like images but also handle sequential data such as speech or video. LSTM, being a special type of RNN, LSTM is designed to address the issue of long-term dependencies. It has the inherent ability to remember information over extended periods of time without the need for explicit training. This means that LSTMs can retain information by default, without any additional intervention.

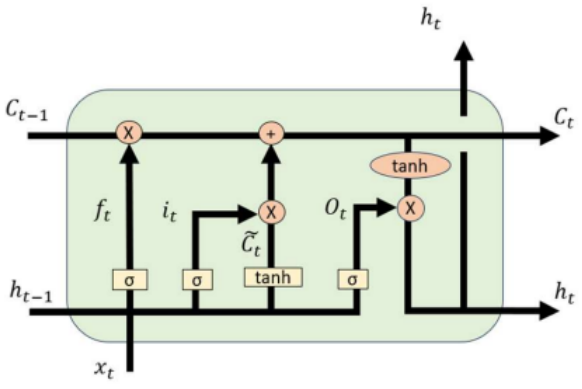

Fig. 6. Model of LSTM architectural [**?**]

In state t of the LSTM model :

Output: c_t, h_t

Input: c_{t-1}, h_{t-1}, x_t

- c_t, h_t c is cell state, h is hidden state.
- c_{t-1}, h_{t-1} output of previous layer.
- x_t input in state t.

f_t, i_t, o_t corresponding to forget gate, input gate, and output gate [15].

- Forget gate :

$$f_t = \sigma(U_t * x_t + W_t * h_{t-1} + b_f) \quad (9)$$

- Input gate:

$$i_t = \sigma(U_i * x_t + W_i * h_{t-1} + b_i) \quad (10)$$

- Output gate:

$$o_t = \sigma(U_o * x_t + W_o * h_{t-1} + b_o) \quad (11)$$

9) Gated Recurrent Unit (GRU)

GRUs are very similar to Long Short Term Memory(LSTM) [13]. Just like LSTM, GRU uses gates to control the flow of information. They are relatively new as compared to LSTM. This is the reason they offer some improvement over LSTM and have simpler architecture.

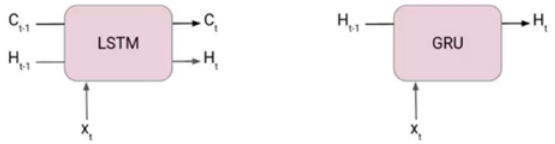

Fig. 7. Difference between LSTM and GRU

Another Interesting thing about GRU is that, unlike LSTM, it does not have a separate cell state (Ct). It only has a hidden state(Ht). Due to the simpler architecture, GRUs are faster to train.

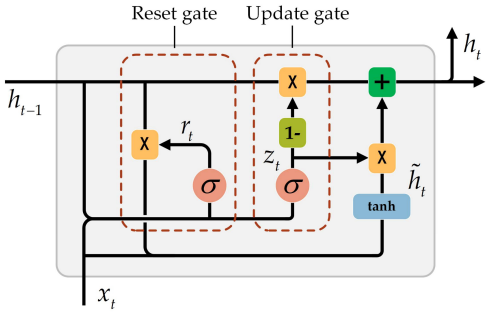

Fig. 8. Model of GRU architectural

In state t of the GRU model:

Output: h_t

Input: h_{t-1}, x_t

- h is the hidden state.

- h_{t-1} output of previous layer.
- x_t input in state t.

r_t, u_t corresponding to reset gate, and update gate.

- Reset Gate (Short-term memory) [14]

$$r_t = \sigma\left(U_r * x_t + W_r * h_{t-1}\right) \tag{12}$$

- Update Gate (Long-term memory)

$$u_t = \sigma\left(U_u * x_t + W_u * h_{t-1}\right) \tag{13}$$

D. Evaluation forecasting models

To evaluate the accuracy of the models, we use three parameters including root mean square error (RMSE), mean absoluate error (MAE), mean absolute percentage error (MAPE). The algorithm with the lowest value of those three parameters has the best performance.

E. Results

TABLE IV
RESULTS WHEN TRAINING ON AMAZON DATASETS

Model	Train : Test : Valid	Measure		
		MAE	MAPE	RMSE
ARIMA	7 : 2 : 1	9.66	6.11	11.50
	6 : 3 :1	20.45	11.96	22.08
	5 : 3 : 2	8.68	5.25	10.59
LN	7 : 2 : 1	20.63	13.72	25.98
	6 : 3 :1	9.03	5.23	11.18
	5 : 3 : 2	28.02	17.05	28.85
SVR	7 : 2 : 1	9.28	7.82	11.16
	6 : 3 :1	7.46	5.80	9.51
	5 : 3 : 2	29.32	18.09	30.14

Model	Train : Test : Valid	Measure		
		MAE	MAPE	RMSE
CNN	7 : 2 : 1	6.01	4.60	7.41
	6 : 3 :1	5.02	3.51	6.23
	5 : 3 : 2	3.69	2.29	4.81
LSTM	7 : 2 : 1	3.59	2.77	4.58
	6 : 3 :1	3.22	2.23	4.08
	5 : 3 : 2	3.13	1.98	4.11
GRU	7 : 2 : 1	2.97	2.23	3.99
	6 : 3 :1	2.61	1.87	3.56
	5 : 3 : 2	2.73	1.71	3.72

Model	Train : Test : Valid	Measure		
		MAE	MAPE	RMSE
ARIMAX	7 : 2 : 1	1.06	0.69	1.32
	6 : 3 : 1	0.59	0.35	0.77
	5 : 3 : 2	0.77	0.48	1.04
HMM	7 : 2 : 1	2.62	1.64	3.31
	6 : 3 :1	1.66	0.98	2.11
	5 : 3 : 2	2.04	1.29	2.65
KNN	7 : 2 : 1	1.69	1.28	2.32
	6 : 3 :1	1.78	1.23	2.51
	5 : 3 : 2	1.75	1.07	2.48

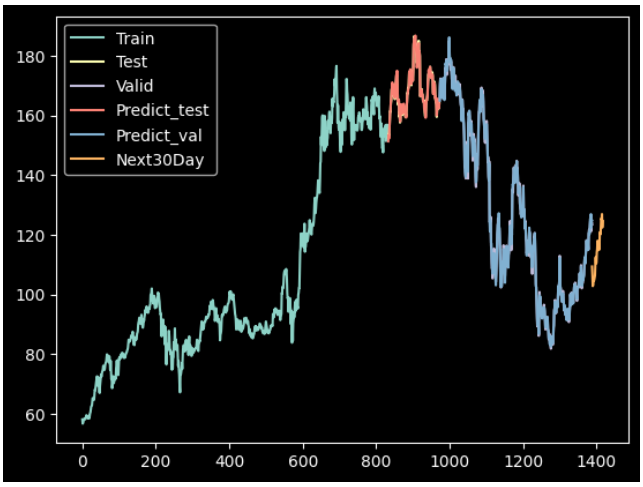

Fig. 9. ARIMAX on AMAZON DATASET WITH TRAIN:TEST:VALID - 6:3:1

TABLE V
RESULTS WHEN TRAINING ON NVIDIA DATASETS

Model	Train : Test : Valid	Measure		
		MAE	MAPE	RMSE
ARIMA	7 : 2 : 1	57.75	20.83	67.23
	6 : 3 :1	61.67	31.56	68.79
	5 : 3 : 2	35.95	19.26	50.89
LN	7 : 2 : 1	88.69	32.67	97.41
	6 : 3 :1	64.53	33.63	69.36
	5 : 3 : 2	79.28	48.34	84.79
SVR	7 : 2 : 1	42.28	26.16	47.11
	6 : 3 :1	33.76	14.46	45.08
	5 : 3 : 2	38.70	16.83	48.17

Model	Train : Test : Valid	Measure		
		MAE	MAPE	RMSE
CNN	7 : 2 : 1	17.44	6.03	21.74
	6 : 3 :1	21.43	7.48	28.73
	5 : 3 : 2	26.64	10.61	32.68
LSTM	7 : 2 : 1	11.29	3.83	14.35
	6 : 3 :1	8.68	3.26	11.38
	5 : 3 : 2	25.20	10.32	32.46
GRU	7 : 2 : 1	7.25	3.40	9.53
	6 : 3 :1	7.84	2.83	10.79
	5 : 3 : 2	10.31	3.92	14.27

TABLE VI
RESULTS WHEN TRAINING ON TESLA DATASETS

Model	Train : Test : Valid	Measure		
		MAE	MAPE	RMSE
ARIMA	7 : 2 : 1	66.80	19.40	75.53
	6 : 3 :1	26.41	11.02	32.48
	5 : 3 : 2	87.02	37.43	97.98
LN	7 : 2 : 1	96.52	28.41	105.00
	6 : 3 :1	62.32	26.94	64.42
	5 : 3 : 2	145.86	66.92	151.27
SVR	7 : 2 : 1	20.56	7.73	26.63
	6 : 3 :1	75.26	28.20	85.26
	5 : 3 : 2	58.42	22.96	66.25

Model	Train : Test : Valid	Measure		
		MAE	MAPE	RMSE
ARIMAX	7 : 2 : 1	3.18	1.24	4.13
	6 : 3 :1	1.06	0.58	1.31
	5 : 3 : 2	1.07	0.71	1.41
HMM	7 : 2 : 1	7.59	2.83	9.82
	6 : 3 :1	2.88	1.70	3.61
	5 : 3 : 2	2.37	1.81	3.02
KNN	7 : 2 : 1	1.67	1.28	2.32
	6 : 3 :1	1.77	1.23	2.51
	5 : 3 : 2	1.75	1.07	2.48

Model	Train : Test : Valid	Measure		
		MAE	MAPE	RMSE
CNN	7 : 2 : 1	17.44	6.03	21.74
	6 : 3 :1	21.43	7.48	28.73
	5 : 3 : 2	26.68	10.61	32.68
LSTM	7 : 2 : 1	11.29	3.83	14.35
	6 : 3 :1	8.68	3.26	11.38
	5 : 3 : 2	25.20	10.32	32.46
GRU	7 : 2 : 1	7.25	3.40	9.53
	6 : 3 :1	7.84	2.83	10.79
	5 : 3 : 2	10.31	3.92	14.27

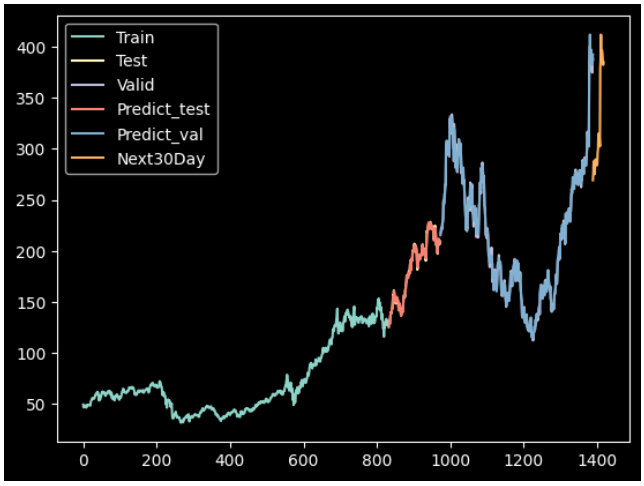

Fig. 10. ARIMAX on NVIDIA DATASET WITH TRAIN:TEST:VALID - 6:3:1

Model	Train : Test : Valid	Measure		
		MAE	MAPE	RMSE
ARIMAX	7 : 2 : 1	4.32	1.34	5.44
	6 : 3 : 1	1.56	0.69	2.03
	5 : 3 : 2	2.10	1.02	2.76
HMM	7 : 2 : 1	11.43	3.51	14.78
	6 : 3 :1	4.47	2.03	5.89
	5 : 3 : 2	4.81	2.59	6.24
KNN	7 : 2 : 1	1.69	1.28	2.32
	6 : 3 :1	1.77	1.23	2.51
	5 : 3 : 2	1.75	1.07	2.48

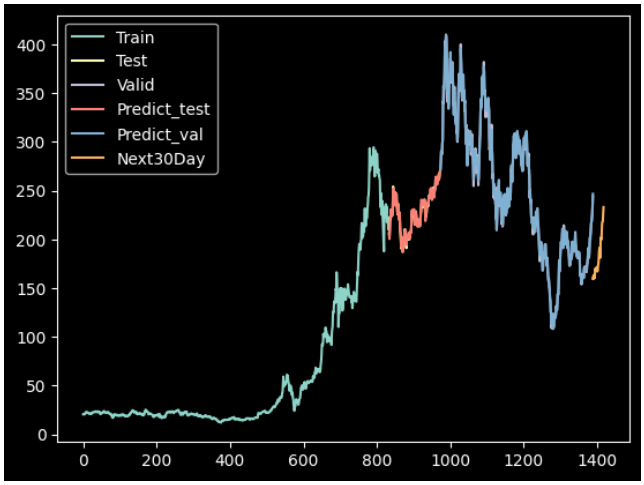

Fig. 11. ARIMAX on TESLA DATASET WITH TRAIN:TEST:VALID - 6:3:1

IV. Conclusion

Stocks present themselves as attractive investment opportunities, but they also come with inherent risks stemming from their volatile and unpredictable nature. Fortunately, these advanced algorithms leverage the power of neural networks to capture complex patterns and trends in stock price data. By analyzing historical data and identifying anomalies, these algorithms can provide valuable insights for forecasting future price movements. The robust performance. ARIMAX algorithms underscore their effectiveness in handling the dynamic and volatile nature of stock markets. With their ability to deliver accurate predictions, these algorithms hold significant potential for informing investment decisions and risk management strategies in the stock predict. Among the nine models tested (Linear Regression, ARIMA, ARIMAX, SVR, KNN, LSTM, CNN, GRU, and HMM), the results of this study demonstrate that the ARIMAX models were the most suitable for accurately predicting the future prices of. AMAZON, NVIDIA, TESLA stocks in the given time series. This study highlights the importance of exploring diverse modeling approaches when predicting stock prices and highlights the potential effectiveness of employing ARIMAX models for forecasting future stock prices. The findings suggest that these models can provide valuable insights for investors and analysts in making informed decisions in the stock market. The utilization of regression and artificial intelligence algorithms, including ARIMAX, has shown significant effectiveness in forecasting stock prices. However, additional research and implementation of further techniques are necessary to enhance these models by incorporating additional external variables that impact stock prediction. With ongoing advancements in data analysis, artificial intelligence, and the continuous improvement of accurate prediction techniques, investors and businesses can confidently navigate the complex financial landscape and achieve high success in their investment endeavors.

References

[1] Everything you need to Know about Linear Regression! (accessed Jun. 15, 2023).
[2] Harris Drucker, Chris J.C.Burges, Linda Kaufman, Alex Smola, Vladimir Vapoik: “Support Vector Regression Machines”
[3] ” Chih-Wei Hsu, Chih-Chung Chang and Chih-Jen Lin ”A Practical Guide to Support Vector Classification (2010)
[4] Jacek Biesiada và Adam Wierzbicki ”k-Nearest Neighbor Algorithm for Real-Time Continuous Data Stream Regression” (2008).
[5] ”Local k-Nearest Neighbors Regression” ca S. Shekhar và S. Chawla (1998)
[6] Hidden Markov Models with Python https://medium.com/@natsunoyuki/hidden-markov-models-with-python-c026f778dfa7 (accessed Jun. 18, 2023).
[7] Hidden Markov Model in Machine learning https://www.geeksforgeeks.org/hidden-markov-model-in-machine-learning/ (accessed Jun. 19, 2023).
[8] Autoregressive Integrated Moving Average (ARIMA) Prediction Model https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asptoc-how-to-build-an-arima-model (accessed Jun. 14, 2023).
[9] What Is an ARIMAX Model? https://365datascience.com/tutorials/python-tutorials/arimax/ (accessed Jun. 19, 2023).
[10] Yann LeCun, Yoshua Bengio và Geoffrey Hinton ”Deep Learning” (2015)
[11] Pramit Saha and Dipti Prasad Mukherjee ”A Comparative Analysis of Activation Functions for Deep Neural Networks” (2018).
[12] ”Long Short-Term Memory” Sepp Hochreiter và Jürgen Schmidhuber (1997)
[13] Introduction to Gated Recurrent Unit (GRU)?https://www.analyticsvidhya.com/blog/2021/03/introduction-to-gated-recurrent-unit-gru/ (accessed Jun. 19, 2023).
[14] Introduction to Gated Recurrent Unit (GRU)?
[15] The Ultimate Guide to Building Your Own LSTM Models: https://www.projectpro.io/article/lstm-model/832