Predict Technology Company Stock Prices using Statistical, Machine Learning and Deep Learning Models

Truong Gia Huy, Phan Nguyen Lam Ha, and Nguyen Duy Khanh
[1]Faculty of Information Systems, University of Information Technology, (e-mail: 21522172@gm.uit.edu.vn)
[2]Faculty of Information Systems, University of Information Technology, (e-mail: 21522030@gm.uit.edu.vn)
[3]Faculty of Information Systems, University of Information Technology, (e-mail: 21522208@gm.uit.edu.vn)

ABSTRACT The stock market allows buyers and sellers to interact and transact shares that represent their ownership of a business. The creation of trustworthy prediction models for the equities market enables investors to make better choices. This research aims to predict the future stock prices of the three largest technology businesses (MSFT, GOOG, APPL) in the world. In this research, we first review the share prices of the above business in recent years. After that, we use statistical and descriptive methods, machine learning and deep learning algorithms such as Linear Regression, Autoregressive Integrated Moving Average (ARIMA), Long Short Term Memory (LSTM), Regression Neural Networks (RNN), Gated Regression Units (GRU), Generalized Autoregressive Conditional Heteroskedasticity (GARCH), Time-series Dense Encoder Model (TiDE), Multi-layer Perceptron (MLP) for analysis and predict time series stock prices in the futures.

INDEX TERMS Stock Prices, Machine Learning, Deep Learning, Linear Regression, ARIMA, LSTM, RNN, GRU, GARCH, TiDE, MLP

I. INTRODUCTION

The Stock Market is a highly complex system, where huge chunks and volumes of information. Data is generated instantaneously and constantly changes in small proportions with different factors and diversity. Since the stock market is primarily dynamic, nonlinear, complex, nonparametric, and chaotic in character, stock market prediction is regarded as a tough task of the financial time series prediction process. The stock market is additionally influenced by a variety of macroeconomic factors, including political developments, corporate policies, general economic conditions, investor expectations, institutional investment preferences, movement in other stock markets, investor psychology, etc Stock Price Prediction is the task of forecasting future stock prices based on historical data and various market indicators. The goal of stock price prediction is to help investors make informed investment decisions by providing a forecast of future stock prices.

Many algorithms and techniques help us predict stock prices. In this paper, we will use statistical models and machine learning algorithms like Linear Regression, ARIMA, LSTM, RNN, GRU, GARCH, TiDE, and MLP models to predict the stock closing prices of three corporations: Microsoft(MSTF), Apple(APPL), Google(GOOG) for the next 30, 60 and 90 days.

Finally, we use the RMSE, MAPE(%) and MAE evaluation metrics to determine which forecasting model performs best on each available data set.

II. RELATED WORKS

Several studies have been conducted to explore various algorithms, data preprocessing techniques, feature selection methods, and evaluation metrics in the context of stock price prediction. This section presents a review of the related work conducted in this paper.

The ARIMA model is widely used to predict linear time series data. ARIMA is integrated from other models such as Automatic Regression (AR), Integrated (I) and Moving Average (MA). The accuracy of the ARIMA model to predict stock prices and the accuracy of over 85%, we see that the ARIMA model is also a good model [1].

S. Hochreiter and J. Schmidhuber [2] investigated the effectiveness of long short-term memory (LSTM) neural networks for stock price prediction. They proposed an LSTM-based model that incorporated both historical stock price data and technical indicators as input features. The study demonstrated that LSTM is a powerful architecture for sequential data analysis, addressing the limitations of traditional RNNs.

Sima Siami-Namini and Akbar Siami Namin (2018) [3] discuss the use of ARIMA and LSTM, two popular time series forecasting techniques, and highlights the advantages of each method, with ARIMA being useful for short-term forecasting of stationary data, and LSTM being particularly effective for capturing long-term dependencies and non-linear patterns in the data.

MSc. Nguyen Thi Hien, PhD. Dang Thi Minh Nguyet, Tran Thi Lan, Khuat Thi Vy, Tran Thi Linh of the University of Commerce in Vietnam chose the ARIMA(1,1,2), ARCH (3), GARCH (2,1) model. , GARCH (2,1)-M, TGARCH (2,1) to analyze the return rate fluctuations of the VN30F1M index with the data set during the period from August 2017 to September 2021. The study shows that the GARCH model is superior for making forecasts for the conditional variance of returns, this result is similar to previous results of Karmakar (2007), Goudarzi (2010), Sohail Chand, Shahid Kamal and Imran Ali (2012), Tran Sy Manh and Do Khac Huong (2013), Pham Chi Khoa (2017). The results of the GARCH (2,1) model show that the past rate of return determines the current rate of return and the fluctuation of the current rate of return but is affected by a decrease in the past, when there is past shocks, the return rate of the VN30F1M hybrid could change by about 13.33%. [4]

Xiangxi Jiang (2020) [5] developed deep learning models, including MLP, RNN (with LSTM extension), and GRU, to forecast Bitcoin prices. The MLP model with two layers of GRU achieved the best result, with an RMSE of 19.02.

Linear Regression: Bhawna Panwar et al used two models, Linear Regression and SVM (State Vector Machine) to predict and evaluate these two models. With the dataset being Amazon's stock price in October 2019. The results show that the accuracy of Linear Regression is 98.76%, higher than the accuracy of SVM is 94.3%. [6]

Yongqiong Zhu [7] applied an RNN model to forecast the stock prices of Apple. The training dataset covered Apple's stock prices (AAPL) spanning 10 years, from August 9, 2009, to August 12, 2020, with 65% allocated for training and the remaining 35% for testing. The model architecture included a two-layer RNN, with the initial layer featuring 50 units and the second layer containing 100 units. Employing 50 epochs, Adam optimization, and Mean Squared Error (MSE) as the loss function, the model achieved highly favorable outcomes. The predictive accuracy surpassed 95%, and the loss value was reported at 0.1%.

III. MATERIALS

A. DATASET

The historical stock price of three technology businesses: Apple Inc. (APPL), Alphabet Inc.(GOOG), and Microsoft Inc. (MSFT) from 01/03/2018 to 01/06/2024 will be applied. The data contains columns such as Date, Price, Open, High, Low, and Volume. As the goal is to forecast close prices, only data relating to column “Close" ($) will be processed.

B. DESCRIPTIVE STATISTICS

TABLE 1. Apple, Google, Microsoft's Descriptive Statistics

	Apple	Google	Microsoft
Count	1511	1511	1511
Mean	114.324	93.969	222.184
Std	51.237	32.441	86.011
Min	35.548	48.811	87.180
25%	55.619	60.826	137.575
50%	126.900	91.397	231.650
75%	156.060	124.735	289.130
Max	198.110	154.840	420.550
Skewness	-0.129	0.197	0.100
Kurtosis	-1.490	-1.451	-1.066

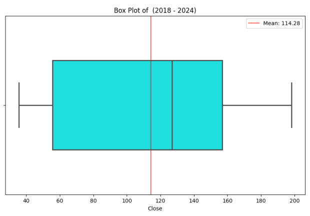

FIGURE 1. Apple stock price's boxplot

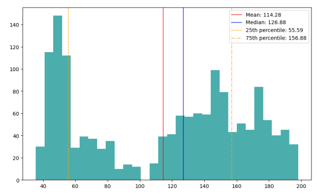

FIGURE 2. Apple stock price's histogram

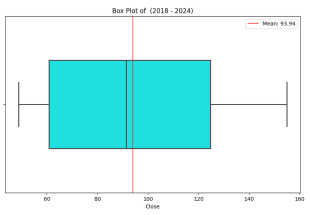

FIGURE 3. Google stock price's boxplot

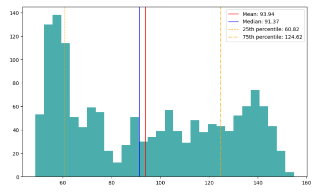

FIGURE 4. Google stock price's histogram

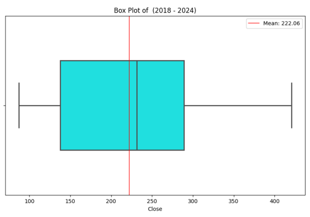

FIGURE 5. Microsoft stock price's boxplot

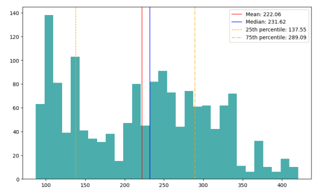

FIGURE 6. Microsoft stock price's histogram

IV. METHODOLOGY

A. LINEAR REGRESSION

Regression analysis is a tool for building mathematical and statistical models that characterize relationships between a dependent variable and one or more independent, or explanatory, variables, all of which are numerical. This statistical technique is used to find an equation that best predicts the y variable as a linear function of the x variables. A multiple linear regression model has the form:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \varepsilon$$

Where:

- Y is the dependent variable (Target Variable).
- $X_1, X_2, \ldots, X_k$ are the independent (explanatory) variables.
- β_0 is the intercept term.
- $\beta_1, ..., \beta_k$ are the regression coefficients for the independent variables.
- ε is the error term.

B. ARIMA

The Autoregressive Integrated Moving Average (ARIMA) is a model used in statistics and econometrics to measure events that happen over a period of time. It uses time-series data and statistical analysis to interpret the data and make future predictions. The ARIMA model aims to explain data by using its past values and uses linear regression to make predictions. For instance, the ARIMA model might seek to predict a stock's future prices based on its past performance or forecast a company's earnings based on past periods.

To understand ARIMA, it is beneficial to examine the name:

- The "AR" stands for autoregression, which refers to the model that shows a changing variable that regresses on its own prior or lagged values.
- The "I" stands for integrated, which means that the data is stationary. Stationary data refers to time-series data that's been made "stationary" by subtracting the observations from the previous values.
- The "MA" stands for moving average model, indicating that the forecast or outcome of the model depends linearly on the past values. Also, it means that the errors in forecasting are linear functions of past errors.

To implement ARIMA modeling effectively, it is necessary to clearly recognize the ARIMA parameters (p, d, q):

- The parameter p is the number of autoregressive terms or the number of "lag observations."
- The parameter d is known as the degree of differencing. it indicates the number of times the lagged indicators have been subtracted to make the data stationary.
- The parameter q is the number of forecast errors in the model.

General formula of ARIMA(p, d, q):git

$$\mathrm{Y}_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t$$

Where:

- Y_t is the value of the time series at time t.
- c is the constant term.
- $\phi_1, \phi_2, \ldots, \phi_p$ are the autoregressive coefficients for lagged values.
- $\theta_1, \theta_2, \ldots, \theta_q$ are the moving average coefficients for lagged errors.
- ϵ_t is the random error term at time t.

C. GARCH

GARCH stands for Generalized Autoregressive Conditional Heteroskedasticity, which is an extension of the ARCH model (Autoregressive Conditional Heteroskedasticity).

GARCH includes lag variance terms with lag residual errors from a mean process, and is the traditional econometric approach to volatility prediction of financial time series.

Mathematically, GARCH (p,q) can be represented as follows:

$$\sigma_t^2 = \omega + \sum_i^q \alpha_i \epsilon_{t-i}^2 + \sum_i^p \beta_i \sigma_{t-i}^2 \tag{1}$$

Where:

- σ_t^2 is the variance at time step t
- t and ϵ_{t-i}^2 is the model residuals at time step $t-1$.
- α is the coefficient for the squared model residuals at time step t-1
- β is the coefficient for the variance at time step t-1.
- ω is the long-term variance.

GARCH(1,1) only contains first-order lagged terms and the mathematical equation for it is:

$$\sigma_t^2 = \omega + \alpha \epsilon_{(t-1)}^2 + \beta \sigma_{(t-1)}^2 \tag{2}$$

Where

- α, β, and ω sum up to 1.
- ω is the long-term variance.

GARCH is generally regarded as an insightful improvement on naively assuming future volatility will be like the past, but also considered widely overrated as predictor by some experts in the field of volatility. GARCH models capture the essential characteristics of volatility: volatility tomorrow will be close to what it is today (**clustering**), and volatility in the long term will probably **mean revert** (meaning it'd be close to whatever the historical long-term average has been).

The basic GARCH model assumes positive and negative news have similar impact on volatility. However, in reality the market tends to "*take the stairs up and the elevator down*". In other words, the impact is usually asymmetric, and negative impacts tends to affect the volatility more than positive ones.

There's another member in the GARCH family that accounts for assymmetry of shocks reponses called **GJR-GARCH** (short for **Glosten-Jagannathan-Runkle GARCH**).

Additional inputs can be used to construct other models. In this case, by setting o to 1, which means the model would include one lag of an asymmetric shock which transforms a GARCH model into a GJR-GARCH model with variance dynamics.

D. RECURRENT NEURAL NETWORK

Recurrent Neural Network(RNN) is a type of **Neural Network** where the output from the previous step is fed as input to the current step. In traditional neural networks, all the inputs and outputs are independent of each other. Still, in cases when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus RNN came into existence, which solved this issue with the help of a **Hidden Layer**.

The main and most important feature of RNN is its Hidden state, which remembers some information about a sequence. The state is also referred to as Memory State since it remembers the previous input to the network. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

E. TIME-SERIES DENSE ENCODER

TiDE is similar to Transformers (implemented in TransformerModel), but attempts to provide better performance at lower computational cost by introducing multilayer perceptron (MLP)-based encoder-decoders without attention.

It is a multivariate time-series model that is able to use static covariates (e.g. brand of a product) and dynamic covariates (e.g. price of a product) which can be known or unknown for the forecast horizon.

The **Encoder** is responsible for mapping the past and the covariates of a time-series into a dense representation of features through two key steps:

- **Feature Projection** which reduces the dimensionality of the dynamic covariates for the whole look-back and the horizon.

$$\tilde{x}_{1:L+H} = \text{ResidualBlock}(x_t)$$

  Where:
  - x_t is the $r - dimensional$ dynamic covariates of time-series i at time t
- **Dense Encoder** which receives the output of the Feature Projection concatenated with static covariates and the past of the time-series and maps them into an embedding:

$$e = \text{Encoder}(y_{1:L}; \tilde{x}_{1:L+H}; a)$$

  Where:
  - $y_{1:L}$ is the look-back of the $i - th$ time-series
  - a is static attributes of a time-series.

The **Decoder** receives the embedding from the encoder and converts it into future predictions through two operations:

- **Dense Decoder** which maps the embedding into a vector per time-step in the horizon

$$g = \text{Decoder}(e)$$

- **Temporal Decoder** which combines the output of the Dense Decoder with the projected features of that time-step to produce the predictions.

$$\hat{y}_{L+t} = \text{TemporalDecoder}(d_t, \tilde{x}_{1:L+t})$$

The **Residual Connection** linearly maps the look-back to a vector with the size of the horizon which is added to the output of the Temporal Decoder to produce the final predictions.

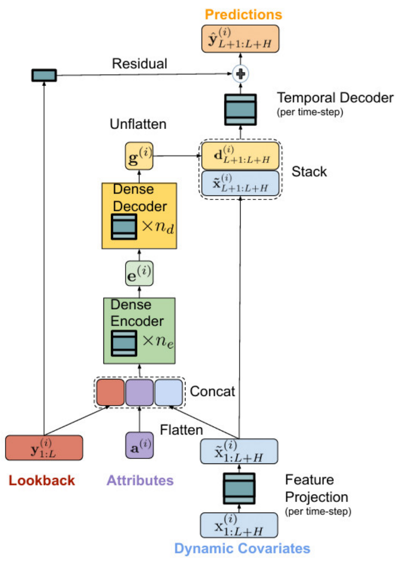

FIGURE 7. TiDE architecture

F. LONG SHORT-TERM MEMORY(LSTM)

Long Short-Term Memory(LSTM) networks are a type of recurrent neural network (RNNs) designed to overcome the limitations of traditional RNNs, particularly in learning long-term dependencies.

Traditional RNNs are good at handling sequential data by maintaining a hidden state that captures information from previous time steps. However, they suffer from two main problems:

- **Vanishing Gradient Problem**: Gradients can become very small during backpropagation, making it hard for the model to learn long-term dependencies.
- **Exploding Gradient Problem**: Gradients can become very large, causing instability during training.

The Structure of LSTM Cells:

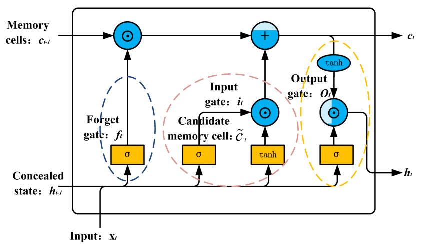

FIGURE 8. Long Short-Term Memory Cell [1]

LSTM networks address these issues through a more complex structure that can maintain long-term dependencies more effectively. Each LSTM cell contains several components:

- Cell State (Ct): This carries long-term information throughout the sequence. It's controlled by gates that can add or remove information.
- Forget Gate (ft): Decides what information to discard from the cell state. It's a sigmoid layer that outputs a number between 0 and 1 for each number in the cell state Ct-1.
- Input Gate (it): Decides which new information to add to the cell state. It consists of a sigmoid layer (input gate) and a tanh layer (input modulation gate) that create the candidate values (C t).
- Output Gate (ot): Decides what part of the cell state to output. It controls the hidden state (ht) for the next time step.

Key Advantages of LSTM:

- Handling Long-Term Dependencies: The use of gates allows LSTM networks to retain information over long periods, effectively addressing the vanishing gradient problem.
- Selective Memory: The forget, input, and output gates provide a mechanism for selective memory, enabling the model to decide what to remember and what to forget.
- Versatility: LSTM networks are widely used in various applications such as time series prediction, natural language processing, speech recognition, and more.

G. MULTILAYER PERCEPTRON (MLP)

Multilayer Perceptron (MLP) is a class of feedforward artificial neural network (ANN). It consists of at least three layers of nodes: an input layer, a hidden layer, and an output layer. Each node, or neuron, in one layer connects with a certain weight w to every node in the next layer. MLPs are capable of modeling complex non-linear relationships and are widely used for both classification and regression tasks.

The MLP model applied in time series prediction:

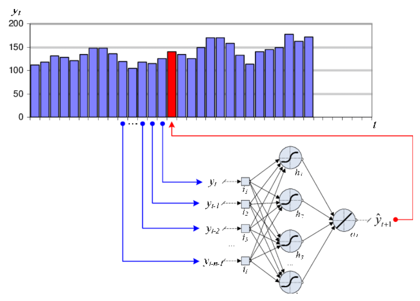

FIGURE 9. Multilayer Perception Achitechture for time series prediction [2]

Multilayer Perceptron Architecture:

- **Input Layer**: The input layer consists of neurons that receive the input features. In time series forecasting, each neuron corresponds to a feature in the input data, which are the historical values of the time series.
- **Hidden Layer**: Each hidden layer consists of multiple neurons, and each neuron in a hidden layer is connected to every neuron in the previous layer and the next layer. These layers apply non-linear transformations to the input data, capturing complex patterns in the time series.
- **Output Layer**: The output layer produces the predicted value for the next time step.

The network learns by adjusting the weights and biases of the connections between neurons through a process called backpropagation. This allows the model to capture nonlinear relationships and make predictions based on historical data.

H. GATED RECURRENT UNIT (GRU)

The basic idea behind GRU is to use gating mechanisms to selectively update the hidden state of the network at each time step. The gating mechanisms are used to control the flow of information in and out of the network. The GRU has two gating mechanisms, called the reset gate and the update gate.

The reset gate determines how much of the previous hidden state should be forgotten, while the update gate determines how much of the new input should be used to update the hidden state. The output of the GRU is calculated based on the updated hidden state. The equations used to calculate the reset gate, update gate, and hidden state of a GRU are as follows [7]:

- **Reset gate:**

$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r)$$

- **Update gate:**

$$z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z)$$

- **Candidate hidden state:**

$$\tilde{h}_t = \tanh(W_{xh}x_t + W_{hh}(r_t \odot h_{t-1}) + b_h)$$

- **Hidden state:**

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t$$

where:

- $W_{xr}, W_{xz}, W_{xh}, W_{hh}$ are learnable weight matrices,
- x_t is the input at time step t,
- h_{t-1} is the previous hidden state, and
- $\odot$ denotes the Hadamard (element-wise) product.

I. EVALUATION METHODS

Mean Percentage Absolute Error (MAPE): is a measure of prediction accuracy in a forecasting model. It is calculated as the average of the absolute percentage errors between the actual and predicted values..

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100$$

Where:

- n is the number of observations.
- y_i is the actual value.
- $\hat{y}_i$ is the predicted value.

Root Mean Squared Error (RMSE): is the square root of average value of squared error in a set of predicted values.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

where:

- y_i is the actual value,
- $\hat{y}_i$ is the predicted value,
- n is the number of observations.

Mean Absolute Error (MAE): is a metric used to evaluate the accuracy of a predictive model. It is calculated by taking the average of the absolute differences between the predicted values and the actual values.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Where:

- n is the number of observations in the dataset.
- y_i is the true value.
- $\hat{y}_i$ is the predicted value.

J. AAPL STOCK DATASET

AAPL Stock Dataset's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MAE
LR	7:3	34.11	19.09	31.23
	8:2	**19.28**	**9.22**	**16.06**
	9:1	20.51	9.69	17.08
GARCH	7:3	9.098	44.958	7.299
	8:2	5.841	45.883	4.939
	9:1	**5.444**	**43.078**	**4.452**
GRU	7:3	3.92	1.712	3.081
	8:2	3.92	1.744	2.95
	9:1	**3.34**	**1.44**	**2.546**
ARIMA	7:3	12.866	6.381	10.339
	8:2	35.172	18.345	33.385
	9:1	**11.885**	**5.419**	**9.629**
LSTM	7:3	3.042	1.447	2.411
	8:2	**2.277**	**0.969**	**1.717**
	9:1	3.667	1.687	3.090
MLP	7:3	4.654	2.320	3.837
	8:2	**2.994**	**1.311**	**2.340**
	9:1	3.265	1.453	2.614
RNN	7:3	3.405	1.590	2.726
	8:2	**3.039**	**1.322**	**2.409**
	9:1	3.184	1.379	2.405
TiDE	**7:3**	**20.969**	**10.386**	**18.170**
	8:2	28.144	14.441	26.238
	9:1	24.58	10.891	20.219

TABLE 2. AAPL Stock Dataset's Evaluation

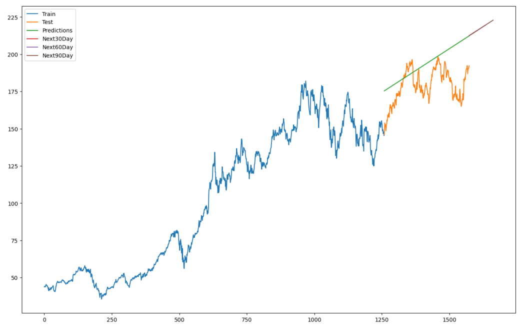

FIGURE 10. Linear model's result with 8:2 splitting proportion

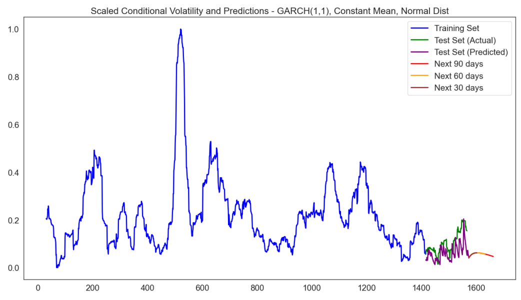

FIGURE 11. Garch model's result with 9:1 splitting proportion

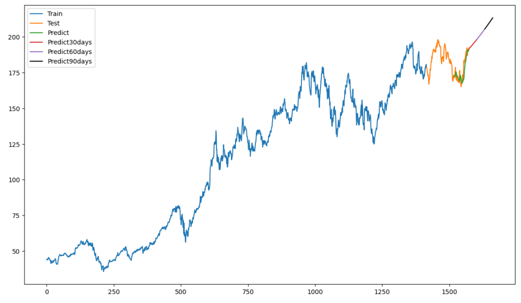

FIGURE 12. GRU model's result with 9:1 splitting proportion

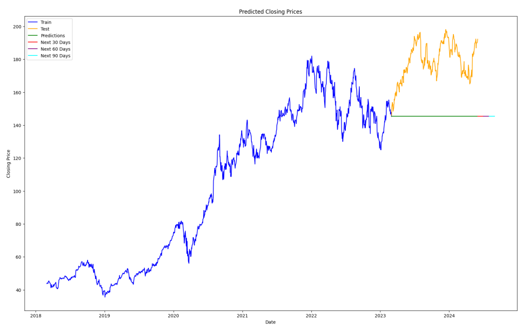

FIGURE 13. ARIMA model's result with 8:2 splitting proportion

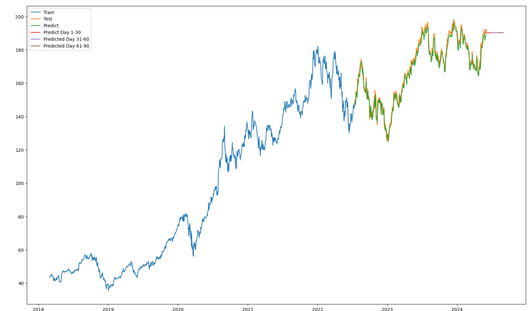

FIGURE 14. LSTM model's result with 7:3 splitting proportion

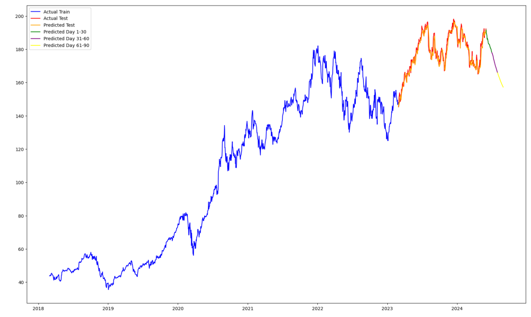

FIGURE 15. MLP model's result with 8:2 splitting proportion

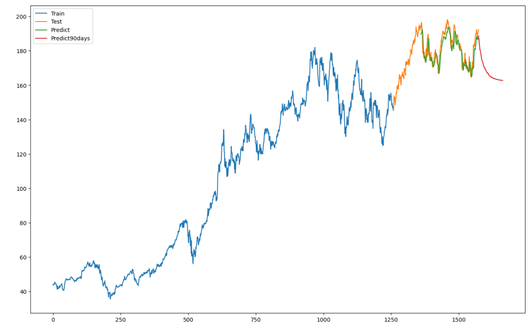

FIGURE 16. RNN model's result with 8:2 splitting proportion

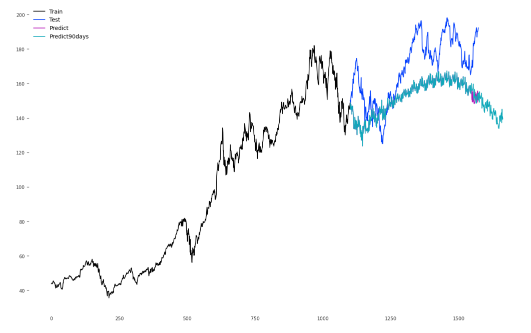

FIGURE 17. TiDE model's result with 7:3 splitting proportion

K. MSFT STOCK DATASET

MSFT Stock Dataset's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MAE
LR	7:3	61.02	18.65	51.03
	8:2	**34.57**	**7.96**	**28.71**
	9:1	47.52	10.83	43.71
GARCH	7:3	11.235	46.99	9.312
	8:2	7.808	43.18	6.495
	9:1	**5.569**	**41.434**	**4.839**
GRU	7:3	7.111	1.735	5.646
	8:2	**6.638**	**1.417**	**5.247**
	9:1	6.944	1.265	5.211
ARIMA	7:3	54.538	12.988	44.736
	8:2	117.390	29.029	107.4226
	9:1	**11.885**	**5.419**	**9.629**
LSTM	7:3	5.805	1.474	4.608
	8:2	**4.911**	**1.096**	**3.822**
	9:1	5.700	1.193	4.697
MLP	7:3	8.290	2.209	6.656
	8:2	6.075	1.379	4.855
	9:1	**5.728**	**1.120**	**4.408**
RNN	**7:3**	**6.292**	**1.521**	**5.107**
	8:2	13.278	3.176	12.093
	9:1	15.062	3.294	13.804
TiDE	**7:3**	**74.635**	**15.772**	**54.885**
	8:2	75.311	14.130	55.084
	9:1	127.371	29.642	119.175

TABLE 3. MSFT Stock Dataset's Evaluation

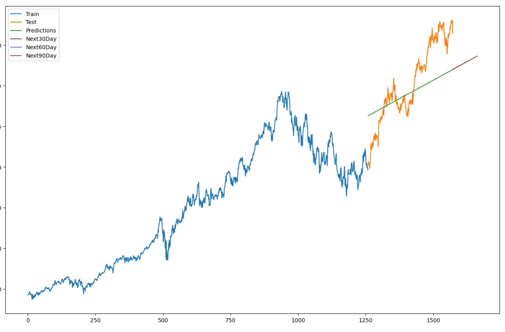

FIGURE 18. Linear model's result with 8:2 splitting proportion

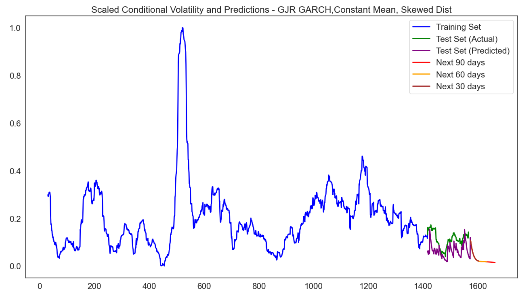

FIGURE 19. GARCH model's result with 9:1 splitting proportion

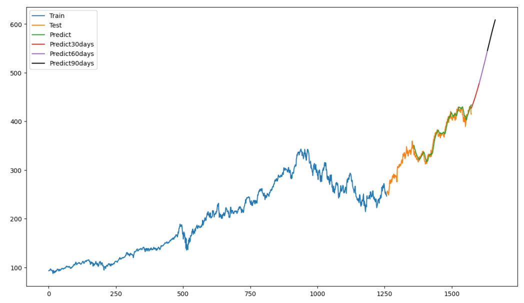

FIGURE 20. GRU model's result with 8:2 splitting proportion

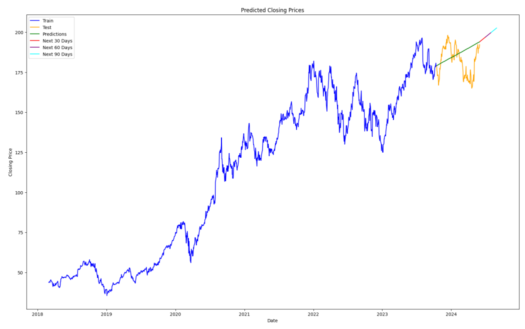

FIGURE 21. ARIMA model's result with 9:1 splitting proportion

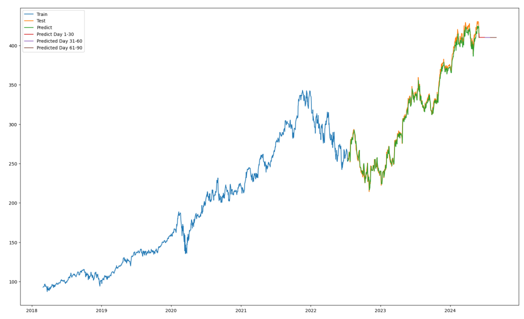

FIGURE 22. LSTM model's result with 7:3 splitting proportion

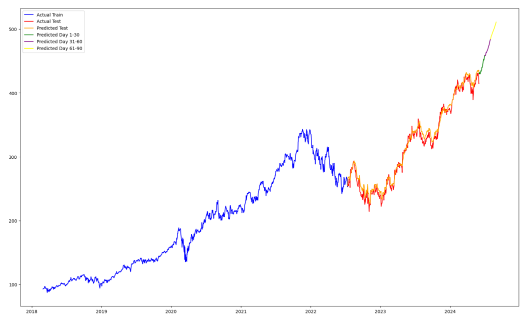

FIGURE 23. MLP model's result with 7:3 splitting proportion

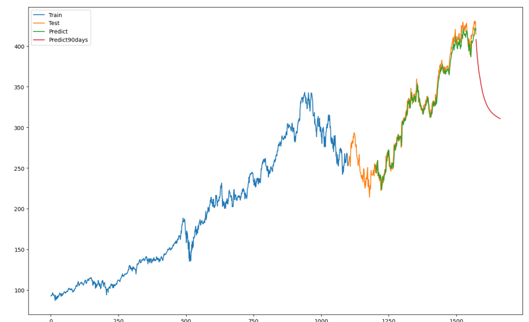

FIGURE 24. RNN model's result with 7:3 splitting proportion

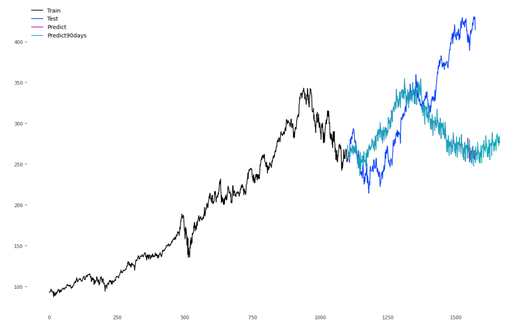

FIGURE 25. TiDE model's result with 7:3 splitting proportion

L. GOOG STOCK DATASET

GOOG Stock Dataset's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MAE
LR	7:3	36.39	30.33	33.86
	8:2	15.25	9.82	12.01
	9:1	**13.53**	**6.20**	**9.76**
GARCH	7:3	16.158	46.673	13.691
	8:2	12.666	40.463	10.831
	9:1	**13.320**	**38.568**	**10.831**
GRU	7:3	3.463	2.131	2.630
	8:2	3.410	1.811	2.564
	9:1	**3.392**	**1.608**	**2.566**
ARIMA	7:3	25.446	16.146	20.922
	8:2	47.351	30.924	43.353
	9:1	**12.202**	**5.727**	**8.884**
LSTM	**7:3**	**2.678**	**1.642**	**1.987**
	8:2	2.763	1.500	1.980
	9:1	2.847	1.257	1.849
MLP	7:3	4.720	3.294	3.869
	8:2	5.006	3.056	4.226
	9:1	**3.551**	**1.855**	**2.742**
RNN	7:3	2.776	1.668	2.080
	8:2	**2.886**	**1.450**	**2.103**
	9:1	3.678	1.744	2.836
TiDE	**7:3**	**16.034**	**9.718**	**11.815**
	8:2	40.055	28.255	38.367
	9:1	23.957	10.385	16.536

TABLE 4. GOOG Stock Dataset's Evaluation

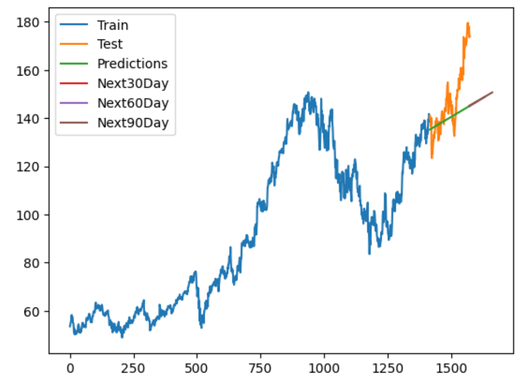

FIGURE 26. Linear model's result with 9:1 splitting proportion

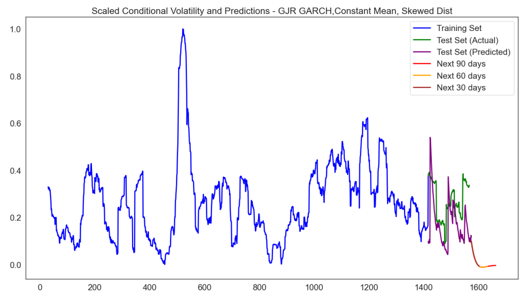

FIGURE 27. GARCH model's result with 9:1 splitting proportion

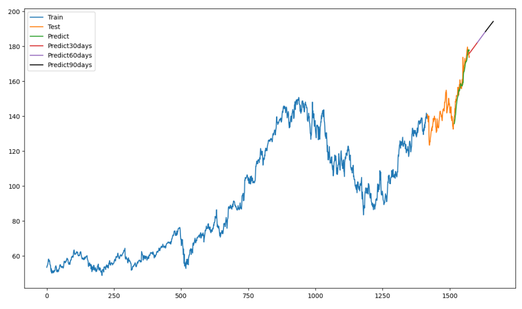

FIGURE 28. GRU model's result with 9:1 splitting proportion

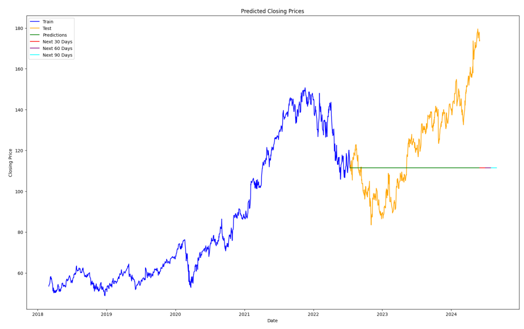

FIGURE 29. ARIMA model's result with 7:3 splitting proportion

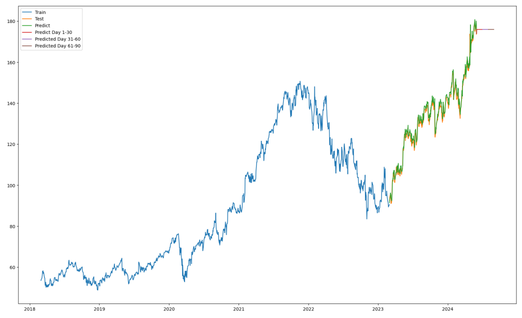

FIGURE 30. LSTM model's result with 8:2 splitting proportion

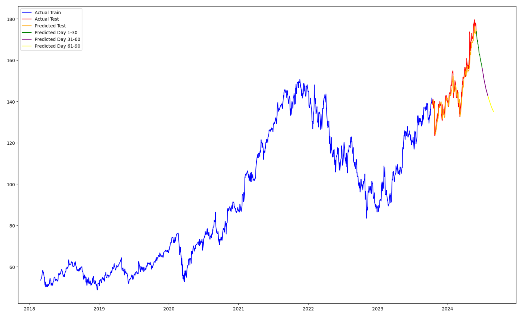

FIGURE 31. MLP model's result with 9:1 splitting proportion

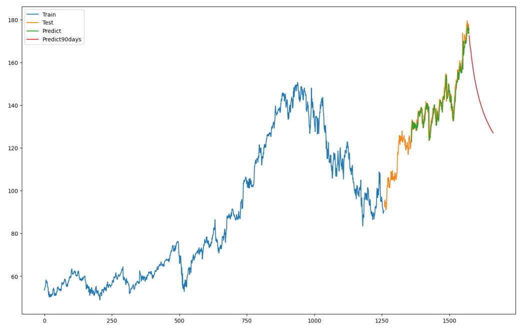

FIGURE 32. RNN model's result with 8:2 splitting proportion

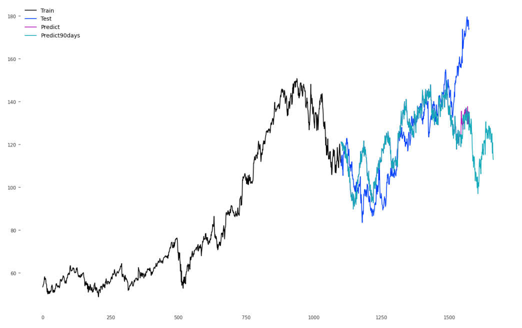

FIGURE 33. TiDE model's result with 7:3 splitting proportion

V. CONCLUSION

A. SUMMARY

In the achievement of forecasting stock prices, the exploration of diverse methodologies, ranging from traditional statistical models to advanced machine learning algorithms, has been aimed. Among the performed models, Linear Regression, Autoregressive Integrated Moving Average (ARIMA), Long Short Term Memory (LSTM), Regression Neural Networks (RNN), Gated Regression Units (GRU), Generalized Autoregressive Conditional Heteroskedasticity (GARCH), Time-series Dense Encoder Model (TiDE), and Multi-layer Perceptron (MLP), it becomes evident that Regression Neural Networks (RNN), Long Short Term Memory (LSTM), and Multi-layer Perceptron (MLP) emerge as the most promising and effective models for predicting stock prices.

Regression Neural Networks (RNN), Long Short Term Memory (LSTM), and Multi-layer Perceptron (MLP) each offer distinct advantages in the realm of time-series stock price forecasting. RNNs are particularly adept at capturing and modeling sequential dependencies within time-series data, making them well-suited for financial markets where past prices influence future trends. LSTMs, an advanced type of RNN, excel in managing long-term dependencies and mitigating issues related to vanishing gradients, thereby providing more accurate and stable predictions over longer periods. MLPs, while simpler, are highly effective in capturing nonlinear relationships within the data, offering flexibility and robust performance across various market conditions. Collectively, these models leverage their unique strengths to deliver superior forecasting accuracy, adaptability to volatile market dynamics, and valuable insights for investors and analysts.

As evidenced by the evaluation metrics, including RMSE, MAPE, and MAE, the RNN, LSTM, and MLP models consistently demonstrate superior performance across various aspects of forecasting accuracy. Their adaptability to handle the inherent uncertainties of stock markets positions them as formidable tools for investors and analysts seeking reliable predictions.

B. FUTURE CONSIDERATIONS

In our future research, it is crucial to prioritize further optimization of the previously mentioned models. This optimization effort should specifically focus on:

• Enhancing the accuracy of the model. While the above algorithms have demonstrated promising results in predicting stock prices, there is a need to further improve the model's accuracy to ensure more precise forecasting outcomes.

• Exploring alternative machine learning algorithms or ensemble techniques. Ensemble techniques, such as combining multiple models or using various ensemble learning methods, can also improve the robustness and accuracy of the forecasts.

• Researching new forecasting models. The field of forecasting continuously evolves, with new algorithms and models being researched and developed. It is crucial to stay updated with these approaches and explore new forecasting models that offer improved accuracy and performance.

By continuously exploring and incorporating new features, data sources, and modeling techniques, we can strive for ongoing optimization of the forecasting models and enhance their ability to predict stock prices with greater precision and reliability.

ACKNOWLEDGMENT

First and foremost, we would like to express our sincere gratitude to **Assoc. Prof. Dr. Nguyen Dinh Thuan** and **Mr. Nguyen Minh Nhut** for their exceptional guidance, expertise, and invaluable feedback throughout the research process. Their mentorship and unwavering support have been instrumental in shaping the direction and quality of this study. Their profound knowledge, critical insights, and attention to detail have significantly contributed to the success of this research.

This research would not have been possible without the support and contributions of our mentors. We would like to extend our heartfelt thanks to everyone involved for their invaluable assistance, encouragement, and belief in our research. Thank you all for your invaluable assistance and encouragement.

REFERENCES

[1] L. Li, Y. Yabin, Y. Ou, Q. Li, Y. Zhou, and D. Chen, "Research on machine learning algorithms and feature extraction for time series." In *Proceedings of the International Conference on Machine Learning and Cybernetics*, 2018, pp. 123-128.

[2] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[3] S. S. Namin and A. S. Namin, “Forecasting Economic Variables with Long Short Term Memory (LSTM) Neural Networks," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2018, pp. 1241-1246, doi: 10.1109/ICMLA.2018.00203.

[4] N. T. Hien, "Application of ARCH-GARCH model to analyze the fluctuation of VNIndex," *Journal of Science and Technology*, vol. 126, no. 1MEIS, pp. 18-25, 2019. [Online]. Available: https://tckhtm.tmu.edu.vn/upload/news/files/126-b2.pdf

[5] X. Jiang, “Bitcoin Price Prediction Based on Deep Learning Methods,” *Journal of Mathematical Finance*, vol. 10, no. 2, pp. 252-263, 2020. [Online]. Available: https://www.scirp.org/pdf/jmf_2020021013592656.pdf

[6] D. Bhuriya, G. Kaushal, A. Sharma, and U. Singh, "Stock market prediction using a linear regression," in *2017 International Conference of Electronics, Communication and Aerospace Technology (ICECA)*, vol. 2, 2017, pp. 510-513. doi: 10.1109/ICECA.2017.8203539.

[7] Y. Zhu, "Stock price prediction using the RNN model," in *2020 International Conference on Applied Physics and Computing (ICAPC 2020)*, 2020. doi: 10.1088/1742-6596/1650/3/032103.

[8] Long Short Term Memory Cell. *Designs*, vol. 14, no. 11, p. 1696, Nov. 2023. [Online]. Available: https://www.mdpi.com/2073-4433/14/11/1696.

[9] A. Das, S. Ghosh, and S. Sen, "Long-term Forecasting with TIDE: Time-series Dense Encoder," arXiv preprint arXiv:2304.08424, 2023.

[10] R. Guedes, "TIDE: the ’embarrassingly’ simple MLP that beats Transformers," *Medium*, Mar. 01, 2024. [Online]. Available: https://towardsdatascience.com/tide-the-embarrassingly-simple-mlp-that-beats-transformers-7db77d588079

[11] "Time Series Forecasting using Recurrent Neural Networks (RNN) in TensorFlow," *GeeksforGeeks*, Jul. 31, 2023. [Online]. Available: https://www.geeksforgeeks.org/time-series-forecasting-using-recurrent-neural-networks-rnn-in-tensorflow/