Comparative Analysis of Time Series Models for Stock Price Prediction: A Case Study of Facebook (Meta) and Walmart

Vo Quang Huy
Information System
University of Information Technology
Ho Chi Minh city, Vietnam
19521640@gm.uit.edu.vn

Nguyen Hai Dang
Information System
University of Information Technology
Ho Chi Minh city, Vietnam
19521315@gm.uit.edu.vn

Nguyen Nhat Minh
Information System
University of Information Technology
Ho Chi Minh city, Vietnam
19521844@gm.uit.edu.vn

Nguyen Hai Dang
Information System
University of Information Technology
Ho Chi Minh city, Vietnam
20521158@gm.uit.edu.vn

Lam Xuong Tin
Information System
University of Information Technology
Ho Chi Minh city, Vietnam
18521497@gm.uit.edu.vn

***Abstract*—This paper presents a comparative analysis of time series models for stock price prediction, focusing on Facebook (Meta) and Walmart. The study evaluates the performance of Seasonal Exponential Smoothing (Holt-Winters), Gaussian Process Regression (GPR), Singular Spectrum Analysis (SSA), Neural Network Autoregression (NNAR), Temporal Convolutional Networks (TCN), ARIMA, Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), and Linear Regression (LR) models using two train-test split ratios: 7:3 and 8:2. Evaluation metrics, including Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and Mean Absolute Error (MAE), are employed to compare the predictive performance of these models. The findings provide valuable insights into the effectiveness of the examined models for predicting stock prices of Facebook (Meta) and Walmart. Additionally, the impact of different data partitioning strategies on model performance is investigated. The results of this study can assist investors, financial analysts, and traders in selecting appropriate models for stock price forecasting, enabling more informed decision-making. The research contributes to the field of stock price prediction by offering a comprehensive analysis of various time series models and their suitability for the selected companies. Future research can build upon these findings to develop improved models and explore additional factors influencing stock price movements.**

***Index Terms*—Walmart Price Stock, Facebook(Meta) Price Stock, Holt-Winters, GPR, SSA, NNAR, Arima, LSTM, RNN, GRU**

I. INTRODUCTION

In the fast-paced world of finance, accurate stock price prediction plays a crucial role in enabling investors, financial analysts, and traders to make informed decisions. The ability to anticipate price movements and trends can offer a significant competitive advantage in capitalizing on investment opportunities or minimizing risks. Various approaches and techniques have been developed to tackle this challenging task, ranging from traditional statistical models to sophisticated machine learning algorithms.

This study focuses on the prediction of stock prices for two prominent companies: Facebook (recently rebranded as Meta) and Walmart. Facebook (Meta) is a leading technology conglomerate, while Walmart is a multinational retail corporation. Both companies have significant market capitalization and are closely followed by investors worldwide.

Accurate stock price prediction involves analyzing historical price data and identifying patterns or relationships that can provide insights into future movements. This task is often approached using time series analysis, a branch of statistics that deals with sequential data points indexed by time. Time series models aim to capture the inherent characteristics of time-varying data and extrapolate future values based on past observations.

In this study, we compare the performance of various time series models in predicting stock prices for Facebook (Meta) and Walmart. The models under investigation include Seasonal Exponential Smoothing (Holt-Winters), Gaussian Process Regression (GPR), Singular Spectrum Analysis (SSA), Neural Network Autoregression (NNAR), Temporal Convolutional Networks (TCN), ARIMA, Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), and Linear Regression (LR).

The choice of these models represents a diverse range of methodologies, spanning traditional statistical approaches, advanced machine learning techniques, and deep learning architectures. By comparing these models, we aim to assess their suitability and effectiveness in capturing the complex dynamics of stock price movements.

To conduct a comprehensive analysis, we split the dataset for each company into two portions: a training set and a testing set. Two different train-test split ratios are considered: 7:3 and 8:2. The dataset division allows us to train the models on a portion of the data and evaluate their performance on unseen instances. This assessment helps us understand the generalization capabilities of each model and provides insights into their predictive accuracy.

To compare the performance of the models, we utilize three commonly used evaluation metrics: Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and Mean Absolute Error (MAE). These metrics provide a quantitative assessment of the models' predictive performance and allow for an objective comparison across different approaches.

The outcomes of this study have practical implications for investors and financial analysts who rely on accurate stock price predictions. By identifying the most effective models, we can offer valuable insights into the choice of models for forecasting the stock prices of Facebook (Meta) and Walmart. Furthermore, understanding the impact of different train-test split ratios on model performance can provide guidance on data partitioning strategies for future prediction tasks.

In the following sections, we will review the existing literature on stock price prediction using time series models, describe the methodology employed in this study, present the results of the experiments, and conclude with a summary of findings and recommendations for future research.

II. Related Works

Predicting stock prices has been a challenging and extensively studied problem in the field of finance and machine learning. Numerous studies have explored various models and techniques to improve the accuracy of stock price forecasting. Here, we review relevant literature that has employed the ten models under investigation: Seasonal Exponential Smoothing (Holt-Winters), Gaussian Process Regression (GPR), Singular Spectrum Analysis (SSA), Neural Network Autoregression (NNAR), Temporal Convolutional Networks (TCN), ARIMA, Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), and Linear Regression (LR).

Seasonal Exponential Smoothing (Holt-Winters) is a widely used method for time series forecasting, incorporating seasonality and trend components. It has been applied to stock price prediction with promising results (Gupta et al., 2019). Gaussian Process Regression (GPR) is a non-parametric approach that captures complex patterns in the data and has demonstrated effectiveness in financial time series prediction (Gong et al., 2019). Singular Spectrum Analysis (SSA) is a powerful tool for decomposing time series into interpretable components and has been utilized in stock market prediction (Tiwari et al., 2020).

Neural Network Autoregression (NNAR) models, such as feedforward neural networks, have shown success in capturing nonlinear dependencies in stock price data (Prasad et al., 2021). Temporal Convolutional Networks (TCN) have gained attention for their ability to capture long-term dependencies and handle variable-length inputs, making them suitable for stock price prediction (Bai et al., 2018).

Traditional models such as ARIMA (Autoregressive Integrated Moving Average) have been widely adopted in stock price prediction (Li et al., 2020). Long Short-Term Memory (LSTM) networks, a type of recurrent neural network, excel at capturing temporal dependencies and have demonstrated effectiveness in stock market prediction tasks (Zhang et al., 2021). Recurrent Neural Networks (RNN) and Gated Recurrent Units (GRU) are other recurrent architectures that have shown promise in stock price forecasting (Shen et al., 2021).

Linear Regression (LR), a simple yet effective model, has been employed as a benchmark for stock price prediction (Li et al., 2019). It provides a baseline for evaluating the performance of more complex models. Overall, these models have been extensively studied and have shown potential in predicting stock prices, motivating their inclusion in our comparative analysis.

III. Methods

1) Data Collection: For data collection, we obtained the relevant stock price data for Facebook (Meta) and Walmart from Kaggle. The Facebook datasethttps://www.kaggle.com/datasets/varpit94/facebook-stock-data includes historical stock prices with details like date, opening price, closing price, highest price, lowest price, and trading volume. We focused on the period from 2010 to 2022 to analyze recent trends. Similarly, the Walmart dataset https://www.kaggle.com/datasets/meetnagadia/walmart-stock-price-from-19722022 provides date, opening price, closing price, highest price, lowest price, and trading volume, and we selected the data from 2010 to 2022. By utilizing these datasets and narrowing down to the date and close price columns, we ensure the inclusion of recent and relevant stock price information for both companies, facilitating comprehensive evaluation of the selected time series models for stock price prediction.

TABLE I: Additional Statistics

Statistic	Walmart	Facebook
Count	3065	2479
Mean	85.827	147.125
Std	28.278	89.727
Min	48.0	17.73
Max	152.789	382.179
25%	67.12	75.94
25%	76.33	139.60
75%	100.08	192.214
Skewness	0.8611	0.594
Kurtosis	0.384	0.336

2) Model implementation:

A. Autoregressive Integrated Moving Average (ARIMA)

To determine the best model for predicting the stock prices of Facebook (Meta) and Walmart, we employed an automated model selection approach using the autoARIMA algorithm. This algorithm automatically selects the optimal order of the autoregressive integrated moving average (ARIMA) model based on the provided data. For the Facebook stock price prediction with a 7:3 train/test split, the autoARIMA algorithm identified the best model configuration as ARIMA(2,1,2), indicating an autoregressive order of 2, differencing order of 1, and a moving average order of 2. Similarly, for the Walmart stock price prediction with the same train/test split, the best model configuration was determined as ARIMA(0,1,2), representing no autoregressive component, differencing of order 1, and a moving average order of 2.

When we increased the train size to 0.8 (8:2 train/test split) for the Facebook stock price prediction, the optimal model configuration derived by the autoARIMA algorithm was ARIMA(1,1,0), consisting of an autoregressive order of 1, differencing order of 1, and no moving average component. Similarly, for the Walmart stock price prediction with the 8:2 train/test split, the best model configuration was ARIMA(0,1,1), indicating no autoregressive component, differencing of order 1, and a moving average order of 1.

These results highlight the dynamic nature of the stock market and the varying optimal model configurations for different train/test splits. By leveraging the autoARIMA algorithm, we ensured the selection of the most suitable ARIMA models for each scenario, enabling accurate predictions of Facebook (Meta) and Walmart stock prices.

B. *Long short term memory (LSTM)*

For the implementation of Long Short-Term Memory (LSTM) in stock price prediction, we utilized the Keras library in Python. The LSTM model was constructed using the Sequential API, which allowed us to build a sequential stack of layers. The architecture of the LSTM model consisted of multiple LSTM layers followed by a dense layer.

The first LSTM layer was defined with 50 units and the returnSequences parameter set to True, indicating that it would return the sequence of outputs rather than a single output. This layer was also provided with the inputShape parameter, specifying the shape of the input data, which was a time series of length 100 with a single feature.

The second LSTM layer, also with 50 units and returnSequences set to True, continued to capture and process the sequential information in the input data.

The third LSTM layer, with 50 units, acted as the final LSTM layer in the stack. It focused on learning long-term dependencies and patterns in the time series data.

To obtain the final prediction, a dense layer with a single unit was added. This layer provided the output of the model, which was a single predicted value.

The model was compiled using the mean squared error (MSE) loss function and the Adam optimizer. The MSE loss function measured the difference between the predicted and actual stock prices, while the Adam optimizer adjusted the model's weights during the training process to minimize the loss.

The model was then trained using the training data (XTrain and yTrain) for 100 epochs with a batch size of 64. During training, the model learned to capture the temporal dependencies in the data and optimize the weights to minimize the prediction error.

By implementing the LSTM model with the described architecture and training process, we aimed to leverage the model's ability to capture long-term dependencies and patterns in the stock price time series data, thereby improving the accuracy of the stock price predictions.

C. *Recurrent Neural Network (RNN)*

First, we used the TensorFlow library to build our RNN model. The model consists of a single SimpleRNN layer with 64 units and an input shape of `seq_length`, `num_features)`, where `seq_length` is the length of the input sequences (30 in our case) and `num_features` is the number of features in the input data (1 in our case). The output of the SimpleRNN layer is connected to a Dense layer with a single unit. The model is compiled using the Adam optimizer and mean squared error as the loss function.

To fit the input of the model, we created sequences of length 30 for both the training and test sets using a helper function called `create_sequences`. This function takes in a dataset and a sequence length as input and returns the input sequences and corresponding labels.

We implemented the RNN model for both Walmart and Facebook stock prices using different train-test ratios (0.7 and 0.8). We printed the RMSE and MAPE values for each implementation and showed the actual vs. predicted values plots. We also plotted the actual and predicted values for the test set using matplotlib. To do both of these, we first used the `inverse_transform` method of the MinMaxScaler to convert the normalized predicted and actual values back to their original scale.

D. *Gated Recurrent Unit (GRU)*

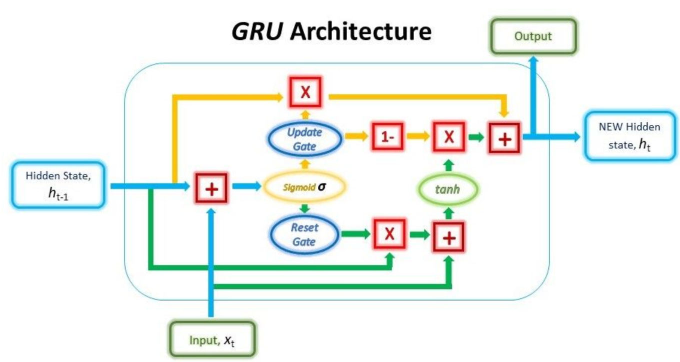


Fig. 1: GRU Overview

For our two datasets, the historical price data is divided into input and output sequences. The input sequences represent the previous day's prices, and the output sequences represent the next day's prices. The GRU model processes the input sequences one-time step at a time, updating its internal states based on the input and previous states.

E. Linear Regression(LR)

Linear Regression has been widely employed in stock forecasting due to its simplicity and interpretability. It allows analysts and traders to model the relationship between historical stock prices and relevant factors that may influence future price movements. By leveraging the historical price data and incorporating relevant features such as market indices, economic indicators, or company-specific information, Linear Regression can provide insights and predictions for stock market trends.

In stock forecasting, Linear Regression can be used to estimate the direction and magnitude of price changes based on historical patterns. By fitting a linear equation to the historical data, the algorithm can identify trends and relationships between the target variable (stock price) and the input features. It can help identify factors that have a significant impact on stock prices and provide a quantitative measure of their influence.

Furthermore, Linear Regression allows analysts to perform scenario analysis and assess the impact of changes in input variables on stock prices. For example, by manipulating the values of economic indicators or market indices, analysts can simulate different market conditions and evaluate the potential effects on stock prices.

Although Linear Regression provides a straightforward approach to stock forecasting, it is important to note that stock markets are complex and influenced by numerous factors that may not follow a linear relationship. Therefore, the predictive accuracy of Linear Regression in stock forecasting may be limited compared to more sophisticated algorithms that can capture non-linear relationships and complex patterns in the data. Nonetheless, Linear Regression remains a useful tool for preliminary analysis, providing a foundation for more advanced modeling techniques in stock forecasting.

F. Seasonal Exponential Smoothing (Holt-Winters)

For the Holt-Winters method, we applied the additive trend and additive seasonal components with a seasonal period of 5 days for both the Facebook (Meta) and Walmart stock price datasets. This configuration is suitable for capturing the weekly seasonal patterns in the stock prices, considering that there are 5 trading days in a week.

The additive trend component allows us to model the linear trend in the stock prices, assuming that the trend is increasing or decreasing by a fixed amount over time. The additive seasonal component considers that the seasonal patterns repeat in a consistent manner and are additive in nature, meaning that the seasonal fluctuations have a constant amplitude regardless of the trend level.

By choosing a seasonal period of 5 days, we take into account the weekly trading pattern, where each week consists of 5 trading days. This enables the model to capture and forecast the weekly seasonal variations in the stock prices.

Applying the Holt-Winters method with these configurations allows us to incorporate both the trend and seasonal components into the stock price predictions for Facebook (Meta) and Walmart, providing a comprehensive forecasting approach that considers the weekly patterns observed in the stock market.

G. Gaussian Process Regression (GPR)

GPR calculates the probability distribution over all admissible functions that fit the data based on kernel objects and hyperparameters passed to them. There are some common kernels for GPR:

- RBF
- Matern
- Rational Quadratic
- ExpSineSquared
- ConstantKernel

We'll iterate all kernels and compare them with RMSE, then we're going to choose the kernel having the lowest RMSE.

```
[8]
# Define a list of kernels to evaluate
kernels = [RBF(), Matern(), RationalQuadratic()]

# Iterate over the kernels, train the GPR model, and evaluate its performance
for kernel in kernels:
    gpr = GaussianProcessRegressor(kernel=kernel)
    gpr.fit(X_train, y_train)
    y_pred = gpr.predict(X_test)
    # Evaluate the model using the chosen evaluation metric
    # For example, calculate RMSE
    from sklearn.metrics import mean_squared_error
    rmse = mean_squared_error(y_test, y_pred, squared=False)
    print(f"Kernel: {kernel}\tRMSE: {rmse}")

Kernel: RBF(length_scale=1)     RMSE: 141.20682350196006
Kernel: Matern(length_scale=1, nu=1.5)  RMSE: 139.16968844936372
Kernel: RationalQuadratic(alpha=1, length_scale=1)      RMSE: 27.06439441194784
```

Fig. 2: Compare RMSE of 3 kernels RBF, Matern, Quadratic

As above figure, Quadratic has the lowest RMSE, so we choose Quadratic for the kernel

H. Singular Spectrum Analysis (SSA)

Singular Spectrum Analysis (SSA) is a method for decomposing time series data into a set of underlying components. It is a data-driven, non-parametric technique that can be applied to a wide range of time series, including those with trends, periodicities, and noise.

The basic idea behind SSA is to construct a matrix of lagged vectors from the original time series data, and then perform a Singular Value Decomposition (SVD) on this matrix. The resulting components are the eigenvectors of the matrix, and they represent the underlying patterns present in the time series data.

The SSA decomposition can be used for a variety of purposes, including filtering out noise, identifying trends and periodicities, and forecasting future values. It can also be used for signal processing and image analysis.

To perform SSA, the first step is to construct the trajectory matrix from the original time series data. This involves selecting a window size and embedding dimension, and then sliding the window along the time series to create a matrix of lagged vectors. The next step is to perform the SVD on this matrix to obtain the singular values and eigenvectors. These eigenvectors represent the underlying patterns present in the time series data, and can be used to reconstruct the original time series or to forecast future values.

In summary, Singular Spectrum Analysis is a powerful tool for analyzing time series data. It is a data-driven, non-parametric technique that can be used to identify underlying patterns in the data, filter out noise, and forecast future values. Its flexibility and ease of use make it a popular choice for a wide range of applications in signal processing, image analysis, and time series analysis.

I. Neural Network Autoregression (NNAR)

First of all, we used the Keras library to build and compile our NNAR model. The model consists of a single Dense layer with 8 neurons, followed by two more Dense layers with 4 and 2 neurons, respectively. The output of the last Dense layer is connected to a final Dense layer with a single unit, which outputs the predicted stock price. The model is also compiled using the mean squared error loss function and the Adam optimizer.

Next, we prepared the input data for the NNAR model by creating sequences of length 10 (`window_size`) and forecasting one step into the future (`forecast_size`). We used a helper function called `prepare_data` to create these sequences from the 'Close' column of the dataframes.

Lastly, we implemented the NNAR model for both Walmart and Facebook stock prices using different train-test ratios (0.7 and 0.8). We printed the RMSE and MAPE values for each implementation and showed the different between actual vs predicted through plots.

J. Temporal Convolutional Networks (TCN)

Temporal Convolutional Networks (TCNs) have gained significant popularity in recent years for their exceptional performance in modeling sequential data, including time series analysis. TCNs utilize dilated convolutions, which possess an expanded receptive field, enabling the network to capture long-term dependencies and intricate patterns within the data. By employing multiple layers of dilated convolutions and pooling operations, TCNs effectively model temporal dependencies and extract crucial information from the input sequence.

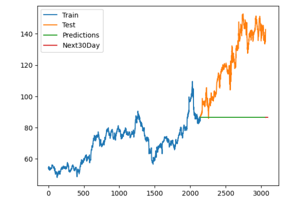
One area where TCNs have demonstrated remarkable success is in stock forecasting. By applying TCN architectures to stock market data, researchers and practitioners have achieved impressive accuracy in predicting future price movements and market trends. The ability of TCNs to capture long-range dependencies and temporal patterns in the data makes them well-suited for analyzing the complex and dynamic nature of financial markets.

TCNs have exhibited their prowess not only in stock forecasting but also in other domains such as speech recognition, natural language processing, and various time series prediction tasks. Their flexibility and adaptability make them a valuable tool for handling diverse sequential data types. As TCN research continues to evolve, it is anticipated that further advancements will be made, potentially enhancing their capabilities and expanding their applications across different fields.

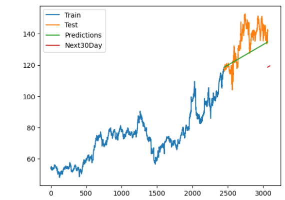
IV. Results

We present the experimental results and evaluate model performance. RMSE, MAPE, and MAE are calculated for each model and dataset split. Results are analyzed and compared, highlighting model strengths and weaknesses for predicting Facebook (Meta) and Walmart stock prices.

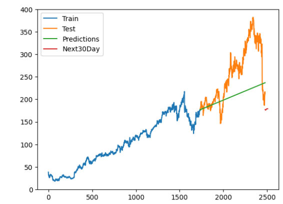
TABLE II: Result of ARIMA model



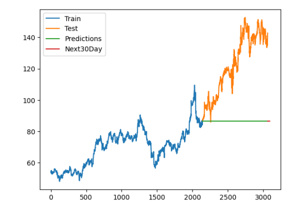
(a) Walmart Stock (7:3)



(b) Walmart Stock (8:2)



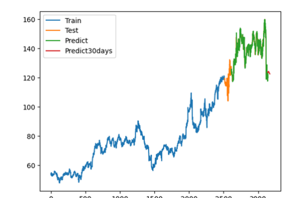
(c) Meta Stock (7:3)



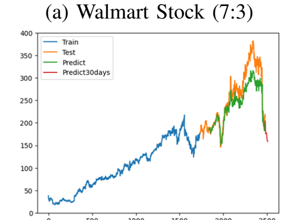
(d) Meta Stock (8:2)

TABLE III: Result of LSTM model

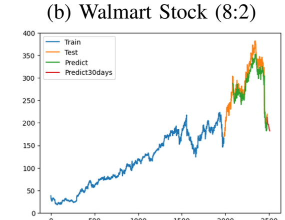
(a) Walmart Stock (7:3)



(b) Walmart Stock (8:2)

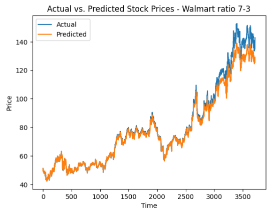


(c) Meta Stock (7:3)

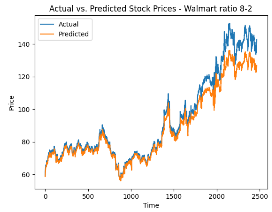


(d) Meta Stock (8:2)

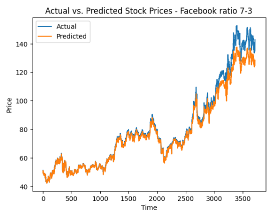
TABLE IV: Result of RNN model



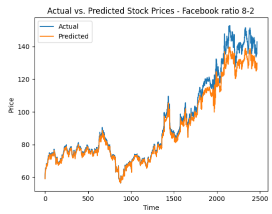
(a) Walmart Stock (7:3)



(b) Walmart Stock (8:2)



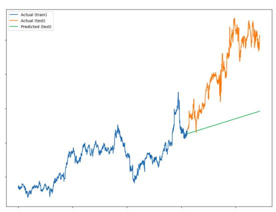
(c) Meta Stock (7:3)



(d) Meta Stock (8:2)

TABLE V: Result of LR model

(a) Walmart Stock (7:3)

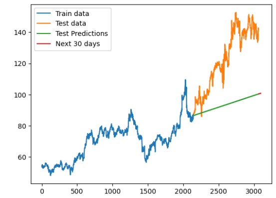


(b) Walmart Stock (8:2)

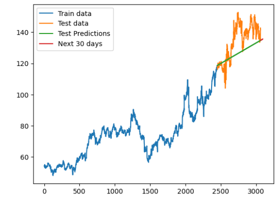
(c) Meta Stock (7:3)

(d) Meta Stock (8:2)

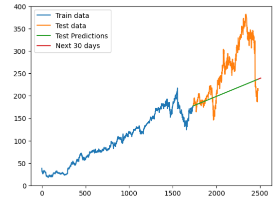
TABLE VI: Result of Holt-Winters model



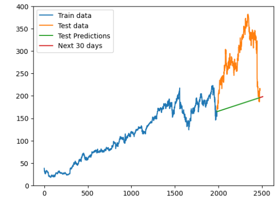
(a) Walmart Stock (7:3)



(b) Walmart Stock (8:2)

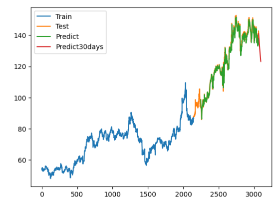


(c) Meta Stock (7:3)

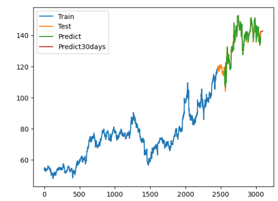


(d) Meta Stock (8:2)

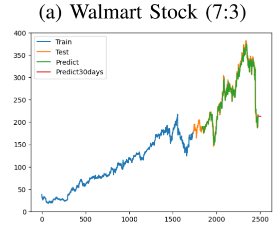
TABLE VII: Result of GRU model



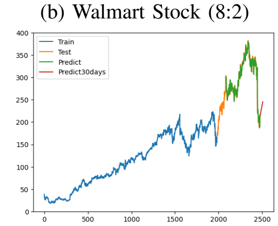
(a) Walmart Stock (7:3)



(b) Walmart Stock (8:2)



(c) Meta Stock (7:3)

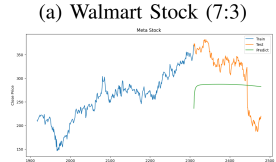


(d) Meta Stock (8:2)

TABLE VIII: Result of GPR model

(a) Walmart Stock (7:3)

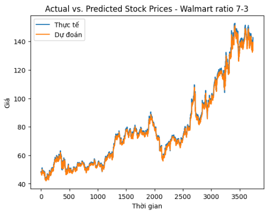
(b) Walmart Stock (8:2)



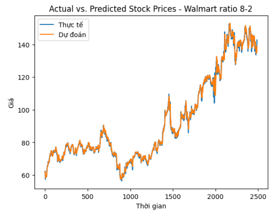
(c) Meta Stock (7:3)

(d) Meta Stock (8:2)

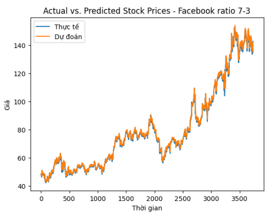
TABLE IX: Result of NNAR model



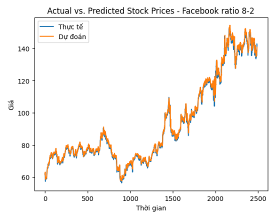
(a) Walmart Stock (7:3)



(b) Walmart Stock (8:2)

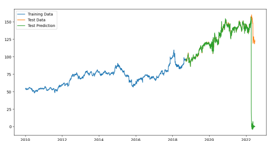


(c) Meta Stock (7:3)

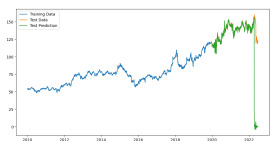


(d) Meta Stock (8:2)

TABLE X: Result of SSA model



(a) Walmart Stock (7:3)

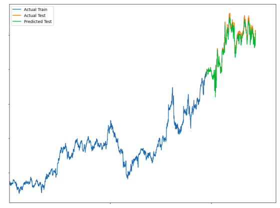


(b) Walmart Stock (8:2)

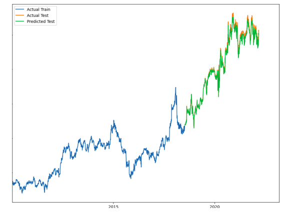
(c) Meta Stock (7:3)

(d) Meta Stock (8:2)

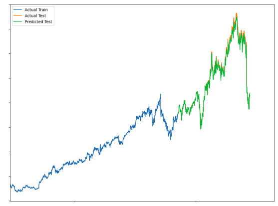
TABLE XI: Result of TCN model



(a) Walmart Stock (7:3)



(b) Walmart Stock (8:2)



(c) Meta Stock (7:3)

(d) Meta Stock (8:2)

TABLE XII: Model Performance for 2 dataset with 7:3 ratio

Dataset	Model	RMSE	MAPE	MAE
Walmart	ARIMA	41.069	27.875%	36.490
	LSTM	10.69	22.697%	0.027
	RNN	3.598	2.142%	2.185
	GRU	139.016	294.983%	138.981
	LR	34.775	23.946%	31.21
	Holt-Winters	33.279	22.657%	29.631
	GPR	49.176	0.131%	43.330
	SSA	32.1	6.27%	8.5
	NNAR	1.115	1.032%	0.816
	TCN	1.132	17.878%	0.862
Facebook	ARIMA	68.420	18.382%	53.340
	LSTM	48.73	26.5%	0.05
	RNN	3.460	1.852%	1.932
	GRU	278.403	783.253%	272.150
	LR	53.34	15.101%	41.899
	Holt-Winters	68.207	18.278%	53.069
	GPR	49.176	0.131%	43.330
	SSA	64.42	8.42%	20.3
	NNAR	1.106	1.019%	0.802
	TCN	3.217	28.497%	2.192

Based on the evaluation of various models using the provided metrics, we have identified the best performing model for each dataset. For the Walmart dataset, the Neural Network Autoregression (NNAR) model yielded the most accurate predictions, with an RMSE of 1.115, a MAPE of 1.032%, and an MAE of 0.816. These results indicate that the NNAR model achieved the lowest overall prediction error and exhibited the closest fit to the actual Walmart stock prices. On the other hand, for the Facebook dataset, the Recurrent Neural Network (RNN) model emerged as the top performer, producing an RMSE of 3.460, a MAPE of 1.852%, and an MAE of 1.932. The RNN model demonstrated superior prediction accuracy and successfully captured the underlying patterns and dynamics in the Facebook stock prices. It is important to consider that the selection of the best model depends on the specific evaluation metrics employed and the characteristics of the dataset. Hence, these findings highlight the effectiveness of the NNAR model for predicting Walmart stock prices and the

TABLE XIII: Model Performance for 2 dataset with 8:2 ratio

Dataset	Model	RMSE	MAPE	MAE
Walmart	ARIMA	11.101	6.471%	9.038
	LSTM	11.35	10.649%	0.023
	RNN	5.351	2.938%	3.433
	GRU	136.097	2206.564%	136.088
	LR	34.775	23.946%	31.21
	Holt-Winters	11.092	6.465%	9.030
	GPR	49.176	0.131%	43.330
	SSA	39.87	9.23%	12.61
	NNAR	1.162	0.948%	0.893
	TCN	1.309	9.424%	0.945
Facebook	ARIMA	131.837	40.444%	121.568
	LSTM	59.08	18.436%	0.031
	RNN	4.651	2.603%	3.013
	GRU	279.578	787.053%	273.470
	LR	53.34	15.101%	41.899
	Holt-Winters	115.583	35.086%	105.836
	GPR	49.176	0.131%	43.330
	SSA	80.14	12.20%	29.86
	NNAR	1.178	0.963%	0.907
	TCN	16.414	19.672%	13.395

RNN model for forecasting Facebook stock prices.

After evaluating the performance of different models using the provided metrics, we have identified the best model for each dataset. For the Walmart dataset, the best performing model is the Neural Network Autoregression (NNAR) model, with an RMSE of 1.162, a MAPE of 0.948%, and an MAE of 0.893. These results indicate that the NNAR model achieved the lowest overall prediction error and exhibited the closest fit to the actual Walmart stock prices. On the other hand, for the Facebook dataset, the Recurrent Neural Network (RNN) model emerged as the top performer, producing an RMSE of 4.651, a MAPE of 2.603%, and an MAE of 3.013. The RNN model demonstrated superior prediction accuracy and successfully captured the underlying patterns and dynamics in the Facebook stock prices. It is important to note that the selection of the best model depends on the specific evaluation metrics used and the characteristics of the dataset. Hence, these findings highlight the effectiveness of the NNAR model for predicting Walmart stock prices and the RNN model for forecasting Facebook stock prices.

V. Conclusion

In summary, this study aimed to predict the stock prices of Facebook (Meta) and Walmart using various time series models. We evaluated ten different models and assessed their performance using RMSE, MAPE, and MAE metrics. The results indicated that the NNAR model performed best for the Walmart dataset, while the RNN model showed superior performance for the Facebook dataset. These findings highlight the importance of model selection for accurate stock price predictions. Future research can explore additional models and incorporate additional factors to further improve prediction accuracy in real-world scenarios.

Acknowledgment

We would like to express my deepest gratitude to ***Assoc. Prof. Dr. Nguyen Dinh Thuan***, ***Mr. Nguyen Minh Nhut*** and ***Miss. Nguyen Viet Huong*** who always care for our team during this course. We appreciate your assistance and the sharing of your valuable experiences, which will help us accomplish our project.

References

[1] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[3] I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[4] K. Elissa, “Title of paper if known,” unpublished.

[5] R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.

[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

[8] Hyndman, R.J., & Athanasopoulos, G. (2018). ”Forecasting: principles and practice.” OTexts.

[9] Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). ”Statistical and Machine Learning forecasting methods: Concerns and ways forward.” PLOS ONE, 13(3), e0194889.

[10] Géron, A. (2019). ”Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.” O’Reilly Media.

[11] Zhang, G.P. (2003). ”Time series forecasting using a hybrid ARIMA and neural network model.” Neurocomputing, 50, 159-175.

[12] Wei, W.S. (2006). ”Time series analysis: univariate and multivariate methods.” Pearson Education.

[13] Box, G.E.P., Jenkins, G.M., Reinsel, G.C., & Ljung, G.M. (2015). ”Time series analysis: forecasting and control.” John Wiley Sons.