Forecasting U.S Pharmaceutical Stock Prices Using Machine Learning

NGUYEN TRUNG NGUYEN[1], LUU VINH PHAT[2], AND PHAM THUY Y VY[3]

[1]Faculty of Information Systems, University of Information Technology, (e-mail: 20521678@gm.uit.edu.vn)
[2]Faculty of Information Systems, University of Information Technology, (e-mail: 20521733@gm.uit.edu.vn)
[3]Faculty of Information Systems, University of Information Technology, (e-mail: 20522183@gm.uit.edu.vn)

ABSTRACT - With the rapid growth of the market economy, an increasing number of investors are looking to participate in the stock market. Given the complex and volatile nature of stock market trends, there is a significant risk of financial loss. This study aims to develop a robust and user-friendly tool for stock price prediction, specifically targeting pharmaceutical companies in the United States. The pharmaceutical sector is chosen due to its critical role in public health, significant research and development investments, and heightened market sensitivity to regulatory approvals, clinical trial outcomes, and emerging health crises. By leveraging a variety of models, including ARIMA, Random Forest (RF), Linear Regression (LN), Convolutional Neural Networks (GRU), Support Vector Regression (SVR), and Recurrent Neural Networks (RNN), Gauss-Newton Method Non-Linear, N-Beast, Stacking Algorithm the performance of these models in forecasting stock prices is compared to draw meaningful conclusions.

INDEX TERMS Stock price prediction, Stacking, ARIMA, RF, LR, GRU, SVM, RNN, Gauss-Newton Method Non-Linear, N-Beast, The pharmaceutical.

I. INTRODUCTION

The stock market, characterized by its inherent complexity and volatility, presents both opportunities and challenges for investors. With the growth of the market economy, more investors are venturing into the stock market, seeking to capitalize on potential financial gains. However, the unpredictable nature of stock market trends can lead to significant financial losses, especially for those without sophisticated tools and models to guide their investment decisions.

In the realm of stock market prediction, various models and techniques have been developed and utilized to forecast stock prices. These models range from traditional statistical approaches to advanced machine learning algorithms. The aim of this study is to develop a comprehensive and less error-prone tool for predicting stock prices, with a particular focus on pharmaceutical companies in the United States.

The pharmaceutical sector has been chosen for several compelling reasons. Firstly, it plays a critical role in public health, continuously developing and providing essential medications and vaccines. This sector's performance is often closely tied to regulatory approvals, clinical trial outcomes, and emerging health crises, which can lead to significant volatility in stock prices. Secondly, pharmaceutical companies invest heavily in research and development (RD), resulting in substantial fluctuations in their stock values based on the success or failure of their research efforts. Lastly, the sector's growth potential remains high due to increasing global health demands and an aging population.

To achieve accurate and reliable stock price predictions, this study employs a diverse set of models, including ARIMA, Random Forest (RF), Linear Regression (LN), Convolutional Neural Networks (GRU), Support Vector Regression (SVR), and Recurrent Neural Networks (RNN), Gauss-Newton Method Non-Linear, N-Beast, Stacking Algorithm. By comparing the performance of these models, the study aims to identify the most effective approaches for forecasting stock prices in the pharmaceutical sector.

II. RELATED WORKS

In the realm of stock price prediction, various AI techniques have been extensively studied. Zhang et al. (2017) applied Multilayer Perceptron (MLP) and NBEATS models, achieving prediction accuracies between 70% to 80%. Lipton et al. (2015) also used MLPs in a deep learning framework, reporting accuracies exceeding 80%. Dietterich (2000) explored Support Vector Machines (SVM) in ensemble methods, showing 5% to 10% accuracy improvement over individual models. Box and Jenkins (1970) pioneered ARIMA models, achieving 60% to 70% accuracy in capturing short to medium-term price movements. Recent studies by Hu et al. (2018) and Kamalov (2020) demonstrated the effectiveness of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) models, respectively, in improving stock price forecasting accuracy. Xue et al. (2020) compared

LSTM with traditional methods, affirming LSTM's high accuracy in financial time series prediction.

III. MATERIALS

A. DATASET

This study focuses on three prominent pharmaceutical companies: Alnylam Pharmaceuticals, Inc. (ALNY), Regeneron Pharmaceuticals, Inc. (REGN), and Vertex Pharmaceuticals Incorporated (VRTX). These companies have been selected due to their significant impact on the pharmaceutical sector, extensive research and development activities, and substantial market presence. The dataset for this study includes historical stock prices and various financial indicators for these companies.

The data for ALNY, REGN, and VRTX was collected from reputable financial databases from Yahoo Finance. The original data consists of 1259 rows and 7 columns. After the normalization process, the data includes 1827 rows and 7 columns. The time frame for the dataset spans from March 1, 2019, to March 1, 2024. This period captures a broad range of market conditions, including economic expansions, recessions, and the impacts of significant global events like the COVID-19 pandemic.

TABLE 1. Stock Market Feature Descriptions

FEATURE	DESCRIBE
Date	Trading day
Open	Opening price of the stock
Close	Closing price of the stock
High	Highest price during the trading day
Low	Lowest price during the trading day
Adj Close	Closing price adjusted for dividends and stock splits
Volume	Number of shares traded

B. DESCRIPTIVE STATISTICS

TABLE 2. ALNY, REGN, VRTX's Descriptive Statistics

	ALNY	REGN	VRTX
Count	1827	1827	1827
Mean	153.3136	593.1	258.8405
Median	151.76	605.95	245.1
Mode	147	615.29	209.67
Std	42.8942	168.887	65.75337
Min	65.86	273.46	164.61
Max	241.31	993.35	446.08
25%	127.75	483.135	208.075
50%	151.76	605.95	245.1
75%	189.38	726.58	296.685
Kurtosis	-0.7448	-0.683	-0.26836
Skewness	-0.1976	-0.10113	0.658198
Sample Variance	1839.916	28523.02	4323.506
Standard Error	1.0035	3.951195	1.538327

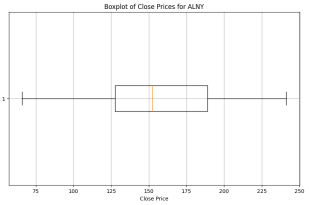

FIGURE 1. ALNY stock price's boxplot

FIGURE 2. ALNY stock price's histogram

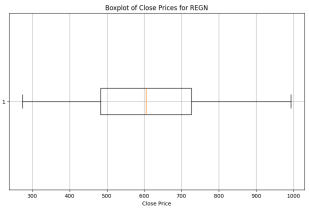

FIGURE 3. REGN stock price's boxplot

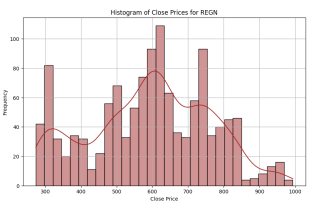

FIGURE 4. REGN stock price's histogram

FIGURE 5. VRTX stock price's boxplot

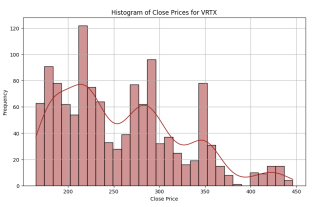

FIGURE 6. VRTX stock price's histogram

IV. METHODOLOGY

A. LINEAR REGRESSION

Linear regression is an important statistical method in data science and machine learning, used for analysis and prediction. This method models the relationship between a dependent variable (Y) and one or more independent variables (X). The goal of linear regression is to find the optimal line (or hyperplane in the case of multiple independent variables) to predict the value of the dependent variable (Y) based on the values of the independent variables (X). 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. RANDOM FOREST

Random Forest is a machine learning algorithm used for classification and regression problems in the field of machine

learning. It is a combination of multiple decision trees to create a more powerful prediction model.

Decision trees in Random Forest are tree types that divide data based on their attributes and values. What's special about Random Forest is the use of multiple decision trees in the model building process, thereby making general predictions based on the results of each member decision tree. The final result of Random Forest is calculated based on averaging or combining the results from all member trees.

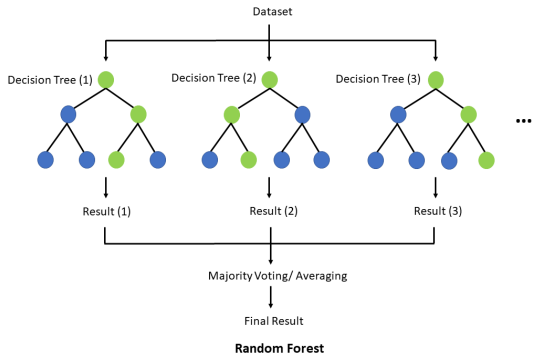

FIGURE 7. Random Forest Explain

The development of decision trees involves splitting the initial training set into smaller subsets, where at each split, only a randomly selected subset of predictors is used. The decision trees are grown continuously without pruning, up to a predefined stopping criterion set by the programmer. Common stopping criteria include Root Mean Squared Error, Gini Diversity Index, or Mean Square Error. Trees with poor predictive performance are discarded, and only those with sufficient predictive accuracy are retained in the final Random Forest model. The random selection of predictors and the aggregation of results from multiple decision trees help eliminate the overfitting problem associated with individual decision trees.

C. SUPPORT VECTOR REGRESSION

The Support Vector Regression (SVR) model is a regression method based on the Support Vector Machine (SVM) model, a supervised learning algorithm first proposed by Vladimir N. Vapnik and widely used for solving nonlinear problems. The SVM algorithm has two main steps. First, the input data is mapped to a higher-dimensional space using kernel tricks, making it easier to find the optimal hyperplane. Then, the algorithm searches for the hyperplane that best separates the data by evaluating the distance from the data points to this hyperplane.

Objective Function: The goal in SVR is to find a function $f(x)$ that has at most ϵ deviation from the actual target y_i for all training data, and at the same time is as flat as possible. The function $f(x)$ can be defined as:

$$f(x) = \langle w, x \rangle + b = w^T x + b$$

Loss Function: SVR uses the ϵ-insensitive loss function, which ignores errors that are within a certain distance ϵ from the true value. This can be formulated as:

$$L_\epsilon(y, f(x)) = \max(0, |y - f(x)| - \epsilon)$$

Optimization Problem: To find the optimal w and b, SVR solves the following optimization problem:

Minimize:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$$

Subject to:

$$\begin{cases} y_i - \langle w, x_i \rangle - b \leq \epsilon + \xi_i \\ \langle w, x_i \rangle + b - y_i \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases}$$

where:
- $\|w\|^2$ is the regularization term to ensure flatness. - ξ_i and ξ_i^* are slack variables that represent the degree to which predictions are allowed to exceed ϵ (for positive and negative deviations respectively). - C is a regularization parameter that determines the trade-off between the flatness of $f(x)$ and the amount up to which deviations larger than ϵ are tolerated.

Dual Problem: The above optimization problem is typically solved in its dual form using Lagrange multipliers:

Minimize:

$$\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\langle x_i, x_j \rangle + \epsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) - \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*)$$

Subject to:

$$\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0$$

$$0 \leq \alpha_i, \alpha_i^* \leq C$$

where: - α_i and α_i^* are the Lagrange multipliers.

Decision Function: After solving the optimization problem, the decision function for a new input x can be expressed as:

$$f(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\langle x_i, x \rangle + b$$

In practice, the dot product $\langle x_i, x \rangle$ is often replaced by a kernel function $K(x_i, x)$ to handle non-linear relationships. Common Kernel Functions

- Linear Kernel: $K(x_i, x) = \langle x_i, x \rangle$
- Polynomial Kernel: $K(x_i, x) = (\langle x_i, x \rangle + 1)^d$
- Radial Basis Function (RBF) Kernel: $K(x_i, x) = \exp\left(-\frac{\|x_i - x\|^2}{2\sigma^2}\right)$
- Sigmoid Kernel: $K(x_i, x) = \tanh(\kappa\langle x_i, x \rangle + \theta)$

D. ARIMA

An autoregressive integrated moving average (ARIMA) model is a form of regression analysis that measures the relationship between one dependent variable and other changing variables. The model aims to forecast future movements in securities or financial markets by analyzing the differences between values in the series rather than the actual values themselves. The general model of ARIMA is written as follows:

$$\Delta Y_t = \phi_1 \Delta Y_{t-1} + \phi_2 \Delta Y_{t-2} + \ldots + \phi_p \Delta Y_{t-p} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q}$$

Where:

- ΔY_{t-i} is the difference value.
- ε_{t-i} are white noise.
- $\phi_1, \phi_2, \ldots, \phi_p$ are the autoregressive coefficients for the differenced series (Δ).
- $\theta_1, \theta_2, \ldots, \theta_q$ are the moving average coefficients.

ARIMA combines three main components:

• AutoRegression (AR): A value of the time series is regressed on its own lagged values. The maximum lag (or order of autoregression) is denoted as p.

• Integrated (I): This is the order needed to make the differenced series stationary. For example, if the first-order difference (ΔY_t) is stationary, then the integration order is d = 1.

• Moving Average (MA): This is the order of the moving average. It is denoted as q.

An ARIMA(p,d,q) model is understood as an autoregressive model of order p, integrated of order d, and moving average of order q.

E. LONG SHORT-TERM MEMORY (LSTM)

Long Short-Term Memory (LSTM) networks are an advanced type of Recurrent Neural Networks (RNNs) that effectively capture long-term dependencies. First introduced by Hochreiter and Schmidhuber in 1997, LSTMs have been further developed and widely adopted through extensive research. These networks are utilized in various domains, including robot control, time series forecasting, speech recognition, anomaly detection in time series, and more. The difference between LSTM and pure RNN is that LSTM has a "forget gate layer".

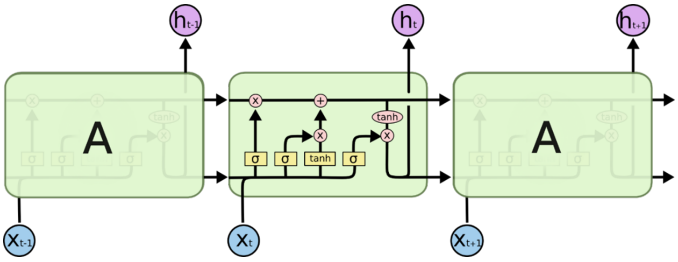

FIGURE 8. LSTM Architecture

The LSTM architecture features a memory cell managed by three distinct gates: the input gate, the forget gate, and the output gate.

• The input gate regulates what information is incorporated into the memory cell.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (1)$$

Here, f_t is the forget gate vector, σ is the sigmoid function, W_f is the weight matrix, h_{t-1} is the output from the previous time step, x_t is the current input, and b_f is the bias.

• The forget gate determines what information is discarded from the memory cell.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (3)$$

Here, i_t is the input gate vector, and $\tilde{C}_t$ is the candidate cell state.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \quad (4)$$

The new cell state C_t is computed by combining the old cell state C_{t-1} and the new candidate cell state $\tilde{C}_t$.

• The output gate governs what information is retrieved from the memory cell.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = o_t \cdot \tanh(C_t) \quad (6)$$

The output of the current time step h_t is computed by combining the new cell state C_t and the output gate vector o_t.

F. RNN

Recurrent Neural Networks (RNNs) are a class of neural networks designed for processing sequences of data by leveraging temporal dependencies. Unlike traditional feedforward neural networks, RNNs incorporate feedback loops, allowing information to persist across time steps.

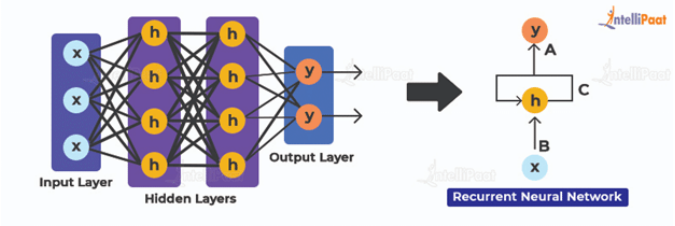

FIGURE 9. RNN Architecture

• Input Layer: Networks have only one input layer.
• Hidden Layer: Networks have one or more hidden layers.
• Output Layer: Networks only have one output layer.

RNNs address the limitation of traditional neural networks in handling sequences by using their internal state to capture

dependencies between successive inputs.

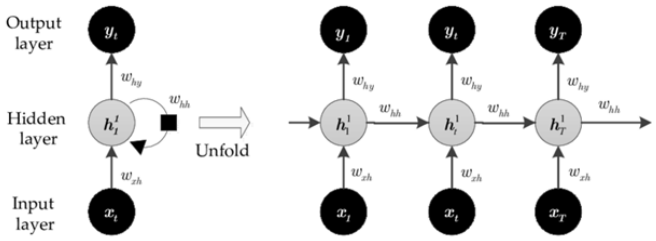

FIGURE 10. RNN Layer

The essence of RNNs lies in their recurrent nature, where the hidden state ht at time step t is computed as follows:

$$h_t = \sigma(W_h h_{t-1} + W_x x_t + b_h)$$

Where:

- h_t is the hidden state at time step t,
- h_{t-1} is the hidden state from the previous time step,
- x_t is the input at time step t,
- W_h and W_x are weight matrices,
- b_h is the bias term,
- σ is a non-linear activation function, such as $\tanh$ or ReLU.

G. STACKING ALGORITHM

Stacking in machine learning involves combining predictions from multiple base models, often referred to as first-level models or base learners. This technique begins by training several base models on the same dataset. Their predictions are then used as inputs for a higher-level model, known as a meta-model or second-level model, which generates the final prediction. Stacking aims to enhance predictive performance by leveraging the diverse insights captured by different base models, surpassing what a single model can achieve on its own.

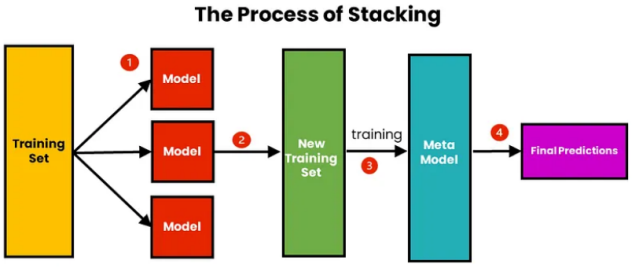

FIGURE 11. The Process of Stacking

H. GAUSS-NEWTON METHOD NON-LINEAR

In nonlinear regression, the connection between the dependent variable and the independent variables is represented by a nonlinear function, unlike linear regression, which assumes a linear relationship. The primary difficulty in nonlinear regression is determining the parameters of the nonlinear function accurately.

The Gauss-Newton algorithm is a numerical method used to solve non-linear least squares problems. These problems involve minimizing a sum of squared function values, which is equivalent to finding the best fit for a model given observed data.

Consider the objective function:

$$f(x) = \frac{1}{2}\sum_{i=1}^{m} r_i(x)^2$$

where $r_i(x)$ are the residual errors.

Initialization:

Start with an initial guess $x^{(0)}$ for the parameters x.

Iteration (for $k = 0, 1, 2, \ldots$)

1) Compute Residuals: Evaluate the residual vector $r(x^{(k)})$:

$$r(x^{(k)}) = \begin{bmatrix} r_1(x^{(k)}) \\ r_2(x^{(k)}) \\ \vdots \\ r_m(x^{(k)}) \end{bmatrix}$$

2) Jacobian Matrix: Construct the Jacobian matrix $J(x^{(k)})$, where $J_{ij} = \frac{\partial r_i}{\partial x_j}$:

$$J(x^{(k)}) = \begin{bmatrix} \frac{\partial r_1}{\partial x_1}(x^{(k)}) & \frac{\partial r_1}{\partial x_2}(x^{(k)}) & \cdots & \frac{\partial r_1}{\partial x_n}(x^{(k)}) \\ \frac{\partial r_2}{\partial x_1}(x^{(k)}) & \frac{\partial r_2}{\partial x_2}(x^{(k)}) & \cdots & \frac{\partial r_2}{\partial x_n}(x^{(k)}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial r_m}{\partial x_1}(x^{(k)}) & \frac{\partial r_m}{\partial x_2}(x^{(k)}) & \cdots & \frac{\partial r_m}{\partial x_n}(x^{(k)}) \end{bmatrix}$$

3) Update Step Direction Δx: Compute the step direction Δx using the equation:

$$J(x^{(k)})^T J(x^{(k)}) \Delta x = -J(x^{(k)})^T r(x^{(k)})$$

4) Update Parameters: Update the parameters x using:

$$x^{(k+1)} = x^{(k)} + \Delta x$$

I. N-BEATS

N-BEATS (Neural Basis Expansion Analysis Time Series) is an advanced deep learning model specifically designed for time series forecasting. N-BEATS was developed to address the limitations of traditional forecasting and previous deep learning methods. It is a domain-agnostic approach, meaning it requires no specific assumptions about the input data, making it highly flexible and powerful.

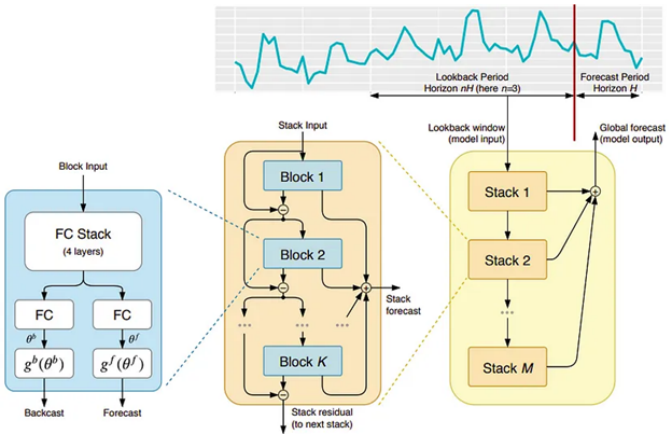

FIGURE 12. The Process of Stacking

N-BEATS Model:
The basic formula of N-BEATS involves dividing the input data $Y_{1:T}$ using deep learning blocks. Each k block will generate a prediction $\hat{y}_k$ and a residual r_0. Basic Block:
Each basic block in N-BEATS consists of two main components: trend and seasonality.
Trend Component:

- **Input:** Time series $Y_{1:T}$
- **Forecasts the trend** $\hat{y}_{\text{trend}}$ of the time series.
- Simple formula:

$$\hat{y}_{\text{trend}} = f_{\text{trend}}(X) \tag{7}$$

where X is the input to the trend layer and $f_{\text{trend}}(X)$ is a neural network function.

Seasonality Component:

- **Input:** Time series $Y_{1:T}$
- **Forecasts the seasonality** $\hat{y}_{\text{seasonality}}$ of the time series.
- Simple formula:

$$\hat{y}_{\text{seasonality}} = f_{\text{seasonality}}(X) \tag{8}$$

where X is the input to the seasonality layer and $f_{\text{seasonality}}(X)$ is a neural network function.

Doubly Residual Stack:

- The input to each block in the stack is the time series $Y_{1:T}$.
- The basic blocks are stacked on top of each other.
- Each block's forecast is an aggregation of the predictions from the preceding blocks.
- The final forecast is the sum of the forecasts from all blocks:

$$\hat{Y}_{1:T} = \sum_{k=1}^{K} \hat{Y}_k \tag{9}$$

V. RESULT

A. EVALUATION METHODS

Mean Percentage Absolute Error (MAPE): is the average percentage error in a set of predicted values.

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

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

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

Mean Absolute Error (MAE):is the average of the absolute values of the error (difference) between the predicted value and the actual value.

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

Where:

- n is the number of data points.
- y_i is the actual value at data point i.
- $\hat{y}_i$ is the predicted value at data point i.
- $|y_i - \hat{y}_i|$ is the absolute error between the actual value and the predicted value.

B. ALNY DATASET

ALNY Dataset's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MAE
LN	7:3	0.0200	1.9738	0.0139
	8:2	0.0200	2.0261	0.0130
	6:4	0.2065	2.2949	0.0163
SVR	7:3	0.1720	15.9285	0.1278
	8:2	0.0447	5.6009	0.0362
	6:4	0.2083	20.5189	0.1683
StackingSVR	7:3	0.2755	32.3626	0.2473
	8:2	0.0843	11.0605	0.0731
	6:4	0.2265	21.8084	0.1744
StackingRF	7:3	0.0557	4.9487	0.0386
	8:2	0.332	4.0935	0.0267
	6:4	0.0728	6.6908	0.0514
LSTM	7:3	3.4332	1.2804	2.4774
	8:2	4.2690	1.8940	3.3700
	6:4	5.3405	1.7294	3.3949
GRU	7:3	3.6253	1.4031	2.7097
	8:2	3.3714	1.2688	2.2735
	6:4	4.9797	1.5325	2.9835
ARIMA	7:3	24.8588	11.1675	1.5325
	8:2	16.5917	7.2845	12.6638
	6:4	39.8252	17.1811	33.7679
RNN	7:3	3.3688	1.2589	2.4191
	8:2	3.6903	1.5966	2.8875
	6:4	5.3246	1.6212	3.1383
Gauss Newton	7:3	3.1785	1.1193	2.1391
	8:2	3.1863	1.1603	2.0841
	6:4	4.5032	1.2813	2.4712
N-BEATS	7:3	32.2708	13.1399	25.8738
	8:2	13.6873	5.9355	11.4763
	6:4	11.631	5.6545	9.1351

TABLE 3. ALNY Dataset's Evaluation

C. REGN DATASET

REGN Dataset's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MAE
LN	7:3	0.0141	1.1493	0.0082
	8:2	0.0100	0.9776	0.0074
	6:4	0.1441	1.4642	0.0094
SVR	7:3	0.3665	45.5500	0.3439
	8:2	0.2718	28.2701	0.2326
	6:4	0.3652	44.6079	0.3248
StackingSVR	7:3	0.3890	47.9611	0.3627
	8:2	0.2814	27.3666	0.2281
	6:4	0.3856	46.4032	0.3392
StackingRF	7:3	0.1288	11.7344	0.0949
	8:2	0.854	7.9160	0.0644
	6:4	0.1786	19.2970	0.1450
LSTM	7:3	19.0095	2.0559	16.7611
	8:2	13.4438	1.3603	11.2067
	6:4	15.2466	1.4349	10.8038
GRU	7:3	11.5763	1.0889	8.8147
	8:2	9.6931	0.7569	6.2892
	6:4	12.3703	1.1409	8.7342
ARIMA	7:3	207.5631	24.0651	194.3134
	8:2	88.6437	7.8934	68.1368
	6:4	160.6834	17.1281	136.5513
RNN	7:3	11.6255	1.0149	8.2971
	8:2	13.8926	1.3236	11.2112
	6:4	14.7109	1.3627	10.6102
Gauss Newton	7:3	10.1511	13.8859	95.9753
	8:2	8.8330	0.6927	5.7507
	6:4	17.5815	2.1919	13.8981
N-BEATS	7:3	106.8106	10.71	0.015
	8:2	119.3929	13.9902	110.2641
	6:4	7933.49	7.47	0.007

TABLE 4. REGN Dataset's Evaluation

D. VRTX DATASET

VRTX Dataset's Evaluation				
Model	Training:Testing	RMSE	MAPE (%)	MAE
LN	7:3	0.0173	1.6484	0.0105
	8:2	0.0173	1.4266	0.0107
	6:4	0.0100	0.9776	0.0074
SVR	7:3	0.4211	56.8420	0.3884
	8:2	0.4190	50.9100	0.3907
	6:4	0.2718	28.2701	0.2326
RNN	7:3	6.4707	1.3391	4.8455
	8:2	5.9341	1.0671	4.1404
	6:4	8.1173	1.8893	6.5132
StackingSVR	7:3	0.4286	57.2759	0.3929
	8:2	0.3914	45.7314	0.3548
	6:4	0.2814	27.3666	0.2281
StackingRF	7:3	0.2234	23.7679	0.1763
	8:2	0.2220	23.7763	0.1889
	6:4	0.0854	7.9160	0.0644
LSTM	7:3	8.1622	1.7476	6.3897
	8:2	8.9099	1.8534	7.1713
	6:4	11.5816	2.7926	9.7210
GRU	7:3	5.0478	0.9888	3.5383
	8:2	5.3239	0.9564	3.6393
	6:4	5.1485	1.1018	3.7096
ARIMA	7:3	71.8767	15.9376	58.2597
	8:2	79.1692	18.5883	69.8934
	6:4	104.3654S	27.1862	92.6581
Gauss Newton	7:3	5.1358	1.0261	3.6836
	8:2	5.2868	0.9492	3.6171
	6:4	5.1399	1.0896	3.6825
N-BEATS	7:3	7.8154	2.1761	6.2601
	8:2	24.876	6.7509	21.6279
	6:4	4.0617	1.3258	3.2207

TABLE 5. VRTX Dataset's Evaluation

FIGURE 14. Data comparison across ALNY, REGN and VRTX sequentially from above with different ratios from left 6:4, 8:2, 7:3 using SVR model

3) Random Forest

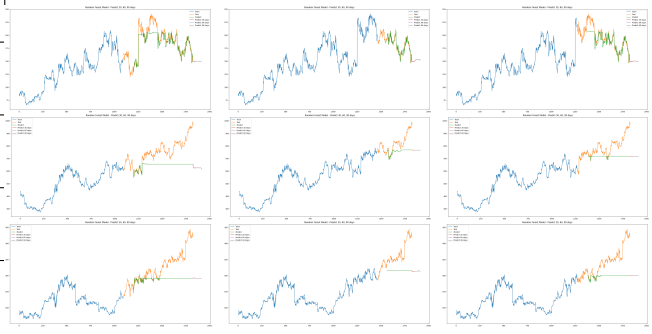

FIGURE 15. Data comparison across ALNY, REGN and VRTX sequentially from above with different ratios from left 6:4, 8:2, 7:3 using RF model

VI. RESULT

A. MODEL SETTING

1) Linear Regression

FIGURE 13. Data comparison across ALNY, REGN and VRTX sequentially from above with different ratios from left 6:4, 8:2, 7:3 using LN model

2) SVR

4) StackingSVR

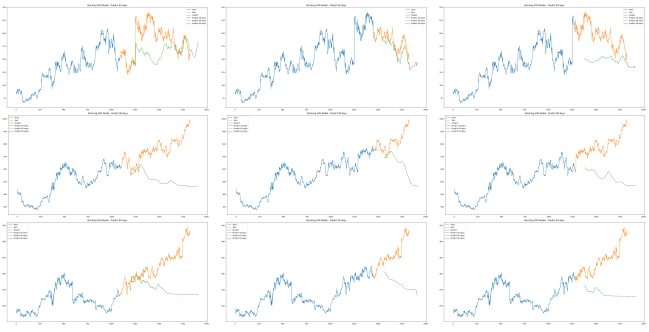

FIGURE 16. Data comparison across ALNY, REGN and VRTX sequentially from above with different ratios from left 6:4, 8:2, 7:3 using StackingSVR model

5) StackingRF

FIGURE 17. Data comparison across ALNY, REGN and VRTX sequentially from above with different ratios from left 6:4, 8:2, 7:3 using StackingRF model

6) LSTM

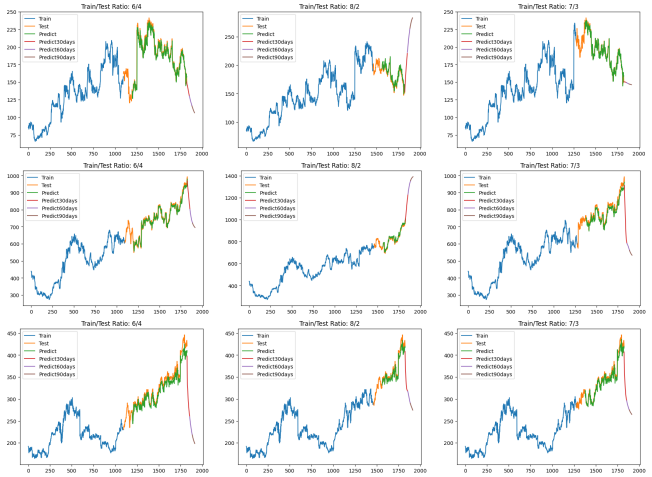

FIGURE 18. Data comparison across ALNY, REGN and VRTX sequentially from above with different ratios from left 6:4, 8:2, 7:3 using LSTM model

7) GRU

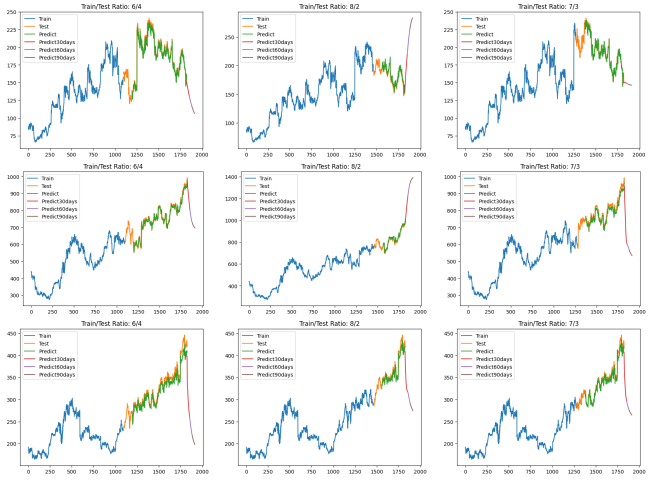

FIGURE 19. Data comparison across ALNY, REGN and VRTX sequentially from above with different ratios from left 6:4, 8:2, 7:3 using GRU model

8) ARIMA

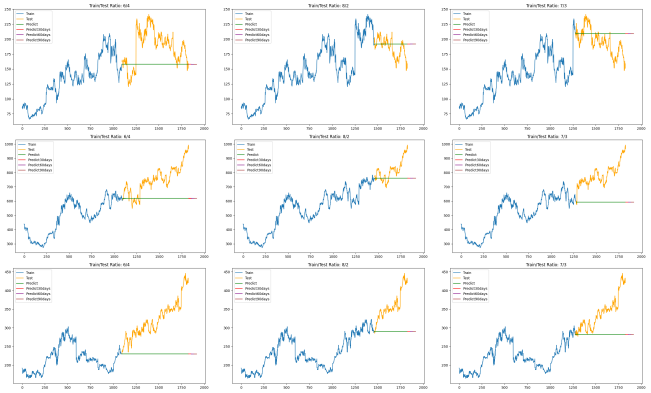

FIGURE 20. Data comparison across ALNY, REGN and VRTX sequentially from above with different ratios from left 6:4, 8:2, 7:3 using ARIMA model

9) RNN

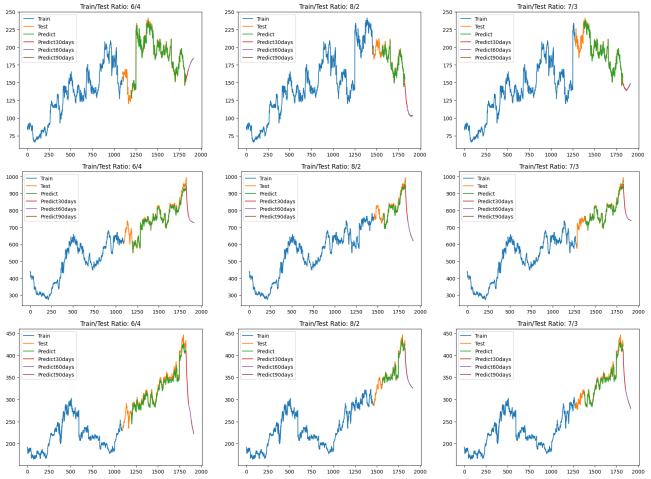

FIGURE 21. Data comparison across ALNY, REGN and VRTX sequentially from above with different ratios from left 6:4, 8:2, 7:3 using RNN model

10) Gauss Newton Non-Linear

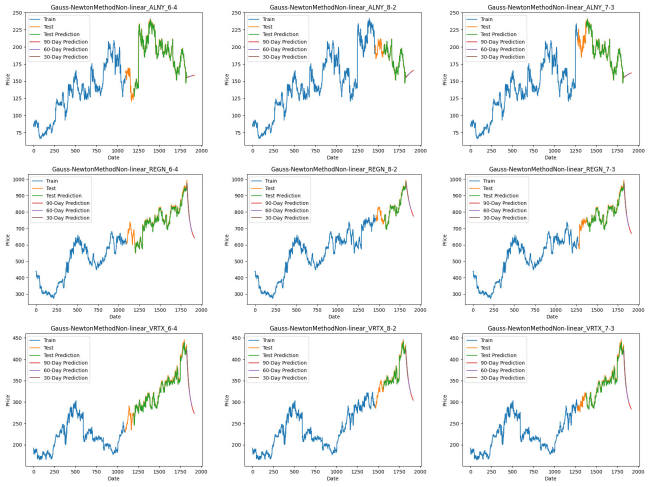

FIGURE 22. Data comparison across ALNY, REGN and VRTX sequentially from above with different ratios from left 6:4, 8:2, 7:3 using Gauss Newton Non-Linear model

11) N-BEAST

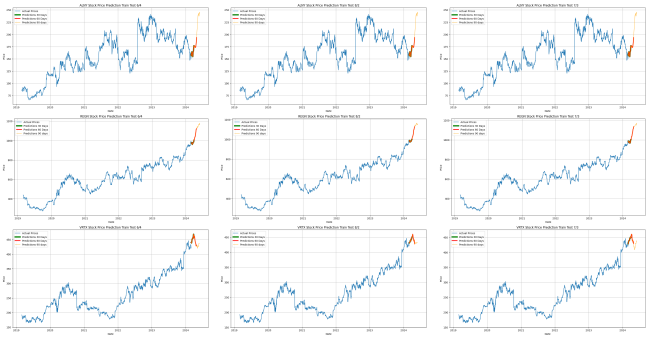

FIGURE 23. Data comparison across ALNY, REGN and VRTX sequentially from above with different ratios from left 6:4, 8:2, 7:3 using NBEAST model

VII. CONCLUSION

A. SUMMARY

Through experiments conducted on all 11 models using datasets from the three companies ALNY, REGN, and VRTX, it was found that the three models yielding the best stock price predictions are Linear Regression (LN), Stacking Random Forest (StackingRF), and GRU. This indicates that the development of statistical and machine learning models has indeed resulted in more accurate prediction models. In the future, there are plans to explore additional new models to determine their applicability to stock price data.

B. FUTURE CONSIDERATIONS

In the future, there will be a focus on exploring advanced models and methodologies to enhance the accuracy and robustness of stock price predictions. This includes leveraging cutting-edge machine learning algorithms such as deep learning architectures and ensemble methods, which are designed to capture complex patterns and dependencies in financial data more effectively.

Furthermore, the research will involve integrating novel data sources and features to gain deeper insights into market behaviors and trends. This comprehensive approach aims not only to improve prediction accuracy but also to deepen understanding of the underlying factors influencing stock prices.

Additionally, there will be a concerted effort to evaluate the scalability and practical applicability of these models across diverse market conditions and sectors. Rigorous testing and validation will be conducted to ensure reliability and consistent performance.

By embracing innovation and continuously refining methodologies, the goal is to contribute significant advancements to the field of financial forecasting. The ultimate objective is to provide decision-makers with reliable insights to facilitate informed investment decisions in dynamic and competitive markets.

REFERENCES

[1] V. Gururaj, "Stock Market Prediction using Linear Regression and Support Vector Machines" vol. 14, no. 8, 2019.

[2] YURTSEVER, M., 2021. Gold price forecasting using LSTM, Bi-LSTM and GRU. Avrupa Bilim ve Teknoloji Dergisi, (31), pp.341-347.

[3] Kishanna, H., RamaParvathyb, L. and SIMATS, C., 2022. A Novel Approach for Correlation Analysis on FBProphet to Forecast Market Gold Rates with Linear Regression.

[4] A. O. A. A. A. Ariyo, "Stock Price Prediction Using the ARIMA Model" ,2014. [Online]. Available:https://ieeexplore.ieee.org/document/7046047..

[5] M. S. S. S. A. F. K. Senthamarai Kannan, "Comparison Of Fuzzy Time Series And ARIMA"August 2019. [Online]. Available:https://www.ijstr.org/final-print/aug2019/Comparison-Of-Fuzzy-Time-Series-And-Arima-Model.pdf. [Accessed 19 June 2023].

[6] B. M. Henrique, V. A. Sobrero, and H. Kimura, "Stock price prediction using support vector regression on daily and up to the minute prices" J. Finance Data Sci., vol. 4,no. 3, pp. 183–201, Sep. 2018, doi: 10.1016/j.jfds.2018.04.003. 5

[7] Avner Abrami, Aleksandr Y. Aravkin, Younghun Kim, "Time Series Using Exponential Smoothing Cells", 9 June 2017.

[8] Professor Thomas B. Fomby, "Exponential Smoothing Models", June 2008.

[9] Bauer, E., Kohavi, R. "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants". Machine Learning 36, 105–139 (1999). https://doi.org/10.1023/A:1007515423169.

[10] Buja, A., and Stuetzle, W. "Observations on bagging". University of Pennsylvania and University of Washington, Seattle. 2002.

[11] B. M. Henrique, V. A. Sobrero, and H. Kimura, "Comparison Of Fuzzy Time Series And ARIMA", August 2019. Available:https://www.ijstr.org/final-print/aug2019/Comparison-Of-Fuzzy-Time-Series-And-Arima-Model.pdf. [Accessed 19 June 2023]. 4

[12] Jason Brownlee, "How to Create an ARIMA Model for Time Series Forecasting in Python", November 18, 2023. Available:https://www.ijstr.org/final-print/aug2019/Comparison-Of-Fuzzy-Time-Series-And-Arima-Model.pdf.

[13] Jason Brownlee, "A Gentle Introduction to SARIMA for Time Series Forecasting in Python", August 21, 2019.

[14] Alexandra M. Schmidt and Hedibert F. Lopes, "Dynamic models", 2019.

[15] Timothy O. Hodson, "Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not", 2022, https://doi.org/10.5194/gmd-15-5481-2022.

[16] Priya Pedamkar,"Support Vector Regression", March 24, 2023. Retrieved from $https : //www.educba.com/support - vector - regression/?fbclid = IwAR0ibzdmqpaaDKq2 - Q4JRcjxQcVt - C7TrHNEc90q_tCSrn8rds9x2AG8Y78$

[17] Seok-Ho Han, Husna Mutahira, Hoon-Seok Jang, "Prediction of Sensor Data in a Greenhouse for Cultivation of Paprika Plants Using a Stacking Ensemble for Smart Farms", Applied Sciences, vol.13, no.18, pp.10464, 2023.

[18] Floater, M. S. (2018, November 12). Non-linear least squares and the Gauss-Newton method. Retrieved from https://www.studocu.com/vn/document/ho-chi-minh-city-university-of-technology/phuong-phap-tinh/non-linear-least-squares-numerical-analysis/88992478

[19] Wikipedia. (n.d.). Gauss–Newton algorithm. In Wikipedia, The Free Encyclopedia. Retrieved from https://en.wikipedia.org/wiki/Gauss

[20] GeeksforGeeks. (2023, March 2). Gated recurrent unit networks. Retrieved June 20, 2024, from https://www.geeksforgeeks.org/gated-recurrent-unit-networks/

[21] PyTorch. (n.d.). GRU. Retrieved from https://pytorch.org/docs/stable/generated/torch.nn.GRU.html