A Comparison of artificial neural network models, recurrent neural network models, and support vector machine for stock price prediction

June 6, 2024

Contents

List of Figures

List of Tables

Abstract

Prediction of stock price has been a big and consistent issue over time. One of the most resonating issues with stock price prediction is the volatile nature of the stock market. Novel approaches using machine learning have helped in breaching this gap by leveraging machine learning algorithms. In this paper, we used LSTM, Multilayer Perceptron, and SVM to predict the stock price of Total Energies company. The LSTM Model was able to capture more patterns and trends in the data by giving us a more reduced mean squared error than the MLP and SVM on both the training and test data. The implications of this research are that, it has potentials of improve the stock market and aiding stock investment in the financial sector at a high probability of avoiding risks of losing money.

CHAPTER 1

Introduction

1.1 Background of the study

Many researchers have tried predicting stock prices accurately but from all indications, it has been noticed that the volatile nature of stock prices makes it difficult to predict accurately. The belief of most researchers is that stock price cannot be accurately predicted as a result of the degree of divergence of stock price. Another difficulty with stock price prediction is the very many dynamic factors that determines the price of a stock in the financial market. Over the years, most researchers have leveraged statistical time series models like ARIMA for stock prediction, and ARCH and GARCH models for gauging the volatility of stock price but these methods have not been adequate enough. Some other researchers used traditional single machine learning models like SVM, etc. but there is still a gap in the way stock prices are being predicted. Hence, It became a need for me to come up with not just an accurate but precise way of predicting the stock prices of different companies by leveraging deep learning models that can accurately capture the patterns and trends in such time series data. This is why careful analysis will be carried out to give versatile and clear answers to various research questions that this project aims to tackle using practical steps.

1.2 Significance of the study

There are a lot of gaps that many researchers have not been able to fill up when dealing with stock volatility. Some researchers have used volatility gauging models and some have tried just traditional models, while some have approached it using hybrid methods but in this research, we will compare outstanding models that have been tested and proven to capture sequential data very well. Hence, this research has potential in the financial market to aid stock prediction with very low-risk levels. Forecasting errors are one important thing to deal with when predicting stock prices, this research uses ANN, RNN, and SVM to breach that gap by forecasting stock prices at a very small forecasting error. This study exposes the need for feature engineering in any time series project especially the case of stock prediction as it is very vital to engineer features that can capture the hidden patterns and trends in any kind of sequential data. This research reveals the theoretical impact of volatility on stock prediction by leveraging both domain expertise and practical approaches. The impact of this research on society cannot be over-emphasized. If we have a machine learning model or application that can accurately predict the stock prices of different companies, it will aid the economy to have a good level of confidence in investing in the stock market and will also help the companies have a lot of people investing in their company and this will then be like a symbiotic association where the company gets a lot of investors and the investors are investing with high level of confidence. The above is what the potential of this research aims to achieve.

1.3 Scope of the study

This research will leverage the approaches below to achieve its aim

- Data and Preprocessing: The data to be used is from Yahoo Finance, specifically Total Energies company data. Data pre-processing techniques like data assessment, handling missing values, feature engineering, and feature selection will be considered
- Machine Learning Models: artificial neural networks with enough layers will be used and with different batch sizes iteratively, RNN(LSTM) will also be used and

lastly we will use support vector regressor

- Evaluation Metrics: This research will use the mean absolute error and mean squared error to evaluate the performance of each model
- Experiment Design: This research will follow the path: data collection, data assessment, data pre-processing, model selection and training, hyperparameter optimization

1.4 Aim and Objectives

Aim:

- The aim of this research is to Compare artificial neural network models, recurrent neural network models, and support vector machine for stock price prediction

Objectives

- Train LSTM, MLP, and Support vector machine for
- Evaluate the performance of each model using mean squared error and mean absolute error
- Establish the best features for the project

1.5 Statement of The problem

Traditional machine learning algorithms do not give reduced forecasting errors on stock data

CHAPTER 2

Literature Review

In this aspect of this research, we will explore the thorough explanation of basic terminologies for stock prices. More so, we will discuss the related works and explore the gaps within each of the researches.

Stock volatility One of the most important considerations and concerns of many researchers has been the presence of volatility in stock prices. Stock price volatility refers to the degree of variation of a stock's price over time. It is a statistical quantity that has the ability to capture different types of fluctuations in stock prices. High volatility indicates that a stock's price can change dramatically over a short period, both in terms of upward and downward movement. Conversely, low volatility implies that a stock's price changes are more gradual and less significant. Hence, it is mandatory to have models like neural networks that can operate well on sequential data other than just data with fixed sizes like LSTM. Historical Volatility is calculated based on historical prices over a specific period. It is usually expressed as an annualized standard deviation of daily price changes. Historical volatility gives investors an idea of how much the stock price has varied in the past, which can be useful for risk assessment and strategy development. Stock market volatility increases with financial leverage, as predicted by Black and Christie, although this factor explains only a small part of the variation in stock volatility. In addition, interest rate and corporate bond return volatility are correlated with stock return volatility. Finally, stock market volatility increases during recessions. None of these factors, however, plays a dominant role in explaining the

behavior of stock volatility over time Schwert, G. W. (1989).

2.1 Related Works

This paper explores the significance of technology in finance, particularly in financial risk management and stock prediction. It focuses on two prominent algorithms used for stock forecasting: Artificial Neural Networks (ANN) and Support Vector Machines (SVM). The paper highlights hybrid models like ANN-MLP (Multi-Layer Perceptron), GARCH-MLP, and combinations of the Backpropagation algorithm with Multilayer Feed-forward networks, which yield better results.SVMs feature simple decision boundaries that help avoid over-fitting. They achieve about 60%-70% accuracy in stock prediction. The paper suggests that integrating ANN and SVM with other novel techniques could result in more potent hybrid methodologies. The paper also addresses challenges in using these algorithms, such as time constraints, data limitations, and the cold start problem. Kurani et al. (2023). This paper introduces a novel machine-learning technique for stock forecasting using a fine-tuned version of support vector regression (SVR) on time series data. By applying a grid search technique to the training dataset, the optimal kernel function and parameters are selected and validated using a validation dataset. This optimization enhances the model's accuracy and efficiency, reducing time and memory requirements while preventing overfitting. The method is applied to analyze various stock market performance parameters, including daily and monthly returns, cumulative monthly returns, volatility, and associated risk. The approach is tested on eight large datasets from different domains, demonstrating superior accuracy in stock prediction compared to similar methods. Additionally, the proposed technique is significantly faster than its counterparts, highlighting its efficiency and effectiveness Dash et al. (2023). This paper addresses the challenge of predicting long-term stock index prices, which is a significant departure from the more common focus on next-day stock price predictions. The study aims to forecast daily prices up to a year ahead using daily close prices of global stock indices. This long-term forecasting is essential for practical applications in financial markets and fills a gap in existing research. The paper highlights the superior performance of a rolling forward-validation approach over traditional cross-validation for long-term predictions. It introduces an

optimized Genetic Algorithm for Support Vector Regression (OGA-SVR) to forecast multi-step ahead prices efficiently. The model's performance is compared with several other methods: standard Support Vector Regression (SVR), Grid Search based SVR (GS-SVR), Genetic Algorithm-based SVR (GA-SVR), and Long Short-Term Memory (LSTM) algorithms. These models are tested on five global stock indices: Nifty, Dow Jones Industrial Average (DJIA), DAX performance index (DAX), Nikkei 225 (NI225), and Shanghai Stock Exchange composite index (SSE). The evaluation metrics used are Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE). Empirical results demonstrate that the OGA-SVR model outperforms the other models in long-term price prediction. The OGA-SVR model is particularly effective in forecasting the long-term underlying patterns of index prices, offering valuable insights for building trading and risk mitigation systems for investors and traders Beniwal et al. (2023). This paper presents a model for predicting stock market values using time series forecasting, aiming to provide a safer investment environment with minimized risk. The model utilizes real-time datasets from approximately fifteen companies across various sectors to forecast future stock prices, helping users decide on investments for the next quarter. The model incorporates three main concepts for forecasting: Holt-Winters Triple Exponential Smoothing: This method accounts for stocks showing periodic changes throughout the season. It considers three factors: Base Level, Trend Level, and Seasoning Factor. The Holt-Winters Algorithm decomposes these factors to predict future stock prices. Recurrent Neural Network (RNN): Specifically, the Long Short-Term Memory (LSTM) model is used. Unlike normal neural networks, LSTM networks have memory cells that retain values through feedback loops, improving the prediction of time series data. Recommendation System: This system filters data and predicts ratings based on various factors, enhancing the accuracy of stock price forecasts Mohan et al. (2023). This research addresses the challenge of predicting stock market prices, given their inherent volatility and the influence of numerous factors like market trends, supply and demand, the global economy, public sentiments, corporate results, and historical prices. The study employs new technologies, such as data mining and machine learning, to develop a more accurate prediction model that can reduce human errors. The primary focus is on predicting the closing prices of specific stocks using a supervised machine learning algorithm, particularly a Recurrent Neural Network

(RNN) with Long Short-Term Memory (LSTM) cells, applied to time-series data. The study involves predicting the next day's stock prices for each day of a year and validating these predictions against actual prices. The developed RNN-LSTM model predicts future stock prices with high accuracy, achieving a mean absolute percentage error below 2%. The model's performance is validated using various metrics, including the coefficient of determination (R-squared), Pearson's correlation coefficient, Spearman's rank correlation coefficient, and explained variance error, all of which indicate a strong correlation between actual and predicted prices. The model's predictions are further validated using statistical tests such as the chi-square goodness of fit and the Wilcoxon signed-rank test, which support the accuracy of the predictions. The study suggests that machine learning, specifically RNN-LSTM models, can effectively predict stock prices one day ahead based solely on historical data. Future work aims to incorporate investor sentiments on sensitive financial disclosures to further refine the prediction model, addressing the complexities that historical prices alone cannot explain Jahan, I. (2018).

2.2 Research Gap

The research papers collectively explore various machine learning techniques for stock market prediction, each presenting unique methodologies and focus areas. Despite their contributions, several gaps can be identified. The papers highlight different machine learning algorithms such as Artificial Neural Networks (ANN), Support Vector Machines (SVM), and Recurrent Neural Networks (RNN) with Long Short-Term Memory (LSTM). However, there is a lack of comparative studies that extensively evaluate the performance of these diverse algorithms under similar conditions. While some papers propose hybrid models (e.g., ANN-MLP, GARCH-MLP, and combinations with SVM), there is limited research on systematically integrating these models and evaluating their collective performance. The potential of combining multiple algorithms to enhance prediction accuracy remains underexplored. Techniques like grid search for parameter optimization are discussed (e.g., fine-tuned SVR). However, the specific impact of different optimization techniques on the performance of hybrid models or across various datasets is not thoroughly investigated. Most studies focus on short-term predictions,

such as next-day prices, while others attempt long-term forecasts up to a year. There is a gap in understanding how methodologies that perform well for short-term predictions adapt to long-term forecasting and vice versa. The papers use various validation metrics and approaches (e.g., rolling forward-validation, cross-validation). A standardized validation framework that can be universally applied to evaluate the robustness of different models is lacking. Although some studies use real-time datasets and aim to provide practical investment advice, there is a gap in bridging the theoretical models with real-world applications. The implementation of these models in live trading environments and their adaptability to market changes need further exploration. While some studies plan to incorporate investor sentiments and other external factors, current research primarily focuses on historical price data. The integration of qualitative data (e.g., news, and economic indicators) into the prediction models is still an emerging area. The efficiency and scalability of these models, particularly when applied to large datasets or real-time streaming data, are not comprehensively addressed. The computational requirements and potential trade-offs between accuracy and speed need further investigation. Although some studies employ statistical tests to validate predictions, a more in-depth analysis of the statistical properties and hypothesis testing across different market conditions is necessary to generalize the findings. There is a limited focus on sector-specific stock predictions. While some studies include diverse sectors, a targeted approach to understanding sectoral variations in prediction accuracy could provide more actionable insights for investors. By addressing these gaps, future research can develop more comprehensive, robust, and practical stock market prediction models that leverage the strengths of various machine learning techniques and adapt to dynamic market environments.

2.3 Comparison of Related Works

Table 2.1: Comparison of Research Papers on Stock Prediction Algorithms

Author	**Algorithms Used**	**Significant Results**
Kurani (2023)	Artificial Neural Networks (ANN), Support Vector Machines (SVM), Hybrid models like ANN-MLP, GARCH-MLP, Backpropagation with Multilayer Feed-forward networks	Hybrid models yield better results, SVM achieves 60%-70% accuracy, suggests integrating ANN and SVM with novel techniques
Dash (2023)	Support Vector Regression (SVR) with grid search for parameter optimization	Enhanced accuracy and efficiency, reduced time and memory requirements, superior performance on eight large datasets
Beniwal (2023)	Optimized Genetic Algorithm for Support Vector Regression (OGA-SVR), standard SVR, Grid Search based SVR (GS-SVR), Genetic Algorithm-based SVR (GA-SVR), Long Short-Term Memory (LSTM)	OGA-SVR outperforms other models in long-term price prediction, effective for building trading and risk mitigation systems
Mohan (2023)	Holt-Winters Triple Exponential Smoothing, Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM), Recommendation System	Accurate quarterly forecasts for 15 companies, LSTM improves prediction of time series data, recommendation system enhances accuracy
Jahan (2018)	Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM)	High accuracy with mean absolute percentage error below 2%, validated with multiple metrics and statistical tests, strong correlation between actual and predicted prices

CHAPTER 3

Methodology

In this section of this research, will explore the methods used for this work.

3.1 Data Collection and presentation

The data used for this work was gotten from Total Energies companies on Yahoo Finance website. The data has 7000 observations and 5 columns. **First Five Rows**

Table 3.1: Data First Five Rows

Date	Open	High	Low	Close	Adj Close	Volume
1994-02-28	13.8125	14.0625	13.8125	14.0000	3.785994	480800
1994-03-01	13.8750	13.8750	13.6875	13.7500	3.718387	231000
1994-03-02	13.2500	13.5000	13.1875	13.5000	3.650780	176800
1994-03-03	13.4375	13.4375	13.2500	13.4375	3.633877	98200
1994-03-04	13.3750	13.5000	13.3750	13.5000	3.650780	88400

Table 3.2: Last Five Rows

Date	Open	High	Low	Close	Adj Close	Volume
2024-02-20	63.930000	63.930000	63.110001	63.259998	63.259998	1282700
2024-02-21	63.610001	64.139999	63.389999	64.089996	64.089996	1905700
2024-02-22	64.160004	64.419998	63.599998	64.139999	64.139999	1253800
2024-02-23	64.199997	64.730003	63.830002	64.510002	64.510002	1004700
2024-02-26	63.770000	64.320000	63.639999	64.300003	64.300003	1026900

3.2 Data Description

	Open	High	Low	Close	Volume
Count	7551.000000	7551.000000	7551.000000	7551.000000	7.551000e+03
Mean	46.641429	47.004245	46.261206	46.646166	1.667459e+06
Std	16.235481	16.335954	16.109003	16.219168	1.437190e+06
Min	12.875000	12.937500	12.687500	12.812500	0.000000e+00
25%	35.000000	35.325001	34.685001	34.947501	7.316000e+05
50%	49.340000	49.730000	48.919998	49.310001	1.321000e+06
75%	57.455000	57.939999	57.045000	57.540001	2.199850e+06
Max	90.779999	91.339996	89.790001	89.940002	2.705530e+07

Table 3.3: Statistical summary of stock data

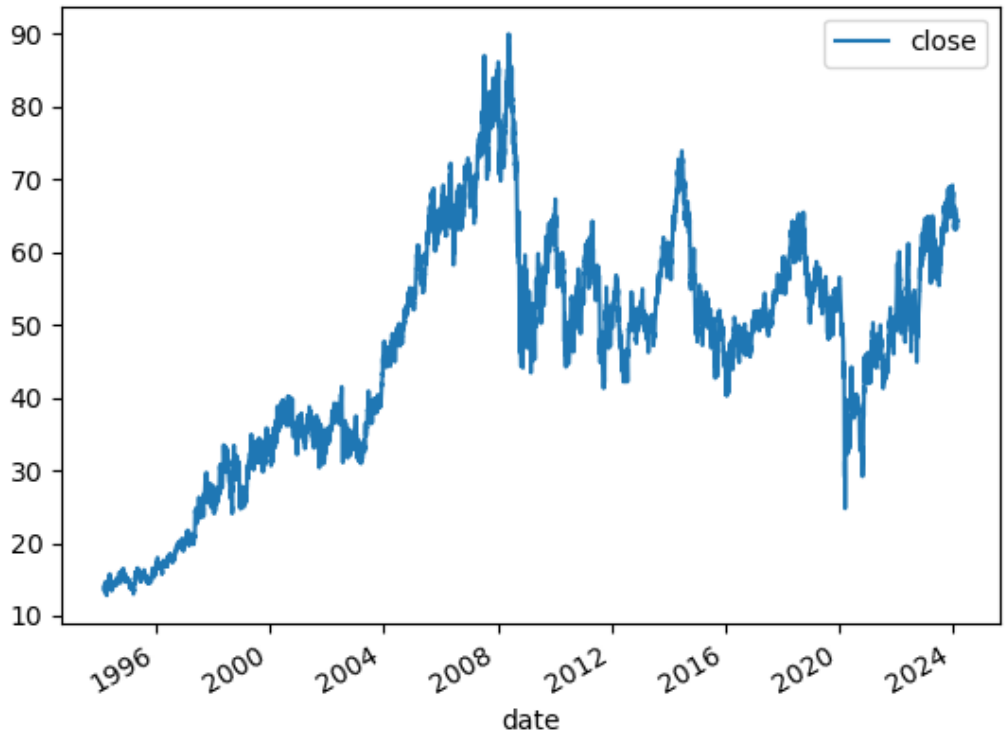

Figure 3.1: Data presentation

Explanation of Columns

Column	Explanation
Open	The opening price of the stock at the beginning of the trading day.
High	The highest price the stock reached during the trading day.
Low	The lowest price the stock reached during the trading day.
Close	The closing price of the stock at the end of the trading day.
Volume	The total number of shares traded during the trading day.

Table 3.4: Explanation of stock data columns

3.3 Data Preprocessing

In this research, we employed feature engineering techniques to create features like relative strength index, momentum, exponential moving average and simple moving average in order to capture both linear and non-linear patterns in the data. Missing values were removed after some or the features were created.

3.4 Machine Learning Algorithms

3.4.1 Support Vector Machine

The support vector algorithm is a nonlinear generalization of the generalized portrait algorithm developed in the 1960s [1]. It is firmly grounded in the framework of statistical learning theory. Its present form was developed at AT and T Bell Laboratories by Vapnik and co-workers in the early 1990s [2]. Statistical learning theory characterizes the properties of learning machines which enable them to generalize well to unseen data Roy et al. (2023). If the data is linearly separable (meaning it needs a hyperplane to separate the classes), then the Support Vector Machine (SVM) is simple, and it finds the decision boundary which is the most distant from the points nearest to the said decision boundary from both classes. If the data is not linearly separable (non-linear), then Kernel Trick is used in SVM, which involves mapping the feature space in the current dimension to higher dimensions such that they are easily separable using a decision boundary. One of the benefits of Support Vector Machines is that they work very well in cases with limited datasets. Data points closer to the hyperplane that influence the position and orientation of the hyperplane are called Support vectors. SVM classification is robust to outliers. **Kernel Trick**:

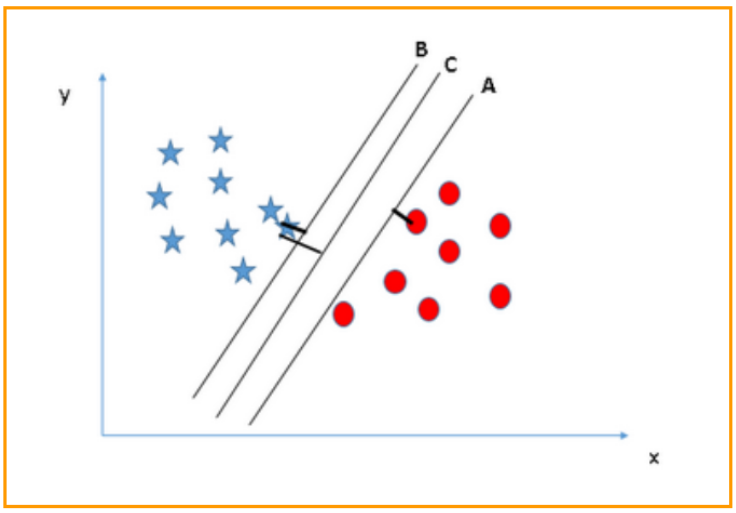

Figure 3.2: Kernel Trick 1

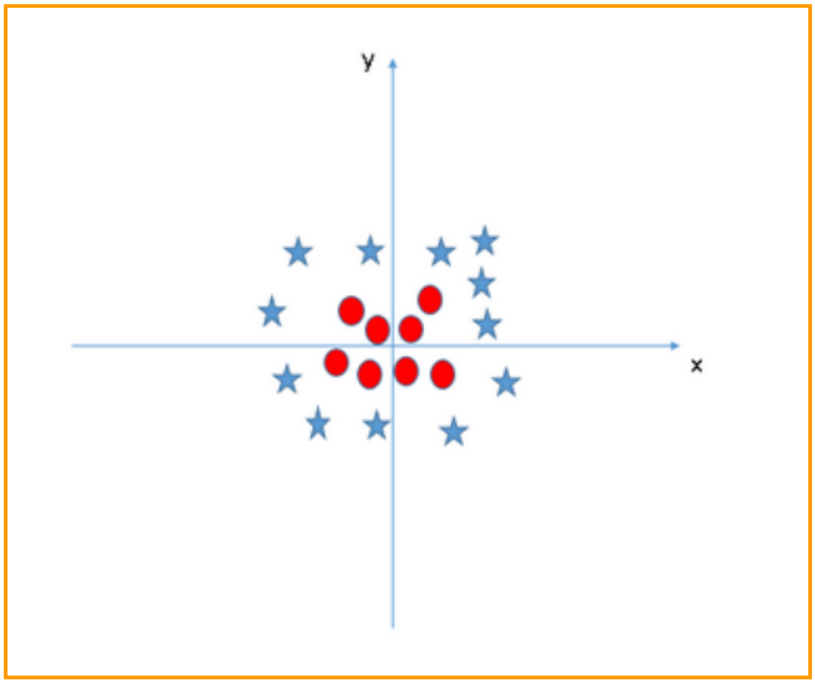

Figure 3.3: Kernel Trick 2

Maximizing the distances between the nearest data point (either class) and hyper-plane will help us decide the right hyper-plane. This distance is called the Margin. The margin for hyper-plane C is high compared to both A and B. Hence, we name the right hyper-plane C. In the case below, we do not have a linearly separable boundary. Here, the Kernel Trick will come to the rescue. Support vector machine has been chosen based on its practical importance and mathematical models. Support Vector Machines are supervised learning models used for classification and regression analysis. They are particularly well-known for classification tasks. The goal of SVM is to find the hyperplane that best separates the classes in the feature space. For a linearly separable dataset, SVM aims to find a hyperplane defined by the equation:

$$wx + b = 0$$

- w is the weight vector.
- x is the input feature vector.
- b is the bias term.

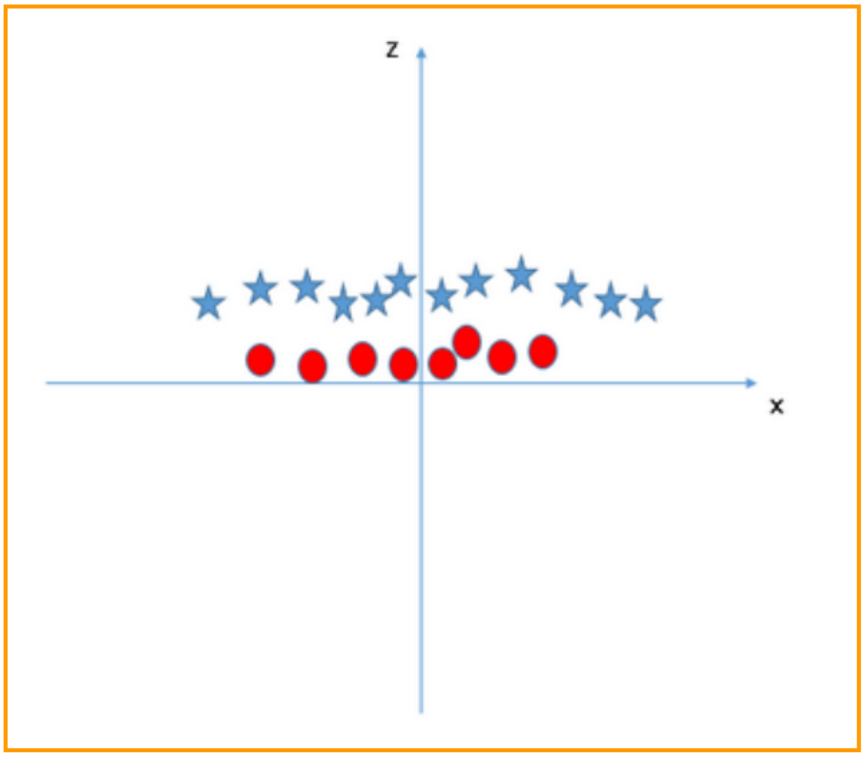

Figure 3.4: Kernel Trick 3

The objective is to maximize the margin, which is the distance between the hyperplane and the nearest data point of each class. This can be formulated as a convex optimization problem. Minimize:

The Support Vector Machine

The SVM is the distance between the hyperplane and the closest points from each class (support vectors). For the hyperplane

$$wx + b = 0$$

the margin can be defined as: $M = \frac{2}{||W||}$ where $||W||$ is the Euclidean norm of the weight vector.

Maximizing the Margin: The goal of SVM is to maximize the margin, which translates into minimizing $||W||$ To ensure the margin is maximized while correctly classifying the data points, the following constraints must be satisfied:

$$y_i(W.x_i + b) \leq 1$$

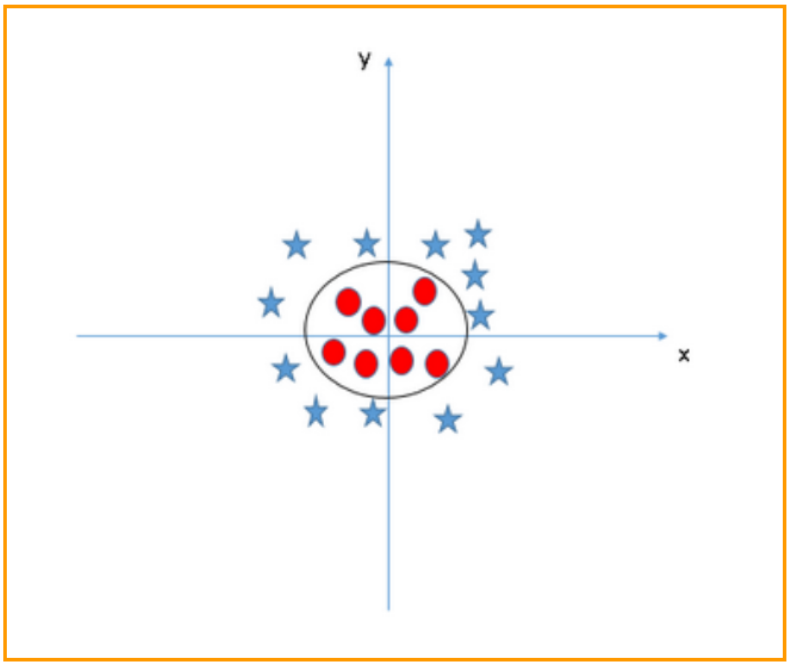

Figure 3.5: Kernel Trick 4

for all i

Recurrent Neural Network(RNN)

Recurrent Neural Networks (RNNs) are powerful sequence models that were believed to be difficult to train, and as a result, they were rarely used in machine learning applications. Recurrent Neural Networks (RNNs) are artificial neural network models that are well-suited for stock price prediction tasks whose inputs and outputs are sequences. The importance of developing methods for predicting stock price is exemplified by tasks such as speech recognition, speech synthesis, named-entity recognition, language modeling, and machine translation. An RNN represents a sequence with a high-dimensional vector (called the hidden state) of a fixed dimensionality that incorporates new observations using an intricate nonlinear function. RNNs are highly expressive and can implement arbitrary memory-bounded computation, and as a result, they can likely be configured to achieve nontrivial performance on difficult sequence tasks. However, RNNs have turned out to be difficult to train, especially on problems with complicated long-range temporal structures â precisely the setting where RNNs ought to be most

useful. Since their potential has not been realized, methods that address the difficulty of training RNNs are of great importance (Sutskever, I. (2013)). There are at least three streams of RNN research: binary, linear, and continuous-nonlinear. Linear Widrow (1962) drew inspiration from the brain to introduce the gradient descent Adeline adaptive pattern recognition machine. Anderson (1968) initially described his intuitions about neural pattern recognition using a spatial cross-correlation function. Concepts from linear system theory were adapted to represent some aspects of neural dynamics, including solutions of simultaneous linear equations $Y = AX$ using matrix theory, and concepts about cross-correlation. Kohonen (1971) made a transition from linear algebra concepts such as the Moore-Penrose pseudoinverse to more biologically motivated studies that are summarized in his books (Kohonen, 1977, 1984). These ideas began with a mathematically familiar engineering framework before moving towards more biologically motivated nonlinear interactions (Grossberg, S. (2013)).

LSTM Mathematical Formulas Staudemeyer et al. (2019)Forget Gate:

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

Input Gate:

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

Candidate Cell State:

$$C_t = tanh(W_c.[h_{t-1}, x_t] + b_C)$$

Update Cell State:

$$C_t = f_t.C_{t-1} + i_t.C_t$$

Output Gate:

$$O_t = \sigma(W_o.[h_{t-1}, x_t] + b_o)$$

Hidden State:

$$h_t = O_t.tanh.(C_t)$$

where: X_t: is the input at time step t f_t, i_t, C_t, O_t, h_t represent the forget gate, input gate, candidate cell state, cell state, output gate, and hidden state at the time step t respectively Sherstinsky et al. (2020). W_f, W_i, W_C, W_o are the weight matrices associated with the forget gate, input gate, cell state, and output gate, respectively. b_f, b_i, b_c, b_o are

the bias terms associated with the forget gate, input gate, cell state, and output gate, respectively. σ denotes the sigmoid activation function, and tanh denotes the hyperbolic tangent activation function Smagulova et al. (2019).

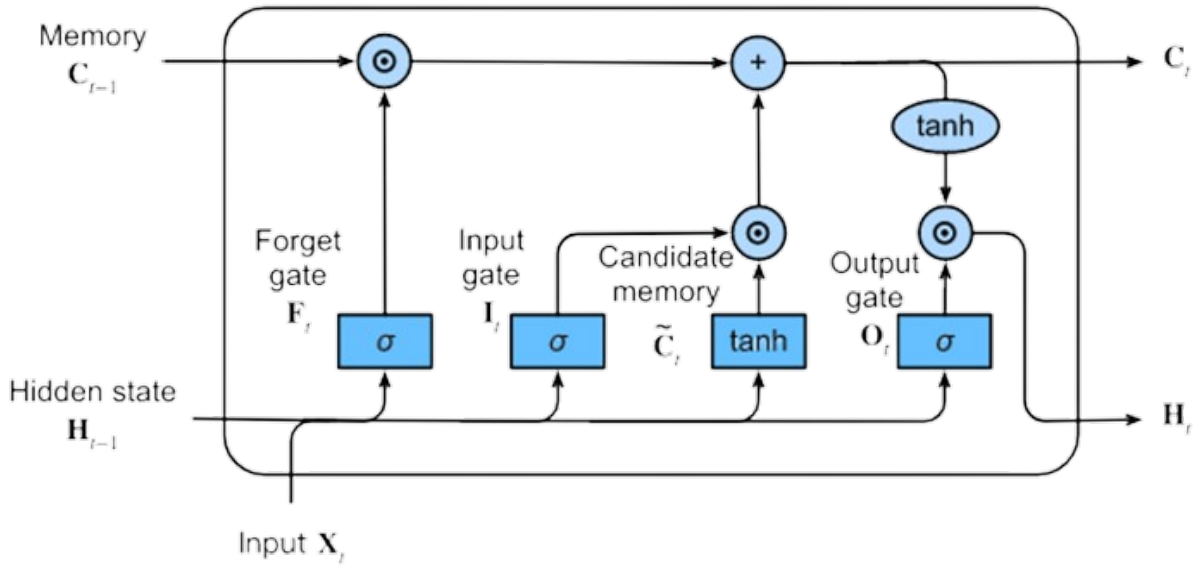

Figure 3.6: LSTM architecture diagram

Feed Forward Artificial Neural Network

Artificial neural networks (ANNs) are essential tools in machine learning that have drawn increasing attention in many researches. Besides offering powerful techniques for data analysis, ANNs provide a new approach for stock prediction to build models for complex behaviors, and heterogeneous activities as well as to explore optimization in neural systems, in ways that traditional models are not designed for. In this paper, we introduce ANNs and employ multilayer perception Yang et al. (2020). ANN models are easily applicable with their higher capability to identify nonlinear relationships between inputs and designated outputs to predict choice behaviors Lee et al. (2018).

Multi-Layer Perceptron (MLP): A Multi-Layer Perceptron (MLP) is a class of feed-forward artificial neural network (ANN) that consists of at least three layers of nodes: an input layer, one or more hidden layers, and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLPs are the quintessential deep learning models, capable of capturing complex relationships in data

by learning from examples. **Input Layer**:
The input layer consists of nodes (neurons) that receive the input data. Each node in this layer corresponds to a feature in the dataset. The input layer does not perform any computations; it just passes the data to the first hidden layer.

Hidden Layers:
An MLP has one or more hidden layers, which are the core computational units of the network. Each neuron in a hidden layer receives input from every neuron in the previous layer and sends its output to every neuron in the next layer. The neurons in the hidden layers apply a weighted sum to their inputs and pass this sum through an activation function, which introduces non-linearity into the model, allowing it to learn complex patterns.

Output Layer:
The output layer is the final layer that provides the networkâs prediction. The number of neurons in this layer depends on the task: for regression problems, there is typically one neuron, while for classification problems, there may be one neuron per class (in the case of binary classification) or a set of neurons equal to the number of classes (in the case of multi-class classification). **Training an MLP**

Training an MLP involves adjusting the weights of the neurons to minimize the error in predictions. This process is typically carried out using the following steps:

- Forward Propagation: Input data passes through the network layer by layer. Each neuron computes a weighted sum of its inputs and applies an activation function to produce an output. The output of each layer becomes the input to the next layer.

- Loss Calculation: After forward propagation, the loss (or error) between the predicted output and the actual output is computed. Common loss functions include Mean Squared Error (MSE) for regression tasks and Cross-Entropy Loss for classification tasks.

- Backpropagation: Backpropagation is an algorithm used to compute the gradient of the loss function with respect to each weight by the chain rule, moving backward from the output layer to the input layer. The gradients indicate how much a change

in each weight would impact the loss, helping in updating the weights correctly.

- Weight Update: The weights are updated using optimization algorithms like Stochastic Gradient Descent (SGD), Adam, or RMSprop. The goal is to reduce the loss by iteratively adjusting the weights in the direction that reduces the loss.

Advantages of MLP

MLPs can be used for both regression and classification problems. They are capable of approximating any continuous function given sufficient neurons in hidden layers and sufficient training data (Universal Approximation Theorem). By adding more hidden layers and neurons, MLPs can model complex, non-linear relationships in the data. With proper regularization techniques (e.g., dropout, L2 regularization), MLPs can generalize well to unseen data. **Disadvantages of MLP** Training deep MLPs can be computationally expensive and time-consuming, especially with large datasets. Without adequate regularization, MLPs can overfit the training data, performing poorly on new, unseen data. MLPs typically require large amounts of training data to perform well, which might not always be available.

3.5 Evaluation Metrics

Evaluation Metrics are quantities used to check the performance of machine learning algorithms on either the train test data set. For a regression task like this, we have restricted ourselves to using the mean squared error and mean absolute error.

3.5.1 Mean Squared Error

It measures the average squared difference between the observed actual outcomes and the predicted outcomes by the model. The MSE is calculated using the following formula:

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

Lower MSE: Indicates a better fit of the model to the data, as it suggests that the predicted values are close to the actual values. Higher MSE: Indicates a worse fit, suggesting that the predictions deviate significantly from the actual values. Sensitivity to Large Errors: Because errors are squared, larger errors have a disproportionately

large impact on the MSE, making it sensitive to outliers. Mathematical Simplicity: It is straightforward to compute and mathematically convenient for optimization, especially in the context of gradient-based learning algorithms.

3.5.2 Mean Absolute Error

The Mean Absolute Error (MAE) is another common metric used to evaluate the accuracy of predictive models, especially in regression analysis. It measures the average absolute difference between the observed actual outcomes and the predicted outcomes by the model. Lower MAE: Indicates a better fit of the model to the data, suggesting that the predicted values are close to the actual values. Higher MAE: Indicates a worse fit, suggesting that the predictions deviate significantly from the actual values. Robust to Outliers: Unlike MSE, MAE does not square the errors, so it is less sensitive to outliers. Interpretability: MAE is in the same units as the target variable, making it more interpretable compared to MSE. It is calculated using the formula :

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

3.6 Exploratory Data Analysis

In this section, we will explore and analyze several relationships and patterns among different variables in the data.

From the time series plot below, one of the first things to do in a time series project is to plot the time series data in order to avoid trouble in the long run. The plot shows that the data has upward and downward trends at different time intervals. The close price rose between 1996 and 2008 though it faced a very brief noise within these periods. The trend between 2008 and 2020 was that of sinusoidal with its highest amplitude above 70 close price.

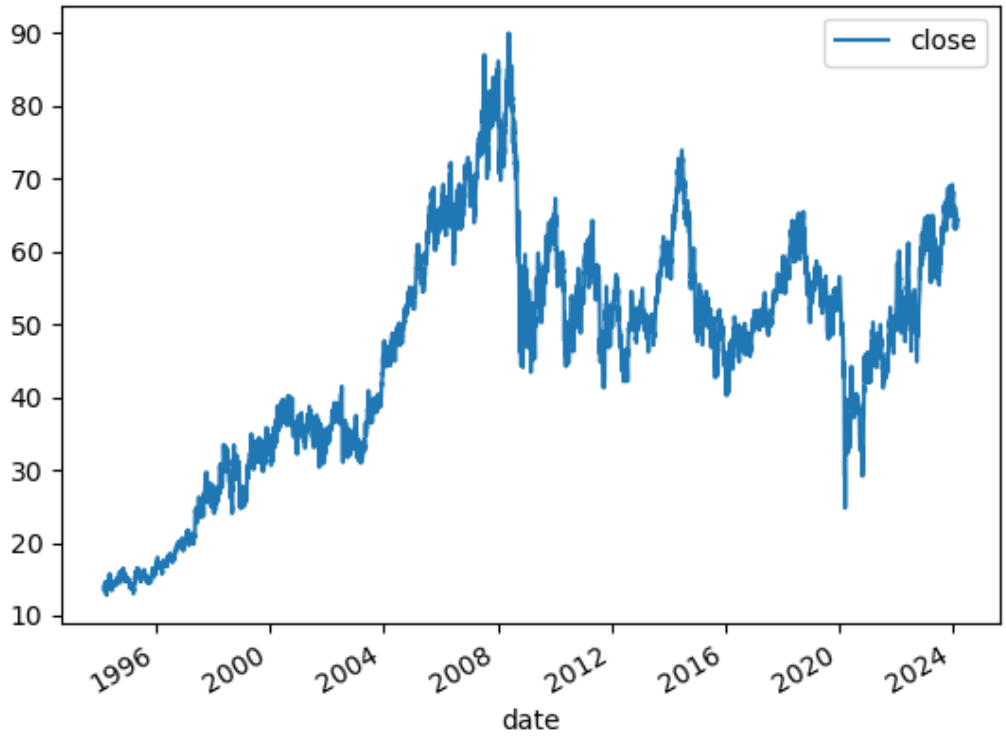

Figure 3.7: plotting the time series data first

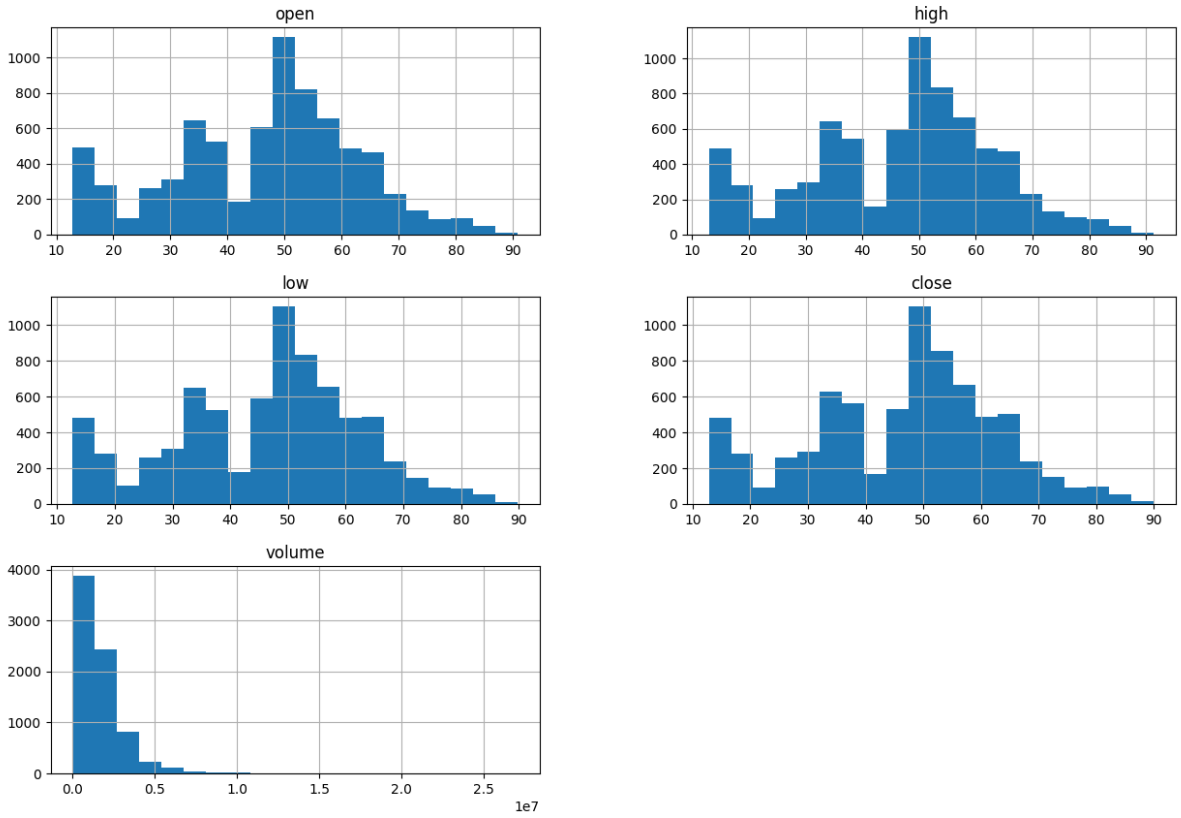

Figure 3.8: Distribution plots

from the distribution plots of the open, low, high, close, and volume above, we could see that volume is skewed to the right, while the remaining variables are kind of normal though not really normal. hence, the data needs to been standardized or we say scaled so that our model will not move outside the spread of its prediction. Anyways, domain experts have it that it is not advisable to use low, open, high, and volume to predict close price, they claim that it will lead to biasedness. The practical truth behind this is presented in a chart below that shows how each of the columns gave a perfect relationship against close prices. Knowing fully well that models are estimators, instead of expecting them to give estimates based on their probabilistic nature, they will be giving overfitting values.

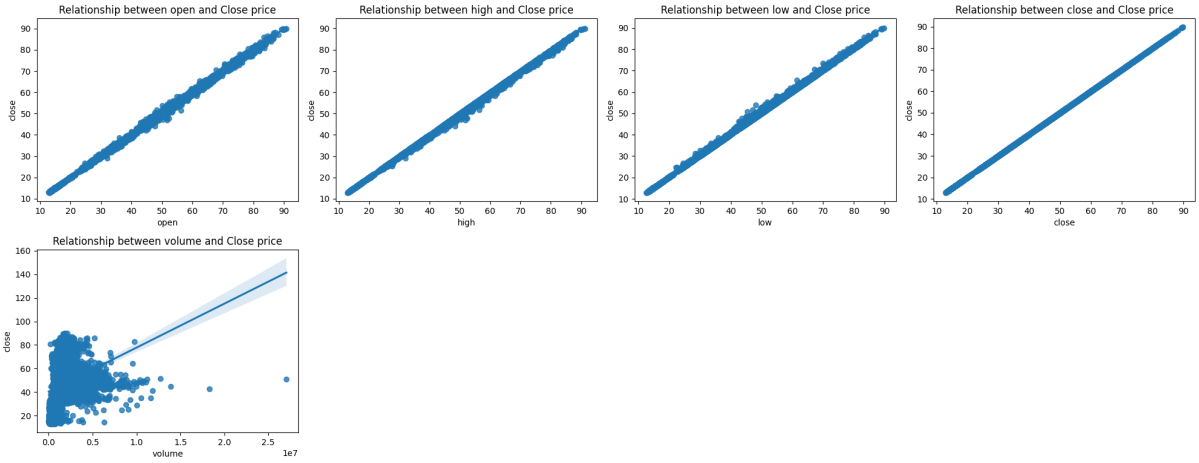

Figure 3.9: Relationship between default variables

Just like we said much earlier, the low, high, volume, and open price are showing perfect relationship with the close price. This is a deceit as they will make the model learn unnecessary things in the data. Hence, subsequently, we will drop these columns and engineer new features that has potential of capturing the trends and patterns in the data.

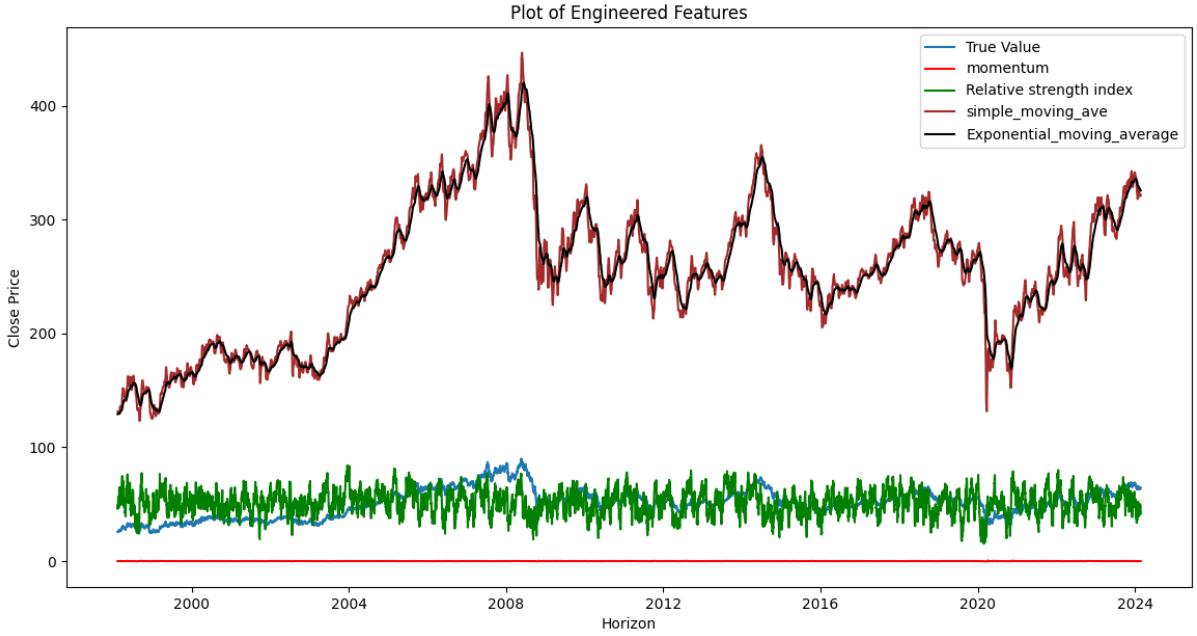

Figure 3.10: Chart of Engineered Features against the close price

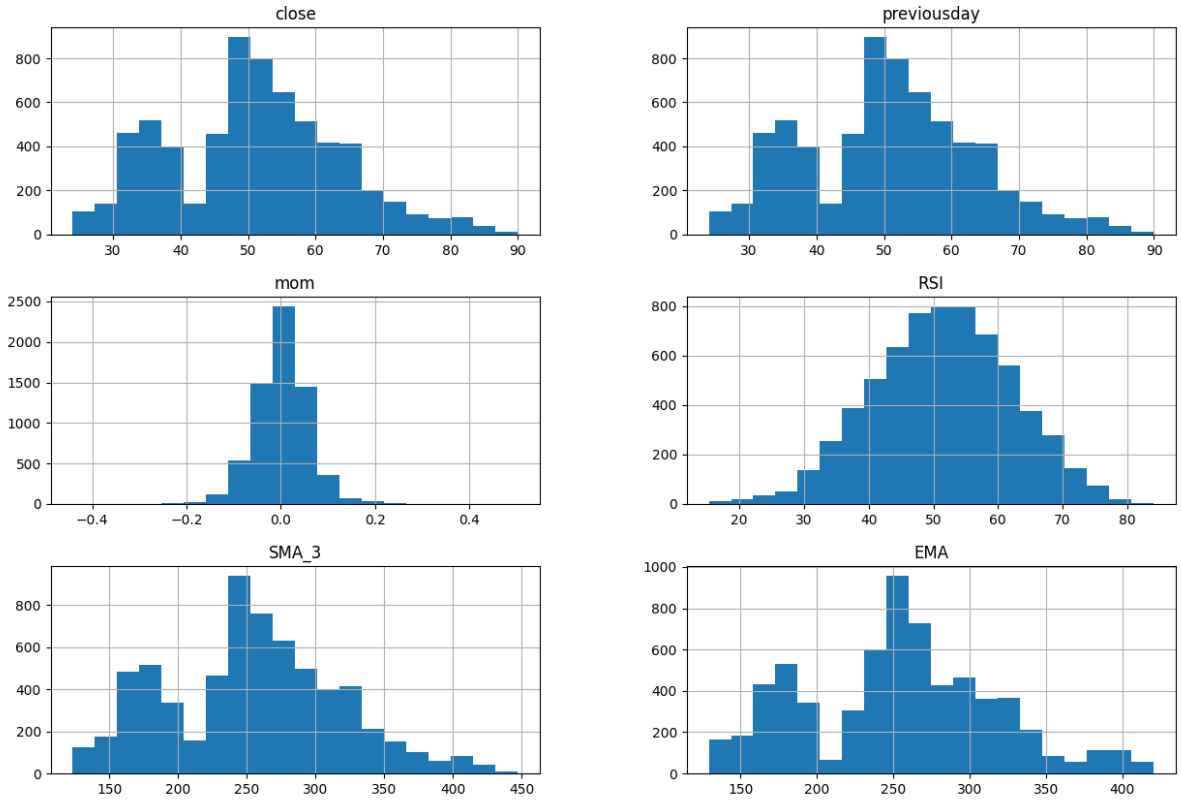

Figure 3.11: Distribution plot of engineered features

CHAPTER 4

Discussion of Results

In this section, we will discuss the results generated from this research. We will explain the simulation and other relevant results.

4.1 Pre Simulation Results using LSTM

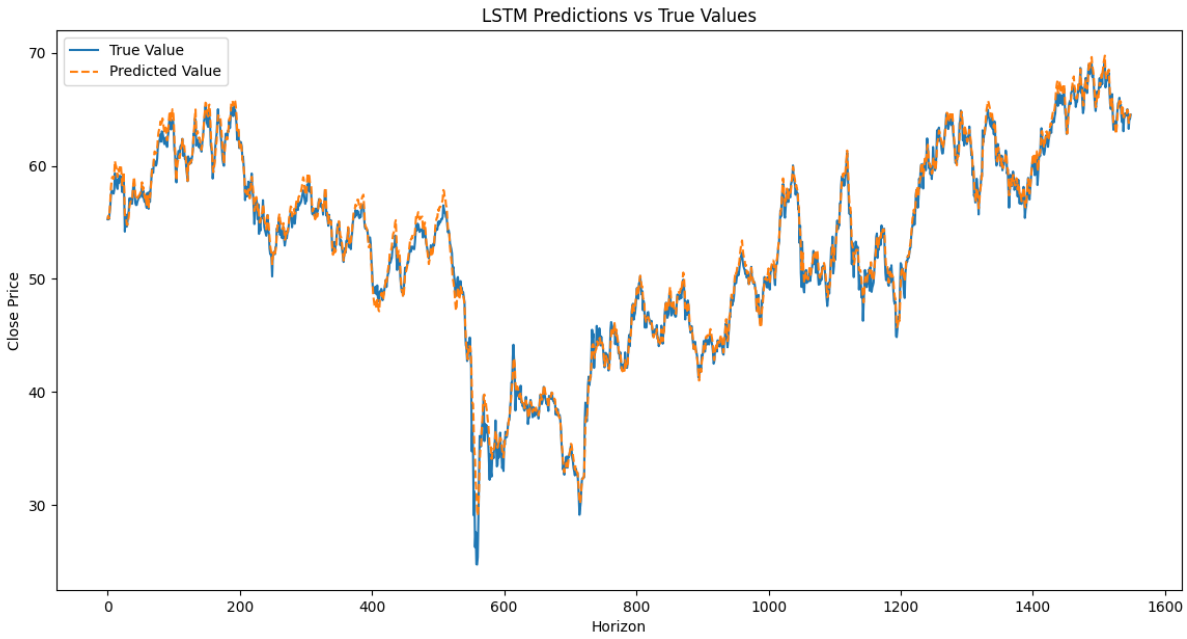

Figure 4.1: Pre-simulation result Lstm

The chart above shows the LSTM predictions' results against the actual stock close price for Total Energies company. The actual close price is in blue while the predicted close price is in orange, we can see that it is obvious that the LSTM Model was able to almost perfectly mimic the direction of movements of the actual close price, which shows that the model is giving perfect predictions.

4.2 Pre Simulation Results using MLP

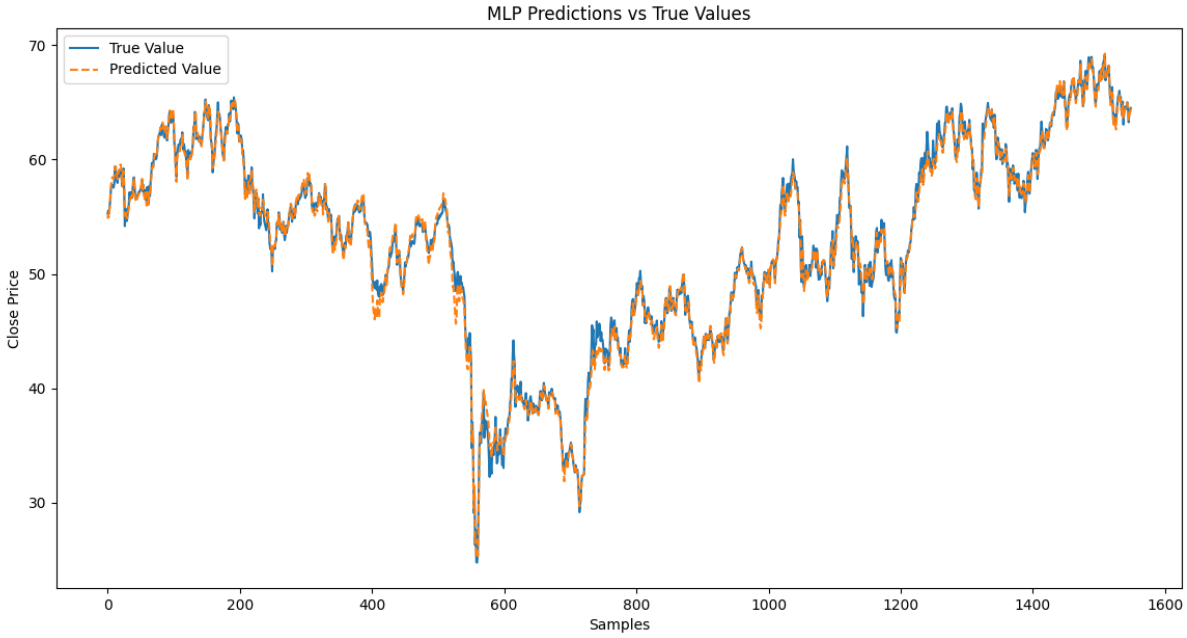

Figure 4.2: Pre-simulation result MLP

The chart above is a plot of the predictions of Multilayer perception against the actual close price at the stage of simulation, hence showing that the model is performing well by being able also to mimic the close price accurately.

4.3 Results after Simulation, Training and Evaluation

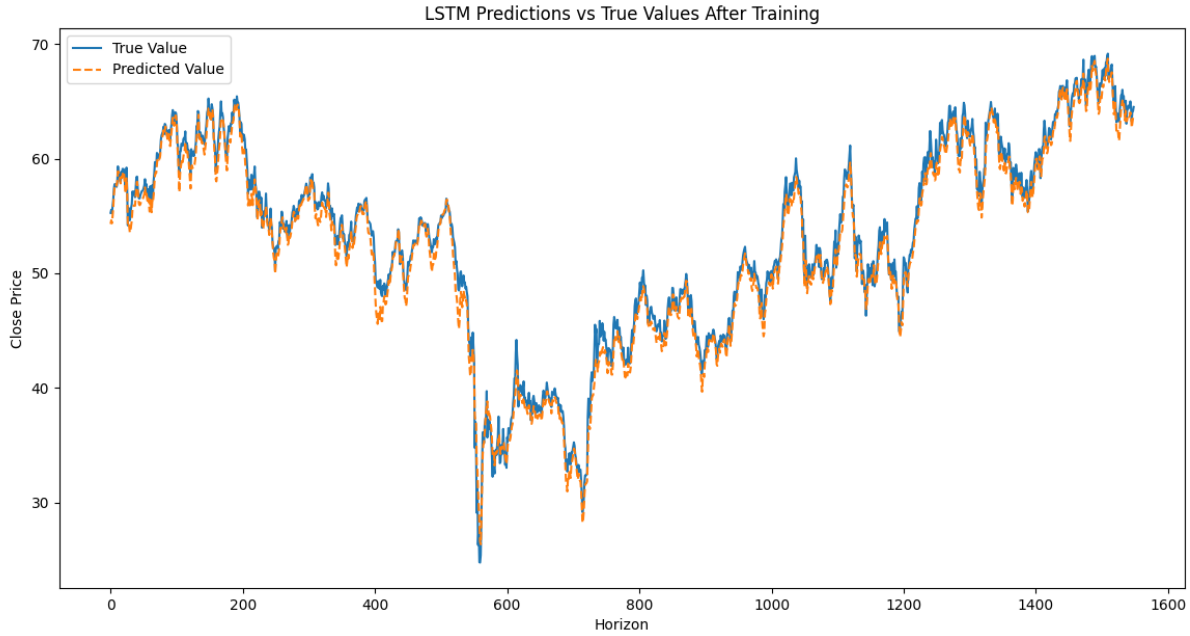

Figure 4.3: After-simulation result LSTM

The plot above is the result of the best model being the LSTM model which showed significant improvement and prediction than the Multilayer Perceptron. The LSTM still showed good predictions even after simulation and was able to achieve more reduced error.

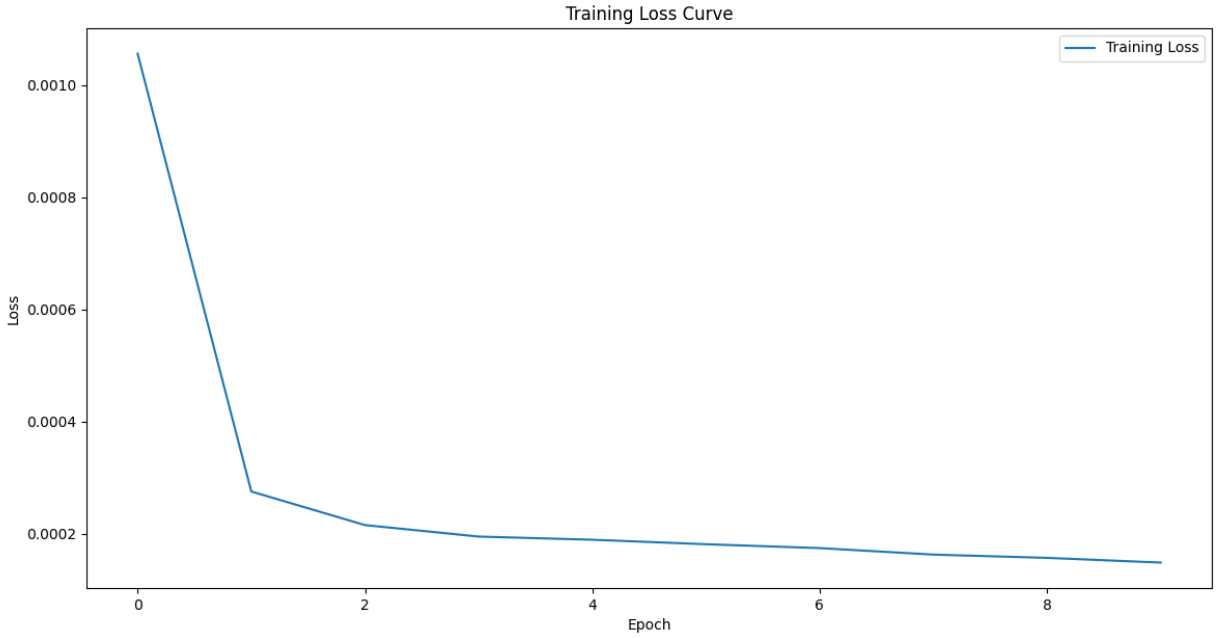

Figure 4.4: Loss Curve

The loss curve was able to show that at the beginning of the training, the loss is usually high because the model parameters (weights and biases) are initialized randomly or with small values. The predictions of the LSTM model are far from the actual values. As the training begins, the optimizer updates the model parameters in a direction that minimizes the loss. During this phase, the model was able to learn basic patterns in the data, resulting in a steep decline in the loss. The downward slope signifies that the LSTM model is effectively learning from the training data. The model parameters are being optimized to minimize the prediction error. The downward slope indicates successful learning and optimization of the model parameters. The model is progressively improving its predictions, capturing complex patterns in the data, and ideally converging to a state where it has minimized the prediction error as much as possible. This behavior is a positive indicator of the model's ability to learn and generalize from the training data.

Model	**Train (RMSE)**	**Test (RMSE)**
LSTM	0.90	1.03

Table 4.1: LSTM Model error metrics

The table shows the root mean squared error values of LSTM being the best model amongst the SVM and MLP. The MSE values show the variance between the predicted close price and that of the actual close price for Total Energies stock price is small.

Bibliography

Schwert, G. W. 1989. Why does stock market volatility change over time? *The journal of finance*, *44*(5),1115-1153.

Kurani, A., Doshi, P., Vakharia, A., & Shah, M. (2023). A comprehensive comparative study of artificial neural network (ANN) and support vector machines (SVM) on stock forecasting. *Annals of Data Science*,*10*(1),183-208.

Dash, R. K., Nguyen, T. N., Cengiz, K., & Sharma, A. (2023) Fine-tuned support vector regression model for stock predictions. *Neural Computing and Applications*,*35*(32),23295-23309.

Beniwal, M., Singh, A., & Kumar, N. (2023). Forecasting long-term stock prices of global indices: A forward-validating Genetic Algorithm optimization approach for Support Vector Regression. *Applied Soft Computing*,*145*,110566.

Mohan, M., Raja, P. K., Velmurugan, P., & Kulothungan, A. (2023). Holt-winters algorithm to predict the stock value using recurrent neural network.*methods*,*8*(10).

Jahan, I. (2018). Stock price prediction using recurrent neural networks.

Roy, A., & Chakraborty, S. (2023). Support vector machine in structural reliability analysis: A review. *Reliability Engineering and System Safety*,*233*, 109126.

Grossberg, S. (2013). Recurrent neural networks. *Scholarpedia 8*(2),188

Sutskever, I. (2013). Training recurrent neural networks (pp. 1-101). *Toronto, ON, Canada: University of Toronto.*

Smagulova, K., James, & A. P. (2019). A survey on LSTM memristive neural network architectures and applications. *The European Physical Journal Special Topics*, *228*(10), 2313-2324

Staudemeyer, R. C., Morris,& E. R. (2019). Understanding LSTM–a tutorial into long short-term memory recurrent neural networks. *arXiv preprint arXiv:*,1909.09586.

Sherstinsky, & A. (2020). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *The European Physical Journal Special Topics*, *404*,132306.

Yang, G. R., & Wang, X. J. (2020). Artificial neural networks for neuroscientists: a primer. *Neuron*,*107*(6),1048-1070.

Lee, D., Derrible, S., & Pereira, F. C. (2018). Comparison of four types of artificial neural network and a multinomial logit model for travel mode choice modeling. *Transportation Research Record*,*2672*(49),101-112.