Journal of Administrative and Financial Sciences

Volume: 10 / N°: 01 (2026), P: 103-119

Stock Price Prediction to Support Investment Decisions: A Model Based on LSTM Networks

BERBOUCHA Meriem [1] [*]

[1] Annaba University (Algeria), meriem.berboucha@univ-annaba.dz

Received: 26/05/2026 ***Accepted: 14/06/2026*** ***Published: 30/06/2026***

Abstract:

This study aimed to predict stock price movements using one of the deep learning models called "Long Short-Term Memory" (LSTM), in order to support investment decisions in the financial market.

In this study, the movement of stock prices of three leading technology companies "Apple Inc, Tesla Inc, and Microsoft Corp" was predicted using LSTM model. The model also generated trading signals that act as a tool to support investment decisions in the financial market.

The study found that LSTM models are efficient and capable of predicting and analyzing complex patterns and time dependencies in stock prices. This confirms that LSTM recurrent neural networks are suitable for analyzing financial time series and can be considered as supporting tools to improve investment decisions.

Keywords: Stock price; forecasting; RNNs; Investment decisions; LSTM networks.

JEL Classification : G17; G12; C53; C45.

[*] ***Corresponding author.***

Introduction:

The process of investment decision-making represents one of the modern trends in contemporary financial thought. It involves determining the type of assets that an institution should hold, whether they are tangible assets, such as buildings and machinery, or intangible assets, such as securities. Investment decisions also concern the optimal utilization of these assets to achieve the maximum possible return with the lowest risk, which makes the decision-making process inherently complex. Achieving uniformity in decision-making is nearly impossible and requires specialized technical skills, due to the influence of the surrounding circumstances. **(Azhar, Imamul Haque, Umer, & Ishfq, 2021, p. 49)**

Similarly, making investment decisions in the financial market is characterized by a high degree of uncertainty and complexity, stemming from the intricate nature of financial markets in general and stock markets in particular. There are numerous factors beyond financial indicators, which can influence stock price fluctuations, such as economic and political conditions and the behavioral characteristics of the investor making the decision. **(Konstantinou, 2017, p. 02)**

This makes the process of predicting stock price movements using traditional models based on linear and multiple regression ineffective. However, price movements become predictable if the non-linear and dynamic relationships that characterize financial markets are uncovered. This motivates the use of artificial neural networks to analyze financial time series **(Bing , Hao , & Zhang , 2012, p. 1055)**. In this field, deep learning models are superior to traditional machine learning techniques such as Support Vector Machines (SVM), Random Forest, Auto regression (AR), Autoregressive Moving Average (ARMA), and Autoregressive Integrated Moving Average (ARIMA). This is because stock prices move in non-linear directions and in a random manner. For this reason, several researchers have proposed deep learning models capable of tracking this non-linearity, most notably Recurrent Neural Networks (RNN) and Long Short-Term Memory LSTM networks. The complexity of stock price movement requires time series analysis for prediction **(Gourav , Rinkle , & Himanshu , 2023, p. 9728)**

Since the variable is the stock price, it can be univariate or multivariate. Univariate data includes information about one specific stock only, while multivariate data includes stock prices of more than one company over different time periods. Time series data analysis helps identify patterns, trends, and periods or cycles present in the data in the case of the stock market. Early knowledge of an upward or downward trend helps in investing money wisely. Pattern analysis also helps identify the best-performing companies for a specific period. This makes time series

analysis using deep learning models suitable for the nature of the variable we wish to predict **(Gourav , 2020, p. 211)**

Therefore, this study aims to explore the feasibility of using the Long Short-Term Memory (LSTM) as a type of the Recurrent Neural Networks (RNNs) to predict stock price movements in order to rationalize investment decisions in the financial markets.

1. The theoretical framework of the study.

Deep learning is an advanced branch of machine learning that relies on Artificial Neural Networks to simulate how the human brain processes information.

1.1 Artificial Neural Networks (ANNs)

Artificial Neural Networks (ANNs) fall within the fields of artificial intelligence and machine learning, and they are characterized by diverse architectures and patterns **(Bing , Hao , & Zhang , 2012, p. 1056)**. ANNs are inspired by biological learning systems and simulate their fundamental functions; they consist of complex networks of neurons organized into layers: an input layer, an output layer, and at least one hidden layer **(Ralf C & Eric, 2019, p. 02)**.

An artificial neural network is a machine designed to mimic the way the human brain performs a particular task or function. It is typically implemented using electronic components or simulated through software on a computer. Thus, ANNs are flexible, non-parametric modeling tools capable of performing complex functions **(Sabek & Saihi, 2021, p. 17)**.

This makes artificial neural networks well suited to this study and to the nature of the variable under investigation (stock prices), represented by opening and closing prices which are characterized by high randomness and non-parametric data. Artificial neural networks can function as universal approximators of complex functions and are capable of capturing nonlinear relationships between inputs and outputs. Moreover, ANNs offer several advantages in this field, as they can detect all possible interactions among input variables. This includes the use of training algorithms derived from back propagation techniques to enhance the predictive performance of the model.

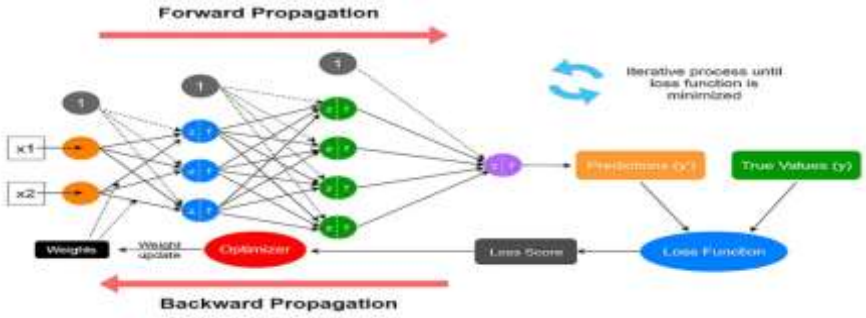
The process that employs gradient descent learning in artificial neural networks is known as back propagation. The training network consists of two main components: the input part and the output part. Initial weights are randomly selected within the range of −1 to 1.

The network's output depends on the input layer, hidden layers, network weights, and the activation function. The difference between the computed outputs and the actual outputs is referred to as the network error.

Back propagation takes this error and propagates it backward through the network, where the errors at each neuron are used to adjust the weights. This process is repeated iteratively until the overall network error reaches its minimum level **(Okasha & Yassen, 2013, p. 162)**.

The following figure illustrates the architecture of back propagation artificial neural networks:

Figure number (01) : architecture of the ANNs



Source : Omkar , O. (2024, 01 07). *Training ANNs: A deep dive into Backpropagation and Gradient descent*. Retrieved 03 27, 2026, from medium.com: https://medium.com/@omkarsoak/training-anns-a-deep-dive-into-backpropagation-and-gradient-descent-bea84ff5273c

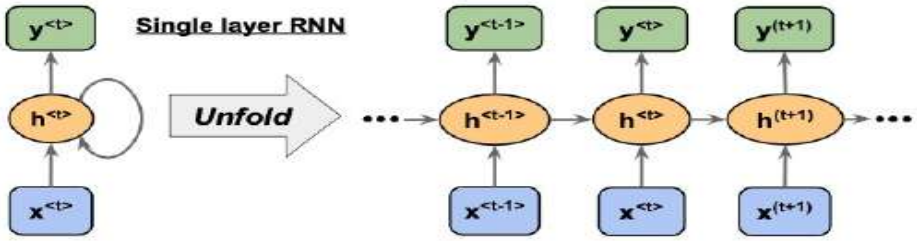
1.2 Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) are a type of artificial neural network that incorporate a feedback loop, allowing the output of the current node to be fed back into the input. RNNs enable the network to retain information from previous time steps and use it to predict future values. They apply a recurrent relationship at each time step to process sequences. RNNs also contain hidden states that act as memory stores for all computed information **(Ashish , Zeinab , & Devinder , 2024, p. 31)**.

RNNs are distinguished by their unique ability to process sequential data, making them indispensable in applications that require an understanding of time and order. They possess a form of memory that allows them to utilize information from previous inputs to guide future outputs. This is achieved through the integration of loops within the network, enabling information to persist across time steps. This feature allows RNNs to predict future events in a sequence based on the context of past events **(Partha, 2024, p. 49)**.

The following figure illustrates the architecture of the Recurrent Neural Network RNN:

Figure number (02): architecture of the RNNs



Source: Ashish , K., Zeinab , Z., & Devinder , K. (2024). Long Short-Term Memory (LSTM) Based Deep Learning Models for Predicting Univariate Time Series Data. *International Journal of Machine Learning, 14*(01), P. 31

The recurrent cell is the fundamental unit within Recurrent Neural Networks RNNs. Its primary function is to store information from previous time steps through what is known as the hidden state. The cell receives the current input x_t and the previous hidden state h_t, and then produces a new hidden state h_{t+1} and the output y_{t+1}.

The cell feeds the hidden state back into itself in the subsequent time step, enabling it to retain past inputs and capture temporal dependencies within sequences such as sentences, speech, and sequential data. Here, h denotes the hidden state, x the input, and y the output

The key feature of RNNs is that they use the same function with the same set of parameters at each time step. They are capable of modeling sequential data through propagation over time namely forward and backward propagation across units from the final processing stage to the initial one. This process aims to update the weights based on the accumulated error over time and to train the network to capture long-term dependencies in sequential data **(Khan & Yairi, 2018, p. 258)**.

Although Recurrent Neural Networks RNNs offer several advantages compared to other artificial neural networks, they are also prone to overfitting, where the model performs well on training data, but poorly on new unseen data. They may also suffer from the vanishing gradient problem, a phenomenon that prevents the network from learning long-term dependencies in the data **(Partha, 2024, p. 53)**.

In short-term models, the gradient of the loss function is used at each iteration to update the network parameters, enabling the model to learn. The gradient measures the extent to which the output of a function changes in response to a small change in the input. The vanishing gradient problem occurs when the gradient generated during back propagation fails to reach

earlier states and becomes extremely small, causing the learning process to slow down or even stop **(Ashish , Zeinab , & Devinder , 2024, p. 31)**.

Consequently, RNNs struggle to learn long-term dependencies and face limitations in short-term memory retention. In contrast, Long Short-Term Memory (LSTM) networks can effectively overcome these issues.

1.3 The Long Short-Term Memory (LSTM)

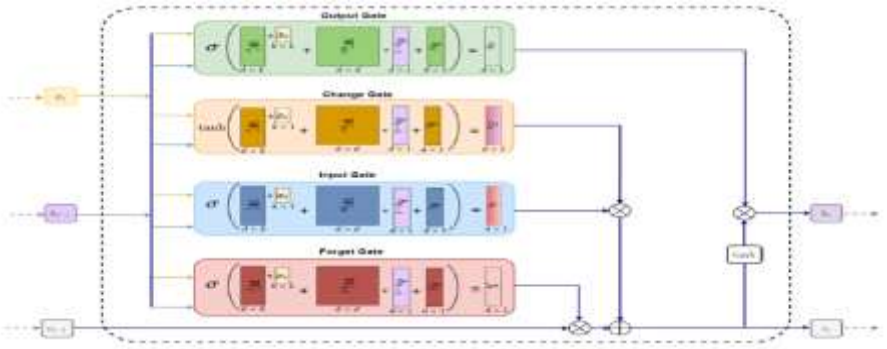
The Long Short-Term Memory network (LSTM) is a specific type of recurrent neural network RNN and has a wide range of applications similar to time series analysis, document classification, speech and audio recognition **(Nabipour, et al., 2020, p. 08)**. The LSTM model is derived from recurrent neural networks and operates on sequential data. Researchers proposed the LSTM model, which includes a memory unit capable of retaining a certain amount of training data to address the long-term dependency problem faced by RNNs. One of the advantages of LSTM is his ability to automatically learn when to forget information associated with a particular sequence **(Hossain, Rezaul , Ruppa Thulasiram, Neil , & Yang , 2018, p. 1838).**

Hochreiter and Schmidhuber developed the Long Short-Term Memory LSTM model in 1997. It differs fundamentally from traditional Recurrent Neural Networks RNNs in terms of his neuronal architecture. In traditional RNNs, there is no distinction between the individual neuron and the general neural network; however, in LSTM networks, each neuron acts as a memory cell that connects previous information to the current task **(Lai, Chen, & Caraka, 2019, p. 03)**. Studies have proven that the LSTM model outperforms most individual economic machine learning models and neural networks regarding time-series analysis **(Fausto & Giancarlo , 2025, p. 03)**.

The LSTM network effectively addresses the problems of vanishing and exploding gradients in traditional recurrent neural networks by introducing three gate units and one memory unit that selectively retain and discard historical information. The input gate controls the cell state at the current time step, while the forget gate regulates the cell state from the previous time step. The two values are then combined to obtain the current cell state value. Finally, the output gate controls the output of the cell state at the current time step **(Yuqi , Bingzhen , Juncheng , & Weiping , 2026, p. 06)**.

The following figure illustrates the architecture of the Long Short-Term Memory network:

Figure number (03): Architecture of the Long Short-Term Memory Network



Source : Bhandari, H. N., Rimal, B., Pokhrel, N. R., Rimal, R., Keshab, R., & Khatri, R. (2022). Predicting stock market index using LSTM. Machine Learning with Applications(09), p 04.

An LSTM cell consists of the input gate, cell state, forget gate, and output gate. It also includes a sigmoid layer, a tanh layer, and a point-wise multiplication operation **(Selvin, Vinayakumar , Gopalakrishnan, Vijay , & Soman, 2017, p. 1645)**.

The cell state functions as long-term memory, as it stores long-term dependencies and patterns. It also transfers important information across the data sequence, allowing information from an earlier time step to be carried forward to the final step in the sequence, thereby reducing the impact of short-term memory. Within the cell state, the gradient is preserved, which helps avoid the problems of vanishing or exploding gradients **(Ashish , Zeinab , & Devinder , 2024, p. 31)**.

The sigmoid function produces values between 0 and 1, where this value controls the flow of data: a value close to 0 means that nothing is passed through, while a value close to 1 means that everything is passed **(Sunny, Mirza, & Alharbi, 2020, p. 89)**.

The different gates and their functions are as follows:

- **Forget Gate:** The forget gate is the unit responsible for removing information from the cell state. It takes two inputs: h_{t-1} which is the hidden state of the previous cell, and x_t, which represents the input at the current time step. The weight matrices are multiplied by the inputs and added to the bias, after which the sigmoid function is applied. The output of the sigmoid function ranges between 0 and 1, corresponding to each value in the cell state. The sigmoid function determines which values should be retained and which should be discarded. After that, the

cell state is multiplied by the output vector of the sigmoid function **(Mahadik, Vaghela, & Mhaisgawali, 2021, p. 1596)**.

The output value of the forget gate is obtained after computation, as shown in the following equation **(Wenjie , Jiazheng , Yifan , Aijun , & Jingyang , 2020, p. 03)** :

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

Where:

- w_f : the weight of the forget gate
- h_{t-1}: the hidden state of the previous cell
- x_t: the input at the current time step
- b_f: the bias of the forget gate

❖ **Input Gate:** The input gate unit determines the information that should be stored in the cell. It consists of two parts. The first part is a sigmoid unit, which functions similarly to the forget gate and determines which information should be updated or retained. The second part uses the hyperbolic tangent (tanh) function to create a new candidate feature for the current time step **(Yuqi , Bingzhen , Juncheng , & Weiping , 2026, p. 07)**. The tanh function is centered around 0, with a range between -1 and +1, using the exponential base e **(Iqbal, Alnafisah, Akbar, & Sial, 2026, p. 07)**.

After performing element-wise multiplication between the new candidate feature of the current time step and the input gate, the remaining information for the current step is obtained. This remaining information is then added to the cell state from the previous time step to update the current cell **(Yuqi , Bingzhen , Juncheng , & Weiping , 2026, p. 07)**.

The outputs of the input gate are obtained as shown in the following equations **(Wenjie , Jiazheng , Yifan , Aijun , & Jingyang , 2020, p. 03)**:

$$i_t = \left(w_i . [h_{t-1}, x_t] + b_i\right)$$

$$\tilde{C}_t = \tanh(w_c . [h_{t-1}, x_t] + b_c)$$

Where:

- w_i: weight of the input gate
- b_i: bias of the input gate
- w_c: weight of the candidate input
- b_c: bias of the candidate input

The cell state is then updated as follows:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

- **Output Gate:** The output gate unit determines the information to be produced from the current cell state. This output depends on the cell state, which is controlled by the input and forget gates. First, the sigmoid function produces a value between 0 and 1 based on the current input and the hidden output from the previous time step. Then, the cell state is passed through the tanh function to transform the values into the range between −1 and 1. Finally, it is multiplied by the output of the sigmoid function, achieving a selective effect **(Yuqi , Bingzhen , Juncheng , & Weiping , 2026, p. 07)**.
  The output gate receives h_t and x_t as inputs at time step t, and the output of the output gate o_t is obtained as follows **(Wenjie , Jiazheng , Yifan , Aijun , & Jingyang , 2020, p. 03)**:

$$o_t = (w_o.[h_{t-1}, x_t] + b_o)$$

Where:

- w_o: weight of the output gate
- b_o: bias of the output gate

The LSTM network output h_t is then obtained by combining the output gate and the cell state, as shown in the following equation:

$$h_t = o_t * \tanh(C_t)$$

2. The experimental framework for the study.

This study relies on historical data covering the period from 2010 to 2025 for the stocks of three leading technology companies: Apple Inc., Tesla Inc., and Microsoft Corp. The data include the opening price, closing price, highest and lowest prices, trading volume, and adjusted closing price. These data were obtained from Yahoo Finance using the yfinance library in the Python environment. The dataset comprises a total of approximately 4,023 trading days for both Apple and Microsoft, and 3,901 trading days for Tesla.

To enhance the predictive capability of the model, six technical indicators were derived from the original data, namely:

- **Moving Averages (MA)** over periods of 7, 20, and 50 days to capture short-, medium-, and long-term trends;
- **Relative Strength Index (RSI)** over a 14-day period to measure trend strength and identify overbought and oversold conditions;
- **Bollinger Bands**, consisting of the middle, upper, and lower bands, to measure price volatility;
- **Moving Average Convergence Divergence (MACD)** to identify price momentum;
- **Trading Volume Indicator**, represented by the moving average of trading volume;
- **Volume Ratio**, used to confirm the strength of the trend

After removing missing values to ensure data quality and applying normalization to scale all values within a unified range between 0 and 1, the dataset was partitioned into 80% for training and 20% for testing. A look-back window of 60 trading days was adopted to train the LSTM model. The following table details the data partitioning for each company:

Table number (01): Study Data Partitioning

company	Training data	Testing data	Training data Shape	Testing data
Appel Inc	3229	808	(3169, 60 ,1)	(748, 60, 1)
TeslaInc	3132	783	(3072, 60 ,1)	(723, 60, 1)
Microsoft corp	3229	808	(3169, 60 ,1)	(748, 60, 1)

Source: Compiled by the author based on Google Colab model results.

The model architecture consists of three consecutive LSTM layers. The first and second layers each contain 100 neurons with sequence return enabled, while the third layer includes 50 neurons without returning sequences. To prevent overfitting, a dropout layer (20%) was added between each pair of LSTM layers. This is followed by two dense layers: the first with 25 neurons using the ReLU activation function, and the second with a single neuron using a linear activation function to output the predicted value.

The model was trained using the Adam optimizer with an initial learning rate of 0.001 and a Mean Squared Error (MSE) loss function. The batch size was set to 32 samples, and the model was trained for 100 epochs, with an early stopping mechanism implemented to monitor validation performance and automatically stop training if no improvement was observed for 20 consecutive epochs.

The model's performance was evaluated using Mean Absolute Error (MAE), Coefficient of Determination (R^2), and Mean Absolute Percentage

Error (MAPE). Additionally, trading signals were generated namely: Buy, Sell, and Hold decisions, based on the expected price changes and their direction, in order to guide investment decisions.

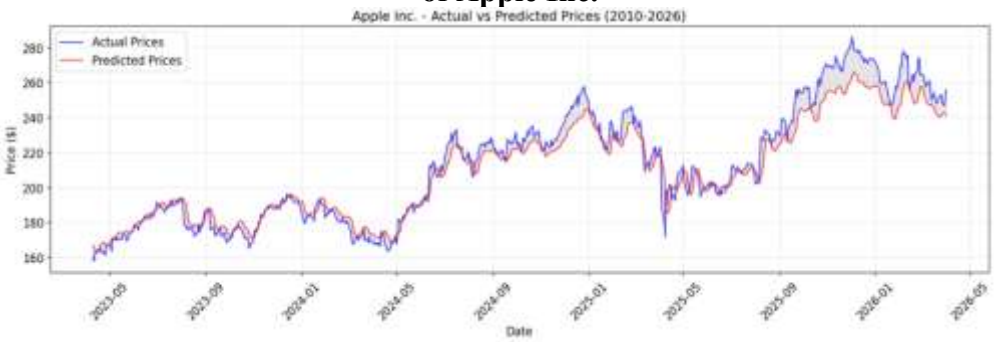
3. Results and Discussion.

Through this section, the results of forecasting the studied stock prices will be presented and evaluated, in addition to displaying the trading signals derived from the model.

3.1 Prediction Results

The following figures show a comparison between the predicted values and the actual values for each of the three models:

Figure number (04): Comparison of Actual and predicted Stock Prices of Apple Inc.



Source: Model outputs generated via Google Colab.

The figure 04 illustrates a comparison between the actual prices and the predicted prices of Apple Inc. stock throughout the study period. A clear convergence between the two curves is observed, reflecting the LSTM model's capability to accurately track the general trend of price movements. This close alignment between actual and predicted prices, notwithstanding minor discrepancies, underscores the model's sensitivity to sharp and stochastic price fluctuations.

The recorded values indicate that the model is adept at detecting both upward and downward trends, as it successfully synchronized predicted values with actual data with an absolute difference of 5$ and an error rate of approximately 2%. Nevertheless, it is noted that the model exhibited a slight tendency toward overestimation during periods of rapid appreciation, rendering it marginally less responsive to acute volatility and sudden market downturns

Figure number (05): Comparison of Actual and Predicted Stock Prices of Tesla Inc



Source: Model outputs generated via Google Colab.

The figure 05 represents a comparison between the actual and predicted prices of Tesla Inc. stock during the study period. The general trajectory of both curves demonstrates a high degree of consistency between the actual and predicted values, indicating substantial statistical efficiency in simulating the fluctuations that characterized the stock's price movement. This is particularly evident during the price surges that surpassed the 450 $ threshold in late 2024 and 2025. Overall, the model successfully tracks the general price trend with high precision, while recording minimal standard deviations during periods of relative stability.

Figure number (06): Comparison of Actual and Estimated Stock Prices for Microsoft Corp.



Source: Model output generated via Google Colab.

The figure 06 presents a comparative temporal analysis between the actual and predicted prices of Microsoft Corp. stock throughout the study period. The predictive model demonstrates a robust capability in simulating the general trends of the stock price trajectory; however, it exhibits a noticeable deviation gap during the period from May 2025 to January 2026. During this interval, actual prices consistently exceeded projections, indicating a state of buying momentum that surpassed the conservative

mathematical modeling. In general, the model is considered capable of tracking the overall price trend, provided that the error coefficient resulting from unforeseen fluctuations is taken into account when analyzing these data.

3.2 Evaluation of the Model Results

The evaluation results of the Long Short-Term Memory (LSTM) models, trained on stock data for Apple, Tesla, and Microsoft from 2010 to 2026, demonstrated varying performance levels across different evaluation metrics. These results are detailed in the following table:

Table number (02): Evaluation Results of Long Short-Term Memory (LSTM) Models

Stock	R^2 Score	RMSE ($)	MAE ($)	MAPE (%)	MSE
Apple Inc.	0.9384	8.1722	6.1757	2.7539	66.7852
Tesla Inc.	0.9796	12.8904	10.2277	3.7721	166.1623
Microsoft Corp.	0.9382	15.3059	12.2568	2.9187	234.2692

✓ Saved: detailed_metrics.csv

Source: Model outputs generated via Google Colab.

The evaluation results of the Long Short-Term Memory (LSTM) models, trained on stock data for the studied companies from 2010 to 2026, demonstrated overall strong performance. The Coefficient of Determination (R^2) values ranged between 0.938 and 0.979, indicating a high explanatory power that accounts for up to 97.9% of the variance in actual prices. Such results are considered exceptional in the field of financial markets. Tesla stock achieved the highest (R^2) value at 0.979, despite its high volatility, followed by Apple and Microsoft, both recording 0.938.

Furthermore, the Mean Absolute Percentage Error (MAPE) recorded low values, ranging from 2.753% for Apple to 3.772% for Tesla and 2.918% for Microsoft. Since all these values remain below the 5% threshold, they are considered an excellent benchmark in financial forecasting.

Table number (03): Model Training Efficiency Results

Stock	Actual Epochs	Best Epoch	Best Validation Loss	Final Validation Loss	Train Improvement (%)	Val Improvement (%)
Apple Inc.	79/100	49	0.000856	0.003318	96.5%	77.5%
Tesla Inc.	86/100	56	0.000696	0.002138	96.3%	51.8%
Microsoft Corp.	41/100	11	0.000858	0.006243	94.3%	40.5%

✓ Saved: convergence_analysis.csv

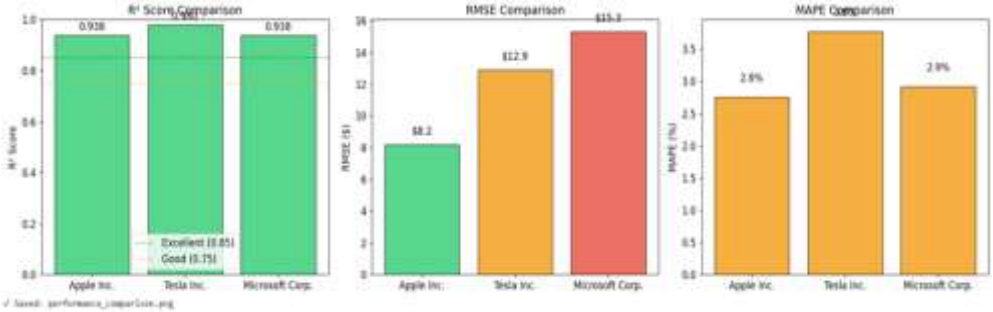
Source: Model outputs generated via Google Colab.

The models demonstrated high training efficiency, as the Early Stopping mechanism was triggered at epoch 41 for Microsoft, 79 for Apple, and 54 for Tesla out of a total of 100 epochs. This reflects the models' ability to rapidly learn price patterns without the need for full training cycles. Furthermore, the Loss Function values remained low, ranging between 0.000696 and 0.000858, which confirms the absence of overfitting issues.

Based on these results, it can be concluded that LSTM models have proven highly efficient in forecasting the price movements of the studied stocks, particularly for Apple and Microsoft. However, it is recommended to exercise caution when applying the model to highly volatile stocks such as Tesla.

The following figure illustrates the performance comparison between the three models:

Figure number (07): Model Performance Comparison



Source: Model outputs generated via Google Colab.

3.3 Trading Signal Results Derived from LSTM Models

By leveraging the LSTM model to forecast the price movements of the studied companies, specific trading signals namely: Buy, Sell, or Hold, were extracted. these signals are based on the expected price change and its trend direction, with the objective of guiding investment decisions within the financial market.

The following table summarizes these results:

Table number (04): Trading Signals Table

Stock	Symbol	Current Price	Predicted Price	Expected Change	Trend	Signal	Action	Confidence	Model R²
Apple Inc.	AAPL	$255.63	$241.12	-5.47%	Downward	Strong Sell	SELL	54.7%	0.938
Tesla Inc.	TSLA	$381.26	$369.37	-2.04%	Downward	Sell	SELL	70.0%	0.980
Microsoft Corp.	MSFT	$369.37	$382.52	5.20%	Downward	Hold	HOLD	50.0%	0.938

✓ Saved: trading_signals.csv

Source: Model outputs generated via Google Colab.

The results presented in the table above indicate that the Coefficient of Determination (R^2) is relatively high for all three studied companies, ranging between 0.938 and 0.980. This demonstrates that the three models are capable of explaining more than 93% of the variance in the studied stock prices, while approximately 7% of the variance is attributed to other factors that the models cannot account for—providing clear evidence of the models' efficiency.

The first model recorded a projected decline in Apple Inc. shares by 5.47%, while the second model anticipated a 2.04% decrease in Tesla Inc. stock prices; these projections justify the "Sell" signals generated by the both models. Conversely, the third model for Microsoft Corp. issued a "Hold" signal, despite an expected positive price change of 5.20%. This decision is attributed to a lower confidence level of 50%, which represents a critical threshold that necessitates a cautious stance. This contrasts with the higher confidence levels observed in the Tesla and Apple forecasting models, which reached 70% and 54.7%, respectively.

Conclusion

This study investigates the application of Long Short-Term Memory (LSTM) recurrent neural networks for forecasting the price movements of three leading U.S. technology stocks: Apple Inc., Tesla Inc., and Microsoft Corp. The analysis was based on daily trading data covering the period from 2010 through the first quarter of 2026.

The proposed LSTM model comprises a three-layer stacked architecture with 100, 100, and 50 units in each layer, respectively. To mitigate overfitting, a dropout layer with a rate of 20% was applied between consecutive LSTM layers. The model was trained over 100 epochs, employing early stopping and an adaptive learning rate reduction strategy to optimize predictive performance.

This study demonstrated the ability of LSTM models to capture complex patterns and temporal dependencies in stock price movements, confirming that deep learning techniques are suitable for financial time series analysis. The results showed that the trained models achieved varying levels of performance, which is attributed to differences in volatility and stability levels of the studied stock prices, explaining the inverse relationship between volatility levels and prediction accuracy. Furthermore, the learning curves exhibited remarkable convergence across all models, as the loss function value decreased sharply during the first twenty epochs, followed by a gradual stabilization period, indicating the model's ability to learn effectively without encountering the problem of overfitting.

In contrast, the trading signals extracted from the three models offer practical value to stock market investors as a decision-support tool for their investment choices. The models generated sell recommendations for both Apple and Tesla shares due to the downward trend in the movement of these stocks. Meanwhile, the model for Microsoft Corporation shares indicated a hold recommendation despite the upward price trend, which is attributed to his low confidence level.

Bibliographie

Ashish , K., Zeinab , Z., & Devinder , K. (2024). Long Short-Term Memory (LSTM) Based Deep Learning Models for Predicting Univariate Time Series Data. *International Journal of Machine Learning, 14*(01).

Azhar, I. B., Imamul Haque, S., Umer, J. G., & Ishfq, G. (2021). Decision Making in Financial Markets: A Thematic. *03*(1-2), 49.

Bhandari, H. N., Rimal, B., Pokhrel, N. R., Rimal, R., Keshab, R., & Khatri, R. (2022). Predicting stock market index using LSTM. *Machine Learning with Applications*(09).

Bing , Y., Hao , J., & Zhang , S. (2012). Stock Market Prediction Using Artificial Neural Networks. *Advanced Engineering Forum, 6-7*, 1055.

Fausto , R., & Giancarlo , S. (2025). An Advisor Neural Network framework using LSTM-based Informative Stock Analysis. *Expert Systems With Applications*(259).

Gourav , B. (2020). Stock Price prediction using LSTM and SVR. *2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC).*

Gourav , B., Rinkle , R., & Himanshu , A. (2023). Stocks of year 2020: prediction of high variations. *Multimedia Tools and Applications*. Retrieved from https://doi.org/10.1007/s11042-022-12390-5

Hossain, M. A., Rezaul , K., Ruppa Thulasiram, T., Neil , D., & Yang , W. (2018). Hybrid Deep Learning Model for Stock Price. *2018 ieee symposium series on computational intelligence (ssci).*

Iqbal, J., Alnafisah, H., Akbar, M. H., & Sial, M. S. (2026). Predicting Stock Market Returns Using Sentiment in Business News Articles: An LSTM Machine Learning Approach. *SAGE Open, 16*(01).

Khan, S., & Yairi, T. (2018). A review on the application of deep learning in system health management. *Mechanical Systems and Signal Processing*(107).

Konstantinou, R. (2017). Stock Market prediction using Artificial Neural. Department of Computer Science and Engineering, Gothenburg, Sweden: Chalmers University of Technology.

Lai, C. Y., Chen, R. C., & Caraka, R. E. (2019). Prediction Stock Price Based On Diffrent Index Factors Using LSTM . *International conference on machine learning and cybernetics (ICMLC).*

Mahadik, A., Vaghela, D., & Mhaisgawali, A. (2021). Stock Price Prediction using LSTM and ARIMA. *Proceedings of the Second International Conference on Electronics and Sustainable Communication Systems (ICESC).*

Nabipour, M., Nayyeri, P., Jabani, H., Mosavi, A., Salwana, E., & Shahab , S. (2020). Deep Learning for Stock Market Prediction. *Entropy, 22*(8).

Okasha, M., & Yassen, A. (2013). The Application of Artificial Neural Networks In Forecasting Economic Time Series. *International Journal of Statistics and Analysis, 03*(02).

Omkar , O. (2024, 01 07). *Training ANNs: A deep dive into Backpropagation and Gradient descent*. Retrieved 03 27, 2026, from medium.com: https://medium.com/@omkarsoak/training-anns-a-deep-dive-into-backpropagation-and-gradient-descent-bea84ff5273c

Partha, M. (2024). *Deep Learning for Managers, Transforming leadership in the ai era.* Copyright.

Ralf C, S., & Eric, R. M. (2019). Understanding LSTM a tutorial into Long Short-Term Memory. *arXiv*. Retrieved from https://arxiv.org/abs/1909.09586

Sabek, A., & Saihi, Y. (2021). Using Artificial Neural Network to predict the financial distress: the case of some. *Journal Of North African Economies, 17*(27).

Selvin, S., Vinayakumar , R., Gopalakrishnan, E., Vijay , K. M., & Soman, K. (2017). STOCK PRICE PREDICTION USING LSTM,RNN AND CNN-SLIDING WINDOW MODEL. *international conference on advances in computing, communications and informatics.*

Sunny, M., Mirza, M., & Alharbi, A. (2020). Deep Learning-Based Stock Price Prediction Using LSTM and Bi-Directional LSTM Model. *2nd Novel Intelligent and Leading Emerging Sciences Conference.*

Wenjie , L., Jiazheng , L., Yifan , L., Aijun , S., & Jingyang , W. (2020). A CNN-LSTM-Based Model to Forecast Stock Prices. *Complexity, 1*.

Yuqi , G., Bingzhen , S., Juncheng , B., & Weiping , D. (2026). Hybrid stock price forecasting model using LSTM of attention mechanism combining noise removal and neighborhood rough set. *Applied Intelligence*.