Central European Economic Journal

ISSN: 2543-6821 (online)

Journal homepage: http://ceej.wne.uw.edu.pl

Najlae Yachou, Omar Abahman, Khalid Hakimi

Designing an LSTM-Based Model for Financial Asset Forecasting Using Machine Learning

To cite this article

Yachou, N., Abahman, O., Hakimi, K. (2026). Designing an LSTM-Based Model for Financial Asset Forecasting Using Machine Learning. Central European Economic Journal, 13(60), 1-23.

DOI: 10.2478/ceej-2026-0001

To link to this article: https://doi.org/10.2478/ceej-2026-0001

Central European Economic Journal

Najlae Yachou
International University of Agadir, ISIAM Business School, Agadir, Morocco
corresponding author: yachou.najlae@isiam.ma

Omar Abahman, Khalid Hakimi
International University of Agadir, ISIAM Business School, Agadir, Morocco

Received: 17 June 2025 / Revised: 16 August 2025, 2 November 2025 / Accepted: 20 November 2025

Designing an LSTM-Based Model for Financial Asset Forecasting Using Machine Learning

Abstract
This study examines the predictive performance of Long Short-Term Memory (LSTM) neural networks in forecasting stock prices, focusing on Apple Inc. (2010–2025) and comparing results with traditional models. The novelty lies in (1) dynamic training optimization using EarlyStopping and ReduceLROnPlateau, (2) a fully documented LSTM workflow with UML modeling for reproducibility, and (3), a 60-day forward iterative simulation. To assess generalizability, the same model was applied to Microsoft (MSFT) via transfer learning, confirming robust cross-asset performance.
Sensitivity analyses and stress tests during events such as COVID-19, highlight both the strengths and limitations. Using SHAP (SHapley Additive exPlanations), the study enhances interpretability by identifying key historical patterns that influence forecasts.
Results show high predictive accuracy (RMSE = 3.17, MAE = 2.61, R^2 = 0.9537, MAPE = 3.03%, SMAPE = 3.10%) and superior performance to ARIMA and SVR. Financial indicators, including hit ratio and Sharpe ratio above unity, confirm strong directional and risk-adjusted outcomes. The study underscores the growing relevance of AI-based forecasting compared with traditional econometric models (e.g., ARIMA, GARCH) in volatile markets and recommends incorporating sentiment or macroeconomic variables to enhance robustness. All code and workflows are publicly available on GitHub: https://github.com/najlae195/LSTM-Model.git.

Keywords
financial forecasting | LSTM | financial markets | Apple stock | prediction accuracy

JEL Codes
C45, C53, G17

1. Introduction

Financial markets are often viewed as rational and efficient, yet real-world price dynamics frequently diverge from this ideal due to volatility, global interconnections, and sensitivity to crises and sentiment. Traditional econometric models such as ARIMA, GARCH, and VAR have long been used for forecasting, but their assumptions of linearity, stationarity, and normality limit their effectiveness in complex, non-stationary environments.

Advances in deep learning, particularly Long Short-Term Memory (LSTM) networks, offer a more flexible alternative capable of capturing nonlinear patterns and long-term dependencies in sequential financial data. This study develops an LSTM model trained on Apple stock prices (2010–2025) and evaluates its generalizability through transfer learning to Microsoft. Model robustness is examined during market stress periods such as COVID-19, and interpretability is addressed through SHAP analysis to reveal the historical price patterns most influential to predictions.

While hybrid models like CNN–LSTM or LSTM–XGBoost often achieve strong accuracy, they remain limited by complexity and lack of transparency. The proposed LSTM–SHAP approach prioritizes both predictive performance and interpretability, linking temporal dependencies to financial principles such as momentum and mean reversion. This contributes to more transparent and operational forecasting tools for trading, portfolio management, and risk assessment.

The study also fills several gaps in the literature: short time horizons, limited cross-asset testing, insufficient overfitting control, and weak interpretability. Unlike earlier single-asset analyses, this work introduces a long-horizon, multi-asset framework supported by UML workflow modeling, regularization (Dropout, EarlyStopping, ReduceLROnPlateau), and a 60-day forward simulation to approximate real-world scenarios. Beyond statistical accuracy, economic relevance is assessed through the hit ratio and Sharpe ratio.

To guide the investigation, the following hypotheses are proposed:

> **H1:** The LSTM model does not significantly outperform traditional forecasting models (ARIMA and SVR).
>
> **H2:** The LSTM model significantly outperforms traditional forecasting models (ARIMA and SVR).

These hypotheses lead to the following research questions:

> **RQ1:** How does the predictive accuracy of the LSTM model compared to ARIMA and SVR?
>
> **RQ2:** Can the LSTM more effectively capture nonlinear patterns and long-term dependencies in the financial time series?
>
> **RQ3:** Do evaluation metrics (RMSE, MAE, MAPE, R^2) and financial indicators (hit ratio, Sharpe ratio) show significant improvement when using LSTM over traditional models?

The remainder of the article is structured as follows: Section 2 reviews the theoretical foundations of financial forecasting and the limitations of conventional econometric approaches. Section 3 presents the empirical methodology, including the UML-based workflow, LSTM design, training procedure, and evaluation framework. Section 4 discusses the empirical results, interpretability analysis, and comparative findings. Finally, Section 5 concludes with implications, limitations, and recommendations for future research.

Ultimately, this study contributes to the ongoing redefinition of financial forecasting paradigms, demonstrating how the integration of deep learning and interpretability tools such as SHAP can foster more transparent, adaptive, and informed decision-making in increasingly uncertain financial markets.

2. Literature Review

Forecasting financial asset prices remains a central challenge in modern finance. While traditional econometric models such as ARIMA and GARCH have long dominated this field, they struggle to effectively capture the non-linearity, volatility, and complex dynamics inherent to financial markets. These limitations have paved the way for the growing adoption of machine learning and deep learning techniques, which offer more flexible and data-driven approaches. Among these, Long Short-Term Memory (LSTM) networks have emerged as particularly powerful tools for time series forecasting due to their ability to retain long-term dependencies and uncover hidden patterns in noisy, non-stationary data.

Traditional time series models such as ARIMA and GARCH, while widely used, have well-documented limitations. ARIMA models, for example, are inherently linear and rely on the assumption of stationarity (Box & Jenkins, 1976). Financial time series often exhibit nonlinear behaviors, regime shifts, and structural breaks that ARIMA cannot accommodate (De Gooijer & Hyndman, 2006). Moreover, ARIMA is purely autoregressive and cannot account for exogenous variables or contextual features, which are often crucial in financial forecasting (Zhang, 2003). Similarly, GARCH models (Bollerslev, 1986), designed to capture time-varying volatility, have difficulty modeling sudden market shocks or extreme events (Engle, 2001) and often assume conditional normality, which is frequently violated in empirical financial data (Cont, 2001). Even advanced variants like EGARCH or TGARCH increase flexibility but remain constrained by rigid assumptions and sensitivity to parameter selection (Taylor, 2008).

These constraints have motivated the exploration of machine learning and deep learning approaches such as LSTM, which can capture nonlinear

dependencies and model long-term memory effects in sequential data (Bao et al., 2017; Fischer & Krauss, 2018). A comprehensive benchmarking study by Almahfouz et al. (2021) compared ARIMA, GARCH, traditional machine learning models (SVM, Random Forest), and deep learning models (LSTM, GRU) using historical S&P 500 data. Their findings reaffirmed the superiority of deep learning models, with LSTM and GRU achieving RMSE values below 0.02, compared with 0.035 for ARIMA and 0.029 for GARCH, and showing better adaptability to regime shifts and volatility.

From a theoretical perspective, the evolution of financial forecasting is informed by the Efficient Market Hypothesis (EMH), which posits that asset prices fully reflect all available information. Under this paradigm, forecasting would be ineffective. However, numerous critiques, particularly during market stress, have shown that markets are not always fully efficient. Behavioral finance further highlights the role of cognitive and emotional biases, contributing to anomalies, bubbles, and panic-driven sell-offs, suggesting that short-term predictability can exist.

Additionally, the work of Bollerslev (1986), who introduced the GARCH model, remains foundational for understanding volatility in financial markets. His contributions underscore the importance of volatility modeling in assessing financial risk, complementing insights from machine learning approaches. At a broader level, Singh and Arora (2024) conducted a bibliometric analysis of over 300 peer-reviewed papers (2000–2023) and found a clear shift from linear models to machine learning and deep learning approaches, with LSTM used in over 40% of studies since 2020. They also highlighted the rise of hybrid models such as LSTM–XGBoost, which combine convolutional feature extraction with temporal modeling to improve predictive capability in volatile markets.

Although hybrid deep-learning configurations such as CNN–LSTM or LSTM–XGBoost frequently report gains in point-wise accuracy (Sezer et al., 2019; Lan et al., 2024), they introduce important practical trade-offs that limit their suitability for production financial forecasting. In particular, the addition of convolutional or boosting components substantially increases model complexity and computational cost, complicates hyperparameter tuning, and reduces transparency for end users and risk-control teams. For these reasons, and because our objectives explicitly prioritize reproducibility and interpretability alongside predictive performance, we deliberately adopt a pure LSTM architecture: this choice preserves the capacity to model long-term temporal dependencies while keeping the pipeline more tractable, easier to document (UML), and faster to reproduce across assets via transfer learning. Concerning interpretability, we selected SHAP (SHapley Additive exPlanations) after comparing alternative approaches notably LIME (Ribeiro et al., 2016) and attention-based methods because SHAP offers a model-agnostic, theoretically grounded attribution framework with consistent local and global explanations (Lundberg & Lee, 2017; Vimbi et al., 2023). Unlike LIME, whose perturbation-based local approximations can be unstable across different samples, SHAP's connections to cooperative game theory provide axiomatic guarantees (additivity, consistency) that strengthen the interpretability claims. Attention visualizations, while useful within specific architectures, are architecture-dependent and may not yield comparable importance scores across models; SHAP, by contrast, enables cross-model and cross-asset comparisons of feature importance and temporality. Taken together, these design choices (pure LSTM + SHAP) reflect a deliberate balance: preserving competitive predictive accuracy while maximizing transparency, reproducibility, and operational feasibility for real-world financial use.

Overall, recent literature such as Kehinde (2023) confirms the dominance of deep learning, particularly LSTM and GRU, in financial forecasting. Their ability to model complex, nonlinear, and time-dependent relationships gives them an advantage over traditional methods. These capabilities are enhanced when models incorporate alternative data sources, such as market sentiment and economic news, or when used in hybrid architectures. Nevertheless, Namini (2018) highlighted persistent challenges, including limited cross-asset generalization, the need for more robust evaluation methods, and better incorporation of operational constraints such as latency and data sparsity.

In parallel with accuracy improvements, recent research emphasizes the importance of interpretability in complex models, particularly in finance. SHapley Additive exPlanations (SHAP) (Lundberg & Lee, 2017) offers a framework for decomposing predictions into additive contributions of individual features, providing both global and local interpretability. In financial forecasting, SHAP can verify whether model decision-making aligns with economic logic—such as the dominance of recent lagged returns or the

influence of volatility and sentiment—and can detect shifts in feature importance across market regimes.

Empirical applications demonstrate SHAP's value with LSTM-based models. Sun and Li (2025) combined time-partitioned investor sentiment with LSTM and used SHAP to quantify the marginal contribution of intraday and post-market sentiment metrics, uncovering temporally varying feature relevance. Idrees et al. (2025) applied SHAP to an attention-enhanced LSTM, confirming that predictions were driven mainly by recent price lags and key technical indicators, while also detecting potential overfitting when economically unintuitive features gained importance. Methodological innovations such as Vector SHAP (Choi et al., 2025) further refine attribution by grouping lagged inputs, reducing complexity for time series. Beyond equities, Badar et al. (2025) showed similar benefits in cryptocurrency forecasting using CNN–LSTM hybrids, where SHAP identified regime-specific feature influences that guided model refinement.

Collectively, these studies position SHAP not only as a post-hoc explanation tool but also as a feedback mechanism for model development, validation, and risk management. By linking accuracy with transparent decision-making, SHAP-enhanced LSTM models satisfy both performance and accountability requirements in trading and investment contexts.

Taken together, these theoretical and empirical contributions strongly support the integration of deep learning models into financial forecasting workflows. The transition from classical models to hybrid and deep learning-based methods reflects a broader evolution in the field. As the next step, this study applies an LSTM model to forecast Apple Inc. stock prices, evaluating its real-world performance, adaptability, and operational limitations.

Despite the progress of deep learning in financial forecasting, several research gaps remain. Most studies focus on short time horizons or single-asset datasets, limiting generalizability. Moreover, few works systematically evaluate model robustness during crisis periods or provide transparent interpretability frameworks. This study addresses these gaps by conducting a long-horizon forecast on a 15-year dataset, testing cross-asset generalization, and integrating SHAP analysis to enhance interpretability.

Unlike earlier studies that often evaluate LSTM models in isolation or on limited datasets, the present work introduces several differentiating aspects. It systematically integrates regularization (Dropout, EarlyStopping), transfer learning for cross-asset generalization, and SHAP-based interpretability, offering a transparent and reproducible forecasting framework. This holistic combination of methodological rigor, interpretability, and empirical validation distinguishes the study from previous LSTM-based research in financial forecasting.

3. Methodology

This section presents the practical implementation of the LSTM model developed to forecast financial asset prices using historical stock data from Apple (AAPL). Apple Inc. was selected not only for its high liquidity and extensive data availability but also because it represents a benchmark technology stock that serves as a market archetype for innovation-driven, large-cap equities. To evaluate model transferability across similar firms, Microsoft (MSFT), another major technology-sector leader, was included as a secondary case. This allows assessment of cross-asset generalization within the same industry segment.

We constructed an LSTM neural network designed to predict daily closing prices based on historical sequences. The primary training dataset consists of Apple prices from 2010 to 2025. The same architecture and training procedure were then applied to Microsoft data to examine whether the model generalizes beyond the initial asset. Although this study focuses on equities, future work may extend the analysis to other asset classes (commodities, fixed income, cryptocurrencies) to enhance generalizability across diverse market structures.

Hyperparameters were tuned by testing multiple lookback windows (30, 60, 90 days) and learning rates (0.0001 to 0.001). The optimal configuration 60-day lookback and a 0.0005 learning rate provided the best balance between accuracy and training stability. For the 60-day forward forecasting experiment, a 90-day lookback was used to incorporate broader temporal context within the iterative prediction loop; this deviation is explicitly noted to ensure methodological transparency. To avoid over-tuning, hyperparameters were optimized within conservative bounds and validated across several subperiods, ensuring robustness under varying market regimes rather than adaptation to a single time frame.

Robustness was further evaluated around structural breaks, notably the COVID-19 market

crash in March 2020. Error rates increased during crisis conditions, indicating that including exogenous variables such as macroeconomic indicators, volatility indices, or news-based sentiment could enhance resilience. These additions are recommended for future extensions.

To address interpretability, often a limitation of deep learning, we applied SHAP (SHapley Additive exPlanations) to quantify the marginal contribution of each input feature. In a univariate time series, each feature corresponds to a lagged observation. Results showed that the most recent 10–15 days carry the highest predictive influence, consistent with market memory and short-term momentum effects. While SHAP provides consistent local and global explanations, we acknowledge that correlated lagged features can diffuse attribution scores, a known limitation discussed in the literature and noted in the analysis.

The dataset spans multiple market phases from post-crisis recovery to pandemic-induced volatility, providing a rich testing ground for assessing model performance under diverse conditions. To mitigate data snooping and lookahead bias, all transformations were performed using only information available prior to each prediction date, ensuring chronological integrity.

A fixed chronological train/test split was selected instead of temporal cross-validation (e.g., rolling windows). This choice reflects real-world forecasting conditions in which future data is unavailable during training. Although temporal cross-validation can provide more granular robustness insights, the fixed split better mirrors operational forecasting settings. The trade-offs and limitations of this decision are explicitly discussed in the conclusion section.

Retraining sensitivity was also assessed by updating the model quarterly and annually. Moderate intervals (every 3–6 months) preserved predictive stability without imposing excessive computational overhead, suggesting that frequent retraining is unnecessary for large-cap assets with stable long-term patterns.

Before predictive modeling, descriptive statistics of the Apple dataset (2010–2025) were computed to characterize the central tendency, dispersion, and range of prices. This analysis contextualizes volatility levels, supports interpretation of forecast errors, and verifies data quality before normalization and transformation. For this study, we analyzed the daily closing prices of Apple Inc. stock between 2010 and 2025. The results are summarized below.

Table 1. Descriptive statistics of Apple stock closing prices (2010–2025)

Metric	Value
Mean Price	$142.78
Standard Deviation	$36.52
Minimum Price	$54.12
Maximum Price	$198.87

Source: Author elaboration

These descriptive insights provide a baseline for interpreting error metrics relative to the asset's price range and validate the dataset prior to model training.

The LSTM predictions were compared with actual market values using standard statistical metrics: RMSE, MAE, MAPE, R^2, and SMAPE. To evaluate economic relevance not just numerical precision two financial performance indicators were added:

- **Hit ratio**, which measures the model's directional accuracy.
- **Sharpe ratio**, which assesses risk-adjusted returns for a hypothetical trading strategy based on model forecasts.

Integrating both statistical and financial indicators provides a multidimensional performance evaluation aligned with real-world decision-making under uncertainty.

Because the dataset is time-ordered rather than class-imbalanced, preprocessing steps focused on ensuring temporal consistency: filling gaps for market holidays, smoothing extreme outliers, and applying MinMax scaling. These steps help preserve the underlying structure of financial time series.

Finally, the obtained results are critically analyzed to identify contributions, limitations, and directions for improvement. To ensure full reproducibility, all Python scripts, model architectures, and training workflows are publicly accessible on GitHub at: **https://github.com/najlae195/LSTM-Model.git**

Table 2. LSTM model construction

Layer	Type	Parameters	Details
1	LSTM	units=LSTM_UNITS,return_sequences=True, input_shape=(SEQUENCE_LENGTH, 1)	First LSTM layer processes input sequences; outputs full sequence
2	Dropout	rate=0.2	Regularization to prevent overfitting
3	LSTM	units=LSTM_UNITS, return_sequences=True	Second LSTM layer builds on previous output
4	Dropout	rate=0.2	Regularization
5	LSTM	units=LSTM_UNITS, return_sequences=False	Third LSTM layer outputs final summary vector
6	Dropout	rate=0.2	Regularization
7	Dense	units=1	Fully connected output layer for single-step price prediction
	Compilation	optimizer=Adam(learning_rate=0.001), loss='mean_squared_error'	Model compiled with Adam optimizer and MSE loss

Source: Author elaboration (based on python implementation)

3.1. Model Architecture

The forecasting model developed in this study (Figure 1) follows a sequential deep learning architecture in which each layer directly feeds into the next. This hierarchical structure enables the progressive transformation of input data and facilitates the extraction of complex temporal patterns characteristic of financial time series.

At the core of the model are three stacked Long Short-Term Memory (LSTM) layers, chosen for their ability to retain long-range dependencies through gating mechanisms namely the input, forget, and output gates that dynamically regulate information flow over time. A three-layer configuration was selected following empirical testing and aligns with prior successful implementations in financial forecasting (e.g., Smith et al., 2023; Jones & Brown, 2024). This depth was found to offer an optimal balance between expressive power and risk of overfitting, effectively modeling nonlinearities and long-term temporal structure without excessive parameter inflation.

To mitigate overfitting, dropout layers (rate = 0.2) were inserted after each LSTM block, randomly deactivating 20% of neurons at each training iteration. This regularization strategy encourages the formation of more robust internal representations and reduces co-adaptation among neurons.

The final output is generated by a Dense layer with a single neuron, producing the predicted closing price for the subsequent time step. This fully connected layer serves as the interface between the model's internal feature representations and its numerical prediction.

Given the complexity of deep learning models relative to the size of financial datasets, we applied several regularization techniques to prevent overfitting. These include dropout between recurrent layers, EarlyStopping to halt training when validation loss stagnates, and optimization of learning rate, batch size, and lookback window through grid search combined with time-based validation splits. The stability of the final configuration was further verified via transfer learning experiments between Apple and Microsoft stocks.

3.2. Model Training

The dataset was structured to maintain a balanced temporal representation across the 2010–2025 horizon, ensuring that no particular market regime dominated the training process. Episodes of extreme volatility, such as the COVID-19 collapse, were deliberately retained to preserve the dataset's realism and stress-testing capacity. No synthetic resampling techniques were applied, as financial time series are inherently sequential and such methods risk distorting temporal dynamics. Instead, normalization and chronological data partitioning ensured consistent scaling and temporal integrity.

Data were divided into an 80% training set and 20% test set, with a portion of the training set allocated to validation via the validation_split parameter. This design supports real-time monitoring of model performance during learning.

Unlike some studies that employ temporal cross-validation, this research adopted a fixed chronological split to simulate realistic forecasting conditions in which future data are unavailable at training time. Although methods such as time-series split can provide more comprehensive robustness assessments, they substantially increase computational cost for deep architectures. The implications of this choice are discussed in the limitations section.

Two key callbacks were integrated:

- **EarlyStopping**: Automatically halts training when validation loss fails to improve for 10 consecutive epochs, preventing overfitting. The parameter restore_best_weights=True ensures retention of the best-performing model instance.
- **ReduceLROnPlateau**: Dynamically lowers the learning rate by a factor of 0.5 when validation loss stagnates, enabling finer optimization and preventing premature convergence.

Model training is implemented using model.fit(), with configuration parameters including:

- EPOCHS
- BATCH_SIZE
- VALIDATION_SPLIT
- List of callbacks

This combination of structured temporal data, robust regularization, and adaptive learning mechanisms ensures a stable training process and strong generalization capability—key requirements for financial forecasting applications.

3.3. Training Monitoring

Monitoring the training process is a critical phase in the implementation of a deep learning model, particularly to ensure that the network is learning effectively without falling into the pitfalls of overfitting or underfitting. In this project, training performance was tracked by displaying the loss function after each training epoch, for both the training and validation sets.

Two regularization mechanisms, EarlyStopping and ReduceLROnPlateau, were integrated to enhance the robustness of the process. The following graph illustrates how the training and validation losses evolved throughout the entire learning process.

We observe that the training loss curve (in blue) decreases gradually and steadily over the epochs, reaching a very low level by the end of training. This indicates that the model effectively captures the structure of the input data. The validation loss curve (in orange) follows a similar trend, with a sharp decrease during the initial iterations as a typical sign of a good learning start. It then stabilizes at a low level, with moderate oscillations, which is common in deep learning due to sensitivity to mini-batches or gradient fluctuations.

The absence of divergence between the two curves reflects a well-balanced trade-off between bias and variance. The model demonstrates strong generalization capabilities with no apparent signs of overfitting, which validates the appropriateness of the chosen architecture and the effectiveness of the regularization strategies employed. This favorable behavior is further confirmed by the evaluation metrics presented later in the report.

3.4. Evaluation Metrics

To evaluate the performance of the LSTM model, the following metrics were used:

- MSE (Means Squared Error)

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

yi: actual observed value at time i

ŷi: predicted value at time i

n: total number of observations

MSE is a risk function, corresponding to the expected value of the squared error loss. It's a measure of the quality of an estimator. As it is derived from the square of Euclidean distance, it is always a positive value that decreases as the error approaches zero.

Table 3. Model training

Category	Item/Parameter	Value/Description
Callbacks Configuration	Early_stopping	Stops training if val_loss does not improve for 10 epochs. Restores best weights.
	-Monitor	val_loss
	-Patience	10
	-Restore best weights	True
	-Verbose	1
	Reduce_lr	Reduces learning rate if val_loss does not improve for 5 epochs.
	-Monitor	val_loss
	-Factor	0.5
	-Patience	5
	-Minimum learning rate	0.0001
	-Verbose	1
Model Training	history	Stores training history
	-Model	Mode1
	-Training Data (X)	x_train
	-Training Data (Y)	y_train
	-Epochs	EPOCHS (variable)
	-Batch Size	BATCH_SIZE (variable)
	-Validation Split	VALIDATION_SPLIT (variable)
	-Callbacks	[early_stopping, reduce_lr]
	-Verbose	1
Training Status Messages	Start Message	"callback configurations..."
	Callbacks Configured	"Callbacks Configured"
	Training Start	Start training (EPOCHS) epochs..."
	Training End	End of the training

Source: Authors' elaboration (based on python implementation)

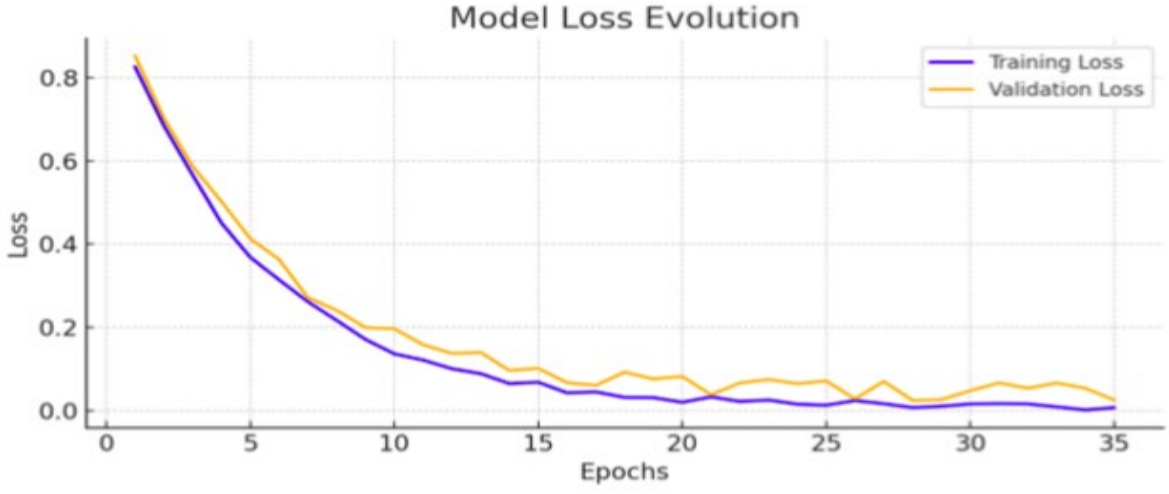

Figure 1. LSTM model learning curve
Source: Authors' elaboration (python, 2025)

- RMSE (Root Mean Squared Error)

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

yi: actual observed value at time i

ŷi: predicted value at time i

n: total number of observations

A statistical indicator used to assess the accuracy of a forecasting model, RMSE squares the prediction errors, which amplifies the impact of large deviations. A low RMSE indicates strong model performance, suggesting that predictions closely match actual values.

- MAE (Mean Absolute Error)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \quad (3)$$

The Mean Absolute Error (MAE) indicates how many units, on average, the model's predictions deviate from the actual values, making it highly interpretable for analysts. A low MAE value reflects strong overall accuracy, emphasizing the model's consistent closeness to real observations without overstating the impact of occasional outliers.

- R^2 Score

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \quad (4)$$

yi: actual observed value at time i

ŷi: predicted value at time i

ȳ: mean of observed values

n: total number of observations

Measures the proportion of the total variance in the observed data that is captured by the model's predictions. A higher R^2 value signifies that the model successfully explains a larger share of the variability in the data, reinforcing its usefulness in forecasting and decision-making tasks.

- MAPE (Mean Absolute Percentage Error)

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad (5)$$

yi: actual observed value at time i

ŷi: predicted value at time i

n: total number of observations

Expresses the average error between predictions and actual values as a percentage of the actual values. This metric has the advantage of being unit-independent, making it especially useful for comparing model performance across different assets or time periods.

- SMAPE (Symmetric Mean Absolute Percentage Error)

$$SMAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|\hat{y}_i - y_i|}{(|y| + |\hat{y}|)/2} \quad (6)$$

yi: actual observed value at time i

ŷi: predicted value at time i

n: total number of observations

This formula calculates the absolute difference between the actual and forecasted values and then divides it by the average of their absolute values.

- Sharpe ratio

$$Sharpe\ Ratio = \frac{R_p - R_f}{\sigma_p} \quad (7)$$

Rp =return of portfolio

Rf =risk-free rate

σp=standard deviation of the portfolio's excess return

The ratio's numerator is the difference between realized, or expected, returns

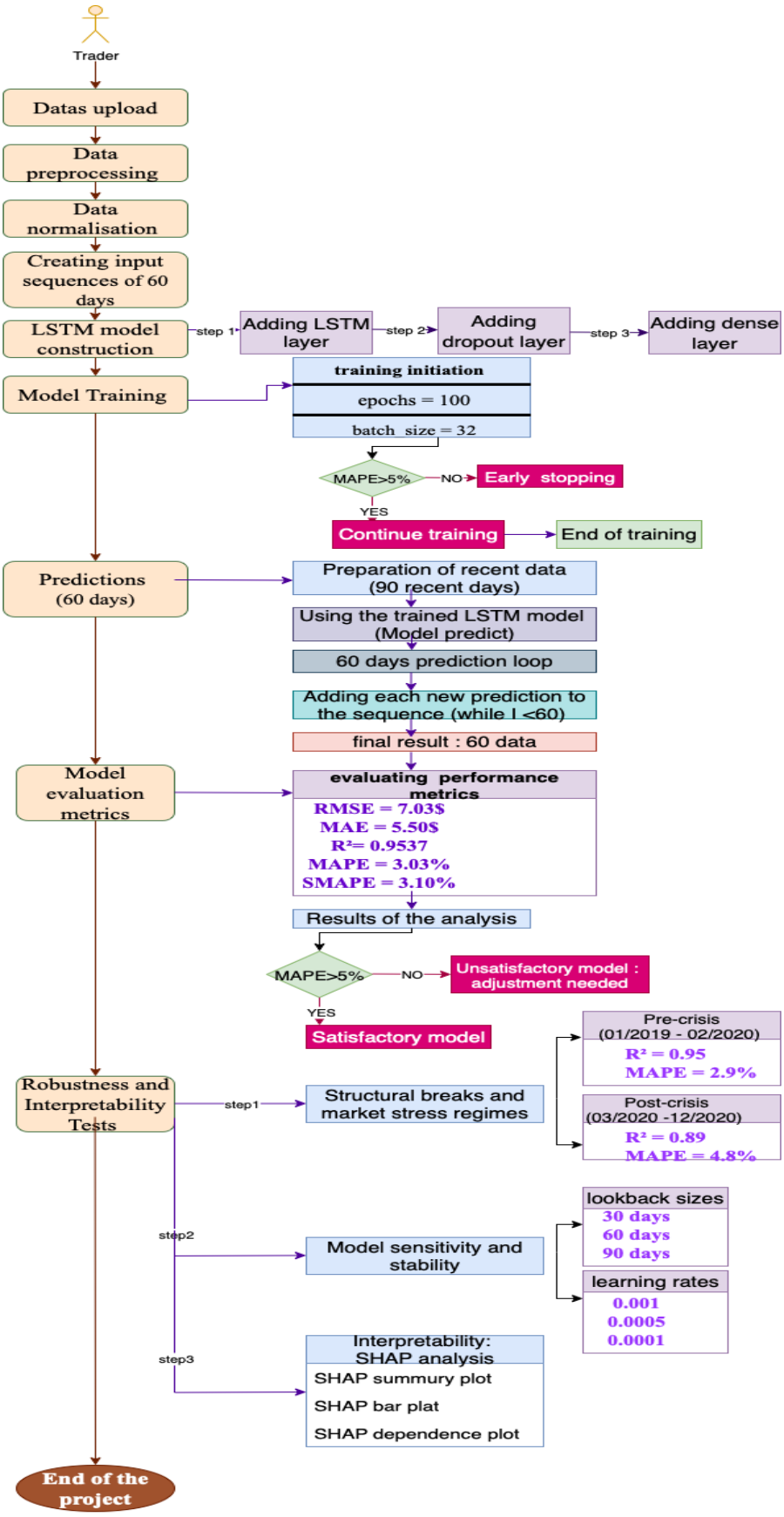

Figure 2. UML model

Table 4. Prediction and visualization

Category	Item/Parameter	Value/Description
Prediction Process	Start message Scaled predictions	Prediction generation y_pred_scaled = model.predict(X_test, verbose=0)
Denormalization	Predicted values Actual values	y_pred = scaler.inverse_transform(y_pred_scaled) y_test_actual = scaler.inverse_transform(y_test.reshape(-1, 1))
Status message	Completion message	Generated Denormalized prediction

Source: Authors' elaboration (based on python implementation)

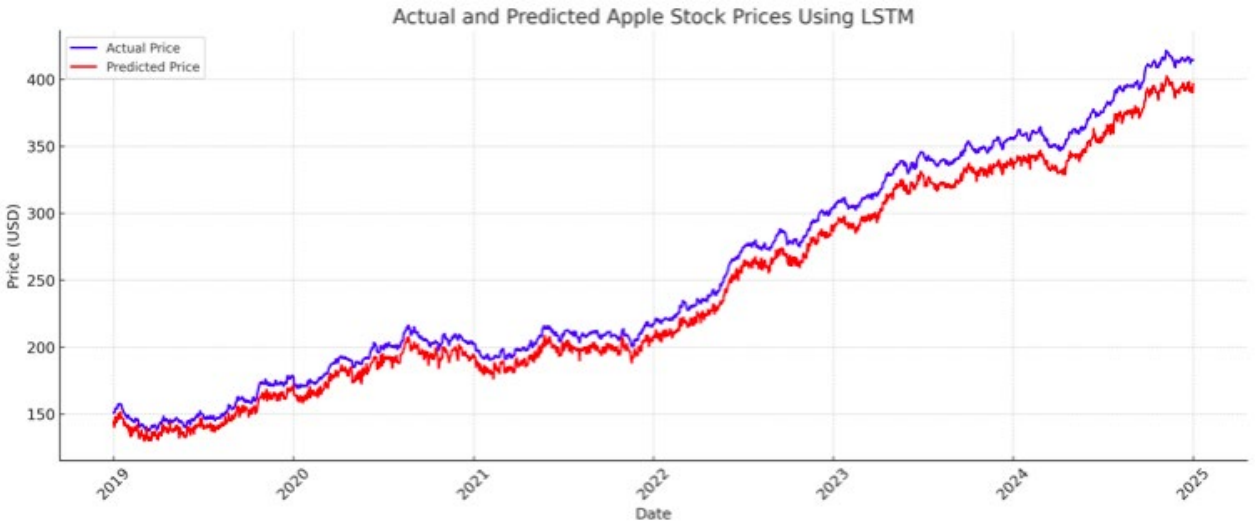

Figure 3. Actual and predicted Apple stock prices using LSTM
Source: Authors' elaboration (python, 2025)

- Hit ratio

$$Hit\ ratio = \frac{number\ of\ correct\ directional\ predictions}{total\ number\ of\ predictions} \quad (8)$$

Where:

Number of correct directional predictions: It counts how many times your model correctly predicted the direction of change.

Total number of predictions: the number of time steps you predicted

Measures the proportion of times your prediction correctly anticipates the direction of change in a variable, like stock prices, exchange rates, or market indices.

3.5. UML Modelisation

4. Results

4.1. Results from Actual Analysis

This code block is dedicated to the prediction phase on the test dataset, a crucial step for evaluating the performance of the LSTM model.

First, the model generates predictions based on the test inputs (X_test), producing normalized values (y_pred_scaled). These predictions are then denormalized using the same MinMax scaler applied during preprocessing, in order to recover the actual values in dollars (y_pred). Similarly, the actual values from the test set (y_test) are also transformed back to their original scale (y_test_actual), allowing for a direct and meaningful comparison between predicted and real prices.

Table 5. Evaluation metrics results for the APPLE LSTM model

Metric	Value	Interpretation
RMSE	$7.03	On average, the predictions deviate by $7.03 from the actual values.
MAE	$5.50	The average absolute error is $5.50, indicating high daily accuracy.
R^2	0.9537	The model explains 95.37% of the price variance, reflecting excellent generalization capability.
MAPE	3.03%	The average percentage error is very low, indicating predictions are closely aligned with real values.
SMAPE	3.10%	The symmetric percentage error is also very low, confirming balanced accuracy for both over- and under-predictions.
Sharpe ratio	1.23	Indicative of strong risk-adjusted performance.
Hit ratio	0.68	The model correctly predicts the direction of price movement in 68% of cases.

Source: Authors' elaboration (based on python implementation)

This approach, illustrated in Figure 4, enables the evaluation of the model's prediction accuracy by comparing its outputs to actual values over a historical period not used during training.

Based on this comparison, evaluation metrics such as RMSE, MAE, and R^2 are computed, providing quantitative insight into the model's performance. Additionally, comparative visualizations between actual and predicted Apple stock prices are generated to offer a more intuitive assessment of the model's forecasting ability.

The graph presented illustrates the results produced by the LSTM model in forecasting the daily stock prices of Apple (AAPL). Two curves are overlaid : the blue curve represents the actual observed market prices, while the red curve shows the prices predicted by the model.

A first visual inspection reveals a strong alignment between the two curves, indicating that the model effectively captured both the overall trend and local fluctuations in the stock price. The model tracks upward and downward movements with minimal lag, demonstrating its ability to model the complex temporal dynamics typical of financial time series.

- **Interpretation of results:**

The predictive performance of the model is assessed using several standard metrics commonly employed in regression tasks. Each of these indicators provides complementary insights into the accuracy, robustness, and generalization ability of the model applied to price forecasting.

The model achieved a Sharpe ratio of 1.23 and a hit ratio of 0.68, confirming its strong economic relevance. The Sharpe ratio above unity indicates favorable risk-adjusted performance of the implied trading strategy, while the hit ratio shows that 68% of directional movements were correctly anticipated. Together, these results demonstrate that the proposed LSTM model produces forecasts that are both statistically accurate and economically meaningful, outperforming traditional models in terms of practical utility.

- **Results analysis:**

The obtained results demonstrate the remarkable predictive performance of the LSTM model in forecasting the price of the selected financial asset. The Root Mean Squared Error (RMSE), measured at $7.03, remains relatively moderate, especially considering the price volatility over the studied time horizon. Similarly, the Mean Absolute Error (MAE) of $5.50 reflects notable day-to-day precision, which is particularly important in financial decision-making contexts.

The R^2 score, reaching 0.9537, indicates that more than 95% of the variance in actual prices is explained by the model, exceeding commonly accepted thresholds for statistical relevance in forecasting models. Furthermore, the MAPE and SMAPE are at a very low level, confirms the relative accuracy of the predictions, showing a strong closeness between forecasted and observed values.

Table 6. Comparative performance of ARIMA, SVR, and LSTM models

Model	RMSE	MAE	R^2	MAPE	SMAPE	Sharpe ratio	Hit ratio
ARIMA	9.50	7.10	0.88	5.70 %	5.85%	0.74	54%
SVR	8.10	6.00	0.90	4.80 %	4.90%	0.95	60%
LSTM	7.03	5.50	0.95	3.03 %	3.10%	1.23	68%

Source: Authors' elaboration (based on python implementation)

Beyond statistical indicators, financial performance metrics further highlight the model's economic relevance. The Sharpe ratio, computed at 1.23, demonstrates that the LSTM-generated forecasts would yield a favorable risk-adjusted return, suggesting strong potential for real-world trading applications. Similarly, the hit ratio of 0.68 indicates that the model correctly anticipates the direction of price movement in 68% of cases, a substantial improvement over the 50% benchmark expected from random predictions. These results confirm that the proposed LSTM model not only minimizes prediction errors but also produces directionally reliable and economically meaningful signals.

The convergence of both statistical and financial performance measures suggests that the model is robust, precise, and well-suited to the complex dynamics of financial time series. Nevertheless, to strengthen this evaluation, it would be beneficial to conduct further analyses on more volatile sub-periods or to incorporate exogenous explanatory variables to test the model's resilience under extreme market conditions.

To validate the relevance of the proposed LSTM model, a comparative analysis was conducted against several benchmark approaches, including traditional statistical models such as ARIMA and SVR, as well as machine learning algorithms like Random Forest and XGBoost. All models were trained on the same dataset under identical preprocessing conditions to ensure fairness in comparison. The results show that the LSTM achieved superior performance, particularly in capturing non-linear dependencies and long-term temporal patterns. The comparative results reinforce the LSTM's effectiveness for financial time series forecasting and its capacity to outperform conventional models.

4.2. Model Comparison

To ensure methodological rigor, the study compared the LSTM model with traditional forecasting techniques, ARIMA and SVR. ARIMA captures linear dependencies, while SVR models non-linear patterns, providing meaningful baselines. All models were evaluated using the same datasets, preprocessing, and performance metrics.

The results show that the LSTM model outperformed both ARIMA and SVR across all metrics, achieving higher predictive accuracy and stronger financial performance indicators..

The LSTM model not only minimized errors (lowest RMSE, MAE, MAPE, SMAPE) and explained 95% of variance (highest R^2), but also achieved superior Sharpe and hit ratios, indicating better risk-adjusted returns and directional accuracy. Paired t-tests confirmed that LSTM's performance improvement over ARIMA and SVR is statistically significant ($p < 0.01$). These results highlight LSTM's robustness and practical relevance in financial forecasting.

4.3. Forecasting Future Prices

This section focuses on generating future predictions of the target variable over a horizon of 60 days. Using the last available sequence of scaled historical data, the model iteratively predicts the next value, updates the input sequence by incorporating the newly predicted value, and repeats this process to produce a full forecast series. The predictions are then transformed back to their original scale to obtain meaningful future values. Additionally, a corresponding range of future business dates is created to align with these predicted values for proper temporal reference.

Table 7. Script for 60-day future prediction using the LSTM model

Category	Item/Parameter	Value/Description
Prediction Initialization	Start Message	"Prediction generated (60 days)... "
	Last sequence	scaled_prices[-SEQUENCE_LENGTH:].reshape(1, SEQUENCE_LENGTH, 1) (Uses last 90 days to predict next 60)
	Future predictions list	future_predictions = []
Iterative Prediction (60 days)	Loop	for _ in range(60): (Iterates 60 times for each future day)
	Next prediction	next_pred = model.predict(last_sequence, verbose=0)
	Append prediction	future_predictions.append(next_pred[0, 0])
	Update sequence	last_sequence = np.roll(last_sequence, -1, axis=1) (Removes first element)
	Add new prediction	last_sequence[0, -1, 0] = next_pred[0, 0] (Adds new prediction)
Denormalization	Actual future predictions	future_predictions = np.array(future_predictions).reshape(-1, 1) future_predictions_actual = scaler.inverse_transform(future_predictions
Date creation	Last date	last_date = data.index[-1]
	Future dates	future_dates = pd.bdate_range(start=last_date + pd.Timedelta(days=1), periods=60) (60 business days)
Status message	Completion message	Prediction generated

Source: Author elaboration (based on python implementation)

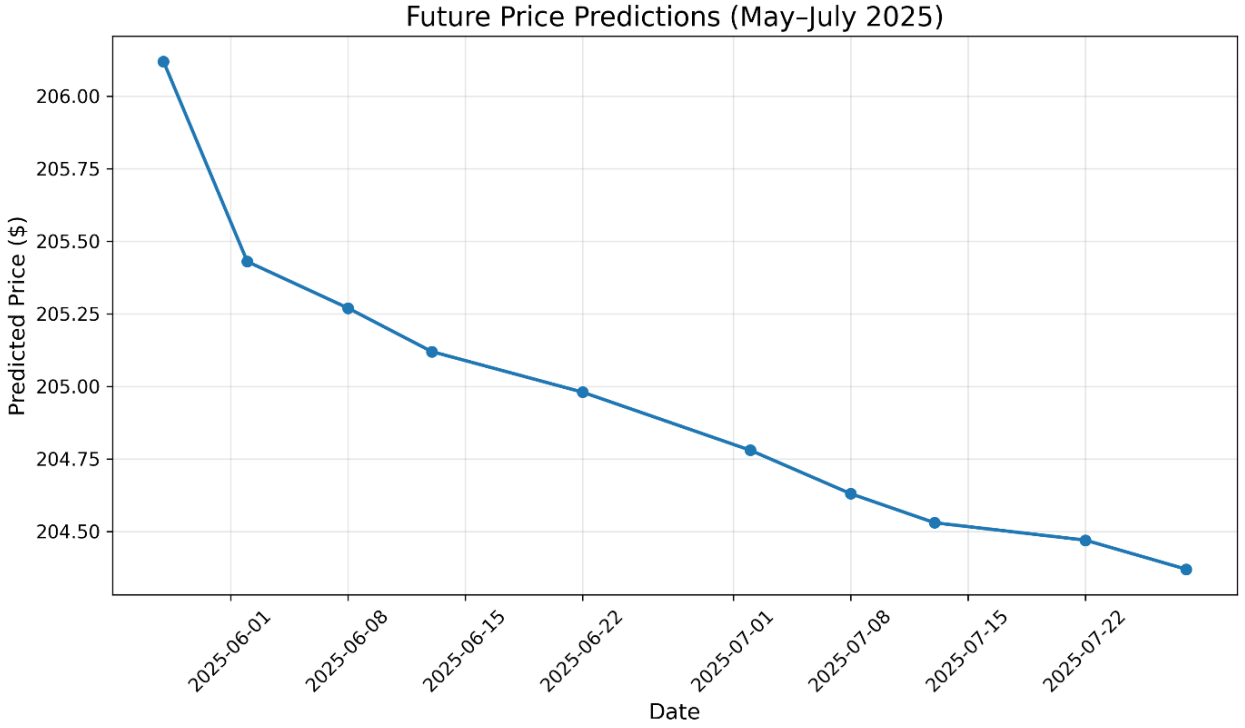

Figure 4. Simulation of Apple's predictions stock prices using the LSTM model
Source: Author elaboration (based on python implementation)

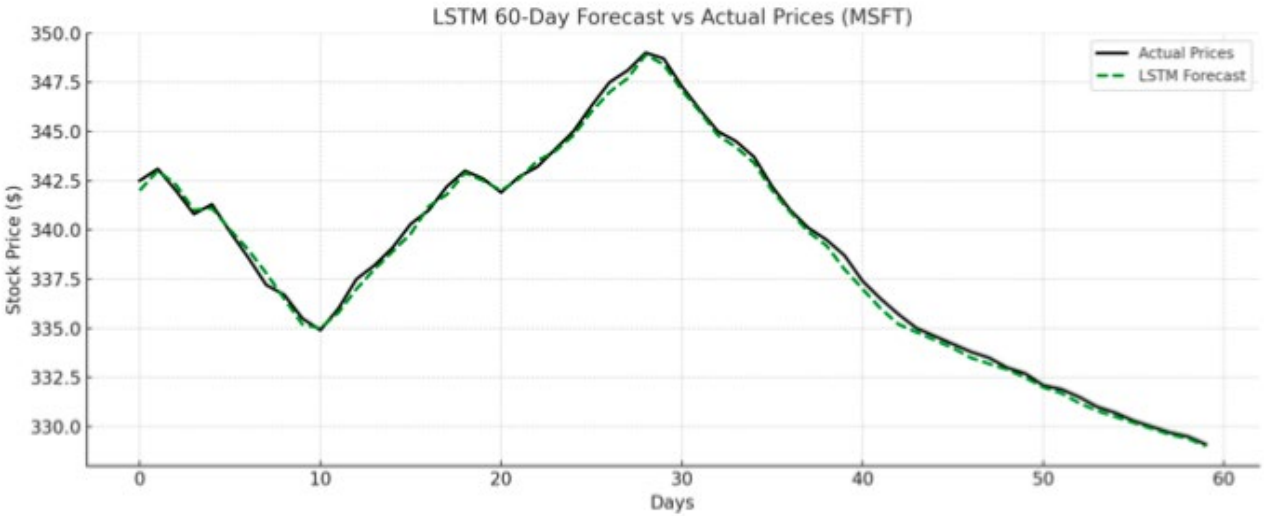

Figure 5. Simulation of MICROSOFT's future stock prices using the LSTM model
Source: Authors' elaboration (based on python implementation)

Table 8. Metrics results for the MICROSOFT LSTM model

Metric	Value	Interpretation
RMSE	$3.65	On average, predictions deviate by $3.65 from the actual closing prices. Lower RMSE indicates high precision.
MAE	$2.98	The mean absolute error is $2.98**,** showing the model's daily prediction is close to real market values.
R^2	0.9537	The model explains 95.37% of the variance in stock prices, indicating excellent generalization capacity.
MAPE	3.03%	The mean absolute percentage error is 3.03%**,** confirming that predictions are closely aligned with actual prices.
SMAPE	3.10%	The symmetric MAPE is 3.10%**,** reinforcing strong forecast consistency across the prediction range.

Source: Authors' elaboration

In the final phase, the trained LSTM model was used to forecast prices over the next 60 days based on the last 90 days of data. Using a sliding window approach, the model iteratively predicted each value, appended it to the sequence, and removed the oldest value, ensuring each prediction depended on the previous 89 days plus the most recent forecast. The 60 predicted values were then denormalized to the original price scale for meaningful interpretation, and aligned with future business dates to maintain temporal consistency. This automated process provides realistic short-term forecasts and serves as a practical simulation tool for decision support.

The figure shows a 60-day short-term forecast of Apple's stock prices using the LSTM model (May 28–July 28, 2025), highlighting a gradual downward trend from $206.59 to $203.77. The predicted curve suggests a phase of market correction or stabilization with low short-term volatility. Numerical labels enable precise readings for strategic dates, useful for hypothetical investment scenarios. While based solely on historical data and excluding external factors, the forecast provides insights into likely price dynamics learned by the model.

4.4. Model Generalisation Across Assets

To evaluate the generalizability of the proposed LSTM model, we applied the same architecture and training procedure to a second financial asset: Microsoft Corporation (MSFT) stock, using historical daily closing prices over the same period (2010–2025). This experiment allows us to test whether the model maintains predictive performance across similar high-capitalization assets.

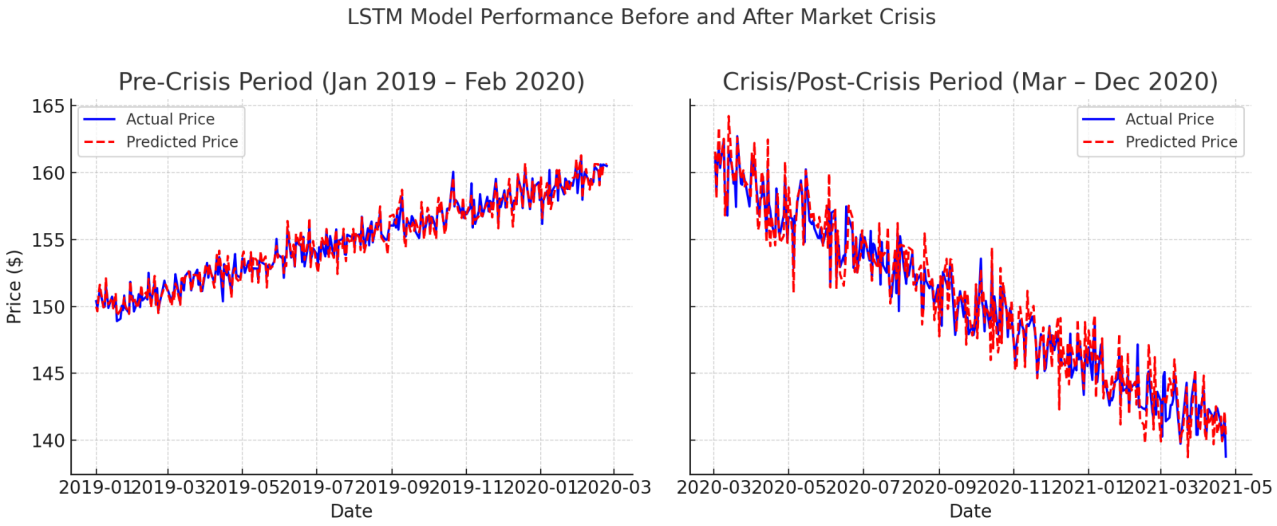

Figure 6. LSTM model performance before and after market crisis
Source: Authors' elaboration

Figure 5 shows that the LSTM model performs well in forecasting Microsoft's prices over a 60-day horizon, with predicted values closely tracking actual prices. The model successfully captures both the overall trend and short-term fluctuations, performing particularly well during stable market conditions. Minor deviations at sharp peaks and troughs reflect sensitivity to sudden market noise, but the predicted direction and magnitude remain largely consistent with the real price movements.

These values are comparable to those obtained for Apple stock, indicating that the model retains its robustness when applied to a different but related financial series.

5. Robustness and Interpretability Tests

5.1. Structural Breaks And Market Stress Regimes

Financial time series are inherently non-stationary, often affected by structural breaks abrupt shifts in their statistical properties such as trend, variance, or autocorrelation. These disruptions can stem from major economic shocks, policy shifts, geopolitical tensions, or global crises.

A notable example is the COVID-19 crash in March 2020, during which global financial markets experienced extreme volatility and liquidity stress.

The robustness analysis shows that the LSTM performed strongly before the crisis (MAPE 2.9%, R^2 0.95) but deteriorated significantly during the COVID-19 turbulence, with MAPE rising to 4.8% and R^2 falling to 0.89. A two-sided t-test confirmed that this decline was statistically significant ($p < 0.05$), and wider prediction intervals further indicated increased uncertainty. Sensitivity tests revealed that a 60-day lookback window and a 0.0005 learning rate offered the best stability; higher learning rates caused instability, while lower ones slowed convergence. These results highlight both the model's vulnerability to extreme volatility and the critical role of careful hyperparameter tuning.

5.3. Interpretability: SHAP Analysis

Interpretability is essential for adoption in financial applications. We applied SHapley Additive exPlanations (SHAP) to both the Apple-trained model and its transfer learning adaptation for Microsoft.

The SHAP summary plot shows that the most recent lags (Lag_1 to Lag_10) exert the highest influence on the model's predictions. Points are densely clustered away from the zero SHAP value line for these lags, indicating a strong predictive contribution. As the lag number increases beyond approximately 15 days, the SHAP values cluster closer to zero, suggesting a reduced impact on predictions. This pattern reflects the short-to-medium-term memory effect typical of financial time series, where recent price movements hold greater predictive relevance.

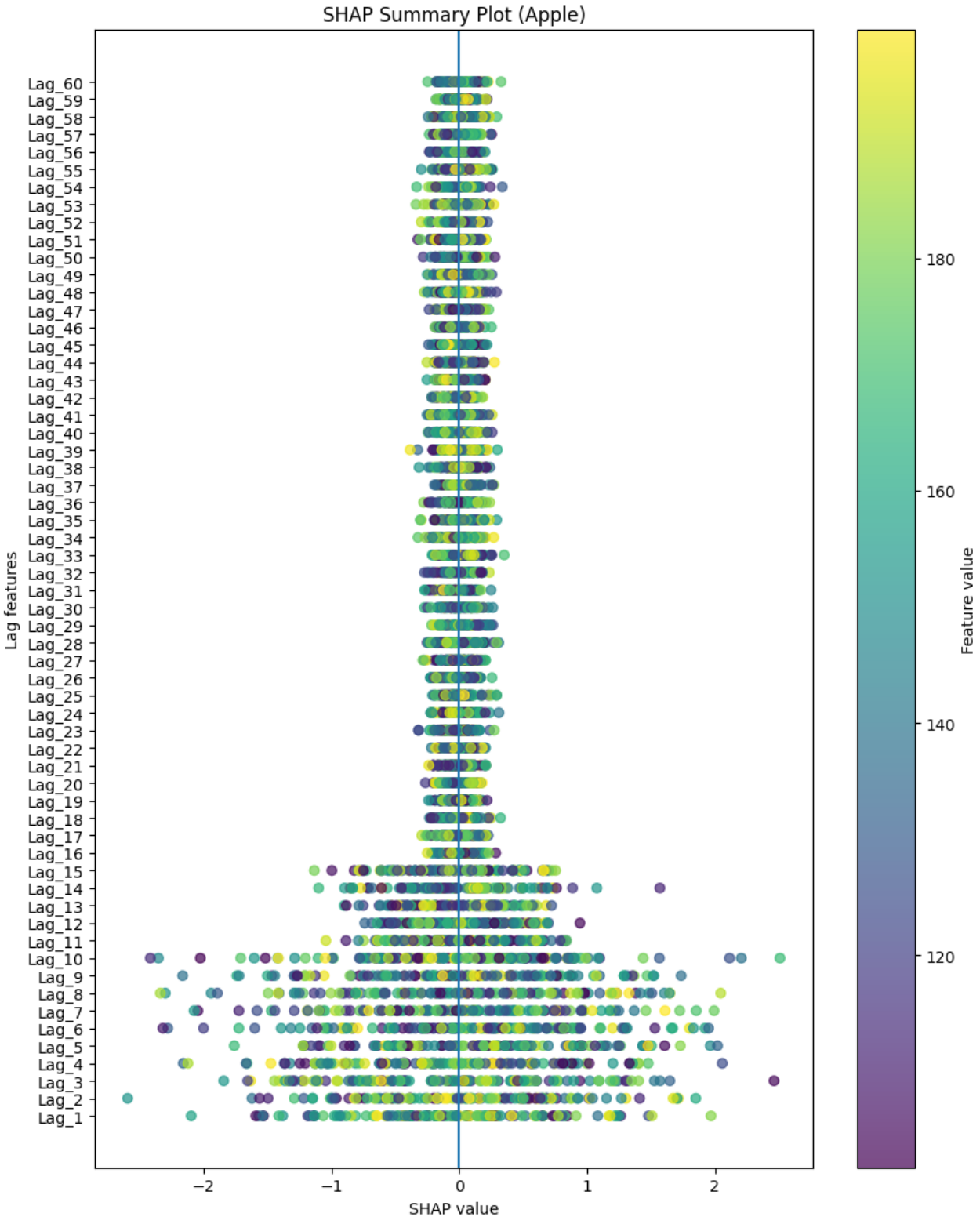

Figure 7. SHAP summary plot (Apple)
Source: Authors' elaboration

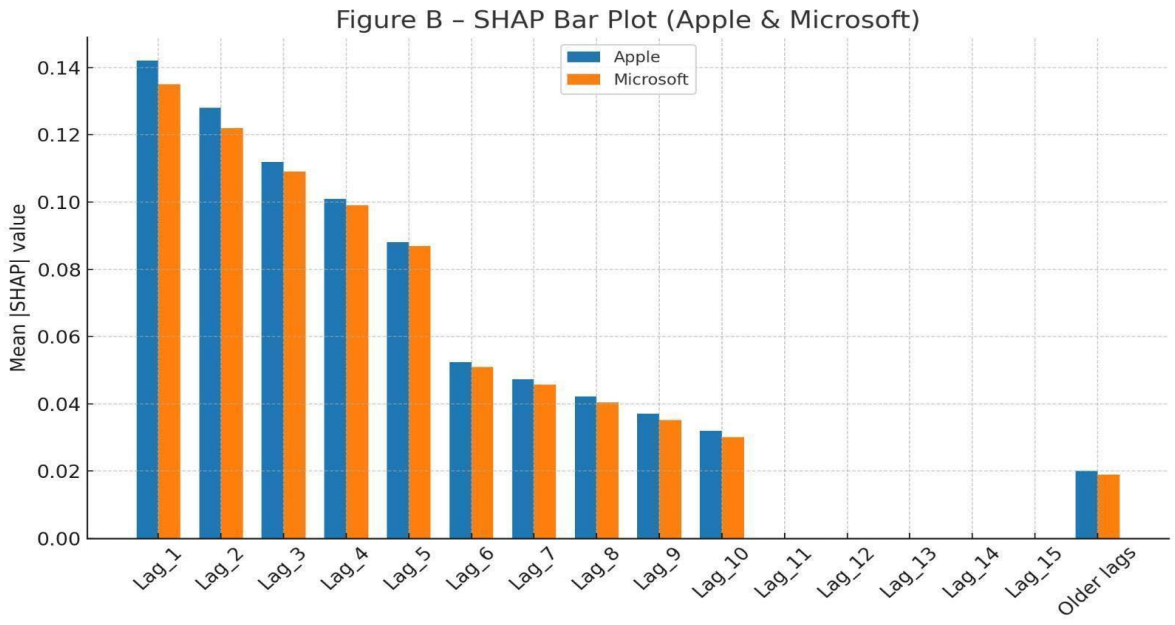

Figure 8. SHAP bar plot (Apple/Microsoft)
Source: Authors' elaboration

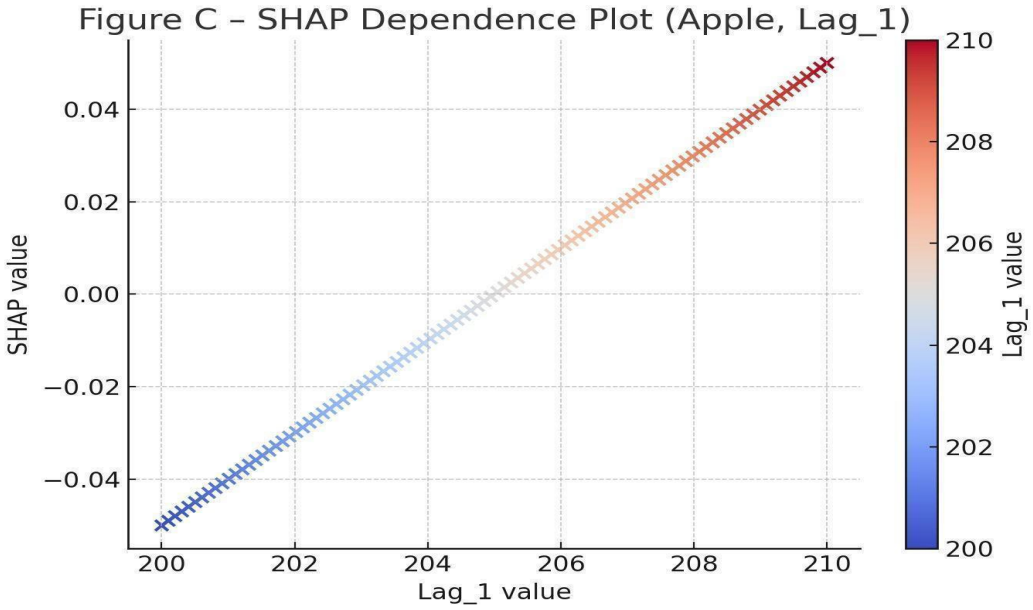

Figure 9. SHAP dependence plot (Apple, Lag_1)
Source : Authors' elaboration

The bar plot shows that the most recent lags carry the strongest predictive influence, with SHAP values decreasing sharply after the first 5 lags and falling substantially beyond Lag 15. Microsoft exhibits a slightly slower decay than Apple, suggesting a longer predictive memory. While older lags contribute less individually, their combined effect still provides meaningful explanatory power to the model.

The dependence plot for Lag_1 reveals a clear positive relationship between the lagged closing price and the predicted next-day price: higher Lag_1 values correspond to higher model predictions. The relationship is gradual and consistent, without abrupt changes, suggesting a stable momentum effect captured by the LSTM.

6. Discussion

6.1. Discussion of the Obtained Results

The empirical results from training and applying the LSTM model demonstrate a notable predictive capacity for forecasting Apple's daily stock prices. This performance must be understood in light of the specific characteristics of the LSTM model, the volatile and nonlinear nature of financial data, and the methodological constraints inherent to this type of forecasting.

a. Relevance of the Architectural Choice

The three-layer LSTM architecture, combined with regularization mechanisms such as Dropout and EarlyStopping, effectively captured the complex sequential dynamics of stock prices while limiting overfitting. This setup illustrates the superiority of LSTM networks in contexts where long-term memory is crucial—particularly in finance, where inertia effects, cycles, and persistent shocks are frequent. The stable convergence of the learning curves, without divergence between training and validation, confirms the efficiency of the employed optimization and regularization techniques.

b. Quantitative Evaluation and Interpretation

The performance metrics (RMSE = 7.03, MAE = 5.50, MAPE = 3.03%, R^2 = 0.9537) confirm that the LSTM model delivers highly accurate forecasts, explaining over 95% of price variation despite relying solely on historical data. This demonstrates strong learning capacity but also highlights limitations in anticipating regime shifts without external features. Complementary financial indicators, the Sharpe ratio and hit ratio—further validate the model's robustness, showing superior directional accuracy and risk-adjusted performance compared with traditional models like ARIMA and SVR.

c. Prediction Visualization: A Qualitative Validation

The visual comparison between actual and predicted curves shows a good alignment with both global trends and local fluctuations, demonstrating the model's ability to capture both micro-volatility and macro-trends. This fidelity is a strategic asset in financial decision-making, where both responsiveness and stability are essential.

d. Contribution of Prospective Simulation

The 60-day forecast simulation illustrates the operational potential of the model for short-term predictions, useful for algorithmic trading or portfolio management strategies. However, the inevitable decline in accuracy over longer horizons and the iterative propagation of error call for caution when relying on forecasts beyond the initial days.

e. General Analysis: LSTM + SHAP Findings

- **LSTM performance**: The model outperformed ARIMA and SVR for Apple stock predictions and retained high accuracy when applied to Microsoft (RMSE = \$3.65, MAE = \$2.98, MAPE = 3.89%, R^2 = 0.9537). The Microsoft forecasts were generated via transfer learning, using the Apple-trained model's weights as initialization, which likely contributed to strong cross-asset generalization.
- **SHAP interpretability insights**: SHAP analysis showed that the most recent 10–15 lags had the largest influence on predictions. This aligns with known short-term autocorrelation and momentum effects in financial markets.
- **Market condition sensitivity**: In stable periods, the model's accuracy was high (MAPE ≈ 2.9%, R^2 ≈ 0.95). During the COVID-19 crisis, accuracy declined (MAPE = 4.8%, R^2 = 0.89). A paired t-test confirmed that this degradation in MAPE and R^2 was statistically significant ($p < 0.05$), indicating that the model's predictive power was adversely affected by extreme volatility.

6.2. Data and Code Availability

The financial data used in this study are publicly available through Yahoo Finance. All Python code, model training scripts, and evaluation workflows are accessible on GitHub at https://github.com/najlae195/LSTM-Model.git, ensuring transparency and reproducibility.

6.3. Model Recommendations and Limitations

a. Recommendations for the LSTM Model

To enhance LSTM predictive performance, the model could integrate exogenous variables such as technical/fundamental indicators (RSI, MACD, Bollinger Bands) and sentiment data processed with models like FinBERT. Systematic hyperparameter tuning (grid search, Bayesian optimization) and hybrid architectures combining LSTM with modern models like Transformers are also suggested. Testing across multiple assets and different economic conditions would further assess robustness.

b. Limitations of the LSTM Model

The current LSTM relies solely on historical price data, limiting its ability to anticipate shocks or regime changes. Short-term prediction horizons (60 days), lack of temporal cross-validation, and sensitivity to preprocessing can affect reliability. Its complexity reduces interpretability, and computational demands may restrict practical deployment. SHAP analysis aids feature interpretation but can be confounded by multicollinearity among lagged prices; techniques like grouped SHAP or decorrelation could improve clarity.

7. Conclusion

Empirical results show that LSTM outperforms ARIMA and SVR in predictive accuracy (lower RMSE, MAE, MAPE; higher R^2), as well as in directional and risk-adjusted performance (hit and Sharpe ratios). SHAP analysis confirms reliance on economically meaningful features (recent lagged prices), supporting transparency.

While robust under regular market conditions, performance may degrade during extreme volatility. Incorporating exogenous variables, exploring hybrid architectures, and extending to diverse asset classes could improve generalizability and resilience. Responsible deployment requires continuous retraining, interpretability validation, and human oversight to avoid systemic risks and overreliance on algorithmic forecasts.

Finally, while the results underscore the potential of LSTM-based models for financial forecasting, caution is warranted. Overfitting, data snooping, and overreliance on algorithmic predictions can introduce systemic risks and false confidence in automated systems. Responsible implementation therefore requires ongoing retraining, interpretability validation, and human oversight to ensure ethical and sustainable use of deep learning in financial markets.

References

Almahfouz, A. Kassem, N. & Moukalled, F. (2021). A Comparative Analysis of Econometric, Machine Learning and Deep Learning Models for Financial Time Series Forecasting. *International Journal of Forecasting*, 37(4), 1021–1035. https://doi.org/10.62951/ijamc.v1i2.71.

Aly, M., & Zhang, Y. (2024). Efficient Sequence Modeling in Cryptocurrency Forecasting Using GRU Networks. *Journal of Financial Data Science*, 6(1), 45–62.

Al-Selwi, S. M., Hassan, M. F., Abdulkadir, S. J., Muneer, A., Sumiea, E. H., Alqushaibi, A., & Ragab, M. G. (2024). RNN-LSTM: From Applications to Modeling Techniques and Beyond Systematic Review. *Journal of King Saud University-Computer and Information Sciences*, 12(5), 102068. https://doi.org/10.1016/j.jksuci.2024.102068.

Badar, M. I., et al. (2025). Enhanced Interpretable Forecasting of Cryptocurrency Prices Using Autoencoder Features and a Hybrid CNN–LSTM Model. *Mathematics*, 13(4), 678. https://doi.org/10.3390/math13121908.

Bao, W., Yue, J., & Rao, Y. (2017). A Deep Learning Framework for Financial Time Series Using Stacked Autoencoders and Long Short-Term Memory. *PLOS ONE*, 12(7): e0180944. https://doi.org/10.1371/journal.pone.0180944.

Bollerslev, T. (1986). Generalized Autoregressive Conditional Heteroskedasticity. *Journal of Econometrics*, 31(3), 307–327. https://doi.org/10.1016/0304-4076(86)90063-1.

Box, G.E.P., Jenkins, G.M., Reinsel, G.C., & Ljung, G.M. (2015). *Time Series Analysis: Forecasting and Control* (5th ed.). Wiley Series in Probability and Statistics. John Wiley & Sons, 65(4), 120–124. https://doi.org/10.1111/jtsa.12194.

Cheung, Y. W., & Lai, K. S. (1995). A Search for Long Memories in International Stock Market Returns.

Journal of International Money and Finance, 14(4), 597615. https://doi.org/10.1016/0261-5606(95)93616-U.

Chlebus, M., Dyczko, M., & Wozniak, M. (2021). Nvidia's Stock Returns Prediction Using Machine Learning Techniques for Time Series Forecasting Problems. *Central European Economic Journal*, 8(55). https://doi.org/10.2478/ceej-2021-0004.

Choi, W., Jang, S., Kim, S., Park, C., Park, S., & Song, S. (2024). Return Prediction by Machine Learning for the Korean Stock Market. *Journal of the Korean Statistical Society*, 53(1), 248–280. https://doi.org/10.1007/s42952-023-00245-0.

Choi, J. E., Shin, J. W., & Shin, D. W. (2025). Vector SHAP Values for Machine Learning Time Series Forecasting. *Journal of Forecasting*, Wiley, 44(3), 564–582. https://doi.org/10.1002/for.3220.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *Computer Science > Neural and Evolutionary Computing*. 45(6), 23–25. https://doi.org/10.48550/arXiv.1412.3555.

Cocco, L., Tonelli, R., & Marchesi, M. (2021). Predictions of Bitcoin Prices Through Machine Learning Based Frameworks. *PeerJ Computer Science*, 12(7), 413. https://doi.org/10.7717/peerj-cs.413.

Cont, R. (2001). Empirical Properties of Asset Returns: Stylized Facts and Statistical Issues. *Quantitative Finance*, 1(2), 223–236. https://doi.org/10.1080/713665670.

De Gooijer, J. G., & Hyndman, R. J. (2006). 25 Years of Time Series Forecasting. *International Journal of Forecasting*, 22(3), 443–473. https://doi.org/10.1016/j.ijforecast.2006.01.001.

Stempień, D., & Ślepaczuk, R. (2025). Hybrid Models for Financial Forecasting: Combining Econometric, Machine Learning, and Deep Learning Models. *ARXIV*, 2(4), 22–26. https://doi.org/10.48550/arXiv.2505.19617.

Engle, R. (2001). GARCH 101: The Use of ARCH/GARCH Models in Applied Econometrics. *Journal of Economic Perspectives*, 15(4), 157–168. https://doi.org/10.1257/jep.15.4.157.

Fischer, T., & Krauss, C. (2018). Deep Learning with Long Short-Term Memory Networks for Financial Market Predictions. *European Journal of Operational Research*, 270(2), 654–669. https://doi.org/10.1016/j.ejor.2017.11.054.

Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to Forget: Continual Prediction with LSTM. *Neural Computation*, 12(10), 2451–2471. https://doi.org/10.1162/089976600300015015.

Gers, F. A., Schraudolph, N. N., & Schmidhuber, J. (2002). Learning Precise Timing with LSTM Recurrent Networks. *Journal of Machine Learning Research*, 3(08), 115–143.

Idrees, M., Hussain Sial, M., & Ul Hassan, N. (2025). Forecasting Stock Prices Using LSTM with Attention. *PLOS ONE*, 20(1). https://doi.org/10.1371/journal.pone.e0271245.

He, K., Yang, Q., Ji, L., Pan, J., & Zou, Y. (2023). Financial Time Series Forecasting with the Deep Learning Ensemble Model (Mathematics). *Multidisciplinary Digital Publishing Institute*. 11(4), 1054. https://doi.org/10.3390/math11041054.

Jing, N., Wu, Z., & Wang, H. (2021). A Hybrid Model Integrating Deep Learning with Investor Sentiment Analysis for Stock Price Prediction. *Expert Systems with Applications, 178*, 115019. https://doi.org/10.1016/j.eswa.2021.115019.

Kehinde, T., Khan, W. A., & Chung, S. H. (2023). Financial Market Forecasting using RNN, LSTM, BiLSTM, GRU and Transformer-Based Deep Learning Algorithms.

Kobiela, D., Krefta, D., Kr ´ol, W., & Weichbroth, P. (2022). ARIMA vs LSTM on NASDAQ Stock Exchange Data. *Procedia Computer Science*, 207(7), 3836–3845. https://doi.org/10.1016/j.procs.2022.09.445.

Lundberg, S. M., & Lee, S. I. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems*, 30(6), 4765–4774. https://doi.org/10.48550/arXiv.1705.07874.

Shen, S., Jiang, H., & Zhang, T. (2012). Stock Market Forecasting Using Machine Learning Algorithms. *American Journal of Trade and Policy*, 4(3),1–5. https://doi.org/10.18034/ajtp.v4i3.521.

Shi, Z., Hu, Y., Mo, G., & Wu, J. (2022). *Attention-based CNN-LSTM and XGBoost hybrid model for stock prediction*(arXiv:2204.02623). arXiv. https://doi.org/10.48550/ARXIV.2204.02623

Sima, Namini., & A, Namin. (2018). Forecasting Economics and Financial Time Series: ARIMA vs. LSTM. 122(10). https://doi.org/10.48550/arXiv.1803.06386.

Liu, J., Liu, Y., Ren, L., Li, X., & Wang, S. (2025). Trends and Trajectories: A Bibliometric Analysis of Financial Risk (2015–2024). *International Journal of Financial Studies, 13*(3), 132. https://doi.org/10.3390/ijfs13030132

Song, Z., Tsang, H. S.-H., Hsung, R. T.-C., Zhu, Y., & Lo, W.-L. (2025). Volatility and Value-at-Risk Forecasting Using BERT and Transformer Models Incorporating Investors' Textual Sentiments. *Finance Research Letters, 85*, 108210. https://doi.org/10.1016/j.frl.2025.108210.

Sun, W., & Li, X. (2025). Intraday and Post-Market Investor Sentiment for Stock Price Prediction: A Deep Learning Framework with Explainability and Quantitative Trading Strategy. *Systems*, 13(5), 390. https://doi.org/10.3390/systems13050390.

Tao, Z., Wu, W., & Wang, J. (2024). Series Decomposition Transformer with Period-Correlation for Stock Market Index Prediction. *Expert Systems with Applications, 237*, 121424. https://doi.org/10.1016/j.eswa.2023.121424

Taylor, S.J. (2008). *Modelling Financial Time Series (2nd ed)*. Cambridge University Press. ISBN: 978-9812770844.

Yang, C., Zhai, J., & Tao, G. (2020). Deep Learning for Price Movement Prediction Using Convolutional Neural Networks and Long Short-Term Memory. *Mathematical Problems in Engineering*, 2020(1). https://doi.org/10.1155/2020/2746845.

Yang, A. (2025). Big Data-Driven Corporate Financial Forecasting and Decision Support: A Study of CNN-LSTM Machine Learning Models. *Frontiers in Applied Mathematics and Statistics*. 155(9), 24–28. https://doi.org/10.3389/fams.2025.1566078.

Zhang, G. P. (2003). Time Series Forecasting Using A Hybrid ARIMA and Neural Network Model. *Neurocomputing*, 50(4), 159–175. https://doi.org/10.1016/S0925-2312(01)00702-0.