Comparative Study of ARIMA and LSTM Models for Stock Price Prediction

1. Introduction

Stock price prediction is a challenging time series forecasting problem due to the presence of noise, non-linearity, and sudden market fluctuations. Traditionally, statistical models such as ARIMA have been widely used for time series forecasting because of their mathematical simplicity and interpretability. In recent years, deep learning models—especially Long Short-Term Memory (LSTM) networks—have gained attention for their ability to model long-term dependencies and complex non-linear patterns in sequential data.

The objective of this assignment is to implement both ARIMA and LSTM models on the same stock price dataset, evaluate their performance using standard error metrics, and compare their predictive behavior. This comparison helps in understanding the practical trade-offs between classical statistical approaches and modern deep learning techniques in financial time series forecasting.

2. Dataset Description and Preprocessing

2.1 Data Collection

The dataset used in this assignment consists of historical stock prices of **Apple Inc. (AAPL)** obtained from **Yahoo Finance**. The data spans approximately three years of daily trading activity.

The **Adjusted Close Price** was selected as the target variable because it reflects the true economic value of the stock by accounting for corporate actions such as dividends, stock splits, and bonuses. Using Adjusted Close ensures consistency and avoids artificial price jumps that could negatively affect model learning.

2.2 Data Cleaning and Formatting

The dataset was examined for missing values and inconsistencies. No missing entries were found, and the data was already indexed by date in datetime format, making it directly suitable for time series analysis.

2.3 Train–Test Split

To evaluate generalization performance, the dataset was split chronologically as follows:

- **Training set:** 80%
- **Testing set:** 20%

Chronological splitting was used instead of random splitting to preserve the temporal structure of the data, which is essential for time series forecasting tasks.

2.4 Normalization

Normalization using Min–Max scaling was applied **only for the LSTM model**, scaling values to the range [0, 1]. Neural networks are sensitive to input scale, and normalization improves training stability and convergence speed.

Normalization was **not applied to the ARIMA model**, as ARIMA operates on raw numerical values and relies on differencing rather than scaling for stationarity.

3. ARIMA Model

3.1 Stationarity Analysis using ADF Test

ARIMA models assume that the time series is stationary. To verify this assumption, the **Augmented Dickey–Fuller (ADF) test** was applied to the training data.

- **Null Hypothesis (H_0):** The series is non-stationary
- **Alternative Hypothesis (H_1):** The series is stationary

The obtained p-value was greater than 0.05, indicating non-stationarity. Consequently, **first-order differencing** was applied to stabilize the mean of the series.

3.2 Parameter Selection using ACF and PACF

After differencing:

- The **PACF plot** showed significant spikes up to lag 5, suggesting an autoregressive order **p = 5**.
- The **ACF plot** did not show strong significant lags beyond the first few, leading to **q = 0**.
- The differencing order was set to **d = 1** based on the stationarity test.

Thus, the final model selected was **ARIMA(5,1,0)**.

3.3 Model Training and Forecasting

The ARIMA model was trained on the training dataset and used to forecast prices over the testing period. Predictions were generated sequentially and compared with actual observed prices.

3.4 Residual Analysis

Residual plots were analyzed to verify model assumptions. The residuals were centered around zero with no strong deterministic patterns, indicating a reasonable model fit. However, some volatility clustering was observed, which is common in financial time series and highlights the limitations of linear models like ARIMA.

4. LSTM Model

4.1 Sliding Window Formulation

To convert the time series into a supervised learning problem, a **sliding window approach** was used. A window size of **60 days** was chosen, meaning the model uses the past 60 trading days to predict the next day's price.

This window size provides a balance between capturing medium-term trends and avoiding excessive model complexity.

4.2 Model Architecture

The LSTM model consisted of:

- Two stacked LSTM layers with 50 neurons each
- One fully connected Dense output layer
- Adam optimizer
- Mean Squared Error (MSE) loss function

Stacked LSTM layers were used to allow the model to learn higher-level temporal representations from the data.

4.3 Training and Learning Curves

The model was trained for **20 epochs** with a batch size of **32**, using a validation dataset to monitor performance. The learning curves showed a steady decrease in both training and validation loss, indicating stable learning and minimal overfitting.

5. Model Evaluation and Comparison

5.1 Evaluation Metrics

Both models were evaluated using:

- Mean Absolute Error (MAE)
- Mean Squared Error (MSE)
- Root Mean Squared Error (RMSE)

These metrics were chosen because they directly quantify prediction error magnitude. Mean Absolute Percentage Error (MAPE) was not emphasized due to its instability when actual values are small.

5.2 Quantitative Performance

The LSTM model achieved lower MAE, MSE, and RMSE values compared to the ARIMA model, indicating better predictive accuracy on the test dataset.

5.3 Visual Analysis

- **Stock price vs prediction plots** showed that ARIMA produced smoother predictions and lagged during rapid price changes.
- LSTM predictions followed the actual price movement more closely, especially during volatile periods.
- **Residual plots** indicated smaller and more evenly distributed errors for the LSTM model.
- **Learning curves** confirmed stable training behavior for the LSTM model.

6. Comparative Discussion

ARIMA Model

Strengths:

- Simple and interpretable
- Low computational cost
- Effective for linear trends

Limitations:

- Assumes linearity

- Requires stationarity
- Poor performance during high volatility

LSTM Model

Strengths:

- Captures non-linear relationships
- Handles long-term dependencies
- Better suited for complex time series

Limitations:

- Computationally expensive
- Requires large datasets
- Less interpretable

7. Conclusion

This assignment highlights the fundamental differences between statistical and deep learning approaches to time series forecasting. ARIMA serves as a strong baseline model and performs well for stable, linear patterns. However, stock price data often exhibits non-linearity and volatility, where LSTM models demonstrate superior performance.

While LSTM models provide better predictive accuracy, they come at the cost of higher computational complexity and reduced interpretability. Therefore, model selection should depend on the specific application requirements, data availability, and interpretability needs.

Future work could include incorporating technical indicators, experimenting with bidirectional LSTM architectures, or applying attention mechanisms to improve both performance and interpretability.