Time Series Forecasting for Johnson & Johnson and Amazon Using ARIMA, LSTM and GRU

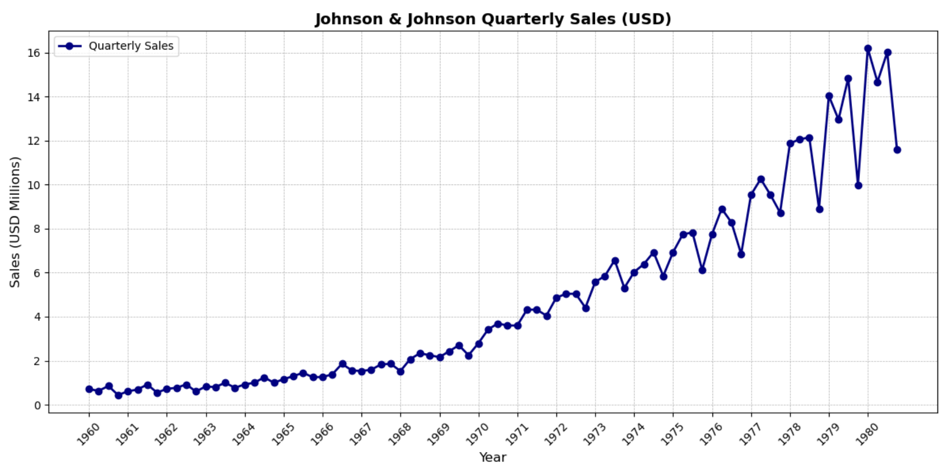
Student Name – Chinni Raj Paul Juthiga
Student ID – 20067368
Github Link – https://github.com/chinnirajpaul/Time-series-case-study-.git

Introduction

Time series forecasting is a critical tool in financial analysis allowing stakeholders to make informed decisions based on expected trends. In this report, we model **Amazon's stock price** and **Johnson & Johnson's quarterly sales** using both traditional statistical and machine learning techniques: **ARIMA, LSTM, and GRU**. Each method is evaluated and compared using **RMSE, MAE, and MAPE** metrics. The goal is to identify which forecasting methods are most effective for different types of financial data and to provide actionable insights based on the comparative performance of these models.

Initial Plots of the Dataset

Johnson & Johnson's quarterly sales spanning from 1960 to 1980



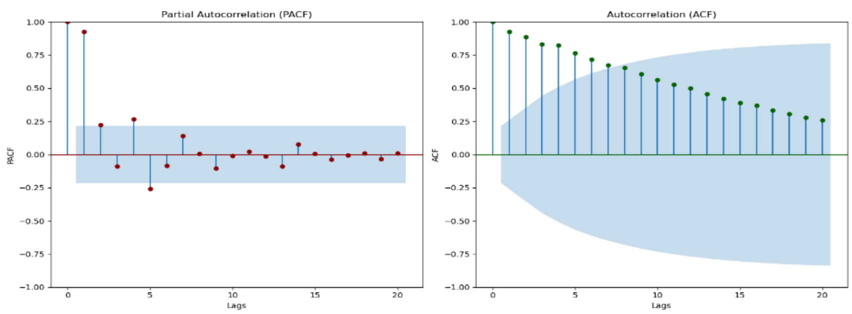
Amazon's stock prices from 2018 through 2023.

Stationarity Analysis and Transformation

Johnson & Johnson (JJ) Sales Data

To assess the suitability of time series models like ARIMA the stationarity of the data was tested using the Augmented Dickey-Fuller (ADF) test and visualized through Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots.

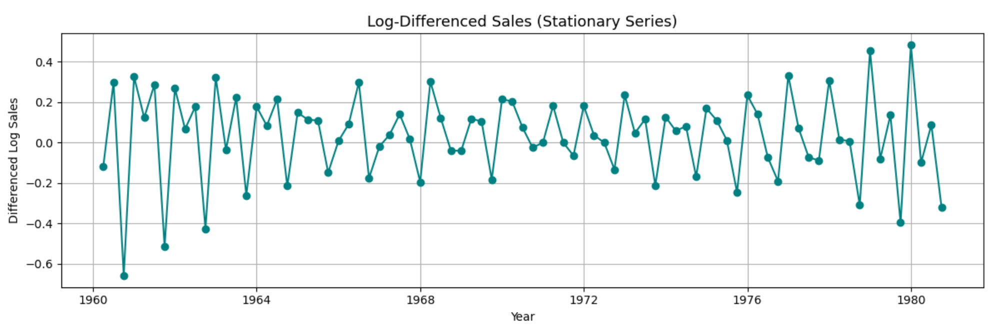
Figure 1 shows the ACF and PACF plots for the original JJ sales series, indicating strong serial correlation and non-stationarity.



ADF Test Results on Original Series

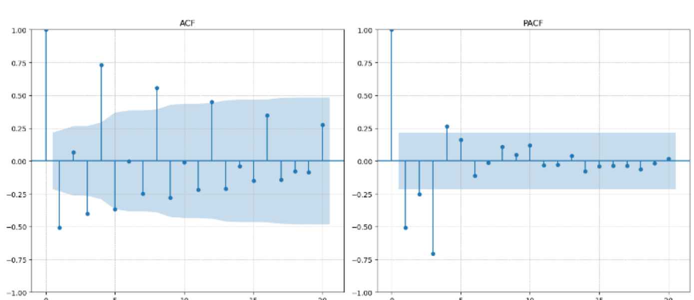
ADF Statistic: 2.742, **p-value**: 1.0000, The data is non-stationary ($p > 0.05$). To transform the series, a natural logarithm was applied followed by first-order differencing.

Figure 2 shows the log-differenced series, now appearing stationary.



ADF Test on Transformed Series - **ADF Statistic**: -4.317, **p-value**: 0.0004, The transformed data is now stationary ($p < 0.05$).

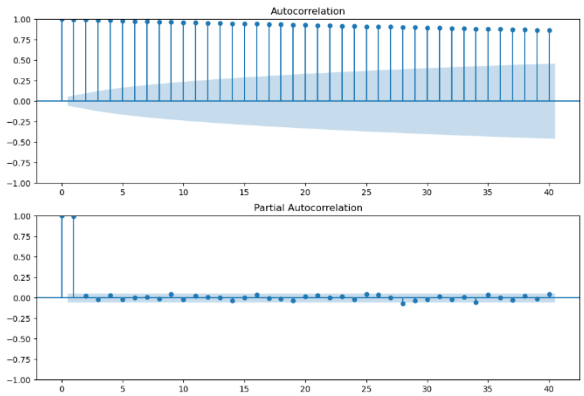
Figure 3 shows the ACF and PACF plots for the log-differenced JJ series, providing guidance for selecting ARIMA parameters p and q.



Amazon Stock Price Data

A similar procedure was applied to Amazon's stock price series. The ADF test on the original closing prices gave the following results:

ADF Test on Original Series- **ADF Statistic**: -1.658, **p-value**: 0.453, The series is non-stationary (p > 0.05).

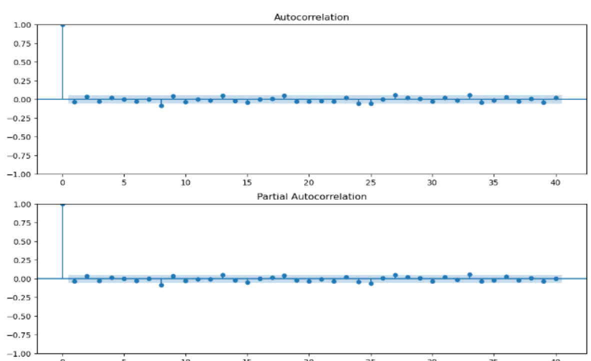


A log transformation followed by first-order differencing was applied.

ADF Test on Transformed Series - **ADF Statistic**: -36.640, **p-value**: 0.0000,

The data is now stationary (p < 0.05).

Figure 4 presents ACF and PACF plots of the differenced Amazon log prices. The sharp drop-off in the ACF and significant lags in the PACF suggest ARIMA model applicability and help identify p and q values.



Methods Overview

ARIMA (AutoRegressive Integrated Moving Average)

ARIMA models are linear models used for univariate time series forecasting. The ARIMA(p,d,q) formulation represents:

- **p**: number of autoregressive terms
- **d**: number of differences to make the data stationary
- **q**: number of moving average terms

This method assumes linearity and stationarity which were addressed by applying log transformation and first differencing.

LSTM (Long Short-Term Memory Networks)

LSTM is a special type of Recurrent Neural Network (RNN) capable of learning long-term dependencies. It is particularly suited for time series problems where patterns evolve over time.

Why LSTM? Because Amazon's stock prices show non-linear, long-term trends and fluctuations, LSTM is expected to outperform linear models like ARIMA.

GRU (Gated Recurrent Unit)

GRU is a simplified variant of LSTM that reduces computational complexity while often delivering similar performance.

Why GRU? It trains faster than LSTM and can be beneficial when long sequences or fewer data points are involved.

LSTM and GRU Model Development

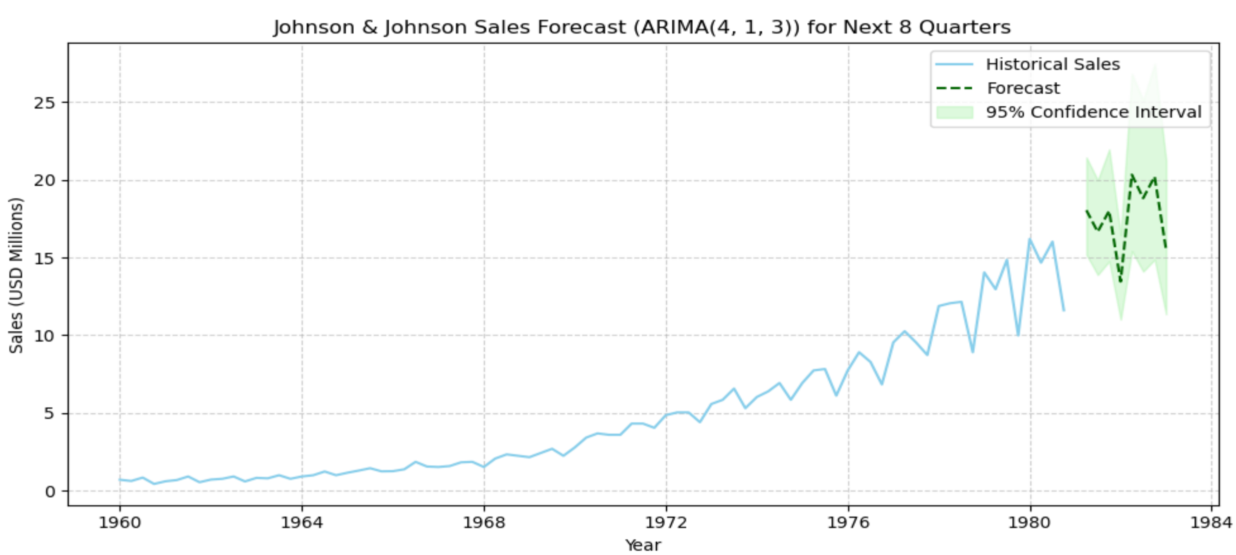
The Amazon stock price data was first scaled to the [0, 1] range using **MinMaxScaler** from scikit-learn. Sequences of 60 time steps were created to serve as input features for both models. The dataset was split into 90% training and 10% testing sets.

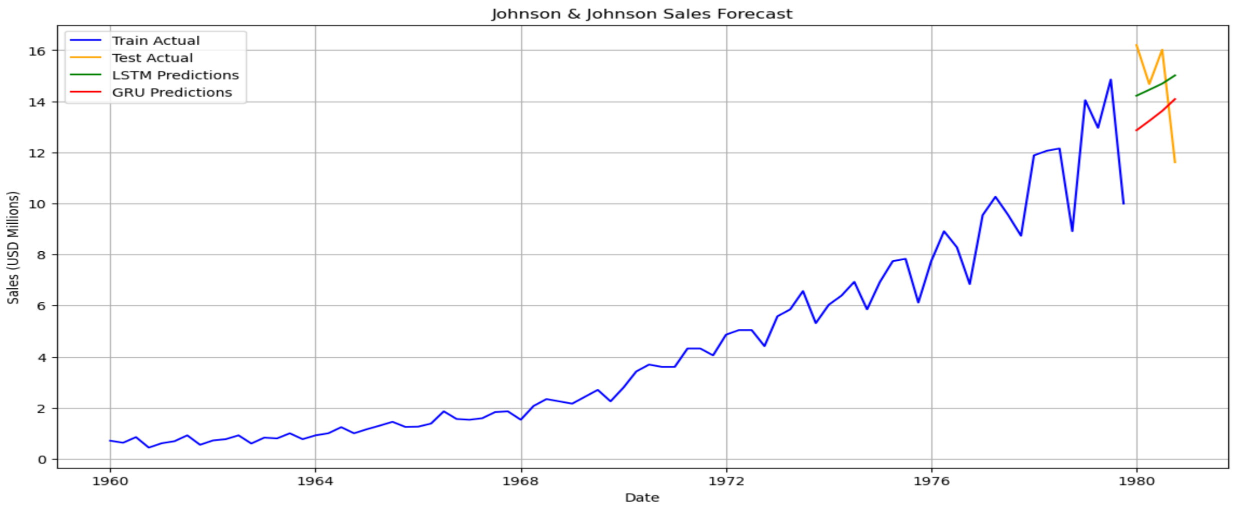
The **LSTM** model was built using **TensorFlow/Keras** with two stacked LSTM layers (50 units each) followed by a Dense output layer. It was compiled with the Adam optimizer and mean squared error loss, then trained for 50 epochs with a batch size of 32. After training, the model predicted the stock prices for the last 24 test sequences. Predictions were inverse-scaled to the original price range and evaluated using **RMSE, MAE, and MAPE** metrics.

Similarly, the **GRU** model was constructed with two stacked GRU layers and a Dense output layer using the same compilation, training parameters, and evaluation approach. The GRU model generally trained faster while producing comparable forecasting accuracy.

Both models forecasts were plotted alongside the historical Amazon stock prices for visual comparison.

Johnson & Johnson Sales Forecast Performance



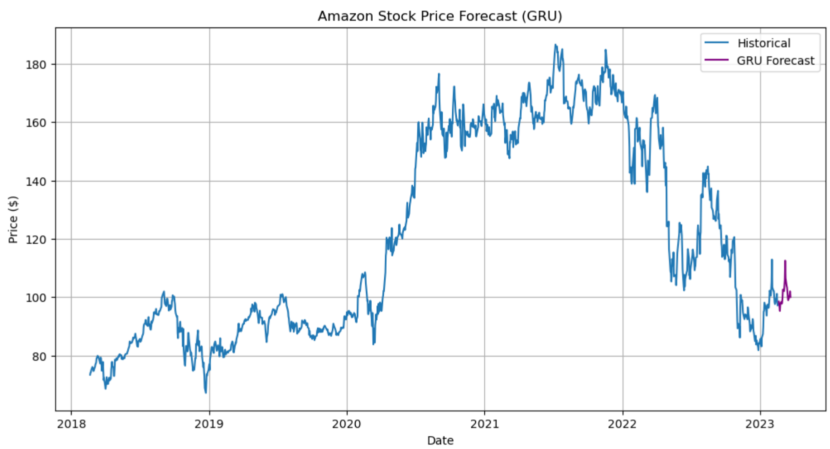


Model	RMSE	MAE	MAPE (%)
ARIMA	2.37	2.27	0.46%
LSTM	2.08	1.74	12.84%
GRU	2.51	2.41	16.68%

Interpretation:

- For JJ's quarterly sales (a stable and smoother time series), the ARIMA model performed significantly better than deep learning models in terms of MAPE, despite slightly higher RMSE.
- Both LSTM and GRU yielded higher percentage errors, suggesting that the deep learning models may be overfitting or misaligned due to limited data (quarterly series are much shorter than daily financial data).
- The MAPE of 0.46% for ARIMA is extremely low and indicates excellent accuracy, making it more suitable for this type of dataset.

Amazon Stock Price Forecast Performance



Model	RMSE	MAE	MAPE (%)
ARIMA	5.47	5.12	5.09%
LSTM	3.17	2.54	2.49%
GRU	3.22	2.49	2.47%

Interpretation:

- The ARIMA model captured the general trend but produced higher error across all metrics. Its limitation in capturing non-linearities is evident in Amazon's volatile price series.
- The LSTM model significantly improved performance, reducing MAPE to 2.49%, highlighting its ability to learn long-term dependencies and temporal structure.
- GRU offered nearly identical performance to LSTM, with a slightly lower MAE and MAPE, confirming its efficiency in learning from sequential data with fewer parameters.

Improvements and Future Work

- Cross-Validation: Implement time series cross-validation to better assess model generalisation and avoid overfitting.
- Hyperparameter Tuning: Use grid search or Bayesian optimization to fine-tune model parameters like layer units, learning rate, and dropout.
- External Features: Integrate additional features e.g.,economic indicators, trading volume to enhance model accuracy.

- Longer Forecast Horizon: Extend forecasts to longer periods and test model stability across varied time spans.

Conclusion

ARIMA performed best on Johnson & Johnson's stable, linear sales data, demonstrating its strength in modeling predictable and stationary time series. On the other hand, LSTM and GRU outperformed ARIMA on Amazon's volatile and non-linear stock prices, effectively capturing intricate temporal patterns and long-term dependencies. LSTM showed strong predictive power, while GRU offered nearly equal accuracy with reduced computational time, making it efficient for scenarios requiring faster training.

Reference:

Box, G. E., Jenkins, G. M., & Reinsel, G. C. (2015). *Time Series Analysis: Forecasting and Control.* *https://www.researchgate.net/publication/299459188_Time_Series_Analysis_Forecasting_and_Control5th_Edition_by_George_E_P_Box_Gwilym_M_Jenkins_Gregory_C_Reinsel_and_Greta_M_Ljung_2015_Published_by_John_Wiley_and_Sons_Inc_Hoboken_New_Jersey_pp_712_ISBN_*

Hochreiter, S., & Schmidhuber, J. (1997). *Long short-term memory.* Neural Computation. https://dl.acm.org/doi/10.1162/neco.1997.9.8.1735

Cho, K. et al. (2014). *Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation.https://aclanthology.org/D14-1179/*

Prudhviraju Srivatsavaya (2023). *LSTMvsGRU,Medium.*

https://medium.com/@prudhviraju.srivatsavaya/lstm-vs-gru-c1209b8ecb5a