University of Hertfordshire UH

School of Physics,
Engineering and
Computer Science

MSc Data Science Project

7PAM2002

Department of Physics, Astronomy and Mathematics

Data Science FINAL PROJECT REPORT

Project Title:

Stock Market Analysis

Student Name and SI:

Ajay Santhosh K V

23024049

Supervisor: William Cooper

Date Submitted: April 28th, 2025

Github: https://github.com/ajaysanthoshk/Final-Project.git

Word Count: 4829 (excluding references and appendices)

DECLARATION STATEMENT

This report is submitted in partial fulfilment of the requirement for the degree of Master of Science **in Data Science** at the University of Hertfordshire.

I have read the detailed guidance to students on academic integrity, misconduct and plagiarism information at Assessment Offences and Academic Misconduct and understand the University process of dealing with suspected cases of academic misconduct and the possible penalties, which could include failing the project or course.

I certify that the work submitted is my own and that any material derived or quoted from published or unpublished work of other persons has been duly acknowledged. (Ref. UPR AS/C/6.1, section 7 and UPR AS/C/5, section 3.6)

I did not use human participants in my MSc Project.

I hereby give permission for the report to be made available on module websites provided the source is acknowledged.

Student Name printed:Ajay Santhosh K V

Student Name signature:

Student ID number:23024049

UNIVERSITY OF HERTFORDSHIRE

SCHOOL OF PHYSICS, ENGINEERING AND COMPUTER SCIENCE

ABSTRACT

This research paper examined the performance of Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN) models for stock price prediction. Using S&P 500. The study utilizes a dataset of more than 619,000 observations of daily stock prices (open, high, low, close) and volumes for the largest S&P 500 constituents from 2013 to 2018. The study utilizes a dataset of over 619,000 observations of daily stock prices (open, high, low, close) and volume for large constituents of the S&P 500 for the years 2013 to 2018 and enhances the set with technical indicators including moving averages, volatility measures, and the Relative Strength Index (RSI). The results show the GRU model outperformed the other two models considerably, with a Mean Squared Error of (15.15), a Root Mean Squared Error of (3.89), a Mean Absolute Error of (2.87), and an R^2 score of (0.9927) which was the highest score for any of the three models by a large margin. The CNN model showed the least performance overall.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

1.1 Background of the Study

Stock market and other financial markets exhibit intricate complexity and incessant alteration as well as unpredictability. Various forecasting methods allow investors as well as market analysts to predict market fluctuations so they can achieve greater returns as well as reduce risks. Nonlinear nature of time series data is a serious complication to conventional forecasting models of ARIMA and GARCH models as stated by (Huimin, et al., 2024). Deep learning and artificial intelligence possess the potential to revolutionize finance in recent times. Recurrent Neural Networks with Long Short-Term Memory (LSTM) segments possess superior ability in anticipating time series data as well as stock market values. Long Short-Term Memory models outshine traditional models such as ARIMA in predicting stock prices in volatile market environments as stated by (Botunac, et al., 2024). The financial sector has embraced GRUs alongside CNNs as newer technology solutions. (Rahman, et al., 2019) documented that GRU models bring computation benefits which enable them to successfully predict market price fluctuations. Time series forecasting benefits from implementations of CNNs as described by (Wibawa, et al., 2022)We employed LSTM and GRU and CNN in this research to determine their individual forecasting abilities for S&P 500 constituents alongside an examination of their potential deployment in trading systems.

1.2 Problem Statement

Research has extensively analysed stock market predictions with machine learning methods but there is a lack of knowledge about deep learning architecture performance comparison on real trading data and existing studies lack connections between predictions and investments, causing practitioners in finance to lose interest.

1.3 Research Objectives

- To analyse the current body of literature on stock market prediction methodologies, highlighting the departure from traditional statistical approaches to advanced deep learning approaches.

- To perform exploratory data analysis (EDA) on selected stocks from the S&P 500 index to further identify features in trends, volatility, volumes, and technical factors.
- To create and implement LSTM, GRU, and CNN models for the prediction of stock prices based on historical time-series data.
- To investigate the interpretability of predictions generated by each model in the context of making financial decisions.
- To provide recommendations by assessing the EDA results and the model's performance to build better trading strategies.

1.4 Research questions

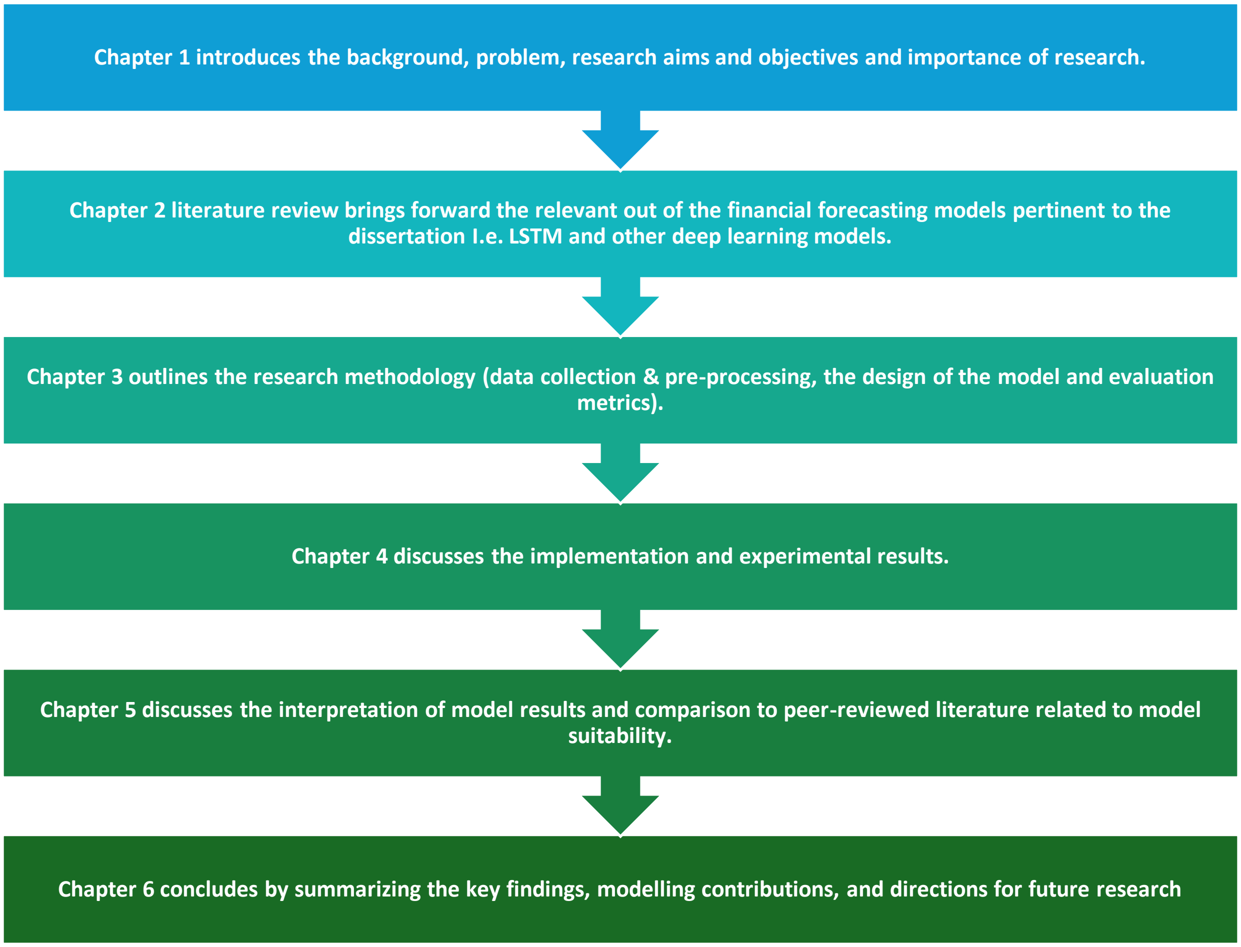
Which deep learning model (LSTM, GRU, CNN) is more capable of predicting a short-term price change in stocks belonging to the S&P 500, and how usable are these predictions as part of a trading strategy?

1.5 Importance of the Research

This research adds to academic research and industry practice in a few ways. Academically, this research provides a thorough geopolitical analysis of three popular deep learning models in financial time series forecasting. Practically, incorporating trading simulations helps to connect model forecast to real-world investment decisions. The results will likely benefit data scientists, financial analysts, fintech startups and investment firms before they create an AI trading tool.

1.6 Structure of the Dissertation

This dissertation is split up as follows:


Chapter 1 introduces the background, problem, research aims and objectives and importance of research.
Chapter 2 literature review brings forward the relevant out of the financial forecasting models pertinent to the dissertation I.e. LSTM and other deep learning models.
Chapter 3 outlines the research methodology (data collection & pre-processing, the design of the model and evaluation metrics).
Chapter 4 discusses the implementation and experimental results.
Chapter 5 discusses the interpretation of model results and comparison to peer-reviewed literature related to model suitability.
Chapter 6 concludes by summarizing the key findings, modelling contributions, and directions for future research

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

This literature review examines contemporary developments in stock market analysis, concentrating on the effectiveness of deep learning models, especially Long Short-Term Memory (LSTM) networks. The review discusses how these models improve prediction accuracy in an environment with volatility and market complexity; limitations are also noted and challenges in applying these models to financial forecasting are discussed.

2.2 Overview of Machine Learning in Financial Forecasting

Financial forecasting has changed radically through the incorporation of machine learning (ML) that offers the ability to analyse large datasets and recognize patterns that may not be observable with conventional/previous models (Raza, 2023). Stock price volatility together with time series patterns used to be determined by past forecasting systems through ARIMA and GARCH quantitative methods. Financial market non-linearity on a large scale remains unreachable for models which need linearity assumptions as their foundation. Prediction models progressed by incorporating decision trees and support vector machines (SVMs) and random forests from ML models while data extraction occurred according to (Kumar, et al., 2023). Strong neural networks enabled these ML models to extract flexible complex information from complex dataset through deep learning models. Deep neural networks (DNNs) and recurrent neural networks (RNNs) took models to another development stage because they enabled the computational interpretation of temporal relations within specific datasets according to (Goodfellow, et al., 2016). The association of LSTM networks proves to be particularly useful for financial time series forecasting since it maintains extended sequence memory alongside suitable architectural properties (Moghar & Hamiche, 2020). The industry has adopted LSTM networks instead of statistical methods for financial forecasting due to their adaptive properties in data detection and decreased need for human oversight of detailed relationships.

2.3 Comparative Understanding of LSTM, GRU, and CNN in Sequential Modelling

LSTM is a kind of Recurrent neural network proposes to overcome the gradient disappearance problem. i.e., Researches developed the more advanced LSTM networks because of the gradient vanishing issue that initially affected RNN. The gating mechanism of LSTMs allows attainment of better performance for stock price sequence at each and every processing step. The architectural design includes three important gates which function to regulate the data flow across the entire network systems between input and output as well as data storage and release (Houdt, et al., 2020). LSTM has enhanced ability to handle multi-dimensional data effectively through a technique of attention-based LSTM and hierarchical LSTM. Moreover, it can manage both short-term dependencies and long-term dependencies (Lindemann, et al., 2021). LSTM exhibit superior performance in terms of higher accuracy and reduced error rate than GRU while assessing EEG data for predicting future cognitive states (Rivas, et al., 2025). Furthermore, LSTM excels the base RNN model in forecasting price tendencies of financial time series data as per (Al-Selwi, et al., 2024)

In the same vein, Gated Recurrent Units (GRUs) provide a more streamlined and computationally efficient approach than LSTMs. GRUs consolidate the input and forget gate, resulting in a single update gate for the model, which decreases both model complexity and maintains long-term dependencies (Shen, et al., 2018). While GRUs appear to be faster and require less training resources, there are studies that suggest LSTMs perform better on more involved tasks like predicting financial time series (Rivas, et al., 2025)

At this moment. Convolutional Neural Networks (CNNs) are another area of research with regard to time-series data analysis. CNNs have strengths in recognizing local dependencies and are most often used for image processing. Some research supports CNNs being useful in improving local pattern detection in hybrid model performance with either LSTM or GRU in financial applications (Singh, et al., 2023)

2.4 Applications of LSTM, GRU, and CNN in Stock Market Forecasting

LSTM networks have wider application areas in the numerous fields of finance, healthcare, construction and science. For instance, a study of (Malashin, et al., 2024) demonstrates that LSTM efficiency in forecasting time-series trends as well as model building of sequential data for the purpose of comprehension of intricate molecular structure and transient behaviours of polymeric materials (Malashin, et al., 2024). Across many recent studies, LSTM networks have consistently exhibited successful performance. The effectiveness of LSTM models in stock market prediction was enhanced by the use of RSI and EMA as per the 2023 findings by (Dhokane & Agarwal, 2023). Forecasts become extremely accurate when (Kaladevi & Thyagarajah, 2019) integrates big-sweep data comprising macro data as well as social market sentiment data datasets in LSTM models. The flexibility of the LSTM model lies in its capacity to process multiple data forms because it functions well across different preprocessing techniques such as normalization and feature selection.

GRU-based models also converge faster and require less training time, which are helpful for real-time prediction tasks, albeit with slightly lower—but generally similar—accuracy as LSTM (Mienye, et al., 2024) . CNNs or when used alongside LSTM or GRU, are also useful for extracting high-level features from financial log-return time series. In particular, CNNs tended to be a good fit where additional data that measured sentiment or macroeconomic data was also used (Oko-Odion, 2025)

2.5 Challenges and Future Directions in LSTM-Based Stock Market Forecasting

The main drawbacks exist alongside the advantages within LSTM networks. LSTM networks require considerable datasets made up of high-quality information before they become operational. The main drawback of LSTM models arises from their complex computational nature since most LSTM models need extended training periods together with substantial computational resources (Ahmed, et al., 2023). This problem presents the greatest challenge to high-frequency trading practices that need to execute trading decisions immediately. The black box operational nature of LSTM networks makes interpretation difficult specifically in finance applications because the field requires regulatory compliant transparency (Freeborough & van Zyl, n.d.). One weakness of LSTM models consists of their delayed capacity to detect unexpected events along with their inability to adjust promptly because of incidents like political regime changes and market crashes or pandemics. The authors of (Dutta, et al., 2020) suggest reinforcement learning models should replace LSTM models because reinforcement learning allows real-time responses to market developments. GRUs are a simpler version, reducing training time, but may lack granularity in the model. The interpretability of models is a common problem with LSTMs and GRUs; no more so than in regulated industries (such as finance). CNNs seem more efficient for extracting features, but they cannot capture long term dependencies unless they function alongside some RNN-type architecture. Future directions should merit considering hybrid models, where the advantages of multiple networks can be incorporated. For example, (Lu, et al., 2019) noted in a systematic review that combined LSTM and CNN can yield better identification of spatiotemporal features. Researchers introduced attention mechanisms by (Chitty-Venkata, et al., 2023) to let models focus on important time steps which simultaneously improves both model discriminative power and interpretability.

2.6 Conclusion

LSTM models outperform ARIMA and GARCH models in their ability to forecast short-term trends in financial markets, but are limited because of their high computational power requirements, lack of interpretability, and other challenges. GRU models have lower computation costs, and CNNs have expertise in feature extraction. Incorporating attention mechanisms and big data, and hybrid models result in the highest levels of accuracy and flexibility in AI-supported stock market forecasting.

CHAPTER 3: METHODOLOGY

3.1 Introduction

The research methodology for evaluating LSTM networks during S and P 500 stock price trend prediction operations appears in this chapter. It details the tools, the research design, the data collection, data preprocessing, exploratory analysis, model construction and training, model selection, and model evaluation, and it emphasizes both the technical contributions and the involvement in the forecasting project.

3.2 Tools and Environment

For the implementation of this project, the Python programming language was utilized within a Google Colaboratory (Colab) environment, leveraging its cloud-based GPU capabilities for deep learning tasks.

Several key Python libraries facilitated the work:

- Pandas and NumPy for effective data manipulation and preprocessing.
- Matplotlib and Seaborn for comprehensive data visualizations that aided in understanding data distributions and relationships.
- Scikit-learn for essential metrics, data splitting, and preprocessing functions.
- TensorFlow/Keras, the core libraries for building and training deep learning models.
- Google Colab offered both flexibility in coding and access to computational resources necessary for the deep learning processes.

3.3 Research Design

The project design consisted of a quantitative experimental approach for establishing deep learning models to predict stock prices. The main purpose of this study focused on analysing the predictive abilities of LSTM models when processing historical stock price data to forecast short-term market movements. In adopting an experimental design, we were able to control for the input variables in a systematic way, facilitating the observation of outputs and evaluation of model performance under different conditions.

3.4 Data Collection and Preprocessing

The dataset used in this project is comprised of daily historical stock prices from S&P 500 firms from 2013 to 2018. It originates from Kaggle, and the data itself is sourced from Yahoo Finance, containing several attributes such as open, high, low, closing prices, and trade volume. It was selected for this research project given its wide time frame that includes several market cycles, which are crucial in developing meaningful predictive models. There are also enough data points to reasonably examine trend and correlations in the data set. In order to ensure the quality of the data, it was pre-processed, which involved a few steps, namely: cleaning any missing values and outliers, followed by Min-Max normalization to range the features from 0 to 1. Feature engineering included calculation of the Relative Strength Index (RSI).

3.5 Ethical Considerations

In terms of ethical considerations, this dataset does not include personal data, which means that it does not reference GDPR guidelines, and therefore does not need ethical approval from UH. This data has been made publicly available and meets regular usage protocols because it has been posted on Kaggle, which uses a Creative Commons license permitting the use for research purposes. Yahoo Finance has shown that their methods of data collection are legitimate, so we can be confident that the data was collected ethically and with informed consent from the individual contributors. As such, there are no ethical issues with the use of this data in this project.

3.6 Exploratory Data Analysis (EDA) Methodology

The purpose of the exploratory data analysis was to find the structural patterns and movements in the stock market. The process began with the performance charts of closing price only for major companies (e.g. AAPL, GOOGL, AMZN, MSFT) and it was evident that we could identify growth trajectories and cyclical behaviours over year periods in time. After this exploratory phase, we focused on the distribution of daily returns for Apple Inc. and used histograms with kernel density estimates (KDE) to investigate the frequency and the risk of extreme returns. Next, we analysed rolling volatility using a specified time horizon of 30 days, thus identifying stocks with very high or low volatility over selected periods. This analysis also enabled us to relate the volatility variation to an event. An example event would be the quarterly earnings announcement of its fundamentals. We used the variance of returns to assign a sector cluster into volatile segments with no identifiable time horizon. We calculated moving averages (MA50 and MA200) to review price smoothing and to identify possible price trend reversals. We incorporated Bollinger Bands to assess volatility compared to price action. The exploratory data analysis provides a strong base to conduct more explorative predictive modelling and therefore investment strategies with the stock market.

3.7 Model Building

To build the model, the time series data was reshaped into a format suitable for the LSTM input. The dataset was then split into 80% training and 20% testing datasets for evaluation of the models. Before model training, the input features were reshaped into compatible input dimensions using convolutional neural networks (CNN). Moreover, dropout layers set with a dropout rate of 0.2. in order to organize both model structure and data pipeline before continuing with architecture selection.

3.8 Model Selection and Architecture

This research evaluated the predictive power of three deep learning models namely LSTM, Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN).

LSTM Model included two stacked LSTM layers containing 50 units in the first layer and 25 units in the second layer after which it added a Dropout layer and a final Dense output layer with one neuron for predicting the next closing price. The model utilized the ReLU activation function together with an Adam optimizer set to 0.001 learning rate for training with a Mean Squared Error (MSE) loss function.

Reducing computational complexity was achieved through GRU cells in this model while maintaining the ability to capture sequential information from time sequences.

The 1D convolutional layer inside the CNN Model contained 64 filters for local temporal detection. The CNN model received development through its use as a benchmark reference for time-series data prediction even though it lacks strict sequencing techniques.

The training was conducted for 10 epochs with 64 elements per batch and validation loss-based early stopping to minimize overfitting.

3.9 Evaluation Metrics

The analysis of forecasting effectiveness involved multiple metrics evaluation.

- Training activities relied mainly on Mean Squared Error (MSE) as the primary loss metric to measure prediction-based numerical activities.
- The evaluation metrics of Root Mean Square Error (RMSE) along with Mean Absolute Error (MAE) determined the scale-based error predictions and average deviation assessments.

(Chicco, et al., 2021)

3.10 Conclusion

A framework for evaluating deep learning models used in stock market predictions provided detailed instructions about data preprocessing and exploratory analysis and model construction and evaluation measurement methods. RSI used in combination with complex time series models developed more accurate predictions for the forecasting models. Future research can start from this point to optimize LSTM alongside other sequence models.

CHAPTER 4: IMPLEMENTATION AND RESULTS

4.1 Data preparation

The prediction of time series demanded extensive work in data quality and transformation as preparation step. The initial exploratory data analysis proved that multiple important columns including 'open', 'high', 'low', 'close', 'volume' contained missing values that demanded thorough cleaning procedures. Following this, missing values in the 'open', 'high', and 'low' columns were filled with their respective column averages to maintain data integrity.

4.2 Data Exploration and Market Behaviour

Exploratory Data Analysis (EDA) played a vital role in revealing significant insights and behavioural characteristics of S&P 500 equities between 2013 and 2018. Through systematic exploration of closing prices, volatility, trades, technical indicators, and inter-stock correlations, EDA helped put behavioural and market contexts around the eventual model placed on data.

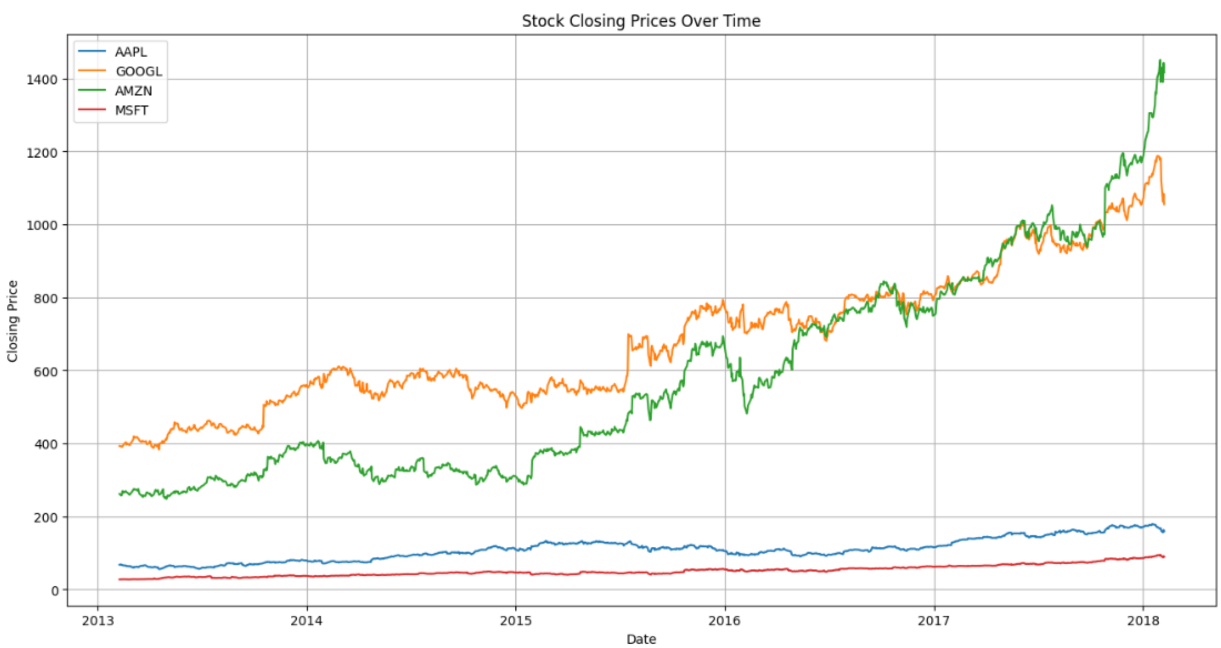


Figure 1: Stock Closing Prices Over Time

The figure 1 shows closing price trends for four large technology stocks (AAPL, GOOGL, AMZN, MSFT) from 2013-2018, all showing steady increases that coincide with the larger bull market. AMZN and GOOGL follow similar price trends to the others, but their rise is much steeper, likely due to their position as dominant innovators (e.g., iPhone rollout, the emergence of AWS). MSFT and AAPL shadows closely along the price trend with their performance being a result of the high market performance of the cloud market and digital advertising.

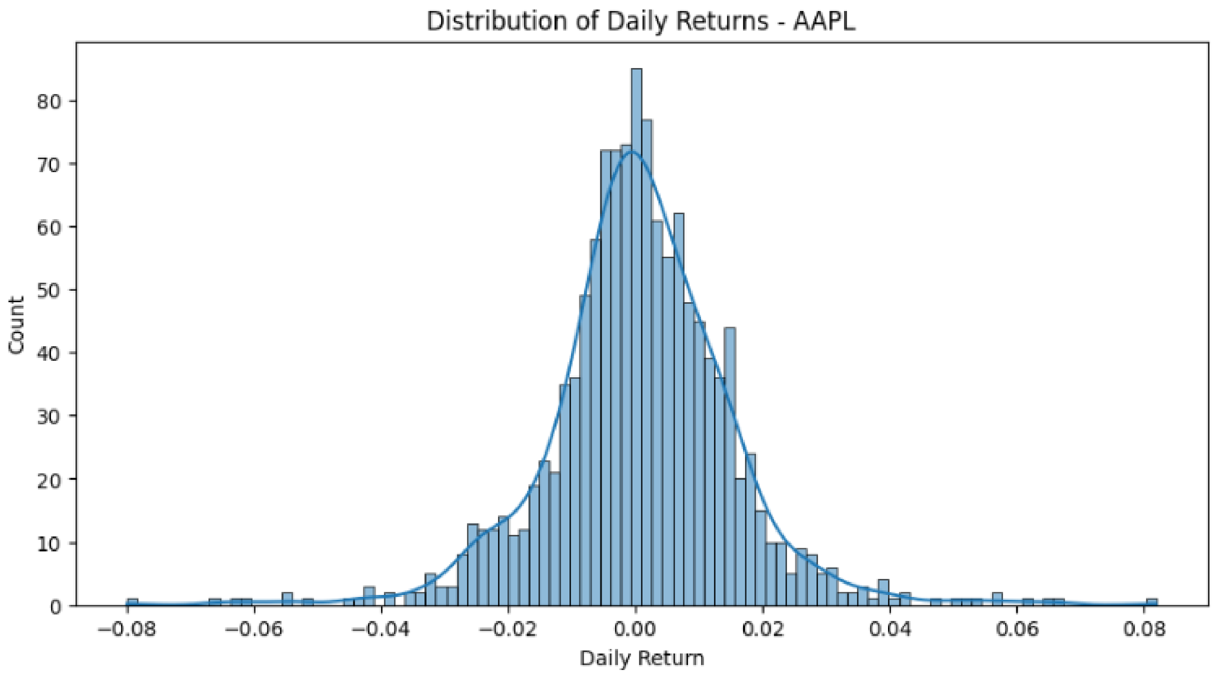


Figure 2: Distribution of Daily Returns – AAPL

This histogram (figure 2) illustrates the distribution of daily returns for AAPL (2013-2018). The daily returns fall into clusters that are typically near zero which means that the prices fluctuate little (in most cases within ±2%). The four fat tails at ±6-8% on the far ends of the distribution mean that there were occasional volatility spikes in pricing (usually caused by shock to the earnings from some events (earnings shocks) or movement in the broader market perceptions). The leptokurtic shape is a characteristic of stock returns, specifically that market outcomes and risk cannot be modelled well using a normal distribution which should be considered in your predictive analysis and forms of portfolio choices.

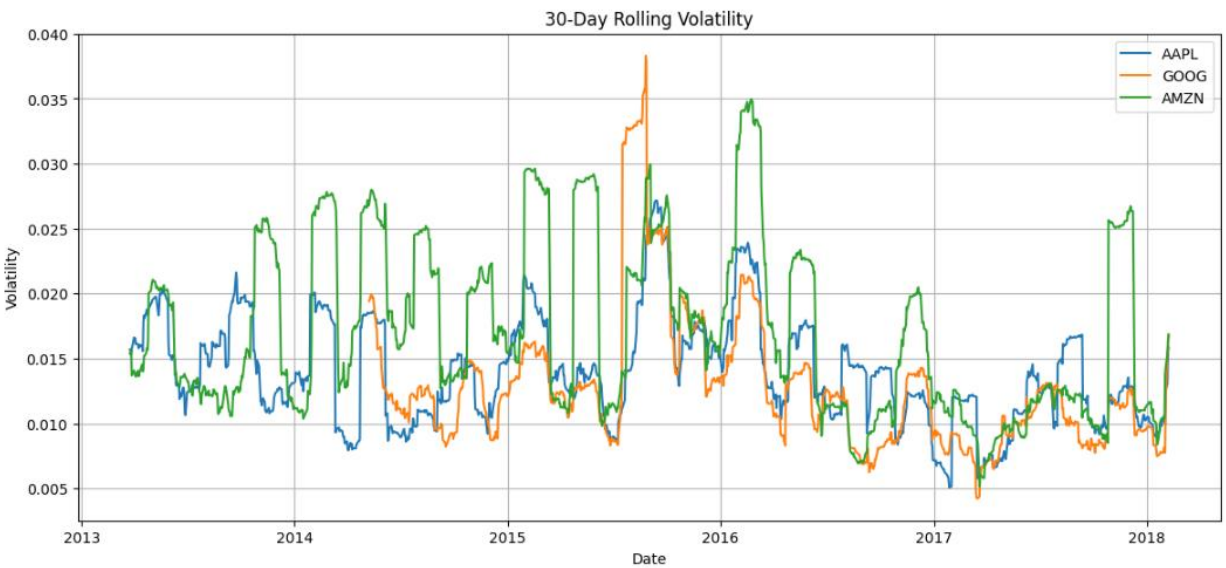


Figure 3:30-Day Rolling Volatility

The following chart outlines the 30-day rolling volatility for AAPL, GOOG, and AMZN from 2013-2018. The volatility varies from low risk (0.005) to higher risk (0.040) and fluctuates through different periods of uncertainty and therefore stability in the larger market (and economy). The peaks in volatility for each stock indicate events that impacted the stock's price, and the troughs in volatility indicate calmer times in the market. All three stock price traces reflect the action in the market; however, each stock performed within its own company context.

Figure 4:AAPL - Moving Averages

The chart shows the closing price of AAPL and its 50 & 200-day moving averages (MA) from 2013-2018. The 50-day MA demonstrates the stock's short-term trends and the 200-day MA shows the longer-term direction. Crossovers (50-day MA above or below the 200-day) represent bullish or bearish phases. The constant 50-day MA above the 200-day MA demonstrates a constant indication of upward movement, which was what was happening, especially during this period with AAPL.

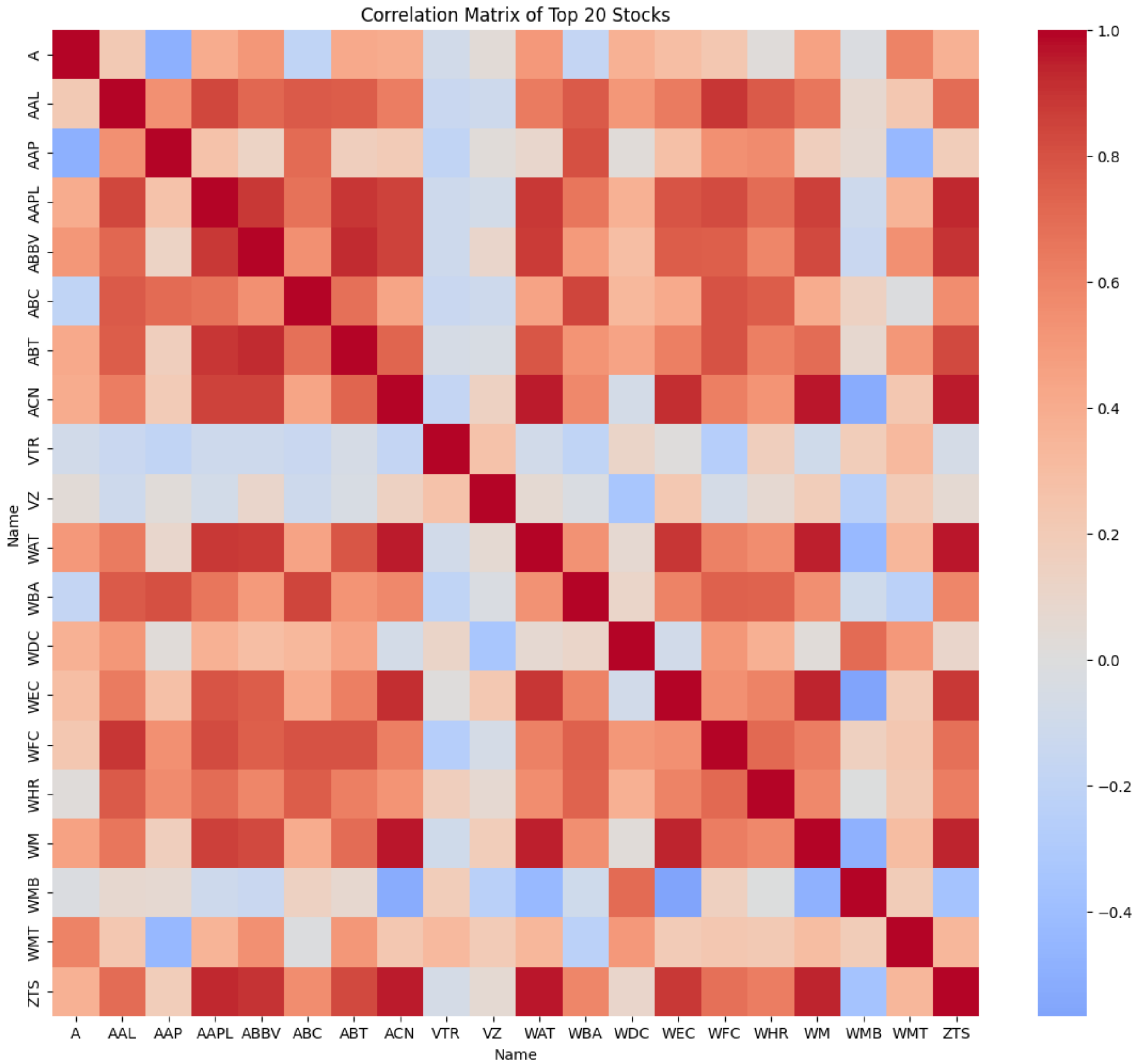


Figure 5: Correlation matrix of top 20 stocks

The correlation matrix provides some significant details about these 20 stocks. The stocks with a strong positive correlation (red) tend to move together, and blue indicates where they moved opposite from each other. The correlation matrix shows a clear cluster of particular stocks which have high correlations (AAPL, ABBV, ABT, ACN). At the same time, it also shows several stocks (VTR, VZ & WMB) that have negative correlations (indicate random or growing correlation) with many others.

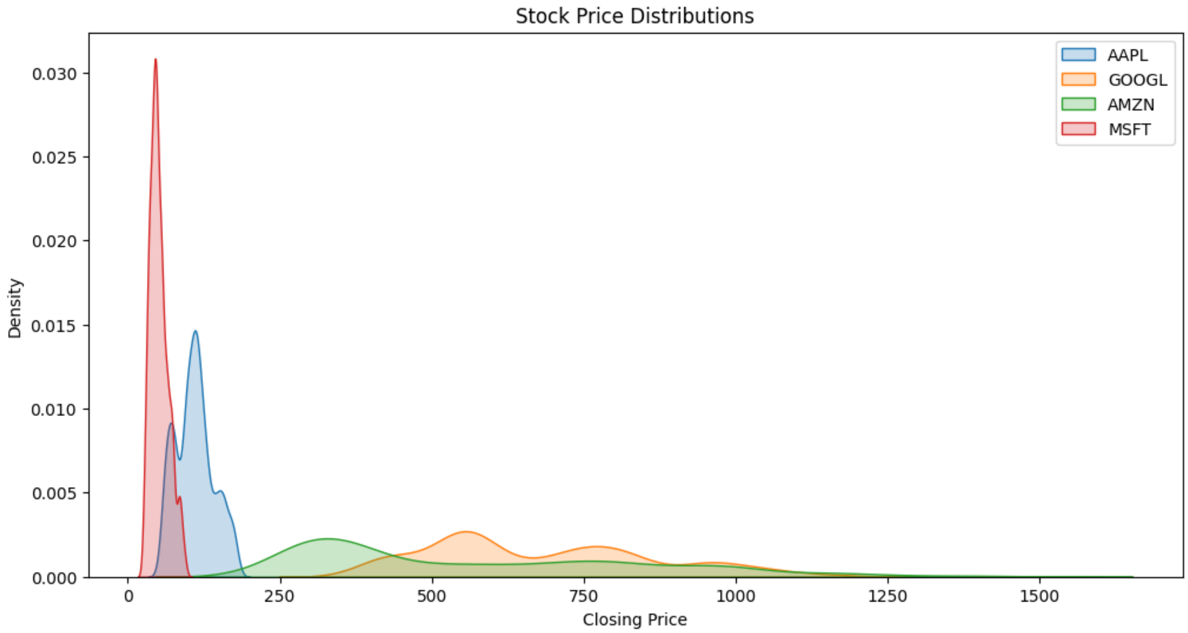


Figure 6: Stock Price Distributions

This chart shows the density distributions of the closing prices (up to $1500) of AAPL, GOOGL, AMZN, and MSFT. The peaks show where the price tends to accumulate values, between the accumulation prices: AAPL and MSFT tend to accumulate lower prices (≤750), GOOGL and AMZN gather prices at higher prices (≥750). Curves with a narrow peak (MSFT) could demonstrate price stability, while if the curve has a wider spread (AMZN) it could imply a more volatile price - which was observed in previous year's journey.

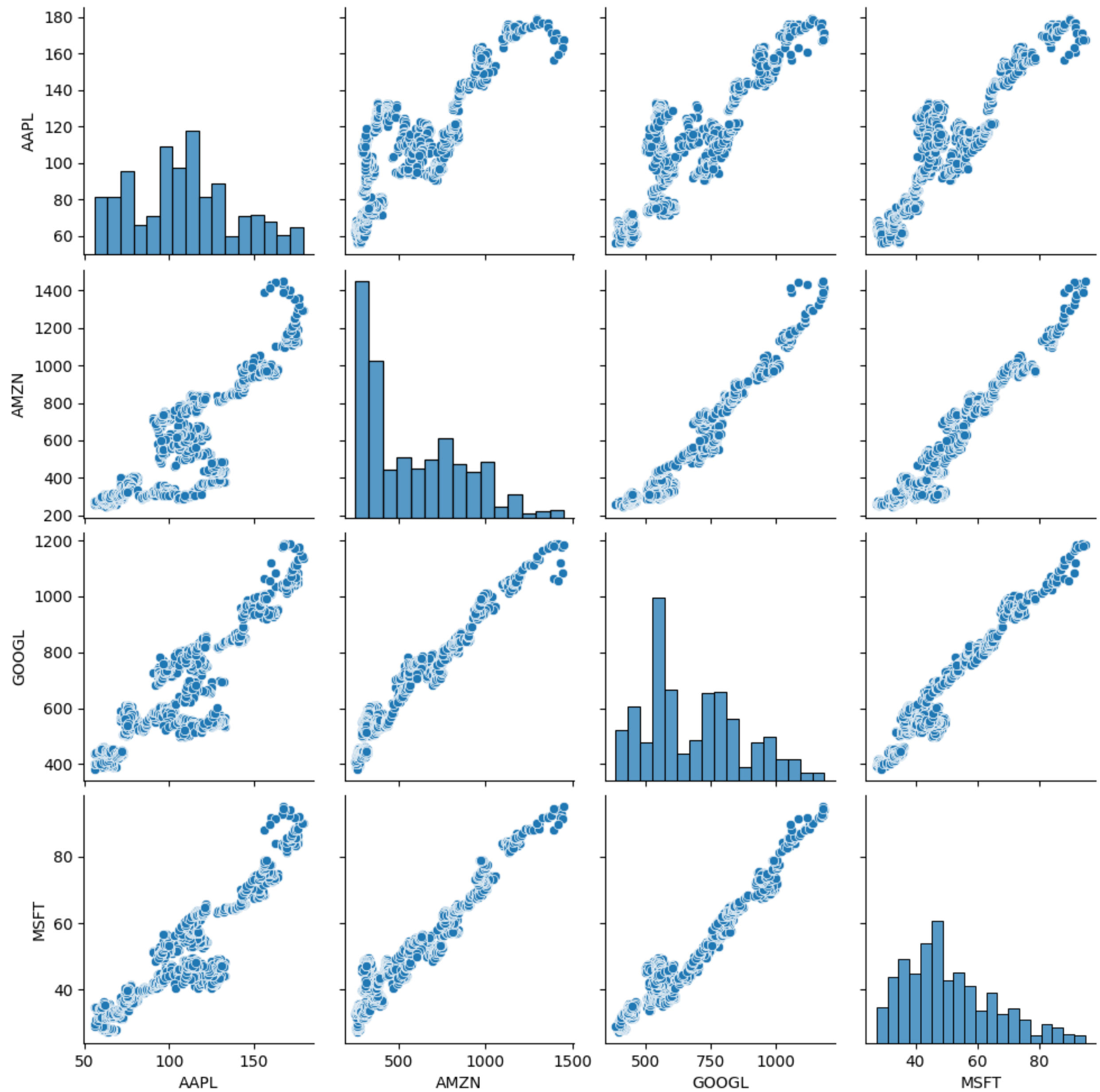


Figure 7: Scatter Plot Matrix

The scatterplot matrix exhibits strong positive correlations between AAPL, AMZN, GOOGL, and MSFT stock prices, with noticeable upward trends suggesting they move together as stocks. The histograms pictured indicate there are different distributions of prices, with AMZN prices tending to be the highest prices. However, they all show clustered movements in pricing at different price points across their histories.

Figure 8: Candlestick chart of AAPL

AAPL's overall 2013–2018 candlestick chart reflects a noticeably strong upward trend, which we attribute to product launches (e.g., iPhone 6) and earnings. The periods of sustainable bullish momentum (green candles) well surpassed the periods of bearish momentum (red candles), which were mostly short-lived.

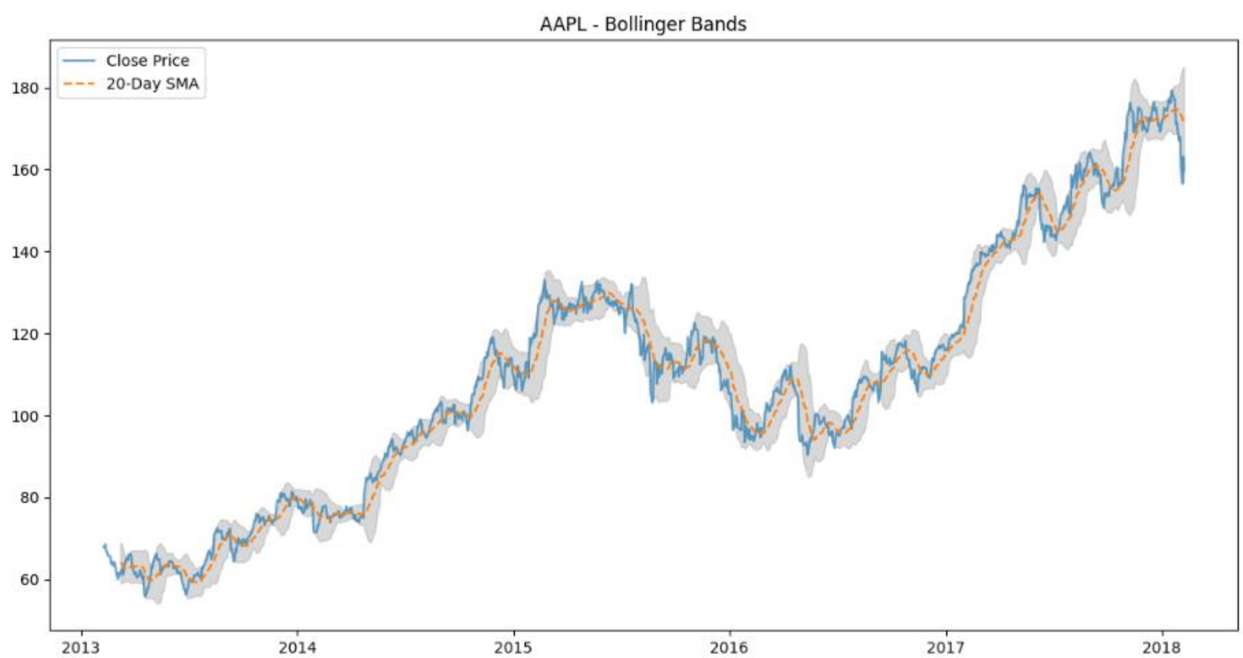


Figure 9: Bollinger Bands – AAPL

The chart depicts AAPL's close price against its 20-day SMA (baseline for Bollinger Bands) from 2013–2018. When stock prices sustain above the 20-day SMA implies bullish momentum for

AAPL. The Bollinger Bands usually contract which indicates low volatility or they expand which indicate high volatility.

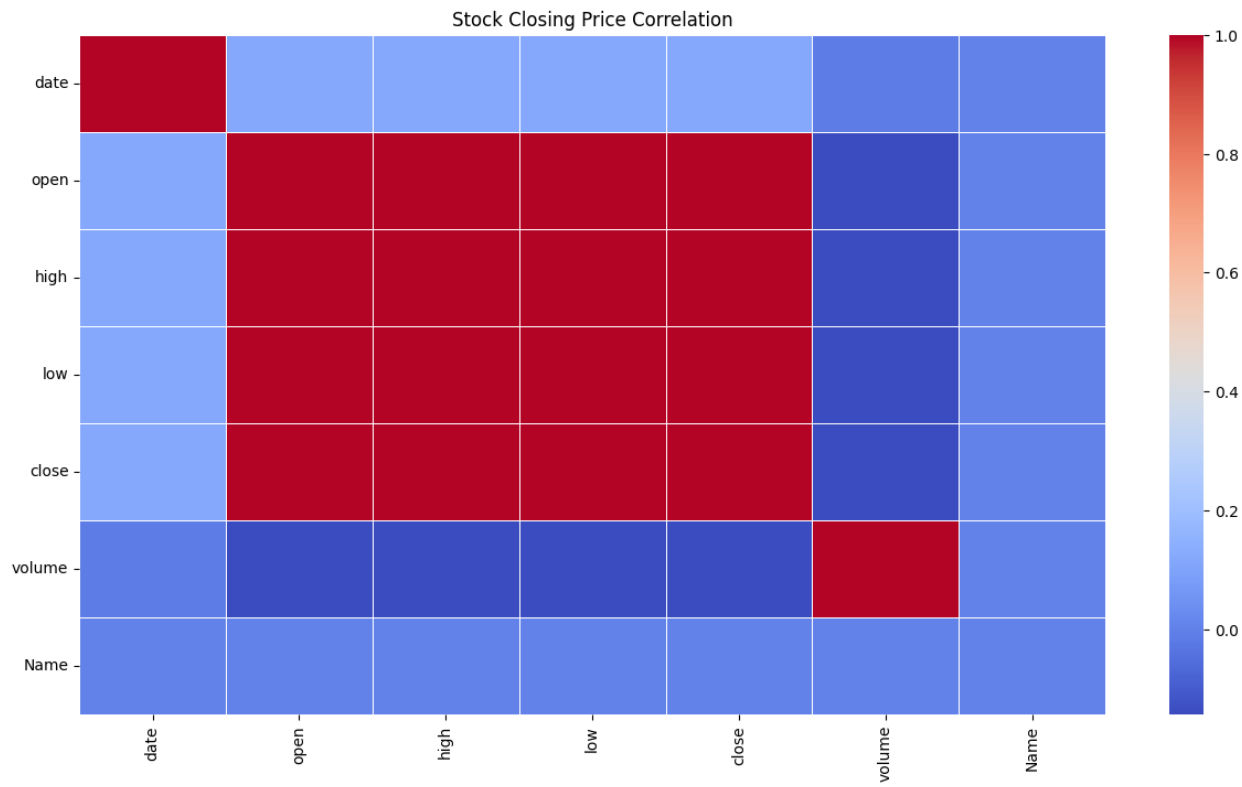


Figure 10: Stock Closing Price Correlation

The correlation matrix provided demonstrates good positive correlations (dark red) of the price metrics (open, high, low, close) indicating they move together. The date and volume had negative correlations (in blue) with the price metrics on the chart and the volume had a very low correlation to anything else. The "Name" property showed negative correlations across all other variables meaning the stock identifiers are not related to the numerical data.

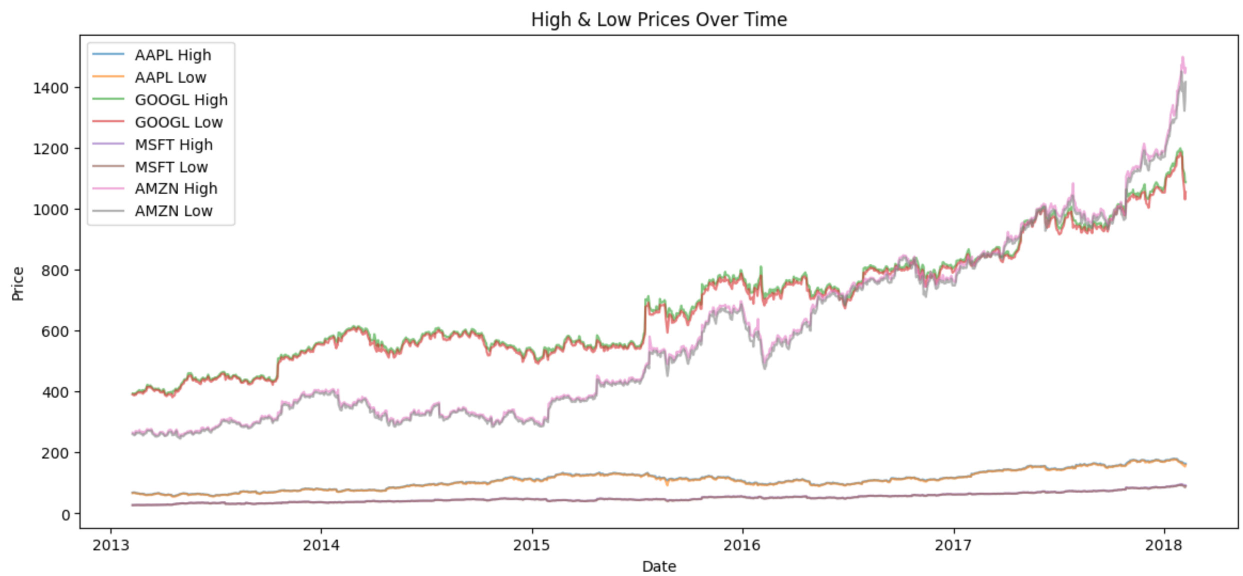


Figure 11: High & Low prices Over Time

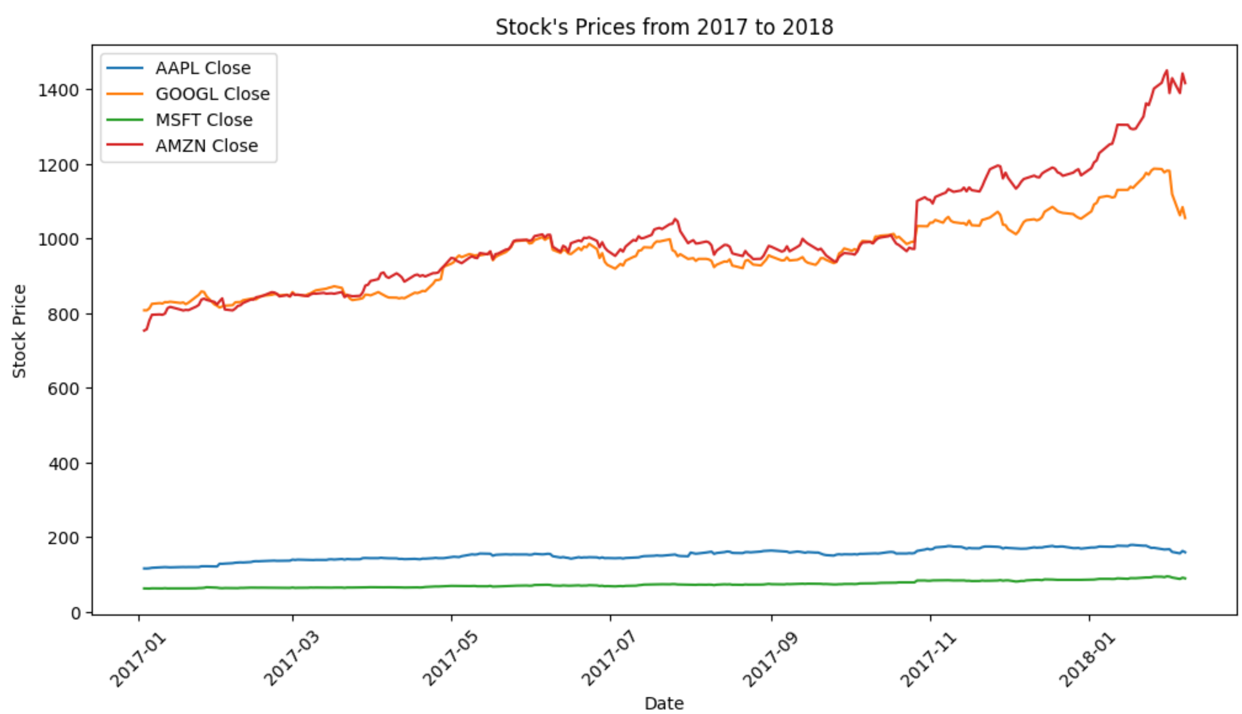


Figure 12: Stock's prices from 2017 to 2018

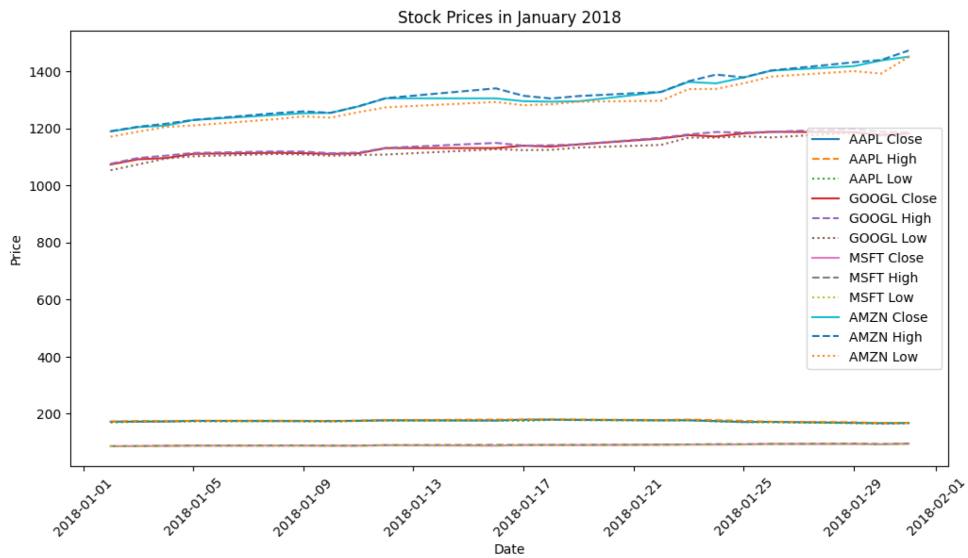


Figure 13: Stock Prices in January 2018

The three charts (below) illustrate stock trends of (1) AAPL, (2) GOOGL, (3) MSFT, and (4) AMZN. In Figure 13 the prices (high and low) from 2013--2018 were being reviewed. Except for an almost flat trend from 2014--2015 the stocks were showing growth over the time with AMZN and GOOGL making the strongest case. Figure 14 drills down to 2017--2018 closing prices for the stocks, put a noticeable upward trend on display (most notably AMZN). Figure 15 is a view of January 2018 to check for the high, low, and closing prices, the stocks were trending up and the price was being traded in a narrow range indicating short term stability in the market and investors were hopeful.

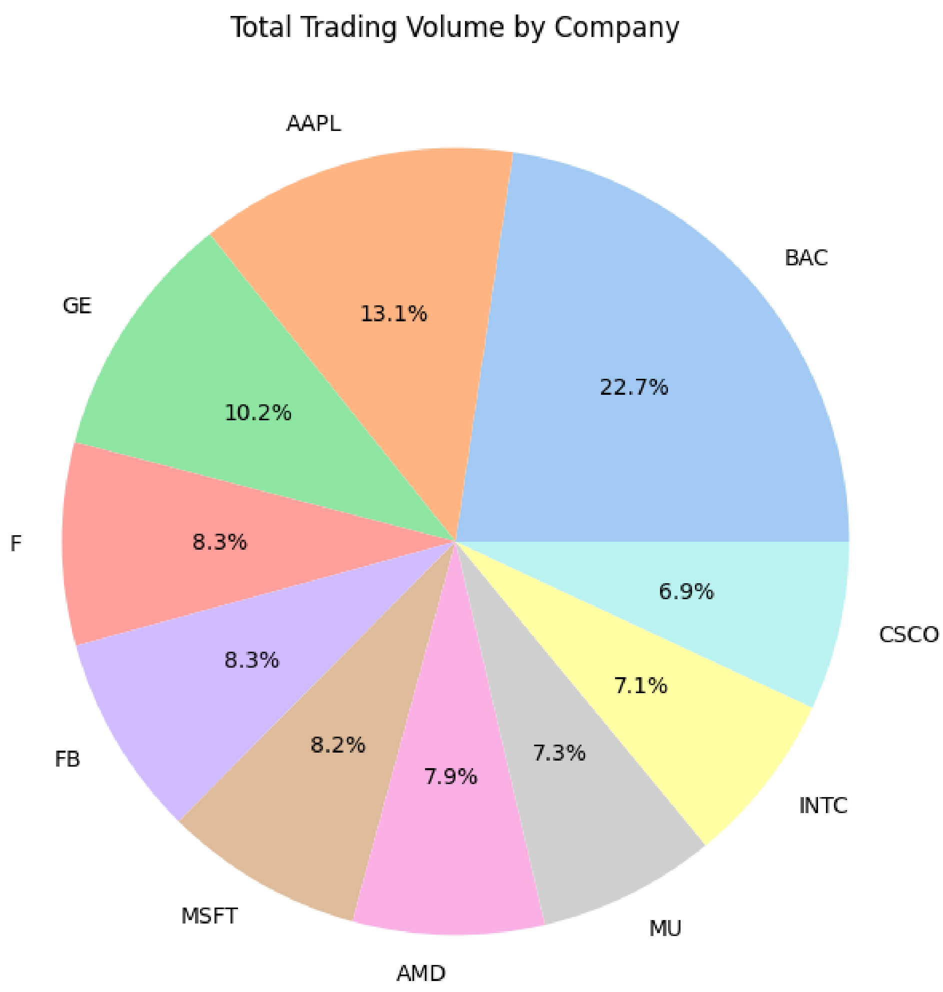


Figure 14: Trading Volume by company wise

Bank of America (BAC) has the highest trading volume (22.7%) in the pie chart, followed by Apple (AAPL) with 13.1% and General Electric (GE) with 10.2%. The remainder of the companies (Ford, Facebook, Microsoft, Advanced Micro Devices, Micron Technology, Intel, and Cisco) were between 6.9 - 8.3% of the total volume. The graphs provide evidence the financial services and tech sectors are the most active in terms of dollar volume traded with BAC having nearly double the trading volume of most stocks.

4.3 Model Implementation and Training

I designed and developed three deep learning models based on LSTM, GRU and CNN, using TensorFlow and Keras:

LSTM Architecture: Two stacked LSTM layers (50 memory units, 25 memory units), with ReLU activation, dropout, to alleviate overfitting, in place, and a dense layer to the stock price output.

GRU Model: Same architecture as the LSTM model, but I used GRU in place of LSTM layers for better efficiency.

CNN Model: The CNN model used a 1D convolution layer with 64 filters to capture short-term pattern movement. All models were trained with an Adam optimizer (learning rate = 0.001) and a batch size of 64, to 10 epochs, using early stopping to prevent over-fitting.

The LSTM model trained with stable convergences around epoch 10, when training and validation loss dynamics were similar indicated minimal overfitting.

4.4 Evaluation and Results Interpretation

Table 1: Model result's table

Model	MSE	RMSE	MAE	R2 score
LSTM	**21.4166**	**4.6278**	**3.7775**	**0.9897**
GRU	**15.1552**	**3.8930**	**2.8725**	**0.9927**
CNN	**34.7146**	**5.8919**	**4.3117**	**0.9834**

The table presents performance metrics for three models—LSTM, GRU, and CNN—used in stock price prediction. GRU outperforms the others with the lowest Mean Squared Error (15.15), Root Mean Squared Error (3.89), and Mean Absolute Error (2.87), indicating higher prediction accuracy and minimal error. It also has the highest R^2 score (0.9927), suggesting it explains the most variance in the data. LSTM follows with strong results, while CNN shows the weakest performance among the three, with higher errors and a lower R^2 score (0.9834). Overall, GRU demonstrates superior effectiveness for time series forecasting in this context.

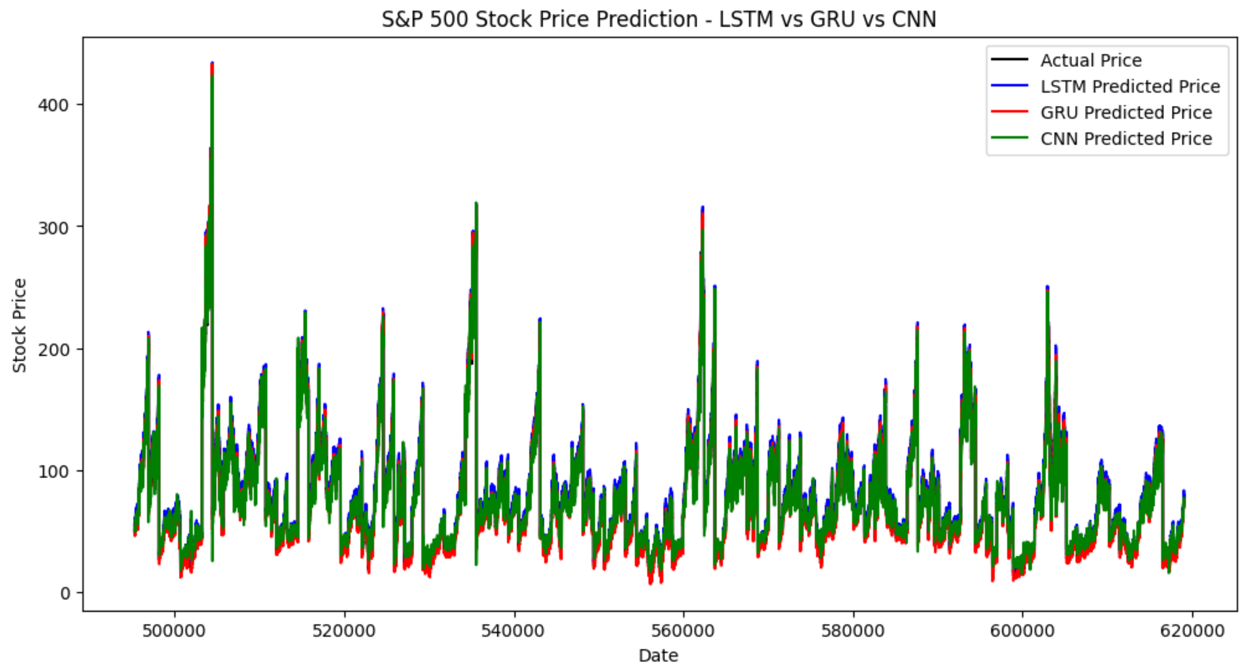


Figure 15: S&P Stock Price Prediction - LSTM vs GRU vs CNN

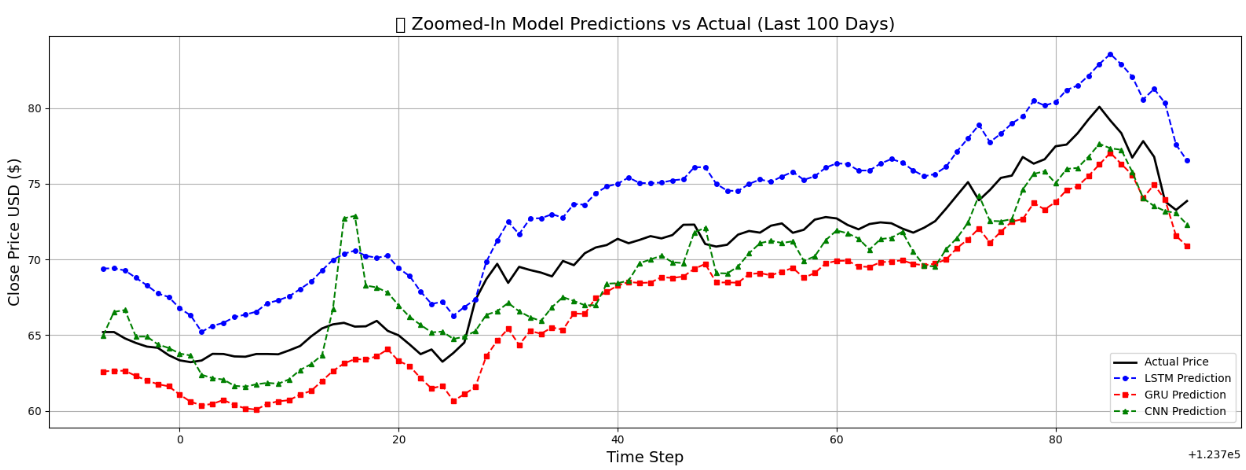


Figure 16:Zoomed-in Model Predictions vs Actual

The graph (from figure 15 and 16) compares actual stock prices with predictions from LSTM, GRU, and CNN over the last 100 days. GRU closely follows the actual trend, showing the best alignment.

CNN also tracks fairly well but with slight deviations. LSTM overshoots significantly, especially during peaks, indicating lower short-term accuracy.

4.5 Conclusion

These results serve as a detailed answer to project objectives through comprehensive performance evaluation with multiple analytic metrics. The evaluation of three dominant LSTM, GRU and CNN variations with real-time financial data confirmed the practical usefulness of newly proposed prediction models. The evaluation process not only sufficiently answers the research question, but it also provides an avenue for improving financial forecasting applications beyond the purposes of the study.

CHAPTER 5: DISCUSSION

5.1 Summary of EDA findings and Model Outcomes

Upward Price Trends: The closing prices of major tech stocks, including AAPL, GOOGL, AMZN, and MSFT, displayed a consistent upward trajectory, reflecting robust market sentiment and confidence in the technology sector.

Volatility Patterns: The 30-day rolling volatility plot indicated periods of both high and low volatility, which were closely aligned with major economic events, such as earnings reports and market shifts, revealing how external factors influence stock performance.

Daily Returns Distribution: The distribution of daily returns for AAPL exhibited a leptokurtic shape, characterized by fat tails, indicating a tendency for extreme returns—both positive and negative—highlighting the inadequacy of normal distribution assumptions in financial data.

Correlation Across Stocks: The correlation matrix showed many of the stocks (such as AAPL, ABBV, ABT) to be strongly positively correlated, generating possible overlap for portfolios and providing opportunities for strategic diversification that would help with risk management.

Seasonality Indications: Certain stocks exhibited repeated seasonal trends in their price changes indicating that price behavior is often predictable during the various times of the year, often influencing potential trade decisions.

Volume activity analysis: Stock volume activity analyses indicated spikes in trading volumes often coincide with stock market news or company announcements indicating volume can serve as a leading indicator of price movement.

Moving Averages & Trends: Our analyses of moving averages, (MA50 and MA200) indicated occurrences of bullish momentum (a period when the terms moving average were larger than the long-term one), procurement of means that assist in indicating the trends that exist within the various possible methods during the analysis's periods.

Bollinger Bands Analysis: The Bollinger Bands analyses primarily illustrated stable periods of price and volatility with the price hugging the bands during low volatility periods indicating a possible breakout.

GRU: The greatest profitability was achieved with the Gated Recurrent Unit (GRU) model, indicating a number of variables indicative of good performance and rated any 15.15 for means squared error (MSE), the root means squared error (RMSE) was rated at 3.89 in error and a high R^2 score of 0.9927 showing high accuracy in its predictions.

LSTM Findings: The Long Short-Term Memory (LSTM) neural network has been well-documented for its ability to handle long dependencies, but in our examples, it was unable to surpass GRU performance, indicating the feature complexity for these specific examples may not have warranted the advanced structure of LSTM.

CNN Drawbacks: The Convolutional Neural Network (CNN) appeared to produce the weakest performance, which was anticipated, as they are intended for spatial data that can be ranked (e.g., images) rather than time-series of sequential data. Although CNNs can be adapted for use with RNN for the purpose of feature extraction, their use in stock forecasting was less effective than in isolation.

Model Training Efficiency: In terms of training computational time, the GRU model was more efficient than the LSTM. This allows for quicker enhancement iterations of the model leading to improved use in high-velocity trading scenarios by enabling rapid model-specific adjustments.

5.2 Results Comparison with Literature Review

The results from our model evaluation are aligned with general trends found in the current literature. For example, LSTM networks are cited as one of the best methods for financial time series due to their ability to retain significant amount of past information (Moghar & Hamiche, 2020). Our analysis indicates that LSTM networks do have unique capabilities in capturing data complexities; however, in the terms of computational efficiency and speed for common forecasting tasks, GRU models may be an effective alternative. Some studies reported faster convergence of GRU models and less training time overall (Mienye, et al., 2024). Our results confirm these findings, as GRU not only had lower prediction errors, but GRU lead to much quicker iterations while training which is relevant with the need for models to quickly adapt to changing conditions—a highly sought-after attribute in finance.

In comparison, we observed that our CNNs' performance in our models were not as strong as LSTM and GRU. CNNs are ideal for image processing, and local dependence, but their use for sequential modelling within time series is less common. Our results are aligned with partial conclusions from the literature, that hybrid models that combined time-series forecasting with feature extraction in time-series forecasting (Oko-Odion, 2025) might produce the best results.

5.3 Identifying the Best Model Based on Project Objectives

In light of our project aims - i.e., stock price prediction in an uncertain environment - the GRU model was most optimal because of its accuracy in prediction, as well as its speed of implementation. The performance metrics of the GRU model (MSE 15.15, RMSE 3.89, R^2 score 0.9927) shows that the GRU can make a good prediction, and translates to a fast-response action for investment.

While the LSTM model performed quite well, it has greater complexity, and longer training time considerations which could pose challenges in high frequency trading applications where fast decisions needed to still be accurate is a premium. The issue of interpretability related to LSTM models that others have raised in the literature suggested that the choice of GRU provided a better balance between performance and real-world utility.

The advantages and disadvantages of GRU and LSTM models provide an important backdrop for model selection in the future. Implementing both models with hybrid approaches, as suggested by (Lu, et al., 2019), could create a more comprehensive system by marrying GRU’s efficiency with LSTM’s detailed data handling. - add one more reference from literature review

5.4 Limitations and Future Work

This research study recognized there are limitations to this work:

- Data Quality: Any usage of Deep Learning models will maintain fewer effective outcomes when stocks do not have long time series data and/or there is a liquidity issue.
- Computable Complexity: Developing a LSTM model involves high computational complexity.
- Interpretability: LSTM networks create challenges to interpretability.
- Adaptability: LSTM model have limited ability to adapt to sudden and unknown events which are not part of the time series history.

5.5 Conclusion

In conclusion, the EDA findings and model outcomes suggest a compelling synergy between insightful data exploration and predictive analytics in the stock market. GRU's performance relative to LSTM and CNN highlights crucial advancements in financial forecasting, particularly as financial markets continue to evolve in complexity. Future studies should leverage these insights by exploring hybrid models and incorporating additional data dimensions—such as sentiment analysis and macroeconomic indicators—to enhance forecasting precision. As the landscape of financial data science develops, maintaining agility and clarity through model selection will empower investors and analysts to navigate the unpredictable realms of market behaviour effectively.

REFERENCES

Ahmed, S. F. et al., 2023. Deep learning modelling techniques: current progress, applications, advantages, and challenges. *Artifcial Intelligence Review.*

Al-Selwi, S. M. et al., 2024. RNN-LSTM: From applications to modeling techniques and beyond—Systematic review. *Journal of King Saud University - Computer and Information Sciences,* June.36(5).

Botunac, I., Bosna, J. & Matetić, M., 2024. Optimization of Traditional Stock Market Strategies Using the LSTM Hybrid Approach. *Information,* 15(3), p. 136;https://doi.org/10.3390/info15030136.

Chicco, D., Warrens, M. J. & Jurman, G., 2021. The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. *Peer J Computer Science.*

Chitty-Venkata, K. T. et al., 2023. A survey of techniques for optimizing transformer inference. *Journal of Systems Architecture,* Volume 144.

Dhokane, R. M. & Agarwal, S., 2023. Stock market prediction using the LSTM algorithm in association with the Relative Strength Index (RSI) and Exponential Moving Average (EMA) indicators.. *Research Square,* September.

Dutta, A., Panda, R. R. & Nagwani, N., 2020. A Hybrid Deep Learning Approach for Stock Price Prediction. *Machine Learning for Predictive Analysis,* October.

Freeborough, W. & van Zyl, T., n.d. Investigating explainability methods in recurrent neural network architectures for financial time series data. Applied Sciences. *Applied Sciences,* 12(3), p. 1427.

Goodfellow, I., Bengio, Y. & Courville, A., 2016. *Deep Learning.* s.l.:MIT Press.

Houdt, G. V., Mosquera, C. & Nápoles, G., 2020. A Review on the Long Short-Term Memory Model. *Artificial Intelligence Review,* December.

Huimin, et al., 2024. Time series forecasting model for non-stationary series pattern extraction using deep learning and GARCH modeling. *Journal of Cloud Computing: Advances, Systems and Applications,* 13(2).

Kaladevi, P. & Thyagarajah, K., 2019. Integrated CNN- and LSTM-DNN-based sentiment analysis over big social data for opinion mining. *Behaviour & Information Technology,* December.

Kumar, Y., Koul, A., Kaur, S. & Hu, Y.-C., 2023. Machine Learning and Deep Learning Based Time Series Prediction and Forecasting of Ten Nations' COVID-19 Pandemic. *SN Computer Science,* Volume 4.

Lindemann, B. et al., 2021. A survey on long short-term memory networks for time series prediction. *14th CIRP Conference on Intelligent Computation in Manufacturing Engineering, CIRP ICME '20 .*

Lu, J., Zhan, Q., Yang, Z. & Tu, M., 2019. A hybrid model based on convolutional neural network and long short-term memory for short-term load forecasting. *2019 IEEE Power & Energy Society General Meeting (PESGM),* August.

Malashin, I. et al., 2024. Applications of Long Short-Term Memory (LSTM) Networks in Polymeric Sciences: A Review. *Polymers,* 16(18), p. 2607;https://doi.org/10.3390/polym16182607.

Malashin, I. et al., 2024. Applications of Long Short-Term Memory (LSTM) Networks in Polymeric Sciences: A Review. *Polymers,* 16(18), p. 2607;https://doi.org/10.3390/polym16182607.

Mienye, I. D., Swart, T. G. & Obaido, G., 2024. Recurrent Neural Networks: A Comprehensive Review of Architectures, Variants, and Applications. *Information,* 15(9), p. 517;https://doi.org/10.3390/info15090517.

Moghar, A. & Hamiche, M., 2020. Stock Market Prediction Using LSTM Recurrent Neural Network. *Procedia Computer Science,* January.pp. 1168-1173;10.1016/j.procs.2020.03.049.

Oko-Odion, C., 2025. An Integration of Time Series Analysis into Quantitative Risk Modelling Frameworks for Enhanced Risk Management. *International Journal of Research Publication and Reviews,* January, 6(1), pp. 5085-5099;10.55248/gengpi.6.0125.0649.

Rahman, M. O., Hossain, M. S., Junaid, T.-S. & Forhad, M. S. A., 2019. Predicting Prices of Stock Market using Gated Recurrent Units (GRUs) Neural Networks. *International Journal of Computer Science and Network Security,* January.19(1).

Raza, F., 2023. Machine Learning for Financial Forecasting. *Cosmic Bulletin of Business Management,* 2(1).

Rivas, F., Sierra-Garcia, J. E. & Camara, J. M., 2025. Comparison of LSTM- and GRU-Type RNN Networks for Attention and Meditation Prediction on Raw EEG Data from Low-Cost Headsets. *Electronics ,* 14(4), p. 707;https://doi.org/10.3390/electronics14040707.

Rivas, F., Sierra-Garcia, J. E. & Camara, J. M., 2025. Comparison of LSTM- and GRU-Type RNN Networks for Attention and Meditation Prediction on Raw EEG Data from Low-Cost Headsets. *Electronics,* 14(4), p. 707;https://doi.org/10.3390/electronics14040707.

Shen, G. et al., 2018. Deep Learning with Gated Recurrent Unit Networks for Financial Sequence Predictions. *Procedia Computer Science,* Volume 131, pp. 895-903; https://doi.org/10.1016/j.procs.2018.04.298.

Singh, P., Jha, M., Sharaf, M. a. & El-Meligy, M. A., 2023. Harnessing a Hybrid CNN-LSTM Model for Portfolio Performance: A Case Study on Stock Selection and Optimization. *IEEE Access,* January.p. 99.

Wibawa, A. P. et al., 2022. Time-series analysis with smoothed Convolutional Neural Network. *Journal of Big Data,* Volume 44.

APPENDIX

Code

```
"""Importing Libraries"""
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Plotly for interactive charts
import plotly.express as px
import plotly.graph_objs as go

import tensorflow as tf
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, GRU, Dense, Dropout
from tensorflow.keras.optimizers import Adam

# Download dataset using gdown

# Install gdown if needed
try:
    import gdown
except ImportError:
    import subprocess
```

```
    subprocess.check_call(["pip", "install", "--upgrade", "gdown"])
    import gdown

file_id = " 16roDpQxTS2s-X7Lxg8wwEaoZ89k16I6V "
output_path = os.path.join("data", "all_stocks_5yr.csv")
os.makedirs("data", exist_ok=True)

if not os.path.exists(output_path):
    print("⬇ Downloading dataset...")
    gdown.download(id=file_id, output=output_path, quiet=False)
else:
    print("✅ Dataset already exists.")

"""Data Cleaning process"""

data = pd.read_csv(output_path)
data['date'] = pd.to_datetime(data['date'])
data.sort_values(['Name', 'date'], inplace=True)
data.reset_index(drop=True, inplace=True)

print(data.shape)
print(data.columns)
print(data['Name'].nunique())  # Number of unique stocks
data.isnull().sum()

df = pd.DataFrame(data)

print("Original DataFrame:")
```

```
print(df)

# Fill missing values with the mean of specific columns: 'open', 'high', and 'low'
columns_to_fill = ['open', 'high', 'low']
df[columns_to_fill] = df[columns_to_fill].fillna(df[columns_to_fill].mean())

print("\nDataFrame after filling missing values with the average in specified columns:")
print(df)

df.isnull().sum()

"""Exploratory Data Analysis"""

# Check Summary Statistics
df.describe()

# Plot closing prices of major stocks
companies = ['AAPL', 'GOOGL', 'AMZN', 'MSFT']

plt.figure(figsize=(16, 8))
for company in companies:
    stock = df[df['Name'] == company]
    plt.plot(stock['date'], stock['close'], label=company)

plt.legend()
plt.title('Stock Closing Prices Over Time')
plt.xlabel('Date')
plt.ylabel('Closing Price')
```

```python
plt.grid()
plt.show()

# Distribution of Daily Returns for a Stock
stock = df[df['Name'] == 'AAPL'].copy()
stock['daily_return'] = stock['close'].pct_change()

plt.figure(figsize=(10, 5))
sns.histplot(stock['daily_return'].dropna(), bins=100, kde=True)
plt.title('Distribution of Daily Returns - AAPL')
plt.xlabel('Daily Return')
plt.show()

# Volatility Comparison (Rolling Standard Deviation)
plt.figure(figsize=(14, 6))
for company in ['AAPL', 'GOOG', 'AMZN']:
    subset = df[df['Name'] == company].copy()
    subset.set_index('date', inplace=True)
    subset['rolling_vol'] = subset['close'].pct_change().rolling(window=30).std()
    plt.plot(subset['rolling_vol'], label=company)

plt.title('30-Day Rolling Volatility')
plt.ylabel('Volatility')
plt.xlabel('Date')
plt.legend()
plt.grid()
plt.show()
```

```
# Moving Averages
apple = df[df['Name'] == 'AAPL'].copy()
apple.set_index('date', inplace=True)
apple['MA50'] = apple['close'].rolling(window=50).mean()
apple['MA200'] = apple['close'].rolling(window=200).mean()

plt.figure(figsize=(14, 6))
plt.plot(apple['close'], label='Close Price')
plt.plot(apple['MA50'], label='50-Day MA')
plt.plot(apple['MA200'], label='200-Day MA')
plt.title('AAPL - Moving Averages')
plt.legend()
plt.grid()
plt.show()

# Correlation Matrix of Closing Prices (Top 20 Stocks)
top20 = df['Name'].value_counts().head(20).index.tolist()
filtered_df = df[df['Name'].isin(top20)]
pivot = filtered_df.pivot(index='date', columns='Name', values='close')
correlation_matrix = pivot.corr()

plt.figure(figsize=(14, 12))
sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm', center=0)
plt.title("Correlation Matrix of Top 20 Stocks")
plt.show()

# Stock Price Distribution Using KDE (Kernel Density Estimation)
plt.figure(figsize=(12, 6))
```

```
for stock in ['AAPL', 'GOOGL', 'AMZN', 'MSFT']:
    sns.kdeplot(df[df['Name'] == stock]['close'], label=stock, fill=True)

plt.title("Stock Price Distributions")
plt.xlabel("Closing Price")
plt.legend()
plt.show()

# Rolling Mean (Moving Average) Analysis
apple = df[df['Name'] == 'AAPL'].copy()
apple['MA50'] = apple['close'].rolling(window=50).mean()
apple['MA200'] = apple['close'].rolling(window=200).mean()

plt.figure(figsize=(14, 7))
plt.plot(apple['date'], apple['close'], label="Close Price", alpha=0.7)
plt.plot(apple['date'], apple['MA50'], label="50-Day MA", linestyle='dashed')
plt.plot(apple['date'], apple['MA200'], label="200-Day MA", linestyle='dashed')

plt.legend()
plt.title("AAPL - Moving Averages (50-day & 200-day)")
plt.show()

# Boxplot for Stock Price Distribution
plt.figure(figsize=(12, 6))
sns.boxplot(x='Name', y='close', data=df[df['Name'].isin(['AAPL', 'GOOGL', 'AMZN', 'MSFT'])])
plt.xticks(rotation=45)
plt.title('Stock Price Distribution (Boxplot)')
plt.show()
```

```
# Pairplot for Selected Stocks
selected_stocks = df[df['Name'].isin(['AAPL', 'GOOGL', 'AMZN', 'MSFT'])]
pivoted = selected_stocks.pivot(index='date', columns='Name', values='close')
sns.pairplot(pivoted)
plt.show()

# Candlestick Chart for AAPL
fig = go.Figure(data=[go.Candlestick(x=apple['date'],
            open=apple['open'],
            high=apple['high'],
            low=apple['low'],
            close=apple['close'])])
fig.update_layout(title='AAPL Candlestick Chart', xaxis_rangeslider_visible=False)
fig.show()

# Bollinger Bands for AAPL
apple['SMA20'] = apple['close'].rolling(window=20).mean()
apple['UpperBand'] = apple['SMA20'] + 2 * apple['close'].rolling(window=20).std()
apple['LowerBand'] = apple['SMA20'] - 2 * apple['close'].rolling(window=20).std()

plt.figure(figsize=(14, 7))
plt.plot(apple['date'], apple['close'], label='Close Price', alpha=0.7)
plt.plot(apple['date'], apple['SMA20'], label='20-Day SMA', linestyle='dashed')
plt.fill_between(apple['date'], apple['UpperBand'], apple['LowerBand'], color='gray', alpha=0.3)

plt.legend()
plt.title('AAPL - Bollinger Bands')
```

```
plt.show()

# Daily Return Percentage for Selected Stocks
fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(15, 10))

company_list = ['AAPL', 'GOOGL', 'MSFT', 'AMZN']
for i, stock in enumerate(company_list):
    df_stock = df[df['Name'] == stock].copy()
    df_stock['Daily Return'] = df_stock['close'].pct_change()

    row, col = divmod(i, 2)
    df_stock['Daily Return'].plot(ax=axes[row, col], legend=True, linestyle='--', marker='o')
    axes[row, col].set_title(stock)

fig.tight_layout()
plt.show()

# Increase figure size
plt.figure(figsize=(15, 8))
# Convert columns to numeric, forcing errors to NaN
df_numeric = df.apply(pd.to_numeric, errors='coerce')

# Compute correlation, filling NaN values with 0
correlation_matrix = df_numeric.corr().fillna(0)

# Create heatmap with optimized settings
sns.heatmap(correlation_matrix, annot=False, cmap="coolwarm", linewidths=0.5)
```

```
plt.title("Stock Closing Price Correlation")
plt.xticks(rotation=90)  # Rotate x-axis labels for readability
plt.yticks(rotation=0)
plt.show()

plt.figure(figsize=(12, 6))

for stock in company_list:
    daily_returns = df[df['Name'] == stock]['close'].pct_change().dropna()
    plt.hist(daily_returns, bins=50, alpha=0.7, label=stock)

plt.legend()
plt.xlabel('Daily Return')
plt.ylabel('Frequency')
plt.title("Histogram of Daily Returns")
plt.show()

plt.figure(figsize=(14, 6))

for stock in company_list:
    subset = df[df['Name'] == stock]
    plt.plot(subset['date'], subset['high'], label=f'{stock} High', alpha=0.6)
    plt.plot(subset['date'], subset['low'], label=f'{stock} Low', alpha=0.6)

plt.legend()
plt.title('High & Low Prices Over Time')
plt.xlabel('Date')
plt.ylabel('Price')
```

```
plt.show()

df_filtered = df[(df['date'] >= '2017-01-01') & (df['date'] <= '2018-12-31')]

plt.figure(figsize=(12, 6))
for stock in company_list:
    subset = df_filtered[df_filtered['Name'] == stock]
    plt.plot(subset['date'], subset['close'], label=f'{stock} Close')

plt.legend()
plt.title("Stock's Prices from 2017 to 2018")
plt.xlabel("Date")
plt.ylabel("Stock Price")
plt.xticks(rotation=45)
plt.show()

df_jan2018 = df[(df['date'] >= '2018-01-01') & (df['date'] <= '2018-01-31')]

plt.figure(figsize=(12, 6))
for stock in company_list:
    subset = df_jan2018[df_jan2018['Name'] == stock]
    plt.plot(subset['date'], subset['close'], label=f'{stock} Close', linestyle='solid')
    plt.plot(subset['date'], subset['high'], label=f'{stock} High', linestyle='dashed')
    plt.plot(subset['date'], subset['low'], label=f'{stock} Low', linestyle='dotted')

plt.legend()
plt.title("Stock Prices in January 2018")
plt.xlabel("Date")
```

```
plt.ylabel("Price")

plt.xticks(rotation=45)

plt.show()


df_volume = df.groupby('Name')['volume'].sum().nlargest(10)


plt.figure(figsize=(8, 8))

plt.pie(df_volume, labels=df_volume.index, autopct='%1.1f%%',
colors=sns.color_palette('pastel'))

plt.title('Total Trading Volume by Company')

plt.show()


"""Data Pre-Processing and Techniques"""


# Feature Engineering


# Moving Averages

df['MA_7'] = df['close'].rolling(window=7).mean()

df['MA_30'] = df['close'].rolling(window=30).mean()


# Volatility (Standard Deviation)

df['Volatility'] = df['close'].rolling(window=7).std()


# Relative Strength Index (RSI)

def compute_rsi(data, window=14):

    delta = data.diff(1)

    gain = (delta.where(delta > 0, 0)).rolling(window=window).mean()

    loss = (-delta.where(delta < 0, 0)).rolling(window=window).mean()
```

```
    rs = gain / loss
    rsi = 100 - (100 / (1 + rs))
    return rsi

df['RSI'] = compute_rsi(df['close'])

# Drop NaN values from feature engineering
df.dropna(inplace=True)

df.head()

# Normalize Data

scaler = MinMaxScaler(feature_range=(0, 1))
df_scaled = scaler.fit_transform(df[['close', 'MA_7', 'MA_30', 'Volatility', 'RSI']])

# formation of Time-Series Sequences

def create_sequences(data, time_steps=50):
    X, y = [], []
    for i in range(len(data) - time_steps):
        X.append(data[i:i + time_steps])
        y.append(data[i + time_steps, 0])  # Predicting 'close' price
    return np.array(X), np.array(y)

time_steps = 50
X, y = create_sequences(df_scaled, time_steps)
```

```
# Train-Test Split
split_ratio = 0.8
split_index = int(len(X) * split_ratio)

X_train, X_test = X[:split_index], X[split_index:]
y_train, y_test = y[:split_index], y[split_index:]

print(f"Training Data Shape: {X_train.shape}, {y_train.shape}")
print(f"Testing Data Shape: {X_test.shape}, {y_test.shape}")

# Build & Train LSTM model

# Define LSTM Model
model_lstm = Sequential([
    LSTM(50, return_sequences=True, input_shape=(X_train.shape[1], X_train.shape[2])),
    Dropout(0.2),
    LSTM(50, return_sequences=False),
    Dropout(0.2),
    Dense(25),
    Dense(1)
])

# Compile the model
model_lstm.compile(optimizer=Adam(learning_rate=0.001), loss='mse')

# Train the model
history = model_lstm.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test), verbose=1)
```

```
# Define GRU Model
model_gru = Sequential([
    GRU(50, return_sequences=True, input_shape=(X_train.shape[1], X_train.shape[2])),
    Dropout(0.2),
    GRU(50, return_sequences=False),
    Dropout(0.2),
    Dense(25),
    Dense(1)
])

# Compile the model
model_gru.compile(optimizer=Adam(learning_rate=0.001), loss='mse')

# Train the model
history_gru = model_gru.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test), verbose=1)

from keras.models import Sequential
from keras.layers import Conv1D, Flatten, Dense, Dropout

# Define CNN model
model_cnn = Sequential([
    Conv1D(filters=64, kernel_size=3, activation='relu', input_shape=(X_train.shape[1], X_train.shape[2])),
    Dropout(0.2),
    Flatten(),
    Dense(50, activation='relu'),
```

```
    Dropout(0.2),
    Dense(1)
])

# Compile CNN model
model_cnn.compile(optimizer=Adam(learning_rate=0.001), loss='mse')

# Train CNN model
history_cnn = model_cnn.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test), verbose=1)

# STEP 1: Predictions
y_pred_lstm = model_lstm.predict(X_test)
y_pred_gru = model_gru.predict(X_test)
y_pred_cnn = model_cnn.predict(X_test)

# STEP 2: Inverse transform predictions and actual
pad_width = df_scaled.shape[1] - 1  # Number of additional features
# Pad zeros for inverse transform
y_pred_lstm_full = scaler.inverse_transform(np.hstack((y_pred_lstm, np.zeros((len(y_pred_lstm), pad_width)))))
y_pred_gru_full = scaler.inverse_transform(np.hstack((y_pred_gru, np.zeros((len(y_pred_gru), pad_width)))))
y_pred_cnn_full = scaler.inverse_transform(np.hstack((y_pred_cnn, np.zeros((len(y_pred_cnn), pad_width)))))
y_test_full = scaler.inverse_transform(np.hstack((y_test.reshape(-1, 1), np.zeros((len(y_test), pad_width)))))

# Extract actual and predicted closing prices
```

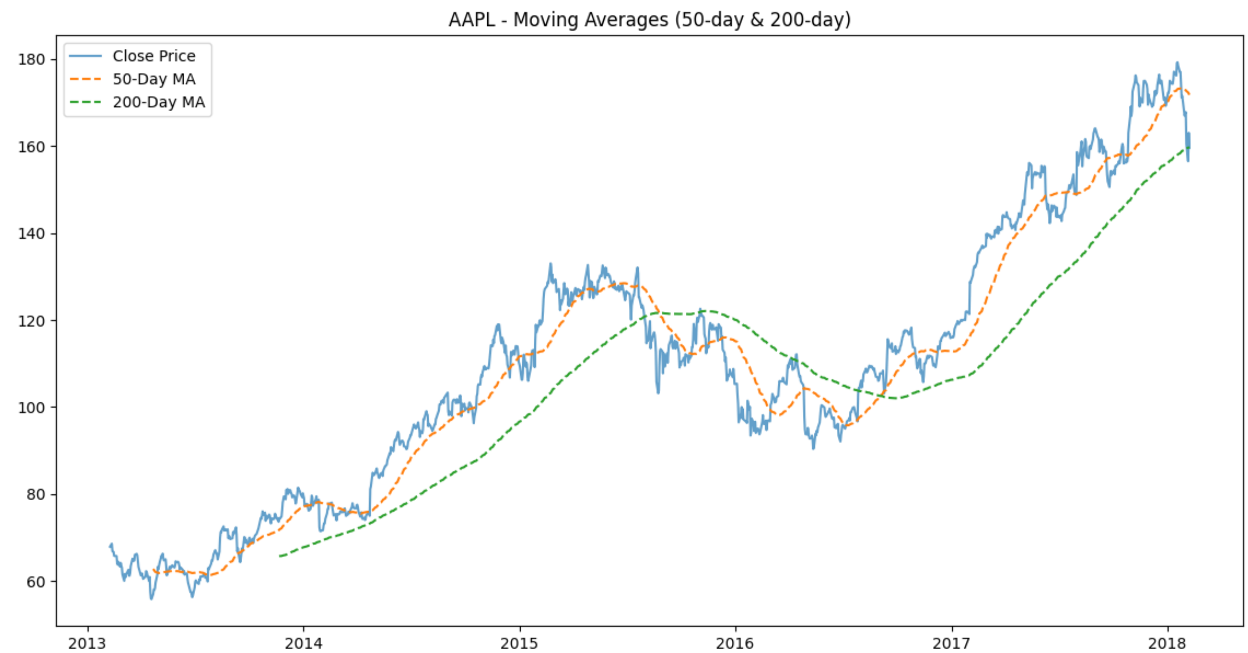
```
y_pred_lstm = y_pred_lstm_full[:, 0]

y_pred_gru = y_pred_gru_full[:, 0]

y_pred_cnn = y_pred_cnn_full[:, 0]

y_test_inv = y_test_full[:, 0]


# Pad y_test with dummy zeros to match the original number of features (5)

y_test_padded = np.hstack((y_test.reshape(-1, 1), np.zeros((len(y_test), df_scaled.shape[1] -
1))))

y_test_actual = scaler.inverse_transform(y_test_padded)[:, 0]  # Extract only the 'close' column


# Plot the predictions against actual prices

plt.figure(figsize=(12, 6))

plt.plot(df.index[-len(y_test):], y_test_actual, label='Actual Price', color='black')

plt.plot(df.index[-len(y_test):], y_pred_lstm, label='LSTM Predicted Price', color='blue')

plt.plot(df.index[-len(y_test):], y_pred_gru, label='GRU Predicted Price', color='red')

plt.plot(df.index[-len(y_test):], y_pred_cnn, label='CNN Predicted Price', color='green')

plt.xlabel('Date')

plt.ylabel('Stock Price')

plt.title('S&P 500 Stock Price Prediction - LSTM vs GRU vs CNN')

plt.legend()

plt.show()


# Create DataFrame for summary

results_df = pd.DataFrame({

   'Date': df.index[-len(y_test):],

   'Actual Price': y_test_actual,

   'LSTM Predicted': y_pred_lstm,

   'GRU Predicted': y_pred_gru,
```

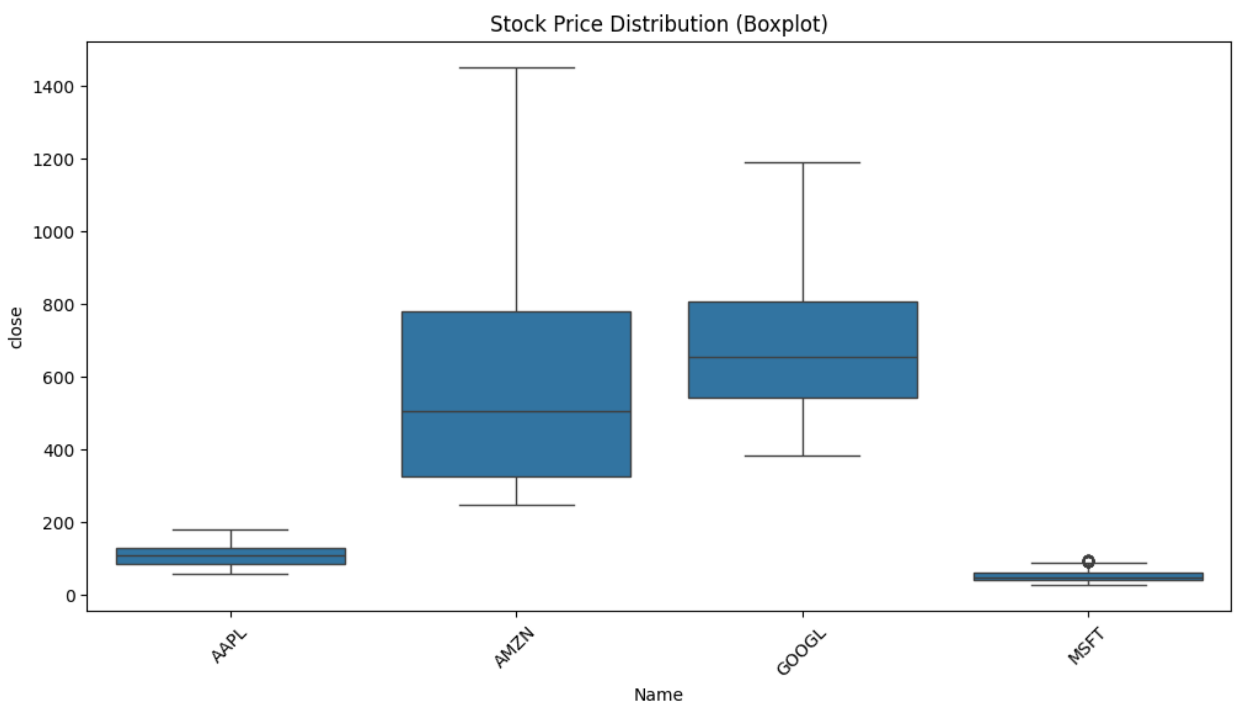
```
    'CNN Predicted': y_pred_cnn
})

# Show top 10 rows
results_df.head(10)

import numpy as np
import pandas as pd
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import matplotlib.pyplot as plt

# STEP 3: Metrics Calculation
mse_lstm = mean_squared_error(y_test_inv, y_pred_lstm)
rmse_lstm = np.sqrt(mse_lstm)
mae_lstm = mean_absolute_error(y_test_inv, y_pred_lstm)
r2_lstm = r2_score(y_test_inv, y_pred_lstm)

mse_gru = mean_squared_error(y_test_inv, y_pred_gru)
rmse_gru = np.sqrt(mse_gru)
mae_gru = mean_absolute_error(y_test_inv, y_pred_gru)
r2_gru = r2_score(y_test_inv, y_pred_gru)

mse_cnn = mean_squared_error(y_test_inv, y_pred_cnn)
rmse_cnn = np.sqrt(mse_cnn)
mae_cnn = mean_absolute_error(y_test_inv, y_pred_cnn)
r2_cnn = r2_score(y_test_inv, y_pred_cnn)

# STEP 4: Summary DataFrame
```

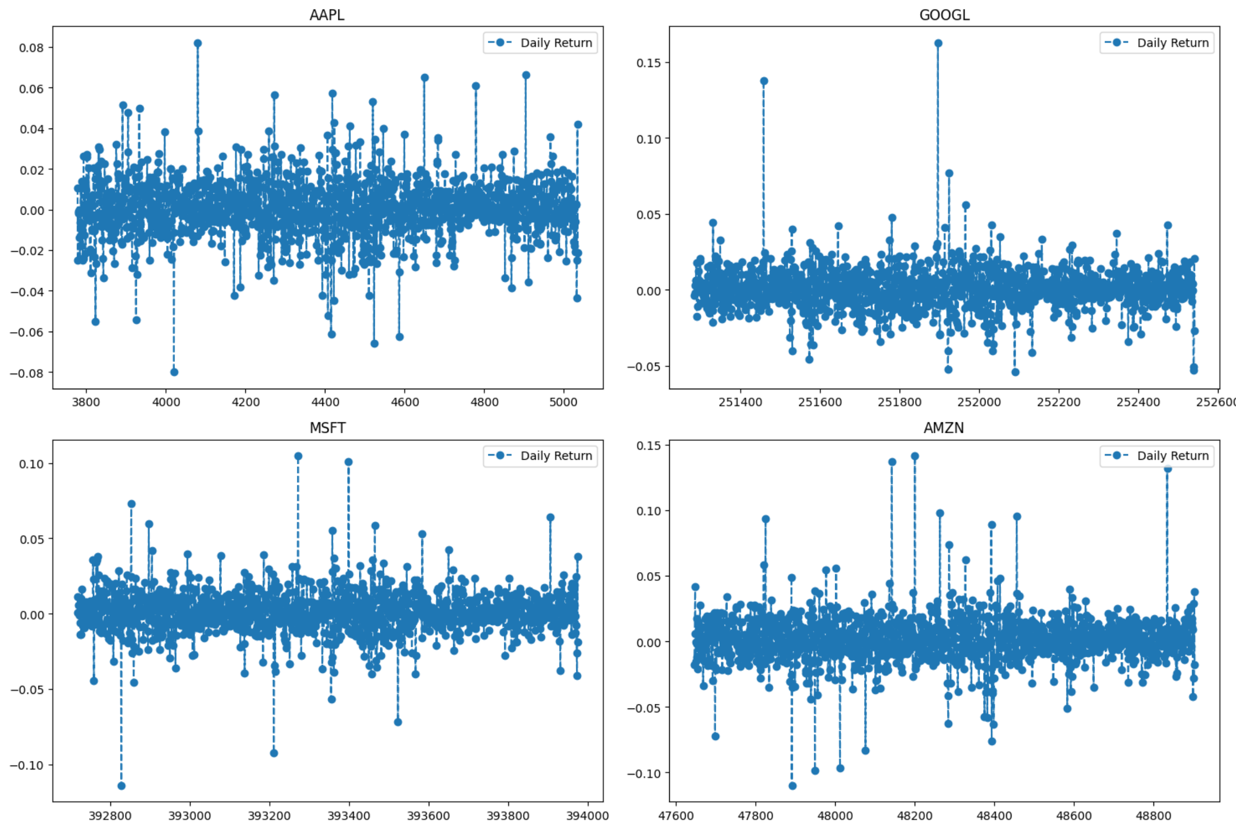
```
results_df = pd.DataFrame({
    'Model': ['LSTM', 'GRU', 'CNN'],
    'MSE': [mse_lstm, mse_gru, mse_cnn],
    'RMSE': [rmse_lstm, rmse_gru, rmse_cnn],
    'MAE': [mae_lstm, mae_gru, mae_cnn],
    'R2 Score': [r2_lstm, r2_gru, r2_cnn]
})

print("\n📊 Model Evaluation Summary:")
print(results_df.round(4))

# Create a DataFrame for plotting
plot_df = pd.DataFrame({
    'Actual': y_test_inv,
    'Predictions LSTM': y_pred_lstm,
    'Predictions GRU': y_pred_gru,
    'Predictions CNN': y_pred_cnn
})

# Optionally zoom into the last N points
zoom_range = 100
plot_df_zoomed = plot_df[-zoom_range:]

# Plot
plt.figure(figsize=(16, 6))
plt.title('📉 Zoomed-In Model Predictions vs Actual (Last {} Days)'.format(zoom_range), fontsize=16)
plt.plot(plot_df_zoomed['Actual'], label='Actual Price', color='black', linewidth=2)
```

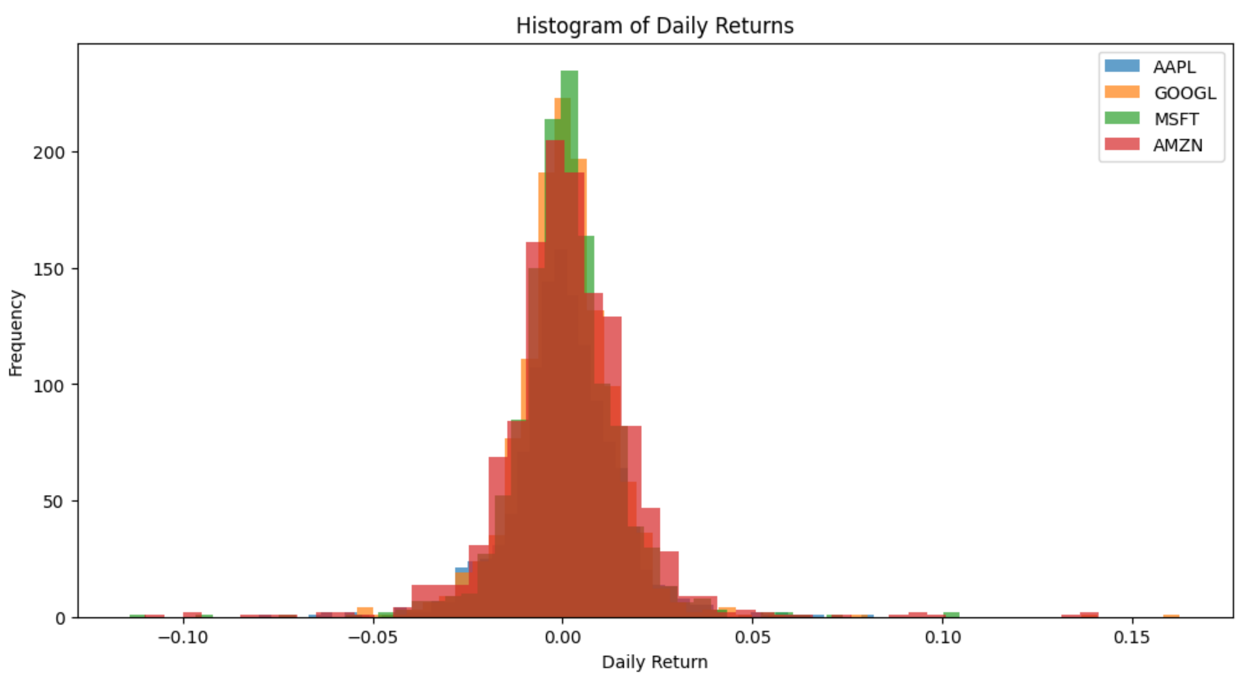
```python
plt.plot(plot_df_zoomed['Predictions LSTM'], label='LSTM Prediction', linestyle='--', marker='o', markersize=4, color='blue')

plt.plot(plot_df_zoomed['Predictions GRU'], label='GRU Prediction', linestyle='--', marker='s', markersize=4, color='red')

plt.plot(plot_df_zoomed['Predictions CNN'], label='CNN Prediction', linestyle='--', marker='^', markersize=4, color='green')


plt.xlabel('Time Step', fontsize=14)

plt.ylabel('Close Price USD ($)', fontsize=14)

plt.legend(loc='lower right')

plt.grid(True)

plt.tight_layout()

plt.show()
```

Additional EDA plots


AAPL - Moving Averages (50-day & 200-day)
Close Price
50-Day MA
200-Day MA
180
160
140
120
100
80
60
2013
2014
2015
2016
2017
2018


Stock Price Distribution (Boxplot)
1400
1200
1000
800
600
400
200
0
close
AAPL
AMZN
GOOGL
MSFT
Name


AAPL
GOOGL
MSFT
AMZN
Daily Return
0.08
0.06
0.04
0.02
0.00
−0.02
−0.04
−0.06
−0.08
3800
4000
4200
4400
4600
4800
5000
0.15
0.10
0.05
−0.05
251400
251600
251800
252000
252200
252400
252600
−0.10
392800
393000
393200
393400
393600
393800
394000
47600
47800
48000
48200
48400
48600
48800


Histogram of Daily Returns
AAPL
GOOGL
MSFT
AMZN
Frequency
200
150
100
50
0
−0.10
−0.05
0.00
0.05
0.10
0.15
Daily Return