NOVA IMS Information Management School

MDSAA

Master Degree Program in

Data Science and Advanced Analytics

Evaluation of LSTM Networks for Stock Price Forecasting

A Benchmark Against traditional econometric models (ARIMA and PROPHET)

Stepan Kuznetsov

Master Thesis

presented as partial requirement for obtaining a Master's Degree in Data Science and Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School

Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

Evaluation of LSTM Networks for Stock Price Forecasting

A Benchmark Against traditional econometric models (ARIMA and PROPHET)

by

Stepan Kuznetsov

Master Thesis presented as partial requirement for obtaining the Master's degree in Data Science and Advanced Analytics, with a specialization in Business Analytics.

Supervised by

Mauro Castelli, PhD, NOVA Information Management School

November, 2025

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledged the Rules of Conduct and Code of Honor from the NOVA Information Management School.

[Lisbon, November 2025]

Stepan Kuznetsov

DEDICATION

First of all, I would like to thank my family. Your love, support and confidence in me have been the foundation of everything I have managed to achieve. Thank you for your patience and for staying by my side, especially when I doubted myself.

I am also very grateful to my supervisor, Professor Mauro Castelli, for his guidance, patience and professionalism throughout this work.

I would also love to acknowledge myself for not giving up when things became overwhelming. Working on this thesis has shown me the value of persistence and of keeping going even when progress feels slow.

To all of you, a very big and sincere thank you.

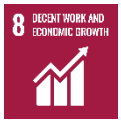
ABSTRACT

This thesis explores the effectiveness of the use of Long Short-Term Memory (LSTM) neural networks for prediction of daily stock closing prices and compares their performance with several traditional time series models, such as AutoRegressive Integrated Moving Average (ARIMA), its extended versions and Facebook PROPHET. The analysis is based on historical stock data from a selected group of companies listed in the S&P 500 index, collected from publicly available Yahoo Finance databases. All models are trained using a consistent preprocessing and forecasting pipeline with the objective of predicting the next day closing price. Specially selected accuracy metrics are used to evaluate the size of prediction errors and how well the models spot trends and movements. The results indicate that while traditional models can model linear relationships and seasonal changes, they tend to underperform with the messy behavior and dynamics seen in financial markets. In contrast, the LSTM model can pick up on complicated time patterns straight from the data, without need to make assumptions like stationarity. This study highlights the growing relevance of machine learning in financial areas, especially forecasting. It offers a comparative analysis that reflects the strengths and limitations of traditional, hybrid and modern approaches. The results received may be useful for researchers and practitioners, who can leverage the presented findings and model designs in their own time series forecasting applications, including closing stock price prediction.

KEYWORDS

Stock Price Forecasting; Time Series Prediction; LSTM; ARIMA; PROPHET; Machine Learning; S&P 500.

Sustainable Development Goals (SDG):



INDEX

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ARIMA	AutoRegressive Integrated Moving Average
SARIMA	Seasonal ARIMA
SARIMAX	SARIMA with exogenouos variables
GARCH	Generalized Autoregressive Conditional Heteroskadasticity
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
MAPE	Mean Absolute Percentage Error
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
ROC	Rate of Change
RSI	Relative Strength Index
SMA	Simple Moving Average
EMH	Efficient Market Hypothesis
AMH	Adaptive Market Hypothesis

1. INTRODUCTION

In today's fast-paced and data-saturated financial environment the ability to make well-informed, data-driven decisions has become essential for maintaining a competitive advantage. Financial markets are characterized by high volatility, driven by a wide range of determinants, such as macroeconomic indicators, geopolitical events and investor sentiment that shift rapidly. The forecasting models have been critical tools for market participants to deal with this uncertainty. Accurate stock price prediction supports portfolio optimization, risk management and algorithmic trading. This helps institutional investors and individual traders to better anticipate and predict market movements. With technological advancements, modern forecasting techniques have more frequently made use of the advantages of artificial intelligence (AI) and machine learning (ML). These techniques provide practically meaningful findings and optimized pricing tactics rather than earlier dependence on gut feeling or traditional heuristics.

The shift toward data-driven forecasting has brought several ML methods to the forefront in the analysis of financial time series (Deloitte, 2023). As the volumes of historical and real time financial data have grown rapidly, new methods have offered ways to uncover patterns that are difficult to detect using traditional models. One of the most effective approaches for analyzing time series data is the Long Short-Term Memory (LSTM) network. LSTM is a specialized type of recurrent neural network (RNN) designed to capture long-range dependencies in sequential data. Originally developed by Hochreiter and Schmidhuber (1997), the LSTM was created to overcome the limitations of conventional RNNs, such as the vanishing gradient problem, by introducing a memory cell structure capable of retaining important information over extended periods. In the context of financial forecasting, this feature is especially valuable as market behavior is often influenced by events or trends that unfold gradually over time. In contrast to classical statistical models like ARIMA, which rely on assumptions of linearity and stationarity, LSTM networks can learn directly from raw, nonlinear financial data without the need for significant preprocessing. This ability to adapt to complex, time-varying dynamics makes LSTM highly effective for modeling stock prices, which are frequently noisy, non-stationary and affected by hidden variables. Empirical studies have shown that LSTM models can outperform classical methods in various forecasting tasks involving equities, indices and even cryptocurrencies (Fischer & Krauss, 2018; Nelson et al.,

2017). Reflecting this growing trend, the market for AI in fintech is projected to exceed $64 billion by 2030 with applications ranging from investment strategy development to algorithmic trading (Statista, 2024).

While machine learning approaches, particularly LSTM, are increasingly adopted in financial forecasting, there are still traditional time series models like ARIMA and hybrid frameworks like PROPHET that continue to serve as a baseline in academic and industry contexts. These models remain in use due to their simplicity, interpretability and effectiveness in modeling linear patterns and seasonality in time series data. ARIMA, developed within the Box-Jenkins framework, models linear dependencies in stationary time series, while PROPHET, often referred to as a hybrid model extends this approach by combining trend, seasonality and holiday effects within a flexible additive framework (Taylor & Letham, 2018).

However, these assumptions, mainly those related to linearity and stationarity, limit their performance in environments characterized by complex, nonlinear dynamics, which frequently occur in financial markets (Makridakis, et al, 2018). These limitations justify the need to benchmark traditional methods against more flexible and adaptive machine learning models such as LSTM, particularly in complex and dynamic forecasting tasks like stock price prediction, which we consider as the main theme in this work.

The current study compares forecasting models used with daily stock price data to address the limitations of traditional methods in capturing nonlinear patterns, adapting to volatile market dynamics and generalizing across multiple assets. Specifically, it analyzes how effectively a LSTM neural network performs in comparison to two widely used econometric models: ARIMA with its improved family of frameworks and PROPHET across several stocks. The research is conducted on a selection of stocks from S&P 500 index using historical data sourced from Yahoo Finance. All models are trained to forecast the closing values of the subsequent day. By adopting a multi-asset approach, the study aims to evaluate each model's generalizability and resilience across varying financial scenarios. The evaluation is based on standard error metrics such as Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and R-squared measure.

This study aims at use and comparison of well-known models inside a single experimental framework rather than introduction of a novel forecasting technique. Building on the research gap identified earlier, the objective is to gain a deeper understanding of each model's

limitations and practical behavior when applied to erratic, real-world financial data. The study is relevant in environments where financial strategy and risk management depend heavily on both predictive accuracy and model transparency as well as reasonably limited computing power for calculations.

The thesis implements a consistent modeling pipeline across all selected assets from the perspective of methodology. Each stock series goes through similar preprocessing steps handling missing values, scaling and splitting into training and test sets. While ARIMA and PROPHET are fitted to stationary or seasonally adjusted versions of the data as needed, the LSTM network is trained using a sequence-based supervised learning approach. To account for temporal variance and changing market conditions, performance is evaluated using multiple rolling prediction windows and the results are then analyzed to identify which models perform best under different scenarios. The work also contributes to the field of stock price forecasting by presenting a structured, comparative evaluation of deep learning and traditional time series models using real-world financial data. By assessing model performance across multiple domains, it highlights the importance of applicability across different market contexts, an aspect often overlooked in single-stock studies. Although traditional models may retain advantages in low-data or highly stable environments, LSTM is expected to perform more effectively under volatile and nonlinear conditions due to its ability to capture complex temporal dependencies. The insights gained from this study may support data scientists, financial analysts and researchers in selecting forecasting tools that best match the characteristics of their specific use cases, as well as in refining their models through parameter choices and structural adjustments.

The following chapter of this master thesis presents the “Literature Review”, which investigates the theoretical foundations of time series forecasting, discusses the development of traditional econometric models and deep learning approaches and reviews key empirical studies in the context of stock price prediction. The “Methodology” chapter outlines the forecasting models implemented namely ARIMA and its improved models, PROPHET, and LSTM describing their underlying principles, model configurations and the performance metrics used for evaluation. The research hypotheses point out three statements which are then evaluated versus known recent works in the next day prediction field and then compared with the results of this work. Results chapter analyses the results of the modeling for all

explored models for the chosen selection of top US market tickers and discusses the limitations of considered approaches. Ehe final chapter summarizes achieved results, evaluates them versus hypotheses and offers suggestions for the future directions of research in the field of financial series forecasting.

2. LITERATURE REVIEW

The purpose of this literature review is to analyze the key theoretical and empirical contributions related to stock price forecasting with a particular focus on traditional statistical models, machine learning algorithms and deep learning techniques. In addition to academic and technical studies, this chapter also considers the insights of influential practitioners and investors such as Benjamin Graham, John C. Bogle, Warren Buffet and others, whose foundational work continues to shape investment philosophy and market analysis. This dual perspective helps to frame the evolution of forecasting practices within both academic research and real world financial strategy.

The chapter is divided into six major sections. The first two investigate overall Stock market forecasting and Stock market nature considering Effective Market Hypothesis and perspectives of fundamental and technical analysis. The third section creates the bridge from forecasting accuracy towards the introduction of the theoretical foundations of time series forecasting models applied in finance, including classical approaches such as ARIMA, and its enhancement family of models. Then the hybrid model PROPHET is considered, followed by a discussion of LSTM networks as a representative of deep learning model and concluded with consideration on competitive studies.

The subsequent section reviews recent empirical studies that evaluate the performance of these models in stock price prediction tasks. This analysis aims to identify key research gaps and provide a conceptual basis for the comparative modeling approach used in this thesis.

2.1 STOCK PRICE FORECASTING

Stock price forecasting has been a critical task in both financial practice and academic research for a long time. Equity markets that are characterized by high volatility, non-stationarity and complex non-linear relations make forecasting valuable not only for practical investment decisions but also for the broader goal of understanding and anticipating market dynamics

In the recent decade the world has seen significant advancement in data availability, computing power and methodological diversification, which makes practical choice of the forecasting approach more valuable in the field.

Traditionally, forecasting has relied on two dominant approaches: fundamental analysis and technical analysis. Fundamental analysis deals with intrinsic value based on financial statements or the company analysis, economy indicators and industry conditions. ("Technical Analysis vs. Fundamental Analysis", 2015). Warren Buffet and Benjamin Graham are well known for their support of fundamental analysis approaches (Graham & Dodd, 2008). Recent studies have shown forecast improvements when extending traditional fundamental analysis with alternative signals such as ESG and sustainability factors, which have become an integral part of almost every annual report of publicly listed companies since around 2020.

In contrast, technical analysis is based mostly on the use of multiple statistical instruments and research of the trends and price levels. It assumes that all known information is already reflected in the stock price and market behavior repeats over time. (Murphy, John J., 1999). Technical analysis is often concerned with the names of the most well-known traders like John Bollinger, Richard Dennis, and Paul Tudor Jones and others, who made their names and fortune on the stock market based on the Effective Market Hypothesis (Fama, 1991). Research over the recent 10-15 years has shown further use and perception of technical indicators as engineered features and parameters in the machine learning forecasting models (Nelson et al., 2017).

The third approach has emerged, driven by the availability of large-scale financial data and advances in computational power. Contemporary machine learning and deep learning tools can learn from experience while the latter can create layered neural networks and ingest unstructured data. This approach has taken the best of fundamental and technical analysis, and it has found its realization in the modern forecast frameworks.

The theme of stock market price and trend forecasting attracts significant attention from the professional financial market players like brokers and traders as well as wider public, which become involved in personal management of their savings and investments due to the complexity of stock market behavior: chaotic, non-linear in nature, sensitive to geopolitics, economy trends and investor sentiment, which makes forecasting a persistent challenge (Guegan, 2009). Academic interest also fits well in the theme because of the high variety of approaches and the potential of practical application of new artificial intelligence methods. It makes the research in this area a fruitful field for innovations and a perspective for practical recommendations for the above-mentioned groups of professional and seasonal investors

Theoretical advancement and practical tools may significantly contribute to informed decision-making in financial markets, creating potential real-world impact and methodological innovation.

2.1.1 The Stock Market

Stock market structural complexities are not exhaustive in explanation of the investors' often irrational actions. While Efficient Market Hypothesis (Fama, 1970) states that stock market incorporates all available information and it is reflected in the stock value, empirical studies (Shiller, 2003; Barberis & Thaler, 2003) suggest that phycological biases like overconfidence or fear may drive the price movements in one or another direction from the true stock value. These behavioral indiscretions bring additional complexity and increase the overall price change dynamics, which makes statistical and linear methods less applicable and accurate, while increasing the applicability of machine learning models, which are capable of detecting non-obvious patterns in historical data without the data rationality or equilibrium assumptions. Adaptive Market Hypothesis (Andrew Lo, 2012) argues that the "Markets are not always efficient but adapt over time", which makes market efficiency context-depended rather than universal. The most important concept differences are related to "episodic predictability" versus "random walk" and rare or brief inefficiencies compared to natural and cyclic, which create much broader opportunities for accurate forecasting framework creation.

Financial market can be categorized in two major regimes: bull market, usually associated with optimistic movement in the stock market. According to (Fama and French, 1988) bull market exhibits positive serial correlation and returns and more predictable trending patterns (Andrew Lo, 2012) due to lower noise and more market stability, which in turn could make statistical time series forecasting models a good fit. Bear market, on the contrary, is associated with stock prices plummeting rapidly, which brings investors' making money on mostly quick selling of the stock. Bear market may be associated with panic behavior (De Bondt & Thaler, 1985). Breakdown of linear relationships, high volatility and negative skewness are the empirical characteristics of this regime. As a result, forecasting frameworks that model better performing heteroskedasticity would be outperforming standard autoregressive algorithms. These regimes are often referred to as recovery and crisis ones, which by default cannot be categorized as linear and time dependent. Volatility clustering phenomenon (Cont, 2001) shows that high volatility events are likely to occur in close succession further followed by

calmer periods. That makes adaptive methods with the use of artificial intelligence and machine learning more applicable because of their ability to model long-term dependencies and non-linear relationships.

The above considerations demonstrates that the stock market continues to be a demanding research field in time series forecasting and complex financial environments.

2.1.2 Economic Significance of Accurate Forecasting

The task of predicting the behavior of certain stocks and making a forecast on the future price has perpetually been one of the most difficult problems in scientific research of finance because of its importance and high volumes. In “A Random Walk Down Wall Street” Malkiel claimed that the stock price cannot be forecasted with over 50% of probability based on the analysis and patterns of the previous periods. Makiel’s setting the right benchmark against which the contemporary forecasting models must justify themselves and prove that it can consistently outperform the market.

Analysts and academic researchers have managed to introduce numerous models over time to predict future stock price movements. These cover the transition from the initial statistical models to the machine and deep learning models that are now in use. The development of new forecasting methods has been maintained for decades, driven by the complexity and unpredictability of financial markets. Although there are arguments on forecast efficiency from Efficient Market Hypothesis (Fama, 1970), Intelligent Investor (Benjamen Graham, 2006), The little book of common-sense investing (Bogle, 2007) to behavioral finance theories (Shiller, 2003), many present the finding that the financial data contains some hidden patterns or structures within financial data which may then be used to increase overall forecast accuracy. Thus, stock price prediction after decades of many studies is still one of the most interesting and evolving research areas in finance.

Interest in accurate forecasting is fueled by many factors including investment strategy, capital allocation, risk management, algorithmic trading and other economic aspects. Financial markets are concerned with continuous complexity increase and data abundance, which in turn require more precise and accurate models with clear specializations for specific conditions. In the following paragraphs the forecasting approaches will be addressed in more

detail starting from the classical time series models and further to adaptive artificial intelligence machine learning practices used in finance.

2.2 TRADITIONAL TIME SERIES MODELS

Traditional time series models are fundamental statistical frameworks used for analysis, modeling and forecasting of stock market parameters including price and volumes. Time series forecasting refers to the process of using historical data to make informed predictions about future values. It plays a crucial role in a wide range of industries, enabling decision-makers to anticipate future trends in areas such as sales, inventory planning, weather forecasting and financial markets. By identifying patterns, cycles and shifts in historical data, time series methods help transform raw chronological data into meaningful insights for both short-term and long-term strategic planning.

In the context of stock market analysis, time series forecasting aims to capture how prices evolve over time by detecting structure in past behavior. Among the commonly used techniques are moving averages, exponential smoothing and more complex models like ARIMA with many enhancement variants. Statistical models dominated empirical finance and econometrics (Hamilton, 1994; Box & Jenkins, 1976) before machine learning frameworks have taken the helm and they are still highly popular for multiple use cases and benchmarking for forecasting accuracy and interpretability. Most time series models work by decomposing data into three primary components: trend, seasonality and irregular fluctuations. The trend represents a long-term directional movement in the data, either upward, downward or flat. For example, a multi-year increase in a company's stock price may reflect sustained growth in revenue, market confidence or macroeconomic improvements (Box & Jenkins, 1976). Seasonality refers to patterns that repeat at regular intervals such as weekly, monthly or yearly cycles. These are often linked to consumer behavior or calendar-based events. For instance, retail sales typically surge during the end-of-year holiday season, and many companies prepare for this by increasing inventory or working capital (McLaney, 1979).

Finally, irregular or unexpected events capture random shocks or structural breaks that cannot be predicted using historical data alone. A striking example was the COVID-19 pandemic, which triggered a global stock market crash in early 2020, disrupting nearly every forecast model in use. Similarly, geopolitical events such as military conflicts in oil-producing regions

can lead to sudden spikes in energy prices, directly affecting the stock performance of companies in related sectors.

Understanding these components is essential for selecting appropriate forecasting models. While some techniques are well-suited for capturing trends and seasonality, others are better equipped to adapt to nonlinearity or respond to sudden, unstructured changes. The following section begins with an overview of traditional models that have long served as the foundation of time series forecasting in financial analysis.

2.2.1 ARIMA

The Autoregressive Integrated Moving Average (ARIMA) model is one of the most established statistical tools for time series forecasting, particularly suited to univariate data where past observations can be used to predict future outcomes. Introduced by Box and Jenkins in the 1970s, ARIMA remains widely used in economics and finance due to its strong theoretical foundation, diagnostic tools and effectiveness in modeling short-term dependencies. It is especially valuable for forecasting financial metrics such as stock prices or earnings, where recent trends and autocorrelations provide meaningful signals.

An ARIMA model is a type of regression analysis that measures one dependent variable's strength against other changing variables. The model aims to forecast financial market movements by analyzing value differences within the series, not the actual values.

In compact polynomial notation, an ARIMA (p, d, q) process can be written as:

$$\phi(B)(1 - B)^d y_t = \theta(B)\varepsilon_t$$

where B is the backshift (lag) operator, such that $By_t = y_{t-1}$ and ε_t is a white noise error term.

$\phi(B)$ is the autoregressive (AR) polynomial of order p:

$$\varphi(B) = 1 - \varphi^1 B - \varphi^2 B^2 - \cdots - \varphi_{\mathrm{p}} B^{\mathrm{p}}$$

It collects the AR coefficients $\phi_1, \ldots, \phi_p$ and represents how the current value y_t depends on its own past values $y_{t-1}, \ldots, y_{t-p}$.

$\theta(B)$ is the moving-average (MA) polynomial of order q:

$$\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q$$

It collects the MA coefficients $\theta_1, \dots, \theta_q$ and describes how y_t depends on current and past shocks $\varepsilon_t, \varepsilon_{t-1}, \dots, \varepsilon_{t-q}$. The factor $(1 - B)^d$ represents d-fold differencing of the series, which is used to remove trends and achieve stationarity (If $d = 1$, $(1 - B)y_t = y_t - y_{t-1}$ corresponds to first differences).

Together these components define an ARIMA (p, d, q) model as a differenced series driven by a combination of autoregressive dynamics and moving-average noise:

- *Autoregression (AR):* describes how the current value of the time series depends on its own past values. In an *AR* model y_t is regressed on its lagged observations, such as $y_{t-1}, y_{t-2}, \dots$
- *Integrated (I):* refers to the differencing of the original observations to achieve stationarity. Instead of modeling the raw series, the model uses differences such as $y_t - y_{t-1}$, which removes deterministic trends.
- *Moving average (MA):* models the dependence between the current values of the series and past forecast errors. In *MA* y_t is expressed as a function of current and lagged error terms $\varepsilon_t, \varepsilon_{t-1}, \dots$

In ARIMA models each component is associated with a parameter that follows a standard notation. The general ARIMA(p, d, q) specification uses the integers p, d and q to describe the model order:

- *p* is the number of lagged observations in the model.
- *d* is the number of times the original series are differenced; also known as the degree of differencing. The majority of models do not require more than one order of differencing to achieve stationary and become applicable for *ARIMA* modeling. (Box, Jenkins, Reinsel, 2008), (Hyndman & Athanasopoulos, (2018)
- *q*: is the order of moving-average component, which specifies how many forecast residuals enter the model. It defines the relationship between the current observation and the residual terms from previous observations.

For example, an ARIMA $(1,1,1)$ model applies first-order differencing and uses both the most recent lag and error term to forecast the next value. In practice a variety of tools such as the Augmented Dickey-Fuller (ADF) test (Dickey & Fuller, 1981) are used to assess stationarity, while autocorrelation (ACF) and partial autocorrelation (PACF) plots assist in determining

appropriate values for p and q. Model selection is often guided by minimizing criteria like the Akaike Information Criterion (AIC), (Akaike, 1974).

In an ARIMA model data is differenced to achieve stationarity. A stationary model shows consistent data over time. Most economic and market data show trends, so the purpose of differencing is to remove any trends or seasonal structures. Seasonality or when data shows regular and predictable patterns that repeat over a calendar year or longer periods, could negatively affect the regression model. Without evident stationarity trend presence hinders accurate computations. Most financial time series are non-stationary in their raw form, so differencing is typically applied to stabilize the data. Seasonal effects and external variables are not directly captured by standard ARIMA, though extensions like SARIMA (Seasonal ARIMA) and SARIMAX (SARIMA with exogenous variables) have been developed to address these limitations.

While ARIMA has the advantage of requiring only historical data and is straightforward to implement in most statistical software, it has clear drawbacks. It is not designed for long-term forecasting, tends to underperform in the presence of nonlinearities or sudden regime shifts and is not suitable for modeling volatility without integration with other models like GARCH. Despite these limitations, ARIMA continues to serve as a strong benchmark and a valuable tool for evaluating the performance of more advanced forecasting techniques.

2.2.2 SARIMA

The Seasonal AutoRegressive Integrated Moving Average (SARIMA) model extends the basic ARIMA framework by explicitly incorporating seasonality factors into the time series. These models are more effective in the environment, where seasonality can be clearly identified, which became demanded in many different economies and real-life domains including finance sector and economic indicators. Important advantage of the models concerns with its ability to integrate non-seasonal and seasonal components, making the model robust and enhancing its flexibility of complex time series data (Brockwell & Davis, 2016)

A Seasonal ARIMA (SARIMA) model is typically written as:

$$\text{SARIMA}(p, d, q) \times (P, D, Q)_s$$

where:

- p, d, q are non-seasonal autoregressive order, differencing order and moving-average order.
- P, D, Q are seasonal autoregressive order, seasonal differencing order, and seasonal moving-average order.
- S is the length of the seasonal cycle (number of observations per season), for example $S = 12$ for monthly data with annual seasonality.

Recent work in the time series literature offers a clear and modern discussion of the advantages of SARIMA models but also points out several important limitations. First, SARIMA generally requires relatively long time series to estimate the parameters reliably and to detect seasonal patterns. Second, the model is built on linear relationships and may struggle to capture more complex or irregular forms of seasonality that often appear in real data. Third, selecting the orders p, d, q, P, D, Q can be challenging in practice: the search space is large, the procedure is sensitive to modeling choices, and there is a risk of overfitting if the model becomes too complex (Majka, 2023).

2.2.3 SARIMAX

The Seasonal AutoRegressive Integrated Moving Average model with exogenous variables (SARIMAX) incorporates both seasonal and external factors for time series predicting modeling. SARIMAX has been created to address external factors like market sentiment, company fundamentals and macroeconomic parameters on top of historical patterns (Hyndman & Athanasopoulos, 2018).

A Seasonal ARIMA with exogenous regressors (SARIMAX) model can be written as:

$$\text{SARIMAX}(p, d, q) \times (P, D, Q)_s + X_t$$

where:

- p, d, q are non-seasonal autoregressive (AR), differencing (I) and moving average (MA) orders.
- P, D, Q are seasonal autoregressive, seasonal differencing, and seasonal moving-average orders.
- S is the length of the seasonal cycle (for example, $s = 12$ for monthly data with yearly seasonality or $s = 52$ for weekly data with yearly seasonality).

- X_t is a vector of exogenous regressors observed at time t.

SARIMAX proved to be highly effective in capturing seasonality in time series models like quarterly effects, monthly volatility cycles, annual taxation cycles and others. Exogenous drivers like interest rates, unemployment rates, oil prices and other market indices contribute to forming structurally informative forecasts. Finally, SARIMAX is one of the few linear models that manage to combine trends, seasonality, outside events and its influence as well as moving average noise and autocorrelation (Shumway & Stoffer,2017).

2.2.4 GARCH

Although, Generalized Autoregressive Conditional Heteroskedasticity GARCH model is not a focus of this research work, it is imperative to provide the basic description and the key distinguishing features as it will be referred to through the thesis. GARCH model was developed as an extension to ARIMA model to address the volatility of financial returns. While ARIMA targets modeling the levels of time series and their dependencies, GARCH focuses on modeling volatilities of the residuals from this mean process (Bollerslev, T. (1986).

In contrast to ARIMA, GARCH models introduce a mechanism to model the time-varying volatility of the residuals. The GARCH (p, q) model can be expressed as follows:

- Mean Equation:

$$y_t = \phi + \sum_{\{i=1\}}^{p} \alpha_i y_{\{t-i\}} + \sum_{\{j=1\}}^{q} \beta_j \varepsilon_{\{t-j\}} + \varepsilon_t$$

- Variance Equation:

$$\sigma_t^2 = \omega + \sum_{\{i=1\}}^{\{p\}} \alpha_i \varepsilon_{\{t-i\}}^2 + \sum_{\{j=1\}}^{q} \beta_j \sigma_{\{t-j\}}^2$$

Where y_t represents the time series, ε_t are the residuals, σ_t^2 indicates the conditional variance and the parameters ϕ, α_i, β_j, and ω satisfy certain non-negativity constraints. (Tsay, 2010). In contemporary practices GARCH model can be viewed as an extension of ARIMA framework for the cases where the focus shifts from predicting the level of a time series to predicting its conditional variance. In financial contexts like, for example stock value forecasting considered in this thesis, after fitting an ARIMA model to capture the mean

dynamics, it could be a good practice to apply a GARCH model to the residuals of this ARIMA model to capture the associated time-varying volatility. Instead of this relatively common approach this thesis considers another more contemporary hybrid model PROPHET and further explores its characteristics versus LSTM model.

2.3 PROPHET

PROPHET is a time series forecasting model developed by Facebook's Core Data Science team and introduced by Taylor and Letham (2018) with the goal of making accurate forecasting accessible to analysts and practitioners without deep statistical training. The model was designed to be highly automated, robust to missing data and outliers and capable of capturing complex seasonal patterns in business or financial time series. PROPHET has become widely adopted in applied data science settings due to its ease of use and flexibility, particularly in situations where quick deployment and interpretable forecasts are important. Its architecture is based on an additive model that decomposes a time series into distinct components such as trend, seasonality, holiday effects and random noise, making it especially suitable for use cases like sales forecasting, web traffic prediction, and financial time series where these patterns are commonly observed.

PROPHET models time series data using an additive framework, where the observed value is expressed as a sum of interpretable components:

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

Here, $g(t)$ represents the trend component, $s(t)$ accounts for seasonality, $h(t)$ captures holiday effects and ϵ_t is the error term assumed to be normally distributed. This modular design enables users to isolate and interpret each pattern independently which is useful in applied business and financial contexts.

The trend component in PROPHET can be either linear or "logistic growth", and it automatically detects changepoints: locations in time where the trend shifts significantly. The model allows users to control the sensitivity to changepoints via the changepoint_prior_scale parameter, enabling a balance between flexibility and overfitting.

The seasonality component models periodic patterns using a Fourier series. PROPHET supports both yearly and weekly seasonality by default, and users can define additional custom seasonalities (e.g., quarterly or monthly cycles). Unlike traditional time series models

PROPHET does not require the data to be strictly periodic, which gives it an advantage in handling real-world, non-stationary data (Taylor and Letham, 2018). The holiday component is optional but highly flexible. Users can supply domain-specific holidays (e.g., Black Friday, earnings announcement days) which may introduce temporary but impactful deviations in the time series. PROPHET treats these holidays as binary regressors that influence the model during specified windows.

This flexible structure allows PROPHET to handle complex time series dynamics while maintaining a high degree of interpretability, making it attractive for forecasting problems where transparency and customizability are key. PROPHET does not require the time series to be stationary or linear, unlike ARIMA family frameworks. Instead, it assumes that the data can be decomposed into distinct additive components: trend, seasonality, and holiday effects. Through this decomposition PROPHET model represent structural changes in data, as sudden trend shifts by introducing changepoints. The model also assumes that seasonal patterns are periodic and can be captured using Fourier terms. While this works well for business and economic time series with stable cycles, it may be less suitable for purely noisy or non-seasonal financial data. Finally, the assumption of component additivity enhances interpretability but limits PROPHET's ability to capture nonlinear interactions. One of the key PROPHET advantages relates to its being very fast in computations and its not requiring any data preprocessing, which is the case for ARIMA family frameworks.

PROPHET has gained traction in financial forecasting tasks that require quick implementation, interpretability and robustness to missing or irregular data. It has been applied to predict stock prices, trading volume, volatility trends and macroeconomic indicators such as exchange rates or commodity prices. Its ability to model seasonality and detect structural changes makes it well suited for time series that exhibit calendar-driven behavior, such as end-of-quarter effects, earnings announcements or holiday-related trading anomalies. For instance, Patel et al (2020) applied PROPHET to forecast Bitcoin closing prices, while Mansour and Weng (2021) applied it against LSTM and ARIMA models, finding it effective in capturing trend and weekly seasonality. PROPHET was also referenced in exploration forecasting of S&P500 equities in the comparative benchmarking studies by Krauss et al. (2017).

PROPHET's primary advantage lies in its simplicity and accessibility. Its modular structure, automatic detection of changepoints and ability to incorporate custom holidays and

seasonalities make it particularly appealing in applied financial settings where interpretability and ease of deployment are essential. The model handles missing data and outliers gracefully and requires minimal manual tuning, which makes it useful for building automated forecasting pipelines or generating quick insights into dynamic business environments.

However, PROPHET also has notable limitations. It is primarily designed for univariate time series and may struggle with high frequency financial data that exhibits irregular volatility or lacks clear seasonal structure. The additive framework, while intuitive, can oversimplify complex interactions in the data and the model's performance may degrade in scenarios dominated by nonlinear dependencies or abrupt market shocks (Mansour & Weng ,2021, Taylor & Letham, 2018). In such cases, more flexible approaches like machine learning or deep learning models may offer better predictive accuracy.

Despite these constraints, PROPHET remains a valuable baseline model for time series forecasting, particularly in financial domains that benefit from transparency, interpretability, and quick iteration.

2.4 DEEP LEARNING MODELS

The recent rise of deep learning, particularly Recurrent Neural Networks (RNNs) and their extensions, has opened new avenues for modeling such complexity. Among these, Long Short-Term Memory (LSTM) networks have emerged as one of the most promising architectures for time series prediction. Originally introduced by Hochreiter and Schmidhuber (1997), LSTMs are specifically designed to retain information across long sequences and overcome the vanishing gradient problem that affects traditional RNNs. Their ability to model sequential dependencies and uncover nonlinear structures makes them highly suited for financial forecasting tasks.

As stock prices are influenced by a mix of macroeconomic variables, investor sentiment and historical patterns, LSTM models offer a powerful alternative to classical techniques. They learn directly from raw data, require fewer statistical assumptions, and have demonstrated strong empirical performance in various studies involving equity markets, indices, and cryptocurrencies (McNally et al, 2018; Nelson et al., 2017).

2.4.1 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) were among the first deep learning architectures specifically designed to handle sequential data, making them a natural candidate for time series forecasting (Elman, 1990). Unlike traditional feedforward networks, RNNs process inputs in order and maintain a hidden state that is updated at each time step, allowing the network to retain information from previous observations. This makes them well-suited for tasks where context and temporal dependencies matter, such as speech recognition, language modeling and financial data analysis (Graves, 2013).

However, standard RNNs face a critical limitation when it comes to learning long-term dependencies: the vanishing or exploding gradient problem. As gradients are backpropagated through many time steps during training, they tend to diminish or grow exponentially, which makes it difficult for the model to learn from distant past events. This issue severely affects the stability and performance of RNNs, particularly in financial applications where key signals may emerge over extended time horizons (Bengio et al.,1994).

In addition to gradient problem basic RNNs have limited capacity to differentiate between short-term noise and long-term structure. It is a crucial challenge in financial markets, where trends often unfold gradually and may be temporarily obscured by volatility (Hochreiter et al., 2001). These drawbacks highlighted the need for more robust sequence models, leading to the development of more advanced architecture such as Long Short-Term Memory networks.

2.5 LONG SHORT-TERM MEMORY NETWORKS

Long Short-Term Memory networks were introduced by Hochreiter and Schmidhuber (1997) as an enhancement to standard RNNs, designed to address their difficulty in learning long-term dependencies. The key innovation of LSTM lies in its internal architecture, which introduces a memory cell regulated by three gates: the input gate, forget gate, and output gate. These gates control how much information is stored, updated or discarded at each time step. This architecture allows LSTM models to capture both short and long-term patterns in sequential data with greater stability and effectiveness.

Compared with simpler recurrent architectures, LSTM models provide more stable gradient flow across time. It makes them especially attractive for noisy time series data. While Gated

Recurrent Units (GRUs) offer a lighter alternative with fewer parameters, LSTM models remain the dominant architecture due to greater capacity and robustness.

In financial forecasting LSTMs independence from assumptions of linearity or stationarity is particularly valuable as stock prices often reflect delayed responses to macroeconomic indicators, investor sentiment or external events. LSTM networks learn the temporal structure directly from historical data, identifying complex relationships, nonlinear dynamics and volatility patterns. The sliding window approach used for training allows the LSTMs to extract predictive information from sequences of past observations and multiple input features, such as price, volume and technical indicators.

Despite the advantages, LSTMs have several challenges. They are sensitive to hyperparameters and scaling choices, and require large training datasets, which are computationally demanding. As they learn patterns implicitly, they provide limited interpretability compared to classical models. Their performance can also be highly impacted when the training data contains multiple market regimes and structural breaks, which leads to overfitting.

Several empirical studies have demonstrated the effectiveness of LSTM models in stock market prediction. For instance, Fischer and Krauss (2018) applied an LSTM architecture to S&P 500 stock data and found that it significantly outperformed traditional methods such as logistic regression and random forests in terms of directional accuracy and profitability. Similarly, Nelson et al. (2017) compared LSTM to ARIMA and other neural models on the Brazilian stock market and reported superior performance in terms of both RMSE and prediction stability.

2.6 COMPARATIVE STUDIES

There are dozens of recent studies which explore the use of statistical and machine learning models for stock price forecasting, with many comparing different ARIMA-class models to more advanced LSTM architectures. These comparisons are often limited in scope, market and time horizon and focus on a single asset or on evaluating models under very specific conditions. While the findings are useful, they may not always reflect broader applicability.

This thesis follows on one side a similar line of research by comparing ARIMA and LSTM models on daily stock price data from selected S&P500 companies but also extends the scope by adding PROPHET model to consideration. Only a few research papers were identified that cover all three models and each of them was too generic and provided limited practical guidance. The goal here is not to introduce new methods but to observe how established models perform in the experimental setting, using the same preprocessing steps, prediction targets, and evaluation metrics. This allows for a more balanced comparison and contributes to the growing body of work on time series forecasting in finance and provides practical advice to market practitioners and market professionals willing to use modeling tools for financial forecasting.

3. METHODOLOGY

Aligned with the goal of comparing the forecasting performance of different models on stock price data, this chapter is structured to follow the full path from raw data to model implementation. The research methodology is strategically divided into several interconnected parts: the collection of the data, the computational environment in which the analysis is carried out, the initial exploration of the series, the definition of evaluation criteria, and finally the specification of the forecasting models. This sequence reflects the practical workflow of the study and make each step in the empirical process transparent.

3.1 DATA COLLECTION

Data collection usually becomes the first step in the task of forecasting the next day stock values. The abundance of data availability both free of charge and commercially often makes the task of choosing the right source a challenge due to high potential cost of mistake. This thesis having an academic nature relies on the Yahoo Finance for collection. Yahoo Finance provides reliable historical stock market data, which is convenient for this thesis modelling exercise and does not require the utmost accuracy or real time access to the ongoing trading, which the commercial traders would need. Additional important reason for choosing Yahoo Finance was the support of Pandas data frame for prices, which was a good fit for further Python modelling.

The empirical foundation of this study is built upon historical market data for number of tickers, selected for its liquidity, relevance and extensive publicly available trading history. The research is based on daily trading records covering more than twenty years. The period was chosen to capture a sufficiently long and diverse market history that covers distinct market phases expansions, downturns, and episodes of heightened volatility. The dataset includes standard OHLCV (Open, High, Low, Close, Volume) fields without dividend or split adjustment.

3.2 PYTHON

Python is a high-level programming language that has become a natural choice for machine learning applications. Its prototyping approach creates the bulk of ML scripting, making programming more accessible and faster than in languages like C++ or Java. In addition, Python benefits from a large and highly active professional community, as well as a rich

ecosystem of mature libraries for numerical computing and machine learning. Although artificial intelligence is still in a relatively early stage of its broader industrial adoption, it is expected to grow rapidly in the coming years (Raschka, Patterson, & Nolet, 2020), and Python's characteristics position it well to remain at the center of this development. Recent work also suggests that, thanks to its high-performance, production-ready libraries, Python is suitable not only for prototyping but also for large-scale research and deployment (Chen et al., 2024).

3.2.1 Python Tools

NumPy is the fundamental part of Python, which provides the tools for fast numerical computations and operations with arrays. Ability to provide high-performance performance for multi-dimensional arrays object tools makes this package central in education, scientific research field and practical applications (Van der Walt, Colbert & Varoquaux 2011). Python's ability to rely on efficient vectorized operations, supported by low-level implementations in C and FORTRAN and additional optimized linear algebra libraries allows it to achieve high computational performance and makes it one of the leading tools for machine learning, even on standard hardware.

For modelling in this thesis, Pandas (Python Data Analysis) package is used. It was chosen because of the open-source analysis and manipulation framework availability. Pandas is built on top of NumPy and provide means for managing large data sets, which is important for this work. Data Cleaning, Analysis and Transformation tools as well as the data range generation and frequency conversion make this package useful and relevant for the modelling. Its seamless integration with other core libraries such as NumPy, Matplotlib and Plotly, all of which are used in the practical part of this thesis, further underscores its suitability for the present work. The key functions that this package contributed to the modelling are reading the data from databases, cleaning and preparation of data, merging, joining and reshaping data sets and performing statistical analysis. Matplotlib and Plotly packages are used for modelling result visualization in this work. It is a natural choice to choose these instruments due to its being a part of the open-source Python framework.

Scikit-learn or "sklearn" library is used in the work for classification, regression, clustering and dimensionality reduction functions. It is built on top of NumPy, SciPy and Matplotlib libraries

and primarily used for machine learning tasks like learning, data preprocessing and model evaluation. Though, the package has certain limitations for large data sets and complexity of customization it is not the limitations or restraints for the current research.

Beyond the libraries discussed above, the Python ecosystem includes a wide range of additional tools that cover almost every aspect of data analysis and machine learning. It would be neither feasible nor necessary to describe each of them in detail, but several of these libraries were used in the present work for model building, tuning, and evaluation.

3.3 DATA EXPLORATION

The dataset consists of daily OHLCV observations for several large capitalization companies listed in the *S&P500* index. Daily observations were downloaded for the period between 2004 and 2025 for selected companies as long as historical data was available. This long period includes different market conditions, such as financial crises, high volatility phases and the post-COVID recovery.

Each dataset contains the following variables:

Table 3.3-1 *Variables in the dataset*

Variable	Description
Date	Trading day timestamp
Open	Opening price of the stock
High	Highest price recorded during the trading session
Low	Lowest price recorded during the trading session
Close	Closing price of the stock (main target variable in all models)
Adj Close	Adjusted closing price, including dividends and stock splits
Volume	Number of shares traded on that day

Before conducting any statistical analysis, the raw data for each stock was first checked for missing records, duplicated rows and irregularities in the timestamp. As expected for U.S. equities listed on Nasdaq or the NYSE no genuine missing values were found. Days without entries correspond to weekends or official market holidays, which is a normal characteristic

of financial time series. There were no duplicated timestamps found, and the price columns also did not contain zeros or negative values, which can occur in poorly constructed datasets. Therefore, no imputation or corrective preprocessing was needed at this stage. The only adjustment made when loading the data was to reset the index and make sure that all observations were chronologically ordered.

Basic descriptive statistics were calculated for all numerical variables to get a first idea of how each stock behaves. As expected, the values differ across companies, but some general patterns are similar. Closing prices show strong long-term upward tendency for all selected stocks, which reflects the overall growth in the technology sector and the wider equity market over the period. The Standard deviation of daily price changes, however, varies quite a lot between assets. More volatile stocks, such as Nvidia (NVDA), naturally show wider ranges and larger swings, while more established companies like Apple (AAPL) tend to move in a narrower band.

Another exploration step was carried out for all selected stocks to see how each series evolved over time and to review its basic time-series properties. Overall, the series shared a consistent pattern. In the early years, especially before 2015, volatility, smaller price movements were smaller and structural changes tended to occur more gradually. This is in line with macroeconomic and market conditions that were relatively calmer compared to recent years. From around 2019 onwards, the behavior of the series changed: price swings became more volatile, trends reversed more abruptly, and the market reacted more strongly to events such as the COVID-19 pandemic, interest rate shifts and geopolitical tensions.

These observations also have implications for forecasting. Using the entire historical dataset without distinction might distort model performance, as it combines different market regimes, strong non-stationarity and dynamics that are no longer relevant to current market conditions. This issue affects all forecasting model. Even flexible approaches such as LSTM struggle when the underlying data distribution changes significantly over time. In practice, very long input histories do not always help - they can slow down training, increase memory usage and worsen accuracy, for instance when model is forced to learn from outdated or inconsistent patterns, especially in financial markets where recent data is often more informative than distant history.

To address this issue all datasets were restricted to a common five-year window starting on 1 January 2019. This period still contains rich daily dynamics but is more in line with current market conditions. The resulting dataset is better suited to the objectives of the present study.

In summary, the exploratory analysis provided a basic but clear understanding of how the selected stock prices behave over time. We checked data quality, looked at descriptive statistics, visualized price and return dynamics and examined key time-series properties such as trends and volatility. Based on these steps, the dataset can be considered ready for use in the empirical part of the thesis.

3.4 METRICS

The most commonly used metrics for time series forecasting, regression modelling, neural networks address the needs of comparison of the models like ARIMA family of frameworks, PROPHET and LSTM, which are researched in this thesis. The effectiveness of these metrics is highly recognized in the literature (Hyndman & Koehler, 2006) and create the proper evaluation environment for machine learning and time series data modelling in Python (McKinney, W. 2022). For the purpose of comparing the forecasting models examined in this study, the following accuracy metrics will be used:

$$MAE = \frac{1}{n}\sum_{\{t=1\}}^{n} | y_t - \{\hat{y}\}_t |$$

The Mean Absolute Error (MAE) is the average absolute difference between the actual values y_t and the corresponding forecasts $\hat{y}_t$. It is an intuitive and scale dependent measure that determines the typical size of the forecast error in the same units as the original series.

$$MSE = \frac{1}{n}\sum_{\{t=1\}}^{n} (y_t - \{\hat{y}\}_t)^2$$

The Mean Squared Error (MSE) is the average of the squared differences between the actual values y_t and the corresponding forecasts $\hat{y}_t$. Because the errors are squared, MSE is particularly sensitive to large deviations. It penalizes occasional large errors more strongly than many small ones. Lower MSE values indicate better forecasting performance, although the metric is expressed in squared units of the original series and is therefore less directly

interpretable than MAE. For this reason, MSE is often used in conjunction with its square root, RMSE, which restores the original scale of the data.

$$RMSE = \sqrt{\{MSE\}}$$

The Root Mean Squared Error (RMSE) is the square root of the MSE and therefore combines the sensitivity of squared errors to large deviations with the advantage of being expressed in the same units as the original series. It is one of the most commonly used accuracy measures in time series forecasting, as it provides a clear indication of the typical size of the forecast error on the original scale of the data (Hyndman & Athanasopoulos, 2006).

$$MAPE = \frac{100}{n} \sum_{\{t=1\}}^{n} \left| \frac{\left((y_t - \widehat{y_t})\right)}{y_t} \right|$$

The Mean Absolute Percentage Error (MAPE) measures the average absolute forecast error as a percentage of the actual values. It is a scale-independent metric and is therefore useful for comparing performance across series with different units or magnitudes. However, MAPE can become unstable when actual values are close to zero, which requires some caution in interpretation.

$$R^2 = 1 - \frac{\left(\sum_{\{t=1\}}^{n} (y_t - \widehat{y_t})^2\right)}{\left(\sum_{\{t=1\}}^{n} (y_t - \bar{y})^2\right)}$$

Casella, Georges (2002) call R^2 "a measure of the goodness of fit of a model". Its value resides in the explanation of the variance in the observed data rather than magnitude and is subject to meaningful interpretation. The closer value to 1 or 100%, the better is forecasting accuracy. Though negative values and low values also have valuable interpretations.

3.5 Development of ARIMA Models

The ARIMA (AutoRegressive Integrated Moving Average) model is used as the traditional benchmark for time series forecasting. The historical series are first decomposed into two components: a smooth trend and a residual term. The residuals are later modelled with an ARIMA model, and the final forecast is obtained by adding the predicted residuals back to the smoothed trend.

The idea behind this setup is straightforward. Stock prices typically evolve on two layers at the same time: a slow, underlying direction (for example, a gradual upward trend) and a noisy

pattern of day-to-day fluctuations. By smoothing the series, the long-term trend becomes easier to identify and work with, while the residual component captures the more irregular short-term movements. This separation allows ARIMA to focus on modelling the high-frequency dynamics, rather than trying to fit both long and short-term behavior simultaneously.

A key practical requirement of ARIMA models is that the underlying time series should be stationary. In practice, financial time series like stock prices tend to drift over time which means the stationarity assumption is almost never met in the raw data (Box & Jenkins, 1970; Box, Jenkins & Reinsel, 2008). Thus, the modelling pipeline first test whether the series have a unit root and, if it does, determine how many times the series needs to be differenced to make it stationary. In our study this issue is addressed by applying the Augmented Dickey Fuller (ADF) test to formally assess stationarity and to determine the appropriate order of differencing required for the series.

3.5.1 ARIMA Modelling on Residual Dynamics

After extracting the short-term residuals of the stock price series using a moving average trend decomposition, the next step is to model these residual fluctuations with an ARIMA model. The model is estimated only on the residuals rather than fitting the model directly to the original closing prices, because closing prices are usually non-stationary and driven by long-term trends. This thesis relies on Python version of automated selection procedure, often referred to as auto_arima, by analogy with R but not related to it anyhow. It restricts the search over the AR and MA parameters, while fixing the differencing order to the value (usually 0, 1 or 2) calculated on the previous stage of stationary analysis. To reduce the computational costs compared to the full grid processing the staged approach is used and by minimizing the Akaike Information Criterion (AIC) the best model is selected.

Performance of the modelling highly depends on the chosen parameters p, d, q. That makes the process of choosing sometimes a complex task, which needs to be approached wisely to avoid overfitting risk on noisy high-frequency data, which is a common characteristic of stock market. Another reason for keeping p and q parameters relatively low value relates to weak serial dependence of returned stock. Many studies constrain p and q values by a certain small value, for example 5, to constrain complexity. Empirical studies (Lo & MacKinlay, 1988; Fama,

1970) find only weak, short-range serial dependence in security returns, while Roll, 1984; Aït-Sahalia et al., point out that high p and q values tend to overfitting, because market microstructure noise may induce spurious autocorrelations at high sampling frequencies. Hyndman & Athanasopoulos (2018) recommend the ranges of $\max(p, q) = 5$, which will be used as the upper limits in this research.

Once the ARIMA model is trained, it is used to generate forecasts for the residual component over the chosen forecast horizon. However, the ultimate objective of the study is to forecast future closing prices rather than residual on their own. Therefore, a reconstruction step is applied: the last available value of the smoothed trend is used as an anchor and the ARIMA model provides a sequence of predicted residuals over the forecast horizon. These residual forecasts are then accumulated over time and added to the last trend value, effectively building a forward path for the closing price.

Formally, the forecast for the closing price at horizon h is given by

$$\hat{y}_{\{t+h\}} = Trend_t + \sum_{\{j=1\}}^{\{h\}} \hat{\varepsilon}_{t+j}$$

where Trend_t is the last observed value of the smoothed trend and $\hat{\varepsilon}_{t+j}$ are the ARIMA-based residual forecasts for each future step.

That approach provides basis for calculation of a stable level of trend with underlying direction of the stock price. Forecast of cumulative residuals describes how the series are expected to move around this level in the short term. Thus, the resulting forecast function combines a smooth long-run trend level with short-term autocorrelated deviations, producing more stable and easier to interpret results.

3.5.2 SARIMAX Modelling

In the quest for forecasting accuracy improvement of basic ARIMA framework several extended models were introduced. These model extensions are well justified and described in the literature sources both empirically and practically demonstrate the influence of earlier non-accounted external factors and seasonal structures on model performance (Hyndman & Athanasopoulos, 2018; Box, Jenkins & Reinsel, 2008). SARIMAX became the major outcome of the evolution incorporating seasonal components and allowing exogenous factors.

Incorporation of the seasonal parameters and exogenous regressors into extended SARIMAX model provides for capabilities of capturing recurring patterns and additional sources of residual series variations. The background for these improvements is related to model improvement expectations due to advanced representation of seasonal cycles which is expected to improve accuracy of residuals forecast and as a result increase quality of future closing prices. In order to find the optimal combination of p, d, q non-seasonal parameters for SARIMAX a systematic grid search is fulfilled. The values were limited by the values of 0 and 1 to limit the processing power consumption and based on the empirical recommendation for this model. A similar restraint for seasonal parameters are introduced to find the optimal combination of P, D, Q among two values 0 and 1. The seasonal period s is set to either 7 for weekly or 30 for monthly cycles respectively. Initially, no exogenous parameters were added to this improvement function, in a way, restricting its functionality to the one of SARIMA model. The grid search of the described SARIMAX specification helps exclude the arbitrary choice of the parameter combination, instead of relying on the weighted comparison of the credible results. With such approach the model is more likely to demonstrate more reliable price value prediction.

As the next improvement step, the incorporating external variables into time series models - SARIMAX has been shown to improve predictive accuracy by accounting for additional variance that cannot be captured by autoregressive components alone (Hyndman, R.J., & Athanasopoulos, G. (2021).

Table 3.3-2 *SARIMAX model with exogenous features*

```
data["pct_change_5"]   = data["Close"].pct_change(periods=5)
data["rolling_vol_10"] = data["Close"].pct_change().rolling(10).std()
data["sma_10"]         = data["Close"].rolling(10).mean()
# RSI (14-day)
delta = data["Close"].diff()
gain  = delta.where(delta > 0, 0)
loss  = -delta.where(delta < 0, 0)
rsi_14    = gain.rolling(14).mean() / loss.rolling(14).mean()
data["rsi_14"] = 100 - (100 / (1 + rs))
exog_vars = ["pct_change_5", "rolling_vol_10", "sma_10", "rsi_14"]
# --- SARIMAX Model Specification ---
model = SARIMAX(
    residual,
    order=(p, d, q),
    seasonal_order=(P, D, Q, s),
    exog=train_exog[exog_vars],
    enforce_stationarity=False,
    enforce_invertibility=False
).fit(disp=False)
```

The following set of exogenous features is engineered based on the underlying structure of the stock's recent price:

$$\text{PCT\_Change\_5}_t = \frac{P_t - P_{t-5}}{P_{t-5}}$$,where P_t is the closing price at time t.

Percentage Change serves as an effective proxy for price momentum. Momentum reflects the tendency of assets that have performed well or poorly in the recent past to continue showing similar behavior in the near future. This idea is strongly supported by the seminal work of Jegadeesh and Titman (1993), who demonstrated that past returns contain statistically significant predictive power for subsequent price movements.

$$\text{Rollling\_Vol\_10}_t = std(r_t, r_{t-1}, \dots, r_{t-9}), \text{ where } r_t = \frac{P_t - P_{t-1}}{P_{t-1}}.$$

Rolling Volatility reflects the degree of uncertainty of risk associated with price movements. As volatility evolves over time, this feature gives the models flexibility to account for sudden increases in market uncertainty (Andersen, et al, 2003).

$$\text{SMA\_10}_t = \frac{1}{10}\sum_{i=0}^{9} P_{t-i}$$,where P_t is the closing price at time t.

The Simple Moving Average serves to smooth price fluctuations and reveal trend. It reduces the risk of overfitting to transient price movements. The use of SMAs as predictive tools is supported by the study of Brock et al, 1992), who found that simple technical trading rules based on moving averages hold statistically significant predictive power in financial markets.

$$\text{RSI}_t = 100 - \frac{100}{1 + RS_t}, \quad RS_t = \frac{\text{Average Gain}_t}{\text{Average Loss}_t},$$

$$\text{Average Gain}_t = MA_{14}(\max(P_t - P_{t-1}, 0)), \quad \text{Average Loss}_t = MA_{14}(\max(P_{t-1} - P_t, 0)).$$

The Relative Strength Index (RSI) is a momentum oscillator commonly used in financial analysis to evaluate the speed and magnitude of recent price movements. It quantifies overbought and oversold conditions on a bounded scale. In this study, the 14-day RSI is included as a feature to capture latent momentum dynamics and investor sentiment extremes. The practical relevance and interpretive strength of RSI are supported by Park and Irwin (2007), whose meta-analysis of technical trading rules confirms RSI's frequent profitability in short-term forecasting scenarios.

3.6 DEVELOPMENT OF PROPHET MODELS

The PROPHET model is mainly designed to break a time series into three parts: trend, seasonality, and noise. Unlike ARIMA and LSTM, which in this research are trained on daily stock prices, PROPHET performs better when the data is aggregated to a monthly frequency. PROPHET model requires the input data to follow a specific schema with two core columns: a time stamp named *ds* and a target variable named *y*. So, we are grouping daily closing prices by month and keeping the last close price of each month as the target value. This aggregation reduces random fluctuations and noise and is consistent with PROPHET's focus on modeling smooth trends and recurring seasonal patterns over longer horizons.

Trend and seasonality alone are not enough to reflect recent market behavior, especially during periods of rapid price swings. A set of lagged price-based regressors is implemented to deal with this limitation. These lagged values represent the stock's price from 1 to 6 periods earlier. Including these lags helps PROPHET capture some autoregressive patterns, because the model by itself does not directly model autocorrelation the way ARIMA does.

Table 3.3-3 *Lagged OHLC features*

Date	Close	CloselLag1	OpenLag1	HighLag1	LowLag1	CloseLag2	CloseLag3
2019-12-02	63.736	0.000	0.000	0.000	0.000	0.000	0.000
2019-12-03	62.599	63.736	64.486	64.723	63.565	0.000	0.000
2019-12-04	63.152	62.599	62.324	62.619	61.837	63.736	0.000
2019-12-05	64.079	63.152	62.990	63.531	62.896	62.599	63.736
2019-12-06	65.316	64.079	63.647	64.153	63.391	63.152	62.599

The lags provided a simple and interpretable way to include past price information while still preserving the model structure consistent with PROPHET's design.

For the baseline specification, PROPHET is configured with an additive structure and the two seasonalities that are included by default: weekly and yearly. Weekly seasonality is sometimes relevant for stock data because trading behavior often differs between the beginning and end of the week, while yearly seasonality captures general recurring patterns such as earnings cycles or end-of-year movements. We intentionally keep this baseline simple to serve as a reference point, allowing later to check whether adding additional features actually improves prediction accuracy.

Table 3.3-4 *PROPHET base model*

```
model = PROPHET (
    yearly_seasonality=True,
    weekly_seasonality=True,
    seasonality_mode='additive'
)
```

After forming the baseline, we started to extend the model to better match the behavior of stock prices. PROPHET allows trend adjustments at points called changepoints, so their flexibility was further controlled using the changepoint_prior_scale parameter. A slightly higher value makes the model more responsive to structural shifts, such as sudden increases or decreases in the stock price. On top of the default trend, two custom seasonalities were introduced: a “monthly” component with the period of 30.5 days and Fourier order 6 and the “quarterly” component with a period of 90 days and Fourier order 8.

PROPHET representation of truncated Fourier series:

$$s_P(t) = \sum_{k=1}^{K} \left(a_k \cos\left(\frac{2\pi kt}{P}\right) + b_k \sin\left(\frac{2\pi kt}{P}\right) \right),$$

The choice of these parameters for Fourier Order K=6 for the monthly and K=8 for quarterly provides more flexibility compared to the default parameters. They fit in the moderate complexity increase option versus default values of 3 and 10 for weekly and annual seasonality and became a healthy compromise between flexibility and overfitting. These values demonstrated the best tradeoff between prediction accuracy and model parsimony (Taylor& Letham, 2018). The US federal holidays were also added as additional regressors to evaluate potential disruptions around official closures. The seasonality_mode was switched to multiplicative, reflecting the belief that the magnitude of seasonal effects scales proportionally with the level of stock price.

3.6.1 PROPHET Model Improvements

Just like in the ARIMA part of the project, we added several engineered features that describe recent price dynamics. These included: percentage change between open and close prices

(pct_change), a 10-day simple moving average (SMA_10), a 12-period rolling volatility measure (volatility_12) and a 3-day lagged momentum indicator (Momentum_3).

An important choice was to exclude the Relative Strength Index (RSI) feature. While RSI worked in ARIMA experiments, in PROPHET it caused problems because it is a nonlinear indicator and overlapped with information already captured by volatility and momentum. PROPHET model tend to behave better with simpler and more linear features, so RSI feature was not being added to reduce the risk of multicollinearity and unnecessary complexity. In contrast, a 3-day lagged momentum indicator is created and perfectly aligns with PROPHET's structure since it reflects only past information (the value is shifted by one period).

Table 3.3-5 *PROPHET model extensions*

```
lag_features = [col for col in data.columns if "lag" in col]
extra_features = ["pct_change", "SMA_10", "volatility_12", "momentum_3"]
all_regressors = lag_features + extra_features
model = PROPHET(
    changepoint_prior_scale=0.1,
    holidays_prior_scale=10.0,
    seasonality_mode='multiplicative',
    yearly_seasonality=True,
    weekly_seasonality=True,
    daily_seasonality=False
)
model.add_country_holidays(country_name='US')
model.add_seasonality(
    name='monthly', period=30.5, fourier_order=6, mode='additive'
)
model.add_seasonality(
    name='quarterly', period=90, fourier_order=8, mode='additive'
)
```

To sum up, the PROPHET setup described in this section results in a ready for use model, aligned with the objectives of this thesis. The settings for trend, seasonality, holidays and additional regressors are chosen based on previous empirical work and on the patterns observed in the data of the analyzed tickers. At the same time, the final model setup is kept as close as possible to the setups used for ARIMA and LSTM, so that the models can be compared fairly in the empirical analysis and modeling results.

3.7 DEVELOPMENT OF LSTM MODELS

The LSTM model represents the deep learning component of this thesis and approaches forecasting in a purely data-driven way without specifying an explicit trend or seasonal structure as in ARIMA or PROPHET. Instead of decomposing the series, the LSTM learns directly from multivariate daily sequences of prices and technical indicators, making it particularly suitable for capturing financial markets' peculiarities.

The LSTM model requires the input data to be represented as fixed-length historical sequences. For this purpose the daily dataset was first normalized and then reorganized into overlapping windows of 60 consecutive trading days. Each window contains the full set of available features, while the corresponding output is the next day Close price. This design allows the model to learn how current price behavior depends on the patterns observed over the previous three months of market activity. The construction of training, validation and test sequences were created with strict chronological order to preserve that each prediction used only information that would have been realistically available at that point in time.

Table 3-6 *LSTM base model.*

```
model = Sequential()
model.add(Input(shape=(X_train.shape[1], X_train.shape[2])))
model.add(LSTM(100))
model.add(Dropout(0.1))
model.add(Dense(1))
model.compile(optimizer="adam", loss="mean_squared_error")
```

The first LSTM specification used in this thesis is a relatively simple one-layer architecture that serves as the base deep learning model for all forecasting experiments. The idea behind choosing a simpler structure was to keep the model expressive enough to learn daily stock price patterns, but not so complex that it would overfit the data. The model takes sequences of 60 consecutive trading days as input, with each day represented by all available normalized features. These sequences are processed by a single LSTM layer with 100 units, which allows the network to capture short-term trends and nonlinear relationships in the data. We applied a small dropout rate of 0.1 to improve generalization and to avoid memorizing noise in the

training set. The output layer contains just one neuron, which predicts the next-day Close price. The model is trained using the Adam optimizer and the mean squared error loss function, both of which are commonly used and work well for regression tasks like stock price forecasting.

3.7.1 LSTM Model Improvements

After the base LSTM model was defined, its informational content was extended with additional technical features in an attempt to improve forecasting performance. A 60-day simple moving average (SMA-60) of the Close price was calculated to represent the medium-term trend and the relative price to this moving average was derived to indicate whether the current price is trading above or below its recent trend. In addition, a 60-day rolling standard deviation of the Close price was computed as a proxy for recent volatility and a 60-day z-score was formed by standardizing the distance between the current price and its moving average.

Several alternative technical indicators were considered, but the decision was made to focus specifically on the relative price to the 60-day moving average and the 60-day z-score for three main reasons. Both indicators directly related the current price to its medium-term trend and volatility, which are fundamental components of short-term market behavior. This made them highly relevant for a sequence-based model that attempts to learn changes around local trends rather than long-range seasonal patterns. Second, these features were simple, interpretable and avoid introducing excessive noise. Many technical indicators (such as MACD, RSI, stochastic oscillators, or multiple SMA/EMA combinations) rely on overlapping information or very short lookback windows, which led to unstable inputs and increased the overfitting when tested. Finally, preliminary experiments showed that adding a large number of engineered features would not improve performance and often would make the model harder to train.

With the extended feature set in place third LSTM specification incorporated a light hyperparameter tuning procedure. Different combinations of key model parameters (including the number of LSTM units, dropout rate, learning rate and batch size) were explored with the aim of obtaining a smoother training process and lower validation error. The purpose of this step was not to design a highly complex network but rather to calibrate the architecture

more carefully so that it trains smoothly and generalizes better to unseen data. The resulting model is therefore treated as a refined version of the feature-enhanced LSTM.

During the development process, several additional deep learning configurations were explored to assess whether more sophisticated architectures could improve forecasting performance. Although these models were not retained for the final comparison, they provided valuable insights into the behavior and limitations of different LSTM variants in the context of this dataset.

A bidirectional LSTM architecture was tested to investigate whether learning patterns in both forward and backward directions would improve predictive power. In a bidirectional network, two LSTM layers are run in parallel: one processes the input sequence in the standard forward direction, while the other processes it in reverse. The outputs of both layers are then combined and the model can incorporate information from the entire sequence. This architecture is known to be effective in tasks such as speech recognition, machine translation and text classification, where the full context of a sequence, including both past and future elements, is available at the time of prediction (Goodfellow, Bengio, & Courville, 2016). However, this logic does not translate well to financial forecasting, as future observations cannot be accessed when predicting the next-day price. In practice, the bidirectional model did not yield meaningful improvements and weas not retained for the final comparison.

Convolutional layers were tested to assess whether local temporal patterns could be extracted more effectively before passing the data to the recurrent component. In a CNN–LSTM architecture, the convolutional part acts as a feature extractor: it applies sliding filters across the temporal dimension to detect short-term structures, such as abrupt price movements or localized volatility spikes. The resulting filtered sequences are then passed to an LSTM layer, which is responsible for modelling longer-range temporal dependencies. This hybrid design is well established in sequence modelling and has been shown to perform effectively in applications such as human activity recognition, audio processing and multivariate time-series classification (Bai, Kolter, & Koltun, 2018).

In practice, however, the CNN–LSTM variants tested in this thesis did not lead to more accurate or stable forecasts. The hybrid model turned out to be quite configuration-dependent and small changes in the convolutional part often led to unstable results and signs of overfitting with training error decreasing while validation performance stagnated or

deteriorated. As a result, the hybrid architecture required substantially more tuning than the simpler LSTM models without delivering consistent gains in out-of-sample accuracy. For these reasons, the CNN–LSTM specification was excluded from the final set of models considered in the comparative analysis

Lastly, a more complex LSTM architecture was explored in order to assess whether a deeper network could capture richer temporal patterns in the data. In this specification, several LSTM layers were stacked on top of each other, with dropout applied after each layer. The idea behind this design was that the lower layers would learn more local dynamics and short-term fluctuations, while the upper layers could extract more abstract representations of medium-term structure in the price series. The model was again trained to predict the next-day Close price based on 60-day input sequences of the same feature set.

Finally, more complex stacked architectures with two, three and four LSTM layers were also tested, with the four-layer version referred to as LSTM-4. The motivation for these models was that a deeper network might learn more structured representations of the data: lower layers could focus on short-term price movements, while upper layers capture broader patterns over longer horizons. These deeper networks were included mainly to see whether increasing the capacity of the LSTM architecture would lead to meaningful improvements over the simpler one-layer models. The empirical performance of LSTM-4 and its comparison with ARIMA, PROPHET and the other LSTM specifications is broader discussed in the subsequent chapters.

4. EMPIRICAL STUDY

Deep literature review defined certain expectations from the results of practical comparison of three commonly used models for next day forecasting. After research and creation of the robust Python models for ARIMA/SARIMAX, PROPHET and LSTM described in Methodology this chapter tests the hypotheses with specific next day financial data. The design of the modelling experiment creates the ground for following evaluations:

- H1: ARIMA and LSTM. LSTM is expected to outperform ARIMA for complex financial series such as daily stock prices of large-cap S&P500 technology companies in terms of next day accuracy measured as RMSE, MAE, R^2, and MAPE.
- H2: ARIMA and PROPHET. Because of strong calendar effects PROPHET is expected to be at least on par with ARIMA or even outperform it for assets with noticeable seasonality.
- H3: LSTM, PROPHET and ARIMA. Recent multi-model comparisons question the absolute supremacy of LSTM over ARIMA and PROPHET, stressing out the dependencies on data and horizon specifics, which may provide for ARIMA and PROPHET outperforming LSTM on some datasets.

The evaluation is done on 4 top US technology stocks: APPLE (AAPL), META (META), AMAZON (AMZN) and NVIDIA (NVDA). These tickers were chosen on purpose to address the spectrum of volatility and market behavior. NVIDIA and AMAZON are highly volatile with high dynamics dependent on speculations and news. APPLE and META are more stable high liquidity stocks, though, with far from linear behavior and different seasonality. Use of this combination of top stock tickers on common daily frequency creates a testbed for evaluating the linear time-series models (ARIMA-class), decomposable additive (PROPHET) and recurrent neural network (LSTM) cope with short-horizon forecasting in different regimes.

The model estimation and forecast generation followed the modern recommended practice of rolling-origin, expanding window schemes. Historical and actual data were used to evaluate each model. One day ahead forecast was generated based on this history. The received value was evaluated and the estimation value was expanded to include new observation. The repetition of this procedure on training and test periods generated unique out-of-sample forecasts for each of the model. Hyperparameters were tuned via time series cross validation

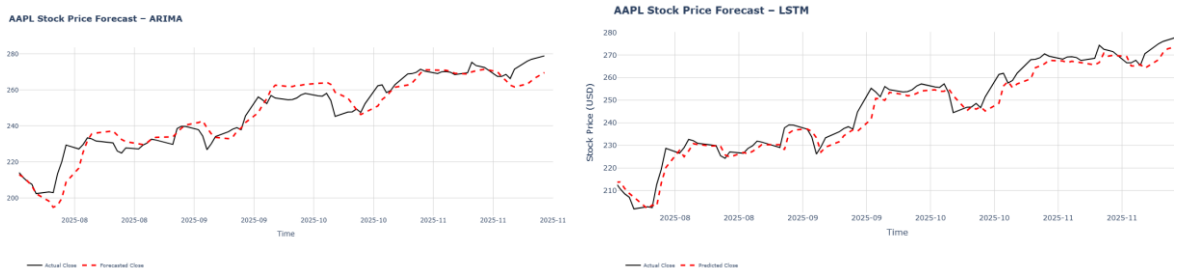
and SARIMA orders were selected from the modest grids based on the out-of-sample performance and combination criteria. PROPHET's key parameters and orders were evaluated based on suggestions of Taylor and Letham (2017). Finally, LSTM parameters (window length, number of units, number of layers, dropout and learning rate) were chosen based on the carefully selected small grids. The proposed approach is consistent with Siami-Namini et al. (2018), emphasizing the importance of tuning for comparison of ARIMA and LSTM models.

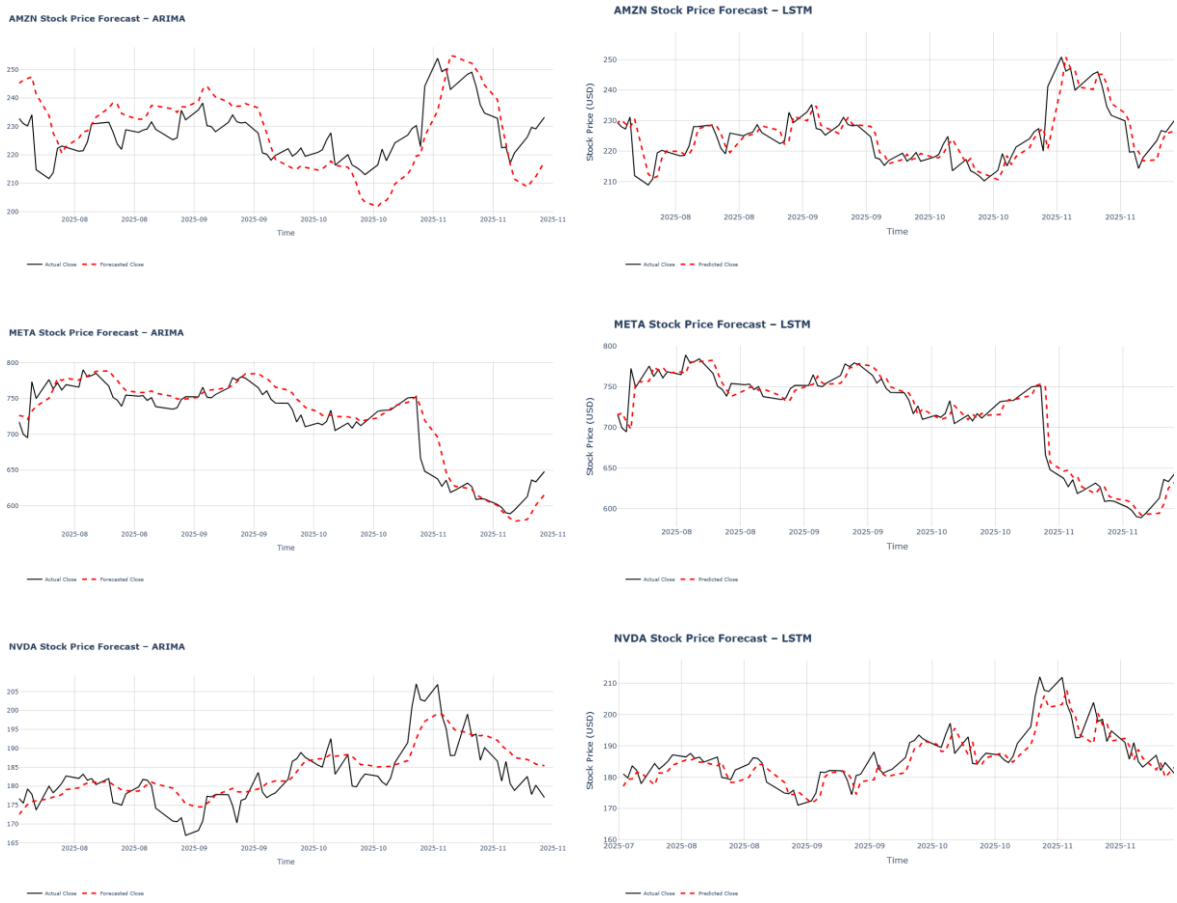
4.1 ARIMA AND LSTM

Expectation of the H1 confirmation was highly biased by multiple comparative studies including. Siami-Namini et al. (2018) reported consistent LSTM forecast error reduction comparing to ARIMA by 80% or more for the broad series of financial and economic series. In research of 5 major indices during 2015-2022, Zhang (2023) got to a similar outcome in the context of index prices predictions with distinct nonlinearity and volatility.

An identical approach to model design and implementation, including rolling windows, careful hyperparameter tuning and the use of the same training and test samples, was adopted to provide a strict test of H1. The findings that LSTM outperformed both ARIMA and SARIMAX for META, AMAZON, NVIDIA and APPLE in terms of RMSE, MAE, R^2, and MAPE were consistent with deep learning advantages reported by Siami-Namini et al. (2018) and Zhang (2023). The choice of evaluation in the specific niche of high-value technological sector rather than a broader index contributed to this list of evidence.

Table 4-1 *ARIMA VS LSTM*





4.2 ARIMA AND PROPHET

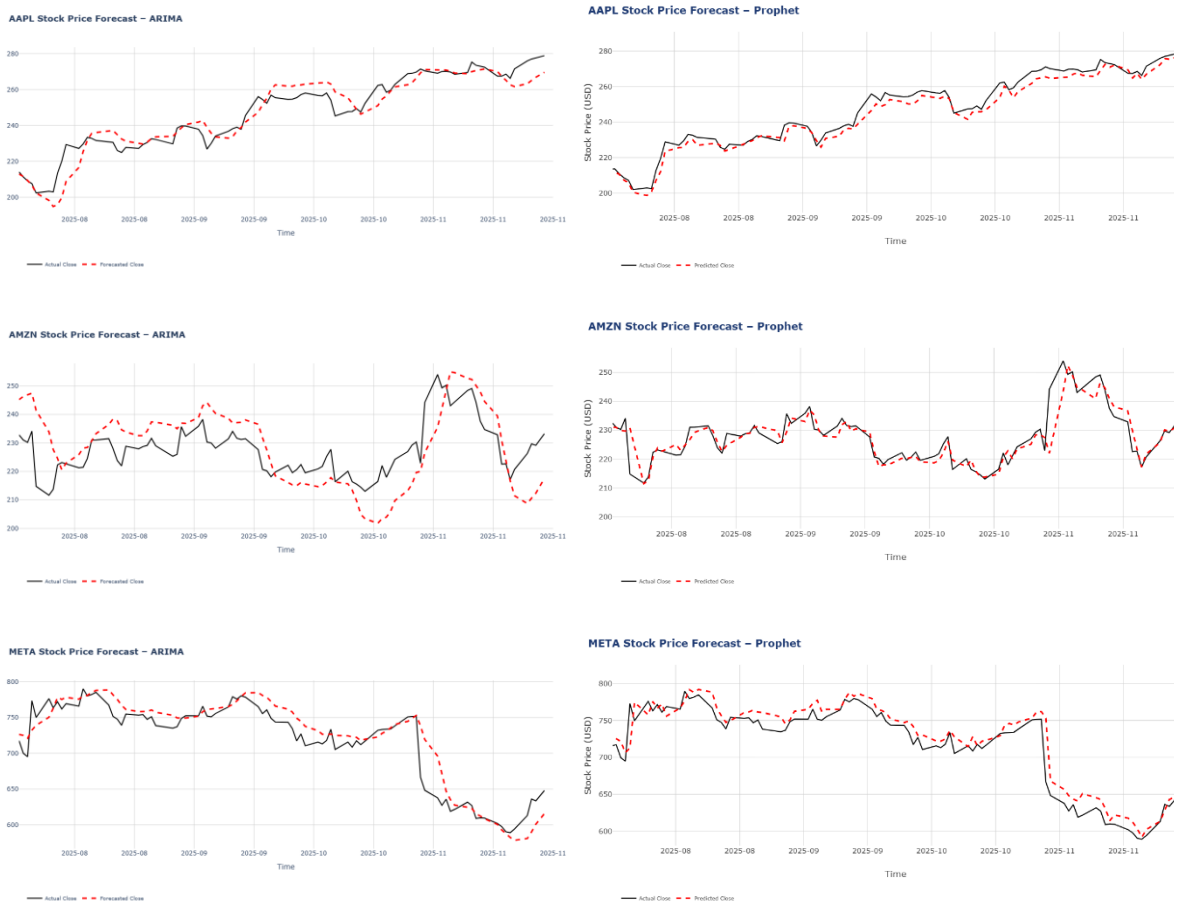
PROPHET designed as a decomposable additive model with explicit trend and seasonal components has proven its effectiveness in cyclic structure domains. Taylor and Letham (2018) reported on PROPHET seniority over classical ARIMA models. Chaturvedi et al. (2022) in comparative assessment of ARIMA, PROPHET and LSTM for Indian monthly energy demand found that ARIMA and LSTM generated higher prediction errors. Chan's (2020) study of Myanmar stock data concluded the applicability of both ARIMA and PROPHET models for the short term daily prediction with PROPHET outperforming ARIMA.

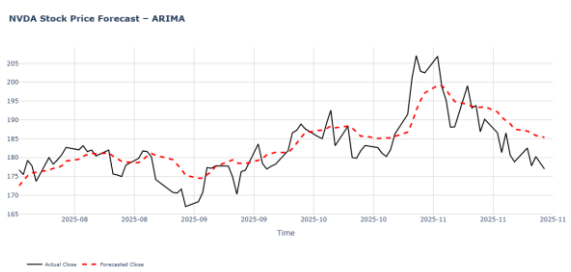
In the considered empirical setting PROPHET's model moving averages captured short-term trend, while the volatility regressors captured medium-horizon risks. Both types of the information are widely used in the financial practice and by incorporating it and as a result, PROPHET addressed the call for interpretable structure and extended variables, which explained the hybrid nature of the model.

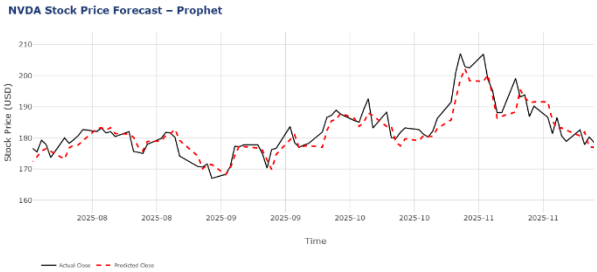
Very close performance of the models were registered for APPLE and META, while PROPHET significantly overperformed ARIMA/SARIMA on much more dynamic and volatile AMAZON and NVIDIA. PROPHET augmented with SMA (10) and volatility regressors consistently outperformed all four tickers. This pattern is in line with Chaturvedi et al. (2022), where PROPHET's advantage becomes clear in relevant calendar and seasonal structures and with Chan's (2020), when models are properly tuned.

As a result, the findings fully confirm H2 hypothesis that PROPHET is superior to ARIMA-class models. It becomes at least competitive with and often exceeds ARIMA-family models when it is enriched with meaningful regressors and well-tuned. This study demonstrated the proof of even basic PROPHET model had been outperforming respective optimized SARIMAX for the chosen US stock market tickers.

Table 4-2 *ARIMA VS PROPHET*







4.3 LSTM, ARIMA AND PROPHET

The majority of research and model competitions in time series next day forecasting consider LSTM more advanced and accurate compared to ARIMA and PROPHET. Though, Sunki et al. (2024) examined ARIMA, LSTM and PROPHET for stock-market forecasting and found that no single model dominated for all stocks and configurations. Suryawan et al. (2024) narrated that their ARIMA (1,0,2) in sales forecasting application outperformed PROPHET and LSTM in MAPE, MSE and RMSE, in spite of the fact that machine learning and particularly deep learning models like LSTM should prevail in that type of tasks.

Menculini et al. (2021) comparative studies found that there were no single leader among ARIMA, PROPHET and LSTM deep learning model for all explored domains and scenarios Similarly, Uzel (2023) in traffic flow studies proved that ARIMA outperformed both PROPHET and LSTM.

Against the portfolio of the chosen tickers of APPLE, META, AMAZON and NVIDIA the empirical results demonstrated consistent ranking:

- LSTM consistently outperformed ARIMA
- PROPHET consistently outperformed ARIMA
- Comparison of LSTM and PROPHET performance on the chosen stocks demonstrated mixed results. LSTM underachieved PROPHET in majority of stock samples but also showed very high potential for improvement and further tuning. That will be one of the recommendations for future research.

The latter ranking may seem to be partially supporting H3. Taking into account that H3 is supposed to cover much broader and general assertion, which assess the model performance across multiple domains and assets, the results rather support the H3 than negate.

The results achieved are in line with Siami-Namini et al. (2018) and Zhang (2023) that reported about highly liquid, high volume, tech stocks, which when properly tuned would be outperforming ARIMA-class and PROPHET models at next day predictions. On the other hand, it only partially varied with the findings and reports of Uzel (2023), Menculini et al. (2021) and Sunki et al. (2024).

5. RESULTS AND DISCUSSION

This chapter examines the results of the modelling and shows the path to improvement for different models. At the final stage the best models' results are compared and the evaluation of the formulated Hypotheses is provided. As the first step the basic setup of each of the models (ARIMA, PROPHET and LSTM) is considered and evaluated against the same class models with further improvement and additional parameters. Then the best models in each class are compared among each other to support or negate the main hypotheses of the research. Graphical values of the results will be provided for better visibility during intra-class assessment. Though this study considered a limited number of tickers for the demonstration of model behavior, all tickers correspond to the strict criteria of being a part of technology sector, high volume trading, among top capitalization companies in the world, traded on the US stock market and as a result have high quality historical data.

5.1 ARIMA-CLASS MODELS RESULTS

Four different models from ARIMA-class were created during this research work and then applied to several chosen tickers of the US stock market.

Table 5.5-1 *ARIMA/SARIMAX Model Performance for AAPL*

	ARIMA-0	SARIMAX-1	SARIMAX-2	SARIMAX-ALL
RMSE	7.771	5.809	5.741	5.448
MAE	7.047	4.668	4.720	4.454
MAPE	2.67%	1.81%	1.79%	1.69%
R^2	0.257	0.585	0.594	0.635

The results of the modelling demonstrate gradual accuracy increase and model improvement across all metrics, which supports the thesis of accuracy improvement with model tuning and parameter optimization. The best results are achieved by SARIMAX model with exogenous variables and proper seasonal configuration further referred as SARIMAX-ALL.

A combination of low MAPE and the highest for this comparison, R^2, indicate strong parameters for financial data series. R^2 improvement across the models has shown more than

a factor of 2. For the other accuracy metrics the improvement between the best and the worst values were in the range of 25% to 32%, demonstrating the meaningful performance gains.

Stability of AAPL even during this turbulent time interval in many ways could contribute to explanation of the received results and the dynamics of improvement.

Table 5.5-2 *ARIMA/SARIMAX Model Performance for META*

	ARIMA-0	SARIMAX-1	SARIMAX-2	SARIMAX-ALL
RMSE	74.911	54.641	21.573	21.911
MAE	56.520	47.792	13.929	14.797
MAPE	8.93%	7.30%	2.08%	2.24%
R^2	0.901	0.115	0.842	0.837

The modelling results for META ticker produced interesting observations. Very low R^2 in SARIMAX-1 demonstrated high residual variances, meaning that the noise has not been properly eliminated, showing that that the model was close to “random walk”. Another observation showed that SARIMAX-ALL, with the highest level of optimization, was outperformed in explanatory power by the less tuned model SARIMAX-2.

Table 5.5-3 *ARIMA/SARIMAX Model Performance for AMZN*

	ARIMA-0	SARIMAX-1	SARIMAX-2	SARIMAX-ALL
RMSE	14.932	9.703	9.585	7.737
MAE	11.476	8.613	7.710	6.614
MAPE	4.80%	3.74%	3.35%	2.85%
R^2	-0.400	0.409	0.423	0.624

Similarly to AAPL, the modelling results for AMZN demonstrated gradual improvement across the implementation of the model improvement. Strong MAPE and R^2 parameter values in line with accurate predictions in both absolute and squared error terms supported the model robustness in ability to detect trend, seasonality and external factors. Negative R^2 value for basic ARIMA model showed worse results than naïve baseline prediction.

Very impressive improvements on the path from basic model to the best performing demonstrated between 41% to 48% key metrics quality increase including significant growth in explanatory power R^2.

Table 5.5-4 *ARIMA/SARIMAX Model Performance for NVDA*

	ARIMA-0	SARIMAX-1	SARIMAX-2	SARIMAX-ALL
RMSE	8.552	7.841	5.896	6.986
MAE	6.207	6.337	4.557	5.728
MAPE	3.16%	3.29%	2.36%	3.01%
R^2	-0.093	0.081	0.481	0.271

NVDA modelling result were quite similar to the ones achieved for AMZN. Being very volatile, it clearly matched the dynamics of the latter and demonstrated that over tuned SARIMAX-ALL model may lose to less optimized, but probably better catching the seasonality SARIMAX-2. All four accuracy metrics demonstrated improvement for over 25% comparing to basic model and leave the room for improvement in better generalization and better choices of exogenous parameters.

As a result of this exercise the best ARIMA-class models were chosen for further comparison with PROPHET and ARIMA models. It is important to note that two models came very close to each other, and each was more accurate in less and more volatile tickers respectively. SARIMAX-ALL showed absolute supremacy in absolute figures, while SARIMAX-2 demonstrated better performance in META and NVDA.

Table 5.5-5 *ARIMA* b*est performing models per metric, per ticker*

	AAPL	META	AMZN	NVDA
RMSE	SARIMAX-ALL	SARIMAX-2	SARIMAX-ALL	SARIMAX-2
MAE	SARIMAX-ALL	SARIMAX-2	SARIMAX-ALL	SARIMAX-2
MAPE	SARIMAX-ALL	SARIMAX-2	SARIMAX-ALL	SARIMAX-2
R^2	SARIMAX-2	SARIMAX-2	SARIMAX-ALL	SARIMAX-2

Table 5.5-6. *Absolute best results for ARIMA class models*

	SARIMAX-ALL
RMSE	5.448
MAE	4.454
MAPE	1,69%
R^2	0.635

5.2 PROPHET MODELS RESULTS

This section explores the evaluation of the results of three Facebook PROPHET models for the same group of US stock market tickers in the similar way as for ARIMA-class models. Consistent approach to modelling and use of similar accuracy metrics explained in detail in the previous thesis chapters provides for weighted comparison.

Three PROPHET models used in this work modelling and evaluation: Basic, Tuned and Features. Basic model is a standard PROPHET forecaster with yearly and weekly additive seasonalities, enhanced by including multiple lagged price features as external regressors. Its purpose is to let PROPHET capture short-term autocorrelation patterns in stock prices that seasonal components alone cannot model. The tuned PROPHET model extends the basic setup by applying the optimal hyperparameters found through a structured grid search. It uses an optimal changepoint prior, a strong holiday prior and multiplicative seasonality, combined with monthly and quarterly seasonal components and all lagged regressors. The final features model represents the most advanced specification in the thesis: PROPHET is combined with a rich set of lagged variables and engineered features (pct-change, SMA, volatility, momentum). This configuration enables model to learn both short-term autoregressive behavior and higher-level market signals.

Table 5.5-7 *PROPHET Model Performance for AAPL*

	Basic	Tuned	Features
RMSE	4.030	3.540	2.720
MAE	3.150	2.420	1.900
MAPE	1.18%	0.91%	0.71%
R^2	0.641	0.723	0.887

Despite strong performance of basic PROPHET model, it has demonstrated high potential for result improvement, which was common for all tickers researched in this work. All metrics demonstrated consistent improvement with model enhancement, resulting in around 40% increase.

Table 5.5-8 *PROPHET Model Performance for META*

	Basic	Tuned	Features
RMSE	20.420	20.310	17.220
MAE	12.750	11.830	9.410
MAPE	2.00%	1.82%	1.46%
R^2	0.869	0.871	0.907

The model demonstrated relatively low forecast precision (RMSE) compared to other stocks considered. That may be explained by high volatility of the ticker and becomes an outlier for PROPHET models, created in the study. Nevertheless, all other metrics demonstrated strong result and relative potential for improvement, which can be attributed to adding features rather than tuning alone.

Table 5.5-9 *PROPHET Model Performance for AMZN*

	Basic	Tuned	Features
RMSE	5.790	5.630	5.330
MAE	4.180	3.950	3.080
MAPE	1.79%	1.69%	1.30%
R^2	0.769	0.781	0.804

AMZN, being a stock known for both high growth, high trading and volatility, enjoyed the strong modeling result of the basic model. Tuning of the seasonality and changepoints provided certain but not dramatic improvement, while exogenous factors demonstrated a change.

Table 5.5-10 *PROPHET Model Performance for NVDA*

	Basic	**Tuned**	**Features**
RMSE	5.000	5.060	4.030
MAE	4.130	4.200	3.200
MAPE	2.16%	2.19%	1.68%
R^2	0.667	0.660	0.784

The results for NVDA demonstrated a rare case for PROPHET family modelling in this work, when the enhanced model didn't provide metric improvement. We attribute it to overfitting or suboptimal tuning, which was consistent for all stocks explored in this work.

All PROPHET models demonstrated very consistent R^2, never falling lower than 0.59, which was a significant differentiation with all other models of ARIMA family and LSTM group models. MAPE achieved good to exceptional values for all considered tickers and all created models. The value remained within 0.71% to 2.63% interval, which is considered very strong achievement. Compared to the peer group of algorithms across the same stock market companies, every created PROPHET model achieved more stable results with higher accuracy than counterparts.

Table 5.5-11 PROPHET b*est performing models per metric, per ticker*

	AAPL	**META**	**AMZN**	**NVDA**
RMSE	Features	Features	Features	Features
MAE	Features	Features	Features	Features
MAPE	Features	Features	Features	Features
R^2	Features	Features	Features	Features

Table 5.5-12 *Absolute best results for PROPHET class models*

	SARIMAX-ALL
RMSE	2.270
MAE	1.900
MAPE	0,71%
R^2	0.887

5.3 LSTM MODELS RESULTS

This section reviews the results of four LSTM specifications: LSTM-1, a one-layer baseline model using price and volume features, IMP-1, which extends LSTM-1 with two additional indicators (relative price to the 60-day moving average and the 60-day z-score), IMP-2, which builds on LSTM-2 and includes light hyperparameter tuning of the main model parameters and LSTM-4, a deeper four-layer LSTM designed to test whether a more complex architecture can improve forecasting performance.

Table 5.5-13 *LSTM Model Performance for AAPL*

	LSTM-1	IMP-1	IMP-2	LSTM-4
RMSE	4.313	4.261	4.169	5.793
MAE	3.119	3.137	3.099	4.642
MAPE	1.28%	1.28%	1.26%	1.88%
R^2	0.955	0.956	0.958	0.918

All LSTM models demonstrated a very high level of performance. The results indicate that the four-layer LSTM model is excessively complex and likely suffers from overfitting. Since four-layer model requires much deeper fine tuning, improvements over one-layer models are easier achieved that we will see through the analysis of the LSTM results.

Table 5.5-14 *LSTM Model Performance for META*

	LSTM-1	IMP-1	IMP-2	LSTM-4
RMSE	15.754	15.998	15.916	20.956
MAE	9.718	10.092	9.982	16.197
MAPE	1.38%	1.43%	1.42%	2.26%
R^2	0.925	0.923	0.924	0.868

Similar to the previous case, design inefficiency for four-layer model is clearly seen. On contrary to the previous stock the simplest one-layer model demonstrated best performance, which may be attributed to specifics of the ticker. Very high level of explainability R^2 for the whole series of LSTM sets out these two stock.

Table 5.5-15 *LSTM Model Performance for AMZN*

	LSTM-1	IMP-1	IMP-2	LSTM-4
RMSE	4.864	4.901	4.903	5.218
MAE	3.522	3.522	3.453	3.96127
MAPE	1.51%	1.56%	1.53%	1.76%
R^2	0.705	0.693	0.693	0.661

Consistently with previous results, simpler models outperformed complexity. Interestingly, the basic and simplest LSTM one-layer model surpassed both enhanced models. Model to model improvements had relatively low impact, which is in correlation with earlier statement about overall deep learning model complexity and flexibility.

Table 5.5-16 *LSTM Model Performance for NVDA*

	LSTM-1	IMP-1	IMP-2	LSTM-4
RMSE	4.527	5.231	4.397	27.120
MAE	3.515	4.225	3.477	25.834
MAPE	1.86%	2.23%	1.84%	13.65%
R^2	0.711	0.614	0.727	-9.375

LSTM-4 results are complete outliers in this modelling, which is consistent with previous stocks. Besides the strong performance of all other models, almost similar performance of basic LSTM-1 and the most advanced Imp-2 deserves pointing out. This important observation confirms some of the findings made by researchers of time series forecasting, particularly that sometimes “less can be more”, which relates to better tuning potential of lower and better researched hyperparameters. Important general observation for researched LSTM models showed steady but much lower model to model improvements, where detected, which proves high potential for use of variety of parameter tuning and external factors.

This highlights that deeper research and broader considerations should be taken into account during the design phase to achieve a high-performing model.

5.4 ARIMA vs LSTM (H1)

This section looks at numerical results supporting the first hypothesis of this work. Each of the four tickers will be compared among the best performing models from ARIMA and LSTM classes respectively.

Table 5.5-17 *H1: AAPL & AMZN*

AAPL	SARIMAX-ALL	IMP-2	**AMZN**	SARIMAX-ALL	LSTM-1
RMSE	5.448	4.169	**RMSE**	7.737	4.864
MAE	4.454	3.099	**MAE**	6.614	3.522
MAPE	1.69%	1.26%	**MAPE**	2.85%	1.51%
R^2	0.635	0.958	**R^2**	0.624	0.705

For better visibility the results were grouped by pairs. Best SARIMAX-ALL model were outperformed by simplest LSTM-1 and the most featured IMP-2. Though, SARIMAX-ALL demonstrated a solid performance and build a decent baseline. The improvement for AAPL is in the range of around 21% - 23%, while for AMZN it grows up to approximately 37% - 48% range.

Table 5.5-18 *H1: NVDA & META*

NVDA	SARIMAX-2	IMP-2	**META**	SARIMAX-2	LSTM-1
RMSE	5.896	4.397	**RMSE**	21.573	15.754
MAE	4.557	3.477	**MAE**	13.929	9.718
MAPE	2.36%	1.84%	**MAPE**	2.08%	1.38%
R^2	0.481	0.727	**R^2**	0.842	0.925

The result of comparing the remaining two pairs of high growth stock reenforce LSTM advantage over ARIMA class models. The outperformance clearly visible on high volatile NVDA, where price is sensitive to external events and as a result - highly volatile. For more stable META ARIMAX-2 demonstrates very solid performance, though still concede.

The principle difference between the ARIMA and LSTM allows the latter to better catch nonlinear trends and sequential dependencies, which based on the research and modelling result fully support the first hypothesis H1 of this work.

5.5 ARIMA vs PROPHET(H2)

Comparison of results of ARIMA and PROPHET class of models demonstrated absolute superiority of the latter. Despite strong performance of ARIMA family algorithms and the results improvements it showed inside of their class, it failed to beat any of the four metric parameters of the top PROPHET Features model.

Table 5.5-19 *H2: AAPL & AMZN*

AAPL	SARIMAX-ALL	Features	**AMZN**	SARIMAX-ALL	Features
RMSE	5.448	2.720	**RMSE**	7.737	5.330
MAE	4.454	1.900	**MAE**	6.614	3.080
MAPE	1.69%	0.71%	**MAPE**	2.85%	1.30%
R^2	0.635	0.887	**R^2**	0.624	0.804

Achieved results showed consistent PROPHET outperforming ARIMA. For both AAPL and AMZN PROPHET model demonstrated around 50% improvement comparing to ARIMA. AMZN RMSE remained the outlier displaying “only” 30%.

Table 5.5-20 *H2: NVDA & META*

NVDA	SARIMAX-2	Features	**META**	SARIMAX-2	Features
RMSE	5.896	4.030	**RMSE**	21.573	17.220
MAE	4.557	3.200	**MAE**	13.929	9.410
MAPE	2.36%	1.68%	**MAPE**	2.08%	1.46%
R^2	0.481	0.784	**R^2**	0.842	0.907

Very consistent performance was detected for the pair of NVDA and META tickers in ARIMA and PROPHET data series modelling. The latter produced close to 30% improvement except for majority of accuracy comparison metrics. The only exception became META RSME, which was improved by around 20%. Research results support second hypothesis about PROPHET superiority (H2) over ARIMA. Interestingly, even the basic PROPHET models demonstrated better performance than the advanced ARIMAX-ALL, which also demonstrated solid performance across the researched samples of US stock market tickers.

5.6 LSTM vs ARIMA vs PROPHET(H3)

After analysis of comparison of ARIMA versus LSTM and ARIMA versus PROPHET considered in the previous sections this section will concentrate mostly on LSTM vs PROPHET. To support the argument all three PROPHET models will be added for the visual judgement.

Table 5.6-21 AAPL All Models Comparison

AAPL	Arima-Class	PROPHET			LSTM
	SARIMAX -2	Basic	Tuned	Features	Imp-2
RMSE	5.448	4.030	3.540	2.270	4.169
MAE	4.454	3.150	2.420	1.900	3.100
MAPE	1.69%	1.18%	0.91%	0.71%	1.26%
R^2	0.635	0.641	0.723	0.887	0.958

Observation of the results demonstrates significant supremacy of LSTM IMP-2 model with a fit of almost 0.96 value. The other metrics of PROPHET showed better results including estimated and actual value percentage, better consistency of prediction and lower absolute

error. Notably, that LSTM model managed to beat consistency prediction only for the basic PROPHET model in this research.

Table 5.6-22 *META All Models Comparison*

META	Arima-Class	PROPHET			LSTM
	SARIMAX -2	Basic	Tuned	Features	LSTM-1
RMSE	10.701	20.420	20.310	17.220	15.754
MAE	8.988	12.750	11.830	9.4100	9.718
MAPE	2.05%	2.00%	1.82%	1.46%	1.38%
R^2	0.620	0.869	0.871	0.907	0.925

For this ticker despite full dominance of PROPHET and LSTM models SARIMAX-ALL demonstrated very strong performance in RMSE and MAE and surpassed the other models. On the other hand, LSTM model, which turned out to be the most basic one outperformed PROPHET models in RMSE, MAPE and R^2, slightly missing on MAE to the most advanced PROPHET model. This ticker became a visible example of mixed performance of the compared models in different metrics. Depending on the priorities of the model architects it may provide valuable advice on choosing the right methods of prediction modelling.

Table 5.6-23 *AMZN All Models Comparison*

AMZN	Arima-Class	PROPHET			LSTM
	SARIMAX -2	Basic	Tuned	Features	LSTM-1
RMSE	7.737	5.790	5.630	5.330	4.864
MAE	6.614	4.180	3.950	3.080	3.384
MAPE	2.85%	1.79%	1.69%	1.30%	1.51%
R^2	0.624	0.769	0.781	0.804	0.705

For AMAZON time series modelling PROPHET model Features demonstrated the top performance among others. It dominated in MAE, MAPE and R^2, missing only on RMSE to LSTM basic model. It is worth stressing out strong performance of LSTM-1 versus both lower-level PROPHET models, which lost to LSTM in three metric parameters for this ticker.

Table 5.6-24 *NVDA All Models Comparison*

NVDA	Arima-Class	PROPHET			LSTM
	SARIMAX -2	Basic	Tuned	Features	Imp-2
RMSE	5.896	5.000	5.060	4.030	4.397
MAE	4.557	4.130	4.200	3.200	3.477
MAPE	2.36%	2.16%	2.19%	1.68%	1.84%
R^2	0.481	0.667	0.660	0.784	0.727

Unexpectedly for high volatile ticker, PROPHET model had dominated over the others. Though demonstrating lower explanatory power R^2 than in for the other tickers from the research sample, it was sufficient to surpass the results of the best LSTM IMP-2 model. In comparison with the basic and tuned PROPHET models LSTM consistently outperformed. As it was stated early with the proper calculate resource and deeper tuning LSTM has significant improvement resource.

Summarizing the analyses of ARIMA, PROPHET and LSTM classes of models results, no single leader was detected. That corresponds in full to the third hypothesis (H3) of this work. Surprisingly, PROPHET model demonstrated better results than LSTM, which is attributed to higher simplicity of the modelling and limitations of the compute power to produce more variances for LSTM experiments. Even under this restraints LSTM exhibited strong performance in accuracy and model fit across the sampled US stock market high-tech tickers.

6. CONCLUSIONS AND FUTURE WORKS

This thesis used the Python simulation frameworks to create working models for next day stock price prediction. The modelling comprised four ARIMA class algorithms, three Facebook PROPHET and four LSTM models. Four prominent US stock market tickers data series were used for the research and experimental work. Three hypotheses explored the deemed effectiveness of selected algorithmic model classes across chosen group of US stock market tickers. ARIMA-class, PROPHET and LSTM and its performance for real life US stock market were to be empirically and practically assessed.

The accomplished work sustained the hypotheses H1 and H2 on LSTM and hybrid PROPHET models supremacy over ARIMA class frameworks and confirmed their consistent outperformance across all accuracy parameters. Despite strong performance of ARIMA family models, particularly SARIMAX, it was outperformed by all range of Facebook PROPHET models including the basic one. The final hypothesis H3 has also been confirmed. The results of the modelling explicitly showed that there is no one single best model which can be recommended for any context. Even relatively scarce, but significant, top US stock market tickers' sampling showed deviating results both inside the considered model classes and outside it.

The limitations of this study concerned with the constrained availability of the compute power of the researcher, which restricted the capability of much wider LSTM simulations. Other limitations relate to intentional restriction of this work to top selected market tickers, statistical, hybrid and artificial intelligence machine learning models and the time frame used for learning, verification and testing of the created models. Selection of variables' values, hyperparameters and other factors, directly impacting the model performance were initially derived from fundamental and practical research sources and then either retained as recommended or adjusted based on the completed experimental work.

The practical value of this study is associated with meaningful advice for individual investors and market practitioners willing to create their own time series models for next day forecasting, which require low compute power, acquaintance with Python modelling and resilience to create separate models for seemingly comparable market tickers. From educational and academic perspective, it encourages further research of new machine learning and deep learning methods within the context of the researched fields.

Though researched LSTM model had not shown absolute supremacy to ARIMA class and hybrid PROPHET models as expected, it had shown the capability to continuously improve depending on the integration and tuning of the external factors and hyperparameters. Broader research and experiments in this field will be subject of the future value.

BIBLIOGRAPHICAL REFERENCES

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control, 19*(6), 716–723. https://doi.org/10.1109/TAC.1974.1100705

Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling.

Barberis, N., & Thaler, R. (2003). A survey of behavioral finance. In G. M. Constantinides, M. Harris, & R. M. Stulz (Eds.), *Handbook of the economics of finance* (Vol. 1B, pp. 1053–1128). Elsevier.

Bogle, J. C. (2007). *The little book of common sense investing: The only way to guarantee your fair share of stock market returns.* John Wiley & Sons.

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. (2008). *Time series analysis: Forecasting and control* (4th ed.). Wiley.

Brockwell, P. J., & Davis, R. A. (2016). *Introduction to time series and forecasting* (3rd ed.). Springer.

Casella, G., & Berger, R. L. (2002). *Statistical inference* (2nd ed.). Duxbury/Thomson Learning.

Chaturvedi, S., Rajasekar, E., Natarajan, S., & McCullen, N. (2022). A comparative assessment of SARIMA, LSTM RNN and Fb PROPHET models to forecast total and peak monthly energy demand for India. *Energy Policy, 168*, 113097. https://doi.org/10.1016/j.enpol.2022.113097

Cheng, J., Tiwari, S., Khaled, D., Mahendru, M., & Shahzad, U. (2024). Forecasting Bitcoin prices using artificial intelligence: Combination of ML, SARIMA, and Facebook PROPHET models. *Technological Forecasting and Social Change, 198*, 122938. https://doi.org/10.1016/j.techfore.2023.122938

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 Bondt, W. F. M., & Thaler, R. (1985). Does the stock market overreact? *The Journal of Finance, 40*(3), 793–805. https://doi.org/10.1111/j.1540-6261.1985.tb05004.x

Deloitte. (2023). Algorithmic forecasting in a digital world: Improving the forecasting process with predictive analytics. *Deloitte Insights.*

Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance, 25*(2), 383–417. https://doi.org/10.1111/j.1540-6261.1970.tb00518.x

Fama, E. F. (1991). Efficient capital markets: II. *The Journal of Finance, 46*(5), 1575–1617. https://doi.org/10.1111/j.1540-6261.1991.tb04636.x

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

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning.* MIT Press.

Graham, B. (2006). *The intelligent investor* (Rev. ed.). HarperBusiness.

Graham, B., & Dodd, D. L. (2008). *Security analysis* (6th ed.). McGraw-Hill.

Graves, A., Mohamed, A.-r., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. *ICASSP Proceedings.*

Guégan, D. (2009). Chaos in economics and finance. *Annual Reviews in Control, 33*(2), 89–93.

Hamilton, J. D. (1994). *Time series analysis.* Princeton University Press.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: Principles and practice* (2nd ed.). OTexts.

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting, 22*(4), 679–688.

Lo, A. W. (2012). Adaptive markets and the new world order. *Financial Analysts Journal, 68*(2), 18–29.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). Statistical and machine learning forecasting methods. *PLoS ONE, 13*(3), e0194889.

McKinney, W. (2022). *Python for data analysis* (3rd ed.). O'Reilly.

McLaney, E. (2017). *Business finance: Theory and practice* (11th ed.). Pearson.

Menculini, L., et al. (2021). Comparing PROPHET and deep learning to ARIMA. *Forecasting, 3*(3), 644–662.

Murphy, J. J. (1999). *Technical analysis of the financial markets.* New York Institute of Finance.

Nelson, D. M. Q., Pereira, A. C. M., & de Oliveira, R. A. (2017). Stock market prediction with LSTM. *IJCNN Proceedings.*

Peixeiro, M. (2023). *Time series forecasting in Python.* Manning / O'Reilly.

Raschka, S., Patterson, J., & Nolet, C. (2020). Machine learning in Python. *Information, 11*(4), 193.

Shiller, R. J. (2003). From efficient markets to behavioral finance. *Journal of Economic Perspectives, 17*(1), 83–104.

Shumway, R. H., & Stoffer, D. S. (2017). *Time series analysis and its applications* (4th ed.). Springer.

Statista. (2024). AI in fintech – market size forecast worldwide.

Sunki, A., et al. (2024). Stock market forecasting using ARIMA, LSTM and PROPHET. *MATEC Web of Conferences, 392*, 01163.

Suryawan, I. G. T., et al. (2024). Performance comparison of ARIMA, LSTM, and PROPHET. *Sinkron, 8*(4), 2410–2421.

Taylor, S. J., & Letham, B. (2018). Forecasting at scale. *The American Statistician, 72*(1), 37–45.

Van der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). The NumPy array. *Computing in Science & Engineering, 13*(2), 22–30.

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa