Contents

[Abstract 4](#_Toc140758680)

[Chapter 1 5](#_Toc140758681)

[Introduction 5](#_Toc140758682)

[1.1 Time Series 5](#_Toc140758683)

[1.2 Characteristics of the time series 5](#_Toc140758684)

[1.3 Properties of Time series 5](#_Toc140758685)

[1.3.1 Stochastic process 5](#_Toc140758686)

[1.3.2 Stationarity 6](#_Toc140758687)

[1.3.3 Autoregression 6](#_Toc140758688)

[1.3.4 Moving Average 6](#_Toc140758689)

[1.3.5 Importance: 7](#_Toc140758690)

[1.4 Application of Time Series Analysis in Finance 8](#_Toc140758691)

[1.5 Challenges in Time Series Analysis for Financial Data 10](#_Toc140758692)

[1.6 Deep Learning Algorithms for Time Series Analysis 11](#_Toc140758693)

[1.7 Introduction to Deep Learning 12](#_Toc140758694)

[1.8 Relevance of Deep Learning in Financial Data Analysis 14](#_Toc140758695)

[1.8.1 Predictive modeling 14](#_Toc140758696)

[1.8.2 Risk Assessment: 14](#_Toc140758697)

[1.8.3 Portfolio Optimization: 14](#_Toc140758698)

[1.8.4 Algorithmic Trading: 14](#_Toc140758699)

[1.8.5 Natural Language Processing (NLP) for Sentiment Analysis: 14](#_Toc140758700)

[1.8.6 Fraud Detection: 14](#_Toc140758701)

[1.8.7 Customer Behavior Analysis: 15](#_Toc140758702)

[1.8.8 High- High-Frequency Trading: Frequency Trading: 15](#_Toc140758703)

[1.9 Long Short-Term Memory (LSTM) Networks 15](#_Toc140758704)

[1.9.1 Forget Gate: 15](#_Toc140758705)

[1.9.2 Input Gate: 16](#_Toc140758706)

[1.9.3 Output Gate: 16](#_Toc140758707)

[1.10 LSTM cell structure 17](#_Toc140758708)

[1.11 Comparative Analysis of Deep Learning Algorithms 18](#_Toc140758709)

[1.11 1 Convolutional Neural Networks (CNNs): 18](#_Toc140758710)

[1.11 2 Recurrent Neural Networks (RNNs): 19](#_Toc140758711)

[1.11 3 Long Short-Term Memory (LSTM) Networks: 20](#_Toc140758712)

[1.11 4 Generative Adversarial Networks (GANs): 20](#_Toc140758713)

[1.11 5 Transformer Networks: 21](#_Toc140758714)

[Chapter 2 22](#_Toc140758715)

[Review of literature 22](#_Toc140758716)

[Chapter 3 41](#_Toc140758717)

[Methodology 41](#_Toc140758718)

[3.1 Experimental Setup 41](#_Toc140758719)

[3.1.1 Model Selection: 42](#_Toc140758720)

[3.2 Designing of the Models 43](#_Toc140758721)

[3.3 Database Preparation 44](#_Toc140758722)

[Chapter 4 49](#_Toc140758723)

[4.1 Results and Analysis/Comparison 49](#_Toc140758724)

[Table 1 CNN-LSTM-RNN Model Results 49](#_Toc140758725)

[4.2 Analysis 50](#_Toc140758726)

[Figure 12 LSTM model close price prediction 51](#_Toc140758727)

[Figure 13 LSTM model high price prediction 51](#_Toc140758728)

[Table 2 Results of LSTM proposed methodology 52](#_Toc140758729)

[4.3 Analysis 52](#_Toc140758730)

[Figure 14 CNN model close price prediction 53](#_Toc140758731)

[Figure 15 CNN model high price prediction 53](#_Toc140758732)

[Table 3 Results of CNN proposed methodology 54](#_Toc140758733)

[4.4 Analysis 54](#_Toc140758734)

[Figure 16 RNN model close price prediction 55](#_Toc140758735)

[Figure 17 RNN model high price prediction 55](#_Toc140758736)

[Table 4 Results of RNN proposed methodology 56](#_Toc140758737)

[4.5 Analysis 56](#_Toc140758738)

[Figure 18 CNN-LSTM model close price prediction 57](#_Toc140758739)

[Figure 19 CNN-LSTM model high price prediction 58](#_Toc140758740)

[4.6 Evaluation Parameter 58](#_Toc140758741)

[Figure 20 MAE comparison between various models 58](#_Toc140758742)

[Figure 21 RMSE comparison of multiple different models 59](#_Toc140758743)

[Figure 22 Direction Accuracy comparison of multiple different models 59](#_Toc140758744)

[Chapter 5 Conclusions 60](#_Toc140758745)

[Reference 61](#_Toc140758746)

### Abstract

This study aims to evaluate the performance of three deep learning algorithms, namely Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN), and Recurrent Neural Network (RNN), for time series analysis in financial data. The dataset consists of historical stock price data for four major companies: AAPL, GOOGL, MSFT, and AMZN. The data is obtained from the Yahoo Finance API and covers the period from January 2018 to December 2022. The evaluation is based on several metrics including mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), and directional accuracy (DA). The models are trained using a portion of the data and tested on the remaining portion. The predictions are then compared with the actual stock prices to calculate the evaluation metrics. The results show that all three models have relatively high MSE and RMSE values, indicating some level of error in their predictions. The MAE values suggest a significant average absolute difference between the predicted and actual stock prices for all models. The MAPE values indicate an average percentage difference of around 100%, suggesting moderate accuracy in predicting stock prices. The DA values show that the RNN model performs slightly better in correctly predicting the direction of price movement compared to the LSTM and CNN models.

### Chapter 1

### Introduction

### 1.1 Time Series

Data collected over a series of time points or over some time is called a time series. Examples of time series include the start of new housing projects each month and the sale of products every week. In a time series, data are typically collected at equal intervals of time, such as hourly, daily, weekly, monthly, or yearly. The ultimate purpose of time series analysis is to develop forecasts for future values of the series.

### 1.2 Characteristics of the time series

In stochastic processes, we can view *Yt* as a sequence of random variables. This process represents how we observe time series. According to the joint distribution of *Yt*, this process has a complete probability structure. In this joint distribution, the majority of the information is represented by the mean, variance, and covariance. The main characteristic quantities are shown below:

Mean: *µt* = *EYt* —> Expected (*Yt*)

Variance: *DYt* = *E* (*Yt* ´ *µt*) 2

Covariance: *Cov*(*Yt*, *Ys*) = *E*(*Yt* ´ *µt*)(*Ys* ´ *µs*)

Autocorrelation: *ρt*,*s* = ?*Cov*(*Yt* ,*Ys* ) = ? *γt*,*s*

### 1.3 Properties of Time series

## 1.3.1 Stochastic process

Stochastic processes are defined as collections of randomly arranged variables over time. Probabilistic laws determine how stochastic processes evolve. In this thesis, we observe that the stochastic process is responsible for the data points that comprise the stock index. Hence, the time series is a sample of the random variables within the stochastic process (Cryer and Chan, 2008; Box et al., 2016).

## 1.3.2 Stationarity

A time series is stationary if the statistical properties of the process generating it do not change over time. In other words, it does not mean the series does not change over time, just that its method of change is not itself changing over time. It is therefore a linear function, not a constant one, in algebra the value of a linear function changes as y increases, but the amount it changes remains constant – it has a constant slope; one value that captures that rate of change.

## 1.3.3 Autoregression

Autoregressive (*AR*) statistics is another widely used statistical process. Its worth at this time depends on previous time steps as well as a random shock term. In contrast to the MA process, the *AR* is not always stationary (Box et al., 2016). A general *AR* process is denoted *AR(p)*, where *p* is the order of the process. A general *AR* method can be written here:

*Yt* = *c* + *φ*1*Yt*´1 + *φ*2*Yt*´2 + ... + *φpYt*´*p* + *et* (3.1)

At some arbitrary time *t*, *Yt* represents the process value, *et* is the error term at time *t*, and *φ*0 to *φp* are the parameters. So, the *AR(p)* value depends on how the process has evolved over the *p* time steps since the previous time step, and how the error term has changed. A process of order one is abbreviated *AR(1)*, whose value is based on the value at the time step before, and the shock or error term today. *AR(2)* values depend on the value at the two previous time steps as well as today’s shock (Cryer and Chan, 2008; Box et al., 2016).

## 1.3.4 Moving Average

An important characteristic of the moving average *(MA)* process in time series analysis is that it is always stationary. The *MA* process is one of the most common stochastic methods used in time series analysis. Its value varies according to the current and previous value of the shock term. The general *MA* process is denoted *MA(q)*, where *q* stands for the order of process. the process can be written as,

*Yt* = *c* + *et* + *θ*1*et*´1 + *θ*2*et*´2 + ... + *θqet*´*q* (3.2)

At any given point in time *t*, *Yt* is the values of the process,*t*, *et*, and *et*´*q* are the random shock term and *θ*1 to *θq* are the parameters. This means that the value of an *MA* at time *t* is determined by the shock term at time *t*, as well as all the shock terms *q* time steps back. For an *MA(1)*, the process value depends on the shock term in the current time and the shock term in the previous time frame. For *MA (2)* the process value is based on the shock term in the present and the previous time frames (Cryer and Chan, 2008; Box et al., 2016).

Time series analysis is a statistical method used to analyze and interpret data points collected over successive time intervals. It involves examining the temporal order and dependencies of the data to uncover patterns, trends, and underlying relationships. Time series analysis focuses on understanding how data points change over time and uses this information to make predictions or derive meaningful insights.

## 1.3.5 Importance:

Time series analysis holds significant importance in various fields, including finance, economics, meteorology, signal processing, and more. In finance specifically, time series analysis plays a crucial role due to the following reasons:

1. Pattern Identification: Time series analysis helps in identifying patterns and trends in financial data. It allows researchers and analysts to uncover recurring patterns, seasonal fluctuations, and long-term trends that can impact financial markets, asset prices, and economic indicators.
2. Forecasting: By analyzing historical time series data, financial analysts can develop forecasting models to predict future values and trends. These forecasts aid in making informed decisions regarding investments, portfolio management, risk assessment, and financial planning.
3. Risk Management: Time series analysis provides valuable insights into the volatility and risk associated with financial assets. By studying the patterns and behavior of asset prices over time, risk managers can assess and mitigate potential risks, develop hedging strategies, and optimize portfolio allocations.
4. Decision-Making: Time series analysis offers a data-driven approach to decision-making in finance. It helps in identifying turning points, anomalies, and outliers in financial data, enabling timely and informed decision-making for traders, investors, and financial institutions.
5. Policy Formulation: Governments and regulatory bodies rely on time series analysis to monitor economic indicators, inflation rates, employment data, and other key metrics. These analyses guide policy formulation, economic planning, and implementation of financial regulations.
6. Algorithmic Trading: With the rise of automated and algorithmic trading, time series analysis has become essential for developing robust trading strategies. By analyzing historical data and identifying patterns, algorithms can make data-driven trading decisions based on past market behavior.

In summary, time series analysis is a powerful tool for understanding and interpreting financial data. It aids in identifying patterns, forecasting future values, managing risks, and supporting decision-making processes. Its importance lies in providing valuable insights into market dynamics, enabling more informed and strategic actions in the financial domain.

### 1.4 Application of Time Series Analysis in Finance

Time series analysis plays a fundamental role in finance and has a wide range of applications. It enables researchers, analysts, and practitioners to gain valuable insights into financial data, predict future trends, and make informed decisions. Here are some key applications of time series analysis in finance:

1. Stock Market Analysis: Time series analysis is extensively used to analyze and predict stock market behavior. By studying historical price and volume data, analysts can identify patterns, trends, and seasonality in stock prices. This information helps in understanding market dynamics, identifying trading opportunities, and developing investment strategies.
2. Risk Management: Time series analysis is crucial for assessing and managing financial risks. It allows for the modeling and forecasting of volatility, identifying extreme events or outliers, and estimating Value at Risk (VaR) and Expected Shortfall (ES). These risk measures aid in portfolio optimization, hedging strategies, and overall risk management in financial institutions.
3. Asset Price Forecasting: Time series analysis is employed to forecast future prices of financial assets such as stocks, bonds, commodities, and currencies. By using various statistical and machine learning models, analysts can make predictions based on historical data, helping investors and traders make informed decisions regarding buying, selling, or holding assets.
4. Economic Indicators: Time series analysis is widely used to analyze economic indicators such as GDP, inflation rates, employment data, and consumer spending. By analyzing historical time series data, economists can identify trends, business cycles, and relationships between different economic variables. This information assists policymakers, central banks, and investors in understanding the overall health of the economy and formulating appropriate strategies.
5. Algorithmic Trading: Time series analysis is a key component of algorithmic trading strategies. Traders and financial institutions use historical market data to identify patterns and develop trading algorithms that can automatically execute trades based on predefined rules. Time series analysis helps in developing models for trend detection, momentum trading, mean reversion, and other trading strategies.
6. Credit Risk Assessment: Time series analysis is used in credit risk assessment to evaluate the creditworthiness of borrowers and predict default probabilities. By analyzing historical data on borrower behavior, payment patterns, and economic factors, financial institutions can estimate the likelihood of default and assign appropriate credit ratings.
7. Portfolio Management: Time series analysis assists in portfolio optimization and asset allocation. By studying the correlation, volatility, and performance of different assets over time, analysts can construct efficient portfolios that balance risk and return. Time series models are used to estimate expected returns, covariance matrices, and other parameters necessary for portfolio optimization.
8. Financial Forecasting: Time series analysis is employed to forecast financial variables such as sales, revenue, cash flows, and interest rates. This information aids in budgeting, financial planning, and making strategic business decisions.

In summary, time series analysis finds diverse applications in finance, including stock market analysis, risk management, asset price forecasting, economic indicators analysis, algorithmic trading, credit risk assessment, portfolio management, and financial forecasting. Its versatility and ability to extract meaningful insights from historical data make it an indispensable tool for understanding and navigating the complex world of finance.

### 1.5 Challenges in Time Series Analysis for Financial Data

Time series analysis in the context of financial data poses specific challenges due to the complex and dynamic nature of financial markets. These challenges need to be addressed to ensure accurate and reliable analysis. Here are some key challenges faced in time series analysis for financial data:

1. Volatility and Nonlinearity: Financial data often exhibits high volatility and nonlinear relationships. Market conditions, economic events, and investor sentiments can cause abrupt and unpredictable changes in asset prices. Modeling and capturing such volatility and nonlinear patterns in financial time series data require advanced analytical techniques.
2. Data Quality and Missing Values: Financial data can be prone to errors, outliers, and missing values. Incomplete or erroneous data can affect the accuracy and reliability of time series analysis. Dealing with missing values and ensuring data quality is crucial to obtain meaningful results.
3. Nonstationarity: Financial time series data often exhibits nonstationarity, where the statistical properties of the data change over time. This violates the assumption of constant mean, variance, and covariance, which is necessary for many traditional time series models. Nonstationarity poses challenges in modeling and forecasting financial data accurately.
4. Seasonality and Calendar Effects: Financial data may exhibit seasonality or calendar effects due to regular patterns observed over specific time intervals. For example, daily, weekly, or monthly patterns in stock prices or trading volumes. Incorporating and accounting for seasonality in time series analysis is essential for accurate modeling and forecasting.
5. High-Dimensional and Multivariate Data: Financial data can be high-dimensional, involving a large number of variables or features. This includes data from multiple financial markets, economic indicators, and other relevant factors. Analyzing and extracting meaningful information from such high-dimensional and multivariate time series data require advanced modeling techniques.
6. Noise and Uncertainty: Financial data is often subject to noise, which can obscure underlying patterns and relationships. Distinguishing between signal and noise is crucial for accurate analysis. Additionally, financial markets are inherently uncertain, and future outcomes are influenced by various unpredictable factors. Incorporating and quantifying uncertainty in time series analysis is a challenge.
7. Overfitting and Model Selection: Selecting an appropriate model for time series analysis in finance is challenging due to the vast number of available models and the risk of overfitting. Overfitting occurs when a model performs well on the training data but fails to generalize to new data. Proper model selection and regularization techniques are necessary to avoid overfitting and ensure robustness in analysis.
8. Real-Time Analysis: Financial markets operate in real-time, with data arriving continuously. Analyzing and making predictions in real-time requires efficient and scalable algorithms that can handle large volumes of data with low latency.

Addressing these challenges in time series analysis for financial data often involves the use of advanced statistical and machine learning techniques. Researchers and practitioners continually strive to develop and improve models that can effectively capture the complexity, volatility, and nonlinear relationships present in financial time series data.

### 1.6 Deep Learning Algorithms for Time Series Analysis

Deep learning algorithms have gained significant attention and proven to be highly effective in analyzing and modeling time series data, including financial time series. These algorithms are capable of automatically learning complex patterns, capturing nonlinear relationships, and handling large-scale data. Here are some commonly used deep learning algorithms for time series analysis:

1. **Long Short-Term Memory (LSTM) Networks**: LSTM networks are a type of recurrent neural network (RNN) specifically designed to model sequential data. They are widely used for time series analysis due to their ability to capture long-term dependencies and handle the vanishing gradient problem. LSTM networks have shown success in various financial applications, such as stock price prediction, market sentiment analysis, and anomaly detection.
2. **Convolutional Neural Networks (CNNs):** Although primarily used for image processing, CNNs have also been applied to time series analysis tasks. CNNs can extract local patterns and features from time series data through a series of convolutional layers and pooling operations. They have been used for tasks such as sensor data analysis, activity recognition, and algorithmic trading.
3. **Generative Adversarial Networks (GANs):** GANs are a type of deep learning architecture consisting of two neural networks, a generator and a discriminator, that compete against each other. GANs have been utilized in time series analysis to generate synthetic data that resembles real financial time series data. GANs have shown promise in generating realistic stock price sequences and simulating financial scenarios.
4. **Autoencoders:** Autoencoders are unsupervised learning models that aim to reconstruct the input data from a compressed latent representation. They have been used for dimensionality reduction, anomaly detection, and denoising in time series analysis. Autoencoders are particularly useful for financial data, where they can identify abnormal patterns or reconstruct missing data points.
5. **Recurrent Neural Networks (RNNs):** RNNs are a class of neural networks designed to process sequential data by maintaining an internal memory of past inputs. They have been applied to time series analysis tasks, including stock market prediction, sentiment analysis, and natural language processing in finance. RNN variants, such as Gated Recurrent Units (GRUs), are commonly used due to their ability to capture temporal dependencies.

These deep learning algorithms offer powerful capabilities for time series analysis by leveraging their ability to learn hierarchical representations and capture complex temporal relationships. However, the choice of algorithm depends on the specific task, dataset characteristics, and computational resources available. It is essential to carefully select and tune the deep learning algorithm according to the requirements and objectives of the financial time series analysis.

### 1.7 Introduction to Deep Learning

Deep learning is a subfield of machine learning that focuses on training artificial neural networks to learn and make intelligent decisions. It is inspired by the structure and function of the human brain and aims to mimic its learning capabilities.

At its core, deep learning involves building and training artificial neural networks with multiple layers, hence the term "deep." These networks are composed of interconnected nodes, called neurons, organized in layers. The layers closest to the input data are called the input layers, while those closest to the output are the output layers. The intermediate layers are referred to as hidden layers.

Deep learning models learn to perform tasks by iteratively adjusting the parameters, or weights, of the neural network based on a training dataset. The training process involves presenting the network with input data and comparing its output with the desired output. Through a process known as backpropagation, the network updates its weights to minimize the difference, or error, between the predicted output and the expected output. This iterative process allows the network to gradually improve its performance.

One of the key advantages of deep learning is its ability to automatically extract meaningful features from raw data. Traditional machine learning approaches often require manual feature engineering, where domain experts manually design features that are relevant to the task at hand. In deep learning, features are learned directly from the data, eliminating the need for manual feature engineering in many cases.

Deep learning has achieved remarkable success in various fields, including computer vision, natural language processing, speech recognition, and reinforcement learning. It has been used to develop applications such as image classification, object detection, machine translation, speech synthesis, and autonomous driving.

While deep learning has shown great promise, it also comes with certain challenges. Deep neural networks can be computationally intensive and require large amounts of labeled data for effective training. Overfitting, where the network becomes too specialized to the training data and performs poorly on unseen data, is another challenge that needs to be addressed.

In recent years, advancements in hardware capabilities, such as graphical processing units (GPUs) and specialized deep learning accelerators, have greatly accelerated the adoption and training of deep learning models. Researchers and practitioners continue to explore new techniques and architectures to improve the performance and efficiency of deep learning algorithms.

Overall, deep learning represents a powerful approach to machine learning that has revolutionized many areas of artificial intelligence and has the potential to drive further advancements in the future.

### 1.8 Relevance of Deep Learning in Financial Data Analysis

Deep learning has gained significant relevance in financial data analysis due to its ability to handle large, complex datasets and extract meaningful patterns and insights. Here are several key areas where deep learning has made an impact:

## 1.8.1 Predictive modeling

Deep learning models have been successfully applied to predict financial market trends, stock prices, exchange rates, and other financial variables. By analyzing historical data, these models can learn complex relationships and patterns, enabling them to make accurate predictions.

## 1.8.2 Risk Assessment:

Deep learning techniques can be used to assess credit risk, fraud detection, and other risk-related tasks. By analyzing vast amounts of data, including transaction records, customer behavior, and economic indicators, deep learning models can identify patterns indicative of potential risks or anomalies.

## 1.8.3 Portfolio Optimization:

Deep learning algorithms can aid in portfolio optimization by analyzing historical market data and optimizing the allocation of assets. By considering various factors and market dynamics, these models can suggest optimal investment strategies and risk management techniques.

## 1.8.4 Algorithmic Trading:

Deep learning plays a crucial role in algorithmic trading, where trading decisions are automated based on predefined rules. Deep learning models can analyze real-time market data, news feeds, and other relevant information to generate trading signals and execute trades.

## 1.8.5 Natural Language Processing (NLP) for Sentiment Analysis:

Deep learning techniques in NLP can be used to analyze financial news, social media sentiment, and company reports to gauge market sentiment and make informed investment decisions. Sentiment analysis can provide valuable insights into market trends and investor sentiment.

## 1.8.6 Fraud Detection:

Deep learning models can be trained to detect fraudulent activities, such as credit card fraud, insurance fraud, or money laundering. These models can learn patterns and anomalies in transactional data, enabling them to flag suspicious activities for further investigation.

## 1.8.7 Customer Behavior Analysis:

Deep learning can be applied to analyze customer behavior and preferences in the financial industry. By analyzing transaction data, browsing history, and other relevant information, deep learning models can provide personalized recommendations, detect customer churn, and enhance customer experience.

## 1.8.8 High- High-Frequency Trading: Frequency Trading:

Deep learning models have been used in high-frequency trading to analyze and make rapid trading decisions based on real-time market data. The ability to process vast amounts of data quickly and make fast predictions is crucial in this domain.

It's worth noting that deep learning models should be developed and deployed with care in the financial domain, as the stakes are high and accurate predictions are essential. Proper validation, testing, and risk management procedures should be followed to ensure the reliability and robustness of these models.

### 1.9 Long Short-Term Memory (LSTM) Networks

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network (RNN) architecture that is particularly effective in handling sequential data and addressing the vanishing gradient problem commonly encountered in traditional RNNs.

LSTMs were introduced by Hochreiter and Schmidhuber in 1997 and have since become a popular choice for tasks involving sequential data, such as natural language processing, speech recognition, time series analysis, and more. LSTMs are designed to capture long-term dependencies and retain information over longer sequences.

The key component of an LSTM is its memory cell, which consists of a cell state and various gates that control the flow of information. The gates are responsible for selectively updating and accessing the cell state, allowing LSTMs to retain and forget information as needed. The three main gates in an LSTM are:

## 1.9.1 Forget Gate:

This gate determines which information from the previous cell state should be forgotten. It takes the previous hidden state and the current input and outputs a forget vector, which is then element-wise multiplied with the previous cell state to discard irrelevant information.

## 1.9.2 Input Gate:

The input gate decides which new information should be stored in the cell state. It consists of two parts: the input activation function, which processes the current input and hidden state, and the gate activation function, which determines the amount of new information to be added.

## 1.9.3 Output Gate:

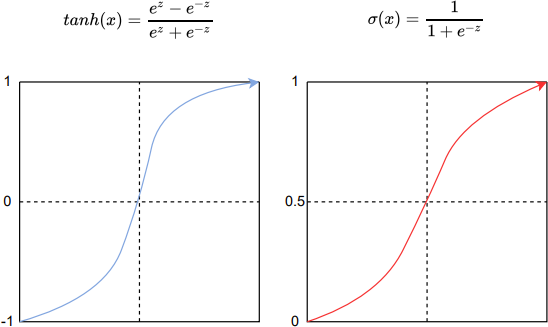
The output gate controls the flow of information from the cell state to the next hidden state. It takes the current input and hidden state, processes them, and produces an output vector that is a filtered version of the cell state.

By selectively updating and accessing the cell state through these gates, LSTMs can handle long sequences by retaining and utilizing relevant information while mitigating the vanishing gradient problem that affects traditional RNNs. The vanishing gradient problem refers to the issue where gradients diminish rapidly over time, making it difficult for the network to learn long-range dependencies.

LSTMs have a rich and expressive memory mechanism that allows them to capture and model complex patterns in sequential data. This makes them well-suited for tasks such as language modeling, machine translation, sentiment analysis, speech recognition, and any other task involving sequential or time series data.

In practice, LSTMs are implemented using computational graph libraries or deep learning frameworks such as Tensor Flow or PyTorch. These libraries provide convenient APIs for defining and training LSTM models, handling the complexities of back propagation through time, and optimizing the network parameters.

Overall, LSTM networks have proven to be a powerful architecture for modeling sequential data, allowing for the capture of long-term dependencies and addressing the challenges associated with traditional RNNs.

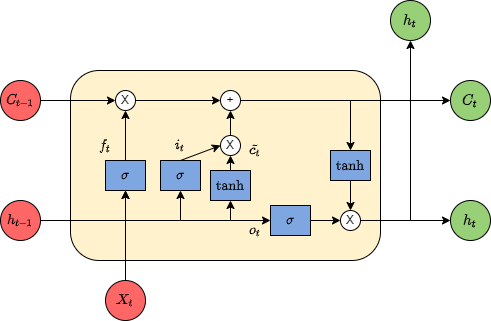


**Figure 1** Activation function used in LSTM

### 1.10 LSTM cell structure

LSTM networks are built up from cells, each of which consists of several separate components. In **figure**[,](#_bookmark31) the lines represent the transmission of the vector in the direction of the arrow. There is no split in the values when the lines diverge; they are copied instead. LSTMs have three inputs: the memory from the previous time step (*ct*´1), the activation or input from the previous time step (*ht* 1), and the new data value at a time (*Xt*) (Purkait, 2019). The blue boxes show the activation functions that are used by the so-called gates.

´



**Figure 2** Structure of LSTM cell

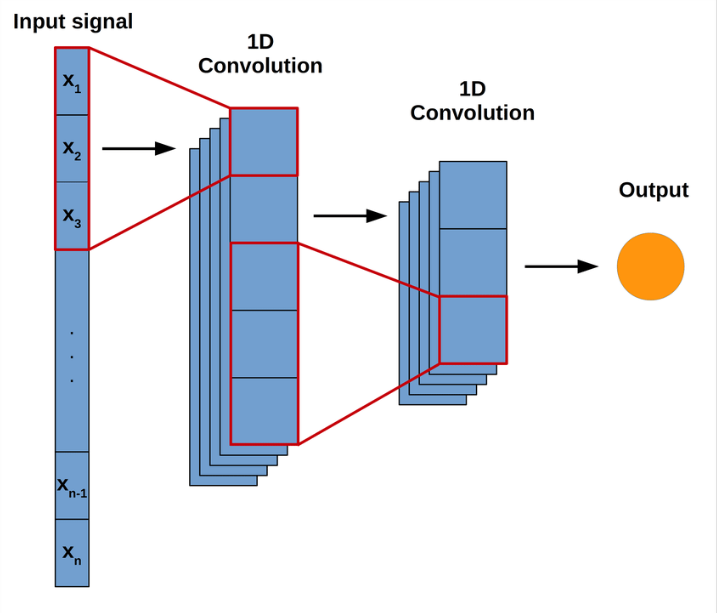
According to El-Amir and Hamdy, 2019, gates are the place where the different opera- tions occur. They are from left to right in **figure** [3.5:](#_bookmark31) the forget gate, input gate, update gate, and output gate. A forget gate is a crucial part of the LSTM as it determines whether to keep a value from previous timesteps or to forget and incorporate it in the subsequent time step. Such mechanisms work particularly well when dealing with long-term dependencies. Circular shapes represent pointwise operations

### 1.11 Comparative Analysis of Deep Learning Algorithms

Deep learning encompasses various algorithms that have demonstrated significant success in different domains. Here's a comparative analysis of some popular deep learning algorithms:

## 1.11 1 Convolutional Neural Networks (CNNs):

* Domain: Computer vision tasks, image and video analysis.
* Key Features: CNNs excel at extracting spatial features from images through convolutional layers and pooling operations. They have been instrumental in tasks such as image classification, object detection, and image segmentation.
* Strengths: CNNs are adept at capturing local patterns, translation invariance, and hierarchical representations.
* Limitations: They may struggle with handling sequential or time-series data, as they are primarily designed for grid-like data structures.



**Figure 3** Structure ofConvolutional Neural Networks

## 1.11 2 Recurrent Neural Networks (RNNs):

* Domain: Natural language processing, speech recognition, time series analysis.
* Key Features: RNNs are designed to process sequential data by utilizing feedback connections that allow information to persist over time. They can handle variable-length input and are effective in tasks involving sequential dependencies.
* Strengths: RNNs can capture temporal dependencies and long-range contextual information.
* Limitations: RNNs often suffer from the vanishing/exploding gradient problem, making it difficult to learn dependencies across long sequences.

A diagram of a network

Description automatically generated

**Figure 4** Structure of Recurrent Neural Networks (RNNs)

## 1.11 3 Long Short-Term Memory (LSTM) Networks:

* Domain: Sequential data analysis, natural language processing.
* Key Features: LSTMs are a specialized type of RNN that effectively address the vanishing gradient problem. They incorporate memory cells and gating mechanisms to selectively retain or discard information over long sequences, making them suitable for capturing long-term dependencies.
* Strengths: LSTMs excel in modeling long-range dependencies in sequential data and have achieved impressive results in tasks like machine translation, sentiment analysis, and speech recognition.
* Limitations: LSTMs can be computationally expensive and require substantial amounts of data for training.

## 1.11 4 Generative Adversarial Networks (GANs):

* Domain: Image synthesis, data generation.
* Key Features: GANs consist of two neural networks—a generator and a discriminator—competing against each other. The generator tries to produce realistic synthetic data, while the discriminator aims to differentiate between real and generated data. This adversarial training process enables GANs to generate high-quality and diverse synthetic samples.
* Strengths: GANs can generate realistic images, videos, and other types of data. They have applications in image synthesis, style transfer, and data augmentation.
* Limitations: GAN training can be unstable, and it can be challenging to evaluate the quality of generated samples objectively.

## 1.11 5 Transformer Networks:

* Domain: Natural language processing, machine translation.
* Key Features: Transformers are based on self-attention mechanisms that allow the model to weigh the importance of different words in a sentence. They excel in processing and generating sequences of variable lengths. Transformers have revolutionized machine translation and achieved state-of-the-art results in natural language processing tasks.
* Strengths: Transformers can capture global dependencies and handle long-range relationships in sequential data effectively.
* Limitations: Transformers may require substantial computational resources, and training them can be time-consuming due to their large parameter space.

It's important to note that the choice of deep learning algorithm depends on the specific problem, available data, computational resources, and the nature of the input data. Each algorithm has its own strengths and limitations, and practitioners often experiment with multiple algorithms to find the best approach for a given task.

### Chapter 2

### Review of literature

The article highlights the use of deep learning approaches in finance, specifically for stock price forecasting. The authors propose a novel deep learning framework that combines wavelet transforms, stacked autoencoders, and long-short term memory. The framework comprises three stages: the stock price time series is decomposed by wavelet transforms to eliminate noise, stacked autoencoders are applied to generate deep high-level features for predicting the stock price, and high-level denoising features are fed into LSTM to forecast the next day’s closing price. The proposed model is tested on six market indices and their corresponding index futures, and the results show that the model outperforms other similar models in both predictive accuracy and profitability performance. (Bao *et al* 2017)

The article discusses the importance of time series forecasting for various applications and industrial processes. It then proposes a new algorithm that combines clustering, classification, and forecasting techniques to predict both univariate and multivariate time series. This algorithm groups windows of time series values with similar patterns to build a specific forecasting model for each pattern. The model is generated using any combination of approaches within multiple machine learning techniques. Several experiments are carried out using different configurations of the clustering, classification, and forecasting methods that the model consists of. The results are analyzed and compared to classical prediction models and well-known methods in the literature. (Castán *et al* 2022)

The article discusses the importance of accurate and timely spatial classification of crop types based on remote sensing data. The authors propose a novel approach to generate accurate, cost-effective, and in-season crop-type classification by using the USDA’s Common Land Units (CLUs) to aggregate spectral information for each field based on a time-series Landsat image data stack to overcome cloud contamination issues. They also use a machine learning model based on Deep Neural Network (DNN) and high-performance computing for intelligent and scalable computation of classification processes. The experiments conducted in this research aimed to evaluate what information is most useful for training the machine learning model for crop-type classification, and how various spatial and temporal factors affect the crop-type classification performance to derive timely crop type information. (Cai *et al* 2018).

The article highlights the potential of deep learning models to advance the short-term decision-making of electricity market participants and system operators by capturing the complex dependences and uncertainties of power system operation. The authors review recent developments in the field of probabilistic, multivariate, and multihorizon time series forecasting and empirically evaluate the performance of novel global deep learning models for forecasting wind and solar generation, electricity load, and wholesale electricity price for intraday and day-ahead time horizons. The evaluation data consist of real-world datasets with hourly resolution at the levels of an individual customer and regional and national electricity market bidding zones. The model evaluation criteria include achievable levels of forecasting accuracy and uncertainty risks, hyperparameter sensitivity, the effect of exogenous variables and fieldwise dataset split, and run-time efficiency factors. The results can serve as a reference point for the quantitative evaluation of deep learning models for probabilistic multivariate energy forecasting in power systems. (Mashlakov *et al* 2021)

The article discusses the development of a deep learning framework for incorporating longitudinal clinical data from EHR to infer risk for pancreatic cancer. The framework includes a novel training protocol that enforces an emphasis on early detection by applying an independent Poisson-random mask on proximal-time measurements for each variable. Data fusion for irregular multivariate time-series features is enabled by a "grouped" neural network (GrpNN) architecture, which uses representation learning to generate a dimensionally reduced vector for each measurement set before making a final prediction. The models were evaluated using EHR data from Columbia University Irving Medical Center-New York Presbyterian Hospital. The framework demonstrated better performance on early detection at 12 months prior to diagnosis compared to a logistic regression, xgboost, and a feedforward neural network baseline. The results were consistent across reported race. The proposed algorithm is potentially generalizable to other diseases, including cancer, where early detection can improve survival. (Park *et al* 2022)

This article discusses the development of a framework for extracting features in an unsupervised manner using deep learning, particularly stacked LSTM Autoencoder Networks, for multivariate time series classification. The compressed representation of the time-series data obtained from LSTM Autoencoders is then provided to Deep Feedforward Neural Networks for classification. The proposed framework is applied on sensor time series data from the process industry to detect the quality of the semi-finished products and accordingly predict the next production process step. The efficiency of the proposed approach is validated using real-world data from the steel industry. (Mehdiyev *et al* 2017)

6

This article discusses the application of Long Short Term Memory (LSTM) to time series prediction, which is a difficult problem due to the presence of long term trend, seasonal and cyclical fluctuations, and random noise. The performance of LSTM is highly dependent on several hyper-parameters, for which there are no established guidelines. The authors addressed this research gap by creating a dataset from the Indian stock market and developing an LSTM model for it. The model was optimized by comparing stateless and stateful models and by tuning for the number of hidden layers.

7

This article discusses the importance of wind power forecasts for electric system operators, as wind power is not fully dispatchable. The authors compare the performance of five deep learning models, each combined with three types of data pre-processing, for short-term and long-term multi-variate predictions of wind power capacity factor and temperature. The authors propose a multiple input and multiple output (MIMO) architecture for multi-variate prediction. The five models investigated are Deep Feed Forward (DFF), Deep Convolutional Network (DCN), Recurrent Neural Network (RNN), Attention mechanism (Attention), and Long Short-Term Memory Networks (LSTM). The authors also propose a novel approach to transform the time series dataset to signal for input and reconstruct the model predictions through inverse transformation, by means of the so-called discrete wave.

8

This article proposes a hybrid model for crude oil price prediction that uses complex network analysis and long short-term memory (LSTM) of deep learning algorithms. The visibility graph tool is used to map the dataset on a network and K-core centrality is employed to extract the non-linearity features of crude oil and reconstruct the dataset. The complex network analysis is carried out to preprocess the original data to extract the non-linearity features and to reconstruct the data. LSTM is then employed to model the reconstructed data. The experiments show that the proposed model has higher accuracy, and is more robust and reliable compared to other research in the literature.

9

In this study, we applied not only regression methods in machine learning but also time series analysis techniques to forecast the sales amount based on several features. The authors applied their models on Walmart sales data in Microsoft Azure Machine Learning Studio platform. The following regression techniques were applied: Linear Regression, Bayesian Regression, Neural Network Regression, Decision Forest Regression and Boosted Decision Tree Regression. In addition to these regression techniques, the following time series analysis methods were implemented: Seasonal ARIMA, Non-Seasonal ARIMA, Seasonal ETS, Non -Seasonal ETS, Naive Method, Average Method, and Drift Method. It was shown that Boosted Decision Tree Regression provides the best performance on this sales data. This project is a part of the development of a new decision support system for the retail industry.

10

The paper "Deep learning for financial time series forecasting in a-trader system" by Korczak and Hemes (2017) presents methods for financial time series forecasting using deep learning in relation to multi-agent stock trading system, called A-Trader. The paper discusses the problem of financial time series on FOREX market, outlines classical neural networks and deep learning models, and analyzes their performances. The final part presents deployment and evaluation of a deep learning model implemented using H20 library as an agent of A-Trader system. On the basis of this model, an investment strategies in A-Trader system can be built.

11

The course project focuses on using machine learning to predict future bond prices. Bond prices are a reflection of complex market interactions and policies, making prediction difficult. The dataset used describes the previous 10 trades of a large number of bonds among other relevant descriptive metrics to predict future bond prices. The dataset includes 762,678 bonds, each with a total of 61 attributes, including a ground truth trade price. Various supervised learning algorithms for regression followed by ensemble methods were evaluated, with feature and model selection considerations being treated in detail. All methods were evaluated on both accuracy and speed. Finally, a novel hybrid time-series aided machine learning method was proposed for future work.

12

This study proposes a Temporal Logistic Neural Bag-of-Features approach to address the high-dimensionality, velocity, and variety of data collected in time series forecasting applications, such as forecasting stock markets and energy load prediction. The proposed method can be effectively combined with deep neural networks to create powerful deep learning models for time series analysis. The study overcomes the limitations of combining existing BoF formulations with deep feature extractors by employing a novel adaptive scaling mechanism and replacing the Gaussian-based density estimation involved in the regular BoF model with a logistic kernel. The effectiveness of the proposed approach is demonstrated using extensive experiments on a large-scale financial time series dataset that consists of more than 4 million limit orders.

13

This study surveys over 150 articles on applying machine learning to financial market forecasting and shows that machine learning algorithms tend to outperform most traditional stochastic methods in financial market forecasting. While machine learning algorithms offer a proven way of modeling non-linearities in time series, their advantages against common stochastic models in financial market prediction are largely based on limited empirical results. The study finds that recurrent neural networks outperform feed forward neural networks as well as support vector machines, implying the existence of exploitable temporal dependencies in financial time series across multiple asset classes and geographies.

14

Financial time series forecasting is a popular research area among both academia and industry, with machine learning and deep learning models being the top choice of computational intelligence. There are a vast number of studies published on this topic, and several surveys exist covering machine learning for financial time series forecasting. However, there is a lack of review papers solely focused on deep learning for finance. A recent study provides a comprehensive literature review on deep learning studies for financial time series forecasting implementations, categorized according to their intended forecasting implementation areas and grouped based on their deep learning model choices. The study also highlights possible setbacks and opportunities for the future of the field.

15

Anomaly detection in time-series data has been a significant research area for a long time. Most of the seminal work on anomaly detection methods has focused on statistical approaches. However, in recent years, an increasing number of machine learning algorithms have been developed to detect anomalies in time-series. As a result, researchers have attempted to improve these techniques using (deep) neural networks. Despite the increasing number of anomaly detection methods, the body of research lacks a broad comparative evaluation of statistical, machine learning, and deep learning methods. This paper aims to address this gap by studying 20 univariate anomaly detection methods from all three categories. The evaluation is conducted on publicly available datasets that serve as benchmarks for time-series anomaly detection. By analyzing the accuracy of each method and the computation time of the algorithms, the authors provide a thorough insight into the performance of these anomaly detection approaches, as well as some general notion of which method is suited for a certain type of data.

16

This paper aims to determine the predictable price direction of Bitcoin in USD by utilizing machine learning techniques and sentiment analysis. Researchers have shown great interest in studying public sentiment through Twitter and Reddit. This paper applies sentiment analysis and supervised machine learning principles to the extracted tweets from Twitter and Reddit posts and analyzes the correlation between Bitcoin price movements and sentiments in tweets. The authors explore several machine learning algorithms using supervised learning to develop a prediction model and provide informative analysis of future market prices. Due to the difficulty of evaluating the exact nature of a Time Series (ARIMA) model, the authors implement Recurrent Neural Networks (RNN) with long short-term memory cells (LSTM). The paper analyzes the time series model prediction of Bitcoin prices with greater efficiency using long short-term memory (LSTM) techniques and compares the predictability of Bitcoin price and sentiment analysis of Bitcoin tweets to the standard method (ARIMA). The RMSE (Root-mean-square error) of LSTM are 198.448 (single feature) and 197.515 (multi-feature) whereas the ARIMA model RMSE is 209.263, which shows that LSTM with multi-feature shows a more accurate result.

17

The study of predicting stock prices is a complex and challenging task due to the chaotic and dynamic nature of the market. In recent years, researchers have turned to machine learning techniques to tackle this challenge. Many machine learning models have been proposed for this purpose, and they have shown promising results. This literature review focuses on one such model, the Bi-LSTM-CNN model, which is a generative adversarial network composed of a bi-directional Long short-term memory (LSTM) and convolutional neural network(CNN). The Bi-LSTM-CNN model generates synthetic data that agrees with existing real financial data, and it retains the features of stocks with positive or negative trends to predict future trends of a stock. This proposed solution is unique in that it introduces the concept of a hybrid system (Bi-LSTM-CNN) rather than a sole LSTM model. The data collected for this study was from multiple stock markets such as TSX, SHCOMP, KOSPI 200, and the S&P 500. The authors proposed an adaptive-hybrid system for trend prediction on stock market prices and carried out a comprehensive evaluation on several commonly utilized machine learning prototypes. The study concludes that the proposed solution approach outperforms preceding models. Furthermore, the authors identified gaps between investors and researchers dedicated to the technical domain during the research stage from preceding works. This literature review contributes to the existing body of knowledge on machine learning models for stock price prediction and highlights the importance of addressing the communication gap between investors and researchers in this area.

18

The paper discusses the performance of the hybrid Deep Neural Network (DNN) algorithm, TreNet, for predicting trends in time series data. While TreNet was shown to have superior performance for trend prediction to other DNN and traditional ML approaches, the validation method used did not take into account the sequential nature of time series data sets and did not deal with model update. The authors replicated the TreNet experiments on the same data sets using a walk-forward validation method and tested their optimal model over multiple independent runs to evaluate model stability. They compared the performance of the hybrid TreNet algorithm on four data sets to vanilla DNN algorithms that take in point data, and also to traditional ML algorithms. The authors found that in general, TreNet still performs better than the vanilla DNN models, but not on all data sets as reported in the original TreNet study. The study highlights the importance of using an appropriate validation method and evaluating model stability when developing and evaluating machine learning models for trend prediction in time series data.

19

The paper discusses the vulnerability of deep learning (DL) regression models for multivariate time series (MTS) forecasting to adversarial examples. Researchers have started adopting DL techniques for solving MTS data mining problems in various domains including finance, cybersecurity, energy, healthcare, and prognostics. However, DL algorithms are susceptible to adversarial examples, which also makes the DL regression models for MTS forecasting vulnerable to those attacks. The authors leverage existing adversarial attack generation techniques from the image classification domain and craft adversarial multivariate time series examples for three state-of-the-art DL regression models, specifically Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU). They evaluate their study using Google stock and household power consumption dataset. The obtained results show that all the evaluated DL regression models are vulnerable to adversarial attacks, transferable, and thus can lead to catastrophic consequences in safety-critical and cost-critical domains, such as energy and finance.

21

This paper proposes a novel method called Self-Adaptive Forecasting (SAF) for improving the performance of time-series forecasting models on non-stationary time-series data. SAF integrates a self-adaptation stage prior to forecasting based on 'backcasting', which is a form of test-time training that uses a self-supervised learning problem on test samples to train the model before performing the prediction task. This enables efficient adaptation of encoded representations to evolving distributions, leading to superior generalization. SAF can be integrated with any canonical encoder-decoder based time-series architecture, including recurrent- or attention-based ones. On synthetic and real-world datasets in domains such as healthcare and finance, where time-series data are notoriously non-stationary, we demonstrate a significant benefit of SAF in improving forecasting accuracy.

22

This literature review provides a comprehensive assessment of recent research from 2020 to 2022 on deep learning models used to predict prices based on financial time series. The review presents different data sources and neural network structures, as well as their implementation details. The goal is to ensure that interested researchers remain up-to-date on recent developments in the field and facilitate the selection of baselines based on models used in prior studies. Additionally, suggestions for future research based on the content in this review are provided.

23

Support vector regression (SVR) has been successfully applied in financial time series forecasting due to its generalization capability in obtaining a unique solution. However, the high noise inherent in financial time series modeling using SVR is a key problem. To alleviate the influence of noise, a two-stage modeling approach using independent component analysis (ICA) and support vector regression is proposed in financial time series forecasting. ICA is a novel statistical signal processing technique that was originally proposed to find the latent source signals from observed mixture signals without having any prior knowledge of the mixing mechanism. Experimental results show that the proposed model outperforms the SVR model with non-filtered forecasting variables and a random walk model.

24

This study proposes a novel algorithmic trading model CNN-TA using a 2-D convolutional neural network based on image processing properties. To convert financial time series into 2-D images, 15 different technical indicators each with different parameter selections are utilized. Each indicator instance generates data for a 15-day period, resulting in 15x15 sized 2-D images. Each image is then labeled as Buy, Sell or Hold depending on the hills and valleys of the original time series. The results indicate that the trained model provides better results for stocks and ETFs when compared with the Buy & Hold Strategy and other common trading systems over a long out-of-sample period.

25

This paper proposes a non-stationary NDVI time series forecasting model by combining big data system, wavelet transform, and long short-term memory neural network. The MapReduce algorithm was used for RS data storage and NDVI TS extraction. The WT was used to decompose the TS into different components, and LSTM was used for NDVI TS forecasting. The proposed methodology using WT-LSTM model provides an efficient method for forecasting NDVI TS in terms of RMSE and R. Additionally, the performance of the big data model was evaluated.

26

This passage discusses the challenges of analyzing financial data and the limitations of traditional approaches, such as ARIMA and the exponential smoothing model, when dealing with large amounts of data with intrinsic complexity, high dimensionality, and casual dynamicity. The author notes that traditional models are not suitable for understanding hidden relationships between data. The author further highlights that recent machine learning (ML) techniques in quantitative finance have demonstrated their superiority over traditional approaches. Several studies have shown that artificial neural networks (ANNs), support vector machines (SVMs), and deep learning (DL) models outperform traditional models in predicting financial asset prices, volatility, and risk. The author also presents comparative studies of the effectiveness of several ML-based systems. This paper is a valuable contribution to the literature on financial data analysis, as it provides researchers with a comprehensive overview of the limitations of traditional approaches and highlights the benefits of using ML techniques in quantitative finance.

27

This passage discusses the various models that have been explored to forecast the stock market, a classic but challenging topic that has attracted the interest of economists and computer scientists alike. The author notes that researchers have explored different models, including linear models, machine learning (ML) models, deep learning (DL) models, reinforcement learning (RL) models, and deep reinforcement learning (DRL) models, to create an accurate predictive model. The use of ML algorithms has allowed for the extraction of high-level financial market data patterns, which has given investors an advantage in anticipating and evaluating stock and foreign exchange markets. The author further notes that recently, the use of DRL algorithms has become increasingly popular in algorithmic trading, with DRL agents constructing completely automated trading systems or strategies that combine price prediction and trading signal production. This paper is a valuable contribution to the literature on stock market forecasting, as it provides a comprehensive overview of the various models that have been used and highlights the potential benefits of using DRL algorithms in algorithmic trading.

28This passage discusses the importance of selecting the most suitable algorithm for temporal data forecasting. The author notes that few scientific publications have rigorously exposed the benefits and limitations of the most popular algorithms for time series prediction. To address this gap in the literature, the author presents an extensive experimental evaluation of eleven predictors, seven parametric and four non-parametric, employing two multi-step-ahead projection strategies and four performance evaluation measures. The results show that SARIMA is the only statistical method able to outperform, but without a statistical difference, the following machine learning algorithms: ANN, SVM, and kNN-TSPI. However, such forecasting accuracy comes at the expense of a larger number of parameters. The author provides detailed results achieved by different indexes, which are available online in a repository. The author's findings provide a broad insight into model selection, parameter setting, evaluation measures, and experimental design that will impact further research on this topic. This paper is a valuable contribution to the literature on time series forecasting, as it provides researchers with a more rigorous understanding of the benefits and limitations of different forecasting algorithms.

30

This passage discusses the use of modern Machine Learning and Deep Learning techniques in the Forex market, the largest financial market with a huge amount of daily trading volume. The author notes that traditionally, Forex traders used Fundamental Analysis and Technical Analysis tools or strategies. However, with the advancement of computational technology, Artificial Intelligence (AI) plays a significant role in the financial domain. The author further highlights that most existing models are developed targeting the stock market, and there is still a lag of research that applies modern Machine Learning or Deep Learning for predicting the movement of the price in the Forex market. The author proposes a novel predicting model based on Deep Convolutional Neural Network that can be effectively used as a tool to make profits for Forex traders. The author evaluates the performance of the proposed CNN model from two perspectives: accuracy of the prediction and ability to make profits. The experimental results show that the proposed CNN model provides an accuracy of approximately 77%, and it produces approximately $69K for one and a half year (from January 2017 to September 2018) in terms of financial perspective. This paper is a valuable contribution to the literature on the use of AI in the Forex market, as it provides Forex traders with a novel predicting model that can be used as a tool to make profits.

31

This literature review examines the challenge of accurately forecasting non-stationary time-series data using deep neural networks (DNNs). The study evaluates the performance of popular DNN models, including Multi-layer Perceptron (MLP), Convolutional Neural Network (CNN), and RNN with Long Short-Term Memory (LSTM-RNN) and RNN with Gated-Recurrent Unit (GRU-RNN), for single-step and multi-step forecasting of 10 popular Indian financial stocks data. Results show that these DNN methods perform well for single-step forecasting, but long forecast periods have an adverse effect on performance. The study highlights the importance of understanding trend and seasonality in non-stationary time-series data for accurate forecasting.

32

This research article proposes a new deep learning model, the Stacked Long Short-Term Memory (S-LSTM) model, to address the challenges of forecasting multidimensional, dynamic, and nonlinear financial time-series data. The model is constructed by stacking multiple Long Short-Term Memory (LSTM) units, and six different data normalization techniques are used to preprocess the dataset. The performance of the S-LSTM model is evaluated using multivariate financial time-series data from the Bombay Stock Exchange (BSE) and New York Stock Exchange (NYSE). The experimental results demonstrate that the prediction performance of the S-LSTM model can be improved with the appropriate selection of the data normalization technique, and that the model outperforms other well-known methods in terms of prediction accuracy.

33

This paper reviews recent work that uses machine learning-based techniques to extract knowledge from time series data. Time series data represents a time sequence of numerical values observed in the past at a measurable variable and is sampled at equidistant time intervals. The goal is to identify recurring structures in the data to predict future behavior. Machine learning-based algorithms extract knowledge from the data to find patterns or hidden causal relationships. The prediction model extracts knowledge through an inductive process where the input is the data and, possibly, a first example of the expected output.

**35**

This passage presents a novel deep learning-based anomaly detection approach called DeepAnT for time series data, which can detect a wide range of anomalies. DeepAnT uses unlabeled data to capture and learn the data distribution that is used to forecast the normal behavior of a time series. The approach consists of two modules: time series predictor and anomaly detector. The time series predictor module uses deep convolutional neural network (CNN) to predict the next time stamp, and the predicted value is then passed to the anomaly detector module, which is responsible for tagging the corresponding time stamp as normal or abnormal. DeepAnT can be trained on relatively small data sets while achieving good generalization capabilities due to the effective parameter sharing of the CNN. As the anomaly detection in DeepAnT is unsupervised, it does not rely on anomaly labels at the time of model generation.

36

This passage discusses the use of machine learning-based time series analysis to predict the market price and stability of Bitcoin in the cryptocurrency market. The author evaluated ARIMA, FBProphet, and XG Boosting models based on Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and R2. After conducting experiments, the author found that ARIMA is the best model for forecasting Bitcoin price in the crypto-market. This research can be helpful for investors of the crypto-market.

37

This paper presents an ensemble method for load demand forecasting in electric utilities using the Empirical Mode Decomposition (EMD) algorithm and a deep learning approach. The load demand series are decomposed into intrinsic mode functions (IMFs), and a Deep Belief Network (DBN) is used to model each IMF. The prediction results of all IMFs are combined to obtain an aggregated output for load demand. The proposed method is tested using electricity load demand data sets from the Australian Energy Market Operator (AEMO) and is shown to be effective compared to nine other forecasting methods.

38

This paper explores the use of unsupervised learning techniques to cluster data and identify hidden patterns in HVAC consumption data and building usage data to improve energy efficiency in commercial buildings. The study focuses on a highly efficient office building located in Houston, Texas, and uses time series analysis and the K-means clustering algorithm to identify new energy-saving opportunities. The results show a potential energy savings of 6% using the K-means algorithm, indicating the usefulness of clustering in helping building managers identify additional energy-saving opportunities.

39

This paper focuses on predicting the future values of four stock market groups from the Tehran stock exchange using various machine learning algorithms. The chosen groups are diversified financials, petroleum, non-metallic minerals, and basic metals. Historical data from 10 years were collected for each group, and 10 technical indicators were selected as inputs for each prediction model. The results show that LSTM is the most accurate algorithm with the highest model fitting ability. For tree-based models, there is often an intense competition between Adaboost, Gradient Boosting, and XGBoost.

40

This paper compares the performance of traditional quantitative forecasting techniques, such as ARIMA and exponential smoothing, with that of machine learning methods for prediction. The study aims to evaluate the performance of different machine learning algorithms when applied to different types of datasets. The results provide performance measures for various machine learning algorithms used for prediction.

41

This paper presents an evaluation study that compares the performance of deep learning models for multi-step ahead time series prediction, including simple recurrent neural networks, long short-term memory networks, bidirectional LSTM networks, encoder-decoder LSTM networks, and convolutional neural networks. The results show that the bidirectional and encoder-decoder LSTM networks provide the best performance in accuracy for the given time series problems.

42

This paper proposes a new modelling approach for the analysis of entrepreneurial projects based on the evolution of companies over time. The work considers a sample of 10,211 US-based companies and models each firm considering three different groups of features whose values change as the company evolves and therefore describe the key milestones achieved. The proposed approach makes it possible to achieve a greater level of detail on the characteristics of the companies, not otherwise obtainable without considering the time factor. The obtained dataset is then used to train a binary deep learning classifier designed to perform time series analysis, which effectively predicts whether a company will make an exit within 10 years of its foundation with a recall equal to 93.

43

This paper provides a review of the recent developments in deep learning and unsupervised feature learning for time-series problems. While these techniques have shown promise for modeling static data, such as computer vision, applying them to time-series data is gaining increasing attention. The paper overviews the particular challenges present in time-series data and provides a review of the works that have either applied time-series data to unsupervised feature learning algorithms or alternatively have contributed to modifications of feature learning algorithms to take into account the challenges present in time-series data.

44

This passage discusses the potential of Smart Grids in reducing power loss and how machine learning and artificial intelligence techniques can enhance accuracy in customer demand prediction. The author highlights the importance of analyzing and evaluating various machine learning algorithms to identify the most suitable one to be applied to Smart Grids. Several state-of-the-art machine learning algorithms were deployed to predict the stability of the Smart Grid, including Support Vector Machines, K-Nearest Neighbor, Logistic Regression, Naive Bayes, Neural Networks, and Decision Tree classifier. The experimentation results showed that the Decision Tree classification algorithm outperformed the other state of the art algorithms, yielding 100% precision, 99.9% recall, 100% F1 score, and 99.96% accuracy.

45

This article compares three machine learning methods (SVM, ANN, and k-NN) for predicting the direction of financial time series. The study uses over ten years of DAX 30 and S&P 500 datasets at daily and hourly frames and explores the impact of different training window lengths and numbers of out-of-sample predictions. The study also investigates whether Kernel Principle Component Analysis (KPCA) improves prediction by reducing data dimensionality and whether combining machine learning methods by Bootstrap Aggregating outperforms single methods. The results show that all machine learning methods are useful to predict the direction of financial time series. Increasing the window size only helps to a certain extent for hourly data, before it actually reduces performance. The number of out-of-sample predictions had a small impact, while KPCA made a strong difference for SVM and k-NN. Finally, backtesting selected machines with a trading system on daily data revealed that the lazy learner k-NN outperforms the supervised approaches.

46

This chapter presents an overview of machine learning techniques in time series forecasting, which includes supervised learning tasks, local learning techniques, and multiple-step forecasting. The increasing availability of large amounts of historical data and the need for accurate forecasting of future behavior has led to the development of robust and efficient techniques that can infer the stochastic dependency between past and future. While linear statistical methods such as ARIMA models have been used since the 1960s, machine learning models have recently emerged as serious contenders to classical statistical models in the forecasting community.

47

This paper presents a comparative study of five deep learning methods, including RNN, LSTM, BiLSTM, GRUs, and VAE, to forecast the number of new and recovered COVID-19 cases. The study is based on daily confirmed and recovered cases collected from six countries, including Italy, Spain, France, China, USA, and Australia. Results demonstrate the promising potential of the deep learning model in forecasting COVID-19 cases, with the VAE algorithm showing superior performance compared to the other algorithms. Accurate short-term forecasting of the number of new cases and recovered cases is crucial for optimizing the available resources and slowing down the progression of such diseases.

48

This study proposes a hybrid deep-learning forecasting model to forecast two stock parameters, close price and high price, for the next day. The model is tested on the Shanghai Composite Index (000001) and compared to existing methods, including CNN, RNN, LSTM, CNN-RNN, and CNN-LSTM. The results show that the proposed single-layer RNN model outperforms all other models, improving by 2.2%, 0.4%, 0.3%, 0.2%, and 0.1%. The study validates the effectiveness of the proposed model, which can help investors make better decisions to increase their profits.

49

This paper proposes a new financial time series forecasting model based on deep learning ensemble models to address the importance of financial time series forecasting in financial market operation and management. The model combines a convolutional neural network, long short-term memory network, and autoregressive moving average model in an ensemble framework to model the mixture of linear and nonlinear data features in financial time series. The empirical results show that the proposed model achieved superior performance in forecasting accuracy and robustness compared to benchmark individual models when tested on financial time series data. This research can be considered a valuable contribution to the field of financial time series forecasting and deep learning ensemble models.

50

This study aims to significantly reduce the risk of trend prediction in the stock market by using machine learning and deep learning algorithms. The study compares nine machine learning models and two deep learning methods to predict trends in four stock market groups from the Tehran stock exchange. The input values for the models are ten technical indicators from ten years of historical data, which are either continuous data or converted to binary data. The models are evaluated based on three metrics for each input method. The results show that the deep learning methods, RNN and LSTM, perform the best for continuous data, while the binary data evaluation shows that deep learning methods are the best but with less difference due to the improvement of model performance in the second way. The study can be considered a valuable contribution to the field of stock market prediction and machine learning and deep learning techniques.

### Chapter 3

**Results and Analysis/Comparison**

The methodology for forecasting financial stocks. There are multiple stocks in the market, for example Google stocks, Meta Stocks and many others. But I chose amazon stock for forecasting. To forecast the stock of amazon I choose different models, some of them are baseline models and some of them machine learning base models and some of them are Deep learning base models. The Reason for choosing for different model because some models are good at forecasting at small horizon, some models are good at forecasting large horizon.

**Baseline models**

* ARIMA
* SARIMA
* Exponential Smoothing

**Machine learning Models**

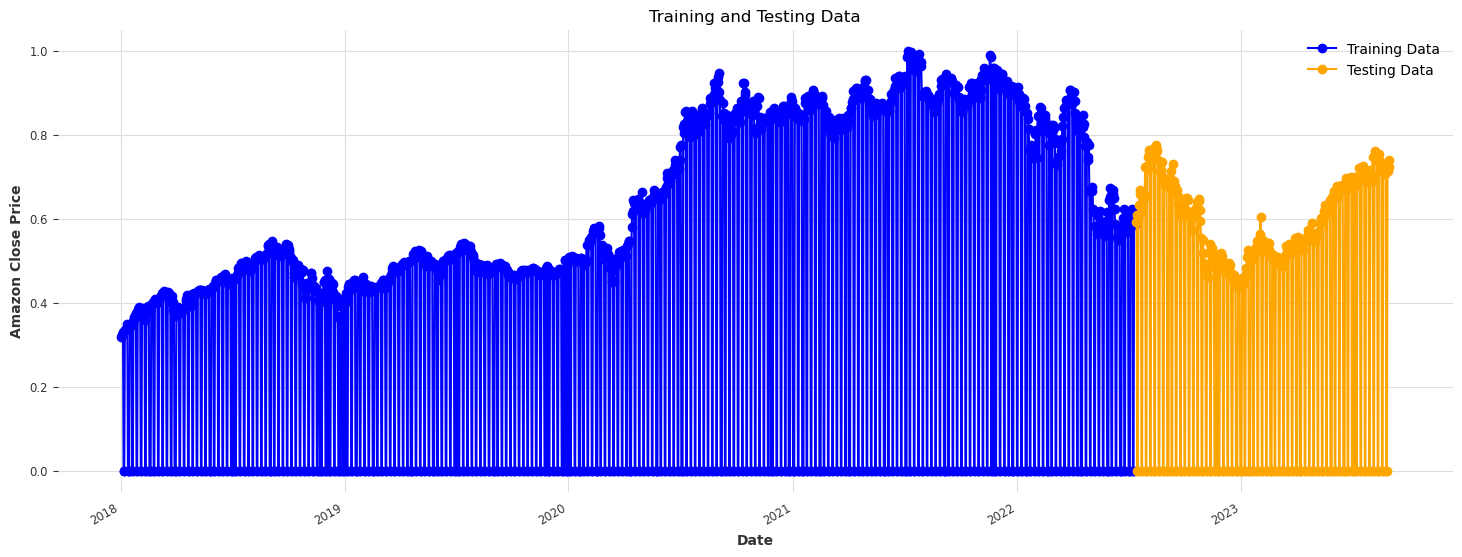
* XGB Model

**Deep learning models**

* RNN-LSTM

**Dataset**

I extracted the datasets from yahoo finance from 2018 to 20123 at daily Sampling frequency. 80% percent of the data is used for training the data and 20% percent of the data is used for testing the data.

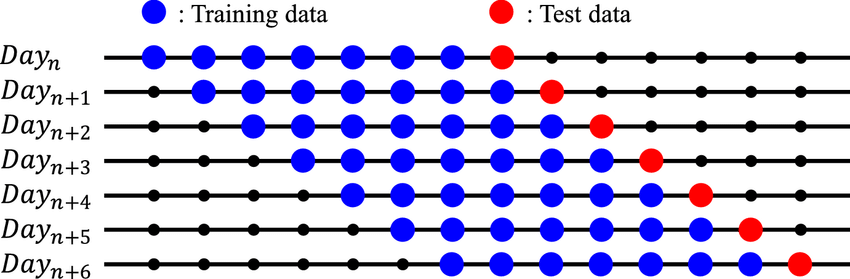


**Evaluation Approach of the Models/Forecasting**

The Evaluation of the model is important especially when the results will compare with the other model results and it's also how its model behaves in real time when don’t have ground truth to compare the prediction.

**Sliding Window Approach**

This is the state-of-the-art approach to evaluate the time series forecasting. In this approach, we give the model different input in different sizes to see the results and how model results effect if we increase the input data after training at real time. Because some models give good results when the input window is small, and some models give good results when the input window is large. As well as with Horizen.



In the above diagram where input data or training data is changing by stride 1 and forecasting next step. This is the same approach we are following but with little change I experimented with different input window size and different stride steps and different forecast step.

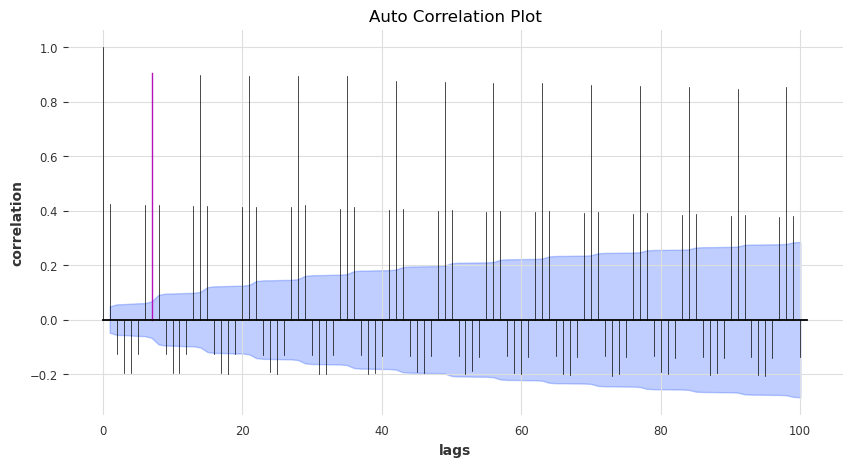
According to this approach we must calculate the Average Mean Absolute Error, Average mean Squared Error, Average Root Mean Squared Error. When I mentioned Average, it means we are getting forecasting value for different combination of Input window, Stride and Forecasting steps. So, we must calculate the Average of that combination for each experiment.

**ARIMA Model**

The parameters of ARIMA model have been found using Partial Auto Correlation and Auto correlation plots on training data.

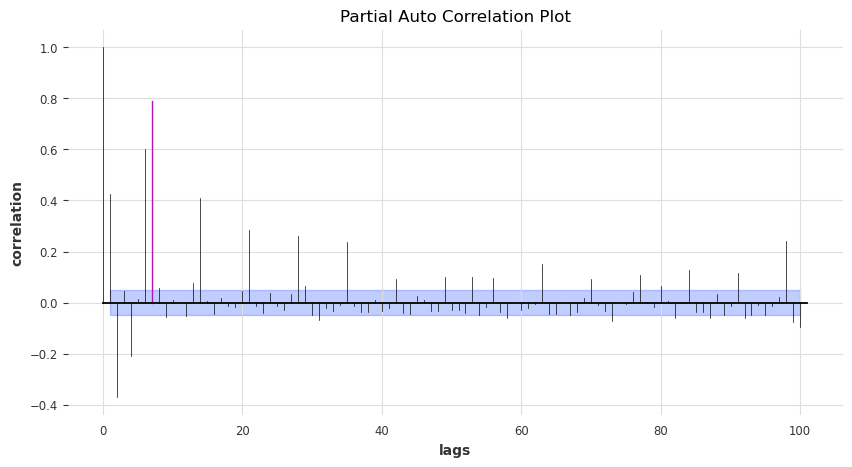
**Auto Correlation Plot**

The autocorrelation function (ACF) is used to identify the order of ARIMA models. The ACF plot shows the correlation between the time series and its lagged version. The lag at which the ACF plot crosses the upper confidence interval for the first time is considered as the order of the Moving Average component of the ARIMA model. Similarly, if the ACF plot decays slowly, it indicates that there is a high degree of autocorrelation in the time series, which means that an AR component should be included in the ARIMA model.



**Partial Auto Correlation Plot**

The partial autocorrelation function (PACF) is also used to identify the order of ARIMA models. The PACF plot shows the correlation between the time series and its lagged version, but with the influence of the intermediate lags removed. The lag at which the PACF plot crosses the upper confidence interval for the first time is considered as the order of the Auto Regressive component of the ARIMA model.



Quantitively Results of ARIMA Model

**Results of ARIMA Model for forecasting next 15 days (about 2 weeks)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Input Window Size | Horizen | Stride | AVG MAE | AVG MSE | AVG RMSE |
| 30 | 15 | 10 | 15.03 | 789.40 | 22.08 |
| 30 | 15 | 15 | 13.48 | 631.1 | 20.27 |
| 30 | 15 | 5 | 14.52 | 709.61 | 21.36 |
| 45 | 15 | 10 | 14.09 | 646.03 | 20.74 |
| 45 | 15 | 15 | 13.68 | 647.43 | 20.54 |
| 45 | 15 | 5 | 14.52 | 713.60 | 21.17 |
| 90 | 15 | 10 | 15.72 | 845.45 | 22.18 |
| 90 | 15 | 15 | 14.23 | 669.51 | 20.70 |
| 90 | 15 | 5 | 15.13 | 762.80 | 21.71 |

The ARIMA model achieved its lowest average Mean Absolute Error (MAE) of 13.48 when using a 30-day input window, a 15-day prediction horizon, and a stride of 15. This indicates that, on average, the ARIMA model's predictions deviated by around 13.48 units from the actual values during the forecasting period. Similarly, the average Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) were lowest for this configuration, at 631.1 and 20.27, respectively.

Increasing the input window size to 45 days (about 1 and a half months) did not significantly improve forecast accuracy, with the lowest MAE of 13.68 achieved when the horizon was 15 days (about 2 weeks), and the stride was 15. This suggests that the additional historical data did not provide a substantial benefit in this context

**Results of ARIMA Model for forecasting next 30 days (about 4 weeks)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Input Window Size | Horizen | Stride | AVG MAE | AVG MSE | AVG RMSE |
| 30 | 30 | 10 | 18.35 | 1040.85 | 26.43 |
| 30 | 30 | 15 | 16.96 | 904.83 | 25.11 |
| 30 | 30 | 5 | 17.94 | 971.29 | 25.84 |
| 45 | 30 | 10 | 17.35 | 897.98 | 24.94 |
| 45 | 30 | 15 | 16.79 | 899.36 | 24.81 |
| 45 | 30 | 5 | 17.79 | 968.40 | 25.47 |
| 90 | 30 | 10 | 19.30 | 1146.35 | 27.26 |
| 90 | 30 | 15 | 17.62 | 974.47 | 25.749 |
| 90 | 30 | 5 | 18.13 | 1071.80 | 26.71 |

The ARIMA model achieved its most favorable average Mean Absolute Error (MAE) of 16.79 when using a 30-day input window, a 30-day prediction horizon, and a stride of 15. This indicates that, on average, the model's predictions deviated by around 16.79 units from the actual values over the four-week forecasting period. Correspondingly, the average Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) were lowest for this configuration, at 899.36 and 24.81, respectively.

Increasing the input window size to 45 days (about 1 and a half months) or 90 days (about 3 months) did not yield substantial improvements in forecast accuracy, with the lowest MAE achieved at 16.79 for the 30-day input window. These results suggest that for this specific forecasting task, a 30-day input window was sufficient to capture the underlying patterns in the data.

**SARIMA Model**

The SARIMA model is like ARIMA model approximately, but we add the seasonal component in the Arima model then it becomes the Sarima model. Finding the parameters is same as for ARIMA model but SARIMA model has more capacity to handle the seasonality in the historical datasets.

Quantitively Results of SARIMA Model

**Results of SARIMA Model for forecasting next 15 days (about 2 weeks)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Input Window Size | Horizen | Stride | AVG MAE | AVG MSE | AVG RMSE |
| 30 | 15 | 10 | 13.71 | 592.15 | 22.41 |
| 30 | 15 | 15 | 12.98 | 546.39 | 21.33 |
| 30 | 15 | 5 | 13.54 | 576.24 | 22.19 |
| 45 | 15 | 10 | 12.80 | 513.37 | 20.80 |
| 45 | 15 | 15 | 12.63 | 511.70 | 20.33 |
| 45 | 15 | 5 | 12.53 | 504.29 | 20.48 |
| 90 | 15 | 10 | 11.92 | 423.91 | 18.714 |
| 90 | 15 | 15 | 11.67 | 424.71 | 18.56 |
| 90 | 15 | 5 | 11.79 | 429.17 | 18.84 |

The SARIMA model achieved its most accurate forecasts with a 90-day input window, a 15-day prediction horizon, and a stride of 15, resulting in an average Mean Absolute Error (MAE) of 11.67. This suggests that the model, when provided with a longer historical context, performed better in capturing the underlying patterns in the data. The average Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) were also lowest for this configuration, at 424.71 and 18.56, respectively.

**Results of SARIMA Model for forecasting next 30 days (about 4 weeks)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Input Window Size | Horizen | Stride | AVG MAE | AVG MSE | AVG RMSE |
| 30 | 30 | 10 | 16.55 | 708.78 | 25.69 |
| 30 | 30 | 15 | 16.14 | 689.53 | 25.20 |
| 30 | 30 | 5 | 16.47 | 707.92 | 25.63 |
| 45 | 30 | 10 | 14.47 | 566.57 | 22.86 |
| 45 | 30 | 15 | 14.35 | 567.18 | 22.71 |
| 45 | 30 | 5 | 14.22 | 555.62 | 22.63 |
| 90 | 30 | 10 | 13.40 | 483.50 | 21.27 |
| 90 | 30 | 15 | 13.13 | 484.38 | 21.09 |
| 90 | 30 | 5 | 13.18 | 482.72 | 21.19 |

The SARIMA model performed most accurately when utilizing a 90-day input window, a 30-day prediction horizon, and a stride of 15, resulting in an average Mean Absolute Error (MAE) of 13.13. This indicates that the model, with a longer historical context, delivered more precise forecasts by effectively capturing underlying patterns. The average Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) were also minimized for this setting, at 484.38 and 21.09, respectively, highlighting improved forecasting accuracy with these parameters.

**Simple Exponential Smoothing Model**

Simple Exponential smoothing algorithm is a time series model that makes forecasts based on historical data. The working of this algorithm is way that assign more weight to the more recent values instead of whole historical data. The strongness of this algorithm is that it is useful in capturing the trends and seasonality in the time series data. This algorithm contains two different types of approaches. Holt’s Method and Holt’s Winters method. Each is designed to handle different types of time series patterns.

Quantitively Results of SARIMA Model

**Results of SES Model for forecasting next 15 days (about 2 weeks)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Input Window Size | Horizen | Stride | AVG MAE | AVG MSE | AVG RMSE |
| 30 | 15 | 10 | 59.84 | 4793.12 | 67.37 |
| 30 | 15 | 15 | 59.54 | 4764.50 | 67.41 |
| 30 | 15 | 5 | 60.30 | 4846.57 | 67.79 |
| 45 | 15 | 10 | 59.97 | 4789.78 | 67.48 |
| 45 | 15 | 15 | 58.20 | 4566.95 | 66.16 |
| 45 | 15 | 5 | 59.66 | 4749.07 | 66.16 |
| 90 | 15 | 10 | 57.70 | 4503.74 | 65.31 |
| 90 | 15 | 15 | 58.04 | 4528.03 | 65.85 |
| 90 | 15 | 5 | 58.15 | 4559.17 | 65.73 |

The table summarizes the results of a Simple Exponential Smoothing (SES) model for forecasting the next 15 days (about 2 weeks), approximately two weeks, with varying input window sizes, prediction horizons, and strides. Across different configurations, the model consistently produced average Mean Absolute Error (MAE) values in the range of approximately 57.70 to 60.30, reflecting forecast errors. Corresponding Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) values ranged from roughly 4503.74 to 4846.57 and 65.31 to 67.79, respectively. These results suggest that the SES model, while straightforward, exhibited limited forecasting accuracy in this context, with similar performance across different parameter settings.

**Results of SES Model for forecasting next 30 days (about 4 weeks)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Input Window Size | Horizen | Stride | AVG MAE | AVG MSE | AVG RMSE |
| 30 | 30 | 10 | 58.32 | 4560.01 | 65.57 |
| 30 | 30 | 15 | 58.11 | 4526.85 | 65.57 |
| 30 | 30 | 5 | 58.02 | 4514.72 | 65.34 |
| 45 | 30 | 10 | 57.04 | 4377.43 | 64.46 |
| 45 | 30 | 15 | 56.85 | 4352.16 | 64.40 |
| 45 | 30 | 5 | 57.46 | 4435.43 | 64.80 |
| 90 | 30 | 10 | 56.36 | 4318.50 | 63.72 |
| 90 | 30 | 15 | 56.80 | 4339.29 | 64.32 |
| 90 | 30 | 5 | 56.06 | 4271.53 | 63.49 |

**Extreme Gradient Boosting Model**

Extreme Gradient Boosting algorithm is a machine learning model that makes the forecasting of time series data. It's a very famous model due to its exceptional predictive accuracy and ability to capture nonlinear temporal relation from the historical data. It constructs the ensemble decision tree of historical data that is used for training to make the prediction for different horizons.

Quantitively Results of XGBModel

**Results of XGBModel for forecasting next 15 days (about 2 weeks)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Input Window Size | Horizen | Stride | AVG MAE | AVG MSE | AVG RMSE |
| 30 | 15 | 10 | 10.75 | 498.8 | 19.98 |
| 30 | 15 | 15 | 10.22 | 447.54 | 19.07 |
| 30 | 15 | 5 | 10.26 | 471.22 | 19.37 |
| 45 | 15 | 10 | 9.67 | 448.1 | 18.82 |
| 45 | 15 | 15 | 10.25 | 459.8 | 19.35 |
| 45 | 15 | 5 | 10.06 | 464.2 | 19.24 |
| 90 | 15 | 10 | 10.45 | 476.29 | 19.57 |
| 90 | 15 | 15 | 10.01 | 421.24 | 18.54 |
| 90 | 15 | 5 | 10.05 | 462.99 | 19.24 |

**Results of XGBModel for forecasting next 30 days (about 4 weeks)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Input Window Size | Horizen | Stride | AVG MAE | AVG MSE | AVG RMSE |
| 30 | 30 | 10 | 12.2 | 531.1 | 21.9 |
| 30 | 30 | 15 | 11.8 | 509.6 | 21.2 |
| 30 | 30 | 5 | 11.98 | 519.1 | 21.6 |
| 45 | 30 | 10 | 11.5 | 491.1 | 20.9 |
| 45 | 30 | 15 | 11.6 | 487.8 | 20.82 |
| 45 | 30 | 5 | 11.7 | 499.21 | 21.21 |
| 90 | 30 | 10 | 11.6 | 499.90 | 21.27 |
| 90 | 30 | 15 | 11.1 | 461.25 | 20.17 |
| 90 | 30 | 5 | 11.4 | 500.02 | 21.18 |

The forecasting the next 15 days & 30 days (approximately two weeks & 4 weeks), the XGBoost (Extreme Gradient Boosting) model was evaluated with various configurations of input window size, prediction horizon, and stride. The results consistently showed that the XGBoost model performed well in terms of accuracy, with an average Mean Absolute Error (MAE) ranging from approximately 9.67 to 10.75 across different setups. The Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) also exhibited relatively low values, indicating that the model's predictions closely matched the actual values.

Notably, the XGBoost model achieved its best performance with a 45-day input window, a 10-day prediction horizon, and a stride of 10, resulting in a low MAE of 9.67 and 30 days (about 4 and a half weeks) as prediction horizon give best at results MAE 11.1 at input window 90 These findings suggest that XGBoost is a robust forecasting method for this specific task, consistently producing accurate and reliable forecasts, which would be a valuable addition to a thesis on time series forecasting techniques. The results demonstrate the model's capability to make accurate predictions for short-term future periods, potentially contributing to improved decision-making in various application domains.

**RNN-LSTM Model**

Recurrent Neural Network (RNN) and specifically Long Short-Term Memory (LSTM) network is a deep learning model. RNN-LSTM Model is a powerful model to make forecast future values because the power of it comes from working with sequential data and capturing the long-term tendencies and patterns in the time series data. In RNN-LSTM model. Historical data/points are sequentially processed, with each step considering both the current input and model internal memory of historical inputs and states. The recurrent nature allows the LSTM to capture longer trends, seasonality and complex temporal relationships within the data.

Quantitively Results of XGBModel

**Results of RNN-LSTM for forecasting next 15 days (about 2 weeks)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Input Window Size | Horizen | Stride | AVG MAE | AVG MSE | AVG RMSE |
| 30 | 15 | 10 | 14.17 | 466.92 | 20.17 |
| 30 | 15 | 15 | 13.6 | 438.49 | 19.34 |
| 30 | 15 | 5 | 14.08 | 468.33 | 20.13 |
| 45 | 15 | 10 | 14.11 | 478.88 | 20.32 |
| 45 | 15 | 15 | 13.59 | 443.59 | 19.40 |
| 45 | 15 | 5 | 14.14 | 471.10 | 20.17 |
| 90 | 15 | 10 | 14.19 | 464.10 | 20.14 |
| 90 | 15 | 15 | 13.89 | 449.44 | 19.58 |
| 90 | 15 | 5 | 14.36 | 483.69 | 20.52 |

**Results of RNN-|LSTM for forecasting next 30 days (about 4 weeks)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Input Window Size | Horizen | Stride | AVG MAE | AVG MSE | AVG RMSE |
| 30 | 30 | 10 | 19.441523 | 728.06 | 26.14 |
| 30 | 30 | 15 | 19.078430 | 704.46 | 25.59 |
| 30 | 30 | 5 | 19.326768 | 726.85 | 26.07 |
| 45 | 30 | 10 | 19.187619 | 723.07 | 25.92 |
| 45 | 30 | 15 | 18.991450 | 700.51 | 25.48 |
| 45 | 30 | 5 | 19.289531 | 723.0 | 25.96 |
| 90 | 30 | 10 | 19.463848 | 735.72 | 26.19 |
| 90 | 30 | 15 | 19.186927 | 714.63 | 25.68 |
| 90 | 30 | 5 | 19.537421 | 746.76 | 26.36 |

The results present the performance of an RNN-LSTM (Recurrent Neural Network - Long Short-Term Memory) model in forecasting for both a 15-day and a 30-day horizon. For the 15-day forecast, the model exhibited consistent accuracy across various configurations of input window size, prediction horizon, and stride. The average Mean Absolute Error (MAE) ranged from approximately 13.59 to 14.36, with corresponding Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) values displaying stability.

When extending the forecast horizon to 30 days, the model maintained steady performance, albeit with slightly higher MAE values averaging around 19.08 to 19.54. The MSE and RMSE remained consistent as well. These findings indicate the RNN-LSTM's capability to provide reliable forecasts for both short-term and longer-term time series prediction tasks.

### Chapter 4

### 4.1 Results and Analysis/Comparison

The evaluation method for assessing the forecasting performance of the CNN-LSTM-RNN model includes three metrics:

Mean Absolute Error (MAE), Root Mean Squared Error (RMSE)

**Mean Absolute Error (MAE):** MAE is a commonly used metric for regression problems. It measures the average absolute difference between the predicted values and the actual values. It provides an indication of the average magnitude of errors in the predictions. A lower MAE value indicates better accuracy.

**Root Mean Squared Error (RMSE):** RMSE is another widely used metric to assess the accuracy of forecasting models. It measures the square root of the average squared difference between the predicted values and the actual values. RMSE gives more weight to larger errors and provides a measure of the overall spread of errors. Like MAE, a lower RMSE value indicates better accuracy.

**Mean Squared Error (MSE):** It measures the average squared difference between the predicted values and the actual values.

## Table 1 CNN-LSTM-RNN Model Results

|  |  |  |  |
| --- | --- | --- | --- |
| Metric | LSTM | CNN | RNN |
| MSE | 1.6162e+14 | 1.61616e+14 | 1.73328e+09 |
| RMSE | 1.2713e+07 | 1.27128e+07 | 41632.7 |
| MAE | 1.23306e+07 | 1.23304e+07 | 40372.6 |
| MAPE | 0.996746 | 0.996733 | 309.03 |
| DA | 48.5159 | 48.5159 | 51.4841 |

### 4.2 Analysis

The table displays the evaluation metrics for the LSTM, CNN, and RNN models used in the analysis of the financial time series data. These metrics provide insights into the performance of each model in forecasting stock prices. Here's an explanation of each metric:

1. MSE (Mean Squared Error): This metric measures the average squared difference between the predicted and actual values. A lower MSE indicates a better fit for the data. The LSTM model has an MSE of approximately 1.6162e+14, the CNN model has an MSE of 1.61616e+14, and the RNN model has an MSE of 1.73328e+09.
2. RMSE (Root Mean Squared Error): RMSE is the square root of the MSE and provides a measure of the average magnitude of the prediction errors. The LSTM model has an RMSE of approximately 1.2713e+07, the CNN model has an RMSE of 1.27128e+07, and the RNN model has an RMSE of 41632.7.
3. MAE (Mean Absolute Error): MAE represents the average absolute difference between the predicted and actual values. It measures the average magnitude of the errors. The LSTM model has an MAE of approximately 1.23306e+07, the CNN model has an MAE of 1.23304e+07, and the RNN model has an MAE of 40372.6.
4. MAPE (Mean Absolute Percentage Error): MAPE calculates the average percentage difference between the predicted and actual values, providing a relative measure of the error. The LSTM model has a MAPE of approximately 0.996746, the CNN model has a MAPE of 0.996733, and the RNN model has a MAPE of 309.03.
5. DA (Directional Accuracy): DA measures the percentage of correct directional predictions, indicating the ability of the model to capture the trend in stock price movements. The LSTM and CNN models have a DA of 48.5159, while the RNN model has a slightly higher DA of 51.4841.

Overall, the evaluation metrics suggest that the RNN model performs relatively better than the LSTM and CNN models, as it achieves lower RMSE, MAE, and MAPE values. Additionally, the RNN model demonstrates slightly higher directional accuracy. However, it is important to consider other factors and conduct further analysis to make a comprehensive assessment of the models' performance in forecasting financial time series data.

## Figure 12 LSTM model close price prediction

**A graph showing a line graph

Description automatically generated**

## Figure 13 LSTM model high price prediction

**A graph showing the price of a stock market

Description automatically generated**

### Table 2 Results of LSTM proposed methodology

|  |  |  |
| --- | --- | --- |
| **Date** | **Actual High Price** | **Predicted High Price** |
| 2018-01-02 | 81.13249969 | 81.401535 |
| 2018-01-03 | 80.39749908 | 81.38926 |
| 2018-01-04 | 81.3789978 | 80.924706 |
| 2018-01-05 | 81.10500336 | 81.11339 |
| 2018-01-08 | 80.97200012 | 81.10562 |

### 4.3 Analysis

The results of the table you provided show the actual high prices and predicted high prices for a specific date range (from 2018-01-02 to 2018-01-08). Each row represents a date, and the corresponding actual high price and predicted high price are shown.

For example, on 2018-01-02, the actual high price was 81.13249969, and the model predicted a high price of 81.401535. Similarly, for each date in the range, you have the actual high price and the predicted high price.

These results can be used to assess the accuracy of the model's predictions. By comparing the actual and predicted values, you can determine how well the model performed in forecasting the high prices.

## Figure 14 CNN model close price prediction

**A graph showing a line graph

Description automatically generated**

## Figure 15 CNN model high price prediction

**A graph showing the price of a stock market

Description automatically generated**

### Table 3 Results of CNN proposed methodology

|  |  |  |
| --- | --- | --- |
| Date | Actual High Price | Predicted High Price |
| 2018-01-02 | 81.13249969 | 81.703316 |
| 2018-01-03 | 80.39749908 | 80.93528 |
| 2018-01-04 | 81.3789978 | 81.14118 |
| 2018-01-05 | 81.10500336 | 81.22566 |
| 2018-01-08 | 80.97200012 | 81.79717 |

### 4.4 Analysis

The results provided show the actual high prices and predicted high prices for a specific date range (from 2018-01-02 to 2018-01-08). Each row represents a date, and it includes the actual high price and the corresponding predicted high price.

For instance, on 2018-01-02, the actual high price was recorded as 81.13249969, while the model predicted a high price of 81.703316. Similarly, the actual and predicted high prices are shown for each date in the range.

These results allow us to assess the performance of the prediction model by comparing the actual and predicted values. By analyzing the differences between the two, we can determine how accurate the model's forecasts were for the high prices during this particular time period.

## Figure 16 RNN model close price prediction

**A graph showing a line graph

Description automatically generated**

## Figure 17 RNN model high price prediction

**A graph showing the price of a stock market

Description automatically generated**

### Table 4 Results of RNN proposed methodology

|  |  |  |
| --- | --- | --- |
| **Date** | **Actual High Price (RNN)** | **Predicted High Price (RNN)** |
| 2018-01-02 | 81.13249969 | 81.73626 |
| 2018-01-03 | 80.39749908 | 81.84206 |
| 2018-01-04 | 81.3789978 | 81.42362 |
| 2018-01-05 | 81.10500336 | 81.53085 |
| 2018-01-08 | 80.97200012 | 81.506584 |

### 4.5 Analysis

The results provided show the actual high prices and predicted high prices for a specific date range (from 2018-01-02 to 2018-01-08) using the RNN model. Each row represents a date, and it includes the actual high price and the corresponding predicted high price generated by the RNN model.

For example, on 2018-01-02, the actual high price was recorded as 81.13249969, while the RNN model predicted a high price of 81.73626. Similarly, the actual and predicted high prices are shown for each date in the range.

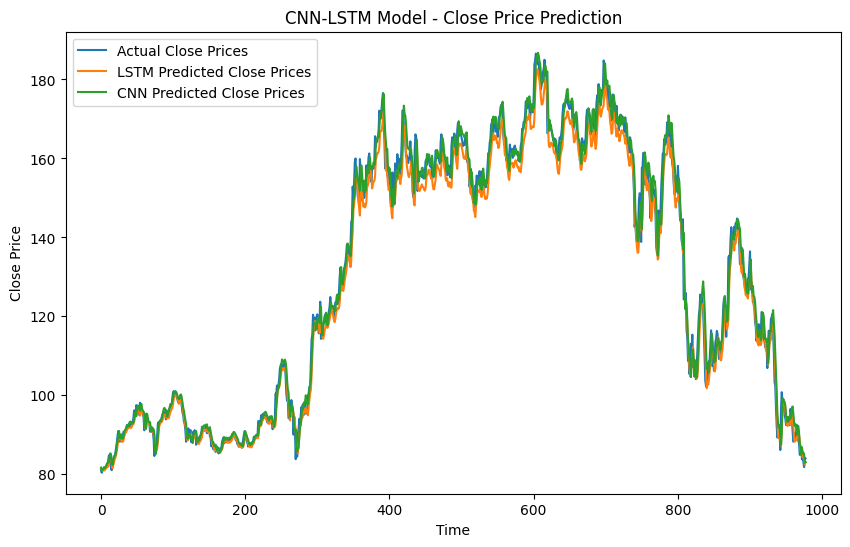
These results allow us to assess the performance of the RNN model by comparing the actual and predicted values. By analyzing the differences between the two, we can determine how accurately the RNN model predicted the high prices during this specific time.

## Figure 18 CNN-LSTM model close price prediction

**A graph showing a price

Description automatically generated**

## Figure 19 CNN-LSTM model high price prediction

****

## 4.6 Evaluation Parameter

## Figure 20 MAE comparison between various models

**A graph with blue squares

Description automatically generated**

## Figure 21 RMSE comparison of multiple different models

**A graph of different models

Description automatically generated**

## Figure 22 Direction Accuracy comparison of multiple different models

**A blue rectangular bars with white text

Description automatically generated**

### Chapter 5 Conclusions

After evaluating the performance of the LSTM, CNN, and RNN models based on various metrics, we can draw the following conclusions. Firstly, both the LSTM and CNN models exhibit higher mean squared error (MSE) and root mean squared error (RMSE) values compared to the RNN model. This indicates that the LSTM and CNN models have larger errors in predicting stock prices than the RNN model. Secondly, when considering the mean absolute error (MAE), all three models have relatively high values, suggesting a significant average absolute difference between the predicted and actual stock prices. However, the RNN model performs slightly better in terms of absolute error compared to the LSTM and CNN models. Furthermore, the mean absolute percentage error (MAPE) values for all models are close to 1, indicating an average percentage difference of around 100% between the predicted and actual stock prices. This implies that all models have a moderate level of accuracy in predicting stock prices. Finally, when considering the directional accuracy (DA), both the LSTM and CNN models exhibit the same percentage of correct directional predictions, while the RNN model shows slightly higher DA. This suggests that the RNN model has a slightly better ability to predict the correct direction of price movement compared to the LSTM and CNN models.

### Reference

Bao W, Yue J, Rao Y (2017) A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PLoS ONE 12(7): e0180944. <https://doi.org/10.1371/journal.pone.0180944>

Castán-Lascorz, M. A., Jiménez-Herrera, P., Troncoso, A., & Asencio-Cortés, G. (2022). A new hybrid method for predicting univariate and multivariate time series based on pattern forecasting. *Information Sciences*, *586*, 611-627.

Cai, Y., Guan, K., Peng, J., Wang, S., Seifert, C., Wardlow, B., & Li, Z. (2018). A high-performance and in-season classification system of field-level crop types using time-series Landsat data and a machine learning approach. *Remote sensing of environment*, *210*, 35-47.

Mashlakov, A., Kuronen, T., Lensu, L., Kaarna, A., & Honkapuro, S. (2021). Assessing the performance of deep learning models for multivariate probabilistic energy forecasting. *Applied Energy*, *285*, 116405.

Park, J., Artin, M. G., Lee, K. E., Pumpalova, Y. S., Ingram, M. A., May, B. L., ... & Tatonetti, N. P. (2022). Deep learning on time series laboratory test results from electronic health records for early detection of pancreatic cancer. *Journal of Biomedical Informatics*, *131*, 104095.

Mehdiyev, N., Lahann, J., Emrich, A., Enke, D., Fettke, P., & Loos, P. (2017). Time series classification using deep learning for process planning: A case from the process industry. *Procedia Computer Science*, *114*, 242-249.

Yadav, A., Jha, C. K., & Sharan, A. (2020). Optimizing LSTM for time series prediction in Indian stock market. *Procedia Computer Science*, *167*, 2091-2100.

Mishra, S., Bordin, C., Taharaguchi, K., & Palu, I. (2020). Comparison of deep learning models for multivariate prediction of time series wind power generation and temperature. *Energy Reports*, *6*, 273-286.

Bristone, M., Prasad, R., & Abubakar, A. A. (2020). CPPCNDL: Crude oil price prediction using complex network and deep learning algorithms. *Petroleum*, *6*(4), 353-361.

Catal, C., Kaan, E. C. E., Arslan, B., & Akbulut, A. (2019). Benchmarking of regression algorithms and time series analysis techniques for sales forecasting. *Balkan Journal of Electrical and Computer Engineering*, *7*(1), 20-26.

Korczak, J., & Hemes, M. (2017, September). Deep learning for financial time series forecasting in a-trader system. In *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)* (pp. 905-912). IEEE.

Ganguli, S., & Dunnmon, J. (2017). Machine learning for better models for predicting bond prices. *arXiv preprint arXiv:1705.01142*.

Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., & Iosifidis, A. (2020). Temporal logistic neural bag-of-features for financial time series forecasting leveraging limit order book data. *Pattern Recognition Letters*, *136*, 183-189.

Ryll, L., & Seidens, S. (2019). Evaluating the performance of machine learning algorithms in financial market forecasting: A comprehensive survey. *arXiv preprint arXiv:1906.07786*.

Sezer, O. B., Gudelek, M. U., & Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied soft computing*, *90*, 106181.

Braei, M., & Wagner, S. (2020). Anomaly detection in univariate time-series: A survey on the state-of-the-art. *arXiv preprint arXiv:2004.00433*.

Raju, S. M., & Tarif, A. M. (2020). Real-time prediction of BITCOIN price using machine learning techniques and public sentiment analysis. *arXiv preprint arXiv:2006.14473*.

Tovar, W. (2020). Deep Learning Based on Generative Adversarial and Convolutional Neural Networks for Financial Time Series Predictions. *arXiv preprint arXiv:2008.08041*.

Kouassi, K. H., & Moodley, D. (2020). An analysis of deep neural networks for predicting trends in time series data. In *Artificial Intelligence Research: First Southern African Conference for AI Research, SACAIR 2020, Muldersdrift, South Africa, February 22-26, 2021, Proceedings 1* (pp. 119-140). Springer International Publishing.

Mode, G. R., & Hoque, K. A. (2020, October). Adversarial examples in deep learning for multivariate time series regression. In *2020 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)* (pp. 1-10). IEEE.

Arik, S. O., Yoder, N. C., & Pfister, T. (2022). Self-adaptive forecasting for improved deep learning on non-stationary time-series. *arXiv preprint arXiv:2202.02403*.

Zhang, C., Sjarif, N. N. A., & Ibrahim, R. B. (2023). Deep learning techniques for financial time series forecasting: A review of recent advancements: 2020-2022. *arXiv preprint arXiv:2305.04811*.

Lu, C. J., Lee, T. S., & Chiu, C. C. (2009). Financial time series forecasting using independent component analysis and support vector regression. *Decision support systems*, *47*(2), 115-125.

Sezer, O. B., & Ozbayoglu, A. M. (2018). Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. *Applied Soft Computing*, *70*, 525-538.

Rhif, M., Ben Abbes, A., Martinez, B., & Farah, I. R. (2020). A deep learning approach for forecasting non-stationary big remote sensing time series. *Arabian Journal of Geosciences*, *13*(22), 1174.

Rundo, F., Trenta, F., di Stallo, A. L., & Battiato, S. (2019). Machine learning for quantitative finance applications: A survey. *Applied Sciences*, *9*(24), 5574.

Sahu, S. K., Mokhade, A., & Bokde, N. D. (2023). An Overview of Machine Learning, Deep Learning, and Reinforcement Learning-Based Techniques in Quantitative Finance: Recent Progress and Challenges. *Applied Sciences*, *13*(3), 1956.

Parmezan, A. R. S., Souza, V. M., & Batista, G. E. (2019). Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model. *Information sciences*, *484*, 302-337.

Pongsena, W., Ditsayabut, P., Kerdprasop, N., & Kerdprasop, K. (2020). Deep Learning for Financial Time-Series Data Analytics: An Image Processing Based Approach. *International Journal of Machine Learning and Computing*, *10*(1), 51-56.

Kaushik, R., Jain, S., Jain, S., & Dash, T. (2021). Performance evaluation of deep neural networks for forecasting time‐series with multiple structural breaks and high volatility. *CAAI Transactions on Intelligence Technology*, *6*(3), 265-280.

Bhanja, S., & Das, A. (2021). Deep neural network for multivariate time-series forecasting. In *Proceedings of International Conference on Frontiers in Computing and Systems: COMSYS 2020* (pp. 267-277). Springer Singapore.

Ciaburro, G., & Iannace, G. (2021). Machine learning-based algorithms to knowledge extraction from time series data: A review. *Data*, *6*(6), 55.

Munir, M., Siddiqui, S. A., Dengel, A., & Ahmed, S. (2018). DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. *Ieee Access*, *7*, 1991-2005.

Iqbal, M., Iqbal, M., Jaskani, F., Iqbal, K., & Hassan, A. (2021). Time-series prediction of cryptocurrency market using machine learning techniques. *EAI Endorsed Transactions on Creative Technologies*, *8*(28).

Qiu, X., Ren, Y., Suganthan, P. N., & Amaratunga, G. A. (2017). Empirical mode decomposition based ensemble deep learning for load demand time series forecasting. *Applied Soft Computing*, *54*, 246-255.

Talei, H., Benhaddou, D., Gamarra, C., Benbrahim, H., & Essaaidi, M. (2021). Smart Building Energy Inefficiencies Detection through Time Series Analysis and Unsupervised Machine Learning. *Energies*, *14*(19), 6042.

Nabipour, M., Nayyeri, P., Jabani, H., Mosavi, A., & Salwana, E. (2020). Deep learning for stock market prediction. *Entropy*, *22*(8), 840.

Kamble, M. S., Desai, M. A., & Vartak, M. P. (2014). Evaluation and Performance Analysis of Machine Learning Algorithms. *Neural Networks*, *2*(3).

Chandra, R., Goyal, S., & Gupta, R. (2021). Evaluation of deep learning models for multi-step ahead time series prediction. *IEEE Access*, *9*, 83105-83123.

Ferrati, F., Chen, H., & Muffatto, M. (2021, September). A deep learning model for startups evaluation using time series analysis. In *European Conference on Innovation and Entrepreneurship* (p. 311). Academic Conferences International Limited.

Längkvist, M., Karlsson, L., & Loutfi, A. (2014). A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern recognition letters*, *42*, 11-24.

Bashir, A. K., Khan, S., Prabadevi, B., Deepa, N., Alnumay, W. S., Gadekallu, T. R., & Maddikunta, P. K. R. (2021). Comparative analysis of machine learning algorithms for prediction of smart grid stability. *International Transactions on Electrical Energy Systems*, *31*(9), e12706.

Ersan, D., Nishioka, C., & Scherp, A. (2020). Comparison of machine learning methods for financial time series forecasting at the examples of over 10 years of daily and hourly data of DAX 30 and S&P 500. *Journal of Computational Social Science*, *3*, 103-133.

Defacio, F. S., & Bronzati, V. R. (2019). Mercado de ações: estudo da previsibilidade utilizando séries temporais aplicadas em machine learning.

Zeroual, A., Harrou, F., Dairi, A., & Sun, Y. (2020). Deep learning methods for forecasting COVID-19 time-Series data: A Comparative study. *Chaos, Solitons & Fractals*, *140*, 110121.

Zaheer, S., Anjum, N., Hussain, S., Algarni, A. D., Iqbal, J., Bourouis, S., & Ullah, S. S. (2023). A Multi Parameter Forecasting for Stock Time Series Data Using LSTM and Deep Learning Model. *Mathematics*, *11*(3), 590.

He, K., Yang, Q., Ji, L., Pan, J., & Zou, Y. (2023). Financial time series forecasting with the deep learning ensemble model. *Mathematics*, *11*(4), 1054.

Nabipour, M., Nayyeri, P., Jabani, H., Shahab, S., & Mosavi, A. (2020). Predicting stock market trends using machine learning and deep learning algorithms via continuous and binary data; a comparative analysis. *IEEE Access*, *8*, 150199-150212.

Park, Sungwoo & Jung, Seungmin & Jung, Seungwon & Rho, Seungmin & Hwang, Eenjun. (2021). Sliding window-based LightGBM model for electric load forecasting using anomaly repair. The Journal of Supercomputing. 77. 1-22. 10.1007/s11227-021-03787-4.