INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Enhancing Financial Trading Strategies: A Study of Convolutional Neural Networks with Feature Selection and Other Machine Learning Techniques

Gonçalo de Sousa Silva Valério Rosado

Master in Data Science

Supervisor:
PhD Diana E. Aldea Mendes, Associate Professor,
ISCTE - University Institute of Lisbon

Co-Supervisor:
PhD Tomás Gomes da Silva Serpa Brandão, Assistant Professor,
ISCTE - University Institute of Lisbon

March, 2025

Department of Quantitative Methods for Management and Economics

Department of Information Science and Technology

Enhancing Financial Trading Strategies: A Study of Convolutional Neural Networks with Feature Selection and Other Machine Learning Techniques

Gonçalo de Sousa Silva Valério Rosado

Master in Data Science

Supervisor:

PhD Diana E. Aldea Mendes, Associate Professor,

ISCTE - University Institute of Lisbon

Co-Supervisor:

PhD Tomás Gomes da Silva Serpa Brandão, Assistant Professor,

ISCTE - University Institute of Lisbon

March, 2025

“Horseness is the whatness of allhorse”

– James Joyce

Acknowledgment

This Dissertation is the result of many years of curiosity about the stock market and a passion for analysing and solving problems with data. My interest in this topic stems from the unpredictability of the stock market, the numerous factors and participants that influence it, and the diverse approaches created over time to predict and comprehend its movements. From speculative bubbles to remarkable earnings, the market exhibits endless complexities, challenges, and opportunities, which have always fascinated me.

This curiosity, combined with an even greater passion for data science, which was fostered and strengthened throughout my master's studies, thanks to the faculty members who guided and inspired me.

With that in mind, I would first like to express my sincere gratitude to my supervisors, Diana Aldea Mendes and Tomás Serpa Brandão, for their unwavering support, guidance, and availability throughout this entire process. Their encouragement and insightful feedback motivated me to explore my topic further and strive to produce the best Dissertation possible.

In addition to their instrumental supervision, I would also like to thank Professor Diana Aldea Mendes for inspiring me during the Time Series Analysis and Forecasting course, which sparked the first ideas for this Dissertation. Equally, I am deeply grateful to Professor Tomás Serpa Brandão for broadening my knowledge on integrating image processing into my research.

To my friends, thank you for every moment we shared. Your companionship, laughter and support helped me push through the challenges of this journey.

To my dear Raquel, I cannot find the words to truly express my gratitude. Your endless support throughout this process, the countless hours you spent by my side while I wrote, and your words of encouragement pushed me to do my best. A simple "thank you" will never be enough.

Finally, I could not conclude this without expressing my deepest appreciation to my family. Without you, I would not be the person I am today, and everything I do is in the hope of making you proud. To my parents, thank you for always allowing me to be myself, for encouraging me to follow my dreams, and for supporting me in every decision, even from across continents.

To my cousins, uncles, and grandparents, thank you for always being there for me.

To all, my sincerest gratitude.

Resumo

O foco principal deste estudo é o uso de Redes Neuronais Convolucionais (CNNs) para previsão do mercado de ações, com ênfase nos tipos de variáveis usadas como inputs do modelo, no impacto da replicação de condições específicas de mercado no desempenho do modelo e num extenso processo de *feature selection*. Para estruturar o estudo, foi utilizada a metodologia CRISP-DM, que forneceu uma *framework* que orientou a tese desde o *business understanding* a *data understanding, preparation, modelling, and evaluation*. Para iniciar este estudo, foi efetuada uma revisão exaustiva da literatura para estabelecer o estado atual da arte. Após a revisão da literatura, o estudo envolveu a análise das variáveis habitualmente utilizadas nas previsões do mercado ações, incluindo indicadores técnicos, macroeconómicos e fundamentais, bem como a investigação das estratégias de *trading* e do sentimento das notícias em relação ao mercado de ações. Uma secção crítica da investigação foi o processo de *feature selection*, que utilizou vários métodos diferentes, abrangendo abordagens de *filter*, de *wrapper* e *embedded*. Foi realizada uma análise de *cross-selection* entre estes métodos de forma a identificar as variáveis mais representativas, garantindo um *dataset* robusto e optimizado. Em seguida, foram estabelecidos três *datasets* distintos, com diferentes variáveis *target* e variáveis independentes derivadas das várias estratégias de *trading*. Para melhorar ainda mais a interpretabilidade e o desempenho do modelo, os dados de séries temporais foram transformados em imagens usando três métodos diferentes: *Gramian Angular Fields* (GAF), *Recurrence Plots* (RP), e *Markov Transition Fields* (MTF). O modelo *Long Short-Term Memory Network* (LSTM) de base e o modelo CNN foram testados e optimizados nos vários *datasets*. Os resultados demonstraram a superioridade do modelo CNN utilizando imagens GAF, alcançando melhorias significativas de *accuracy* de 7-10% em relação à linha de base LSTM, particularmente nas estratégias de *stock price direction* e MACD *Crossover*. Além disso, a utilização de dados sintéticos revelou-se valiosa, apresentando uma elevada fidelidade e desempenho.

Abstract

This Dissertation's focus is using Convolutional Neural Networks (CNNs) for stock market prediction, emphasizing the types of variables used as model inputs, the impact of replicating specific market conditions on model performance, and an extensive feature selection process. To structure the study, the CRISP-DM methodology was employed, providing a systematic framework that guided the progression from business understanding to data understanding, preparation, modelling, and evaluation. To begin this Dissertation, a comprehensive literature review was conducted to establish the current state of the art. Following the literature review, the study involved analysing variables commonly used in stock market predictions, including technical, macroeconomic, and fundamental indicators, as well as investigating trading strategies and news sentiment. A critical section of the research was the feature selection process, which employed various methods spanning filter, wrapper, and embedded approaches. A cross-selection analysis was conducted across these methods to identify the most representative variables, ensuring a robust and optimized input feature set. Following this, three distinct datasets were established, featuring different target features and independent variables derived from the various trading strategies. To further enhance model interpretability and performance, time-series data were transformed into images using three different methods: Gramian Angular Fields (GAF), Recurrence Plots (RP), and Markov Transition Fields (MTF). A baseline Long Short-Term Memory Network (LSTM) and CNN architectures were tested and optimized on the various datasets. Results demonstrated the superiority of the CNN model utilizing GAF images, achieving significant accuracy improvements of 7-10% over the LSTM baseline, particularly in stock price direction and MACD crossover strategies. In addition, the use of synthetic data proved valuable, displaying high fidelity and contributing to enhanced model performance.

Table of Contents

List of Figures

List of Tables

Glossary of Abbreviations and Acronyms

AD – Accumulation/distribution

ADF – Augmented Dickey Fuller

AMD – Advanced Micro Devices

APO – Absolute Price Oscillator

CCI – Commodity Channel Index

CNN – Convolutional Neural Network

DPO – Detrended Price Oscillator

EC – Evolutionary Computation

EMA – Exponential Moving Average

GA – Genetic Algorithm

GAF – Gramian Angular Field

GAN – Generative Adversarial Network

GP – Genetic Programming

GRU – Gated Recurrent Units

HMA – Hull Moving Average

LSTM – Long Short:Term Memory

MACD – Moving Average Convergence Divergence

MA – Moving Average

MFM – Money Flow Multiplier

ML – Machine Learning

MTF – Markov Transition Field

OBV – On Balance Volume

PCA – Principal Component Analysis

PSAR – Parabolic SAR

PSO – Particle Swarm Optimization

RP – Recurrence Plot

RSI – Relative Strength Index

SMA – Small Moving Average

SVM – Support Vector Machine

TEMA – Triple Exponential Moving Average

WMA – Weighted Moving Average

Chapter 1

Introduction

In the last decades, Machine Learning (ML) has emerged as a popular tool in the field of financial trading, as it assists in decision-making processes and facilitates the recognition of patterns in financial data. There is a vast range of machine learning models that have been applied to financial trading, like support vector machines (SVMs), decision trees, and deep learning approaches such as long short-term memory (LSTM) networks (Kumbure et al., 2022). All of these algorithms have shown potential working with financial data and have their own advantages and disadvantages depending on the specific problem at hand.

The main focus of this Dissertation will be primarily on the use of convolutional neural networks (CNNs) for stock price movement prediction. Initially developed for image recognition tasks, CNNs have been successfully applied to various financial tasks, including stock price prediction.

Predicting the financial market is a challenging and complex task. Instead of focusing solely on obtaining the highest accuracy, this Dissertation addresses the following research questions focused on the use of CNNs for financial trading:

1. Which feature selection algorithms can improve stock market prediction?
2. How can CNNs be used to identify patterns in financial market data?
3. How do CNNs compare to other machine learning algorithms regarding predictive performance and accuracy in financial trading?
4. How can Generative Adversarial Networks (GANs) generate synthetic data to improve model performance?

Besides the literature review, the first step in this Dissertation is the data collection. With that in mind, it is essential to know which variables impact the stock price movement the most, since using different input variables can make the model perform completely differently. Thus, developing the optimal model is an arduous task.

So, for this Dissertation, it was considered fundamental to conduct an in-depth analysis of the variables used in stock market predictions, such as technical, macroeconomic, and fundamental indicators.

According to the study by Tsai and Hsiao (2010), the most common analytical approaches to stock price analysis are fundamental and technical analysis. For this work, since the target variables were based on trading strategies, the focus was on technical analysis, which has been part of financial practice for many decades, examining a stock's historical price and volume movements. In addition to the technical indicators, sentiment from news is also used.

According to Chen et al. (2019), pre-processing data, usually leads to more effective predictions. Considering that, it is fundamental to select which group of attributes should be chosen as input variables. Performing feature selection is an important task in machine learning since it improves the model performance and also enhances data visualization and understanding (Xue et al.,2016). In this Dissertation, a vast collection of feature selection methods is applied to the datasets.

The main goal of the Dissertation is to enhance stock price movement prediction using CNNs. To achieve this, it is essential to transform time series data into images, and in this Dissertation, three different methods were selected: Gramian Angular Field (GAF), Recurrence Plot (RP), and Markov Transition Field (MTF). These methods capture spatial dependencies between the various features in the image. The images are then used as input for the 2D CNN.

The model used daily stock price data from Advanced Micro Devices (AMD), covering the period from 1/1/2000 and 31/12/2023, and was evaluated across multiple datasets with different target variables to assess its performance under various trading strategies, including the Stochastic Oscillator and the Moving Average Convergence Divergence (MACD) Crossover. In addition, Generative Adversarial Networks (GANs) were employed to generate synthetic data, replicating market conditions to assess their potential in improving machine learning model performance.

This Dissertation offers a strong foundation for exploring the application of CNNs in financial trading strategies and valuable insights for future research, particularly in developing hybrid models with sentiment analysis and a wider use of synthetic data in different markets and contexts.

To summarize, this Dissertation is structured as follows: a literature review covering feature selection methods, technical indicators, machine learning approaches for stock prediction, CNNs, and the use of GANs for generating synthetic data; the methodology used for developing training data and evaluating the model's performance; the presentation of the model's results and discussion; and finally, the conclusions drawn from the findings, along with suggestions for future research directions.

Chapter 2

Literature Review

This chapter serves as the base for the proposed Dissertation, trying to answer some of the questions mentioned in the introduction. In this literature review, a wide range of studies were examined related to the application of machine learning techniques in financial trading, with a specific focus on the use of Convolutional Neural Networks (CNNs) for stock market prediction. In addition, an in-depth analysis of articles about the variables used in stock market predictions was also conducted, from technical indicators to macroeconomic, fundamental indicators, and also new types of variables. Moreover, it is provided an extensive assessment of feature selection methods used in stock market forecasting, since selecting the best features for the model is a complex yet essential task (Peng et al.,2021).

Finally, it was also examined the use of Generative Adversarial Networks (GANs) to improve the model's performance, since this technique has been used in financial applications, such as generating synthetic financial time series data.

The literature review is structured as follows: first, the search strategy is defined, followed by the search queries utilized, and then the inclusion and exclusion criteria for articles. Next, the methodology is briefly described, followed by a discussion and presentation of the review's findings. Lastly, conclusions are provided based on the results and discussion, along with potential directions for future research.

2.1. Search Strategy

The search strategy for this literature review was designed to identify a suitable and relevant set of studies to help answer the established research questions. The search process consisted of two stages. In the first stage, a preliminary set of research papers, such as those by Jansen (2020), Henrique et al. (2019), and Kumbure et al. (2022), were manually selected. In the second stage, the initial set of articles was used to locate additional studies through a process called snowballing (Henrique et al., 2019).

2.1.1. Manual Search

The primary set of studies included some scientific articles, chapters of books, and other reviews, such as Jansen (2020), Kumbure et al. (2022), Sezer and Ozbayoglu (2018), and Lu et al. (2021), among others.

These articles were found through a manual search of the literature review on CNN for financial trading, which included topics as data sources, feature selection, stock market prediction and deep learning approaches.

The following phase was performing backward snowballing on this set of articles. In the backward snowballing process, the first step was to look at the reference list of the initial set of articles and include any studies that met the established inclusion and exclusion criteria. In this step, only the titles of the articles were analysed.

After deciding which articles were relevant, it was done a further analysis of them, by reading the abstract and other meaningful parts. The significant articles were included in the final set.

2.1.2. Automated Search

The search process was extended using an automated search to broaden the sample of relevant studies. For this strategy, seven databases were used, namely: Google Scholar, IEEE Xplore, Science Direct, ArXiv, Springer, Research Gate, and Scopus.

The process began by selecting the search queries and identifying relevant articles using the inclusion and exclusion criteria. The abstracts (and other necessary parts) of the selected articles were evaluated, and the articles that met the criteria were included in the final set.

2.2. Search Queries

The search queries were selected to reflect the aim of this literature review. To determine the relevant terms, it was used the current understanding of the topic, as well as the information provided in the titles, keywords, and abstracts of articles found through the manual search. The most used terms were "Forecasting" and "Prediction", although both were used separately for each data source.

In the search queries, the main objective was to seek articles that discussed "Machine Learning", "Stock Price", "Trend Prediction", "Convolutional Neural Networks", "Feature Selection", "Indicators", "Generative Adversarial Networks", since it was expected to find articles that were based on forecasting models for the stock market that employed machine learning techniques.

After some research, it was found that using machine learning aligned with the other terms expanded the number of relevant articles retrieved.

2.3. Inclusion and Exclusion Criteria

For this literature review, inclusion and exclusion criteria were used to determine which articles were considered significant.

Initially, it was determined that only articles from 2000 onward would be considered for the literature review.

The second criterion for selecting articles for the literature review was to only include those that had been published, such as in journals, theses, or book publications. Additionally, only articles written in English were considered.

Finally, only articles that were available in full text were considered for inclusion in the literature review, as it was deemed essential to have access to the complete texts in order to conduct a thorough analysis. After combining the final sets of articles from the manual and automated searches and the inclusion-exclusion criteria, 44 articles were selected.

2.4. Related Work

This section presents the findings of the analysis conducted on the literature review in relation to the research questions established previously.

The review was conducted using information taken from 44 selected studies. The findings of this review are presented and discussed in three subtopics:

1. Details about the studies, such as bibliographic information.
2. Information about the data used in the studies, mainly the type of indicators applied.
3. An overview of the machine learning methods used in the studies and developments that were made in these methods.

2.4.1. Bibliographic information

In this subtopic, the selected articles were analysed based on the year of publication and the most frequently used keywords.

This information can be found in the figures below (Figure 2.1 and Figure 2.2). Figure 2.1 shows that more than 50% of the selected articles were published between 2017 and 2023. This suggests that there has been a significant rise in the number of studies focused on forecasting stock markets using machine learning, particularly CNN approaches.

Figure 2.2 presents the top 10 keywords used by an author in their work and the number of times they were used. The figure shows that the keywords "Stock Market/Price Prediction," "Deep Learning," "Convolutional Neural Network" and "Feature Selection" are among the most used. Additionally, the top keywords often include variations, such as synonyms and different forms of specific keywords, such as "Convolutional Neural Networks" and "CNNs," indicating that keeping these distinctions in mind may assist in searching for publications.

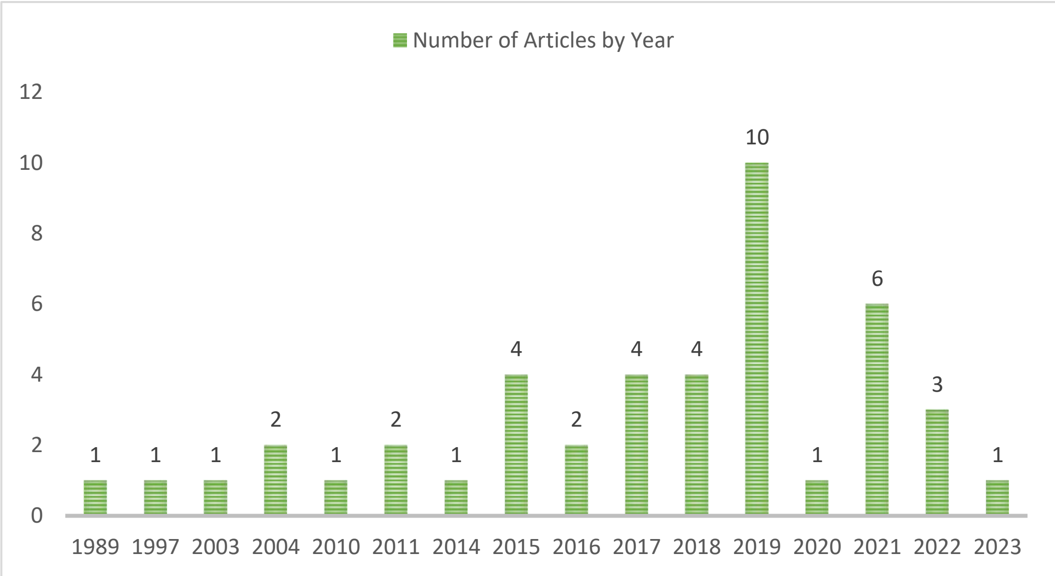

Figure 2.1. Number of articles by year.

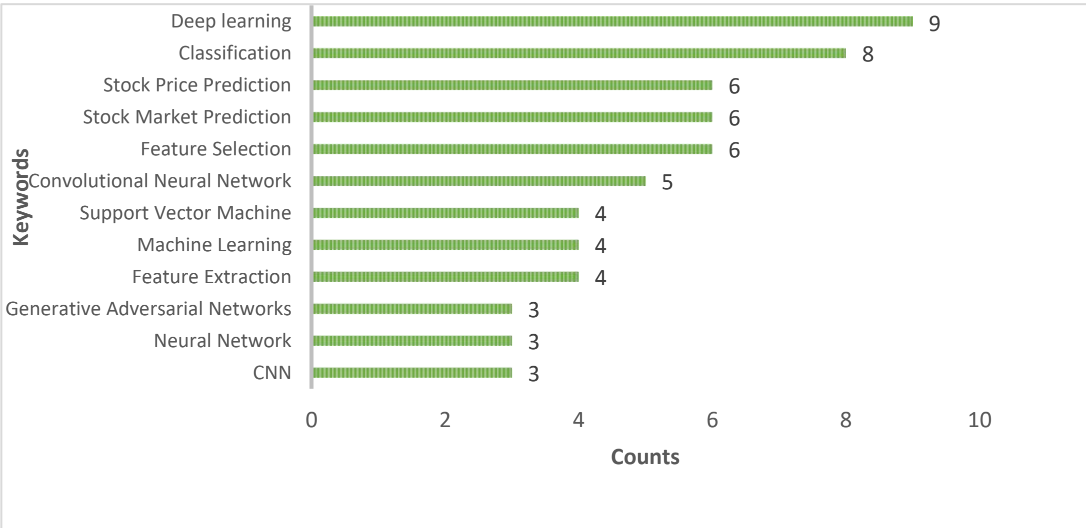

Figure 2.2. Most used keywords in selected research articles.

2.4.2. Indicator Variables

The first step in the workflow of a stock market prediction model is data collection. Taking that into account, the type of variables that have the most impact on the stock price movement were analysed during this review.

Sedighi et al. (2019) selected the most essential technical indicators, 20, from different types, such as: trend, volume, support and resistance, momentum and volatility. Furthermore, the chosen indicators encompass all stock data, including indicators from the four classes: Oscillator, Index, Overlay, and Cumulative. Examples of the indicators used in this Dissertation include the Absolute Price Oscillator (APO), Fibonacci Fan, Commodity Channel Index (CCI), and Williams %R.

A work done by Chang and Wu (2014), adopted other technical indicators aside from the essential ones, such as the differences of technical indices and the same indicators but with different timeframes, for instance, a 6-day Relative Strength Index (RSI) and a 12-day RSI.

According to a study done by Peng et al. (2021), utilizing technical indicators compiled by the combination of other indices can be considered as a replacement of their constitute counterpart, since it lowers the levels of redundant information considered for the models and possibly produces better predictive results and asset allocation. Moreover, the technical indicators that were most frequently chosen by feature selection methods in recent studies between 2008 and 2019 were the Detrended Price Oscillator (DPO), Hull Moving Average (HMA), and Money Flow Multiplier (MFM). However, more traditional indicators, such as the Simple Moving Average (SMA) and Weighted Moving Average (WMA), were only selected a few times.

As previously mentioned, feature selection methods tend to prefer indicators that combine multiple sources of information, such as the HMA, a combination of WMAs for different window sizes.

The main disadvantage of using only technical indicators (technical analysis) is that it only considers the stock's price movement and ignores fundamental factors related to the company (Barak et al., 2017).

In addition to the technical indicators, financial and macro-economic variables received substantial attention in the development of stock market studies, such as credit ratings, money supply levels, and T-bill rates. (Enke et al., 2011; Tsai et al., 2011; Zhong & Enke, 2017).

The logic behind fundamental analysis is that if a company has strong fundamentals, then investing in its stock for the long-term will be more secure and stable (Barak et. Al, 2017). This work utilized fundamental indicators such as liquidity ratios, activity ratios, profit margins, growth rates, earnings per share (EPS), dividend per share (DPS), stock book value, etc.

As shown in Figures 2.3 and 2.4, most studies reviewed utilize data from various stock markets in the United States. However, there has been an increase in studies that predict stock performance using data from Asia, when compared to previous literature reviews (Henrique et al. ,2019; Kumar et al., 2021).

In addition, Figure 2.5 supports the trend of using technical indicators in stock market prediction research and also highlights an increase in the utilization of other variables, such as news and tweets.

In conclusion, recent research on the stock market suggests that many factors are associated with future stock prices (Enke et al., 2011). However, it is crucial to use feature selection to identify the indicators that have the strongest forecasting capability, as using too many financial, technical and economic indicators can overburden the prediction system (Thawornwong et al., 2003).

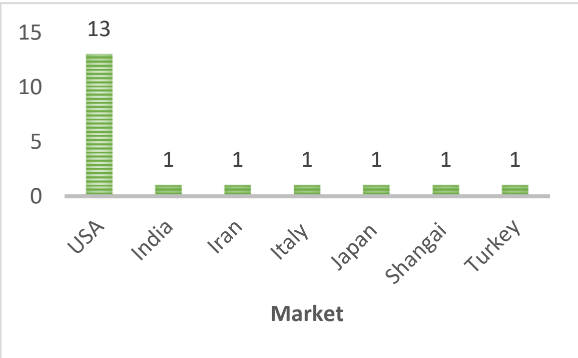

Figure 2.3. Frequency by market.

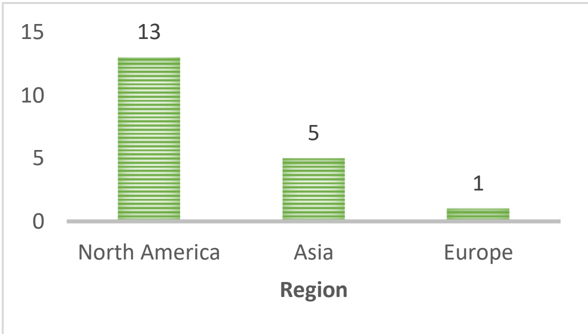

Figure 2.4. Frequency by region.

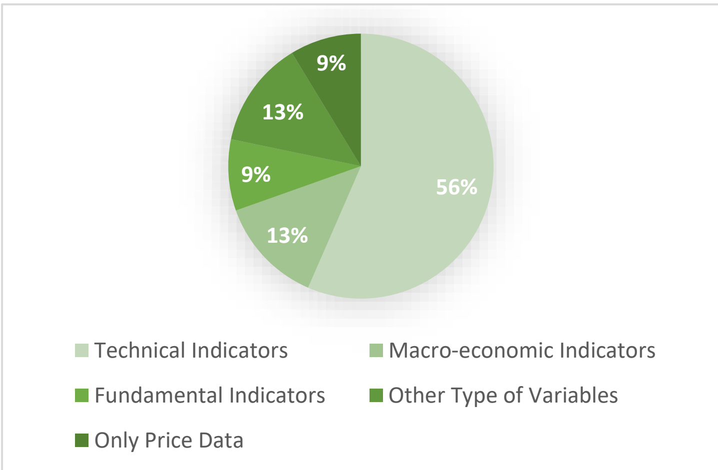

Figure 2.5. Types of variables in the selected articles.

2.4.3. Feature Selection

Financial variables are hard to predict (Peng et al., 2021). Having that in mind, it is essential to select which group of attributes should be chosen as input variables.

Feature selection is a challenging task primarily due to a large search space, where the overall number of potential solutions is 2^n for a dataset with *n* features. (Dash & Liu, 1997). In recent years, this task has become even more difficult, as *n* is increasing in many fields due to advances in data collection techniques and the enhanced complexity of machine learning problems. (Xue et al., 2016)

Although a range of search techniques have been applied to feature selection, such as complete search, heuristic search, random search and greedy search, most of them still suffer from stagnation in local optima or high computational cost (Too et al., 2019). Having that in mind, an efficient global search technique is required to better solve this type of problem.

The feature selection community has widely acclaimed evolutionary computation (EC) techniques due to their global search ability and potential (Xue et al., 2016). The two most popular EC methods in feature selection are Genetic Algorithms (GAs) and particle swarm optimization (PSO).

According to the evaluation criteria, feature selection algorithms can be divided into filter and wrapper approaches. The primary distinction between both is that wrapper approaches contain a classification/learning algorithm in the feature subset evaluation step (Guyon & Elisseeff, 2003).

Another challenging aspect of feature selection is feature interaction. It happens frequently, and it can be a two-way, three-way, or a complex multiway interaction among features. For example, a feature that is almost irrelevant to the target variable by itself could improve the accuracy of the model if it is utilized with some complementary features. Hence, the two main factors in a feature selection approach are the search technique and the evaluation criteria (Xue et al., 2016).

Concerning search techniques, in recent years, the EC techniques have been the most effective methods to solve feature selection problems, although they have some limitations such as scalability. Regarding the evaluation criteria, for wrapper feature selection approaches, the classification performance of the chosen features is the evaluation criteria. Filter feature selection, on the other hand, are independent of any classification algorithm and apply other scientific methods, such as information theory-based measures, distance measures, or even correlation measures (Dash & Liu, 1997, Tareq et al., 2018).

The most popular approach has recently been using EC algorithms such as GAs and Genetic Programming (GP) to address feature selection tasks with thousands of features, improving the representation and the classifiers (wrapper feature selection approach). Furthermore, combining feature selection methods with feature extraction/construction can enhance the classification performance (Xue et al., 2016).

In the study by Tsai and Hsiao (2010), the main goal was to combine different feature selection methods to identify more representative variables to better predict stock price movements. While Principal Component Analysis (PCA) and Genetic Algorithms (GA) are not strictly categorized as feature selection methods, they can be utilized as such to reduce dimensionality and optimize the selection of features. Alongside decision trees (CART), these methods were combined using three strategies: union, intersection, and multi-intersection.

The results showed that the intersection of PCA and GA, as well as the multi-intersection of PCA, CART, and GA, attained the highest performance, with an accuracy of 79% and 78.98%, respectively. Furthermore, both approaches removed nearly 80% of unrepresentative features, highlighting their effectiveness in reducing data complexity for better predictive performance.

Figure 2.6 supports the widespread use of Evolutionary Computation (EC) algorithms for selecting features in stock market prediction.

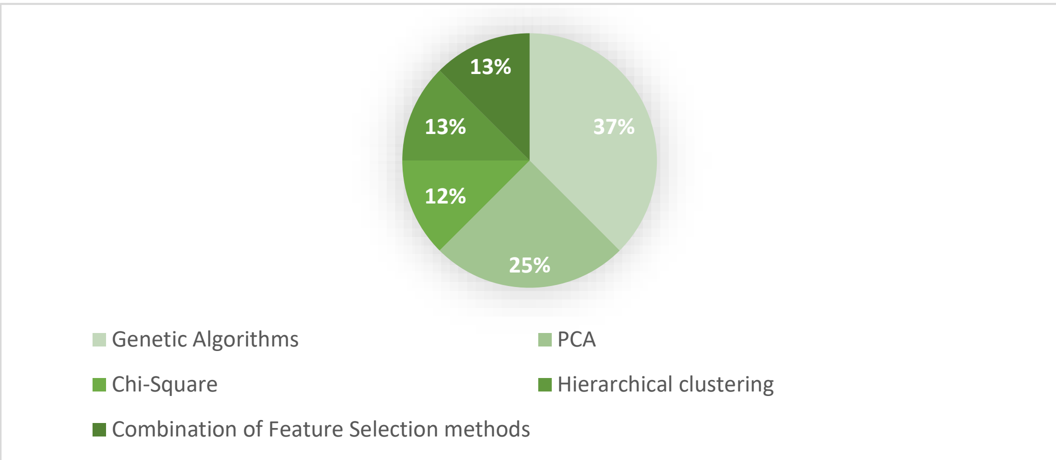

Figure 2.6. Types of Feature Selection Methods present in articles.

2.4.4. Machine Learning Approaches

Machine learning approaches are crucial for stock market prediction as they enable the analysis of large amounts of data, identify complex patterns, and improve the accuracy of predictions.

Before deep-diving into the CNN approaches, Zhong and Enke (2019) utilized a big data analytic process aligned with a DNN to forecast the daily price direction of the SPDR S&P 500 ETF, an exchange-traded fund that replicates the performance of the S&P 500, based on 60 financial and economic features. The authors utilized the PCA to transform the data and construct a low-dimensional representation of the data while preserving the maximum variance and covariance shape.

According to the authors, the DNN-based classification for the PCA-represented data set with 31 principal components achieves the highest accuracy, and in general, the DNN classifiers outperform the ANN classifiers in predicting the SPDR S&P 500 ETF price movement. The authors also noted that a pattern regarding the classification accuracy appears while increasing the number of hidden layers, with the overfitting issue remaining under control.

As mentioned above, the CNNs are another research focus on deep learning regarding financial trading, which has been applied widely in the field of image processing, speech recognition, and recently in time-series data (Kumbure et al., 2022).

2.4.5. Convolutional Neural Networks

As stated, CNNs have been used for time-series data and have been known to achieve state-of-the-art results on time-series classification.

In the context of utilizing CNN architecture to create a financial trading model, there is a growing body of research that is studying the high potential of this approach and its limitations.

For example, Gunduz et al. (2017), proposed a CNN approach to predict intra-day movements of the Borsa Istanbul 100 index. The researchers utilized feature correlations and hierarchical agglomerative clustering to order the input features and tested three techniques (L2 regularization, dropout, and early stopping) to prevent overfitting. They also compared the effectiveness of a CNN model using correlated features (CNN-corr) to a CNN model using randomly ordered features (CNN-rand) and found that the CNN-corr model performed better.

In 2019, Hoseinzade and Haratizadeh proposed a CNN approach to predict the price movement of five stock indices—S&P 500, NASDAQ, DJI, NYSE, and RUSSELL—using two different feature representation methods, 2D-CNN and a 3D-CNN, while emphasizing that the filter size should be defined based on the financial interpretation of features and their attributes. These CNNs have four significant steps: input data representation, daily feature extraction, durational feature extraction, and final prediction.

The remarkable part of this study is that the first layer of both frameworks is assigned to combine the daily features into higher-level features for representing each single day of the dataset. According to the authors, their model outperformed the benchmarks in all five indices by about 3% to 11% in terms of F1 score, which shows that the model not only accurately identifies positive instances but also minimizes false positives.

In another study, Cao and Wang (2019) explored the use of CNNs for financial forecasting, showing that they can effectively handle both categorical and continuous variables and achieve strong prediction results. They tested two different models, a CNN and a CNN-SVM, to predict stock index prices and found that both models performed well.

Methab et al. (2021) developed a CNN with a walk-forward validation to predict NIFTY 50 stock price movements in the National Stock Exchange of India. The authors developed three approaches, varying in the number of variables used in forecasting the price movements, the number of sub-models, and the size of the input training data.

According to the authors, the results show that CNN-based multivariate model is the most effective and accurate. This model consisted of two convolutional layers with 32 filter maps followed by a pooling layer, then again, another convolutional layer with 16 feature maps and pooling. It is essential to mention that this was a multi-step time series forecasting approach since it uses the prior time series data to forecast the values for the next week. This research also explored the power of Generative Adversarial Networks (GAN) to improve prediction performance.

One of the more advanced deep learning algorithms is the LSTM, and according to the literature, Chen et al. (2019), created a stock price trend prediction model (TPM) that combines a CNN with an LSTM. The TPM consists of two phases. First, it uses a piece-wise linear regression method (PLR) to extract long-term temporal features and a CNN to extract short-term spatial market features. These two methods work together as a dual feature extraction method. In the second phase, an encoder-decoder framework, which is formed by an LSTM, is employed to select and combine significant features and make trend predictions. The model differs from traditional methods since it can extract relevant features for mining the financial time series.

A paper by Lu et al. (2021) went even further and proposed a CNN-BiLSTM-AM model to predict the Shanghai Composite Index stock closing price for the next day. As the acronym suggests, the model combines a Convolutional Neural Network, a Bi-directional Long Short-Term Memory (BiLSTM) and an

Attention Mechanism (AM). The CNN is utilized to extract features from the data, and then the BiLSTM applies those features to forecast the stock closing price of the following day. Finally, the AM, captures the impact of feature states on the stock closing price at various times in the past to enhance the accuracy of the model. This method achieved an RMSE of 0.31694 and 0.9804 R^2, which surpasses every benchmark.

Sezer and Ozbayoglu (2018) proposed an innovative approach to CNN architectures by transforming time series data of alpha factors into a two-dimensional format, leveraging the model's ability to detect local patterns. The model proposed was a CNN-TA, which computes 15 technical indicators for different intervals and utilizes hierarchical clustering to find indicators with similar behaviour in a two-dimensional grid. The architecture was composed of nine layers: one input layer, two convolutional layers, a max pooling, two dropout layers (to prevent overfitting), fully connected MLP layers, and finally, an output layer. In this Dissertation, a 3x3 filter was utilized in the CNN, which helps capture more details of the images.

The model was evaluated using two criterias: Computational Model Performance and Financial Evaluation. Regarding Computational Model Performance, it was verified that the recall values of the classes “Buy” and “Sell” were better compared with the “Hold” class. Accuracy results were 0.58 for the Dow30 dataset, which consists of 30 major publicly traded U.S. companies that make up the Dow Jones Industrial Average (DJIA), a key benchmark for the U.S. stock market and 0.62 for the ETFs. Financially, the proposed model presented an average annualized return almost three times higher compared with average annualized return of the benchmark models. According to the authors (Sezer and Ozbayoglu, 2018), the proposed model could present better results if the structural parameters were optimized. They suggest boosting the data representation for “Buy”, “Sell” and “Hold” points for better trade signal creation performance, potentially via the use of GANs to boost trade signal creation.

A similar study was done by Chandar (2022), which developed a stock trading model by combining Technical Indicators and a CNN (TI-CNN). The first step was obtaining ten technical indicators from historical data and taking them as feature vectors. Then, those feature vectors were transformed into an image using the Gramian Angular Field (GAF) method and were employed as input data for CNN.

2.4.6. Generative Adversarial Networks

GANs were created by Goodfellow et al. (2014), and in recent years, Yann LeCun et al. (2015) stated that these networks were the “most exciting idea in AI in the last ten years”. GANs, as mentioned by Stefan Jansen (2020), train two neural networks, called the generator and discriminator, in a competitive setting. The generator network produces samples until the discriminator network can’t differentiate it from a given training data class. The outcome is a generative model that can develop synthetic samples of an individual class distribution without expense. Yoon and Jarret (2019) proposed a novel GAN architecture to model time-series data, the TimeGAN.

The main difference between this novel approach and the other GAN architectures, is that TimeGAN proposes the concept of supervised loss, where the model depicts time conditional distribution within the data by applying the original dataset as supervision.

The authors demonstrated the application of the TimeGAN models to financial data by using 15 years of daily Google stock prices, targeting synthetic series with 24-time steps. The two autoencoder components and the generator element of the adversarial network, consisted of an RNN with three hidden layers and 24 Gated Recurrent Units (GRU) units. The supervisor component of RNN only differs in the number of hidden layers, which is 2.

This novel model presented by the authors demonstrated consistent and significant improvements in terms of performance when compared to state-of-the-art benchmarks.

In addition, Staffini (2022) presented a Deep Convolutional Generative Adversarial Network (DCGAN) architecture as a solution for forecasting stock prices. This architecture demonstrated improved performance in both single-step and multi-step forecasting when compared to standard methods. For the generator network, the author selected a CNN-BiLSTM architecture, which, according to Lu et al. (2021), achieves excellent results. On the other hand, for the discriminator network it was selected a simple CNN architecture.

Lin et al. (2021) developed a stock prediction model that employs a GAN architecture. A GRU is utilized as the generator, which takes historical stock prices as input and generates predictions for future prices. A CNN is the discriminator, trained to distinguish between real and generated stock prices. In their study, the authors found that training with a 1D-CNN discriminator improves the performance of basic recurrent models in stock prediction. Additionally, using a GRU-based generator results in more stable training and better test performance.

The noteworthy part of their study is the use of the loss function from the Wasserstein GAN with the Gradient Penalty (WGAN-GP) model as an alternative to a simple GAN, providing more stable and improved performance for multi-step ahead predictions.

The main conclusion from these studies, is that although financial time series forecasting may benefit from using GANs, training this type of model remains a challenging task due to the need to adjust multiple hyperparameters while maintaining a balance between the generator and discriminator networks.

2.4.7. Review Summary

The literature review for this Dissertation was essential in determining the direction and focus on stock market prediction models. It emphasized the value of leveraging diverse data sources, including technical indicators, fundamental analysis, and even sentiment data, to enhance predictive capabilities. For instance, it emphasised the effectiveness of using the same indicator across different timeframes to capture a broader range of patterns and the importance of incorporating additional variables, such as news sentiment, to improve model accuracy.

The review also stressed the critical role of feature selection in optimizing model performance, exploring various techniques such as Genetic Algorithms (GAs) alongside traditional methods like embedded, wrapper, and filter approaches. These insights inspired the idea of combining feature selection techniques to minimize redundant information and prioritize the most representative variables, ultimately boosting the model's accuracy and efficiency.

Moreover, the review deepened the understanding of advanced machine learning models, including CNNs and LSTM models, and their applications in time-series data. It showcased the CNNs ability to process financial features and integrate technical indicators while highlighting the potential of GANs for producing synthetic data to further enhance model performance. This initial knowledge was instrumental in guiding the development of the Dissertation, providing clarity on variable selection, the importance of feature selection, and the potential of both CNNs and GANs for financial data analysis.

Chapter 3

Data Selection Methodology

For this Dissertation, to guarantee a systematic approach and considering its focus on Data Science, the CRISP-DM (Cross-Industry Standard Process for Data Mining) framework will be employed. This widely used methodology in the industry provides a structured procedure for organizing the dissertation's workflow through its key stages: Business Understanding, Data Understanding, Data Preparation, Modelling and Evaluation.

Given the emphasis placed on the data preparation phase in this Dissertation, particularly on feature engineering, feature selection and image generation, a detailed deep dive into the data preparation process will be presented in a separate section. Similarly, the modelling and evaluation stage will also be discussed in a separate section. This approach ensures that these critical steps are fully described.

3.1. Business Understanding

The Business Understanding phase is a central part of the Dissertation, laying the grounds for its structure. The first purpose of the study is to develop a novel algorithmic trading model using a CNN architecture to evaluate stock price movements. This architecture will utilize various trading strategies and assess its performance, aiming to top the effectiveness of benchmark models, such as an LSTM model.

A second objective of this Dissertation is to improve the model performance by applying GANs, particularly the TimeGAN architecture. This method aims to generate synthetic data providing a more thorough assessment of the trading model's effectiveness.

A selection process was conducted to adopt the stock utilized in the proposed method, analysing Global Industry Classification Standard (GICS) sector returns, volatility, trading volume, and other metrics. Although the detailed process and the selected stock will be stated in the next section, a brief overview is provided here for context. The preferred stock is Advanced Micro Devices, Inc. (AMD), an American semiconductor company founded in 1969 by Jerry Sanders. AMD operates globally, offering a variety of digital semiconductors, including microprocessors, graphics processing units (GPUs), data center solutions, embedded processors, and other graphics solutions for desktops, laptops, and gaming consoles.

Recently, AMD has been focusing on expanding its AI and data center capabilities, leading hedge funds and analysts to keep a bullish position on its stock. The acquisition of Xilinx in 2022 has further diversified AMD's business, strengthening its position in key markets. It's also essential to mention that the company faced significant challenges in the late 2000s and early 2010s, only recovering market share due to the success of its Ryzen processors.

Finally, the company's financials in July 2024 reinforce the buy recommendations from various analysts. AMD reported a revenue of $22.8 billion, a net income of $1.12 billion, a profit margin of 4.9%, and a debt-to-equity ratio of 5.34%.

Moving to the construction of the model, the features of the proposed model are defined by producing a wide range of different technical indicators with distinct time intervals, daily news sentiment, previous logarithmic returns, and previous closing prices. Eleven feature selection methods are then employed, and cross-selection is performed to identify the most significant features for each strategy.

The next step before using the CNN model is to generate images using techniques such as Recurrence Plots (RP), Gramian Angular Field (GAF) and Markov Transition Field (MTF). Afterward, the model is tested by different trading strategies, each utilizing various image generation methods. The final step involves generating synthetic data and assessing its fidelity to warrant its reliability and accuracy.

To conclude, as shown in the figure below (Figure 3.1), the proposed workflow is divided into various steps: Data Extraction, Stock Selection, Feature Creation, Data Labelling, Feature Selection, Image Creation and Model Development.

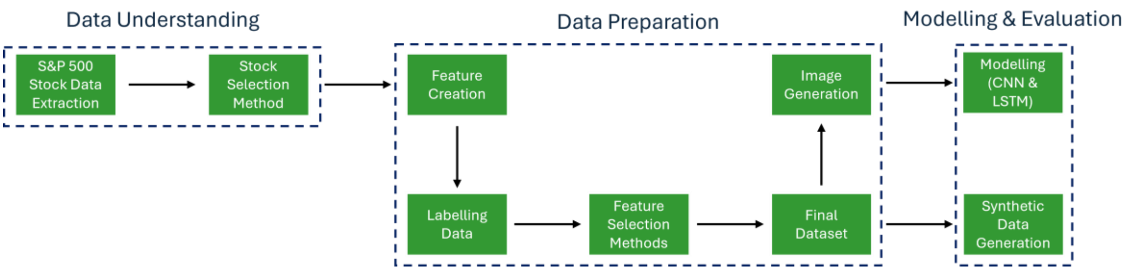

Figure 3.1. Proposed Workflow.

3.2. Data Understanding (Dataset Extraction and Stock Selection)

For this Dissertation, the daily stock prices of S&P 500 stocks were selected for training and testing, between 1/1/2000 and 31/12/2023. Before extracting the daily prices of S&P 500 stocks from Yahoo Finance API, a stock selection process was carried out.

Initially, the focus was on choosing the most liquid stocks within the S&P 500. This involved selecting stocks with the highest trading volume, ensuring they could be readily bought or sold without significantly impacting their market price.

The subsequent stage involved choosing stocks that had experienced a substantial movement over the past year. More precisely, stocks with a price movement exceeding 100% over the past year were selected to identify those exhibiting significant growth.

This stage starts by calculating the five-year (Appendix A) and one-year (Figure 3.2) returns for each stock in the S&P 500. The second step was to identify the GICS Sectors with the highest growth. As exhibited in the image below, the GICS Sectors with the highest average 1-year return are Information Technology and Consumer Discretionary. It is important to note that both these sectors were also in the top 3 in terms of the 5-year average return.

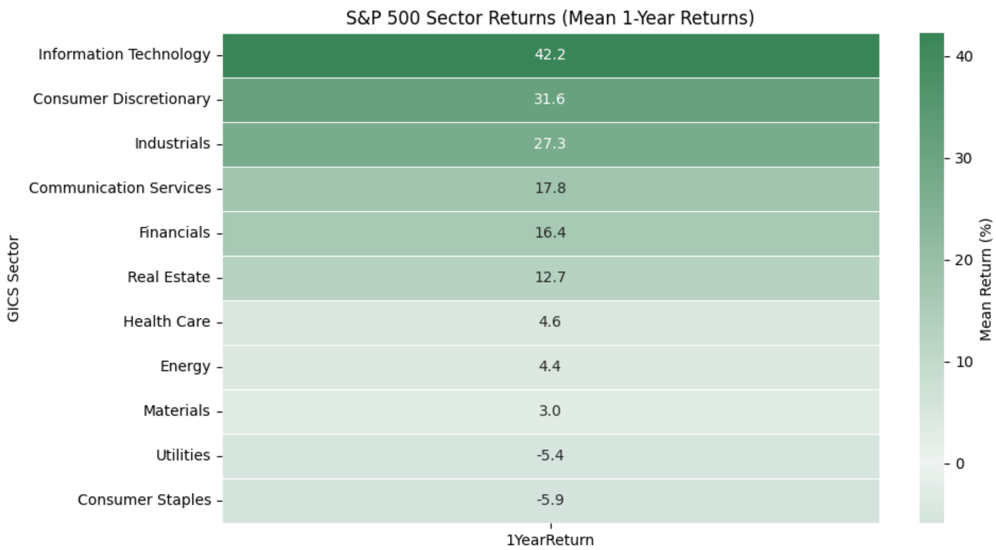

Figure 3.2. S&P 500 one-year sector returns.

The next step was to refine the list of stocks such that it only included those with a price movement exceeding 150% over 2023. This task was accomplished by computing the delta percent metric, which captures the relative change in stock prices between their highest and lowest points within the year of

2023. Although more volatile stocks could be more complicated to predict, it should help reduce the possibility of class imbalance and create more defined samples. The delta percent is defined as:

$$Delta\ Percent = \frac{High - Low}{Low} \times 100 \tag{3.1}$$

In parallel, another method for selecting stocks was developed. The most "momentum stocks" were selected, by choosing the top 5 stocks, sorted by average trading volume, with above-median cumulative returns and trading volume in the top 20% in the last semester of 2023. Ultimately, AMD, NVDA (Nvidia), and TSLA (Tesla) met all criteria, and AMD was chosen for this Dissertation.

Chapter 4

Data Preparation

4.1. Stock Analysis

After selecting the stock, it is essential to perform a time series analysis. In this case, analysing both the evolution of the stock's closing price and daily returns is crucial for evaluating trends, volatility, and potential trading opportunities.

Regarding the stationarity of AMD's closing price, the Augmented Dickey-Fuller (ADF) test results indicate that the stock price is non-stationary, as the p-value of 0.99 is notably higher than the 0.05 threshold. The ADF statistic of 0.76 also exceeds the critical values at the 1%, 5%, and 10% confidence levels, verifying that the null hypothesis (which says that a unit root is present) cannot be rejected. This indicates that AMD’s stock price follows a strong trend component and does not revert to a constant mean over time. Hence, traditional time series forecasting models like ARMA/ARIMA would not be suitable unless differencing or transformations are employed to achieve stationarity, an approach that will not be utilized in this Dissertation.

The seasonal decomposition plot, presented in Figure 4.1, delivers additional insights into AMD's stock price behaviour. The trend component illustrates a long-term upward movement, particularly after 2016, which aligns with AMD’s significant growth during that period. The seasonality component shows a relatively regular range, suggesting the presence of short-term cyclic patterns in stock price movements. These patterns can be influenced by various factors, such as earnings reports, macroeconomic conditions, and market sentiment, which can be captured through technical indicators to recognize trading opportunities.

The residual component reveals periods of increased volatility, predominantly in recent years, indicating increased price fluctuations Given this volatility, implementing trading strategies that capitalize on price fluctuations while minimizing risk could be advantageous. Additionally, incorporating technical indicators tailored to these market conditions could enhance the effectiveness of trading strategies.

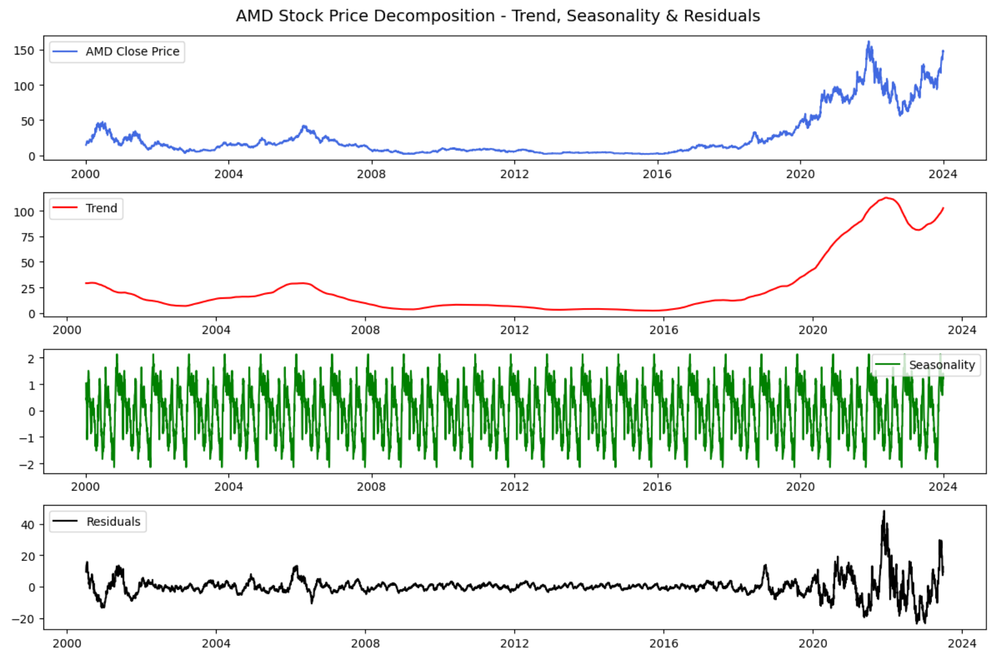

Figure 4.1. AMD closing price seasonal

Concerning the summary statistics of AMD's closing price and daily returns, the data show that the stock has experienced high volatility, with a mean closing price of 25.43 and a standard deviation of 31.92, reflecting significant price fluctuations over the 23-year period. The minimum closing price of 1.62 and maximum of 161.91 further highlight the stock's wide trading range over time. The returns data also exhibit considerable variation, with a mean return of 0.0373%, a minimum of -39.16%, and a maximum of 42.06%, reinforcing the presence of both strong upward and downward price movements over this extended timeframe. The 25th percentile return (-1.87%) and 75th percentile return (1.98%) indicates that most daily returns fall within a relatively moderate range, while extreme values highlight occasional large price swings, presenting trading opportunities.

To further analyse daily returns, two examples will be utilized: 2023 (Figure 4.2 and Table 4.1), representing a bull market scenario, and 2022 (Figure 4.3 and Table 4.2), illustrating a bear market scenario.

In 2023, the AMD stock presented a change in price of 82.75$, an average return of 0.33% per day and a cumulative return of 127.98%. While in 2022, the scenario was the complete opposite, the stock presented showed a change in price of 71.93$, an average daily return of -0.32% and a cumulative return of -52.43%.

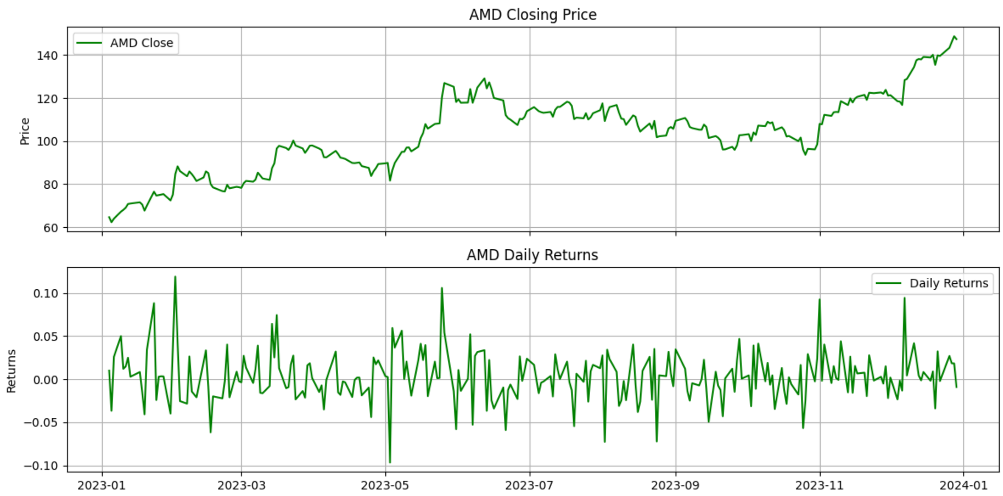

Figure 4.2. AMD stock performance in 2023.

Table 4.1. AMD stock performance metrics in 2023.

Metric	*Value*
Change in Price ($)	82.75
Average Daily Return (%)	0.33
Cumulative Return (%)	127.98

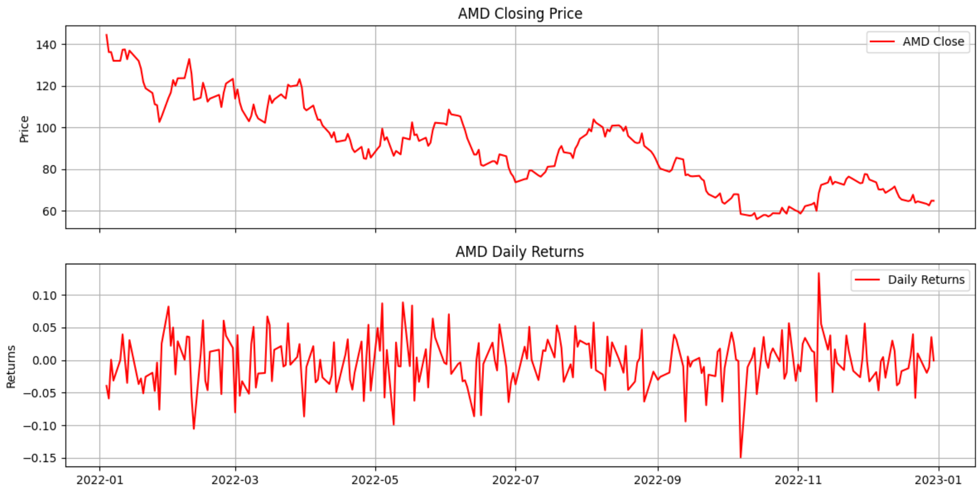

Figure 4.3. AMD stock performance in 2022.

Table 4.2. AMD stock performance metrics in 2022.

Metric	*Value*
Change in Price ($)	-71.38
Average Daily Return (%)	-0.32
Cumulative Return (%)	-52.43

4.2. Technical Indicators and Trading Strategies

The next step in the data preparation process, is the addition of new features. During this step, 12 different technical indicators were added, some with multiple timeframes and others with various features. This resulted in developing and adding 25 features to enhance the dataset (see Table 3).

Since the moving averages-based strategies are highly favoured among investors, the first technical indicators added were the Simple Moving Average (SMA), Exponential Moving Average (EMA), and Weighted Moving Average (WMA) with various periods.

Regarding the Moving Average (MA), the primary objective behind its calculation is to smooth the price data, by continually generating an updated average price. However, given the high volatility of the stock market, implementing trading rules solely based on a moving average is not advisable. This approach could lead to the generation of too many signals, some of which could be misleading.

As a result, most studies and investors usually employ certain moving averages (MAs) when applying trading rules (Stanković et al., 2015). The EMA consists of a technical indicator that tracks how the price of an asset changes over time and unlike an SMA, the EMA gives more emphasis to recent data points. Like the EMA, the WMA also assigns greater significance to recent data points. However, unlike the exponential decrease in the EMA, the weights in WMA are designed to ensure that their sum totals 1.

These indicators can be calculated using the subsequent formulas:

$$SMA_t = \frac{1}{N}\sum_{i=0}^{N-1} P_{t-i} \tag{4.1}$$

$$EMA_t = P_t \times \left(\frac{2}{N+1}\right) + EMA_{t-1} \times \left(1 - \frac{2}{N+1}\right) \tag{4.2}$$

$$WMA_t = \frac{\sum_{i=0}^{N-1}(W_i \times P_{t-i})}{\sum_{i=0}^{N-1} W_i} \tag{4.3}$$

In these formulas, N refers to the number of periods considered for the moving average calculation, P indicates the current price of the stock, W represents the weights assigned to past values in the WMA, and t denotes the current time index or period for which the moving average is being calculated.

Furthermore, this Dissertation used more momentum indicators, such as the Moving average convergence divergence, a widely known trend-following indicator. This indicator calculates the difference between two EMAs of a stock's price to show the connection between them.

The Triple EMA, also rooted in the EMA, which is crafted to exhibit even quicker responsiveness to price fluctuations, effectively signalling short-term price movements, was also used in the proposed model.

$$TEMA_t = 3 \times EMA_t - 3 \times EMA(EMA_t) + EMA(EMA(EMA_t)) \tag{4.4}$$

Additionally, in the Dissertation context, the parabolic SAR was also utilized. This indicator completes the goal of identifying potential trend reversals. Functioning as a trend following (lagging) indicator, it can establish trailing stop losses and make informed decisions about entry or exit points (Jansen, 2020).

$$SAR_t = SAR_{t-1} + \alpha(EP - SAR_{t-1}) \tag{4.5}$$

According to the study by Kumbure et al. (2022), the Relative Strength Index (RSI) with a 14-period setting is the most frequently utilized technical indicator in machine learning studies. As a result, it was incorporated into the proposed model, alongside the Williams Percent Range (Williams %R) and Stochastic Oscillator %K, which are also considered momentum indicators.

The RSI is a momentum indicator used to detect overbought and oversold market conditions by measuring the speed and change of stock price movements. The Williams %R also helps identify entry and exit points in the stock market by determining overbought and oversold levels. It measures the difference, in percentage, between the current closing price and the highest price of a given period. The Stochastic Oscillator %K serves a similar purpose as Williams %R. It compares the most recent closing price to the range of prices over a particular period.

Continuing with momentum indicators, the proposed model also incorporated the commodity channel index. This indicator determines the difference between the current price and the historical average price. The last momentum indicator used in the model was the Percentage Price Oscillator, which shows the relationship in percentage between two exponential moving averages (26–period and 12–period).

Moving to the volatility indicators, the most common one, according to Kumbure et al. (2022), is the Bollinger Bands. Bollinger Bands blend a moving average (MA) with upper and lower bands to represent the moving standard deviation (Jansen, 2020).

Regarding volume and liquidity indicators, this Dissertation incorporated two factors: the accumulation/distribution (AD) and the on-balance volume (OBV). The first indicator helps measure the cumulative flow of money into and out of a stock. On the other hand, the OBV is a cumulative indicator that establishes a connection between volume and changes in price.

Concerning Technical Indicators, the ones employed in this Dissertation are as follows. Table 4.3 provides an overview of each indicator (a total of 12), along with their respective periods and specific types, ending in 25 unique features. The figures in Appendix B illustrate the performance of various technical indicators on AMD stock during the bullish period of 2023.

Table 4.3. Technical indicators utilized.

Indicator	*Feature Descriptions & Periods*	*Type*
Bollinger Bands	Upper Band; Middle Band; Lower Band	Volatility
Stochastic Oscillator	%K; %D	Momentum
Small Moving Average (SMA)	SMA 20-days; SMA 50-days; SMA 150-days	Trend
Exponential Moving Average (EMA)	EMA 20-days; EMA 50-days; EMA 150-days	Trend
Weighted Moving Average (WMA)	WMA 20-days; WMA 50-days; WMA 150-days	Trend
Triple Exponential Moving Average (TEMA)	TEMA 20-days; TEMA 50-days; TEMA 150-days	Trend
Relative Strength Index (RSI)	RSI	Momentum
Williams %R	Williamns %R	Momentum
Parabolic SAR (PSAR)	PSAR	Trend
Acumulation/Distribution (AD)	AD	Volume
On-Balance Volume (OBV)	OBV	Volume
Moving Average Convergence Divergence (MACD)	MACD; MACD Histogram; MACD Signal line	Trend/Momentum

4.3. Lagged Variables

Although the primary objective was to use technical indicators as features for the Classifier, enhancing the proposed model with lagged returns was a must, given their consistent status as highly informative variables (Jansen, 2020), alongside technical indicators and news sentiment.

To capture historical price trends over various time periods, returns were computed for lag intervals of 1, 2, 3, 4, 5, 21, and 63 trading days. Furthermore, these returns were converted into binary format to visually represent their directional movement, with a positive return represented as 1 and a negative return represented as -1. In addition to the lagged returns, lagged adjusted close prices were also computed for various intervals.

4.4. News Sentiment

The integration of news sentiment gains significance due to its perceived effectiveness among investors in explaining stock price movements. Moreover, the efficiency exhibited by the stock market in quickly processing information further emphasizes its importance. Therefore, incorporating this variable into the model aims to uncover its influence on investment decisions, contributing to a comprehensive understanding of its prediction ability.

To get news sentiment, the FinBERT AI NLP model (Araci, 2019) was employed. This choice was motivated by the considerable challenge of conducting sentiment analysis in the financial domain, characterized by its unique language usage and limited availability of labelled data. The utilization of a pre-trained language model is pivotal in capturing stock sentiment accurately, given these complexities.

Before conducting sentiment analysis, the initial step involved the extraction of daily financial news from the eodhistoricaldata.com API. This data collection process encompassed the period spanning from 2010 to 2023.

To extract news from the API, a function, fetch_news(), was developed to fetch a maximum of 10 news articles per day for the entire date range. Since the articles aren't scattered consistently, some days may have no news, and others may have a lot. Figure 4.4 depicts a sample of news headlines extracted. When there was no news, a forward fill of the sentiment variable was applied after the sentiment analysis.

```
8              AMD Launches Its Best High-End Graphics Card in Years
2                 Is This the Best Thing to Happen to AMD Investors?
1     10 Tech Stocks to Buy Now According to Billionaire Steve Cohen
0    Millennial, Gen Z Investors Favor Green Energy, EV Stocks In Q3
8                          10 Semiconductor Stocks to Buy on the Dip
```

Figure 4.4. News example.

The following step was to write the functions that perform the concrete NLP sentiment analysis based on the article headlines retrieved.

The sentiment analysis returned three variables: positive, neutral and negative. To simplify the sentiment analysis and reduce multicollinearity, the positive, neutral, and negative sentiment variables were combined into a single variable, "sentiment." This new variable provides a normalized overall sentiment regarding AMD stock.

A “weekly sentiment” feature, resampled and aggregated every week, was also created in order to provide a more stable and thorough view of the stock market.

The line plot shown in Figure 4.5 illustrates that, despite some volatility in the adjusted close price, the sentiment regarding AMD stock remained mainly positive. This positivity might be linked to the tech rally that began in early 2023. The linear correlation between the variables was approximately 0.26, which is not a substantial (linear) relationship. This low correlation can be attributed to the previously mentioned bullish sentiment toward AI companies. In the scatter plot presented in Figure 4.6, while the adjusted close price and sentiment show some clustering of points, indicating a potential positive association, the overall dispersion of the points highlights the variability in sentiment despite the price fluctuations.

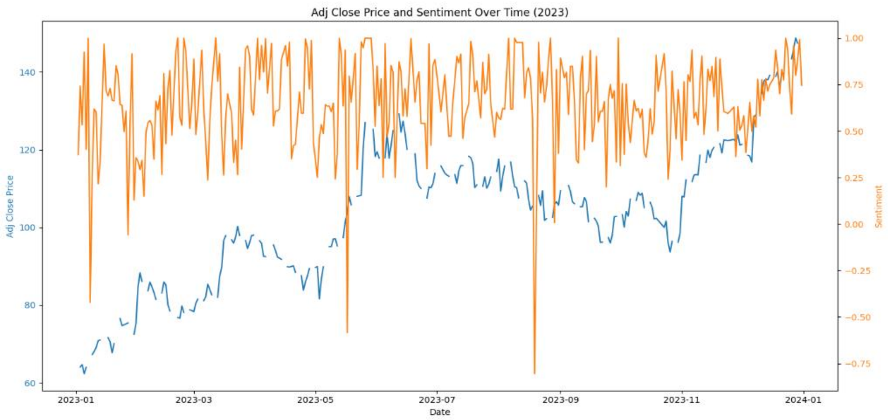

Figure 4.5. Adj. Close Price vs. Sentiment line

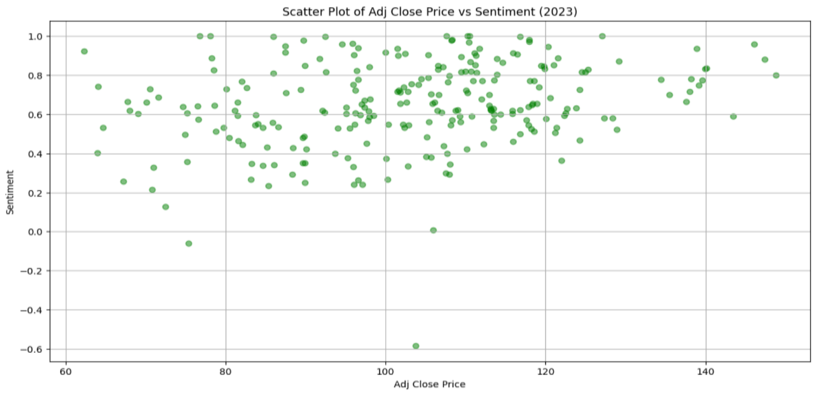

Figure 4.6. Adj. Close Price vs. Sentiment scatter

4.5. Labelling Method

After gathering the stock data and creating the necessary variables for the specified timeframe, the next step was to develop the target variables. The strategies adopted as benchmarks in this Dissertation for the labelling process, were five of the most common technical shift indicator strategies, along with a simple buy & hold strategy.

The first labelling process categorizes daily adjusted closing prices as either "Buy" or "Sell" based on their respective daily returns, or in other words, whether the market is going “Up” or “Down”. A value of -1 denotes negative returns (downward movement), while 1 signifies positive returns (upward movement). This labelling method was chosen because alternative strategies, such as those based on technical indicators, may not consistently perform well across different market conditions. While this method presents a enormous challenge for accurate predictions, a correct forecast will always result in a profitable trade, making it a highly robust approach when successful.

Moving on to the technical indicator strategies, four different methods were selected. In the short term, these strategies can be highly efficient. Creating multiple approaches to the labelling process, provides flexibility and adaptability for the proposed study.

It is important to highlight that this binary labelling approach ("Buy" or "Sell") was chosen for all strategies, including the technical indicators strategies, for several key reasons. First, it simplifies decision-making by concentrating solely on actionable outcomes, decreasing the ambiguity linked with a "Hold" class. Second, it aligns with the Dissertation’s focus on a more active trading approach, prioritizing profitability from market movements over periods of stagnation. Third, excluding a "Hold" class minimizes noise, allowing the model to better identify significant trends.

The methodologies, key indicators, and conditions for generating buy and sell conditions for the four selected technical indicator strategies are summarized below:

- **Exponential Moving Average (EMA) crossover:**
  - Relies on two EMAs: a short-term EMA and a long-term EMA.
  - **Buy signal:** Triggered when the short-term EMA crosses above the long-term EMA, indicating an upward trend.
  - **Sell signal:** Triggered when the short-term EMA falls below the long-term EMA.

The most common combination for a short-term EMA and long-term EMA is a 12-period and 26-period, respectively (Stankovic et al. 2015).

- **Moving Average Convergence Divergence (MACD) Strategy:**
  - Uses the MACD indicator and its 9-period EMA signal line.
  - **Buy signal:** Triggered when the MACD value is greater than the 9-period signal line.
  - **Sell signal:** Triggered when the MACD value falls below the 9-period signal line.
- **Stochastic Oscillator Strategy:**
  - Based on the relationship between the %K and %D lines.
  - **Buy signal:** Triggered when the %K line crosses above the %D line, indicating upward momentum.
  - **Sell signal:** Triggered when the %K line crosses below the %D line, indicating downward momentum.
- **Mean Reversion Strategy:**
  - Based on the premise that stock prices revert to their historical mean or average over time. Utilizes Bollinger Bands to identify overbought and oversold conditions.
  - **Buy signal:** Triggered when the adjusted close price falls below the lower Bollinger Band (oversold).
  - **Sell signal:** Triggered when the adjusted close price rises above the upper Bollinger Band (overbought).

To test the performance of the various strategies, two distinct periods were selected: a bullish period from the beginning of 2023 to the end of the first semester, and a bearish period from July 2022 to the end of December 2022. Both scenarios compared the trading strategies to the market returns.

Regarding the bullish period, as expected, the market returns show a strong upward trend. Both the SMA Crossover strategy (yellow line in Figure 4.7) and the EMA Crossover strategy (green line in Figure 4.7) underperform compared to the market, ending with a cumulative return of -0.1 by June 2023. The yellow line is not visible in the figure, as it overlaps with the green line, indicating that both strategies delivered identical results. The Stochastic Oscillator (red line in Figure 4.7) shows a moderate performance but is still well below the market performance. In contrast, the MACD Crossover strategy (purple line in Figure 4.7) demonstrates a solid performance, showing its effectiveness in this bullish period. Finally, although the Mean Reversion strategy (brown line in Figure 4.7) starts with a period of gains, it ends the semester with the worst performance out of all the strategies, which suggests that the Mean Reversion only works in shorter periods.

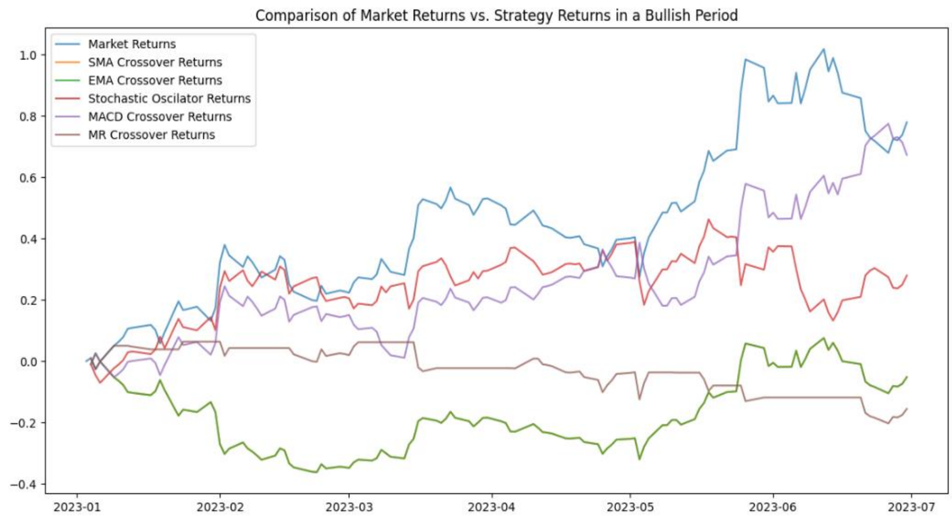

Figure 4.7. Market returns vs. Strategy returns in bullish period.

The Sharpe Ratio is a measure used to evaluate an investment's performance, such as a trading strategy, relative to its risk. It is calculated by dividing the difference between the investment's returns (R_p) and the risk-free rate (R_f), which for this case was selected the annual value of 5.48% (current yield on a 3-month US Treasury bill), by the standard deviation of the investment's returns (σ_p).

$$Sharpe\ Ratio = \frac{R_p - R_f}{\sigma_p} \tag{4.6}$$

In terms of results, the market outperformed all the trading strategies established, achieving a Sharpe ratio of 2.43. However, the MACD Crossover strategy also demonstrated strong performance with a high Sharpe ratio of 2.18, indicating effective risk management. The detailed metrics can be seen in Table 4.4. The remaining trading strategies present poor risk-adjusted performance compared to the market and the MACD Crossover strategy.

Table 4.4. Trading strategy Sharpe ratio in bullish period.

Trading Strategy	*Sharpe Ratio*
Market	2.43
Simple Moving Average	-0.05
Exponential Moving Average	-0.05
MACD Crossover	2.18
Stochastic Oscillator	1.12
Mean Reversion	-1.11

Contrary to the bullish period, the bearish period, shown in Figure 4.8, demonstrates all the trading strategies outperforming the market returns except the Mean Reversion strategy.

This analysis highlights the challenges of achieving positive returns in bear market conditions and also the possibility of achieving brief periods of high returns with certain trading strategies such as the Stochastic Oscillator and Mean Reversion Strategy. Finally, the finding emphasizes the importance of finding robust strategies like the MACD Crossover that can adapt to market volatility and complex bearish market conditions.

The Sharpe Ratio results, as shown in Table 4.5, highlight the difficulty of achieving positive risk-adjusted returns during a bearish market period. The MACD Crossover strategy performed the best, with a Sharpe ratio of 2.49, followed by the Stochastic Oscillator strategy.

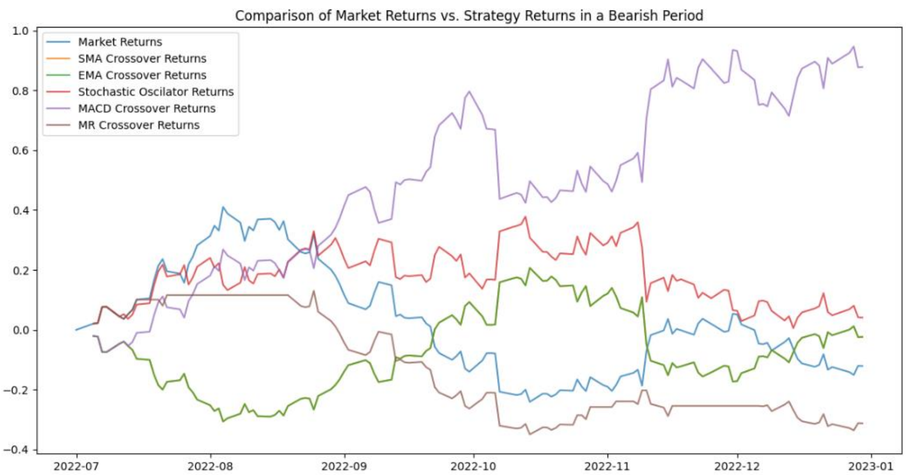

Figure 4.8. Market returns vs. Strategy returns in bearish period.

Table 4.5. Trading strategy Sharpe ratio in bearish period.

Trading Strategy	*Sharpe Ratio*
Market	-0.29
Simple Moving Average	0.09
Exponential Moving Average	0.09
MACD Crossover	2.49
Stochastic Oscillator	0.32
Mean Reversion	-1.62

After analysing both scenarios, it was apparent that the MACD Crossover strategy was the most robust, adapting well to both bullish and bearish conditions, and providing high cumulative returns. Even though the Stochastic Oscillator strategy lacked the resilience to sustain high gains over longer periods, it proved the potential for significant short-term gains.

Considering these findings, both the MACD Crossover and Stochastic Oscillator strategies were selected as the target variables, along with price direction, to be tested in the CNN model. This selection aims to influence the robust performance of the MACD Crossover strategy and the short-term effectiveness of the stochastic Oscillator strategy to augment the model's predictive accuracy.

4.6. Feature Selection

Selecting the right features for the proposed model is a challenging task but an essential aspect of any machine learning pipeline. As is often the case in many studies, including this one, some features are redundant. These redundant features can introduce noise, complicating the model's development and interpretation. It is essential to identify the relevant features for the problem at hand, ensuring that the model relies solely on high-quality inputs.

The performance of a comprehensive range of feature selection methods was analysed, including filter, wrapper, and embedded techniques and Genetic Algorithms (GA). The goal was to cross-select the top 10 most frequently selected features across these methods. Figure 4.9 presents an overview of the methods and processes applied, highlighting the sequential flow of the methodology, starting from the initial datasets and progressing through various feature selection techniques to the final datasets containing the top selected features.

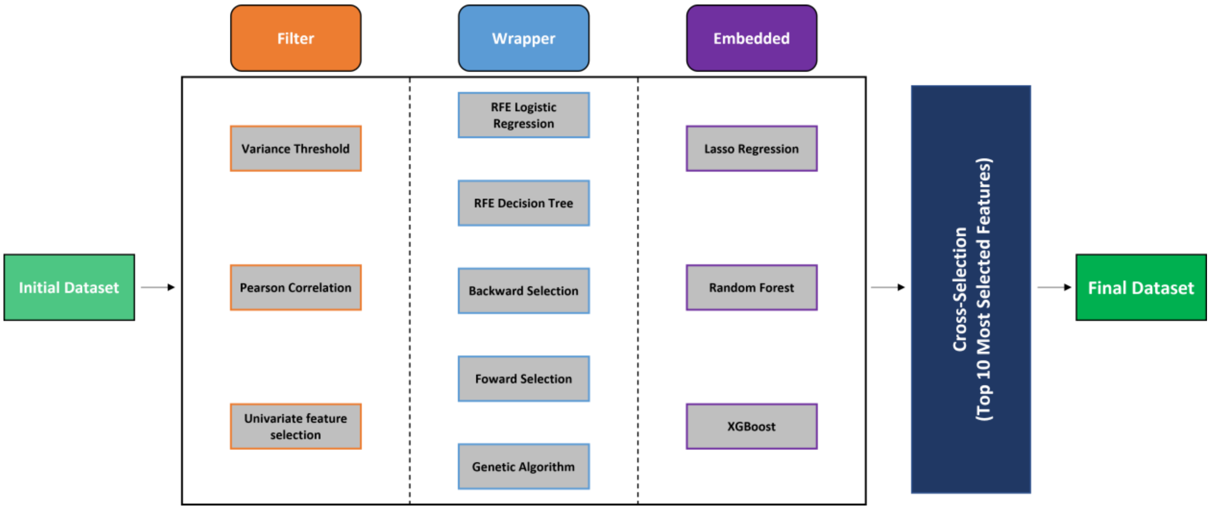

Figure 4.9. Overview of the feature selection process

Before applying any feature selection methods, the target variable in each strategy (Price Direction, MACD Crossover, Stochastic Oscillator) was shifted backward by one day. This shift warrants that the model uses today's features to predict tomorrow's results, preserving a realistic predictive framework, causality and avoiding data leakage.

Firstly, the performance of filter methods was analysed, including Variance Threshold, Correlation Coefficients and Univariate Selection.

The Variance Threshold, as the name suggests, is a feature selection method that removes features that present low variance. In this Dissertation, a variance threshold of 0.3 was selected, resulting in the selection of 57 out of 99 features for each strategy. The 99 features include the technical indicators developed specifically for this Dissertation, as well as lagged returns, lagged adjusted close prices, standard stock variables (such as volume and open prices), and, finally, news sentiment data.

The subsequent method chosen was the Pearson correlation coefficient, which is used to assess the linear relationship between features and target variables. For this Dissertation, features with a correlation of 0.2 or greater with the target variables were selected, focusing solely on continuous variables.

In the dataset with stock price direction as the target variable, no variables were selected, likely due to the complex and non-linear relationship between the features and the target. For the dataset based on the Stochastic Oscillator strategy, 3 variables were identified as relevant, while the MACD

Crossover strategy dataset yielded 11 relevant variables. The correlation matrices for the Stochastic Oscillator and the MACD Crossover strategy are presented in Appendix C.

The last filter method applied was the Univariate feature selection, which evaluates each feature independently to test its relationship with the target variables. This method utilizes the ANOVA F-test to assess if the means of the target variables vary substantially across the values of a feature. After the test, the features are ranked based on their significance with the target variable, in this case, the top 10 features for each dataset. The features selected for each dataset are displayed in Appendix B.

Next, the wrapper methods were tested. Recursive Feature Elimination (RFE) was applied using Logistic Regression and a Decision Tree Classifier as estimators. RFE is a method that selects the most relevant features by recursively considering smaller sets of features and applying an estimator to assess the importance of each feature. In this Dissertation, for each estimator, the RFE selected the top 10 features based on their importance.

Next, backward and forward selections were applied. These are standard techniques used in feature selection for developing predictive models. They are types of stepwise regression methods that help determine the most relevant features for a model.

Forward selection is an iterative method that starts with no features and increases them one at a time based on the accuracy of the logistic regression classifier. This process continues until adding new features no longer significantly improves the model's performance. In contrast, the backward elimination, starts with all available features and removes them one at a time based on the model’s accuracy.

For this Dissertation, both methods were employed to leverage their specific benefits. Backward elimination is beneficial for large datasets as it effectively reduces features, while forward selection incrementally builds the model, warranting that each added feature contributes to improved accuracy. In both methods, the top 10 features were selected based on their influence.

The final wrapper method implemented was a genetic algorithm with a decision tree classifier. As mentioned in the literature review by Xue et al., (2016), genetic algorithms have gained widespread acclaim in the feature selection community. Genetic algorithms are optimization methods inspired by natural selection and genetics. The features selected by the Genetic Algorithm for each dataset are displayed in Appendix C.

To begin, a population of possible solutions (individuals) is generated; in this Dissertation, the population size was set to 100. Each individual is evaluated using a fitness function, which in this context is the accuracy of the decision tree classifier. Fitter solutions, those with higher accuracy, have a higher likelihood of being chosen for the next generation, ensuring that better solutions propagate. The next step is crossover, which involves mixing pairs of selected solutions to exchange parts of their genetic information and establish new individuals. In this Dissertation, a crossover probability of 0.5 was chosen. Additionally, random mutations are introduced to some of the newly created individuals with a mutation probability of 0.2. This helps preserve genetic diversity within the population and allows the algorithm to explore new areas of the solution space.

These steps are repeated for many generations. In this Dissertation, the algorithm was set to stop if there was no improvement for 20 consecutive generations. The features selected by the Genetic Algorithm for each dataset are displayed in Appendix C.

The last type of feature selection technique tested for the considered datasets was the embedded methods. Unlike wrapper methods, which evaluate combinations of features using a predictive model, embedded methods integrate feature selection straight into the model training process.

The Lasso Regression (L1 Regularization) was the first embedded method applied to the datasets to identify the most relevant features. This technique adds a penalty equivalent to the absolute value of the magnitude of coefficients, reducing some of them to zero. By applying that penalty, the Lasso Regression not only does regularization but also feature selection, keeping only the features with non-zero coefficients.

The last two techniques applied before cross-feature selection were Random Forest and XGBoost feature importance. Random Forest, although it is a machine learning method, intrinsically performs feature selection by evaluating the importance of each feature. The XGBoost is another ML method that utilizes decision trees. The main difference between both learning methods is that Random Forest uses mean decrease in impurity and accuracy to determine feature importance. And the XGBoost utilizes metrics like gain, frequency, and cover to estimate feature importance. The results of both methods for each dataset, along with the others, are presented in Appendix C.

After applying 11 feature selection methods, the next phase was to perform a cross-selection of features. The aim is to determine and select the top 10 features most frequently chosen by each feature selection method for each dataset. This approach warrants that the most consistently significant features across different methods are identified and employed in the image creation.

The final version of each dataset is displayed in Table 4.6. The selected variables for the Stock Price Direction dataset emphasize volume, price and momentum indicators to identify market activity and its direction. Features like volume, Accumulation/Distribution Line (AD), and On-Balance Volume (OBV) are significant in understanding trading activity and historical price movements, essential for predicting future market direction.

In the Stochastic Oscillator dataset, while there is still certain emphasis on volume-related variables such as the AD and OBV, the primary focus shifts to momentum indicators, such as the %K, %D and Williams %R, which are essential for analysing price direction and generating buy and sell signals. To conclude, the MACD Crossover dataset strongly relies on MACD-related features and momentum indicators to detect trading opportunities. The MACD histogram, %K, %D, and the MACD signal line are fundamental in identifying trend strength and changes, enabling the detection of bullish and bearish market conditions.

Table 4.6. Final version of each dataset.

Features	*Stock Price Direction Dataset*	*Stochastic Oscillator Dataset*	*MACD Crossover Dataset*
1	Volume	%D	MACD Histogram
2	AD (Accumulation/Distribution Line)	Williams %R	%K
3	OBV (On-Balance Volume)	%K	%D
4	Lagged Adjusted Close day 2	Returns	Williams %R
5	Lagged Return Day 1	Volume	Returns
6	Lagged Return Day 2	PPO (Percentage Price Oscillator)	Lagged Return Day 1
7	Close	AD (Accumulation/Distribution Line)	Volume
8	Adjusted Close	OBV (On-Balance Volume)	OBV (On-Balance Volume)
9	%D	MACD Histogram	MACD Signal Line
10	TEMA 20-day (Triple Exponential Moving Average)	Lagged Return Day 1	PPO (Percentage Price Oscillator)

4.7. Image Creation

After selecting the most important features in each dataset, employing the 2D-CNN is essential for transforming the time-series data into image. To perform this transformation, three different approaches were explored and performed.

The approaches explored were the Gramian Angular Field (GAF), Recurrence Plots (RP) and Markov Transition Field (MTF). As seen in the literature review, by converting time series data into images, these CNN models can accomplish effective results when utilized for various financial tasks, including stock price prediction.

The first approach employed was the GAF. In recent years, this method has become a widespread technique for converting time series data into images to leverage the advantages of CNNs for time series prediction. Proposed by Wang and Oates (2015), this time series encoding method nets the temporal relationships between each time point within the data by normalizing the time series and representing its values as coordinates in a polar coordinate system.

Applying this base, the Gramian Angular Summation Field (GASF) and Gramian Angular Difference Field (GADF) matrices can be established.

Xu et al. (2023) explain that GASF and GADF matrices capture the correlations between time series values at various periods using cosine and sine operations, creating an image representation of the temporal signal. However, this operation may risk losing some information. The main difference between the two matrices is that GASF highlights cumulative patterns, while GADF emphasizes variations, resulting in different representations of the time series.

For this Dissertation, it was decided to generate multivariate GADF images with overlapping windows from the AMD stock data. It was proposed a window of 22 days, considering a swing trader's perspective, which is a style of trading that attempts to capture short to medium term gains using mainly technical indicators. Selecting a 22-day overlap with a step size of 1 ensures that maximum overlap is achieved, facilitating CNN in recognizing patterns and trends that occur over longer periods. Additionally, with this overlap, the model is probably better equipped to detect subtle shifts and anomalies and grant more accurate and robust predictions for future time windows. This continuity permits CNNs to use the shared information across overlapping windows to reinforce learning and improve their capacity to generalize across different periods.

The core distinction between classical GAF generation and the method proposed in this Dissertation is that the GAFs will be multivariate. This is accomplished by creating individual GAF images for each variable and combining them into multichannel images. Creating a multivariate image allows the CNN model to learn from the relationships between different variables, increasing its ability to capture complex patterns and dependencies within the data.

With the proposed method, each generated image produces a distinct representation of the time series, visualizing the progression of different variables up to the 22nd day. This thorough visualization allows the model to analyse the multivariate relationships and trends, ultimately predicting a more robust trading decision for the next day.

The images generated for each dataset, representing the first variable in the first segment, are displayed below in Figure 4.10. The three images reflect the effectiveness of the GAF in capturing intricate details from the time series data.

Both the images for the Stochastic Oscillator dataset and the MACD Crossover dataset show clear patterns with well-defined regions, highlighting the GAF's capability to capture temporal dependencies effectively. In contrast, the image corresponding to the stock price direction dataset reveals a more complex pattern, which is naturally more challenging to predict. This complexity is anticipated, as the Stochastic Oscillator and MACD Crossover datasets include variables that are more directly correlated with the target variable, enhancing the GAF's ability to identify and represent these temporal relationships.

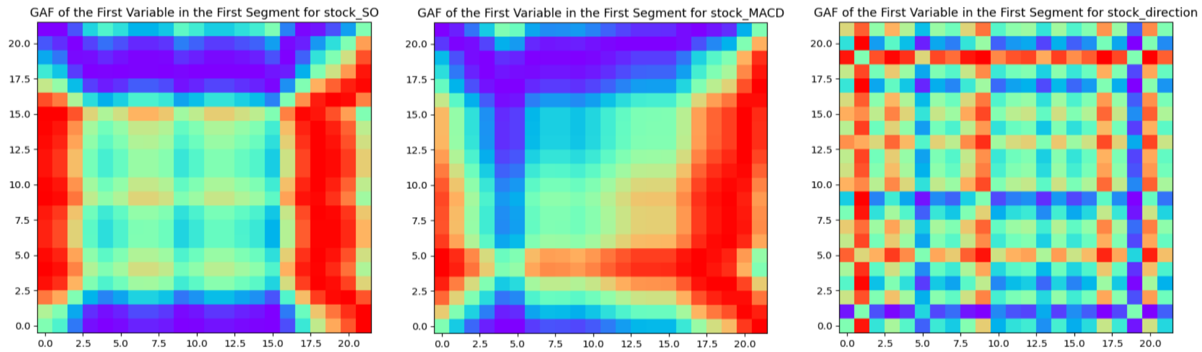

Figure 4.10. GAF Images for each dataset.

The following method for encoding time series data into images is recurrence plots. Introduced by Eckmann et al. in 1987, recurrence plots provide a way to visualize the recurrences of a dynamical system. This method offers a simple and easily estimable approach to characterize the system's dynamics. Initially, it was based solely on the measured time series and was intended to complement other contemporary methods.

Over the years, as highlighted by Goswami (2019), the use of recurrence plots has increased significantly due to their intuitive visual appeal and the growing interest in nonlinear time series analysis. Recurrence plot-based methods have been applied to a wide range of problems, including finance, which is the focus of this Dissertation, to detect and visualize recurring patterns. The ability of recurrence plots to provide a detailed visual representation of temporal relationships makes them a valuable tool in time series prediction.

For this Dissertation, similar to the approach used for generating GAF, the aim was to develop multivariate recurrence plots to capture the temporal dynamics of various variables within the time series data. Following the methodology used for Gramian Angular Fields (GAFs), we proposed using an overlapping window of 22 days with a step size of 1 day. Consequently, each new window overlaps the previous one by 21 days. Combining each variable recurrence plot into a multichannel recurrence plot makes it possible to capture the complex relationships among the variables over time, while maintaining continuity with the overlapping window.

For this approach, a threshold method was set to 'point', and a percentage of 20 was selected. This mechanism establishes the points in the phase space that are believed to be recurrent by measuring the distances between them and picking the top 20% closest pairs. This allows the recurrence plots to highlight significant recurrences by filtering out noise.

Figure 4.11 displays the images generated for each dataset, representing the ninth variable in the ninth segment. For the recurrence plot of the Stochastic Oscillator dataset, the structured pattern with a diagonal cross indicates regular and predictable recurrence. The recurrence plot for the MACD Crossover dataset also reinforces the idea of predictability. Strong diagonal bands suggest a significant amount of periodic behaviour, indicating that the ninth variable in this segment presents regular and recurring patterns over time. On the other hand, the recurrence plot for the Stock Price Direction dataset, although it has some periodic behaviour, still exhibits complex and non-linear dynamics.

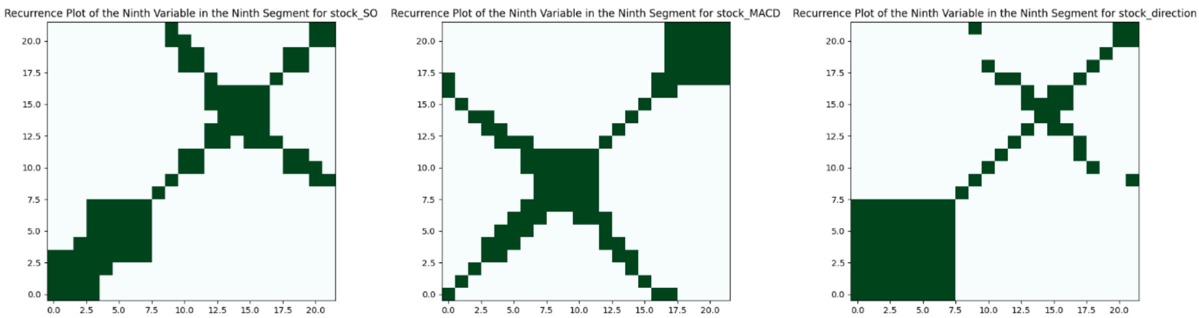

Figure 4.11. RP Images for each dataset.

The Markov Transition Field (MTF) was the final image generation method. As noted by Lu et al. (2018), the MTF enhances the Markov matrix by aligning each probability according to the temporal order of the time series. While the Markov matrix encodes the dynamical transition statistics, it does not account for the conditional relationships and temporal dependencies between time steps. The MTF, as described by Wang and Oates (2015), sequentially represents the Markov transition probabilities to maintain information within the time domain. In addition, Wang and Oates (2015) demonstrated that the MTF method produces highly competitive results compared to other time series classification approaches.

This Dissertation closely followed the parameters selected from the approaches above. Once again, the aim was to generate multivariate images, specifically multivariate MTF images, using a 22-

day overlapping window with a step size 1. Each window captures the transition dynamics of the time series over 22 days, and the overlapping ensures that temporal dependencies are preserved.

Figure 4.12 exhibits the images generated for each dataset, representing the third variable in the third segment. For the MTF of the Stochastic Oscillator dataset, the image displays a mix of structured and random transitions, indicating some predictability and some volatility. The MTF matrix of the MACD Crossover demonstrates more predictable patterns, such as some high-probability transitions (yellow region), which ensures higher consistency for time series prediction. Lastly, the MTF matrix for the Stock Price Direction dataset, exhibits the most complex and structured patterns, which suggests strong temporal dependencies with high probability transitions (yellow and green regions).

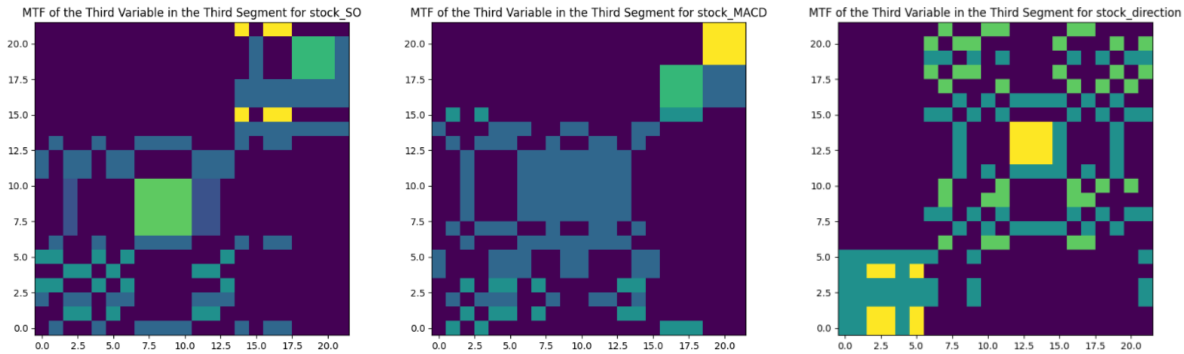

Figure 4.12. MTF Images for each dataset.

The generated images in all approaches (GAF, Recurrence Plot, MTF) exhibit the potential of these methods in capturing structured patterns and powerful temporal dependencies, indicating high predictability. However, they also reveal that some variables and specific time windows can be less clear, making predictions more challenging. This is due to the inherent volatility and randomness of the stock market. By using multivariate images, some images may compensate for others, thus enhancing overall predictability.

Chapter 5

Modelling and Evaluation

The next phase in the CRISP-DM methodology is modelling. For this Dissertation, three models were employed, all utilizing Deep Learning techniques. The first model used was an LSTM, which served as the baseline for the study. Next, a CNN model was applied to the various image techniques previously developed and finally, to enhance the accuracy of the CNN model, a GAN architecture, the TimeGAN, was employed to generate data simulating different market scenarios. Each of these models was applied to the created datasets: Stock Price Direction dataset, MACD Crossover strategy dataset and Stochastic Oscillator strategy dataset.

5.1. Train-Test Splitting and Performance Evaluation Metrics

Before applying any model, it was essential to define the train and test sets and the metrics for evaluating and comparing the model's performance. For all models tested, two train-test split methods were performed on the three datasets. Additionally, a validation set was included during training to monitor the model's performance on unseen data.

The validation set was created by splitting 10% of the training data, guaranteeing that it remained separate from the training and test sets. This set granted an intermediary checkpoint to evaluate the model's generalization ability after each epoch. The validation loss was closely monitored, and early stopping was employed to prevent overfitting. Training was halted if no improvement in validation loss was observed for 20 consecutive epochs, and the best-performing model weights were restored. This process warranted that the model did not overtrain on the training data while maintaining optimal performance on unseen data. Including the validation set was fundamental for fine-tuning the model and ensuring good generalization.

The datasets contained complete data without missing values from July 6, 2015, to December 28, 2023. However, data prior to July 6, 2015, had missing values due to the generation of features such as technical indicators and sentiment data, which needed historical data for calculation.

In the first approach, a train-test split of 90/10 was performed on all datasets. Given the unpredictable nature of stock data and the focus on short-term predictions, this split was considered suitable. For the tabular and image data, the split resulted in 1903 training samples and 212 test samples, which can be regarded as not a high volume of data.

Regarding the second approach, shown in Figure 5.1, to accomplish a more robust evaluation, a Rolling Cross-Validation was applied to the datasets. Rolling Cross-Validation is a technique that maintains the temporal order of the data, which is fundamental for time series data, avoiding data leakage and over-optimistic performances.

This approach involves multiple evaluations, four in this Dissertation, offering a more comprehensive understanding of the model's performance across different market scenarios and assessing its stability. In each fold, the training set includes the start of the dataset up to a specific time t, and the validation set incorporates data points immediately after t. The process is repeated four times, moving the split point forward in each fold. For instance, in the first fold, the model trains data until t_1 and tested on data from $t_1 + 1$ to t_2. In the second fold, the train data include observations up to t_2, and the validation data ranges from $t_2 + 1$ to t_3, and so on, until the last fold.

In conclusion, Rolling Cross-Validation allows for a better evaluation of the model's robustness and reliability, by closely simulating how the model will be used in practice. In the case of the time-series data converted in images, a walk-forward validation was used, which is almost identical to the Rolling Cross-Validation but is suited for image-like data.

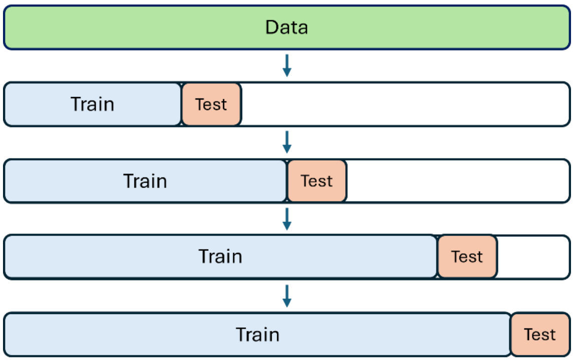

Figure 5.1. Rolling Cross-Validation/Walk-Forward Validation.

Regarding the metrics selected to evaluate the model's performance, several generally used classification metrics were picked. The first metric is accuracy, which is the ratio of correctly predicted observations to the total observations.

$$Accuracy = \frac{True\ Positives\ (TP) + True\ Negatives(TN)}{TP + TN + False\ Postives\ (FP) + False\ Negatives(FN)} \quad (5.1)$$

Next were Precision and Recall (Sensitivity). Precision measures the ratio of correctly predicted positive instances to the total predicted positives. Recall, on the other hand, measures the ratio of correctly predicted positive instances to all the instances in the actual class.

$$Precision = \frac{TP}{TP + FP} \quad (5.2)$$

$$Recall = \frac{TP}{TP + FN} \tag{5.3}$$

Lastly, the metric Area Under the ROC Curve (AUC-ROC) was chosen. This metric helps understand the trade-offs between the true positive rate and false positive rate at different levels, indicating how well the model can distinguish between classes. A higher AUC means the model better distinguishes between positive (Buy) and negative (Sell) classes. This metric is valuable as it helps assess the model's performance in different market scenarios.

5.2. Long Short-Term Memory (LSTM) Baseline Model

The first model developed for this Dissertation was an LSTM model, which served as the baseline. LSTM neural networks are a specialized architecture of recurrent neural networks (RNNs), which are designed to better capture long-term dependencies in sequential data. While traditional RNNs can theoretically use information from long sequences, they often struggle with long-term dependencies in practice. This difficulty surfaces from the problem of gradient vanishing or exploding during backpropagation over many time steps, as highlighted by Jansen (2020).

Multiple RNN design techniques have been developed to address this challenge, with the most successful ones employing gates trained to control how much past information is preserved in the current state and when to reset this information. The LSTM is the most popular example of this approach. It uses input, output, and forget gates to manage dependencies between elements in the input sequence, allowing recurrent decisions. Precisely, the forget gate controls how much of the cell's state should be discarded, the input gate updates the cell state based on the current input and previous hidden state, and the output gate filters the updated cell state to produce the final output, as explained by Jansen (2020). This gated mechanism allows LSTMs to successfully handle long-term dependencies, making them fit for stock price prediction.

Since the LSTM was not the primary focus of this Dissertation, hyperparameter tuning for this model was not emphasized; its primary role was to provide a baseline performance.

The LSTM model for all datasets was constructed using the Sequential API from Keras, featuring a simple architecture containing a single LSTM layer with 50 units and a ReLU activation function. This was followed by a dense layer with a sigmoid activation function to create binary classification outputs. The model was compiled with the Adam optimizer and binary cross-entropy loss, with accuracy tracked as a performance metric. To prevent overfitting, early stopping was employed, monitoring the validation loss with a patience of 10 epochs.

The training process involved fitting the model to the training data of all the datasets for 200 epochs with a batch size of 32. A validation set compromising 10% of the data was used to validate the model during training.

5.3. Convolutional Neural Network (CNN)

As mentioned in the beginning of this Dissertation, CNNs were initially developed to process image data and in computer vision has achieved exceptional performance. As noted by Jansen (2020), time-series data has a grid-like structure very alike to that of images. With that in mind, CNNs have been successfully applied to one-, two- and three-dimensional representations of temporal data. In this Dissertation, to take advantage of the grid-like structure of multivariate time-series data, where each time series is a channel, a 2D CNN was developed to leverage its ability to detect local patterns and relationships between the different channels.

Regarding the theory behind the CNNs, as explained by Jansen (2020), these networks are a specialized type of neural networks that excel at learning spatial hierarchies in data, making them effective at image and sequence data tasks. Remarkably similar to feedforward neural networks (NNs), CNNs consist of units with weights and biases as parameters, which are adjusted during the training process to optimize the network's output for a specific input.

In the time-series context, as stated above, CNNs leverage the assumption that local patterns (could be represented as autocorrelation or other non-linear relationships at relevant intervals) are essential to predict the outcome.

In this Dissertation, a 2D CNN architecture was utilized to leverage its capability to capture local patterns in stock data, such as technical indicators, and compare its performance with the baseline model (LSTM) and various image generation techniques. Each stock dataset (Stock Price Direction, MACD crossover strategy, Stochastic Oscillator strategy) was used to train and test the 2D CNN. For each stock dataset, the relevant features, already transformed into GAFs, RPs or MTF, were selected, and the number of channels was defined based on these features used to generate the images (10 channels). The CNN model was developed utilizing the Sequential API from Keras, with two convolutional layers followed by pooling layers, a flattening layer, and, finally, dense layers.

The original CNN architecture consisted of a first convolutional layers with 32 filters, a kernel size of 3x3, followed by a max pooling layer of 2x2. The second convolutional layer had 64 filters, kernel size of 3x3, and a max pooling layer of 2x2. The input matrices for the CNN had dimensions of 22 × 22 × 10, representing 22 time-steps, 22 windows, and 10 features. The model constructed incorporated a flattening layer, a dense layer with 64 units, and a final dense layer with a sigmoid function to deliver binary classification outputs. Then, the model was compiled utilizing the Adam optimizer, binary cross-entropy loss function, and accuracy as the selected performance metric. Lastly, during the training process, the CNN was fitted to the training data for 200 epochs with a validation split of 10%. Table 5.1 provides the parameters selected for the initial CNN:

Table 5.1. Parameters for the initial approach of the CNN model.

Layer Type	*Parameters*	*Activation Function*
Convolutional Layer 1	32 filters, kernel size 3x3	ReLU
Max Pooling Layer 1	Pool size 2x2	-
Convolutional Layer 2	64 filters, kernel size 3x3	ReLU
Max Pooling Layer 2	Pool size 2x2	-
Flattening Layer	-	-
Dense Layer	64 units	ReLU
Output Dense Layer	1 unit	Sigmoid
Compilation	Optimizer: Adam, Loss: Binary Cross-Entropy	-
Training	Epochs: 200, Validation Split: 10%	-

In the initial approach, the model exhibited significant overfitting, especially with the Stock Price Direction dataset and the Stochastic Oscillator strategy dataset. In order to resolve this issue, a more sophisticated CNN architecture was developed.

This second approach was given special focus in the study, requiring careful attention to achieve optimal results. The initial CNN's tendency to easily overfit to two of three datasets showed how critical is hyperparameter tuning to guarantee the best possible performance across all datasets.

To mitigate overfitting and improve model's generalization capacity, several adjustments were made. Batch normalization layers were included after each convolutional layer to soothe the learning process and reduce sensitivity to initialization. In addition, dropout layers were also employed to randomly drop units during the training process, helping to reduce overfitting. Finally, an early stopping callback was applied to monitor the validation loss and stop training if it did not improve for 20 consecutive epochs, thus preventing overfitting.

This enhanced approach conducted multiple experiments with various hyperparameters, including a grid search employing the Keras Classifier form TensorFlow. The model architecture featured two convolutional layers with diverse combinations of filters (16 and 32, 32 and 64, 64 and 128) to capture different levels of feature complexity. Regarding the kernel size in each convolutional layer, three kernel sizes were tested (2x2;3x3;5x5). Each convolutional layer was followed by a max pooling layer with a pool size of 2x2, batch normalization layers of 16, 32 or 64 units, and a dropout layer with rates of 0.1, 0.25 or 0.5 to observe their influence on the training process. Identical to the initial approach, both convolutional layers used a ReLU activation function and were followed by a flattening layer to convert the 2D matrix into a vector. Before the final dense layer, an additional batch normalization and dropout layers were employed with various rates.

To conclude, the model was built and trained with the same optimizer and loss function as the initial approach. Two hundred epochs were used for training, but due to the employment of early stopping, training was halted earlier when no improvement in validation loss was observed for 20 consecutive epochs, as previously mentioned. By realizing extensive experiments with a vast range of hyperparameters, it was possible to optimize the model's performance. Table 5.2 provides the parameters selected for the CNN model constructed.

Table 5.2. Parameters for the final approach of the CNN model.

Layer Type	*Parameters*	*Activation Function*
Convolutional Layer 1	Filters: 16,32,64 Kernel size: 2x2,3x3.5x5	ReLU
Max Pooling Layer 1	Pool size: 2x2	-
Convolutional Layer 2	Filters: 32,64,128 Kernel size: 2x2,3x3.5x5	ReLU
Max Pooling Layer 2	Pool size: 2x2	-
Batch Normalization	Units: 16,32,64	
Dropout Layer	Rates:0.1,0.25,0.5	
Flattening Layer	-	-
Dense Layer	Units: 64	ReLU
Output Dense Layer	Units: 1	Sigmoid
Compilation	Optimizer: Adam, Loss: Binary Cross-Entropy	-
Training	Epochs: 200 Validation Split: 10%	-

5.4. Generative Adversarial Network (TimeGAN)

This model was constructed to generate synthetic data to improve the CNN's model accuracy and to produce data realistic enough to simulate various market scenarios. As mentioned in the literature review, in 2019, Yoon and Jarret proposed a novel GAN architecture to model time-series data, the TimeGAN. This approach allows to generate data that capture the various datasets feature distributions within each time-point and catch the complicated dynamics of those features across time.

TimeGAN differs from other GAN architectures because it introduces the concept of supervised loss, meaning that the model is incentivized to catch time conditional distribution within the data by utilizing the original data as supervision. Additionally, the presence of an embedding network reduces the adversarial learning space dimensionality. Another benefit of utilizing the TimeGAN architecture is its lower sensitivity to hyperparameter changes and greater stability during training.

The TimeGAN consists of four main network components: the embedding function, the recovery function, the sequence generator, and the sequence discriminator. The first two components, known as the autoencoding components, are trained together with the latter two, known as the adversarial, as part of the overall architecture. This process allows the TimeGAN to simultaneously learn to encode features, generate representations, and iterate across time, as explained by Yoon and Jarret (2019). In addition, the embedding network supports the latent space, the adversarial network works within this space, and the latent dynamics of both real and synthetic data are synchronized through a supervised loss.

In this Dissertation, the TimeGAN was utilized solely to simulate the bull run using data from 2023. This scenario was chosen to ensure consistency, as the same data used in the CNN model was also employed for the TimeGAN simulation. This dataset was scaled using the Min-Max Scaler to ensure consistency and reliability in the simulation.

To simulate this scenario, the YData Fabric platform was utilized to generate precise and realistic synthetic data (YData, n.d.). The parameters from the model are shown in Table 5.3.

Table 5.3. Parameters for the TimeGAN.

Parameter	*Value*
Sequence Length	22
Number of Sequences	11
Hidden Dimension	22
Gamma Value	1
Noise Dimension	32
Layer Dimension	128
Batch Size	128
Learning Rate	$5e^{-4}$
Number of Epochs	500
Samples Generated	250

Chapter 6

Results and Discussion

6.1. Baseline LSTM

In this chapter, the results from the proposed models are analysed and discussed. A total of 11 models were evaluated, considering various frameworks and datasets. Starting with the baseline model, the LSTM, all datasets were tested using this architecture. The model was tested using two different approaches: a standard train-test split of 90/10 and Rolling Cross-Validation.

In the first approach, the Stochastic Oscillator strategy presented an accuracy of 66% on the test set. While this may not seem very high, it is a reasonable performance given the natural difficulty and unpredictability of the stock market.

Regarding the recall metric, the LSTM exhibited a higher recall for class 0 (71%), indicating better performance in identifying sell or no-buy signals. Precision was balanced between the classes, with an overall score of 65%. The AUC-ROC score was 66%, indicating that the model is reasonably capable of distinguishing between buy and sell signals.

When using the Rolling Cross Validation, the Stock Oscillator strategy achieved an overall accuracy of 59%, gradually improving accuracy until the final fold. The recall and AUC-ROC metrics followed a similar trend, reaching a value of 67% and 63%, respectively, in the last fold. Regardless of this performance in the final fold, the average recall was 50% and the average AUC-ROC was 59%. For the precision metric, the overall performance was 62%, with the best performance happening in the second fold at 70%.

Although the Rolling Cross-Validation technique achieved moderate performance, with the overall metrics hovering around 60%, it still shows the inherent difficulty of predicting stock strategy movements and achieved a worse performance than the train-test split approach. This performance difference can be credited to the fact that in Rolling Cross-Validation, each fold progressively raises the amount of training data, but the initial folds have less data to train on compared to the train-test split. Another factor is the strong temporal dependencies in stock market data.

In conclusion, while the Rolling Cross-Validation is a good technique to measure the model robustness and performance across multiple subsets of data, the complexity and variability of stock market data in this case, may benefit more from a stable, consistent, and longer training approach such as the train-test split. Table 6.1 displays the results for the Stochastic Oscillator strategy.

Table 6.1. LSTM Stochastic Oscillator strategy results.

	Stochastic Oscillator Strategy		
Metrics	Train-Test Split	Rolling CV Last Fold	Rolling CV Overall
Accuracy	66%	63%	59%
Precision	65%	60%	62%
Recall	60%	67%	50%
AUC-ROC	66%	63%	59%

In the case of dataset of MACD Crossover strategy, the Rolling Cross-Validation approach achieved an accuracy of 83%, which is two percentage points higher than the 81% accuracy obtained from the train-test split approach. Although the metrics are very similar in both methods, the difference in accuracy can be primarily justified by the difference in recall. In the train-test split approach, the model had a recall of 68% for identifying buy signals, while the Rolling Cross-Validation approach attained a recall of 90% in the last fold.

In general, the model for the MACD Crossover strategy presents a strong performance with a high precision for buy signals, which are essential for trading strategies. Even though the model presents a substantial AUC-ROC value, there is room to improve the recall to guarantee it identifies more actual buying opportunities. The metrics for the MACD Crossover strategy are shown in Table 6.2.

Table 6.2. LSTM MACD Crossover strategy results.

	MACD Crossover Strategy		
Metrics	Train-Test Split	Rolling CV Last Fold	Rolling CV Overall
Accuracy	81%	87%	83%
Precision	93%	86%	88%
Recall	68%	90%	77%
AUC-ROC	81%	87%	83%

The improvement with the Rolling Cross-Validation can be clarified by the progressive increase in training data with each fold, allowing the model to learn more effectively from a larger dataset over time. Unlike the Stochastic Oscillator strategy, where the train-test split was the more effective approach, the greater performance of the MACD Crossover strategy with Rolling Cross-Validation can be credited to the nature of the MACD signals. MACD signals are more stable and predictable, with less variation, specifically in their ability to accurately recall buy signals. This allows the MACD Crossover strategy to perform well even from the initial folds. This underlines the importance of selecting the appropriate validation approach for each strategy.

The last strategy tested was the Stock Price Direction, which by default is the hardest to predict due to the inherent variability of the stock price market and because it is based solely on stock returns rather than any technical strategy.

As expected, this strategy achieved the worst performance. The accuracy in either approach didn't go beyond the 53% mark. However, the Rolling Cross-Validation approach presented a much better performance in terms of recall. With the train-test split technique, the model achieved a recall of only 12%, indicating that it missed many actual buy signals, making it ineffective at capturing buy signals. Additionally, the train-test split presented an accuracy of 49%, which is lower than random guessing.

Overall, the Rolling Cross-Validation demonstrated a modest improvement in the Stock Price Direction strategy, likely by allowing the model to learn from a more varied dataset. Regardless of these improvements and the inherent complexity of stock price direction classification, the results are inadequate. With the use of CNNs, different image generation techniques, and synthetic data, the study aims to achieve more reliable predictions. The metrics supporting these results for the Stock Price Direction strategy are stated in Table 6.3.

Table 6.3. LSTM Stock Price Direction strategy results.

	Stock Price Direction Strategy		
Metrics	Train-Test Split	Rolling CV Last Fold	Rolling CV Overall
Accuracy	49%	52%	53%
Precision	58%	52%	55%
Recall	12%	47%	58%
AUC-ROC	51%	52%	52%

6.2. CNN

Regarding the performance of the CNN model across multiple strategies, two architectures were tested. The initial architecture exhibited signs of overfitting, particularly with the Stock Price Direction dataset and the Stochastic Oscillator strategy dataset. To mitigate this issue, a more refined architecture was implemented. The results analysed and discussed in this section refer to the performance of this improved model.

For the improved CNN, two testing approaches were employed: a standard 90/10 train-test split and walk-forward validation. Furthermore, all developed image generation techniques were applied and tested using this model (GAF, RP, MTF). To optimize the performance of the best-performing image generation method, a grid search was conducted to fine-tune the hyperparameters.

6.2.1. CNN-GAF

Firstly, this model was applied to the GAF generated images for each trading strategy. In the train-test split approach, the Stochastic Oscillator strategy accomplished an accuracy of 58% on the test set, which is lower when compared to the baseline LSTM model. Other metrics, such as precision, recall, and AUC-ROC, underperformed relative to the LSTM baseline, suggesting that classifying buy and sell signals in GAF image data using the Stochastic Oscillator strategy is more challenging.

Nevertheless, this idea turns when analysing the results from the walk-forward validation, where the strategy accomplished an accuracy of 67%, very similar to the best result from the LSTM baseline. The major improvement was in the AUC-ROC, which reached nearly 70%, indicating the model's strong ability to differentiate between buy and sell signals.

Overall, the rest of the metrics were on par with the LSTM baseline. The use of GAF images applied to the CNN model did not show a significant improvement when compared to LSTM. This could be related to the Stochastic Oscillator strategy's high volatility, with frequent buy and sell signals during the 22-day intervals captured in the images, making it more difficult for the model to discover spatial patterns successfully. The Stochastic Oscillator strategy is known to have frequent fluctuations since it reacts to short-term price movements, creating noisy data, which makes it harder for the model to identify clear patterns.

Although there is no improvement compared to the LSTM baseline, it is still a solid performance for the trading classification model. The metrics for the Stochastic oscillator strategy are shown in Table 6.4.

Table 6.4. CNN-GAF Stochastic Oscillator strategy results.

	Stochastic Oscillator Strategy		
Metrics	Train-Test Split	Walk Forward Last Fold	Walk Forward Overall
Accuracy	58%	67%	57%
Precision	51%	65%	57%
Recall	65%	58%	59%
AUC-ROC	61%	69%	63%

Regarding the second strategy, the MACD Crossover strategy, all approaches yielded strong results, consistently achieving at least 90% accuracy on the test set. Furthermore, the CNN-GAF model revealed high precision, with a median of 91%, showcasing its ability to identify true buy signals while minimizing false positives.

Across all metrics, the CNN model reliably outperformed the baseline LSTM, mainly excelling in recall (95%) and AUC-ROC (98%).

The CNN-GAF, especially in the last fold of the walk-forward validation, significantly shined, meaning it successfully captured nearly all true buy signals, making it a highly reliable model for trading decisions. However, the biggest improvement was in the AUC-ROC, where the model showed a near-perfect performance, indicating exceptional capability to distinguish between buy and sell signals.

These results not only highlight the model's ability in interpreting GAF images but also in precisely understanding the market dynamics inherent in the MACD Crossover strategy. The improvement in performance compared to the Stochastic Oscillator strategy can be attributed to the fact that MACD Crossover strategy usually produces fewer but more reliable buy and sell signals. This reduces noise in the data and enables the CNN to capture cleaner and more structured patterns. The MACD strategy's more consistent signals align with the CNN's strength in identifying spatial patterns, leading to better overall results.

In conclusion, the CNN-GAF excelled with the MACD Crossover strategy, identifying almost all buy and sell signals and consistently outperforming the baseline LSTM model. The metrics supporting these results for the MACD crossover strategy are presented in Table 6.5.

Table 6.5. CNN-GAF MACD Crossover strategy results.

	MACD Crossover Strategy		
Metrics	Train-Test Split	Walk Forward Last Fold	Walk Forward Overall
Accuracy	91%	92%	90%
Precision	89%	91%	91%
Recall	94%	95%	90%
AUC-ROC	98%	98%	96%

The final dataset tested with the CNN-GAF model was the stock price direction, which is inherently the most difficult due to the strategy's dependence on the price movement of the prior day. As anticipated, the stock price direction strategy performed the worst of the three assessed strategies but still outperformed the baseline LSTM model across all metrics. Particularly, the CNN-GAF model achieved a 57% accuracy and a 77% recall in the walk-forward validation, emphasizing its strength in identifying true buy signals and more trading chances, which is fundamental to maximizing profit opportunities.

The AUC-ROC was also moderately better when compared to the LSTM baseline, with the best value reaching 57%. While this suggests that the model has some difficulty distinguishing between buy and sell signals, this level of performance is appropriate in the context of the stock market, where price direction prediction is particularly difficult.

Across the validation approaches, the CNN-GAF model showed consistent performance, a marked contrast to the LSTM model, which presented more significant variability. This stability further supports the case for CNN-GAF as a more reliable and robust model for stock price prediction.

Overall, the CNN-GAF model's ability to capture market trends makes it a significantly better candidate for trading strategies than the LSTM baseline. Table 6.6 shows the results for the stock price direction strategy.

Table 6.6. CNN-GAF Stock Price Direction strategy results.

	Stock Price Direction Strategy		
Metrics	Train-Test Split	Walk Forward Last Fold	Walk Forward Overall
Accuracy	57%	57%	51%
Precision	57%	59%	53%
Recall	60%	77%	60%
AUC-ROC	57%	54%	51%

6.2.2. CNN-RP

Next, the CNN model was tested with Recurrence Plots across all strategies (datasets). The CNN-RP model consistently showed worse performance compared to the GAF images.

In the Stochastic Oscillator strategy, as detailed in Table 6.7, the CNN-RP model struggled to exceed 50% accuracy in each evaluation method. Although the model attained high recall in the Walk Forward Last Fold, this came at the cost of low precision, revealing that while the model identified many true positives, it also created many false positives. Despite numerous adjustments, the model tended to overfit the training data, which was apparent from the inadequate generalization. The AUC-ROC varied between 49%-54%, indicating the CNN-RP's ability to distinguish between buy and sell signals was only slightly better than random guessing.

A similar trend was observed in the Stock Price Direction dataset, as shown in Table 6.9, where the CNN-RP model performed worse than the CNN-GAF. The high recall observed in the Train-Test Split possibly resulted from overfitting, as it dropped sharply in Walk Forward Validation, confirming the model's difficulty in generalizing to test data. In this strategy, the CNN-GAF was demonstrated to be a more reliable model, offering better generalization and capturing more true buy signals, which is essential in a stock trading model.

In the case of MACD Crossover strategy, since it creates more precise signals and with less noise in data, as shown in Table 6.8, the CNN-RP presented a good performance but also was below the CNN-GAF only achieving an accuracy of 80% but with a very good AUC-ROC, indicating that the model was highly effective at distinguishing between buy and sell signals.

To conclude, the inferior performance of recurrence plots compared to GAF can be attributed to GAF’s strength in translating both the direction and magnitude of changes into a visual representation.

This capability is fundamental in volatile and complex environments like the stock market, where even slight price variations can cause significant changes. GAF's ability to identify trends over time and angular relationships aligns well with CNN architectures, allowing them to detect more meaningful and evident patterns.

On the other hand, recurrence plots excel at capturing periodicity and repeated patterns, which may not be as effective in the non-repetitive and volatile nature of stock market data. Sudden changes, outliers, and trends in the market frequently do not follow cyclical patterns, reducing the effectiveness of recurrence plots in this domain. While these characteristics are beneficial in other contexts, it is clear that for predicting buy and sell signals in the stock market, the CNN-GAF consistently outperformed the CNN-RP. As mentioned in the text, the results for each strategy are shown in Tables 6.7, 6.8, and 6.9.

Table 6.7. CNN-RP Stochastic Oscillator strategy results.

	Stochastic Oscillator Strategy		
Metrics	Train-Test Split	Walk Forward Last Fold	Walk Forward Overall
Accuracy	50%	48%	53%
Precision	43%	47%	49%
Recall	45%	97%	59%
AUC-ROC	54%	49%	54%

Table 6.8. CNN-RP MACD Crossover strategy results.

	MACD Crossover Strategy		
Metrics	Train-Test Split	Walk Forward Last Fold	Walk Forward Overall
Accuracy	82%	82%	80%
Precision	80%	82%	85%
Recall	87%	84%	78%
AUC-ROC	87%	92%	89%

Table 6.9. CNN-RP Stock Price Direction strategy results.

	Stock Price Direction Strategy		
Metrics	Train-Test Split	Walk Forward Last Fold	Walk Forward Overall
Accuracy	51%	53%	53%
Precision	51%	52%	54%
Recall	99%	45%	65%
AUC-ROC	55%	52%	52%

6.2.3. CNN-MTF

Lastly, the CNN model was tested with MTF images across all strategies, and once again, the CNN-GAF was demonstrated to be the most reliable model for stock price prediction due to its ability to identify both short-term fluctuations and long-term trends. The drop in performance of the CNN-MTF model can be attributed to the fact that MTF captures transitions between states, which is not an ideal representation for stock price movements. Stock prices are usually less about distinct "state transitions" and more about continuous trends or momentum shifts. As a result, MTF fails to capture key details that GAF images effectively retain.

In the Stochastic Oscillator strategy, the CNN-MTF model struggled considerably to capture the short-term fluctuations and long-term trends of the AMD stock. This resulted in low AUC-ROC scores across both evaluation methods, varying from 47%-55%, which indicates the model was only marginally better than random guessing when distinguishing between buy and sell signals.

For the MACD Crossover strategy, there was also a noticeable drop in performance for the CNN-MTF, especially when compared to the CNN-RP model. This decline is likely appointed to the recurrence plot's ability to capture recurrence, which is more effective in detecting the patterns involved in the MACD Crossover strategy. The CNN-RP achieved 80% accuracy, while the CNN-MTF only reached the mid-60% range.

Finally, in the stock price direction strategy, the CNN-MTF model faced the same challenges as in the other strategies, being outperformed by both the CNN-GAF and CNN-RP models. Accuracy for CNN-MTF hovered between 47% and 51%, which is close to random guessing and shows a lack of generalization.

Reiterating, the MTF performed worse primarily because of its inherent design, which focuses on capturing transitions between discrete states. This approach works well when transitions are clearly defined, but such clarity is sporadic in the volatile and complex nature of the stock market. In financial markets, prices tend to move in continuous trends, and the MTF's emphasis on state transitions causes the model to miss crucial details about the direction, momentum, and magnitude of price movements. This leads to the model's inability to effectively distinguish between potential buy and sell signals. On the other hand, GAF not only captures angular relationships but also preserves the magnitude of changes, making it much better suited for stock price prediction. The results supporting these findings for each strategy can be seen in Tables 6.10, 6.11, and 6.12.

Table 6.10. CNN-MTF Stochastic Oscillator strategy results.

	Stochastic Oscillator Strategy		
Metrics	Train-Test Split	Walk Forward Last Fold	Walk Forward Overall
Accuracy	57%	55%	49%
Precision	0%	51%	48%
Recall	0%	52%	75%
AUC-ROC	47%	55%	51%

Table 6.11. CNN-MTF MACD Crossover strategy results.

	MACD Crossover Strategy		
Metrics	Train-Test Split	Walk Forward Last Fold	Walk Forward Overall
Accuracy	62%	65%	63%
Precision	63%	62%	63%
Recall	64%	91%	76%
AUC-ROC	66%	76%	69%

Table 6.12. CNN-MTF Stock Price Direction strategy results.

	Stock Price Direction Strategy		
Metrics	Train-Test Split	Walk Forward Last Fold	Walk Forward Overall
Accuracy	45%	49%	51%
Precision	45%	49%	52%
Recall	44%	100%	63%
AUC-ROC	47%	54%	54%

6.3. Synthetic Data

The last step in this Dissertation was to generate synthetic data to enhance the predictive power of the CNN model. As mentioned before, the YData Fabric platform was utilized to generate the synthetic dataset. For this step in the study, only the stock direction dataset was used since it was the strategy that would need to be improved.

Regarding the sample, only data from the year 2023 was considered to generate additional synthetic data that simulated the bull run in the information technology sector during that year.

After configuring the parameters for synthetic data generation and completing the data creation process, the fidelity of the generated data was assessed using two key metrics: correlation similarity and distance distribution. Furthermore, the Qscore was used to measure the utility of the synthetic data.

The first metric, the correlation similarity, measures how similar the correlation matrices of the synthetic data are to those of the original dataset. The score varies between 0 and 1, and the higher the value, the higher the fidelity. In this case, the synthetic data achieved a perfect score of 1.0, which means that the relationships between features in the synthetic data mirror those in the original dataset.

The Distance distribution measures the similarity between the feature distributions between both datasets. A value close to 1 indicates that the synthetic data follows the same distribution patterns as the actual data. Once again, the synthetic data achieved a near perfect score, indicating that the feature distributions closely resemble those in the original dataset.

Lastly, the QScore evaluates the utility of the synthetic data by comparing the results of random aggregation queries performed on both the synthetic and original datasets. A score above 0.8 indicates that future queries on the synthetic data will maintain the same statistical characteristics as those conducted on the original dataset, ensuring high data fidelity. Table 6.13 presents the metrics for synthetic data generated.

Table 6.13. Synthetic data profiling.

Synthetic data profiling

Metrics	*Train-Test Split*
Correlation Similarity	1.00
Distance Distribution	0.99
QScore	0.83

After evaluating the various metrics, a Principal Component Analysis (PCA) algorithm was applied to reduce the dimensionality of both datasets. The first two principal eigenvectors together accounted for approximately 71% of the total variance in the data. In the dimensionality plot below (Figure 6.1), it is evident how closely the distribution of the synthetic dataset mirrors that of the original dataset.

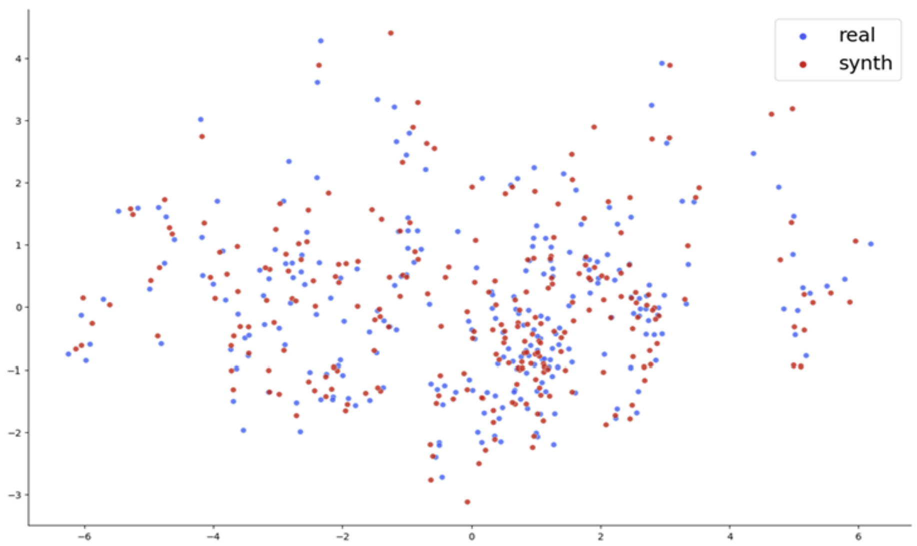

Figure 6.1. Dimensionality plot comparing real data vs. synthetic data.

The final metric evaluated using YData Fabric was the Train Synthetic Test Real, which accesses the AUC-ROC score across different estimators. The average AUC-ROC score was 67%, ranging from

59% to 73%. This exhibits a clear improvement over the AUC-ROC results achieved in both the CNN and LSTM models using only real data, emphasizing the significant potential of synthetic data in improving machine learning model performance and robustness.

All performance measures evaluated point to the leading value of synthetic data generation in enhancing machine learning model accuracy. By generating high-fidelity synthetic datasets that capture the distribution and relationships of the actual data, synthetic data allows models to be trained on more precise and additional signals. This is particularly valuable in volatile and complex environments such as the stock market, where data may be noisy or sparse. Synthetic data can be designed to simulate specific market scenarios, ensuring that models are trained on a more diverse dataset and are more representative of real-world market dynamics.

Chapter 7

Conclusion

This Dissertation aimed to study the use of Convolutional Neural Networks for stock market price prediction, integrating technical indicators, sentiment analysis, image-based time-series transformations, and synthetic data generation.

Several achievements were made during the course of this Dissertation. As the primary objective, a CNN model was developed for price movement prediction using various image generation techniques (GAF, RP, MTF) to convert time-series data into visual representations. As a secondary objective, a broad set of technical indicators with diverse time frames and sentiment variables derived from financial news using the FinBERT library were created. This provided valuable insights into how sentiment could be leveraged as a predictive variable, although its effective integration proved to be challenging.

Another key goal was to develop and analyse trading strategies capable of withstanding short-term fluctuations under different market conditions. Among the strategies tested, the MACD Crossover and the Stochastic Oscillator proved robust across backtesting scenarios. Additionally, TimeGAN, implemented using the YData platform, was used to generate synthetic stock market data that mimicked real market behaviour.

The findings of this study underscore the importance of a thorough feature selection process. The extensive analysis significantly improved data input quality and predictive performance. This allowed for a deeper understanding of various selection techniques and confirmed that combining multiple methods can enhance variable selection, as suggested by Tsai and Hsiao (2010), ultimately improving stock prediction by identifying the most relevant financial indicators.

Among the models tested, the CNN-GAF combination proved to be the most effective. GAF's ability to capture angular trends and temporal relationships aligned well with CNN's strengths and the dynamic nature of stock market data. The use of synthetic data was also validated. In line with the findings of Lin et al. (2021) and Yoon and Jarrett (2019), TimeGAN was able to generate high-fidelity data, reinforcing the potential of synthetic data as a tool for simulating diverse market scenarios and improving model training.

Nonetheless, this Dissertation presents several limitations. Only two trading strategies were implemented, which may have limited the ability to capture more complex market behaviour or achieve stronger predictive signals. The analysis focused primarily on technical indicators, while fundamental indicators were excluded due to their complexity and limited availability. Although sentiment analysis was initially considered, it was excluded from the final model during feature selection, reducing the opportunity to evaluate its true contribution. The model's performance was also limited by the specific market conditions under which it was tested, namely a bull market, raising concerns about its ability to generalize across different time periods and market scenarios without ongoing recalibration.

Future research should aim to explore a wider and more dynamic set of trading strategies, develop hybrid models that combine CNN-GAF architectures with sentiment analysis, and validate model performance across different markets and economic environments. Additional studies should consider ensemble learning approaches or attention-based models, which may better adapt to the complexity of financial data. Evaluating these models under varied market conditions, using both real and synthetic data, will be essential to assess their robustness and to explore the full potential of synthetic data in stock price movement prediction.

In summary, this Dissertation establishes a strong foundation for understanding the potential of CNNs in financial trading strategies. When combined with appropriate image encoding methods such as GAF, CNNs show a strong capacity to identify patterns and relationships within financial time series. Moreover, synthetic data offers great value to stock market prediction, providing a way to simulate specific market scenarios and reduce dependence on historical datasets.

References

Barak, S., Arjmand, A., & Ortobelli, S. (2017). Fusion of multiple diverse predictors in stock market. Information Fusion, 36, 90–102. https://doi.org/10.1016/j.inffus.2016.11.006.

B. Xue, M. Zhang, W. N. Browne and X. Yao, "A Survey on Evolutionary Computation Approaches to Feature Selection," in IEEE Transactions on Evolutionary Computation, vol. 20, no. 4, pp. 606-626, Aug. 2016, doi: https://doi.org/10.1109/TEVC.2015.2504420.

Cao, J., & Wang, J. (2019). Stock price forecasting model based on modified convolution neural network and financial time series analysis. International Journal of Communication Systems, https://doi.org/10.1002/dac.3987

Chandar, S.K. Convolutional neural network for stock trading using technical indicators. Autom Softw Eng 29, 16 (2022). https://doi.org/10.1007/s10515-021-00303-z

Chang, PC., Wu, JL. A critical feature extraction by kernel PCA in stock trading model. Soft Comput 19, 1393–1408 (2015). https://doi.org/10.1007/s00500-014-1350-5

Chen, Y., Lin, W., & Wang, J. Z. (2019). A dual-attention-based stock price trend prediction model with dual features. IEEE Access, 7, 148047–148058. https://par.nsf.gov/servlets/purl/10149715.

Enke, D., Grauer, M., & Mehdiyev, N. (2011). Stock market prediction with multiple regression, fuzzy type-2 clustering and neural networks. Procedia Computer Science, 6, 201–206. http://dx.doi.org/10.1016/j.procs.2011.08.038.

Ken-Ichi Funahashi, On the approximate realization of continuous mappings by neural networks, Neural Networks, Volume 2, Issue 3, 1989, Pages 183-192, ISSN 0893-6080, https://doi.org/10.1016/0893-6080(89)90003-8.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. "Generative Adversarial Nets." In Advances in Neural Information Processing Systems 27, edited by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, 2672–2680. Curran Associates, Inc. http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

Goswami, B. (2019). A brief introduction to nonlinear time series analysis and recurrence plots. Potsdam Institute for Climate Impact Research, Transdisciplinary Concepts & Methods. https://www.mdpi.com/1099-4300/21/12/1218

Gunduz, H., Yaslan, Y., & Cataltepe, Z. (2017). Intraday prediction of Borsa Istanbul using convolutional neural networks and feature correlations. Knowledge-Base Systems, 137, 138–148. http://dx.doi.org/10.1016/j.knosys.2017.09.023

Guyon, I., & Elisseeff, A. (2003). An Introduction to Variable and Feature Selection. https://www.jmlr.org/papers/volume3/guyon03a/guyon03a.pdf.

Henrique, B. M., Sobreiro, V. A., & Kimura, H. (2019). Literature review: Machine learning techniques applied to financial market prediction. Expert Systems With Applications, 124, 226–251. https://doi.org/10.1016/j.eswa.2019.01.012

Hoseinzade, E., & Haratizadeh, S. (2019). CNNPred: CNN-based stock market prediction using a diverse set of variables. Expert Systems With Applications, 129, 273–285. http://dx.doi.org/10.1016/j.eswa.2019.03.029.

Hu, H., Tang, L., Zhang, S., & Wang, H. (2018). Predicting the direction of stock markets using optimized neural networks with Google Trends. Neurocomputing, 285, 188–195. http://dx.doi.org/10.1016/j.neucom.2018.01.038.

Jansen, S. (2020). Chapter 18: CNNs for Financial Time Series and Satellite Images. In Machine Learning for Algorithmic Trading: Predictive models to extract signals from market and Alternative Data for systematic trading strategies with Python, 2nd edition. essay, Packt.

Jiang, W. (2021). Applications of deep learning in stock market prediction: Recent progress. Expert Systems With Applications, 184, Article 115537. http://dx.doi.org/10.1016/j.eswa.2021.115537.

Kitchenham, B. (2004). Procedures for performing systematic reviews. Keele University, UK and National ICT Australia.

Kubat, M. (2017). An introduction to machine learning (2nd Ed.). Springer Publishing Company, Incorporated.

Kumar, D., Sarangi, P. K., & Verma, R. (2021). A systematic review of stock market prediction using machine learning and statistical techniques. Materials Today: Proceedings, http://dx.doi.org/10.1016/j.matpr.2020.11.399.

Lin, H., Chen, C., Huang, G. & Jafari, A. (2021). Stock price prediction using Generative Adversarial Networks. Journal of Computer Science, 17(3), 188-196. https://doi.org/10.3844/jcssp.2021.188.196

Liu, Lu.; Wang, Zhiguang (2018) Encoding temporal Markov dynamics in graph for visualizing and mining time series, https://arxiv.org/pdf/1610.07273.

Lu, W., Li, J., Wang, J. et al. A CNN-BiLSTM-AM method for stock price prediction. Neural Comput & Applic 33, 4741–4753 (2021). https://doi.org/10.1007/s00521-020-05532-z.

M. Dash, H. Liu, Feature selection for classification, Intelligent Data Analysis, Volume 1, Issues 1–4, 1997, Pages 131-156, ISSN 1088-467X, https://doi.org/10.1016/S1088-467X(97)00008-5.

Mailagaha Kumbure, Mahinda & Lohrmann, Christoph & Luukka, Pasi & Porras, Jari. (2022). Machine learning techniques and data for stock market forecasting: A literature review. Expert Systems with Applications. 197. 116659. https://doi.org/10.1016/j.eswa.2022.116659.

Mehtab, Sidra; Sen, Jaydip (2021): Stock Price Prediction Using Convolutional Neural Networks on a Multivariate Time Series. TechRxiv. Preprint. https://doi.org/10.36227/techrxiv.15088734.v1

Qiu, M., & Song, Y. (2016). Predicting the direction of stock market index movement using an optimized artificial neural network model. PLoS ONE, 11(5), 1–11. http://dx.doi.org/10.1371/journal.pone.0155133.

Sedighi, Mojtaba & Jahangirnia, Hossein & Gharakhani, Mohsen & Fard, Saeed. (2019). A Novel Hybrid Model for Stock Price Forecasting Based on Metaheuristics and Support Vector Machine. Data. 4. 75. https://www.researchgate.net/publication/333316620_A_Novel_Hybrid_Model_for_Stock_Price_Forecasting_Based_on_Metaheuristics_and_Support_Vector_Machine.

Sezer, Omer & Ozbayoglu, Murat. (2018). Algorithmic Financial Trading with Deep Convolutional Neural Networks: Time Series to Image Conversion Approach. Applied Soft Computing. 70. 10.1016/j.asoc.2018.04.024.

Shi, L., Teng, Z., Wang, L., Zhang, Y., & Binder, A. (2019). DeepClue: VIsual interpretation of text-based deep stock prediction. IEEE Transactions On Knowledge And Data Engineering, 31, 1094–1108. http://dx.doi.org/10.1109/TKDE.2018.2854193.

Staffini, Alessio. (2022). Stock Price Forecasting by a Deep Convolutional Generative Adversarial Network. Frontiers in Artificial Intelligence. 5:837596. https://doi.org/10.3389/frai.2022.837596.

Tareq Abed Mohammed, Shaymaa Alhayali, Oguz Bayat, Osman N. Uçan, "Feature Reduction Based on Hybrid Efficient Weighted Gene Genetic Algorithms with Artificial Neural Network for Machine Learning Problems in the Big Data", Scientific Programming, vol. 2018, Article ID 2691759, 10 pages, 2018. https://doi.org/10.1155/2018/2691759

Thawornwong, Suraphan & Enke, David. (2004). The adaptive selection of financial and economic variables for use with artificial neural networks. Neurocomputing. 56. 205-232. 10.1016/j.neucom.2003.05.001.

Too J, Abdullah AR, Mohd Saad N. A New Co-Evolution Binary Particle Swarm Optimization with Multiple Inertia Weight Strategy for Feature Selection. Informatics. 2019; 6(2):21. https://doi.org/10.3390/informatics6020021.

Tsai, C. F., & Hsiao, Y. C. (2010). Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches. Decision Support Systems, 50(1), 258–269. http://dx.doi.org/10.1016/j.dss.2010.08.028.

Wang, W., Zhang, M., Wang, D. et al. Kernel PCA feature extraction and the SVM classification algorithm for multiple-status, through-wall, human being detection. J Wireless Com Network 2017, 151 (2017). https://doi.org/10.1186/s13638-017-0931-2

Wang, Z., & Oates, T. (2015). Imaging Time-Series to Improve Classification and Imputation. arXiv preprint arXiv:1506.00327. Retrieved from https://arxiv.org/abs/1506.00327

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep Learning." Nature 521 (7553): 436–44. https://doi.org/10.1038/nature14539.

Xu, Z., Wang, Y., Feng, X., Wang, Y., Li, Y., & Lin, H. (2023). Quantum-Enhanced Forecasting: Leveraging Quantum Gramian Angular Field and CNNs for Stock Return Predictions. https://doi.org/10.48550/arXiv.2310.07427

Yaohao Peng, Pedro Henrique Melo Albuquerque, Herbert Kimura, Cayan Atreio Portela Bárcena Saavedra, Feature selection and deep neural networks for stock price direction forecasting using technical analysis indicators, Machine Learning with Applications, Volume 5, 2021, 100060, ISSN 2666-8270, https://doi.org/10.1016/j.mlwa.2021.100060.

YData. (n.d.). YData – The Data-Centric AI Company. https://ydata.ai

Yeh, C. Y., Huang, C. W., & Lee, S. J. (2011). A multiple-kernel support vector regression approach for stock market price forecasting. Expert Systems With Applications, 38(3), 2177–2186. http://dx.doi.org/10.1016/j.eswa.2010.08.004.

Yoon, Jinsung, Daniel Jarrett, and Mihaela van der Schaar. 2019. "Time-Series Generative Adversarial Networks." In Advances in Neural Information Processing Systems 32, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, 5508–5518. Curran Associates, Inc. http://papers.nips.cc/paper/8789-time-series-generative-adversarial-networks.pdf.

Zhiguang Wang and Tim Oates. (2015) Imaging time-series to improve classification and imputation. arXiv preprintarXiv:1506.00327.

Zhong, X., & Enke, D. (2019). Predicting the daily return direction of the stock market using hybrid machine learning algorithms. Financial Innovation, 5(1), http://dx.doi.org/10.1186/s40854-019-0138-0.

Appendix A

Figure of S&P 500 Sector Returns

Figure A.1. S&P 500 five-year sector returns.

Appendix B

Figures of Technical Indicators Performance on the AMD Stock

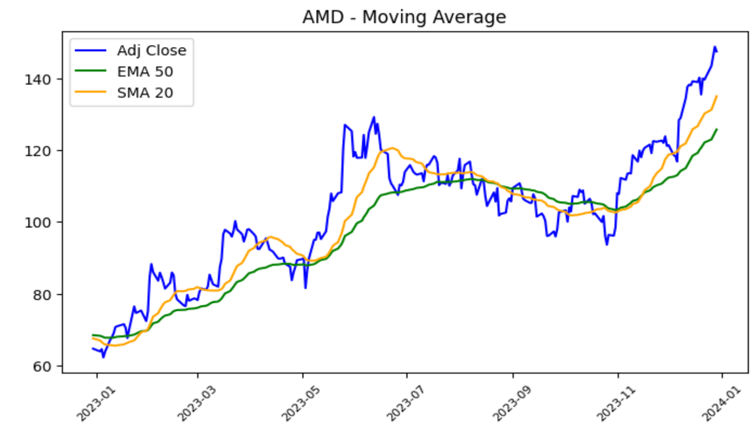

Figure B.1. Moving Average Plot.

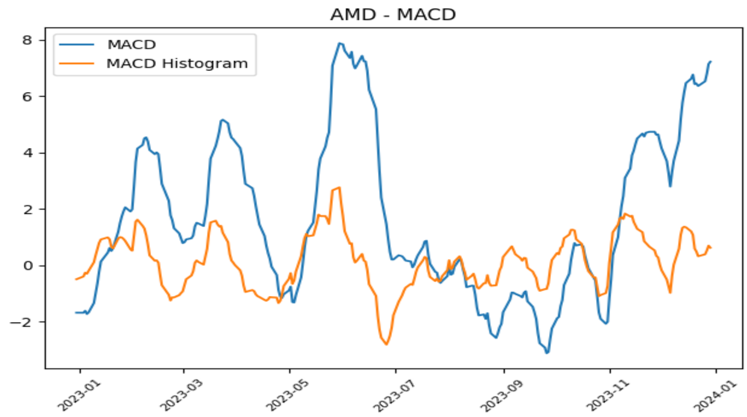

Figure B.2. MACD Plot.

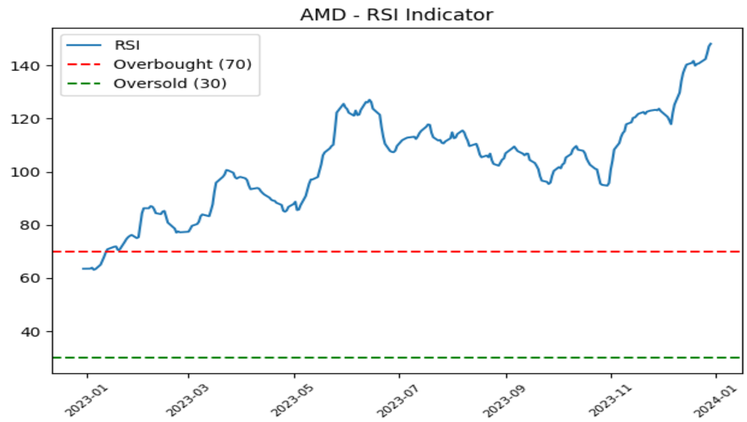

Figure B.3. RSI Plot.

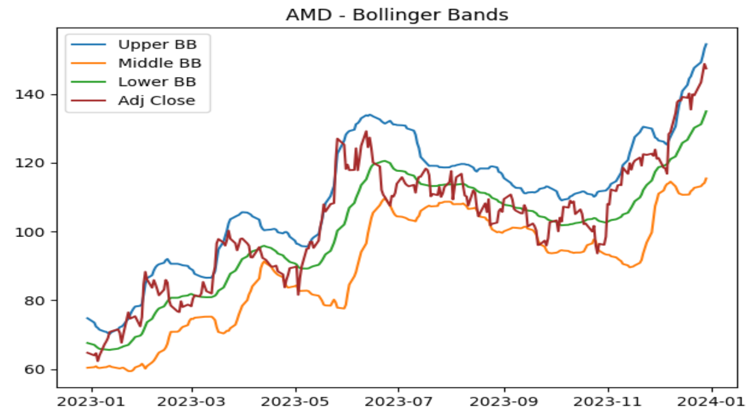

Figure B.4. Bollinger Bands Plot.

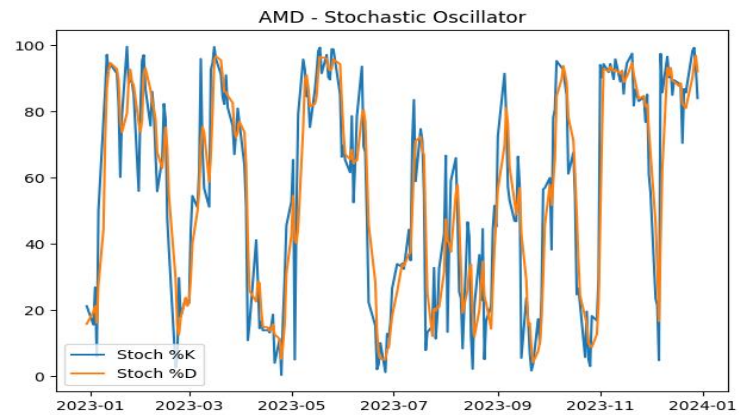

Figure B.5. Stochastic Oscillator Plot.

Appendix C

Figures of Results from Feature Selection Methods Applied

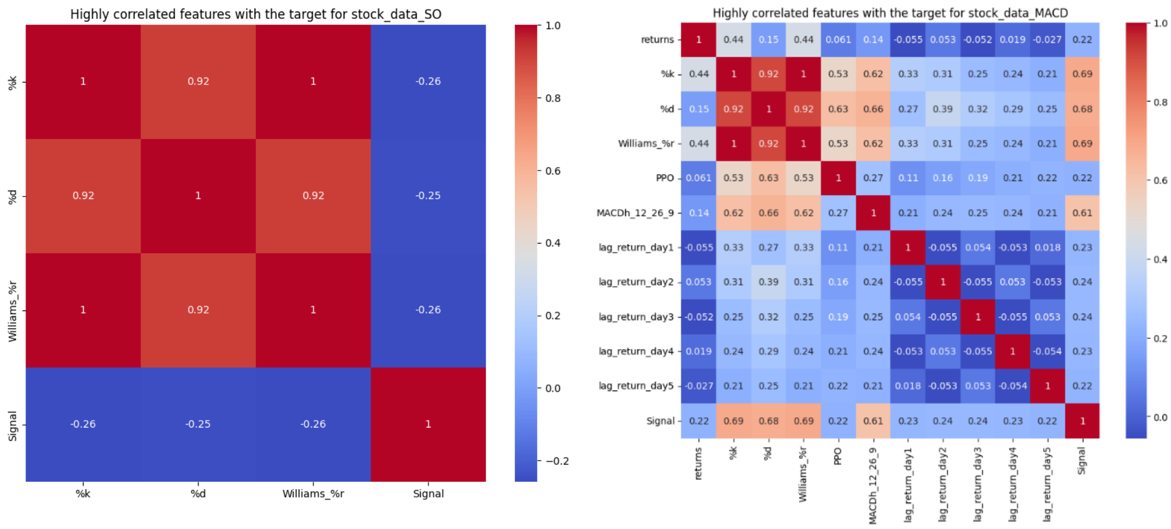

Figure C.1. Highly correlated features with the Stochastic Oscillator and MACD Crossover strategies.

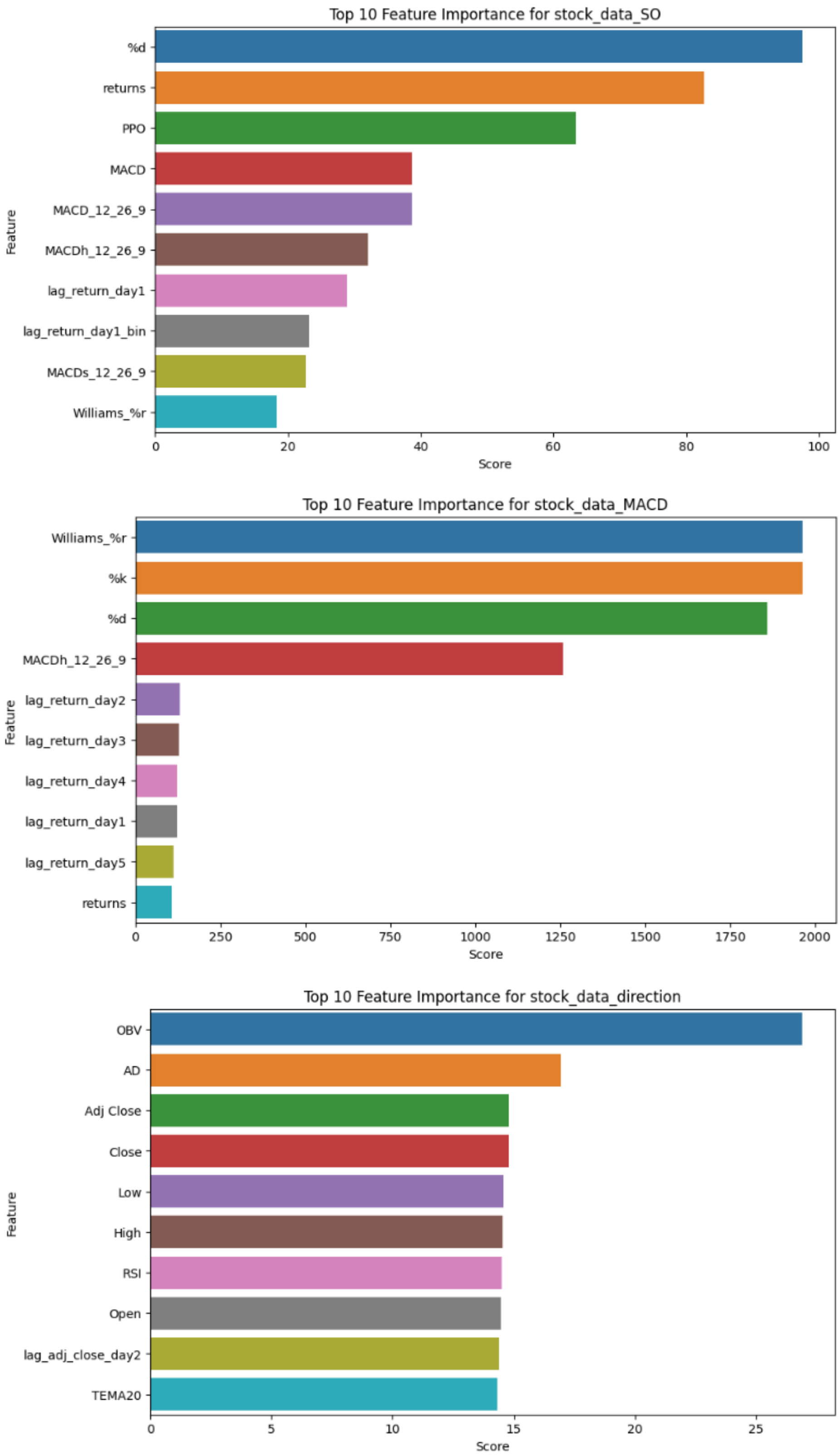

Figure C.2. Top 10 features selected using Univariate Feature Selection for the Stochastic Oscillator, MACD Crossover, and Price Direction strategies.

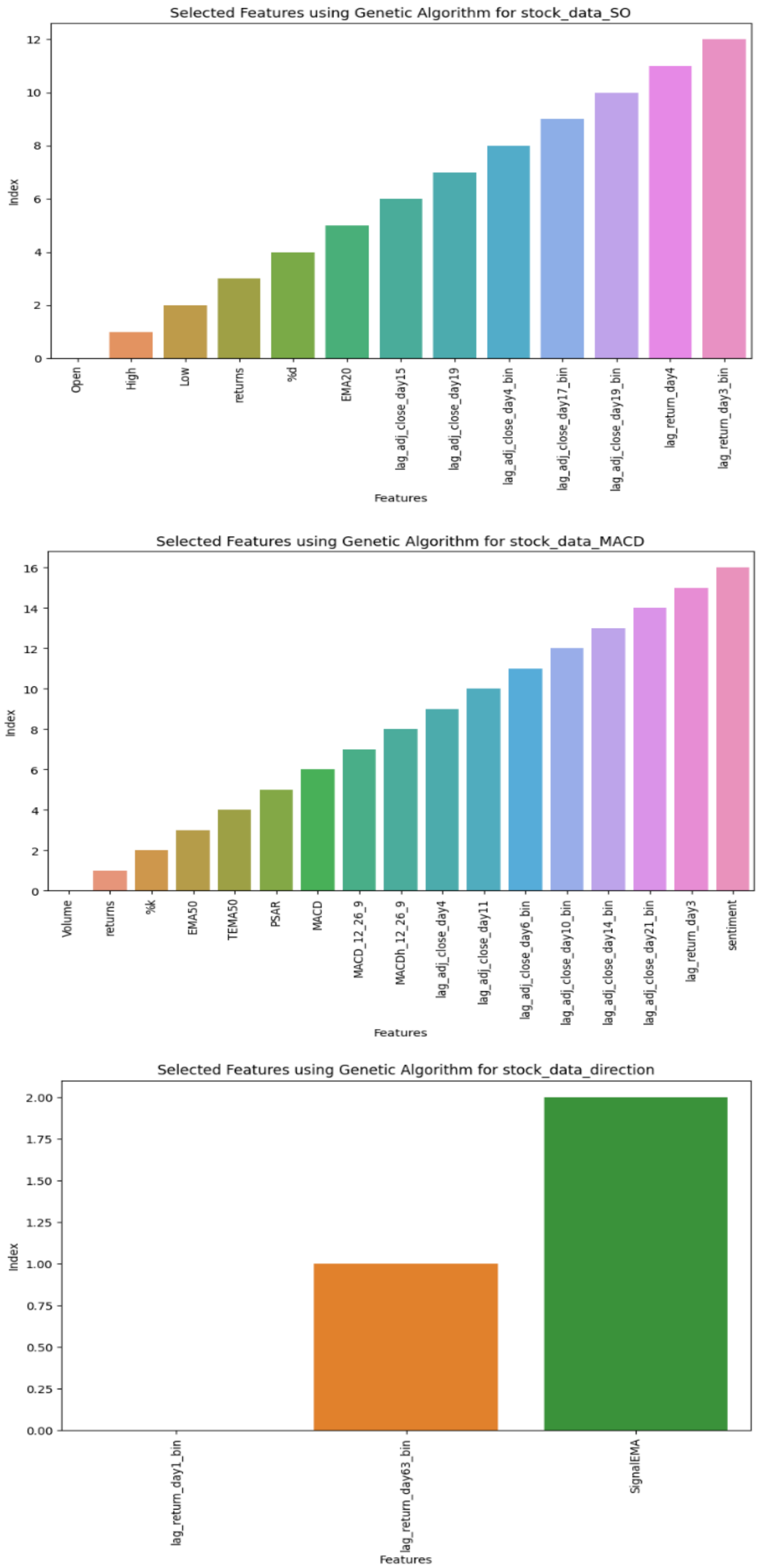

Figure C.3. Features selected using GA for the Stochastic Oscillator, MACD Crossover, and Price Direction strategies.

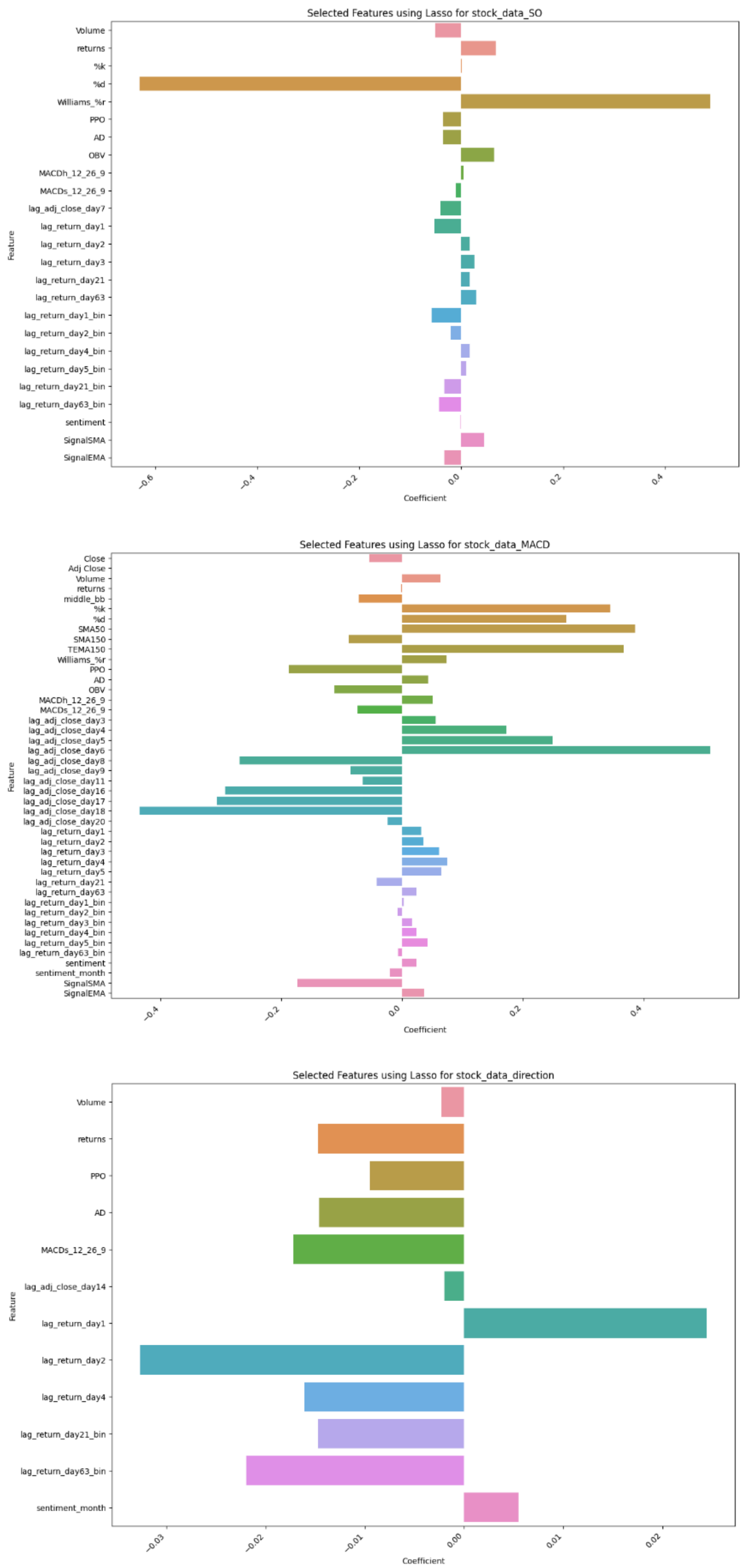

Figure C.4. Features selected using Lasso Regression for the Stochastic Oscillator, MACD Crossover, and Price Direction strategies.

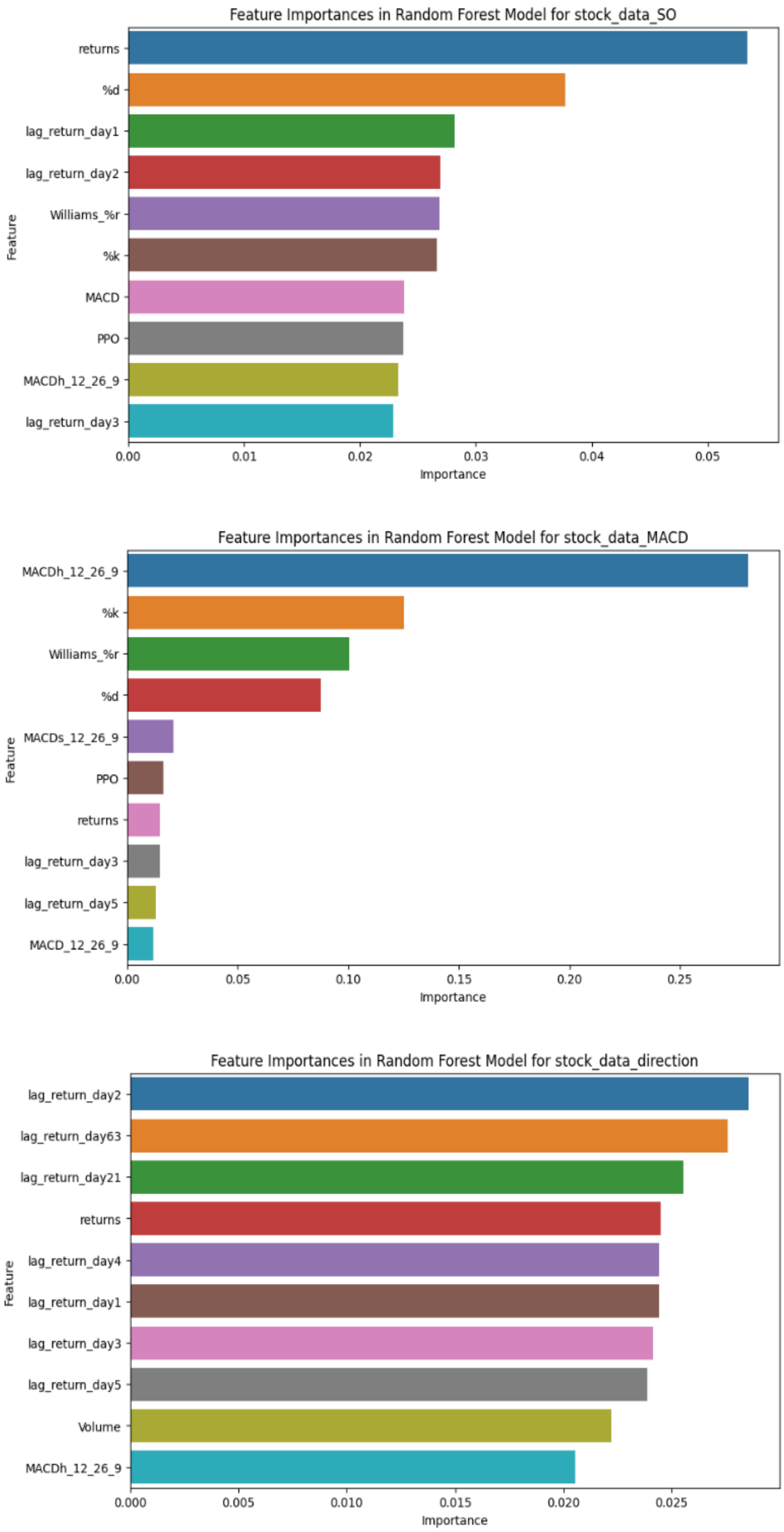

Figure C.5. Top 10 features selected using Random Forest for the Stochastic Oscillator, MACD Crossover, and Price Direction strategies.

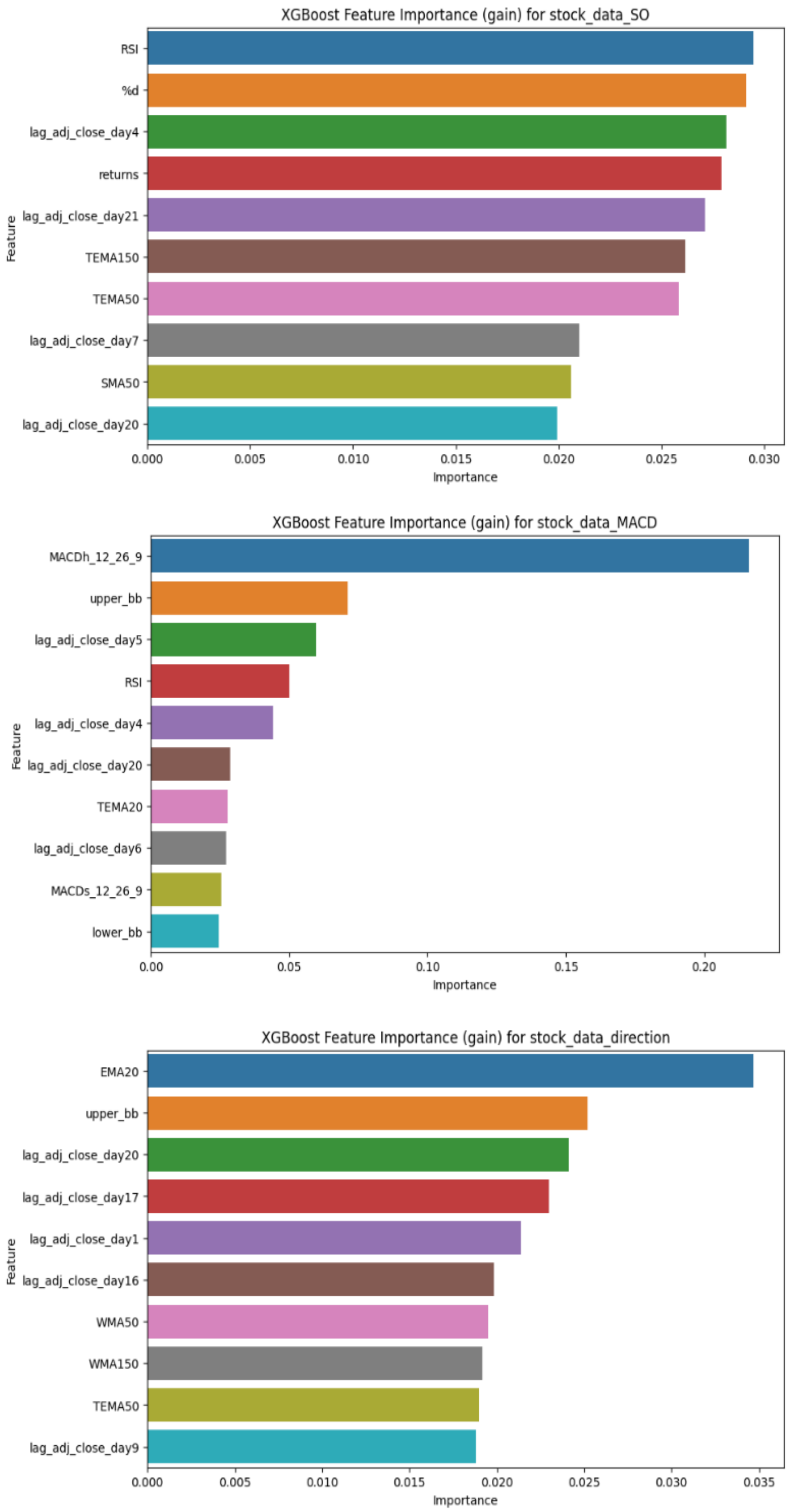

Figure C.6. Top 10 features selected using XGBoost for the Stochastic Oscillator, MACD Crossover, and Price Direction strategies.

Appendix D

Figures of Results from the various models

D.1. Confusion Matrices for the LSTM Baseline Model

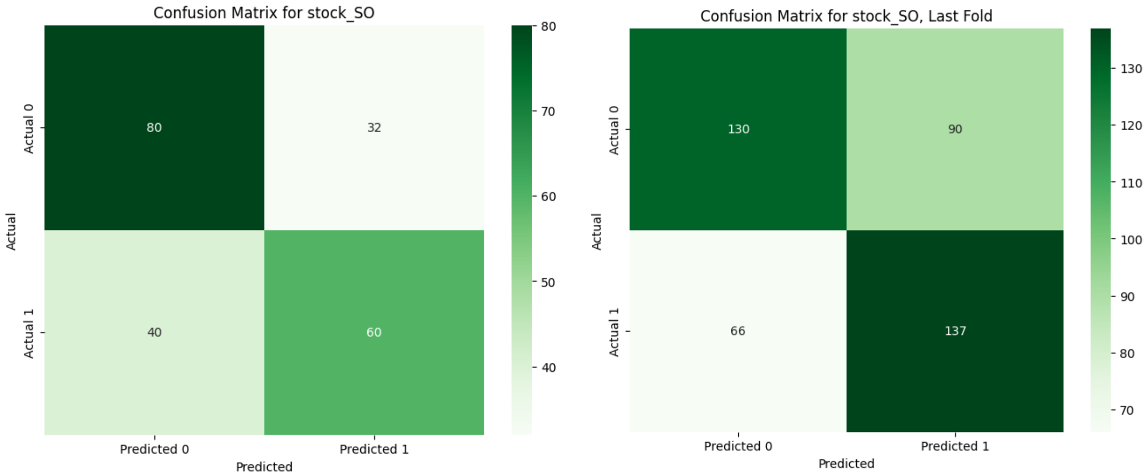

Figure D.1. Confusion matrix for the train-test split and the final fold of the rolling CV for the Stochastic Oscillator strategy.

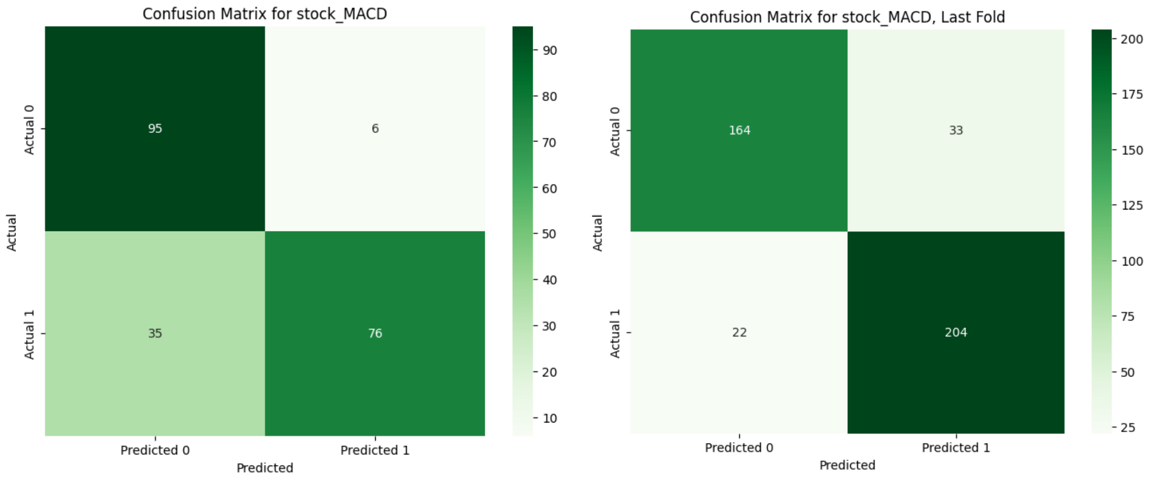

Figure D.2. Confusion matrix for the train-test split and the final fold of the rolling CV for the MACD Crossover strategy.

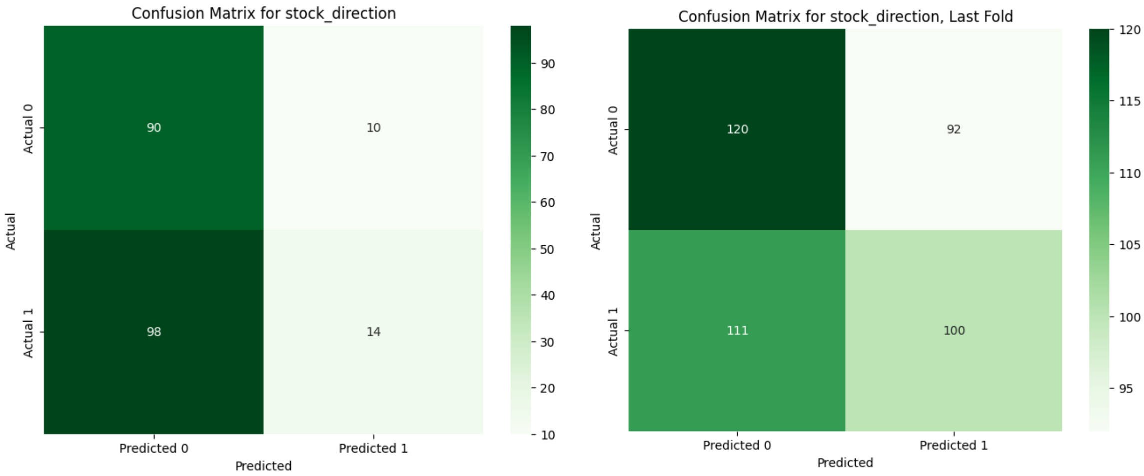

Figure D.3. Confusion matrix for the train-test split and the final fold of the rolling CV for the Price Direction strategy.

D.2. Confusion Matrices for the CNN-GAF model

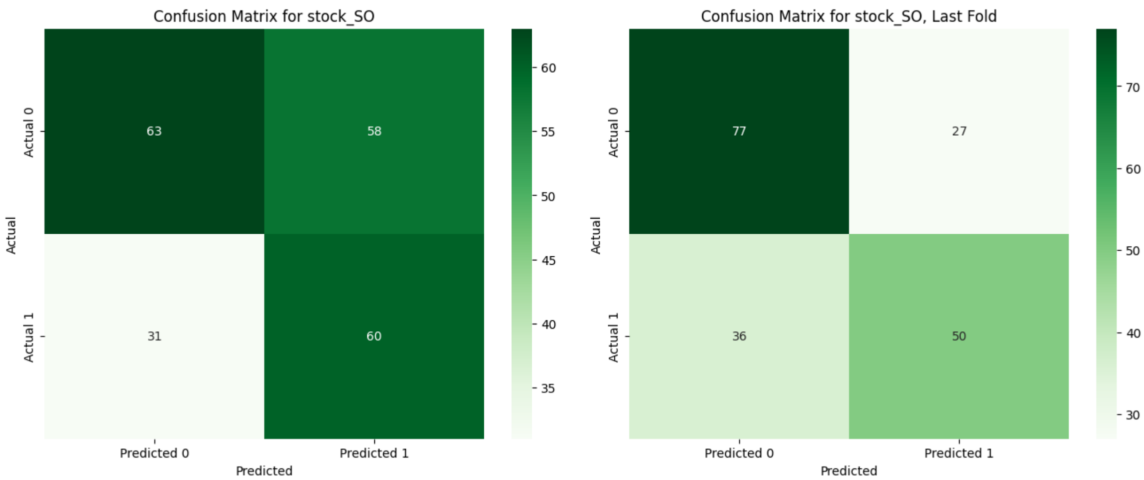

Figure D.4. Confusion matrix for the train-test split and the final fold of the rolling CV for the Stochastic Oscillator strategy.

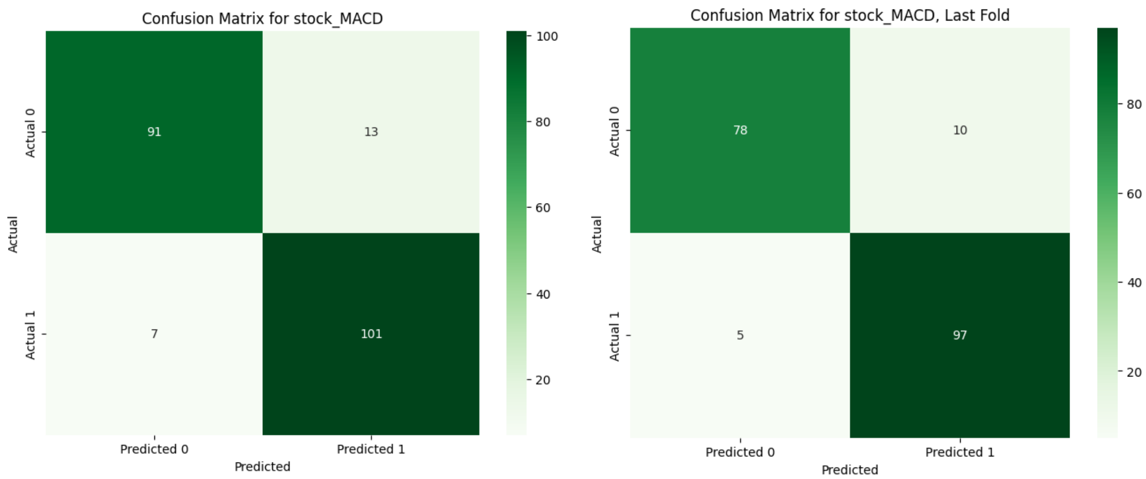

Figure D.5. Confusion matrix for the train-test split and the final fold of the rolling CV for the MACD Crossover strategy.

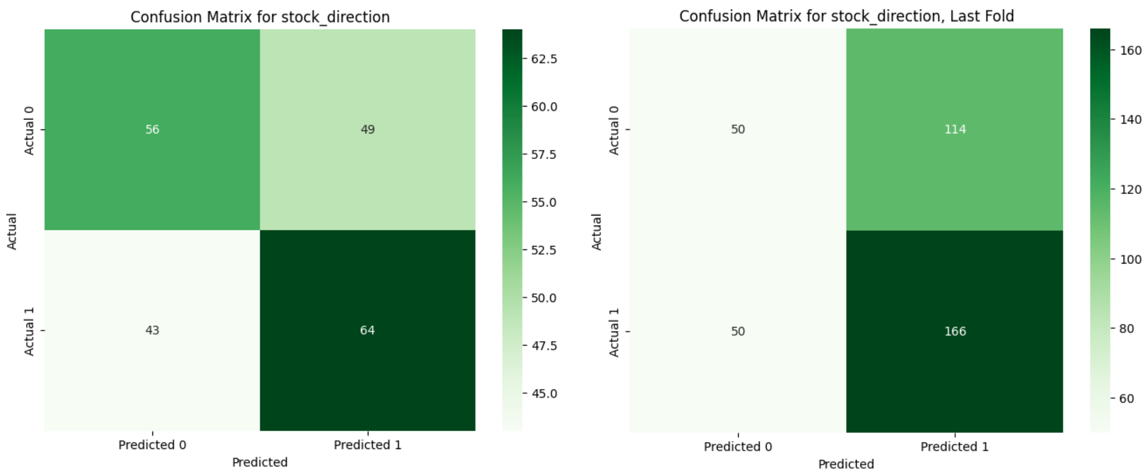

Figure D.6. Confusion matrix for the train-test split and the final fold of the rolling CV for the Price Direction strategy.