Fundamental, Sentiment and Technical Analysis for Algorithmic Trading Using Novel Genetic Programming Algorithms

Evangelia Paraskevi Christodoulaki

A thesis submitted for the degree of

Doctor of Philosophy

at the School of Computer Science and Electronic Engineering

University of Essex

Date of submission for examination January 2024

Abstract

This thesis explores genetic programming (GP) applications in algorithmic trading, addressing significant advancements in the field. Investors typically rely on fundamental analysis (FA) or technical analysis (TA) indicators, with sentiment analysis (SA) gaining recent attention. Consequently, algorithms have become the primary method for developing pre-programmed trading strategies, leading to substantial financial benefits. While each analysis type has been studied individually, their combined exploration remains limited. Our motivation is to assess if integrating FA, SA, and TA indicators can improve financial profitability. Therefore, we propose the use of novel GP algorithms for the combination of the three analysis types, along with the use of a novel fitness function, and a novel GP operator that encourages active trading by injecting trees into the GP population that perform a high number of trades while achieving high profitability at low risk. To evaluate our GP variants' performance, we conduct experiments on stocks of 42 international companies, comparing the novel algorithm with the GP variants introduced in the same chapter. Moreover, we compare the proposed GP algorithm against four machine learning benchmarks and a financial trading strategy. The evaluation employs three financial metrics: Sharpe ratio, rate of return, and risk. Results consistently show that the proposed GP algorithms in each chapter enhance the financial performance of trading strategies, surpassing the benchmarks.

Acknowledgements

I would like to express my sincere appreciation to Dr. Michael Kampouridis, my supervisor, for his invaluable guidance, support, and insights throughout my journey. I am truly grateful to have had such an exceptional mentor and I could not have asked for a better one. Additionally, I would like to extend my thanks to Dr. Maria Kyropoulou and Dr. Panagiotis Kanelopoulos for their collaboration on research papers, which has been extremely enriching. I would also like to acknowledge the friends and lecturers who have shared this PhD adventure with me, whose advice and discussions have made the experience even more rewarding. To my family, I am deeply grateful for your support. Finally, I want to express my heartfelt gratitude to my partner for their love and encouragement. Thank you for believing in me! To everyone, I am truly grateful you have been part of this journey.

List of Publications

- Christodoulaki, E., Kampouridis, M., " Fundamental, Technical and Sentiment Analysis for Algorithmic Trading with Genetic Programming.”, IEEE Congress on Evolutionary Computation (CEC) (2023)
- Christodoulaki, E., Kampouridis, M., Kyropoulou M., "Enhanced Genetic Programming for Algorithmic Trading.", Genetic and Evolutionary Computation Conference (GECCO) (2023)
- Christodoulaki, E., Kampouridis, M., "Using strongly typed genetic programming to combine technical and sentiment analysis for algorithmic trading.", IEEE Congress on Evolutionary Computation (CEC) (2022)
- Christodoulaki, E., Kampouridis, M., "Combining Technical and Sentiment Analysis under a Genetic Programming algorithm.”, UK Workshop on Computational Intelligence (UKCI) (2022)
- Christodoulaki, E., Kampouridis, M., Kanellopoulos, P., "Technical and Sentiment Analysis in Financial Forecasting with Genetic Programming.", IEEE Symposium on Computational Intelligence for Financial Engineering & Economics (CIFEr) (2022)

List of Works Under Review

- Christodoulaki, E., Kampouridis, M., Kyropoulou, M., "Different data types Strongly Typed Genetic Programming for Algorithmic Trading", Swarm and Evolutionary Computation

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Algorithmic trading refers to the use of computer programs and algorithms to execute trades in financial markets. It involves creating and implementing a set of predefined rules and strategies that guide the buying and selling of financial instruments such as stocks, bonds, commodities, currencies, and derivatives. These algorithms are designed to analyse market data, identify patterns, make decisions, and execute trades at high speeds and frequencies that would be nearly impossible for a human trader to achieve.

1.1 Motivation

In the field of algorithmic trading research and implementation, it is common to apply the indicators derived from on economic measures (fundamental analysis), events, such as news (sentiment analysis), and historical prices (technical analysis) in order to infer future prices, individually. Such applications have yielded significant results, and they play a pivotal role in the accumulation of profits in the stock market. Integrating the individual fundamental, sentiment, and technical analysis types in trading strategies has been limited due to their distinct approaches and philosophies in understanding and analysing market behaviour. However, recent studies have suggested that combining these analysis types may lead to improved overall performance.

Thus, as the first thesis contribution, we introduce a novel genetic programming (GP) algorithm that incorporates indicators from all three analysis types. This integration serves as the motivation for Chapter 4.

Chapter 5 introduces the second thesis contribution, where the focus lies on enhancing the trading performance of GP algorithms through the adoption of a strongly-typed architecture. The strongly-typed GP architecture enables the separate handling of FA, SA, and TA indicators in distinct branches of the GP algorithmic tree, thereby facilitating better exploration and exploitation within each indicator type's search space. Moreover, we introduce a novel fitness function that allows GP to guide the search towards trading strategies with strong performance across all three components. It achieves that by considering not only the performance of individual trees but also the performance of FA, SA, and TA subtrees. This approach ensures that all analysis indicators contribute to the overall performance of a GP individual. The objective is to evolve individuals that demonstrate strong performance in each component (FA, SA, and TA subtrees), as well as in the overall individual.

The third thesis contribution is shown in Chapter 6. The proposed GP algorithm builds upon these improvements by encouraging the algorithm to follow an active trading approach, which monitors and analyses market conditions to identify short-term trading opportunities. Such trading strategies from taking advantage of price movements within relatively short timeframes, thereby increasing their profitability. To achieve this, we create a new GP operator. This novel operator identifies the FA, SA, and TA subtrees that are using highly active trading strategies, achieving high profitability at low risk, and injects them into the following generation by combining them into a new tree.

The primary goal of this research is to demonstrate the capability of the proposed GP variants to generate unique and profitable trading strategies that leverage information from all three analysis types. Performance comparisons are conducted against GP benchmarks in Chapters 4 - 6, including the corresponding GP variants, machine learning, and financial benchmarks. Experiments are conducted on the stocks of 42 international companies, and results are analysed based on three financial metrics: Sharpe ratio, rate of return, and risk. The research seeks to establish the effectiveness of the proposed GP algorithm and its potential to enhance trading strategies in financial markets.

1.2 Thesis overview

The thesis is structured into eight distinct chapters, each addressing various aspects of the research. These eight chapters are separated into two main parts, Part I and Part II.

Part I starts with Chapter 2, which focuses on presenting financial forecasting and algorithmic trading as the topics we try to explore in the thesis. Moreover, the chapter includes a comprehensive review of the literature on algorithmic trading and genetic programming, comprised of sections dedicated to each of the analysis types, namely fundamental, sentiment, and technical analysis, and it introduces the data derived from the three financial analysis types and the generated indicators. Additionally, the chapter introduces studies that have conducted pairwise combinations of the individual analysis types, as well as studies for the integration of all three. A critical review of existing literature is mentioned at the end of the chapter as a means to highlight gaps and motivate the research direction of this thesis. Chapter 3 is dedicated to the genetic programming methods used in literature, aiming to provide a concise introduction to genetic programming algorithms. More specifically, the chapter investigates the description of a GP algorithm, including its representation, initialisation methods, operators, breeding methods, and selection strategies.

Part II is dedicated to showcasing the contributions this PhD thesis makes, and it is divided into three chapters, namely Chapter 4, 5 and 6. Starting, Chapter 4 introduces the individual GP algorithms and compares them with the combination of the analysis types indicators using a novel GP algorithm. Following, in Chapter 5, we focus on comparing GP algorithms of Chapter 4 with the two strongly-typed GP algorithms introduced in Chapter 5 to showcase the advantages of the strongly-typed architecture, along with the novel fitness function. Lastly, in Chapter 6, we propose a novel GP algorithm that utilises a novel GP operator used within a strongly-typed GP architecture, which focuses on active trading.

Chapter 7 presents the conclusion of the research. This chapter offers sections that summarise the work conducted and the results found in Chapters 4 - 6, while it provides a final summary of the research and delves into discussions regarding future potential research areas.

Part I

Background Information

Chapter 2

Financial Trading

Incorporating fundamental, sentiment, and technical analysis within algorithmic trading, individually and as a combination, has revolutionised the financial markets by enabling market participants to navigate the complexities of the modern digital era. As markets become increasingly complex and competitive, the need for algorithmic trading strategies that swiftly adapt to changing conditions has never been more pronounced. Each financial analysis type has become a subject of intense research and innovation, especially within the algorithmic trading domain. This literature review will explore the evolving landscape of algorithmic trading, focusing on the advancements made in incorporating these three financial analytical domains.

In this chapter, we will delve into the concepts of financial forecasting and algorithmic trading, as well as the financial indicators employed in our research as solutions to the above topics, and relevant literature review related to them. More specifically, we will present the background information about financial forecasting in Section 2.1 and then discuss how forecasting can be used as part of algorithmic trading in Section 2.2. Finally, we will introduce previous papers and research on the three financial analysis types: fundamental in Section 2.3, sentiment in Section 2.4.2, and technical in Section 2.5, along with the financial indicators we will use in our research. In the following chapters, we also address the methodology of extracting the financial data in our study.

2.1 Financial Forecasting

Based on Plummer, [7], financial forecasting is part of the wide world of finance and business management. It mainly involves the estimation of future financial outcomes for a company, project, or investment, typically based on historical data and analysis of current trends and market conditions. The objective of financial forecasting is to make informed decisions to optimise stock market investments and produce stock market profits, strategic planning, setting long-term goals, and scenario analysis. It also involves the preparation of possible outcomes, evaluating the financial viability of projects, managing the cash flow and budgeting, risk management, and assessing the overall financial health of a company. The part of financial forecasting we focus on in this PhD thesis is related to the stock market, where many organisations use financial models and software to streamline the forecasting process. These models can predict stock prices, perform data analysis, and scenario testing.

Continuing with Plummer, [7], these forecasts involve the evaluation of a financial instrument's intrinsic value by utilising economic indicators and financial statements. This method is known as fundamental analysis (FA). Furthermore, stock price data are exceptionally common to seek patterns and trends to forecast future market movements. This technique is known as technical analysis (TA) . Based on Blascoa et al, [8], extracting valuable insights from news sentiment or social media chatter can also be used in financial forecasting. The above falls into the category of sentiment analysis (SA). All three analysis types have distinct characteristics that support and assist in predicting the stock market and creating trading strategies. Thus, in our study we use all three types of financial analysis data. More details can be found in Sections 2.3 - 2.5.

In more conventional methods, forecasts are based on assumptions about various economic and financial factors, such as interest rates, inflation rates, market demand, and industry trends. The accuracy of a forecast depends heavily on the validity of these assumptions. In our methods, we heavily employ machine learning algorithms, with a focus on genetic programming algorithms, aiming for a more accurate and reliable financial forecast. Moreover, the timeframe of such forecasts can vary and cover any time period, depending on the specific needs of the project. In our research, the timeframe covers 5 years, which is considered a medium to long period.

It needs to be noted that financial forecasts must be regularly monitored and revised as new information becomes available or as circumstances change. This adaptability is crucial for accurate financial planning and predictions.

2.2 Algorithmic Trading

Algorithmic trading, also known as "algo trading" or "automated trading", is a method of executing trading strategies using computer algorithms. Based on Ritter, [9], algorithmic trading involves the use of mathematical models and computer programs to automate the process of buying or selling financial instruments, such as stocks, options, futures, and currencies. These algorithms are designed to analyse market data, identify patterns, make decisions, and execute trades at high speeds and frequencies that would be nearly impossible for a human trader to achieve. This transformation in trading practices is driven by the three critical analytical domains, fundamental analysis, sentiment analysis, and technical analysis. When harnessed within algorithmic trading systems, these three analytical approaches empower market participants to make informed decisions, capitalise on emerging opportunities, and mitigate risks in an ever-evolving financial landscape. In this study, we focus on the implementation of algorithmic trading in the stock market.

Continuing, as stated by Ritter, [9], algorithmic trading has emerged as a popular and promising sector in the financial industry, driven not only by its potential for high returns but also by its continuous development and variety. Researchers have been exploring different data sets, timelines, techniques, and methodologies to predict stock market behaviour, financial instrument prices, potential global crises, bankruptcies, and company profits. Accurate predictions hold a distinct advantage in generating profits and mitigating substantial losses, making such forecasting methods highly valuable in both the stock market and individual company contexts. Advancements in technology have significantly enhanced the accuracy of financial forecasting and the profits generated by algorithmic trading, moving away from traditional mathematical methods to leverage technological innovations over time.

The main reason traders choose to utilise algorithms in their trading is the speed and efficiency they provide. Algorithms can execute trades in a matter of microseconds, allowing traders to take advantage of even the smallest market opportunities. Based on Aldridge, [10], the speed of technology is also used to minimise trading execution delays (latency), meaning the time lag between the moment a trading order is submitted and when it is executed. This is important especially in High-Frequency trading (HFT), [1] since even a small delay can

[1]High-Frequency Trading (HFT) is a subset of algorithmic trading that focuses on extremely high-speed execution. HFT firms often make a large number of small trades in a very short period. Their profitability depends on

impact the profitability of a strategy. Moreover, algorithms use historical and real-time data to make trading decisions, analyse vast amounts of data, identify patterns, and make predictions. Risk management is another reason for the use of algorithms in trading, as they can be programmed to set stop-loss orders, control position sizes, and limit exposure to market risks. Furthermore, the use of algorithms can provide liquidity to markets by constantly quoting buy and sell prices, this way maintaining a liquid market and earning profits from the bid-ask spread [2]. Machine learning and artificial intelligence are increasingly used in algorithmic trading to develop predictive models and adapt strategies to changing market conditions. At the same time, intensive backtesting is fundamental to evaluate how the algorithm would have performed in the past and optimise the trading strategies. Overall, algorithmic trading can have a significant impact on markets, and many firms (hedge funds, proprietary trading firms, and financial institutions) invest heavily in research and technology to gain a competitive edge over new strategies. Algorithms have revolutionised the financial markets by increasing profits while reducing trading costs.

2.3 Fundamental Analysis

As seen in Krantz, [11], Financial forecasting and algorithmic trading rely on the use of different information provided by the companies. The study of this information is called fundamental analysis (FA). More specifically, fundamental analysis is a method of evaluating and analysing securities based on intrinsic factors that influence their value. It involves assessing various qualitative and quantitative factors related to the underlying asset, such as financial

the speed and low-latency access to markets.

[2]Bid-ask spread is the difference between the highest price a buyer is willing to pay (the bid price) and the lowest price a seller is willing to accept (the ask price) for a particular asset.

statements, industry trends, competitive advantage, management team, economic indicators, and market conditions. The goal of fundamental analysis is to determine the intrinsic value of an asset and assess whether it is overvalued or undervalued in the market. [3]

FA is commonly used by investors to make long-term investment decisions and identify potential opportunities based on the underlying fundamentals of the asset. Fundamental analysis can also be used in conjunction with other methods, such as sentiment and technical analysis, to gain a comprehensive understanding of an asset's investment potential.

2.3.1 Literature Review on fundamental analysis

Fundamental analysis is a crucial approach used by investors to evaluate the intrinsic value of a stock, analysing various factors that can impact a company's financial performance and, consequently, its price. It focuses on the objective and quantitative aspects of a company's overall performance and external economic events. It is the earliest form of studying the stock market and a company's values. Researchers have recognised the importance of incorporating fundamental company information into predicting its future stock value through traditional and machine learning techniques. The scientific papers shown in this section are presented based on their topic and research focus.

There have been studies underlying the importance of fundamental analysis in stock markets and its potential for generating significant returns, such as Ng et al, [12], Yan and Zheng, [13]. There are also many machine learning applications in fundamental analysis, as we see from Chen et al, [14], in which the authors address data accessibility challenges and employ

[3]The intrinsic value of a stock refers to the estimated underlying value of a company's stock based on its fundamental characteristics, such as its earnings, assets, growth prospects, and other relevant factors. It represents an investor's assessment of the stock's true worth, independent of its current market price.

machine learning models to forecast one-year-ahead earnings changes and stock returns.

Artificial Neural Networks (ANN) are a popular domain when exploring the use of fundamental analysis indicators. An early paper for value-based stock selection criteria is of Eakings and Stansell, [15], in which the authors conclude that value-based stock selection can lead to superior risk-adjusted returns. Moreover, the study of Quah, [16], explored the application of Neural Networks in stock selection, demonstrating its consistent outperformance of market benchmarks over time by effectively identifying top-performing stocks, and they managed to achieve positive compounded excess returns. Similarly, the authors of Eng et al, [17], also used an ANN, but this time to predict daily foreign exchange rates, implementing interest rates, Gross Domestic Product (GDP), quarterly trade balance numbers and the Consumer Price Index (CPI). In the work of Abe and Nakayama, [18], the authors utilised 25 indicators, like Book-to-market ratio, Earnings-to-price ratio, Sales-to-price ratio and Cash flow-to-price ratio. They showed that the accuracy increases when using more layers in a Deep Learning Model. Hybrid models are also popular. For example, in He et al, [19], the authors proposed a fusion of linear regression and an XGBoost model that, firstly, serve as base models individually and later are fused. The researchers showed the high performance of their fusion model and improved accuracy with data from the Balance Sheet, Income Statement and Cash Flow Statement of companies.

Moreover, fuzzy systems have also been used as well, such as in Qyah, [20], where the authors used feed-forward neural networks (FNN) and adaptive neural fuzzy inference systems (ANFIS) for stock selection, evaluating three soft-computing models. More specifically, the evaluation included multi-layer perceptrons (MLP), ANFIS, and general growing and pruning radial basis function (GGAP-RBF), considering fundamental attributes while also introducing the concept of using relative operating characteristics (ROC) to systematically select equities based on the strength of predicted output values from the neural network models. In the use of FNN and ANFIS models, [21] evaluated their performance in predicting stock performance using fundamental financial ratios. Both models effectively distinguish winning and losing stocks and outperform the benchmark index, with FNN displaying superior performance in constructing "Buy" and "Sell" portfolios.

Many popular algorithms have been used in algorithmic trading implementation, as well as novel algorithms. One such algorithm is that of McGroarty et al, [22], which incorporated an agent-based simulation model to explore algorithmic trading strategies using fundamental analysis to predict financial asset price movements. The model features five types of agents representing various trading behaviours and successfully replicates statistical properties of high-frequency order-driven markets, including extreme price spikes and volatility clustering.

The effect of fundamental indicators has also been studied. In their work, Fama and French, [23], reported that the high fundamental ratio stocks, such as price-to-earnings (P/E), have higher average returns than the stocks with low ratios due to their growth rating in the stocks. Based on the F-tests in the later research of Herawati et al, [24], indicators like Debt to Equity Ratio (DER), Return on Assets (ROA), Current Ratio (CR), Price Earnings Ratio (PER), and Total Assets Turnover (TATO) were found to affect the fluctuations in the stock prices, while based on the t-statistic test results ROA and TATO partially influence the share price. Similarly, Luckieta et al, [25] analysed the Return on Assets (ROA), Earning Per Share (EPS), Price Earning Ratio (PER), and Debt To Equity Ratio (DER) on stock prices. Their findings suggested that EPS, PER, and DER have a positive and significant effect. For accounting ratios, Ajekwe and Ibiamke, [26], used financial statement information and predicted one-year returns with an accuracy of 76.6% using multiple logic models. Huang et al, [27], addresses the prediction of stock performance using machine learning methods with a focus on long-term predictions based on fundamental analysis. A substantial dataset of 22 years' worth of stock financial data is used, and three machine learning algorithms are experimented with, with Random Forest (RF) achieving the best performance. The aggregated model shows promising results, suggesting the potential for using historical financial data.

2.3.2 Financial information on Fundamental analysis

When conducting fundamental analysis for stocks, investors often examine financial ratios, such as the Price-to-Earnings (P/E) ratio, Price-to-Book (P/B) ratio, and Debt-to-Equity ratio, to assess a company's financial health and valuation. By considering both quantitative and qualitative factors, fundamental analysis provides a well-rounded assessment of an asset's intrinsic value, helping investors make informed decisions in the financial markets.

Thus, within our research, a comprehensive array of twelve distinct and widely adopted financial indicators, as seen by Damodaran, [28], is employed for fundamental analysis (FA). These indicators include the Net Profit ratio (NPR), Return on Equity (ROE), Quick ratio, Debt to Equity ratio, Price-Earnings ratio, Price to Book ratio, Price-Sales ratio, Total Revenues, Levered Free Cash Flow, Diluted Earnings Per Share (EPS), Earnings Before Interest, Taxes, Depreciation, and Amortization (EBITDA), and Research and Development (R&D) expenses. A concise summary of all indicators can be found in Table 2.1.

The data utilised in this research was procured from the 10-K filings of the esteemed financial content service company, "Seeking Alpha"[4]. These filings served as a valuable source of information pertaining to the companies under investigation. Among the twelve aforementioned indicators, namely Diluted EPS, Total Revenues, EBITDA, Levered Free Cash Flow, and R&D Expenses, these particular metrics were readily accessible within the 10-K filings. For

[4]https://seekingalpha.com. Last accessed: May 2023.

Table 2.1: Fundamental Analysis indicators

Fundamental analysis indicators	
$\frac{\text{NetIncome}_y}{\text{Revenue}_y}$	Net Profit Ratio
$\frac{\text{NetIncome}_y}{\text{ShareholderEq}_y}$	Return on Equity
$\frac{CurrentAssets_y - Inventory_y}{CurrentLiabilities_y}$	Quick ratio
$\frac{TotalDebt_y}{ShareholderEq_y}$	Debt to Equity
$\frac{price_j}{EPS_y}$	Price-Earnings ratio
$\frac{price_j}{BVPS_y}$	Price to Book ratio
$\frac{price_j}{Revenue_y}$	Price-Sales ratio
	Total Revenues, Levered free Cash Flow
	EBITDA, Diluted EPS, R&D Expenses

the remaining seven indicators, we derived them employing the following equations, where y denotes the financial year, $price$ represents the stock value, and j corresponds to a specific day within the five-year period.

More specifically, Diluted EPS, measures a company's earnings available to the shareholders after accounting for the potential dilution of convertible securities. The metric provides a more conservative measure of a company's earnings per share, considering the potential impact of dilutive securities, while it is important for investors to assess the company's financial health and profitability on a per-share basis. It is calculated by dividing the net income available to common shareholders by the weighted average number of diluted common shares outstanding, and it is available in the 10-K filings.

Total Revenues, also provided in the 10-K filings, represent the total amount of money generated by a company from its primary operations (sales of goods or services to customers), and they reflect the company's ability to generate income from its core operations. They are typically recorded on a company's income statement and represent the sum of all sales transactions during a specific accounting period.

EBITDA is a metric that measures a company's operating performance, indicating its earnings before accounting for interest expenses, taxes, depreciation, and amortization [5], providing a clearer view of a company's operational profitability. It is calculated by adding back interest, taxes, depreciation, and amortization to the net income.

Levered Free Cash Flow measures the cash flow available to a company's investors, including equity and debt holders, after accounting for operating expenses, capital expenditures, and interest expenses. It helps assess a company's ability to generate cash flow to cover both its operating and financial obligations, evaluating financial health and the potential for dividends or debt repayments. The metric is calculated by subtracting operating expenses, capital expenditures, and interest expenses from the company's operating cash flow.

Research and Development Expenses show the costs incurred by a company in researching and developing new products, services, technologies, or processes, which are significant for innovation and future growth. They indicate a company's commitment to innovation and its investment in future growth, with high numbers of R&D spending signifying a company's focus on staying competitive and developing new revenue-generating opportunities. These expenses are reported on a company's income statement as part of its operating expenses.

The Net Profit ratio (NetProf) stands as a pivotal profitability metric, evaluating a com-

[5]Amortization is an accounting and budgeting process that refers to the gradual reduction or repayment of liability or the allocation of the cost of an intangible asset over a specific period. It is used to spread out the cost of certain expenses or assets over time rather than recognising them as a one-time expense.

pany's financial performance and efficiency. A higher Net Profit ratio is generally more favourable, indicating that the company is generating a larger profit margin relative to its revenue. Businesses use the Net Profit ratio to identify areas where profitability can be improved and to make pricing adjustments, cost-cutting measures, and expansion plans. It is calculated by dividing the Net Income by the Revenue, ultimately measuring the portion of each dollar of revenue that remains as profit after covering all operating expenses (interest, taxes, and other costs).

$$\text{NetProf}(y) = \frac{NetIncome_y}{Revenue_y} \tag{2.1}$$

Return on Equity (RoE) is an imperative profitability measure employed to quantify the return generated on the shareholders' investment. It is a key profitability and financial performance indicator used by investors, analysts, and businesses to assess how efficiently a company uses shareholders' equity to generate earnings. A higher RoE indicates that the company uses shareholders' equity effectively to generate profit. It is conventionally calculated by dividing the Net Income by the Shareholders' Equity.

$$\text{RoE}(y) = \frac{NetIncome_y}{ShareholderEq_y} \tag{2.2}$$

The Quick ratio (QuickR) serves as a vital liquidity ratio, assessing a company's short-term liquidity and its ability to meet its immediate financial obligations without relying on the sale of inventory. This means how easily the company can utilise its cash or quick assets to promptly settle immediate liabilities. It is a valuable measure of financial health and liquidity, especially when a company faces sudden financial stress or economic downturns. This ratio is calculated by dividing the Current Assets minus Inventory by the Current Liabilities.

$$\text{QuickR}(y) = \frac{CurrentAssets_y - Inventory_y}{CurrentLiabilities_y} \tag{2.3}$$

Debt to Equity (DebtEq) measures a company's financial leverage or capital structure. It represents a critical metric elucidating the proportion of a company's assets financed through shareholders' equity and debt. The Debt to Equity ratio provides insights into a company's capital structure, showing the degree to which it relies on debt financing compared to equity. A high ratio indicates greater financial leverage, while a low ratio suggests less reliance on debt. This metric is determined by dividing the Total Debt by the Shareholders' Equity.

$$\text{DebtEq}(y) = \frac{TotalDebt_y}{ShareholderEq_y} \tag{2.4}$$

The Price-Earnings ratio (P/E) stands as a significant valuation metric utilised to ascertain whether a company's stock is overvalued or undervalued, assessing the relative value of a company's stock by comparing its market price (per share) to its earnings per share (EPS). The P/E ratio is a key indicator of a company's valuation. It is a fundamental tool for investors when evaluating stocks since it shows how much investors are willing to pay for each dollar of earnings. It is calculated by dividing the Share Price (p) by the Earnings per Share (EPS).

$$\text{P/E}(j) = \frac{price_j}{EPS_y} \tag{2.5}$$

The Price to Book ratio (P/B) holds significance as a valuation metric employed to discern potential investment opportunities, providing insight into the market's perception of a company's assets and their market worth. It represents the total value of the company's assets minus its liabilities, and it is a metric of a company's net worth. It is ascertained by dividing the Share Price (p) by the Book Value per Share (BVPS).

$$\text{P/B}(j) = \frac{price_j}{BVPS_y} \tag{2.6}$$

The Price-Sales ratio (P/S) holds significance as a valuation metric used to assess the value assigned by financial markets to each dollar of a business's revenues, ultimately answering how the market values a company's top-line revenue. The P/S ratio is particularly useful when a company has low or negative earnings, in which traditional valuation metrics like the Price-to-Earnings (P/E) ratio are not applicable. The P/S ratio measures the company's revenue as a measure of its value. It is computed by dividing the Share Price (p) by the Revenue per share (Revenue).

$$\text{P/S}(j) = \frac{price_j}{Revenue_y} \tag{2.7}$$

All FA indicators were normalised between $[-1, 1]$. This normalisation process ensures that each indicator is scaled uniformly, facilitating meaningful comparisons and analyses across the diverse financial metrics used in the study. Normalising between $[-1, 1]$ is important because it standardises the values, making it easier to consistently compare and analyse different metrics.

2.4 Sentiment Analysis

As Vinodhini and Chandrasekaran, [29], and Liu state, [30], sentiment analysis (SA), also known as opinion mining, is a natural language processing (NLP) technique used to determine the sentiment or emotional tone expressed in a piece of text, such as a review, comment, social media post, or news article. The goal of sentiment analysis is to classify the sentiment as positive, negative, neutral, or sometimes more nuanced emotions like happy, sad, and angry. This analysis helps understand public opinion, customer feedback, and overall sentiment about a particular topic, product, brand, or event. However, it's important to note that sentiment analysis has many challenges, like context, sarcasm, irony, and cultural nuances, which can make the accuracy of finding a text's sentiment complex.

As a result, the accuracy of sentiment analysis models can vary based on the quality of training data and the sophistication of the algorithms used, which involve text collection, pre-processing, tokenisation, feature extraction and sentiment classification.

2.4.1 Literature Review on Sentiment analysis

Macroeconomic factors, global events, and human behaviour can influence the stock market movement. Hence, estimating the stock market can be a challenging task. Many researchers have studied the importance of events and news in predicting the stock market by developing sophisticated algorithms, combining neural networks, fuzzy systems and evolutionary computation with news and events happening in a local and global spectrum. The publications in this section will be presented based on their machine learning applications on algorithmic trading as a topic.

One of the most important and earliest papers to dive into the overall idea of sentiment analysis was Kohara et al, [31], which searched for ways of increasing the predictive power of multivariate models for financial forecasting with prior knowledge from newspaper headlines and neural networks. More recently, Ding et al, [32], produced a model from event-driven stock market prediction. They extracted events from news and used a deep convolutional neural network (CNN) to model the short-term and long-term influences on price movements. Similarly, Peng and Jiang, [33], used DNNs to predict stock price movements through historical prices and online financial news, showing that adding financial news into a standard financial data set can improve the model's accuracy. Furthermore, Mohan et al, [34], used RNN - LSTM (Recurrent Neural Network with Long Short-Term Memory) and Facebook Prophet [6], in conjunction with news sentiment analysis for enhancing stock price prediction accuracy. The

[6]Facebook Prophet, also known as Prophet, is an open-source tool from Facebook used for forecasting time series data which helps businesses understand and possibly predict the financial market.

authors emphasised the significant correlation between stock price movements and the publication of news articles, highlighting the inadequacy of predicting stock prices using historical data or textual information in isolation. The same year, Souma et al, [35], investigated the predictive potential of historical news sentiments derived from financial market performance for forecasting future market behaviour, utilising recurrent neural networks (RNN) with long short-term memory (LSTM) units.They employed a combination of deep learning methods to extract news sentiments (positive or negative) and utilised them as inputs for their model. Mehta et al, [36], proposed algorithm that leverages public sentiment, opinions, news, and historical stock prices to forecast future stock prices. The study highlighted the effectiveness of sentiment analysis (SA) in predicting stock price changes and emphasised the relationship between public sentiment and stocks, employing the LSTM (Long Short-Term Memory) deep learning technique combined with social media and financial news data. Sharaff et al, [37], also uses a comprehensive dataset that includes financial news headlines, and it aims to predict the sentiment score of the news headlines in the financial domain while discussing the impact of COVID-19 on financial events and the use of sentiment analysis for collective risk detection.

Support Vector Machines have been widely used in the algorithmic trading domain. An example is the work of Xie et al, [38], where the authors explored using semantic frame parsers to generalise sentences to scenarios. This way, they could detect a company's role, either positive or negative. Another one is of Zhang et al, [39], which demonstrated a Principal Component Analysis (PCA) and Support Vector Machines (SVM) combination model into China A-share and Hong Kong stock data. They extracted the events from internet news and the sentiment from social media to study the stock price changes. Liapis et al, [40], used Twitter, international newspapers, and hacker forums on both the dark web and the surface web to explore the prediction of stock price direction using various machine learning techniques and sentiment analysis (e.g. SVMs, LSTMs, CNNs, ARIMA). Their findings suggested that incorporating sentiment variables can lead to up to 18% better predictions of stock price direction. The study also discussed the influence of sentiment analysis on stock market prediction during health crises such as H1N1 and COVID-19, highlighting the potential links between social media posts and closing stock prices at specific time horizons.

Researchers also resort to Open Information Extraction techniques, like Etzioni et al, [41] and Ding et al, [42]. The authors compared SVM and Deep Neural Networks with bag of words technique and event features in 3 different intervals, 1 day, 1 week and 1 month. The concluded that event features are better predictors than bag of words in the stock market domain and that Deep NN are better than the SVM. Futhermore, that the quality of information is better than the quantity of information, as well as the relativity of information. Kirange et al, [43], delved into sentiment analysis of news headlines and its link to stock price prediction. The paper introduces an efficient prediction model that assesses emotions from real-time publicly available news. The research employed various classifiers for emotion classification, including Naive Bayes, k-Nearest Neighbours (KNN), and Support Vector Machines (SVM). The study underscored the significance of artificial intelligence and data mining techniques in analysing stock market data. In a similar manner, Huang and Liu, [44], implemented text mining technology to quantify social media opinions on stock-related news and incorporated them into a logistic regression model for improved prediction. Following, the study of Farid et al, [45], explored stock price forecasting in an emerging market using data mining techniques, specifically decision tree analysis. While traditional models are utilised for predictions, data mining methods showed promise for improving forecasting accuracy.

In the topic of Evolutionary computation, Hochreiter, [46], Yang et al, [47], Christodoulaki et al, [5], Christodoulaki and Kampouridis, [4], are among the few studies that have utilised sentiment analysis indicators as inputs to a GP algorithm for algorithmic trading. While their implementation and trading strategies differ, all studies successfully demonstrated the financial profitability of sentiment analysis. Hybrid models also exist. For example, Sharma et al, [48], introduced the integration of genetic algorithms with artificial neural networks to enhance stock market forecasting, with potential applications in algorithmic trading strategies. Huang et al, [49], introduced an innovative approach that integrates social media sentiment analysis, a hybrid genetic algorithm (HGA), and deep learning using Long Short-Term Memory (LSTM) to predict stock price changes. The study notably used chip-based indicators from the semiconductor industry and sentiment variables extracted from social media data in its predictive model.

Previously, Wang et al, [50], proposed their algorithm, ExpertRank, to evaluate news based on its content and the authority of the source, stating that the origin of the news source matters; if the news comes from an official source, is leaked or rumoured. Findings that were also established as part of Li et al, [51], research. The paper of Kim et al, [52], explored the use of sentiment analysis through news articles to predict stock prices. It involved constructing a sentiment dictionary and establishing a correlation between news articles' positive index and the subsequent day's stock price returns. For Twitter news sentiment, Bollen et al, [53], explored the correlation between public sentiment on Twitter and stock market movements, specifically focusing on the Dow Jones Industrial Average (DJIA). Similarly, Bogle and Potter, [54], investigated the use of sentiment analysis and machine learning to predict the stock market based on Twitter comments. The study introduced a hybrid predictive model called SentAMaL, which combined sentiment analysis with machine learning algorithms like decision trees, neural networks, and support vector machines, aiming to leverage the instantaneous impact of social media comments on stock markets. Pagolu et al, [55], also investigated the relationship between sentiments expressed on Twitter and stock price fluctuations, highlighting a robust correlation, where positive news and tweets related to a company were found to drive investment and boost stock prices. The sentiment analyser demonstrated its effectiveness using machine learning algorithms like Random Forest, Logistic Regression, and Sequential Minimal Optimization (SMO). Furthermore, Karalevicius et al, [56], suggested a connection between media sentiment and Bitcoin's price, revealing that investors tend to react strongly to news in the short term. The study employed sentiment analysis of news articles and blog posts to determine sentiment scores based on the articles' positive and negative language. Most recently, Das et al, [57], focused on predicting stock market movements during the COVID-19 pandemic by analysing public sentiments from various online sources. The authors utilised a variety of data sources for sentiment analysis and stock market prediction, including headlines, tweets, financial news, Facebook comments, stock Tweets for tweets of Bitcoin, asset-related news from media, and sentiment scores from different web scraped data sources. The findings indicated that the sentiment scores from the different sources significantly impact stock market movement prediction.

Zhang and Skiena, [58], compared sentiment from blogs and news and used a large-scale Natural Language Processing text analysis system to study how a company's media presence reflects on its stocks' trading volumes and returns. They concluded that data obtained from the internet are highly informative and confirm their performance. Moreover, in the work of Gross-Klussmann and Hautsch, [59], the authors highlighted the significant impact of intraday company-specific news on high-frequency trading activity, affecting returns, volatility, trading volumes, and bid-ask spreads. Using a high-frequency Vector Autoregressive (VAR) model that accounts for the high proportion of zero variables, the authors analyse the relationship between high-frequency trading activity and intra-day volatility dynamics in financial markets, offering insights into the dynamics and cross-dependencies underlying market reactions to news events. In the topic of news sources, Day and Lee, [60], showed that different finance news sources have different characteristics, because of their business principles, the Review's team knowledge and specialisation on the topic, the different writing and wording style of their journalists, and the sensitivity of market trends of the media.

2.4.2 Financial information on Sentiment analysis

In our study, we first needed to acquire the needed data to generate and implement sentiment analysis indicators. To achieve that, we employed a Python-based web scraper, integrated with the Google Search Console API, to retrieve articles pertaining to the selected companies. The web scraper effectively accessed the first twenty pages of daily Google search results, utilising each company's name as a keyword. These articles were obtained for the same time-frame as the technical analysis indicators. Moreover, not only were the articles downloaded, but also their corresponding titles and summaries were retrieved, thereby facilitating the creation of polarity, subjectivity, and sentiment indicators not only for the body of each article but also for its title and summary.

In order to focus solely on relevant articles related to the selected companies, certain criteria were applied. Primarily, articles with a minimum length of 500 characters were considered. Additionally, to ensure relevance, these articles were required to mention both the company's name and its stock market ticker. This criterion helped filter out articles that were not directly relevant to the companies of interest or those that might have been mistakenly downloaded.

To align the sentiments expressed in the articles with the corresponding stock price data, a meticulous synchronisation of the publication dates of the articles with the relevant stock prices was undertaken. In cases where an article was published on a weekend when the stock market was closed, its sentiment was assimilated as part of the sentiment for the preceding Friday. This approach was designed to capture any potential impact on the stock price during the subsequent trading day, typically occurring on Monday.

When multiple articles were published for the same company on the same date, we adopted the approach of calculating the average sentiment value of these relevant articles. Conversely, for days with no published articles, we assigned a sentiment value of zero (0) to indicate neutrality or the absence of any specific sentiment, thereby ensuring continuity in our data points.

Let us now proceed to the presentation of the sentiment analysis indicators. Two commonly used indicators in sentiment analysis are sentiment polarity and subjectivity of given texts. Sentiment polarity captures the overall sentiment inclination, categorising the text as positive, negative, or neutral. On the other hand, subjectivity measures the degree to which the text expresses a personal opinion rather than objective facts. Our analysis utilises indicators based on these concepts while distinguishing between different calculation methods. The definitions of the respective methods are provided below.

In sentiment analysis classification research, specialised sentiment analysis programs are commonly employed to compute the polarity and subjectivity of text. Three popular tools in this domain are *TextBlob* by Loria, [61], *SentiWordNet* by Baccianella et al, [62], and *AFINN sentiment* by Nielsen, [63]. *TextBlob* represent a Python library that offers a straightforward API for determining the polarity and subjectivity of text. *SentiWordNet 3.0,* on the other hand, constitutes a lexical resource based on the English language's lexical taxonomy, *WordNet*, specifically designed for sentiment classification and opinion mining. It comprises a list of words classified as positive, negative, or neutral, and the overall sentiment of a given text is calculated as a weighted average of these words. *AFINN sentiment* is a widely utilised sentiment lexicon developed by Finn Årup Nielsen, encompassing over 3300 words, each assigned a polarity score. In our research, we leverage the built-in function for the *AFINN sentiment* lexicon, which is readily available in Python.

In our study, our attention is directed towards examining 12 distinct sentiment analysis (SA) indicators, which are concisely summarised in Table 2.2. These indicators are derived from the polarity and subjectivity levels obtained using *TextBlob*, in addition to the sentiment polarity extracted through *SentiWordNet* and *AFINN sentiment*. We conduct separate analyses on the articles, titles, and summaries, culminating in a total of 12 SA indicators.

Table 2.2: Sentiment Analysis Indicators. The designation "TEXT" refers to the complete article, "TITLE" pertains to the title of the respective article, and "SUMM" corresponds to the summary extracted from the Google Search results for that particular article.

Analysis type	**Indicator**
Sentiment Analysis (TextBlob)	TEXTpol, TEXTsub
	TITLEpol, TITLEsub
	SUMMpol, SUMMsub
Sentiment Analysis (SentiWordNet)	TEXTsenti, TITLEsenti, SUMMsenti
Sentiment Analysis (AFINN)	TEXTafinn, TITLEafinn, SUMMafinn
	ERC

All SA indicators were normalised between $[-1, 1]$.

2.5 Technical Analysis

Based on Murphy, [64], technical analysis (TA) is a method used in financial markets to analyse and predict price movements and trends by examining historical market data, primarily focusing on price and trading volume. Unlike fundamental analysis, which looks at underlying economic and financial factors, technical analysis is based on the idea that historical price and volume patterns can provide insights into future price movements. Traders widely use technical analysis, especially in short-term and day trading, to make buying and selling decisions. Technical analysts use various tools and techniques to identify potential entry and exit points for trading decisions. The main concepts and components of technical analysis include price charts, trends, chart patterns, volume analysis, and technical indicators, as in the case of our study.

2.5.1 Literature review on Technical analysis

Machine learning algorithms have been applied to technical analysis, enabling the identification of complex patterns and the development of predictive models that adapt to changing market conditions based on technical analysis indicators. In this section, we will refer to previous research on technical analysis, which will be presented based on their methods.

Developed by Vapnik, [65], Support Vector Machines (SVM) are supervised learning models for classification and regression analysis problems. In the work of Wang and Choi, [66], the authors proposed a SVM model with a principal component analysis (PCA) feature selection method, wanting to predict the stock price movements in the Korean and Hong Kong stock market, using 10 years' worth of data for 1 day ahead estimations. They found high hit ratios in the movements' predictions and they were able to verify a co-movement effect between these markets and the American stock market. Similarly, Yu et al, [67], used PCA in an SVM model in order to extract the low-dimensional and efficient feature information because of the sensitivity of the SVM accuracy on the quality of the training set. Wand and Shang, [68], explored the idea of the Least Square Support Vector Machine (LSSVM) and tested it in the estimation of the daily movement of the China Security Index 300 (CSI 300). They selected 10 Technical Indicators as input variables and compared their model with a Probabilistic Neural Network (PNN) and two Discriminant Analysis models. Moreover, Kumar et al, [69], conducted a comparative analysis of five supervised machine learning algorithms (Support Vector Machine, Random Forest, K-Nearest Neighbors, Naive Bayes, and Softmax) for stock market trend prediction. The results highlighted that the Random Forest algorithm excels with large datasets, while the Naive Bayes algorithm performs best with smaller datasets. The accuracy of each algorithm is influenced by the number of technical indicators used for feature extraction.

Neural Networks have allowed researchers to conduct a wide scale of experiments in financial forecasting. The earliest NNs approach in the stock market estimation was produced by White, [70], who created a Fead Forward NN (FFNNs) to decode undetected occurrences in the price movements, such as their fluctuations. Other Neural Network models, except FFNNs, were also Backprogation models (BPNNs) and Recurrent Neural Networks (RNNs) used in financial forecasting. In their research, Mostafa, [71], used a Long Short Term Memory model due to its usefulness in time series classification, and especially financial time series. Furthermore, McDonald et al, [72], proposed a novel hybrid model for estimating that combines the ARIMA model and a self-organising fuzzy neural network (SOFNN). The authors evaluated their model with different data sets, including financial data, concluding that the performance of a hybrid model is effective for time series forecasting. The same year, Adebiyi, [73], compared the ARIMA model with ANN, finding that ANN outperforms the ARIMA model in the New York Stock Exchange stock data. Moreover, Lasheras et al, [74], concluded the same results when they compared the ARIMA model with a Multilayer Perceptron NN and Elman NN on copper spot prices from the New York Commodity Exchange. The key findings of Ayala et al, [75], indicated that integrating machine learning techniques with technical analysis indicators enhances the generation of trading signals, resulting in a more robust and profitable trading strategy. The hybrid approach combines machine learning methods like Multivariate Linear Regression, Artificial Neural Networks, Support Vector Regression, and Random Forest. In their research, Dixon, [76], applied NNs in high-frequency trading and Hu et al, [77], highlighted the effectiveness of combining LSTM with various deep learning methods, like DNN and reinforcement learning, for stock price prediction. Interestingly, the application of technical indicators in stock price forecasting by CNN has indicated no discernible positive impact. At the same time, LSTM-based models have consistently outperformed other approaches, including MLP, Random Forest, and SVM models.

In their work, Chen et al, [78], used a LSTM model in order to predict stock returns in the Chinese market, transforming the data into 30-days-long sequences with 10 learning features and 3-day earning rate labelling. They showed that the accuracy increased by adding the indices and that different data sets showed different accuracy values. Following, Kim and Won, [79], proposed a hybrid LSTM model with various generalised autoregressive conditional heteroscedasticity (GARCH)-type models of stock price volatility. They enhanced this predictive power by combining NN with multiple econometric models. Their GEW-LSTM and LSTM models with three GARCH-type models had the lowest prediction errors. Nelson et al, [80], used price history and technical indicators as inputs in a LSTM model to forecast the future trends of stock prices. They compared their model with a Multi-Layer Perceptron, Random Forest, and a pseudo-random model, as well as investment strategies. More specifically, Buy and hold (buy at the first time step and sell at the latest), Optimistic (if prices went up on the previous time step, then perform a buy and sell it on the following step) and Pseudo-random (perform a trading operation based on probabilities according to the class distribution). Mallikarjuna and Rao, [81], showed the results of 5 different models in predicting the stock returns in 24 markets that had categorised in 3 groups, selecting daily closing prices of 19 years. These models were linear, which was Autoregressive Integrated Moving Average (ARIMA), nonlinear Self-exciting threshold autoregressive (SETAR), Artificial Intelligence with Neural Networks, frequency domain, which is the Singular Spectrum Analysis (SSA) and a hybrid model. From these models, none of them can be used uniformly for all the markets in the groups chosen by the researchers, but SETAR and ARIMA perform well in most of the markets. Finally, Alessandretti et al, [82], used machine learning, specifically LSTM models, to predict cryptocurrency prices. The study found that three prediction methods outperformed a basic strategy, with LSTM consistently yielding the best results. Long et al, [83], used a Deep Stock-trend Prediction Neural Network (DSPNN) model, which combines transaction records and market information with knowledge graph and graph embedding techniques. Kumbhare et al, [84], used technical indicators on the application of algorithmic trading with an LSTM neural network for stock market prediction, and it provided evidence of the effectiveness of technical indicators in the topic. Kurani et al, [85], and Valli et al, [86], both explored many algorithms, such as SVM, MLP, RNN, and LSTM algorithms, finding all of them indicating promising results.

In evolutionary computation, the study of Li and Tsang, [87], demonstrated that GP algorithms could outperform commonly employed technical techniques. Similar results were achieved in other studies such as Kampouridis and Tsang, [88], Kampouridis et al, [89], Kampouridis and Otero, [90]. As demonstrated in Berutich et al, [91], and Brabazon et al, [92], GP algorithms can develop trading strategies, generate solutions that survive extreme market conditions, and create new solutions while optimising the solution parameters. Similarly, Long et al, [93], showed that genetic programming provides profitable results and outperforms 9 other machine learning benchmarks in terms of risk and Sharpe ratio. In Long et al, [94], the authors presented a novel GP algorithm using directional change based indicators for trading strategies, able to outperform physical time strategies and the buy and hold strategy. In the same topic, Long et al, [95], presented a multi-objective genetic programming approach for trading strategy optimisation and demonstrated its effectiveness in achieving higher cumulative returns and rates of return compared to single-objective optimisation and financial strategies. More recently, Ari and Alagoz, [96], presented an evolutionary hyper-parameter optimal genetic programming-based framework for a-day-ahead trend forecasting in the ISE100 and BIST100 datasets. Gene expression programming has also been used, especially in the work of Mostafa and El-Masry, [97], where the model outperformed traditional statistical techniques, neural networks, and autoregressive integrated moving average (ARIMA) models in predicting oil prices.

The topic of evolutionary computation has been used in combination with other models, such as in Kim and Han, [98], where the authors predicted the Korea Composite Stock Price Index 200 (KOPSI200) by combining genetic algorithms and neural networks, achieving 82% of accuracy in estimating weekly fluctuations in the stock market. More recently, Hadavandi et al, [99] used Genetic Fuzzy Systems with Artificial Neural Networks for predicting stock market prices to create a system that wanted to use minimum required input data and the least complex stock market model. Zhang, [100], used a Genetic Deep Neural Network (GDNN) with different activation functions to optimise the parameters and selected the best activation function combination for different neurons that can perform better than one using a single activation function. Furthermore, Chen and Wang, [101], proposed a Genetic Network Programming (GNP) and Mean Conditional Value-at-Risk Model (GNP–CVaR), combining evolutionary algorithms and a statistical model that is useful for investors that have different risk attitudes. Experimenting on five stock indices, the GNP and maximum Sharpe Ratio model performs the best in a bull market, contradicting the GNP and the global minimum risk portfolio that performed the best in a bear market. The system they proposed improved the efficiency of the original GNP. A hybrid deep learning model was created by Kamara et al, [102], that incorporated technical analysis for financial forecasting, using two stocks in their experiments. [103], proposed a genetic algorithm (GA) optimisation-based deep learning technique (CNN-LSTM) for predicting the next day's closing stock price. The findings indicated that the GA-based CNN-LSTM model outperforms single CNN and LSTM models and the CNN-LSTM model without GA optimisation regarding prediction accuracy. Moreover, Deng and Huang, [104], used various data to investigate the feasibility of leveraging the synergistic effect of a hybrid system for stock market prediction. The study explored the integration of neural networks and genetic algorithms to create a hybrid system for stock market prediction. Overall, the results suggested that the hybrid system is capable of learning and exhibiting promising behaviour for stock market prediction.

2.5.2 Financial information on Tehcnical analysis

In this research, we thoroughly examine of six widely adopted technical analysis indicators: Moving Average, Momentum, Rate of Change, Williams %R, Midprice, and Volatility. These indicators can be found at Murphy's work [64] and Fang et al, [105]. The equations are precisely defined in Equations (2.8) to (2.13), and their computations rely on historical data obtained from selected companies, specifically the (adjusted) close prices, highest and lowest daily prices. The datasets are sourced from Yahoo! Finance[7], and further elaboration on the datasets can be found in Section 4.3.1. Each indicator undergoes evaluation using look-up windows of $n = 5$ and $n = 10$ days, ultimately culminating in a comprehensive set of 12 TA indicators, as succinctly summarised in Table 2.3.

Table 2.3: Technical Analysis indicators. Each indicator is considered for two distinct lookup windows denoted by the variable n.

lookup windows $n = 5$ **and** $n = 10$	
$\frac{\sum_{i=j-n}^{j} p_i}{n}$, for $j \geq n$.	Moving Average
$p_j - p_{j-n}$	Momentum
$\sqrt{\text{Var}\left(\left\{\frac{p_{j-i}}{p_{j-n}} - 1\right\}_{i\in\{0,\dots,n-1\}}\right)}$	ROC
$-100 \cdot \frac{hh_{n,j}-cl_j}{hh_{n,j}-ll_{n,j}}$	Williams %R
$\sqrt{\text{Var}\left(\left\{\frac{p_{j-i}}{p_{j-n}} - 1\right\}_{i\in\{0,\dots,n-1\}}\right)}$	Volatility
Midprice$(n, j) = \frac{hh_{n,j}-ll_{n,j}}{2}$	Midprice

The Moving Average is a widely employed technical analysis indicator utilised to smooth out stock price data, thereby facilitating the identification of trends by mitigating the influence of noise. The adjusted closing price of the stock on the j-th day is denoted as p_j.

$$\text{Moving Average}(n, j) = \frac{\sum_{i=j-n}^{j} p_i}{n}, \text{ for } j \geq n. \tag{2.8}$$

The Momentum indicator computes the difference between the most recent adjusted closing price and the adjusted closing price n days ago. It provides insights into the strength

[7]https://finance.yahoo.com. Last accessed: July 2023

or weakness of a price trend. Positive Momentum values suggest upward price momentum, while negative values indicate downward momentum. For instance, if the Momentum is calculated as 10, the current price is 10 units higher than the price n days ago.

$$\text{Momentum}(n, j) = p_j - p_{j-n}, \tag{2.9}$$

while the Rate of Change (ROC) normalises the momentum.

$$\text{ROC}(n, j) = \left(\frac{p_j}{p_{j-n}} - 1\right) \cdot 100 \tag{2.10}$$

The Williams $\%$R indicator, as defined in Equation (2.11), is designed to compare the most recent closing price cl_j on day j to the highest high price $hh_{n,j}$ observed within the look-up window ending on day j. Additionally, it considers the lowest low price $ll_{n,j}$ observed within the same look-up window. Williams %R oscillates between -100 and 0, with readings below -80 considered oversold and above -20 as overbought. This indicator helps traders identify potential reversal points in price trends. For instance, if Williams %R reaches -85, the asset may be oversold, indicating a possible buying opportunity. Conversely, a reading above -15 might signal an overbought condition, suggesting a potential time to sell.

$$\text{Williams }\%\text{R}(n, j) = -100 \cdot \frac{hh_{n,j} - cl_j}{hh_{n,j} - ll_{n,j}} \tag{2.11}$$

Volatility is a statistical measure that quantifies the dispersion of returns over a specific period of time. It helps traders assess the level of risk associated with an asset. High volatility implies greater price fluctuations and risk, while low volatility suggests more stable prices. It is worth noting that while high volatility can offer trading opportunities, it also carries higher risk.

$$\text{Volatility}(n, j) = \sqrt{\text{Var}\left(\left\{\frac{p_{j-i}}{p_{j-n}} - 1\right\}_{i\in\{0,\ldots,n-1\}}\right)}, \tag{2.12}$$

where Var defines the sample variance over a dataset.

The Midprice, as defined in Equation (2.13), is computed as the midpoint value between the highest high price, $hh_{n,j}$, and the lowest low price, $ll_{n,j}$, observed across all days within the look-up window ending at day j. The Midprice can provide insights into price equilibrium within a given time frame. For instance, if the Midprice is closer to the high price within the look-up window, it suggests bullish sentiment, indicating potential upward price movement. Conversely, if the Midprice is closer to the low price, it may signal bearish sentiment and potential downward movement.

$$\text{Midprice}(n, j) = \frac{hh_{n,j} - ll_{n,j}}{2} \tag{2.13}$$

All TA indicators were normalised between $[-1, 1]$.

We generated 12 indicators from each analysis type to compare the algorithms fairly, and the indicators were selected based on their use and popularity in previous research papers and similar studies. The 36 indicators are used as individual and combined terminal sets when run within the GP variants.

2.6 Combinations

In addition to the individual applications of FA, TA, and SA, there have also been some limited works that have combined these techniques and will be presented next. In most cases, news sentiment are used as input alone, less often imported along with technical analysis indicators, and even more rarely combined with fundamental analysis. In earlier years, researchers would primarily focus on fundamental or technical analyses to generate profit, usually comparing them with each other or sticking to one of them based on their investment style. Only recently have traders and researchers started to analyse the stock market using fundamental and technical analyses. At the beginning of the research, sentiment analysis was used as the only method of production in a paper, along with the historical prices, in order to prove the efficiency of the technique. In recent years, sentiment analysis has been used in combination with the other two methods.

2.6.1 Fundamental analysis and Sentiment analysis

Most papers examine the movements of quantitative information to determine the value of an investment and its returns. In their reports, though, companies also provide written information in detail, information that can be added to the news about the companies. Apart from the fundamental analysis indicators, researchers also try to analyse the sentiment of annual reports based on how each company produces 10-K filings. This is the primary way researchers have implemented sentiment analysis within fundamental analysis, while there are exceptions that have used indicators derived from news articles, too. This section will present the papers based on the machine learning algorithms they used.

An example is the work of Loughran et al, [106], where the authors examined how often managers were using ethics-related terms in the 10-K annual reports of their companies ,in order to find a correlation between the accuracy of the words and how possible is for these companies to be a "sin" stock. They found that companies that used ethics-related terms were more likely to be the object of class action lawsuits and to score poorly on corporate governance measures. Later, Jegadeesh and Wu, [107], presented a new approach in order to quantify document tone. They obtained 10-K filings using a web crawler, resulting in a sample of $45,860$ filings from $7,606$ unique firms, with a mean market value of \$3.09 billion and a book-to-market ratio of \$0.65 billion. Also, they used a set of variables to reflect recent events: the size (the natural logarithm of the market before the 10-K filing), BM (ratio of the book value of equity of the fiscal year end in the 10-K), the volatility (standard deviation of the firm, estimated by using 60 months worth of returns as of the end of the month before the filing). Their research discovered a relationship between their tone measurements in the 10-K filings and how the market reacts to negative and positive words.

Regarding the challenges of sentiment analysis in the financial domain, Loughran and McDonald, [108], show that negative words, commonly used in other studies to measure the tone of a text, misclassify common words in the financial domain. They examined 10-K filings , where more than half of the words were identified as negative by the Harvard Dictionary that has been commonly used, but these words are not considered as negative in finance. They considered that tone could be used simultaneously with accounting numbers to drive information. Textual analysis can help understand the influence of information on stock returns. If the tone does not directly cause returns, it can be used as an additional way to capture information.

Corporate annual reports and sentiment analysis are a popular combination, as evident from Hoberg and Maksimovic, [109], that created variables extracted from the "Management's Discussion and Analysis" section of 10-K filings, Azimi and Agrawal, [110], which stated that corporate annual reports are more informative than it was found in previous studies. They trained an RNN model using 8,000 manually labelled sentences that got selected randomly from 10-K filings, and they achieved an in-sample accuracy of 91%. They separated the effects of negative and positive sentiment and observed that the market overreacted to negative sentiment and underreacted to the positive one during that period.

An interesting addition is that of Hajek and Munk, [111], where the authors utilised 1278 earnings conference calls of the 40 largest US companies. The data included linguistic indicators, vocal cues extracted from these earnings conference calls, and traditional financial indicators from corporate financial statements. The study proposes a deep learning architecture that combines managerial emotional states extracted from speech emotion recognition with FinBERT-based sentiment analysis. The study's findings suggested that managerial emotions are essential in predicting financial distress, even when compared with sentiment indicators obtained from text.

2.6.2 Fundamental Analysis and Technical Analysis

The comparison of fundamental and technical analysis has been the most seen application of the two analysis types in studies. However, there have been researchers who have combined the two. This section presents the papers that only combine fundamental and technical analyses based on the machine learning algorithms they use for algorithmic trading and finance.

To begin with the comparison element of the two financial analysis types, many researchers have examined their different performances, such as in the work of De et al, [112], where the authors observed that technical analysis performs better in terms of short-term trading. In contrast, fundamental analysis outperforms technical analysis regarding long-term investments. Likewise, Zhu and Zhou, [113], and Menkhoff, [114], observed that technical analysis outperformed fundamental analysis in short-term predictions, and that technical analysis rules are robust for modelling specifications.

There are fewer articles that combine technical analysis and fundamental analysis in contrast to those that compare them. One such study is by Bettman et al, [115], which confirmed the complementary nature of these techniques and concluded that while they perform well in isolation, models combining the techniques had greater explanatory power. In fuzzy systems, Zarandi et al, [116], incorporated fundamental and technical indicators into a type-2 fuzzy rule-based expert system to estimate stock price movements accurately. Through the experiments, their model accurately estimated the stock prices of different sectors and showed its robustness, flexibility and error minimisation. Similarly, in Ahmad et al, [117], where the authors combined technical analysis and fundamental analysis within a fuzzy system to make trading more rewarding and Pothumsetty, [118], which highlighted the advantages of algorithmic trading over traditional discretionary methods, emphasising the potential for enhanced efficiency and profitability.

The study of Beyaz et al, [119], compared the individual and combined performances of the two analysis types by using ANN and SVR, where it was found that fundamental analysis indicators outperformed those of technical analysis. In contrast, their combination outperforms the individual ones 95% of the cases with lower RMSE. Additionally, the authors demonstrated the differences among industries, highlighting that Health Care and Information Technology companies performed better when tested with fundamental indicators, but this trend was not as prominent for financial and energy companies. In evolutionary computation, Larijani and Dehghani, [120], utilised fundamental and technical analysis indicators for stock market prediction, using ANN, Fuzzy NN, genetic algorithms, and the firefly algorithm. The results indicated that the proposed combined algorithm, utilising the firefly algorithm and genetic algorithm, demonstrated higher prediction accuracy than previous algorithms.

2.6.3 Sentiment and Technical Analysis

Although not many studies have delved into the advantages of combining sentiment and technical analysis, this section presents notable papers that have done so. The papers are presented based on the machine learning algorithms they implement in algorithmic trading.

The most popular method of combining these analysis types is by integrating their indicators, as demonstrated in Vargas et al, [121]. The authors employed text mining on Reuters news related to the S&P500 index in a hybrid RNN and CNN model to predict prices and intraday directional movements. Using financial news articles and a set of technical indicators in their hybrid model, they showed that it performed better than CNN in the exact implementation. Furthermore, the authors showed how functional both technical and sentiment analysis are in financial forecasting. Similarly, Vargas et al, [122], explored the application of deep learning techniques to stock market prediction, focusing on algorithmic trading strategies. The authors compared two models for stock market forecasting: SI-RCNN, a hybrid model combining a Convolutional Neural Network (CNN) for processing financial news titles and a Long Short-Term Memory (LSTM) network for technical indicators, and I-RNN, a LSTM network that uses only technical indicators. The study finds that SI-RCNN outperforms I-RNN in terms of accuracy and profitability.

Another example is provided by Atkins et al, [123], where the authors indicated that information extracted from news sources is more effective in anticipating shifts in market volatility direction than predicting asset price movements. The authors developed machine learning models employing Latent Dirichlet Allocation (LDA) to represent information from news feeds and straightforward naïve Bayes classifiers for predicting movement directions. Moreover, the paper underscored the significance of considering the temporal aspect of news data, implementing a decay function to weigh the importance of past news articles and assign greater significance to recent ones, given their likely relevance for predicting market movements. Kim et al, [124], used sentiment analysis in news articles to predict stock prices, emphasising the construction of a specialised sentiment dictionary tailored for stock news. The study reveals a meaningful correlation between the positive index derived from news articles and subsequent day's stock price returns, considering the complexities of stock data and analysing the Korean language. The paper underlined the potential of sentiment analysis in the stock market and the superiority of a domain-specific sentiment dictionary. The paper of Ashtiani and Raahmei, [125], explored the utilisation of news sentiment, historical stock prices, technical indicators, and news text as data sources for stock market prediction, showing the significant enhancement in machine learning model performance achieved through sentiment analysis of news articles when predicting stock market trends. LSTM (Long Short-Term Memory) models, particularly when incorporating a substantial number of news headlines, have consistently exhibited superior performance compared to other deep learning models. Furthermore, Verma et al, [126], used various data for stock market forecasting, including historical stock data, technical indicators, sentiment analysis, Google search trends, COVID-19 cases, and domestic and foreign market behaviour. The study explored the use of deep learning models for predicting stock prices and identified various influential factors such as sentiment analysis and exceptional situations like epidemics. The authors of Wang et al, [127], proposed a 3-phase hybrid model for stock trending prediction, incorporating technical indicators and sentiments from social media text. The model leveraged machine learning algorithms such as XGBoost, LSTM, and CNN, demonstrating the proposed method's effectiveness. The results also indicated that integrating social opinions with technical indicators is a promising direction for enhancing stock market predictions.

Another way of combining the two analyses is an interesting concept, and it is implemented by creating probable scenarios, such as in Teymourian et al, [128]. The authors used event knowledge and standard company information in the paper to create a specific scenario. This information is available in their system, and the users can express their specific interests in companies with similar characteristics. The fusion of the events and the background knowledge assists in their event processing so they can know more about incoming events. The authors proposed an external knowledge base to provide information on the events so these events can be detected based on reasoning. The query from the user is pre-processed and rewritten into a single new query, so the user can define event queries without caring for some details. The authors created a lightweight ontology for the companies in the system, and the information about them comes from DBpedia, which extracts the wanted knowledge from Wikipedia. The events of the companies can be seen in the company's characteristics.

In the field of evolutionary computation and genetic programming, Christodoulaki et al, [5], showed that sentiment analysis produced better and statistically significant average results than technical analysis in terms of Sharpe ratio and risk. When it comes to combining technical and sentiment analysis indicators for algorithmic trading, the work of Christodoulaki and Kampouridis, [4], showed that the GP using the combined analysis types statistically outperformed the individual performance of the technical and sentiment analysis under several different financial metrics, as well as the financial benchmark of buy and hold. Another combination of examples are those presented in Yang et al, [47], and Christodoulaki and Kampouridis, [3]. More specifically, in Yang et al, [47], the authors found that using news and Twitter for sentiment analysis is more financially profitable than performing technical analysis alone or combining TA and SA indicators while considering a simple GP architecture. Nevertheless, in Christodoulaki and Kampouridis, [3], the authors demonstrated that combining technical and sentiment analysis indicators under a strongly-typed genetic programming framework is more financially profitable. Another study is that of Huang et al, [49], where the authors proposed a method that combines social media sentiment analysis, genetic algorithm, and deep learning to predict changes in stock prices, achieving higher accuracy after adding the sentiment analysis method.

2.6.4 Fundamental Analysis, Technical Analysis and Sentiment Analysis

Finally, the publications on all the methods combined will be presented based on the algorithms used during the research.

In the study by Weng et al. in [129], the authors used information on publicly traded companies while checking the public's interest by web traffic statistics. Combining online data sources with traditional time series and technical indicators for stocks can be essential for more effective trading. They evaluated their data fusion using Decision trees, Neural Networks, and SVM.

Following, Chlebus et al, [130], used many models with variables from fundamental analysis, technical analysis and behaviour analysis from Google trends regarding Nvdia, along with its marketing news. The authors used SVR, KNN, XGBoost, LightGBM, LSTM models and the econometric methods ARIMA and ARMAX. Another example is that of Picasso et al, [131], who, through Machine Learning techniques, produced more than 80% annualised return when running a high-frequency trading simulation.

In evolutionary computation, the work of Christodoulaki and Kampouridis, [1], used a novel genetic programming algorithm to combine the three financial analysis types and showcased the financial advantages of the combination.

Finally, Nazareth et al, [132], provided a comprehensive overview of findings and indicators related to various finance and machine learning topics, as discussed in different research papers. Key findings include the effectiveness of combining stock market and trader information for prediction, the impact of community sentiments and economic/political conditions on stock market volatility, and the rapid response to positive news compared to delayed reactions to negative news. The paper highlighted the utility of LSTM models in stock market prediction and mentioned using machine learning models like Random Forest, SVM, and Decision Trees for various financial predictions.

2.7 Critical Review

From the investigation on algorithmic trading with machine learning, we concluded that technical analysis serves as a better short-term trading indicator, and fundamental analysis is more advantageous for long-term investments. This can be because of the timeline the researchers examined, the indicators, and what they aimed to find. One approach could involve using fundamental analysis to identify promising companies with strong indicators and then applying technical analysis to exploit arbitrage opportunities. This approach appears logical, as the stock market accommodates various types of investors and traders, some risk-averse, while others are not. Some investors may have higher incomes than others, thus, to afford more risk. The showcased publications show that stocks with good fundamental rates can be appropriately timed into investing with technical analysis to generate higher profits.

There have been many ways to conduct sentiment analysis, mostly on social media or based on news articles. Since there is a need for more data on classified lengthy news articles, most researchers follow the dictionary-based approach. When using natural language processing tools, researchers focus on only one of these techniques and do not use their combined results. Recognising this as a gap in the literature, this study aims to combine the results of these tools to enhance accuracy and leverage additional indicators to achieve higher profits. Although many machine learning algorithms have been used to experiment with sentiment analysis indicators, the involvement of genetic programming has not been researched in that much depth.

There has been extended work on technical analysis, the best indicators, and the best models to use, while less work has been done on fundamental and sentiment analysis. Even fewer researchers have used the three methods as combinations, which allows us to explore their benefits even more, especially since the research on their fusion has been found to be more effective than their individual use. Considering the profitability of these methods individually and the possibilities that arise from combining them, we identify the above as gaps in the literature. Thus, we aim to create novel genetic programming algorithms to leverage the advantages of fundamental, sentiment, and technical analysis by examining their individual properties and their combination.

Finally, the most popular state-of-the-art models, especially for technical analysis indicators, seem to be LSTM models, which will be one of the benchmarks used in this thesis to compare the novel GP algorithms in Chapters 4 and 5. Of course, as mentioned in the literature review, hybrid models can increase accuracy. Furthermore, there has been less research on genetic programming related to sentiment analysis and fundamental analysis data, which we can experiment with since the technical analysis indicators have proven financially profitable.

Chapter 3

Genetic Programming

This chapter provides background information on genetic programming (GP) techniques used in literature and focuses on the ones used in the experiments of this PhD thesis. We are not aiming to offer a long and extended review of genetic programming but rather to introduce the GP concepts to the reader and also to present the main parts of the GP algorithms, which we used to facilitate our experiments. The rest of this chapter is organised as follows: Section 3.1 presents general information about genetic programming and Section 3.2 discusses the different GP representations. Section 3.3 introduces different initialisation methods, Section 3.4 presents the main GP operators, and Section 3.5 covers different breeding methods. Section 3.6 introduces the evaluation process of genetic programming, Section 3.7 presents the different selection strategies available, followed by Sections 3.8 and 3.9, showcasing the iteration and termination process of the algorithmic set, respectively. Section 3.10 presents the different data types a GP algorithm can utilise, and Section 3.11 briefly discusses the different choices of grammar that a GP system can use. Finally, Section 3.12 concludes this chapter. It needs to be noted that the specific methodology required for the experiments of each thesis contribution will be mentioned in the methodology parts of that specific chapter so as not to confuse the reader with the mentioned methods.

3.1 General Information

As stated in Poli et al, [133], Genetic programming is a computational technique inspired by the process of natural evolution that allows computers to evolve solutions to complex problems. It falls under the broader umbrella of evolutionary computation and artificial intelligence. The main idea behind genetic programming algorithms is to use principles of genetics and natural selection to automatically generate computer programs or solutions to various problems, where said computer programs act as the individuals of a population.

In GP, a population of candidate programs or solutions is evolved over multiple generations to find increasingly better solutions. This process mimics the way biological evolution operates in nature. The steps the program follows are standard, while in-between operations can be altered or added to further support the algorithms' performance and their outputs. During initialisation, the GP algorithm generates a random or semi-random population of trees/individuals, which represent solutions to the problem the algorithm tries to solve. This is achieved by using terminal sets and functions appropriate to the problem domain, where the former are variables and constants of the algorithms, and the latter are responsible for processing the values of the system (either terminals or other functions' output). Functions can be arithmetic operations (+, -, *, /), mathematical functions (sin, cos, exp, log), boolean operations (AND, OR, NOT), a conditional operator (If-Then-Else), functions causing iteration (Do-Until), comparison operators (>, <, =) [133].

Next is the Evaluation phase, in which each individual in the population is evaluated based on a predefined fitness function that measures how well each individual in the population solves the problem. The fitness function provides a quantitative measure of the individual's quality. After Evaluation, there is Selection, where individuals are chosen to produce new offspring programs based on the problem the algorithm tries to solve. In our case, the fitness function maximises of a specific financial metric (Sharpe ratio), which we will introduce in detail in Chapter 4. Thus, during Selection, the individuals with the higher fitness scores are more likely to be selected for reproduction, mirroring the concept of "survival of the fittest". This step aims to promote the traits of the better performing individuals in the population [133].

The next step is using the operators in genetic programming, where the individuals are manipulated by genetic operators such as crossover and mutation. These operations introduce diversity and novelty into the population, while there are more operators a researcher can implement or alter the current ones in order to create a GP architecture that works the best for their problem. In this thesis, we also create a novel GP operator, introduced in Chapter 6, [133].

Subsequently, there is the iteration of Evaluation and Selection for a set number of generations or until a stopping criterion is met. This is achieved by using the children produced by the previous generation. As the generations progress, the population ideally evolves towards better solutions to the problem as better performing traits become more prevalent. Finally, there is Termination, where the evolution process concludes when a satisfactory solution is found or when the algorithm reaches a predetermined stopping point, in which the final evolved individual is the solution to the problem [133]. The steps are better summarised below:

- Initialisation of a random or semi-random population.
- Evaluation of each individual and fitness assignment.
- Selection of individuals in order to produce new offspring by the means of different operators. These offspring form the new population.
- Iteration of the steps from generation to generation until a number of generations has been reached or the problem requirements have been achieved.
- Termination of the process with the generation of the final evolved solution.

Genetic programming can be applied to a wide range of tasks, including symbolic regression (finding mathematical expressions that fit given data), machine learning algorithm design and optimisation problems. It is a flexible technique that allows researchers and practitioners to automate the creation of programs without needing to manually design them. An important challenge in genetic programming is selecting suitable program representations and a suitable fitness function to guide the evolution effectively. Additionally, managing the balance between exploration (diversity) and exploitation (improving existing solutions) is crucial for achieving optimal results. Overall, genetic programming is a powerful approach that harnesses evolutionary principles to solve complex problems and create innovative solutions in various fields [133].

3.2 Genetic Programming Representations

As mentioned above, genetic programming is a machine learning technique that involves evolving populations of computer programs to solve complex problems. The process hinges on the concept of representation, which refers to how the genetic makeup (genotype) of individuals in the population is symbolised or encoded. This encoded form determines the structure and behaviour of the evolved programs.

Thus, John Koza's pioneering work in 1992 [134] better showcased the efficiency and effectiveness of the *tree-based* representation, leading to its widespread adoption among researchers. The *tree-based* GP representation has become a staple in the field due to its flexibility and expressiveness. The main advantage is that it allows for the evolution of programs with varying levels of complexity, which is crucial for tackling diverse problem domains. In this work, we use this *tree-based* GP representation due to its popularity amongst researchers, as it strikes a balance between capturing intricate program structures and being amenable to genetic operations like crossover and mutation.

Another variant is the graph-based GP, where individuals are encoded as graphs; the programs are represented as *directed acyclic graphs* (DAGs), where nodes represent functions or operations, and edges represent the data flow or control between nodes. This representation allows for more flexible and complex program structures than traditional *tree-based* GP, which can be particularly beneficial when dealing with large and complex data as seen in Teller and Veloso, [135]. A further innovation is the *Cartesian* GP, which was developed by Miller, [136]. Cartesian genetic programming can be considered an extension of the graph-based GP, and the algorithmic set utilises a grid-like structure for representing programs. In *Cartesian* GP, a program is represented as a set of nodes arranged in rows and columns, a multi-dimensional grid, where each node can perform a specific function, and the connections between nodes define the program's logic flow. The programs are executed by traversing this grid. This representation can provide benefits in terms of efficiency and direct parallelism, making it particularly suited for specific hardware implementations as seen by Miller et al in their works, [136], [137], [138], [139].

As shown by Cramer, [140], and Banzhaf et al, [141], beyond the *tree-based* representation, several alternative representations have been explored in the literature. One such representation is *linear* GP, which arranges genetic material sequentially. This representation is reminiscent of traditional genetic algorithms and can be advantageous for certain types of problems.

A more recent work is that of Franccoso, [142], where the authors examine the different GP representations, such as the *Cartesian genetic programming*, the *Linear genetic programming* (LGP), evolving graphs through graph programming, and traditional GP. The researchers empirically investigate and compare the performance of these techniques in various configurations, with a focus on their interactions with three different Evolutionary Algorithms (EAs): generational, steady-state, and $(1 + \lambda)$, revealing that the choice of technique and algorithm depends on the problem type. It also exhibits the advantages of *graph* GP methods, particularly in combination with specific evolutionary algorithms, and underscores the significance of reusing intermediate results for solving complex problems effectively.

Moreover, another study is that of Altenberg et al, [143], where the idea presented in the paper explores how genetic programming systems can evolve programs that are not only solutions to specific tasks but also have the capacity to evolve new solutions more easily or efficiently in the future, relating to the evolvability of the programs themselves. Evolvability has led to co-evolution, evolving populations of solutions together with other populations; diversity, which can lead to more evolvable solutions; modularity, which promotes the evolution of modular and reusable components within programs to support evolvability by allowing easy recombination of parts to solve different problems.

Finally, as of Hildebrandt and Branke, [144], the authors present a novel way to integrate surrogate models with genetic programming by using phenotypic characterisations instead of direct numerical representations, especially for computationally expensive fitness evaluations. Surrogate models are mathematical approximations of the fitness function that can be computed much more efficiently than the actual fitness evaluations. At the same time, they are usually created using statistical or machine learning techniques based on samples of fully evaluated solutions. Surrogate models typically require numerical representations, which do not align well with the tree structure of GP, so the paper uses a phenotypic characterisation to provide an efficient way to describe the traits or features of the evolved programs without relying on numerical values. More specifically, it involves creating a decision vector for each rule, quantifying how similar the rule's decisions are to those of the reference rule. If a rule's decisions match the reference rule, the elements in the vector are set to 1. Suppose a rule consistently chooses the job with the lowest priority according to the reference rule. In that case, the vector elements are set to the maximum value, corresponding to the number of jobs waiting in the decision situation. This approach abstracts away from the actual numeric values the rule produces, focusing solely on the decisions.

3.3 Initialisation of the population

In genetic programming, the process of initialising the population is a critical step that sets the stage for the evolutionary process to follow. Various methods have been devised to initialise GP populations, and three prominent approaches have been widely adopted: Full, Grow, and Ramped-Half-And-Half. These initialisation techniques were originally formalised by Koza in [134].

The first method, the Full initialisation, involves the systematic construction of trees with a uniform and regular structure. The process begins by randomly selecting a function from the available set of functions as the root of the tree. This chosen function serves as the parent node, and additional functions are selected as its children. This process is repeated iteratively, adding functions as internal nodes until one level before the maximum allowed depth for the trees. Subsequently, terminal nodes are randomly selected to complete the leaves of the tree. This method results in trees with a consistent depth and a uniform structure along any path from the root to a leaf. This structured approach is beneficial for maintaining a certain level of predictability in the initial population's composition [134]. An example of the Full initialisation method can be found in Figure 3.1.

Conversely, the Grow initialisation method introduces more diversity and variability into the population's initial structure. Like the Full method, a tree root is randomly selected, but the choice can come from either the set of functions or the set of terminals. If a function is chosen as the root, its arguments (child nodes) are then populated with random functions or terminals. This process continues recursively, adding functions and terminals along the branches until the tree reaches its maximum depth. As a result, the trees produced through the Grow method can vary widely in their shapes and sizes, allowing for greater exploration of the solution space [134]. An example of the Grow initialisation method can be found in Figure 3.2.

The Ramped-Half-And-Half initialisation method is a hybrid approach that seeks to strike a balance between the regularity of the Full method and the diversity of the Grow method. In this approach, trees are initialised using both the Full and Grow methods, with a controlled degree of variation in their shapes. Specifically, for each depth level within the range of two to the maximum initial depth, half of the trees are created using the Full method, and the other half are created using the Grow method. This combination ensures that the population contains both trees with a predictable structure and trees with more varied configurations

[134].

It is worth noting that beyond the above established initialisation methods, other approaches like the uniform initialisation have been proposed and studied in the literature; however, they are not used in this thesis. In the uniform initialisation method, individuals are generated uniformly at random, allowing maximum diversity but potentially needing more structured guidance provided by the Full and Grow methods. In the context of generating initial populations for Grammar-Guided genetic programming (GGGP) systems, as Garcia, [145], introduces a novel initialisation process in the generated individuals adhere to the grammar of the problem and do not exceed a predefined maximum size. This approach ensures computational savings and a more uniform distribution of individuals in terms of size and search space. Similarly, Ramos et al, [146], present their novel algorithm, which creates an initial population of individuals that adhere to the grammar's rules while maintaining uniformity and controlling bloat. This process can help set a solid foundation for subsequent genetic programming operations and improve the overall effectiveness of grammar-guided genetic programming.

The choice of the initialisation method can significantly impact the performance and behaviour of the GP algorithm, influencing the population's diversity, exploration capabilities, and convergence speed.

3.4 Genetic Operators

Genetic operators play a pivotal role in driving the evolutionary process by manipulating individuals in the population to generate new offspring. According to Banzhaf et al, [141], the primary genetic operators are reproduction, crossover, and mutation. Each operator contrib-

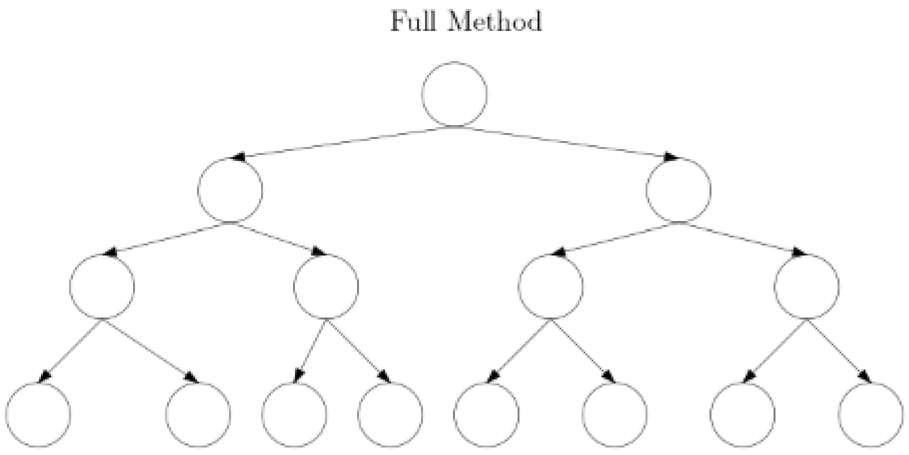

Figure 3.1: The Full method of population initialisation, as illustrated in [6].

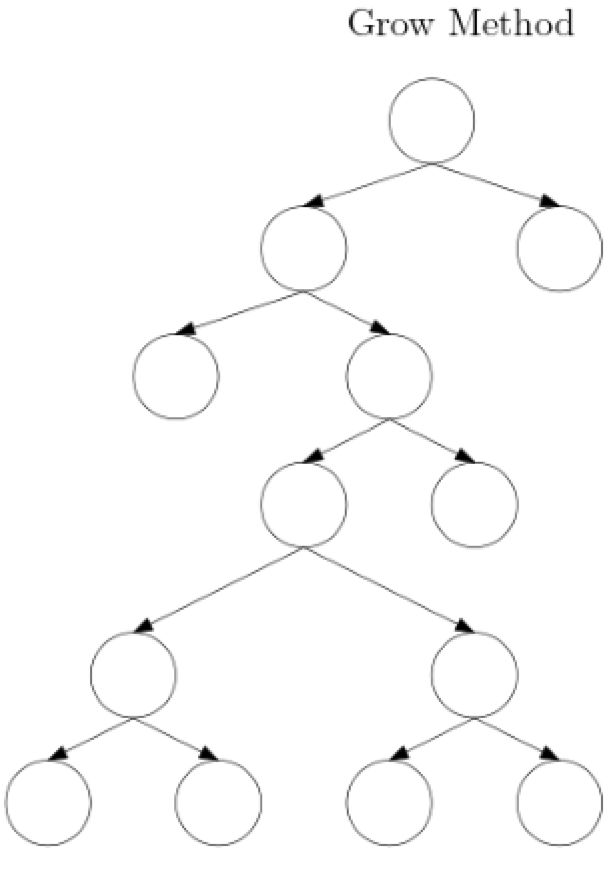

Figure 3.2: The Grow method of population initialisation, as illustrated in [6].

utes to the exploration and exploitation of the solution space in different ways.

Crossover is a key genetic operator that promotes the recombination of genetic material from two parent individuals to create offspring, with many variations. In our research, we use the subtree-crossover, which is a process that involves selecting two parents and a crossover point within their genetic structures. This point, referred to as the cross-point, acts as the division point in the parents' genetic material. An illustration of the division point can be found in Figure 3.3. By cutting the parents at this cross-point, two subtrees are obtained: the subtree from the root to the cross-point and the subtree from the cross-point onward. Two offspring are then generated. The first offspring combines the rooted subtree from the first parent with the subtree after the cross-point from the second parent. The second offspring is formed by the remaining parts of the two parents. Crossover facilitates the exchange of genetic material, potentially leading to the creation of novel and improved solutions [141]. An illustration of the crossover operator can be found in Figure 3.4. A lot of early information related to different crossover variations (uniform crossover [1], one-point crossover [2] and various combinations of these methods) can be found in the work of Spears and Anand, [147]. Finally, one more variation can be found in Poli et al, [148], where the smooth uniform crossover is introduced, in which two parents combine in a way that aims to produce offspring with less disruption to the structure of their genetic makeup.

Mutation involves altering a single individual by making small changes in its genetic makeup, such as altering a subtree. Unlike crossover, which involves two parents, mutation requires only one parent. The process begins by selecting a node within the individual's genetic structure. A randomly generated subtree then replaces the subtree rooted at this selected node. Mutation introduces small-scale variations in the population, enabling the ex-

[1]Uniform crossover combines genetic material from both parents with a certain probability for each gene

[2]One-point crossover combines two parent chromosomes by selecting a random crossover point and exchanging the genetic material between the parents at that point.

ploration of nearby regions in the solution space [141]. Furthermore, a variation of mutation is the one-point mutation, where only one node within the individual's genetic structure is altered. Unlike the standard mutation, which might replace any node in the structure, one-point mutation focuses on making a single change, often limited to a specific node, as seen in Poli et al, [149]. An illustration of the mutation operator can be found in Figure 3.5. Further mutation variations include the multi-mutation, introduced by Chellapilla, [150], which involves introducing multiple mutations to an individual's genetic material. This approach aims to inject a higher degree of diversity into the population by evolving programs through mutation and selective reproduction. Moreover, there is the Lévy mutation by Lee and Yao, [151], which applies a Lévy distribution (a probability distribution used to model heavy-tailed, long-range dependence phenomena) to mutate an individual's genetic material. This mutation strategy offers an innovative way to explore the solution space.

Elitism is the simplest of the genetic operators. It involves copying the best performing individual directly from the current population to the population of the new generation without any modifications. In other words, the individual remains unchanged. While this operator does not introduce diversity or variation, it ensures the preservation of well-performing individuals in subsequent generations. In our research, we use elitism in all thesis contribution chapters as part of all GP algorithms, as seen in Poli et al, [133].

Moreover, the work of Crawford and Spector, [152], extends on Langdons' work, [153], and introduces size control, where the authors investigate different operators and aim to produce offspring with similar sizes to their parents, thereby addressing issues related to program bloat in genetic programming. The "Size Fair" mutation operator described is a modification of an existing operator that produces mutations of length 1/4 of the original subtree length. The "Fair Crossover" operator is introduced to ensure that the average size of children resulting from crossover remains similar to the parents. Considering size limitations, this operator attempts to replace subtrees from the first parent with suitable subtrees from the second parent.

Additionally, a breeding method known as broad-selection, as proposed by Altenberg, [143], showcases a distinct approach to selecting individuals for reproduction, potentially yielding unique results compared to the previously discussed technique. Another mechanism someone can apply to the GP operators is by Cruz and Perdomo, [154], introducing the self-adaptation concept of genetic operators, meaning for the operators to change their behaviour over time. Hence, the algorithms eventually learn how to improve themselves while working.

Finally, Agapitos, [155] offers a comprehensive discussion of the aforementioned breeding methods and their respective merits, providing an in-depth discussion of their strengths, limitations, and applications.

While these additional variations of the genetic operators are not discussed in the context of the mentioned thesis, they highlight the ongoing efforts to design effective methods for evolving populations in genetic programming. Thus, in addition to the three main genetic programming operators, this thesis introduces a novel operator, which is made to solve the problem at hand, and it is introduced and explained in the methodology section of Chapter 6.

3.5 Breeding Methods

Breeding methods in genetic programming involve the strategies that determine which individuals are allowed to mate and how genetic operators are applied to generate offspring. Two primary criteria are used to classify and differentiate among various breeding methods.

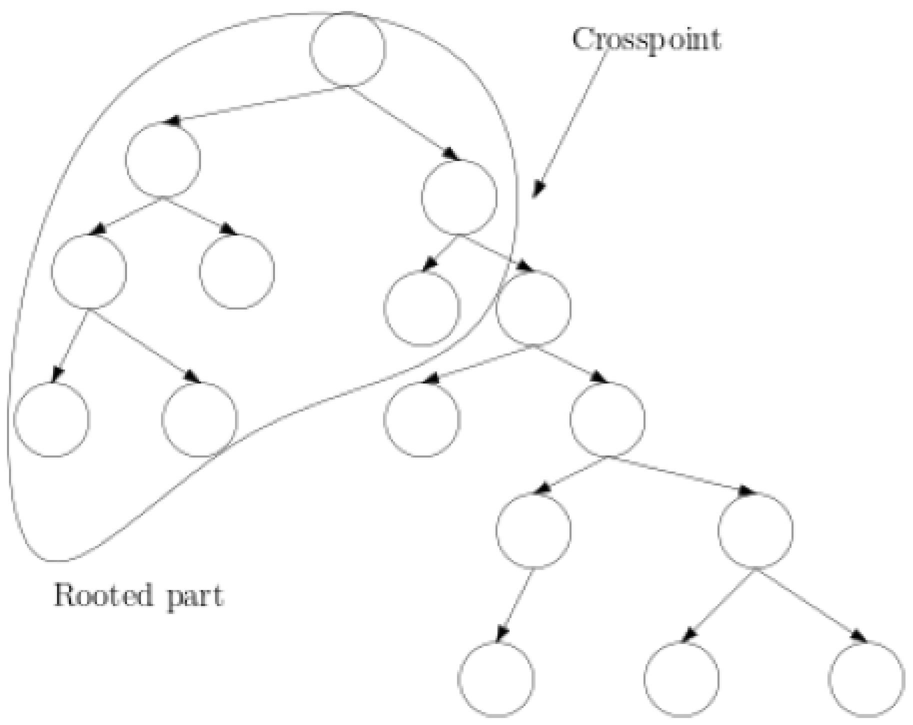

Figure 3.3: The cross-point in a tree, as illustrated in [6].

The first criterion is the mating restrictions between population members because breeding methods can exhibit different levels of restrictions on mating among individuals in the population. Based on Reynolds, [156], and Syswerda, [157], there are two different categories. The first one is that breeding can be either generational or steady. In generational breeding, generations do not overlap. Offspring are generated in a separate intermediate population, entirely replacing the old population. In contrast, steady state breeding involves a continuous influx of new offspring into the existing population. New offspring can immediately become parents to generate new individuals. To maintain a constant population size in steady state breeding, an existing member is usually replaced by each new offspring. Continuing with the first criterion, based on Goldberg, [158], and Langdon, [153], breeding can be either panmictic or be under some form of restricted mating. In panmictic breeding, any individual from the population is allowed to mate with any other individual from the popu-

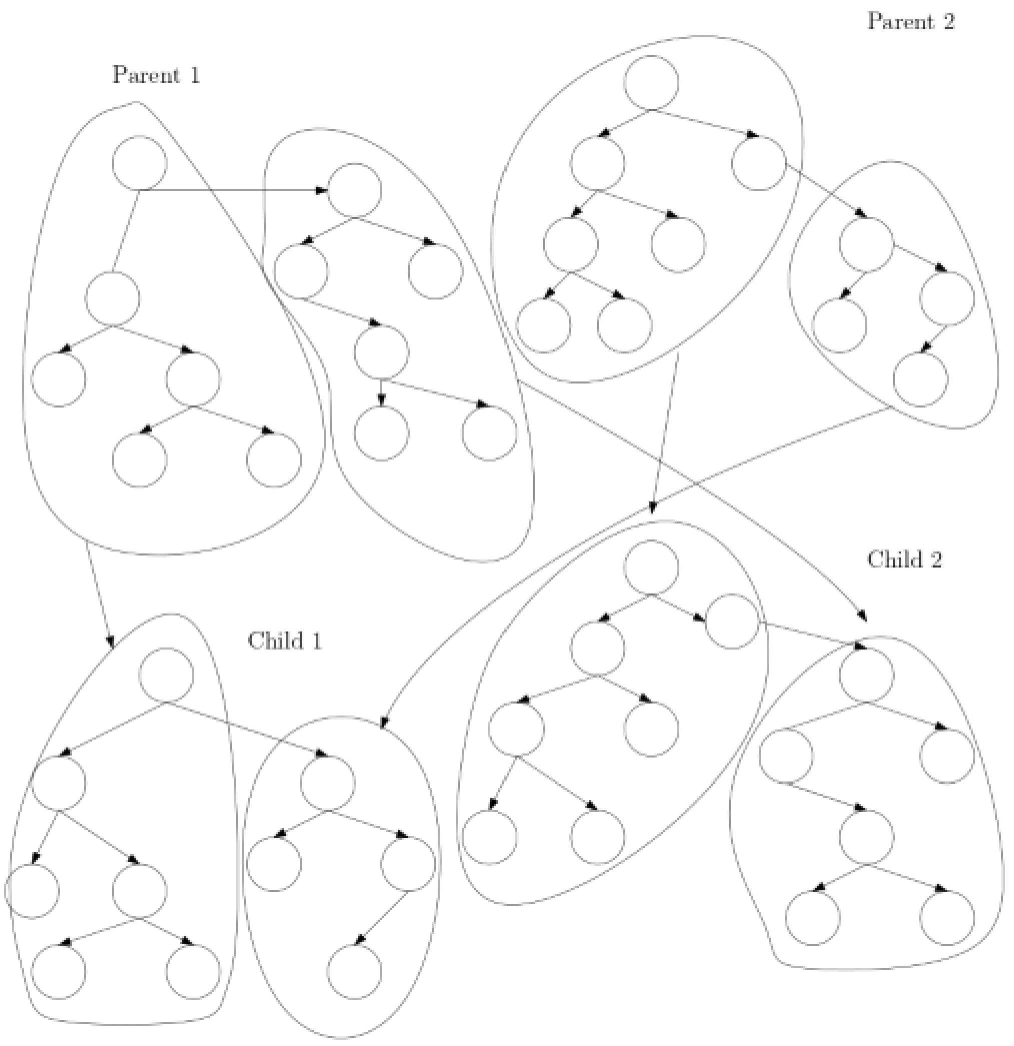

Figure 3.4: The crossover operator between two parents, as illustrated in [6].

lation. Contradictory, in restricted mating there are limitations on mating choices. Various strategies fall under restricted mating, such as Demic Sub-Populations, where the population is divided into demes or sub-populations, and most recombination occurs within these demes. Baseon on Langdon, [159], this promotes local exploration.

Expanding on the previous methods of the first criterion, another technique is of Ryan, [160], introducing pygmy algorithms, in which the population is partitioned into two sub-populations, and crossover is only permitted between individuals from different sub-populations.

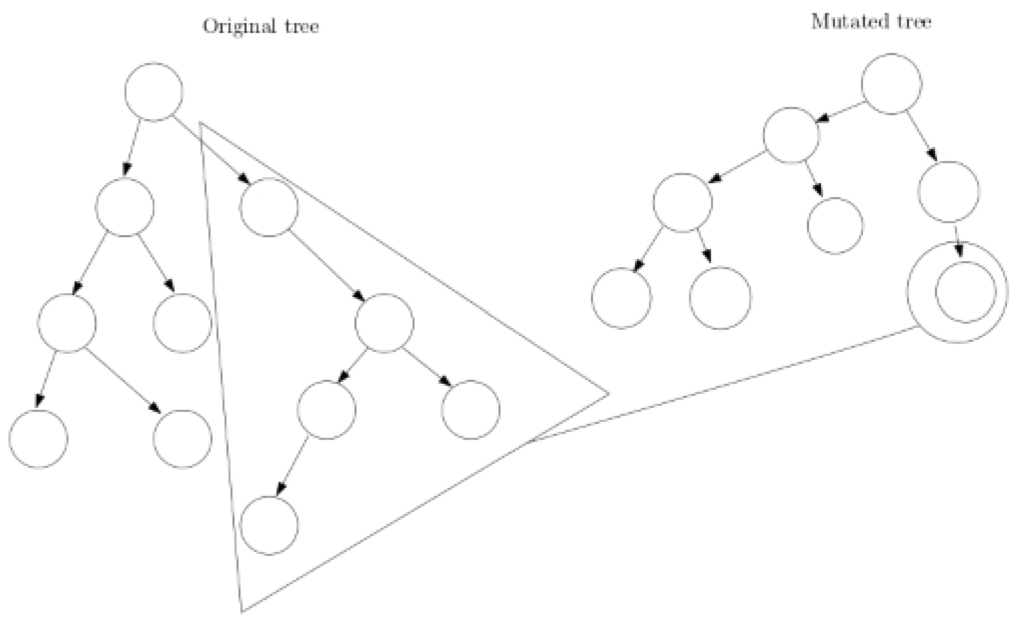

Figure 3.5: The mutation operator, as illustrated in [6].

This strategy encourages diversity by preventing excessive crossover within sub-populations. Moreover, the Genetic Lineage Strategies are introduced in McPhee et al, [161], in which parents are selected based on their genetic lineage, which refers to their evolutionary history. This approach emphasises retaining successful genetic lineages. More different breeding strategies can be studied in Luke and Spector, [162], where the authors created three different breeding strategies: the "Clones", the "Free", and the "Restricted".

The second criterion is the use of different types of operators, which are used depending on some probability settings, as seen in Poli et al, [133]. This happens to balance exploration and exploitation of the solution space. By adjusting the probabilities of applying various operators, GP algorithms can tailor their behaviour to the problem at hand and the specific goals of the optimisation process. Genetic operators, such as crossover and mutation, are applied to individuals to create offspring. The distinction here is whether an operator involves a single parent (as in mutation) or a set of parents (as in crossover). Different operator setups can be employed based on probability settings. In Koza's work, [134], the author introduces one widely used breeding method, the crossover, extended with mutation. This approach involves selecting two parents for crossover, applying crossover to produce offspring, and then applying mutation to the offspring. The offspring inherit genetic material from both parents through crossover, and mutation introduces small-scale variations. This method balances the exploration of new solutions (crossover) and fine-tuning existing solutions (mutation). A work that introduces an alternative to the traditional mutation operator is of Kocsis and Swan, [163], where the authors employ a deterministic proof search in the sequent calculus to yield semantics-preserving transformations on algebraic data types.

This criterion can also be applied to operator selection (the algorithm controls the probabilities of these operators so that it can emphasise different types of changes to the genetic material), elitism and reproduction (preserving a random individual from one generation to the next), and for adaptive probabilities (the probabilities of using different operators can be adjusted dynamically as the algorithm progresses, to monitor the algorithm's performance and adjust operator probabilities based on how well the current population is evolving).

These breeding methods can be combined and adapted to suit specific problem domains and research objectives in GP. The choice of breeding method and operator setup significantly influences the exploration-exploitation trade-off in the evolutionary process.

3.5.1 Combination of Subtree Crossover and Point Mutation

A commonly employed breeding strategy in genetic programming is the crossover operator as seen in Koza's work, [134], which is often paired with mutation. Even though John Koza's initial experiments in 1992 did not incorporate mutation, subsequent research has demonstrated the advantages of introducing a small amount of mutation. This can be particularly helpful in rejuvenating the genetic diversity within the population, based on Agapitos, [155]. Therefore, it has become a prevalent practice to combine crossover and mutation techniques, where the likelihood of applying crossover is set relatively high, for instance, around 90%, while the remaining portion (usually around 10%) is allocated to the probability of mutation.

3.6 Evaluation of the individuals

The evaluation process in genetic programming constitutes an important step for assessing the quality and efficacy of evolved solutions, as seen by Koza, [134]. This process involves subjecting candidate solutions, represented as individuals within the population, to a battery of fitness criteria, as seen in the workd of Banzhaf et al, [141]. The specific criteria are tailored to the problem domain and the overarching objectives of the evolutionary process.

As written by Poli et al, [133], at the heart of the evaluation process lies the fitness function, which quantitatively assesses the calibre of a candidate solution. This function assigns a numerical value that signifies how well an individual addresses the problem's requirements. Notably, the fitness values assigned to individuals profoundly influence the selection process, steering the algorithm's focus towards individuals with higher fitness scores for further reproduction and evolutionary progression. The formulation of the fitness function is a pivotal consideration, substantially impacting the GP algorithm's performance and the solutions it yields. Designing an effective fitness function necessitates deep comprehending the problem domain, objectives, and any pertinent constraints.

Several key factors need to be taken into consideration in the creation of a fitness function, such as the problem-relevant metrics, balancing trade-offs, mitigating premature convergence, accounting for noise and robustness, and efficient evaluation. Firstly, problem-relevant metrics should be used as part of the fitness function to emphasise the problem at hand. For instance, in classification tasks, metrics like accuracy, precision, recall, or the F1-score could hold relevance. Following, certain problems may entail trade-offs between multiple objectives. The fitness function must find an equilibrium between these competing objectives, reflecting their relative significance. Additionally, it is important to avert premature convergence, where the algorithm converges to sub-optimal solutions. Thus, the fitness function should be devised to discourage such outcomes. Finally, given the computational expense of evaluating individuals, particularly for intricate problems, the fitness function should find a balance between accuracy and computational efficiency [133].

Notably, the efficacy of the evaluation process significantly shapes the outcomes of the GP algorithm. Since GP is employed across diverse domains, the evaluation process lacks a one-size-fits-all approach, with researchers and practitioners customising the process to suit the idiosyncrasies of their specific challenges [141].

3.7 Selection Strategy

As written by Banzhaf et al, [141], the selection process within genetic programming holds a pivotal role in determining the individuals to whom genetic operators are applied. This process centres around evaluating the fitness of each program in comparison to their counterparts within the population . This important step essentially decides which individuals will contribute to the next generation's genetic makeup, which is critical in shaping the population's genetic makeup and driving the evolutionary progression.

Several selection methods are commonly employed in GP, each contributing to the algorithm's dynamics and outcomes. Two prominent selection strategies normally employed in GP are roulette wheel selection and tournament selection. The roulette wheel selection hinges on the principle that an individual's probability of selection is directly proportional to its fitness value. This mechanism mirrors the sections of a roulette wheel being sized according to the chances of landing on each respective section [141]. This method is particularly effective for emphasising the influence of higher-fitness individuals while still allowing lesser-fitness individuals to be chosen. In Koza, [134], we find the concept of tournament selection, which is the second widely embraced strategy we employ in our study. It involves assembling a subset of the population based on a predefined tournament size. Within this subset, individuals engage in competitive evaluations. The individual with the highest fitness value within the tournament emerges as the victor and is subsequently selected . Worth noting is that this method does not involve direct reproduction, but it does incorporate elitism. This implies that the most outstanding individual from each generation is directly duplicated into the subsequent generation. This helps maintain the strongest solutions in the population across generations and aids in preserving the overall quality of the evolving population. Furthermore, this strategy introduces an exploration element by allowing candidates of varying fitness levels to compete for selection. More specifically, larger tournament sizes may increase exploration, while smaller sizes may increase exploitation. Overall, tournament selection is simple and effective in maintaining diversity while favouring higher-fitness individuals.

Based on Langdon, [153], the choice of selection method profoundly influences the convergence rate and the diversity of the evolving population. Tournament selection and proportional selection, among others, strike different balances between exploration (diversification) and exploitation (convergence). Selecting the most appropriate method depends on the problem characteristics, the desired convergence rate, and the level of exploration required. In their experiments, Sokolov et al, [164], concluded that the variance of a selection method significantly impacts genetic algorithm performance. Unbiased tournament selection is a technique that reduces this variance and improves algorithm results, particularly for small population sizes and tournament sizes of 2.

The tournament selection method, which, as said, has been adopted as the preferred selection strategy in this thesis, aligns with the overarching goal of selecting candidates with superior fitness values to steer the evolutionary process toward improved solutions. Tournament selection ensures a balanced synergy between exploration and exploitation, and its effectiveness is showcased by its widespread application across diverse problem domains.

3.8 Generations

As seen in Branzhaf et al, [141], the process in GP forms the core of the evolutionary algorithm, driving the gradual refinement and improvement of candidate solutions across successive generations. It is of great significance due to its central role in achieving optimisation and problem-solving objectives. This is achieved by search and exploration, where the iteration process enables a systematic search through the solution space. By repeatedly generating new candidate solutions through crossover and mutation, GP explores a wide range of potential solutions. Moreover, it allows the evolving population to adapt to the problem's requirements and changing conditions. Over successive generations, genetic material is refined and diversified, leading to solutions that are better suited to addressing the problem at hand, and the algorithm progresses toward improved results. By selecting individuals with higher fitness values as parents, the algorithm guides the population towards solutions that possess desirable traits. The process aids in escaping local optima that might otherwise trap an algorithm, as the genetic diversity can help the algorithm "escape" stagnation points and explore new regions of the solution space.

Overall, the iterating from generation to generation is well-suited to navigating problem spaces and finding satisfactory solutions, allowing for the incorporation of domain-specific knowledge and striking a balance between exploration (searching for new solutions) and exploitation (refining known solutions).

3.9 Termination

The termination criterion in genetic programming is the stopping point for an algorithm's iteration process. It determines when the evolutionary process should conclude, indicating that the algorithm has either sufficiently converged to a satisfactory solution or has reached a point where further iterations are unlikely to yield substantial improvements. The choice of termination criterion depends on the quality of the solution, which is based on achieving a specific level of solution quality, where the algorithm halts once a solution surpasses a predetermined fitness threshold or meets a desired objective, in which case domain-specific insights guide the termination criteria. Similarly, it can be based on convergence criteria, where the Termination is triggered when the population's fitness values stabilize or show minimal improvement over several generations, indicating that further iterations are unlikely to lead to significant enhancements. In these cases, early stopping can be applied, where the algorithm may halt if there is a lack of progress or improvement in solution quality over a defined number of generations, preventing unnecessary computational efforts. In some instances, if GP continues to evolve for too many generations, it might overfit the training data, reducing its ability to generalize to new, unseen data. Early stopping helps prevent this by terminating the algorithm before overfitting occurs. Since computational resources may play a role, too, Termination might occur when a specified time limit or computational budget is reached, which is particularly important for resource-intensive problems. Following, a common termination criterion is applying an iteration limit, where the algorithm halts after a certain number of generations, allowing a limited number of iterations to refine solutions.

Selecting an appropriate termination criterion involves a trade-off between achieving optimal solutions and computational efficiency. Terminating too early might lead to suboptimal solutions, while prolonging the process unnecessarily might waste computational resources. In this thesis, the termination criterion is the reach of a specific number of independent runs, meaning we applied an interaction limit, which was tuned as part of the parameter tuning (Section 4.3, Chapter 4) and found out to be 50 runs.

3.10 Data Types in Genetic Programming

Continuing with Banzhaf et al, [141], data types are an important concept in genetic programming . Like in traditional programming, data types define the values that can be represented and manipulated within the GP framework. They determine the kind of operations that can be performed on individuals and play a crucial role in ensuring the correctness and functionality of evolved solutions. However, in GP, the concept of data types is often more flexible and dynamic compared to traditional programming languages.

In traditional programming, data types like integers, floating-point numbers, characters, and strings are well-defined and fixed. In contrast, GP allows for the evolution of programs where the data types are not predefined but emerge from the evolutionary process itself, as seen in Koza's work, [134]. As the author states, in order for GP to be applied to a specific problem, the concept of closure must be satisfied. This means that the arguments for functions and the values returned from these functions must be of the same data type. In this way, all functions can deal with any values that they receive as input. As a result, early works in this area required only one type of function to generate trees. For instance, all functions should only return and accept boolean values.

In order to address the issue of closure, Koza, [134], suggested using constrained syntactic structures. These syntactic rules describe the terminals and non-terminals that can be used as the children nodes of each non-terminal. The author then applied these syntactic constraints during tree initialisation and while genetic operations took place. Similarly, structure-preserving crossovers make sure that any crossover between trees is legal. For instance, once the point of crossover from the first parent is chosen, the crossover point from the second parent is randomly chosen among points of the same type. In this way, subtrees that are going to be exchanged are going to be of the same type, and thus, any offspring derived from this crossover will be valid.

Soon after, more research took place in order to introduce further constraints in the data types to improve the performance of GP. Montana, [165], extended Koza's idea of John Koza by proposing the strong typed genetic programming (STGP). As Montana states in the study, John Koza's GP and STGP are similar approaches, with the difference that the latter does not need to specify directly which children each non-terminal can have. Instead, this is done indirectly in STGP by specifying the data types of the arguments of each non-terminal and the data types returned by terminals and non-terminals. This flexibility meant that functions could operate on any input received, where arguments and return values had the same data type. Early GP implementations often limited themselves to a single data type, like boolean values.

Strongly-typed genetic programming algorithms' key distinction lies in its ability to enforce data type constraints more explicitly and systematically, ensuring that only valid data types can interact within a program. This approach enhances the robustness of evolved solutions by preventing incompatible data types from causing errors. Additionally, STGP makes designing GP systems for complex problem domains with intricate data type requirements easier, ensuring that evolved programs adhere to these requirements. Overall, STGP strikes a balance between GP's evolutionary flexibility and the need for structured data typing, making it a valuable tool in various applications of GP.

For data types in grammar-based GP, Espada et al, [166], present a novel approach to improve embedding grammars within the target programming language using object-oriented types. This enhances development through type-checking, integration with tools, and better ergonomics. Introducing "Meta-Handlers" further extends the algorithm's expressive power by allowing controlled value generation.

In this thesis, we use both GP and STGP to generate trading strategies in algorithmic trading.

3.11 Grammar-based Genetic Programming

Grammar-based genetic programming (GGP) is a variant of genetic programming that employs formal grammars to guide the generation and evolution of program structures. In GGP, the syntax and structure of programs are defined using a context-free grammar, which specifies the valid combinations of functions, terminals, and their arrangement within the evolving solutions, as seen in Aneline and Kinnear, [167], Whigham et al, [168], Backus et al, [169]. This approach adds structure and constraints to the genetic programming process, allowing for more controlled and structured evolution.

In GGP, the grammar defines the syntax rules that dictate how various elements (functions and terminals) can be combined to form valid solutions. The grammar serves as a blueprint for generating individuals and ensures that they adhere to syntactic correctness. Furthermore, GGP enforces a hierarchical structure in evolved programs, ensuring that the resulting solutions are well-structured and semantically meaningful. This contrasts with traditional GP, where the structure of individuals is determined solely by *tree-based* representations. Grammar rules specify which functions or terminals can be used at different positions within the program. This enables GGP to generate solutions with predefined patterns or structures, which can be particularly advantageous in domains where specific structures are important. By constraining the space of possible solutions through grammatical rules, GGP can guide the search towards more relevant and meaningful solutions. This controlled exploration can lead to faster convergence and improved solutions. Following, it can be customised for specific problem domains by designing grammars tailored to the problem's characteristics. This allows for the incorporation of domain-specific knowledge into the search process. In GGP, the grammar itself can also evolve over the generations. Genetic operators such as crossover and mutation can be applied to the grammar rules, allowing the evolution of both the program structures and the grammar that defines them.

As seen in McKay et al, [170], Grammar-based genetic programming has been applied to a variety of domains, including symbolic regression, program synthesis, image processing, and more. It is particularly useful in domains where grammar rules can naturally capture the problem structure. However, the choice of grammar in GGP can influence the types of solutions generated. While grammar provides advantages like closure and controlled exploration, it can also limit the diversity of solutions.

Strongly-typed genetic programming can be considered a part of grammar-based genetic programming (GGP) since Both STGP and GGP involve the use of formal grammar to guide the evolution of programs. However, they emphasize different aspects of the programming process. The strongly-typed architecture can be seen as a specific instance of GGP where the grammar is designed to enforce strong type checking, ensuring that only operations compatible with specific data types are allowed.

3.12 Chapter Summary

In this chapter, we introduced the concept of genetic programming algorithms, a technique that serves as the foundation for all chapters in this thesis. Our exploration commenced with a comprehensive overview of the algorithm's core principles, exploring specific facets of this set of algorithms. The information outlined specific topics within the GP framework, including representations, operators, breeding methods, selection strategies, data types, and grammar. Each area contributes distinct insights to the broader understanding of how GP operates and evolves solutions to complex problems. This overview of genetic programming algorithms provides the necessary groundwork for the subsequent research chapters, where we build upon these concepts to investigate and propose innovative solutions in the financial domain.

Part II

Thesis Contributions

Chapter 4

Combination of the financial analysis types

4.1 Chapter motivation

In this chapter, we will investigate the outcomes generated by genetic programming algorithms that utilise three distinct financial analysis types: fundamental, sentiment, and technical analysis. Considering the efficacy of each individual analysis type in identifying profitable trading strategies from the literature review in Chapter 2, our motivation behind this chapter lies in exploring the potential enhancement of financial profitability that can be achieved by combining their respective indicators. Thus, our underlying objective is to demonstrate the financial benefits arising from the fusion of these three analysis types, thereby establishing the foundation of our research.

Hence, we propose a novel genetic programming (GP) algorithm that integrates the aforementioned three analysis types. We exhibit the advantages resulting from their combination, as measured by three key financial metrics: the Sharpe ratio, rate of return, and risk. Our experimental investigations utilise information from 42 companies, and we present the results in the following sections.

The chapter's structure is as follows: Section 4.2 introduces the methodological framework employed in our study, including model representation, genetic programming operators, the trading strategy of the algorithms, and the fitness function. Section 4.3 presents the experimental setup of the research, wherein we introduce the data and its extraction, the benchmarking algorithms, and the parameter tuning. Following, Section 4.4 presents the empirical findings and analysis, offering an evaluation of the GP algorithms' performance and an assessment of the efficacy of combining the three financial analysis types. Finally, Section 4.5 summarises the research findings, the contribution of this chapter and the motivation for future research.

4.2 Methodology

GP algorithms have been widely adopted in various financial tasks, including algorithmic trading [92]. In this chapter, we propose a novel genetic programming algorithm referred to as GP-FASATA. In this nomenclature, "FA" stands for fundamental analysis, "SA" denotes sentiment analysis, while "TA" signifies technical analysis.

For the remainder of this section, we will first present the model representation in Section 4.2.1, followed by introducing the GP operators used in our experiments in Section 4.2.2, and an explanation of the trading process in Section 4.2.3. Finally, we will discuss the fitness function of the proposed GP algorithm, along with the metrics utilised to evaluate the GP individuals in Section 4.2.4.

4.2.1 Model representation

When constructing the trees used in this study to represent individuals, we employ logical functions such as `AND`, `OR`, Greater than (`GT`), and Less than (`LT`) for the internal nodes, as shown in the first part of Table 4.1. The terminal set, detailed in the second part of Table 4.1, incorporates indicators from three analysis types, along with an Ephemeral Random Constant (ERC). The ERC generates random real values within the range of -1 to 1 and acts as a threshold for these indicators. Specifically, the algorithm evaluates whether an indicator's value exceeds (or falls below) this threshold, thereby contributing to the maximisation of the Sharpe ratio.

The illustration of a sample evolved GP tree in Part 1 of Figure 4.1 depicts the root node as `AND`, with the initial branch starting from the `GT` node, indicating "greater than," connected to the SA indicator and the ERC. The second branch of the tree includes the `OR` node followed by the `LT` node, indicating "less than," in both parts. These nodes are associated with the FA and TA indicators, along with their corresponding ERC values.

Each GP individual undergoing evolution is incorporated into another tree, wherein an If-Then-Else (ITE) statement functions as the root. The first branch of this composite tree represents the evolved GP tree (Part 1). The second and third branches correspond to the buy (1) and hold (0) actions, respectively, which will be further explained in Section 4.2.3.

Notably, Part 2 of Figure 4.1 remains unaltered throughout the evolutionary process, rendering its inclusion within the GP algorithm unnecessary.

Table 4.1: Function and Terminal sets

Function set	
GP-FASATA root node	AND, OR
GP-FASATA function nodes	AND, OR, LT, GT
Terminal set	
FA	Net Profit ratio, Return on Equity, Quick ratio, P/E ratio, P/B ratio, P/S ratio Debt to Equity, Total Revenues, EBITDA, Levered free Cash Flow, Diluted EPS, R&D Expenses, ERC
SA-textBlob	TEXTpol, TEXTsub, TITLEpol,TITLEsub SUMMpol,SUMMsub
SA-SentiWordNet	TEXTsenti, TITLEsenti, SUMMsenti
SA-AFINN	TEXTafinn, TITLEafinn, SUMMafinn ERC
TA (for 5 and 10 days)	Moving Average, Momentum, ROC Williams' %R, Volatility, Midprice, ERC

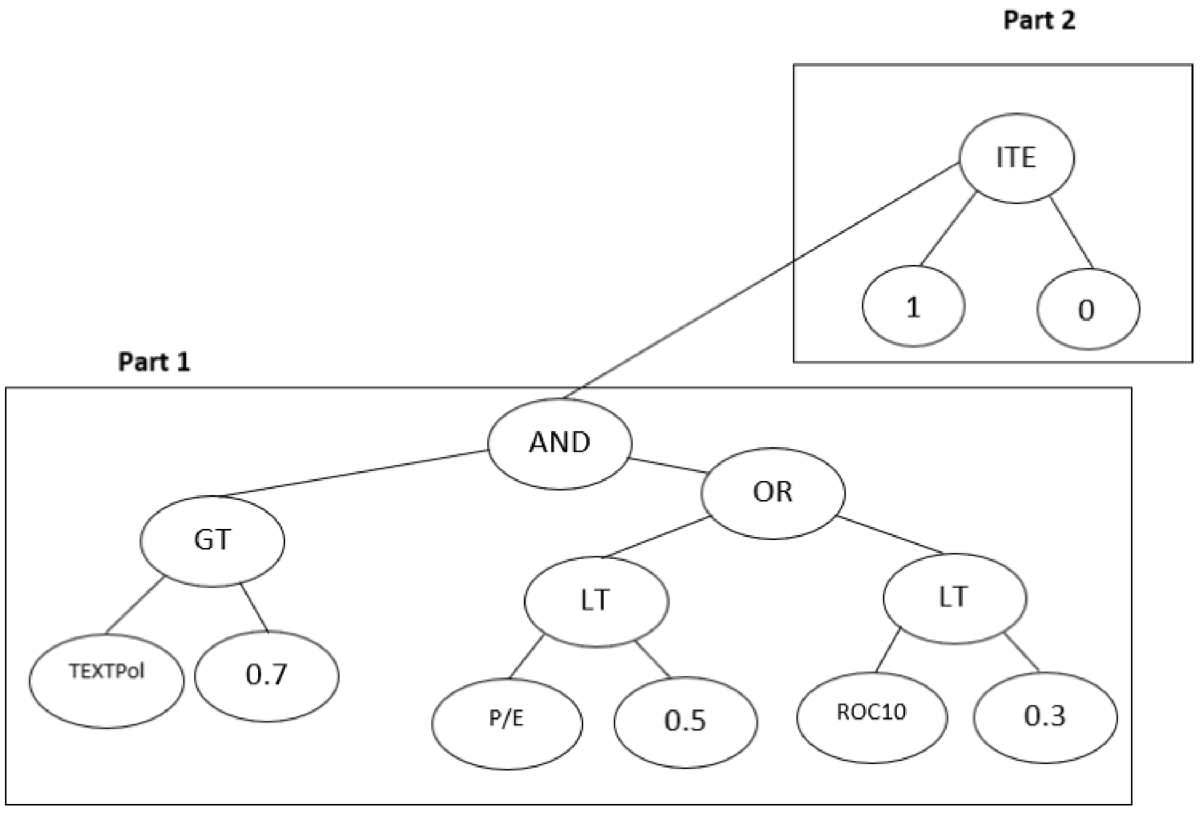

Figure 4.1: The depicted individual is a sample GP tree highlighting the combination of FA, SA, and TA indicators, showcasing a tree generated by the GP-FASATA algorithm. More specifically, there is an AND statement that, at its root, requires both branches of Part 1 to be TRUE for a trade to occur. This means the tree checks if the first branch with the SA indicators, TEXTpol, is greater than 0.7 and if the second branch with the FA and TA indicators, P/E and ROC over a 10-day period, are less than the ERC of 0.5 and 0.3, respectively.

4.2.2 GP operators

To remind the reader, in our research, the chosen individual undergoes sub-tree crossover with a probability denoted p, while point mutation is applied with a probability of $1 - p$. Additionally, we incorporate elitism, a strategy whereby the best individual from each generation is preserved and directly copied to the next generation. This ensures that the fittest individual continues contributing to the evolving population in subsequent generations. These operators were selected due to their popularity and results in genetic programming research. More information can be found in Chapter 3.

4.2.3 Trading algorithm

When the GP tree (Part 1) produces a TRUE value, it corresponds to the second branch (1) of the ITE tree, signalling a buy recommendation. Consequently, the STGP algorithm initiates the purchase of the stock. To determine the optimal selling time, the algorithm applies a predefined rule based on a fixed reference increase rate r and a specified time period of d days. If the stock price during the next d days yields a return that exceeds the threshold r, the stock is sold on that particular day. However, if the return does not surpass r within d days, the stock is sold at the end of the specified period. For instance, with $d = 30$ and $r = 0.05$, the rule can be interpreted as follows: "If the stock price increases by more than 5% of the buying price within the next 30 days, sell the stock on that day. Otherwise, sell the stock at the end of the 30-day period."

It should be noted that the parameters d and r are fine-tuned during the validation phase and remain constant for all GP algorithms considered. However, these parameters can vary across different companies, as discussed in Section 4.3.3.

We maintain a list of returns to evaluate the trading performance, as indicated by Equation 4.1. From this record, we calculate essential metrics such as the Sharpe ratio (Equation 4.2), rate of return, and risk. The trading algorithm's effectiveness assessment is based on these metrics, and their results are compared accordingly. Further details and the presentation of these results can be found in Section 4.4.

4.2.4 Fitness function and Metrics

In the subsequent sections, we introduce the metrics of rate of return, risk, and the Sharpe ratio, which are defined as follows:

The returns denoted R, measure the profitability of a trade as a percentage of the amount invested. The calculation of profit considers the transaction cost, which amounts to 0.025% of the selling price (c_t). Specifically, the return of a trade is determined using Equation (4.1), where V_f represents the final value or the price at which the stock was sold, and V_i represents the initial value or the price at which the stock was bought.

$$R = \frac{(1 - c_t)V_f - V_i}{V_i}. \tag{4.1}$$

The rate of return denoted RoR, represents the sample mean of the returns obtained from all trades during a specific time period under consideration. The risk is quantified as the standard deviation of the returns, calculated as the square root of the variance of R, denoted as $\sqrt{var[R]}$.

The Sharpe ratio denoted S_a, is a metric that compares the expected value of the excess return over the risk-free return, R_f, to the level of risk. It is formally defined as follows:

$$S_a = \frac{\mathrm{E}[R - R_f]}{\sqrt{var[R]}}, \tag{4.2}$$

,where $E[R - R_f]$ represents the expected value of the excess return, and $\sqrt{var[R]}$ denotes the risk, as previously defined. The Sharpe ratio measures the risk-adjusted return and provides insights into the performance of a trading strategy by considering both the expected return and the associated risk.

The *fitness function* employed in the GP-FASATA algorithm is defined as the maximisation of the Sharpe ratio. The Sharpe ratio, a well-known financial concept, assesses the return an investment strategy achieves in relation to its associated risk. By considering the risk of a particular stock or company, the Sharpe ratio helps investors make informed decisions by determining whether the potential return justifies the level of risk involved. As such, it maximises the Sharpe ratio as the fitness function aims to identify trading strategies that offer favourable risk-adjusted returns, thereby assisting in the selection of potentially profitable investment approaches.

4.3 Experimental Setup

This section will remind the reader of the experimental setup steps of data collection in Section 4.3.1. We will introduce the GP benchmark algorithms, the four machine learning benchmarks and one financial strategy in Section 4.3.2, and we will present the parameter tuning process in Section 4.3.3.

4.3.1 Data

In our research, we conducted an analysis on a dataset comprising 42 international companies listed on various stock exchanges. The dataset incorporated information from 10K-filings, historical stock prices, and relevant news articles. The research period we spanned 5 years, commencing on January 1st, 2015, and concluding on January 31st, 2020. Notably, the period excluded the COVID-19 pandemic to maintain consistency between the training/validation sets and the test set, thereby ensuring reliable parameter tuning.

To gather fundamental analysis data, we obtained 10K-filings from Seeking Alpha. We retrieved daily closing price data from Yahoo! Finance for technical analysis. Sentiment analysis was performed by utilising a Python scraper in conjunction with the Google Search Console API to scrape articles, titles, and summaries. Once all the necessary data was collected, we generated a total of 36 relevant indicators, with 12 indicators corresponding to each analysis type, as outlined in Section 4.2.1. Subsequently, we partitioned the dataset for the 42 companies into three sequential parts: 60% for training, 20% for validation, and 20% for testing purposes.

4.3.2 Benchmarks

The proposed GP-FASATA algorithm is benchmarked against three other GP algorithms employed in this study:

- **GP-FA**: A GP algorithm that only includes fundamental analysis indicators in its terminal set. This benchmark aims to evaluate the trading performance of a GP algorithm that solely uses fundamental analysis indicators.
- **GP-SA**: A GP algorithm that only includes sentiment analysis indicators in its terminal

set. Similar to the previous benchmark, the motivation here is to evaluate the performance of a GP algorithm using only sentiment analysis indicators.

- **GP-TA**: A GP algorithm that only includes technical analysis indicators in its terminal set. Similar to the previous benchmarks, the motivation here is to assess the performance of a GP algorithm using only technical analysis indicators.

Additionally, the study includes four additional algorithmic benchmarks:

- **Multilayer perceptron (MLP)**
- **Support vector machine (SVM)**
- **eXtreme Gradient Boosting (XGBoost)**
- **Long short-term memory (LSTM)**

All of the machine learning benchmark algorithms were applied to address a binary classification problem: predicting whether the stock price will increase by a certain percentage ($r\%$) within the next d days. A Class 1 prediction indicates a buy action, while a Class 0 prediction represents a hold action. As mentioned in Section 4.2.3, the sell action is executed as part of the trading strategy.

Finally, the proposed GP-FASATA is evaluated against the following financial benchmark:

- **Trading-Strategy**d, r (**TS**d, r): Buy at the beginning of every trading period. Sell when the price increases by more than the rate of reference r or after d days have passed, whichever happens sooner.

The TS$_{d,r}$ benchmark is a baseline trading strategy without the learning component. It allows us to examine the added value of our GP algorithm when it is separated from the pure trading element of the strategy.

4.3.3 Parameter tuning

A two-step grid search conducted on the validation set determined the optimal GP parameters for the proposed algorithm and the other GP variants. The grid search involved adjusting several parameters, including the population size, number of generations, tournament size, maximum depth of the trees, and the crossover probability (p)[1]. During the tuning process, the trading parameters, specifically d (number of days) and r (percentage increase), were kept constant at 30 and 0.05, respectively. This was done to reduce the time required for parameter tuning. After conducting the grid search, a set of parameters performed equally well for all GP variants on the validation set was identified without observing any statistical differences. These parameters were then used in all runs for all GP algorithms and companies. The specific GP parameters used in the experiments are provided in Table 4.2. These parameters remained consistent across all runs for all GP algorithms and companies.

[1]Since the mutation probability can be calculated as 1-p, it was not required to include it as a separate parameter during the tuning process.

Table 4.2: GP Parameters for GP-FASATA

GP Parameters	
Population size	1000
Crossover probability	0.95
Mutation probability	0.05
Generations	50
Tournament size	4
Maximum tree depth	6

Table 4.3: Parameters d and r for each company.

Company	***Days (d)***	***Rate (r)***	***Company***	***Days (d)***	***Rate (r)***
AAPL	20	0.02	KERING	10	0.035
ADIDAS	5	0.03	KODAK	25	0.015
ALIBABA	20	0.025	MCDON	25	0.045
AMAZON	25	0.015	NESTLE	30	0.005
ASUS	20	0.025	NFLX	30	0.05
ATVI	10	0.01	NINTENDO	15	0.01
BLACKB	30	0.035	NYT	20	0.025
COKE	10	0.02	PANASONIC	15	0.02
EBAY	30	0.015	SAINSBURY	30	0.045
ESTEE	20	0.005	SHISEIDO	30	0.05
FORD	25	0.01	SUBARU	25	0.01
FUJIFILM	10	0.0.4	SUZUKI	20	0.05
FUJITSU	20	0.045	TENCENT	15	0.01
GM	30	0.03	TESCO	30	0.02
GOOGL	30	0.015	TESLA	20	0.035
HITACHI	15	0.05	TOYOTA	15	0.02
HONDA	30	0.01	UBISOFT	25	0.01
HSBC	5	0.005	WALMART	30	0.005
HYUNDAI	15	0.02	XEROX	10	0.05
IBM	25	0.035	YAMAHACO	30	0.005
INTC	15	0.01	YAMAHAMO	30	0.015

To enhance the trading performance, the parameters d and r of the trading strategy were adjusted independently for each company. These parameters were selected based on their overall performance across all GP algorithms, and this process was carried out using the validation set. Table 4.3 shows the d and r parameters for each company.

In addition to the GP algorithms, the machine learning benchmarks (MLP, SVM, XGBoost, and LSTM models) were also tuned separately using binary classification. The performance of each model was evaluated on the validation set, and the best model was selected based on its predictive ability. The predicted class from the selected model was then used as signals for the trading strategy, employing the same d and r parameters as the GP variants. The tuning process for these machine learning algorithms for trading was based on the methodology described in [93].

4.4 Results and Discussion

In this section, we present the results of our experiments, which involved comparing the performance of GP-FASATA with the benchmarks discussed in Section 4.3.2. To assess the effectiveness of GP-FASATA, we conducted 50 independent runs on the training set of each of the 42 companies for each algorithm. Each run represented a distinct trading strategy derived from the respective algorithm. Subsequently, these trading strategies were applied to the test set for further analysis and evaluation. Tables 4.4 and 4.5 showcase the mean Sharpe ratio values of the runs in each company, separated into the first and last 21 companies for increased readability. Moreover, the median Sharpe ratio values of all the runs across all companies can be found in Table 4.6. Similarly, the median values for rate of return and risk can be found in Sections 4.4.2 and 4.4.3, respectively.

To ensure the accuracy of our statistical analysis and avoid distortions, we implemented certain exclusion criteria for calculating the mean values presented in Tables 4.6, 4.10, and 4.14. Specifically, runs in which the GP algorithms did not execute any trades due to potential losses were not considered when calculating the mean values for the Sharpe ratio, rate of return, and risk. Additionally, runs with only one trade were excluded from the risk calculation, as a minimum of two trades is required. The tables provide each algorithm's mean, median, standard deviation, maximum, and minimum values across the 42 companies. However, certain algorithms did not generate any results for specific companies in some cases, resulting in rows with mean values of 0. These cases indicate that the corresponding algorithm did not perform trades for those companies.

We performed a two-sample Kolmogorov-Smirnov (KS) test on all runs across companies for each algorithm to validate our findings. At the same time, we excluded values of 0, which are runs that did not result in any trading action. We chose the KS test because it is sensitive to differences in the shape of the empirical cumulative distribution of two samples and reports the maximum difference arising between the samples' respective distributions.

To account for the issue of multiple comparisons, we applied the Holm-Bonferroni correction. This correction method helps control the overall Type I error rate when conducting multiple hypothesis tests. We conducted three separate comparisons for each financial metric, comparing the GP-FASATA algorithm with three different GP benchmarks. The Holm-Bonferroni correction involves determining the minimum p-value threshold for statistical significance at a 5% significance level, considering the number of comparisons being made. For each rank, which represents the magnitude order of the p-values (with 1 being the smallest and k being the largest), the minimum acceptable p-value threshold is calculated using the formula $\alpha(rank) = \frac{0.05}{k-rank+1}$. Here, the denominator k represents the total number of comparisons being performed. By applying the Holm-Bonferroni correction, we adjust the significance threshold for each comparison, ensuring that the overall Type I error rate is controlled within the desired range.

To assess the statistical significance of differences between two samples at a 5% significance level, we compared the p-values obtained from each comparison to the corresponding minimum acceptable p-value for its rank. In this part of the analysis, where we compare one control algorithm to 3 other GP variants, the denominator k has the value of 3. More specifically, the first-ranked p-value should be less than 0.0166, the second-ranked p-value should be less than 0.025, and the third-ranked p-value should be less than 0.05. This approach ensures that the probability of obtaining a false positive result is maintained below a predetermined threshold, resulting in more reliable statistical findings.

4.4.1 Sharpe ratio

Tables 4.4 and 4.5 present the median Sharpe ratio values over the 50 runs for the 42 companies. A zero (0) value for a company means that the relevant algorithm decided not to trade at all to avoid losses. As we can observe, the algorithm which has the highest occurrence with the best Sharpe ratio values is GP-SA with 17 companies, followed by GP-TA (9), GP-FA (13), whereas GP-FASATA (4) is last. However, it must be noted that the mean Sharpe ratio values are not that far away in some cases. It is also worth mentioning that GP-SA, GP-TA, and GP-FASATA have three 0 Sharpe ratio values, indicating that they are slightly more conservative algorithms than GP-FA. As mentioned above, this enables these algorithms to avoid losses by not trading at all in situations they consider to be unfavourable. As a result, this can positively affect their average and median performance, as we will see next.

Table 4.4: Median values for Sharpe ratio per company. Boldface is used to denote the best value for the particular dataset. First 21 companies.

Company	*GP-FA*	*GP-SA*	*GP-TA*	*GP-FASATA*
AAPL	**6.747**	0.35	5.29	5.06
ADIDAS	0.045	0.099	**0.256**	0.2109
ALIBABA	0.5242	**12.872**	0.4731	3.3684
AMAZON	1.179	0.155	**19.44**	1.457
ASUS	-1.375	0.3301	**0.5215**	-1.375
ATVI	0.3558	2.063	**3.506**	0.4423
BLACKB	**1.027**	-0.212	0.875	0.7
COKE	**13.07**	-0.581	0.615	1.214
EBAY	**0.670**	-0.142	0	-0.549
ESTEE	0.043	**2.05**	0.77	0.371
FORD	-0.123	**8.301**	-0.595	0.0989
FUJIFILM	**1.044**	0.0551	-1.472	-0.630
FUJITSU	0.7532	3.707	**14.624**	13.584
GM	-0.25	**0.858**	-20.91	-0.238
GOOGL	0.052	**0.0823**	0.0162	-0.11
HITACHI	**0.097**	0.355	0	**0.097**
HONDA	2.217	**9.044**	3.341	0.594
HSBC	-0.225	**3.939**	0.0372	0.0133
HYUNDAI	0.2059	0.7194	-0.6451	**2.5834**
IBM	0.5286	0.194	**7.216**	0.1768
INTC	0.6264	**1.5177**	1.31	0.1648

Table 4.5: Median values for Sharpe ratio per company. Boldface is used to denote the best value for the particular dataset. Last 21 companies.

Company	*GP-FA*	*GP-SA*	*GP-TA*	*GP-FASATA*
KERING	0.5836	0.9582	**0.9863**	0.3646
KODAK	0.9546	1.4971	**1.9477**	0.11
MCDON	-0.05	**0.0982**	-4.3045	0
NESTLE	-0.209	**2.3209**	-0.1383	-0.0065
NFLX	**8.0470**	0.0801	-0.2085	**8.0470**
NINTENDO	0.1078	**1.1455**	-0.4265	0.2703
NYT	0.3043	**3.4819**	0.1407	0.2134
PANASONIC	-0.004	**2.505**	0.9242	0.9076
SAINSBURY	**3.2401**	2.7915	1.246	1.0172
SHISEIDO	-0.7148	-0.1614	**0.5601**	-0.9181
SUBARU	-0.3436	-0.4698	**1.133**	0
SUZUKI	**7.353**	-0.783	-0.1657	0.804
TENCENT	0.2961	0	-0.554	**2.528**
TESCO	0.194	**0.9316**	0.35	-0.257
TESLA	1.6276	**3.106**	2.140	1.086
TOYOTA	**3.0495**	0.3721	0	-0.344
UBISOFT	**1.856**	0	0.8739	0.9205
WALMART	-0.0558	**0.6368**	0.3542	0.090
XEROX	**2.1182**	0.054	-0.8289	0.5039
YAMAHACO	0.6108	**1.4**	0.0588	1.095
YAMAHAMO	**0.7899**	0	0.5951	0

Table 4.6 presents the statistical measures of each algorithm's mean and median Sharpe ratio, along with the corresponding standard deviation, maximum, and minimum values. The table encompasses the results obtained from 50 independent runs conducted for each company, considering all eight GP algorithms.

Table 4.6: Summary statistics of Sharpe ratio across all 50 GP runs and all 42 companies. Boldface is used to denote the best value for each statistic.

Algorithm	*Average*	*Median*	*StDev*	*Maximum*	*Minimum*
GP-FA	2.18	0.67	10.37	83.59	-52.54
GP-SA	2.65	0.57	10.75	173.13	-14.16
GP-TA	2.32	**0.70**	**5.62**	31.47	-20.9
GP-FASATA	**2.99**	0.58	12.1	**206.25**	**-9.21**

When considering Sharpe ratio results, it is important to remember that this metric can be sensitive to the number of trades and can experience large values (either positive or negative) if very few trades are performed. This is because the risk, calculated as the standard deviation of returns experienced within a given period, can be extremely small if an algorithm performs very few trades (e.g. $2-4$ trades throughout the test set). Given that the risk is the denominator of the Sharpe ratio, a minimal decimal number of risks can lead to a very high value for the Sharpe ratio.

The results presented in Table 4.6 show that the proposed GP-FASATA algorithm exhibits the highest mean value for the Sharpe ratio. However, it is worth noting that it does not yield the highest median value among all the algorithms considered. The median represents the middle value when the Sharpe ratios are arranged in ascending order, providing a measure of central tendency less affected by extreme values. That being said, we do notice that the median values of the GP algorithms are not that far apart, either.

Regarding the variability of the algorithms' performances, GP-FASATA stands out with the highest standard deviation. A higher standard deviation indicates greater dispersion or variability across the runs in the Sharpe ratio values. Alongside GP-SA, GP-FASATA also exhibits the highest maximum values, indicating the presence of runs with exceptionally high Sharpe ratio values. Conversely, these algorithms also have the lowest minimum values, suggesting the existence of runs with relatively poor trading performance.

The KS test assesses the statistical differences between two samples, with the null hypothesis stating that the two samples are drawn from the same population. A p-value below the predetermined significance level indicates a rejection of the null hypothesis, suggesting that the two samples are statistically different. In our analysis, GP-FASATA is chosen as the control algorithm, as it exhibits the highest mean Sharpe ratio value. The KS test compares it pairwise with the other GP variants.

The results in Table 4.7 reveal that the distribution of the control algorithm GP-FASATA is statistically significant different than the distributions of the benchmark algorithms, as indicated by the significantly lower p-values (second column) compared to the corresponding significance level values (fourth column). These p-values are calculated by comparing the two samples and assessing whether they are drawn from the same population.

Table 4.7: Kolmogorov-Smirnov test p-values on Sharpe ratio of the proposed GP-FASATA algorithm against the 3 GP benchmarks. Statistical significance changes are based on the Holm-Bonferroni correction. Statistically significant differences at the 5% level are indicated in boldface.

Algorithm	*GP-FASATA p-values*	**Rank**	*Significance level*
GP-FA	**8.45e-07**	2	0.025
GP-SA	**1.10E-08**	1	0.05
GP-TA	**7.10E-06**	3	0.016

4.4.2 Rate of Return

Tables 4.8 and 4.9 showcase the median rate of return values over the 50 runs for the 42 companies. Again, a zero (0) value for a company means that the relevant algorithm decided not to trade to avoid losses. As we can see, the algorithm that has the highest occurrence with the best rate of return values is GP-FA with 14 companies, followed by GP-SA (13), GP-TA (10), and GP-FASATA (6) comes last. Again, it must be noted that in more cases than previously observed, the median rate of return values are close to each other. Again, GP-SA, GP-TA, and GP-FASATA have three 0 Sharpe ratio values, while GP-FA has zero 0 observations.

Table 4.8: Median values for rate of return per company. Boldface is used to denote the best value for the particular dataset. First 21 companies.

Company	***GP-FA***	***GP-SA***	***GP-TA***	***GP-FASATA***
AAPL	**0.0254**	0.0129	0.0240	0.0245
ADIDAS	0.0014	0.0017	**0.0107**	0.0083
ALIBABA	0.0185	0.0303	0.0158	**0.0326**
AMAZON	**0.0295**	0.0007	0.0173	0.0256
ASUS	-0.033	0.0096	**0.0190**	-0.018
ATVI	0.0081	0.0102	**0.0157**	0.0100
BLACKB	**0.0355**	-0.018	0.0346	0.0180
COKE	**0.0239**	-0.040	0.0077	0.00809
EBAY	**0.0088**	-0.004	0	-0.0284
ESTEE	0.0013	0.0098	**0.0107**	0.00746
FORD	-0.004	**0.0140**	-0.028	0.00289
FUJIFILM	**0.0317**	0.0308	-0.027	-0.0136
FUJITSU	0.0355	0.0645	**0.1018**	**0.10181**
GM	-0.013	**0.0171**	-0.089	-0.0124
GOOGL	0.0015	**0.0027**	-0.0004	-0.0022
HITACHI	0.0054	**0.0176**	0	0.0054
HONDA	**0.0175**	0.0114	0.01473	0.0120
HSBC	-0.0042	0.0125	0.00046	**0.0133**
HYUNDAI	0.0069	0.0124	-0.020	**0.0270**
IBM	0.01676	0.0132	**0.03612**	0.0111
INTC	0.0180	**0.0195**	0.0179	-0.0046

Table 4.9: Median values for rate of return per company. Boldface is used to denote the best value for the particular dataset. Last 21 companies.

Company	***GP-FA***	***GP-SA***	***GP-TA***	***GP-FASATA***
KERING	**0.0217**	0.0176	0.0163	0.0151
KODAK	0.0232	**0.0534**	0.03832	0.00576
MCDON	-0.00195	**0.00575**	-0.0003	0
NESTLE	-0.0049	**0.00820**	-0.0021	0.000113
NFLX	**0.06071**	0.0080	-0.02305	0.06071
NINTENDO	0.0035	**0.02509**	-0.03231	0.0153
NYT	0.0126	**0.0385**	0.006101	0.00910
PANASONIC	-2.24E-06	**0.0269**	0.01694	0.01822
SAINSBURY	**0.0491**	0.040347	0.01798	0.02039
SHISEIDO	-0.0448	-0.01658	**0.02077**	-0.0448
SUBARU	-0.0150	-0.02840	**0.0314**	0
SUZUKI	**0.05504**	-0.03080	-0.0133	0.0373
TENCENT	0.0079	0	-0.0543	**0.0182**
TESCO	0.00682	**0.0163**	0.01033	-0.0096
TESLA	0.0661	0.0527	**0.07098**	0.04592
TOYOTA	**0.01723**	0.01065	0	-0.0028
UBISOFT	0.0185	0	**0.0288**	0.02499
WALMART	-0.000805	**0.00447**	0.00363	**0.00447**
XEROX	**0.0327**	0.003	-0.0174	0.0320
YAMAHACO	0.0128	0.01209	0.0025	**0.0144**
YAMAHAMO	**0.0170**	0	0.01025	0

Table 4.10 presents the statistical summary of the rate of return for each algorithm, including the mean, median, standard deviation, maximum, and minimum values. These values are calculated based on 50 independent runs for each company across the eight GP algorithms.

Table 4.10: Summary statistics of rate of return across all 50 GP runs and all 42 companies. Boldface is used to denote the best value for each statistic.

Algorithm	*Average*	*Median*	*StDev*	*Maximum*	*Minimum*
GP-FA	0.014	**0.015**	0.038	0.069	-0.30
GP-SA	0.009	0.010	**0.028**	**0.10**	-0.12
GP-TA	0.013	0.014	0.033	**0.10**	**-0.089**
GP-FASATA	**0.015**	0.014	0.029	**0.10**	-0.10

The summary statistics in Table 4.10 indicate less variability in the performance of the algorithms when considering the rate of return metric. GP-FASATA, GP-FA, and GP-TA exhibit higher mean and median values, suggesting relatively better performance in generating returns. Additionally, GP-SA and GP-FASATA show lower standard deviation values, indicating less variability in the returns generated by these algorithms. The maximum rate of return is the same for GP-SA, GP-TA, and GP-FASATA, suggesting that these algorithms have the potential to achieve higher returns in certain cases. On the other hand, GP-TA has the lowest minimum rate of return, indicating a relatively lower downside risk than other algorithms.

For the KS tests, GP-FASATA is, again, the control algorithm for the rate of return results, showing the difference in the distributions of GP-FASATA and the distributions of the benchmarks. In contrast, the former's average and median performance values are close to GP-FASATA.

Table 4.11: Kolmogorov-Smirnov test p-values on rate of return of the proposed GP-FASATA algorithm against the 3 GP benchmarks. Statistical significance changes are based on the Holm-Bonferroni correction. Statistically significant differences at the 5% level are indicated in boldface.

Algorithm	*GP-FASATA p-values*	**Rank**	*Significance level*
GP-FA	**3.23E-10**	1	0.016
GP-SA	**1.10E-08**	3	0.05
GP-TA	**7.10E-06**	2	0.025

4.4.3 Risk

Tables 4.12 and 4.13 present the median risk values for all runs of the 42 companies. As we can observe, the algorithm with the highest occurrence in best risk values is GP-SA with 19 companies [2], GP-TA (12), followed by GP-FA (7), and GP-FASATA (6). However, we observe that the median risk values between the algorithms are similar and much closer than for the Sharpe ratio and rate of return.

[2]Zero (0) risk values, which indicate that an algorithm performed no trading, are not taken into account when comparing values to identify the best performing algorithm.

Table 4.12: Median values for risk per company. Boldface is used to denote the best value for the particular dataset. First 21 companies.

Company	*GP-FA*	*GP-SA*	*GP-TA*	*GP-FASATA*
AAPL	0.005697	0.03179	**0.00464**	0.00469
ADIDAS	0.02695	**0.01415**	0.02963	0.02471
ALIBABA	0.0350	**0.00404**	0.03700	0.01055
AMAZON	0.02484	0.02694	**0.00088**	0.01745
ASUS	0.02482	0.02310	0.03615	**0.01410**
ATVI	0.02240	**0.00359**	0.00825	0.02212
BLACKB	0.0326	0.08741	**0.02545**	**0.02545**
COKE	**0.00181**	0.07011	0.01291	0.00775
EBAY	**0.01286**	0.02923	0	0.05331
ESTEE	0.02691	**0.00452**	0.01466	0.01525
FORD	0.02799	**0.00166**	0.04818	0.02722
FUJIFILM	0.03016	0.05543	**0.01883**	0.02205
FUJITSU	0.0515	0.01736	**0.00697**	**0.00697**
GM	0.05325	0.01967	**0.00428**	0.05317
GOOGL	0.0253	**0.01223**	0.04068	0.02199
HITACHI	0.0533	**0.04914**	0	0.0533
HONDA	0.0080	**0.00123**	0.00539	0.01996
HSBC	0.01989	**0.00312**	0.00662	0
HYUNDAI	0.0325	0.02157	0.04458	**0.01387**
IBM	0.03129	0.06740	**0.00202**	0.06177
INTC	0.0285	**0.01275**	0.01479	0.06467

Table 4.13: Median values for risk per company. Boldface is used to denote the best value for the particular dataset. Last 21 companies.

Company	***GP-FA***	***GP-SA***	***GP-TA***	***GP-FASATA***
KERING	0.037776	**0.02704**	0.03209	0.03977
KODAK	0.023521	0.03556	**0.01956**	0.05659
MCDON	0.043232	0.04056	**0.02985**	0
NESTLE	0.024797	**0.00344**	0.01703	0.01620
NFLX	**0.007517**	0.07614	0.10521	0.00755
NINTENDO	0.030983	**0.03228**	0.03854	0.03330
NYT	0.040852	**0.01115**	0.04322	0.04118
PANASONIC	0.044524	**0.01086**	0.01809	0.02114
SAINSBURY	0.014848	0.02298	**0.01425**	0.01732
SHISEIDO	0.063066	0.10319	0.05848	**0.05783**
SUBARU	0.044433	0.06091	**0.02083**	0
SUZUKI	**0.00745**	0.08093	0.06445	0.04614
TENCENT	0.026062	0	0.09871	**0.00723**
TESCO	0.034039	**0.01888**	0.02890	0.04154
TESLA	0.040500	**0.01702**	0.03452	0.04737
TOYOTA	**0.01268**	0.01963	0	0.02756
UBISOFT	**0.01128**	0	0.03468	0.02992
WALMART	0.018379	**0.00927**	0.00949	0.02483
XEROX	0.023120	0.06069	**0.02131**	0.03093
YAMAHACO	0.020637	**0.00956**	0.04052	0.01841
YAMAHAMO	0.027458	0	**0.01588**	0

Table 4.14 presents the summary statistics for the risk values obtained from 50 independent runs for each company across the four GP algorithms. The table includes the mean, median, standard deviation, maximum, and minimum values.

Table 4.14: Summary statistics of risk across all 50 GP runs and all 42 companies. Boldface is used to denote the best value for each statistic.

Algorithm	***Average***	***Median***	***StDev***	***Maximum***	***Minimum***
GP-FA	0.029	0.026	**0.024**	0.21	0.0007
GP-SA	0.034	0.022	0.034	0.25	0.0003
GP-TA	**0.027**	**0.018**	0.025	0.18	0.0006
GP-FASATA	0.030	0.023	0.027	**0.17**	**0.00007**

In terms of summary statistics, we can observe that GP-TA has the lowest mean and median values. Furthermore, GP-FA has the lowest standard deviation value, closely followed by the remaining algorithms. The lowest maximum and minimum risk values, however, are observed for GP-FASATA. Thus, GP-TA is the control algorithm for risk.

Table 4.15: Kolmogorov-Smirnov test p-values on risk of the proposed GP-TA algorithm against the 3 GP benchmarks. Statistical significance changes are based on the Holm-Bonferroni correction. Statistically significant differences at the 5% level are indicated in boldface.

Algorithm	***GP-FASATA p-values***	**Rank**	***Significance level***
GP-FA	**7.10E-06**	2	0.025
GP-SA	**6.88E-05**	1	0.05
GP-FASATA	**3.68E-05**	3	0.016

For the risk results, the control algorithm is GP-TA, having the lowest mean and median values. The KS test does reveal that the distributions of GP-TA and the benchmarks are statistically significant different at level of 5% significance level.

4.4.4 Results of each market

Given that the data for all 42 companies originates from the same time period, it is worth noting that considerable variation exists among their respective price series. Certain companies demonstrate a tendency towards positive price movements, while others exhibit an overall negative movement. In light of these divergent market profiles, we also assess the performance of GP algorithms across these distinct scenarios. We can gain valuable insights into the algorithms' robustness and adaptability by examining their effectiveness in navigating diverse market conditions.

To facilitate the analysis, we employed a methodology that involved examining the initial and final prices of the test set for each company and calculating the corresponding return value. Based on these return values, we subsequently categorised the companies into three distinct groups:

- Group 1, which includes those companies whose price experienced a long-term increase of at least 20%.
- Group 2, which includes those companies whose price experienced a long-term increase between 0% and 19.99%.
- Group 3, which includes those companies whose price experienced a long-term decrease. , i.e. had a negative return.

After defining the above groups, 19 companies were placed in Group 1, 14 in Group 2, and 9 in Group 3.

We proceeded to report the average values of three key metrics, namely the Sharpe ratio, rate of return, and risk, for each GP algorithm across the datasets within each group. Examining Table 4.16, we observe that in Group 1, characterised by robust positive price movements, there is no outright winner in terms of rate of return or risk. However, GP-TA demonstrates the highest mean Sharpe ratio. In Group 2, GP-FASATA exhibits the highest Sharpe ratio value and the lowest level of risk, alongside GP-FA. Despite underperforming in terms of rate of return for both Group 2 and Group 3, GP-SA displays the highest average rate of return in Group 1. In Group 3, GP-FA attains the best performance in terms of both the Sharpe ratio and rate of return, while GP-TA exhibits the lowest level of risk.

This indicates that the GP-FASATA algorithm demonstrates favourable performance in certain aspects. In Group 2, GP-FASATA algorithm exhibits the highest Sharpe ratio value, which indicates a potentially attractive risk-adjusted return, while showcasing the lowest level of risk among the considered algorithms in the same group, suggesting a potentially more stable investment strategy. However, it is important to note that GP-FASATA underperforms in terms of the rate of return for both Group 2 and Group 3, implying that while it may provide a more balanced risk-return profile, it may not generate the highest returns when compared to other algorithms under those particular market conditions.

Table 4.16: Separated average results per metric per trend group.

Market	*Algorithm*	*Sharpe ratio*	*Rate of return*	*Risk*
Group 1 (>20%)	GP-FA	1.63	**0.016**	**0.028**
	GP-SA	2.2	0.013	0.029
	GP-TA	**2.7**	0.012	0.029
	GP-FASATA	2.49	0.014	0.029
Group 2 (0% - 19.99%)	GP-FA	1.09	0.003	0.031
	GP-SA	3.8	**0.010**	0.034
	GP-TA	1.89	0.0035	0.039
	GP-FASATA	**4.86**	0.006	**0.030**
Group 3 (<0%)	GP-FA	**2.99**	**0.010**	0.030
	GP-SA	2.26	0.001	0.036
	GP-TA	-1.32	-0.001	**0.025**
	GP-FASATA	1.43	0.006	0.035

4.4.5 Non-GP Benchmarks

GP-FASATA compared to machine learning benchmarks

In addition to the GP variants, we also incorporated the MLP, SVM, XGBoost, and LSTM algorithms from the scikit-learn and TensorFlow libraries in Python for further benchmarking purposes.

Upon analysing Table 4.17, we observed that the average values of MLP across the 42 companies were as follows: 0.30 for the Sharpe ratio, 0.009 for the rate of return, and 0.041 for the risk. These values stand in contrast to those obtained from the GP algorithms, which exhibited significantly higher values for the Sharpe ratio and risk. To ascertain the statistical significance of these differences, we conducted Kolmogorov-Smirnov tests. The results of Chapter 5.17 indicated that the distribution of GP-FASATA was statistically significant different from MLP for the Sharpe ratio and risk values while not statistically significant different for the rate of return.

Similarly, the SVM algorithm yielded average values of 0.37 for the Sharpe ratio, 0.01 for the rate of return, and 0.038 for the risk. Once again, there were disparities between the performances of the novel GP algorithm and SVM. The KS tests confirmed that the distribution of the GP-FASATA algorithm was statistically significant different from SVM for the Sharpe ratio values. At the same time, there are not statistically significant differences in terms of the rate of return and risk.

The XGBoost algorithm had average values of 0.36 for the Sharpe ratio, 0.010 for the rate of return, and 0.041 for the risk. GP-FASATA does perform better in the financial metrics, while the novel GP algorithm's distribution is statistically significant different to XGBoost for the Sharpe ratio values (as shown in Table 5.17), while the difference is not statistically significant for the rate of return and risk.

The LSTM algorithm had average values of 0.23 for the Sharpe ratio, 0.006 for the rate of return, and 0.043 for the risk. GP-FASATA does perform better the algorithm in each of the three financial metrics. However, based on the KS tests in Table 5.17, the distributions are only statistically significantly different for the Sharpe ratio values, as before, and not for the rate of return or risk.

These results suggest that the GP-FASATA algorithm performs better than the four machine learning benchmarks in terms of the Sharpe ratio, rate of return, and risk, showing there is a disadvantage of these four algorithms compared to GP algorithms when it comes to algorithmic trading and creating strategies using metrics that consider both the returns and the risk. However, there is no statistical difference between the distributions of the GP algorithm regarding the rate of return and risk values of the machine learning benchmarks.

Table 4.17: Comparison of average values for GP-FASATA and algorithmic benchmarks.

Algorithm	**Sharpe ratio**	**Rate of return**	**Risk**
MLP	0.30	0.009	0.041
SVM	0.37	0.010	0.038
XGBoost	0.36	0.010	0.041
LSTM	0.23	0.006	0.043
GP-FASATA	**2.99**	**0.015**	**0.030**

Table 4.18: Kolmogorov-Smirnov test p-values on all financial metrics of the proposed GP-FASATA algorithm against the 4 machine learning benchmarks. Statistical significance changes are based on the Holm-Bonferroni correction. Statistically significant differences at the 5% level are indicated in boldface.

Financial Metric	*Algorithm*	*GP-FASATA p-values*	Rank	*Significance level*
Sharpe ratio	MLP	**3.9e-6**	1	0.05
	SVM	**1.27e-05**	3	0.016
	XGBoost	**1.25e-05**	4	0.0125
	LSTM	**3.68E-05**	2	0.025
Rate of return	MLP	**0.79**	4	0.0125
	SVM	**0.29**	1	0.05
	XGBoost	**0.43**	3	0.016
	LSTM	**0.035**	2	0.025
Risk	MLP	**0.0089**	1	0.05
	SVM	**0.064**	4	0.0125
	XGBoost	**0.018**	2	0.025
	LSTM	**0.035**	3	0.016

GP-FASATA compared to the financial trading strategy

As explained in Section 4.2.3, the TS$_{d,r}$ trading strategy involves purchasing one unit of stock on the first trading day and selling it when there is a specific price increase of $r\%$ within a period of d days, or at the end of the d days period.

When comparing GP-FASATA and TS$_{d,r}$, the first thing we observe is that the latter performs on average many more trades (240), while the former performs only 29, while it is evident from Table 4.19 that GP-FASATA achieves superior values across all three metrics. Notably, GP-FASATA outperforms the TS$d, r$ strategy regarding the Sharpe ratio and risk. The differences in the distributions observed are statistically significant, as confirmed by the KS test, with p-values of $1.3e-10$ for the Sharpe ratio, 0.004 for the rate of return and $3.8e-05$ for risk. No Holm-Bonferroni correction took place, as there is only one pairwise comparison.

These findings suggest that GP-FASATA performs better than the TS$_{d,r}$ strategy regarding risk-adjusted returns, as indicated by the higher Sharpe ratio and lower risk. While the difference in the distribution of rate of return is not statistically significant, GP-FASATA still maintains an advantage across all three metrics. Therefore, based on the statistical analysis and the observed performance, it can be concluded that GP-FASATA offers a more favourable trading strategy as compared to the TS$_{d,r}$ approach, considering both risk and return.

Table 4.19: Average values of GP-FASATA and TS$d, r$.

Algorithm/Metric	***Sharpe ratio***	***Return***	***Risk***
TS$d, r$	0.10	0.006	0.067
GP-FASATA	**2.99**	**0.15**	**0.030**

GP-FASATA compared to the SP500 market index

The SP500 index involves purchasing one unit during the test set period, and selling at the end of it. Meaning only one trade took place, thus no KS test can take place. The value of the SP500 can be found in Yahoo Finance!. At the end of January 2019 that the testing period begins, the price of the index was $\$2784.49$ and by the end of February 2020, the value of the index was $\$2954.22$. The change indicates an increase of 0.06, in contract with the average rate of return of GP-FASATA which is 0.15. Thus, we understand that GP-FASATA performs better than the just buying the market index. No risk is involved of course, as we only buy the SP500 at the beginning of the testing period and selling it at the end. Even if we were buying more than one stock, the results would be the same as the period is the same.

4.4.6 Summary of findings

In conclusion, as seen in Tables 4.6 - 4.19 and focusing on the findings from GP-FASATA, the results have been summarised into 2 categories based on the GP variants' results and those of other benchmarks.

When comparing the GP variants, it is evident that:

- GP-FASATA has higher mean values regarding the Sharpe ratio and rate of return, though not the median. At the same time, the distributions of the algorithms are statistically different, as exhibited by the KS-tests. On the other hand, GP-TA has the lowest values in risk compared to the other GP algorithms, from which its distribution is also statistically significant different.

- Developing trading strategies that incorporate information from multiple indicator types (fundamental analysis, sentiment analysis, and technical analysis) can lead to more profitable and less risky strategies as compared to strategies that rely solely on a single indicator type (such as GP-FA, GP-SA, or GP-TA).

- While the GP-FASATA results in terms of the Sharpe ratio and rate of return were encouraging, it was not always clear that GP-FASATA was the best algorithm across all metrics, while the algorithm showcases limitations. A particular limitation is that the current GP representation might not allow for all three analysis types (FA, SA, and TA) to be used in a tree, which might potentially mean that the GP is not taking full advantage of these types. Thus, we are motivated to look for better ways to allow the GP algorithm to use all three types. We will do this in Chapter 5.

Regarding the comparison of GP-FASATA to the non-GP benchmarks, we observe the following:

- That the GP-FASATA algorithm demonstrates superior performance compared to four commonly used algorithmic benchmarks, namely MLP, SVM, XGBoost and LSTM, and its distribution shows statistical significance compared to theirs. Furthermore, the higher risk associated with all algorithms is evident through their low Sharpe ratio values, which are below 0.4. In contrast, the novel GP algorithm exhibits a rate of return value twice as high as that obtained from the four algorithmic benchmarks.

- The GP-FASATA algorithm offers greater financial advantages compared to the trading strategy $\text{TS}_{d,r}$. The distribution of the algorithm is statistically significant different, and in addition with the higher mean and median values we can state that it outperformed $\text{TS}_{d,r}$ in terms of key financial metrics such as the Sharpe ratio, rate of return, and risk.

4.5 Conclusion and Further Experiments

This chapter delves into the performance evaluation of a genetic programming algorithm that integrates fundamental, sentiment, and technical analysis indicators. The experimentation on a dataset encompassing 42 companies reveals that combining these three data types produces competitive outcomes compared to individual analyses or algorithms that combine only two analysis types.

In particular, the GP-FASATA algorithm surpasses the benchmark measures across the Sharpe ratio and rate of return, but not in terms of risk. This finding underscores the advantages derived from the amalgamation of indicators from fundamental analysis (FA), sentiment analysis (SA), and technical analysis (TA), especially when addressing the Sharpe ratio. While each analysis type can generate favourable trading results independently, the outcomes can be further enhanced by combining all three terminal sets. This observation suggests that while there may be individual advantages in utilising the aforementioned analysis types, the true advantages emerge when incorporating all three, as they exhibit complementary characteristics.

In the next chapter, we develop a strongly typed GP, wherein each FA, SA, and TA terminal set will be assigned a distinct branch. This approach will enable the search process to focus on each indicator type separately, aiming to refine the search quality and yield even more favourable trading outcomes. Furthermore, we will introduce a novel fitness function that focuses not only on the whole tree's fitness but also on the performances of each tree component, meaning the FA, SA, and TA indicators. This investigation aims to uncover the specific benefits arising from integrating these three analysis types, shedding light on potential synergies and the complementarity arising between them.

Chapter 5

Non-strongly and strongly-typed genetic programming

5.1 Chapter motivation

The combination of individual analysis types has generated optimism regarding the potential for achieving improved performance. However, as seen in the previous chapter, while GP-FASATA demonstrated better mean results in terms of Sharpe ratio and rate of return, its median results were not as good. In addition, in terms of risk, GP-TA was the algorithm with the lowest risk value. Thus, there seem to be limitations when integrating all 36 indicators into a non strongly-typed GP algorithm.

To address these limitations, this chapter proposes a novel strongly-typed genetic programming algorithm, one which integrates fundamental analysis (FA), sentiment analysis (SA), and technical analysis (TA). To remind the reader, a strongly-typed genetic programming (STGP) architecture ensures that only operations compatible with specific data types are allowed, meaning that the grammar explicitly defines and enforces data types of various elements within a program. STGP ensures that only "valid" combinations (combinations of the same analysis type) of functions and terminals are generated and that the resulting programs adhere to a predefined type structure.

The strongly-typed nature of GP enforces the use of FA, SA, and TA indicators into distinct branches or subtrees within the GP model. Consequently, each GP tree incorporates a dedicated branch exclusively focused on FA indicators, another dedicated branch for SA indicators, and a third dedicated branch addressing TA indicators. This type of constraint facilitates targeted exploration within each indicator type's search space, thereby promoting more effective exploration and exploitation. Type constraints are incorporated during crossover and mutation operations to ensure the validity of generated trees.

Moreover, we introduce a new fitness function, one which considers not only the performance of a given individual (tree), as it usually happens in evolutionary algorithms, but also the performance of the FA, SA and TA subtrees. As a result, the GP evolves individuals, which ensures that *all* analysis indicators contribute to the overall performance of a GP individual. This is particularly important because it guarantees good performance for each component of an individual (FA, SA, and TA subtrees) and also good performance for the overall individual, which contains the aforementioned FA, SA, and TA subtrees.

The primary objective of this chapter is to demonstrate the efficacy of the strongly-typed GP architecture when combining the three financial analysis types, as well as to introduce a novel fitness function whereby all components are equally taken into consideration. To assess its performance, a comparative analysis is conducted, pitting the proposed GP algorithm against the GP-FASATA and GP-TA algorithms introduced in the preceding chapter. The experimental evaluation is conducted on the same dataset as in the previous chapter, comprising stock data from 42 international companies and evaluating the same three financial metrics (i.e., Sharpe ratio, rate of return, and risk) that are employed to assess and report the performance of the algorithms.

The rest of this chapter follows the structure of Chapter 4 and it is organised as follows: First, we present the methodology we follow (Section 5.2), followed by the experimental setup (Section 5.3), and, finally, we present the results and the analysis (Section 5.4) before the conclusion and future works (Section 5.5).

5.2 Methodology

This chapter presents the STGP-FASATA-S, an innovative genetic programming (GP) algorithm that incorporates a strongly-typed architecture capable of distinguishing between fundamental, sentiment, and technical analysis types. We contend that the adopting such an architecture enhances the algorithm's capacity to explore each search space effectively, leading to models that fully exploit the data type indicators.

The research methodology comprises three main components. Firstly, Section 5.2.1 offers an overview of the STGP methodology and the model representation, while the GP operators are introduced in Section 5.2.2. Subsequently, Section 5.2.3 reminds the reader of the trading signals and the trading strategy utilised in the thesis. Finally, Section 5.2.4 details the fitness function and metrics that will be taken into consideration during the evaluation process, which are novel in this thesis contribution.

5.2.1 Model representation

Part 1 of Figure 5.1 illustrates a representative tree structure generated by the STGP-FASATA algorithm. The algorithm's strongly-typed architecture imposes specific constraints on the tree's composition. Specifically, the root node of the tree is configured to have three children, each corresponding to a distinct category, specifically fundamental analysis (FA) indicators, sentiment analysis (SA) indicators, and technical analysis (TA) indicators. To enforce this structure, the root node is consistently set as a `3-AND` function that serves as the junction point for the three branches. Notably, the first branch of the `3-AND` function is exclusively dedicated to FA-related indicators. In contrast, the second branch is solely designated for SA-related indicators, and the third branch is exclusively assigned to TA-related indicators.

The function nodes are based on a 3-input `AND` function with a Boolean output (we call it `3-AND`), as well as binary `AND`, `OR`, Greater than (`GT`) and Less than (`LT`) functions, each with different variants allowing for each indicator type. In particular, our algorithm uses $\texttt{AND}_{\texttt{FA}}$, $\texttt{OR}_{\texttt{FA}}$, $\texttt{GT}_{\texttt{FA}}$, $\texttt{LT}_{\texttt{FA}}$ function nodes in the FA branch, $\texttt{AND}_{\texttt{SA}}$, $\texttt{OR}_{\texttt{SA}}$, $\texttt{GT}_{\texttt{SA}}$, $\texttt{LT}_{\texttt{SA}}$ function nodes in the SA branch and it uses $\texttt{AND}_{\texttt{TA}}$, $\texttt{OR}_{\texttt{TA}}$, $\texttt{GT}_{\texttt{TA}}$, $\texttt{LT}_{\texttt{TA}}$ function nodes in the TA branch.

The `3-AND` function takes 3 children. The first is of type FA, the second is of type SA, and the third is of type TA. Thus, the first branch can take any of the relevant FA-type functions, i.e. $\texttt{AND}_{\texttt{FA}}$, $\texttt{OR}_{\texttt{FA}}$, $\texttt{GT}_{\texttt{FA}}$, or $\texttt{LT}_{\texttt{FA}}$. The same principle applies to the second (SA) and third (TA) branches. All three branches of the `3-AND` function need to evaluate TRUE in order for the function to yield a TRUE outcome.

The function set used in our algorithm is summarised in Table 5.1. This carefully designed function set ensures that the algorithm generates models that fully utilise all indicator types (i.e., FA, SA, and TA) while enforcing type consistency at the same time. In doing so, we can prevent type errors and enhance the exploration of the search space.

Table 5.1: Function set for the ActTrade algorithm.

Explanation	**Function nodes**
Root node	3-AND
FA, SA and TA type for AND	AND_{FA}, AND_{SA}, AND_{TA}
FA, SA and TA type for OR	OR_{FA}, OR_{SA}, OR_{TA}
FA, SA and TA type for Greater Than	GT_{FA}, GT_{SA}, GT_{TA}
FA, SA and TA type for Less Than	LT_{FA}, LT_{SA}, LT_{TA}

The terminal sets consist of the same indicators as in Chapter 4, which can be found in Table 4.1. However, the key change is that different sets are allowed at different branches of the corresponding trees. In the FA branch (Table 2.1), the terminal set consists of specific indicators permitted to be used within that branch. Similarly, the SA branch (Table 2.2) and the TA branch (Table 2.3) have their own terminal sets of permissible indicators. Additionally, all terminal sets include an Ephemeral Random Constant (ERC) variable, which serves as a threshold value for the indicators and consists of random values between -1 and 1.

Compared to a non-strongly typed GP, our proposed algorithm has the advantage of fully utilising the search space of each individual indicator type. The three branches corresponding to the FA, SA and TA indicators are connected using the 3-AND function at the root of the tree. This integration creates a foundation for better exploration and exploitation of the search space. Consequently, our algorithm can generate a broader range of trading strategies, which are not only diverse but also more effective and adaptable.

5.2.2 GP operators

As in Chapter 4, our novel algorithm again utilises elitism, as well as subtree crossover and one-point mutation. The difference of this novel algorithm's GP architecture, though, is that the subtree crossover does not allow the indicators to "interact" and exist within the same branch but ensures the exchanged nodes must be of the same type (e.g., a terminal node with another terminal node) and data type (e.g., FA branch with FA branch). This is implemented to maintain the validity of the tree. To ensure the legality of the tree exchange, we first exchange the FA branches of the two selected parents, and once that process is complete, we exchange the SA branches of the two trees, followed by the exchange of TA branches.

Finally, we use point mutation in a strongly-typed setting, where specific limitations are, again, in place. For example, a function node such as $\mathtt{OR}_{\mathtt{SA}}$ can only be changed to $\mathtt{AND}_{\mathtt{SA}}$, while a function node like $\mathtt{GT}_{\mathtt{TA}}$ can be replaced only with $\mathtt{LT}_{\mathtt{TA}}$ (and similarly for other function nodes), an ERC can only be replaced with another ERC, and a terminal variable can only be replaced with another variable of the same indicator type. The algorithm ensures that valid data types replace the mutated nodes. Point mutation occurs in one of the three branches per tree.

The individuals selected as parents for these operators are determined through tournament selection. A selected individual will undergo crossover with a probability of p and will undergo mutation with the remaining probability of $1 - p$.

5.2.3 Trading signals and trading strategy

Trading signals are generated using the STGP-FASATA-S trees, where the outcome of the root 3-AND function, which evaluates to either TRUE or FALSE, is utilised to provide a recommendation regarding whether to buy a stock or hold it. As mentioned in Chapter 4, this signal is determined by employing an If-Then-Else (ITE) node as the root of a separate tree structure. An example STGP-FASATA-S tree can be found in Figure 5.1.

The evolved tree within Part 1 of Figure 5.1, the root node is denoted as `3-AND`. The FA branch commences with the $\texttt{LT}_{\texttt{FA}}$ node, as indicated by orange-coloured nodes. Similarly, the SA branch begins with the $\texttt{GT}_{\texttt{SA}}$ node, also represented by orange-coloured nodes. Finally, the TA branch corresponds to the $\texttt{LT}_{\texttt{TA}}$ node, distinguished by yellow-coloured nodes.

The rest of the trading strategy is the same as in the first as in Chapter 4. To remind the reader, each evolving STGP model is integrated into a tree structure, wherein an If-Then-Else (ITE) node serves as the root. The second and third branches of this ITE statement are static and represent the decisions to buy (1) and hold (0) stock, respectively, as depicted in Figure 5.1. The trading strategy examines whether the price exhibits an increase of $r\%$ within the subsequent d days, or if the price does not meet this condition, the strategy opts to sell the stock on the d^{th} day. The parameters d and r are fine-tuned during the validation phase and remain constant for all considered GP algorithms, as discussed in Section 4.3.3.

5.2.4 Fitness function and Metrics

As seen in the previous chapter (Chapter 4) in Section 4.2.4, our analysis considers the metrics of rate of return, risk and Sharpe ratio.

To remind the reader, the *return*, R, of a trade, captures the profit made as a percentage of the amount invested. The *rate of return* (RoR) represents the sample mean of the returns of all trades , and the *risk* is measured as the standard deviation of the returns. The *Sharpe ratio*, S_r, is the expected value of the excess return compared to the risk free return, R_f, over the risk, as introduced in Chapter 4, Equation 4.2.

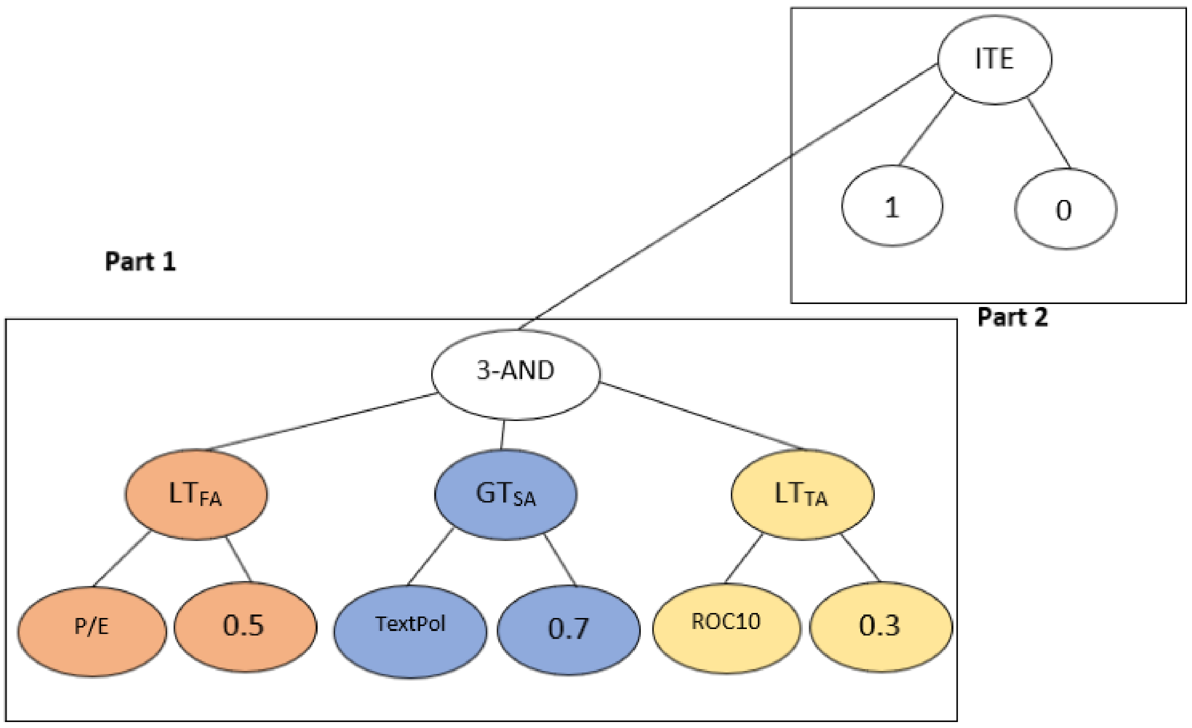

Figure 5.1: Sample tree of STGP-FASATA-S. The first child of the 3-AND function is enforced to be FA-related, while the second child is SA-related, and the third is enforced to be TA-related. This sample tree checks if the P/E indicator is less than the ERC of 0.5 if the TEXTpol indicator is greater than the ERC 0.7, and if the ROC10 indicator is less than the ERC 0.3. If all three of them are TRUE, then the recommendation will be to buy (1) otherwise it will be to hold (0).

The difference with Chapter 4 lies in the fact that the new fitness function does not aim to maximise the Sharpe ratio of the complete tree solely. Instead, the new fitness function is the summation of the Sharpe ratio, S_r^C, of the complete tree, which combines FA, SA and TA indicators; the Sharpe ratio, S_r^{FA} of the subtree with FA indicators, S_r^{SA}, the Sharpe ratio of the subtree with only SA indicators, the Sharpe ratio, S_r^{TA}, of the substree that considers only TA indicators, with weights w_c, w_{fa}, w_{sa}, and w_{ta}. Formally,

$$f_{sum} = w_c \cdot S_r^C + w_f a \cdot S_r^{FA} + w_{sa} \cdot S_r^{SA} + w_{ta} \cdot S_r^{TA} \tag{5.1}$$

For example, in Figure 5.1, S_r^C corresponds to the subtree with root node `3-AND` (i.e. Part 1), S_r^{FA} corresponds to the subtree with root node $\texttt{LT}_{\texttt{FA}}$ (orange-coloured nodes), S_r^{SA} corresponds to the subtree with root node $\texttt{GT}_{\texttt{SA}}$ (blue-coloured nodes), and S_r^{TA} corresponds to the subtree with root node $\texttt{LT}_{\texttt{TA}}$ (yellow-coloured nodes).

The advantage of using this fitness function is that it allows us to evolve trees that maximise all four Sharpe ratios. As a result, the GP can guide the search towards trading strategies that exhibit strong performance across all four components of the fitness function. This is particularly important because if, for example, the fitness function was only the Sharpe ratio of the whole tree (as it usually happens in such cases in the literature), then the GP would be able to identify well-performing trees, but would not necessarily take advantage of its strongly typed nature that ensures that there are always FA, SA, and TA indicators present.[1]

To conclude, the proposed GP algorithm is a strongly typed GP which aims to maximise the Sharpe ratio. The fitness function is determined by the weighted sum of the FA, SA, and TA subtrees and the complete tree comprising the three subtrees. The strongly typed structure of the GP enables more effective exploration and utilisation of all three indicator types. At the same time, the fitness function emphasises each indicator's contribution to enhancing the

[1]In fact, early experiments have indicated exactly this: when the fitness function was only the Sharpe ratio of the whole tree, very frequently, the best tree would have a large and well-performing subtree on a single analysis type, e.g. the SA side, but a small and bad-performing subtree on the FA and TA sides. In addition, the performance of the overall tree was no better than the performance of a non-strongly typed GP that allowed the presence of FA, SA, and TA indicators, thus making the use of the strongly typed feature redundant. Our proposed fitness function overcomes this limitation.

algorithm's trading performance.

5.3 Experimental Setup

As in Chapter 4, in the experimental setup section, we will quickly mention the data collection steps again, Section 5.3.1, introduce the GP benchmarks of this chapter in Section 5.3.2, and we will finally remind the reader of the parameter tuning process in Section 5.3.3.

5.3.1 Data

The data collection process is the same as in Chapter 4, and can be found in detail in Section 4.3. As mentioned before, we gathered data from 42 international companies, derived from different stock exchanges, between January 1st, 2015, and concluding on January 31st, 2020. The fundamental analysis data were gathered from "Seeking Alpha". A web scraper was used for the sentiment analysis data collection, and Yahoo! Finance was the tool employed to collect the daily closing prices to create the technical analysis data. With these tools, we obtained 36 indicators in total, 12 from each analysis type, which we subsequently divided into three sequential portions.

5.3.2 Benchmarks

While GP-FASATA outperformed GP-FA, GP-SA, and GP-TA in the mean values of Sharpe ratio and rate of return, it did not outperform them in terms of the median values. This is why we cannot consider GP-FASATA to be the outright winner of Chapter 4. Thus, we will benchmark the proposed STGP-FASATA-S algorithm against all four non strongly-typed algorithms from Chapter 4 and the strongly-typed version of GP-FASATA, namely STGP-FASATA. We remind the reader of the following algorithms:

- **GP-FA** is a GP algorithm that only includes fundamental analysis indicators in its terminal set. The motivation for this benchmark is to evaluate the trading performance of a non strongly-typed GP that only uses fundamental analysis indicators.
- **GP-SA** is a GP algorithm that only includes sentiment analysis indicators in its terminal set. As mentioned above, the motivation for this benchmark is to evaluate the performance of a GP using only SA indicators.
- **GP-TA** is a GP algorithm that only includes technical analysis indicators in its terminal set. The motivation here is similar to the use of GP-FA and GP-SA.
- **GP-FASATA** is a (non-strongly typed) GP algorithm that combines indicators of fundamental, sentiment and technical analysis. The motivation for using this algorithm is to evaluate the GP's trading performance when it uses all three indicator types in its terminal set, albeit in a non strongly-typed manner. As there are no branch types, the GP can choose any and as many indicators as it sees fit, regardless of whether they are derived from fundamental, sentiment, or technical analysis.
- **STGP-FASATA** is a strongly-typed GP algorithm that combines fundamental, sentiment and technical analysis indicators. This algorithm does not use our novel fitness function, and thus, it acts as a valuable benchmark to enable us to understand the added value of the aforementioned innovations of the proposed GP. This is essentially the algorithm's divergence from the proposed STGP-FASATA-S algorithm.

The motivation behind using these algorithms is to assess the trading performance of a GP model when all three indicator types are available in its terminal set without the constraints of type enforcement and with the constraints of the strongly-typed GP architecture.

The STGP-FASATA-S algorithm employs the same strongly-typed architecture as the STGP-FASATA algorithm yet uses a novel fitness function. By comparing the STGP-FASATA-S algorithm with both GP-FASATA and STGP-FASATA algorithms, we can evaluate the impact of type enforcement on the GP models' trading performance and establish the advantages of a more elaborate fitness function. These comparisons provide insights into the advantages and limitations of employing a strongly-typed GP in utilising different indicator types and generating effective trading strategies.

Furthermore, as seen in Chapter 4, the study also included four additional algorithmic benchmarks with the same prediction strategy, namely, the Multilayer perceptron (MLP), Support vector machine (SVM), eXtreme Gradient Boosting (XGBoost), and Long short-term memory (LSTM), as well as a financial benchmark, ($TS_{d,r}$). However, the proposed STGP-FASATA-S will only be compared against the machine learning benchmarks, as the financial benchmark and the SP500 index have already been outperformed across the financial metrics by GP-FASATA and will not be included in this chapter.

5.3.3 Parameter Tuning

As in Chapter 4, a two-step grid search was conducted using the validation set to determine the optimal parameters for the GP algorithm. The steps of the parameter tuning have been described in Chapter 4, while the parameters remain the same and can be found in Table 5.2.

Table 5.2: GP Parameters for the eight GP algorithms.

GP Parameters	
Population size	1000
Crossover probability	0.95
Mutation probability	0.05
Generations	50
Tournament size	4
Maximum tree depth	6

Again, tuning the trading strategy parameters (d and r) is conducted independently for each company across the algorithms employed in the study.

5.4 Results and Discussion

In this section, we present the experimental results of our study, focusing on the comparison between the strongly-typed GP algorithms, STGP-FASATA-S and STGP-FASATA, and the non strongly-typed algorithms, GP-FA, GP-SA, GP-TA, and GP-FASATA. To assess the performance of each algorithm, we conducted 50 independent runs on the training set for each of the 42 companies, resulting in a total of 50 distinct trading strategies for each algorithm. As in Chapter 4, we include Tables 5.3 and 5.4 to showcase the mean Sharpe ratio values of the runs in each company, separated in the first and last 21 companies. Moreover, the median Sharpe ratio values of all the runs across all companies can be found in Table 5.5. Similarly, the median values for rate of return and risk can be found in Sections 5.4.2 and 5.4.3, respectively.

As in Chapter 4, to ensure the accuracy and reliability of our statistical analysis, certain runs were excluded from the calculation of mean values presented in Tables 5.5, 5.9, and 5.13. Again, this exclusion was done to avoid distortions in the results and maintain the integrity of the statistical measures. More specifically, runs where the GP and STGP algorithms did not make any trades due to potential losses were not considered in calculating the mean values of the Sharpe ratio, rate of return, and risk. The provided tables present the mean, median, standard deviation, maximum, and minimum values for each algorithm across the 42 companies.

Again, to validate our findings, we employed the non-parametric Kolmogorov-Smirnov (KS) test, which is suitable for comparing two independent groups, and applied it to all runs across companies for each algorithm. The KS test helps to ensure the robustness and reliability of our results.

Since the comparisons made are between the STGP-FASATA-S, STGP-FASATA, and the non-strongly GP algorithms, we, again, need to take into consideration the multiple comparisons. Therefore, there is a need for the application of a correction method, such as the Holm-Bonferroni correction. In this section, we conducted five separate comparisons for each financial metric, comparing the STGP-FASATA-S algorithm with the three other GP benchmarks studied herein.

For each rank, which represents the magnitude order of the p-values (in this case, with 1 being the smallest and 5 being the largest), the minimum acceptable p-value threshold is calculated using the formula $\alpha(rank) = \frac{0.05}{5-rank+1}$. To assess the statistical significance of differences between two samples at a 5% significance level, we compared the p-values obtained from each comparison to the corresponding minimum acceptable p-value for its rank. Specifically, the first-ranked p-value should be less than 0.01, the second-ranked p-value should be less than 0.0125, the third-ranked p-value should be less than 0.016, the fourth-ranked p-value should be less than 0.025, and the fifth-ranked p-value should be less than 0.05.

5.4.1 Sharpe ratio

Tables 5.3 and 5.4 present the median Sharpe ratio values over the 50 runs for the 42 companies. As we can observe, the algorithm that has the most optimal Sharpe ratio values is GP-SA with 13 companies, GP-FA (11), followed by GP-TA (7), STGP-FASATA (5), STGP-FASATA-S (5), and GP-FASATA (4). As mentioned in Chapter 4, it needs to be noted that in some cases, the mean Sharpe ratio values are close. In this chapter, we observe that STGP-FASATA-S has more companies with 0 Sharpe ratio values, indicating that this is a more conservative algorithm than the benchmarks.

Table 5.5 displays the mean and median Sharpe ratio for each algorithm, as well as the standard deviation and maximum and minimum values of 50 runs for each company across the eight GP algorithms.

Table 5.3: Median values for Sharpe ratio per company. Boldface is used to denote the best value for the particular dataset. First 21 companies.

Company	*GP-FA*	*GP-SA*	*GP-TA*	*GP-FASATA*	*STGP-FASATA*	*STGP-FASATA-S*
AAPL	**6.747**	0.35	5.29	5.06	2.379	4.932
ADIDAS	0.045	0.099	**0.256**	0.2109	0.183	0
ALIBABA	0.5242	**12.872**	0.4731	3.3684	0.8336	2.5453
AMAZON	1.179	0.155	**19.44**	1.457	1.468	0
ASUS	-1.375	0.3301	**0.5215**	-1.375	-0.5085	-0.5439
ATVI	0.3558	2.063	**3.506**	0.4423	0.5166	0.3749
BLACKB	1.027	-0.212	0.875	0.7	1.3265	**1.7034**
COKE	**13.07**	-0.581	0.615	1.214	1.20975	1.094
EBAY	**0.670**	-0.142	0	-0.549	-0.2471	0.43
ESTEE	0.043	2.05	0.77	0.371	2.277	**3.8917**
FORD	-0.123	**8.301**	-0.595	0.0989	6.487	0
FUJIFILM	**1.044**	0.0551	-1.472	-0.630	0.1352	-1.7363
FUJITSU	0.7532	3.707	14.624	13.584	2.443	**24.5146**
GM	-0.25	**0.858**	-20.91	-0.238	-0.6598	0
GOOGL	0.052	0.0823	0.0162	-0.11	**0.0886**	-0.1268
HITACHI	**0.097**	0.355	0	**0.097**	-1.4215	0
HONDA	2.217	**9.044**	3.341	0.594	2.1395	2.1479
HSBC	-0.225	**3.939**	0.0372	0.0133	0.3209	0
HYUNDAI	0.2059	0.7194	-0.6451	**2.5834**	0.3972	1.124
IBM	0.5286	0.194	**7.216**	0.1768	0.1098	3.2115
INTC	0.6264	1.5177	1.31	0.1648	**2.2122**	0

Table 5.4: Median values for Sharpe ratio per company. Boldface is used to denote the best value for the particular dataset. Last 21 companies.

Company	*GP-FA*	*GP-SA*	*GP-TA*	*GP-FASATA*	*STGP-FASATA*	*STGP-FASATA-S*
KERING	0.5836	0.9582	0.9863	0.3646	**1.16502**	0.3959
KODAK	0.9546	1.4971	**1.9477**	0.11	1.26547	1.0454
MCDON	-0.05	**0.0982**	-4.3045	0	-0.0125	0
NESTLE	-0.209	**2.3209**	-0.1383	-0.0065	2.17719	-0.1383
NFLX	**8.0470**	0.0801	-0.2085	**8.0470**	2.44843	5.94198
NINTENDO	0.1078	**1.1455**	-0.4265	0.2703	0.35842	0.71975
NYT	0.3043	**3.4819**	0.1407	0.2134	0.18615	0.45961
PANASONIC	-0.004	**2.505**	0.9242	0.9076	0.09342	0.2917
SAINSBURY	**3.2401**	2.7915	1.246	1.0172	1.02793	1.4162
SHISEIDO	-0.7148	-0.1614	0.5601	-0.9181	-0.4704	**1.89246**
SUBARU	-0.3436	-0.4698	**1.133**	0	-0.3314	0.090107
SUZUKI	**7.353**	-0.783	-0.1657	0.804	0	6.32750
TENCENT	0.2961	0	-0.554	**2.528**	0.98139	0.3919
TESCO	0.194	0.9316	0.35	-0.257	**15.4991**	0
TESLA	1.6276	**3.106**	2.140	1.086	1.22883	1.4788
TOYOTA	**3.0495**	0.3721	0	-0.344	3.0060	0
UBISOFT	1.856	0	0.8739	0.9205	**2.03319**	-0.432
WALMART	-0.0558	**0.6368**	0.3542	0.090	0.31503	0.12854
XEROX	**2.1182**	0.054	-0.8289	0.5039	1.4751	0
YAMAHACO	0.6108	**1.4**	0.0588	1.095	1.1658	1.1852
YAMAHAMO	**0.7899**	0	0.5951	0	-0.209	-0.13029

Table 5.5: Summary statistics of Sharpe ratio across all 50 GP runs and all 42 companies. Boldface is used to denote the best value for each statistic.

Algorithm	*Average*	*Median*	*StDev*	*Maximum*	*Minimum*
GP-FA	2.18	0.67	10.37	83.59	-52.54
GP-SA	2.65	0.57	10.75	173.13	-14.16
GP-TA	2.32	0.70	**5.62**	31.47	-20.9
GP-FASATA	2.99	0.58	12.1	**206.25**	-9.21
STGP-FASATA	3	0.64	9.9	144.32	**-8.81**
STGP-FASATA-S	**3.78**	**1.07**	11.57	83.59	-27.80

Table 5.6: Kolmogorov-Smirnovtest p-values on Sharpe ratio of the proposed STGP-FASATA-S algorithm against the 5 GP benchmarks. Statistical significance changes are based on the Holm-Bonferroni correction. Statistically significant differences at the 5% level are indicated in boldface.

Algorithm	*STGP-FASATA-S p-values*	Rank	*Significance level*
GP-FA	**1.70E-45**	4	0.025
GP-SA	**3.34E-38**	5	0.05
GP-TA	**1.03E-48**	3	0.016
GP-FASATA	**1.82E-84**	1	0.01
STGP-FASATA	**4.31E-65**	2	0.0125

As has been noted before, when examining the results of the Sharpe ratio, it is crucial to bear in mind that when analysing the Sharpe ratio's results, one must account for its sensitivity to the number of trades. When the number of trades is limited, the risk calculation can produce exceptionally small values, potentially resulting in significantly high Sharpe ratios, as risk serves as the denominator in the ratio.

Referring to Table 5.5, it is evident that STGP-FASATA and GP-FASATA have similar average Sharpe ratios, while the median Sharpe ratio of STGP-FASATA slightly outperforms that of the GP-FASATA and the rest of the non-strongly typed GP variants. On the other hand, STGP-FASATA-S outperforms all other algorithms in the mean and median values for the Sharpe ratio. However, the standard deviations of the algorithms differ, indicating a variance in the distribution of Sharpe ratios around the respective means, wherein the GP-TA has the lowest standard deviation value.

When conducting the Kolmogorov-Smirnov tests we compare the distribution of STGP-FASATA-S, as the control algorithm, with the distributions of the other benchmarks. In this case the p-value of the KS test, found in Table 5.6, suggesting that there are statistically significant differences in the distributions of STGP-FASATA-S and the benchmark algorithms.

5.4.2 Rate of Return

Tables 5.7 and 5.8 showcase the median rate of return values over the 50 runs for the 42 companies. The algorithm with the most best rate of return values is again GP-SA with 11 companies, followed by STGP-FASATA-S (9), STGP-FASATA (8), GP-FA (7), GP-TA (6), and GP-FASATA (2). As indicated, in many cases, the median rate of return values are close.

Table 5.7: Median values for rate of return per company. Boldface is used to denote the best value for the particular dataset. First 21 companies.

Company	*GP-FA*	*GP-SA*	*GP-TA*	*GP-FASATA*	*STGP-FASATA*	*STGP-FASATA-S*
AAPL	0.0254	0.0129	0.0240	0.0245	0.0240	**0.0285**
ADIDAS	0.0014	0.0017	**0.0107**	0.0083	0.0041	0
ALIBABA	0.0185	0.0303	0.0158	0.0326	0.0248	**0.0329**
AMAZON	**0.0295**	0.0007	0.0173	0.0256	0.0236	0
ASUS	-0.033	0.0096	**0.0190**	-0.018	0.0013	0.00092
ATVI	0.0081	0.0102	**0.0157**	0.0100	0.0120	0.00785
BLACKB	**0.0355**	-0.018	0.0346	0.0180	0.0307	0.0307
COKE	**0.0239**	-0.040	0.0077	0.00809	0.01310	0.00839
EBAY	0.0088	-0.004	0	-0.0284	-0.0127	**0.0102**
ESTEE	0.0013	0.0098	0.0107	0.00746	**0.0117**	0.01004
FORD	-0.004	**0.0140**	-0.028	0.00289	0.0162	0
FUJIFILM	**0.0317**	0.0308	-0.027	-0.0136	0.0036	-0.0308
FUJITSU	0.0355	0.0645	0.1018	0.10181	0.08038	**0.1038**
GM	-0.013	**0.0171**	-0.089	-0.0124	-0.03538	0
GOOGL	0.0015	**0.0027**	-0.0004	-0.0022	0.0021	-0.0033
HITACHI	0.0054	**0.0176**	0	0.0054	-0.020306	0
HONDA	0.0175	0.0114	0.01473	0.0120	**0.01766**	0.01652
HSBC	-0.0042	0.0125	0.00046	**0.0133**	0.00192	0
HYUNDAI	0.0069	0.0124	-0.020	0.0270	0.01908	**0.034**
IBM	0.01676	0.0132	0.03612	0.0111	0.00756	**0.0466**
INTC	0.0180	**0.0195**	0.0179	-0.0046	0.0185	0

Table 5.8: Median values for rate of return per company. Boldface is used to denote the best value for the particular dataset. Last 21 companies.

Company	*GP-FA*	*GP-SA*	*GP-TA*	*GP-FASATA*	*STGP-FASATA*	*STGP-FASATA-S*
KERING	0.0217	0.0176	0.0163	0.0151	**0.0297**	0.0191
KODAK	0.0232	**0.0534**	0.03832	0.00576	0.0331	0.0227
MCDON	-0.00195	**0.00575**	-0.0003	0	-0.0125	0
NESTLE	-0.0049	**0.00820**	-0.0021	0.000113	0.0076	-0.00213
NFLX	**0.06071**	0.0080	-0.02305	**0.06071**	0.0437	0.0590
NINTENDO	0.0035	**0.02509**	-0.03231	0.0153	0.0113	0.0069
NYT	0.0126	**0.0385**	0.006101	0.00910	0.00772	0.01596
PANASONIC	-2E-06	**0.0269**	0.01694	0.01822	0.00453	0.01152
SAINSBURY	**0.0491**	0.040347	0.01798	0.02039	0.01676	0.01967
SHISEIDO	-0.0448	-0.01658	0.02077	-0.0448	-0.0258	**0.0282**
SUBARU	-0.0150	-0.02840	**0.03148**	0	-0.0226	0.00364
SUZUKI	0.05504	-0.03080	-0.0133	0.0373	0	**0.0645**
TENCENT	0.0079	0	-0.0543	0.0182	**0.0139**	0.00987
TESCO	0.00682	0.0163	0.01033	-0.0096	**0.0476**	0
TESLA	0.0661	0.0527	**0.07098**	0.04592	0.0480	0.0656
TOYOTA	0.01723	0.01065	0	-0.0028	**0.0333**	0
UBISOFT	0.0185	0	**0.0288**	0.02499	0.0126	-0.0442
WALMART	-0.0008	0.00447	0.00363	0.00447	**0.0049**	0.00198
XEROX	0.0327	0.003	-0.0174	0.0320	**0.04045**	0
YAMAHACO	0.0128	0.01209	0.0025	0.0144	0.0130	**0.0159**
YAMAHAMO	**0.0170**	0	0.01025	0	-0.00363	-0.00229

Table 5.9 presents the mean, median, standard deviation, maximum, and minimum values of the rate of return obtained from 50 iterations for each company, considering the application of the six genetic programming (GP) algorithms.

Table 5.9: Summary statistics of rate of return across all 50 GP runs and all 42 companies. Boldface is used to denote the best value for each statistic.

Algorithm	***Average***	***Median***	***StDev***	***Maximum***	***Minimum***
GP-FA	0.014	**0.015**	0.038	0.069	-0.30
GP-SA	0.009	0.010	**0.028**	**0.10**	-0.12
GP-TA	0.013	0.014	0.033	0.10	-0.089
GP-FASATA	0.015	0.013	0.029	0.10	-0.10
STGP-FASATA	0.014	0.0125	**0.028**	0.10	-0.18
STGP-FASATA-S	**0.021**	**0.015**	0.035	**0.11**	**-0.087**

Upon reviewing the summary statistics presented in Table 5.9 and the statistical tests of Table 5.10, it is evident that STGP-FASATA-S does demonstrate better mean and median values for the rate of return. At the same time, GP-FASATA performs slightly better than the remaining GP benchmarks. Moreover, GP-FA has the same median value as STGP-FASATA-S, followed by GP-TA. However, the standard deviation value of STGP-FASATA-S is higher than that of GP-FASATA and STGP-FASATA, which are similar and fall closer to that of the GP-TA algorithm. Furthermore, STGP-FASATA-S exhibits a higher maximum value and a lower minimum value.

Based on the statistical tests presented in Table 5.10, we again compare the distribution of the control algorithm STGP-FASATA-S with the distributions of the benchmarks. All boldfaced

Table 5.10: Kolmogorov-Smirnovtest p-values on rate of return of the proposed STGP-FASATA-S algorithm against the 5 GP benchmarks. Statistical significance changes are based on the Holm-Bonferroni correction. Statistically significant differences at the 5% level are indicated in boldface.

Algorithm	*STGP-FASATA-S p-values*	Rank	*Significance level*
GP-FA	**4.35E-51**	3	0.025
GP-SA	**1.34E-44**	5	0.05
GP-TA	**3.51E-49**	4	0.016
GP-FASATA	**7.97E-84**	1	0.01
STGP-FASATA	**1.74E-64**	2	0.0125

p-values signify statistical significance at the 5% level after applying the Holm-Bonferroni correction, as an adjustment for controlling the family-wise error rate in multiple tests. Based on the results, the distribution of the STGP-FASATA-S algorithm demonstrated statistically significant differences in the rate of return compared to the distributions the GP benchmarks.

5.4.3 Risk

Tables 5.11 and 5.12 showcase the average rate of return values over the 50 runs for the 42 companies. The algorithm with the most best rate of return values is again GP-SA with 11 companies, followed by STGP-FASATA-S (9), STGP-FASATA (7), GP-FA (7), GP-TA (6), and GP-FASATA (2). Again, as indicated in many cases, the median risk values are close to each other.

Table 5.11: Averages for risk per company. Boldface is used to denote the best value for the particular dataset. First 21 companies.

Company	*GP-FA*	*GP-SA*	*GP-TA*	*GP-FASATA*	*STGP-FASATA*	*STGP-FASATA-S*
AAPL	0.005697	0.03179	**0.00464**	0.00469	0.01399	0.00541
ADIDAS	0.02695	**0.01415**	0.02963	0.02471	0.02399	0
ALIBABA	0.0350	**0.00404**	0.03700	0.01055	0.02960	0.01312
AMAZON	0.02484	0.02694	**0.00088**	0.01745	0.01914	0
ASUS	0.02482	0.02310	0.03615	0.01410	**0.01285**	0.01620
ATVI	0.02240	**0.00359**	0.00825	0.02212	0.02209	0.02006
BLACKB	0.0326	0.08741	0.02545	0.02545	0.02391	**0.01794**
COKE	**0.00181**	0.07011	0.01291	0.00775	0.00927	0.00747
EBAY	**0.01286**	0.02923	0	0.05331	0.04924	0.02337
ESTEE	0.02691	0.00452	0.01466	0.01525	0.00613	**0.00304**
FORD	0.02799	**0.00166**	0.04818	0.02722	0.00351	0
FUJIFILM	0.03016	0.05543	0.01883	0.02205	0.02563	**0.01786**
FUJITSU	0.0515	0.01736	0.00697	0.00697	0.03339	**0.00429**
GM	0.05325	0.01967	**0.00428**	0.05317	0.05301	0
GOOGL	0.0253	**0.01223**	0.04068	0.02199	0.02179	0.02787
HITACHI	0.0533	0.04914	0	0.0533	**0.01443**	0
HONDA	0.0080	**0.00123**	0.00539	0.01996	0.00883	0.00793
HSBC	0.01989	**0.00312**	0.00662	0	0.00532	0
HYUNDAI	0.0325	0.02157	0.04458	**0.01387**	0.03814	0.03038
IBM	0.03129	0.06740	**0.00202**	0.06177	0.06206	0.01446
INTC	0.0285	0.01275	0.01479	0.06467	**0.00829**	0

Table 5.12: Averages for risk per company. Boldface is used to denote the best value for the particular dataset. Last 21 companies.

Company	*GP-FA*	*GP-SA*	*GP-TA*	*GP-FASATA*	*STGP-FASATA*	*STGP-FASATA-S*
KERING	0.037776	0.02704	0.03209	0.03977	**0.02533**	0.04783
KODAK	0.023521	0.03556	**0.01956**	0.05659	0.02755	0.03425
MCDON	0.043232	0.04056	**0.02985**	0	0	0
NESTLE	0.024797	**0.00344**	0.01703	0.01620	0.00574	0.01703
NFLX	**0.007517**	0.07614	0.10521	0.00755	0.01763	0.01007
NINTENDO	0.030983	0.03228	0.03854	0.03330	**0.01963**	0.05054
NYT	0.040852	**0.01115**	0.04322	0.04118	0.04170	0.03246
PANASONIC	0.044524	**0.01086**	0.01809	0.02114	0.02605	0.03873
SAINSBURY	0.014848	0.02298	0.01425	0.01732	0.02259	**0.01231**
SHISEIDO	0.063066	0.10319	0.05848	0.05783	**0.05050**	0.06252
SUBARU	0.044433	0.06091	**0.02083**	0	0.04449	0.03800
SUZUKI	**0.00745**	0.08093	0.06445	0.04614	0	0.01017
TENCENT	0.026062	0	0.09871	**0.00723**	0.01701	0.02464
TESCO	0.034039	0.01888	0.02890	0.04154	**0.00306**	0
TESLA	0.040500	**0.01702**	0.03452	0.04737	0.07283	0.05365
TOYOTA	0.01268	0.01963	0	0.02756	textbf0.01102	0
UBISOFT	**0.01128**	0	0.03468	0.02992	0.02372	0.08165
WALMART	0.018379	**0.00927**	0.00949	0.02483	0.01310	0.02351
XEROX	0.023120	0.06069	**0.02131**	0.03093	0.02727	0
YAMAHACO	0.020637	**0.00956**	0.04052	0.01841	0.01740	0.01586
YAMAHAMO	0.027458	0	**0.01588**	0	0.02108	0.02177

Table 5.13 presents the summary statistics for the risk values obtained from 50 iterations for each company, considering the four algorithms introduced in Chapter 4 and the two GP algorithms introduced in this chapter.

Table 5.13: Summary statistics of risk across all 50 GP runs and all 42 companies. Boldface is used to denote the best value for each statistic.

Algorithm	*Average*	*Median*	*StDev*	*Maximum*	*Minimum*
GP-FA	0.029	0.026	**0.024**	0.21	0.0007
GP-SA	0.034	0.022	0.034	0.25	0.0003
GP-TA	0.027	0.018	0.025	0.18	0.0006
GP-FASATA	0.030	0.023	0.027	0.17	0.00007
STGP-FASATA	0.028	0.022	0.026	0.19	**0.000067**
STGP-FASATA-S	**0.026**	**0.017**	0.026	**0.126**	0.00046

When examining the summary statistics presented in Table 5.13, along with the statistical tests between the control algorithm STGP-FASATA-S and the other five GP benchmarks in Table 5.14, a similar observation to that of the rate of return emerges. The STGP-FASATA-S algorithm demonstrates lower mean and median values for risk while exhibiting similar standard deviation values with the other algorithms, except GP-SA, while GP-TA performs with the lowest standard deviation. These results are followed by STGP-FASATA, which has a lower mean and median risk than the GP-FA, GP-SA, and GP-FASATA algorithms but not from GP-TA, in contrast to the rate of return results.

Table 5.14 shows the KS results when control algorithm is again STGP-FASATA-S in comparison to the five other GP benchmarks. The p-values generated when comparing the

Table 5.14: Kolmogorov-Smirnovtest p-values on risk of the proposed STGP-FASATA-S algorithm against the 5 GP benchmarks. Statistical significance changes are based on the Holm-Bonferroni correction. Statistically significant differences at the 5% level are indicated in boldface.

Algorithm	*STGP-FASATA-S p-values*	Rank	*Significance level*
GP-FA	**1.34E-44**	4	0.025
GP-SA	**2.16E-37**	5	0.05
GP-TA	**7.38E-47**	3	0.016
GP-FASATA	**1.82E-84**	1	0.01
STGP-FASATA	**1.55E-64**	2	0.0125

distribution of STGP-FASATA-S to the distribution of the benchmarks signify a significantly statistical difference in risk performance between the algorithms, accounting for the Holm-Bonferroni correction.

5.4.4 Results of each market

Similar to the previous chapter, it is crucial to assess the performance of each algorithm across different market groups due to the considerable variation in price series among companies and analyse the performance of the genetic programming (GP) algorithms across diverse market profiles.

In order to achieve this, we used the market groups mentioned in Chapter 4, wherein we examined the initial and final prices of the test set for each company and calculated the corresponding return value, creating three distinct groups. To remind the reader, the first group (Group 1) includes companies whose price experienced a long-term increase of at least 20%, the second group (Group 2) includes companies whose price experienced a long-term increase of between 0% and 19.99%. In contrast, the last group (Group 3) includes those companies whose price experienced a long-term decrease.

Following the aforementioned classification criteria, we assigned 19 companies to Group 1, 14 companies to Group 2, and 9 companies to Group 3.

We proceeded to calculate the average values of each metric (Sharpe ratio, rate of return, and risk) for each GP algorithm across the datasets of each group. Table 5.15 illustrates the results. Upon examining the table, it is evident that the proposed STGP-FASATA-S algorithm demonstrates strong performance in terms of the aggregate metric of the Sharpe ratio for Group 1 and Group 3. Conversely, in Group 2, the GP-FASATA algorithm emerges as the frontrunner. Regarding the rate of return, the strongly typed algorithms achieve the highest value for Group 1, while STGP-FASATA-S continues to be the winning algorithm in Groups 2 and 3. On the other hand, STGP-FASATA's performance in Group 3 is close to zero and negative, as is the case for GP-TA. Finally, in terms of risk, STGP-FASATA consistently exhibits the lowest value across the two first groups. GP-TA performs with the lowest risk in Group 3, with the greatest disparity observed when compared to the risk value of GP-FASATA.

Overall, it is worth noting that STGP-FASATA-S and STGP-FASATA, the strongly-typed genetic programming architectures, exhibit better performance when there are positive price movements in the market. This observation aligns with the finding that STGP-FASATA-S and STGP-FASATA are performing strongly in terms of the aggregate metric of the Sharpe ratio, which combines the rate of return and risk. In conclusion, STGP-FASATA-S demonstrates a competitive performance, particularly in those scenarios with more extreme market condi-

Table 5.15: Separated average results per metric per trend group.

Market	*Algorithm*	*Sharpe ratio*	*Rate of return*	*Risk*
Group 1 (>20%)	GP-TA	2.7	0.012	0.029
	GP-FASATA	2.49	0.014	0.029
	STGP-FASATA	2.55	**0.017**	**0.023**
	STGP-FASATA-S	**3.3**	**0.017**	0.024
Group 2 (0% - 19.99%)	GP-TA	1.89	0.0035	0.039
	GP-FASATA	**4.86**	0.006	0.030
	STGP-FASATA	3.7	0.008	**0.029**
	STGP-FASATA-S	1.8	**0.02**	**0.029**
Group 3 (<0%)	GP-TA	-1.32	-0.001	**0.025**
	GP-FASATA	1.43	0.006	0.035
	STGP-FASATA	2.26	-0.001	0.033
	STGP-FASATA-S	**3.5**	**0.011**	0.030

tions (favourable or negative), showcasing its potential as a promising algorithm for generating positive returns while managing risk effectively.

5.4.5 Non-GP Benchmarks

STGP-FASATA-S compared to machine learning benchmarks

In Chapter 4, GP-FASATA statistically outperformed the MLP (Multi-Layer Perceptron) algorithm in Sharpe ratio and risk but not in rate of return. Moreover, the GP algorithm did not statistically outperform the SVM (Support Vector Machine), eXtreme Gradient Boosting (XG-Boost), and Long short term memory (LSTM) algorithms in terms of rate of return and/or risk, either. Thus, we investigate the results arising between the MLP, SVM, XGBoost, and LSTM against STGP-FASATA-S to find if the latter can *statistically* outperform the machine learning benchmarks. We again present the performance results in Table 5.16.

Upon examining Table 5.16, it is evident that the average values of MLP for the 42 companies yield a Sharpe ratio of 0.30, a rate of return of 0.009 and for risk 0.041. Notably, these values contrast significantly with those obtained from the GP algorithms, which exhibit considerably higher values for the Sharpe ratio and risk values. Statistical analysis using the Kolmogorov-Smirnov test reveals that the distribution of the STGP-FASATA-S algorithm is statistically different than MLP's distribution regarding Sharpe ratio, rate of return, and risk values.

As mentioned in the Chatper 4, the average Sharpe ratio is 0.37, the rate of return is 0.01, and the risk is 0.038. This time, the KS tests confirm that the distribution of STGP-FASATA-S is statistically different than the distribution of SVM in all financial metrics assessed.

Table 5.16: Comparison of average values for STGP-FASATA-S and algorithmic benchmarks.

Algorithm	**Sharpe ratio**	**Rate of return**	**Risk**
MLP	0.30	0.009	0.041
SVM	0.37	0.010	0.038
XGBoost	0.36	0.010	0.041
LSTM	0.23	0.006	0.043
STGP-FASATA-S	**3.78**	**0.021**	**0.026**

Table 5.17: Kolmogorov-Smirnov test p-values for all financial metrics of the proposed STGP-FASATA-S algorithm measured against the 4 machine learning benchmarks. Statistical significance changes are based on the Holm-Bonferroni correction. Statistically significant differences at the 5% level are indicated in boldface.

Financial Metric	*Algorithm*	*GP-FASATA p-values*	**Rank**	*Significance level*
Sharpe ratio	MLP	**2.21e-11**	1	0.05
	SVM	**2e-11**	3	0.016
	XGBoost	**5.19e-14**	4	0.0125
	LSTM	**5e-14**	2	0.025
Rate of return	MLP	**0.0007**	4	0.0125
	SVM	**0.0007**	1	0.05
	XGBoost	**7.4e-10**	3	0.016
	LSTM	**3.7e-9**	2	0.025
Risk	MLP	**3.9e-6**	1	0.05
	SVM	**0.008**	4	0.0125
	XGBoost	**0.0001**	2	0.025
	LSTM	**0.0003**	3	0.016

For the XGBoost algorithm, again, we observe average values of 0.36 for the Sharpe ratio, 0.010 for the rate of return, and 0.041 for the risk. STGP-FASATA-S performs better in the financial metrics and its distribution is statistically significant different than XGBoost in all financial metrics.

Regarding the LSTM algorithm, we recall the average values of 0.23 for the Sharpe ratio, 0.006 for the rate of return, and 0.043 for the risk. STGP-FASATA-S does outperform the algorithm in the three key financial metrics, while the novel GP algorithm's distribution is statistically different than LSTM's distribution for Sharpe ratio, rate of return, and risk.

In summary, the results again indicate that all algorithms, namely MLP, SVM, XGBoost, and LSTM, exhibit lower average values for the financial metrics than the GP algorithms. The statistical tests further support the conclusion that the distributions of the ML benchmarks are statistically significant different than the distribution of the STGP-FASATA-S algorithm regarding the Sharpe ratio, rate of return, and risk, something GP-FASATA did not achieve in Chapter 4.

This observation highlights the relative disadvantage(s) of the machine learning benchmarks when it comes to algorithmic trading and the development of strategies that consider both returns and risk metrics. The GP algorithms, specifically STGP-FASATA-S, outperforms the ML benchmarks in these aspects, making it more suitable for generating effective trading strategies.

5.4.6 Summary of findings

In conclusion, based on the findings presented in Tables 5.5 - 5.15, the results can be summarised into two categories in relation to the performance of GP variants and their performance in relation to other benchmark algorithms.

When comparing the GP variants amongst themselves, we observe that:

- The proposed STGP-FASATA-S algorithm demonstrates strong performance across all three financial metrics, namely the Sharpe ratio, rate of return, and risk. Additionally, it outperforms the benchmark GP algorithms in terms of the performance metrics.

- The strongly-typed architecture of the proposed STGP-FASATA-S and the STGP-FASATA algorithms enable them to conduct a more effective search of the space encompassing fundamental analysis, sentiment analysis, and technical analysis indicators. This advantage is not present in the GP-TA and GP-FASATA algorithms, which allow all indicator types in its terminal set without the strongly-typed architecture.
- The novel fitness function of the STGP-FASATA-S algorithm can improve the financial performance of the GP.

Regarding the comparison of STGP-FASATA-S to the non-GP benchmarks, we observe the following:

- The STGP-FASATA-S algorithm outperforms MLP, SVM, XGBoost, and LSTM, and their distributions are shown to be statistically significant different, as indicated by the Kolmogorov-Smirnov tests. Furthermore, the algorithms show higher risk associated, something evident from the lower Sharpe ratio values.

5.5 Conclusion and Further Experiments

In conclusion, this chapter aimed to investigate and compare the performance of the trading strategies developed by two GP algorithms that integrate fundamental, sentiment, and technical analysis indicators. We introduced a novel strongly-typed GP algorithm that assists where each of the FA, SA, and TA terminal sets will be assigned a distinct branch, allowing for better exploration and exploitation within each indicator type's respective search space. This enhancement is anticipated to yield even more robust and efficient trading strategies, ultimately contributing to advancing algorithmic trading techniques.

The results indicated that our proposed STGP-FASATA-S algorithm exhibited competitive performance over the five GP benchmark algorithms and the four ML benchmark algorithms in the three financial metrics. This finding underscores the potential effectiveness of our approach in developing robust trading strategies.

Combining the indicators can enhance the models' knowledge and create financially more advantageous trading strategies. Moreover, based on our analysis, it is not profitable enough to simply combine the different types of indicators, as GP-FASATA does, and a strongly-typed architecture is essential in achieving improved performance in terms of Sharpe ratio, which combines both rates of return and risk. Finally, the use of the novel fitness function significantly improved results across all three metrics of Sharpe ratio, rate of return, and risk.

Although STGP-FASATA-S demonstrated better results than the benchmark algorithms and it is able to outperform all other GP algorithms (including individual analysis types, e.g. GP-TA), we are now motivated to investigate whether we can further improve its performance on rate of return and risk. In the next chapter, we will address by proposing a novel genetic operator that encourages active trading.

Chapter 6

Strongly-typed genetic programming variant

6.1 Chapter motivation

Considering the effectiveness of STGP-FASATA and STGP-FASATA-S in Chapter 5, we are motivated to investigate the creation of operators that may increase the profitability of the genetic programming algorithms, especially when considering the median value of the rate of return and the mean value of the risk of the STGP-FASATA-S algorithm. Thus, in this chapter, we introduce another strongly-typed GP algorithm with the previously mentioned novel fitness function, but this time, proposing to incorporate a novel GP operator. This GP operator encourages active trading by injecting into the GP population trees that are able to perform a high number of trades while achieving high profitability at low risk at the same time.

More specifically, we are interested in encouraging the algorithm to follow an active trading approach that actively monitors and analyses market conditions to identify short-term trading opportunities. The advantage of such trading strategies is that they can leverage price movements over relatively short timeframes, thereby increase their profitability. To achieve this, we create a new GP operator that works alongside crossover and mutation. This novel operator identifies the FA, SA, and TA subtrees, which are using highly active trading strategies to achieve high profitability at low risk and injects them into the following generation by combining them into a new tree.

Our objective is to showcase the proposed GP algorithm's capability of generating unique and profitable trading strategies that leverage information from all three analysis types. We compare its performance to STGP-FASATA-S, which is the best performed GP variant in Chapter 5 and STGP-FASATA since it is an algorithm incorporating the strongly-typed architecture. Experiments take place on the same 42 international companies' stocks, and results are reported across the same three financial metrics, namely Sharpe ratio, rate of return, and risk.

The rest of the chapter is organised similarly with Chapters 4 and 5 as Section 6.2, which details the methodology employed in our study, Section 6.3 with the experimental setup, and the results and analysis of the study are presented in Section 6.4. Finally, Section 6.6 summarises the main findings of this thesis contribution and explores potential future scientific work.

6.2 Methodology

This chapter presents the ActTrade algorithm, a novel genetic programming algorithm incorporating a strongly-typed architecture.

The research methodology includes three main components. Firstly, Section 6.2.1 offers an overview of the GP methodology, encompassing the model representation, while the novel GP operator is introduced in Section 6.2.2. Subsequently, Section 6.2.3 delves into the trading signals and trading strategy employed. Finally, Section 6.2.4 details the fitness function and metrics that will be taken into consideration during the evaluation process.

6.2.1 Model representation

As in Chapter 5, the function nodes are based on a 3-input `AND` function with a Boolean output called `3-AND`, as well as binary `AND`, `OR`, Greater than (`GT`) and Less than (`LT`) functions, with different variants allowing for each indicator type. To remind the reader, the `3-AND` function takes 3 children, with the first being of type FA, the second of type SA, and the third of type TA, with all of the branches taking relevant types of functions. All three branches of the `3-AND` function need to evaluate TRUE in order for the function to yield a TRUE outcome. The function set used in our algorithm is summarised in Table 5.1 of Chapter 5 designed to ensure that the algorithm generates models that fully utilise all indicator types while enforcing type consistency.

As in Chapter 5, the strongly-typed architecture of the novel algorithm has the advantage of fully utilising the search space of each individual indicator type. The three branches corresponding to the FA, SA and TA indicators are connected using the `3-AND` function at the root of the tree. This integration creates a foundation for better exploration and exploitation of the search space. As a result, the model can generate more diverse, effective, and adaptable trading strategies.

6.2.2 GP operators

This section is separated into two parts: the traditional GP operators and the novel GP operator. The traditional GP operators have been introduced in Chapter 4, while the novelty of this chapter lies in Section 6.2.2, where we introduce the new GP operator.

Traditional GP operators The traditional GP operators and function set presented in Chapter 6 are parts of this chapter, too. More information can be found in Chapters 3, 4, and 5.

A novel GP operator for active trading As mentioned in the introduction of this chapter, one of our motivations is to create an algorithm that follows highly active trading strategies, as this offers a potential for higher profits. Thus, we focused on creating offspring that seek to maximise the number of trades, as well as the Sharpe ratio.

In order to achieve this, we undertake the following process: at every generation, we iterate through each individual in the population and place its three subtrees (FA, SA, TA) into three respective lists, one for each analysis type. We then calculate the number of trades and the Sharpe ratio for each subtree. Once we have done this for all trees in the current generation, we identify the FA subtree with the highest number of trades, as well as the FA subtree with the highest Sharpe ratio. This process is repeated for the SA and TA subtrees. It is also worth noting that in order to encourage smaller and easier to read trees, when there are subtrees that perform the same number of trades, we choose the one with the smaller size. Next, we create three offspring by combining subtrees that maximise the Sharpe ratio and the number of trades. More specifically, the first offspring consists of the FA subtree with the highest Sharpe ratio (from all FA subtrees), the SA subtree with the highest number of trades (from all SA subtrees), and the TA subtree with the highest number of trades (form all

TA subtrees). These three subtrees are placed under the ternary `3-AND` function and copied into the next generation. The second offspring follows a similar process, whereby the SA subtree with the highest Sharpe ratio forms a tree along with the FA and TA subtrees with the highest number of trades. Finally, the third offspring consists of the TA subtree with the highest Sharpe ratio, along with the FA and SA subtrees with the highest number of trades. It should be noted that this operator is applied at every generation during training. This means that the above three generated offspring will be copied into the next generation before crossover and mutation takes place. This process is summarised in Algorithms 1 - 3.

By adopting this approach, the GP operator imparts two distinct characteristics that are desired in the individuals of the population, namely, a high number of trades and a high Sharpe ratio.[1] This strategy aims to enhance the quality of the evolving trees in subsequent generations, promoting individuals that exhibit both traits to achieve more effective and profitable trading outcomes.

[1]During the early experimentation phase, we considered alternatives to this approach, such as creating trees where all three subtrees would either maximise the number of trades or maximise the Sharpe ratio. We found that in those cases, only a limited number of cases would evaluate TRUE and subsequently perform a trading action because the root of these trees was the function `3-AND`. As a result of this highly restrictive situation, the GP was overly passive and would frequently choose not to trade at all. Other alternatives we considered were (i) using an `OR` function as the root node, but this would not guarantee that all three indicator types would be simultaneously considered during a trading action, and (ii) using a non strongly-typed GP solution. Neither of these two approaches yielded very good financial performance. We have left any further investigation of alternative approaches as future work.

Algorithm 1 Active trading genetic operator procedure - Part1

```
Require: Population of individuals, Analysis types: FA, SA, TA
 1: Initialise empty lists for FA, SA, and TA subtrees.
 2:
 3: for each individual in the population do
 4:     Extract FA, SA, and TA subtrees from the individual.
 5:     Append these subtrees to the respective lists for the analysis type.
 6: end for
 7:
 8: for each subtree in the corresponding analysis type (FA, SA, TA) list do
 9:     Calculate the number of trades and Sharpe ratio.
10: end for
11:
12: for each list (FA subtrees, SA subtrees, TA subtrees) do
13:     Identify the subtree with the highest number of trades (max_trades_subtree) and
        the subtree with the highest Sharpe ratio (max_sharpe_subtree).
14: end for
15:
```

Algorithm 2 Active trading genetic operator procedure - Part2

```
1: Create three offspring (named Offspring1, Offspring2, Offspring3):
2: for each offspring do
3:     Select corresponding subtrees:
4:     if Offspring1 then
5:         FA subtree: max_sharpe_subtree from FA subtrees
6:         SA subtree: max_trades_subtree from SA subtrees
7:         TA subtree: max_trades_subtree from TA subtrees
8:     else if Offspring2 then
9:         FA subtree: max_trades_subtree from FA subtrees
10:        SA subtree: max_sharpe_subtree from SA subtrees
11:        TA subtree: max_trades_subtree from TA subtrees
12:    else
13:        # Offspring3
14:        FA subtree: max_trades_subtree from FA subtrees
15:        SA subtree: max_trades_subtree from SA subtrees
16:        TA subtree: max_sharpe_subtree from TA subtrees
17:    end if
18:
19:    Combine the three subtrees (FA, SA, TA) under the ternary 3-AND function and enter
       them into the population of the next generation.
20: end for
```

Selection

In the previous section, we explained how our proposed genetic operator creates three offspring, which are subsequently placed into the population of the next generation. We call them *Offspring1, Offspring2,* and *Offspring3*. As these three individuals might not always make a large impact on the search, we decided to also probabilistically allow them to act as parents during crossover and mutation. In this way, we allow their genetic material to be copied into more trees in the population.

To achieve the above, we assign four probabilities p_1, p_2, p_3, and p_4,[2] to select the first parent for crossover, and the single parent for mutation. More specifically, with probability p_1, we use tournament selection, wherein k individuals (trees) are randomly selected from the population. Then, the individual with the highest fitness value in the tournament is selected as a parent to undergo crossover/mutation. In addition, we assign a probability p_2, p_3, and p_4, to each one of Offspring1, Offspring2, and Offspring3, respectively. In this way, the proposed genetic operator can also affect the creation of new offspring by carrying part of its genetic material to the new offspring.

For the second parent, we always use tournament selection, i.e. $p_1 = 1$. Algorithm 3 summarises the selection process.

To conclude, the proposed GP algorithm is a strongly typed GP that aims to maximise the Sharpe ratio. The strongly typed structure of the GP enables more effective exploration and utilisation of all three indicator types. At the same time. The proposed operator allows for offspring to be created that focus both on maximising the Sharpe ratio, as well as maximising the number of trades.

[2]The sum of $p_1 + p_2 + p_3 + p_4$ is always equal to 1.

Algorithm 3 Parents selection procedure

```
Require: Probabilities $p_1$, $p_2$, $p_3$, $p_4$, Offspring1, Offspring2, Offspring3
1: if Selecting the first parent for crossover or parent for mutation then
2:     With probability $p_1$: Perform tournament selection to select parent
3:     With probability $p_2$: Select Offspring1 as parent
4:     With probability $p_3$: Select Offspring2 as parent
5:     With probability $p_4$: Select Offspring3 as parent
6: else
7:     Perform tournament selection to select the second parent for crossover
8: end if
```

6.2.3 Trading signals and trading strategy

The trading strategy is the same as in Chapter 5, where the trading signals are generated using the GP trees, where the result of the root `3-AND` function represents a TRUE/FALSE value. The strongly-typed algorithms are all able to produce the same sample GP tree with STGP-FASATA-S, also included in Figure 6.1. We remind the reader of the said tree with Figure 6.1.

6.2.4 Fitness function and Metrics

After improving the results from STGP-FASATA to STGP-FASATA-S in Chapter 5, the same fitness function of STGP-FASATA-S is implemented in ActTrade. Our analysis still considers the return, risk and Sharpe ratio metrics to evaluate the algorithms' respective performances. As in Chapters 4 and 5, first we find the list of returns (R) from each trade. The rate of return is the mean value of this list, while the risk is the standard deviation of the list. The Sharpe

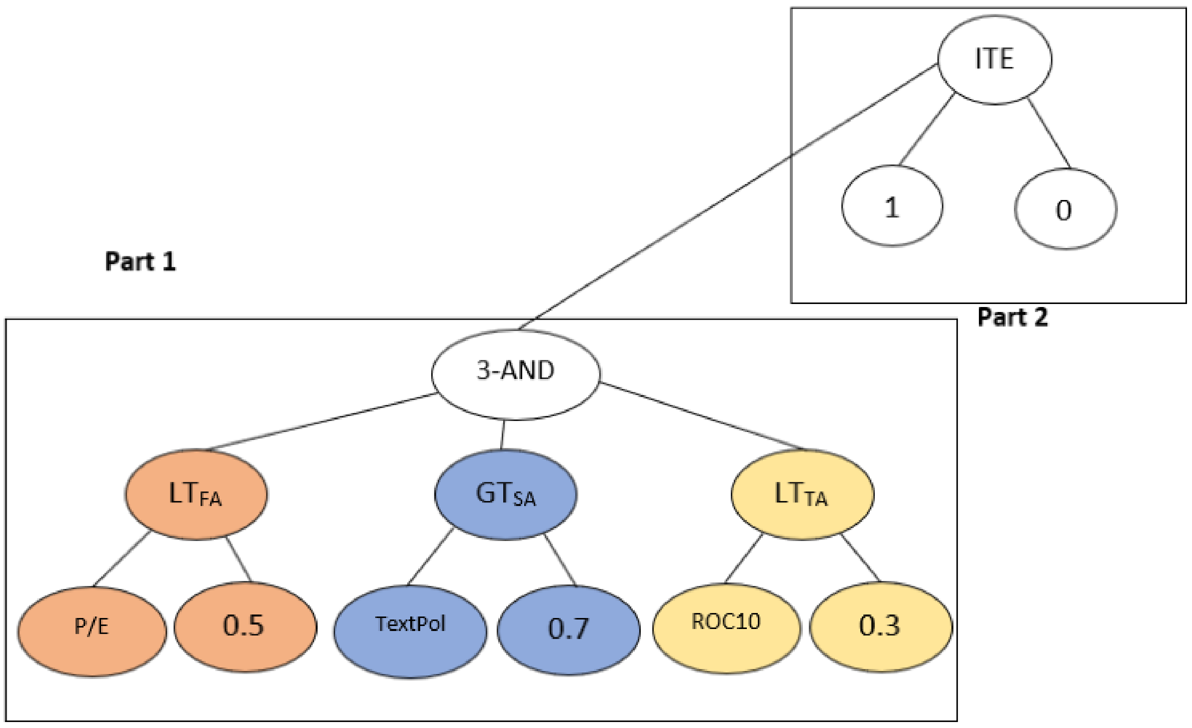

Figure 6.1: ActTrade sample tree. The first child of the 3-AND function is enforced to be FA-related, the second child is SA-related, and the third is enforced to be TA-related. This sample tree checks if the P/E indicator is less than the ERC of 0.5, and if the TEXTpol indicator is greater than the ERC 0.7, and if the ROC10 indicator is less than the ERC 0.3. If all three of them are TRUE, then the recommendation will be to buy (1), otherwise it will be to hold (0).

ratio is calculated as the ratio of the expected value of the excess return compared to the risk free return over the risk. Again, the *fitness function*, f_{sum}, is determined by the weighted sum of the FA, SA, and TA subtrees and the complete tree comprising the three subtrees and it aims to emphasise the contribution of each indicator in enhancing the trading performance of the algorithm. More specifically, it is defined as the summation of the Sharpe ratio (SR^C) of the complete tree, and the Sharpe ratios $(SR^{FA}$, SR^{SA}, $SR^{TA})$ of the FA, SA, and TA subtrees, with weights w_c, w_{FA}, w_{SA}, and w_{TA}, respectively.

6.3 Experimental Setup

As in Chapter 5, in the experimental setup section, we will briefly mention the data collection again in Section 6.3.1, and thereafter introduce the GP benchmarks of this chapter in Section 6.3.2. Finally, we present the parameter tuning process for this chapter in Section 6.3.3.

6.3.1 Data

The data collection process is the same as in Chapters 4 and 5, and can be found in more detail in Section 4.3.

6.3.2 Benchmarks

We call our proposed GP algorithm ActTrade, due to its active trading operator. We compare its performance to the proposed GP variants STGP-FASATA and STGP-FASATA-S strongly-typed algorithms, introduced in Chapter 5. We present all of these benchmarks next.

- **STGP-FASATA** is a strongly-typed GP algorithm that combines fundamental, sentiment and technical analysis indicators. This algorithm does not use our novel fitness function nor the novel active trading genetic operator. It thus acts as a valuable benchmark to enable us to understand the added value of the aforementioned innovations of the proposed GP.

- **STGP-FASATA-S** is similar to the aforementioned STGP-FASATA algorithm, with the addition of the fitness function that our proposed ActTrade also uses (summation of Sharpe ratio fitness functions). Thus, the only difference between STGP-FASATA-S and ActTrade is the use of the active trading operators, serving as a benchmark to reveal the added value of this GP operator.

Furthermore, Chapters 4 and 5 included four machine learning and one financial benchmark. However, since we have shown these benchmarks being statistically outperformed for all financial metrics in Chapter 5 by STGP-FASATA-S, we do not include them in this chapter.

6.3.3 Parameter Tuning

Table 6.1: GP Parameters for the eight GP algorithms.

GP Parameters	
Population size	1000
Crossover probability	0.95
Mutation probability	0.05
Generations	50
Tournament size	4
Maximum tree depth	6
Tournament probability (p_1)	0.4
Active trading offspring probability (p_2, p_3, p_4)	(0.2, 0.2, 0.2)

Table 6.1 presents the GP parameters used in the experiments of this article. This section is a little different from the corresponding parameter tuning detailed in Sections 4.3.3 and 5.3.3. The current section contains the previously discussed parameters and also the tournament probability (p_1), and active trading offspring probability (p_2, p_3, p_4). To find the optimal GP parameters, a two-step grid search was conducted in the validation set for the parameters of population size, crossover probability, number of generations, tournament size, and maximum tree depth. Furthermore, the probabilities p_1, p_2, p_3, and p_4 for selecting the first parent

for crossover or mutation (p_1: selecting tournament; p_2, p_3, and p_4 are for the three generated offspring from the active trading genetic operator. As outlined in the previous chapters, the trading parameters d and r were held constant at 30 and 0.05, respectively, to expedite the tuning process.

The second step of parameter tuning can be found in Chapter 4.

6.4 Results and Discussion

In this section, we present our experiments' results in comparing the proposed ActTrade with the benchmarks discussed in Section 6.3.2. We follow the same process as for the experiments detailed in Chapters 4 and 5, by running 50 independent runs for each one of the 42 companies for each GP algorithm. This way, each run returned a distinct trading strategy, which we then evaluated on the test set. Tables 6.2 and 6.3 showcase the mean Sharpe ratio values of the runs in each company, while the median Sharpe ratio values of all the runs across all companies can be found in Table 6.4. The median values for the rate of return and risk can be found in Sections 6.4.2 and 6.4.3, respectively.

To be consistent with the previous chapters, we again validate our findings by performing a two-sample Kolmogorov-Smirnov (KS) test on all runs across companies for each algorithm. We again address the issue of multiple comparisons by applying the Holm-Bonferroni correction for pairwise comparisons. We individually compared the ActTrade algorithm with the two GP algorithmic benchmarks (STGP-FASATA and STGP-FASATA-S) for each financial metric.

Consequently, for a 5% significance level, the following p-values apply: the first-ranked p-value should be less than 0.025, and the second-ranked p-value should be less than 0.05. This approach ensures that the probability of obtaining a false positive result is maintained below a predetermined threshold, resulting in more reliable statistical findings.

6.4.1 Sharpe ratio

Tables 6.2 and 6.3 present the median Sharpe ratio values over the 50 runs for the 42 companies. As we can observe, the algorithm that has the most optimal Sharpe ratio values is ActTrade with 16 companies, followed by STGP-FASATA (14) and STGP-FASATA-S (13). We again notice that some of the values are close to each other, while in some cases where STGP-FASATA has a negative value, we observe that STGP-FASATA-S and ActTrade algorithms choose not to trade at all.

Table 6.4 displays summary statistics over each company's 50 GP runs across the seven GP algorithms. We have excluded in the statistic's calculation runs with a Sharpe ratio equal to 0, as in those cases, the algorithms did not perform any trading.

As we can see from Table 6.4, the proposed ActTrade algorithm has the highest mean and median values, indicating a strong risk-adjusted performance compared to the other algorithms and highlighting its robustness and consistency in delivering favourable risk-adjusted returns. Generally, the GP algorithms of this chapter experience higher mean values, which indicates the advantage of the architecture in comparison to the non strongly-typed GP variants we show in the previous chapters. ActTrade's standard deviation is also the lowest.

To better understand the above results, we perform pairwise comparisons of the ActTrade (best/control algorithm due to having the best average and median values) with each other GP algorithm. We present the test results in Table 6.5. As mentioned in Chapter 4, we use the Kolmogorov-Smirnov (KS) non-parametric test, along with the Holm-Bonferroni correction, to account for the multiple pairwise comparisons (two in total). As we can observe, the distribution of the proposed ActTrade is statistically significant different than the distributions

Table 6.2: Median values for Sharpe ratio per company. Boldface is used to denote the best value for the particular dataset. First 21 companies.

Company	***STGP-FASATA***	***STGP-FASATA-S***	***ActTrade***
AAPL	2.3797	**4.9321**	2.677
ADIDAS	**0.1836**	0	0
ALIBABA	0.833	2.5453	**11.814**
AMAZON	**1.468**	0	0
ASUS	-0.5085	-0.5439	**6.3521**
ATVI	0.5166	0.3749	**2.698**
BLACKB	1.3265	**1.7034**	-0.2127
COKE	**1.2097**	1.0946	1.1638
EBAY	-0.2471	**0.4302**	0.1362
ESTEE	2.2770	**3.8917**	2.123
FORD	**6.4874**	0	3.2563
FUJIFILM	0.1352	-1.7363	**0.1399**
FUJITSU	2.4430	**24.514**	12.9103
GM	**-0.6598**	0	0
GOOGL	0.0886	-0.1268	**5.994**
HITACHI	**-1.4215**	0	0
HONDA	2.1395	**2.1479**	0
HSBC	0.3209	0	**0.0372**
HYUNDAI	0.3972	1.124	**2.3572**
IBM	0.1098	**3.211**	1.7280
INTC	**2.2122**	0	-0.0318

Table 6.3: Median values for Sharpe ratio per company. Boldface is used to denote the best value for the particular dataset. Last 21 companies.

Company	***STGP-FASATA***	***STGP-FASATA-S***	***ActTrade***
KERING	**1.16502**	0.3959	0
KODAK	1.26547	1.0454	**1.4971**
MCDON	**-0.0125**	0	-1.2986
NESTLE	**2.17719**	-0.1383	-0.0065
NFLX	2.44843	**5.94198**	4.14847
NINTENDO	0.35842	**0.71975**	0.5744
NYT	0.18615	**0.45961**	0.24952
PANASONIC	0.09342	**0.2917**	0.1192
SAINSBURY	1.02793	**1.4162**	**1.4162**
SHISEIDO	-0.4704	**1.89246**	-0.6254
SUBARU	-0.3314	0.090107	**2.0503**
SUZUKI	0	6.32750	**9.1026**
TENCENT	0.98139	0.3919	**3.7368**
TESCO	**15.4991**	0	-14.513
TESLA	1.22883	1.4788	**2.031**
TOYOTA	**3.0060**	0	0
UBISOFT	**2.03319**	-0.432	0
WALMART	0.31503	0.12854	**2.2318**
XEROX	**1.4751**	0	0.31883
YAMAHACO	1.1658	1.1852	**2.2302**
YAMAHAMO	-0.209	-0.13029	**0.1512**

Table 6.4: Summary statistics of Sharpe ratio across all 50 GP runs and all 42 companies. Boldface is used to denote the best value for each statistic.

Algorithm	*Average*	*Median*	*StDev*	*Maximum*	*Minimum*
STGP-FASATA	3	0.64	9.9	144.32	**-8.81**
STGP-FASATA-S	3.78	1.07	11.57	83.59	-27.80
ActTrade	**3.9**	**1.318**	**8.22**	64.35	-14.51

of STGP-FASATA and STGP-FASATA-S, with p-values (second column) significantly lower than the corresponding significance level values (fourth column).

Table 6.5: Kolmogorov-Smirnov test p-values on Sharpe ratio of the proposed ActTrade algorithm against the 2 GP benchmarks. Statistical significance changes are based on the Holm-Bonferroni correction. Statistically significant differences at the 5% level are indicated in boldface.

Pairwise comparison	*KS test p-value*	Rank	*Significance level*
ActTrade vs STGP-FASATA	**9.92E-33**	1	0.025
ActTrade vs STGP-FASATA-S	**5.04E-06**	2	0.05

In conclusion, ActTrade has the highest average and median values of Sharpe ratio, and its distribution is statistically significant different than that of the benchmarks. Thus, we can state ActTrade statistically outperforms the other 2 GP variants, subsequently outperforming the GP non-strongly GP algorithms that have already been outperformed in Chapter 5. Additionally, ActTrade operator generates an average of 21 trades during the test period, while

STGP-FASATA and STGP-FASATA-S generate an average of 17 trades each. Thus, showcasing the effectiveness of the new operator in forming high performing models.

6.4.2 Rate of return

Tables 6.6 and 6.7 present the median rate of return values over the 50 runs for the 42 companies. The algorithm that has the most optimal rate of return values is again ActTrade with 16 companies, followed by STGP-FASATA (15) and STGP-FASATA-S (14). We notice that some values are closer to each other than their value difference in Section 6.4.1.

Table 6.8 summarises the results by presenting the mean, median, standard deviation, maximum and minimum values across all 50 runs for each company. As we can observe, the proposed ActTrade again has the best mean value and a median value very close to the best observed (0.0145 vs 0.015). Its standard deviation (0.031) is comparable to the lowest value of 0.028, showing that it has relatively low volatility. The maximum rate of return value is the same for all algorithms, and ActTrade has the lowest minimum value for the rate of return.

Table 6.8: Summary statistics of rate of return across all 50 GP runs and all 42 companies. Boldface is used to denote the best value for each statistic.

Algorithm	***Average***	***Median***	***StDev***	***Maximum***	***Minimum***
STGP-FASATA	0.0135	0.0125	**0.028**	**0.11**	-0.18
STGP-FASATA-S	0.021	**0.015**	0.035	**0.11**	-0.087
ActTrade	**0.0225**	0.0145	0.031	**0.11**	**-0.06**

Again, to evaluate the significance of the above results, we performed the Kolmogorov-Smirnov non parametric test, along with the Holm-Bonferroni correction, as presented in

Table 6.6: Median values rate of return per company. Boldface is used to denote the best value for the particular dataset. First 21 companies.

Company	*STGP-FASATA*	*STGP-FASATA-S*	*ActTrade*
AAPL	0.0240	0.0285	**0.0267**
ADIDAS	**0.0041**	0	0
ALIBABA	0.0248	**0.0329**	0.0325
AMAZON	**0.0236**	0	0
ASUS	0.0013	0.00092	**0.0264**
ATVI	0.0120	0.00785	**0.0259**
BLACKB	**0.0307**	**0.0307**	-0.0183
COKE	**0.01310**	0.00839	0.0118
EBAY	-0.0127	**0.0102**	0.00261
ESTEE	**0.01170**	0.01004	0.01092
FORD	0.0162	0	**0.01892**
FUJIFILM	0.0036	-0.0308	**0.0039**
FUJITSU	0.08038	**0.1038**	0.08283
GM	-0.03538	**0**	**0**
GOOGL	0.00211	-0.0033	**0.0167**
HITACHI	-0.020306	**0**	**0**
HONDA	**0.01766**	0.01652	0
HSBC	**0.00192**	0	0.0004
HYUNDAI	0.01908	0.034	**0.0355**
IBM	0.00756	**0.0466**	0.0250
INTC	**0.0185**	0	-0.0026

Table 6.7: Medain values rate of return per company. Boldface is used to denote the best value for the particular dataset. Last 21 companies.

Company	*STGP-FASATA*	*STGP-FASATA-S*	*ActTrade*
KERING	**0.0297**	0.0191	0
KODAK	0.0331	0.0227	**0.0534**
MCDON	**-0.0125**	0	-0.0563
NESTLE	**0.0076**	-0.00213	0.000113
NFLX	0.0437	**0.0590**	0.0556
NINTENDO	0.0113	0.0069	**0.0179**
NYT	0.00772	**0.01596**	0.00949
PANASONIC	0.00453	**0.01152**	0.00601
SAINSBURY	0.01676	**0.01967**	0.01676
SHISEIDO	-0.0258	**0.0282**	-0.0468
SUBARU	-0.0226	0.00364	**0.0291**
SUZUKI	0	**0.0645**	0.0402
TENCENT	0.0139	0.00987	**0.0186**
TESCO	**0.0476**	0	-0.0302
TESLA	0.0480	0.0656	**0.0669**
TOYOTA	**0.0333**	0	0
UBISOFT	**0.0126**	-0.0442	0
WALMART	0.0049	0.00198	**0.00967**
XEROX	**0.04045**	0	0.0173
YAMAHACO	0.0130	**0.0159**	0.0130
YAMAHAMO	-0.00363	-0.00229	**0.00734**

Table 6.9. For the pairwise comparisons ActTrade was the control algorithm once again and when comparing the distributions of ActTrade with the distributions of STGP-FASATA and STGP-FASATA-S we find them to be statistically significant different.

Table 6.9: Kolmogorov-Smirnov test p-values on the rate of return of the proposed ActTrade algorithm against the 2 GP benchmarks. Statistical significance changes are based on the Holm-Bonferroni correction. Statistically significant differences at the 5% level are indicated in boldface.

Pairwise comparison	***KS test p-value***	**Rank**	***Significance level***
ActTrade vs STGP-FASATA	**1.37E-36**	1	0.025
ActTrade vs STGP-FASATA-S	**1.75E-06**	2	0.05

6.4.3 Risk

Tables 6.10 and 6.11 present the median risk values over the 50 runs for the 42 companies. The algorithm that has the most lowest risk values is STGP-FASATA with 17 companies, followed by ActTrade (15) and STGP-FASATA-S (10). Similarly to Section 6.4.2, we notice that some values are closer to each other.

As in Sections 6.4.1 and 6.4.2, Table 6.12 displays the summary statistics for the values of risk of all 50 runs for each company across the three GP algorithms tested. We get a very similar picture to what we have seen so far in that the proposed ActTrade algorithm has the lowest mean and median values, as well as the lowest standard deviation. STGP-FASATA-S observes the lowest maximum risk value, while the lowest minimum risk is by STGP-FASATA.

Table 6.10: Median values for risk per company. Boldface is used to denote the best value for the particular dataset. First 21 companies.

Company	***STGP-FASATA***	***STGP-FASATA-S***	***ActTrade***
AAPL	0.01399	**0.00541**	0.0058
ADIDAS	**0.02399**	0	0
ALIBABA	0.02960	0.01312	**0.0027**
AMAZON	**0.01914**	0	0
ASUS	0.01285	0.01620	**0.00424**
ATVI	0.02209	0.02006	**0.00942**
BLACKB	0.02391	**0.01794**	0.08741
COKE	0.00927	**0.00747**	0.01027
EBAY	0.04924	0.02337	**0.01760**
ESTEE	0.00613	**0.00304**	0.00593
FORD	**0.00351**	0	0.00574
FUJIFILM	0.02563	**0.01786**	0.02678
FUJITSU	0.03339	**0.00429**	0.00803
GM	**0.05301**	0	0
GOOGL	0.02179	0.02787	**0.00286**
HITACHI	**0.01443**	0	0
HONDA	0.00883	**0.00793**	0
HSBC	**0.00532**	0	0.00662
HYUNDAI	0.03814	0.03038	**0.01499**
IBM	0.06206	0.01446	**0.01435**
INTC	**0.00829**	0	0.09021

Table 6.11: Median value for risk per company. Boldface is used to denote the best value for the particular dataset. Last 21 companies.

Company	***STGP-FASATA***	***STGP-FASATA-S***	***ActTrade***
KERING	**0.02533**	0.04783	0
KODAK	**0.02755**	0.03425	0.0355
MCDON	0	0	**0.0435**
NESTLE	**0.00574**	0.01703	0.0162
NFLX	0.01763	**0.01007**	0.0138
NINTENDO	**0.01963**	0.05054	0.0314
NYT	0.04170	**0.03246**	0.0371
PANASONIC	**0.02605**	0.03873	0.0271
SAINSBURY	0.02259	0.01231	**0.0116**
SHISEIDO	**0.05050**	0.06252	0.0757
SUBARU	0.04449	0.03800	**0.0141**
SUZUKI	0	**0.01017**	0.0313
TENCENT	0.01701	0.02464	**0.0049**
TESCO	0.00306	0	**0.0020**
TESLA	0.07283	0.05365	**0.0327**
TOYOTA	**0.01102**	0	0
UBISOFT	**0.02372**	0.08165	0
WALMART	0.01310	0.02351	**0.0047**
XEROX	**0.02727**	0	0.0538
YAMAHACO	0.01740	0.01586	**0.0058**
YAMAHAMO	**0.02108**	0.02177	0.0227

Table 6.12: Summary statistics of risk across all 50 GP runs and all 42 companies. Boldface is used to denote the best value for each statistic.

Algorithm	***Average***	***Median***	***StDev***	***Maximum***	***Minimum***
STGP-FASATA	0.028	0.022	0.026	0.19	**0.000067**
STGP-FASATA-S	0.026	0.0177	0.026	**0.126**	0.00046
ActTrade	**0.023**	**0.016**	**0.023**	0.18	0.0004

Table 6.13: Kolmogorov-Smirnov test p-values on the risk of the proposed ActTrade algorithm against the 2 GP benchmarks. Statistical significance changes are based on the Holm-Bonferroni correction. Statistically significant differences at the 5% level are indicated in boldface.

Pairwise comparison	***KS test p-value***	**Rank**	***Significance level***
ActTrade vs STGP-FASATA	**2.98E-31**	1	0.025
ActTrade vs STGP-FASATA-S	**1.22E-06**	2	0.05

Table 6.13 shows the differences in the distributions between ActTrade, as the control algorithm, and the two benchmark algorithms. The small p-values indicate that the distributions are statistically significant different, at the 5% significance level.

6.4.4 The role of the active trading operator

Given the above positive results for the ActTrade algorithm, we are interested in understanding the effects of the proposed genetic operator during the learning (training) phase of the models. To do this, we track the fitness value (Sharpe ratio) of the population for each generation.

As previously mentioned in Subsection 6.2.2, the novel genetic operator yields offspring through two distinct mechanisms. First, it generates offspring directly by incorporating the three newly created individuals into the next generation. These individuals are identified as follows: Offspring1 (where the FA subtree maximises the Sharpe ratio, while the SA and TA subtrees maximise the number of trades); Offspring2 (with the SA subtree maximising the Sharpe ratio, and the FA and TA subtrees maximising trades); and Offspring3 (where the TA subtree maximises the Sharpe ratio, while the FA and SA subtrees maximise trades). Second, the operator indirectly influences offspring generation by using Offspring1, Offspring2, and Offspring3 as parents for crossover and mutation. Consequently, at each generation, we encounter not only Offspring1, Offspring2, and Offspring3 but also several individuals that were generated by having these offspring as parents. Furthermore, traditional tournament selection also contributes to the creation of individuals in each generation.[3]

To organise our analysis effectively, we categorise the offspring in each generation in the following categories:

- Offspring1
- Offspring2
- Offspring3
- Tournament Selection Group (Group-Tourn): Individuals in this group are the result of

[3]There is also an elitist individual that exists at each generation, but for the purpose of this section's analysis, we focus solely on understanding the performance of individuals derived through the proposed genetic operator and how they compare to those generated by tournament selection.

conventional tournament selection.

- FA Group (Group-FA): This group comprises offspring whose first parent is Offspring1. Given that this is a selection method, several offspring are generated in this way, hence the word 'group' in the naming.

- SA Group (Group-SA): The SA group includes offspring whose first parent is Offspring2.

- TA Group (Group-TA): Offspring in this group have parents whose first parent is Offspring3.

Tables 6.14 and 6.15 present the average Sharpe ratio values for each offspring group across all generations over the 50 GP runs. Our primary objective is to rigorously compare the performance of offspring generated through our proposed operator with those generated via traditional tournament selection. Note that we exclude duplicate models (i.e. models/trees that exist more than once in a given group). As we can observe, all categories of offspring exhibit high average Sharpe ratio values. More importantly, there are many cases in which offspring generated through tournament selection (Group-Tourn) have, on average, lower fitness values than the other categories. This is an important finding because it indicates that offspring generated by the proposed genetic operator (either directly or indirectly through acting as a parent for crossover and mutation) has the ability to create competitive models that often outperform models derived from tournament selection.

Table 6.14: Average Sharpe ratio values of the different offspring groups. Boldface is used to denote the best value for the particular company. This table presents the first 21 companies.

Company	***Offspring1***	***Offspring2***	***Offspring3***	***Group-Tourn***	***Group-FA***	***Group-SA***	***Group-TA***
AAPL	5.33	**39.28**	11.63	15.20	15.67	16.85	11.63
ADIDAS	**19.43**	2.73	6.06	13.62	15.89	13.86	15.29
ALIBABA	14.22	13.08	**16.29**	11.39	13.26	12.39	15.81
AMAZON	5.32	32.30	**42.95**	24.27	25.43	27.13	26.56
ASUS	12.13	5.60	**23.25**	11.83	12.63	12.35	12.07
ATVI	6.95	10.54	**13.31**	8.54	8.50	8.63	8.37
BLACKB	5.68	16.89	**28.64**	14.96	15.33	16.98	16.83
COKE	8.61	9.48	**21.29**	10.62	11.13	10.65	13.24
EBAY	8.68	20.11	**34.07**	16.18	16.81	17.33	18.46
ESTEE	**8.52**	7.74	6.80	6.49	7.78	7.67	6.99
FORD	6.60	10.07	**18.90**	9.54	10.98	11.05	8.92
FUJIFILM	11.03	13.54	**15.00**	12.00	14.53	14.52	10.32
FUJITSU	17.42	10.06	**50.89**	20.25	24.53	24.21	19.32
GM	27.42	4.34	**63.08**	28.01	31.12	28.40	33.35
GOOGL	10.20	**39.83**	11.62	24.43	25.25	27.69	25.90
HITACHI	**206.79**	3.81	72.04	141.38	173.94	159.22	169.06
HONDA	14.99	16.94	**20.11**	16.33	18.42	18.98	17.17
HSBC	3.18	4.28	**11.64**	5.55	6.14	6.39	6.23
HYUNDAI	5.23	6.31	**19.49**	11.54	12.28	12.34	12.62
IBM	13.74	**106.57**	26.19	46.14	41.85	50.43	26.60
INTC	14.62	**45.51**	7.78	19.78	21.86	23.82	17.26

Table 6.15: Average Sharpe ratio values of the different offspring groups. Boldface is used to denote the best value for the particular company. This table presents the last 21 companies.

Company	***Offspring1***	***Offspring2***	***Offspring3***	***Group-Tourn***	***Group-FA***	***Group-SA***	***Group-TA***
KERING	11.65	**21.62**	17.77	12.16	13.69	14.58	15.18
KODAK	4.41	**10.41**	6.39	5.16	5.93	6.38	5.21
MCDON	19.96	4.43	**51.78**	31.28	33.16	31.55	35.05
NESTLE	5.44	12.37	**23.92**	11.30	12.55	13.05	10.94
NFLX	17.24	12.92	**24.38**	16.66	17.17	16.11	19.79
NINTENDO	5.04	4.72	**6.98**	4.39	4.86	4.79	4.24
NYT	57.03	8.04	**597.05**	225.10	241.75	233.01	246.84
PANASONIC	8.07	8.03	**15.21**	9.73	9.93	10.31	9.34
SAINSBURY	25.82	29.24	**35.56**	18.64	21.78	22.03	17.83
SHISEIDO	11.54	10.35	**28.96**	16.75	18.76	17.69	17.14
SUBARU	12.11	5.39	**13.89**	10.63	11.39	10.29	9.91
SUZUKI	10.66	17.85	**19.50**	15.66	15.36	15.74	14.34
TENCENT	5.69	**12.82**	12.11	9.30	7.11	8.42	9.47
TESCO	**64.87**	16.83	13.56	36.99	45.71	41.62	50.50
TESLA	13.41	**19.12**	8.58	11.73	13.05	13.78	12.26
TOYOTA	3.61	12.95	**279.70**	161.56	98.91	103.87	258.83
UBISOFT	15.82	10.47	11.49	15.51	17.07	17.02	**18.72**
WALMART	3.26	**11.28**	7.49	6.51	6.23	7.10	6.73
XEROX	19.79	7.84	**49.99**	27.40	29.48	27.81	34.08
YAMAHACO	3.47	5.31	**11.01**	6.18	6.96	6.70	6.30
YAMAHAMO	12.98	8.41	**15.83**	11.16	12.63	12.31	13.53

6.4.5 Results of each market

As we showcased in the previous thesis contributions, although all 42 companies' data comes from the same time period, there is a lot of variation among their price series. Thus, we again examine the algorithms' performance across different market profiles by separating them into Group 1 (companies with an increase of at least 20%), Group 2 (companies with an increase between 0% and 19.99%) and Group 3 (companies with a decrease).

After defining the above groups, 19 companies were placed in Group 1, 14 in Group 2, and 9 in Group 3, respectively.

We then report the average value of each metric (Sharpe ratio, rate of return, and risk) for each GP algorithm across the datasets for each group. As we can observe from Table 6.16, the proposed ActTrade performs strongly in terms of the aggregate metric of Sharpe ratio for Group 1, and it has a high value for Group 3, while in Group 2 the winner is GP-FASATA and in Group 3 the best performance derived from ActTrade. With regards to rate of return, ActTrade has the best value for Groups 1 and 3, while in Group 2, the best performance is returned by that of STGP-FASATA-S. Finally, in terms of risk, ActTrade performs way lower than the rest of the algorithms for Groups 2 and 3, while STGP-FASATA has the lowest risk for Group 1.

In summary, ActTrade has the best performance in terms of the rate of return in two of the groups. Moreover, it delivers low risk values across all groups. Given the variation of results across groups and metrics, looking at the Sharpe ratio as an aggregate metric of both return and risk is useful. As mentioned, ActTrade shows strong performance for datasets that either have very strong positive price movements (Group 1) or negative price movements (Group 3) and perform optimally in the two metrics assayed. It also performs well as regards the degree of risk for average price movements (Group 2). This indicates that our proposed algorithm is able to perform very well in strongly uptrend markets, as well as in downtrend markets. This is an important finding, demonstrating that our algorithm can perform well in opposite types of markets. Finally, the fact that the strongly-typed GP algorithms also perform very well in terms of the Sharpe ratio indicates the importance of the strongly-typed GP architecture when combining different analysis types of indicators.

Table 6.16: Separated average results per metric per trend group.

Market	***Algorithm***	***Sharpe ratio***	***Rate of return***	***Risk***
Group 1 (>20%)	STGP-FASATA	2.55	0.017	**0.023**
	STGP-FASATA-S	3.3	0.017	0.024
	ActTrade	**4.17**	**0.020**	0.027
Group 2 (0% - 19.99%)	STGP-FASATA	**3.7**	0.008	0.029
	STGP-FASATA-S	1.8	**0.02**	0.029
	ActTrade	1.8	0.011	**0.021**
Group 3 (<0%)	STGP-FASATA	2.26	-0.001	0.033
	STGP-FASATA-S	**3.5**	0.011	0.030
	ActTrade	2.18	**0.012**	**0.024**

6.5 Additional analysis

In this section we will explore the results of the ActTrade in more depth, introducing a "best tree analysis" in Section 6.5.1 and the analysis of the computational times in Section 6.5.2.

6.5.1 Best tree results

While the average results presented in the previous sections provide insights into the expected performance of a given algorithm in the machine learning domain, this section also showcases the best results achieved by each GP algorithm. "Best results" here refer to the top-performing tree in terms of fitness among 50 runs in the training set. Subsequently, this selected tree is applied to the unseen test set, effectively representing a single best tree chosen from those 50 runs. In the financial sector, this holds particular significance as investors employing GP algorithms in stock markets typically run the algorithm multiple times and subsequently opt for the best-performing tree or model for trading. Therefore, having an algorithm that excels in terms of the "best tree" is a critical aspect of the financial sector. Consequently, Table 6.17 presents the average performance of the best trees across the 42 datasets for each GP algorithm. To completely understand the algorithm's capabilities, we present additional analysis, including all previous GP algorithms. The reason for the inclusion is that we have not yet separately addressed them as regards the best tree results in the previous thesis to avoid repetition.

In the results, we exclude the companies whose runs gave a value of 0 so as to be consistent with the results from the previous sections. It is to be reminded that the results for each financial metric are presented individually.

As we can observe, the proposed ActTrade algorithm has the highest Sharpe ratio and rate of return values while achieving the lowest risk. As the Sharpe ratio is an aggregate metric that considers both return and risk, the fact that the best tree of ActTrade has the best value makes it a very positive result. It is also worth noting that practitioners pay particular attention to such aggregate metrics [171], thus ActTrade best tree's performance is of particular

Table 6.17: Best trees average performance across the 42 datasets

Algorithm	***Sharpe ratio***	***Return***	***Risk***
STGP-FASATA	3.23	-0.004	0.027
STGP-FASATA-S	1.76	0.020	0.04
ActTrade	**3.51**	**0.039**	**0.021**

importance.

6.5.2 Computational times

Table 6.18 presents the computational times for all algorithms tested in this chapter. As we can observe, the fastest algorithm is STGP-FASATA, which takes around 1.5 minutes. STGP-FASATA-S and ActTrde take more time, at more than 7 minutes per run.

Table 6.18: Computational times per algorithm

Algorithm	**Value (in minutes)**
STGP-FASATA	1.36
STGP-FASATA-S	7.10
ActTrade	7.48

However, such differences are not crucial because, in the algorithmic trading domain, such algorithms usually run offline, and only their models are used in real-time applications. So, while training a model can take up to 7.5 minutes, its application to the test set takes a fraction of a second. Furthermore, genetic programming is a highly parallelisable algorithm, and thus, its computational times can be further reduced through the parallelisation processes [172].

6.5.3 Summary of findings

To sum up, as seen in Tables 6.2 - 6.17 and focusing on the findings from ActTrade, the results have been summarised below. When comparing the GP variants with each other, it is evident that:

- The proposed ActTrade algorithm is able to perform strongly across all three financial metrics of Sharpe ratio, rate of return and risk. Furthermore, its distribution is statistically significant different than the distributions of the two GP variants.
- The proposed genetic operator has the ability to create strong offspring (in terms of Sharpe ratio), which often outperform offspring derived by tournament selection.
- The novel ActTrade algorithm also has the highest values in Sharpe ratio and rate of return in the best tree results while exhibiting the least risk amongst all other algorithms.

6.6 Conclusion

In conclusion, this chapter presented a novel strongly-typed genetic programming algorithm that combines fundamental, sentiment, and technical analysis indicators. On top of the novel fitness function presented in Chapter 5, the proposed algorithm also utilised a novel genetic operator to encourage active trading. As we have established during this chapter, the novel genetic operator leads to improved results across all three metrics of Sharpe ratio, rate of return, and risk, outperforming STGP-FASATA and STGP-FASATA-S, and subsequently outperforming the GP variants showcased in Chapter 5. Additionally, the average trades generated by the ActTrade algorithm were more than the ones generated from STGP-FASATA and STGP-FASATA-S, showcasing the effectiveness of the operator. The importance of combining the three analysis indicator types, which have not been frequently occurring in previous studies, is evident from our findings. Combining the indicators enhances the models' knowledge and creates financially more advantageous trading strategies. Moreover, the novel fitness function, in addition to the active trading operator significantly improves these results.

Chapter 7

Thesis Conclusion

This thesis leveraged genetic programming techniques to address applications within the field of algorithmic trading. The primary objective of this study was to demonstrate the efficacy of the proposed GP algorithms in generating distinctive and profitable trading strategies that incorporate information from the three analysis types. To achieve this, the algorithms were employed to integrate three financial analysis types, namely fundamental, sentiment, and technical analysis, in addition to introducing a novel strongly-typed architecture, a novel fitness function and a novel GP operator. The conclusions drawn from the experiments conducted on these problems are presented in the subsequent sections. The discussion commences with an exploration of the motivation driving each set of experiments. This is followed by an analysis of the novelty and originality exhibited by the presented research. Subsequently, the conclusions of each set of experiments are detailed, offering insights into their outcomes and implications. Finally, this chapter delves into the exploration of future research, providing a comprehensive overview of potential directions for further investigation and advancement in this domain.

7.1 Summary of the combination of the three financial analysis types

7.1.1 Motivation of the presented research

In Chapter 4, the motivation stemmed from exploring the implications of utilising a genetic programming algorithm, one that incorporates the three financial analysis types (fundamental, sentiment, and technical analysis), as well as to show the advantages of their combined utilisation, as this integration had not been extensively investigated in the literature. To address this, we developed a novel GP algorithm, namely GP-FASATA, which is tailored to include all financial analysis indicators in its terminal set. This algorithm was then compared to three other GP benchmarks, four machine learning benchmarks, and a financial strategy to assess its significance and highlight its disadvantages.

7.1.2 Novelty of the Presented Research

While each analysis type can produce favourable trading outcomes individually, their combination yielded even more enhanced performance, especially related to the Sharpe ratio. This observation indicates that the individual strengths of these analysis types are complemented by their synergy when utilised together.

7.1.3 Conclusions

The GP-FASATA algorithm achieved higher mean values for the Sharpe ratio and rate of return metrics, although it did not perform with the highest median values. Moreover, the GP-FASATA algorithms' results revealed a statistically different distribution from the distributions of the other GP algorithms. On the other hand, GP-TA was the algorithm with the lowest mean and median values regarding risk, and its distribution was also statistically significant different to those of the other three GP variants. Thus, we aimed to improve the trading performance further, especially the value of risk, by introducing a strongly-typed GP architecture for the algorithms that combine the three financial analysis types, as well as a new fitness function.

7.2 Summary of the non-strongly and strongly-typed genetic programming

7.2.1 Motivation of the presented research

From Chapter 4, we concluded that GP-FASATA did perform with a higher mean Sharpe ratio and rate of return than the individual GP algorithms. However, the median values for two metrics were not as high and in relation to risk GP-TA attained the lowest risk value. Thus, in Chapter 5, as part of the second thesis contribution, in seeking to create an algorithm that would perform with lower risk, we introduced STGP-FASATA-S. The proposed algorithm incorporates a strongly-typed genetic programming structure and a novel fitness function, thereby maximising the complete tree of an individual and the three subtrees. To assess the importance of the algorithm, we compared it to the algorithms introduced in Chapter 4 and STGP-FASATA, which does possess a strongly-typed structure yet does not incorporate the novel fitness function.

7.2.2 Novelty of the presented research

The novel algorithm was able to conduct a more effective search of the space encompassing fundamental analysis, sentiment analysis, and technical analysis indicators. This advantage is not present in algorithms of Chapter 4, permitting all indicator types within its terminal set without the strongly-typed architecture. While the average and median values of the Sharpe ratio and rate of return are similar in both algorithms, we were more interested in reducing both the average and median values of risk. The strongly-typed architecture of the proposed STGP-FASATA-S algorithm enables a more effective search of the space, combining the three financial analysis types. In contrast, the novel fitness function of the algorithm can improve the financial performance of GP.

7.2.3 Conclusions

In conclusion, STGP-FASATA-S showcased higher mean and median values for all three financial metrics, performing better than the other five GP variants in terms of Sharpe ratio, rate of return, and risk, which we sought to lower in Chapter 5. Moreover, the proposed GP algorithm's results were also better than the results of the four machine learning benchmarks in *all financial metrics* and their distributions were statistically significant different. This part was especially included in Chapter 5 because the proposed algorithm in Chapter 4, GP-FASATA's distribution was not statistically significant different than the distributions of the machine learning benchmarks in all financial metrics. Therefore, we conclude that combining all indicators can enhance the models' knowledge and create financially more advantageous trading strategies. Finally, based on our analysis, it is not sufficiently profitable to simply combine the different types of indicators, as GP-FASATA does. Thus, a strongly typed architecture is essential in achieving an improved performance for the Sharpe ratio, which combines both rate of return and risk. This finding underscores the potential efficacy of our approach in developing robust trading strategies.

Wanting to take advantage of the strongly-typed architecture of the algorithm, along with the novel fitness function, for the next contribution to this thesis, we sought to increase the values of the Sharpe ratio and rate of return even further while decreasing measured risk. We achieved this by introducing a new genetic programming operator, one which encourages active trading.

7.3 Strongly-typed Genetic Programming variants

7.3.1 Motivation of the presented research

In Chapter 5, where we introduced STGP-FASATA and STGP-FASATA-S, the values of the Sharpe ratio and rate of return did increase, while the value of risk decreased. However, we wanted to strengthen these results more and create an algorithm that would further advance these values, especially in terms of the rate of return and risk. Thus, in Chapter 6, for the third and final thesis contribution, our goal was to showcase further financial profits and take full advantage of the strongly-typed architecture; we created the ActTrade algorithm. The ActTrade GP variant has the same fitness function as STGP-FASATA-S but introduces a novel operator that encourages active trading. We compared the ActTrade variant with the two other strongly-typed GP algorithms introduced in Chapter 5.

7.3.2 Novelty of the presented research

The proposed algorithm encourages active trading by injecting into the GP population trees that perform a high number of trades while achieving high profitability with low risk at the same time. Thus, the novel GP variant was able to perform strongly across all three key financial metrics of Sharpe ratio, rate of return and risk, and its distribution was statistically significant different than the other GP variants. The proposed genetic operator can create strong offspring (in terms of the Sharpe ratio), often outperforming offspring derived by tournament selection.

7.3.3 Conclusions

ActTrade performed better than the two other strongly-typed genetic programming algorithms and, subsequently, the non-strongly GP variants, with the addition of the novel GP operator leading to significantly improved results across all three financial metrics. The proposed genetic operator can spawn strong offspring, outperforming traditional GP operators and techniques.

This algorithm also exhibited significant "best tree" results, which is an important aspect referring to the best-performing tree in terms of the Sharpe ratio among 50 runs in the training set. This holds significance as investors in stock market applications using genetic programming algorithms often run the algorithm multiple times and choose the best performing tree for trading.

7.4 Future work

While our research has yielded promising results, it is important to acknowledge that it remains a work in progress for further improvements in the performance of trading strategies. Future research will concentrate on refining the genetic programming (GP) methodology by optimising the utilisation of trees generated through the proposed genetic operator to tackle the similarity between generated trees. This similarity manifests in both their structural configuration (genotype) and the signals they generate (phenotype). To overcome this challenge, we propose a strategy wherein we systematically replace tree nodes responsible for these similarities. By substituting these nodes with different, yet compatible nodes belonging to the same branch type (FA, SA, and TA nodes), we aim to introduce diversity and variability into the evolving population of trees. The careful selection of compatible nodes ensures that the fundamental characteristics of the trees are preserved while introducing the necessary diversity to enhance the overall robustness of the methodology. Our goal is to create a more dynamic and resilient GP framework capable of exploring a broader solution space, ultimately leading to the discovery of more effective and robust trading strategies.

Another avenue is to implement a modification in the GP process, targeting trees that do not meet predefined performance criteria. This strategic intervention seeks to enhance the adaptability and responsiveness of the GP methodology. This could be achieved by substituting nodes with trees from a secondary initialised population that includes additional function nodes compared to the original population. This secondary population is characterised by the inclusion of additional function nodes, such as NOT and XOR, exceeding the known elements of the original population. By infusing this supplementary genetic material into the underperforming individuals, we aim to instigate a form of genetic "rejuvenation", fostering the emergence of more sophisticated and potentially successful trading strategies. This deliberate injection of complexity is intended to create GP-generated trees with enhanced performance. Through these refinements, our research seeks to tackle the challenge of stagnation within the evolutionary process, proactively addressing instances where trees fail to meet desired benchmarks. By employing this targeted modification, we anticipate not only an elevation in the efficiency and effectiveness of individual trees but also a broader enhancement of the overall GP methodology. This iterative and adaptive approach aligns with our goal of optimising trading strategies for improved financial outcomes, contributing to the evolving landscape of algorithmic trading in financial markets.

Finally, in advancing the performance of trading strategies involves the incorporation of Multi-objective Optimisation (MOO). Through the application of MOO algorithms, we seek to identify Pareto-optimal solutions that represent optimal trade-offs between conflicting objectives. More specifically, we aim to maximise the rate of returns while simultaneously minimising risk, thereby presenting a holistic perspective on strategy optimisation. This way we aim to intricate balance between two conflicting objectives of financial decision-making, but both important for devising robust trading strategies. This not only enhances the adaptability of our approach but also contributes to a more comprehensive understanding of the complex dynamics inherent in financial markets.

References

[1] Eva Christodoulaki and Michael Kampouridis. Fundamental, technical and sentiment analysis for algorithmic trading with genetic programming. In *2023 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, 2022.

[2] Eva Christodoulaki, Michael Kampouridis, and Maria Kyropoulou. Enhanced genetic programming for algorithmic trading. In *Genetic and Evolutionary Computation Conference (2023)*, pages 1–8, 2022.

[3] Eva Christodoulaki and Michael Kampouridis. Using strongly typed genetic programming to combine technical and sentiment analysis for algorithmic trading. In *2022 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, 2022.

[4] Eva Christodoulaki and Michael Kampouridis. Combining technical and sentiment analysis under a genetic programming algorithm. In *UK Workshop of Computational Intelligence (UKCI)*, 2022.

[5] E. Christodoulaki, M. Kampouridis, and P. Kanellopoulos. Technical and sentiment analysis in financial forecasting with genetic programming. In *IEEE Symposium on Computational Intelligence for Financial Engineering & Economics (CIFEr)*, 2022.

[6] Alma Lilia Garcia Almanza. *New Classification Methods For Gathering Patterns In The Context Of Genetic Programming*. PhD thesis, Citeseer, 2008.

[7] Tony Plummer. *Forecasting financial markets: the psychology of successful investing*. Kogan Page Publishers, 2009.

[8] Natividad Blasco, Pilar Corredor, and Sandra Ferreruela. Market sentiment: a key factor of investors' imitative behaviour. *Accounting & Finance*, 52(3):663–689, 2012.

[9] Gordon Ritter. Machine learning for trading. *Available at SSRN 3015609*, 2017.

[10] Irene Aldridge. *High-frequency trading: a practical guide to algorithmic strategies and trading systems*, volume 604. John Wiley & Sons, 2013.

[11] Matthew Krantz. *Fundamental analysis for dummies*. John Wiley & Sons, 2023.

[12] Chi Cheong Allen Ng, Jianfu Shen, and S Ghon Rhee. Fundamental analysis and stock returns in international equity markets. *The Hong Kong Polytechnic working paper. Retrieved on*, 20, 2019.

[13] Xuemin Yan and Lingling Zheng. Fundamental analysis and the cross-section of stock returns: A data-mining approach. *The Review of Financial Studies*, 30(4):1382–1423, 2017.

[14] Xi Chen, Yang Ha Cho, Yiwei Dou, and Baruch Lev. Fundamental analysis of detailed financial data: A machine learning approach. *Journal of Accounting Research*, 60(2):467–515, 2022.

[15] Stanley G Eakins and Stanley R Stansell. Can value-based stock selection criteria yield superior risk-adjusted returns: an application of neural networks. *International review of financial analysis*, 12(1):83–97, 2003.

[16] Tong-Seng Quah. Improving returns on stock investment through neural network selection. In *Artificial Neural Networks in Finance and Manufacturing*, pages 152–164. IGI Global, 2006.

[17] Ming Hao Eng, Yang Li, Qing-Guo Wang, and Tong Heng Lee. Forecast forex with ann using fundamental data. In *2008 International Conference on Information Management, Innovation Management and Industrial Engineering*, volume 1, pages 279–282. IEEE, 2008.

[18] Masaya Abe and Hideki Nakayama. Deep learning for forecasting stock returns in the cross-section. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 273–284. Springer, 2018.

[19] Qun He, Zhangli Cai, and Wenjing Liu. Prediction of listed companies' revenue based on model-fused. In *2019 International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*, pages 257–260. IEEE, 2019.

[20] Tong-Seng Quah. Djia stock selection assisted by neural network. *Expert Systems with Applications*, 35(1-2):50–58, 2008.

[21] Yuxuan Huang, Luiz Fernando Capretz, and Danny Ho. Neural network models for stock selection based on fundamental analysis. In *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*, pages 1–4. IEEE, 2019.

[22] Frank McGroarty, Ash Booth, Enrico Gerding, and VL Raju Chinthalapati. High frequency trading strategies, market fragility and price spikes: an agent based model perspective. *Annals of Operations Research*, 282:217–244, 2019.

[23] Eugene F Fama and Kenneth R French. Size, value, and momentum in international stock returns. *Journal of financial economics*, 105(3):457–472, 2012.

[24] Aty Herawati, Angger Setiadi Putra, et al. The influence of fundamental analysis on stock prices: The case of food and beverage industries. *European Research Studies Journal*, 21(3):316–326, 2018.

[25] Meiliani Luckieta, Ali Amran, and Doni Purnama Alamsyah. The fundamental analysis of stock prices. *Test Engineering and Management*, 23:28–721, 2020.

[26] Clement Ajekwe and Adzor Ibiamke. Financial statement analysis to predict stock returns of listed consumer goods firms in nigeria. *Available at SSRN 3247805*, 2018.

[27] Yuxuan Huang, Luiz Fernando Capretz, and Danny Ho. Machine learning for stock prediction based on fundamental analysis. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 01–10. IEEE, 2021.

[28] Aswath Damodaran. *The little book of valuation: how to value a company, pick a stock and profit*, volume 34. John Wiley & Sons, 2011.

[29] G Vinodhini and RM Chandrasekaran. Sentiment analysis and opinion mining: a survey. *International Journal*, 2(6):282–292, 2012.

[30] Bing Liu. *Sentiment analysis: Mining opinions, sentiments, and emotions*. Cambridge university press, 2020.

[31] Kazuhiro Kohara, Tsutomu Ishikawa, Yoshimi Fukuhara, and Yukihiro Nakamura. Stock price prediction using prior knowledge and neural networks. *Intelligent Systems in Accounting, Finance & Management*, 6(1):11–22, 1997.

[32] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. Deep learning for event-driven stock prediction. In *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[33] Yangtuo Peng and Hui Jiang. Leverage financial news to predict stock price movements using word embeddings and deep neural networks. *arXiv preprint arXiv:1506.07220*, 2015.

[34] Saloni Mohan, Sahitya Mullapudi, Sudheer Sammeta, Parag Vijayvergia, and David C Anastasiu. Stock price prediction using news sentiment analysis. In *2019 IEEE fifth international conference on big data computing service and applications (BigDataService)*, pages 205–208. IEEE, 2019.

[35] Wataru Souma, Irena Vodenska, and Hideaki Aoyama. Enhanced news sentiment analysis using deep learning methods. *Journal of Computational Social Science*, 2(1):33–46, 2019.

[36] Pooja Mehta, Sharnil Pandya, and Ketan Kotecha. Harvesting social media sentiment analysis to enhance stock market prediction using deep learning. *PeerJ Computer Science*, 7:e476, 2021.

[37] Aakanksha Sharaff, Tushin Roy Chowdhury, and Sakshi Bhandarkar. Lstm based sentiment analysis of financial news. *SN Computer Science*, 4(5):584, 2023.

[38] Boyi Xie, Rebecca Passonneau, Leon Wu, and Germán G Creamer. Semantic frames to predict stock price movement. In *Proceedings of the 51st annual meeting of the association for computational linguistics*, pages 873–883, 2013.

[39] Xi Zhang, Yunjia Zhang, Senzhang Wang, Yuntao Yao, Binxing Fang, and S Yu Philip. Improving stock market prediction via heterogeneous information fusion. *Knowledge-Based Systems*, 143:236–247, 2018.

[40] Charalampos M Liapis, Aikaterini Karanikola, and Sotiris Kotsiantis. Investigating deep stock market forecasting with sentiment analysis. *Entropy*, 25(2):219, 2023.

[41] Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. Open information extraction: The second generation. In *IJCAI*, volume 11, pages 3–10, 2011.

[42] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. Using structured events to predict stock price movement: An empirical investigation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1415–1425, 2014.

[43] DK Kirange, Ratnadeep R Deshmukh, et al. Sentiment analysis of news headlines for stock price prediction. *Composoft, An International Journal of Advanced Computer Technology*, 5(3):2080–2084, 2016.

[44] Jia-Yen Huang and Jin-Hao Liu. Using social media mining technology to improve stock price forecast accuracy. *Journal of Forecasting*, 39(1):104–116, 2020.

[45] Saqib Farid, Rubeena Tashfeen, Tahseen Mohsan, and Arsal Burhan. Forecasting stock prices using a data mining method: Evidence from emerging market. *International Journal of Finance & Economics*, 28(2):1911–1917, 2023.

[46] Ronald Hochreiter. Computing trading strategies based on financial sentiment data using evolutionary optimization. In *Mendel 2015: Recent Advances in Soft Computing*, pages 181–191. Springer, 2015.

[47] Steve Y Yang, Sheung Yin Kevin Mo, Anqi Liu, and Andrei A Kirilenko. Genetic programming optimization for a sentiment feedback strength based trading strategy. *Neurocomputing*, 264:29–41, 2017.

[48] Dinesh K Sharma, HS Hota, Kate Brown, and Richa Handa. Integration of genetic algorithm with artificial neural network for stock market forecasting. *International Journal of System Assurance Engineering and Management*, 13(Suppl 2):828–841, 2022.

[49] Jia-Yen Huang, Chun-Liang Tung, and Wei-Zhen Lin. Using social network sentiment analysis and genetic algorithm to improve the stock prediction accuracy of the deep learning-based approach. *International Journal of Computational Intelligence Systems*, 16(1):93, 2023.

[50] G Alan Wang, Jian Jiao, Alan S Abrahams, Weiguo Fan, and Zhongju Zhang. Expertrank: A topic-aware expert finding algorithm for online knowledge communities. *Decision support systems*, 54(3):1442–1451, 2013.

[51] Qing Li, TieJun Wang, Ping Li, Ling Liu, Qixu Gong, and Yuanzhu Chen. The effect of news and public mood on stock movements. *Information Sciences*, 278:826–840, 2014.

[52] Jaeyoon Kim, Jangwon Seo, Minhyeok Lee, and Junhee Seok. Stock price prediction through the sentimental analysis of news articles. In *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 700–702, 2019.

[53] Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of computational science*, 2(1):1–8, 2011.

[54] SA Bogle and WD Potter. Sentamal-a sentiment analysis machine learning stock predictive model. In *Proceedings on the international conference on artificial intelligence (ICAI)*, page 610. The Steering Committee of The World Congress in Computer Science, 2015.

[55] Venkata Sasank Pagolu, Kamal Nayan Reddy, Ganapati Panda, and Babita Majhi. Sentiment analysis of twitter data for predicting stock market movements. In *2016 international conference on signal processing, communication, power and embedded system (SCOPES)*, pages 1345–1350. IEEE, 2016.

[56] Vytautas Karalevicius, Niels Degrande, and Jochen De Weerdt. Using sentiment analysis to predict interday bitcoin price movements. *The Journal of Risk Finance*, 19(1):56–75, 2018.

[57] Nabanita Das, Bikash Sadhukhan, Tanusree Chatterjee, and Satyajit Chakrabarti. Effect of public sentiment on stock market movement prediction during the covid-19 outbreak. *Social network analysis and mining*, 12(1):92, 2022.

[58] Wenbin Zhang and Steven Skiena. Trading strategies to exploit blog and news sentiment. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 4, 2010.

[59] Axel Groß-Klußmann and Nikolaus Hautsch. When machines read the news: Using automated text analytics to quantify high frequency news-implied market reactions. *Journal of Empirical Finance*, 18(2):321–340, 2011.

[60] Min-Yuh Day and Chia-Chou Lee. Deep learning for financial sentiment analysis on finance news providers. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1127–1134. IEEE, 2016.

[61] Steven Loria. Textblob documentation. *Release 0.15*, 2, 2018.

[62] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *LREC*. European Language Resources Association, 2010.

[63] Finn Årup Nielsen. A new ANEW: evaluation of a word list for sentiment analysis in microblogs. In Matthew Rowe, Milan Stankovic, Aba-Sah Dadzie, and Mariann Hardey, editors, *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages*, volume 718 of *CEUR Workshop Proceedings*, pages 93–98, May 2011.

[64] John J Murphy. *Technical analysis of the financial markets: A comprehensive guide to trading methods and applications*. Penguin, 1999.

[65] Vladimir N Vapnik. The nature of statistical learning. *Theory*, 1995.

[66] Yanshan Wang and In-Chan Choi. Market index and stock price direction prediction using machine learning techniques: an empirical study on the kospi and hsi. *arXiv preprint arXiv:1309.7119*, 2013.

[67] Huanhuan Yu, Rongda Chen, and Guoping Zhang. A svm stock selection model within pca. *Procedia computer science*, 31:406–412, 2014.

[68] Shuai Wang and Wei Shang. Forecasting direction of china security index 300 movement with least squares support vector machine. *Procedia Computer Science*, 31:869–874, 2014.

[69] Indu Kumar, Kiran Dogra, Chetna Utreja, and Premlata Yadav. A comparative study of supervised machine learning algorithms for stock market trend prediction. In *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pages 1003–1007. IEEE, 2018.

[70] Halbert White. Economic prediction using neural networks: The case of ibm daily stock returns. In *ICNN*, volume 2, pages 451–458, 1988.

[71] Mohamed M Mostafa. Forecasting stock exchange movements using neural networks: Empirical evidence from kuwait. *Expert Systems with Applications*, 37(9):6302–6309, 2010.

[72] Scott McDonald, Sonya Coleman, T Martin McGinnity, Yuhua Li, and Ammar Belatreche. A comparison of forecasting approaches for capital markets. In *2014 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr)*, pages 32–39. IEEE, 2014.

[73] Ayodele Ariyo Adebiyi, Aderemi Oluyinka Adewumi, and Charles Korede Ayo. Comparison of arima and artificial neural networks models for stock price prediction. *Journal of Applied Mathematics*, 2014, 2014.

[74] Fernando Sánchez Lasheras, Francisco Javier de Cos Juez, Ana Suárez Sánchez, Alicja Krzemień, and Pedro Riesgo Fernández. Forecasting the comex copper spot price by means of neural networks and arima models. *Resources Policy*, 45:37–43, 2015.

[75] Jordan Ayala, Miguel García-Torres, José Luis Vázquez Noguera, Francisco Gómez-Vela, and Federico Divina. Technical analysis strategy optimization using a machine learning approach in stock market indices. *Knowledge-Based Systems*, 225:107119, 2021.

[76] Matthew Dixon. A high-frequency trade execution model for supervised learning. *High Frequency*, 1(1):32–52, 2018.

[77] Zexin Hu, Yiqi Zhao, and Matloob Khushi. A survey of forex and stock price prediction using deep learning. *Applied System Innovation*, 4(1):9, 2021.

[78] Kai Chen, Yi Zhou, and Fangyan Dai. A lstm-based method for stock returns prediction: A case study of china stock market. In *2015 IEEE international conference on big data (big data)*, pages 2823–2824. IEEE, 2015.

[79] Ha Young Kim and Chang Hyun Won. Forecasting the volatility of stock price index: A hybrid model integrating lstm with multiple garch-type models. *Expert Systems with Applications*, 103:25–37, 2018.

[80] David MQ Nelson, Adriano CM Pereira, and Renato A de Oliveira. Stock market's price movement prediction with lstm neural networks. In *2017 International joint conference on neural networks (IJCNN)*, pages 1419–1426. IEEE, 2017.

[81] M Mallikarjuna and R Prabhakara Rao. Evaluation of forecasting methods from selected stock market returns. *Financial Innovation*, 5(1):1–16, 2019.

[82] Laura Alessandretti, Abeer ElBahrawy, Luca Maria Aiello, and Andrea Baronchelli. Anticipating cryptocurrency prices using machine learning. *Complexity*, 2018:1–16, 2018.

[83] Jiawei Long, Zhaopeng Chen, Weibing He, Taiyu Wu, and Jiangtao Ren. An integrated framework of deep learning and knowledge graph for prediction of stock price trend: An application in chinese stock exchange market. *Applied Soft Computing*, 91:106205, 2020.

[84] Pranay Kumbhare, Lokendra Kolhe, Sakshi Dani, Pooja Fandade, and Dipti Theng. Algorithmic trading strategy using technical indicators. In *2023 11th International Conference on Emerging Trends in Engineering & Technology-Signal and Information Processing (ICETET-SIP)*, pages 1–6. IEEE, 2023.

[85] Akshit Kurani, Pavan Doshi, Aarya Vakharia, and Manan Shah. A comprehensive comparative study of artificial neural network (ann) and support vector machines (svm) on stock forecasting. *Annals of Data Science*, 10(1):183–208, 2023.

[86] Ms SVS VALLI, Mr S AMRUTH RAJ, Mr N JEEVAN SAiI, Ms K LAVANYA, and Mrs Khadarunnisa. Stock market prediction using machine learning algorithms. *soft computing*, 52(4), 2023.

[87] Jin Li and Edward PK Tsang. Improving technical analysis predictions: An application of genetic programming. pages 108–112, 1999.

[88] Michael Kampouridis and Edward Tsang. Investment opportunities forecasting: Extending the grammar of a gp-based tool. *International Journal of Computational Intelligence Systems*, 5(3):530–541, 2012.

[89] Michael Kampouridis, Abdullah Alsheddy, and Edward Tsang. On the investigation of hyper-heuristics on a financial forecasting problem. *Annals of Mathematics and Artificial Intelligence*, 68:225–246, 2013.

[90] Michael Kampouridis and Fernando Otero. Heuristic procedures for improving the predictability of a genetic programming financial forecasting algorithm. *Soft Computing*, 21:295–310, 2017.

[91] José Manuel Berutich, Francisco López, Francisco Luna, and David Quintana. Robust technical trading strategies using gp for algorithmic portfolio selection. *Expert Systems with Applications*, 46:307–315, 2016.

[92] Anthony Brabazon, Michael Kampouridis, and Michael O'Neill. Applications of genetic programming to finance and economics: past, present, future. *Genetic Programming and Evolvable Machines*, 21(1):33–53, 2020.

[93] Xinpeng Long, Michael Kampouridis, and Delaram Jarchi. An in-depth investigation of genetic programming and nine other machine learning algorithms in a financial forecasting problem. In *IEEE Congress on Evolutionary Computation (CEC)*, 2022.

[94] Xinpeng Long, Michael Kampouridis, and Panagiotis Kanellopoulos. Genetic programming for combining directional changes indicators in international stock mar-

kets. In *International Conference on Parallel Problem Solving from Nature*, pages 33–47. Springer, 2022.

[95] Xinpeng Long, Michael Kampouridis, and Panagiotis Kanellopoulos. Multi-objective optimisation and genetic programming for trading by combining directional changes and technical indicators. *IEEE XPlore*, 2023.

[96] Davut Ari and Baris Baykant Alagoz. Dehypgpols: a genetic programming with evolutionary hyperparameter optimization and its application for stock market trend prediction. *Soft Computing*, 27(5):2553–2574, 2023.

[97] Mohamed M Mostafa and Ahmed A El-Masry. Oil price forecasting using gene expression programming and artificial neural networks. *Economic Modelling*, 54:40–53, 2016.

[98] Kyoung-jae Kim and Ingoo Han. Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index. *Expert systems with Applications*, 19(2):125–132, 2000.

[99] Esmaeil Hadavandi, Hassan Shavandi, and Arash Ghanbari. Integration of genetic fuzzy systems and artificial neural networks for stock price forecasting. *Knowledge-Based Systems*, 23(8):800–808, 2010.

[100] Luna M Zhang. Genetic deep neural networks using different activation functions for financial data mining. In *2015 IEEE International Conference on Big Data (Big Data)*, pages 2849–2851. IEEE, 2015.

[101] Yan Chen and Xuancheng Wang. A hybrid stock trading system using genetic network programming and mean conditional value-at-risk. *European Journal of Operational Research*, 240(3):861–871, 2015.

[102] Amadu Fullah Kamara, Enhong Chen, and Zhen Pan. An ensemble of a boosted hybrid of deep learning models and technical analysis for forecasting stock prices. *Information Sciences*, 594:1–19, 2022.

[103] Heon Baek. A cnn-lstm stock prediction model based on genetic algorithm optimization. *Asia-Pacific Financial Markets*, pages 1–16, 2023.

[104] Pi-Sheng Deng and Tzu-Man Huang. Using neural-genetic hybrid systems for complex decision support. *Neural Computing and Applications*, 35(15):11403–11416, 2023.

[105] Jiali Fang, Yafeng Qin, and Ben Jacobsen. Technical market indicators: An overview. *Journal of behavioral and experimental finance*, 4:25–56, 2014.

[106] Tim Loughran, Bill McDonald, and Hayong Yun. A wolf in sheep's clothing: The use of ethics-related terms in 10-k reports. *Journal of Business Ethics*, 89(1):39–49, 2009.

[107] Narasimhan Jegadeesh and Di Wu. Word power: A new approach for content analysis. *Journal of financial economics*, 110(3):712–729, 2013.

[108] Tim Loughran and Bill McDonald. When is a liability not a liability? textual analysis, dictionaries, and 10-ks. *The Journal of Finance*, 66(1):35–65, 2011.

[109] Gerard Hoberg and Vojislav Maksimovic. Redefining financial constraints: A text-based analysis. *The Review of Financial Studies*, 28(5):1312–1352, 2015.

[110] Mehran Azimi and Anup Agrawal. Is positive sentiment in corporate annual reports informative? evidence from deep learning. *Evidence from Deep Learning (July 2019)*, 2019.

[111] Petr Hajek and Michal Munk. Speech emotion recognition and text sentiment analysis for financial distress prediction. *Neural Computing and Applications*, pages 1–15, 2023.

[112] Gerben De Zwart, Thijs Markwat, Laurens Swinkels, and Dick van Dijk. The economic value of fundamental and technical information in emerging currency markets. *Journal of International Money and Finance*, 28(4):581–604, 2009.

[113] Yingzi Zhu and Guofu Zhou. Technical analysis: An asset allocation perspective on the use of moving averages. *Journal of Financial Economics*, 92(3):519–544, 2009.

[114] Lukas Menkhoff. The use of technical analysis by fund managers: International evidence. *Journal of Banking & Finance*, 34(11):2573–2586, 2010.

[115] Jenni L Bettman, Stephen J Sault, and Emma L Schultz. Fundamental and technical analysis: substitutes or complements? *Accounting & Finance*, 49(1):21–36, 2009.

[116] MH Fazel Zarandi, Babak Rezaee, IB Turksen, and Elahe Neshat. A type-2 fuzzy rule-based expert system model for stock price analysis. *Expert Systems with Applications*, 36(1):139–154, 2009.

[117] Mahfudh Ahmad, Haryono Soeparno, and Togar Alam Napitupulu. Stock trading alert: With fuzzy knowledge-based systems and technical analysis. In *2020 International Conference on Information Technology Systems and Innovation (ICITSI)*, pages 155–160. IEEE, 2020.

[118] Rohan Pothumsetty. Application of artificial intelligence in algorithmic trading. *Int. J. Eng. Appl. Sci. Technol.*, 4(12):140–149, 2020.

[119] Erhan Beyaz, Firat Tekiner, Xiao-jun Zeng, and John Keane. Comparing technical and fundamental indicators in stock price forecasting. In *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pages 1607–1613. IEEE, 2018.

[120] Ata Larijani and Farbod Dehghani. Stock price prediction using the combination of firefly (fa) and genetic algorithms. *Available at SSRN 4448024*, 2023.

[121] Manuel R Vargas, Beatriz SLP De Lima, and Alexandre G Evsukoff. Deep learning for stock market prediction from financial news articles. In *2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, pages 60–65. IEEE, 2017.

[122] Manuel R Vargas, Carlos EM Dos Anjos, Gustavo LG Bichara, and Alexandre G Evsukoff. Deep leaming for stock market prediction using technical indicators and financial news articles. In *2018 international joint conference on neural networks (IJCNN)*, pages 1–8. IEEE, 2018.

[123] Adam Atkins, Mahesan Niranjan, and Enrico Gerding. Financial news predicts stock market volatility better than close price. *The Journal of Finance and Data Science*, 4(2):120–137, 2018.

[124] Jaeyoon Kim, Jangwon Seo, Minhyeok Lee, and Junhee Seok. Stock price prediction through the sentimental analysis of news articles. In *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 700–702. IEEE, 2019.

[125] Matin N Ashtiani and Bijan Raahmei. News-based intelligent prediction of financial markets using text mining and machine learning: A systematic literature review. *Expert Systems with Applications*, page 119509, 2023.

[126] Satya Verma, Satya Prakash Sahu, and Tirath Prasad Sahu. Stock market forecasting with different input indicators using machine learning and deep learning techniques: A review. *Engineering Letters*, 31(1), 2023.

[127] Zhaoxia Wang, Zhenda Hu, Fang Li, Seng-Beng Ho, and Erik Cambria. Learning-based stock trending prediction by incorporating technical indicators and social media sentiment. *Cognitive Computation*, 15(3):1092–1102, 2023.

[128] Kia Teymourian, Malte Rohde, and Adrian Paschke. Knowledge-based processing of complex stock market events. In *Proceedings of the 15th International Conference on Extending Database Technology*, pages 594–597, 2012.

[129] Bin Weng, Mohamed A Ahmed, and Fadel M Megahed. Stock market one-day ahead movement prediction using disparate data sources. *Expert Systems with Applications*, 79:153–163, 2017.

[130] Marcin Chlebus, Michał Dyczko, Michał Woźniak, et al. Nvidia's stock returns prediction using machine learning techniques for time series forecasting problem. Technical report, 2020.

[131] Andrea Picasso, Simone Merello, Yukun Ma, Luca Oneto, and Erik Cambria. Technical analysis and sentiment embeddings for market trend prediction. *Expert Systems with Applications*, 135:60–70, 2019.

[132] Noella Nazareth and Yeruva Yenkata Ramana Reddy. Financial applications of machine learning: A literature review. *Expert Systems with Applications*, page 119640, 2023.

[133] Riccardo Poli, William Langdon, and Nicholas McPhee. A field guide to genetic programming. 2009.

[134] JRGP Koza. On the programming of computers by means of natural selection. *Genetic programming*, 1992.

[135] Astro Teller and Manuela Veloso. Program evolution for data mining. *International Journal of Expert Systems Research and Applications*, 8(3):213–236, 1995.

[136] Julian F Miller et al. An empirical study of the efficiency of learning boolean functions using a cartesian genetic programming approach. In *Proceedings of the genetic and evolutionary computation conference*, volume 2, pages 1135–1142, 1999.

[137] Julian F Miller and Stephen L Smith. Redundancy and computational efficiency in cartesian genetic programming. *IEEE Transactions on evolutionary computation*, 10(2):167–174, 2006.

[138] Julian Francis Miller and Simon L Harding. Cartesian genetic programming. In *Proceedings of the 10th annual conference companion on Genetic and evolutionary computation*, pages 2701–2726, 2008.

[139] Julian Francis Miller. Cartesian genetic programming: its status and future. *Genetic Programming and Evolvable Machines*, 21:129–168, 2020.

[140] Nichael Lynn Cramer. A representation for the adaptive generation of simple sequential programs. pages 183–187. Grefenstette J (ed) Proceedings of an International Conference on Genetic Algorithms and the ApplicationsCarnegie-Mellon University, Pittsburgh, PA, USA, 1985.

[141] Wolfgang Banzhaf, Peter Nordin, Robert E Keller, and Frank D Francone. *Genetic programming: an introduction: on the automatic evolution of computer programs and its applications*. Morgan Kaufmann Publishers Inc., 1998.

[142] Léo Françoso Dal Piccol Sotto, Paul Kaufmann, Timothy Atkinson, Roman Kalkreuth, and Márcio Porto Basgalupp. Graph representations in genetic programming. *Genetic Programming and Evolvable Machines*, 22(4):607–636, 2021.

[143] Lee Altenberg et al. The evolution of evolvability in genetic programming. *Advances in genetic programming*, 3:47–74, 1994.

[144] Torsten Hildebrandt and Jürgen Branke. On using surrogates with genetic programming. *Evolutionary computation*, 23(3):343–367, 2015.

[145] Marc García-Arnau, Daniel Manrique, Juan Rios, and Alfonso Rodríguez-Patón. Initialization method for grammar-guided genetic programming. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 32–44. Springer, 2006.

[146] Pablo Ramos Criado, D Barrios Rolanía, Daniel Manrique, and Emilio Serrano. Grammatically uniform population initialization for grammar-guided genetic programming. *Soft Computing*, 24(15):11265–11282, 2020.

[147] William M Spears and Vic Anand. A study of crossover operators in genetic programming. In *International symposium on methodologies for intelligent systems*, pages 409–418. Springer, 1991.

[148] Riccardo Poli, Jonathan Page, and WB Langdon. Smooth uniform crossover, sub-machine code gp and demes: A recipe for solving high-order boolean parity problems. In *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, pages 1162–1169, 1999.

[149] Riccardo Poli, William B Langdon, et al. On the search properties of different crossover operators in genetic programming. *Genetic programming*, pages 293–301, 1998.

[150] Kumar Chellapilla. Evolving computer programs without subtree crossover. *IEEE Transactions on Evolutionary Computation*, 1(3):209–216, 1997.

[151] Chang-Yong Lee and Xin Yao. Evolutionary programming using mutations based on the lévy probability distribution. *IEEE Transactions on Evolutionary Computation*, 8(1):1–13, 2004.

[152] Raphael Crawford-Marks and Lee Spector. Size control via size fair genetic operators in the pushgp genetic programming system. In *GECCO*, pages 733–739, 2002.

[153] William B Langdon and WB Langdon. Genetic programming—computers using "natural selection" to generate programs. *Genetic Programming and Data Structures: Genetic Programming+ Data Structures= Automatic Programming!*, pages 9–42, 1998.

[154] Andres Felipe Cruz-Salinas and Jonatan Gomez Perdomo. Self-adaptation of genetic operators through genetic programming techniques. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 913–920, 2017.

[155] Alexandros Agapitos, Michael O'Neill, and Anthony Brabazon. Evolutionary learning of technical trading rules without data-mining bias. In *Parallel Problem Solving from Nature, PPSN XI: 11th International Conference, Kraków, Poland, September 11-15, 2010, Proceedings, Part I 11*, pages 294–303. Springer, 2010.

[156] Craig W Reynolds. An evolved, vision-based behavioral model of coordinated group motion. *From animals to animats*, 2:384–392, 1993.

[157] Gilbert Syswerda. A study of reproduction in generational and steady-state genetic algorithms. In *Foundations of genetic algorithms*, volume 1, pages 94–101. Elsevier, 1991.

[158] David E Goldberg. Genetic algorithms in search. *Optimization, Machine Learning*, 1989.

[159] William B Langdon et al. Evolving data structures with genetic programming. In *ICGA*, pages 295–302, 1995.

[160] Conor Ryan. Pygmies and civil servants. In *Advances in Genetic Programming*, pages 243–263. 1994.

[161] Nicholas Freitag McPhee, Nicholas J Hopper, et al. Analysis of genetic diversity through population history. In *Proceedings of the genetic and evolutionary computation conference*, volume 2, pages 1112–1120. Citeseer, 1999.

[162] Sean Luke and Lee Spector. Evolving teamwork and coordination with genetic programming. *Genetic Programming*, 96:150–56, 1996.

[163] Zoltan A Kocsis and Jerry Swan. Genetic programming+ proof search= automatic improvement. *Journal of Automated Reasoning*, 60(2):157–176, 2018.

[164] Artem Sokolov, Darrell Whitley, and Andre' da Motta Salles Barreto. A note on the variance of rank-based selection strategies for genetic algorithms and genetic programming. *Genetic Programming and Evolvable Machines*, 8:221–237, 2007.

[165] David J Montana. Strongly typed genetic programming. *Evolutionary computation*, 3(2):199–230, 1995.

[166] Guilherme Espada, Leon Ingelse, Paulo Canelas, Pedro Barbosa, and Alcides Fonseca. Data types as a more ergonomic frontend for grammar-guided genetic programming. In *Proceedings of the 21st ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences*, pages 86–94, 2022.

[167] Peter J Angeline and Kenneth E Kinnear. On using syntactic constraints with genetic programming. 1996.

[168] Peter A Whigham et al. Grammatically-based genetic programming. In *Proceedings of the workshop on genetic programming: from theory to real-world applications*, volume 16, pages 33–41. Citeseer, 1995.

[169] John W Backus. The syntax and the semantics of the proposed international algebraic language of the zurich acm-gamm conference. In *ICIP Proceedings*, pages 125–132, 1959.

[170] Robert I McKay, Nguyen Xuan Hoai, Peter Alexander Whigham, Yin Shan, and Michael O'neill. Grammar-based genetic programming: a survey. *Genetic Programming and Evolvable Machines*, 11:365–396, 2010.

[171] William F Sharpe. The Sharpe ratio. *Streetwise–the Best of the Journal of Portfolio Management*, 3:169–185, 1998.

[172] James Brookhouse, Fernando EB Otero, and Michael Kampouridis. Working with opencl to speed up a genetic programming financial forecasting algorithm: Initial results. pages 1117–1124, 2014.