Self Adaptive Voting System for Stock Market Investment Strategy based on Evolutionary Computing

Gonçalo Lopes da Cruz Capela

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor: Professor Doutor Nuno Cavaco Gomes Horta

Examination Committee:
Chairperson: Professor Doutor António Manuel Raminhos Cordeiro Grilo
Supervisor: Professor Doutor Nuno Cavaco Gomes Horta
Members of the Committee: Professor Doutor João Paulo Baptista de Carvalho

November 2018
Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.
Resumo

Os movimentos do mercado financeiro são altamente influenciados por fatores complexos que dificultam a tarefa de lucrar desses movimentos. Os investidores costumam usar estratégias que os ajudam a determinar quando comprar ou vender ações. As regras técnicas têm sido amplamente utilizadas nos mercados financeiros há mais de um século como ferramentas analíticas para avaliar a segurança de um determinado investimento.

Este trabalho descreve o desenvolvimento de uma aplicação, baseada em uma técnica de computação evolucionária, em particular, o NSGAI, e visa gerar pontos de entrada e saída de investimentos utilizando indicadores de análise técnica (EMA, RSI, MACD, entre outros). Para validar a solução desenvolvida, é definida uma avaliação completa, comparando a estratégia desenvolvida com outros métodos de investimento, como Buy & Hold. Para testar a solução sob diferentes condições de mercado, incluindo a mais recente quebra financeira foram usados diferentes horizontes temporais. Os resultados são promissores, pois a solução atual pode superar as outras estratégias durante o crash. O estudo de caso mais extenso resultou em um retorno do investimento 55 vezes maior do que a estratégia Buy & Hold.

Palavras-chave: Análise Técnica, Computação evolutiva, mercados financeiros, Otimização, Investimentos.
Abstract

Financial market movements are highly influenced by complex factors that make it difficult to profit from these movements. Investors often use strategies that help them determine when to buy or sell stocks. Technical rules have been widely used in financial markets for more than a century as analytical tools to assess the safety of a given investment.

This work describes the development of an application, based on an evolutionary computation technique, in particular, the NSGAII, and aims to generate entry and exit points of investments using technical analysis indicators (EMA, RSI, MACD, among others). To validate the developed solution, a complete evaluation is defined, comparing the strategy developed with other investment methods, such as Buy & Hold. To test the solution under different market conditions, including the latest financial crash, different time horizons were used. The results are promising, as the current solution outperforms other strategies during the crash. The most extensive case study resulted in a return on investment 55 times higher than the Buy & Hold strategy.

Keywords: Evolutionary Computation, Financial Markets, Investments, Optimization, Technical Analysis.
Acknowledgements

Many factors contributed to the success of this work, but the most important and special is the constant guidance, feedback and support of my supervisor, Professor Nuno Horta.

I would also like to thank my family and friends who have also been a great help in developing this work.
Contents

1. Introduction .................................................................................................................. 1
   1.1. Domain Background ................................................................................................. 1
   1.2. Motivation ................................................................................................................ 2
   1.3. Goals.......................................................................................................................... 2
   1.4. Document Structure ................................................................................................. 3
2. Background .................................................................................................................... 5
   2.1. Market ....................................................................................................................... 5
       2.1.1. Data Series ......................................................................................................... 5
       2.1.2. Transactions ..................................................................................................... 6
       2.1.3. Long vs Short .................................................................................................... 7
   2.2. Market Analysis ....................................................................................................... 7
       2.2.1. Fundamental analysis ....................................................................................... 8
       2.2.2. Technical analysis ............................................................................................ 9
       2.2.3. Strategies ......................................................................................................... 11
       2.2.4. Rate of Return and volatility ............................................................................ 11
3. State-of-the-Art ............................................................................................................ 13
   3.1. General Architecture ............................................................................................... 13
   3.2. Data processing ....................................................................................................... 15
   3.3. Optimization / Machine Learning .......................................................................... 15
       3.3.1. Neural Networks ............................................................................................. 15
       3.3.2. Clustering ......................................................................................................... 16
       3.3.3. Ensemble learning .......................................................................................... 16
       3.3.4. Support vector machine ................................................................................... 17
       3.3.5. Particle swarm optimization ............................................................................. 17
       3.3.6. Genetic algorithm ........................................................................................... 18
   3.4. Other works ............................................................................................................. 20
   3.5. Overview and Discussion ....................................................................................... 21
4. Proposed Architecture .................................................................................................. 25
   4.1. Overall Architecture ............................................................................................... 25
   4.2. Data Flow ............................................................................................................... 26
   4.3. Development Environment ................................................................................... 27
   4.4. Investment Simulator Module ............................................................................... 28
6.2. Future Work ........................................................................................................ 65
6.2.1. Possible Improvements .............................................................................. 65

Bibliography ............................................................................................................. 67

Appendix A – Development Description ................................................................. 71
Appendix B – User Documentation ......................................................................... 73
List of Tables

Table 1 - Some Technical Indicators [12] ...................................................................................................................... 10
Table 2 – Table resuming all references ........................................................................................................................ 23
Table 3 - MA Rule ..................................................................................................................................................... 33
Table 4 - EMA Rule .................................................................................................................................................... 34
Table 5 - RSI Rule ...................................................................................................................................................... 35
Table 6 - Parameters for Case Study I ........................................................................................................................ 50
Table 7 - NSGAII Parameters for Case Study I ......................................................................................................... 51
Table 8 - Results. Intervals with 95% confidence for Case Study I ............................................................................ 52
Table 9 - Genome for each period .............................................................................................................................. 55
Table 10 - Parameters for Case Study II .................................................................................................................. 57
Table 11 - NSGAII Parameters for Case Study II ..................................................................................................... 58
Table 12 - Results for Case Study II. Intervals with confidence of 95% ................................................................. 59
Table 13 - Parameters for Case Study III ................................................................................................................ 61
Table 14 - NSGAII Parameters for Case Study III .................................................................................................. 61
Table 15 - Results for Case Study III. Intervals with confidence of 95% ............................................................. 62
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Fundamental analysis aspects</td>
</tr>
<tr>
<td>7</td>
<td>General architecture for the system</td>
</tr>
<tr>
<td>8</td>
<td>General Architecture for [13] (Top) and [15] (Bottom). Images retrieved from the corresponding references</td>
</tr>
<tr>
<td>9</td>
<td>General Architecture for [14]. Image retrieved from the corresponding reference</td>
</tr>
<tr>
<td>10</td>
<td>Neural Network model [16]</td>
</tr>
<tr>
<td>11</td>
<td>Ensemble learning model of a rule generating system [18]</td>
</tr>
<tr>
<td>12</td>
<td>Generating Rules [20]</td>
</tr>
<tr>
<td>13</td>
<td>Decision tree of the previous paper [21]</td>
</tr>
<tr>
<td>14</td>
<td>An intelligent hybrid trading system for discovering trading rules [24]</td>
</tr>
<tr>
<td>15</td>
<td>General Architecture</td>
</tr>
<tr>
<td>16</td>
<td>Data Flow</td>
</tr>
<tr>
<td>17</td>
<td>Application UML</td>
</tr>
<tr>
<td>18</td>
<td>Investment Simulator Module UML</td>
</tr>
<tr>
<td>19</td>
<td>Sliding Window</td>
</tr>
<tr>
<td>20</td>
<td>Technical Rules Module UML</td>
</tr>
<tr>
<td>21</td>
<td>EMA rule illustration</td>
</tr>
<tr>
<td>22</td>
<td>RSI Rule Illustration</td>
</tr>
<tr>
<td>23</td>
<td>MACD illustration</td>
</tr>
<tr>
<td>24</td>
<td>NSGAII Non Dominated Sorting, [35]</td>
</tr>
<tr>
<td>25</td>
<td>Tournament Selection [36]</td>
</tr>
<tr>
<td>26</td>
<td>Mutation Operator – Flip Bit [37]</td>
</tr>
<tr>
<td>27</td>
<td>Uniform Crossover [38]</td>
</tr>
<tr>
<td>28</td>
<td>Pareto Front</td>
</tr>
<tr>
<td>29</td>
<td>Optimization Module UML</td>
</tr>
<tr>
<td>30</td>
<td>Thresholds illustration</td>
</tr>
<tr>
<td>31</td>
<td>Voting Based Decision Module process visualization</td>
</tr>
</tbody>
</table>
Figure 30 - Voting Based Decision Module UML................................................................. 48
Figure 33 - ROI evolution for Case Study I........................................................................... 51
Figure 34 - Histogram for Case Study I.................................................................................. 52
Figure 35 - Positions taken on each period............................................................................ 53
Figure 36 - Yearly ROI for the voting system and Buy & Hold.................................................. 53
Figure 37 - Investment Evolution of the different indicators ..................................................... 54
Figure 38 - Visualization of the sliding window period on price chart....................................... 54
Figure 39 - ROI Evolution for a strategy without short positions for Case Study I .................. 56
Figure 40 - Detail in ROI Evolution. ....................................................................................... 56
Figure 41 - ROI Evolution for Case Study II .......................................................................... 58
Figure 42 - Positions taken on each period for Case Study II.................................................... 59
Figure 43 - ROI Evolution for a strategy without short positions for Case Study II ................. 60
Figure 44 - ROI Evolution for Case Study III ......................................................................... 62
Figure 45 - Positions taken on each period for Case Study III................................................ 63
Figure 46 - ROI Evolution for a strategy without short positions for Case Study III ............... 63
Figure 47 - Application Start.................................................................................................... 73
List of Acronyms

Optimization Related

- **GA** Genetic Algorithm
- **GP** Genetic Programming
- **SVM** Support Vector Machines
- **ANN** Artificial Neural Networks
- **PSO** Particle Swarm Optimization
- **TVPSO** Time Varying Particle Swarm Optimization
- **EC** Evolutionary Computation
- **EA** Evolutionary Algorithm
- **MO** Multi-objective
- **SO** Single-objective
- **MOOP** Multi-objective Optimization Problem
- **PF** Pareto Frontier
- **NSGAII** Non-dominated Sorting Genetic Algorithm II
- **GNP** Genetic network Programming

Technical Analysis Related

- **SMA** Simple Moving Average
- **EMA** Exponential Moving Average
- **WMA** Weighted Moving Average
- **HMA** Hull Moving Average
- **ROC** Rate of Change
- **RSI** Relative Strength Index
- **MACD** Moving Average Convergence Divergence
- **OBV** On Balance Volume
- **TSI** True Strength Index

Investment Related

- **ROI** Return On Investment
- **B&H** Buy and Hold
- **IS** Investment Simulator

Other

- **UML** Unified Modeling Language
1. Introduction

This chapter works as an introduction to the subject of the financial market. Firstly, a domain background is presented as a way to introduce this study. The motivations and the goals for this work are then described, and finally, the document structure is introduced.

1.1. Domain Background

The financial market is an essential part of the economy and, from investors to central banks, there are many different roles to play in this area. Companies can finance their operations and protect their debts against currency fluctuations, all types of investors can vary their capital allocation, and speculators can profit from short-term moves [1].

Investment in this market is one of the significant investment options for investors to make a high profit. However, the high return rates are associated with high-risk investments. It is very common for stock prices to fluctuate, causing doubts about the investment options, making it difficult for investors to decide on when to buy or sell particular stocks to maximize returns.

The movements of the financial markets are highly influenced by complex factors, which make the task of profiting from these movements challenging. Stock prices are volatile and sensitive, can rise or fall in response to political, macro or microeconomic factors. A change in the interest rate can affect the entire equity market, just as an event of a company can affect the value of its stock or even the value of shares of other companies that have any business relationship. Hence it is complicated to predict all the events that can influence the movement of the values of a stock, and also predict the market response to these events.

Investors often use strategies that help them determine when to buy or sell stocks. Many strategies can be found in the literature, and many more strategies are developed, tailored so that they can be used to assess markets. An essential approach to operating the stock market is the Technical Analysis that is defined [2] as "the study of market action for the purpose of forecasting future price trends "

Technical rules are widely used in financial markets for more than a century as analytical tools for assessing the safety of a given investment. These technical indicators usually try to predict the future trend of market prices by analyzing historical values [2].
In the stock exchange markets, the buy-and-hold approach is a well-known strategy among traders. If a company and its industry seem to be promising, the trader buys and stores their assets over a relatively long period. An alternative approach, known as market timing, is more dynamic and focuses on short-term fluctuations. Through technical analysis, trading rules are designed to generate appropriate buy and sell signals in short periods of time.

1.2. Motivation

Optimizing the investments in the stock market is of extreme importance for investors since they manage to increase the return on investment (ROI). Increasing profits will most often also increase the risk of the investment. To counter this risk, it is possible to develop some rules that support the investor in the decisions to be made. However, these rules are not always easy to compose, much less to optimize. This process of creating strategies can be simplified by using fewer data and fewer indicators, but in doing so may result in losing some of the information. To make this process friendlier to the investor is possible to automate the creation and optimization of the rules, giving the user buy or sell signals.

1.3. Goals

This work is part of the development of an application for the support of the investor in decision-making on the stock exchange.

The module to be developed is one capable of generating and optimizing technical rules and generating buy or sell signals taking into account the historical data from a specified stock. These buy and sell signals will be generated from several technical indicators/rules using a voting system (rewarding indicators with good performance). To do this it is essential to identify which indicators are consistently resulting in good return rates, so it is possible to reward those.

While the work is being developed, is also necessary to do some tests to evaluate its performance. For these tests to be scientifically valid, they must be done within many varied market types. For example, high volatility and value range -> stocks, high volume and limited value range -> forex, limited volume, and value range -> commodities. So, to analyze the performance of the developed module, tests with these characteristics will be conducted.
Goals:

- Identify technical indicators that are consistently associated with good rates of return.
- Create a module for the formulation and optimization of technical rules based on technical indicators, and with them generate buy or sell signals with a voting-based process.
- Test the system with various types of market.

1.4. Document Structure

The document is structured as follows:

- **In chapter 2** a background is presented. This chapter works as a way to introduce to some financial definitions important to really understand the work done in this study.

- **In chapter 3** the State of the Art is shown. Various works already done in the financial area using machine learning algorithms are here described. Many of these works try to support the investor in the decision-making process.

- **The chapter 4** illustrates the solution’s architecture of the developed application. Here the architecture of the module developed is presented, as also as the explanation of each sub-module. All inputs and outputs of the system are here also defined.

- **Chapter 5** proposes the validation procedure to evaluate the developed system by providing an exhaustive study on the solution’s performance and robustness.

- **Chapter 6** summarizes the provided report and supplies the respective conclusion and future work.
2. Background

This chapter addresses some of the most fundamental concepts to understand the two major domains in this work, namely, Computational Finance and Machine Learning. The definitions presented in this chapter are essential to understanding the work presented in this document.

2.1. Market

2.1.1. Data Series

A time series is a sequence of observations of a variable over time. It is a sequence of points (numerical data) in successive order, usually occurring in uniform intervals, for example, one day, one week, one month.

In investing, it is common to use a time series to track the price of a security over time. This can be tracked over the short term, such as the price of a security on the hour throughout a business day, or the long-term, such as the price of a security at close on the last day of every month over the course of five years. Some of the most common variables used in this area are opening price, closing price, maximum, minimum and the volume. All this information can be presented using a candlestick chart.

Figure 1 - Candlestick Chart example with a daily period. Retrieved from investing.com [3]
In Figure 1 an example of a candlestick chart [4] is presented, where each bar represents one day, so it is possible to extract the opening and closing price for each day, and the maximum and minimum price as well. Below the candlestick chart is a bar chart that represents the volume of transactions for each day. Here is important to explain that a green candlestick means that the closing price for the day is higher than the opening price of that same day, the red represents the opposite.

![Candlestick Chart](image)

*Figure 2 – Candlestick. Retrieved from Wikipedia [5]*

In the figure above two candlesticks are presented, one in a bullish market and the other in a bearish market. A bull market is a market in which prices are rising or are expected to rise while a bear market is a market where prices are falling or expected to fall.

All this data must be retrieved from somewhere, and in this case, the source of the data is a fundamental issue. There are many sources for this information online, some of them are “yahoo finance”, “google finance”. The most important thing is to get clear and correct data.

### 2.1.2. Transactions

A trade or transaction occurs after the buyer and seller agree on a price for the security. For this, a buyer can set a bid price, the maximum price that he is willing to pay for a security. A seller can define an asking price, the minimum price that he is willing to receive.

To do this transaction an investor need a broker. A broker is an individual or firm that charges a fee or commission for executing buy and sell orders submitted by an investor.
Transaction costs are expenses incurred when buying or selling a good or service. Transaction costs represent the labor required to bring a good or service to the market, giving rise to entire industries dedicated to facilitating exchanges.

2.1.3. **Long vs Short**

When an investor is in a long position in a stock, he has bought it expecting the price to rise. In a long position, there is the risk of the stock price falling, in which case the investor will lose money. To establish a long position, investors buy shares of stock and wait for the price to rise. Once it does, the investor has a decision to make. The gain exists only unofficially until is converted to cash by selling the shares. It can be sold, skipping the chance to make more money if the price rises further. The other option is to hold the shares in anticipation of higher profits, risking that the price will fall and wipe out all gains.

In a short position, the investor is doing just the opposite. "Going short" is considerably more complicated than going long. First, the investor borrows some shares of the stock from the broker. Then he or she sells those shares on the open market at the market price. Then the investor waits for the stock price to fall. When it does, she or he goes back into the market and buys the same number of shares sold before. Finally, the investor refunds those shares to the broker.

![LONG = BUY vs SHORT = SELL](image)

*Figure 3 - Long vs Short. Retrieved from tourmaketer [6]*

2.2. **Market Analysis**

There are two main schools of thought to financial market analysis, technical and fundamental analysis. The first is focused on the statistical analysis of price movements, while
fundamental analysis involves analyzing financial statements to determine the fair value of the business.

2.2.1. Fundamental analysis

![Diagram showing fundamental analysis aspects]

Figure 4 - Fundamental analysis aspects

The major part of fundamental analysis involves the in-depth analysis of the financial statements to measure its intrinsic value [7]. It uses revenue, expenses, assets, liabilities and all the other financial aspects of a company and other data to determine a company’s underlying value and potential for future performance [8].

Fundamentalists defend that the stock price tends to move in the direction of the “real value” or “intrinsic value”. If the stock price is below the “real value” than the investor can decide to buy the stock because the stock is going to move towards the “real value” (going up), so if the stock price is above the “real value”, the investor can sell a stock because according to this schools of thought the stock price is going down [8]. Because of this, this technique tends to be used generally on the long-term investment.

On the one hand, this technique is a systematic approach, and it can predict changes before they show up in the charts. On the other hand, it is hard to formalize this knowledge for purposes of automation (with machine learning for example).
2.2.2. Technical analysis

Technical analysis [9] is a method of evaluating stocks that involve analyzing statistics generated by market activity, past prices, and volume. This technique does not attempt to measure an intrinsic stock value, but rather, looks for peaks, bottoms, trends and other factors affecting a stock’s price. This assumes that the value of a stock price depends on their past behavior, like past values and other correlated variables [10].

Some of the forms of technical analysis rely on chart pattern, other use technical indicators, and oscillators or some combinations of tools. For any of these previous techniques used the historical price and volume data are used, so the only thing that matter is the past trading data and the information that this data might provide about future price movements.

A technical indicator is a series of data that derived by applying a formula to the price data. Price data can be any combination like the open, high, low or close values. Many technical indicators have been developed, and new variants continue to be developed by traders with the aim of getting better results. New indicators are often backtested on historical prices and volume data to see how effective they would have been.

Although all indicators provide some additional information about the stock, using all the available indicators would make the analyzing process very complicated and slow. So it is crucial to identify which indicators to use without increasing the complexity of the system too much.

There are four basic categories of technical indicators: Trend, Momentum, Volatility, and Volume. These indicators are used to develop strategies, and the type of indicator to be used depends on the strategy intended. For example, a trader preferring a long-term investment might focus on a trend-following strategy. A trader interested in small moves follows a strategy based on volatility.

One of the most known trend indicators is the Moving Average. This indicator smooths the price data, and with this, it does not predict the price direction but instead informs the current direction. The two most used MAs are the simple moving average and the exponential moving average.

The simplest MA is the simple moving average, which gives the same weight to all days, for example, a 50-day moving average sums the closing price of the last 50 days and divides by 50. This indicator lags because it is based on past prices. One indicator with less lag is the exponential moving average. This MA applies more weight to recent prices.
Figure 5 - 15 and 50 day moving average. Retrieved from Investopedia [11]

In the graph above two moving averages are presented, one with a period of 15 days, and another with 50 days. A lot more of technical indicators are available, and here is presented the most commonly used:

Table 1 - Some Technical Indicators [12]

<table>
<thead>
<tr>
<th>Technical Indicators</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moving Average</td>
<td>$MA_n = \frac{\sum_{i=1}^{n} D_i}{n}$</td>
</tr>
<tr>
<td>Exponential Moving Average</td>
<td>$EMA(t) = (\text{Close}(t) - EMA(t-1)) \times \text{multiplier} + EMA(t-1)$</td>
</tr>
<tr>
<td>Moving Average Convergence and Divergence</td>
<td>$MACD \text{ Line } = EMA_{12} - EMA_{26}$</td>
</tr>
<tr>
<td></td>
<td>$\text{Signal Line } = EMA_{9} \text{ of MACD Line}$</td>
</tr>
<tr>
<td>Relative Strength Index</td>
<td>$RSI = 100 - \frac{100}{1 + RS}$</td>
</tr>
<tr>
<td></td>
<td>$RS = \frac{\text{Average Gain}}{\text{Average Loss}}$</td>
</tr>
<tr>
<td>On Balance Volume</td>
<td>$\text{If } \text{Close}(t) &gt; \text{Close}(t-1),$</td>
</tr>
<tr>
<td></td>
<td>$OBV(t) = OBV(t-1) + \text{volume}(t)$</td>
</tr>
<tr>
<td></td>
<td>$\text{If } \text{Close}(t) &lt; \text{Close}(t-1),$</td>
</tr>
<tr>
<td></td>
<td>$OBV(t) = OBV(t-1) - \text{volume}(t)$</td>
</tr>
<tr>
<td></td>
<td>$\text{If } \text{Close}(t) = \text{Close}(t-1),$</td>
</tr>
<tr>
<td></td>
<td>$OBV(t) = OBV(t-1)$</td>
</tr>
</tbody>
</table>

Some may be considered quite easy to compute while others may be seen as more laborious, but one thing in common to all technical indicators is the fact that all can be used to
create strategies, and generate buy or sell signals (define entry and exiting points). The illustration below represents several technical indicators and their values. Accordingly to the value, a recommended action is also presented. Some indicators recommend selling, others suggest buying, and others recommend some other action.

![Table of Technical Indicators](image)

**Figure 6 - Example of Actions from several Technical Indicators. Retrieved from Investing [3]**

### 2.2.3. Strategies

Strategies objectively use indicators to create rules for defining entry and exit points. A strategy is a set of definitive rules that determine the exact conditions upon which there will be a trade. Strategies usually include the use of indicators, or more often multiple indicators, to define when some action occurs.

Using the Moving averages presented in picture 5 a strategy can be developed. When the two moving averages crossover a change in the trend is taking place, in this case signaling an uptrend, this can be used to generate a buy signal.

### 2.2.4. Rate of Return and volatility

A rate of return is the gain or loss on an investment over a specified period, expressed as a percentage of the investment’s cost.
$ROR = \frac{Current\ Price - Original\ Price}{Original\ Price} \times 100 \tag{1}$

Volatility is a statistical measure of the dispersion of returns for a particular security or market index. Volatility can be measured using the standard deviation or the variance between the returns of the same security or market index. Usually, the higher the volatility, the riskier the security or market index is. Many investors should be aware that standard deviation is the typical statistic tool used to measure volatility. The standard deviation is defined merely as the square root of the average squared deviation of the data from its mean.

The Sharpe Ratio is the average return earned over the risk-free rate per unit of volatility or total risk. Subtracting the risk-free rate from the mean return the performance associated with risk-taking activities can be isolated. One intuition of this calculation is that a investment engaging in “zero risk” investment has a Sharpe ratio of precisely zero. Generally, the higher the value of the Sharpe ratio, the more attractive the risk-adjusted return.
3. State-of-the-Art

In recent years there have been numerous attempts to improve the profit of financial trading. In these attempts, all kind of machine learning tools have been used such as Neural Networks, Genetic Algorithms, Support Vector Machines, and others.

The architectures of these attempts can be divided into different parts of data processing so that the solution will incorporate different submodules, each with different functionalities. The first phase, data capture will be responsible for acquiring the data and compute some indicators from them if necessary. Next will be the generation of buy or sell signals with Evolutionary algorithm. Finally, a module to evaluate and simulate the resulting strategy (also to be able to implement a reward-based system).

For this purpose, it is necessary to analyze the state of the art for all the stages of this process, so in this chapter and in a later section some articles will be presented to introduce various Machine Learning methods applied to the financial market.

3.1. General Architecture

The architecture illustrated in figure 7 is the base model for most of the applications explored in this section. Starting with data intake and processing, followed by the optimization process alongside the simulator and ending with outputting the results.

A. Gorgulho et al.[13] proposes a traditional layer architecture composed of three distinct layers to solve the portfolio management and technical analysis optimization. These three layers
consist of Data Processing, Optimization Layer, and Strategy layer, and each layer is composed of several modules.

With a more detailed analysis A. Gorgulho et al. [13], J.M. Pinto et al. [14] and A. Silva et al. [15] propose very similar architectures with minor differences. All these works try to optimize in some way the strategies based on technical indicator rules taking as inputs the user settings and the historical prices of the stock to be optimized. The optimization process of these works is done based on technical rules, and as expected all of them have technical rules modules, also an investment simulator as a way to find the performance of each strategy.

![General Architecture](image)

*Figure 8 - General Architecture for [13] (Top) and [15] (Bottom). Images retrieved from the corresponding references.*
3.2. Data processing

Whether optimizing the parameters of the technical indicators or optimizing strategies based on technical indicators a standard module is the data intake and their respective processing, such as the computation of the technical indicators and their rules.

The work of A. Gorgulho et al. [13] proposes a solution for the data intake process, and it describes an architecture where the data layer is divided into two modules, the financial data processing module and the technical rules module. The financial data processing module is accountable for processing all the financial data downloading a complete history of all the available data on distinct markets. All the financial data relative to the former index was downloaded through the Yahoo Finance Database. In the technical rules module, several technical indicators are computed in order to be used in the optimizations process.

3.3. Optimization / Machine Learning

In this section several works are presented and described

3.3.1. Neural Networks

Several works have been done using Neural Networks to improve return on the financial market. For example, Sabaithip Boonpeng et al. [16] compared several multi-binary classification techniques using neural networks. They used One-Against-One (OAO) and One-Against-All
(OAA) techniques and compared them with the traditional neural network. Results show that OAA-NN outperforms OAO-NN and the multi-class classification using a single NN.

![Figure 10 – Neural Network model [16]](image)

3.3.2. Clustering

Qinghua Huang et al. [17] propose a solution to mine trading rules from trading data. The authors used a biclustering algorithm and the k nearest neighbor to mine patterns as trading rules and classified as three trading rules (buy, sell and no-action). It was implemented on four historical datasets, and then the results were compared with the conventional buy-and-hold strategy. The experimental results demonstrate that the proposed trading system outperforms the buy-and-hold strategy.

3.3.3. Ensemble learning

Ensemble learning is a machine learning method that used multiple learning algorithms to obtain better performance than all algorithms alone. This approach allows a much more flexible structure. Using this method to solve stock trading problems is also a possibility as described in [18]. In this study, several classifiers were created, and the classification is made by combining the results generated by the classifiers. Bootstrap aggregating is one of the ensemble learning methods, which creates several classifiers using bootstrapping samples, and makes classification by majority vote. To generate the rules a genetic network is used. After this, all rules are stored, and a voting method is used to generate the sell or buy signal. The weights of the weighted voting system are the output of a Multi-layer perceptron. After the multi-layer generates the vote weight of each rule the buy or sell rule is easily calculated.
For the results three systems were tested, the proposed solution (GNP with ensemble), GNP without ensemble and “buy-and-hold”. The results show that the ensemble method has better results than the solution without ensemble in 10 of the 16 tested cases. Comparing with the “buy-and-hold” it is shown that the proposed solution is better in 12 out of the 16 tests. This show that the ensemble learning enhances the performance of GNP.

3.3.4. Support vector machine

In [19] the authors use technical indicators as features in defining a predictive model based on Least Squares Support Vector Machines. The LS-SVM classifier is used to predict the trend of the stock indices. The outputs of the LS-SVM are binary signals that can be later used to define strategies. Comparing the results obtained from traditional statistical methods for predicting the trend of financial series and proposed LS-SVM model, it can be concluded that machine learning techniques capture the non-linear models which are dominant in the financial markets in an adequate way. Outperforming the results of Buy & Hold strategy and technical trading strategies, application of LS-SVM in the decision making process on investing in the financial market can significantly contribute to the maximization of profitability on investment.

3.3.5. Particle swarm optimization

In [20] Fei Wang et al. used an improved time variant particle swarm optimization algorithm to determine the best parameter values of Performance-based reward strategy. So in this paper, it was used a PRS to generate the sell/buy signals. It was used the two most popular
classes of technical trading rules - moving average (MA) and trading range break-out (TRB) as inputs in the PRS system, the algorithm then assigns a weight to all inputs, and sums all inputs multiplied by the respective weight. If the sum is above some threshold then is generated a buy signal, or a sell signal otherwise.

![Diagram](image)

*Figure 12 - Generating Rules [20]*

A Particle swarm optimization algorithm to arrange new and better weight combination for the PRS does the optimization of this system. So the PRS parameters are initialized using the particle’s position, and then the annual net profit generated by the PRS on the stock data is returned as the fitness. The out-of-sample test shows that PRS can be successfully optimized by TVPSO.

### 3.3.6. Genetic algorithm

Dmitry Iskrich et al. [21] suggest the use of an evolutionary algorithm to generate and select the most fitting trading rules for interday trading, creating long-term trading rules. For this it was used binary decision trees whose leaf nodes contain decisions, in this case, can be buy or sell decisions. The transition to its right or left node depends on the node’s value. In each non-terminal node, the value of a technical indicator is compared to a certain threshold to decide which child node to choose. To determine the best tree a genetic algorithm is used. The advantages of this approach are the simplicity of the generated rules, and the extensibility since the technical indicators are relative (from VERY LOW to VERY HIGH).
One of the problems in the solution above is that the rules obtained rely on random constant parameters. To handle this issue Sunisa Rimcharoen et al. [22] propose a new hybrid method for generating trading rules. This method uses a genetic algorithm to evolve the rules structure and uses evolution strategy to determine constant parameters. This way a rule is tailored to each particular stock, resulting in a better performance of this method than the MACD strategy.

Suriya Yodphet et al. [23] also proposed a genetic algorithm to find profitable rules, but this team proposes an improvement over the previous solution. This research suggests two modifications to the chromosome encoding resulting in a loss avoidance system. The first modification proposed by the authors is to add an exit signal to avoid loss and increase profitability. The second is to incorporate two more effective indicators: the relative strength index and the average directional index. The results show that the exit rule and stop loss percentage yield more profit.

An intelligent hybrid system is proposed by Kim Youngmin et al. [24] for discovering technical trading rules using a genetic algorithm. The first phase of analysis consists of generating a decision table (conditional attributes and decision attribute). The second stage is the rule discovery mechanism that consists of two steps. First, extract rules using rough set analysis, and then applying GA to discovering optimal decision rules. These two steps are repeated until the stopping condition of the GA is met. The third stage is the trading signal generation from the generated rules. For this study, a six-month training period and a set of 50 decision rules provided the highest annualized return rate compared to other experimental combinations. These results show that rough set analysis can help to generate decision rules in the rule discovery mechanism, while the GA can help to improve the decision rules.
As described before, A. Gorgulho et al. [13] uses GA to optimize technical analysis rules and optimizing portfolio composition. J.M. Pinto et al. [14] uses a Multi-Objective Evolutionary System to predict the future tendency of assets price and optimize a set of Trading or Investment Strategies.

### 3.4. Other works

A. Silva et al. [15] describe a new approach to portfolio management using stocks. A Multi-Objective Evolutionary Algorithms (MOEA) with two objectives, the return, and the risk, are used to optimize the models.

B. J. Almeida et al. [25] proposes a new algorithm capable of generating technical rules to make investments with a given amount of leverage depending on the certainty of the prediction presented. To forecast those predictions, a combination of a Support Vector Machine (SVM) algorithm and a Dynamic Genetic Algorithm.

J. Leitão et al. [26] describes a new pattern discovery approach based on the combination among rules between Perceptually Important Points (PIPs) and the Symbolic Aggregate approximation (SAX) representation optimized by Genetic Algorithm (GA).

J. Carapuço et al. [27] describes a new system for short-term speculation in the foreign exchange market, based on recent reinforcement learning (RL) developments.
3.5. Overview and Discussion

Table 1 shows a summary version of the State of the Art concerning recent approaches to the generation and optimization of investment strategies. It is possible to conclude that genetic algorithms have a high dominance in this area. However, despite this, some works use other algorithms to get the same results.

While analyzing the works described in this section, it has become difficult to compare different solutions to the same problem, because there is a lack of homogeneity in the evaluation of the results. As this makes it difficult to draw up a direct comparison only a table with the main characteristics of the various solutions is presented here.
Table 2 – Table resuming all references

<table>
<thead>
<tr>
<th>Reference</th>
<th>Year</th>
<th>Method</th>
<th>Data</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>[16]</td>
<td>2016</td>
<td>OAA-Neural Network</td>
<td>Buying Data, Holding Data, Selling Data</td>
<td>2 year of Stock Exchange of Thailand</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>OAA-NN outperforms other models</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Stochastic -&gt; 14.64% Return</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RSI -&gt; 28.81% Return</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MACD -&gt; -0.95% Return</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>OAA-NN -&gt; 57.67% Return</td>
</tr>
<tr>
<td>[21]</td>
<td>2017</td>
<td>Genetic Algorithm</td>
<td>Technical Indicators</td>
<td>12 years</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Higher return rate and sharpe ratio than the “buy and hold”</td>
</tr>
<tr>
<td>[22]</td>
<td>2014</td>
<td>Genetic Algorithm, Evolutionary Strategies</td>
<td>Simple Moving Average, Exponential Moving Average, Standard Deviation, Max and Min values</td>
<td>50 Stocks listed in the Stock Exchange of Thailand</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Proposed method – 3.94% Average Profit Return</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Moving Average Convergence</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Divergence - 0.52% Average Profit Return</td>
</tr>
<tr>
<td>[23]</td>
<td>2016</td>
<td>Genetic Algorithm, Evolutionary Strategies</td>
<td>Simple Moving Average, Exponential Moving Average, Standard Deviation, Max and Min values, Relative Strength Index, Average Direction Index</td>
<td>30 Stocks listed in the Stock Exchange of Thailand, High Dividend</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Proposed Technique – 19.53% Average Profit Return</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Moving Average Convergence</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Divergence - 2.10% Average Profit Return</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Annualized return rate</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Random approach - 5.66%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Correlation approach - 20.32%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>GA approach - 10.59%</td>
</tr>
<tr>
<td>[17]</td>
<td>2015</td>
<td>Biclustering, K nearest neighborhood</td>
<td>ROC, EMA, ADX, ATR, SMA, %R, RSI</td>
<td>Four Canadian stocks, Six major world indices, NYSE composite index, S&amp;P 500 index</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>BIC-K-NN better than BAH on all indices</td>
</tr>
<tr>
<td>Reference</td>
<td>Year</td>
<td>Method</td>
<td>Data</td>
<td>Performance</td>
</tr>
<tr>
<td>-----------</td>
<td>-------</td>
<td>--------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| [18]      | 2015  | Ensemble learning – Evolutionary Algorithm and Multilayer Perceptron | ROD, RSI, ROC, Volume, Stochastics, RCI, Golden/Dead cross, MACD and Candle chart | 16 companies of Tokyo exchange | Ensemble GNP-RL – 1.8% Profit Rate  
GNP-RL w/o ensemble - -4.4% Profit Rate  
Buy & Hold profit - -3.4% Profit Rate |
| [19]      | 2014  | LS-SVM                                           | Technical Indicators                                                | Small emerging markets of Southeast Europe | LS-SVM 43.23% higher than the best indicator strategy  
LS-SVM 24.13% higher than the Buy & Hold |
| [20]      | 2014  | Performance-based reward strategy with improved time variant particle swarm optimization | MA and TRB                                                          | 52 stocks of NASDAQ100 | The excess annual net profit generated by PRS on the original stock prices is 0.69% |
Best GA ROI – 62.95 %  
Buy & Hold ROI – 7.17 % |
| [14]      | 2015  | Multi-Objective Genetic Algorithm                | Technical Indicators                                                | NASDAQ, S&P 500, FTSE 100, DAX 30, and also NIKKEI 225 | The results show a return higher than 10% annual for the period of 2006–2014 in the NASDAQ and DAX indexes, in a period that includes the stock market crash of 2008 |
Buy & Hold ROI – 61.9% |
| [27]      | 2018  | Reinforcement Learning                           | Historical Data                                                      | FOREX | Average profit of 114.0±19.6% |
4. Proposed Architecture

The purpose of this chapter is to describe the solution developed to generate buy or sell signals. It starts by giving an overview of the general architecture developed, and the various modules that compose the solution are also described below.

4.1. Overall Architecture

In this chapter the architecture of the system developed to handle the optimization problem is described. It is specified which steps are addressed to construct such a capable system, as well, answer the fundamental questions about which data we can use and what will be the composition of such an application.

The developed application tries to find the best investment strategy for a particular stock (entry and exit points). In an initial phase, the data enters the application, and technical indicators are calculated from data provided. Then there is a training phase, in which the optimal voting powers of each indicator are discovered. These voting powers are then inserted into a voting system, and a weighted average is calculated to find out entry or exit points. In this last phase, two different strategies are built, with different levels of risk and profit. This way different types of investors can choose which strategy suits their taste.

The following diagrams propose the architecture of a possible system which tries to handle the optimization issue.

![General Architecture](image-url)

Figure 15 - General Architecture
As we can see in the previous figure, the system is constituted by several modules, with different functionalities and specializations. Each of the modules is responsible for a different phase throughout the process. It is also possible to identify four different modules whose responsibilities are explained below.

- **Technical Rules Module**: Starts by reading the data from a file, it computes all the technical indicators and transform each indicator into entry and exit signals applying the corresponding technical rules.
- **Optimization Module**: This is the brain of the system. It is responsible for finding the best set of weights/voting power for each indicator. It operates with a genetic algorithm optimizing the return on investment and the associated risk.
- **Voting-Based Decision Module**: This module is responsible for creating the strategy to be tested according to the voting power of each technical indicator used.
- **Investment Simulator Module**: This is the system that is responsible for simulating every strategy. It simulates the strategy given by the optimization module, computing the return on investment and the risk involved in each strategy.

### 4.2. Data Flow

In order to start the application, the user must define the inputs required to execute the optimization algorithm.

- Initially, the user must choose the stock to perform the optimization, also the initial budget, the training and execution window and the period, and also other settings.
- Afterward, the system computes the technical indicators, chosen by the user, and with these values, it computes buy and sell signal from each indicator according to the rules.
- After this process, the GA starts its execution by defining several random individuals, which correspond to different voting powers for each technical rule.
- In order to evaluate each individual, a Strategy Simulation module is needed. In this module, each individual is transformed in a strategy, and it is computed the Return on investment and Maximum Drawdown of each strategy/individual.
- When the GA converges in a final solution, the system executes the investment simulation again, but to the current date period.
The above diagram illustrates the different stages of the optimization process. It is worth stressing that the simulator and the GA process run alongside because it is the simulator that computes the fitness functions to evaluate each strategy. After the GA process is finished, the best strategy must be simulated again (but this time not in the training set) to yield the final results.

4.3. Development Environment

The application presented in this work was developed in Scala. Scala is a modern multi-paradigm programming language designed to express common programming patterns in a concise, elegant, and type-safe way. It smoothly integrates features of object-oriented and functional languages [28].


More details about the code developed are described in Appendix A.
4.4. Investment Simulator Module

In order to be able to evaluate each strategy associated with each individual, an investment simulator is necessary. The simulator applies the signals of buy and sell resulting from the voting system. Simulating the strategies also computes the return on investment and the Maximum Drawdown associated with the simulated strategy.

Several settings must be defined in the simulator depending on the preference of the investor.

- **Budget**: The initial capital available to invest.

- **Short Selling**: This parameter defines if the simulator can take short positions.
- **Transaction Costs:** Used for the consideration of transaction costs. This parameter is used to include the commission costs involved in buying or selling shares. This number defines the percentage to be charged.

- **Training window size:** Size of the period to find the optimum set of weights of the indicator rules

- **Application window size:** Size of the window in which apply the result of the previous optimization.

![Simulator UML](image)

*Figure 18 - Investment Simulator Module UML*

This module runs alongside the optimization module. The way this modules work is:

- First performs the optimization for the first training window, finding the best set of weights for this period. (First GA run)
- Then the weights that were found in the previous step are applied to the first application window.
- After this, both sliding windows are shifted in the same amount as the application window size.
- Then all the processes are repeated until the end of the simulation period
In the first iteration, the two months immediately before January first are used to find the best set of weights. Then those same weights are applied to the first month (First application window).

After all the sliding windows are shifted the second iteration may start, finding the new best weights to the new sliding training window, and then applied again to the new application window. And so on.

4.5. Technical Rules Module

This module is responsible for processing all the financial data. After the stock data is processed from the input files, the technical indicators must be computed and transformed in buy and sell signals for each indicator according to each rule.

4.5.1. Input Files

The first step of the process is to read and store the information about the stock prices. The input files must obey some rules, must be a “.csv” file, and with the following configuration:

<table>
<thead>
<tr>
<th>Date</th>
<th>Open</th>
<th>High</th>
<th>Low</th>
<th>Close</th>
<th>Adj. Close</th>
<th>Volume</th>
</tr>
</thead>
</table>

Where:

- Date is the start date of specific interval, using the format “yyyy-mm-dd”
- Open is the opening price of a specific interval (day, month, year...)
- High is the highest price of a specific interval
- Low is the lowest price of a specific interval
• Close is the closing price of a specific date.
• Adj. Close is the adjusted closing price
• Volume is the number of shares traded in a security during a specific interval.

4.5.2. Implementation and Functionality

As mentioned in the previous chapter, a technical indicator is a formula that is usually applied to the price and volume of the asset. The resulting values are analyzed allowing to identify a perspective in the evolution of the price of the asset. Briefly, a technical indicator tries to understand the behavior of the asset in order to evaluate if it is over or undervalued.

To use a technical indicator is necessary to define some parameters, such as the calculation period. For example, a simple indicator such as the moving average, which calculates the average of the last x days for each day. This indicator depends on the number of days x.

By setting rules for each indicator, is possible to generate buy and sell signals. Analyzing which indicators generated the best buy and sell signals (with a higher return and lower risk) in the past is possible to attribute in the present and future more importance to the signals generated by indicators with a better score.

For each technical indicator calculated for each period (day, week, or month) in the data set, a score was assigned. Three distinct scores were used:

- **Sell Score**: Assigns -1.0 points, indicates a sell/short signal,
- **Out Score**: Assigns 0 points, indicates an out signal,
- **Buy Score**: Assigns 1.0 points, indicates a buy signal.

The voting system of this system needs several technical indicators to function. In this way, several different types of technical indicators were implemented, among them: Exponential Moving Average (EMA), the Simple Moving Average (SMA), the Relative Strength Index (RSI), the Moving Average Convergence Divergence (MACD).

Note that the technical indicators are easily extendable, so it is possible to add new technical indicators to the application. The only requirement is to implement the technical indicator and define its rule to generate input or output signals.
The main class, the Indicator class, is abstract and divided into different classes of technical indicators. This structure makes it easy to implement new technical indicators for the application.

Each QuoteHistory Class contains several Quote objects (with Close, Open, High, Low, Adj Close prices and volume), one for each day/week. It also contains several Indicator Objects, according to the indicators to be used.

As stated before after all indicators are computed must be transformed into buy/sell signals.

Each Indicator object present in the QuoteHistory creates a corresponding new Strategy object and is added to the StrategyList also present in the QuoteHistory. Each Strategy object contains the entry/exit points generated from the corresponding technical indicator rule.

After all the processes in this module are finished there are:

- One QuoteHistory object constructed from the entry file, containing the price history of the corresponding stock.
- Several Indicator objects (list present on the QuoteHistory) constructed from the price history present in the QuoteHistory Object. This object contains the indicator history.
- One StrategyList present in the QuoteHistory
- Several Strategy objects (one from each indicator) constructed from the indicator objects. This class contains the entry and exit points from each technical indicator rule. These objects are aggregated in a list present in the StrategyList object (present in the QuoteHistory)

4.5.3. Moving Averages

The moving average computes the average value of the data over a given period. These are commonly used to mitigate the noises of short-term price fluctuations in order to facilitate the identification and definition of trends.

In this work two types of moving averages are used: simple and exponential.

4.5.3.1. Simple Moving Average

The simple moving average combines daily prices, uses the closing prices of a market as a rule. Is computed with the closing prices of a market for some periods chosen and divides that number by the number of periods. It consists of a moving average, with old data being discarded as soon as new data comes up.

Table 3 - MA Rule.

<table>
<thead>
<tr>
<th>MA</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>MA &gt; Stock Price</td>
<td>Sell Signal (-1)</td>
</tr>
<tr>
<td>MA &lt; Stock Price</td>
<td>Buy Signal (+1)</td>
</tr>
</tbody>
</table>

4.5.3.2. Exponential Moving Average

The Exponential Moving Average (EMA) is a trend following indicator. The goal of this device is to identify that a trend has begun, or it is finishing its cycle. In order to accomplish it, the EMA averages the price data, in order to produce a smooth line which can be easily perceived, in contrast to the irregular curve signaling the prices. The exponential moving average assigns more weight to the most recent data in order to give more importance to it. Its formula can be defined as follows:
\[ EMA_t(N) = EMA_{t-1}(N) \times \left(1 - \frac{2}{N+1}\right) + X_t \times \frac{2}{N+1}. \]

Where:

- \( N \) is the length of the EMA;
- \( X_t \) is the stock price of the corresponding period;
- \( t \) is the considered period (day, week, month).

With this indicator the following rules were applied:

<table>
<thead>
<tr>
<th>EMA</th>
<th>EMA Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMA &gt; Stock Price</td>
<td>Sell Signal (-1)</td>
</tr>
<tr>
<td>EMA &lt; Stock Price</td>
<td>Buy Signal (+1)</td>
</tr>
</tbody>
</table>

The following picture provides an example of the EMA line. As you can see, it defines a smoothing curve which can be easily analyzed, in contrast to the zigzag performed by the stock’s prices.

\[ \text{Figure 21 - EMA rule illustration} \]

4.5.4. Relative Strength Index

The RSI is a technical indicator that uses the closing prices to identify the growth potential of a stock.

The higher the value, the lower is the potential for asset growth will be. There are upper and lower limits that will indicate whether the asset is considered sub or overvalued. Thus, if its
value is below the lower limit, the asset will have a higher valuation potential, so it should be purchased.

The formula used on its calculation is:

$$ RSI_t(N) = 100 - \frac{100}{1 + RS(N)} $$

Where:

- $N$ is the length of the RSI;
- $t$ is the considered period (day, week, month).

For the RSI indicator the following rules were defined:

<table>
<thead>
<tr>
<th>RSI</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSI &gt; 70</td>
<td>Sell Signal (-1)</td>
</tr>
<tr>
<td>RSI &lt; 30</td>
<td>Buy Signal (+1)</td>
</tr>
</tbody>
</table>

4.5.5. **Moving Average Convergence Divergence**

The Moving Average Convergence Divergence (MACD) indicator was developed by Gerald Appel and consists of an extension of strategies that uses the smoothing of moving averages to reduce false business indicators.

This indicator is formed by three exponential moving averages, which appear in the graphs as two lines, the MACD and the signal line, whose crosses emit negotiation signals (buy/sell).

The MACD line is made up of two exponential moving averages and respond to price changes quickly. The signal line is composed of the MACD line adjusted by another exponential moving average and will respond to changes in prices more slowly.
Thus, trading signals are issued when the MACD line crosses the signal line: when the MACD line crosses below the signal line consists of a sell signal, and when it crosses upwards is an indication of the purchase of the stock.

\[
MACD_t(s, l) = EMA_t(s) - EMA_t(l)
\]

\[
Triggert(n) = EMA_t(n) \text{ of } MACD_t(s, l)
\]

\[
Hist_t = MACD_t(s, l) - Triggert(n)
\]

Figure 23 - MACD illustration

4.5.6. Indicator extensibility

As explained before new indicators are easily implemented. To add new indicators, a new class must be coded. This new class must extend the indicator class. To extend the indicator class, several methods capable of computing the indicator values must be implemented. One objected with the same name as the indicator must also be implemented. A detailed guide to add new indicators is now presented.

- 1\textsuperscript{st} - Create an objected with a method named calculate. This method must compute the indicator value for one day / period.

```java
object ClassIndicatorName{
    def calculate(index: Int, qh: QuoteHistory, IndicatorName: ClassIndicatorName, periodLength: Int, otherParametersNeeded): Double = {
        /*Code to compute the indicator value of one period*/
    }
}
```

- 2\textsuperscript{nd} - Create a Class extending the Indicator Class
class ClassIndicatorName(val qh : QuoteHistory, val periodLength : Int, OtherParametersNeeded) extends Indicator(qh) {

    /*variables needed*/
calculate_history()

    override def calculate_history(): Unit = {
        var bar = periodLength -1
        /*Other Variables Needed*/
        while(bar,qh.getSize){
            val indicatorname = ClassIndicatorName.calculate(bar, qh, this, periodLength,
            OtherParametersNeeded)
            addToHistory(indicatorname)
            bar += 1
        }
    }

    override def name: String = {
        this.getClass.getSimpleName + periodLength + OtherParametersNeeded
    }

    override def toStrategy: Strategy = {
        var bar = 0
        val strategy = new Strategy(this.name, qh)
        /*Other Variables Needed*/
        strategy.strategy = breeze.linalg.DenseVector.zeros(qh.getSize+1)
        /*Code to Transform indicator values into buy/sell Signals*/
        strategy
    }
}

4.6. Optimization Module

This module is the core of the application. It is this module that is responsible for finding
the best set of weights and thresholds possible. In this chapter, the various steps of the multi-
objective genetic algorithm [33] are described.

As explained before, the goal of the optimization is to find several optimized solutions
regarding the Return on Investment and the Maximum Drawdown. To do this, this module
performs a multi-objective optimization. The algorithm chosen to perform this task was the
NSGAII [34]. The set of several optimized solutions is called Pareto frontier. From this Pareto
frontier two solutions are chosen, one with high risk and high profit, and other with low risk and
low profit.

4.6.1. NSGAII

Multi-objective optimization is an area of multiple criteria decision making, that is
concerned with mathematical optimization problems involving more than one objective
function to be optimized simultaneously [34].
NSGA-II is an evolutionary algorithm. Evolutionary algorithms where developed because the classical direct and gradient-based techniques have the following problems when leading with non-linearities and complex interactions.

NSGA-II has the following three features:

- It uses an elitist principle; the elites of a population are given the opportunity to be carried to the next generation.
- It uses an explicit diversity preserving mechanism (Crowding distance)
- It emphasizes the non-dominated solutions.

4.6.2. Objective Functions

4.6.2.1. Return on Investment

The Return on Investment (ROI) is used to evaluate the efficiency of different investments during a specific period. The explanation for this concept is based on the relationship between the money earned or lost on a given investment and the amount invested. It consists in dividing the return by the total investment, thus acquiring the value corresponding to the profitability. The bigger the result of the better division.

The standard formula is straightforward and can be defined as follows:

\[
ROI = \frac{(Gain \text{ form investment} - Cost \text{ of investment})}{Cost \text{ of investment}}
\]

4.6.2.2. Maximum Drawdown

Maximum Drawdown is a downside risk indicator for a certain period. This indicator is used to assess the risk of one strategy against another, focusing on the preservation of capital.

This calculates the maximum percentage loss, from a maximum point of the series to the minimum point, before a new maximum is reached. Thus, the estimator will go through all periods of the sample and calculate the return for each period. The maximum drawdown is the lowest value of the return, so the smaller the magnitude of the value, the better.

It is important to note that the MDD will measure the size of the most significant loss, not taking into account the frequency of massive losses nor indicates the recovery time of the loss.

The maximum drawdown can be computed as follows:
4.6.3. Chromosome Representation

The design of the chromosome and its parameters is by necessity specific to the problem to be solved. In the solution presented in this work, the chromosomes are represented in binary as strings of 0s and 1s. The number of bits for each parameter is defined by the user. The search space depends on each parameter. Here the Chromosome represents different weights (importance) given to each technical rule. In addition to the weights described, several thresholds are also encoded in the genome. These thresholds transform the numerical result of the weighted average into buy or sell signals. These signals are then needed to take long or short positions and even to know how much to invest at a certain point in time. Next, the structure of the genome used in this application is presented and described.

<table>
<thead>
<tr>
<th>1st Rule</th>
<th>2nd Rule</th>
<th>...</th>
<th>Last Rule</th>
<th>Strong Buy Threshold</th>
<th>Buy Threshold</th>
<th>Sell Threshold</th>
<th>Strong Sell Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[0, 1]</td>
<td>[-1, 1]</td>
<td>[-1, 1]</td>
<td>[-1, 1]</td>
<td>[-1, 1]</td>
</tr>
</tbody>
</table>

In the representation above the search space for each parameter is presented as an interval, but, the representation is not continuous, so the values that each parameter can take depends on the number of bits used for each parameter. For instance, for the first rule, the search space is [0, 1], and let’s assume that a 2 bit representation is used, this means that the values that it can take are: {0, 1/3, 2/3, 1}, this is because a 2 bit string can only represent 4 different equally spaced values.

The ways this weights and threshold are then transformed in buy or sell signals are later described in the Voting module chapter.

4.6.4. Initialization

Unlike other steps, the initial population is generated only at the beginning of the algorithm, not repeating itself over the next generations. This population is generated randomly, allowing the full range of possible solutions. Its size depends on the source of the problem, but there are often thousands of solutions.
If there are repeated individuals, that is, if an individual already exists in the population, it will be ignored, and another will be created, so that the initial population is composed of the correct number of individuals many different.

### 4.6.5. Fast Non-Dominated Sort

The Fast Non-Dominated Sort selection algorithm analyses all individuals and compares them with each other, ranking them with their degree of rank. Thus, if a given individual is dominated by \( x \) individuals, his rank is \( x \). Thus, if at the end of the first stage the rank of one individual is 0, this individual is not dominated by anybody of the total population, being part of the first front where the best individuals of the population are. This way the individuals are sorted concerning the objectives, meaning that the non-dominated individuals appear at the beginning of the list.

![Figure 24 - NSGAII Non Dominated Sorting.](image)

### 4.6.6. Selection

In each generation, a sample of the population is chosen to create a new generation. The individual solutions created are chosen based on the fitness-based process, in which fitter
solutions are more likely to be selected. As it also a goal to have a diverse set of individuals within the population when selecting the selection process also depends on crowding distance.

The computation of the crowding distance is needed because this is a multi-objective optimization, and the result is not only one individual, but a full set of individuals. As said before we want to guarantee that there are different kinds of solutions in the final population. In this case, we want High Risk / High Profit and Low Risk / Low Profit solution.

The fitness function measures the quality of the solution. In this case, the fitness functions are the ROI and the Maximum Drawdown of each individual.

Finally, to select the individuals for the crossover and mutation operators, a binary tournament selection procedure is used. First, the procedure selects two solutions of the population, and then selects the better according to non-dominated sorting technique and crowding distance value. If both solutions are selected from the same front (same rank), the solution with the higher crowding distance is selected. This is called a Tournament Selection Operator.

![Tournament Selection](image)

**Figure 25 - Tournament Selection [36]**

### 4.6.7. Mutation

In order to maintain genetic diversity between generations, it is necessary to modify the chromosomes in some way. This process is done through the mutation that occurs during the process of evolution. This process involves a probability, which defines the probability of each bit being changed. The method of implementing the mutation operator involves generating a
random variable for each bit in a sequence. This random variable tells whether a particular bit will be modified or not.

![Mutation Operator – Flip Bit](image)

**Figure 26 - Mutation Operator – Flip Bit [37]**

### 4.6.8. Crossover

For this application, the crossover operator chosen was the uniform crossover. This method constructs a new chromosome (child) based on two chromosomes (parents). For each of the child bits, a random variable is generated that defines whether to inherit the bit from one of the parents or the other.

Each bit is chosen from either parent with equal probability. Other mixing ratios are sometimes used, resulting in offspring which inherit more genetic information from one parent than the other.

![Uniform Crossover](image)

**Figure 27 - Uniform Crossover [38]**

### 4.6.9. Evaluation Functions

For the genetic algorithm to be able to choose the best individuals for reproduction, it is necessary that each individual is evaluated, thus allowing the algorithm to converge to an optimal solution. To evaluate each individual two fitness functions were used. One to evaluate
the investment gain, and another to evaluate the risk of each strategy. For the first one was used the Return on Investment, and to evaluate the risk was used the Maximum Drawdown.

The ROI is used to evaluate the efficiency of different investments during a specific period. The Max Drawdown is the maximum loss from a peak to a trough of an investment, before a new peak is attained. Maximum Drawdown (MDD) is an indicator of downside risk over a specified period.

4.6.10. Optimization Result – Pareto Front

![Pareto Front Graph](image)

*Figure 28 - Pareto Front. Adapted from [39]*

The result of multi-objective optimization is not just a solution. The result is a set of solutions that minimize goal functions. This set of solutions is called Pareto Front. In Figure 26 the pareto front is represented in red and it is composed by non-dominated solutions. In this work are considered the two solutions located in each end of the pareto front.
To this optimization module work three classes are needed:

- The StrategyList, containing several strategies (one from each indicator). This defines the entry and exit points from each indicator.

- The NSGAII class responsible for defining the execution procedure of the genetic algorithm, handling the population of chromosomes.

- Also, the Simulator class, accountable for evaluating each individual present in the population (Fitness). More details about this class are presented in the next chapter.
4.7. Voting Based Decision Module

Using several technical indicators to build a strategy seems to be ideal. This is because the technical indicators complement each other. Sometimes during certain periods, a technical indicator produces wrong signals and therefore should be penalized in the following uses, and on the other hand, indicators with good performance should be rewarded, and therefore their signals should be important in decision making.

This module is responsible for implementing the voting system for the indicators. It runs alongside the optimization module because it handles a process needed to compute the fitness of each individual of the population. From the genome resulting from the optimization process, this module computes the trading rules for the next period. The way of calculating the buy/sell signals considers the weights and thresholds defined by the genome.

4.7.1. Implementation and Functionality

For every single period (day, week, month, or another period) a buy/sell must be computed. To do this, the indicators' buy/sell signal (-1, 0 or 1) is multiplied by the corresponding weight/voting power, summed all together and then normalized. This calculation can be defined as:

\[
\text{normalized weighted average} = \frac{\sum_{i=1}^{N} w_i \times s_i}{\sum_{i=1}^{N} w_i}.
\]

Where:

- \( w_i \) is the weight/voting power of the indicator \( i \);
- \( s_i \) is the signal for the indicator \( i \) (-1, 0 or 1);
- \( N \) is the number of indicators.

This formula always yields a resulting signal which is a number between -1 and 1. After this is computed, the resulting numerical signal must be converted into buy/sell signals. There are five different signals: Strong Sell, Sell, Out, Buy, Strong Buy. As explained before, the genome also includes four thresholds, and are these values that defined and transform the numerical signal in the five different signals.
In the figure above, it is possible to understand how each numerical signal is then transformed into a buy/sell signal. It should be noted that short selling is only used if the user allows it. This way if a user does not allow short selling all short positions are transformed into “Out 100%”.

![Thresholds Illustration](image1)

**Figure 30 - Thresholds Illustration**

The result of this process is a set of buy/sell signals (Strategy object), and the corresponding ROI and Maximum Drawdown.

![Voting Based Decision Module Process Visualization](image2)

**Figure 31 - Voting Based Decision Module Process Visualization**
In the figure above is the process of finding the resulting strategy (Buy / Sell Signals) from the genome and the buy/sell signals from each indicator.

This module receives the buy/sell signals from each indicator (of the Application window), the genome (optimized with data from the training window), and finds the resulting strategy for the application window. With the resulting strategy, it also computes the Return on Investment and Maximum Drawdown.

The first step is to apply the normalized weighted average formula on each day, according to the signals (from each indicator) for that same day, and the corresponding weight. This results in a normalized weighted average for each day. Then these values are compared with the thresholds and transformed into buy/sell signals.

The implementation of this module and the Investment Simulator Module was done in such way that all entries (buys) were done at the beginning of the period (Open price), and all exits (sell/shorts) were computed at the end of the period (Close price).

Let’s take the resulting strategy (buy/sell signals) of the previous example. The first signal is a “1” corresponding to a Strong Buy, this means that all available budget is invested in a long position on the beginning of that same period (Open price). The second signal is also a “1”, so no changes, the position is still long. On the third day we get a signal of “0.5” this means that only 50% of the available budget must be invested. Taking into account that until now all the budget is invested, to get to the 50% investment, half of the investment must be sold (this operation is computed with the closing price). On the fourth and fifth days a strong sell signal is present, this means that the simulator must sell the remaining 50% investment (long), and must take now a short position in the amount of all the budget available. In the last day, only half of the budget must be in a short position, so the simulator must close half of the investment made in the previous short operation, remaining 50% of the available budget in a short position.
4.8. Conclusion

In this chapter, various details of the various steps that make up the application were presented. Note that several aspects can be easily modified and added, especially the technical indicators. After the solution is implemented, it is necessary to test intensively. In the following chapter, the reader has access to the results of the system.
5. Validation

In this chapter, we describe the evaluation method to validate the developed system. To validate the application it was necessary to use a test strategy called Backtesting [40]. This process consists of testing a specific strategy in an earlier period to determine its effectiveness. For example, suppose we want to evaluate our strategy against the year's performance. Instead of waiting a full year to do so, we can extract the data from the past and then evaluate the procedure in the periods considered. Applying the strategy developed to the previous data can be substantially beneficial in order to detect strategy failures and improve their potential.

5.1. Performance Measures

To calculate the performance of a financial fund or stock, different measures can be employed, such as ROI, Sharpe Index, among others. To evaluate the system, two of these measures are used, ROI and Maximum Drawdown. In addition, a list of classification parameters is proposed for evaluation.

All the data for these tests was taken from yahoo finance [41].

5.2. Classification Parameters

In addition to the two measures presented to evaluate the performance of the generated strategies, a list of important parameters that can be used to classify the strategies was defined:

- Number of positions: The number of positions taken in the investment within a specific period.
- Percentage of profitable positions: From the total number of positions made in a specific period, it will be essential to observe how many of them are profitable.
- Percentage of non-profitable positions: As to measure the number of winning positions, it is also desirable to determine the non-profitable ones.
- Greatest profit: It will be interesting to observe what was the most significant profit obtained within all the investment period.
• Greatest loss: As the greatest profit, it is also essential to determine the most significant loss.
• Average Profit: Indicates the average profit considering all positions taken within a specific period.

5.3. Strategies Employed

To validate the developed solution, the optimized strategies were compared with the market and other investment methodologies:

Buy and Hold: According to some theories [2], already discussed in the first chapter, prices are independent of each other, which means that one should not use previous data to predict market movements, so in this case the best strategy we can use is to buy and maintain the asset regardless of market fluctuations.

5.4. Case Studies

5.4.1. Case Study I – Dow Jones Industrial Average ETF (DIA) 2003/2009

The case study was simulated during the years 2003 to 2009 and presents the results obtained when evaluating the implemented strategies. For this designed case study, the following configuration was applied:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stock</td>
<td>DIA</td>
</tr>
<tr>
<td>Period</td>
<td>02/01/2003 – 16/06/2009</td>
</tr>
<tr>
<td>Budget</td>
<td>100000 USD</td>
</tr>
<tr>
<td>Short Selling</td>
<td>TRUE</td>
</tr>
<tr>
<td>Commission Costs</td>
<td>0.1% [42]</td>
</tr>
<tr>
<td>Number of executions</td>
<td>100</td>
</tr>
</tbody>
</table>

In respect to the evolutionary strategy, the following parameter specification was employed:
Table 7 - NSGAII Parameters for Case Study I

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population Size</td>
<td>50</td>
</tr>
<tr>
<td>Mutation Rate</td>
<td>10%</td>
</tr>
<tr>
<td>Crossover Rate</td>
<td>90%</td>
</tr>
<tr>
<td>Nº Generations</td>
<td>100</td>
</tr>
<tr>
<td>Sliding Window</td>
<td>125 days / 125 days</td>
</tr>
</tbody>
</table>

Sliding window refers to the combination of the training / validation period. Note that if validation begins in January 2003, the previous training period will be used to train the algorithm. In this case, after six months of validation, the algorithm passes through the same training process.

The graph below shows the ROI evolution for the strategies considered in the test period. Each curve represents the return on investment obtained by the respective investment method.

![ROI Evolution graph](image)

*Figure 33 - ROI evolution for Case Study I*

To evaluate the superiority of the optimized strategy the following histogram demonstrates the ROI obtained for the different 100 executions experienced by each investment method.
The following table summarizes the performance of each strategy according to the parameters described in the first section of this chapter. 100 different runs have been tried to evaluate each methodology completely. The results for these strategies correspond to the confidence interval reached when using a confidence level of 95%.

<table>
<thead>
<tr>
<th></th>
<th>GA (HIGH RISK)</th>
<th>GA (LOW RISK)</th>
<th>BUY &amp; HOLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROI (%)</td>
<td>[61%, 68%]</td>
<td>[7.3%, 13%]</td>
<td>1.2%</td>
</tr>
<tr>
<td>MAX DRAWDOWN</td>
<td>[-16%, -15%]</td>
<td>[-29%, -26%]</td>
<td>-54%</td>
</tr>
<tr>
<td>Nº POSITIONS</td>
<td>[70, 75]</td>
<td>[120, 130]</td>
<td>1</td>
</tr>
<tr>
<td>PROFITABLE POSITIONS</td>
<td>[38%, 40%]</td>
<td>[38%, 39%]</td>
<td>100%</td>
</tr>
<tr>
<td>NON-PROFITABLE POSITIONS</td>
<td>[60%, 62%]</td>
<td>[61%, 62%]</td>
<td>0%</td>
</tr>
<tr>
<td>AVG PROFIT PER POSITION</td>
<td>[0.78%, 0.87%]</td>
<td>[0.10%, 0.16%]</td>
<td>1.2%</td>
</tr>
<tr>
<td>MAX PROFIT</td>
<td>[32%, 34%]</td>
<td>[18%, 20%]</td>
<td>1.2%</td>
</tr>
<tr>
<td>MIN PROFIT</td>
<td>[-5.1%, -4.4%]</td>
<td>[-6.4%, -5.4%]</td>
<td>1.2%</td>
</tr>
</tbody>
</table>

In the next graph is possible to visualize the position taken on each moment of the investment. On the curve below, it is possible to see the position taken on the corresponding time.
In the figure above, it is possible to understand where the voting system is robust and can beat the Buy & Hold strategy. In the first five years a sideways market is dominant, and in this situation, the voting system cannot get better results than the B&H strategy, but in the last three years of the simulation, when the big crash start, the voting system get much better ROI than the B&H.
Above, in figure 36, the evolution of the different strategies corresponding to each indicator used is shown. It is possible to note that these strategies almost follow the buy & hold.

In the figure below is possible to see the different stages of the sliding window, and on the table below the chromosomes for all periods is shown.
### Table 9 - Genome for each period

<table>
<thead>
<tr>
<th></th>
<th>I1</th>
<th>I2</th>
<th>I3</th>
<th>I4</th>
<th>I5</th>
<th>I6</th>
<th>T1</th>
<th>T2</th>
<th>T3</th>
<th>T4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0,00</td>
<td>0,67</td>
<td>0,33</td>
<td>1,00</td>
<td>0,67</td>
<td>0,33</td>
<td>-0,33</td>
<td>-0,33</td>
<td>0,33</td>
<td>1,00</td>
</tr>
<tr>
<td>2</td>
<td>0,33</td>
<td>1,00</td>
<td>1,00</td>
<td>0,00</td>
<td>1,00</td>
<td>1,00</td>
<td>-1,00</td>
<td>-1,00</td>
<td>-0,33</td>
<td>0,33</td>
</tr>
<tr>
<td>3</td>
<td>0,67</td>
<td>1,00</td>
<td>0,33</td>
<td>0,33</td>
<td>0,67</td>
<td>1,00</td>
<td>-1,00</td>
<td>-1,00</td>
<td>-0,33</td>
<td>-0,33</td>
</tr>
<tr>
<td>4</td>
<td>0,00</td>
<td>1,00</td>
<td>1,00</td>
<td>0,00</td>
<td>1,00</td>
<td>0,67</td>
<td>-0,33</td>
<td>-0,33</td>
<td>0,33</td>
<td>1,00</td>
</tr>
<tr>
<td>5</td>
<td>0,00</td>
<td>0,67</td>
<td>1,00</td>
<td>0,67</td>
<td>0,33</td>
<td>0,67</td>
<td>-0,33</td>
<td>0,33</td>
<td>0,33</td>
<td>1,00</td>
</tr>
<tr>
<td>6</td>
<td>1,00</td>
<td>0,00</td>
<td>0,33</td>
<td>0,00</td>
<td>0,67</td>
<td>0,33</td>
<td>-1,00</td>
<td>1,00</td>
<td>1,00</td>
<td>1,00</td>
</tr>
<tr>
<td>7</td>
<td>0,33</td>
<td>0,33</td>
<td>0,33</td>
<td>0,67</td>
<td>0,00</td>
<td>0,00</td>
<td>-1,00</td>
<td>0,33</td>
<td>0,33</td>
<td>1,00</td>
</tr>
<tr>
<td>8</td>
<td>0,00</td>
<td>1,00</td>
<td>0,00</td>
<td>0,67</td>
<td>0,67</td>
<td>0,00</td>
<td>-1,00</td>
<td>-0,33</td>
<td>0,33</td>
<td>1,00</td>
</tr>
<tr>
<td>9</td>
<td>0,00</td>
<td>0,00</td>
<td>1,00</td>
<td>0,00</td>
<td>0,00</td>
<td>0,33</td>
<td>-1,00</td>
<td>-0,33</td>
<td>0,33</td>
<td>1,00</td>
</tr>
<tr>
<td>10</td>
<td>0,33</td>
<td>1,00</td>
<td>0,67</td>
<td>0,33</td>
<td>0,00</td>
<td>0,00</td>
<td>-0,33</td>
<td>-0,33</td>
<td>-0,33</td>
<td>1,00</td>
</tr>
<tr>
<td>11</td>
<td>0,00</td>
<td>1,00</td>
<td>0,33</td>
<td>0,33</td>
<td>0,00</td>
<td>0,00</td>
<td>-1,00</td>
<td>0,33</td>
<td>1,00</td>
<td>1,00</td>
</tr>
<tr>
<td>12</td>
<td>0,67</td>
<td>0,67</td>
<td>1,00</td>
<td>1,00</td>
<td>0,67</td>
<td>0,33</td>
<td>-0,33</td>
<td>1,00</td>
<td>1,00</td>
<td>1,00</td>
</tr>
<tr>
<td>13</td>
<td>0,00</td>
<td>0,67</td>
<td>0,67</td>
<td>0,00</td>
<td>0,00</td>
<td>0,33</td>
<td>-0,33</td>
<td>0,33</td>
<td>0,33</td>
<td>1,00</td>
</tr>
</tbody>
</table>

In the table above:

- **I1** – is MA20
- **I2** – is MA40
- **I3** – is MACD25,12
- **I4** – is MACD50,20
- **I5** – is RSI14
- **I6** – is RSI28
- **T1** – is the Full Short Threshold
- **T2** – is the Half Short Threshold
- **T3** – is the Half Buy Threshold
- **T4** – is the Full Buy Threshold

Looking at Table 9, the genome 12 is a solution that only allows for short positions (Half and Full). This is because T2 is 1.0 (Maximum value), knowing that the weighted average can only take values between -1 and 1. It is worth to note that the optimization for this individual was done in the 11th period, where is presented a situation that also takes the advantage from only is allowed short positions.
Here we can see the optimization with the same setting as before, but in this simulation short selling is not allowed.

**Figure 39 - ROI Evolution for a strategy without short positions for Case Study 1**

In the figure above is a detail of figure 38. In this situation is possible to understand the reason for such rise in the ROI line. This is one example of one excellent move from the voting strategy that allows a very high return Long between two Shorts (In the case that is possible), or between two Outs (In the situation without short selling).

For the first part of the simulation where short positions are allowed, it is possible to see that in the beginning when it is present a sideways market, the optimization algorithm does not
perform better than the buy & hold strategy. However, when the market crashes the evolutionary strategy is capable of maintaining a reasonable profit without collapsing as the competitor strategies; being less risky and volatile, and subsequently obtaining a much higher ROI on the end of the testing period.

Although the evolutionary strategy tries to perform intelligent investment decisions, it has much more transaction costs when compared with the B&H approach, decreasing its profitability when the market is rising. In contrast, when the crash occurs, the intelligent investment decisions are more notorious which allow the strategy to pick the ideal positions while maintaining its profit.

For the second part of the experiment, where short selling is not allowed it is evident that the ROI is not as high as in the first experiment, this is also obvious because we can see that the advantage of the evolutionary algorithm comes when the market crashes. In this simulation it is possible to see that simply instead of taking a short position, it just gets out of the market, avoiding the significant losses that the Buy & Hold cannot avoid.

The complete process of this simulation, from the input file processing to the exportation of the result files, finished in 12 seconds. The experiments was done in a laptop with an Intel Core i7-3615QM and 8,00GB of RAM.

5.4.2. Case Study II – Bausch Health Companies Inc. (BHC) 2013/2015 (Valeant)

The case study was simulated during the years 2013 to 2015 and presents the results obtained when evaluating the implemented strategies. For this designed case study, the following configuration was applied:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stock</td>
<td>BHC</td>
</tr>
<tr>
<td>Period</td>
<td>02/01/2013 – 31/12/2015</td>
</tr>
<tr>
<td>Budget</td>
<td>100000 USD</td>
</tr>
<tr>
<td>Short Selling</td>
<td>TRUE</td>
</tr>
<tr>
<td>Commission Costs</td>
<td>0.1%</td>
</tr>
<tr>
<td>Number of executions</td>
<td>100</td>
</tr>
</tbody>
</table>
In respect to the evolutionary strategy, the following parameter specification was employed:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population Size</td>
<td>50</td>
</tr>
<tr>
<td>Mutation Rate</td>
<td>10%</td>
</tr>
<tr>
<td>Crossover Rate</td>
<td>90%</td>
</tr>
<tr>
<td>nº Generations</td>
<td>100</td>
</tr>
<tr>
<td>Sliding Window</td>
<td>63 days / 63 days</td>
</tr>
</tbody>
</table>

The graph below shows the ROI evolution for the strategies considered in the test period. Each curve represents the return on investment obtained by the respective investment method.

![ROI Evolution](image)

The following table summarizes the performance of each strategy according to the parameters described in the first section of this chapter. 100 different runs have been tried to evaluate each methodology completely. The results for these strategies correspond to the confidence interval reached when using a confidence level of 95%.
Table 12 - Results for Case Study II. Intervals with confidence of 95%

<table>
<thead>
<tr>
<th></th>
<th>GA (HIGH RISK)</th>
<th>GA (LOW RISK)</th>
<th>BUY &amp; HOLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROI (%)</td>
<td>[236%, 262%]</td>
<td>[83%, 97%]</td>
<td>66%</td>
</tr>
<tr>
<td>MAX DRAWDOWN</td>
<td>[-41%, -40%]</td>
<td>[-39%, -37%]</td>
<td>-73%</td>
</tr>
<tr>
<td>N° POSITIONS</td>
<td>[36, 38]</td>
<td>[79, 82]</td>
<td>1</td>
</tr>
<tr>
<td>PROFITABLE POSITIONS</td>
<td>[45%, 47%]</td>
<td>[40%, 41%]</td>
<td>100%</td>
</tr>
<tr>
<td>NON-PROFITABLE POSITIONS</td>
<td>[51%, 53%]</td>
<td>[58%, 59%]</td>
<td>0%</td>
</tr>
<tr>
<td>AVG PROFIT PER POSITION</td>
<td>[4.4%, 4.8%]</td>
<td>[0.9%, 1.1%]</td>
<td>66%</td>
</tr>
<tr>
<td>MAX PROFIT</td>
<td>[64%, 67%]</td>
<td>[38%, 40%]</td>
<td>66%</td>
</tr>
<tr>
<td>MIN PROFIT</td>
<td>[-21%, -19%]</td>
<td>[-14%, -13%]</td>
<td>66%</td>
</tr>
</tbody>
</table>

In the next graph is possible to visualize the position taken on each moment of the investment. On the curve below, it is possible to see the position taken on the corresponding time.

Figure 42 - Positions take on each period for Case Study II
Here we can see the optimization with the same setting as before, but in this simulation short selling is not allowed.

Here in this experiment the evolutionary strategy is put against a profitable buy & Hold strategy. Although there is a significant drop near the middle of the simulation, the buy & hold strategy ends the investment with a 66% ROI. The B&H strategy cannot profit while the market is on a downtrend, this is where the evolutionary strategy gains against the B&H.

For the first part of the simulation where short positions are allowed, it is possible to see that in the beginning when it is present a bullish market, the optimization algorithm does not perform better than the buy & hold strategy. However, when the downtrend begins the evolutionary strategy is capable of maintaining a reasonable profit without collapsing as the competitor strategies; being less risky and volatile, and subsequently obtaining a much higher ROI on the end of the testing period.

Although the evolutionary strategy tries to perform intelligent investment decisions, it has much more transaction costs when compared with the B&H approach, decreasing its profitability when the market is rising. In contrast, when the downtrend begins occurs, the intelligent investment decisions are more notorious which allow the strategy to pick the ideal positions while maintaining its profit.

For the second part of the experiment, where short position are not allowed it is evident that the ROI is not as high as in the first experiment, this is also obvious because we can see that the advantage of the evolutionary algorithm comes when the downtrend starts. In this

![Figure 43 - ROI Evolution for a strategy without short positions for Case Study II](image-url)
simulation it is possible to see that simply instead of taking a short position, it just gets out of the market, avoiding the significant losses that the buy&hold cannot avoid.

5.4.3. Case Study III – EUR/USD (EURUSD=X) 2015/2017

The case study was simulated during the years 2015 to 2017 and presents the results obtained when evaluating the implemented strategies. For this designed case study, the following configuration was applied:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stock</td>
<td>EUR/USD</td>
</tr>
<tr>
<td>Period</td>
<td>02/02/2015 – 15/12/2017</td>
</tr>
<tr>
<td>Budget</td>
<td>100000 USD</td>
</tr>
<tr>
<td>Short Selling</td>
<td>TRUE</td>
</tr>
<tr>
<td>Commission Costs</td>
<td>0.1%</td>
</tr>
<tr>
<td>Number of executions</td>
<td>100</td>
</tr>
</tbody>
</table>

In respect to the evolutionary strategy, the following parameter specification was employed:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population Size</td>
<td>50</td>
</tr>
<tr>
<td>Mutation Rate</td>
<td>10%</td>
</tr>
<tr>
<td>Crossover Rate</td>
<td>90%</td>
</tr>
<tr>
<td>Nº Generations</td>
<td>100</td>
</tr>
<tr>
<td>Sliding Window</td>
<td>125 days / 125 days</td>
</tr>
</tbody>
</table>
The graph below shows the ROI evolution for the strategies considered in the test period. Each curve represents the return on investment obtained by the respective investment method.

**ROI Evolution**

![Figure 44 - ROI Evolution for Case Study III](image)

The following table summarizes the performance of each strategy according to the parameters described in the first section of this chapter. 100 different runs have been tried to evaluate each methodology completely. The results for these strategies correspond to the confidence interval reached when using a confidence level of 95%.

**Table 15 - Results for Case Study III. Intervals with confidence of 95%**

<table>
<thead>
<tr>
<th></th>
<th>GA (High Risk)</th>
<th>GA (Low Risk)</th>
<th>Buy &amp; Hold</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ROI (%)</strong></td>
<td>[-18%, -17%]</td>
<td>[-19%, -18%]</td>
<td>4.1%</td>
</tr>
<tr>
<td><strong>Max Drawdown</strong></td>
<td>[-27%, -26%]</td>
<td>[-25%, -24%]</td>
<td>-11%</td>
</tr>
<tr>
<td><strong>No Positions</strong></td>
<td>[27, 29]</td>
<td>[46, 48]</td>
<td>1</td>
</tr>
<tr>
<td><strong>Profitable Positions</strong></td>
<td>[19%, 21%]</td>
<td>[17%, 18%]</td>
<td>100%</td>
</tr>
<tr>
<td><strong>Non-profitable Positions</strong></td>
<td>[78%, 79%]</td>
<td>[81%, 82%]</td>
<td>0%</td>
</tr>
<tr>
<td><strong>Avg Profit per Position</strong></td>
<td>[-0.69%, -0.65%]</td>
<td>[-0.43%, -0.41%]</td>
<td>4.1%</td>
</tr>
<tr>
<td><strong>Max Profit</strong></td>
<td>[5.0%, 5.3%]</td>
<td>[6.0%, 6.4%]</td>
<td>4.1%</td>
</tr>
<tr>
<td><strong>Min Profit</strong></td>
<td>[-6.0%, -5.8%]</td>
<td>[-2.9%, -2.8%]</td>
<td>4.1%</td>
</tr>
</tbody>
</table>
In the next graph is possible to visualize the position taken on each moment of the investment. On the curve below it’s possible to see the position taken on the corresponding time.

![Positions on price chart](image1)

*Figure 45 - Positions taken on each period for Case Study III*

Here we can see the optimization with the same setting as before, but in this simulation short selling is not allowed.

![Position on price chart](image2)

*Figure 46 - ROI Evolution for a strategy without short positions for Case Study III*
As it is possible to understand this is a much harder situation for the evolutionary strategy. This is a situation where there is not much price variation throughout the time. Besides the difficulty of the situation it is possible to see that in both cases (shorts allowed and not allowed) the algorithm can sometimes get close to the buy & hold strategy.

As the table above shows, the number of positions/transactions is very high relative to the buy and hold strategy. Even with low transactions costs, this is maybe the most significant setback for this strategy in this situation.

Comparing the situation where Shorts are allowed with the situation where shorts are not allowed, is possible to see that that is not much difference. As covered before, the primary advantage of the evolutionary strategy is in the shorts position. In this situation, there is not a perfect position to take a short position, and this leads to a non-existing difference between the short and non-short situation.
6. Conclusion and Future Work

The work presented proposes a potential strategy optimization system that combines several indicators of technical analysis with evolutionary computation algorithms to choose the best entry and exit points for various actions in the market. To evaluate the developed system and its strategies, it was compared with the market itself and several other investment methods. The results are promising, and much more can be done to improve them. The algorithm can be easily extended and parameterized. This chapter proposes several improvements to refine the current system.

6.1. Conclusion

The validation demonstrated in the previous chapter shows that the combination of EC and technical analysis indicators has potential. In this work, several investment methodologies were described, as well as different computational techniques to address the presented problem. While there is a prospect of using this application to manage an investment, human capabilities cannot be replaced entirely. So this application should be used while keeping a close eye on the financial market news to understand if there is any problem that the system is not aware of.

6.2. Future Work

Under this section, several improvements of the algorithm are addressed.

6.2.1. Possible Improvements

Due to the high level of code extensibility, the following features are proposed to improve the current algorithm.

- Extend the chromosome with more technical indicators. This extension can be easily executed, it is only necessary to implement the desired indicator and define the respective rules.
• The system developed and presented in this paper does not allow the user to use leverage to increase earnings. It is essential to overcome this problem to enable the strategies developed to make a profit in every possible way.

• In addition to defining the ideal balance between different technical indicators, the chromosome can also be extended with the time interval assigned to each indicator, thus optimizing the indicator itself. However, this process will increase the execution time of the algorithm.

• To minimize the risk involved in the strategies optimized by the application developed, a possible solution would be to consider the dates scheduled to announce the results of the companies considered. This is very important, since, after the publication of the profits, the stock company may undergo a sudden change.

• Improve the existing GA by trying additional mutation, crossover and selection procedures.
Bibliography


Appendix A – Development Description

The application is divided into 14 different files:

- **DateUtil.scala** – File with an object with methods to handle date and time.
- **Util.scala** – File with an object with methods to handle number and strings.
- **Main.scala** – File with an object with the main method. In this file is the implementation of the user interface. Here the user can change all the settings.
- **Quote.scala** – File with an object and one class. This class stores the information of the prices for each period. This file is also responsible to parse each line of the input file.
- **QuoteHistory.scala** – File with an object and one class. This contains one array of the quote objects, and the methods to handle it.
- **Indicator.scala** – File with an abstract class. This class is the skeleton for all the indicators classes.
- **EMA.scala** – File with the implementation of EMA indicator.
- **MA.scala** - File with the implementation of MA indicator.
- **MACD.scala** - File with the implementation of MACD indicator.
- **RSI.scala** - File with the implementation of RSI indicator.
- **Strategy.scala** - File with the strategy class. Responsible for storing the buy and sell signals, computing the return rate for each period, and the Maximum Drawdown.
- **StrategyList.scala** – File with one class, responsible for storing the list of strategies of each chosen indicator.
- **Simulator.scala** – This file is responsible to simulate all strategies.
- **NSGA.scala** – Implementation of the algorithm NSGA II.
Appendix B – User Documentation

When the user first starts the application, the menu is displayed.

![Figure 47 - Application Start](image)

The default settings and some instructions are printed. To change the settings the user must write the setting to be changed, and then type the new value for the setting being changed.

To start the simulation the user must write “Run”. After the simulation is completed the application exports several “.csv” files with all the information of the simulation.