TÉCNICO LISBOA

# A Divergence-Based Trading Model using Genetic Algorithms and Market Classification

**André Abreu dos Santos**

Thesis to obtain the Master of Science Degree in

**Engenharia Eletrotécnica e de Computadores**

Supervisor(s): Prof. Rui Fuentecilla Maia Ferreira Neves
Prof. Nuno Cavaco Gomes Horta

## Examination Committee

Chairperson: Prof. António Manuel Raminhos Cordeiro Grilo
Supervisor: Prof. Rui Fuentecilla Maia Ferreira Neves
Member of the Committee: Prof. Rui António Dos Santos Cruz

**June 2024**

## Declaration

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

## Declaração

Declaro que o presente documento é um trabalho original da minha autoria e que cumpre todos os requisitos do Código de Conduta e Boas Práticas da Universidade de Lisboa.

# Acknowledgments

First, I want to thank my supervisor Rui Neves for his guidance and patience while working with me through this thesis.

I also want to thank my friends Mariana, Guilherme, Augusto, André, Bruno, Catarina and Fábio for their incredible support and friendship.

I am deeply grateful to my parents for their unwavering support and encouragement throughout my entire life.

Finally, I want to express my heartfelt thanks to Francisca, for her love, patience and support throughout this journey.

... to my uncle Adelino.

# Resumo

Este trabalho propõe um modelo capaz de transacionar no Mercado de Ações baseado na divergência entre o desempenho de ações individuais e o desempenho do índice Standard & Poor's 500 (S&P500). O modelo começa por selecionar um subconjunto de ações do S&P500 com base nos valores do sistema de pontuação fundamental $F_{Score}$. Indicadores técnicos são então calculados para estas ações e para o índice S&P500, para avaliar as tendências e movimento dos seus valores. A cada indicador é atribuído uma pontuação específica com base em regras predefinidas. As pontuações de cada ação são então comparadas com as pontuações do índice S&P500, gerando uma lista de "Pontuações de Divergência". Cada Pontuação de Divergência é então multiplicada pelo seu respetivo peso e somada para produzir uma "Pontuação Total de Divergência", que é usada para medir a divergência entre o desempenho de uma ação e o desempenho do índice S&P500. Os pesos de cada pontuação são otimizados usando Algoritmos Genéticos e são ajustados de acordo com a tendência do índice S&P500 do respetivo dia. Um classificador XGBoost é utilizado para a classificação do índice S&P500 em três categorias: *Downtrend*, *Sideways* e *Uptrend*. Os pesos são então selecionados com base na previsão da tendência do mercado pelo XGBoost. Quando as Pontuações Totais de Divergência excedem os limites predefinidos, o algoritmo inicia uma posição, que é fechada assim que a pontuação ultrapassa os limites de fecho. Este modelo foi testado no período de 3 anos de 2020 a 2022 e alcançou um retorno de investimento de 10,77%.

**Palavras-chave:** Algoritmos Genéticos, FScore, Análise Fundamental, Análise Técnica, XGBoost

# Abstract

This work proposes a trading model based on the divergence between the performance of individual stocks and the performance of the Standard & Poor's 500 (S&P500) index. The model starts by selecting a subset of stocks from the S&P500 based on the values of the fundamental scoring system $F_{Score}$. Technical indicators are then calculated for these stocks and the S&P500 index to assess the current trends and momentum. Each indicator is assigned a specific score based on predefined rules. The scores for each stock are then compared to the scores of the S&P500 index, generating a list of "Divergence Scores". Each Divergence Score is then multiplied by its respective weight and summed to produce a "Total Divergence Score", which is used to measure the divergence between the performance of a stock and the performance of the S&P500 index. The weights of each score are optimized using Genetic Algorithms and are adjusted according to the trend of the S&P500 index of that day. A XGBoost Classifier is used for the categorization of the S&P500 index into three categories: Downtrend, Sideways and Uptrend. The weights are then selected based on the XGBoost market trend prediction. When the Total Divergence Scores exceed predefined thresholds, the algorithm opens a position, which is closed once the score surpasses the closing thresholds. This model was tested in the 3 year period from 2020 to 2022 and it achieved a return on investment of 10.77%.



**Keywords:** Genetic Algorithms, FScore, Fundamental Analysis, Technical Analysis, XGBoost

# Contents

# List of Tables

# List of Figures

# List of Acronyms

AUC-ROC Area Under the Receiver Operating Characteristics Curve

B&H Buy & Hold

CBOE Chicago Board options Exchange

COGS Cost of Goods Sold

DC Double Crossover

DWT Discrete Wavelet Transform

EA Evolutionary Algorithm

EMA Exponential Moving Average

GA Genetic Algorithm

GBDT Gradient Boosted Decision Trees

HMA Hull Moving Average

LR2GBDT Logistic Regression To Gradient Boosted Decision Trees

MACD Moving Average Convergence Divergence

MDD Maximum Draw Down

MOEA Multi-Objective Evolutionary Algorithm

MOO-GA Multi-Objective Optimization Genetic Algorithm

NASDAQ National Association of Securities Dealers Automated Quotations

OBV On Balance Volume

OCF Operating Cash Flow

P/B Price-to-book

P/E Price-to-earnings

PCA Principal Component Analysis

ROA Return on Assets

ROC Rate of Change

ROI Return on Investment

ROR Rate of Return

RSI Relative Strength Index

S&P500 Standard & Poor's

SAX-GA Symbolic Aggregate approXimation Genetic Algorithm

SMA Simple Moving Average

SVM Support Vector Machine

# Chapter 1

# Introduction

The financial markets are an interesting topic for computational intelligence researchers because of their extreme complexity and dynamic nature, with numerous variables that can impact their performance. This complexity makes them ideal for developing and exploring new computational techniques that can help to better understand and predict the behavior of the market. Additionally, even small improvements in predictive accuracy can have significant impacts. Finally, the constantly evolving nature of financial markets, with new data and information becoming available all the time, makes them an interesting and challenging area for research.

A common tool to measure the performance of a market is a market index. A market index provides a comprehensive snapshot of the performance of a group of large publicly traded companies. One of the most popular market indexes is Standard & Poor's 500 (S&P500), this index tracks the performance of 500 of the largest companies in the USA and is considered as the benchmark by many investors and analysts.

The Efficient Market Hypothesis [1] describes that investors and traders cannot consistently "beat the market" by making profitable trades based on publicly available information, because any new information is quickly incorporated into the asset prices. Even though this hypothesis proposes valid premises, there are many works with strategies capable of outperforming the market.

The strategies developed to outperform the market are usually based on two types of analysis or a combination of the two. These types of analysis are Fundamental Analysis and Technical Analysis [2]. Fundamental Analysis involves evaluating the financial health and business model of a company to determine its intrinsic value. On the other hand, Technical Analysis involves studying past market data such as price and volume to identify patterns and predict future market behavior.

Some of the works that combine these two types of analysis have managed to achieve great returns, even in some cases outperforming the S&P500 index, but they usually focus only on the behaviour of the selected assets and do not take into account the overall behaviour of the market.

For this work, it is proposed a combination of these two types of analysis, where it is used Fundamental Analysis to select stocks used for trading, and Technical Analysis to optimize the timing of the positions based on the divergence between each individual stock performance and the S&P500 index

performance. A XGBoost Classifier is also used to optimize the trading algorithm based on the current market trend prediction.

## 1.1 Motivation

The financial markets are complex and constantly evolving, presenting significant challenges for traders and investors seeking to achieve consistent returns. Traditional trading strategies tend to rely on either fundamental analysis or technical analysis but few tend to effectively integrate both to capture market opportunities. Additionally, many existing trading algorithms do not take into account the current market conditions, leading to missing opportunities to increase the returns.

This thesis addresses this gap by proposing a trading model consisting on the use of Fundamental Analysis to select the stocks used for trading, and Technical Analysis to perfect the timing of the positions based on the divergence between each individual stock performance and the S&P500 index performance.

To optimize the returns of this trading algorithm, the calculation of the divergence will differ based on current market trend of the S&P500 provided by a XGBoost Classifier.

## 1.2 Objectives

The objectives of this thesis are to:

- Creation of a divergence-based trading model.
- Study the impact of market trend optimization on the trading algorithm.
- Study the impact of the margins of the opening and closing thresholds on the trading algorithm.
- Understand the impact of the moving average selected for the sliding window scheme.
- Study the influence of the selected stocks for the returns of the trading algorithm.

## 1.3 Contributions

The contributions presented in this thesis are:

- Implementation of a XGBoost Classifier for trend classification of the S&P500 index into three categories: Downtrend, Sideways and Uptrend.
- Usage of the average $F_{Score}$ of the previous three years of the trading session for the selection of the stocks used for trading.
- Creation of a divergence metric called “Total Divergence Score” to measure the divergence of the performance of a stock and the performance of the S&P500 index, using Genetic Algorithms to optimize the weights for the calculation of the “Total Divergence Score”.

## 1.4 Thesis Outline

The thesis presented is structured as following:

- **Chapter 2**: Background - Explains the fundamental concepts and methodologies necessary to understand financial markets, Genetic Algorithms, XGBoost and the previous works developed.
- **Chapter 3**: Architecture - Shows the architecture of the system presented in this work, as well as the flow of the system and a detailed explanation of each module.
- **Chapter 4**: Results - Presents and analyses the Case Studies developed in this work. It starts by explaining the metrics used to evaluate the Case Studies, followed by a detailed explanation and analysis of the results for each Case Study.
- **Chapter 5**: Conclusion - Presents the conclusions of this work, as well as suggestions for future work.

# Chapter 2

# Background

In this chapter it is explained the fundamental concepts necessary to understand the work addressed relative to the stock market and analyze a substantial part of the algorithms applied to the trading system proposed on Chapter 1. This chapter is composed of three sections: Section 2.1 will explain in detail the concepts of trend markets, S&P500 index, Fundamental Analysis and Technical Analysis, along with the description of some commonly used fundamental and technical indicators. Section 2.2 will explain the computational algorithms used for this work. Section 2.3 will mention all works that have influenced the implementation of the algorithm proposed in this work.

## 2.1 Market Analysis

The stock market is essentially a platform where people can buy or sell pieces of ownership called ”shares” in companies. Investors can buy those shares when a company becomes public through the stock market, which grants the investor a portion of the ownership of that company. This ownership grants the investors a portion of the company's profits and voting power on the management of the company.

There are also stock indexes which represent an agglomeration of several stocks, providing insight into how a specific market or sector such as technology, healthcare or finance is behaving. Indexes such as the S&P500 and NASDAQ are well-known and are actively used as benchmarks since they can help understand how the market or sectors are performing [3].

The index that is going to be used as a benchmark in this work will be Standard & Poor's 500, also known as S&P500. This index fund tracks the 500 largest public companies in the USA and is commonly used as a benchmark for multiple hedge funds and studies because of its broader scope [4].

Understanding market trends is a crucial detail for investors. A market trend represents the overall direction of a market and can be categorized into three main types: Uptrend, Downtrend and Sideways [5].

In an Uptrend Market, prices generally increase, characterized by higher highs and higher lows. On the contrary, in a Downtrend Market, the prices generally decrease, characterized by lower highs and

lower lows. Finally, a Sideways market occurs when the prices fluctuate within a relatively narrow range without indicating a clear upward or downward trend, resulting in a horizontal trend. A sideways trend usually happens during prolonged trends, since it is nearly impossible for an upward or downward trend to sustain themselves over the long term [6].

In the stock market, an investor can take either long or short positions. A long position is when an investor buys a stock with the expectation that its value will increase over time. On the other hand, a short position is when an investor sells a stock that they do not own with the expectation that its value will decrease over time. This works by selling a ”borrowed” stock from a broker and buying it back later when the price is lower than the current price.

When investing in the stock market, the goal is to pick the best potential assets within the market in order to avoid losses and maximize returns. There are several different types of analyses that can be used to evaluate a company's stock. Some of the most common include **Fundamental Analysis**, which involves studying the company's financial statements and other metrics to determine its intrinsic value, and **Technical Analysis**, which involves analyzing historical price and volume data to identify trends and make predictions about the stock's future performance.

### 2.1.1 Fundamental Analysis

Fundamental Analysis is the process of evaluating a company's financial health and business model in order to determine its intrinsic value. This involves analyzing a company's financial statements, including its income statement, balance sheet, and statement of cash flows, as well as its management team, competition, and overall industry conditions.

The goal of Fundamental Analysis is to identify companies that are undervalued by the market and therefore have the potential to generate higher returns for investors. To do this, analysts often use ratios such as the price-to-earnings (P/E) ratio and the price-to-book (P/B) ratio to compare a company's stock price to its earnings and book value, respectively.

In addition to analyzing a company's financials, analysts also pay attention to factors such as its growth prospects, level of debt, and management quality. They may also consider external factors such as the overall state of the economy, government regulations, and industry trends.

Overall, Fundamental Analysis is a long-term approach that seeks to identify companies with strong fundamentals and the potential for long-term growth.

Piotroski et al. [7] proposed a fundamental scoring system called $\mathbf{F_{Score}}$ that is used to assess the financial strength of a company based on its annual financial statements. It is composed by the sum of 9 individual binary signals that are derived from financial ratios of the company. Each signal is assigned a value of either 0 or 1, and the signals values are summed up to obtain the company's score. The companies with a score of 8 or 9 are considered to be strong while the companies with a score of 0 to 2 are considered to be weak. The formula for the $\mathbf{F_{Score}}$ is described in the Equation 2.1

$$F_{Score} = F_{ROA} + F_{\Delta ROA} + F_{OCF} + F_{ACCRUAL} + F_{\Delta LEV} + F_{\Delta CURRATIO} + F_{\Delta SHARES} + F_{\Delta GMARGIN} + F_{\Delta ATURN} \tag{2.1}$$

The value assigned to each binary signal (0 or 1) depends on the result of ratios and the meaning behind those ratios. The formulas to calculate each ratio and the rules to attribute the value of each binary signal will be described below.

- **Return on Assets (ROA)** - ROA measures a company's profitability and efficiency in generating profits from its assets. It provides insights into how effectively a company is utilizing its assets to generate earnings. It is calculated by dividing the company's net income by its total assets, as it is shown in the Equation 2.2.

$$ROA = \frac{NetIncome}{TotalAssets} \tag{2.2}$$

  If the ROA value is positive then the binary signal will be assigned the value of 1, and 0 if the value of the ratio is negative.

  The change in Return of Assets ($\Delta$ROA) indicates the difference between the current value of ROA and the value from the previous year. If the result is positive then it will be assigned to the binary signal a value of 1, and 0 otherwise.

- **Operating Cash Flow (OCF)** - OCF measures the cash generated or used by a company's operations during a specific period. It provides insight into a company's ability to generate cash from its regular business activities. It is calculated by subtracting the company's cash revenue with the operating expenses paid in cash, as it is shown in the Equation 2.3.

$$OCF = CashRevenue - OperatingExpenses \tag{2.3}$$

  If the value of the OCF is positive, then it is attributed the value of 1 to the corresponding signal, and 0 if the value is negative.

- **Accruals (ACCRUAL)** - Accruals are accounting entries that reflect the revenue and expenses of a company during a specific time period. It is assigned a value of 1 if the value of OCF $>$ ROA and 0 otherwise.

- **Leverage ($\Delta$LEV)** - Leverage is used to assess the level of debt or financial leverage employed by a company. It provides insight into the extent to which a company relies on borrowed funds to finance its operations and investments. It is calculated by dividing the total debt of a company by its total assets, as it is shown in the Equation 2.4.

$$\Delta LEV = \frac{TotalDebt}{TotalAssets} \tag{2.4}$$

It is attributed the value of 1 if the current result is lower than the result from the previous year, and 0 if the current value is bigger.

- **Current Ratio (∆CURRATIO)** - The Current Ratio is used to assess a company's short-term liquidity or its ability to convert its current assets into cash to cover its current liabilities. It is calculated by dividing the current assets of the company with its current liabilities, as it is shown in the Equation 2.5.

$$\Delta CURRATIO = \frac{CurrentAssets}{CurrentLiabilities} \tag{2.5}$$

If its current value is bigger than the previous year's value it is assigned a value of 1, and 0 if the current value is smaller.

- **Number of Shares (∆SHARES)** - The number of shares is used to know if a company has issued new shares since the previous year. In the case it has been issued new shares, it will be attributed the value of 0 to its corresponding signal and 1 in the case it hasn't been issued new shares

- **Gross Margin (∆GMARGIN)** - The Gross Margin indicator is used to assess the profitability and efficiency of a company's operations. It measures the percentage of revenue that remains after deducting the cost of goods sold (COGS). It is calculated by subtracting the total revenue of the company with the COGS and dividing it by the total revenue, as it is shown in the Equation 2.6.

$$\Delta GMARGIN = \frac{TotalRevenue - COGS}{TotalRevenue} \cdot 100 \tag{2.6}$$

If the current result is higher than the previous year's result it is assigned the value of 1, and 0 otherwise.

- **Asset Turnover (∆ATURN)** - The Asset Turnover measures the company's efficiency in generating sales or revenue relative to its total assets. It provides insights into how effectively a company is utilizing its assets to generate sales. It is calculated by dividing its sales revenue with its total assets, as it is shown in the Equation 2.7.

$$\Delta ATURN = \frac{SalesRevenue}{TotalAssets} \tag{2.7}$$

It is assigned a value of 1 if it's current value is higher than the previous year's value, and 0 otherwise.

### 2.1.2 Technical Analysis

Technical Analysis in the stock market is the study of past market data, such as price and volume, to identify patterns and predict future market behavior. It is a way to analyze the market without taking into account fundamental factors, such as a company's financials or economic conditions [8].

The main tools used in Technical Analysis are charts and indicators, which are graphical representations of market data. These charts and indicators are used to identify trends, support and resistance levels, and other market patterns. For example, a trendline can be used to show the overall direction of a stock's price, while an oscillator can show when the stock is overbought or oversold.

Analysts use these tools to make predictions about future market movements and to identify entry and exit points for their trades. For example, if a stock has been trending upwards and an oscillator indicates that it is overbought, an analyst may sell the stock in anticipation of a price correction.

Just like fundamental indicators, there are hundreds of possible technical indicators to choose from, so the following ones are going to be used to detect the trend and the momentum of the selected companies and the index. Also, for each technical indicator, it is assigned a set of rules to determine the indicator's score, which ranges from Very Low Score (-1), Low Score (-0.5), Neutral Score (0), High Score (0.5) and Very High Score (1). An example of these rules is shown in the Table 2.1.

- **Simple Moving Average (SMA)** - A SMA is a commonly used statistical indicator that helps smooth out fluctuations in a dataset and identify trends over a specified period of time. It is calculated by taking the average of a set of values over a specific time interval or window. The formula of SMA is described in the Equation 2.8:

$$SMA_n = \frac{1}{n} \sum_{i=1}^{n} X_i \tag{2.8}$$

  Where the value of $n$ is the window size and $X(i)$ is the value of the stock price at the time $i$.

  The simple moving average is useful in analyzing trends and identifying potential support and resistance levels in financial markets. It smooths out short-term fluctuations and provides a clearer picture of the underlying price direction.

- **Exponential Moving Average (EMA)** - The EMA is a type of moving average calculation that places more weight on recent data points while decreasing the significance of older data points. It is commonly used in financial analysis and Technical Analysis to smooth out price data and identify trends.

  To calculate the EMA, it is used a SMA as the initial value. Then, for each subsequent data point, it is calculated the EMA using the formula described in the Equation 2.9 that incorporates the previous EMA value, the current data point, and a smoothing factor, denoted as alpha ($\alpha$).

$$EMA_t = \begin{cases} \alpha \cdot X_1 + (1-\alpha) \cdot SMA & t = 1 \\ \alpha \cdot X_t + (1-\alpha) \cdot EMA_{t-1} & t > 1 \end{cases} \tag{2.9}$$

  The smoothing factor determines the weight given to the current and previous EMA values. It is typically a value between 0 and 1, with higher values placing more weight on recent data.

  As new data points are added, the EMA calculation recursively incorporates the most recent data while gradually diminishing the influence of older data. This characteristic makes the EMA more

responsive to recent price movements compared to a simple moving average, which treats all data points equally.

The EMA can be used to identify trends, generate buy or sell signals, and estimate support and resistance levels in Technical Analysis. Shorter EMA periods are more sensitive to price changes and are commonly used for short-term analysis, while longer EMA periods provide a smoother trend indication and are used for longer-term analysis.

The rules presented for EMA are shown in Table 2.1:

Table 2.1: EMA scoring rules.

| **Value** | **Score** | **Description** |
|---|---|---|
| -1 | Very Low Score | Price line crosses below the EMA line |
| -0.5 | Low Score | EMA line is decreasing |
| 0 | Neutral Score | EMA line has the same values since last period |
| 0.5 | High Score | EMA line is increasing |
| 1 | Very High Score | Price line crosses above the EMA line |

- **Hull Moving Average (HMA)** - The HMA is a technical indicator that aims to provide a smoother representation of price trends compared to traditional moving averages. It was developed by Alan Hull and seeks to reduce lag and noise while maintaining responsiveness to price movements.

  The calculation of the HMA starts by calculating the weighted moving average (WMA) of the price data. The WMA assigns higher weights to recent prices and lower weights to older prices. The period used for the WMA is typically half the length of the desired HMA period. Then it is calculated the WMA of the price data again, using the square root of the period as the length of the WMA, subtract the second WMA from the first WMA calculated, and finally calculate the WMA of the difference calculated previously using the square root of the desired HMA period. The formula for HMA is described in the Equation 2.10:

$$HMA_n = \sqrt{n} \cdot WMA\left(2 \cdot WMA(\frac{n}{2}) - WMA(n), \sqrt{n}\right) \tag{2.10}$$

  The HMA is based on the concept of weighted moving averages but incorporates the square root of the period to adjust the smoothing effect. This adjustment helps to reduce lag and produce a more responsive indicator.

  Traders and analysts use the HMA to identify the direction of a trend and generate buy or sell signals. When the HMA is rising, it indicates an uptrend, while a declining HMA suggests a downtrend.

  The rules presented for HMA are shown in Table 2.2:

  In the Figure 2.1 it is shown the comparison of a SMA, EMA and HMA with a 20-day period. It is clear that the EMA is more volatile to the sudden changes in the data compared to the SMA, because of the weights of the most recent data points being more significant than the oldest data points, but it still maintains a clear representation of the direction of the market or asset. The HMA

Table 2.2: HMA scoring rules.

| Value | Score | Description |
|---|---|---|
| -1 | Very Low Score | HMA slope changes to a downward direction |
| -0.5 | Low Score | HMA line is decreasing |
| 0 | Neutral Score | HMA line has the same values since last period |
| 0.5 | High Score | HMA line is increasing |
| 1 | Very High Score | HMA slope changes to an upward direction |

is the most sensitive of the three moving averages to the most recent data, simulating a more smooth representation of the data and with less lag than the other two moving averages.

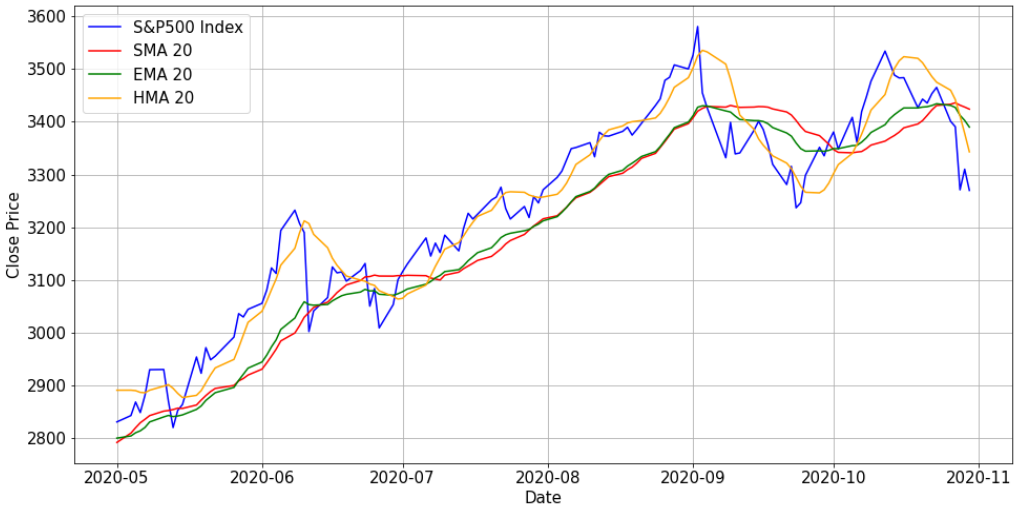


Figure 2.1: The comparison of a SMA, EMA and HMA of S&P500 with a period of 20 days.

- **Rate of Change (ROC)** - The ROC is a technical indicator used in finance to measure the percentage change in the price of a financial asset over a specified period. It provides insights into the momentum or speed at which the price of an asset is changing. The formula for calculating ROC is described in the Equation 2.11:

$$ROC_n = \frac{Close_n - Close_{n-k}}{Close_{n-k}} \times 100 \tag{2.11}$$

The $Close_n$ represents the close price of the stock at the end of the $n$ period, $Close_{n-k}$ represents the close price of $k$ periods before the $n$ period.

A positive ROC indicates an upward price momentum while a negative ROC suggests a downward price momentum. The magnitude of the value also provides information about the strength or weakness of the momentum. In the Figure 2.2 it is shown an example of a ROC with a 20-day period.

Traders and analysts use ROC to identify potential buy or sell signals. A large positive ROC may


Figure 2.2: The ROC of the S&P500 with a 20-day period.

indicate an overbought condition, suggesting that the price has increased too rapidly and could potentially reverse. Conversely, a large negative ROC may signal an oversold condition, indicating that the price has declined too rapidly and could potentially rebound.

The rules presented for ROC are shown in Table 2.3:

Table 2.3: ROC scoring rules.

| **Value** | **Score** | **Description** |
|---|---|---|
| -1 | Very Low Score | ROC line crosses below 0 |
| -0.5 | Low Score | ROC increases while price decreases |
| 0 | Neutral Score | Both ROC and price have the same values since last period |
| 0.5 | High Score | ROC decreases while price increases |
| 1 | Very High Score | ROC line crosses above 0 |

- **Double Crossover** - A double crossover is a technical indicator used in finance to analyze the relationship between two moving averages.

  The double crossover involves two moving averages: a shorter-term moving average and a longer-term moving average. The shorter-term moving average responds more quickly to price changes, while the longer-term moving average provides a smoother trend indication.

  When the shorter-term moving average crosses above the longer-term moving average, it is considered a bullish signal, indicating a potential upward trend in the price. Traders and investors interpret this as a buying opportunity, anticipating further price increases.

  Conversely, when the shorter-term moving average crosses below the longer-term moving average, it is considered a bearish signal. This suggests a potential downward trend in the price and is seen as a signal to sell or take a short position. In the Figure 2.3 it is shown an example of a

Double Crossover of an EMA with a 20-day period and a EMA with a 50-day period.

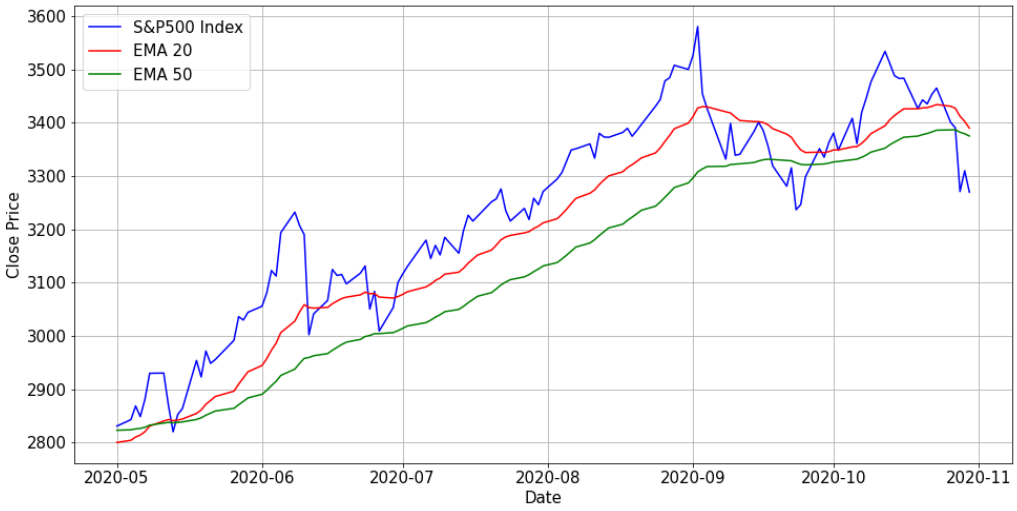


Figure 2.3: Double Crossover of an EMA with a 20-day period and a EMA with a 50-day period.

The rules presented for Double Crossover are shown in Table 2.4:

Table 2.4: Double Crossover scoring rules.

| Value | Score | Description |
|---|---|---|
| -1 | Very Low Score | Shorter EMA crosses below the longer EMA line |
| -0.5 | Low Score | Both EMA's are decreasing |
| 0 | Neutral Score | Both EMA's have the same values since last period |
| 0.5 | High Score | Both EMA's are rising |
| 1 | Very High Score | Shorter EMA crosses above the longer EMA line |

- **Relative Strength Index (RSI)** - The RSI is a popular technical indicator used in financial analysis to assess the momentum and overbought or oversold conditions of a financial instrument, such as a stock or a market index. The RSI is a bounded oscillator that fluctuates between 0 and 100, providing insights into the strength and potential reversal of price movements. The formula for calculating the RSI is described in the Equation 2.12

$$\text{RSI} = 100 - \left( \frac{100}{1 + \frac{\text{Average Gain}}{\text{Average Loss}}} \right) \tag{2.12}$$

Traders and investors use the RSI to identify potential buy and sell opportunities. For example, if the RSI is above 70, it might signal an overbought condition and a potential sell opportunity. Conversely, if the RSI is below 30, it might suggest an oversold condition and a potential buy opportunity. In the Figure 2.4 it is shown an example of a RSI with a 14-day period.

The rules presented for RSI are shown in Table 2.5:

- **Moving Average Convergence Divergence (MACD)** - The MACD is a popular technical indicator

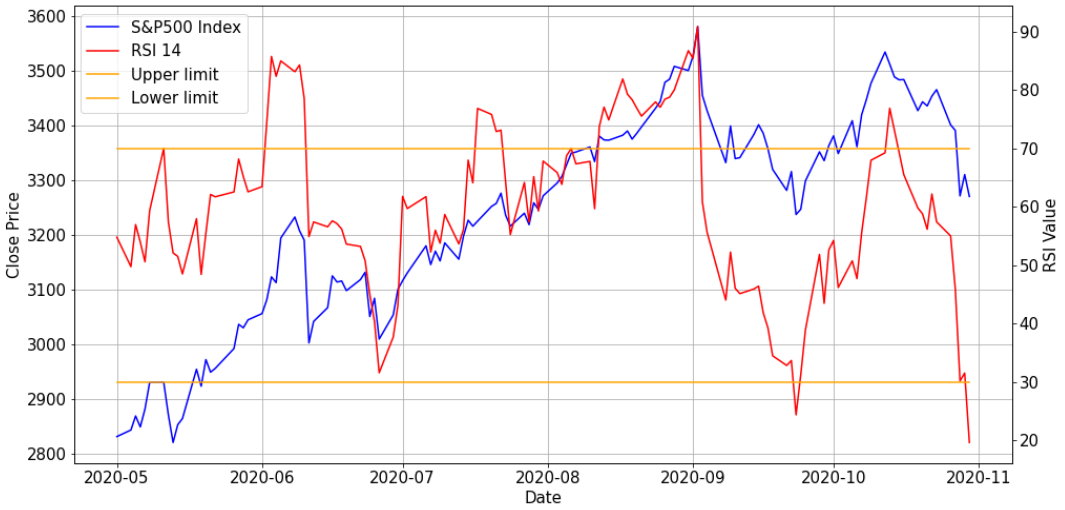


Figure 2.4: The RSI of the S&P500 with a 14-day period.

Table 2.5: RSI scoring rules.

| **Value** | **Score** | **Description** |
|---|---|---|
| -1 | Very Low Score | RSI line crosses below 70 |
| -0.5 | Low Score | RSI line is decreasing between the extreme levels |
| 0 | Neutral Score | RSI line has the same values since last period |
| 0.5 | High Score | RSI line is rising between the extreme levels |
| 1 | Very High Score | RSI line crosses above 30 |

used in finance to analyze price trends and generate trading signals. It consists of two main components: the MACD line and the signal line.

The MACD line is calculated by subtracting a longer-term EMA from a shorter-term EMA of a given financial instrument's price. The most commonly used time periods for the two EMAs are 12 and 26 trading days, respectively. The MACD line reflects the momentum of the price movement and provides an indication of whether the price is trending upwards or downwards.

The signal line is a smoothed average of the MACD line and is typically a 9-period EMA. It acts as a trigger for buy and sell signals. When the MACD line crosses above the signal line, it generates a bullish signal indicating a potential buy opportunity. Conversely, when the MACD line crosses below the signal line, it generates a bearish signal indicating a potential sell opportunity. In the Figure 2.5 it is shown an example of the MACD line and the signal line.

The MACD also includes a histogram, which represents the difference between the MACD line and the signal line. It provides further insight into the strength of the price momentum.

Traders and analysts use the MACD to identify potential entry and exit points in the market. When the MACD line moves above zero, it suggests that bullish momentum is increasing, and traders may consider buying. Conversely, when the MACD line moves below zero, it suggests increasing bearish momentum, and traders may consider selling.

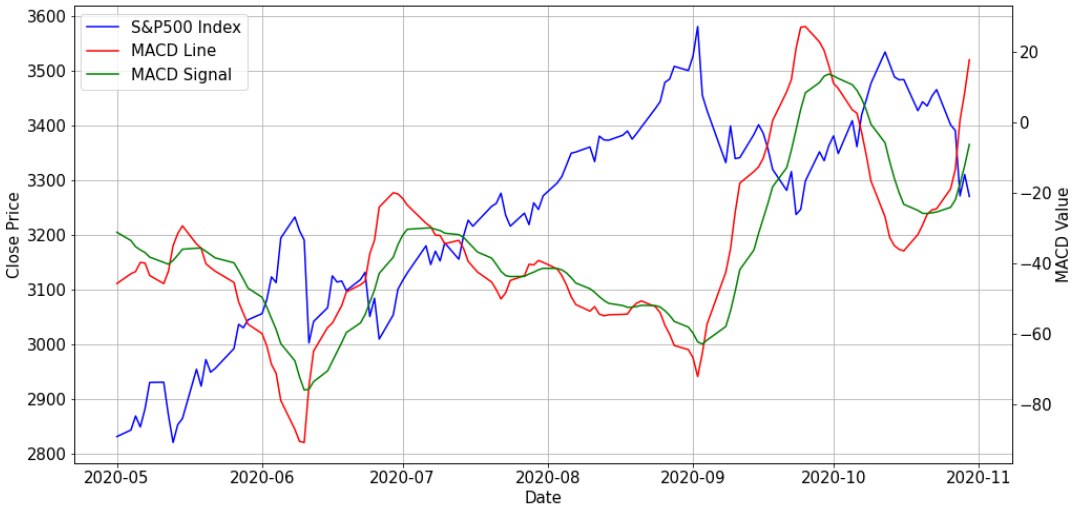


Figure 2.5: The MACD line and the signal line of the S&P500.

The rules presented for MACD are shown in Table 2.6:

Table 2.6: MACD scoring rules.

| Value | Score | Description |
|---|---|---|
| -1 | Very Low Score | Histogram crosses below 0 |
| -0.5 | Low Score | Histogram is decreasing on negative direction |
| 0 | Neutral Score | Histogram has the same values since last period |
| 0.5 | High Score | Histogram is rising on positive direction |
| 1 | Very High Score | Histogram crosses above 0 |

- **On Balance Volume (OBV)** - The OBV is a technical indicator used in financial markets to measure the cumulative buying and selling pressure of an asset. It provides insights into the strength of a price trend by considering volume changes.

  The calculation of OBV involves adding the volume of a stock to a running total if the price moves up, and subtracting the volume if the price moves down. The idea behind OBV is that volume precedes price movement, and changes in volume can often signal a potential trend reversal or continuation. To calculate the OBV, it is used an initial value (typically 0), followed by the rules described in the equation 2.13:

$$OBV_t = OBV_{t-1} + \text{Volume} \times \begin{cases} 1, & \text{if the closing price is higher than the previous period} \\ -1, & \text{if the closing price is lower than the previous period} \\ 0, & \text{if the closing price is the same as the previous period} \end{cases} \quad (2.13)$$

By accumulating volume based on price changes, OBV attempts to identify periods when buying

or selling pressure is increasing or decreasing. Rising OBV suggests a positive trend where buying pressure is stronger, while falling OBV indicates a negative trend where selling pressure is stronger. In the Figure 2.6 it is shown an example of the OBV.

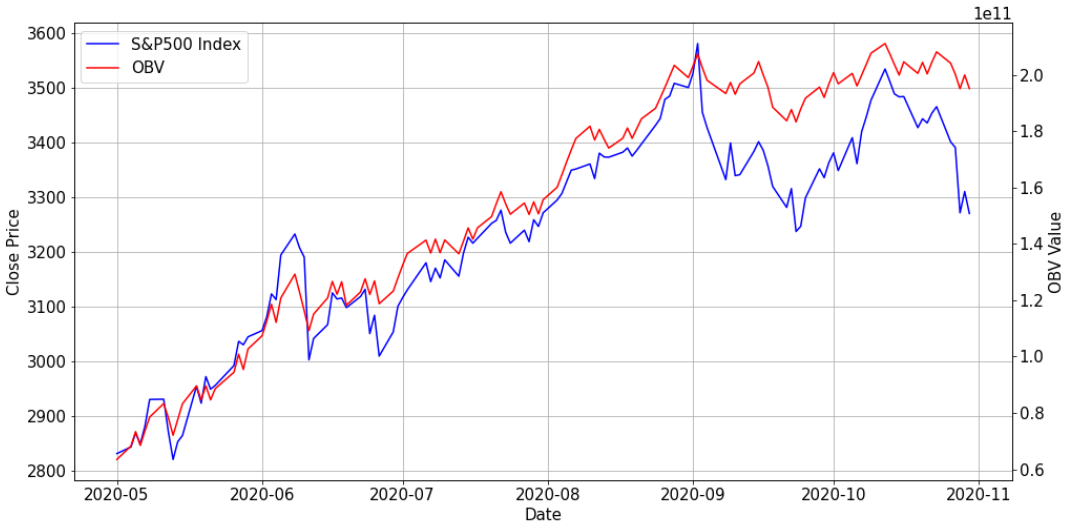


Figure 2.6: The OBV of the S&P500 index.

The rules presented for OBV are shown in Table 2.7:

Table 2.7: OBV scoring rules.

| **Value** | **Score** | **Description** |
|---|---|---|
| -1 | Very Low Score | OBV is falling simultaneously with price indicating a clear down trend |
| -0.5 | Low Score | OBV is decreasing and price is rising indicating a possible uptrend breakout |
| 0 | Score | OBV and price have the same values since last period |
| 0.5 | High Score | OBV is rising and price is declining indicating a possible downtrend breakout |
| 1 | Very High Score | OBV is rising simultaneously with price indicating a clear up trend |

## 2.2 Computational Algorithms

In order to obtain good results when trading stocks based on their performance relative to the index performance, it is crucial to perfect the timing of opening and closing positions. For that purpose, Genetic Algorithms will be used to open and close positions on stocks from the S&P500 in order to maximize the returns of the system.

Genetic Algorithms are very useful to determine optimal or near-optimal solutions in large solution spaces compared to other optimization techniques because of their features inspired by natural evolution, such as: fitness function, crossover and mutation [9]. These features will be explained in detail in the Section 2.2.1.

### 2.2.1 Genetic Algorithm

Genetic Algorithms are a type of computational method that uses principles of natural selection and genetics to solve complex problems. They are inspired by the way that natural evolution occurs, where the fittest individuals are more likely to survive and reproduce, passing on their genetic traits to the next generation.

In Genetic Algorithms, the first step is to generate a randomized population. The population is constituted by individuals where each individual has a chromosome where each chromosome represents a set of genes where a gene represents the value of a variable. The initial population is a set of possible random solutions specific to the problem.

The next step of the algorithm is to evaluate each individual of the population using a fitness function, which is a function that ranks the individual according to the problem. This function is the parameter that needs to be optimized since individuals with higher rankings will be more significant for the next generation.

After all individuals of the population are ranked, the next step of the algorithm will be the selection of the parents of the next generation. The individuals with higher rankings are most likely to be selected since they are closer to the optimal solution. There are several selection methods such as roulette wheel selection, tournament selection and rank selection. In order to choose the best selection method for the designated problem it is necessary to have into account the range of the fitness values. For instance, tournament selection and rank selection are capable of handling negative fitness values, unlike roulette wheel selection. Rank selection is preferable when the fitness values of the population are very similar since it prioritizes individuals based on their ranking instead of their specific fitness values.

Following the selection of the parents, the next step is called crossover which consists on combining the genes of the parents to generate the offspring, which will constitute the next generation. Like in the selection process, there are several methods for the combination of the genes. There is also a mutation operation after this step which consists on modifying randomly selected genes of the offspring based on a random chance previously defined.

The last step of the algorithm is to evaluate the offspring using the fitness function used previously and verify if any solutions have met the criteria or if the algorithm has reached a determined number of

cycles, in which case the algorithm will exit the loop and end. If the solutions have not met the criteria and the number of cycles haven't reached the limit, the offspring will be selected to be parents for the next generation and the cycle will repeat again.

Some of the key advantages of Genetic Algorithms are their ability to find high-quality solutions even in the absence of complete information, their adaptability to changing conditions, and their ability to explore a large search space.

In Figure 2.7 it is shown the flowchart of a genetic algorithm.

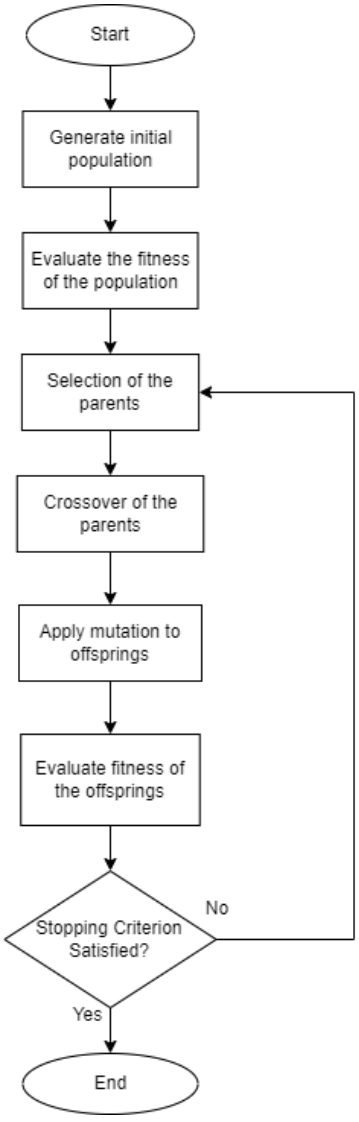


Figure 2.7: Genetic Algorithm Flowchart.

### 2.2.2 XGBoost

In machine learning, ensemble is a technique where multiple predictor models are combined to solve a prediction problem, using the diversity among individual models in order to improve the final solution compared to the performance of a single model.

Boosting is an ensemble technique that combines multiple base models sequentially, where each model corrects the errors made by the previous models to minimize the final prediction errors. During the training, each instance of the training dataset will have its weight adjusted based on the model performance, where the misclassified instances will have its weight increased, so the following models will focus more on these instances. Each model has a certain weight when contributing to the final prediction. The models with better performance will have a higher weight compared to those who perform poorly. Decision Trees are a common base model in boosting algorithms and consist of a hierarchical tree structure where each internal node represents a decision based on a feature of the dataset and each leaf node represents the final prediction.

Gradient Boosted Decision Trees (GBDT) is a boosting method introduced by Friedman [10] that combines the use of decision trees as the base model with a Gradient Descent optimization in order to minimize the loss function. At each boosting stage, the models are updated according to the Equation 2.14

$$F_m(x_i) = F_{m-1}(x_i) + \sum_{j=1}^{J_m} \gamma_{jm} 1 R_{jm}(x) \tag{2.14}$$

where $m$ is the current stage, $J_m$ is the number of leaves in the stage $m$, the $R_{jm}$ are the disjoint regions that the tree partitions the input space into, and $\gamma_{jm}$ is a value chosen to minimize the loss function, calculated using the Equation 2.15

$$\gamma_{jm} = \textit{argmin}_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma) \tag{2.15}$$

XGBoost, which stands for eXtreme Gradient Boosting, is a variation of the Gradient Boosted Decision Trees algorithm and considered as a state of the art method capable of outperforming many other systems with fewer resources [11]. The output of the models in XGBoost are generated similarly with the models of the GBDT with a few different notations, as shown in the Equation 2.16

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i), f_k \in F \tag{2.16}$$

where $K$ represents the number of trees and $f$ is a function in the functional space $F$, which is represented in the Equation 2.17

$$F = \{f(x) = w_{q(x)}\}(q : \mathbb{R}^m \to T, w \in \mathbb{R}^T) \tag{2.17}$$

where $q$ represents a tree structure, $T$ is the number of leaves in the tree and $w$ is the leaf weight.

The objective function to minimize is represented in the Equation 2.18 and it is composed of two

parts: a loss function and a regularization term.

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{2.18}$$

The loss function represented in Equation 2.18 by $l$ is a function measuring the difference between the predicted value $\hat{y}_i$ and the true value $y_i$, for all $n$ instances of the training dataset. The regularization term represented by $\Omega(f_k)$ penalizes the complexity of the model and is one of the key improvements over the GBDT by helping avoiding overfitting, it is calculated by the Equation 2.19

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda ||w||^2 \tag{2.19}$$

where $T$ is the number of leaves of the tree $f_k$, $w$ are the leaf weight and $\lambda$ is the regularization parameter controlling the strength of the regularization.

As the model is trained additively, the Equation 2.18 can be rewritten as the Equation 2.20

$$\mathcal{L}(t) = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \sum_{k=1}^{K} \Omega(f_t) \tag{2.20}$$

where $f_t$ is the function that helps minimize the loss function.

Another key improvement of the XGBoost over the GBDT algorithm is the implementation of the decision trees in a parallel way instead of sequentially, allowing for a faster performance and a scalable implementation. The architecture of the XGBoost is represented in the Figure 2.8

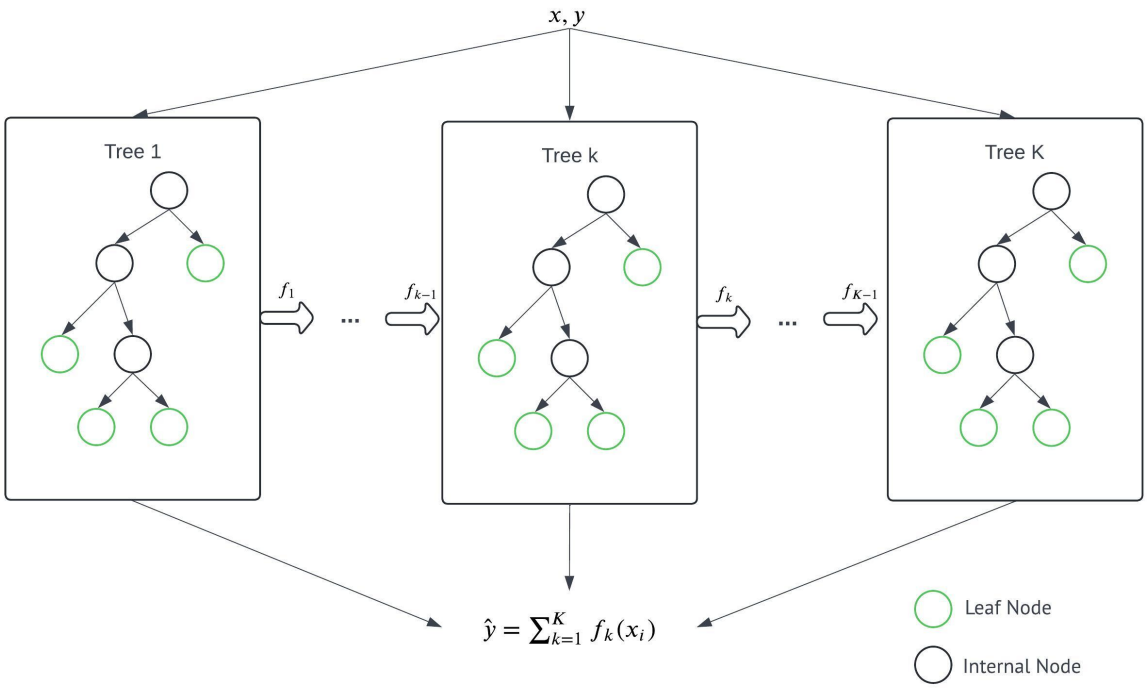


Figure 2.8: XGBoost Architecture.

## 2.3 State of the Art

The use of both Fundamental and Technical Analysis alongside the use of Genetic Algorithms for trading in stock market has shown great results in many works. In Almeida and Neves [12], it was proposed a self-adaptive evolutionary algorithm for portfolio composition, where the probability of mutation and crossover was optimized according to the facing scenario in hope to achieve optimal results. In this work, it was used a fundamental scoring system called F-Score that ranges from 0-9 (9 being the best) in order to select the best companies from the S&P500 index. It was also tested the use of a static portfolio and a dynamic portfolio which changed every year. In the work of Silva et al. [13], instead of using a single-objective Genetic Algorithm, it was proposed a Multi-Objective Evolutionary Algorithm (MOEA) for portfolio management, where the return and the variance of the returns were used as the optimization objectives. That work tested 3 different chromosome models, each chromosome was divided into two parts, the first group was composed of fundamental indicators for the selection of the companies and the second group was composed of technical indicators for the trading parameters. Both Almeida and Neves [12] and Silva et al. [13] showed great results, outperforming the S&P500 index. Almeida and Neves [12] also showed that the use of a static portfolio results in better returns compared to a dynamic portfolio. The main reason for that result is the market seasonality. During a bull market, all companies tend to thrive while in a bear market any company has a chance to decline. And Silva et al. [13] managed to obtain great returns while minimizing the variance of the returns, demonstrating it as a good alternative for single-objective Genetic Algorithms. Those two articles influenced the proposed system of this work by using the fundamental ranking system F-Score for the selection of the companies, along with the use of a Genetic Algorithm to help with the selection and trading.

The complexity of the technical data can affect the performance of algorithms because of its noisy data. In Nobre and Neves [14], the architecture of the proposed system was divided in several layers such as: Data Layer, Business Logic Layer and User Interface Layer. In the Data Layer it was used Principal Component Analysis (PCA) to reduce the dimensionality of the data, and Discrete Wavelet Transform (DWT) to reduce the noise from the data. After the Data Layer, the data are normalized and ready to be used in the Business Logic Layer, where a XGBoost Binary classifier determines if the system should take a Long or a Short position for the next day. A Multi-Objective Optimization Genetic Algorithm (MOO-GA) was used to optimize the parameters of the XGBoost classifier in order to improve the returns and minimize the risk. The results showed that the system performed significantly well, although it did not translate well in a market with a strong uptrend or downtrend.

Market classification can improve a system performance by focusing on specific market conditions instead of having just one model. In Matos and Neves [15], it was tested the use of multiple classifiers to classify the Forex Market in five different categories. Each category was assigned a unique trading algorithm, optimized for that specific market condition by using a Strong Pareto Evolutionary Algorithm 2 to maximize the profits and minimize the losses. The tested classifiers were Support Vector Machines, Multinomial Logistic Regression, Random Forest, Gradient Boosted Decision Trees, XGBoost, Ensemble of Support Vector Machines, Ensemble of Logistic Regression and a Cascading Logistic Regression

to Gradient Boosted Decision Trees. The results of the classifiers showed that the Logistic Regression had the worst overall scores, both in terms of Precision, Recall and Accuracy. The Support Vector Machine had a better performance compared to the Logistic Regression, on par with the Ensemble of Support Vector Machines. The rest of the classifiers achieved very similar results in terms of precision and accuracy, with the best classifier being the Cascade Logistic Regression to Gradient Boosted Decision Trees Classifier with a precision of 79%, a recall of 73% and an accuracy of 74.6%. Although the Cascade Classifier was the best, the XGBoost Classifier, the Gradient Boosted Decision Trees and the Random Forest Classifier achieved results very similar to the Cascade Classifier, with 78% in precision, 73% in recall except the Random Forest which achieved a recall of 72%, and 74.1%, 74.3% and 73.6% of accuracy, respectively. This work achieved a 19.55% return over five years, beating the return of a Forex Hedge Fund Index which only achieved 6.49% in the same period.

In Canelas et al. [16], it was proposed a new strategy for pattern discovery in financial time series. This strategy is called Symbolic Aggregate approXimation Genetic Algorithms (SAX-GA) and consists on converting a time series into a symbolic representation using Symbolic Aggregate approXimation (SAX) and using a Genetic Algorithm to identify the most relevant patterns and establish a set of rules for the investment decisions based on the patterns discovered by the SAX. The case studies shown in that work compare the use of a single chromosome structure and a multi-chromosome structure. The main difference between those two case studies was that the single chromosome structure only allows for one type of investment strategy, while the multi-chromosome structure allows the use of both Long and Short positions. The results showed that the use of a multi-chromosome structure allows better results since it can benefit from the use of Long and Short positions and it can profit in a bear or a bull market conditions simultaneously. This method was able to outperform the Buy and Hold (B&H) strategy, which consists on buying stocks and hold onto them for extended periods of time, generating returns if the value of the stocks increase.

Focusing more on Technical Analysis, Gorgulho et al. [17] proposed a portfolio management strategy based on purely Technical Analysis using Genetic Algorithms. The technical indicators chosen in that strategy were mostly trend following, in order to identify trends in the market, and momentum oscillators which measure the velocity of directional price movement in order to identify the strength of a price movement. Each technical indicator was assigned with four distinct scores: ”Very Low Score”, ”Low Score”, ”High score” and ”Very High Score”. The ”Very Low Score” and ”Low Score” indicate a strong or a potential opportunity to sell or take a short position, while the ”High Score” and ”Very High Score” indicate a strong or a potential opportunity to take a long position. The selection of the assets and the type of positions to take is based on the sum of each score, the optimized weights for each technical indicator and four bound values that indicate the limits to take a long or a short position and limit to close that position. Almeida et al. [18] also proposed a purely Technical Analysis for trading in the Foreign Exchange market (Forex) but it was used a Support Vector Machine classifier (SVM) and three Genetic Algorithms. In this strategy, the SVM classifies the current market trend as uptrend, downtrend, or sideways. Then, the system selects the Genetic Algorithm that has been trained for that market based on the trend chosen, and the selected Genetic Algorithm changes the weights of the set of rules for the

trading strategy and optimizes the leverage multiplier to be used. The study also tested the use of Price Sequence and Technical Indicators as data for the SVM classifier and concluded that the use of Price Sequence yields better results in terms of Precision, Recall, and Accuracy compared to Technical Indicators. Overall, these two works achieved a great Return on Investment (ROI), surpassing the S&P500 performance, proving them as viable solutions for trading. Those two articles have also influenced the proposed system by applying Technical Analysis for trading along with the use of a Genetic Algorithm to optimize the weights of the technical indicators.

On Table 2.8 there's a summary for works mentioned previously, using evaluation metrics such as Rate of Return (ROR) and Maximum Draw Down (MDD).

Table 2.8: Previous Works

| Reference | Year | Financial Markets | Methodology | Dataset | Evaluation Metrics | Results |
|---|---|---|---|---|---|---|
| [12] | 2022 | Some S&P500 Companies | Self-Adaptive EA | 2012 - 2018 | ROR | ROR: 39,49% |
| [13] | 2015 | Some S&P500 Companies | MOEA | 2013 - 2014 | ROI, Variance of returns | 1st model: ROI: 50.24% Var: 19.71% 2nd model: ROI: 36.4% Var: 8.02% 3rd model: ROI: 28.3% Var: 19.6% |
| [14] | 2019 | Brent Crude futures contract, Corn futures contract, Exxon Mobil Corporation stocks, Home Depot Inc. stocks, S&P500 index | PCA, DWT, XGBoost, MOO-GA | Brent Crude: 17/02/2015 - 13/04/2017 Exxon Mobil: 30/01/2015 - 13/04/2017 Home Depot: 30/01/2015 - 13/04/2017 S&P500: 30/01/2015 - 13/04/2017 | ROR, MDD, Accuracy, Sharpe Ratio | Brent Crude: 30.56% Exxon Mobil: 23.71% Home Depot: 38.34% S&P500: 35.21% |
| [15] | 2023 | Forex EUR/USD | LR2GBDT, MOEA | 2015-2020 | ROI, Accuracy | ROI: 19.55%, Accuracy: 74.6% |
| [16] | 2013 | 99 S&P500 Companies, S&P500 index | SAX-GA | 2005 - 2010 | ROI, Accuracy | ROI: 482.41% |
| [18] | 2018 | Forex EUR/USD | SVM, Dynamic GA | 2013 - 2016 | ROI, Accuracy | ROI: 83.5% |
| [17] | 2011 | DJI | GA | 2003 - 2009 | ROI | ROI: 62.95% |

# Chapter 3

# Implementation

This chapter will explain the implementation of the proposed system and how each module will function. It will begin by presenting a general description of the entire architecture, including the order and objectives of each module. Then, each module will be explained in detail.

## 3.1 General Description

The architecture of the proposed system is showed in the Figure 3.1, it consists of four modules: Data Processing Module, Asset Selection Module, Model Training Module and Trading Module.

The first module of the system is the **Data Processing Module** where it extracts information from the S&P500 index and its companies. Then the data is cleaned and validated in order to be processed and used to create the fundamental and technical indicators described in Chapter 2. It also receives the XGBoost Classifier from the **Model Training Module** in order to classify the S&P500 index market trends.

The next module is the **Asset Selection Module** where the selection of the companies stocks are going to take place. This selection will be based on the fundamental ranking system $\mathbf{F_{Score}}$, which will use the fundamental indicators mentioned in the Section 2.1.1. The ranking of the companies is determined by averaging the $\mathbf{F_{Score}}$ of multiple quarters, resulting in the company's score. The companies with the best scores will be selected to trade in the **Trading Module**.

The third module is the **Model Training Module**, where the XGBoost Classifier and the weights of the Genetic Algorithms are trained in order to be used in the **Trading Module**. The XGBoost Classifier uses the data and technical indicators of S&P500 index between the years 1970-01-01 and 2019-12-31 in order to accurately classify the current market trend into 3 categories: **Uptrend**, **Downtrend** and **Sideways**. Afterwords, two sets of weights are trained using Genetic Algorithms, with one being trained during periods where the vast majority of its data points are classified as **Uptrend** by the XGBoost, and the other set of weights is trained during periods where the vast majority of its data points are classified as **Downtrend**. The output of this module is going to be a XGBoost Classifier capable of accurately classifying the current market trend and the weights of the two Genetic Algorithms, where each set of

weights is specialized for a specific market trend.

Finally, the **Trading Module** is where the system will open or close positions according to the market trend classification and the stocks "Divergence Scores", which represent the divergence between the stock performance and the S&P500 index performance, calculated by comparing the technical indicators used for the selected companies and the S&P500 index. The first step is to iterate every stock selected by the **Asset Selection Module**, and then iterate through every day of the trading session. For each trading day, it is calculated the "Divergence Score" for each technical indicator, which is the result of the difference between a stock technical indicator score and the respective S&P500 index technical indicator score. Afterwords, the weights of the Genetic Algorithms are selected according to the market trend classification and they are used to calculate the "Total Divergence Score". This score represents the comparison between the stock performance and the index performance for a specific data point, which allows to determine if a stock performance is diverging or converging with the index performance. When a stock starts to diverge or converge from the S&P500 index it will most likely open or close a position respectively. The type of position, long or short, also depends on whether the value of the indicators is negative or positive, respectively. And because of the uncertainty of the market direction during periods classified as **Sideways**, it was restricted the option to open and close positions only during periods classified as **Uptrend** and **Downtrend**. This module outputs the results of the trading session, including the total profit, the total amount invested, the ROI, as well as the results using the B&H strategy during the trading session.

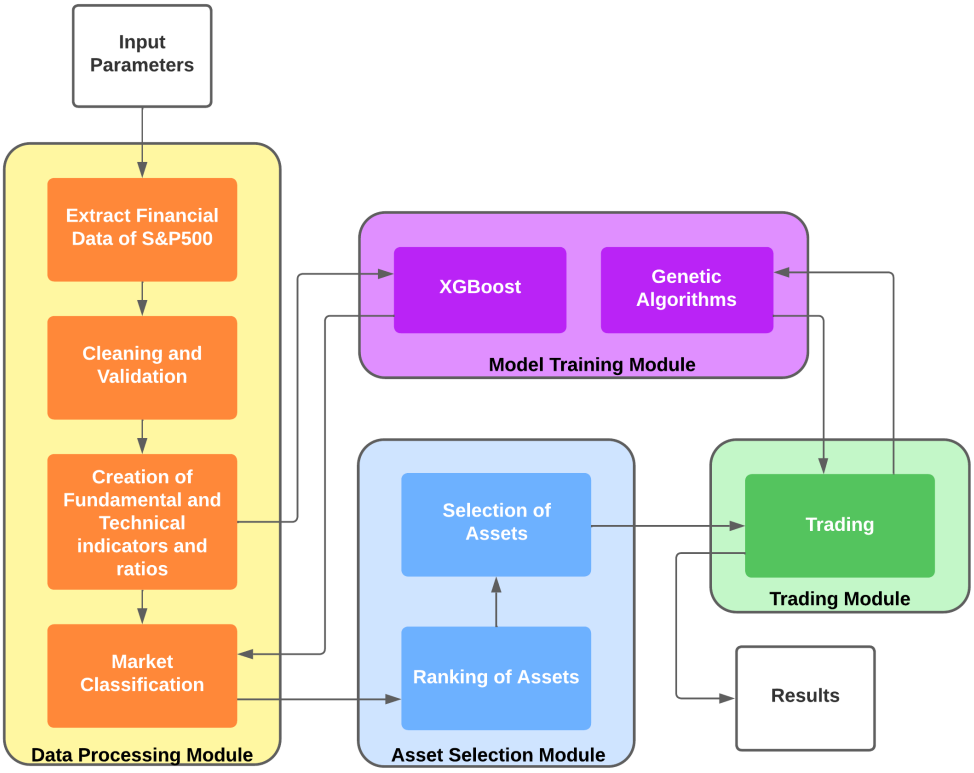


Figure 3.1: Architecture of the proposed system.

## 3.2 Flow Description

The flow of the architecture shown in the Figure 3.1 is represented in the fluxogram shown in the Figure 3.2 and will be described below step by step:

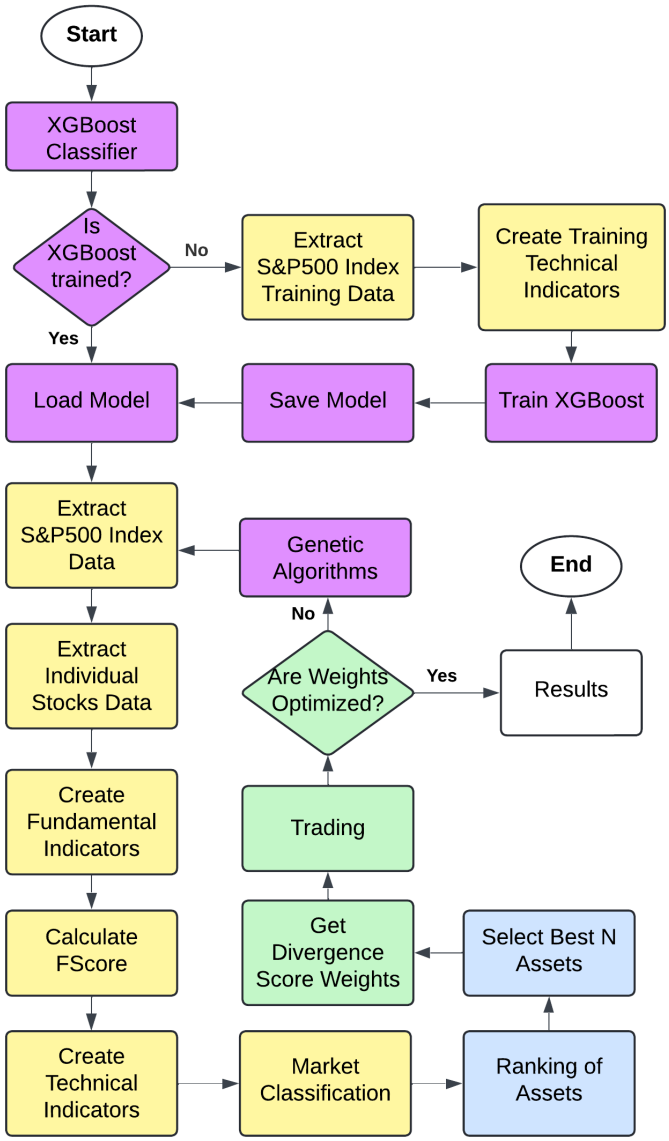


Figure 3.2: Fluxogram of the proposed system.

1. The first step is to instantiate the XGBoost Classifier. If the XGBoost Classifier is already trained then it loads the model from a JSON file. If not, then the model receives the data and technical indicators from the S&P500 index between 1970-01-01 and 2019-12-31 so it can train.

2. After the XGBoost Classifier is instantiated, the next step is to extract the fundamental and technical data of the stocks and the S&P500 index for the periods of the trading session.

3. When the data has been extracted and verified, the next step is to calculate the $\mathbf{F_{Score}}$ of the stocks, as well as the technical indicators of the stocks and the S&P500 index for the trading period.

4. With the technical indicators calculated, classify every data point of the S&P500 index during the trading session.

5. With the $\mathbf{F_{Score}}$ of every stock calculated, the next step is to rank the stocks using an average of the $\mathbf{F_{Score}}$ of each quarter of the trading session.

6. After every stock is ranked, select the best *N* stocks.

7. Get the weights used in the **Trading Module** to calculate the "Total Divergence Score".

8. Calculate the "Total Divergence Score" for each data point of every stock and open or close positions when the scores surpass the defined thresholds. Return the results from the trading session.

9. If the weights were not optimized, use the Genetic Algorithms to optimize them by repeating the process from step 2 for different periods, focusing on periods mainly consisting of data points classified as **Uptrend** or **Downtrend**. Use the results from the trading session to optimize the weights.

## 3.3 Module Description

In the following sections, there is going to be a detailed description of each module of the proposed system, explaining every component and how the modules interact with each other. It is worth specifying that this work was implemented using the Python programming language [19], along with the use of the Anaconda Navigator software [20] for the management of the external Python Libraries used, and Spyder [21] as the Integrated Development Environment (IDE).

### 3.3.1 Data Processing Module

The **Data Processing Module**, is responsible for extracting and selecting the ticker symbols of the stocks in the S&P500 index, obtaining the technical data and financial statements from the selected companies and the S&P500 index, calculating the fundamental and technical ratios mentioned in Chapter 2.1.1 and Chapter 2.1.2, respectively, and to classify the S&P500 index by using the XGBoost Classifier. In order to handle and modify the data used in this system, it was used the Python Library Pandas [22].

This module receives as input the start date and the end date of the trading session, as well as the XGBoost Classifier trained in the module **Model Training Module**.

The first step of this module is to obtain the ticker symbols, which are unique series of letters assigned to a stock for trading purposes [23], for all the stocks in the S&P500 index during the chosen trading period. These tickers are extracted by filtering a list of all the companies that currently belong to the S&P500 index, and selecting the tickers that have been added to the index at least nine months prior to the start date of the chosen trading period. This ensures that the technical ratios calculated afterwards do not have missing values during the trading period. Following this is the extraction of the fundamental and the technical data of the stocks corresponding the selected tickers.

### Fundamental Data

The fundamental data of the stocks are extracted from the website Macrotrends [24] and they are necessary to calculate the fundamental ratios mentioned in Chapter 2.1.1. For each selected ticker, the corresponding data consisting of the balance sheets, cash flow statements, income statements and financial ratios is scrapped from the website. The available data is extracted from 2009-01-01, which is as far back as the website provides, up to the present date. Since not every variable from the extracted data were necessary to calculate the $\mathbf{F_{Score}}$, only the needed variables were stored in a Pandas DataFrame [25]. An example of the necessary fundamental variables of GOOGL are shown in the Table 3.1. Afterwords, the data extracted is filtered to cover the four years prior to the beginning of the trading session.

Table 3.1: Fundamental Variables of GOOGL.

| **Index** | **2019-12-31** | **2019-09-30** | **2019-06-30** | **2019-03-31** | **...** |
|---|---|---|---|---|---|
| Return On Assets | 3.8676 | 2.687 | 3.8689 | 2.7133 | **...** |
| Cash Flow Operations | 14427 | 15466 | 12627 | 12000 | **...** |
| Long Term Debt | 4554 | 4082 | 4074 | 4066 | **...** |
| Total Current Assets | 152578 | 148358 | 147437 | 138207 | **...** |
| Total Current Liabilities | 45221 | 39224 | 37000 | 34910 | **...** |
| Total Assets | 275909 | 263044 | 257101 | 245349 | **...** |
| Shares Outstanding | 14901 | 14894 | 14929 | 14949 | **...** |
| Gross Profit | 25055 | 22931 | 21648 | 20327 | **...** |
| Revenue | 46075 | 40499 | 38944 | 36339 | **...** |

In order to properly evaluate the stock, it is necessary for the filtered data to be complete. If there are any missing values, such as 0 or values that are not a number (NaN), in the data or there is less than four years of data, the corresponding stock is removed from the list of selected stocks. This ensures that every stock that is used during the training session was evaluated correctly and it does not impact the overall performance of the proposed system.

To improve the performance of the extraction of the fundamental data, each stock data was stored in a separate sheet within a ".xlsx" file, identified by its corresponding ticker symbol, allowing the use of this file to access the fundamental data of the stocks for future sessions, preventing the need to scrap the website Macrotrends excessively.

### Technical Data

The technical data of the stocks and the S&P500 index are extracted in this module by using the python package yfinance [26] to access the free database of Yahoo Finance [27]. The data extracted ranges from nine months prior to the starting date of the trading session until it's ending date, ensuring that every technical indicator does not have missing data for every data point during the trading session.

The technical data extracted comes in a time-series format which consists of the open, high, low, close and adjusted close prices for each period, along with the corresponding volume. The data can be sampled in equally spaced intervals, such as one second, one minute, one hour, one day, one week,

one month, etc. For this work, the technical data will be extracted in a daily rate. Following, is a detailed description of the extracted features:

- **Open** - The price of the asset at the start of the sampled period.
- **High** - The highest price the asset has achieved during the sampled period.
- **Low** - The lowest price the asset has achieved during the sampled period.
- **Close** - The price of the asset at the end of the sampled period.
- **Adj Close** - The price of the asset at the end of the sampled period after adjusting it because of corporate actions such as stock splits, dividends and rights offerings [28].
- **Volume** - The amount of shares traded during the sampled period.

The technical data of the S&P500 index uses these features as columns and the date of each day as the index. The technical data of the stocks have a very similar structure as the index technical data except that the features are used as sub-columns instead, grouped by the stock ticker symbol. The Table 3.2 and the Table 3.3 represent the technical data of the S&P500 index and the stocks technical data, respectively, stored in Pandas DataFrames.

Table 3.2: S&P500 Technical Data.

| Date | Open | High | Low | Close | Adj Close | Volume |
|---|---|---|---|---|---|---|
| 2019-05-20 00:00:00 | 2841.94 | 2853.86 | 2831.29 | 2840.23 | 2840.23 | 3293750000 |
| 2019-05-21 00:00:00 | 2854.02 | 2868.88 | 2854.02 | 2864.36 | 2864.36 | 3223050000 |
| 2019-05-22 00:00:00 | 2856.06 | 2865.47 | 2851.11 | 2856.27 | 2856.27 | 3194000000 |
| 2019-05-23 00:00:00 | 2836.70 | 2836.70 | 2805.49 | 2822.24 | 2822.24 | 3899320000 |
| 2019-05-24 00:00:00 | 2832.41 | 2841.36 | 2820.19 | 2826.06 | 2826.06 | 2889230000 |

Table 3.3: Stocks Technical Data.

| Date | META | | | ... | GOOGL | | |
|---|---|---|---|---|---|---|---|
| | Open | ... | Volume | ... | Open | ... | Volume |
| 2019-05-20 00:00:00 | 181.88 | ... | 10352000 | ... | 57.65 | ... | 30602000 |
| 2019-05-21 00:00:00 | 184.57 | ... | 7502800 | ... | 57.72 | ... | 20562000 |
| 2019-05-22 00:00:00 | 184.73 | ... | 9213800 | ... | 57.56 | ... | 18826000 |
| 2019-05-23 00:00:00 | 182.42 | ... | 12768800 | ... | 57.30 | ... | 25214000 |
| 2019-05-24 00:00:00 | 182.33 | ... | 8807700 | ... | 57.60 | ... | 18554000 |

After extracting the technical data, it was verified if there were any errors during the download of the data. All the stocks that raised an error or had missing data, such as 0 or values that are not a number (NaN), were removed from the list of selected stocks.

#### Fundamental Indicators

With the fundamental data already extracted and processed, the next step involves calculating the $\mathbf{F_{Score}}$ for each stock across every quarter. As previously explained in the Chapter 2.1.1, the $\mathbf{F_{Score}}$ is a fundamental scoring system used to assess the financial strength of a company based on its financial

statements. To calculate the $\mathbf{F_{Score}}$, it is used the Equation 2.1, where some of the binary signals depend not only on the financial statements of the current quarter, but the corresponding quarter of the previous year.

The binary signals that constitute the $\mathbf{F_{Score}}$, as mentioned in the Section 2.1.1, are calculated and stored in a Pandas DataFrame. Additionally, a new row is added to the DataFrame representing the sum of all binary signals, resulting in the $\mathbf{F_{Score}}$. An example of the binary signals computed from the fundamental variables of GOOGL is shown in the Table 3.4. After calculating the $\mathbf{F_{Score}}$ for every quarter, the data from the oldest year is removed. This step is necessary since the financial data from the oldest year is incomplete, making it impossible to compare with the previous year data. This ensures that the selection in the **Asset Selection Module** is not affected by incomplete or unreliable data.

Table 3.4: Sample of the binary signals and $\mathbf{F_{Score}}$ Results of GOOGL.

| **Index** | **2019-12-31** | **2019-09-30** | **2019-06-30** | **2019-03-31** | **...** |
|---|---|---|---|---|---|
| ROA | 1 | 1 | 1 | 1 | **...** |
| $\Delta$ROA | 0 | 1 | 0 | 1 | **...** |
| OCF | 1 | 1 | 1 | 1 | **...** |
| ACCRUAL | 1 | 1 | 1 | 1 | **...** |
| $\Delta$LEV | 0 | 0 | 0 | 0 | **...** |
| $\Delta$CURRAT | 0 | 0 | 0 | 0 | **...** |
| $\Delta$SHARES | 1 | 1 | 1 | 1 | **...** |
| $\Delta$GM | 0 | 0 | 0 | 1 | **...** |
| $\Delta$ATURN | 1 | 0 | 1 | 0 | **...** |
| $F_{Score}$ | 5 | 5 | 5 | 6 | **...** |

**Technical Indicators**

The next step of this module is to calculate the technical indicators, mentioned in Chapter 2.1.2, of the selected stocks and the S&P500 index. In order to have a good understanding of the stocks or the index behaviour, it is going to be used multiple Exponential Moving Averages, Hull Moving Averages, RSI and ROC ranging different periods. This ensures that the **Trading Module** will have into account short term fluctuations as well as long term trends. It is going to be used five Exponential Moving Averages and five Hull Moving Averages with periods of 5, 10, 20, 50 and 80 days. To help detect the trend of the asset, it will also be used a Double Crossover between the EMAs with periods of 20 and 50 days. There will be two RSIs with periods of 7 and 14 days, as well as a two ROCs with periods of 10 and 20 days. The final indicators are a MACD with two EMAs of 12 and 26 days, and the OBV to help understand the momentum and the trend of the price movement.

For every technical indicator, it is assigned a score for every data point according to the rules established in the Chapter 2.1.2. These indicators, along with the scores, are added to the Pandas DataFrame containing the technical data of the stock or the S&P500 index.

There is one last step for the S&P500 index data, which is the classification of the market trend using the XGBoost Classifier. The goal is to classify each data point of the S&P500 index into three categories: **Uptrend**, **Downtrend** and **Sideways**. For that to happen, it is necessary to calculate several moving averages with different periods, along with other technical indicators such as the RSI, ROC and MACD

for different periods as well, and the derivatives of maximum and minimum local peaks for different time ranges of the S&P500 index. The technical indicators used are described in more detail in the Chapter 3.3.3.

After those calculations are completed, each data point of the S&P500 index is going to iterated, and the XGBoost Classifier is going to classify each data point and store the result in a list. Since the XGBoost Classifier requires the categories to be encoded into numerical values, the categories **Downtrend**, **Sideways** and **Uptrend** are encoded as the values 0, 1 and 2, respectively. This list is going to be added to the DataFrame containing the S&P500 index technical scores as a new column labeled "Trend". The purpose for this classification is to choose the list of weights for the technical scores, optimized for the classified market trend. A sample of the resulting DataFrame is shown on the Table 3.5.

Table 3.5: Sample of DataFrame of the S&P500 index containing technical scores and classified trend.

| Date | Adj Close | EMA5 | EMA5 Score | ... | MACD | MACD Score | Trend |
|---|---|---|---|---|---|---|---|
| 2019-08-01 | 2953.56 | 2986.26 | -0.5 | ... | 7.65 | 0 | 1 |
| 2019-08-02 | 2932.05 | 2968.19 | -0.5 | ... | 10.89 | 0 | 1 |
| 2019-08-05 | 2844.74 | 2927.04 | -0.5 | ... | 18.10 | 0 | 1 |
| 2019-08-06 | 2881.77 | 2911.95 | -0.5 | ... | 19.45 | 0 | 1 |
| 2019-08-07 | 2883.98 | 2902.63 | -0.5 | ... | 19.22 | -0.5 | 1 |
| 2019-08-08 | 2938.09 | 2914.45 | 1 | ... | 14.67 | -0.5 | 1 |
| 2019-08-09 | 2918.65 | 2915.85 | 0.5 | ... | 12.30 | -0.5 | 1 |

The output of this module consist of two main components:

1. A Pandas DataFrame containing the technical indicators and their corresponding scores of the S&P500 index, along with the market trend classification done by the XGBoost Classifier.

2. A Python Dictionary [29] with two key-value pairs:

    - The first key-value pair, with the key named "fundamental", contains a list of Python Tuples [29]. Each Tuple consists of the ticker symbol of a stock and its respective DataFrame containing the $\mathbf{F_{Score}}$ calculated for every quarter.
    - Similarly, the second key-value pair, with the key named "technical", also consists of a list of Python Tuples. Each Tuple includes the ticker symbol of a stock and its corresponding DataFrame, which covers the technical data, indicators, and associated scores of the stock.

The output of this module is used in the **Asset Selection Module**, for the selection of the stocks based on the values of the $\mathbf{F_{Score}}$ of every quarter, and it is also used by the **Model Training Module**, in order to train the XGBoost Classifier and the Genetic Algorithms.

### 3.3.2 Asset Selection Module

The **Asset Selection Module**, is responsible for the selection of the companies from the S&P500 index. This selection is based on the values of the $\mathbf{F_{Score}}$, which are calculated in the **Data Processing**

**Module**. It receives as input the Python Dictionary that the **Data Processing Module** outputs, and the integer *N* representing the total amount of stocks that are going to be used in the **Trading Module**.

The first step of this module is to iterate through the list of Python Tuples, containing the DataFrames with the $\mathbf{F_{Score}}$ of the stocks. The module then stores the associated ticker symbol and the average $\mathbf{F_{Score}}$ of all quarters into a new list of Python Tuples. Using the average of the $\mathbf{F_{Score}}$ of all quarters improves the evaluation of the stocks and reduces the impact of the fluctuations of the quarters. The average score is calculated by the Equation 3.1, where *Q* represents the total number of quarters and $\mathbf{F_{Scoreq}}$ is the $\mathbf{F_{Score}}$ of the quarter *q*.

$$Score_{avg} = \frac{1}{Q} \sum_{q=1}^{Q} F_{Score}q \tag{3.1}$$

Afterwards, the list is sorted based on the average score of each stock in descending order. Following this, the list is split, selecting only the *N* stocks with the highest average scores. The module then stores the ticker symbols of the selected stocks into a new list.

The final step of this module is to iterate through the lists of Python Tuples containing the DataFrames with the $\mathbf{F_{Score}}$ and the technical scores of the stocks. During this iteration, only the stocks with corresponding ticker symbols stored in the list of selected tickers are retained. The original lists containing the DataFrames are then replaced with these filtered lists.

This ensures that the **Trading Module** will only use the *N* companies with the highest "fundamental" value.

### 3.3.3 Model Training Module

In the work of Matos and Neves [15] it is tested the use of multiple classifiers in order to accurately classify Forex data into 5 different trend categories. Although the classifier that had a better performance was the Cascade Classifier, both XGBoost Classifier and Random Forest Classifier achieved very similar results, such that the results of the full algorithm should be very close regardless of which classifier was chosen between these three mentioned. Therefore, the classifier chosen to classify the S&P500 index market trend is the XGBoost because of its accuracy, speed and easy implementation.

The **Model Training Module**, is the one responsible for training the XGBoost Classifier to accurately classify the current market trend of the S&P500 index, as well as optimizing the list of weights used in the **Trading Module** through two Genetic Algorithms.

This module receives as input the technical indicators of the S&P500 index from the **Data Processing Module** to train the XGBoost Classifier. Additionally, it receives the results from the **Trading Module** in order for the Genetic Algorithms to optimize the list of weights used in the **Trading Module**.

#### XGBoost Classifier

The objective of the XGBoost Classifier is to classify the trend of each data point of the S&P500 index into three different categories: **Uptrend**, **Downtrend** and **Sideways**. An example of each category is

shown in the Figures 3.3. In order to do that, it is necessary to provide the XGBoost Classifier with sufficient information so it can train and learn efficiently.

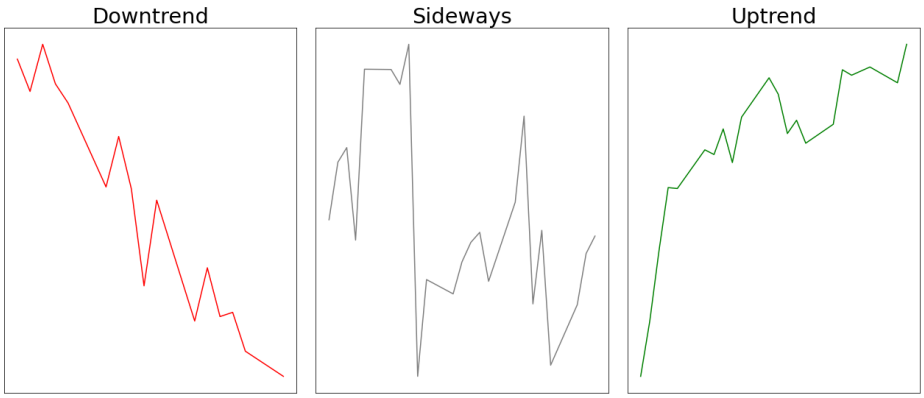


Figure 3.3: Examples of the market trend categories.

The dataset used to train the XGBoost Classifier consists of multiple technical indicators provided by the **Data Preparation Module**. These technical indicators are: SMA, EMA, RSI and ROC with periods of 7, 14, 30, 50 and 80 days, a MACD with periods of 12 and 26 days, and the derivatives of multiple lines passing through the maximum and minimum locals in periods of 7, 14, 30, 50 and 80 days of the S&P500 index. These technical indicators provide the XGBoost Classifier sufficient information for it to learn to classify the market and adjust to the short-term variations of the market, as well as the market direction on the long-term.

To avoid overfitting, it is necessary to provide a large enough dataset, so it can handle a multitude of different scenarios. For that reason, it is provided the data of the technical indicators of the S&P500 index between 1970-01-01 and 2019-12-31, as well as the manual classification of the data points for that period. This period contains multiple intervals classified as Uptrend, Downtrend and Sideways. The Figure 3.4 shows the distribution of each class in the training dataset.

As it can be seen on the Figure 3.4, the class Uptrend constitutes the majority of the training dataset, having more than the double of the frequency compared to the other classes. This is expected as the S&P500 index tends to experience Uptrend periods for much longer durations compared to Downtrend and Sideways periods. Despite this unbalanced dataset, the performance of the XGBoost Classifier is not affected since there is still a significant amount of data points classified as Downtrend and Sideways for the XGBoost Classifier to learn how to accurately predict those classes.

The provided dataset is split in two parts randomly: the first part contains 80% of the dataset and it is going to be used as the training dataset. The remaining 20% is used as the test dataset. The split is conducted randomly so the XGBoost trains and tests using unique datasets every time the training is performed.

The XGBoost Classifier is instantiated using the class “XGBClassifier” from the python package “py-xgboost” [30] with the hyper-parameters *eta*, *n_estimators*, *max_depth*, *gamma*, *min_child_weight*, *reg_lambda* and *subsample*. The description of these parameters, as well as the list of values to explore are defined as follows:

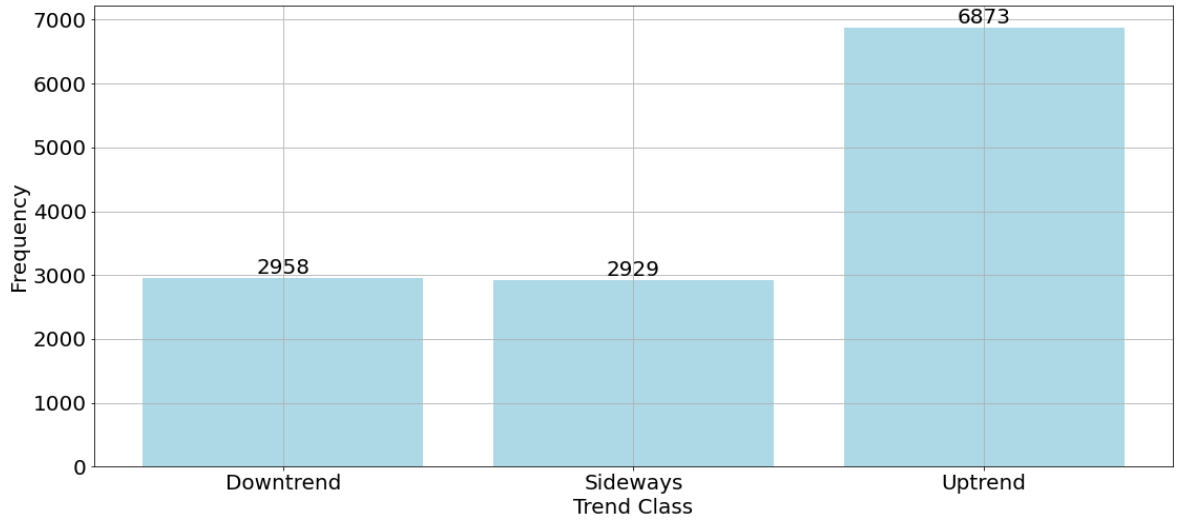


Figure 3.4: Class distribution of the training dataset.

- **eta** - The step size shrinkage used in updates to prevent overfitting. It is also known as learning rate. [0.1, 0.15, 0.2]
- **n_estimators** - The number of boosting stages. [100, 150, 200]
- **max_depth** - The maximum depth of the trees. [3, 5, 7]
- **gamma** - The minimum loss reduction necessary for the partition of the leaf node. [0, 0.5, 1]
- **min_child_weight** - The minimum hessian weight needed in a child for further partitioning. [1, 2, 3]
- **reg_lambda** - The weight of the L2 regularization. [0.5, 1, 1.5]
- **subsample** - the percentage of rows used for each tree construction. [0.5, 0.7, 0.9]

After the XGBoost is instantiated, it will train with the technical indicators of the training dataset and the manual classifications of the market trend of that dataset. Afterwords, it uses the test dataset in order to determine its accuracy, as well as the respective confusion matrix, which is a square matrix where the rows represent the actual classes and the columns represent the predicted classes and it shows the number of instances correctly predicted and the instances incorrectly predicted [31].

To determine the best hyper-parameters for the XGBoost Classifier, it is used a Cross Validation Grid Search procedure. Grid Search involves testing all different combinations of the hyper-parameters values in order to find the optimal set to maximize the model's performance. The training dataset will be split into ten smaller groups, with one serving as the testing dataset in each iteration. Within each iteration, the XGBoost Classifier is evaluated for its accuracy by testing all possible combinations of the hyper-parameters. The set of hyper-parameters with the best average accuracy will be chosen to perform the classification of the market trend of the S&P500 index. The Cross Validation Grid Search is handled by the class "GridSearchCV" from the python package "sklearn.model_selection" [32].

To improve the speed of the system, when the XGBoost finishes training, it stores its model in a JSON file using the method “save_model” from the class “XGBClassifier”. For future runs of the system, the model is loaded from the JSON file using the method “load_model” from the class “XGBClassifier”.

#### Genetic Algorithms

In order to open and close positions with the best timing possible in the **Trading Module**, it is necessary to optimize the weights of the “Divergence Scores”, which are used to calculate the “Total Divergence Score”. The optimization of the weights will be handled by two Genetic Algorithms. Genetic Algorithms are well-suited to determine optimal or near-optimal solutions in complex and large spaces. They mimic the way that natural selection and evolution occurs, allowing them to efficiently explore a large search space and converge towards optimal solutions. Each Genetic Algorithm will focus on optimizing the weights for periods classified as **Uptrend** or **Downtrend**, ensuring that the trading decisions are optimized for the market trend.

The chromosome composition of the Genetic Algorithms have a total of 17 genes, each representing the weight of the “Divergence Score” of a technical indicator calculated in the **Data Preparation Module**. The weights range between 0.001 and 0.3 to ensure the weights are not completely discarded and to not get to large in cases of overfitting. The chromosome composition is shown in the Figure 3.5, where each gene shows the technical indicator it is associated with. For better readability, technical indicators of the same type but with different periods were omitted.


Figure 3.5: Chromosome Composition.

The fitness function, used to evaluate the individuals of each generation, will use the ROI from the trading session of the positions that were opened or closed during the respective market trend of the Genetic Algorithm as parameter that needs to be optimized. The individuals with the best fitness value will pass to next generation.

The crossover operator chosen for the Genetic Algorithms is called Single Point Crossover, it consists of splitting the chromosomes of two individuals from the same generation by randomly choosing a point in chromosome and swapping the genes after that point between both individuals. The resulting individuals will become part of the next generation. An example of the Single Point Crossover is shown in the Figure 3.6.

The chosen mutation operator uses a Gaussian distribution to introduce random values to the genes. When iterating every gene from the chromosome, it is generated a random value between 0 and 1, if this value is equal or smaller than the predefined value of the Mutation Rate, a value from the Gaussian

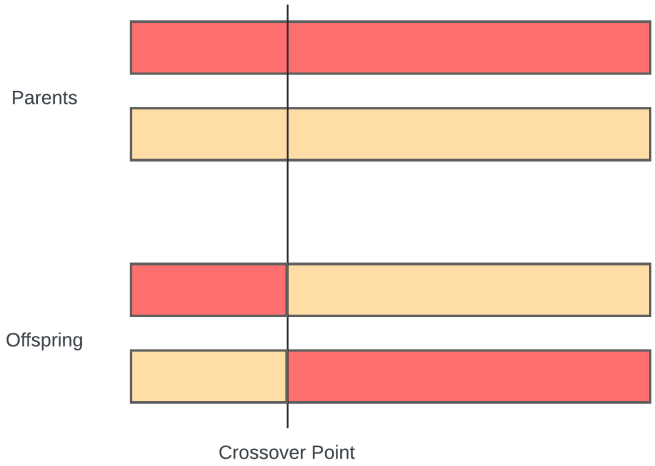


Figure 3.6: Single Point Crossover.

distribution with a mean of 0 and a standard deviation of 0.1 is added to the gene. If the resulting value of the gene is below the minimum value of 0.001, then it is set to the minimum value. This process helps preventing chromosomes to get stuck at a local maxima.

To select the parents for the next generation, the selection operator method chosen is the Tournament Selection method. This method selects a subset of the population and ranks every individual from that subset by their fitness values and selects the two individuals with the highest fitness values to be the parents for the next generation. This method has the advantage of being capable of handling individuals with a negative fitness value.

The method chosen to train the Genetic Algorithms is Repeated Cross Validation [33]. Each Genetic Algorithm receives a list of five periods, mainly consisting of **Uptrend** or **Downtrend** periods, depending on the specific Genetic Algorithm. It will use four of the five periods for training, while the fifth period is used as a test dataset to evaluate its performance. This process is repeated five times, with each iteration rotating the test dataset, ensuring that each period is used as the test dataset once. This method helps determining the performance of the Genetic Algorithms and their variations. The Figure 3.7 shows the flow of this method.

The list of periods for each Genetic Algorithm are defined as follows:

- **Downtrend** - [2015-08-12 to 2015-10-02], [2015-12-02 to 2016-02-12], [2016-09-09 to 2016-11-12], [2018-10-10 to 2019-01-04], [2019-05-08 to 2019-06-04]

- **Uptrend** - [2016-02-17 to 2016-08-16], [2016-11-08 to 2017-03-20], [2017-04-24 to 2018-02-02], [2018-04-24 to 2018-10-08], [2019-01-08 to 2019-05-07]

After the training is complete, this module returns the two lists of weights for the “Divergence Scores”, where each list is optimized for its specific market trend.

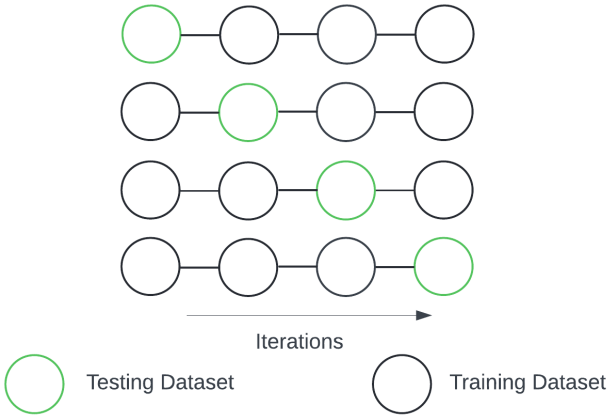


Figure 3.7: Repeated Cross Validation.

### 3.3.4 Trading Module

Finally, the **Trading Module** is the one responsible for analysing the technical indicators and their respective scores, mentioned in the Chapter 2.1.2, for the stocks selected by the **Asset Selection Module** and the S&P500 index. It then takes positions based on this analysis.

Besides the list of selected stocks, the **Trading Module** also receives as input the weights of the technical scores, optimized by the Genetic Algorithms in the **Model Training Module**, for periods classified by the XGBoost Classifier as **Uptrend** and **Downtrend**.

In the work of Gorgulho et al. [17] it was used a purely Technical Analysis of the companies with the use of a Genetic Algorithm to optimize the weights of the indicators and ratios used. This module will also use a purely Technical Analysis of the companies performance, along with the S&P500 index performance.

The first step of this module will be to iterate the list of Python Tuples containing the ticker symbol and the DataFrame containing the technical scores of every selected stock by the **Asset Selection Module**. In each iteration, the technical scores of the respective stock are compared with those of the S&P500 index using the Equation 3.2 to calculate the divergence. The stock technical score is subtracted with the respective index technical score and the result is divided by two, in order to have a normalized value between -1 and 1. These results are called “Divergence Scores” and they represent a numerical value of the divergence between the stock’s performance and the S&P500 index performance. When the Divergence Score is negative, it indicates that the respective stock technical indicator value underperforms the index technical indicator value, and when the Divergence Score is positive it indicates that the respective stock technical indicator value outperforms the index indicator technical value. These Divergence Scores are then stored in a new Pandas DataFrame, containing the divergence of the stocks technical scores and the index technical scores for each data point in the trading session, and using the date of the data points as the index.

$$\text{Divergence Score} = \frac{\text{Stock}_{score} - \text{Index}_{score}}{2} \tag{3.2}$$

With the Divergence Scores calculated, the respective DataFrame containing them is going to be iterated. At the start of the iteration, the Divergence Scores are multiplied by their respective weights, determined by the trend classification from the XGBoost Classifier. This multiplication results in a numerical value representing the total Divergence Score of that data point. These total Divergence Scores are then added to a list. This list of Divergence Scores serves as the basis for setting the thresholds for opening and closing positions. When the list has reached or surpassed the predefined size, $T$, a moving average is applied to the most recent $T$ data points. The opening and closing thresholds are established using a predefined margin percentage, $M$. For example, the top $M$% values of the moving average are set as the opening threshold for long positions and the bottom $M$% values of the moving average are set as the closing threshold for long positions as well. This approach ensures that the opening and closing thresholds are always adapted to the most recent $T$ Divergence Scores, resulting in a sliding window scheme. For this work, the value of $T$ is set to 80 trading days, and the value of $M$ is set to 1. An example of the thresholds being established is shown in the Figure 3.8. The blue line represents the moving average of the total Divergence Scores of a stock, while the green and red lines represent the opening and closing thresholds for long positions set for that 80-days period, respectively.

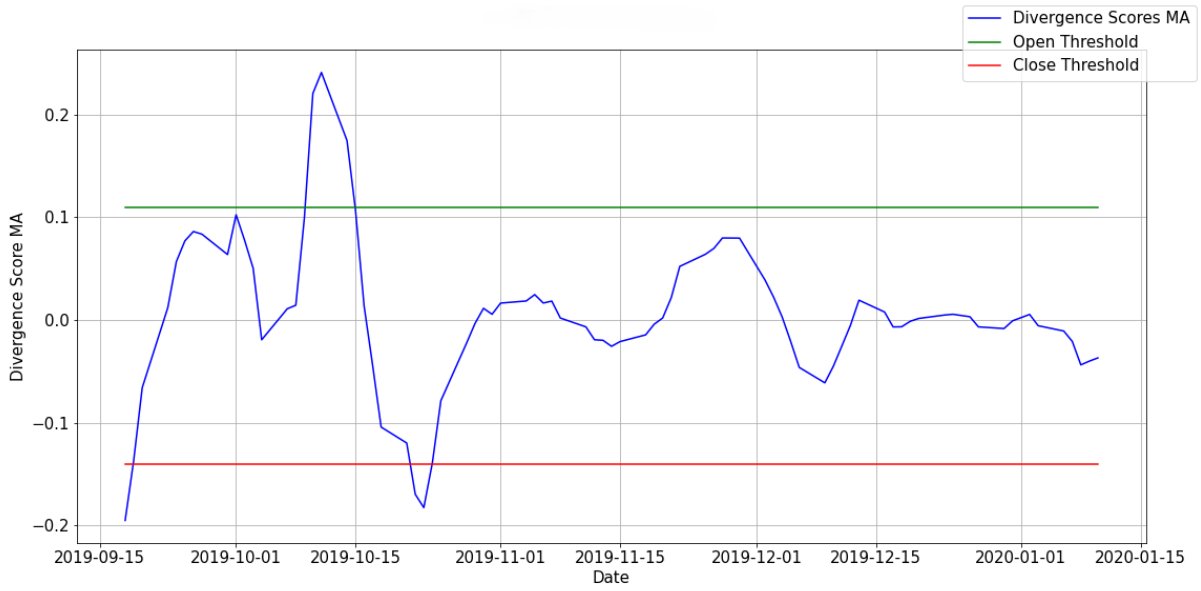


Figure 3.8: Setting the opening and closing thresholds.

To ensure that the opening and closing thresholds are already adapted to the last $T$ data points at the beginning of the trading session, the initial iterations until the starting date of the trading session only serve to calculate the total Divergence Scores. That is why the technical data of the stocks are extracted with a few months prior to start of the trading session in the **Data Processing Module**.

When the system reaches the starting date of the trading session, it is finally possible to open and close positions.

The trading logic of this module is to open or close a position when a stock starts to diverge or converge from the index. When the value of a ratio is close to the opening thresholds, the module will

have more chances of opening a short or long position since it indicates a strong divergence between the stock and the index. When the values of the scores start to approximate the closing thresholds the module will start closing the positions since the stock performance is starting to converge or it is going in the opposite direction with the index performance.

Instead of using the total Divergence Score of just one data point to open or close positions, it is going to be used a moving average, with a predefined size of *W*. This ensures that the daily fluctuations of the price do not impact the performance of the algorithm. For this work, it is used a period of 10 trading days for the moving average.

Once the moving average of the total Divergence Scores surpasses the opening threshold for long positions it is open a long position and it maintains that position until the moving average crosses below the respective closing threshold, at which the position is closed. If the moving average of the total Divergence Scores surpasses the opening threshold for short positions then it opens a short position until the moving average crosses above its closing threshold, at which the position is closed. An example of opening and closing several positions is shown in the Figure 3.9, where the green dots show when a position was bought and the red dots show when a position was sold.


Figure 3.9: Positions of the stock AVGO.

During periods classified as **Sideways**, when the market direction is highly uncertain, the option of opening or close positions is restricted. This ensures that the algorithm focus specially on periods classified as **Uptrend** and **Downtrend**, where the market direction is much more clear. Another restriction added to the trading algorithm was to only open long positions during periods classified as **Uptrend** and to only open short positions during periods classified as **Downtrend**. This restriction improves the performance of the algorithm by preventing positions from being opened against the current market trend.

When the algorithm reaches the end of the trading session for a stock, it stores the results in three

separate Pandas Series [34]. The first Series represents the total profit of the trading session for the corresponding stock, the second and third Series represent the total profit made with long and short positions during the trading session, respectively. The index for these Series is the date of the data point. After these results are stored, the algorithm repeats the process for the next stock and stores its results in three Pandas Series. After all stocks have been iterated, the Pandas Series are added, resulting in an aggregation of the entire profit of the trading session and the profit from the long and short positions taken for the entire trading session. An example of the results made by a stock are shown in the Figure 3.10 and the aggregated results are shown in the Figure 3.11.


Figure 3.10: Total profit history of the trading session using stock AVGO.

To evaluate the algorithm performance, the B&H was used as the benchmark. Each stock was iterated once again, and under the B&H strategy, it was bought once at the start of the trading session and sold at the end of the trading session.

The results of both the system algorithm and the B&H strategy were stored in a Python Dictionary. This Dictionary included the total profit earned during the trading session, the total amount invested, the total amount of positions taken and the ROI for each of these two strategies.


Figure 3.11: Total profit history of a trading session.

# Chapter 4

# Results

This chapter will present the Case Studies of the implemented system, explaining each one in detail and its results. The first part of this chapter will explain the evaluation metrics for each type of Case Study, regarding the classification of the market and the trading results. The second part will focus on the Case Studies and its results.

## 4.1 Metrics

In this section, the metrics used to evaluate the classification of the market using the XGBoost Classifier for the first Case Study will be explained in detail, as well as the metrics used to evaluate the performance of the trading results for the rest of the Case Studies.

### 4.1.1 Classification Metrics

The first Case Study will analyze the classifications of the market using the XGBoost Classifier. The metrics that will be used in order to determine its performance are *Accuracy*, *Precision*, *Recall* and *Area Under the Receiver Operating Characteristics Curve (AUC-ROC Curve)*.

A Confusion Matrix is a table displaying the combinations of the predicted and the actual values. It helps evaluate the performance of the classifier by categorizing the predicted values into four categories:

- **True Positive (TP)** - The predicted value is positive and it is correct.
- **True Negative (TN)** - The predicted value is negative and it is correct.
- **False Positive (FP)** - The predicted value is positive and it is incorrect.
- **False Negative (FN)** - The predicted value is negative and it is incorrect.

The Figure 4.1 shows the description of the Confusion Matrix.

These categories quantify the number of correct and incorrect predictions made by the classifier and help with the calculation of other performance metrics such as *Accuracy*, *Precision* and *Recall*.

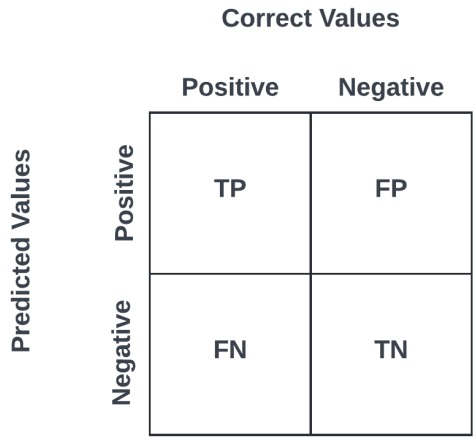


Figure 4.1: Structure of a confusion matrix.

Accuracy measures how often the classifier predicts correctly. It is defined by the ratio of the number of correct predictions and the total number of predictions. This metric provides insight of the overall performance of the classifier. The Equation 4.1 describes the formula to calculate the system accuracy.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

Precision measures how many of the predicted cases are actually positive. It is defined by the number of true positives divided by the number of predicted positives. This metric is useful for evaluating the classifier's ability to avoid FP predictions. The Equation 4.2 describes the formula to calculate the precision for each class.

$$Precision = \frac{TP}{TP + FP} \tag{4.2}$$

Recall measures how many of the positive cases were correctly predicted. It is defined by the number of the true positives divided by the total number of actual positives. This metric helps assess the classifier's ability to retrieve all positive cases. The Equation 4.3 describes the formula to calculate the recall for each class.

$$Recall = \frac{TP}{TP + FN} \tag{4.3}$$

The AUC-ROC Curve is a performance measurement for classification problems at various threshold settings [35]. The ROC is a probability curve and AUC represents a measure of separability. It provides insight of how much the model is capable of distinguishing between two classes. When the AUC is 1, the model is perfectly capable of distinguishing between positive and negative class. When the AUC is 0, the model predicts negative classes as positive and positive classes as negative. When the AUC is 0.5, the model is not capable of distinguishing between the positive and the negative class. Since the XGBoost Classifier classifies the data into three categories, it is used the "One vs All" methodology where each class is compared against the other possible classes as one. For example, the Uptrend

class is compared with Downtrend and Sideways class simultaneously.

### 4.1.2 Trading Metrics

The Case Studies regarding the results of the **Trading Module** will use performance metrics to evaluate its results, in order to obtain insight of the performance of the system algorithm. The performance metrics used are *ROI*, *Sharpe Ratio* and *Maximum Draw Down*.

The ROI is a financial metric used to evaluate the profitability of an investment. It is calculated by subtracting the current value of the investment with the initial cost of the investment and dividing that result with the initial cost of investment. Its formula is represented in the Equation 4.4.

$$ROI = \frac{Current\ Value\ of\ Investment - Initial\ Cost\ of\ Investment}{Initial\ Cost\ of\ Investment} \tag{4.4}$$

The Sharpe Ratio is a financial metric used to assess the risk-adjusted relative returns. It is calculated by subtracting the returns of the investment with a benchmark of risk-free rate of return, such as government bonds, and dividing that result with with the standard deviation of the portfolio's excess returns. Excess returns are the returns above the risk-free rate of return. The formula for the Sharpe Ratio is described in the Equation 4.5.

$$Sharpe\ Ratio = \frac{Returns\ on\ Investment - Risk\text{-}Free\ Rate\ of\ Returns}{Standard\ Deviation\ of\ the\ portfolio's\ excess\ returns} \tag{4.5}$$

The MDD is an indicator of downside risk during a specific time period. It represents the maximum observed loss from a peak to a trough of a portfolio before a new peak is attained. Its formula is shown in the Equation 4.6

$$MDD = \frac{Trough\ Value - Peak\ Value}{Peak\ Value} \tag{4.6}$$

## 4.2 Case Studies

This section will go over all the Case Studies of this work, providing a detailed description of the Case Study as well as a deep analysis of its results. The Case Study 1 will analyse the performance of the XGBoost Classifier given different training datasets with the same type of technical indicators but covering different ranges. The Case Study 2 will analyse the results of the trading algorithm by testing the performance of multiple dynamic opening and closing thresholds. The Case Study 3 will test the impact of using different moving averages in the sliding window scheme when evaluating the stocks "Total Divergence Score". The Case Study 4 will study the influence of the selection of the stocks based on the average $\mathbf{F_{Score}}$. Finally, the Case Study 5 will analyse the certainty of XGBoost classifications on the trading algorithm.

### 4.2.1 Case Study 1 - XGBoost Input Comparison Study

Before proceeding with the comparison of the results of the XGBoost Classifier, it is necessary to set the hyper-parameters of the XGBoost Classifier. As it was mentioned in the Section 3.3.3, it was used a Cross Validation Grid Search to find the optimal set of hyper-parameters in order to maximize the system's accuracy. The training dataset provided was 80% of the dataset containing the technical indicators of the S&P500 index between 1970-01-01 and 2019-12-31. The 20% remaining will be used to validate the performance of the XGBoost Classifier.

The resulting values of the hyper-parameters set by the Cross Validation Grid Search are described here:

- **eta** - 0.2
- **n_estimators** - 200
- **max_depth** - 7
- **gamma** - 0
- **min_child_weight** - 1
- **reg_lambda** - 0.5
- **subsample** - 0.9

With the hyper-parameters set, the next step is to train the XGBoost Classifier. The technical indicators that constitute the training datasets are SMA, EMA, RSI, ROC, MACD and derivatives of the maximum and minimum locals, the only difference being the variations of the ranges.

The first training dataset covers the ranges of 5, 10, 15, 20 and 30 trading days for its technical indicators. This dataset is more focused on short-term variations of the market. The second training dataset has the ranges of 20, 30, 50, 80 and 100 trading days, focusing especially on the market direction on the long-term and decreasing the impact of the short-term variations of the market. The third training dataset has a more balanced approach, focusing on the market direction on the long-term but also taking into account the short-term variations of the market. Its ranges are 10, 20, 30, 50 and 80 trading days. The last dataset has the ranges of all previous datasets to assess the impact of additional features on the XGBoost Classifier.

Each dataset was trained ten times in order to obtain an average of the performance of the XGBoost Classifier. The results of the XGBoost Classifier given the different training datasets are shown in the Table 4.1, displaying the average precision, average recall and average AUC-ROC of the Downtrend, Sideways and Uptrend classes respectively, as well as the total values for each metric.

By observing the results from the table 4.1, it is clear that the dataset focusing on the short-term variations exhibits the worst performance, with an average accuracy of 90.7%. Additionally, it also has the worst average precision with 89.4% and average recall with 88.4%. Despite this, its average AUC-ROC is only 1% lower than the other datasets. This performance is due to the lack of information to

Table 4.1: Results of the XGBoost Classifier with different training datasets.

| | **Short-Term** | **Long-Term** | **Balanced** | **Full** |
|---|---|---|---|---|
| **Avg. Precision per Class (%)** | [90.2, 84.7, 93.3] | [96.7, 93.9, 96.7] | [95.9, 92.4, 96.0] | [95.8, 90.5, 97.6] |
| **Avg. Precision (%)** | 89.4 | 95.8 | 94.8 | 95.0 |
| **Avg. Recall per Class (%)** | [90.6, 78.9, 95.8] | [96.4, 91.2, 98.0] | [95.1, 89.1, 97.7] | [95.9, 90.5, 97.5] |
| **Avg. Recall (%)** | 88.4 | 95.2 | 94.0 | 94.6 |
| **Avg. AUC-ROC per Class (%)** | [98.9, 96.6, 98.6] | [99.7, 99.2, 99.6] | [99.6, 98.7, 99.4] | [99.6, 99.1, 99.5] |
| **Avg. AUC-ROC (%)** | 98.0 | 99.5 | 99.3 | 99.4 |
| **Avg. Accuracy (%)** | 90.7 | 96.1 | 95.2 | 95.6 |

understand the direction of the market trend on the long-term, which results in an increased sensitivity of the noise of the short-term fluctuations of the market. The classification of the market with this dataset between 2020-01-01 and 2022-12-31 is shown in the Figure 4.2.

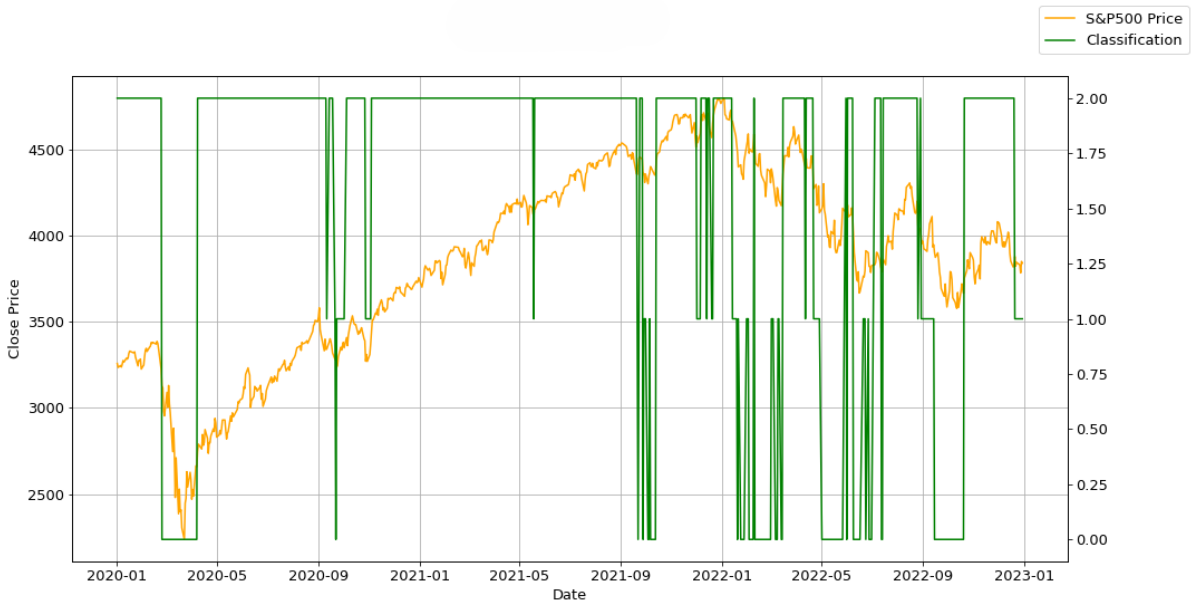


Figure 4.2: Classification of the S&P500 index with the short-term dataset.

The rest of the datasets have achieved very similar results, with the long-term dataset bringing the best performance in every metric by a small margin. Its performance comes from reduction of the impact from the fluctuations of the market and the direction of the market trends in the long-term. The classification of the market with this dataset between 2020-01-01 and 2022-12-31 is shown in the Figure 4.3.

In contrast, the balanced dataset has the third best performance with an average accuracy of 95.2%, an average recall of 94.0% and an average AUC-ROC of 99.3%. This dataset helps the XGBoost Classifier to classify the market correctly based on its long-term trends but still takes into account the

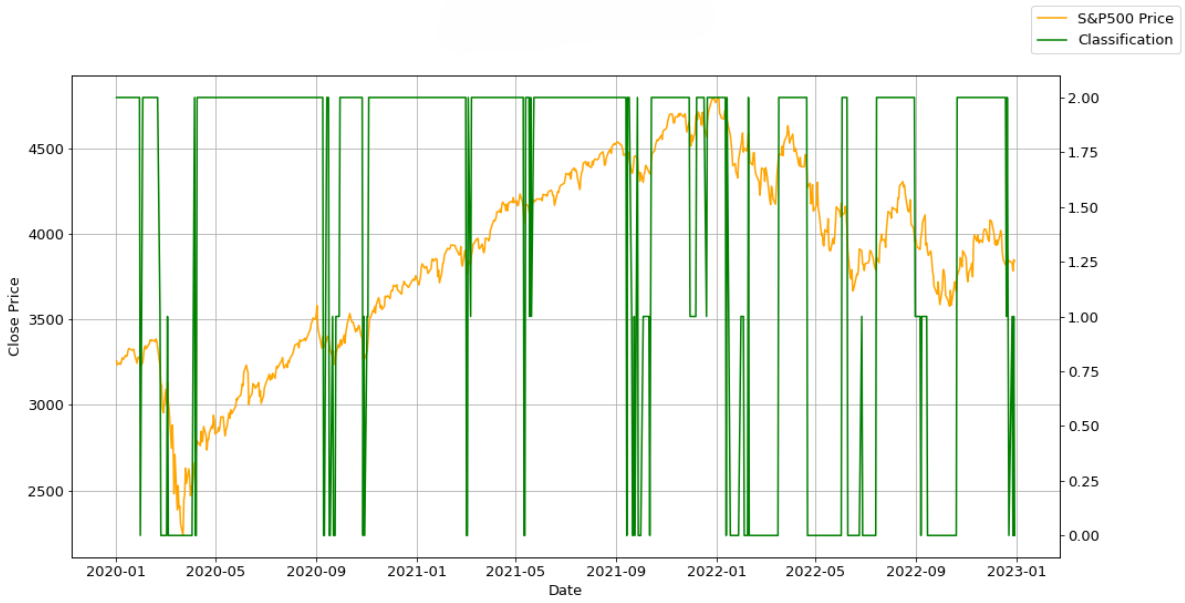


Figure 4.3: Classification of the S&P500 index with the long-term dataset.

short-term fluctuations, which help the classifier to react faster to the sudden changes but it also impacts the overall performance of the classifier because of the noise introduced to the data. The classification of the market with this dataset between 2020-01-01 and 2022-12-31 is shown in the Figure 4.4.

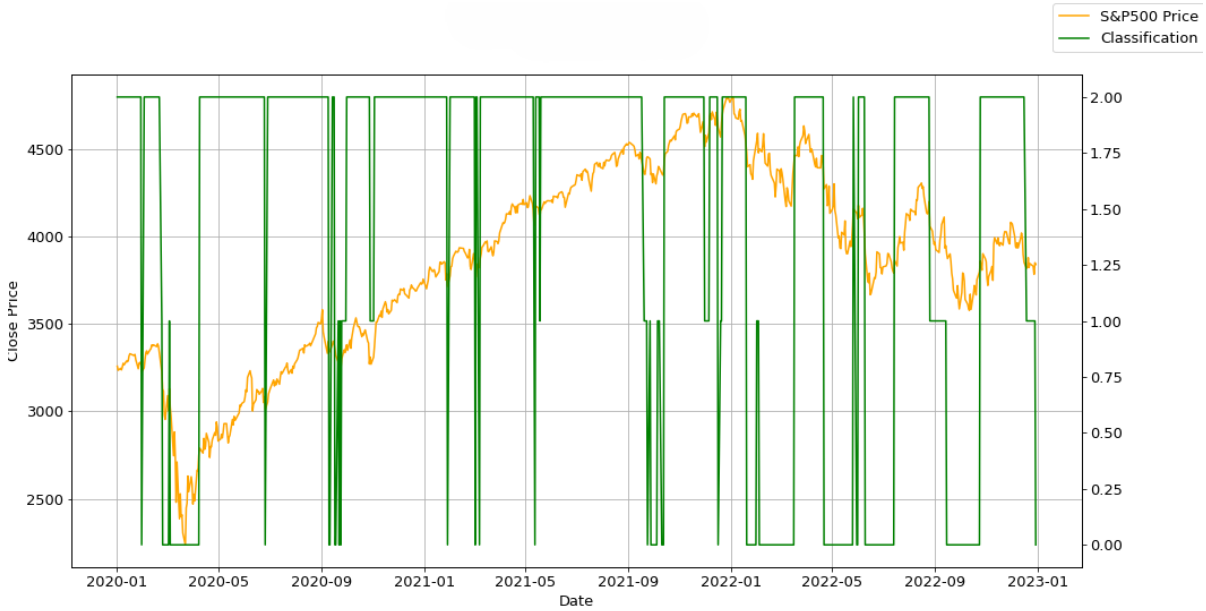


Figure 4.4: Classification of the S&P500 index with the balanced dataset.

The dataset containing all the ranges of the previous datasets has the second best performance, with an average accuracy of 95.6%. This dataset improves the performance of the XGBoost Classifier by a slight margin compared with the balanced dataset, due to the added ranges which give more information of the trend market on the long-term and the sudden changes of the market. However, it performs slightly worse compared to the long-term dataset due to the increased sensitivity to the sudden variations. Additionally, increasing the amount of technical indicators also increased the complexity of

XGBoost Classifier, which can potentially lead to overfitting. The classification of the market with this dataset between 2020-01-01 and 2022-12-31 is shown in the Figure 4.5.

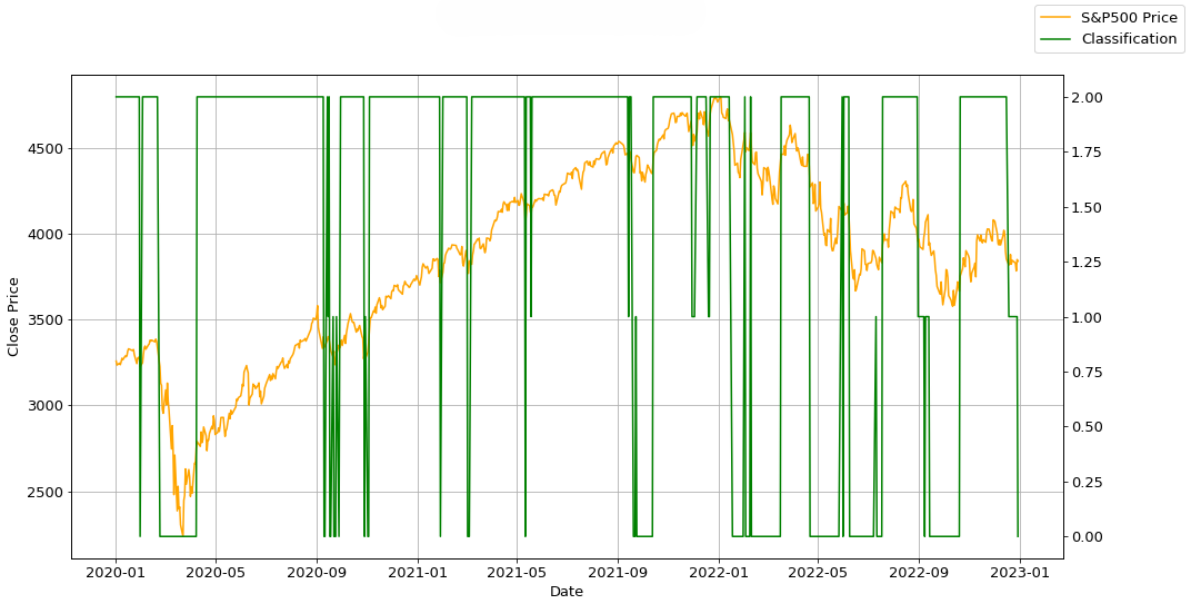


Figure 4.5: Classification of the S&P500 index with the full dataset.

Upon assessing the overall performance of the XGBoost Classifier with different training datasets, the long-term dataset has proven to be the best. It has the best performance based on the evaluation metrics and it is also observable in the Figure 4.3 that is capable of reacting to the abrupt sudden changes in the market without lagging to much compared to the other datasets. Because of this, the following Case Studies will have the long-term dataset has the chosen training dataset for the XGBoost Classifier. The confusion matrices of these training datasets are shown in the Appendix A.1

### 4.2.2 Case Study 2 - Dynamic Thresholds Comparison Study

Before proceeding with the comparison of the results between multiple dynamic thresholds, it is tested the market trend restriction described as opening only long positions during periods classified as “Uptrend” and opening short positions during periods classified as “Downtrend”. In the Figures 4.6 and 4.7, it is displayed the results of the trading session without and with the market trend restriction, respectively. The Table 4.2 also provides an overall comparison between the results of these two trading sessions, showing the number of positions and the ROI obtained for each class, as well as the Sharpe Ratio and MDD corresponding to the years 2020, 2021 and 2022 for each class.

As it can be observed in the Figures 4.6 and 4.7, the long positions typically generate the highest profit of the trading session. They tend to have a similar behaviour as the S&P500 index, only having a bad performance during periods of higher volatility or periods classified as “Downtrend”. The short positions, however, are the ones that tend to generate profits during “Downtrend” periods, while having a bad performance during the rest of the periods as it can be seen on the Figure 4.6.

By adding the market restriction, the trading results have significantly improved, with a ROI of 6.94%, almost the double of the ROI of the trading session without the restriction. This performance is due to the improvement of the shorts positions performance and reducing the amount of positions opened, since by adding the market restriction, the amount of positions opened reduced to less than half. Although the short positions do not increase the total profit of the trading session significantly, it does not suffer significant losses during periods classified as “Uptrend”, as it can be seen in the Figure 4.7. Because of these results, this restriction will be used for the rest of the Case Studies, including this one.


Figure 4.6: Trading without the market trend restriction.


Figure 4.7: Trading with the market trend restriction.

Table 4.2: Results of the trading session with and without market restriction.

| **Metrics** | **Without Restriction** | | | **With Restriction** | | |
|---|---|---|---|---|---|---|
| | **Downtrend** | **Uptrend** | **Total** | **Downtrend** | **Uptrend** | **Total** |
| **#Positions** | 63 | 148 | 211 | 16 | 80 | 96 |
| **ROI (%)** | 2.38 | 4.96 | 3.62 | 2.04 | 8.34 | 6.94 |
| **Sharpe Ratio per year** | [-0.063, -0.187, -0.013] | [-0.004, 0.08, -0.11] | [-0.017, -0.083, -0.118] | [-0.036, -0.077, 0.015] | [-0.008, 0.067, -0.144] | [-0.001, 0.054, -0.145] |
| **MDD per year (%)** | [12.10, 4.35, 1.49] | [30.47, 1.50, 2.04] | [13.52, 0.77, 0.78] | [8.59, 4.96, 4.29] | [28.88, 1.71, 1.26] | [11.54, 1.54, 0.98] |

With the market trend restriction added to the trading system, the next step is to test the performance of the trading algorithm using multiple dynamic thresholds. These thresholds are responsible for the timing of opening and closing positions during the trading session, which can drastically improve the performance of the algorithm. In order to find the best thresholds ranging between 0.01% and 0.2%, the weights used for the calculation of the Total Divergence Scores were optimized for different dynamic thresholds, using the method Repeated Cross Validation, explained in detail in the Section 3.3.3. During the optimization of the weights, it was selected 20 stocks with the highest $\mathbf{F_{Score}}$ from all the stocks that were in the S&P500 index at the beginning of each training period and are still present in the S&P500 index. The list of optimized weights is shown in the Appendix A.2. After the weights were optimized, it was tested the performance of the algorithm using each dynamic threshold during the period 2020-01-01 and 2022-12-31. The results of the trading sessions using the multiple dynamic thresholds are shown in the Table 4.3, with the number of positions and ROI obtained for each class, as well as the Sharpe Ratio and MDD for the years 2020, 2021 and 2022 for each class.

The results from the Table 4.3 show that the trading session using the opening and closing margins of 1% have yielded the best returns during Uptrend periods, with a ROI of 6.02%, and the overall second

Table 4.3: Results of the trading session with multiple dynamic thresholds.

| Thresholds (%) | Periods | #Positions | ROI (%) | Sharpe Ratio per year | MDD per year (%) |
|---|---|---|---|---|---|
| 1 | **Downtrend** | 17 | -6.9 | [-0.069, 0, -0.049] | [32.88, 0, 9.54] |
| | **Uptrend** | 71 | 6.02 | [-0.004, 0.1, -0.181] | [19.33, 1.87, 3.01] |
| | **Total** | 88 | 2.77 | [-0.046, 0.097 -0.232] | [19.25, 1.76, 3.63] |
| 3 | **Downtrend** | 23 | -3.04 | [-0.125, 0, -0.032] | [41.68, 0, 5.1] |
| | **Uptrend** | 91 | 5.39 | [-0.015, 0.084, -0.196] | [29.17, 1.2, 1.69] |
| | **Total** | 114 | 3.46 | [-0.034, 0.076, -0.237] | [28.61, 1.12, 1.14] |
| 5 | **Downtrend** | 26 | -2.13 | [-0.062, -0.307, -0.03] | [33.7, 4.55, 4.61] |
| | **Uptrend** | 104 | 4.16 | [-0.014, 0.064, -0.244] | [25.46, 1.42, 1.89] |
| | **Total** | 130 | 2.59 | [-0.041, 0.036, -0.244] | [23.34, 1.23, 1.49] |
| 10 | **Downtrend** | 35 | -2.07 | [-0.035, -0.174, -0.047] | [14.17, 6.07, 3.54] |
| | **Uptrend** | 119 | 3.48 | [-0.012, 0.029, -0.235] | [22.02, 1.19, 1.79] |
| | **Total** | 154 | 2.08 | [-0.035, -0.017, -0.264] | [13.6, 1.26, 1.23] |
| 20 | **Downtrend** | 62 | -1.78 | [-0.044, -0.316, -0.066] | [13.19, 2.69, 2.16] |
| | **Uptrend** | 170 | 2.56 | [-0.015, -0.015, -0.278] | [17.74, 0.8, 1.41] |
| | **Total** | 232 | 1.34 | [-0.041, -0.08, -0.358] | [10.31, 0.65, 0.97] |

best returns, only falling short to the results using a 3% margin. These results, as expected, have the fewest amount of positions due to its tight margins, only opening and closing positions once the Total Divergence Score reaches the top or bottom 1% of the Total Divergence Scores in the last 80 trading days. Because of these tight margins, these thresholds impact the performance of the algorithm during Downtrend periods, having the worst ROI of -6.9% compared to the other results during these periods, because of missing the opportunities to close the positions, making the positions remaining open for longer, leaving them open for longer and reducing potential profits from short positions as the market recovers.

The results using the 3% margin have the best overall returns, with a ROI of 3.46%. The amount of positions have increased significantly and the negative impact of the positions taken during Downtrend periods was reduced by lesser than half compared to the previous result. This is due to its bigger margins allowing to close the short positions taken during the Downtrend periods and stop the losses before the market recovers more. However, the returns during Uptrend periods have decreased significantly compared to the previous results due to closing its positions earlier than before, missing the chance to increase its returns by a little more.

The results with 5%, 10% and 20% margins had a similar development as the results using 3% margins. As the margins increased, the number of positions increased significantly and the losses during Downtrend periods kept decreasing since it was able to close its short positions sooner and reducing its losses, however, the profits generated during the Uptrend periods were reduced as well due to its long positions closing too soon. On the other hand, as the margins increased, the volatility of the profits in the trading sessions were reduced, as it can be seen from the values of the MDD decreasing, with the results using the 20% margins having the lowest MDD each year of the trading session. In the Figure 4.8 it is shown a direct comparison of the results between each trading session using the different thresholds as well as the S&P500 index.

The results shown in the Figure 4.8 support the observations made previously. With the increase of


Figure 4.8: Results of the trading sessions with multiple dynamic thresholds.

the opening and closing margins, the volatility of the returns were reduced but also reducing the overall profit of the trading session. During the sharp decline in the first quarter of 2020, the results with tighter margins were negatively impacted due to not being able to close the positions in time in order to minimize losses. The results with the 1% margin were not affected as much as the other results during the first quarter of 2020 because of the small amount of positions opened during that period. However, after the market decline, the algorithm was not able to open long positions quickly enough to take advantage of the rise in the market during the first year of the trading session.

In conclusion, by increasing the opening and closing margins of the thresholds, the algorithm suffers fewer losses during Downtrend periods while also decreasing the returns during Uptrend periods, which tend to reduce the overall returns of the algorithm during the trading session. Because of that, the 3% margin for the opening and closing thresholds was chosen for the next Case Studies due to having the best overall returns, with the returns during Uptrend Periods being only slightly inferior compared to the results with the 1% margin and having less than half of the losses during the Downtrend periods.

### 4.2.3 Case Study 3 - Sliding Window Type Comparison Study

For this Case Study, it is tested the use of three different moving averages for the calculation of Total Divergence Score of a stock using the sliding window scheme. The values of the Total Divergence Score also influence the thresholds for opening and closing positions, thereby significantly impacting the performance of the algorithm. The moving averages tested were SMA, EMA and HMA and in order for the results to be compared, the weights used for the calculation of the Total Divergence Scores were optimized again for each moving average by using the method Repeated Cross Validation, explained in the Section 3.3.3. For the optimization of the weights, the process was repeated like in the previous Case Study, it was selected 20 stocks with the highest $\mathbf{F_{Score}}$ from all the stocks that were in the S&P500

index at the beginning of each training period and are still present in the S&P500 index. The list of optimized weights is shown in the Appendix A.2.

After optimizing the weights for each moving average, it was tested the performance of each moving average during the period from 2020-01-01 to 2022-12-31. The results of the trading sessions using the SMA, EMA and HMA for the calculation of the Total Divergence Scores are shown in the Figures 4.9, 4.10 and 4.11, respectively. Each figure shows the total profit of the trading session, as well as the profit generated by long and short positions, compared to the S&P500 index. The Table 4.4 provides more information of the results, such as, the number of positions opened, the ROI, the Sharpe ratio and the MDD for the years 2020, 2021 and 2022 of the trading session, for both Downtrend and Uptrend periods, as well as the results for the total trading session for each moving average.

Table 4.4: Results of the trading session using a SMA, EMA and HMA for the calculation of the Divergence Scores.

| Moving Average | Periods | #Positions | ROI (%) | Sharpe Ratio per year | MDD per year (%) |
|---|---|---|---|---|---|
| **SMA** | **Downtrend** | 22 | -6.6 | [-0.096, -0.136, -0.077] | [30.78, 5.80, 4.95] |
| | **Uptrend** | 78 | 8.492 | [-0.011, 0.099, -0.143] | [21.29, 1.60, 2.39] |
| | **Total** | 100 | 4.161 | [-0.076, 0.079, -0.214] | [19.96, 1.46, 1.84] |
| **EMA** | **Downtrend** | 21 | 0.846 | [-0.066, -0.154, 0.039] | [29.96, 2.53, 4.80] |
| | **Uptrend** | 91 | 4.886 | [-0.013, 0.067, -0.19] | [24.1, 1.35, 1.5] |
| | **Total** | 112 | 3.908 | [-0.037, 0.061, -0.194] | [20.7, 1.28, 1.15] |
| **HMA** | **Downtrend** | 16 | 2.044 | [-0.036, -0.077, 0.015] | [8.6, 4.96, 4.3] |
| | **Uptrend** | 80 | 8.342 | [-0.008, 0.067, -0.145] | [28.8, 1.71, 1.26] |
| | **Total** | 96 | 6.945 | [-0.001, 0.054, -0.145] | [11.54, 1.54, 0.98] |

Using the SMA for the calculation of the Total Divergence Scores has resulted in the second best total returns, with a ROI of 4.161% and 100 positions opened. However, it achieved the best returns during Uptrend periods with a ROI of 8.492%, compared to the other moving averages. Its performance during Downtrend periods is what negatively affects the overall results, with a ROI of -6.6% and the worst MDD values in the first and third years, which have significant periods classified as Downtrend, compared to the other moving averages.

The performance during Downtrend periods is affected by the slow reaction of the SMA to sudden changes in the market. Since the Downtrend periods tend to be shorter and abrupt compared to Uptrend periods, the slow reaction of the SMA to sudden changes can cause the algorithm to miss opportunities for opening or closing short positions, or opening and closing the positions too late, resulting in smaller profits or even losses. Contrarily, the Uptrend periods tend to last longer compared to other periods and exhibit slower price changes, making the slow reaction of the SMA advantageous during these periods since it is less affected by sudden changes and, even if it is slower to open a long position, it remains with that position during a longer time, which usually generates more profit during these periods, compared to the other moving averages.

The EMA has resulted in the worst total returns and the highest number of positions opened, with a ROI of 3.908% and 112 positions opened. It has the worst ROI during Uptrend periods compared to the other moving averages but it has the second best returns during Downtrend periods, surpassing the SMA returns during these periods. Although the EMA reaction to sudden changes in the market is faster

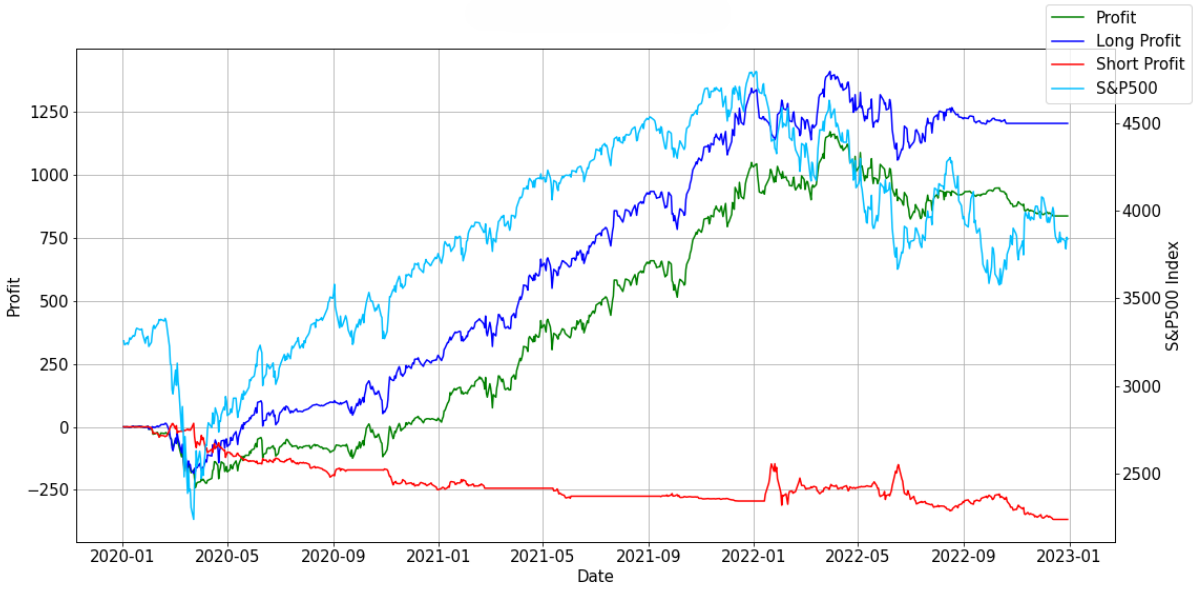


Figure 4.9: Trading results using a SMA for the sliding window.

than the SMA, it is still affected by those drastic changes, as it can be seen in the Figure 4.10 during first quarter of 2020. The algorithm does not react fast enough to open short positions during that fall, although its losses are smaller compared to the SMA, and its long positions are greatly affected by it, impacting significantly the total performance. During the last year of the trading session, the algorithm does not suffer significant losses, even with the abrupt changes, since it is fast enough to open and close short positions during the small Downtrend periods, and to close the long positions before suffering any significant losses.


Figure 4.10: Trading results using a EMA for the sliding window.

The HMA has the best total returns, as well as the returns during Downtrend periods, with a ROI of 6.945% and 2.044%, respectively. Its profit is mostly due to the profit of the long positions, with the short positions only improving slightly the total returns but still performing better compared to the other moving averages. Its performance is due to its faster reaction to the sudden changes in the market, which reduced significantly the impact during the first quarter of 2020. Because of that, the HMA has the best Sharpe Ratio and MDD for the first year, compared to the other moving averages. During the second year of the trading session, its Sharpe Ratio is equal to the EMA's Sharpe Ratio and lower than the SMA's Sharpe Ratio, as well as having the worst MDD by a small margin compared to the other moving averages. And by the third and final year of the trading session, the HMA has the highest Sharpe Ratio as well as the lowest MDD, due to the less volatility of its performance.


Figure 4.11: Trading results using a HMA for the sliding window.

In the Figure 4.12 it is shown a direct comparison of the total profit of the trading sessions using each moving average. It is clear that the HMA outperforms the other moving averages for most of the trading session, resulting in a less volatile performance with higher returns. The EMA and the SMA have achieved very similar returns but the SMA has outperformed the EMA for most of the trading session due to the significant impact of the drastic fall during the first quarter of 2020.

In the Figure 4.13 it is shown the comparison of the Total Divergence Scores calculated using each moving average, as well as the S&P500 index. The graph a) shows a direct comparison of the Total Divergence Scores calculated by using a SMA, EMA and HMA, while the other graphs show more clearly the Total Divergence Score calculated by one of the moving averages while the others are less saturated to help visualize the differences between the use of these three moving averages. In the graph b), it is clear that the SMA reacts much slower to the sudden changes, resulting in Total Divergence Scores much less volatile compared to the other moving averages. In the graph c), the EMA has a significant improvement on its reaction to the sudden changes compared to the SMA. Finally, the HMA has the

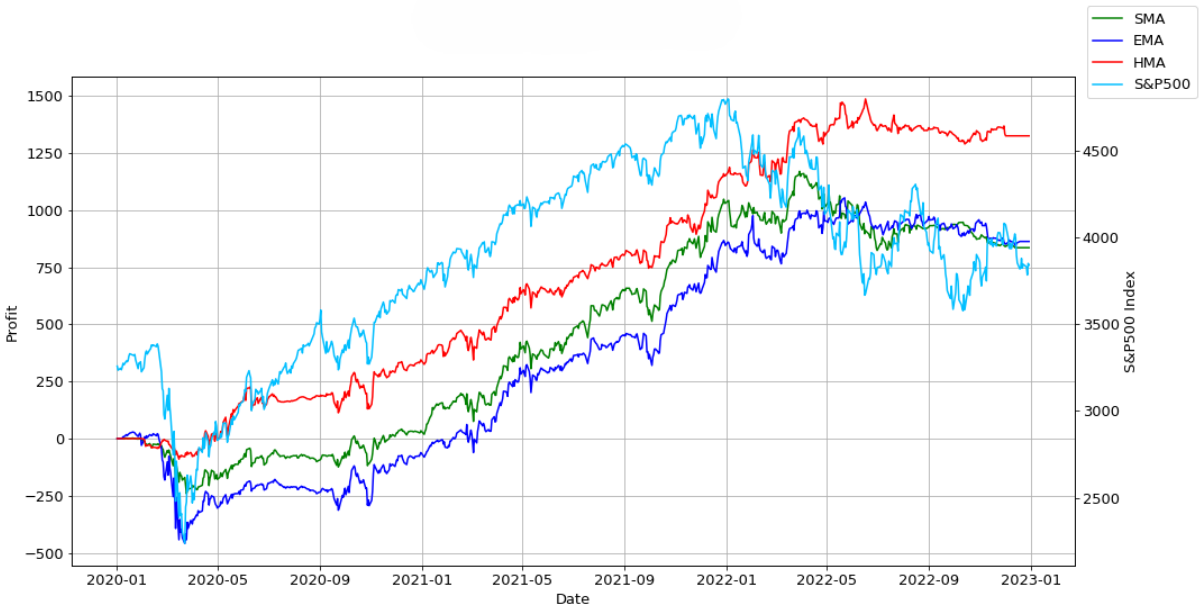


Figure 4.12: Comparison of the trading results of the SMA, EMA and HMA.

fastest reaction compared to the other moving averages, as well as the highest absolute values of the Total Divergence Scores, even achieving values of 0.4, which were never achieved by using the other moving averages.

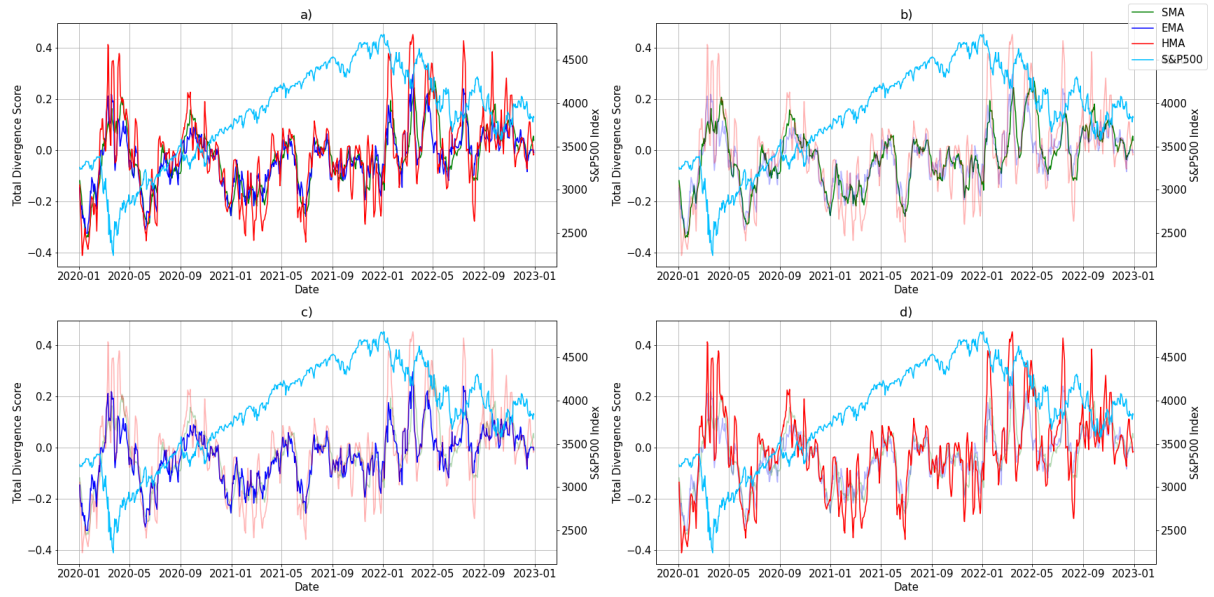


Figure 4.13: a) Comparison of the Total Divergence Scores using the SMA, EMA and HMA; b) Total Divergence Scores SMA; c) Total Divergence Scores EMA; d) Total Divergence Scores HMA;

In conclusion, the HMA has proved to achieve the best returns for the algorithm, surpassing the returns from the other moving averages, due to its fast reaction to the sudden changes in the market. Because of this, the HMA is going to be used to calculate the Total Divergence Scores for the next Case Studies.

### 4.2.4 Case Study 4 - Assets Selection Comparison Study

For this Case Study, it was examined the impact of the stocks selection based on the average $\mathbf{F_{Score}}$ of each stock, on the results of the trading session. The $\mathbf{F_{Score}}$, explained in detail in the Section 2.1.1, is a fundamental scoring system ranging from 0 to 9 that assesses the financial strength of a company based on its annual financial statements. The stocks used during the trading session are selected based on the average of the $\mathbf{F_{Score}}$ of every quarter, three years before the trading session. The selected stocks are the ones with the highest average $\mathbf{F_{Score}}$.

To determine if this selection is the most optimal for the performance of the algorithm, the trading session results were compared using stocks with the average $\mathbf{F_{Score}}$ ranging in following intervals: [3 to 4], [4 to 5], [5 to 6], [6 to 7] and [8 to 9]. The Figure 4.14 shows the distribution of all the stocks that were in the S&P500 index at the beginning of the trading session and are still in the S&P500 index at the ending of the trading session, in the different intervals based on the average $\mathbf{F_{Score}}$ of the stock.

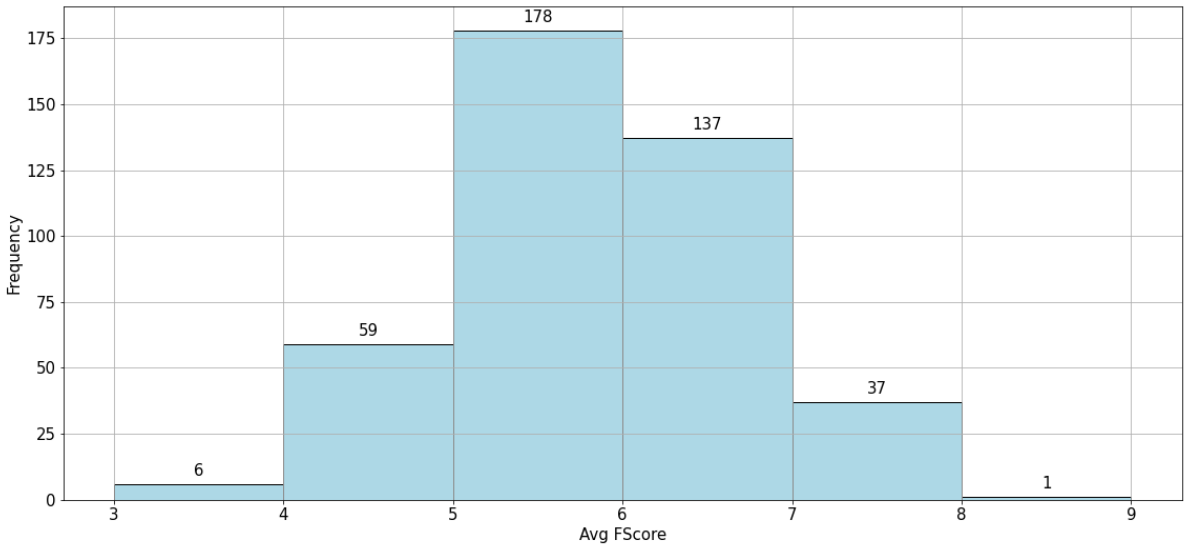


Figure 4.14: Distribution of the average $\mathbf{F_{Score}}$ of the stocks.

It is going to be selected twenty random stocks from each interval and it is going to be compared the results of each trading session in order to determine the impact of the selected stocks for the trading session results. Since there are only six stocks in the interval [3 to 4], that interval will be joined with the interval [4 to 5], resulting in the interval [3 to 5]. Similarly, since the interval [8 to 9] only has one stock, the intervals [7 to 8] and [8 to 9] will be joined resulting in the interval [7 to 9]. Each interval will be tested ten times in order to obtain an average of the performance for each interval. The average of the results of the trading sessions for the stocks from each interval are shown in the Table 4.5, where it is displayed the average of the amount of positions opened, the average ROI, the average Sharpe Ratio per year, the average Maximum Draw Down per year of the trading session, as well as the ROI of the entire trading session using the B&H strategy. It was also selected the trading session with the highest ROI of each interval in order to have a direct comparison between the best performance of each interval.

The stocks from the interval [3 to 5] have the worst returns during the entire trading session, with an

Table 4.5: Average of the results of the trading sessions with stocks from different $\mathbf{F_{Score}}$ intervals.

| Intervals | Periods | #Positions | ROI (%) | B&H ROI (%) | Sharpe Ratio per year | MDD per year (%) |
|---|---|---|---|---|---|---|
| **[3 to 5]** | **Downtrend** | 16 | -1.83 | - | [-0.036, -0.155, -0.069] | [12.36, 3.24, 6.11] |
| | **Uptrend** | 94 | 1.51 | - | [-0.018, 0.003, -0.231] | [36.76, 1.78, 2.30] |
| | **Total** | 110 | 0.87 | 8.29 | [-0.026, -0.009, -0.277] | [31.77, 1.61, 2.13] |
| **[5 to 6]** | **Downtrend** | 19 | -2.76 | - | [-0.049, -0.146, -0.028] | [19.68, 2.87, 6.63] |
| | **Uptrend** | 95 | 2.25 | - | [-0.017, -0.006, -0.187] | [39.98, 2.42, 2.44] |
| | **Total** | 114 | 1.40 | 35.57 | [-0.026, -0.021, -0.209] | [32.29, 2.07, 2.03] |
| **[6 to 7]** | **Downtrend** | 15 | -1.31 | - | [-0.017, -0.150, -0.068] | [9.87, 4.03, 4.26] |
| | **Uptrend** | 93 | 3.92 | - | [-0.007, 0.0217, -0.172] | [37.4, 2.22, 2.94] |
| | **Total** | 108 | 2.92 | 40.94 | [-0.004, 0.002, -0.219] | [31.54, 1.54, 0.98] |
| **[7 to 9]** | **Downtrend** | 19 | -3.29 | - | [-0.025, -0.193, -0.070] | [10.00, 4.16, 6.98] |
| | **Uptrend** | 94 | 1.99 | - | [-0.016, 0.038, -0.163] | [38.94, 1.66, 3.24] |
| | **Total** | 113 | 1.11 | 52.13 | [-0.025 0.026, -0.186] | [30.3, 1.53, 2.94] |

average ROI of 0.87%. The stocks from this interval also have the lowest average ROI using the B&H strategy, at 8.29%. The poor performance of these stocks is evident during Uptrend periods, as they have the worst average ROI during Uptrend periods compared to the stocks from the other intervals. Because of this poor performance, positions opened with these stocks tend to generate very small returns during the trading session, as well as suffering significant losses during Downtrend periods, although they are not affected as much as the stocks from other intervals such as [5 to 6] and [7 to 9] during these periods.

The stocks from the interval [5 to 6] have a better performance compared to the intervals [3 to 5] and [7 to 9], with an average ROI of 1.4% during the entire trading session, as well as -2.76% and 2.25% for Downtrend and Uptrend periods. These stocks tend to generate significant profits but they are more volatile compared to the stocks from the other intervals. This increased volatility is reflected in their having the worst Sharpe Ratios and MDDs for the first two years of the trading session.

The stocks from the interval [6 to 7] have the best overall returns compared to the stocks from the other intervals, with an average ROI of 2.92% during the entire trading session and -1.31% and 3.92% for Downtrend and Uptrend periods, respectively. In contrast to the stocks from the interval [5 to 6], the stocks from the interval [6 to 7] exhibit much less volatility, even during periods of significant market fluctuations, such as the last year of the trading session. They have an average MDD of 0.98% during the last year of the trading session, outperforming the stocks from the other intervals.

The stocks from the interval [7 to 9] surprisingly have the second worst performance, with an average ROI of 1.11%. Since these stocks tend to have a high “fundamental” value, they have a good perfor-

mance during Uptrend periods but end up suffering significant losses during Downtrend periods, having the worst performance during Downtrend periods compared to the stocks from the other periods, with an average ROI of -3.29%.

The Figure 4.15 shows the best results of the stocks from each interval compared to the S&P500 index.

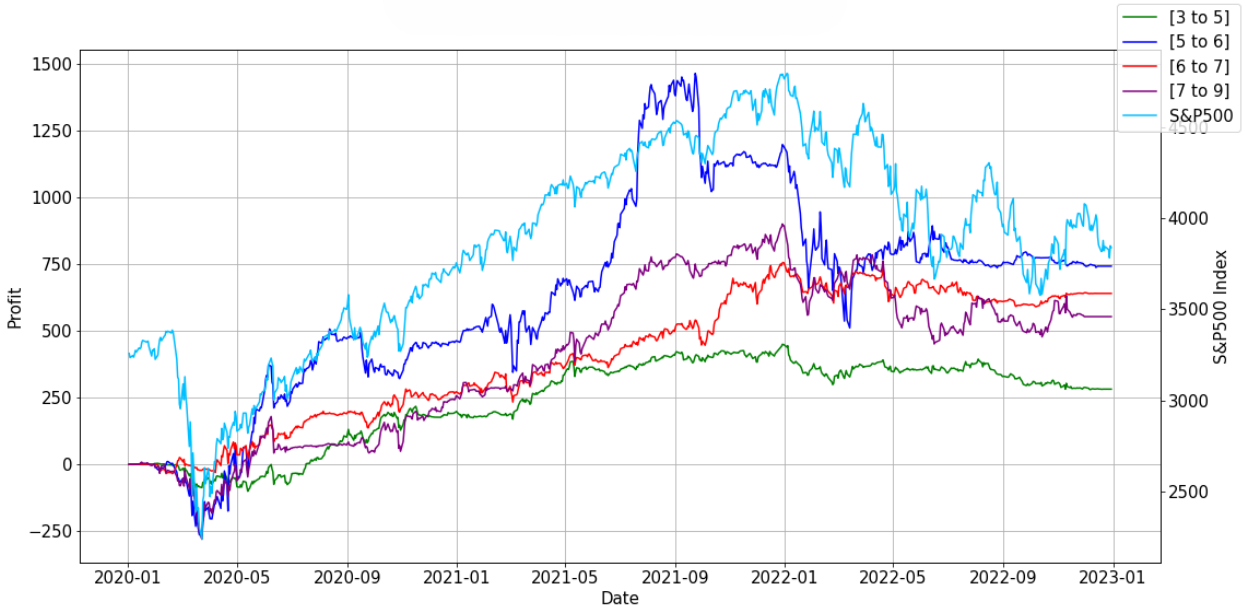


Figure 4.15: Best results of the stocks from each $\mathbf{F_{Score}}$ interval.

The results displayed in the Figure 4.15 reflect the observations made from the Table 4.5. The stocks from the interval [3 to 5] have the lowest returns, generating small profits even during Uptrend periods but suffering smaller losses during Downtrend periods compared to the other stocks. The stocks from the interval [5 to 6] have generated the most profit but are also the most volatile, suffering significant losses during Downtrend periods but also generating the most profit during Uptrend periods. The stocks from the interval [6 to 7] have the second best returns and they have the best performance during Downtrend periods by suffering very small losses compared to the rest of the stocks. Finally, the stocks from the interval [7 to 9] tend to suffer significant losses during Downtrend periods but are capable of generating good returns during Uptrend periods, only falling short to the returns made by the stocks from the interval [5 to 6] in those periods, while being much less volatile.

### 4.2.5 Case Study 5 - XGBoost Certainty Thresholds Study

For this Case Study, it was studied the impact of adding a threshold for opening and closing positions based on the certainty of the XGBoost Classifier. The XGBoost Classifier has a significant influence on the performance of the algorithm since the weights chosen to calculate the Total Divergence Score depend on the market trend classification provided by the XGBoost Classifier. During periods of increased volatility, the performance of the XGBoost Classifier can be affected, decreasing the probability

of accurate predictions and generating incorrect predictions. In the Figure 4.16 it is shown the the trend classification of the S&P500 index provided by the XGBoost Classifier and the probability of the certainty of each prediction.

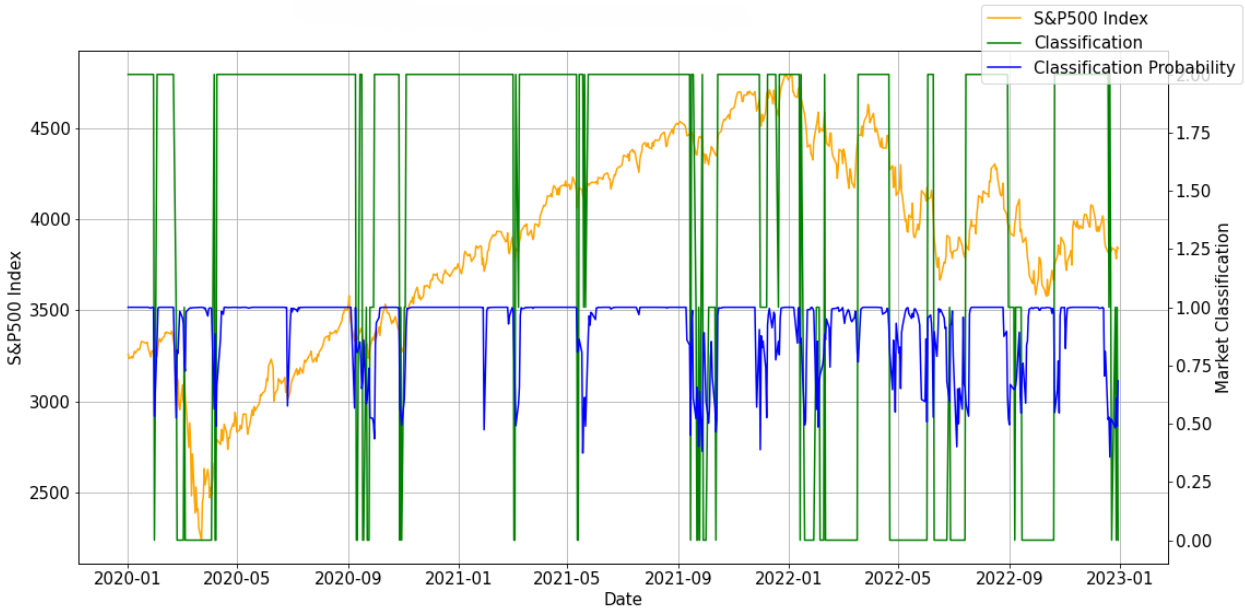


Figure 4.16: Trend Classification of the S&P500 index and the probability of the predictions.

The Figure 4.16 shows that the probability of the classifications sit close to 100% for most of the trading session, only displaying certain levels of uncertainty during periods of increased volatility and before changing its current prediction. During these periods of uncertainty, the XGBoost Classifier can generate wrong predictions, which results in using the incorrect weights for the calculation of the Total Divergence Scores and can cause the opening or closing of positions, affecting the overall performance of the algorithm. To reduce these situations, it was added thresholds for the opening and closing of positions based on the probability of the predictions made by the XGBoost Classifier. In order for a position to be opened or closed, it is necessary for the probability of the prediction of the market trend to be equal or superior to the respective threshold. The ranges of the thresholds are [0%, 50%, 70%, 85%, 90%, 95%] and to test the impact of these thresholds on the performance of the algorithm, it was tested all possible combinations for the values of the opening and closing thresholds. All tests were made using the same twenty stocks, with $\mathbf{F}_{\mathbf{Scores}}$ ranging from the interval [5 to 7], during the period 2020-01-01 to 2022-12-31. The results of these tests are displayed in the Table 4.6, with the number of positions and ROI of each test, as well as the Sharpe Ratio and MDD for the years 2020, 2021 and 2022 of the trading session.

By looking at the results from the Table 4.6, it is clear that the opening threshold has a more significant impact on the performance of the algorithm compared to the closing threshold. When changing the closing threshold while keeping the opening threshold static, the ROI of the trading session is minimally affected, changing at most 0.4%. The amount of positions opened, as well as the Sharpe Ratio and the Maximum Draw Down per year also show negligible variations. The closing threshold that provides

the best returns overall is at 85%, as it achieves the best returns for every opening threshold except for the opening threshold set at 85%, only being surpassed by 0.04%. This performance is due to setting the threshold significantly high in order to avoid closing positions during periods of elevated uncertainty, while also not setting the threshold too high to avoid missing the chances of closing the positions with optimal timing.

The opening threshold has a very significant impact on the performance of the algorithm, with it being able to increase the ROI of the trading session from a 6.85% to a 10.77%. The returns of the algorithm increase with the opening threshold, achieving the best ROI with an opening threshold set at 95% and a closing threshold set at 85%. As the opening threshold increases, the amount of positions opened decreases, reducing the amount of positions from 96 to 61, as expected. The volatility of the algorithm also increases with the reduction of the positions opened as it can be seen from the MDD per year. The first year of the trading session has a MDD of 28.88% for the opening thresholds equal or higher than 70%, while for the opening thresholds equal to 0% and 50% only have 11.5%. During the second year of the trading session, the MDD also tends to increase with the opening threshold, ranging from 1.55%, with an opening threshold set at 0%, to 2.13% with an opening threshold set at 95%. This increased volatility is due to reduction of the amount of positions, reducing the diversity of the trading session. The performance of the opening threshold is due to securing the opening of the positions during periods of less uncertainty, achieving its best returns with a very high margin. In the Figure 4.17 it is shown the best results for each opening threshold with the corresponding closing threshold that maximizes its returns.


Figure 4.17: Comparison of the best results between the opening thresholds.

The results from the Figure 4.17 show that the trading sessions with different opening thresholds have very similar behaviour, having almost the same profit at the end of the trading session. The trading sessions with the opening thresholds set at 0% and 50% have achieved the highest total profit compared to the other trading sessions due to them having a significant increase on the amount of positions.

Table 4.6: Results of the trading sessions using the opening and closing thresholds based on the probability of predictions.

| Opening Threshold (%) | Closing Threshold (%) | #Positions | ROI (%) | Sharpe Ratio per year | MDD per year (%) |
|---|---|---|---|---|---|
| 95 | 95 | 61 | 10.48 | [-0.002, 0.082, -0.147] | [28.88, 2.13, 1.16] |
| | 90 | 61 | 10.76 | [-0.002, 0.088, -0.147] | [28.88, 2.13, 1.15] |
| | 85 | 61 | 10.77 | [-0.002, 0.088, -0.147] | [28.88, 2.13, 1.15] |
| | 70 | 61 | 10.72 | [-0.002, 0.087, -0.153] | [28.88, 2.13, 1.16] |
| | 50 | 61 | 10.65 | [-0.002, 0.087, -0.169] | [28.88, 2.13, 1.16] |
| | 0 | 61 | 10.65 | [-0.002, 0.087, -0.169] | [28.88, 2.13, 1.16] |
| 90 | 95 | 64 | 9.46 | [-0.004, 0.071, -0.134] | [28.88, 2.2, 1.13] |
| | 90 | 64 | 9.73 | [-0.004, 0.076, -0.135] | [28.88, 2.2, 1.12] |
| | 85 | 64 | 9.83 | [-0.004, 0.079, -0.135] | [28.88, 2.2, 1.12] |
| | 70 | 64 | 9.79 | [-0.004, 0.078, -0.141] | [28.88, 2.2, 1.12] |
| | 50 | 64 | 9.82 | [-0.004, 0.08, -0.161] | [28.88, 2.2, 1.12] |
| | 0 | 64 | 9.82 | [-0.004, 0.08, -0.161] | [28.88, 2.2, 1.12] |
| 85 | 95 | 79 | 7.66 | [-0.006, 0.06, -0.13] | [28.88, 1.88, 1.28] |
| | 90 | 79 | 7.98 | [-0.005, 0.066, -0.131] | [28.88, 1.87, 1.28] |
| | 85 | 79 | 7.94 | [-0.006, 0.068, -0.131] | [28.88, 1.88, 1.28] |
| | 70 | 79 | 7.81 | [-0.006, 0.067, -0.137] | [28.88, 1.88, 1.33] |
| | 50 | 80 | 7.75 | [-0.006, 0.07, -0.134] | [28.88, 1.88, 1.04] |
| | 0 | 80 | 7.78 | [-0.006, 0.07, -0.133] | [28.88, 1.88, 1.01] |
| 70 | 95 | 83 | 7.18 | [-0.006, 0.053, -0.136] | [28.88, 1.74, 1.23] |
| | 90 | 83 | 7.37 | [-0.006, 0.059, -0.136] | [28.88, 1.74, 1.23] |
| | 85 | 83 | 7.54 | [-0.006, 0.061, -0.136] | [28.88, 1.73, 1.23] |
| | 70 | 83 | 7.42 | [-0.006, 0.06, -0.142] | [28.88, 1.73, 1.28] |
| | 50 | 84 | 7.44 | [-0.006, 0.062, -0.138] | [28.88, 1.73, 0.99] |
| | 0 | 84 | 7.47 | [-0.006, 0.062, -0.137] | [28.88, 1.73, 0.97] |
| 50 | 95 | 92 | 6.96 | [-0.001, 0.055, -0.152] | [11.38, 1.52, 0.89] |
| | 90 | 92 | 7.02 | [-0.005, 0.061, -0.152] | [11.54, 1.52, 0.89] |
| | 85 | 92 | 7.27 | [-0.001, 0.063, -0.152] | [11.54, 1.52, 0.89] |
| | 70 | 92 | 7.16 | [-0.001, 0.062, -0.156] | [11.54, 1.52, 0.9] |
| | 50 | 93 | 7.18 | [-0.001, 0.065, -0.144] | [11.54, 1.52, 1.04] |
| | 0 | 93 | 7.21 | [-0.001, 0.065, -0.143] | [11.54, 1.52, 1.02] |
| 0 | 95 | 95 | 6.59 | [-0.004, 0.045, -0.153] | [11.38, 1.55, 0.86] |
| | 90 | 95 | 6.84 | [-0.002, 0.051, -0.154] | [11.54, 1.55, 0.86] |
| | 85 | 95 | 6.89 | [-0.004, 0.053, -0.154] | [11.54, 1.55, 0.86] |
| | 70 | 95 | 6.79 | [-0.004, 0.052, -0.158] | [11.54, 1.55, 0.88] |
| | 50 | 96 | 6.82 | [-0.004, 0.054, -0.145] | [11.54, 1.55, 1.01] |
| | 0 | 96 | 6.85 | [-0.004, 0.054, -0.144] | [11.54, 1.55, 0.99] |

# Chapter 5

# Conclusion

In this work, it is proposed a divergence-based trading strategy where the stocks selected using the fundamental scoring system $F_{Score}$ are traded based on the divergence between their performance and the performance of the S&P500 index. This divergence is calculated by subtracting the scores of each technical indicator with the respective scores of the technical indicators of the S&P500 index. The resulting differences are then summed, each multiplied by their respective weights, which were optimized using Genetic Algorithms. Additionally, The list of weights for the resulting scores is selected based on the prediction of the current market trend provided by a XGBoost Classifier.

Regarding the selection of the stocks, it was concluded that the stocks that generated the best returns were not the ones with the highest average $F_{Score}$ but the ones with the scores ranging between 5 and 7. This performance was due to these stocks having a good fundamental value but still generating some divergences between their performances and the performance of the S&P500, resulting in those stocks being the most suited candidates for this divergence-based strategy.

The impact of the classifications made by the XGBoost Classifier on the trading algorithm were also thoroughly tested, as well as the influence of the training dataset provided. By adding a minimum prediction probability restriction for the opening and closing of the positions, the returns of the trading algorithm were significantly improved. This was due to the reduction of positions opened during periods of higher uncertainty, resulting in a better timing and certainty of the opening of the positions.

Still regarding to the classification of the market trend, the tests performed using the XGBoost Classifier with different training datasets showed that the use of long-term focused dataset yields better results due to not being affected by sudden changes in the market and with minimal lag compared to the other training datasets.

It was also tested the influence of the moving average selected to calculate the Total Divergence Score in the sliding window scheme and it was concluded that the use of a HMA leads to the best returns. This is due to the sensitivity of the HMA to the most recent changes in the price, making it easier to detect sudden changes between the divergence of the performances and generate more extreme values for the Total Divergence Score, leading to more distinct thresholds compared to the other moving averages.

The opening and closing thresholds were tested multiple times with different margins in order to understand the best thresholds range to increase the returns of the trading algorithm. The results showed that a 3% margin led to the best overall returns, especially during Uptrend periods, due to restricting the opening of the positions to only the highest moments of divergence between the performances but not high enough to miss the optimal timings to close the positions.

In conclusion, the returns achieved by this work were not ground breaking and did not surpass the B&H strategy but it did achieved interesting results, laying the foundation for future research and development.

## 5.1 Future Work

The dynamic nature of financial markets requires constant improvement and adaptation of trading algorithms in order to maintain their competitive edge. Although the proposed trading algorithm has achieved interesting results, there are several paths for further research and to enhance its performance and robustness. These paths might be:

- Adding volatility indicators, like Chicago Board Options Exchange (CBOE) Volatility Index (VIX), to the trading algorithm in order to stop opening positions during periods of high volatility.
- Using a multi-objective Genetic Algorithm in order to optimize the returns while also reducing the risk.
- Exploring the impact in the returns of the weights of the selected stocks in the S&P500 index.
- Exploring different technical indicators.
- Testing the algorithm with different parameters, such as the size of the sliding window and the size of the moving average used in the sliding window.
- Adapting the XGBoost Classifier to the volatility of the market, decreasing the errors made in periods of higher volatility.

# Bibliography


[1] E. F. Fama. Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, 25(2):383–417, 1970. ISSN 00221082, 15406261.

[2] J. Chen. Stock Analysis: Different Methods for Evaluating Stocks. *Investopedia*, 3 2023. URL https://www.investopedia.com/terms/s/stock-analysis.asp. [Accessed 27-06-2024].

[3] B. Beers. Why Do Investors Use the Samp;P 500 As a Benchmark? *Investopedia*, 9 2022. URL https://www.investopedia.com/ask/answers/041315/what-are-pros-and-cons-using-sp-500-benchmark.asp. [Accessed 27-06-2024].

[4] Investopedia. What Are the Pros and Cons of Using the SP 500 as a Benchmark? Investopedia, 2015. URL https://www.investopedia.com/ask/answers/041315/what-are-pros-and-cons-using-sp-500-benchmark.asp. [Accessed 27-06-2024].

[5] C. Mitchell. Trend: Definition, Types, Examples, and Uses in Trading — investopedia.com. *Investopedia*. URL https://www.investopedia.com/terms/t/trend.asp. [Accessed 27-06-2024].

[6] J. Chen. Sideways Trend: Definition, How Traders Profit, and Example — investopedia.com. *Investopedia*. URL https://www.investopedia.com/terms/s/sidewaystrend.asp. [Accessed 27-06-2024].

[7] J. D. Piotroski. Value investing: The use of historical financial statement information to separate winners from losers. *Journal of Accounting Research*, pages 1–41, 2000.

[8] J. Murphy. *Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications*. New York Institute of Finance Series. Penguin Publishing Group, 1999. ISBN 9780735200661.

[9] M. Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

[10] J. H. Friedman. Greedy Function Approximation: A Gradient Boosting Machine on JSTOR — jstor.org. 2001.

[11] C. G. Tianqi Chen. XGBoost: A Scalable Tree Boosting System — arxiv.org. 2016.

[12] R. L. de Almeida and R. F. Neves. Stock market prediction and portfolio composition using a hybrid approach combined with self-adaptive evolutionary algorithm. *Expert Systems with Applications*, 204:117478, 2022.

[13] A. Silva, R. Neves, and N. Horta. A hybrid approach to portfolio composition based on fundamental and technical indicators. *Expert Systems with Applications*, 42(4):2036–2048, 2015.

[14] J. Nobre and R. F. Neves. Combining principal component analysis, discrete wavelet transform and xgboost to trade in the financial markets. *Expert Systems with Applications*, 125:181–194, 2019.

[15] R. N. Diogo Matos. Multi-class trading strategies optimized through multi-objective genetic algorithms for the forex market. 2022. URL https://fenix.tecnico.ulisboa.pt/downloadFile/2533669927387141/86981-Diogo-Matos-Thesis.pdf.

[16] A. Canelas, R. Neves, and N. Horta. A sax-ga approach to evolve investment strategies on financial markets based on pattern discovery techniques. *Expert Systems with Applications*, 40(5):1579–1590, 2013.

[17] A. Gorgulho, R. Neves, and N. Horta. Applying a ga kernel on optimizing technical analysis rules for stock picking and portfolio composition. *Expert Systems with Applications*, 38(11):14072–14085, 2011.

[18] B. J. de Almeida, R. F. Neves, and N. Horta. Combining support vector machine with genetic algorithms to optimize investments in forex markets with high leverage. *Applied Soft Computing*, 64:596–613, 2018.

[19] python.org. https://www.python.org/, . [Accessed 27-06-2024].

[20] Anaconda Navigator 2014; Anaconda documentation — docs.anaconda.com. https://docs.anaconda.com/free/navigator/index.html. [Accessed 27-06-2024].

[21] S. Team. Home 2014; Spyder IDE — spyder-ide.org. https://www.spyder-ide.org/. [Accessed 27-06-2024].

[22] pandas - Python Data Analysis Library — pandas.pydata.org. https://pandas.pydata.org/, . [Accessed 27-06-2024].

[23] A. Hayes. Stock Symbol (Ticker Symbol): Abbreviation for a Company's Stock. *Investopedia*, 2024. [Accessed 27-06-2024].

[24] The long term perspective on markets. https://www.macrotrends.net/, 2024.

[25] pandas.DataFrame &x2014; pandas 2.2.2 documentation — pandas.pydata.org. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html, . [Accessed 27-06-2024].

[26] R. Aroussi. yfinance. GitHub, 2021. URL https://github.com/ranaroussi/yfinance.

[27] Yahoo finance screener. https://finance.yahoo.com/screener/. [Accessed 27-06-2024].

[28] A. Ganti. Adjusted Closing Price: How It Works, Types, Pros & Cons. *Investopedia*. [Accessed 27-06-2024].

[29] 5. Data Structures — docs.python.org. https://docs.python.org/3/tutorial/datastructures.html, . [Accessed 27-06-2024].

[30] XGBoost Python Package &x2014; xgboost 2.0.3 documentation — xgboost.readthedocs.io. https://xgboost.readthedocs.io/en/stable/python/index.html, . [Accessed 27-06-2024].

[31] Bharathi. Latest Guide on Confusion Matrix for Multi-Class Classification — analyticsvidhya.com. https://www.analyticsvidhya.com/blog/2021/06/confusion-matrix-for-multi-class-classification/. [Accessed 27-06-2024].

[32] sklearn.model_selection.GridSearchCV — scikit-learn.org. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html. [Accessed 27-06-2024].

[33] D. Kepplinger, P. Filzmoser, and K. Varmuza. Variable selection with genetic algorithms using repeated cross-validation of pls regression models as fitness measure. *arXiv preprint arXiv:1711.06695*, 2017.

[34] pandas.Series 2014; pandas 2.2.2 documentation — pandas.pydata.org. https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.Series.html, . [Accessed 27-06-2024].

[35] A. P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997. ISSN 0031-3203. doi: https://doi.org/10.1016/S0031-3203(96)00142-2.

# Appendix A

# Appendix A

## A.1 Confusion Matrices

Table A.1: Confusion Matrix of Short-Term Dataset

| **Actual Classes** | **Predicted Classes** | | |
|---|---|---|---|
| | **Downtrend** | **Sideways** | **Uptrend** |
| Downtrend | 258 | 20 | 7 |
| Sideways | 21 | 234 | 41 |
| Uptrend | 9 | 22 | 663 |

Table A.2: Confusion Matrix of Long-Term Dataset

| **Actual Classes** | **Predicted Classes** | | |
|---|---|---|---|
| | **Downtrend** | **Sideways** | **Uptrend** |
| Downtrend | 279 | 6 | 4 |
| Sideways | 6 | 262 | 19 |
| Uptrend | 3 | 10 | 675 |

Table A.3: Confusion Matrix of Balanced Dataset

| **Actual Classes** | **Predicted Classes** | | |
|---|---|---|---|
| | **Downtrend** | **Sideways** | **Uptrend** |
| Downtrend | 275 | 9 | 6 |
| Sideways | 8 | 252 | 23 |
| Uptrend | 3 | 12 | 680 |

Table A.4: Confusion Matrix of Full Dataset

| **Actual Classes** | **Predicted Classes** | | |
|---|---|---|---|
| | **Downtrend** | **Sideways** | **Uptrend** |
| Downtrend | 273 | 8 | 4 |
| Sideways | 8 | 255 | 19 |
| Uptrend | 34 | 12 | 682 |

## A.2 List of Weights

Table A.5: List of optimized weights for Case Study 2

| **Technical Indicators** | **1%** | | **3%** | | **5%** | | **10%** | | **20%** | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Down. | Up. | Down. | Up. | Down. | Up. | Down. | Up. | Down. | Up. |
| EMA5 | 0.103 | 0.001 | 0.139 | 0.152 | 0.016 | 0.001 | 0.12 | 0.001 | 0.068 | 0.001 |
| EMA10 | 0.001 | 0.001 | 0.045 | 0.001 | 0.001 | 0.001 | 0.047 | 0.006 | 0.024 | 0.001 |
| EMA20 | 0.001 | 0.083 | 0.030 | 0.001 | 0.105 | 0.034 | 0.037 | 0.001 | 0.001 | 0.001 |
| EMA50 | 0.001 | 0.128 | 0.001 | 0.127 | 0.085 | 0.257 | 0.027 | 0.278 | 0.001 | 0.116 |
| EMA80 | 0.008 | 0.029 | 0.017 | 0.245 | 0.001 | 0.001 | 0.099 | 0.001 | 0.070 | 0.130 |
| DC | 0.006 | 0.074 | 0.145 | 0.001 | 0.001 | 0.063 | 0.158 | 0.166 | 0.038 | 0.001 |
| HMA5 | 0.162 | 0.085 | 0.241 | 0.071 | 0.284 | 0.085 | 0.004 | 0.004 | 0.001 | 0.105 |
| HMA10 | 0.040 | 0.014 | 0.001 | 0.099 | 0.001 | 0.149 | 0.001 | 0.051 | 0.001 | 0.030 |
| HMA20 | 0.132 | 0.001 | 0.016 | 0.001 | 0.001 | 0.001 | 0.044 | 0.001 | 0.029 | 0.002 |
| HMA50 | 0.020 | 0.108 | 0.001 | 0.001 | 0.076 | 0.001 | 0.001 | 0.001 | 0.030 | 0.161 |
| HMA80 | 0.076 | 0.001 | 0.265 | 0.001 | 0.244 | 0.084 | 0.297 | 0.116 | 0.244 | 0.300 |
| ROC10 | 0.204 | 0.001 | 0.029 | 0.040 | 0.001 | 0.001 | 0.113 | 0.040 | 0.010 | 0.001 |
| ROC20 | 0.001 | 0.174 | 0.001 | 0.011 | 0.002 | 0.032 | 0.029 | 0.116 | 0.232 | 0.002 |
| RSI7 | 0.113 | 0.054 | 0.001 | 0.174 | 0.001 | 0.005 | 0.052 | 0.001 | 0.075 | 0.143 |
| RSI14 | 0.094 | 0.148 | 0.001 | 0.001 | 0.001 | 0.147 | 0.026 | 0.169 | 0.094 | 0.001 |
| MACD | 0.001 | 0.045 | 0.001 | 0.073 | 0.017 | 0.106 | 0.014 | 0.044 | 0.001 | 0.001 |
| OBV | 0.035 | 0.052 | 0.064 | 0.001 | 0.161 | 0.031 | 0.035 | 0.001 | 0.078 | 0.001 |

Table A.6: List of optimized weights for Case Study 3

| **Technical Indicators** | **SMA** | | **EMA** | | **HMA** | |
|---|---|---|---|---|---|---|
| | Down. | Up. | Down. | Up. | Down. | Up. |
| EMA5 | 0.001 | 0.097 | 0.064 | 0.035 | 0.001 | 0.001 |
| EMA10 | 0.001 | 0.001 | 0.001 | 0.088 | 0.044 | 0.001 |
| EMA20 | 0.001 | 0.005 | 0.057 | 0.065 | 0.055 | 0.073 |
| EMA50 | 0.109 | 0.092 | 0.001 | 0.096 | 0.197 | 0.108 |
| EMA80 | 0.164 | 0.088 | 0.001 | 0.034 | 0.001 | 0.205 |
| HMA5 | 0.001 | 0.001 | 0.180 | 0.051 | 0.082 | 0.001 |
| HMA10 | 0.001 | 0.016 | 0.143 | 0.049 | 0.001 | 0.014 |
| HMA20 | 0.001 | 0.083 | 0.160 | 0.025 | 0.001 | 0.079 |
| HMA50 | 0.001 | 0.102 | 0.001 | 0.054 | 0.117 | 0.001 |
| HMA80 | 0.239 | 0.075 | 0.031 | 0.106 | 0.294 | 0.187 |
| ROC10 | 0.001 | 0.044 | 0.032 | 0.057 | 0.001 | 0.001 |
| ROC20 | 0.001 | 0.028 | 0.001 | 0.079 | 0.021 | 0.020 |
| RSI7 | 0.148 | 0.082 | 0.011 | 0.015 | 0.038 | 0.062 |
| RSI14 | 0.214 | 0.070 | 0.137 | 0.151 | 0.001 | 0.109 |
| MACD | 0.001 | 0.069 | 0.001 | 0.001 | 0.032 | 0.125 |
| OBV | 0.001 | 0.019 | 0.139 | 0.019 | 0.109 | 0.001 |
| DC | 0.115 | 0.125 | 0.037 | 0.072 | 0.001 | 0.013 |