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

André Abreu dos Santos
andre.a.dos.santos@tecnico.ulisboa.pt
Instituto Superior Técnico, Lisboa, Portugal

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 algorithm 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 algorithm was tested in the 3 year period from 2020 to 2022 and it achieved a return of 10.77%.

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

I. 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 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 [1]. 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.

A. 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 algorithm 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.

B. Objectives

The objectives of this thesis are to:

- Analyze the results of a divergence-based trading algorithm.
- 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.

C. Main 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.
- Selection of the stocks used for trading based on the average F_{Score} of the previous three years of the trading session.
- 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".

II. Related Work

This section reviews and analyzes he scientific works gathered during the research and development for this thesis.

A. Fundamental Analysis

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 [2], 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. The work of Almeida and Neves [2] showed great results, outperforming the S&P500 index. They 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.

B. Technical Analysis

Focusing more on Technical Analysis, Gorgulho et al. [3] 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. [4] 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.

In Canelas et al. [5], 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.

C. Market Classification

Market classification can improve a system performance by focusing on specific market conditions instead of having just one model. In Matos and Neves [6], 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.

III. Architecture

In this section it is presented the architecture of the proposed system, as well as a general description of the entire architecture, including the order and objectives of each module.

A. System Architecture

The architecture of the proposed system is showed in the Figure 1, it consists of four modules: Data Processing Module, Asset Selection Module, Model Training Module and Trading Module.

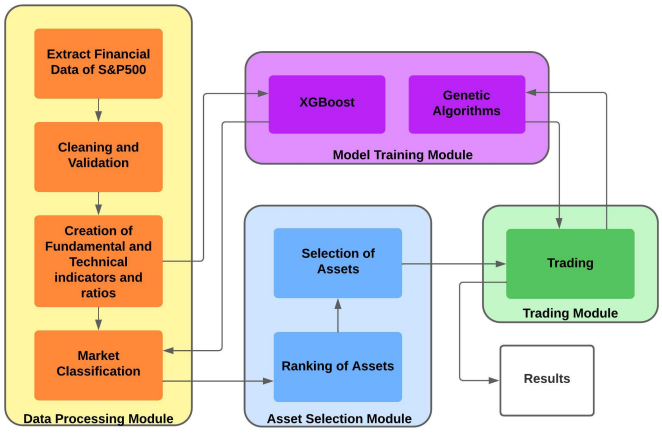

Fig. 1: Architecture of the proposed system.

B. 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, and to classify the S&P500 index by using the XGBoost Classifier.

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 [7], 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.

The fundamental data of the stocks are extracted from the website Macrotrends [8]. 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. Afterwords, the data extracted is filtered to cover the four years prior to the beginning of the trading session.

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.

The technical 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.

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.

With the fundamental data already extracted and processed, the next step involves calculating the $\mathbf{F_{Score}}$ for each stock across every quarter. As mentioned in Almeida and Neves [2], the $\mathbf{F_{Score}}$ is a fundamental scoring system used to assess the financial strength of a company based on its financial statements. The $\mathbf{F_{Score}}$ consists of 9 binary signals, 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. Those binary signals are:

- **Return on Assets (ROA)** - ROA measures a company's profitability and efficiency in generating profits from its assets.
- **Change in Returns on Assets (ΔROA)** - ΔROA indicates the difference between the current value of ROA and the value from the previous year.
- **Operating Cash Flow (OCF)** - OCF measures the cash generated or used by a company's operations during a specific period.
- **Accruals** - Accruals are accounting entries that reflect the revenue and expenses of a company during a specific time period.
- **Leverage (ΔLEV)** - Leverage is used to assess the level of debt or financial leverage employed by a company.
- **Current Ratio (ΔCURRATIO)** - 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.
- **Number of Shares (ΔSHARES)** -The number of shares is used to know if a company has issued new shares since the previous year.
- **Gross Margin (ΔGMARGIN)** - It measures the percentage of revenue that remains after deducting the cost of goods sold (COGS).
- **Asset Turnover (ΔATURN)** - The Asset Turnover measures the company's efficiency in generating sales or revenue relative to its total assets.

The binary signals that constitute the $\mathbf{F_{Score}}$ 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 $\mathbf{F_{Score}}$. 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.

The next step of this module is to calculate the technical indicators, 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, Relative Strength Index (RSI) and Rate of Change (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 Moving Average Convergence Divergence (MACD) with two EMAs of 12 and 26 days, and the On Balance Volume (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 appointed for its technical indicator. 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. 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.

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. 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.

C. 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 1, where Q represents the total number of quarters and $\mathbf{F_{Scoreq}}$ is the $\mathbf{F_{Score}}$ of the quarter q.

$$\text{Score}_{avg} = \frac{1}{Q}\sum_{q=1}^{Q} F_{Score}q \tag{1}$$

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 first *N* stocks. 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.

D. Model Training Module

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 Preparation 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**.

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

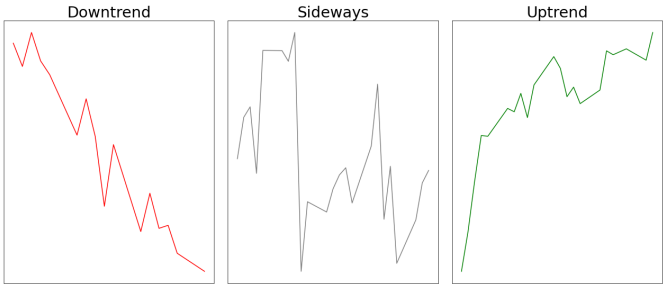

Fig. 2: Examples of the market trend categories.

The dataset used to train the XGBoost Classifier consists on 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 was 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 XGBoost Classifier is instantiated using the class "XGBClassifier" from the python package "py-xgboost" [9] 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:

- **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 [10].

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.

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, 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.

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

Fig. 3: Chromosome Composition.

to be optimized. The individuals with the best fitness value will pass to next generation.

Finally, the **Trading Module** is the one responsible for analysing the technical indicators and their respective scores, mentioned in the Chapter B, 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**.

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 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.

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

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, and the value of *M* is set to 1.

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 its 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.

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 [11]. 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.

IV. Results

This thesis comprehends two important Case Studies: The Assets Selection Comparison Study and the XGBoost Certainty Thresholds Study.

A. 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 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 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 today, in the different intervals based on the average $\mathbf{F_{Score}}$ of the stock.

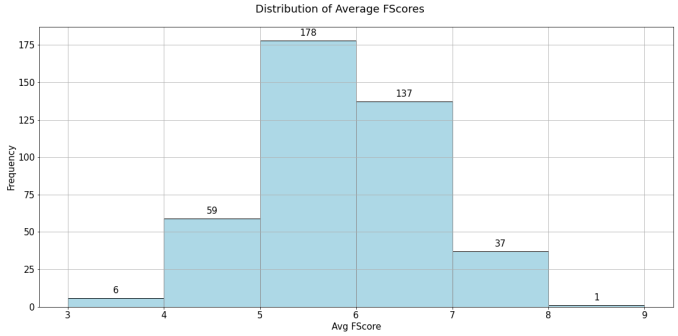

Fig. 4: Distribution of the average $\mathbf{F_{Score}}$ of the stocks.

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 I, 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.

TABLE I: 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]

The stocks from the interval [3 to 5] have the worst returns during the entire trading session, with an 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 Maximum Draw Downs 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 Maximum Down 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 performance 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 5 shows the best results of the stocks from each interval compared to the S&P500 index.

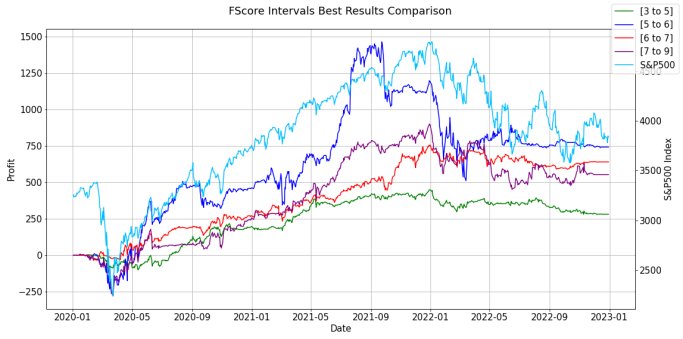

Fig. 5: Best results of the stocks from each $\mathbf{F_{Score}}$ interval.

The results displayed in the Figure 5 reflect the observations made from the Table I. 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 between, 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.

B. 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 6 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.

The Figure 6 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

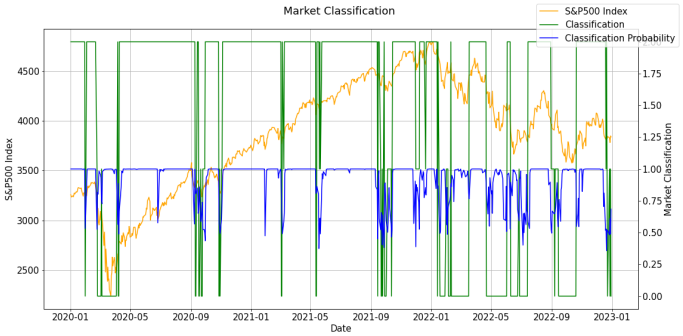

Fig. 6: Trend Classification of the S&P500 index and the probability of the predictions.

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 $\mathbf{F_{Scores}}$ 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 II.

By looking at the results from the Table II, 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

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

Open Threshold (%)	Close Threshold (%)	#Pos.	ROI (%)	Sharpe Ratio per year	MDD per year (%)
95	95	61	10.48	[-0.002, 0.082, -0.147]	[28.9, 2.1, 1.2]
	90	61	10.76	[-0.002, 0.088, -0.147]	[28.9, 2.1, 1.1]
	85	61	10.77	[-0.002, 0.088, -0.147]	[28.9, 2.1, 1.1]
	70	61	10.72	[-0.002, 0.087, -0.153]	[28.9, 2.1, 1.2]
	50	61	10.65	[-0.002, 0.087, -0.169]	[28.9, 2.1, 1.2]
	0	61	10.65	[-0.002, 0.087, -0.169]	[28.9, 2.1, 1.2]
90	95	64	9.46	[-0.004, 0.071, -0.134]	[28.9, 2.2, 1.1]
	90	64	9.73	[-0.004, 0.076, -0.135]	[28.9, 2.2, 1.12]
	85	64	9.83	[-0.004, 0.079, -0.135]	[28.9, 2.2, 1.12]
	70	64	9.79	[-0.004, 0.078, -0.141]	[28.9, 2.2, 1.12]
	50	64	9.82	[-0.004, 0.08, -0.161]	[28.9, 2.2, 1.12]
	0	64	9.82	[-0.004, 0.08, -0.161]	[28.9, 2.2, 1.12]
85	95	79	7.66	[-0.006, 0.06, -0.13]	[28.9, 1.9, 1.3]
	90	79	7.98	[-0.005, 0.066, -0.131]	[28.9, 1.9, 1.3]
	85	79	7.94	[-0.006, 0.068, -0.131]	[28.9, 1.9, 1.3]
	70	79	7.81	[-0.006, 0.067, -0.137]	[28.9, 1.9, 1.3]
	50	80	7.75	[-0.006, 0.07, -0.134]	[28.9, 1.9, 1.0]
	0	80	7.78	[-0.006, 0.07, -0.133]	[28.9, 1.9, 1.0]
70	95	83	7.18	[-0.006, 0.053, -0.136]	[28.9, 1.7, 1.2]
	90	83	7.37	[-0.006, 0.059, -0.136]	[28.88, 1.74, 1.2]
	85	83	7.54	[-0.006, 0.061, -0.136]	[28.88, 1.73, 1.2]
	70	83	7.42	[-0.006, 0.06, -0.142]	[28.88, 1.73, 1.3]
	50	84	7.44	[-0.006, 0.062, -0.138]	[28.88, 1.73, 1.0]
	0	84	7.47	[-0.006, 0.062, -0.137]	[28.88, 1.73, 0.9]
50	95	92	6.96	[-0.001, 0.055, -0.152]	[11.4, 1.5, 0.9]
	90	92	7.02	[-0.005, 0.061, -0.152]	[11.5, 1.5, 0.9]
	85	92	7.27	[-0.001, 0.063, -0.152]	[11.5, 1.5, 0.9]
	70	92	7.16	[-0.001, 0.062, -0.156]	[11.5, 1.5, 0.9]
	50	93	7.18	[-0.001, 0.065, -0.144]	[11.5, 1.5, 1.0]
	0	93	7.21	[-0.001, 0.065, -0.143]	[11.5, 1.5, 1.0]
0	95	95	6.59	[-0.004, 0.045, -0.153]	[11.4, 1.5, 0.9]
	90	95	6.84	[-0.002, 0.051, -0.154]	[11.5, 1.5, 0.9]
	85	95	6.89	[-0.004, 0.053, -0.154]	[11.5, 1.5, 0.9]
	70	95	6.79	[-0.004, 0.052, -0.158]	[11.5, 1.5, 0.9]
	50	96	6.82	[-0.004, 0.054, -0.145]	[11.5, 1.5, 1.0]
	0	96	6.85	[-0.004, 0.054, -0.144]	[11.5, 1.5, 1.0]

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 Maximum Draw Down per year. The first year of the trading session has a Maximum Draw Down 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 Maximum Draw Down 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 7 it is shown the best results for each opening threshold with the corresponding closing threshold that maximizes its returns.

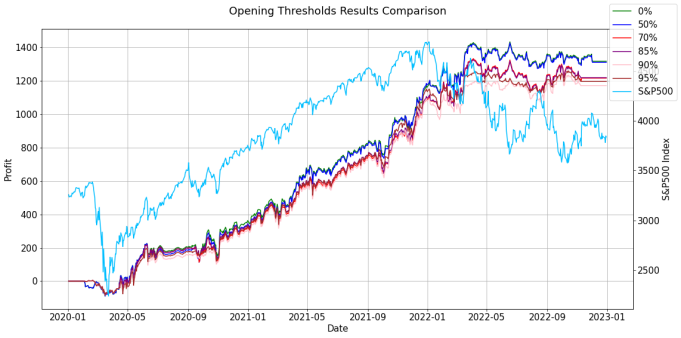

Fig. 7: Comparison of the best results between the opening thresholds.

The results from the Figure 7 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.

V. Conclusions and Future Work

A. Conclusions

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.

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.

B. 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 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.

REFERENCES

[1] James Chen. Stock Analysis: Different Methods for Evaluating Stocks. *Investopedia*, 3 2023.

[2] Rodrigo Lopes de Almeida and Rui Ferreira Neves. Stock market prediction and portfolio composition using a hybrid approach combined with self-adaptive evolutionary algorithm. *Expert Systems with Applications*, 204:117478, 2022.

[3] António Gorgulho, Rui Neves, and Nuno 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.

[4] Bernardo Jubert de Almeida, Rui Ferreira Neves, and Nuno Horta. Combining support vector machine with genetic algorithms to optimize investments in forex markets with high leverage. *Applied Soft Computing*, 64:596–613, 2018.

[5] António Canelas, Rui Neves, and Nuno 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.

[6] Rui Neves Diogo Matos. Multi-class trading strategies optimized through multi-objective genetic algorithms for the forex market. 2022.

[7] Adam Hayes. Stock Symbol (Ticker Symbol): Abbreviation for a Company's Stock — investopedia.com. 2024.

[8] The long term perspective on markets, 2024.

[9] XGBoost Python Package &x2014; xgboost 2.0.3 documentation — xgboost.readthedocs.io.

[10] Bharathi. Latest Guide on Confusion Matrix for Multi-Class Classification — analyticsvidhya.com.

[11] pandas.Series &x2014; pandas 2.2.2 documentation — pandas.pydata.org.