TÉCNICO LISBOA

Hybrid System of Classification Algorithms Voting System with Genetic Algorithm for Dividend Stocks Ranking

Rodrigo Henriques da Silva Lopes

Thesis to obtain the Master of Science Degree in

Eletrical and Computer Engineering

Supervisor: Prof. Rui Fuentecilla Maia Ferreira Neves

Examination Committee

Chairperson: Prof. João Manuel de Freitas Xavier
Supervisor: Prof. Rui Fuentecilla Maia Ferreira Neves
Member of the Committee: Prof. Rui António Dos Santos Cruz

June 2022

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

Acknowledgments

It is with great happiness that I write this message to all those who were present in my academic path and who helped me to finish this course.

I would like to thank my supervisor, Professor Rui Neves, for giving me the opportunity to do this project.

To all my friends who accompanied me during this very important phase of my life, who helped me overcome many difficulties, and it is because of them that I never had to spend any moment, or overcome any difficulty, alone.

And last but not least, I would like to give my most sincere and enormous thanks to my parents, my brother and my whole family. There wasn't a single moment when I felt that I didn't have their huge and unconditional support. Without them none of this would be possible.

Thank you to everyone who has been a part of my journey,

Rodrigo Lopes

Resumo

Nesta tese, é proposto um sistema de votação com algoritmos de classificação (XGBoost e Máquina de Vector de Suporte) que pretende analisar os dados financeiros das empresas que constituem o Standard & Poor's 500 (S&P500), e prever quais (daquelas que apresentam um aumento consecutivo dos dividendos pagos) irão continuar essa série de aumentos, e quais irão quebrá-los. Os algoritmos terão uma fase de optimização de parâmetros, feita por um Algoritmo Genético e uma Pesquisa de Grelha, e depois passarão à fase de previsão. São testados nos anos 2017, 2018 e 2019, e desenvolverão um sistema que atribui uma classificação a cada acção, facilitando assim a escolha dos melhores para investir. Além disso, é implementado um método de janela deslizante, onde o sistema é avaliado para uma sequência de anos de teste, facilitando assim a obtenção de um sistema preciso e mais fiável. Esta janela deslizante contém todos os módulos de classificação e previsão, a fim de executar sequencialmente o sistema várias vezes, melhorando assim as suas previsões com cada iteração.

Palavras-chave: Análise Fundamental, Dividendos, XGBoost, Máquina de Vector de Suporte (SVM), Algoritmo Genético, Pesquisa de Grelha

Abstract

In this thesis, a voting system is proposed with classification algorithms (XGBoost and Support Vector Machine (SVM)) that intends to analyze the financial data of the companies that constitute the Standard & Poor's 500 (S&P500), and predict which ones (of those that present a streak of increasing the dividends paid) will continue that streak and which ones will break it. The algorithms have a parameter optimization phase, done by a Genetic Algorithm and grid search, and then move on to the prediction phase. They are tested in the years 2017, 2018 and 2019, and create a system that assigns a rank to each stock, thus making it easier to choose the best ones to invest in. On top of this, a sliding window method is implemented, where the system is evaluated for a sequence of test years, making it easier to get an accurate and very reliable system. This sliding window contains all the classification and prediction modules in order to sequentially run the system several times, thus improving its predictions with each iteration.

Keywords: Financial Analysis, Dividends, XGBoost, Support Vector Machine (SVM), Genetic Algorithm, Grid Search

Contents

List of Tables

List of Figures

Accronyms

CSR - Corporate Social Responsibility

DBN - Deep Belief Network

DC20 - Dividend Constant 20 years

DR10 - Dividend Raise 10 years

EBIT - Earnings Before Interest and Tax

EBITDA - Earnings Before Interest, Tax, Depreciation and Amortization

GA - Genetic Algorithm

GE - General Electric

IPO - Initial Public Offering

ML - Machine Learning

MLNN - Multi-Layer Neural Networks

MSE - Mean Squared Error

NaN - Not a Number

PR-AUC - Precision-Recall Area-under-Curve

RBF - Radial Basis Function

ROC-AUC - Receiver Operator Characteristic Area-under-Curve

ROI - Return on Investment

RBF - Radial Basis Function

R&D - Research & Development

S&P500 - Standard and Poor's 500

SGA - Selling General and Administrative costs

SMA - Simple Moving Average

SUS - Stochastic Universal Sampling

SVM - Support Vector Machines

Chapter 1

Introduction

The stock market, is a well studied market, where many investors have already participated, as it has always attracted many types of investors. This market has many components and consequently changes due to many factors, which causes many investors, too often, to lose with investments . Still, if one researches and studies the market enough, there is proof that one can profit a lot as well.

These investors differ a lot in types of preferred investments that they tend to keep more often in their portfolios. One way to invest, for example, is in dividend-paying companies. In this case investors will be looking for companies that will continue to pay dividends and keep them rising steadily. This is not always easy to predict, as there are many factors that can lead a company to change its dividend policy.

In this first chapter the following topics will be discussed:

- The **Motivation** that led to the development of this project.
- The **Objectives** that were in mind when developing this project.
- The **Contributions** that were made with the completion of this project.
- The **Document Structure**, which will explain how this project is divided, and a summarized description of its different parts.

1.1 Motivation

A big part that motivated this project was the work that has been done on machine learning (ML) and supervised leaning algorithms. Over the years, many ways have been found to make them more efficient and more able to predict future results. These algorithms can be used to predict a number of different aspects in a large range of different situations, which is why they are becoming increasingly popular when it comes to financial investments. And in a market like the stock market, where there are many different types of data that can be fed to these algorithms, it is normal that in some cases they can be quite efficient. Even so, their predictions are not certain, and probably never will be, and that is why studies such as this project are continuously being done, in order to better optimize prediction processes like this.

In the case of this project, we will apply classification algorithms to the dividend stocks market, in order to try to predict the future situations of companies' dividend policies.

1.2 Objectives

The main objectives of this project are to:

- Study the effect of certain financial ratios on companies' dividend payouts, and consequently, their dividend streaks.

- Through the use of various ML algorithms, analyze the cause/effect relationships mentioned above.
- Tune the algorithms' parameters with aid from genetic algorithms, so that an optimal set of parameters can be achieved.
- Get rankings for all the companies from S&P500, in respect to their future dividend actions and values.
- Assess which companies will have better performances in the future (dividend wise), and the predictability associated with those performances.

1.3 Contributions

The contributions made by this project are the following:

- A better understating on how a multi-algorithm voting system can improve the results of single-algorithm methods.
- A system that is able to identify which companies will continue their dividend streaks, by using classification algorithms with parameter optimization through various processes, such as Genetic Algorithms.
- Assigning a rank to all companies present in the dataset, in respect to their future dividend payments.

1.4 Document Structure

This project is divided into four distinct parts, where each of this parts had their contribution to the completion of the project, and are the following:

- **Chapter 2 -** In this chapter, it is presented the necessary background associated with the stock market, dividend payments and also an overview of the algorithms used in this project.
- **Chapter 3 -** In this chapter, the implementation of the proposed system will be described. In it are complete and extended explanations on the functionality and reasoning behind every module incorporated into this system, and how they all interact with each other.
- **Chapter 4 -** In this chapter, all the metrics used to evaluate the proposed system are described. The performance of the system will be put to test, according to various testing metrics and in different case studies.
- **Chapter 5 -** In this final chapter of the project, there will be a discussion on the conclusions upon the completion of this project. Additionally, some types of possible future improvements on this work will be discussed.

Chapter 2

Background and Related Works

In this chapter, The fundamental information behind all parts of the project are presented and explained succinctly. It starts with an explanation about stock markets, stocks and dividends, and goes on to explain the basis of fundamental analysis, and the main reason why it is such a viable option this days, when looking at investing in the stock market. Finally, it ends with all the needed information behind all the ML concepts, processes and algorithms that will help in the development of this system (e.g., supervised learning, the SVM) algorithm, Genetic Algorithms, etc).

2.1 Stock Market and Stocks

The stock market is where investors buy and sell stocks. A stock is a small fraction of the company that is sold to investors in order to finance the company. Not all companies are listed in a market, therefore, not all companies have stocks. In order to enter any market, the company first has to release an Initial Public Offering (IPO), which refers to the release of a set of shares to be sold to investors.

There are a few reasons for which a company could decide to go public. the most important advantage of going public, is that the company can use the money from selling their stocks to finance and grow the company. There is always the option of asking for a loan at a bank, but the interest rates could put the company in serious debt. Another strong reason for going public is the visibility that the company gets just by going public, as people who have not heard anything of the company, will start to look at its financial statements and maybe invest.

At a fundamental level, supply and demand are the forces that dictate the variations and value of stocks. Basically what this means is that, if at a certain moment, there are more people wanting to buy a stock then there are people wanting to sell, than the stock price will go up, as the demand is greater than the supply. Conversely, if the exact opposite occurs (i.e., supply is higher than demand) than the price would fall.

2.2 Dividends

Dividends are the distribution of some of the profits of a company to its shareholders, seen as a kind of reward for the investment made by the shareholders in the company. Common shareholders are normally entitled to receive these payments, the only requirement being to hold stocks of the company they own on the ex-dividend date (i.e., when dividend eligibility expires). The type of payment is usually in the form of cash, but can also be given in stocks, or other types of property. The type and amount of the dividend is determined within the company by its board of directors.

Dividends normally originate from a company's net profits. Even if most of the profits go to reinvest in the company, the rest can be used to reward its shareholders in the form of dividends. At times, some companies may continue to pay dividends even if they have not made a profit, in order to maintain their reputation as a dividend payer and thus keep shareholders interested in continuing to invest.

2.2.1 Important dates

There is always a chronological order of events/dates related to the most important steps in the dividend payment process.

- Announcement Date - Date on which it is announced by the company management that dividends will be paid.

- Ex-Dividend Date - Date on which dividend eligibility expires. Those who hold stocks of the company up to a day before this date will still receive the dividends in question, whereas if the acquisition of the stocks has occurred on or after the ex-dividend date, then these shareholders will not be eligible to receive the dividends that the company in question has previously announced.

- Record Date - Also known as the cut-off date, is set by the company itself and means the deadline for deciding which of the shareholders are eligible to receive the dividend payments

- Payment Date - The date on which the dividends are paid to the shareholders.

2.2.2 Why do companies pay dividends?

Shareholders may expect dividends as a return on their confidence in the company. Company management may aim to fulfill this sentiment by providing a solid record of dividend payments. Dividend payments have a positive impact on the company and help maintain investor confidence. Dividends are also favored by shareholders because in many countries, dividends are considered tax-free income for shareholders. In contrast, capital gains realized through the sale of rising shares are considered taxable income. Traders seeking short-term gains may also prefer to receive dividend payments that provide immediate tax-free gains.

The announcement of high dividends can indicate that the company is doing well and generating good profits, but this is not always the case. It can mean that the company does not have the right projects in place to generate better returns in the future. Therefore, it is using its money to pay shareholders, rather than reinvesting it in growth.

If a company has a long history of paying dividends, reducing or canceling the amount of dividends can signal to investors that the company is in trouble. On November 13, 2017, General Electric (GE), one of the largest industrial companies in the United States, announced a 50% dividend reduction, and GE's stock price fell more than 6%.

Reducing the amount of dividends or deciding not to pay any dividends at all may not necessarily translate into bad news for the company. Given its financial and operating conditions, the company's management may have a better investment plan. For example, a company's management may choose to invest in a high return project. Compared to the small returns they realize through dividend payments, the project may bring greater returns to shareholders in the long run.

2.2.3 Dividend-paying strategies

The frequency with which dividends are paid changes from company to company, this stems from various types of strategies that can be used according to the vision and priorities of the company in question. Some of the most common strategies are:

- Residual - Companies that abide by this type of dividend policy, usually need the generated equity to reinvest in themselves and only use any residual equity for dividend-paying. On the one hand, this gives the company some capital that can help finance new projects and thus help the company move forward. On the other hand, investors may demand a higher stock price relatively to other companies with a more stable and consistent dividend policy. Another downside to this type of dividend policy is that, the price of the company's stock will probably be more volatile than it should be due to the sporadic nature of this type of dividend policy.
- Stable - In this type of strategy, the company makes dividend payments each year, regardless of the company's earnings. The main goal of this type of policy is to give the investors more confidence knowing that there will be a regular return on their actives. These payments can be either quarterly, semi-annual, or yearly. The payment frequency and amount, are chosen through a thorough earnings forecast and by determining the earnings percentage that is to be payed out to the investors.
- Hybrid - The hybrid approach is a type of combination of the Residual and Stable methods. This is one of the most common approaches in dividend policy, and consists in the company deciding on a small percentage of the company's earnings to give out in a regular manner, and additionally giving out special dividends when the earnings exceed the companies expectations.
- Special - A company can also decide to pay special dividends.These types of dividends can be paid either by a company that pays dividends regularly, or by one that has never paid a dividend, and are therefore independent of the history of dividends paid by a certain company.

2.2.4 Dividend Investing

Recently, investors have started to take on a more Dividend Investing strategy. As stated by Clemens [1], these types of investments can perform better than other types of strategies, such as broader market and value investing. This is because they have lower volatility, which makes them more predictable, while still being able to show good returns. These returns will come from both the dividend payments, and the security increase.

As stated in Lichtenfeld [2], a dividend portfolio has to be as diversified as any other investment portfolio, both in return risk and in terms of the sectors in which companies operate. This way, even if a sector is in some difficulty, or if a once promising company starts to fall, the investor will always be protected and in the long run will have a superior return.

Many times, what happens is that usually, dividend investors try to invest on companies with dividend yields around 10%, which can turn this into a more inconsistent method, like many others. An optimal

scenario would be to choose companies with dividend yields that are not too low, but also not extravagant, such that it can have a signficant growth in the future. As it is said in [2], in the long run, it will always be better to buy stocks from a company that has a starting dividend yield of 4% but goes up 10% every year, from a company that starts with 6% but goes up around 3% per year.

2.3 Fundamental Analysis

Fundamental Analysis is a method, or a group of methods, used to determine the intrinsic value securities (e.g., stocks), in order to understand which ones are being undervalued or overvalued by the market.

For the purpose of finding the "real" value of a certain stock, one must first look at the company's financial statements. Then, through the use of a selected group of financial values and ratios, assess a certain asset. However, analysts typically start by doing a thorough analysis of the economy and the industry in which the stock is inserted to arrive at a good value for the stock. Fundamental Analysis is the favored approach for long-term investors, as it helps to understand the overall strength and durability of an asset.

2.3.1 Fundamental Analysis vs Technical Analysis

Besides Fundamental Analysis, there is also Technical Analysis. Both of these approaches use some kind of historical information. Fundamental Analysis uses the information contained in financial statements in order to evaluate a stock's fundamental value and compare it to the actual market value to see if the stock is overvalued or undervalued, whilst Technical Analysis uses the price and volume over time in order to predict certain trends in the market [3].

Technical Analysis

- Looks into price and volume movements in order to find statistical patterns.
- Based on finding optimal entry points.
- Daily trading, short-term investments.

Fundamental Analysis

- Look into the overall state of a company, through thorough analysis of financial statements.
- Main goal is to find intrinsic value of a company, and check if it is under or overvalued.
- Long-term investments.

2.3.2 Financial Statements

Financial statements are essentially records that show the financial state of a company, and they are used by investors to analyze a company and consequently try and predict the direction the company is in. Financial statements are divided into three separate parts, which are: **Income statement**, **Balance Sheet** and **Cash Flow Statement**.

Income Statement

The income statement shows the overall values of revenues, expenses and the net profit in a given period of time, which is usually three months. Regarding the revenue, there are three factors that come into the equation. First, there is the total or gross revenue, which is the amount of money that entered the company in the given period time, then there is the cost of goods sold, which is how much it costs to the company to buy or manufacture the product that went on to sell later. Finally there is the gross profit, which is the gross revenue minus the cost of goods sold. This value by itself does not give much information, but it is used to calculate the gross profit margin.

The expenses covers all of the hard costs related to Research and Development (R&D), Selling, General and Administrative costs (SGA) and Depreciation. By subtracting the expenses from the revenue, one gets the operating profit, or earnings before income tax (EBIT).

After the taxes are redacted to the EBIT, one gets the net earnings value, which is fundamentally the amount of money generated by the company in the period of time designated to that income statement [4].

An example of an income statement can be seen in Figure 2.1.

Revenue	
Total Revenue	110 360,00
Cost of Revenue, Total	38 353,00
Gross Profit	72 007,00
Operating Expenses	
Selling/General/Admin. Expenses, Total	22 223,00
Research & Development	14 726,00
Total Operating Expense	75 349,00
Operating Income	35 011,00
Income from Continuing Operations	
Total Other Income/Expenses Net	1 416,00
Earnings Before Interest and Taxes	35 058,00
Income Before Tax	36 474,00
Income Tax Expense	19 903,00
Net Income From Continuing Ops	16 571,00
Non-recurring Events	
Discontinued Operations	-
Extraordinary Items	-
Effect of Accounting Changes	-
Other Items	-
Net Income	
Net Income	16 571,00
Preferred Stock and Other Adjustments	-
Net Income Applicable to Common Shares	16 571,00

Figure 2.1: Income Statement and its components of a Company X

Balance Sheet

The balance sheet, unlike the income statement, is not according to a period of time, but instead for a certain instant in time, showing the financial state of a company on that specific date. The balance sheet is broken into assets, liabilities and the shareholders equity. Assets are divided into current assets (which include cash, inventory), receivables and non-current assets (such as property), equipment, intangibles and long-term investments. The liabilities are also divided into current and non-current liabilities. Current liabilities are debts and obligations due within the current fiscal year includes accounts payable, accrued expenses, short-term debt and long-term debt that is due this year. Non-current liabilities is the long-term debt that is due next year or later, including deferred income tax, minority interest, etc. Finally, if subtracted all the liabilities from all of the company's assets, it results in the shareholders equity, or the net worth of the company, which is the total value of the company [4].

An example of an income statement can be seen in Figure 2.2.

Current Assets	
Cash and cash equivalents	11,296
Receivables	4,315
Allowance for doubtful accounts	-65
Inventories	10,396
Other current assets	2,450
	28,392
Non-current assets	
Equipment	60,782
Vehicles	87,5
Accumulated depreciation	-19,387
	128,895
Total Assets	157,287
Current Liabilities	
Interest payables	600
Account payables	4,600
Accruals	2,800
Other current liabilities	4,355
	12,355
Non-current liabilities	
Note payables	35,000
	35,000
Total Liabilities	**47,355**
Equity	
Share capital	65,000
Retained earnings	29,572
Profit/(Loss) current year	15,360
Total Equity	**109,932**
Total Liabilities and Equity	**157,287**

Figure 2.2: Balance Sheet and its components of a Company X

Cash Flow Statement

Cash flow statements, like income statements, refer to an interval of time. Investors use them to get a better understanding of where the money is coming from and where it is being spent. The cash flow statement is separated into operating activities, investing activities and financing activities. Operating activities start with the net income added with the proper depreciation and amortization. Then there is the investing activities, which document all changes in equipment, assets, or investments. Finally, the financing activities include all outward dividend payments, buying and selling of the company's stocks and issuance/repayment of debt.

Adding all the parts from the cash flow statement results in the company's net change in cash [4]. An example of a cash flow statement can be seen in Figure 2.3

Cash Flow From Operations	
Net Earnings	2 000 000
Additions to Cash	
Depreciations	10 000
Decrease in Accounts Receivable	15 000
Increse in Accounts Payable	15000,000
Increase in Taxes Payable	2 000
Subtractions Frm Cash	
Increase in Inventory	-30 000
Net Cash From Operations	2 012 000
Cash Flow From Investing	
Equipment	-500 000
Cash Flow From Financing	
Notes Payable	10 000,00
Cash Flow for FY Ended 31 Dec 2021	**1 522 000**

Figure 2.3: Cash Flow Statement and its components of a Company X

2.3.3 Financial Ratios

After going through the financial statements of a company, investors use certain numerical values to compose financial ratios, in order to preform a quantitative analysis of the financial state of the company.

These ratios can be broken into five different categories:

- Liquidity Ratios - These ratios access a firm's ability to pay off current obligations without the help of external capital. Liquidity ratios help understand the capabilities a firm has to avoid insolvency [5].

- Leverage Ratios - These ratios give the investors some information regarding how the business's operations are being financed, whether coming from debt/liabilities or from equity. In a fundamental level, these type of ratios help determine the amount of debt that a company has, and how much of its financing comes from it [5].

- Efficiency Ratios - These ratios help measure a company's performance and how well it is being managed. It shows how efficient a company is in terms of using its assets and liabilities to maximize its sales and generate income. For example, efficiency ratios can be used to see how fast a company can turn inventory or assets into cash, which can give a better understanding of a company's efficiency and consequently, profitability (These are also called Asset turnover ratios).

- Profitability Ratios - These ratios helps measure how well a company can generate profit from its sales or operations. It helps the investors get a more deep understanding of how likely it is that the company will be able to generate a significant return on its investments. When looking at these ratios' results, the higher they are, the better, although, it gives a better understanding when comparing them with other company's in the same market/sector, or with the history of the company itself [5].

- Market Value Ratios - These ratios are used in order to evaluate the financial status of the company in relation to the market. In other words, they help to understand if the company is being undervalued or overvalued in the marketplace.

2.4 Machine Learning

There are many different ML algorithms, which can be divided into three leading categories:

- Supervised Learning - In supervised learning, the algorithm is trained with the help of labeled datasets, in order to predict or classify an outcome.
- Unsupervised Learning - In unsupervised learning, the algorithm analyzes the dataset and detects patterns or groupings with no need for labels or any sort of human intervention.
- Reinforcement Learning - Reinforcement learning is, in some sense, similar to supervised learning. But instead of using labeled data, the algorithm learns with a series of attempts, while receiving rewards depending on the current performance of the algorithm.

2.4.1 Supervised Learning

Supervised learning is based on learning by example. Usually the algorithm is presented with a training set and a test set, each of which are input/output pairs that the model in the training phase will use to learn, so that when in the testing phase, without having access to the outputs, it will make a prediction.

The composition of the training dataset can be represented by:

$$\mathcal{A}_n = ((x_1, y_i), (x_2, y_2), ..., (x_n, y_n)) \in (\mathcal{X} \times \mathcal{Y})^n \tag{2.1}$$

where

$$n$$

is the number of instances in the dataset, and x_i are input vectors, each with dimension

$$m$$

, that belong in a feature space $\mathcal{X} \in \mathbb{R}$, and y_i is the output of each entry from the dataset, in the case of this project, this variable will be binary as it can only portrait two different values $\{0, 1\}$. The algorithm will then, try to create a function $\mathcal{F}$ to serve as a way to predict the outcomes in the testing phase of the algorithm. This function can be represented as:

$$\mathcal{F} : \mathcal{X} \to \hat{\mathcal{Y}} \tag{2.2}$$

Where $\hat{y}$ is the predicted outcome by the model. Finally, in order to access the performance of the model, the predicted outcome is compared to the actual outcome that was hid from the model during the testing phase. This is usually done through the use of a loss function that checks the error of the prediction. There are many loss functions that can be used in this type of problems, and one of the most common is the Mean Squared Error (MSE).

Algorithms

As stated by Nasteski [6], Supervised Learning algorithms can be sorted into Regression and Classification categories.

- **Regression** based algorithms are used to get a better understanding of possible correlation between certain variables. The output value is a real or a continuous value.
- In **Classification** the output is not a continuous value as in regression. In classification, the result is a label. A classification algorithm, will receive the input values and try to predict the label (or labels) associated with each input.

2.4.2 XGBoost

XGBoost is a scalable machine learning system for tree boosting, which through successive iterations combines a multitude of week learners into a strong learner.

As seen in Figure 2.4, with each iteration of the algorithm, the residue of the current prediction function will be used to improve its performance, in order to minimize the chosen loss function, creating a new function, with new features. This will continue until the stopping criteria is met.

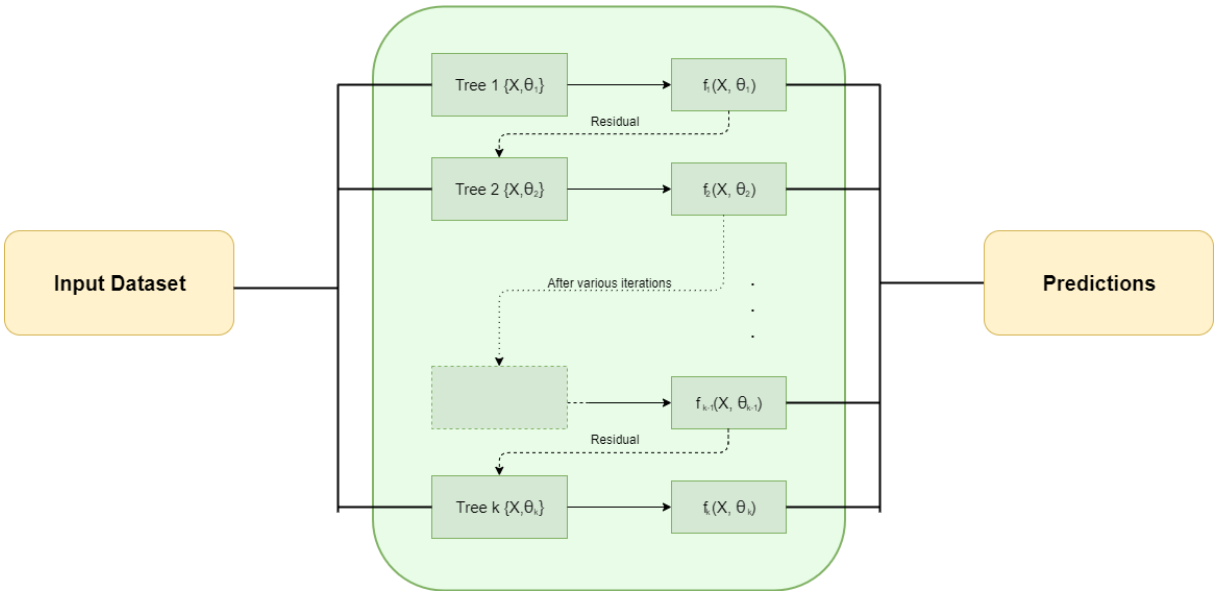

Figure 2.4: XGBoost Algorithm

As stated by Chen and Guestrin [7], given a dataset $\mathcal{D} = \{(x_i, y_i)\} \quad (|D| = n, x_i \in \mathbb{R}^m, y_i \in \mathbb{R})$, where n is the number of examples and m the number of features, an algorithm such as this (tree ensemble algorithm) utilizes K addictive functions in order to make a prediction of the outcome, as shown in Equation 2.3.

$$\hat{y} = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in \mathcal{F} \tag{2.3}$$

where $\mathcal{F}$ is the space of the regression trees, and can be defined as:

$$\mathcal{F} = \{f(x) = w_{q(x)}\}(q : \mathbb{R}^m \to T, w \in \mathbb{R}^T) \tag{2.4}$$

In eq.2.4 the structure of each tree is represented by q, and the number of leaves in each tree, by T. Each f_k will represent a tree with structure q, with leaf weights w. In the specific case of regression trees, the weight (or score) is considered in each leaf node, and in the node i, the weight assigned will be w_i [7].

This algorithm proceeds with the use of the decision rules given by the tree (represented by q) to classify each leaf node, thus assigning a score to each leaf node. Consequently, when all leaf nodes have a score (w) assigned, one sums the scores of all leaf nodes, obtaining the final desired prediction. As one would expect from a ML classification algorithm, the main objective would have to be to minimize a certain loss function, where in this scenario is given by the equation 2.5.

$$\mathfrak{L}(\phi) = \sum_{i} l(\hat{y}_i, y_i) + \sum_{k} \Omega(f_k) \tag{2.5}$$

l here is a convex and differentiable loss function, and Ω is an additional smoothing term that serves to help adjust the computed weights, in order to prevent over-fitting and its mathematical definition is given as:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2 \tag{2.6}$$

The model represented above in equation 2.6, has functions as its parameters, which makes it impossible to solve this optimization problem in a Euclidean space. Consequently, it will be necessary to add f_t, in order to minimize the objective function seen below in equation 2.7.

$$\mathfrak{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}^{(t-1)} + f_t(x_i)) + \Omega(f_t) \tag{2.7}$$

With the term $\hat{y}^{(t-1)}$ being the *i*-th instances' prediction at iteration t. Afterwards, in order to optimise the general objective, a second-order approximation is used in equation 2.8.

$$\mathfrak{L}^{(t)} \simeq \sum_{i=1}^{n} [l(y_i, \hat{y}^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) \tag{2.8}$$

where the equations 2.9 and 2.10 are first and second order gradient statistics on the loss function,

as stated by [7].

$$g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)}) \tag{2.9}$$

$$h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)}) \tag{2.10}$$

The next step is removing the constant terms as means to obtain the simplified objective function at step t , and is seen in equation 2.11.

$$\tilde{\mathfrak{L}}^{(t)} = \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) \tag{2.11}$$

And then, by expanding Ω and defining $I_j = \{i | q(x_i) = j\}$, equation 2.11 can be redefined as in equation 2.12.

$$\begin{aligned} \tilde{\mathfrak{L}}^{(t)} &= \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \\ &= \sum_{j=1}^{T} [(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i + \lambda) w_j^2] + \gamma T \end{aligned} \tag{2.12}$$

Afterwards the optimal weight w_j^* of leaf j is obtained, through the use of the following equation 2.13:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \tag{2.13}$$

Finally, according to [7], the optimal loss reduction is calculated by using the equation 2.14,

$$\tilde{\mathfrak{L}}^{(t)} = -\frac{1}{2} \sum_{j=1}^{T} \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma T \tag{2.14}$$

2.4.3 Support Vector Machines

As stated by Satapathy et al. [8], SVM is one of the most used ML Algorithms. It is based on statistical learning theory and pattern classification and it came to scene in 1995 when it was introduced by Vapnik [9]. The aim of SVM is to take in the data and the label associated with each data entry ant create a dividing hyperplane that optimizes data classification. Obviously there will be several different hyperplanes that can divide the data correctly but the optimal hyperplane will be the one that maximizes the distance between the hyperplane and the closest data entry of each class. In figure 2.5, an hyperplane and the corresponding support vectors are represented.

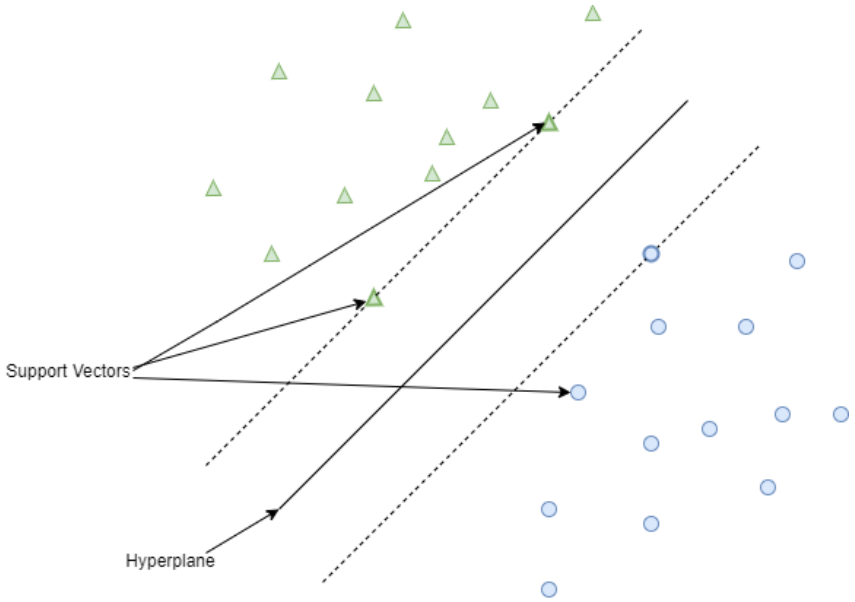

Figure 2.5: Support Vector Machine - Support vectors and hyperplane representation

Linearly Separable Case

Let's take a simple example on a two-dimensional plane, with a linear solution, to be easier to visualize. We can begin a training sample $S = (x_1, y_1), (x_2, y_2), ..., (x_l, y_l)$, where $S \subseteq (X \times Y)^l$. $X \subseteq \mathbb{R}^n$ corresponds to input space, $Y = \{-1, +1\}$, corresponds to the output domain used to classify data, and l is the total number of data entries, Mammone et al. [10].

The points x which are directly on the hyperplane satisfy the condition: $\langle w, x \rangle + b = 0$ with w and $x \in \mathbb{R}^n$ and $b \in \mathbb{R}$, where w is a vector perpendicular to the hyperplane and as one modifies the value of b, the hyperplane moves parallel to its position. This condition is relative to the decision function $f(x) = sign(\langle w, x \rangle + b = 0)$ and one can assume that, for the data points that are closest to the hyperplane, this decision function assumes the value of 1 or -1. Given two points (x^+ and x^-) that are the closest to the hyperplane, in respect to each class (x^+ corresponds to the positive class and x^- corresponds to the negative class), we can conclude the following equations:

$$\langle w, x^+ \rangle + b = 1 \tag{2.15}$$

$$\langle w, x^- \rangle + b = -1 \tag{2.16}$$

$$\gamma = \langle \frac{w}{\|w\|}, (x^+ - x^-) \rangle = \frac{2}{\|w\|} \tag{2.17}$$

Some properties of the SVM algorithm are represented below in figure 2.6

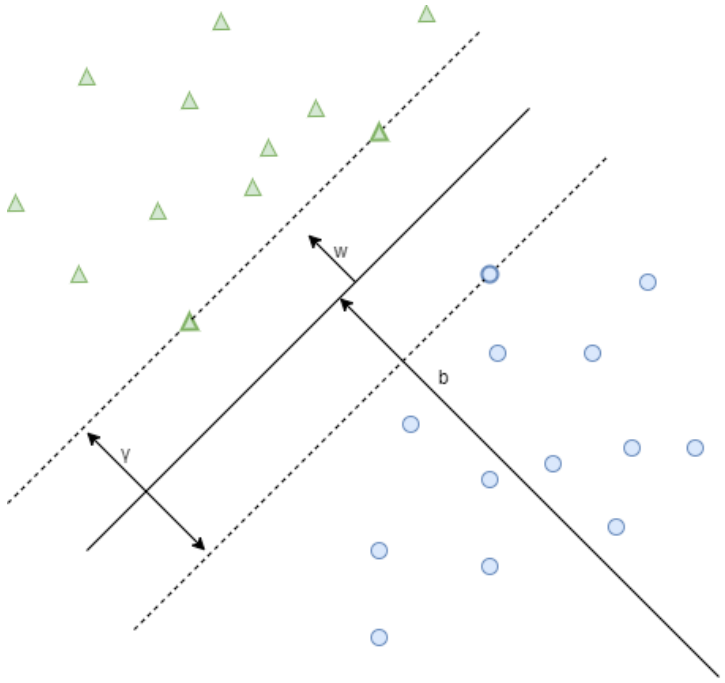

Figure 2.6: Support Vector Machine - w, γ and b representation

One can, this way conclude, that the margin is inversely proportional to $\|w\|$. The main goal of the algorithm is to find the hyperplane that admits the largest margin, which can be solved by minimizing $\|w\|$.

$$\begin{aligned} & min\langle w, w\rangle, \\ s.t. \quad & y_l(\langle x_l, w\rangle + b) - 1 \geq 0 \end{aligned} \tag{2.18}$$

Which can be solved through the primal Lagrangian formulation of the problem,

$$L(w, b, \alpha) = \frac{1}{2}\langle w, w\rangle - \sum_{i=1}^{l} \alpha_i [y_i(\langle w, x_i\rangle + b) - 1]. \tag{2.19}$$

The next step is as easy as calculating the partial derivatives of L in respect to the variables w and b,

$$\frac{L(w, b, \alpha)}{\partial w} = w - \sum_{i=1}^{l} y_i \alpha_i x_i = 0, \tag{2.20}$$

$$\frac{L(w, b, \alpha)}{\partial b} = \sum_{i=1}^{l} y_i \alpha_i = 0, \tag{2.21}$$

and by solving the equations 2.20 and 2.21 we derive the following:

$$\sum_{i=1}^{l} y_i \alpha_i x_i = w, \tag{2.22}$$

$$\sum_{i=1}^{l} y_i \alpha_i = 0, \tag{2.23}$$

Subsequently, when substituting equations 2.22 and 2.23 into equation 2.19, we get the optimization problem with linear constraints:

$$max \quad W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \tag{2.24}$$

$$s.t. \quad \sum_{i=1}^{l} y_i \alpha_i = 0, \tag{2.25}$$

Finally, the Karush-Kuhn-Tucker complementary conditions are required and enough to find the optimal solution (α^*, w^*, b^*) of the primal. The solution has to meet the following condition:

$$\alpha_i^* [y_i(\langle w^*, x_i \rangle + b^*) + 1] = 0, \quad i = 1, ..., l. \tag{2.26}$$

Through a simple analysis of the equation 2.26, one can verify that a certain point can meet the condition either if $\alpha_i^* = 0$ or if $y_i(\langle w^*, x_i \rangle + b^* = 1$. Additionally, only points that are, in fact, closest to the hyperplane, can have $\alpha_i^* = 0$, and can consequently be called *support vectors*, and are the most important part of classifying the sample in question. On the contrary, the other data points in the sample (the ones that are not *support vectors*), if were to be removed and the training repeated the solution would remain the same, as they do not hold any information to the construction of the hyperplane.

Nonlinear Case

In the previous example, the classification problem is very simple, as the different classes can be separated by a straight line. One can see that in a linear-solvable problem, the data appears in the form of an inner product ($\langle x_i, x_j \rangle$). In an example that cannot be solved by a straight line, a new problem arises.

One can start by mapping the data points into a higher dimensional space, by replacing $\langle x_i, x_j \rangle$ with $\langle \phi(x_i), \phi(x_j) \rangle$. $\phi(x)$ is a transformation of x and does not need to be known as it is implicitly determined by the choice of a kernel. The choice of an appropriate kernel will help data go from being linearly non-separable in the input space to being easily separable in a selected feature space.

There are a variety of different kernels that can be chosen to solve problems like this, but the three most commonly used in classification problems, and more specifically in SVM, can be seen in the following Table 2.1 [11]:

Table 2.1: Types of Kernels that can be used for the SVM

Kernel Type	**Kernel Equation**
Linear Kernel	$x^T y + c$
Polynomial Kernel	$(x^T y + c)^n$, where n is the Kernel degree
Radial Basis Function(RBF) Kernel	$e^{-\frac{\|x-y\|}{\sigma}}$

The first step to take in this type of aproach changes the data representation,

$$x = (x_1, x_2, ..., x_n) \longmapsto \phi(x) = (\phi_1(x), \phi_2(x), ..., \phi_N(x)) \tag{2.27}$$

As stated by Vasco Amaral [11], the next steps of the process are similar to what is seen in the previous example, which means that the data classification will merely depend on the “inner product of an unknown vector in a high dimensional space as a function of vectors in the original space”, this inner product will not need to be known as it is represented by the kernel function (as seen in the equation 2.28), and that is all that is going to be needed to calculate the hyperplane that has the maximum distance to the classified data points, as expressed in equation 2.28.

$$K(x, z) = \langle \phi(x), \phi(z) \rangle \tag{2.28}$$

2.5 Genetic Algorithms

Genetic Algorithms (GAs) are stochastic search algorithms inspired by the basic principles of biological evolution and natural selection, Scrucca [12]. GAs simulate the evolution of living beings, where the “fitter” solutions prevail over the weaker ones, while mimicking evolutionary mechanisms such as mutations and selection.

In a GA, each individual is represented by a chromosome. Each chromosome is a fixed array of numerical values, or genes, where each gene, like in biology, represents a specific feature of the individual. Each individual is a certain solution to the problem in question, and as the GA develops, the set of individuals will change, in order for the algorithm to find the optimal solution to the problem. A population of a Genetic Algorithm is represented in figure 2.7

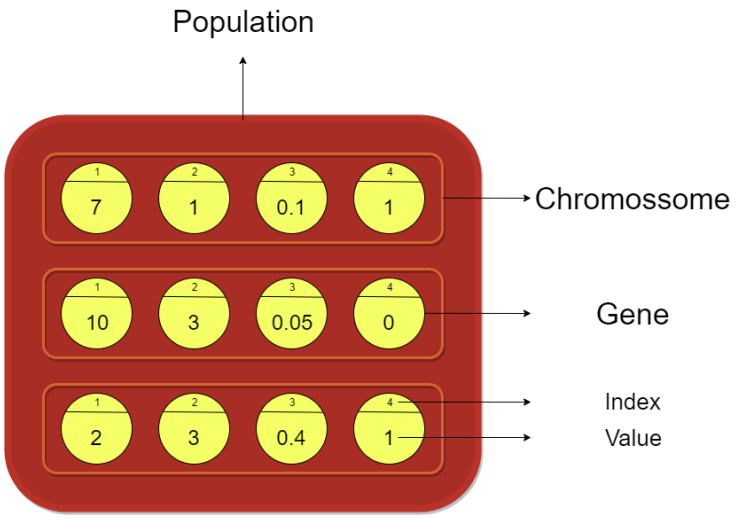

Figure 2.7: Graphical Representation of a Population, its chromosomes and respective Genes

A GA is divided into several stages. In the first stage, the algorithm initializes a set of chromosomes, each with its own set of randomly selected genes. These chromosomes compose the initial population and the first to go through the process of evaluation. During this stage, every chromosome is given a certain fitness value, which is directly associated with the quality of a solution. This value is dictated by a pre-determined fitness function.

The next stage is called **Selection**, which selects the individuals that will serve as parents to produce offspring for the next generation. There are many methods of selecting the parents. The most used ones are: Fitness Proportionate Selection, Stochastic Selection, Ranked Selection and Random Selection.

- **Proportionate Selection -** In Proportionate Selection, or "Roulette Wheel Selection" as it is commonly called, the more fit an individual is, the more probable it is to be chosen as a parent. It starts off by calculating the probability that every individual has to become a parent in each selection. That probability ($P(x)$) is calculated through the equation 2.29, where f_x is the fitness value of the chromosome x and N is the number of chromosome in the population. Then each parent is chosen at a time, amongst the mating pool according to the probabilities in question.

$$P(x) = \frac{f_x}{\sum_{i=0}^{N} f_i} \tag{2.29}$$

- **Tournament Selection -** Tournament Selection is one of the most simple types of selection. It begins by selecting a random set of individuals, then between those individuals, the most fit go to the mating pool. This process is repeated until the mating pool is completely full.

- **Ranked Selection -** Ranked Selection is mostly used when the individuals in the population have very close fitness values. Each individual is given a rank depending on its fitness value (i.e., the

most fit individual is given Rank 1, the second most fit, is given Rank 2, etc). The higher an individual is ranked, the more likely it is to be chosen.

- **Random Selection -** Random selection is the most simple and unreliable type of selection, as the fitness of the individuals plays no part in the selection process. In this strategy, the parents that will go to the mating pool are randomly chosen.
- **Stochastic Selection -** In this type of selection, and like in Proportionate Selection, a selection probability is assigned to each individual according to the respective fitness value, as shown in Table 2.2.

Table 2.2: Stochastic selection - Representation of population and its fitness values and selection probabilities

Individual #	**1**	**2**	**3**	**4**	**5**	**6**
Fitness Value	2.0	1.8	1.6	1.4	1.2	1
Selection Probability	0.22	0.2	0.18	0.16	0.13	0.11

Then, all individuals are mapped into a contiguous line, where each individual's segment line is equal in size to its selection probability. Here, several equally spaced pointers, are set within the same line, with the number of pointers (N) being determined by the number of individuals that are going to be selected. The distance between the pointers is then set as the inverse of the number of individuals chosen ($\frac{1}{N}$), and a random number inside the interval $[0; \frac{1}{N}]$ is generated. The first pointer will be located in the position of that same random number, with the remaining pointer being located according to the distance between pointers mentioned before.

A visual example of that process is represented in Figure 2.8, where in a population with 6 individuals, 4 are meant to be chosen by this selection process. This will result in a pointer distance equal to 0.25.

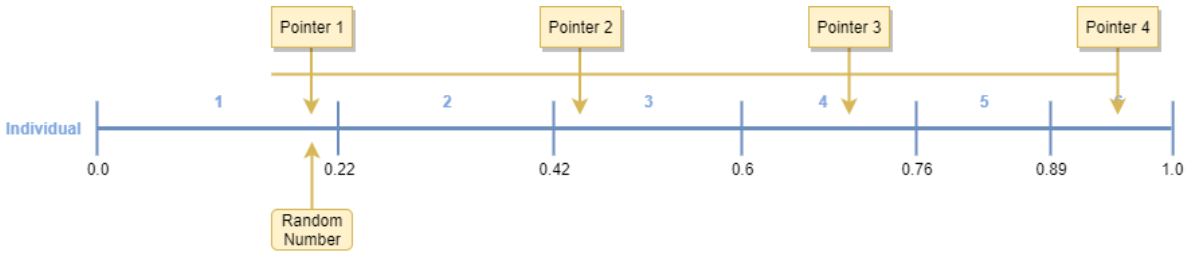

Figure 2.8: Graphical Representation of Stochastic Selection Process - Random number generated for pointer selection equal to 0.2

Then, after selecting the parents, there is the **Crossover** stage where the parents that were previously chosen, combine their genes in order to create the offspring. some genes are chosen from one parent, and some from the other, and the way the genes are chosen can vary depending on the type of crossover used. While there are many types of crossover, the most commonly used are:

- **One Point Crossover -** In One Point Crossover, a random point is selected in the chromosome (e.g., after gene 4, as shown in Figure 2.9), and the genes after that crossover point are switched between both parents to create two new offspring.
- **Two Point Crossover -** In Two point Crossover, two random points are selected in the chromosome, and as in one point crossover, the offspring are the result of both parents changing the genes after every crossover point (e.g., after gene 2 and again after gene 6, as shown in Figure 2.10. In this case, the genes that are going to be changed are the ones between both crossover points.
- **Uniform Crossover -** In Uniform Crossover, there are no crossover points. A crossover vector with the same length as the parents' chromosome is created with random values being assigned to each index between 0 and 1. If the value of a certain index or more is higher than a threshold chosen previously, then the respective genes of the parents are switched to create the offspring. A representation of an uniform crossover is shown in Figure 2.11

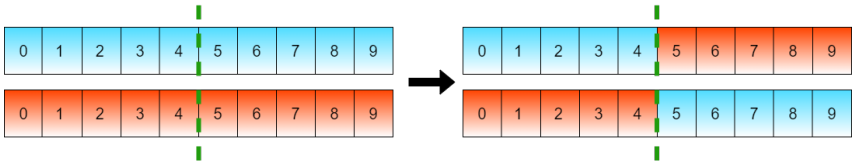

Figure 2.9: One Point Crossover

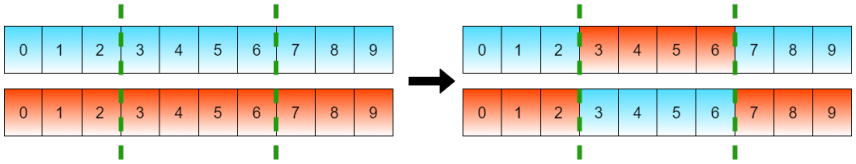

Figure 2.10: Two Point Crossover

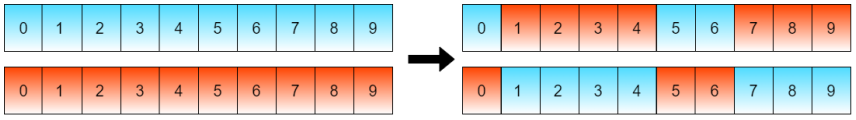

Figure 2.11: Uniform Crossover

After crossover, the **Mutation** stage takes place. In this stage, the resulting offspring have a small chance of changing their genes. This stage serves the purpose of increasing the variety and differences

within the population, but has a small probability of actually happening, so that the algorithm does not become completely random. There are some types of mutations used in GA, but in this specific project the only one to take into account is where a random gene gets changed into another random value.

Then, after the mutation occurs, it is the **Survivor Selection**. This part of the algorithm serves the purpose of deciding which individuals will be taken away from the population in order to be substituted by the new offspring, and which will stay. Some times it is employed an **Elitism** strategy, where the fittest individual, no matter what, stays inside the population in order to preserve the best solution at all times.

The easiest way to select the individuals to be taken away from the population is through a random selection, but that can easily be problematic as it can have certain convergence issues. Therefore, there is usually a strategy applied in order to optimize the algorithm. There are two types of strategies in survivor selection, **Age Based Selection** and **Fitness Based Selection**.

- **Age Based Selection** is based on the premise that each individual should only stay a finite number of generations inside the population, generating new offspring, so the new offspring will substitute the older individuals.

- In **Fitness Based Selection** the new offspring will substitute the individuals with the poorest fitness, as to maintain a good level of individuals inside the population. The individuals to be replaced do not always have to be the ones with the least fitness score, e.g. the selection of the least fit individuals can be done using a tournament selection

Finally, there is the **Termination Condition**, which determines when the algorithm should stop running. GAs normally progress very fast in the first iterations of the algorithm, as better solutions continue to appear, but as the algorithm goes on, fewer good solutions appear in each iteration, as the GA begins to saturate. Consequently, the termination condition is a very determining factor.

The most commonly used termination conditions are:

- When the fitness of the population has shown zero improvement for a certain number of generations

- When a certain number of generations has been reached

- When an individual has reached a certain pre-determined fitness level

2.6 Related Works

In this section, a summary of the research and work that has been done in Dividend Investing and ML over the years. In Section 2.6.1, there will be a extent research on what can drive a company to pay dividends, and what type of strategies there are in relation to dividend payments. Later in Section 2.6.2, the attention will be more focused on how ML algorithms have evolved over the years as well as how well their performances have been, on dividend investments and some other practices.

2.6.1 Dividends

In 2001, Fama and French [13] conducted a study that showed a decrease in the percentage of dividend paying companies. It showed that between the years of 1978 and 1999, the proportion of companies that effectively payed dividends, went down, from 66.8% to 20.8%. Their study presented a variety of factors that could have caused this decrease in dividend paying companies (e.g., "lower transaction costs for selling stocks for consumption purposes" and "larger holdings of stock options by managers who prefer capital gains to dividends"). However, in 2004, DeAngelo et al. [14] continued to investigate the subject in question, and found that, although there was a decrease in the percentage of companies that payed dividends, the aggregate amount of dividends being paid was, in fact, increasing, which happened due to the fact that the majority of payers that stopped paying dividends, corresponded to relatively small firms, and the increase of dividend payments, by much larger firms, "swamp" the reductions of smaller firms dividend payments.

As stated by Allen and Michaely [15], there is a very important statement when discussing dividend policy, that says that in case of a dividend increase there will be a positive reaction by the market and all its constituents, and respectively, in case of a dividend decrease by a company, then the market will react very negatively, as it could mean the some problem relative to the company in question may be resurfacing.

In order to better understand dividend smoothing and what drives certain companies to do so, some studies were made by Javakhadze et al. [16] on a sample of more than two-thousand firms from around the world. As a result from those studies, [16] cites that "Managers of firms with low market-to-book ratios, less cash, low dividend payouts, and few tangible assets engage greater dividend smoothing", while firms that are in their early stages, usually show less dividend smoothing.

Corporate Social Responsibility(CSR), has become more important and relevant over time. Hence, a study was made by Benlemlih [17], in the interest of getting a better understanding on the matter. The study showed that companies with lower CSR appeared to give out smaller dividend payments, than companies with a higher CSR. Low CSR firms also take less time to adjust their dividend payments, which make them less stable then dividends given by high CSR firms. Finally, [17] showed that firms that are involved in controversial products or services, for example alcohol, usually give less dividends.

In 2020, Krieger et al. [18] studied the effects of the COVID-19 pandemic on US-firms' dividend payouts. Results showed that, out of a pool of 1400 dividend paying firms, about 15.2% cut their dividend payments, and 6.6% omitted their dividend payments completely.

Dividend investing has become more and more common each year, and in 2021, Berre [19] developed a study with the sole purpose of comparing several portfolios according to different dividend investing strategies to a classical S&P500 portfolio. Berre [19] used portfolios such as DC20(Dividend Constant 20 years) and DR10(Dividend Raise 10 years). These portfolios were tested in several decades and the results implied that dividend growth portfolio strategies outperformed the typical S&P500, when measuring certain performance metrics such as the annual yield and return, and Sharpe Ratio. The one portfolio strategy that stood out the most was the DC20 as the DR10 sometimes had very little companies and could show some levels of discrepancy, unlike the DC20.

2.6.2 Machine Learning

SVM has been very commonly used in all sorts of classification problems. One typical situation where one might use an algorithm such as this, is in financial related predictions. It is very common to use SVM fed with technical indicators in order to predict future stock prices of one or more companies [20]. And in 2007, Han and Chen [20], conducted a study where, instead of resorting to technical indicators, fed the algorithm with the information from the companies' financial statements instead. This Fundamental Analysis type of approach showed great results, and the SVM model appeared to display a better accuracy than those models who analyzed technical information. Then, in 2008, Ding et al. [21] developed a study, where instead of trying to predict stock prices, wanted to see if one could predict a company's future dividend policy, and if such could be done with good accuracy performances when resorting to an SVM model. Ding et al. [21] came to the conclusion that, even though the algorithm appeared to have great levels of accuracy in its predictions, it also had a substantial error associated with it. The experiment was done using several kernels as well, so as to get the best out of the algorithm and eventually the kernel with best results was the RBF kernel. Furthermore, in 2010, Bae [22] tested several ML classification techniques on how well they would preform in predicting future dividend policies for several companies. The one that appeared to have the best performances out of the rest, was the SVM algorithm.

In order to test the performance of several ML algorithms, Huang and Yen [23] gathered the financial information of several Taiwanese companies, and ran them through various ML algorithms, both supervised and unsupervised ML algorithms, and even tested with an hybrid form of DBN(Deep Belief Network)-SVM, to see which would give the best predictions. Inside the supervised learning algorithms were the SVM and XGBoost, which incidentally were the ones that showed some of the best performances. An also good approach was the hybrid algorithm, as it made better predictions than those of the SVM, although, the XGBoost was the one that evidently stood out from all of them. During recent years, with more incoming studies on XGBoost, it became a very popular algorithm in financial prediction, and in 2021, Ozlem and Tan [24] started to investigate the performance of certain ML algorithms when trying to predict future dividend policies, and of course, one of them would be XGBoost itself, with the other one being Multi-Layer Neural Networks (MLNN). The algorithms would receive data from the companies' financial statements (e.g., cash, sales, earnings per-share), and try to predict if a company would pay dividends in the future or not. Even though both algorithms showed good results, XGBoost was the one that stood out the most, specially when being run with carefully chosen parameters. But this is not the only environment for which the XGBboost evidently exceeds in comparison to other ML algorithms. In 2020, Shimin et al. [25] conducted a study to see if machine learning algorithms could detect and prevent financial fraud, a problem very present and important in the world of finance. The study involving XGBoost also considered several other ML algorithms, such as Naive Bayes and Logistic Regression, but XGBoost clearly outperformed the other algorithms, and again stood out as being the most trustworthy in financial related predictions.

Chapter 3

Implementation

In the next chapter will focus in the implementation of the proposed system and all of its parts. It will start by showing how the overall architecture works and how the various modules interact with each other, and proceeds to give a more thorough analysis and explanation on how each of those modules works.

3.1 Overall Architecture

In this section, the complete architecture of the implemented system will be discussed in a general way. There is going to be presented a general description of all different modules, in order to obtain a good understanding of how each one works and, consequently, how this project was developed. In order to get a better understanding, there is a graphical representation of the architecture in Figure 3.1, along with a brief description of the sequence of events involved in this system.

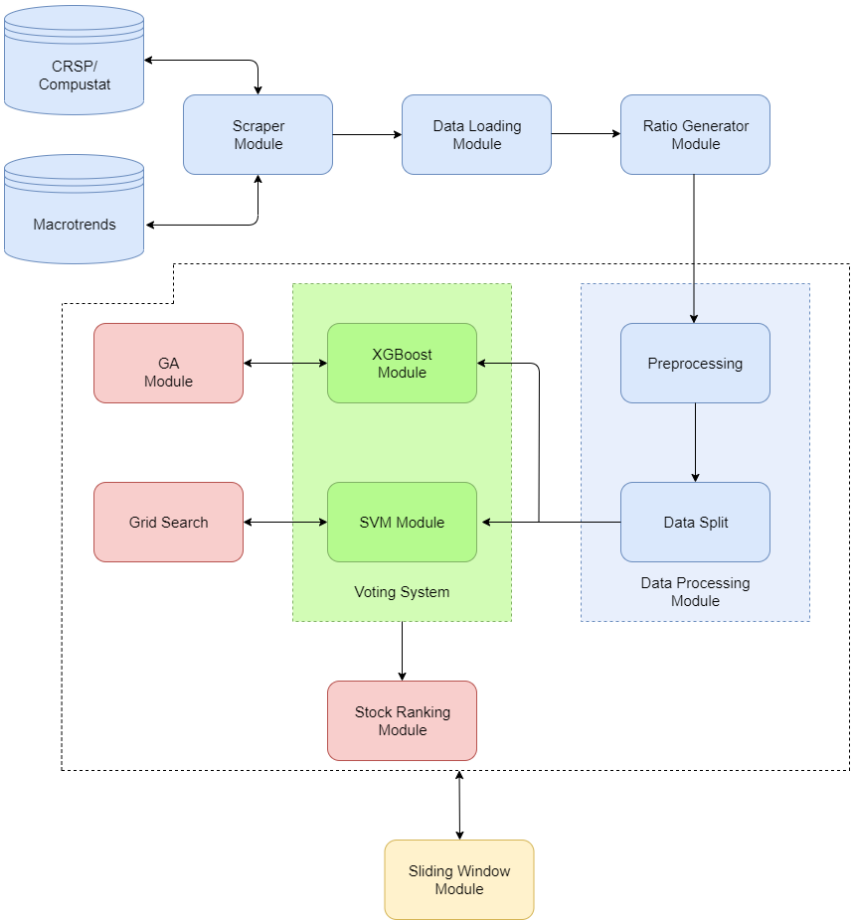

Figure 3.1: Overall Architecture of the Implemented System

The first part of this project is based on obtaining the data that will be needed for the project and processing it so that it is ready to serve as input for the classification modules.

- First, the main part of the resulting dataset, is exported from the CRSP/Compustat[26] merged

database in the form of a *.csv* file. This includes data from the *Income Statement*,*Balance Sheet* and *Cash Flow Statement* which will be further explained in section 3.2.

- The next step is to download all the data needed from the multiple databases. This process is implemented in the *Scraper* module, and it makes possible to download the dividends and prices from *Yahoo Finance*Finance [27], and some additional financial items from the *macrotrends* database.

- Afterwards, the *Ratio Generator* module exerts its functions of calculating the desired financial ratios, the dividend streak indicators and the labels later used for classification. The choice of these ratios (along with all other components) and a better explanation on how they work, can be seen in Section 3.4.

- After generating the financial ratios, all the data needed for the classification algorithms to use is now available. The last step left to complete the data handling section will take place in the data processing module. This module can be divided into two parts. A first pre-processing part, where the data will go through a series of filtering and elimination processes, and a second part where the data will be divided into several datasets (i.e. training, validation and test datasets). This separation is further explained in Section 3.6.

Now that the dataset is completed, the next part is the classification phase. In this part, the data will be fed to the algorithms, and with some aid from the GA for validation purposes, the classification part will handle all the data analysis and predictions. This is the most crucial part, as it will have the most impact on how well the system will preform.

- First the algorithms have to go through a validation phase where the algorithms analyze the data served as input over and over again in order to find the optimal values for their parameters. This is done through the use of a GA and grid search depending on the algorithm.

- Then, after the optimal parameters have been found, the algorithms are ready to make their predictions. Each algorithm will return their own predictions along with a probability assigned to each prediction.

- Afterwards, comes the voting phase. This is where the algorithms come to an agreement on which predictions to use. This is done through analyzing both the predictions and probabilities of each algorithm, and deciding on what should be done relative to the classification problem at hand.

- After the voting, phase, there is now a final prediction from the part of the predictors. This prediction will enter the ranking system and begin by evaluating all the companies, and according to the predictions, compare them all and assign a score to each. This score will be used later to make a ranking of the best companies in the dataset, in terms of future dividend payments.

Finally, the modules mentioned above in the classification part of the system, are all functioning inside the *Sliding Window* module, where the system repeats all the steps all the while increasing the

years where the datasets are placed, along the entire functioning period. With the conclusion of this module, the system returns the stock rankings, in respect to the years inside this same period of time, therefore assigning a rank that takes into account all the years, giving this way, a more complete solution.

In the following sections, each and every module will be described in a more thorough way, in order to get a better understanding of how each of the modules mentioned in this section function and interact with each other.

3.2 Data Loading Module

This module is used to collect Financial data from all companies in the S&P500 index, do some minor processing in the database to adjust the data to what is needed and append all different types of data collected during this phase.

The main database used was the CRSP/Compustat Merged Database, and was used to collect nearly all the data needed. This database contained all information from the Income Statement and Balance Sheet from all the desired companies. The downloaded document was in .csv format where each column was either a data identifier (date of the statement, quarter, year, tic of the company, etc) or a specific financial item of the company (Revenue, Cost of goods sold, Long term debt) and each row was associated with a specific record, identified by the data identifiers. The extracted data, obtained from this .csv file had 32020 rows and 376 columns. A part of this dataframe is show in Figure 3.2.

	tic	datadate	fyearq	fqtr	fyr	indfmt	consol	popsrc	datafmt	datacqtr	datafqtr	actq	atq	cheq	cogsq	cshoq	dlttq
31120	ATVI	30/06/2015	2015	2.0000	12	INDL	C	D	STD	2015Q2	2015Q2	6051.0000	14015.0000	4521.0000	192.0000	728.8700	4077.0000
31121	ATVI	30/09/2015	2015	3.0000	12	INDL	C	D	STD	2015Q3	2015Q3	6309.0000	14302.0000	4519.0000	232.0000	730.9440	4078.0000
31122	ATVI	31/12/2015	2015	4.0000	12	INDL	C	D	STD	2015Q4	2015Q4	3387.0000	15251.0000	1831.0000	410.0000	734.5030	4079.0000
31123	ATVI	31/03/2016	2016	1.0000	12	INDL	C	D	STD	2016Q1	2016Q1	4008.0000	17302.0000	2888.0000	252.0000	738.0070	5777.0000
31124	ATVI	30/06/2016	2016	2.0000	12	INDL	C	D	STD	2016Q2	2016Q2	3420.0000	16607.0000	2285.0000	283.0000	741.2730	4977.0000
31125	ATVI	30/09/2016	2016	3.0000	12	INDL	C	D	STD	2016Q3	2016Q3	5334.0000	18380.0000	4053.0000	239.0000	743.1180	4881.0000
31126	ATVI	31/12/2016	2016	4.0000	12	INDL	C	D	STD	2016Q4	2016Q4	4830.0000	17452.0000	3258.0000	456.0000	745.4870	4887.0000
31127	ATVI	31/03/2017	2017	1.0000	12	INDL	C	D	STD	2017Q1	2017Q1	4356.0000	16921.0000	3248.0000	268.0000	753.5520	4393.0000
31128	ATVI	30/06/2017	2017	2.0000	12	INDL	C	D	STD	2017Q2	2017Q2	4352.0000	16808.0000	3278.0000	256.0000	754.8200	4387.0000
31129	ATVI	30/09/2017	2017	3.0000	12	INDL	C	D	STD	2017Q3	2017Q3	5386.0000	17718.0000	3576.0000	298.0000	756.0570	4388.0000

Figure 3.2: Extracted DataFrame downloaded from the CRSP/Compustat Database

The module proceeds to receive the financial data from the *Scraper* module to continue completing the dataframe with all the needed data. The module then process all that data, and adapts to the format needed to append it to the current dataframe, all the while filtering the data to what the system will definitely need for later uses. This process is represented in figure 3.3

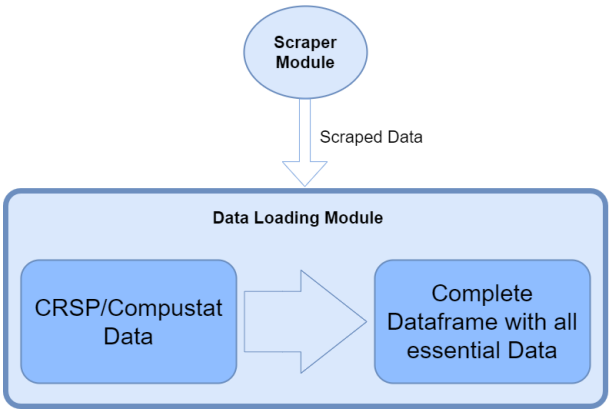

Figure 3.3: Data Loading and Scraper module representation

3.3 Scraper Module

This module is divided into two parts, one scrapes the *Macrotrends* database and the other scrapes Yahoo Finance. Firstly, the former, focuses on scraping the financial data that were not already in the data retrieved from the *CRSP/Compustat* merged database, such as Cash Flow from operating and from investing activities, and the Free Cash Flow (this financial stat, came as a Per-Share value, so the number of shares outstanding had to be retrieved as well so as to calculate the absolute value of Free Cash Flow).

Secondly, the *Yahoo Finance* scraper went on to retrieve the price and the dividends from the Yahoo Finance's Historical Data. The price retrieval is a very straightforward process. Through the python library *yfinance*, and by defining an interval of time, and the tic of a specific stock, one can get the historical prices of said stock. There are a variety of different price data that can be downloaded from *Yahoo Finance*. An explanation on the different price data can be shown bellow on Table 3.1.

Table 3.1: Different prices downloaded from Yahoo Finance

Open Price	Price at which a stock started trading at a certain day
High Price	Highest price a stock was traded during a certain day
Low Price	Lowest price a stock was traded during a certain day
Close Price	Price at which a stock is last traded at a certain day

The price that was chosen for future uses was the Close price.

The next step is to retrieve the dividend history of a stock from *Yahoo Finance* as well. This part is not as straightforward as getting a stock's price. The initial part is very similar to getting the price data. First neeeds to be inserted the time period to which one wants to get the data from, and the tic

of the company, and *yfinance* returns the dividend history. Although, as you download dividends for each of the target companies, one needs to check if there are any stock splits during the time period of the extracted dividends. If there are not any stock splits recorded in the selected time period, then the dividends retrieved from *Yahoo Finance* are returned like that, without need for any type of adjustment.

On the contrary, if there is in fact a record of a stock split or more for a certain stock, in the time period that was previously selected, then an extra step is needed. Typically when a stock split occurs, the dividend also splits roughly in the same ratio than that of the stock split. This can easily cause a problem when trying to calculate dividend streaks, so in order to reduce this problem, the dividend payments before the stock split occurred are divided by the ratio of the stock split in question. In Figure 3.4, there is a graphical representation of the process.

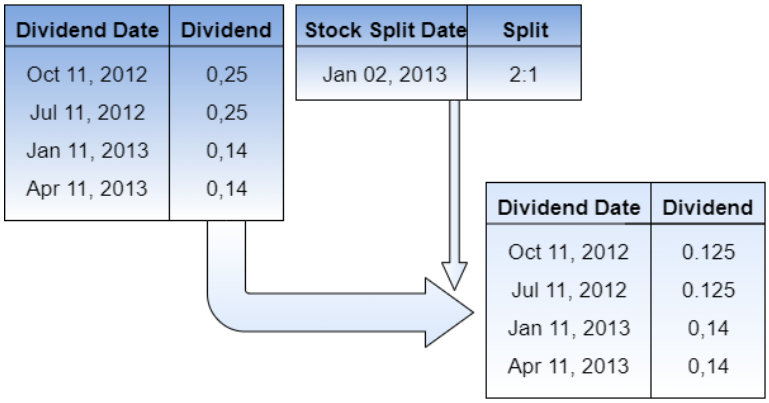

Figure 3.4: Graphical representation of a data split correction for dividends

This module works semi-parallel to the Data Loading module, as it continuously send the scraped data to the Data Loading module in order to load the various scraped financial data into a single dataframe.

3.4 Ratio Generator Module

This module receives the quarterly financial data from the Data Loading module described in Section 3.2, and proceeds to calculate the following metrics:

- **Financial Ratios -** Ratios calculated from the financial data contained in the *Income Statement*, *Balance Sheet* and *Cash Flow Statement*.
- **Dividend Streak Flags -** Flags to help keep track of which stocks have ongoing dividend streaks up to the date in question.
- **Classification Labels -** This classification Labels are what the algorithms will use to classify the data in the training phase, and will later try to predict when in the testing stage

The ratios were chosen based on the studies made by Francisco Silva [28] and after a few experiments using those same ratios, some were added, and a few of those stayed because they had some influence in the decision making process of the algorithm when trying to classify and predict data. All the ratios used in this project, are represented in Table 3.3

In order to make it easier to follow the dividend streaks of the various companies at any given date, certain flags have been created. In order to know if a company is currently on a one, or three year streak of maintaining the dividends paid, or of increasing the dividends paid. These flags can indicate if a company has a dividend paid higher than the one of the previous year, and consequently the one of the previous year being higher than the one of two years ago.

In respect to what the algorithm will try to predict (labels), what matters is whether some company in question, will continue its dividend streak (whether it is of dividend increase or maintenance), or if it will break it in the following year.

There is a more succinct explanation below in Table 3.2, with the aid of graphical representation, to better perceive this process. Since all companies are treated individually but equally, this explanation will be an example for each of the companies in the present data.

Step 1: First of all, in order to calculate the dividend streaks, we just need to use the annual amount of dividends paid, since that will be the way to verify the companies' streaks. Then, for each instance (i.e. for each quarter/year pair), we check if in that quarter, the company in question has paid dividends.

Step 2: Afterwards, if the company has paid dividends, then the next step is to verify if this value is just an isolated value or if it comes from some kind of streak from previous years. In order to check what the present picture looks like, one compares the present value of the dividend with those of previous years. In the Table 3.2 are the conditions necessary to be assigned certain flags.

Table 3.2: Flags that were considered for this project

Flag Description	**Condition**
3 years of consecutive increase of dividend payments	$Div_{year} > Div_{year-1} > Div_{year-2}$
3 years of consecutive maintenance of dividend payments	$Div_{year} \geq Div_{year-1} \geq Div_{year-2}$

Step 3: After assigning the corresponding flags for all the data entries, then the next step is, for all the instances that are already on a dividend streak, to check whether they will continue that same dividend streak in the following year or if they will break it. In order for the label to be set as **True**, the following has to happen: $Div_{year} < Div_{year+1}$ in case of checking for dividend increase, and $Div_{year} \leq Div_{year+1}$ in case of dividend maintenance.

Table 3.3: Ratios calculated in Ratio Generator module that will be used in this project

Ratio Category	**Ratio**	**Descritpion**
Liquidity Ratios	Current Ratio	Current Assets / Current Liabilities
	Quick Ratio	(Receivables + Cash) / Current Liabilities
	Cash Ratio	Cash and Equivalents / Current Liabilities
Leverage Ratios	Debt Ratio	Liabilities / Assets
	Debt-to-Equity Ratio	Liabilities / Shareholders' Equity
	Debt-to-EBITDA Ratio	Liabilities / EBITDA
	Long-Term Debt to Total Assets Ratio	Long Term Debt (LTD) / Assets
	Long-Term Debt Coverage Ratio	EBITDA/(interest + principal)
Efficiency Ratios	Assets Turnover Ratio	Revenue / Mean[Assets(t), Assets(t-1)]
	Inventory Turnover Ratio	Revenue / Mean[Inventory(t), Inventory(t-1)]
	Receivables Turnover Ratio	Revenue / Mean[Receivables(t), Receivables(t-1)]
Profitability Ratios	Gross Profit Margin	(Revenue - Cost of Revenue) / Revenue
	Return on Equity	Net Income / Shareholders' Equity
	Cash Flow Margin	Operating Cash Flow / Revenue
	EBITDA Margin	EBITDA / Revenue
	Adjusted Return on Assets	ROA / σ[ROA]
	Return on Invested Capital	Net Income / Invested Capital
Earnings ratios	Accruals Ratio	[NOA(t) - NOA(t-1)] / Mean[NOA(t),NOA(t-1)]
	Sloan Ratio	(Net Income - Operating CF - Investing CF) / Assets
Market Value Ratios	Market-to-Book Value	(Shares Outstanding * Price)/ (Assets - Liabilities)
	Dividend Yield	Dividend-per-share / Price
Dividend Ratios	Dividend Payout Ratio	Dividend / Net Income
	Dividend-to-Free Cash Flow Ratio	Dividend / Free Cash Flow (FCF)
	Sustainable Dividend Growth Ratio	ROE (1 - DPR)
	Dividends-to-Assets Ratio	Dividend / Assets
Growth Ratios	Revenue Growth	[Revenue(t) - Revenue(t-4)] / Revenue(t-4)
	Net Income Growth	[Net Income(t) - Net Income(t-4)] / Net Income(t-4)
	Debt Ratio Growth	[DR(t) - DR(t-4)] / DR(t-4)
	Debt-Equity Ratio Growth	[DER(t) - DER(t-4)] / DER(t-4)
	Long-Term Debt Growth	[LTD(t) - LTD(t-4)] / LTD(t-4)
	Dividend Growth	[Dividend(t) - Dividend(t-4)] / Dividend(t-4)
Firm Size	Firm Size	log(Assets)

3.5 Data Processing Module

The function of the data processing module is, through various procedures, to receive the data from the ratio generator module, and prepare it to serve as input to the algorithms.

Step 1: The first step of this module is to filter the necessary columns that the dataframe will need to have in order to enter the algorithms, this columns will be divided into two categories, the features and the data identifiers. The features are what will help train and evaluate the data, and the data identifiers, as the name suggests, have the sole purpose of identifying the data inside the dataframe.

Step 2: As it was previous explained in Section 3.4, the prediction label, can be relative to one company maintaining or increasing its dividend payment for the following year. Therefore, this particular step serves the purpose of determining which label the prediction will act on (which in the case of this project, the most commonly used will be the dividend increase streak). Then, depending on the label used for the prediction, the flag associated will determine which companies the dataframe will be keeping (e.g., if one chooses to evaluate the dataset according to the dividend increase label, then the only data being kept in the data set will be of companies with at least 3 years of dividend payments, all the while increasing them each year).

Step 3: Since the algorithms that will be used cannot admit string objects as values for a feature of the algorithm, we proceeded to dummify the variables that require it, which in this case will be the sectors of the companies in question. This dummification is done through the function get_dummies from Python's *pandas* library. A graphical representation of how this function works for the intended case is shown in the Figure 3.5.

Figure 3.5: Dummification of Categorical Values

Step 4: The next step of data processing is to eliminate unnecessary or inconsistent data from the dataframe. First, if there are values inside the dataframe that sit outside a selected threshold($10^{-5} < x < 10^{5}$), then they are switched with a *Not a Number* (NaN), so they are not taken into account. Then it checks if there are any rows with more NaN's than 20% of the number of columns. If there are, all of

them are removed from the dataframe.

As stated before, ML algorithms “learn” from the analysis of input data. Therefore, the variation of all different input variables will have a great influence in the analysis of the model. For instance, if the input data contains variables that come in different units, or if the range of values differs a lot (e.g., they are in different orders of magnitude), the model will have a harder time solving the problem in question, which may result in poor performance.

One way to solve this problem is through data standardization. This process, for each different variable, is based on subtracting each value by the mean (*centering*), and then dividing the result by the standard deviation (*scaling*), as represented in equation 3.1. This process leaves all the data with a mean of zero and a standard deviation of 1. This process can be done through the use of the python library *StandardScaler*

$$z = \frac{x - \mu}{\sigma} \tag{3.1}$$

In this module and many others, there is a verbose mode option for the user. This option if turned on, has the purpose of informing the user of the current status of the program in question.

In this specific module, the verbose option, starts showing the current state of the dataframe to be processed, and afterwards goes on to show which option was chosen to use in the data classification process, and how many companies will still be present after the process of filtering the data to only remain those that are going to be relevant to continue to be used for the case in question.

It has also a whole range of information that is printed for the user, such as:

- the sectors of the companies that are in the dataframe and the ones that seem to be the most predominant, in order to give the user a better understanding of the current state of the data
- the evolution of the dataframe and of its dimensions throughout the data processing stage.
- the number of positive and negative observations in order to get a better undersanding of the balancing state of the dataset.

3.6 Data Split

The main purpose of this module is to prepare the data to then go into the classification algorithms. It receives a dataframe already processed by the data processing module, and separates the data in order to obtain datasets to train, validate and test the algorithms in question.

The data will be separated into three different parts, as mentioned in [29]:

- The **Training Dataset** is the section that the model uses to see and learn from the data. It is used to build the model from the analysis of all the parameters and their associated labels. The trained model is then challenged with the validation dataset in the validation phase.

- The **Validation Dataset** is used to evaluate the model trained in the previous step. The model is tested on the validation dataset, and certain metrics such as accuracy, recall and f1score are verified, which are then used to continuously improve the model, by tuning its parameters. In this case, this validation will be performed in the Genetic Algorithm module, described in Section 3.8.

- After the parameters have been optimized to improve model performance, the model is finally tested on the testing dataset. This is the last test of the model and will serve to determine the predictions for which it is being developed.

As Falessi et al. [30] state, usually in forecasting stock prices and other financial items, the method normally used to separate the dataset for validation is the *Walk-Forward* method. In this method, when validating the model, the separation of all parts of the dataset is done as it is shown in Figure 3.6.

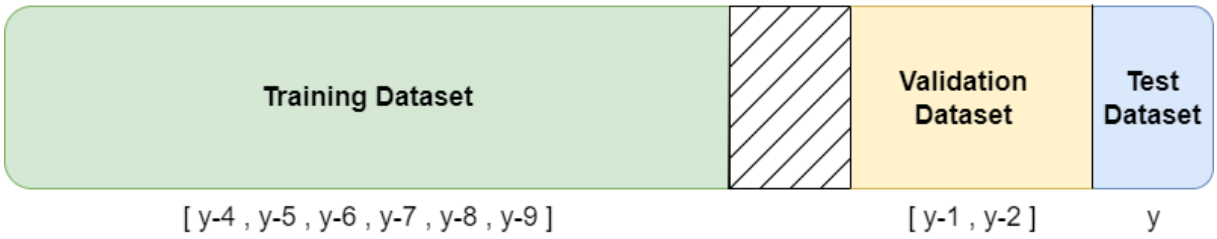

Figure 3.6: Separation of the Data with the *Walk-Forward* method - Validation Stage

As it is shown in Figure 3.6, the test year (year y) is used as a reference when mentioning the years in the rest of the dataset. The years that will be used to validate the dataset will always be the two years before, and the training dataset will have a duration of 6 years. However, they will not be immediately before the validation years, because data in consecutive years are more subject to interdependencies, so a blank year is left between the validation and training sets [28].

When the algorithm has finished the validation phase, and is already in the testing phase, then the datasets will change, because the validation set, as expected for this phase, will not exist. In this phase, the training set will increase by two years (the years taken from the previous validation set), still continuing with a one year gap for the training set, due to the reasons mentioned above. This new arrangement of datasets is represented in Figure 3.7.

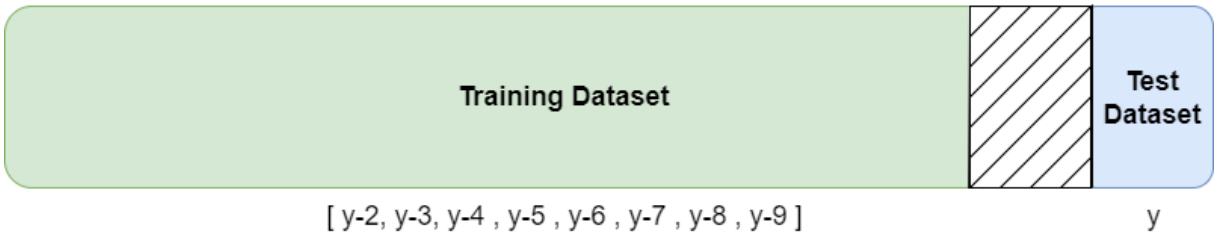

Figure 3.7: Separation of the Data with the *Walk-Forward* method - Testing Stage

3.7 XGBoost Module

This module will implement the **XGBoost Classifier**. It receives the training dataset, the evaluation dataset and the hyper-parameters of the algorithm, builds the classifier and returns the data predictions and its probabilities.

The XGboost's hyperparameters serve the purpose of regulating the learning process in order to optimize the algorithm.

XGBoost is a very powerful ML algorithm, as such, it has a wide range of hyperparameters that can be adjusted. Since it is a decision tree algorithm, the hyperparameters will define certain tree characteristic metrics, such as the maximum depth of each tree, the amount of trees, certain decision metrics involved in the splitting of specific nodes, different hyprparaenters and three different categories: **General Parameters**, **Booster Parameters** and **Learning Task Parameters**. In Table 3.4, there's the list of hyperparameters used in this project, with a brief description and its category.

Table 3.4: XGBoost parameters used in this project

Parameter Type	**Parameter Name**	**Brief Description**
General Parameters	nthread	Number of cores used for parallel processing
Booster Parameters	learning_rate	step size shrinkage used in update to prevent overfitting
	scale_pos_weight	Manages class imbalance. Helps the algorithm to converge
	min_child_weight	Minimum sum of weights required in a child node
	max_depth	Maximum tree depth
	gamma	Loss reduction needed to make a split on a leaf node of the tree
	subsample	Defines ratio of observations to be randomly sampled
	colsample_bytree	Defines the subsample ratio of columns for each tree
	lambda	L2 regularization of weights
	alpha	L1 regularization of weights
	n_estimators	Defines the number of weak learners
Learning Task Parameters	objective	Loss function to be minimized
	seed	Random number seed. Typically used for getting reproducible results

An important parameter for binary classification is the *scale_pos_weight*, as it controls the balance between the two classes. According to Luo et al. [31], the appropriate approach is to set this parameter to the ratio between the number of negatives and positives in the dataset as shown in the equation 3.2.

$$scale\_pos\_wieght = \frac{N_{negatives}}{N_{positives}} \tag{3.2}$$

Another very important topic of ML algorithms such as this, is **feature importance**, which is, as the name suggests, the impact each feature has on the resulting model prediction.

The *xgboost* Python library already has a function that computes the feature importance, which will be used in the framework of this project, in order to get a better understanding which features can be extracted from the dataset so that the predictions can become more accurate.

There are several metrics under which the importance of a feature can be assessed:

- **Weight** is the number of times a feature appears in the model trees. This metric can sometimes be misleading, for example for binary features. This type of features may only appear once in each tree. In this project there are several binary features, so weight will not be of very use.

- **Gain** is related to the importance of a feature in the model in relation to the other features. When comparing the gain of two different features, a higher value of one over the other will mean a higher importance in the prediction process.

- **Cover** measures the relative number of observations in which a certain feature is found. For example, if a feature is used to decide 10 leaf nodes in all trees altogether, then the cover for that particular feature is $\frac{10}{N}$, where N is the sum of the covers for all the features in the model.

3.8 GA Module

In the previous chapter, the hyperparameters of XGBoost, and how its performance varies depending on the choice of these parameters were discussed. In order to have a good classification algorithm, one needs to choose very carefully on which values are assigned to each parameter. A simple way to achieve this, is to test the algorithm with several different values and try to understand which values would lead to a better performance.

However, there are several different methods that can be used to optimise the process in question, such as [32]:

- **Grid Search** is the most traditional method for the optimization of hyperparameters, and consists of doing a complete search over all different values (one usually sets up a range of values for each parameter, so as to optimize the time for the algorithm) for all of them. This type of search is only used when the parameters have a small variety of values.

- In **Random Search**, instead of searching and analyzing all possible combinations of values for the parameters, it searches only for random combinations until it reaches the predetermined limit of iterations combinations.

- **Genetic Algorithm -** As discussed in Section 2.5, GA are evolutionary search algorithms, which use techniques that resemble the process of biological evolution in order to solve optimization problems. GA involves the creation of several individuals each with its score according to the problem to be solved, and through similar processes like “survival of the fittest”, individuals with better scores survive and reproduce amongst themselves, creating better offspring and so on, making algorithm converge on an optimal solution. This method of parameter optimization is much less time consuming than a grid search, and will be more effective in finding optimal solutions than a purely random search algorithm [32] [33]. A fluxogram of a GA is represented in figure 3.8.

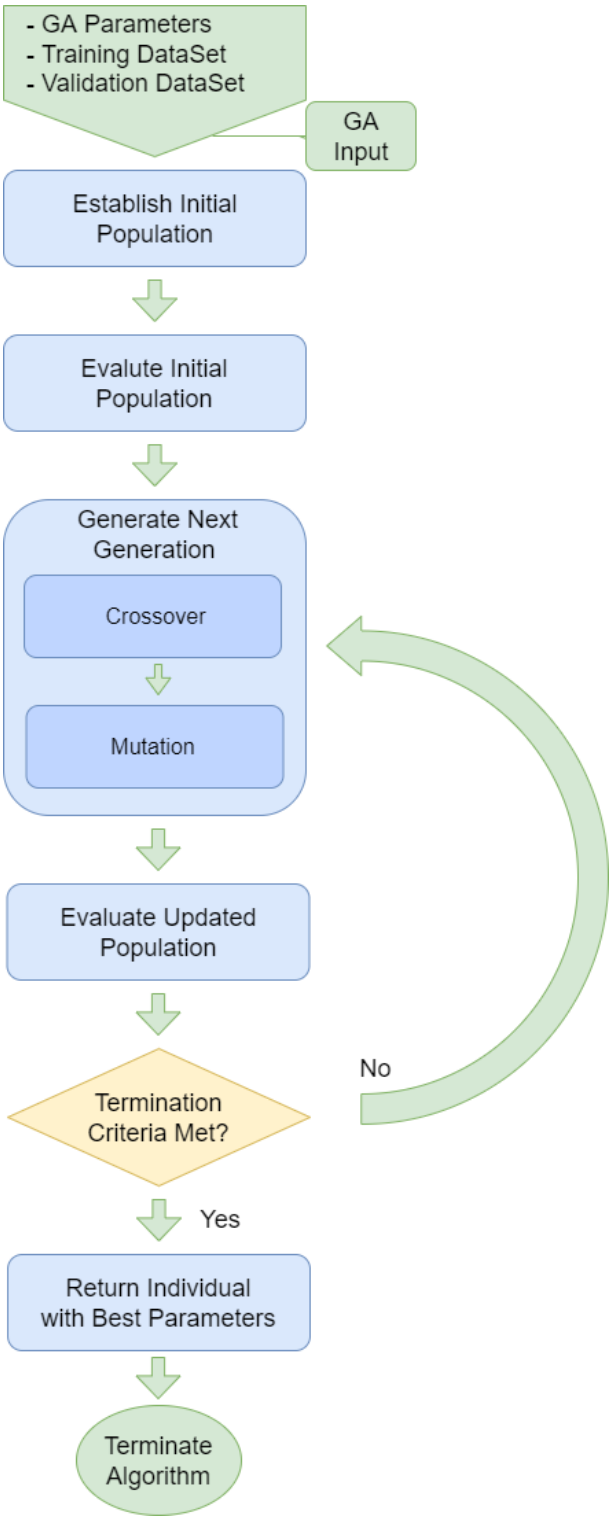

Figure 3.8: Genetic Algorithm Fluxogram

In this project, the process that served the purpose of validating the dataset, was the GA approach. Grid search, is more useful when dealing with discrete variables for features, as such, it will be used for the SVM algorithm, but in the case of XGBoost, the parameters can take a large range of continuous values, therefore the GA will be used to validate it. The random search eliminates the problem of the run time encountered by grid search, but, in the other hand, it is based in a purely random process, so

the probability to find a close-to-optimal solution would be very small.

The GA algorithm can be dividend in the following sequence of events/processes:

First the **GA parameters** are chosen in order for the algorithm to start. The parameters chosen, along with the values for which the algorithm was tested with are represented in Table 3.5.

Table 3.5: GA parameters and assigned values for optimization

Parameter Name	**Adopted values for this study**
Population Size	[50, 100]
Crossover Rate	[0.6, 0.7, 0.8]
Mutation Rate	[0.1, 0.2]
Number of Generations	[50, 100]
Fitness Type	[máx]
Crossover Type	[Two-Point, Uniform]
Selection Type	[Stochastic Universal Sampling, Tournament]
Scoring Metric	[f1score, PR-AUC, ROC-AUC]

Afterwards, a population of random individuals is created. The size of the population was initially set at 50, but was soon changed to 100 as it was showing improvements in the search of an optimal solution. Each individual is given all the parameters of the algorithm for which it is being tested, with each parameter representing different gene inside the chromosome of the respective individual and a random number is assigned to each gene, within a specific window of values depending on the parameter in question. Below, in Figure 3.9, stands a representation of a chromosome for the XGBoost algorithm.

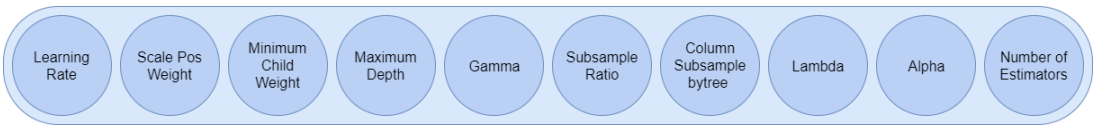

Figure 3.9: Chromossome and gene codification in the GA module for XGBoost validation sequence

The next step is to evaluate every individual inside the current population. There is a wide range of metrics from which one can choose in order to evaluate the performance of a classification algorithm, and the ones chosen in this study include the f1score, the Precision-Recall area-under-curve(PR-AUC) and the Receiver Operator Characteristic area-under-curve(ROC-AUC). The following equations eq.3.3 demonstrate how to calculate each of these metrics.

$$f1score = 2 \times \frac{Precision \times Recall}{Precision + Recall},$$

$$PR \quad Curve = Plot \quad of \quad Recall(x) \quad vs \quad Precision(y), \tag{3.3}$$

$$ROC \quad Curve = Plot \quad of \quad FalsePositive \quad Rate(x) \quad vs \quad TruePositive \quad Rate(y)$$

Precision and Recall are already two widely used metrics for evaluating the performances of ML algorithms. Although, instead of just using each of them separately, both can be maximized at the same time. This can be achieved by calculating their harmonic mean, or *f1score*, or even a Precision-Recall curve and calculating the AUC.

ROC-AUC is one the most commonly used performance metrics to evaluate a model in classification problems, mainly with binary labels, which is the situation in this project. ROC is a probability curve and by measuring the area under it, it gets the separability of the model. This means that, with a higher ROC-AUC, the model will have higher probability to predict $0s$ as $0s$ and $1s$ as $1s$.

After the initial population has all been tested against the selected metric, the best individual inside the population is saved (the fitness, and the parameters that resulted in said fitness).

The next step is the selection of the individuals that will go to the mating pool, to be part of the process of creating the new offspring, which will help populate the next generation of individuals. This selection, as mentioned in Section 2.5, can be done through several processes. The processes considered in this project were Tournament Selection and Stochastic Universal Sampling(SUS) Selection, as shown in Table 3.5.

Afterwards, when the mating pool is complete, one can proceed to the next phase. This phase serves the purpose of creating the new offspring and consists of two separate processes, crossover and mutation. The crossover methods chosen for validating the model were the uniform and two-point crossover methods that are explained in Section 2.5, The crossover and mutation phases have one variable associated with them, which in this project will take the values of $P_c = 0.8$ and $P_m = 0.1$. This two values and both the selection and crossover methods were chosen based on various experiments with many different sets of GA parameters. The set of parameters that found better solutions more often, was the one with the variables above showing those respective values, with a tournament selection and a two-point crossover. The GA would, from then on, be used only with those parameters.

When the next population is already full and ready to be evaluated, the algorithm will keep repeating all the steps above, always trying to find better and better solutions over time. The algorithm will then stop and save the best solution, when either of these two requirements are met: The maximum number of iterations are met or the algorithm is not finding a better solution for five consecutive generations.

3.9 SVM Module

As with the XGBoost module, described in section 3.7, it will also receive the data already separated and ready for validating and testing the model.

The performance of this classification algorithm, as one might expect, depends heavily on the choice of its parameters. The parameters that will be used in this project for this algorithm, are the following, and can be seen in Table 3.6.

Table 3.6: SVM parameters used and considered values

Parameter	**Description**	**Values Tested**
Kernel	The type of kernel used in the algorithm	[RBF, Polynomial]
C	Regularization parameter for the algorithm	[1, 10, 100, 1000]
Gamma	Kernel coefficient	[0.1, 0.01, 0.001, 0.0001]
Degree	Degree of the polynomial kernel funcion(only applicable in polynomial kernel).	[2, 3, 4, 5]

When in the process of calculating the optimal parameters for the SVM model, a grid search approach was used. Firstly, the two kernel were separated as the polynomial has an extra parameter associated. The models created in the grid search showed a variety of results, and were evaluated in respect to the same metrics described in Section 3.8. Below, in the Tables 3.7 and 3.8 are the results for the different kernels with various sets of parameters, when evaluating them according to the ROC-AUC evaluation metric described in Section 3.8, for the testing year 2019.

Table 3.7: Results of a grid search preformed on SVM - RBF Kernel used

	$C = 10^0$	10^1	10^2	10^3
$\gamma = 10^{-1}$	0.612	0.605	0.599	0.597
$\gamma = 10^{-2}$	0.665	0.660	0.669	0.625
$\gamma = 10^{-3}$	0.674	**0.682**	0.668	0.648
$\gamma = 10^{-4}$	0.662	0.660	0.672	0.679

Table 3.8: Results of a grid search preformed on SVM - Polynomial Kernel used

	n=2				n=3				n=4			
	$C = 10^0$	10^1	10^2	10^3	10^0	10^1	10^2	10^3	10^0	10^1	10^2	10^3
$\gamma = 10^{-1}$	0.616	0.622	0.597	0.609	0.605	0.562	0.540	0.540	0.613	0.602	0.585	0.585
$\gamma = 10^{-2}$	0.592	0.599	0.616	0.622	0.639	0.648	0.643	0.605	0.686	0.683	**0.689**	0.678
$\gamma = 10^{-3}$	0.506	0.538	0.592	0.599	0.609	0.611	0.670	0.639	0.494	0.484	0.595	0.675
$\gamma = 10^{-4}$	0.491	0.502	0.506	0.538	0.565	0.577	0.607	0.609	0.482	0.483	0.533	0.529

The best results for each of the kernels can be seen in bold in Table 3.7 and Table 3.8, and is 0.682 for the RBF kernel and 0.689 for the Polynomial kernel. Therefore the polynomial is going to be the one that is going to be used for future purposes, and with the following parameters : $n = 4, \quad C = 10^2, \quad \gamma = 10^{-2}$

3.10 Voting System

This voting system consists of receiving each models predictions, and according to each prediction's probabilities it decides on which model's prediction to take. Every data entry is treated independently, and the voting system analyzes each model's predictions, and consequently saves the prediction of the model that has a higher probability value. An example of this process is represented below in Table 3.9 for 5 distinct data entries.

Table 3.9: Functioning of the implemented Voting System

SVM		**XGBoost**		**Voting System**
Prediction	Probability	Prediction	Probability	Final Decision
1	0.945	1	0.863	1
1	0.899	0	0.736	1
1	0.986	0	0.993	0
0	0.763	0	0.837	0
0	0.803	1	0.798	0

3.11 Stock Ranking Module

This module has the function of assigning a ranking to all the companies present, such that the user gets a better understanding how the companies might behave in the future according to the predictions from the classification models. It will receive the predictions and probabilities coming from the algorithms

or the voting system and merge them with the test dataset (in order to know if a certain company, in fact, will stop presenting dividend payments in the following year), and assign a rank to each company.

This rank is associated to the probabilities that the algorithms provide to each prediction. The data is represented in quarterly statements, as so, the company's rank will be relative to the four quarters of the year that is being evaluated. The way this happens is represented in the Table 3.10.

Table 3.10: Stock Ranking System - An example on how it would preform

Stock 1		**Stock 2**		**Stock 3**		**Stock 4**	
Quarter	Score	Quarter	Score	Quarter	Score	Quarter	Score
Q1	0.932	Q1	0.959	Q1	0.932	Q1	0.846
Q2	0.930	Q2	0.906	Q2	0.899	Q2	0.903
Q3	0.990	Q3	0.924	Q3	0.851	Q3	0.912
Q4	0.903	Q4	0.927	Q4	0.947	Q4	0.899
Final Score = 0.939 Ranking = 1st		Final Score = 0.929 Ranking = 2nd		Final Score = 0.907 Ranking = 3rd		Final Score = 0.890 Ranking = 4th	

If the verbose option for this module is activated, then there will be a series of statistics that will be presented to the user of the program. These are basically, the companies that are at the top and the bottom of the ranking and the amount of companies that will leave their dividend payments at the next observation, always in percentiles chosen previously in order to facilitate the final analysis of the system.

3.12 Sliding Window Module

In the modules described above, all tests are done on a static window, which means that there is a test year, validation years and training years and these are static, i.e. they do not change and the system runs each module only once for each intended test.

However, in order to improve the performance of the system, this Sliding Window module was implemented. When this module is present, the different datasets, for all phases of the program, are no longer static. Instead, the years are incremented as the program runs (e.g., the test year starts as 2016, then becomes 2017 and so on).

The system then does a complete run where it trains, validates, and tests the presented data, and instead of stopping there, it moves on to the next year. All datasets start and finish one year later, and so on. As this happens, the predictions, and therefore the rankings as well, are merged with those of previous years, making the model more reliable with each passing iteration.

This whole process can be described, by succinctly going through all the steps involved for each iteration of the module, and it goes as follows:

- First, the module receives the pre-processed data, along with the specified test years where the predictions will take place.
- Then it proceeds to split the data according to the current test year, into all the different datasets. These datasets will be used later in the different parts of the module
- The next step is to collect the parameters values for the algorithms. In the case of XGBoost, the GA will be executed and eventually find the optimal parameters for the current window. While with SVM, the parameters are predefined, as they will be the same for every step of the process.
- After each algorithm has their parameter values assigned they will provide their predictions, and enter the Voting System module.
- In the voting System module, the final decision for the predictions will be calculated, according to each model's predictions and the respective probabilities. The final prediction will then be used for ranking all the stocks accordingly
- Later, in the case of this iteration not being the first, the current stock ranking will merge with the previous ones, thus updating the score of all stocks. In this step, it is important to emphasize that, in every iteration's ranking phase, the worst stocks get taken out of the ranking system, as they will no longer serve any purpose due to their lower scores.
- Finally, after all iterations have passed, the module will then give out the final rankings as its output.

The final results, will lead to a better and more trustworthy stock ranking, as it takes into account several years, instead of just one, like in a single window system.

Chapter 4

Results

This chapter presents all the results that the implemented system has achieved. There will be various case studies, where each will have a brief explanation on why it was performed, and some expected results, and their outcome when presented to the selected data.

The case studies defined in this project will be the following:

- **Case Study 1 -** Here, the algorithms will be tested in respect to certain metrics, in testing years that are still new to them in order to better perceive how they would behave in the following years, where the future state of the companies' dividend policies are still unknown. Then the algorithms will join each other in a voting system and be evaluated in the same metrics to see if the voting system has given the system an improvement in terms of prediction performance.

- **Case Study 2 -** The second case study serves the purpose of analyzing the predictions, and through the stock ranking module discussed in section 3.11, assign a rank to each stock and compare them according to their ranking and whether or not they broke their dividend streak.

- **Case Study 3 -** This final case study, will be similar to the combination of the ones before, but the algorithms will now be evaluated over a few years of testing. This is possible with the sliding window module described in section 3.12, and will give a more trustworthy set of results on both the models performance and the ranking of the considered stocks.

This different case studies have the purpose of showing how this system behaves in a series of different situations, so as to be able to get conclusions on how a voting system could improve single-algorithm systems.

4.1 Case Study 1 - Single Algorithm System vs Voting System in Single Window

The first part of this case study is to analyze how each of the algorithms performs in respect to the metrics chosen, and consequently show the improvements of the voting system in comparison to just using each algorithm separately. As shown in Sections 3.9 and 3.7, the algorithms had their parameters chosen with the aid of a grid search and a GA respectively. Afterwards, with the set of parameters chosen for each, they went through a series of performance metrics such as *F-score*, *PR-AUC* and *ROC-AUC*.

The results for both algorithms and also for the implemented voting system, are represented in Table 4.1. Firstly, a comparison between both algorithms performance is needed and both of their ROC curves can be observed below in figures 4.1 and 4.2 for SVM and XGBoost

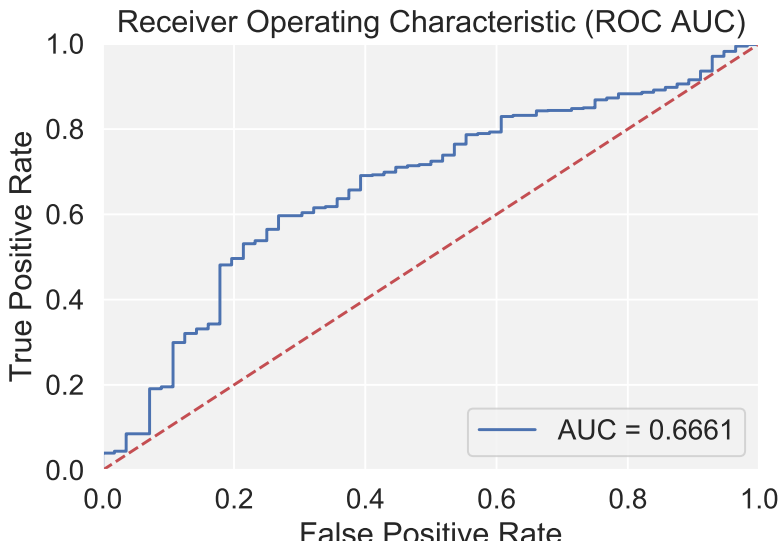

Figure 4.1: ROC-AUC Graphic of SVM predictions - Testing Year = 2017

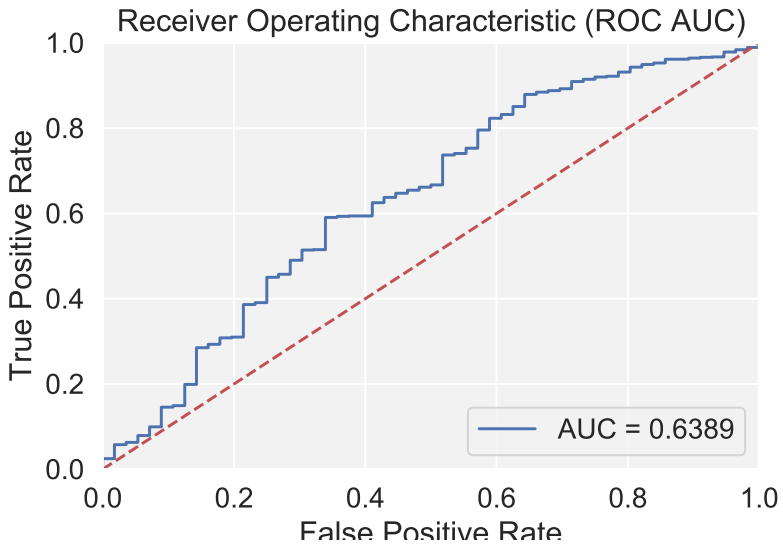

Figure 4.2: ROC-AUC Graphic of XGBoost predictions - Testing Year = 2017

It can already be observed which algorithm showed better results in the chosen testing year, and in this case was the SVM, as it presented a better ROC-AUC value than XGBoost. The testing year was chosen randomly from a selection of years. In the year 2017, the SVM might have better results, but in a different year, XGBoost could have a better performance, as the data that defines the performance of both algorithms varies for each year. This is the main reason why the voting system was implemented, as it can get the best predictions from both algorithms and create a better predictor. The resulting predictions from the voting system generated a ROC curve that can be observed in Figure 4.3

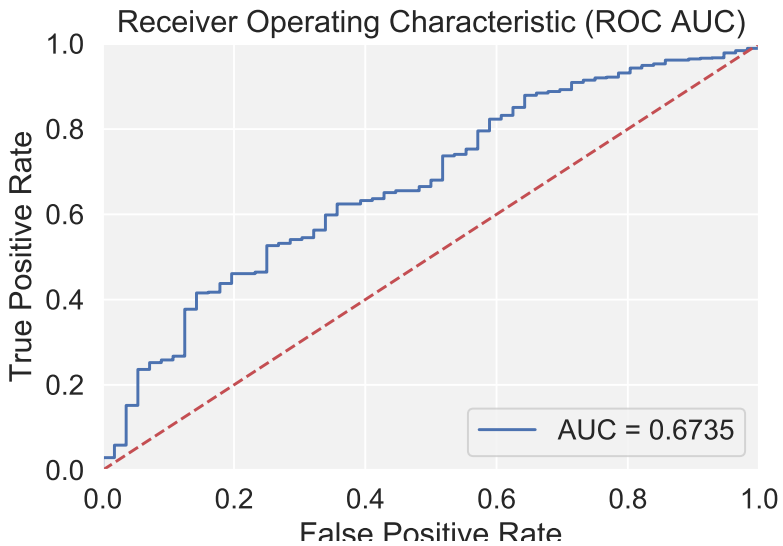

Figure 4.3: ROC-AUC Graphic of the Voting System predictions - Testing Year = 2017

As we can see from the Figure 4.3, the ROC curve resulting from the voting system, although it may not show as good an improvement as expected, it still outperforms the algorithms when trying to predict the results independently, which is a good result overall. Below in Table 4.1 are represented the performances of both algorithms an the voting system.

Table 4.1: Performances of both algorithms in comparison to the Voting System

SVM			**XGBoost**			**Voting System**		
F-Score	ROC-AUC	PR-AUC	F-Score	ROC-AUC	PR-AUC	F-Score	ROC-AUC	PR-AUC
0.9708	0.6661	0.9725	0.9635	0.6389	0.9678	**0.9721**	**0.6735**	**0.9741**

By analyzing this table we can see that the ROC-AUC is not the only metric where the voting system clearly presents superior results. Both in F-Score and PR-AUC, the algorithm has a better performance which means that it has an easier time recognizing when an observation is actually positive or negative, thus having fewer false positives and false negatives, which is quite important in the problem presented, consequently, it will show a better separability of classes (which in this case are only two since it is a binary classification system), which removes some of the error when predicting future values.

4.2 Case Study 2 - Stock Ranking with Voting System

In this case study, we will evaluate the companies present in the dataset, and use the stock ranking module (described in Section 3.11) in order to get a better understanding of which are the best companies in terms of future continuation of the dividend payout increase. The metric used to better

optimize the parameters of the XGBoost was ROC-AUC, and this was the most regularly used metric in this project, as it always showed better results than the others. As to the SVM parameters, the same ones are used in all the studies performed, seeing that the grid search used to calculate the optimal set of parameters has been validated and trained in many different situations, already with the aim of being used in the following processes.

The main part of this case study is to find the best ranked companies. After this process, the information obtained by the system served to construct a bar chart so as to get a better understanding of how the results would vary for these companies and whether any would in the future leave their dividend streak.

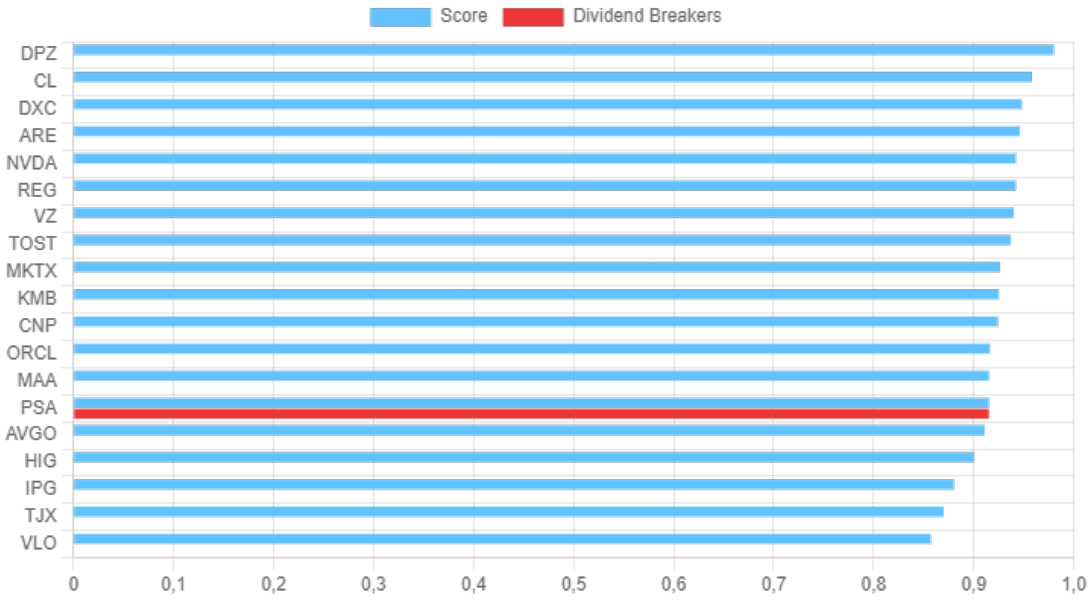

Figure 4.4: Single Window - Top 20 Stocks with corresponding Ranking - Stocks that brake their dividend the following year are signaled with a red bar

The results did not deviate much from what was expected, as the companies do indeed score high and there is not much variation, which would be expected from the companies placed at the top of the rank, given the percentage it represents in the total number of companies processed and analyzed. There is, however, one company (PSA) that will break its dividend streak the following year, which should not happen to one that is so high in the total ranking.

Even so, considering the percentage it represents, one company does not carry that much weight in the total system analysis. However, an analysis of the financial data of this company is a process that should be carried out in order to try to understand why this outlier was present in the section of the top ranking stocks analyzed.

It was verified that, even though PSA has broken its dividend increase streak, it has maintained its dividend payments until now (2021) without ever reducing or completely withdrawing them, which means that the red flag that it appeared to be is not as serious as it could be. However, it still means that the

system implemented can be improved, and way to do so is to perform a more thorough analysis on these red flags, in order to try to understand why the algorithm decided the way it did, and finally, attempt to correct these mistakes.

The next companies that are going to be analyzed are the ones on the bottom of the ranking system. The voting system has ranked these companies so poorly in order to show that investments made in these companies are not so reliable, even if they continue to pay dividends at the next observation. It means that one, or several, financial ratios are at values that the system associates with companies that will break their streak of dividend payments.

Below, in Figure 4.5, is the graph representing these 20 companies, and as it is expected, more than one will break its streak at the next observation.

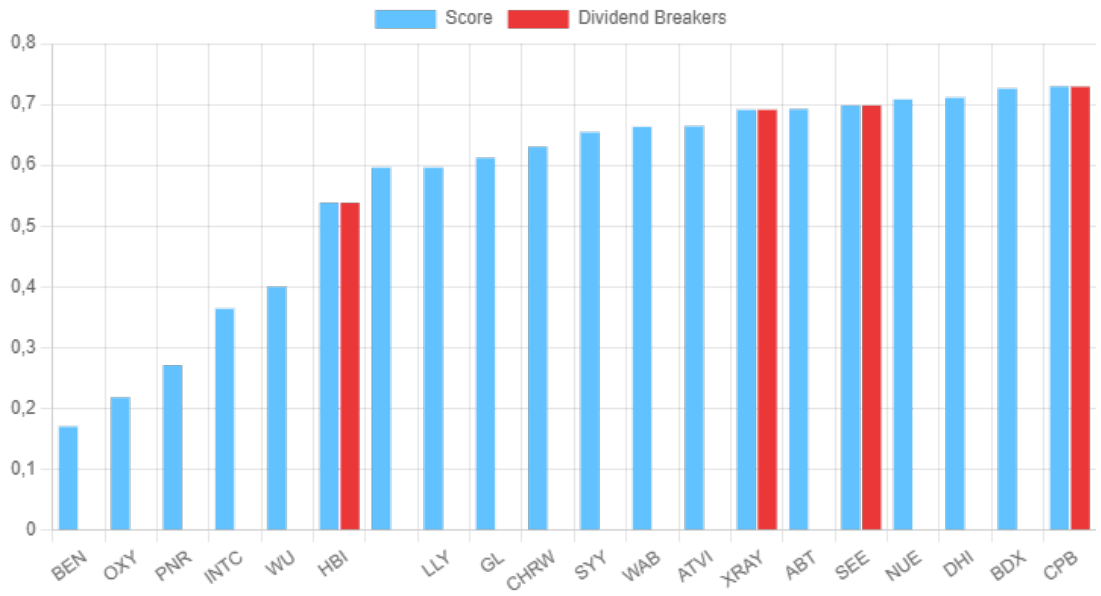

Figure 4.5: Single Window - Bottom 20 Stocks with corresponding Ranking - Stocks that brake their dividend the following year are signaled with a red bar

After gathering the top 20 stocks, the next step is to analyze whether investing in those stocks is a good investment or not. In order to evaluate how good an investment is, the ROI(Return on Investment) is going to be calculated for a portfolio with all those companies, in the year corresponding to the one used in testing. Additionally, not only will the ROI for the entire year be calculated as the one for a 6 month period, in order to perceive the evolution of the ROI.

In order to evaluate this investment the ROI will be compared to the one of the S&P500 for the same year, to understand if the voting system is a viable option, or if it is redundant because it shows worse results than that of S&P500 investing. The ROI values for both investment strategies are represented in Figure 4.6 below.

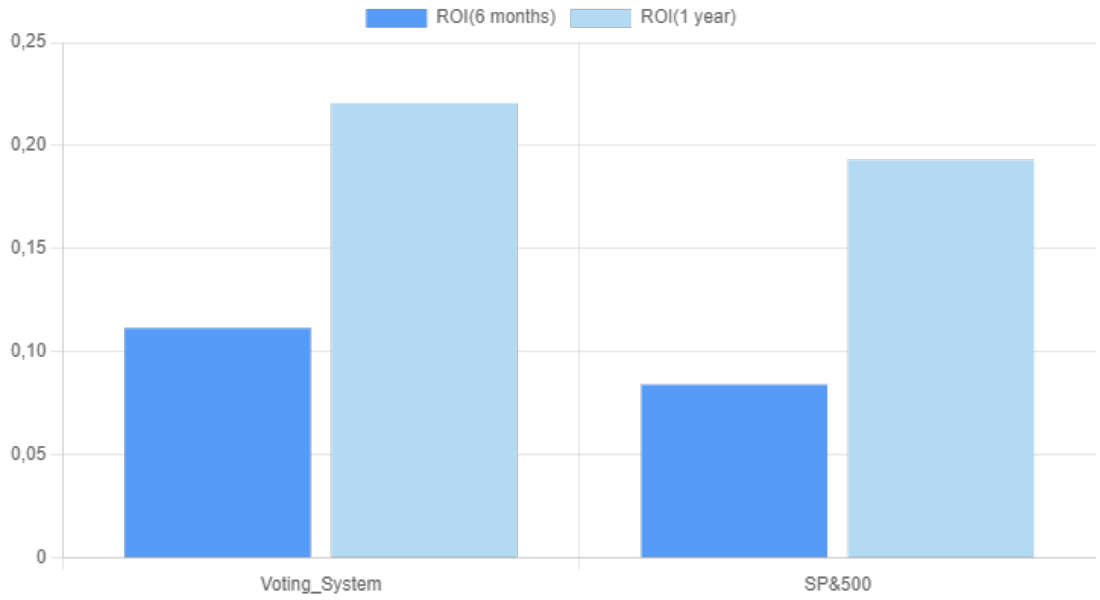

Figure 4.6: Single Window - ROI comparison for the Voting System strategy vs Classical S&P500 typical investment strategy

As it can be observed, the voting system investment strategy, outperforms the classical S&P500 investment. This is a positive result as it shows the viability of the voting system.

In the next case study, the voting system will be tested in a sliding window, in order to see if it can improve its current performance.

4.3 Case Study 3 - Sliding Window Stock Ranking and Performance Analysis

In this last case study, the voting system will again make its predictions, thus building a ranking of the companies. However, now, instead of just doing it once, it will do it several times sequentially, within a sliding window. This method is described in Section 3.12, and will consist in repeating the previous process of ranking the stocks, adding in each iteration a new ranking to the previous ones, always calculating the average of the rankings according to a SMA(Simple Moving Average).

The sliding window used in this project is in respect to three evaluation years ([2017, 2018, 2019]). The metrics used to optimize the parameters of the XGBoost algorithm was again the ROC-AUC, as it appeared to lead the algorithm in a better direction, performance-wise, and again the SVM parameters will stay as in the previous examples.

This method, was proposed, as it should turn the predictions into a more reliable source for future investment strategies. In a sliding window, the system continuously runs the modules needed to build the rankings and the worst ranked stocks will begin to leave the ranking completely, in order to discard any stock, that at any point in time appeared to be inconstant and unreliable. The top 20 stocks can be

seen in Figure 4.7.

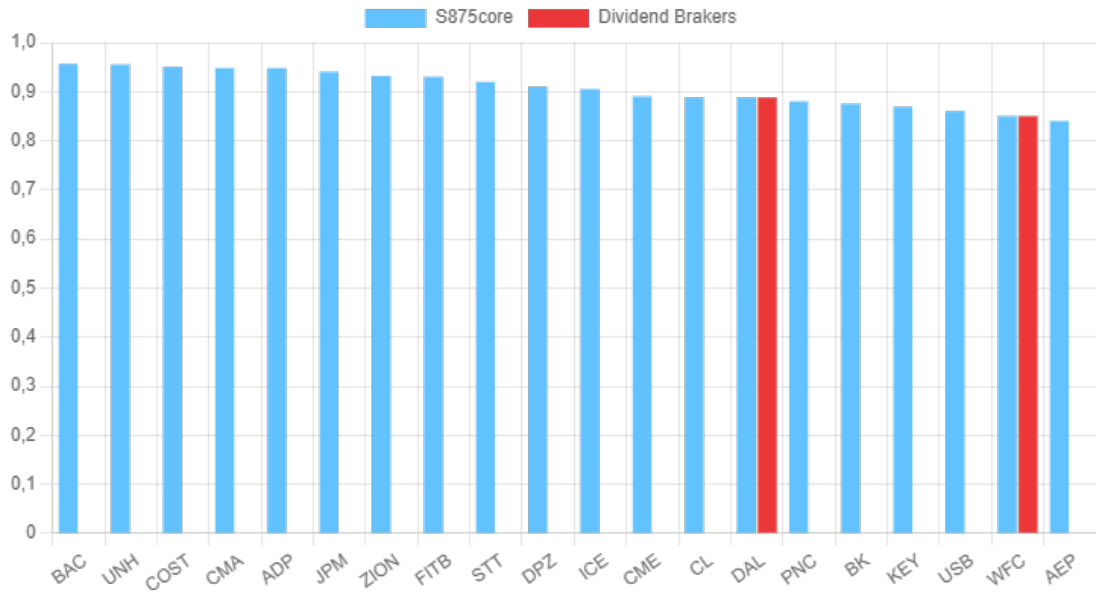

Figure 4.7: Sliding Window - Top 20 Stocks with corresponding Ranking - Stocks that brake their dividend the following year are signaled with a red bar

The resulting stock ranking does not appear to be what it should. There are two of the top 20 best ranking stocks that will in fact break their dividend streak, therefore, they should not be here in the top 20 best stocks. The sliding window was supposed to show more reliable results, but the graph in Figure 4.7 shows exactly the opposite. This graphic should have looked better than the one in Figure 4.4, but as one can observe, it does not.

As it is expected, accordingly to the graphic of this sliding window module, in Figure 4.8, there is evidently some major downside to using this system, something that was not at all expected. In this case study, the S&P500 investing strategy outperforms the voting system with a sliding window, therefore, the system implemented can not be considered as a major investing tool, as someone could just simply invest in the S&P500, and get better results, with a smaller risk.

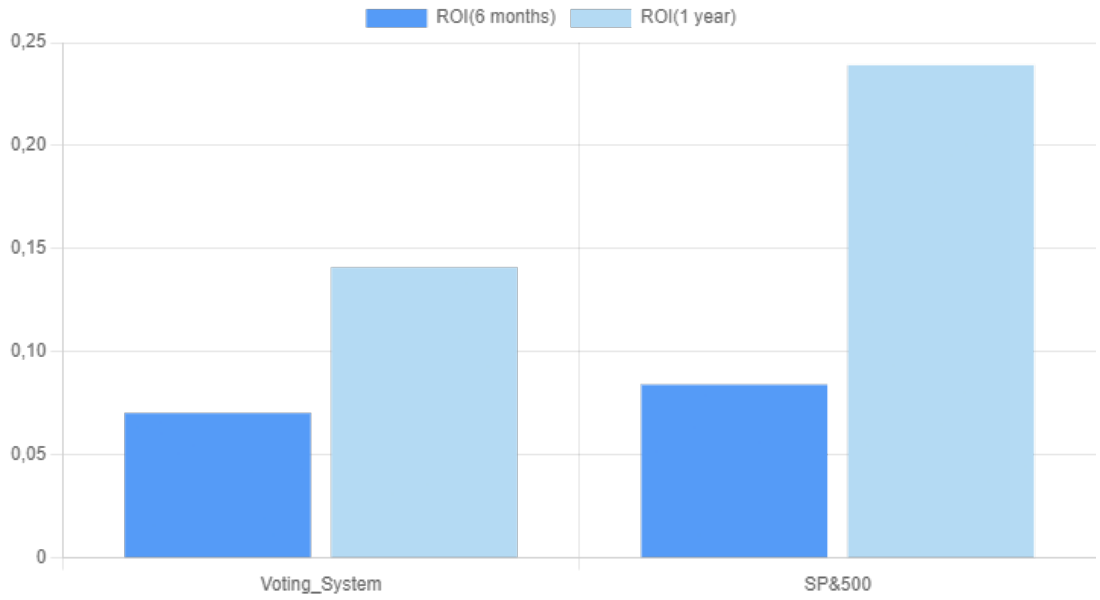

Figure 4.8: Sliding Window - ROI comparison for the Voting System strategy vs Classical S&P500 typical investment strategy

This case study did not go as well as it should have. There are some improvements that need to be done in order for this system to start showing some better results. improvements will be discussed in Section 5.2, but as of now, this system is not a very viable option.

Chapter 5

Conclusion and Future Works

5.1 Conclusion

The final system presents a voting system composed of the SVM and XGBoost algorithms, where parameter optimization is done through a Genetic Algorithm and a Grid Search respectively. This system was mainly tested in the years 2017, 2018 and 2019 and trained/validated in previous years, since 2008 onward.

The algorithms received datasets composed of the financial data of the companies that make up the S&P500, which are previously taken from the CRSP/Compustat and macrotrends databases. The data fed into the algorithms was used to train them, so as to create a model that will predict continued increases in dividend payments. The algorithms also took part in a voting system that provided the system's final predictions.

The algorithms chosen appeared to be very useful for the problem presented, but given the multitude of studies already done with these algorithms, a voting system was suggested where both algorithms could make their predictions and decide which one to use for the final prediction of the system.

The genetic algorithm used to optimize the XGBoost parameters showed a very good ability to find parameters that maximized the performance of the algorithm for several different testing years. Additionally, the grid search method showed some good results, but still seems to be an aspect that can be improved.

The implemented voting system showed improvements over using algorithms individually, which was the expected result back in the beginning of this project. However, the sliding window module fell short of its expectations as the system did not behave as expected.

To conclude, the implemented system showed some good classification ability, but it can still be further optimized, considering possible improvements that will be discussed in the next section.

5.2 Future Work

As always, there are some improvements that could be done so as to improve the results of this system. This improvements could bring forth a better classification model in order to reduce the classification error, and consequently improve the ROI. The improvements considered for this project's future work are the following:

- The first improvement that could be implemented is bringing a GA approach into the SVM's parameters optimization. The grid search would still be used, with a short list of values for each parameter, and sequentially, a GA could be applied in order to search for a more optimal set of parameters in the feature space around the parameters given by the grid search. This would turn the SVM algorithm more adaptable to changes in the dataset, as it could change according to the occurring data variations, specially when changing the testing years.

- A fairly obvious improvement that could be done to this system is adding more different algorithms to the voting system. The current voting system already showed some improvements, and adding

a few more, could only help the system adapt, and make better predictions over time. An algorithm that was considered for this system but ended up not being used is the Random Forrest. It also showed good results in previous studies, so it should be a good addition to the system. Additionally more algorithms could be added, so as to reduce the errors that every algorithm will end up having in some situations.

- Finally, another good improvement would be to add more labels to the dataset. This way, it could give a better understanding on a few more important events by attempting to predict them. This could help get more information on how the companies would behave in the future, therefore, increasing the user´s financial knowledge, and consequently make more thought-out decisions when deciding on which investment opportunities to take, therefore increasing the probability to have better returns.

Bibliography

[1] M. Clemens. Dividend investing: Strategy for long-term outperformance. 2012.

[2] M. Lichtenfeld. *Get Rich with dividends: A proven system for earning double-digit returns*. John Wiley & Sons, 2015.

[3] A. Wafi, H. Hassan, and A. Mabrouk. Fundamental analysis vs technical analysis in the egyptian stock exchange – empirical study. *International Journal of Business and Management Study – IJBMS*, 2:212 – 218, 10 2015.

[4] M. Buffet and D. Clark. *Warren Buffet and the Interpretation of Financial Statements : The Search for the Company with a Durable Competitive Advantage*. Springer, 2^{nd} edition, 2006. ISBN:978-0387303031.

[5] S. Baresa, S. Bogdan, and Z. Ivanovic. Strategy of stock valuation by fundamental analysis. *UTMS Journal of Economics*, 4(1):45–51, 2013. ISSN 1857-6982.

[6] V. Nasteski. An overview of the supervised machine learning methods. *HORIZONS.B*, 4:51–62, 12 2017. doi: 10.20544/HORIZONS.B.04.1.17.P05.

[7] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939785.

[8] S. K. Satapathy, S. Dehuri, A. K. Jagadev, and S. Mishra. Chapter 1 - introduction. In S. K. Satapathy, S. Dehuri, A. K. Jagadev, and S. Mishra, editors, *EEG Brain Signal Classification for Epileptic Seizure Disorder Detection*, pages 1–25. Academic Press, 2019. ISBN 978-0-12-817426-5. doi: https://doi.org/10.1016/B978-0-12-817426-5.00001-6.

[9] V. Vapnik. Support-vector networks. *Machine Learning*, 1995.

[10] A. Mammone, M. Turchi, and N. Cristianini. Support vector machines. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(3):283–289, 2009.

[11] R. N. Vasco Amaral. Hybrid system combining artificial neural networks and support vector machines for trading the forex market intraday. Master's thesis, Instituto Superio Técnico, 2021.

[12] L. Scrucca. Ga: a package for genetic algorithms in r. *Journal of Statistical Software*, 53(1):1–37, 2013.

[13] E. F. Fama and K. R. French. Disappearing dividends: changing firm characteristics or lower propensity to pay? *Journal of Financial Economics*, 60(1):3–43, 2001.

[14] H. DeAngelo, L. DeAngelo, and D. J. Skinner. Are dividends disappearing? dividend concentration and the consolidation of earnings. *Journal of Financial Economics*, 72(3):425–456, 2004.

[15] F. Allen and R. Michaely. Chapter 25 dividend policy. In *Finance*, volume 9 of *Handbooks in Operations Research and Management Science*, pages 793–837. Elsevier, 1995. doi: https://doi.org/10.1016/S0927-0507(05)80069-6.

[16] D. Javakhadze, S. P. Ferris, and N. Sen. An international analysis of dividend smoothing. *Journal of Corporate Finance*, 29:200 – 220, 2014.

[17] M. Benlemlih. Corporate social responsibility and dividend policy. *Research in International Business and Finance*, 47:114 – 138, 2019.

[18] K. Krieger, N. Mauck, and S. W. Pruitt. The impact of the covid-19 pandemic on dividends. *Finance Research Letters*, page 101910, 2020.

[19] M. Berre. *Investing in dividend growth stocks: analysis of portfolio performance using asset pricing models*. PhD thesis, 2021.

[20] S. Han and R.-C. Chen. Using svm with financial statement analysis for prediction of stocks. *Communications of the IIMA*, 7(4):8, 2007.

[21] Y. Ding, X. Song, and Y. Zen. Forecasting financial condition of chinese listed companies based on support vector machine. *Expert Systems with Applications*, 34(4):3081–3089, 2008. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2007.06.037.

[22] J. K. Bae. Forecasting decisions on dividend policy of south korea companies listed in the korea exchange market based on support vector machines. *J. Convergence Inf. Technol.*, 5(8):186–194, 2010.

[23] Y.-P. Huang and M.-F. Yen. A new perspective of performance comparison among machine learning algorithms for financial distress prediction. *Applied Soft Computing*, 83:105663, 2019. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2019.105663.

[24] S. Ozlem and O. Tan. Predicting dividend payout policy of turkish firms using xgboost and mlnn algorithms. In *8th. INTERNATIONAL MANAGEMENT INFORMATION SYSTEMS CONFERENCE*, 2021.

[25] L. Shimin, X. Ke, Y. Huang, and S. Xinye. An xgboost based system for financial fraud detection. In *E3S Web of Conferences*, volume 214, page 02042. EDP Sciences, 2020.

[26] CRSP/Compustat. URL https://www.crsp.org/products/research-products/crspcompustat-merged-database.

[27] Y. Finance. URL https://finance.yahoo.com/.

[28] R. N. Francisco Silva. Selection of sustainable dividend stocks combining xgboost with genetic algorithm. Master's thesis, Instituto Superio Técnico, 2020.

[29] Y. Xu and R. Goodacre. On splitting training and validation set: a comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. *Journal of analysis and testing*, 2(3):249–262, 2018.

[30] D. Falessi, J. Huang, L. Narayana, J. F. Thai, and B. Turhan. On the need of preserving order of data when validating within-project defect classifiers. *arXiv preprint arXiv:1809.01510*, 2018.

[31] G. Luo, S. He, B. L. Stone, F. L. Nkoy, and M. D. Johnson. Developing a model to predict hospital encounters for asthma in asthmatic patients: secondary analysis. *JMIR medical informatics*, 8(1): e16080, 2020.

[32] P. Liashchynskyi and P. Liashchynskyi. Grid search, random search, genetic algorithm: A big comparison for nas, 2019. URL https://arxiv.org/abs/1912.06059.

[33] Q. Wang. Using genetic algorithms to optimise model parameters. *Environmental Modelling Software*, 12(1):27–34, 1997. ISSN 1364-8152. doi: https://doi.org/10.1016/S1364-8152(96)00030-8.