Gonçalo Capela Sanches Pereira da Silva

Licenciado em Engenharia Informática

Decision Support System for Investment Analysis

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática

Orientador:	Carlos Gomes, Mestre em Economia e Finanças ISCTE e NOVA-SBE & Founding Partner GoBusiness Finance, GoBusiness Finance
Co-orientador:	Nuno Cavalheiro Marques, Professor Auxiliar, Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa

Júri

Presidente:	Doutor Pedro Abílio Duarte de Medeiros
Arguentes:	Doutor Miguel Jorge Tavares Pessoa Monteiro
	Mestre Carlos Joaquim da Costa Gomes

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2019

Decision Support System for Investment Analysis

Este documento foi gerado utilizando o processador (pdf)LaTeX, com base no template "novathesis" [1] desenvolvido no Dep. Informática da FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

Tremendous learning. To my future self.

Acknowledgements

I would like acknowledge my thesis coordinators and advisors, Professor Nuno Cavalheiro Marques and Carlos Gomes for their tremendous support, shared knowledge and time dedicated to this thesis. Not only did they help complete this project but have shared competences that I will carry into my career and personal life. I would also like to thank Carlos Gomes, João Maria Vasconcelos and all the people that work at GoBusiness for the opportunity to work there and learn from them. I need to specially thank my Mother, Father, Brother and Grandmother for providing me with their support and patience throughout this project. Lastly, but remarkably important, I want to thank my closest friends and girlfriend, for their support, patience, conversations and opinions throughout the development of this project.

Abstract

The purpose of this thesis lies on selecting and automating a set of Fundamental Analysis indicators and studying related software tools that can help investors understanding market behaviour. The several distinct data-sources, tools and methods will be evaluated using a Decision Making process for Financial Markets.

Sometimes there's not enough data in which we can base the investment decision upon, other times the data lacks quality, while other times, despite having the right data, the problem lies on the process of analyzing the data and then turning that analysis into a concrete decision. Also, since the human decision making process is not well systemized, there are times when both the data and the analysis are well performed, but the results may vary even when confronted with similar data patterns more than once. This is particularly crucial when dealing with fast-paced environments like the Financial Markets. This thesis will therefore study tools for systemizing a Decision Making process based on fundamental analysis indicators over financial markets and will evaluate how such tools help to avoid uncertainty in human decision and to complement lack of data and poor data quality. There are two essential building blocks of such a system: the data set and the model that analyses the data and ultimately, provides information that facilitates the decision making process about a particular investment. Both blocks will be made available in the framework of the research project at GoBusiness Finance.

Keywords: Fundamental, Investment, Analysis, Decision, Support, Systems, Investment, Tree, Markets, Regression

Resumo

O propósito desta dissertação reside na selecção e sistematização de um conjunto de indicadores financeiros para Análise Fundamental, assim como, o estudo de ferramentas que possam ajudar investidores a terem um melhor entendimento do segmento das acções dos Mercados Financeiros. Por vezes, não existe informação suficiente sobre a qual possamos basear as decisões de investimento, por outrem, existem vezes em que a informação existe, mas a qualidade da mesma não pode ser comprovada. Também acontecem casos em que, apesar de possuirmos a informação adequada, o problema recai no processo de análise da informação e na subsequente tomada de decisão. Para além das questões relacionadas com informação, existe também o facto de o processo de decisão desempenhado pelos humanos não ser bem sistematizado. Assim, podem surgir ocasiões em que as decisões resultantes são distintas, mesmo quando confrontados com padrões de informação e resultados de análise semelhantes. Isto é particularmente importante quando lidamos com ambientes em que as decisões são tomadas de forma tremendamente rápida, como é o exemplo dos mercados financeiros. Com isto, esta tese irá estudar ferramentas para sistematizar o processo de tomar decisões relativas a investimentos nos mercados, com base em princípios análise fundamental. Existem duas componentes essenciais para a construção de um sistema de apoio à decisão: o data set e os modelos de análise ao mesmo. Ambas as componentes serão estudadas e disponibilizadas em âmbito empresarial na Gobusiness Finance.

Palavras-chave: Análise, Fundamental, Investimento, Suporte, Decisões, Árvores, Regressão, Mercados

Contents

List of Figures

List of Tables

Listings

Glossary

Asset
Product or resource with economic value.

Balance Sheet
A Balance Sheet is a financial statement that reports a company's assets, liabilities and shareholders' equity at a specific point in time, and provides a basis for computing rates of return and evaluating its capital structure.

Bloomberg Terminal
The Bloomberg Terminal is a software system provided by the financial data vendor Bloomberg L.P. that was created to provide professionals in the financial services sector with tools to monitor and analyze real-time financial market data..

Cash Flow Statetment
The Cash Flow Statement is a financial statement that summarizes the amount of cash and cash equivalents entering and leaving a company.

Decision Tree
A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

Decision Support Systems
Information system that aids in the process of decision making.

Equity
Equity is a measure of the amount of money that would be returned to a company's shareholders if all of the assets were sold and all of the company's debt was paid.

ETF
ETF stands for Exchange Traded Fund, and is a collection of securities (e.g. Stocks) that tracks an underlying index such as the S&P500. An ETF is quoted in the stock market. These are one of ways of achieving a diversified portfolio at a lower cost compared to other funds. ETF's can "track" a group of stocks.

Feature

Distinctive attribute or aspect of a dataset.

Fundamental Indicator

Similarly to the previous definition of *indicator*, a fundamental indicator is simply an indicator that focuses on fundamental analysis. Typically, fundamental indicators, rather than being strictly focused on price, are more related to other metrics such as: revenue, earnings, debt, expenses, cash flow and many others..

Fundamental Analysis

Method for evaluating a security to assess its intrinsic value.

Income Statment

An Income Statment primarily focuses on the company's revenues and expenses during a particular period.

Indicator

"Indicators are statistics used to measure current conditions as well as to forecast future financial or economic trends"..

Investment Manager

An investment manager is a person or company that makes investments in portfolios of securities on behalf of clients.

Investment Fund

An investment fund is a collection of capital that belongs to a group of investors and that is used to collectively purchase assets such as stocks(or others)..

Liabilities

A liability is an obligation of a payment of one company to another(e.g. bank).

Machine Learning

Branch of artificial intelligence that focuses on data analysis and automation of analytical model building.

Pruning

Pruning is a technique in machine learning and search algorithms that reduces the size of decision trees by removing sections of the tree that provide little power to classify instances.

Publicly traded companies

This is company whose ownership is represented via shares of stock which are intended to be traded on the capital markets.

Regression Tree Regression tree is a type of decision tree where the predicted outcome can be considered a real number.

Revenues Revenues is a term that represent all types of income that a business has from the sale of goods and services to customers.

Stock market index A stock index or stock market index is a measurement of a section of the stock market. It is computed from the prices of selected stocks (typically a weighted average).

Stock Market The stock market refers to the collection of markets and exchanges where the regular activities of buying, selling and issuance of shares of publicly held companies take place.

Technical Analysis Method employed to evaluate investments and identify trading opportunities.

Chapter 1

Introduction

1.1 Motivation

The financial market is a broad term describing any marketplace where trading of securities including equities, bonds, currencies and derivatives occurs. The purpose of a financial market is to set prices for global trade, raise capital, and transfer liquidity and risk. That said, the financial market has also, for a long time been a way for investors to allocate their capital with the intent of obtaining capital gains. Almost ever since the financial markets were established, investors have been trying to develop models, strategies and algorithms that could beat the market, however that has proven to be a very difficult task. Nevertheless, algorithmic based investing has proven to have a lot of advantages, namely the removal of the emotional factor inherent to all human analysis, and perhaps even more important, the ability to process large amounts of data a number of times faster a human ever could. There are a number of examples of individuals that have built some of the biggest investment funds entirely based on algorithmic decision support systems. These types of funds are understandably very secretive, as they do not want to disclose their algorithms and methods of investing. There is however one famous example of such a fund: Renaissance Technologies [66, 68], which as of 2018 had 84 billion dollars of assets under management [67], and is increasingly hiring mathematicians, physicists and computer science professionals.

It is becoming more apparent that **Data** is one of the most important commodities of our era. This also coincides with the fact that there has never been a time when there is more data available. In the financial industry, that can be attributed to tools like the **Bloomberg Terminal** and a variety of other tools that provide historical and real-time data about the economy and capital markets. This amount of data creates a need for methods and tools that allow professionals to select and aggregate the most important information that will allow them to solve problems more efficiently.

1.2 Objectives

This thesis will be developed in partnership with a financial consultancy company **GoBusiness** [33]. The philosophy shared with clients of the firm is that they should not look at short term pricing and trading swings but rather, they should take interest in a long term investment approach, as it may take time for the intrinsic value to be realized. This method of investing is usually referred to as begin **"Long"** in the market.

This thesis aims to study how a decision support system, and in particular, a DSS that includes decision trees as a model, could be applied to the Financial Markets. Specifically, what kind of benefit would an investment manager get out of using a DSS to aid them in the analysis of possible investments. Typically, investors use enterprise tools such as Microsoft Excel, as their analysis and storage tool, but as with many other industries, it is becoming increasingly important to develop and apply systems such as a DSS in order to increase speed and efficiency. Investing is increasingly becoming a data science affair and with that, investment managers might take interest in taking advantage of some of the tools and techniques that are available. This is some of what this thesis aims to show.

A decision support system can be sustained by a variety of tools and models. In this case, we are interested in making use of a **Machine Learning** model called a **Decision Tree** and developing a type of workflow that would allow an investment manager to use this model, extract valuable information from it, apply that gathered knowledge and verify how an investment strategy based on the information would have performed in the past. This process is called backtesting and will be executed using a particular platfform described later on. Several fundamental indicators that fit the strategy of value focus funds will be identified. In order to enable the study of these indicators, we will explore ways of automating the process of applying the collected data to the chosen models for fundamental valuation of publicly traded companies.

After the study of the *Fundamental Indicators* is complete, the aim is to identify tools that allow the automatic management of the data set, fit this data to the decision tree model, devise an investment strategy that is based on the information provided by the tree and lastly analyse the results from these backtests.

It must be noted that satisfactory performance in the past does not guarantee similar results in the future, but such tests are a standardized starting point for testing in the financial industry. Therefore, system evaluation will also be cross-checked with the knowledge from domain experts and financial analysts from the partner investment funds. This thesis aims to investigate the following research questions:

1. What are the benefits of automating the investment decision process?

2. Is it helpful to design a schema that accommodates the collected data?

3. How should we deal with pre-processing of financial data?
4. How can regression trees be used for analysis of financial data provided by publicly traded companies?
5. How valuable is the information provided by a research and backtesting system like Quantopian?
6. How should we deal with the problem of scalling data when using decision trees?
7. What is the future of **Decision Support Systems** in the capital markets?

1.3 Thesis Structuring

This first chapter emphasizes some of the objectives of this thesis. The second chapter focuses on introducing some of the financial concepts necessary to undestand the domain of this project and start working on the thesis, literature review and state of the art studies. Chapter 3 focuses on some of the tools and methods that will be frequently used and further studies during the development of the system. Chapter 4 focuses on detailing the proposed system and some of the decisions made about implementing this system. Chapter 5 will focus on reporting and analysing the results obtained from the tests performed. Lastly, chapter 6 provides some concluding thoughts as well as a few suggestions for future work regarding this type of system and for integrating machine learning in the financial markets in general.

Chapter 2

Related Work

2.1 Initial Remarks

2.1.1 Capital Markets

Capital Markets refers to a system where the objective is to perform a variety of activities that gather funds from different entities and distribute them to other entities needing funds. The core function of these types of markets is to improve the efficiency of transactions so that each individual entity does not need to do research and analysis, create legal agreements to complete transactions of funds [47]. They also intend everyone with the opportunity to invest in a variety of assets. These assets are distributed through different types of capital markets(or Financial Markets):

- Stock Markets
- Bond Markets
- Money Markets
- Commodities Markets

This thesis is primarily concerned with the analysis of the **Stock Market**, which is comprised of various *publicly traded companies*. The earliest record of traded securities in the **New York Stock Exchange** goes back to May 17, 1792 [85] and since that very moment, investors have used these markets has a place to allocate capital in order to originate a return. In order to accomplish that, more and more sophisticated methods have been developed in order to provide investors with any type of hedge in the market.

2.1.2 Concepts

- Markets
  The purpose of financial markets is to set prices for global trade of various assets such as stocks. Besides setting prices, markets are also used as a tool to raise capital, transfer liquidity and risk [92] .

- Asset
  An asset is a product or resource with economic value that an entity, such as an individual, company or country, controls with the expectation that it will provide a future benefit [46].

- Stocks
  A stock (can also be referred to as "share"or "equity") is a type of security or asset that represents part ownership in a company. This means there could be a claim for part of the corporations assets and earnings [14].

- Index
  A stock market index is a hypothetical portfolio of stocks that represent a segment of the stock market. It is essentially used as metric of performance for a segment of the market. The calculation of the index value is based on its underlying holdings [91]. **DOW 30** (Dow Jones Industrial Average) and **S&P 500** (Standard & Poor's 500 Index) are two examples of major stock indices. The Dow expanded to 30 companies in 1928, where it remains today, but its composition changes regularly as stocks and the industries it represents fall in and out of favor [12]. The S&P 500 is composed of the 500 largest U.S companies by market-value. The latter is regarded as the the best gauge of the large U.S companies [45].

- Valuation Models
  Valuation Model or Valuation Method is a defined approach and calculation to estimate the value for a company and its stock(or other type pf security) [57]. This thesis focuses mainly on *fundamental analysis models*. "A key assumption of any fundamental value technique is that the value of the security (in this case an equity or a stock) is driven by the fundamentals of the firm's underlying business at the end of the day" [28].

Even though every investor has his/her own method for investing, there are two major schools of thought when it comes to approaching the markets [58], they are **fundamental analysis** and **technical analysis**.

2.1.3 Fundamental Analysis

Fundamental analysis is the approach whereby the investor tries to determine the intrinsic value of a company by looking at the basic economic factors and the fundamentals of that company. Namely [31, 32, 58]:

- Revenues, expenses and income;
- Growth prospects for the company;
- The competitive factors the company faces;
- Expected return on equity or assets in the industry;

The goal of this type of analysis is to establish a value for the stock that would factor in all of these underlying factors. As the approach does not look at short term pricing and trading swings, this is considered a long term investment approach, as it may take time for the intrinsic value to be realized [32, 47]. This type of analysis is typically based on three sources of information [39], and figure 2.1 is an example of such statements:

- Income Statement
  The income statement shows the performance of the business throughout each period, displaying sales revenues, gross income, EBIT (Earnings Before Interests and Taxes), and Net Income;
- Balance Sheet
  The balance sheet shows the financial position of the business. It does this by displaying the company's assets, liabilities and shareholder's equity.

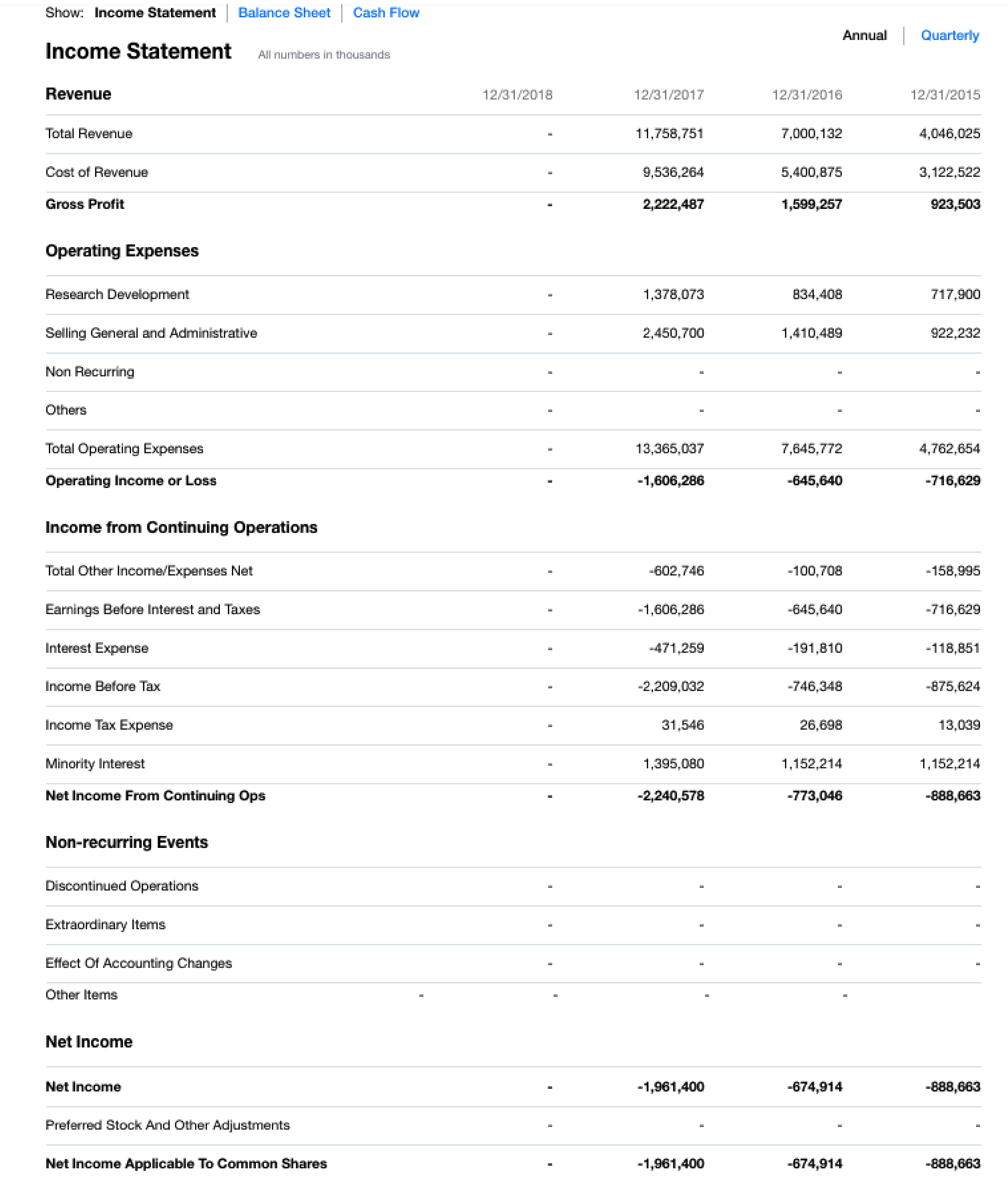
$$Assets = Liabilities + Shareholder'sEquity \tag{2.1}$$

- Cash Flow Statement
  The cash flow statement is used to display the increases and decreases in cash along a period of time (e.g. 1 year, 1 quarter, etc.)

According to the **U.S Securities and Exchange Commission**, publicly traded companies have an obligation to disclose their financials [73]. This disclosure is made through a variety of different types of reports, namely:

- Annual Reports on Form 10-K;
- Quarterly Reports on Form 10-Q;
- Current Reports on Form 8-K;

Extracting this type of information from financial reports, compiling it and centralizing it in order to provide easy access to investors is a difficult task. Fortunately, platforms such as the Bloomberg Terminal already have the infrastructure in place to execute that task, and this is why Bloomberg is considered one of the most reliable sources of information for investment professionals, which is why in order to get access to this data service investors pay a subscription of 25,000$ per year [75]. This opens up the question for a more easily accessible, cheaper, but equally reliable source of data. There are however



Income Statement

All numbers in thousands

Revenue	12/31/2018	12/31/2017	12/31/2016	12/31/2015
Total Revenue	-	11,758,751	7,000,132	4,046,025
Cost of Revenue	-	9,536,264	5,400,875	3,122,522
Gross Profit	**-**	**2,222,487**	**1,599,257**	**923,503**
Operating Expenses				
Research Development	-	1,378,073	834,408	717,900
Selling General and Administrative	-	2,450,700	1,410,489	922,232
Non Recurring	-	-	-	-
Others	-	-	-	-
Total Operating Expenses	-	13,365,037	7,645,772	4,762,654
Operating Income or Loss	**-**	**-1,606,286**	**-645,640**	**-716,629**
Income from Continuing Operations				
Total Other Income/Expenses Net	-	-602,746	-100,708	-158,995
Earnings Before Interest and Taxes	-	-1,606,286	-645,640	-716,629
Interest Expense	-	-471,259	-191,810	-118,851
Income Before Tax	-	-2,209,032	-746,348	-875,624
Income Tax Expense	-	31,546	26,698	13,039
Minority Interest	-	1,395,080	1,152,214	1,152,214
Net Income From Continuing Ops	**-**	**-2,240,578**	**-773,046**	**-888,663**
Non-recurring Events				
Discontinued Operations	-	-	-	-
Extraordinary Items	-	-	-	-
Effect Of Accounting Changes	-	-	-	-
Other Items	-	-	-	-
Net Income				
Net Income	**-**	**-1,961,400**	**-674,914**	**-888,663**
Preferred Stock And Other Adjustments	-	-	-	-
Net Income Applicable To Common Shares	**-**	**-1,961,400**	**-674,914**	**-888,663**

Figure 2.1: Tesla, Inc. (Tesla) Annual Income Statement [22]

many other sources that provide investors with this kind of data, for example **CNBC** ,**Reuters**, **Morningstar**, **Seeking Alpha**, **Yahoo Finance**,**Google Finance** ,**Quantopian** and **Investopedia**. Eventhough some of these services are freely accessible and already provide a good amount of information, investment professionals tend to prefer using premium services like the Bloomberg terminal and Morningstar because they know that the whole industry makes decision based on these services, so essentially, it creates a need for having the same exact data as every other investor in the market and eleminating a possible point of disadvantage.

2.1.4 Technical Analysis

Technical analysis is an investment methodology that evaluates investments purely on the market activity surrounding them, with no consideration of the actual operations or value of the company itself. Relevant factors that would be looked at include:

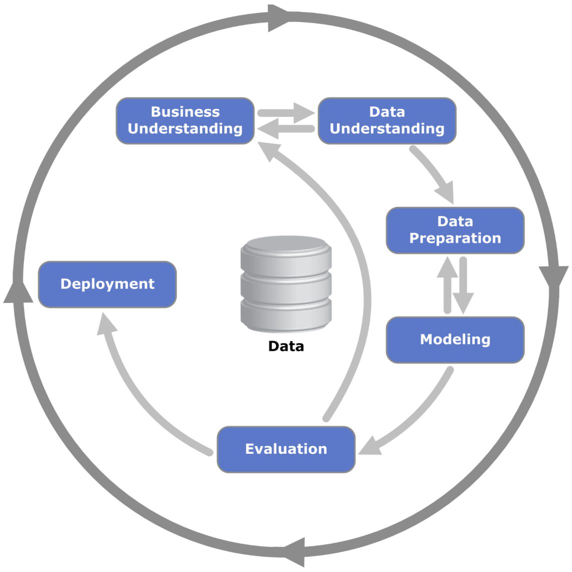
- Historical pricing of the shares;
- Standard deviation of prices in a time period;
- Trading volumes over time;
- News;
- Sentiment data;
- Industry trading trends;
- Market movements;

The goal of this analysis is to capitalize on pricing opportunities and trends that can be identified in the market activity around each company. As this methodology is purely based on historical market activity this is considered to be a backward looking and speculatory practice [13, 32].

2.2 Decision Support Systems

The concept of decision support was started at the Carnegie Institute of Technology in the late 1950s and early 1960s. After that, some researchers started studying the use of computerized quantitative models to aid in decision making (Raymond, 1966; Turban, 1967; Urban, 1967, Holt and Huber, 1969) [16]. The Massachusetts Institute of Technology (MIT) also applied computer technology to decision-making theory in the 1960s. By the 1980s, intensive research on DSS was underway, and new theories and concepts started emerging from single-user models of DSS, including organizational decision support systems (ODSSs), group decision support systems (GDSSs) and executive information systems (EISs) [16, 82]. The researchers' model of decision making process originally

consisted of three phases: **intelligence**, **design** and **choice**. In this model, intelligence is concerned with the search for problems, design involves the development of alternatives, and choice is about analysing the alternatives and selecting one for implementation. This classic problem-solving model of “intelligence-design-choice” has been widely accepted and adopted. Even though the researcher later extended the model with a fourth monitoring phase, DSS research remained primarily focused on the original three-phase model. By 1990 the field of DSS was broadened to include data warehousing(DW) and online analytical processing(OLAP). A well-designed DSS aids decision makers in compiling a variety of data from many sources: raw data, documents, personal knowledge from employees, management, executives and business models. One can think of data wareshousing as the storage system for all the data that is gathered for future analysis and, in turn, decision making. OLAP, is an approach to answer certain queries needed to aid the decision making [82]. The field of *Decision Support Systems* is widely studied and applied to many areas, and there is a standard method of approaching the question of how such systems should be built. The authors of "On Construction a Financial Decision Support System" [49] describe it as:
"The first step is the collection of the data set. If a domain expert is available, then he/she could suggest which fields (attributes, features) are the most informative. If not, then the simplest method is a “brute-force”, which indicates the measuring of everything available and only hopes that the right (informative, relevant) attributes are among them. However, a data set collected by the “brute-force” method is not directly suitable for induction. It comprises in most cases noise and missing values, therefore it needs pre-processing. The choice of which specific data mining algorithm to use is a critical step. Once preliminary testing is judged to be satisfactory, the classifier/regressor is available for routine use. If the testing is not satisfactory, we must return to a previous stage. The earlier stage we return, the more time we spent but the result may be better". This is essentially the concept of **CRISP Data Mining**, or Cross-industry standard process for data mining, which splits the process of data mining in six main phases:



1. Business Understanding
2. Data Understanding
3. Data preparation
4. Modelling
5. Evaluation
6. Deployment

Figure 2.2: CRISP [81]

This methodology allows for a repeatable approach to examine data, remove system inefficiencies and improve models. The following sections in this chapter describe related work and state of the art approaches for each of this CRISP phases.

2.3 Building a Data set

The data set is the basis upon which the proposed system will operate. Therefore, the key in building the appropriate data set rests on selecting the fundamental indicators (features of the data set) that carry the most *weight* (or correlation) with the performance (price) of the stock. In "A Data Warehouse Based Modelling Technique for Stock Market Analysis" [55], the authors list a few of the most important dimensions to consider when designing the DW Schema: **time**, **interest rate**, **price of petrol**, **price of gold**, **government and policy decisions**, **political decisions**, **industry decisions**.

Even though all of these dimensions have an impact on the performance of the market, as mentioned by the authors, in some of them, the weight of the correlation is only significant in a particular set of market conditions. For example, the price of petrol has an impact on the cost of production, but that will probably not have much impact on an online business. The interest rate has particularly interesting effect on the market. If rates go down, the markets typically answers with an increase in prices, with one of the reasons being the fact that companies can borrow money at a lower cost from the Banks, which means profits increase. If rates go up, it has the opposite effect. This is what some investors call "ripple effect" [41].

That said, we should focus the data set on more specific indicators, fundamental indicators directly and specifically related to the company and its own industry. There are a number of well respected fundamental indicators popularized by some of the most successful investors like **Warren Buffet, Peter Lynch, Ray Dalio, Joel Greenblatt, Howard**

Marks, Benjamin Graham and Seth A. Klarman.

As mentioned in section 2.1.3, publicly traded companies provide, at the end of every quarter and at the end of every year, reports that detail their current financials. These reports give investors access to every possible indicator to analyse a company, however there are a few fundamental indicators that are quoted as provinding the best possible comparison between companies that are not present in these reports. The **P/E ratio** or price earning ratio, is cited as one of these indicators. However, working with features such as this one creates a few problems when working a model like a decision tree, which is detailed in section 4.5.2 of chapter 4. In order to somewhat limit complexity of the models that are going to be used, we should limit the range of features included in our data set. So, for this thesis, the considered features are [40]:

- **Cash** Cash and cash equivalents refers to the line item on the balance sheet that reports the value of a company's assets that are cash or can be converted into cash immediately.

- **Cost of Revenue** The cost of revenue is the total cost of manufacturing and delivering a product or service to consumers.

- **Gross Profitability** Gross profit is the profit a company makes after deducting the costs associated with making and selling its products, or the costs associated with providing its services.

- **Free Cash Flow** Free cash flow, a subset of cash flow, is the amount of cash left over after the company has paid all its expenses and capital expenditures (funds reinvested into the company).

- **Current Assets** Current assets represent all the assets of a company that are expected to be conveniently sold, consumed, utilized or exhausted through the standard business operations, which can lead to their conversion to a cash value over the next one year period.

- **Current Liabilities** Current liabilities are a company's debts or obligations that are due within one year or within a normal operating cycle.

- **Total Debt** Total Debt is the aggregate of short and long term debt and liabilities.

- **Total Equity** Equity is referred to as shareholder equity (also known as shareholders' equity) which represents the amount of money that would be returned to a company's shareholders if all of the assets were liquidated and all of the company's debt was paid off.

- **Investment to Asset** Investment to Asset or Return on Investment (ROI) is a performance measure used to evaluate the efficiency of an investment or compare the efficiency of a number of different investments.

- **Sales** Sales is a metric for the overall sales of a company, unadjusted for costs incurred in generating those sales, as well as things like discounts or returns from customers.
- **Share Issuance** Issued shares are the authorized shares sold to and held by the shareholders of a company, regardless of whether they are insiders, institutional investors or the general public, as shown in the company's annual report. Issued shares include the stock a company sells publicly to generate capital and the stock given to insiders as part of their compensation packages.
- **Price** The price of a stock is the price investors have to buy in order to buy one share of a publicly traded security/company.

In order to build an interesting test case, these indicators were chosen based on the following principles:

- These indicators are reliable because they are provided by each company, and are regulated with established accounting methods;
- Do not include ratios that incorporate the price of the stock (the reason why this is important is detailed in section 4.5.2) and are mainly fundamental;
- These indicators are believed by the partner domain experts (GoBusiness Finance) to provide a perspective of the overall state of the company;
- They are easy to understand even for someone without extensive financial knowledge.

These indicators were also chosen in order to compare our model with one already implemented by an investment fund, **ERAAM**. As a reference, in their documentation, the ERRAM [19, 20] fund says "It all starts with data". This fund has a **Stock Universe** composed of 14 000 stock in 2018 of which 6000 are investable, 90% of the world's market capitalisation, and they also have a **proprietary optimised ERAAM dabatase** with 15+ partners that provide data since 1990. Although the indicators used by this fund are not all publicly available, we used two indicators shown on publicly available presentations. Image 2.3 details indicators used by this fund.

After selecting the features to work with, it is necessary to study and subsequently choose how the data selected from these indicators will be warehoused.

2.3.1 Data Warehousing

Financial market data, specifically data related to stocks is of a particular type, called *Time-Series-Data* which is a series of numerical data points that have inherent correlations based on time periods. The term applies to companies and markets data points across a period of time [48]. In the case of stocks, storing this type of data is not overly complex,

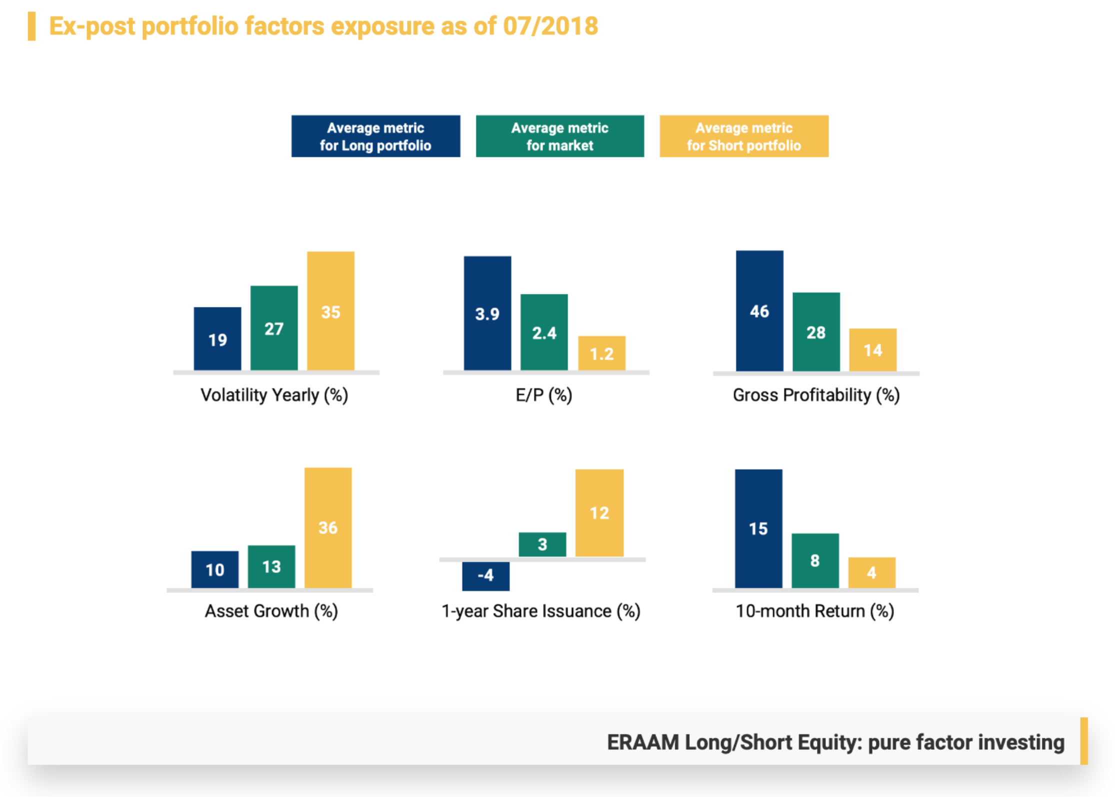


Figure 2.3: ERAAM Long Short Equity Portfolio indicators [19]

as there are only three main dimensions: **symbol**(which indicates the specific company) **time**(the time frame to which the data refers to) and the **features**(which in the case of this system are fundamental indicators, detailed previously). According to the partner domain experts, nowadays Microsoft Excel is the main tool for data warehousing in small investment firms. This is due to the fact that it not only enables professionals to archive financial data(ussualy in the .csv format), but it also enables them to apply their models and formulas already developed in that environment. However, in trying to build a *decision support system*, a lot of data will be collected and stored so there is an obvious need for a scalable and reliable management system for this data. Chapter 4 details the approach to findig an alternative method for storing data. Regarding *database management systems* it is interesting to point out that the Bloomberg data service has outgrown most available "retail"DBMS, so it now uses its own proprietary (but open-source [30]) RDBMS called **Comdb2** which is designed for geographical replication and high availability [72], again showing how much this "new"commodity has become.

2.4 Building the Model

After the parameters of *input data*(i.e. features of the data set) are chosen, there is a need to define the methods and rules to be used in order to perform the analysis of the data, which in this case aim finding valuable publicly traded companies. We can refer to these "methods and rules"as a *Model*. In this context, a model is well defined as "a theoretical construct representing economic processes by a set of variables and a set of relationships between them." [77] According to the authors of "Integration of decision support systems to improve decision support performance" [50], the "central role of models and provision of mechanisms for the management of models have been regarded as what distinguish a DSS from more traditional information processing systems".

The models presented in this section are a few examples that fit this description, and a similar version of the ranking algorithm (detailed in section 2.4.1.2) will be used in the end system that this thesis aims to build. There are however, a few other models that can be applied to the financial markets, but that can only be applied through more powerful computational capabilities and "intelligence", hence computational finance models and methods, described in chapter 3.

2.4.1 Classical Valuation Methods

Given that, decision support systems need a model for data analysis, and given that this thesis is focused on financial analysis of publicly traded companies, a variety of valuation models were studied during the development of the thesis. Peter Lynch, a well renowned investor and author famously says: "Know what you own, and know why you own it" [52]. That is the purpose of a financial valuation model, determining the present value of an asset. If the investor has reason to believe that asset will increase in value in the future then that asset could be considered as a possible investment. Choosing a particular method of valuation can be very challenging because its success depends on a number of factors, for example:

- Economic and market conditions(Upturn vs Downturn);
- Is the model the right one for the investors strategy?
- Does the investor have trustworthy data?
- Can the chosen model be applied to the type of asset the investor is trying to value?

That said, there are a number of valuation methods made famous by some of the most successful investors of our time. For example:

2.4.1.1 DCF Model

The **Discounted Cash Flow** model is used to determine the present value of a company based on future cash flows. It is best used when a company does not pay a dividend (distribution of the company's earnings to investors) or their payments are irregular, because it means that company's earnings are being invested back in growing the business. The organization must have stable, positive and predictable cash flows for this model to be effective. One of the main principles behind this model is that the value of a company is not determined by a function of supply and demand for that company's stock. It is instead, determined by the company's ability to generate cash flow in the future for its stockholders [10, 11, 78]. The formula for DCF for a period of n years is [61]:

$$Value_{t=0} = \sum_{n=1}^{t=n} \frac{CashFlow_t}{(1+r)^t} \tag{2.2}$$

Where:

- r is the discount rate;

2.4.1.2 The Magic Formula (Ranking Algorithm)

One big proponent of the **Value Investing** methodology, Joel Greenblatt [35], puts forward two main indicators. The first one, **Earnings Yield** is the determining factor of whether or not he is buying a business at a *good price*. A company with a High Earnings Yield is what the author calls a "Bargain". However, the author is not only interested in buying businesses cheaply, but he is particularly interested in buying *Good Businesses* cheaply. The author determines that the indicator that points to whether or not a businesses is considered *good* is its **Return on Capital**. A company with a High Return on Capital is considered a good business.

$$ReturnonCapital = \frac{EBIT}{NetWorkingCapital + NetFixedAssets} \tag{2.3}$$

Where [40]:

- *EBIT* is Earnings Before Interest and Taxes;

- *Net Working Capital* is the difference between a company's current assets, such as cash, accounts receivable and inventories of raw materials and finished goods, and its current liabilities, such as accounts payable.

- *Net Fixed Assets* is the purchase price of all fixed assets (i.e. Land, buildings, equipment, machinery, vehicles) less depreciation.

Using these indicators, the author creates what he calls the *Magic Formula*, which is essentially a ranking algorithm.Firstly, the author ranks the companies from highest to lowest earnings yield. Then, the author does the same with return on capital. The result would resemble the following:

Position	Return on Capital	Earnings Yield
1	E	G
2	B	A
3	F	B
4	C	D
5	A	F
6	G	E
7	D	C

As as example, the letters on the table represent publicly traded companies. Where company **E** has the most return on capital, but in number 6 in terms of earnings yield, comapny **B** has the second most return on capital and has the third highest earnings yield, and so on with every other company. In order to combine both ranks, the author adds them up for every company. As a result of this, each company is left with two results:

$$R_{returnoncapital} = NumberofCompaniesontheList - PositionOftheCompanyontheList \tag{2.4}$$

$$R_{earningsyield} = NumberofCompaniesontheList - PositionOftheCompanyontheList \tag{2.5}$$

And lastly, as a final ranking, we are left with:

$$Finalrank = R_{returnoncapital} + R_{earningsyield} \tag{2.6}$$

So the overall ranking would be:

Position	Company	Final Rank
1	B	9
2	A,E,G	7
3	F	6
4	D,C	3

Lastly, the investor defines a threshold where companies that are above it are bought and enter the portfolio, and companies that are not do not enter the portfolio.

2.5 Computational Finance

Computational Finance is generally defined by various universities as a field of finance, or modern quantitative finance, that relies on mathematics, statistics and computing to be applied and solve problems related to financial modelling and algorithmic financial computation [51, 59, 88]. Arguably, the methods of computational finance have been applied since the beginning of finance itself. The main difference is in the kind of computational tools used. Early on, calculations would be performed by hand or they would be assigned to "human computers". Nowadays, technology is taking finance to new levels with advances in cloud computing, machine learning, behavioral science and even blockchain [56]. For instance, Microsoft, with its cloud computing platform, Azure, as entered the field of *cloud computing applied to the markets* [4] offering various solutions for data storage, distributed computing services, AI and machine learning technologies for detecting financial fraud [5].

2.5.1 Technology applied to the markets

Whatever approach (Fundamental or Technical) a fund manager decides is best, he/she will inevitably have to deal with the increasing speed and *democratization* of information present in Financial Markets. Due to the advances in database technology, high-speed internet (and arguably just the web itself) and general availability of computing capabilities [24], the financial markets have "graduated"from an environment where only the most well connected investors could compete to an environment where everyone can play a role.

As mentioned previously, investment managers are usually reliant on enterprise software like Microsoft Excel, and the Bloomberg Terminal. However, according to a 2015 report from the WEF (World Economic Forum) [25], the wealth management industry suffered a significant loss of customer trust following the financial crisis. As a result of that, a number of disruptors emerged to provide low-cost and sophisticated alternatives to traditional wealth managers and to a broader customer base. Some of those innovations were the rise of:

- Automated Management
- Social Trading
- Retail Algorithmic trading

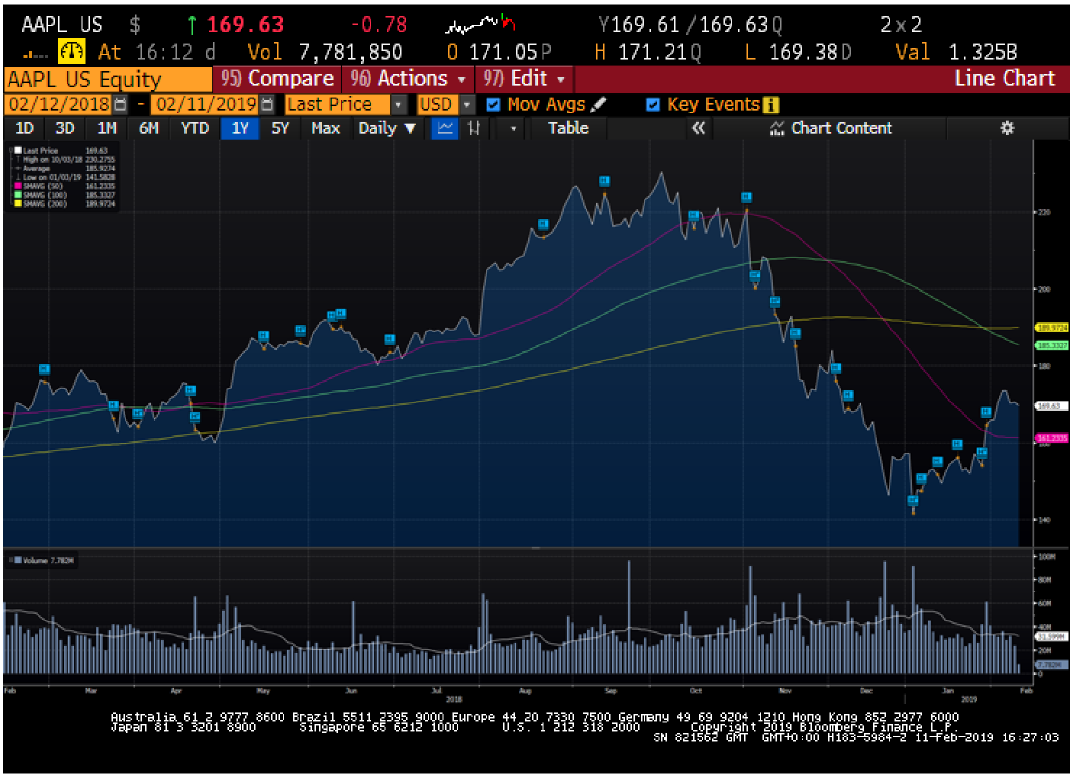
Overall, the introduction of these types of systems has made the Investment Management industry become **hyper-efficient** and **low-fee**. It has also facilitated the offering of more **customized investment portfolios**. More recently, in a 2018 report [26], the WEF also mentions the fact that AI (Artificial Intelligence) has the "potential to democratize access to capital across the global economy by unlocking greater efficiency". So, in order to keep up with the ever-increasing speeds, frequencies and data volumes, financial institutions are now evolving to become technology companies, because technology can bring a competitive advantage to an institution and that is particularly crucial in this very competitive environment.This phenomenon has even earned the "field"its own name. It is called *Fintech*, although that is a much wider field than the one this thesis aims to focus on, which is to study how significant is the development of an investment DSS, and what advantages would it bring to investors.

2.5.2 Examples of DSS applied in Computational Finance

Many people have tried to predict the movement of the stock market but no one has really been able to do it for a sustainable period of time. Some have even attempted to incorporate tools like **AI (Artificial Intelligence)** and **Data Mining** in price prediction. In a 2009 paper [90], "Designing a Decision Support System Model for Stock Investment Strategy"the authors propose that the advances in computing power open up some opportunities in a new era known as financial engineering or computational finance. The paper presents the "study on designing a decision model for investment strategy by utilizing financial methods and computation techniques". The authors defend the premise that "the market itself is its own best source of data"and that all the investors' actions regarding all the available information is already accounted for on the stock price. Even though the paper is focused on **Technical Analysis**, there are some important remarks to note and that will be useful when building the Investment Decision Support System based on Fundamental Analysis, namely the fact that the DSS has two main components:

- The Data Set;
- The Model;

Perhaps the most famous system that already incorporates both of these components is the Bloomberg Terminal. Bloomberg L.P. is a privately held financial, software, data, and media company. The **Bloomberg Terminal** is a computer system that allows investors to access the Bloomberg data service which provides real-time financial data, news feeds, messages and also facilitates the placement of financial transactions. The Bloomberg terminal, from the perspective of the end-user, is a Windows-based application where the investor can get acces to all information services provided. It is also compatible with the popular Excel program, a very important part of the workflow for those in the finance industry. This might be one of the reasons why investment firms still use Microsoft Excel as an archive for financial data. Bloomberg also offers users access to the application online and through mobile devices, via its Bloomberg Anywhere service. For portfolio managers and brokers, having the ability to access real-time market information from almost anywhere in the world, is an incredibly convenient and important advantage of a Bloomberg subscription. GoBusiness Finance will provide access to the Bloomberg Terminal as well as training for using the platform.



AAPL US $ ↑ 169.63 -0.78 Y169.61 / 169.63 Q 2 x 2
At 16:12 d Vol 7,781,850 O 171.05 P H 171.21 Q L 169.38 D Val 1.325B

AAPL US Equity 95) Compare 96) Actions ▾ 97) Edit ▾ Line Chart

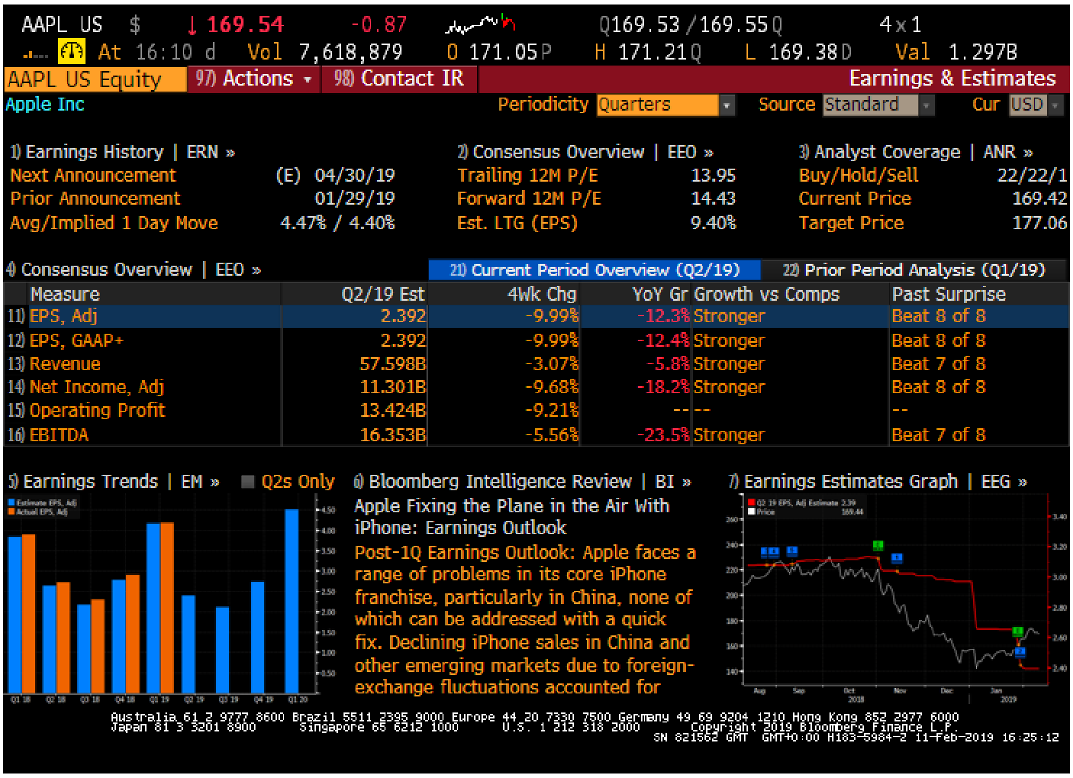
02/12/2018 - 02/11/2019 Last Price USD ☑ Mov Avgs ☑ Key Events

1D 3D 1M 6M YTD 1Y 5Y Max Daily ▼ Table Chart Content

Last Price	169.63
High on 10/03/18	230.2755
Average	185.9274
Low on 01/03/19	141.5828
SMAVG (50)	161.2335
SMAVG (100)	185.3327
SMAVG (200)	189.9724

Australia 61 2 9777 8600 Brazil 5511 2395 9000 Europe 44 20 7330 7500 Germany 49 69 9204 1210 Hong Kong 852 2977 6000
Japan 81 3 3201 8900 Singapore 65 6212 1000 U.S. 1 212 318 2000 Copyright 2019 Bloomberg Finance L.P.
SN 821562 GMT GMT+0:00 H183-5984-2 11-Feb-2019 16:27:03

Figure 2.4: Apple (AAPL US) Price Chart [Bloomberg Terminal]



AAPL US $ ↓ 169.54 -0.87 Q169.53 / 169.55 Q 4 x 1
At 16:10 d Vol 7,618,879 O 171.05 P H 171.21 Q L 169.38 D Val 1.297B

AAPL US Equity 97) Actions ▾ 98) Contact IR Earnings & Estimates

Apple Inc Periodicity Quarters Source Standard Cur USD

1) Earnings History | ERN »		2) Consensus Overview | EEO »		3) Analyst Coverage | ANR »	
Next Announcement	(E) 04/30/19	Trailing 12M P/E	13.95	Buy/Hold/Sell	22/22/1
Prior Announcement	01/29/19	Forward 12M P/E	14.43	Current Price	169.42
Avg/Implied 1 Day Move	4.47% / 4.40%	Est. LTG (EPS)	9.40%	Target Price	177.06

4) Consensus Overview | EEO » 21) Current Period Overview (Q2/19) 22) Prior Period Analysis (Q1/19)

Measure	Q2/19 Est	4Wk Chg	YoY Gr Growth vs Comps		Past Surprise
11) EPS, Adj	2.392	-9.99%	-12.3%	Stronger	Beat 8 of 8
12) EPS, GAAP+	2.392	-9.99%	-12.4%	Stronger	Beat 8 of 8
13) Revenue	57.598B	-3.07%	-5.8%	Stronger	Beat 7 of 8
14) Net Income, Adj	11.301B	-9.68%	-18.2%	Stronger	Beat 8 of 8
15) Operating Profit	13.424B	-9.21%	--	--	--
16) EBITDA	16.353B	-5.56%	-23.5%	Stronger	Beat 7 of 8

5) Earnings Trends | EM » ■ Q2s Only

6) Bloomberg Intelligence Review | BI »

Apple Fixing the Plane in the Air With iPhone: Earnings Outlook

Post-1Q Earnings Outlook: Apple faces a range of problems in its core iPhone franchise, particularly in China, none of which can be addressed with a quick fix. Declining iPhone sales in China and other emerging markets due to foreign-exchange fluctuations accounted for

7) Earnings Estimates Graph | EEG »

Australia 61 2 9777 8600 Brazil 5511 2395 9000 Europe 44 20 7330 7500 Germany 49 69 9204 1210 Hong Kong 852 2977 6000
Japan 81 3 3201 8900 Singapore 65 6212 1000 U.S. 1 212 318 2000 Copyright 2019 Bloomberg Finance L.P.
SN 821562 GMT GMT+0:00 H183-5984-2 11-Feb-2019 16:25:12

Figure 2.5: Apple (AAPL US) Description [Bloomberg Terminal]

Chapter 3

Models and Methods for Computational Finance

3.0.1 Machine Learning

Machine Learning is the science of programming a computer to learn from data.

"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E." [54][**Machine Learning(p.2), Tom Mitchell, 1997]**

There are a few types of Machine Learning [76]:

- **Supervised** In supervised learning, most frequently in classification trees, the training data we feed to the algorithm includes the both the inputs and the desired outputs, called labels. When working on regression problems, a typical task is to predict a target numeric value, such as the price of a car, given a set of features (i.e. kms, age, brand) called predictors. To train the system, we need to give it many examples of cars, including both their predictors and their labels (e.g. their prices);

- **Unsupervised** In unsupervised learning, the training data is not labeled, therefore the algorithms learn form data that is not labeled, categorized or even classified;

- **Semi-supervised** This type of learning deals with partially labeled training data, usually a lot of unlabelled data and a smaller amount of labeled data. This is called semi-supervised learning;

- **Reinforcement Learning** In Reinforcement Learning the learning system, called an agent in this context, can observe the environment, select and perform actions, and get rewards in return (or penalties in the form of negative rewards). It must then learn by itself what is the best strategy, called a policy, to get the most reward over time. A policy defines what action the agent should choose when it is in a given situation.

This thesis deals with a regression problem, in that we have a variety of features that are numerical, countinuous values(described in section 2.3) and the target variable is know, the price, which is also a numerical continuous value. Hence, this problem, associated with the type of model used, the decision tree, falls into the **supervised** type of machine learning.

3.0.2 Machine Learning in Finance

Starting to use any new type of technology tipically has some difficulties. In the case of machine learning, one of the main challenges for companies is to hire the right people, because it is still a field in rapid development. Despite the challenges, many financial companies have begun implementing a variety of Machine Learning solutions. It can help reduce costs of operations due to automation, increase profits due to better productivity, better compliance and reinforced security, so many of the biggest financial firms have begun heavily investing in both current ML systems as well as in more research and development [15, 37]. This makes machine learning directly and indirectly important in the markets. Indirectly because it can help in making publicly traded companies more efficient and more productive, which is positive for investors. Directly because investors themselves can use machine learning models to analyse the markets and extract additional information, somewhat like this thesis aims to do. Besides academic researchers, financial companies have been early adopters of these various Machine Learning tools. However, regarding the applicability of ML to stock market investing, there is still not much information regarding how exactly it is being applied, and that could be due to a number of factors. Firstly, financial companies have a tendency of being very secretive about their practices, and since ML could potentially be revolutionary to the industry, information regarding the subject is somewhat scarce. The second reason could perhaps be the fact ML is still not regularly used as a tool for investing, and financial firms are still waiting for more research on the matter. The reason why machine learning still are not very dominant in the financial markets could be because of the following:

- Complexity of the interaction between Data makes it harder for models to models to establish rules;

- Human behaviour and emotion(social science as opposed to an exact science);

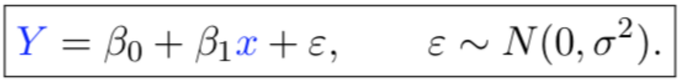
- Complex long-term and short-term market cycles there require economic data to understand;

However, there are also some important arguments to be made about how cyclical economy and the financial markets are [64], and so, machine learning could be a great tool to deal with potential patterns in the markets, and take advantage of that.

In preparation for the development of the system that this thesis aims to build, a variety of models were studied. Some of them do not enter the field of machine learning but are helpful for understanding the problem in question for this thesis.

3.0.3 Linear Regression

Linear regression is a method of modelling the correlation between dependent and independent variables.



$$Y = \beta_0 + \beta_1 x + \varepsilon, \qquad \varepsilon \sim N(0, \sigma^2).$$

Figure 3.1: Simple linear regression formula [27]

Where **Y** is the *independent variable*, **x** is the *dependent variable*. ε is the *error variable* which is meant to add "noise"to the linear relationship between **Y** and **x** [27]. Although ε follows this type of distribution, Benoit B. Mandelbrot [53] disagrees that this can be applied when using linear regression for stock market analysis. β_0 and β_1 are the deterministic parameters and have to be estimated. Although this model assumes that the relationship between the variables is linear, which is not always the case in the stock market, it can be very useful to implement a trend line which represents the long-term movement of time-series data [84]. Besides the representation of a trend line, the linear regression method is also commonly used to determine the *capital asset pricing model* or CAPM, which is used to find the appropriate rate of return of an asset in order to make the decision of adding that asset to a portfolio [79]. The formula for multiple linear regression is [89]:

$$Y_i = \beta_0 + \beta_1 * x_i 1 + \beta_2 * x_i 2 + ... + \beta_p * x_i p + \varepsilon_i \tag{3.1}$$

For i=1,2,...n.

3.0.4 Regression Trees

Linear regression and logistic regression models fail in situations where the relationship between features and outcome is nonlinear or where features interact with each other. This is where decision trees are useful. These trees split the dataset multiple times in certain splliting points, which are dependent on the algorithm used to build the tree. In a decision tree the data ends up in distinct groups that are often easier to understand than points on a linear regression beacause it creates a natural visualization with its nodes and edges.

There are two types of decision trees: **Classification Trees** and **Regression Trees**. Classification trees are used when dependent variables are categorical (e.g. positive or negative), whereas regression trees are used when the dependent variables (in this case, fundamental indicators) are continuous values. The latter type is the one that is applicable to the problem described because as established earlier we are working with continuous numerical values. There are various algorithms with which a decision tree can be created. The ID3 and the C4.5 extension introduced by J.R Quinlan [6] are well known algortihms to create decision trees for classification problems. However, when dealing with regression problems, the appropriate algorithm is usually CART (Classification and Regression Trees). This algorithm is very similar to C4.5, but it is different because it supports numerical target varibales. It constructs binary trees (each node has a maximum of two edges) that yield the maximum information gain at each node. Companys variables, such as its returns, liabilities and price are considered continuous variables because theoretically there is an infinite number of possible values for each of these variables, and regression trees are used to deal with continuous variables [3].

In order to build the regression tree, the CART algorithm has two main tasks: **selecting features** and **finding split points**. At the beggining of the process, the tree is split once using one feature. This is done by computing every possible splitting point for every feature. Then, the best possible split for each feature is chosen, and lastly for the actual split it is picked the feature ans its split point minimize the *impurity* of the node at that point, where the impurity is a measure of how homogeneous the nodes are. This originates two child nodes where each one contains a smaller subset of samples, and each one will be split using the same process [8]. Scikit-Learn (detailed in section 4.1) and its decision tree regressor use **Mean Squarred Error**, which is equal to variance reduction, as a default criterion for measuring the impurity. This regressor has two possible strategies to choose the split at each node: **best** to choose the best split (the smallest impurity) and **random** to choose the best random split. The process of creating the tree using the provided data is called **fitting**, and this process is done using binary recursive partitioning. It is *binary* because parent nodes are always split into two child nodes, and it is recursive because every child node eventually becomes a parent node unless it becomes a leaf node. In regression problems, the value assigned to leaf nodes is the mean of the target variable of all the samples in that node. The algorithm continues this process recursively in every new node until a stop criterion is reached. Possible criteria are [74]:

- If a node becomes pure which means that all cases in that node have identical values of the dependent variable;

- If all cases in that node have the same value for the target variable, which given the fact that in this case is the price, a cotinuous variable, is a somewhat rare stopping criteria;

- If the tree reaches some stopping parameters defined by the programmer. Possible parameters are *maximum depth*, *maximum leaf nodes*, *minimmum samples to split* and some others, detailed in section 5.1.1.

Finding the appropriate parameters for this type of data, along with a technique called pruning (described in section 4.8) can avoid problems like **overfitting** and **underfitting**. Overfitting happens when the model captures patterns that will not be recurrent in the future, whereas underfitting happens when the model fails to capture relevant patterns, both lead to less accurate predictions [42].

After the tree is built, it ends up with three components. The first one represents the decision or rules inferred from the data set and it is called an *edge* or branch. The second component of the tree represents the featues of the data set and it is called *node*. The last main component is a special type of node. It is called *leaf* and it represents the outcome of a certain path of rules [9, 83]. In terms of visualization, the nodes are labeled with the questions about the data set, and branches are labeled with the answers. The root of the tree (top node) has the entire data set samples in it, whereas the subsequent lower nodes have a smaller amount of samples. Each node has the sum of the number samples in the nodes below it.

When preparing to build a regression tree there is one major aspect to take into account, dividing the available data in two groups: **training data** and **testing data**. This is important because we do not want to make the system predict something it already knows the answer to, meaning testing with data that already was a part of data used to create the tree. To avoid this, the data (historical financial data) is split into two groups, and this can be done in a variety of ratios, but common used ratios are 70%/30%, 60%/40% or 50%/50% [34].**Chapter 4** describes how these two sets of data were built for our particular case. Figure 3.2 is a sample decision tree that resulted from an experiment with the **Boston house data set**.

Because of how decision trees are built, they have the ability to tell the user how important each feature is in determining the target variable. Hence the regression tree model will be applied in the proposed decision support system with the main intent of providing the investor with the knowledge of what features (financial indicators) matter most in determining the price. This is described in the following section.

3.0.5 Feature Based Investing

Feature, or factor, based investing relies on selecting features that are higly correlated with the target variable and using those to inform the investor. This technique is already being implemented by a few Investment Funds, like **ACATIS** [2] and **ERAAM** [19, 20] with its Long Short Equity Fund. After the features are selected and ranked by importance they then serve as an input to an investment algorithm or valuation model. This is a different topic, but for the case of this thesis we are going to use the importances of each feature as an input to a ranking model. This technique is based on what was shown in section 2.4.1.2 and is detailed in section 4.10.

Before trying to rank features, we should look at how we could make the job of the regression model (regression tree) easier. One of the ways this is usually done is by

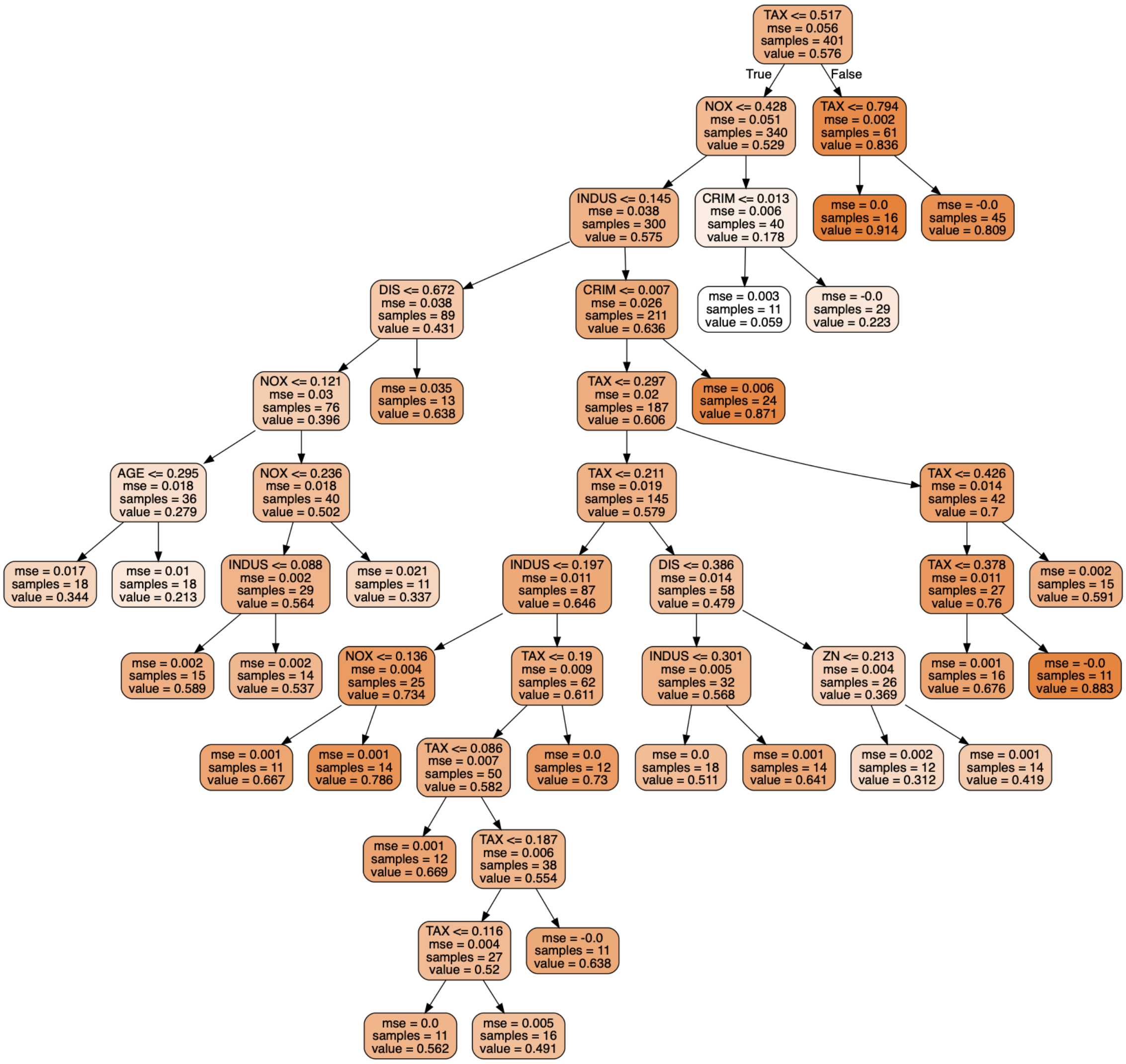


Figure 3.2: Sample Regression Tree produced with python

"pre-selecting"relevant features. Inherently, we know this reduces the complexity of the tree. One of the ways we can select relevant features is by applying some **Regularization** techniques. Some of the popular techniques used are [71]:

- **L1 Regularization**, also known as *Lasso Regression* is described as "a linear model that estimates sparse coefficients. It is useful in some contexts due to its tendency to prefer solutions with fewer non-zero coefficients, effectively reducing the number of features upon which the given solution is dependent";

- **L2 Regularization**, also known as *Ridge Regression* is described as "provides improved efficiency in parameter estimation problems in exchange for a tolerable amount of bias";

By doing this, we can also be preventing overfitting, which as mentioned previously, tends to happen with tree models. However, as detailed in section 2.3, in this case, given the nature of the data (there aren't that many fundamental indicators) and the constraints of collecting data, we are only working with 11 features, plus the price of the stock, which does not create the necessity to apply techniques like regularization. This is why these techniques were not studied extensively. Nevertheless, they are important to mention.

Hence, in order to determine the importance of each feature, we can analyse all the splits for each feature and measure how much this split reduced variance when compared to the root node. This method is fine when looking at a small tree, but difficult to perform when looking at a bigger tree, so each tree also has an attribute *feature_importances_* which directly outputs the importance of each feature (see Image 3.3). The sum of all importances adds to 1.

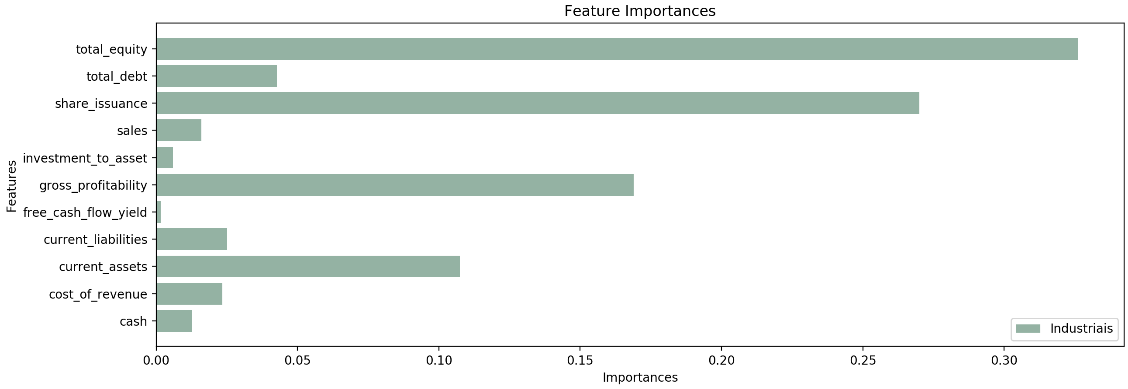


Figure 3.3: Sample of feature importances in one dataset

3.0.6 Python in Finance

Python has been ranked one the most in-demand programming languages in the financial industry [1]. This is mainly due to some key advantages, namely, its **scalability**, **concise code** and **powerful frameworks** [23]. Fund managers acknowledge the importance of an automated system to help take emotions and human error out of the process of valuing a business. This is where applying the strengths of Python to build a decision support system might present immense value. With a language/tool like Python, investment managers are able to implement their trusted models and in turn, make them much faster and much less prone to error. Libraries like Pandas (Python Data Analysis Library) allow managers to analyse the collected data (from an CSV file or SQL database for example) much more easily.

3.0.6.1 Why Python?

Python is a high-level, multipurpose programming language used in multiple fields. It can be used by a begginer programmer as well as highly experienced developers. It it well know by having the following features [36]:

- **Open Source** Python and a big part of its libraries are open source and have flexible and open licenses.
- **Interpreted** CPython is a reference implementation of an interpreter of Python that translates code at runtime to executable byte code.
- **Multiparadigm** Python supports paradigms such as object orientation and imperative, functional and procedural programming.
- **Multipurpose** Pyhton can be used for low-level systems as well as high-level analytics tasks.
- **Cross-Platform** Python is available for the most important operating systems such as Linux, Windows, Mac OS and it can even be run in a Raspberry Pi.
- **Dinamically typed** Types in Python are generally inferred during runtime and are not statically declared as is the case with other programming languages.
- **Garbage Collecting** Python has automated garbage collection so that developers do not have worry about managing memory.
- **Indentation aware** Python used indentation for making code blocks instead of parantheses, brackets and semicolons.

One major advantage of using Python for the type of problems present in the finance industry is that it has a syntax similar to the mathematical syntax used to describe traditional economic and financial models. Other advantages are connected to productivity, efficiency, quality and high performance.

3.1 Model Validation

The goal of this system is to provide investors with analysis support regarding their possible investments, and in order to do that, the chosen model is the regression tree. In order to get an understanding of how well this model fits the data provided by the investor, there are a few metrics that can be used, namely the **mean squarred error** and the R^2. Lastly, but still in the context of model validation, after the user gathers the information provided by the tree and inputs it into the investing model, there is also a need to perform some kind of validation regarding that model, this is described in the **Alpha**, **Beta** and **Backtests** sections.

3.1.1 Regression Score Functions

3.1.1.1 Mean Squared Error - MSE

MSE measures average squared error of our predictions. For each point, it calculates square difference between the predictions and the target and then averages those values.

It is never negative, since we are squaring the individual prediction-wise errors before adding them, but would be 0.0 for a perfect model. The higher this value, the bigger the error of the predictions made by the model. However, this is not absolute, beacause MSE is very influenced by scale of numbers, which means that when dealing with a data set that contains very large numbers, it should not be compared with one with numbers on a different scale.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{3.2}$$

3.1.1.2 Coefficient of Determination - R^2

Since MSE is scale dependent, it is difficult to establish if our model is good or not just by looking at the absolute values of MSE. In order to complement that metric, we also want to measure how our model is when compared with a baseline model like a simple linear regression. The coefficient of determination, or R^2, is another metric we may use to evaluate a model and it is closely related to MSE, but has the advantage of being scale-free, meaning that it does not matter if the input and output values are very large or very small, the R^2 is always going to be between [-∞;1].

The R^2 value typically ranges from 0 to 1, where **1 is the best possible result**. However, R^2 can also be a negative number, so [70]:

- R^2<0 means an horizontal line (like the mean) would better represent the data than our model does;
- R^2=0 our model represents data as well as an horizontal line;
- R^2>0 our model represents data better than an horizontal line (mean of all values);

In order to measure this coefficient sickit learn provides a **r2_score**. It is a measure of how well observed outcomes are replicated by the model, based on the proportion of total variation of outcomes explained by the model [17, 80]. In order to calculate R^2 there are two important variables:

- **RSS** - Residual Sum of Squares which is the sum of the squares of all the subtractions between the *true* value and the *predicted* value.
- **TSS** - Total Sum of Squares is the sum of the squares of all the subtractions between every *true* value and the *average* of all the values.

$$RSS = \sum_{i}(y_i - \hat{y}_i)^2 \tag{3.3}$$

$$TSS = \sum_{i}(y_i - \overline{y})^2 \tag{3.4}$$

And the general definition of the coefficient of determination is:

$$R^2 = 1 - \frac{RSS}{TSS} \tag{3.5}$$

3.1.2 Alpha (Jensen Index)

Alpha, is one of the most used indicators of investment performance. It is defined as the excess return on an investment relative to the return on a benchmark index. For instance, if an investor decides to invest in a stock and it returns 20% while the S&P 500 earned 5%, the alpha is 15. An alpha of -15 would indicate that the investment underperformed by 20%. Alpha is also a measure of risk. In this example, the -15 means the investment was far too risky given the return. An alpha of zero indicates that an investment has earned an appropriate return given the risk taken. Alpha of greater than zero means an investment outperformed. Alpha is one of the five major risk management indicators for mutual funds, stocks and bonds, and in a sense tells investors whether an asset has performed better or worse than its beta predicts [18, 60].

Hedge fund managers use this metric to promote their fund, and then they mention their "high alpha", they usually mean that their managers are good enough to outperform the market. But that raises another important question: when alpha is the "excess"return over an index, what index are they using? For example, a fund manager might say that he/she generated a 20% return when the S&P returned 15%, an alpha of 5. But is the S&P 500 an appropriate index to use? Considering a manager who has invested in Apple Inc. (AAPL) on Aug. 1, 2014. Compared to the S&P 500, the alpha would look quite good: Apple returned 18.14%, while the S&P 500 returned 6.13%, for an alpha of about 12. However, the risk is very significant because it is in just one stock, and not sufficiently diversified. The formula for alpha is [60]:

$$Alpha = Rp - [Rf + (Rm - Rf) * Beta] \tag{3.6}$$

Where:

- Rp= Realized return of portfolio
- Rm = Market return
- Rf = Risk-free rate

3.1.3 Beta

Beta in a measure of relative volatility. It measures how an asset (e.g. stock, ETF or portfolio) moves versus a benchmark (e.g. an index, or another investment fund) [21]. It is a multiplicative factor. A stock with a beta of 2 relative to the S&P 500 goes up or down twice as much as the index in a given period of time. If the beta is -2, then the stock moves in the opposite direction of the index by a factor of two [18, 21]. The formula for Beta is [44]:

$$Beta = \frac{Cov(RaRb)}{Var(Rb)} \tag{3.7}$$

Where:

- Cov(RaRb): Covariance of asset and market
- Va(Rb): Variance of market

3.1.4 Backtests

Creating an investment algorithm and then running that algorithm in a simmulated market in a certain *past* time period is commonly referred to as *backtesting*. It is used to measure how a particular model would have performed in the past. However, a good performance in the past is no assurance of good performance in the future [43], and this is essentially due to the cycles of the market and the economy. When performing a backtest with financial data, getting results is usally not enough. Professional investors also want to know how the model would have performed when compared with the "baseline"or standard, the **S&P500** which is why they use metrics such as Alpha and Beta. Initially, the end goal of this thesis was to create a decision support system that essentially operated as an investment fund. Therefore, in order to determine if the fund would perform as a viable option in a real context, the backtesting (values of Alpha and Beta) would be compared to other comparable funds. This comparison has to be done with one important detail in mind. Funds should only be compared to funds of the same type and same attributes. As an example, a fund with worldwide exposure(one that can invest in every market of the world) can not be compared with one that only invests in the U.S stock market. Some funds past performance is publicly available through different providers like the Bloomberg Terminal, Morningstar and Google Finance. This option was considered and some of it was implemented. However, throughout the development of this paper, a better approach was discovered, which **chapter 5** will detail and show its results. Essentially, instead of appliyng the regression tree model to the whole market, we are going to start with just two sectors of the market. This is useful because we are able to better understand the results from the tree with the knowledge we have about these sectors, the technology sector and the industrials sector.

Chapter 4

Implementation

4.1 Proposed System

This thesis aims to build a decision support system for investment analysis. As such, following the architecture of a decision support system presented in section 2.2, this system will be composed of four main components:

- Data source - Morningstar through Quantopian;
- Local database - SQLite3;
- Data analysis models - Pandas and DecisionTreeRegressor(Scikit-Learn);
- Model validation - Quantopian;

Each of these components faces one of the goals of this project. Findind a reliable, flexible and easy to use data source and database aims to answer the question of how an investor, without professional tools, would get access to financial data in a way that it could be worked on. Data analysis, using python and machine learning models was the main portion of the project. Through the used model it is expected that an investor could extract valuable information about the sectors of the market being analysed, namely:

- How has one particular sector (e.g. Techonology or Industrials sectors) varied throughout the period of [2003-2018];
- What fundamental indicators are more closely related to the price of a stock;
- And ultimately, even though we are not going to invest based on the predictions of the tree, we will use metrics such as the mean squared error (MSE) and coefficient

of determination (R^2) to determine the quality of the predictions made by the regression tree, which allows us to understan how this type of data can be fitted to a regression tree.

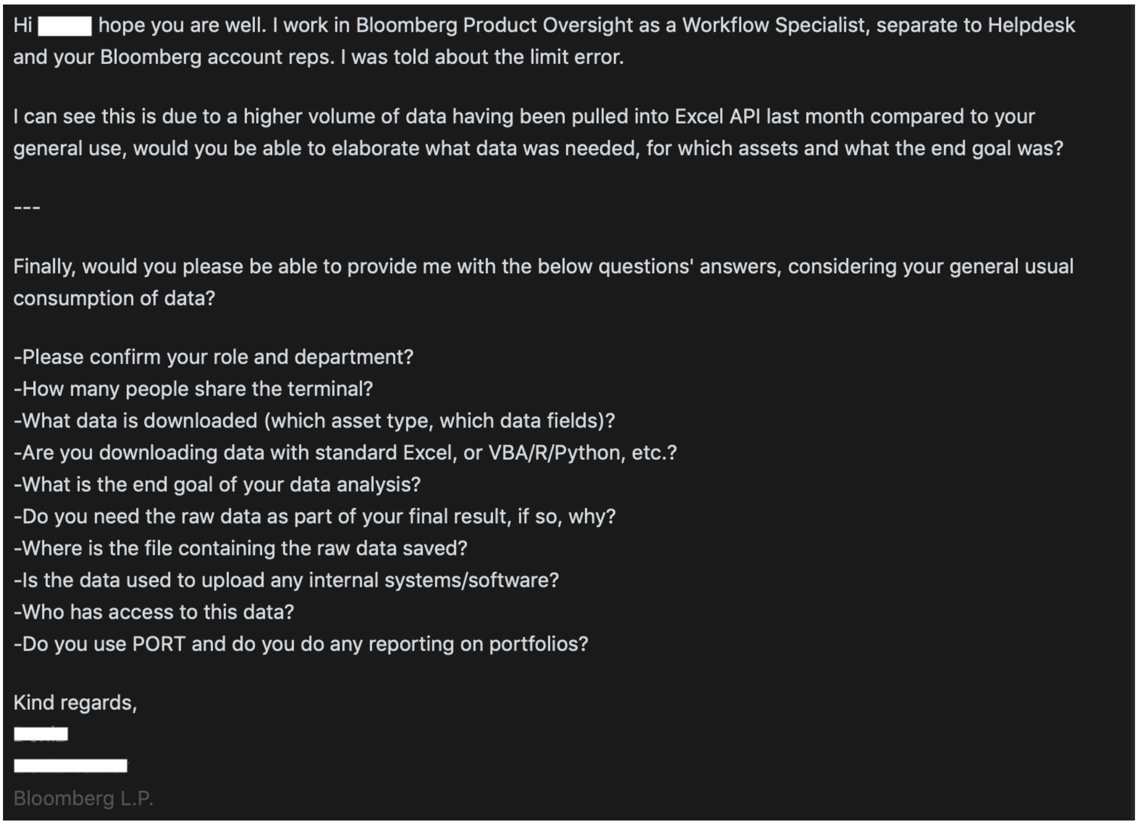
The ranking of features made by the tree is going to be used to create an investment algorithm that ranks companies based on their "performance"regarding each one of the features (i.e. fundamental indicators), similarly to the ranking algorithm detailed in section 2.4.1.2. This algorithm, implemented on a backetesting platform such as Quantopian, will provide a sense of how well this strategy would have performed in the past, which is something very valuable for a potential investor, as well as for a data scientist trying to optimize this strategy. The following sections of this chapter explain the reasoning behind the choices regarding implementation of each one of these components.

4.2 Acquiring Financial Data

The technology era we are currently living in has brought us a new, important and very valuable commodity: **information** (i.e data). This is true for every type of data, be it data for research and development in an academic context, enterprise context or others, and for almost every industry. However, the more direct the path is from collecting and analyzing the data and translating that analysis into revenue, the more valuable it is. That of course is the case with financial data. Without the data with which investors analyze a security (e.g stocks, bonds, foreign exchange, etc), they would not be able to decide how and in which assets to invest, and without these types of investments, the economy and our world would come to a halt.

As shown previously, there are some sources that provide users with that data. However, to remove one substantial possible **point of failure** of our analysis, we needed to collect the data from a reliable source. The Bloomberg Terminal is possibly the worlds most trusted financial data provider. That makes it incredibly expensive, and a very limited and closed system regarding the data that it lets its users transfer out of the system. The main way to extract data from the Bloomberg Terminal for analysis is through a Microsoft Excel plugin. Due to the restrictions in data extraction that Bloomberg establishes is it hard to automate this process. An alternative would be **BQL** - Bloomberg Query Language. This method of extraction has some similarities with SQL, which theoretically would make it a great fit for the type of task we are trying to accomplish. However, it quickly exceeds the volume of data that the available license allows for (see image 4.1).

This caused several problems and delays in the development of the thesis project, which led to the search for an alternative source. The source of data used for this project ended up being Quantopian [63]. Quantopian offers no-setup and in-browser research, backtesting, and live trading environments. It contains US equities priced at a minute level and point-in-time corporate fundamentals as well as access to a number of other datasets through a partnership with Quandl [62]. Fortunately, the data required for this



Hi [redacted] hope you are well. I work in Bloomberg Product Oversight as a Workflow Specialist, separate to Helpdesk and your Bloomberg account reps. I was told about the limit error.

I can see this is due to a higher volume of data having been pulled into Excel API last month compared to your general use, would you be able to elaborate what data was needed, for which assets and what the end goal was?

Finally, would you please be able to provide me with the below questions' answers, considering your general usual consumption of data?

-Please confirm your role and department?
-How many people share the terminal?
-What data is downloaded (which asset type, which data fields)?
-Are you downloading data with standard Excel, or VBA/R/Python, etc.?
-What is the end goal of your data analysis?
-Do you need the raw data as part of your final result, if so, why?
-Where is the file containing the raw data saved?
-Is the data used to upload any internal systems/software?
-Who has access to this data?
-Do you use PORT and do you do any reporting on portfolios?

Kind regards,
[redacted]
Bloomberg L.P.

Figure 4.1: Extracting a large volume of data resulted in inquiry from Bloomberg representatives and a temporary blocking of the license.

thesis is facilitated through Quantopian, but the main provider is Morningstar, which similarly to Bloomberg is very well renowned and trustworthy.

4.2.1 Storing the Data

After verifying the legitimacy of the data, there was a problem of storing it for research with *scikit-learn*. Quantopian provides its users with a very compreensive research toolset by incorporating its API on Jupyter Notebooks. As such, an ideal scenario would allow us to extract, do all the pre-processing necessary, build the decision tree and then analyse it inside Quantopian notebooks. However, in order to avoid the extraction of some proprietary data sets, Quantopian does not allow outputing any kind of files to the users computer, and that also includes files like the one generated by the decision tree. A **tree.dot** file for example.

In order to surpass this issue, there was a need to:

1. Extract data from Quantopian by creating an output to the console in text format;

2. Gather the output in text format;

3. Use a text processing program to insert data into SQLite3 and storing it, only for the purpose of this research project;

In order to build a Regression Tree with a significant number of data points, data was extracted from the beginning of 2003 to the month of December of 2018. That way we can capture various and diverse periods of the markets, with both downturns in the economy and periods of economic growth. Extracting this data however was not a one step process. The extraction process was made in batches of quarters and of sectors, which means something to the effect of:

1. Extracting data for the 1º quarter of 2003, technology sector;
2. Extracting data for the 2º quarter of 2003, technology sector;
3. Extracting data for the 3º quarter of 2003, technology sector

All the way to the 4th quarter of 2018, and then repeating the process for the Industrials sector. Fortunately, we are dealing with fundamental indicators which are provided by companies on a quarterly and annual basis. Otherwise this process would not be feasible.

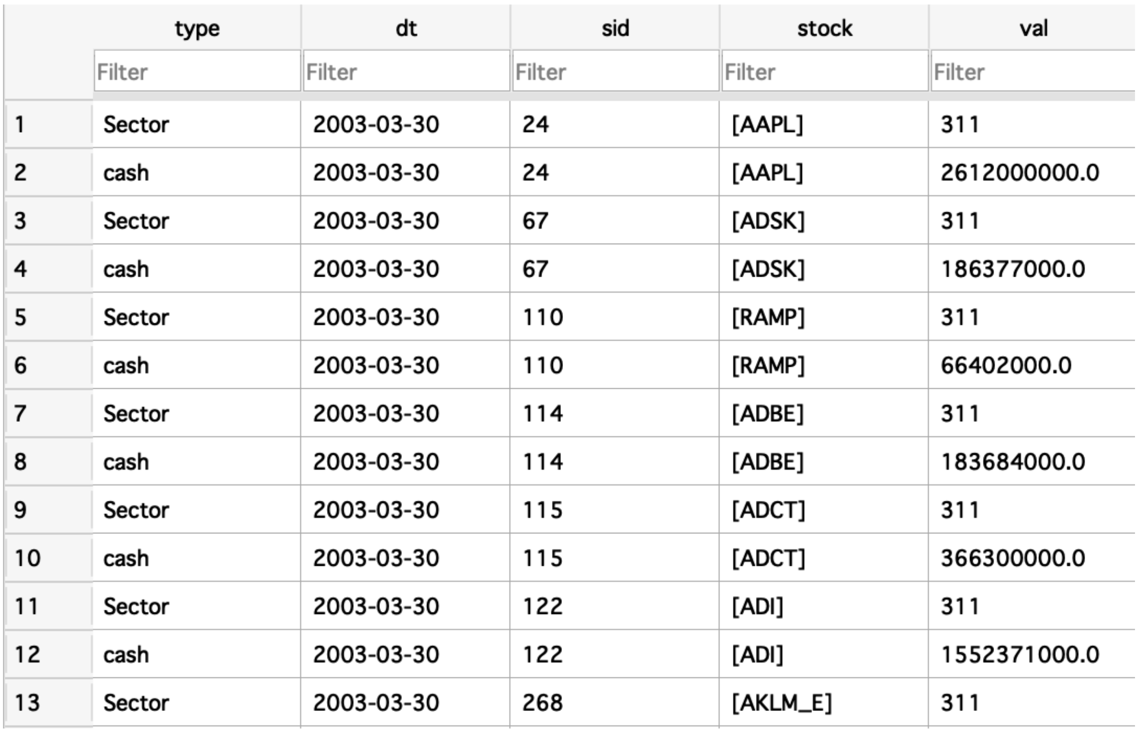
This process creates a data set that ended up containing 106 distinct companies, 56 of which from the Technology Sector and the remainning 50 from the Industrials Sector,with data about these companies from 2003 up to 2018, providing a total of 44268 data points (rows). For the matter of storing the data, two main possibilities were identified during the preparation for the thesis: Microsoft Excel, as it is already used for both storing data and for analysis of the data in the banking and financial industry, and SQLite3, as it reduces complexity, provides portability across all operating systems and provides excellent performance both in reading and writing operations. Because of the amount of data used for this project, Excel did not provide as much flexibility as SQLite3. This was the case for both how much data it could store whilst maintaining sufficient performance, but also in querying data. Because of this, the extracted batches of data were saved in text format and later imported to SQLite3.

Figure 4.3 shows a small sample of the fundamentals table, in which:

- type - identifies the type of financial indicator;
- dt - identifies the corresponding date of the data;
- sid - Stock id, used by Quantopian;
- stock - ticker of the company;
- val - value of the fundamental indicator (type) of company (stock) in date (dt).

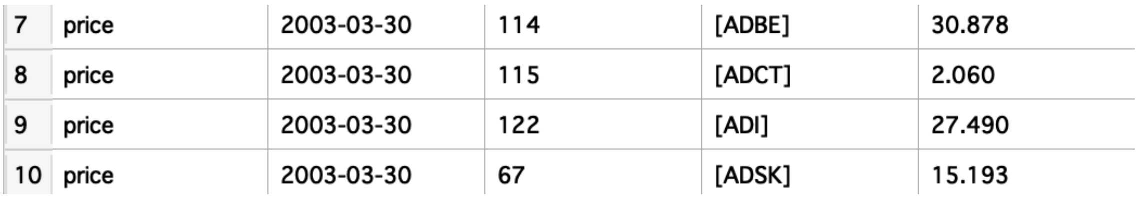
Where the primary key is composed of: type, date, stock. Possible values for "type", which are features of the database are described in section 2.3 of chapter 2.

Price data is stored on a separate table, although the only difference is the fact that it only has one "type", which is **price**, and because of this, the primary key is composed of: date, stock. The pricing data was stored in a separate table (see Figure 4.4).



	type	dt	sid	stock	val
	Filter	Filter	Filter	Filter	Filter
1	Sector	2003-03-30	24	[AAPL]	311
2	cash	2003-03-30	24	[AAPL]	2612000000.0
3	Sector	2003-03-30	67	[ADSK]	311
4	cash	2003-03-30	67	[ADSK]	186377000.0
5	Sector	2003-03-30	110	[RAMP]	311
6	cash	2003-03-30	110	[RAMP]	66402000.0
7	Sector	2003-03-30	114	[ADBE]	311
8	cash	2003-03-30	114	[ADBE]	183684000.0
9	Sector	2003-03-30	115	[ADCT]	311
10	cash	2003-03-30	115	[ADCT]	366300000.0
11	Sector	2003-03-30	122	[ADI]	311
12	cash	2003-03-30	122	[ADI]	1552371000.0
13	Sector	2003-03-30	268	[AKLM_E]	311

Figure 4.2: SQLite3 sample of the database



7	price	2003-03-30	114	[ADBE]	30.878
8	price	2003-03-30	115	[ADCT]	2.060
9	price	2003-03-30	122	[ADI]	27.490
10	price	2003-03-30	67	[ADSK]	15.193

Figure 4.3: SQLite3 sample of pricing data

Perhaps a more typical modelling approach for a data set like this one would have every "type"as a column in the database. Certainly, some datasets used as references, like the Boston House dataset or the Iris Flower dataset present each feature as a column. However, in this case the choice was to have the "type"column which then has for a given stock in a given date, all the different fundamental indicators. Although this type of modelling creates a much larger data set in terms of rows, it was appropriate for this case because it allows for easy expansion of the data set, without the need for a change in the schema. If at some point there is a need for adding another fundamental indicator, there is no need to add another column.

4.3 Programming Language

There is an ongoing debate about what programming language is more suitable for data analysis, modelling and machine learning. The **R** programming language and **Python** are the most commonly used. However, as previously established in section 3.0.6, Python brings a few advantages when dealing with a project like this one. So, in order to analyse the data, the system will use Python and some of its various libraries, namely:

- numpy - Provides access to mathematical funtions as well as support for multi-dimensional arrays;
- matplotlib - Provides tools for plotting the results in graphical form;
- pandas - Offers the main data structures used to manipulate and analyse data after it has been read from the SQLite database;
- scikit-learn - It provides the regression model used: **Decision Tree Regressor**.

These libraries will not only allow to perform the analysis phase, but also the model validation phase in the Quantopian environment.

4.4 Cleaning Data

According to **IBM** [7], data scientists and machine learning developers can spend up to 80% of their time cleaning data. It is something that really must be done before starting to work on fitting the data. There are various factors that create this need for cleaning data, but as a general rule the biggest one is missing values. In this case, even though companies are obligated to publish their quarterly and anual results, sometimes there are missing values for just a particular feature. If we take **dividends** as an example, we know that some companies pay dividends and others do not. So if we tried to extract these values from a company that does not pay a dividend we will most likely have a missing value in our database, and this is just one of many possible cases where this can occur. Fortunately, our data provider has documentation that tells us that missing values are represented in the database by "NaN"as a string, so in order to identify which feature suffers from it the most:

```
#Replace Str 'NaN' for numpy.nan
data_ex.replace("NaN", np.nan, inplace=True)
#Returns the total NaN values for each feature
print(data_ex.isnull().sum())
```

This analysis tells us that:

- **date** 5.1% NaN values
- **sid** 5.3% NaN values
- **stock** 0 % NaN values
- **Sector** 0 % NaN values
- **cash** 5.1% NaN values

- **cost of revenue** 5.2% NaN values
- **current assets** 5.3% NaN values
- **current liabilities** 5.3% NaN values
- **free cash flow yield** 5.2% NaN values
- **gross profitability** 5.12% NaN values
- **investment to asset** 0.4% NaN values
- **sales** 0.5% NaN values
- **share issuance** 4.5% NaN values
- **total debt** 5.5% NaN values
- **total equity** 5.1% NaN values

Tipically, whenever there is a feature with more than 50% of missing data we should think about removing it. Fortunately this is not the case with any of the features we are working with. Instead, one very common strategy we could use to combat missing values is calculating the average of each feature and changing the missing values to that value.

```
#Deal with NaN values in Training Set
    for column in data_training[[features]]:
        data_training[column].fillna(data_training[column].mean(),inplace=True)

    #Deal with NaN values in Testing Set
    for column in data_testing[[features]]:
        data_testing[column].fillna(data_testing[column].mean(),inplace=True)
```

This method does not change the distribution of our dataset which makes it a viable when dealing with missing data.

4.5 Normalizing Data

Commonly refered to as normalization, **feature scaling** is used in statistics for adjusting values in order the compensate for those on different orders of magnitude, removing the impact that outliers, an unusually big or small number has on other data points of the data set. This topic calls for somewhat of a debate because some experts say that all data should be normalized, and some experts say that models like the decision tree do not require normalization. This discussion is specially present when working with tree based methods. An algorithm such as CART measures possible split points based on the maximum reduction of variance. The scale of values in the dataset should not impact the performance of the model. Scaling is commonly referred to as a monotonic transformation of data, because it is a function that transforms real numbers into real numbers, preserving the original order of the data set. This should equate into the lack of necessity to scale data before building the tree. In *Classification and Regression Trees* [8], **Leo Breiman, Jerome H. Friedman, Richard A. Olshen, Charles J. Stone** (cited by **Tom Mitchell** [54]) describe this phenomenon: "The models are invariant under transformations in the predictor space."And in detail: "In the case of regression trees, heteroscedasticity, or the tendency for higher-value responses to have more variation, can be problematic. Because regression trees seek to minimize within-node impurity, there will be a tendency to split nodes with high variance. Yet, the observations within that node may, in fact, belong together. The remedy is to apply variance-stabilizing transformations to the response as one would do in a linear regression problem."This led to the decision to implement a scaling technique in our data set in order to observe the results in the data itself and the produced tree. These results are analysed in chapter 5.

Even when normalizing data, there isn't one single method that fits all purposes and types of data [29, 86]. It is generally reported that when dealing with financial data, the most commonly used method for feature scaling is the **Min-Max Method**.

4.5.1 Rescaling (Min-Max normalization)

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{4.1}$$

Perhaps the simplest scaling method, this technique allows for the scaling of all values to a range in [0,1] or [-1,1], depending on the type of data. This method is applied to each feature of the data set, which in this case corresponds to fundamental indicators, as well as to the target variable. x is the value in consideration for normalization, *min(x)* and *max(x)* are the minimum and maximum values of the series of values of that feature. It is important to note that **min** and **max** values of each feature of the training set are recorded so they can later be applied to perform the normalization on the testing set. This is important because it ensures that the training set is in no way influenced by the testing set (and the opposite as well). In order to better understand the results of applying this method we can look at a general description of the data set before and after normalization:

This description of the training set and testing set after normalization provides a very interesting perspective of the market. Since normalization of the testing set is performed using the same scalar as the one used to perform normalization in the training set, min and max values of the testing set are no longer restricted to a range between 0 and 1. The fact that min values are negative in some features and max values are higher than 1 tells us that the performance of the companies in the testing set varied a substantial amount. This is interesting because it can provide an investment professional with information about the periods of growth and downturns in the market.

Looking at how min and max values also vary in the pricing portion of the testing set (usually named the true values) we can see how pricing varied between the period of the training set [**2003-03** - **2018-03**] and the testing set which is [**2018-12**]. We can

Training set	Cash	Cost of Revenue	Current Assets	...	Share issuance	Total debt	Total equity
count	1680	1680	1680	...	1680	1680	1680
mean	0.365723	0.437091	0.424254	...	0.425954	0.323834	0.460925
std	0.292301	0.298117	0.296994	...	0.345778	0.348258	0.324830
min	0	0	0	...	0	0	0
25%	0.105394	0.215960	0.168549	...	0.071625	0.008061	0.159444
50%	0.332522	0.393429	0.406714	...	0.389905	0.165644	0.454664
75%	0.56180	0.686233	0.647280	...	0.725537	0.592686	0.754825
max	1	1	1	...	1	1	1

Table 4.1: Description of features in training set

Testing set	Cash	Cost of Revenue	Current Assets	...	Share issuance	Total debt	Total equity
count	57	57	57	...	57	57	57
mean	0.527928	0.843537	0.566000	...	0.274833	0.862341	-0.212531
std	0.526108	0.426159	1.186922	...	1.756693	2.342546	5.556403
min	-0.792868	-0.162288	-5.895095	...	-7.932733	-2.000000	-29.374834
25%	0.200990	0.539964	0.371418	...	-0.016141	0.078125	0.733612
50%	0.357203	0.899314	0.773179	...	0.303701	0.620751	0.920018
75%	0.758805	1.135232	1.052129	...	0.942176	0.963397	1.105248
max	1.986132	1.756098	1.586133	...	6.694677	1.423123	1.790965

Table 4.2: Description of features in testing set

Pricing training set	Price
count	1680
mean	0.368945
std	0.280997
min	0.00
25%	0.136769
50%	0.312755
75%	0.569656
max	1.00

Table 4.3: Description of pricing in training set

Pricing true set	Price
count	57
mean	0.859302
std	0.484455
min	-0.233083
25%	0.495483
50%	0.887024
75%	1.114054
max	2.278752

Table 4.4: Description of true value set

see that there are instances where the minimum price was lower, and other times where maximum pricing was higher. However, looking at the mean values of each of the sets we can establish that the time period of [**2018-12**] was priced higher than the previous time period. This information, with this level of detail, is something that only a substatially complex and expensive tool like the Bloomberg terminal would provide an investment professional.

4.5.2 Alternatives to normalization

An alternative to scaling data would be to use ratios instead of the "raw"fundamental indicator values. This approach would also enable us to compare different companies, independently of the scale of values of each one. However, tipically ratios are devised using price, for example, **Price-Earnings Ratio**. This is a commonly used metric to compare companies within the same sector of the market. The problem this brings when using a decision tree is that by introducing the price of the company into a feature that the tree will analyse, we are inherently increasing the correlation between this feature and the target variable, which for a decision tree is not ideal.

4.5.3 Applying Normalization

Before normalizing data, the data set is divided in training and testing set. This is important because we should not, in any way mix both sets of data or influence one another with normalization. We should perform normalization over the training set, and then perform normalization over the testing set, but using **min** and **max** values determined in the training set. This allows us to see whether the model is able to conform to new and "never seen"data points.

```
1. #Normalization of Training Set
2. from sklearn import preprocessing

3. x_training_values = x_training.values #returns a numpy array
4. x_min_max_scaler = preprocessing.MinMaxScaler(copy=True, feature_range=(0,
    1)).fit(x_training_values)
```

```
6   5. x_scaled_training = x_min_max_scaler.transform(x_training_values)
7   6. x_training = pd.DataFrame(x_scaled_training)
8
9   7. y_training_values = y_training.values #returns a numpy array
10  8. y_min_max_scaler = preprocessing.MinMaxScaler(copy=True, feature_range=(0,
       1)).fit(y_training_values)
11  9. y_scaled = y_min_max_scaler.transform(y_training_values)
12  10. y_training = pd.DataFrame(y_scaled)
13      ...
14  11. #Normalization of Testing Set
15  12. x_testing_values = x_testing.values #returns a numpy array
16  13. x_scaled_testing = x_min_max_scaler.transform(x_testing_values)
17  14. x_testing = pd.DataFrame(x_scaled_testing)
```

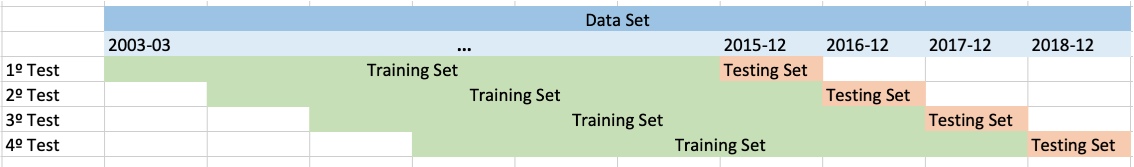
In order to ensure that the previous technique is applied, we first fit the data of the training set (line 4), then transform the data using those parameters. In order to transform the testing set with those same parameters, we use the same variable as before, the **x_min_max_scaler** (line 13), and this process is applied for each company present in the dataset on an individual basis. This explains how the description of the data set shown figure 4.2 has values lower than 0 ang higher than 1. Nonetheless, having values that the tree was never trained with, such as values lower than 0 and higher than 1 would have a big impact on the ranking of features and on the predictions of the tree. To fix this phenomenon a decision was made to perform the following transformation for every x, where x is every possible value of every feature:

- if x <0, x = 0;
- if x >1, x = 1;

4.6 Sliding Window

Given that we are working with time series data, it is interesting to perform tests in different time periods. Commonly referred to as a data transmission protocol, a **sliding window** can actually be used as a method for performing tests in time series data. Although used in a data stream context, one paper [38] describes this method as "With the sliding window method for evaluation we can obtain detailed and precise evaluation over the whole period of training/learning without the influence of earlier errors. With this method we evaluate the model over a test set determined by a window of examples which the algorithm has not used for training. The window of examples manipulates the data like a FIFO (first-in-first-out) queue". We can take this idea from the data stream context and adapt it to our types of tests in the following manner:

As described by the image 4.5, based on the sliding window principle, the performed tests will be: When performing these tests we can also compare r2 score results from the different "windows"and analyse how well our model would have performed in different market environments (detailed in chapter 5).



	Data Set									
	2003-03			...			2015-12	2016-12	2017-12	2018-12
1º Test	Training Set						Testing Set			
2º Test		Training Set						Testing Set		
3º Test			Training Set						Testing Set	
4º Test				Training Set						Testing Set

Figure 4.4: Sliding Window

Test	Training Set	Testing Set
1º	[2003-03 -> 2015-09]	[2015-12]
2º	[2004-03 -> 2016-09]	[2016-12]
3º	[2005-03 -> 2017-09]	[2017-12]
4º	[2006-03 -> 2018-09]	[2018-12]

Table 4.5: Sliding Window based tests

4.7 Building a Regression Tree

We now discuss the process of building a regression tree. Roughly speaking, there are two steps.

1. We divide the data into two sets, the training set and the testing set;
2. Clean the data (fix missing values);
3. Scale data (dependent on the type of test performed);
4. We fit the data from the training data set to the tree;

Given that we are dealing with time-series data, it is imperative that in the first step described, the **time boundaries** are well defined, not making the mistake of using "future"data (or data used for testing) to create the tree. To make sure of this, we divided the training and testing sets of data based on the sliding window principle. For example:

- **Training** : from 2003-03-30 to 2015-09-30
- **Testing** : 2015-12-30

Listing 4.1: Splitting data into Training and Testing Sets with time boundaries

```
#Training Data (Industrials Sector Id == 310.0)
data_training = data_ex[data_ex.Sector == 310.0]
data_training = data_training[data_training.date < "2015-09-30"]
x_training = data_training[['cash','cost_of_revenue', 'current_assets', '
    current_liabilities','free_cash_flow_yield', 'gross_profitability',
'investment_to_asset', 'sales', 'share_issuance', 'total_debt',
'total_equity']]
y_training = data_training[['price']]

#Testing Data (Industrials Sector Id == 310.0)
```

```
data_testing = data_ex[data_ex.Sector == 310.0]
data_testing = data_testing[data_testing.date == "2015-12-30"]
x_testing = data_testing[['cash','cost_of_revenue', 'current_assets', '
    current_liabilities','free_cash_flow_yield', 'gross_profitability',
'investment_to_asset', 'sales', 'share_issuance', 'total_debt',
'total_equity']]
y_testing = data_testing[['price']]
}
```

When working with machine learning algorithms, such as decision trees, it is common to have another set of data called **validation set**. This set is used to tune certain parameters of the regressor. Similarly to the testing set, it should follow the same probability distribution as the training data set [87]. It is used not only for validation but also to avoid overfitting [69]:

"Since our goal is to find the network having the best performance on new data, the simplest approach to the comparison of different networks is to evaluate the error function using data which is independent of that used for training. Various networks are trained by minimization of an appropriate error function defined with respect to a training data set. The performance of the networks is then compared by evaluating the error function using an independent validation set, and the network having the smallest error with respect to the validation set is selected. This approach is called the hold out method. Since this procedure can itself lead to some overfitting to the validation set, the performance of the selected network should be confirmed by measuring its performance on a third independent set of data called a test set."

This process of splitting the original data set into training and validation sets is known as *cross-validation*, for which there are various methods for dividing data. Although this is a useful method for validating the model and working on problems like overfitting, this method implies shuffling data which in the case of time-series data should not be done because it undermines the intrinsic temporal dependencies of the data set.

Listing 4.2: Initializing and fitting a Decision Tree Regressor

```
#This creates the Regression Tree
clf = tree.DecisionTreeRegressor(...)

#This fits the training data to the tree
clf = clf.fit(x_training, y_training)
}
```

4.8 Pruning

The process described above may produce good predictions on the training set, but is likely to overfit the data, leading to poor test set performance. This is because the resulting tree might be too complex. A smaller tree with fewer splits might lead to lower variance and better interpretation at the cost of a little bias. Pruning refers to the removal of those branches in our decision tree which we feel do not contribute significantly to our decision process. There are various methods for carrying out pruning. Some are bottom up, meaning we prune the tree starting from the leaves, others are top down, meaning we start puning from the root of the tree. Additionally, sometimes referred to as "pre-pruning", stopping parameters like **minimum samples to split**, **maximum depth** and others (described in section 5.1.1) can also be helpful to avoid overfitting, although by stopping too early, these parameters can also lead to underfitting. This method will be used in order to decrease the complexity of the tree because it does not require too much optimization by a potential user, with the understanding that additional performance could gained from a more sophisticated and mathematically rigorous method.

4.9 Backtesting - Quantopian

From the results obtained from the **regression tree** we will take into consideration:

1. The importance of each feature in determining the price;
2. The MSE and R^2 Score for each prediction;

Both of these results already give us (and an investment Manager) additional information regarding the evaluation of possible investments, which in essence is the goal of this thesis. For an investment manager, after collecting and analysing this data, it is necessary to test assumptions and be able to get a sense of how his/her model would have performed in the past. As mentioned previously, this process is called backtesting. **Quantopian.com** provides a platform that allows financial data scientists to not only study the data, using tools like Jupyter Notebooks, but also a proprietary backtesting module, which allows for the implementation of an investing algorithm where the investor can place buy and sell orders with a variety of parameters like maximum leverage, cost of trade, stop-loss and many others. This gives the investor the opportunity to test his/her algorithm in various simulated world conditions.

Chapter 5 details some of the results obtained when performing tests in Quantopian. Nevertheless, it is important to establish what metrics we are going to analyse at when looking at those tests. *Alpha* and *Beta* are the main metrics when comparing our "portfolio"to the benchmark. These metrics were already explained in section 3.1. The most common benchmark used in financial markets, and in particular the stock market, is the S&P 500. Even in the Quantopian environment, this is the default benchmark used, although we could change it. This is important because the *Alpha* and *Beta* values returned are "against"the benchmark. The investment algorithm implemented in Quantopian is simple, but very compatible with the information collected from the regression tree. As an example, lets say that the regression trees provides the following importances for each feature:

- **cash** 0.01104782
- **cost of revenue** 0.01107045
- **current assets** 0.01341082
- **current liabilities** 0
- **free cash flow yield** 0
- **gross profitability** 0.24860227
- **investment to asset** 0
- **sales** 0
- **share issuance** 0.34133083
- **total debt** 0.07364987
- **total equity** 0.30088794

We can then take the universe with every company that we could invest in, and the day after quarterly results are announced, we rank each company based on every one of these indicators. It essentially follows the same principle as the investment algorithm coined by **Joel Greenblatt**, described in section 2.4.1.2, but instead of only using two features : Return on Capital and Earnings Yield, we are going to use these 11 indicators, and use the importances provided by the tree to rank our universe of companied based on that.

$$CompanyRanking = \sum_{i=Cash}^{TotalEquity} (Rank_i * Importance_i) \tag{4.2}$$

- i represents each feature;
- *Rank* is the ranking of each company regarding i;
- *Importance* is the importance of i detemined by the regression tree.

From this ranking, in each quarter, we balance our portfolio by selling companies that were there previously and are not in the top 20 of our ranking, and by adding companies that are in the top 20 rank and were not in the portfolio. If some company consistently remains in the top 20 rank, we do not make any changes on its weight in the portfolio.

Chapter 5

Experimental Works

As described previously in section 3.1.4, instead of applying the decision tree model to an hypothetical S&P 500 fund like initially planned, we are starting by applying this model to two sectors of the market. This decision was made based on the fact that is it a lot more difficult to interpret the results of the whole market. And by analysing results from smaller portions of the market we can more easily improve the model. This chapter is dedicated to describing the two case studies used in order to test this system, as well as its results. Firstly, the goal is to see if through analysing data using a regression tree we can extract useful and additional information about the Technology Sector and the Industrials Sector. This is described in the subsections named: **Understanding Companies Fundamentals** and **MSE and R^2 Comparison** .The second phase, described in the subsection named **Backtesting Feature-Based Investing**, intends to check whether building a **feature based investment strategy** using that information would perform well on the financial markets. As mentioned in section 4.5, we are also going to observe the effects that scalling has on the data set and on the regression tree itself.

5.1 Technology Sector

The technology sector is one where investment professionals still have quite a bit of difficulty when analysing companies. This is due to the fact that companies in this sector have not followed the same level of scrutiny as other companies. For example, sometimes, even when companies from this sector do not show any profits and instead show high levels of debt, they still seem to perform well in the market. This is one of the reasons why this sector is interesting and was chosen as a first case study.

5.1.1 Understanding Companies Fundamentals

In order to understand what fundamental indicators have the highest level of correlation with the price of a stock in this sector, we could observe how the generated tree is structured, but we can also make use of a method provided by scikit learn, and can then plot that information. The importance of a feature is computed as the total reduction of variance brought by each feature.

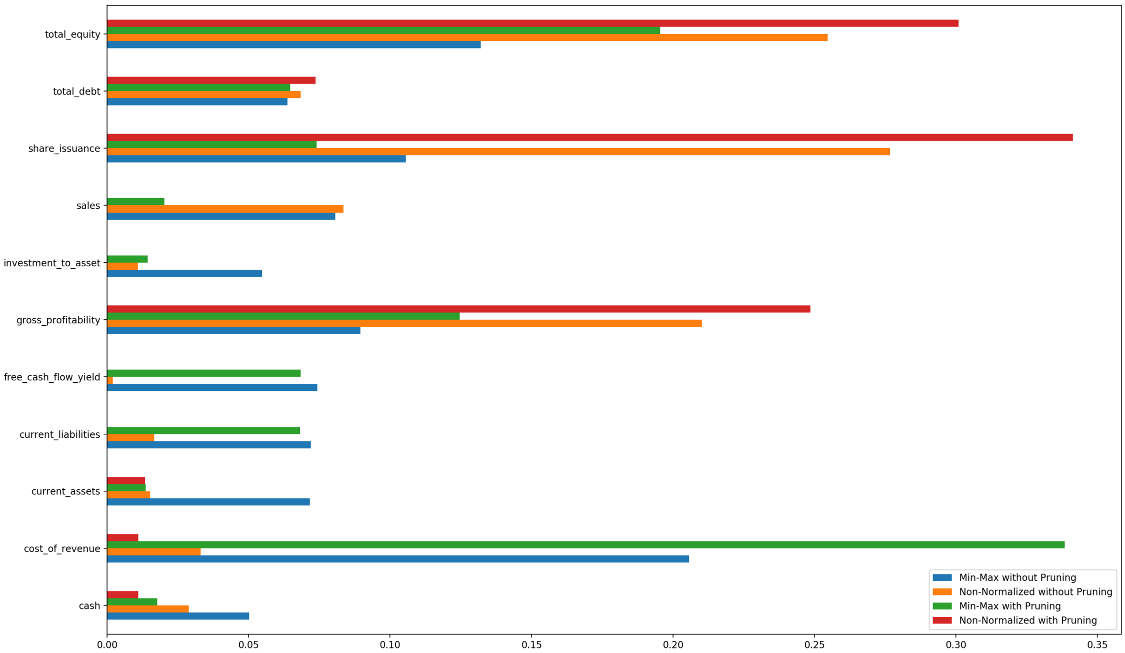


Figure 5.1: Technology Sector Features

The previous plot reports the feature importances from a test performed with the entire time period of collected data, which is from 2003 to the end of 2018, with the aim of getting an overall ranking of features, independently of time period, whereas the results shown in table 5.1 have a more localized time frame that allows to observe variations in importance.

These first results highlight a few interesting aspects about how scalling and pruning have an impact on data. Firstly, we can see that features like **investment to asset** and **sales** would likely lead to overfitting. We can assume this because they have a greater importance on the tree before pruning is executed (with min-max normalization). There are also some features where after pruning is applied the importance of the feature increases. This might point to a feature that would lead to underfitting. This is the case with: **Cost of Revenue**, **Total Equity**. It is also interesting to note that features that have the most importance when the tree is fully formed (without any pruning) still mantain that status after pruning is applied. This is the case with the top three features with the most importance: **Total Equity**, **Share Issuance** and **Gross Profitability**. The bottom three features with less importance are **cash**, **investment to asset** and **current assets**.

This ranking is very interesting because it does align with the perspective that investors have of the market, sometimes claiming that technology companies are evaluated just on growth (e.g. growth of users) and not enough on fundamentals such as actual money earned and stability provided by a balance between debt and assets. Although when comparing features like **current assets** with **total debt** we can clearly see that the debt side of the equation does have more correlation in determining the price. The importance attributed by the tree to each attribute would now have to be studied by an investment professional in order to determine whether its correlation with the price is positive or negative. After that we can then apply positive or negative importance when ranking companies using these features. This is later applied in Quantopian.

The following table shows how the importance of each one of these features has evolved throughout the different periods of testing which refer back to the sliding window tests detailed in section 3.6.

	1º Test	2º Test	3º Test	4º Test	Mean
Cash	0.137	0.124	0.013	0	0.0685
Cost of revenue	0.141	0.148	0	0	0.07225
Current assets	0.026	0.013	0.012	0.115	0.0415
Current liabilities	0.011	0.004	0	0.169	0.046
Free cash flow yield	0.008	0	0	0	0.002
Gross profitability	0.371	0.357	0.049	0.051	0.207
Investment to asset	0.004	0.006	0.004	0	0.0035
Sales	0	0	0.004	0.016	0.005
Share issuance	0.175	0.231	0.360	0.290	0.264
Total debt	0	0.006	0.046	0.114	0.0415
Total equity	0.122	0.107	0.510	0.242	0.245

Table 5.1: Sliding Window based tests with No-normalization

The results from table 4.1 demonstrate how the importance of each feature as evolved during the different time periods. Without a profound understanding of the characteristics of this sector and of what happened in the markets during these different time frames it is difficult to draw conclusions. Nevertheless, we should point to the fact that importance of **total debt** has been growing, which might point to a "maturity"of the technology sector. In contrast, the results show that the importance of **gross profitability** has decreased throughout these time frames, which could indicate that investors are increasingly looking for actual money earned. However, the decrease in importance of **free cash flow** somewhat disproves this hypothesis.

It is important to note that a high value of **Share Issuance** or**Total Equity** is not indicative of a good company or that the price of that company should rise. This only tells us that these are fundamental indicators more correlated with price than other indicators. It should also be noted that this tree has had some pruning, in the form of tuning parameters, in order reduce overfitting. The tree was iteratively prunned until the R2 was near optimal. It is also interesting to note some observations made regarding the

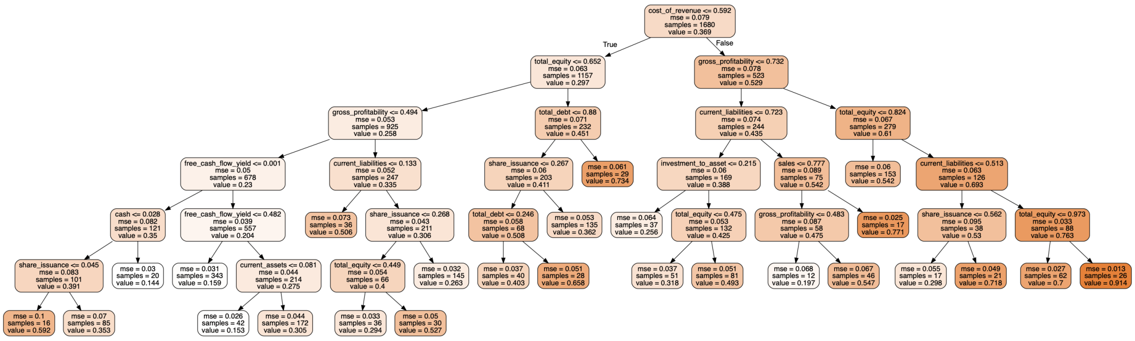


Figure 5.2: Technology Sector Tree example

parameters used for creating the tree:

- Max-Depth of the tree only has an impact on performance until max-depth equals 20;
- Once min_samples_leaf goes beyond 11, the predictive performance of the tree starts to decrease;
- A higher value of the max_leaf_nodes parameter brings some predictive gain, but tends to increase the complexity of the tree, which makes it harder to analyse by looking at it;
- The number of minimum samples necessary for splitting seems to bring the best performance to interpretability ratio with a value of 10.

5.1.2 MSE and R^2 Comparison

When performing the sliding window method for a series of tests, the results are as follows:

Test	Training Set	Testing Set	MSE	R2 Score
1º	[2003-03 -> 2015-09]	[2015-12]	0.080	0.368
2º	[2004-03 -> 2016-09]	[2016-12]	0.155	-0.404
3º	[2005-03 -> 2017-09]	[2017-12]	0.113	-0.650
4º	[2006-03 -> 2018-09]	[2018-12]	0.108	-0.018
Total	.	.	Mean: 0.114	Mean: -0.176

Table 5.2: Sliding Window based tests with Min-Max normalization

The fact that on average, the coefficient of determination (R^2) is slightly negative tells us that how the tree fits our data is not better than a simple linear regression. However, when dealing with data sets that somehow involve human behavior it is difficult to assess how good the predictions are. Furthermore, low R^2 values for a good model and high R^2 values for a bad model are plausible scenarios because this method cannot determine whether the predictions are biased [65]. The MSE mean values of these two tests, with min-max normalization and without any type of scalling should not be compared, because mse is sensitive to scale. However, when looking at R^2 for both tests, with min-max scalling (table 5.2) and without any normalization applied (table 5.3), it seems that the decision tree is able to capture a lot more variance of the data set when there is no normalization applied, because the mean R^2 is significantly higher when compared to the mean R^2 value for tests with scaled data.

Test	Training Set	Testing Set	MSE	R2 Score
1º	[2003-03 -> 2015-09]	[2015-12]	131.99	0.754
2º	[2004-03 -> 2016-09]	[2016-12]	831.32	0.398
3º	[2005-03 -> 2017-09]	[2017-12]	2670.40	0.299
4º	[2006-03 -> 2018-09]	[2018-12]	985.56	0.658
Total	.	.	Mean: 1154.81	Mean: 0.527

Table 5.3: Sliding Window based tests with No normalization

5.1.3 Backtesting Feature-Based Investing

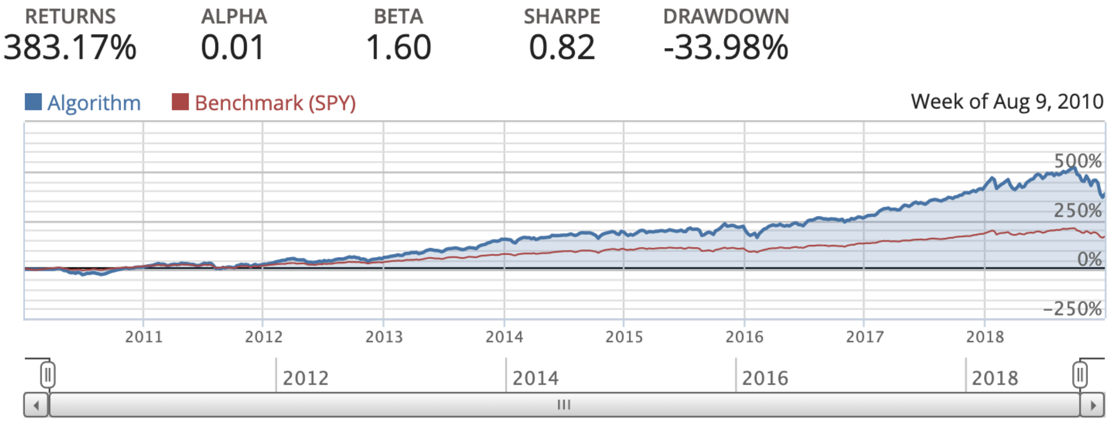


Figure 5.3: Backtesting Technology Sector 2010-2019 with Min-Max performed

The investing algorithm responsible for the results show in Figures 5.3 and 5.4 is a **Ranking algorithm** where the ranking is performed according to the importances that resulted from the regression tree. Where the results of figure 5.3 are based on the importances based on scalled data using the min-max method, and results of figure 5.4 are based on importances from data that was not normalized. These tests were performed in order to further the discussion about the impact of monotonic transformations in a decision tree. As such, features that have more importance have more weight on the ranking of which stocks to buy, and features with less importance have less weight in the ranking. The results from figure 5.3 are based on a ranking that was built with the following feature weights: **cash**:0.0177, **cost of revenue**:0.3383, **current assets**:0.0136,

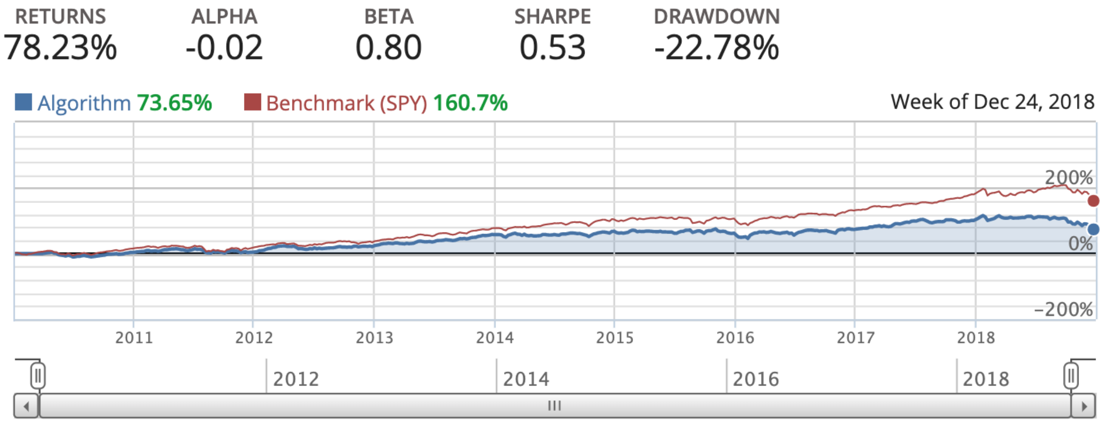


Figure 5.4: Backtesting Technology Sector 2010-2019 with No-normalization performed

current liabilities:0.0681, **free cash flow yield**:0.0684, **gross profitability**:0.1246, **investment to asset**:0.0143, **sales**:0.0202, **share issuance**:0.0740 , **total debt**:0.0647 , **total equity**:0.1954, whereas results from figure 5.4 are based on a ranking with the following feature importances: **cash**:0.0110, **cost of revenue**:0.0110, **current assets**:0.0134, **current liabilities**:0, **free cash flow yield**:0, **gross profitability**:0.2486, **investment to asset**:0, **sales**:0, **share issuance**:0.3413 , **total debt**:0.0736 , **total equity**:0.3008.

As seen by the results of the first backtest, in the time period between 01/01/2010 and 01/01/2019, this feature based ranking approach of investing would give the investor a return of 383.17%. In the same time period, our benchmark, the SPY (an Exchange Traded Fund that closely follows the S&P 500) returned 167.13%, which makes the return of our algorithm very positive. Taking a look at the Alpha and Beta values we can declare the following:

- The fact that the Beta is 0.65, tells us that this algorithm is heavily correlated with the performance of the market. This means that in periods when the returns of overall market (S&P 500) are lower, the returns of our algorithm will probably be lower, and the opposite will likely be true for when returns of the market increase.

- The fact that Alpha has a value of 0.01 tells us that our strategy of investing has earned a return that has more than compensated for the volatility risk taken. Nevertheless, ideally, this value would be higher.

Results of backtesting with data that was not scalled are quite different. Returns for the time period between 01/01/2010 and 01/01/2019 are 78.23%, the *Alpha* value is slightly below zero, which means the investments have earned a return that has not compensated for the amount of risk assumed. This seems to point to a more accurate ranking of features when the data is scalled before fitting it to the tree.

In a hypothetical scenario, the decision about using this algorithm would depend on the preference and strategy of a potential investor. If the investor needs a strategy with no correlation to the market, this might not be right one. However, if the investor is looking for a way of investing that has less risk and is correlated with the overall market, this is a viable option.

5.2 Industrials Sector

Arguably very different than the technology sector, industrials are judged and analysed very differently. This sector is composed of companies that provide industrial and commercial equipment and services, transportation, and distribution operations. Construction, farming machinery, airlines, railroads, and waste management, are some of the types of companies that compose the industrials sector. It is commonly considered a cyclical sector, meaning there is less demand for them in a down economy. It is also know for being a **capital intensive** sector, meaning that companies have to constantly invest money in equipment and infrastructure if they intend to operate. Analysing this sector will provide us information of how the decision tree model can perform when used with data with these inherent characteristics.

5.2.1 Undestanding Companies Fundamentals

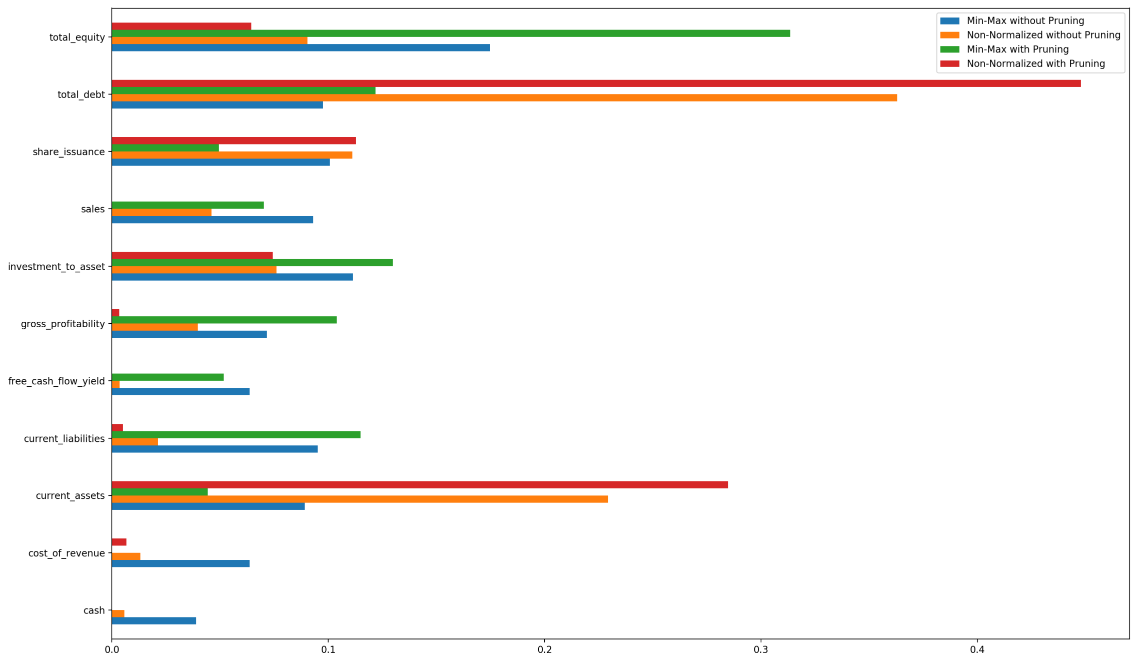


Figure 5.5: Industrials Sector Features

Figure 5.5 shows the results from a test performed in the entire time period of collected data, which is from 2003 to the end of 2018. The goal with this test was to see if it provided us with an overall ranking of features, independently of time period. Looking at the importances that the tree attributed to each feature we learn that **total debt** and **current assets** have very high importances. This is compatible with the assumptions that we established previous about this sector of the market. This means that when quarterly and annual results are announced by companies that are part of this sector, investors should pay close attention to these features. Looking at the definitions of these features (detailed in section 2.3) we can assume that there exists a positive correlation between the current assets value and price of these companies, and there exists a negative correlation with debt and the price of these companies. It is also interesting to note how these two features are some of the lowest ranked in the overall test performed with technology companies. Perhaps this is already well established in the knowledge that investors have about this sector. However, the tree gives us the actual value of importance for these indicators. To different investors this information could be used in different ways. The approach of investing in this thesis will be based on taking the value of importance of each feature and using it for a ranking based investment system. Comparing results from with no-normalization without pruning with those with no-normalization with pruning we can postulate that maybe features like **total debt** and **current assets** would lead to underfitting because importances with pruning are higher than when no pruning was performed, and **total equity** would maybe lead to overfitting because importances with pruning are lower than when pruning was performed. However, loking at the importances of current assets with min-max normalization with pruning and min-max normalization with no pruning we can see that the opposite happens. It is unclear why this happens, but it might lead us back to discussion of whether or not this type of monotonic transformations improve the performance of the tree. And in order to get a better comparison between these two approaches applied to the market we will later compare the results of both in backtesting (see images 5.7 and 5.8).

The following table shows how the importance of each one of these features has evolved throughout the different periods of testing, and the tests mentioned in the header of the table refer back to the sliding window tests detailed in section 4.6.

	1º Test	2º Test	3º Test	4º Test	Mean
Cash	0.016	0	0	0	0.004
Cost of revenue	0.053	0	0.007	0.003	0.015
Current assets	0.028	0.096	0.121	0.116	0.090
Current liabilities	0.017	0.057	0.075	0.204	0.088
Free cash flow yield	0	0	0	0	0
Gross profitability	0.021	0	0	0	0.005
Investment to asset	0.031	0.021	0.047	0.046	0.036
Sales	0	0	0	0	0
Share issuance	0.038	0.540	0.152	0.041	0.192
Total debt	0.653	0.132	0.448	0.453	0.421
Total equity	0.137	0.153	0.147	0.137	0.142

Table 5.4: Sliding Window based tests with No-normalization

5.2.2 MSE and R^2 Comparison

When performing the sliding window method for a series of tests, the results are as follows:

Test	Training Set	Testing Set	MSE	R2 Score
1º	[2003-03 -> 2015-09]	[2015-12]	0.131	-0.257
2º	[2004-03 -> 2016-09]	[2016-12]	0.168	-1.383
3º	[2005-03 -> 2017-09]	[2017-12]	0.184	-2.173
4º	[2006-03 -> 2018-09]	[2018-12]	0.101	0.053
Total	.	.	Mean: 0.146	Mean: -0.94

Table 5.5: Sliding Window based tests with Min-Max normalization

Test	Training Set	Testing Set	MSE	R2 Score
1º	[2003-03 -> 2015-09]	[2015-12]	494.90	0.536
2º	[2004-03 -> 2016-09]	[2016-12]	459.80	0.625
3º	[2005-03 -> 2017-09]	[2017-12]	1555.20	0.543
4º	[2006-03 -> 2018-09]	[2018-12]	1262.68	0.668
Total	.	.	Mean: 943.07	Mean: 0.593

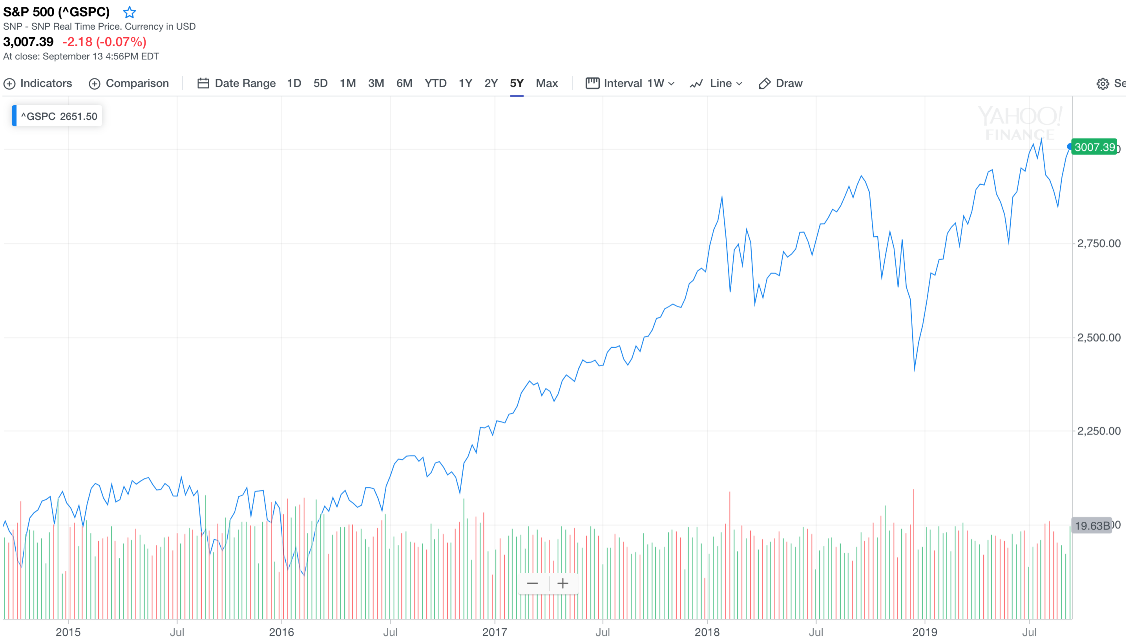
Table 5.6: Sliding Window based tests with No normalization

As we have established in section 4.5.1, **MSE** is sensitive to scale. This explains the high variation between the results shown in table 5.5 and 5.6. They are explained by the fact that the first is the result of scaled data, and the second has much higher values in the data set because scaling was not performed. Nevertheless, the mean squared error is always better the closer it is to 0. The fact that the mse mean between test 1 and 2 is lower than in test 3 and 4, for the test shown in figure 5.6, might point to the fact that the closer the tree gets to time periods between [2016-2018] the harder it becomes to predict. Again, this leads to the question of whether this difficulty is a result of the surge (high growth) in the market in this period (see figure 5.6), providing the tree with values that it was not extensively trained for.

Similarly to the tests performed with the technology sector, the R^2 coefficient is negative when measured with min-max scalling, and positive when measured without normalization. This will remain to be explained, because min-max normalization, a monotonic transformation, should not have this type of impact on the tree. Despite that, these values are close enough to 0 to be considered.

5.2.3 Backtesting Feature-Based Investing

Similarly to the results shown in section 5.1.3, these results show a high correlation to the overall market. This is shown by the *Beta* value of 0.70 in the first test, with no-normalization, and 1.28 with min-max normalization. This might be caused by a variety of factors, namely:



S&P 500 (^GSPC)

SNP - SNP Real Time Price. Currency in USD

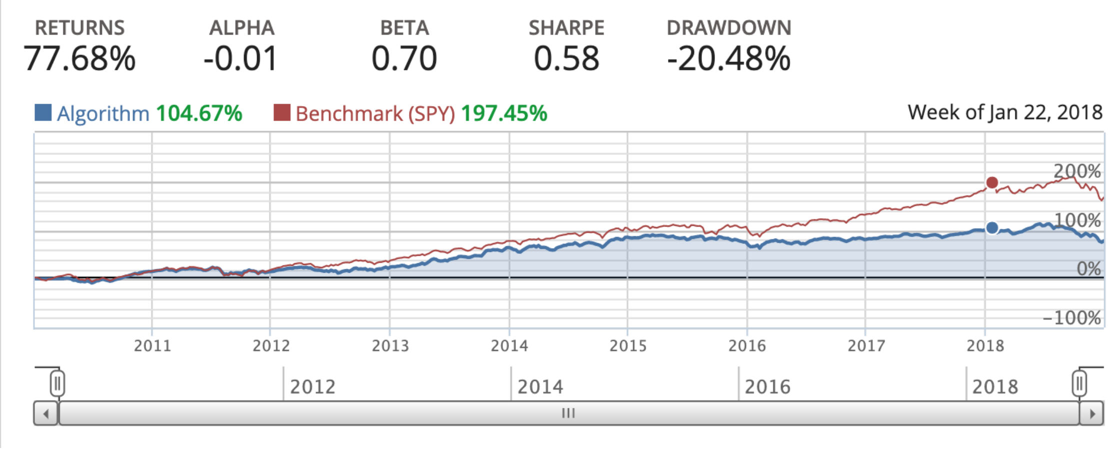
3,007.39 -2.18 (-0.07%)

At close: September 13 4:56PM EDT

Interval 1W

Line

Figure 5.6: S&P 500 surge from 2016

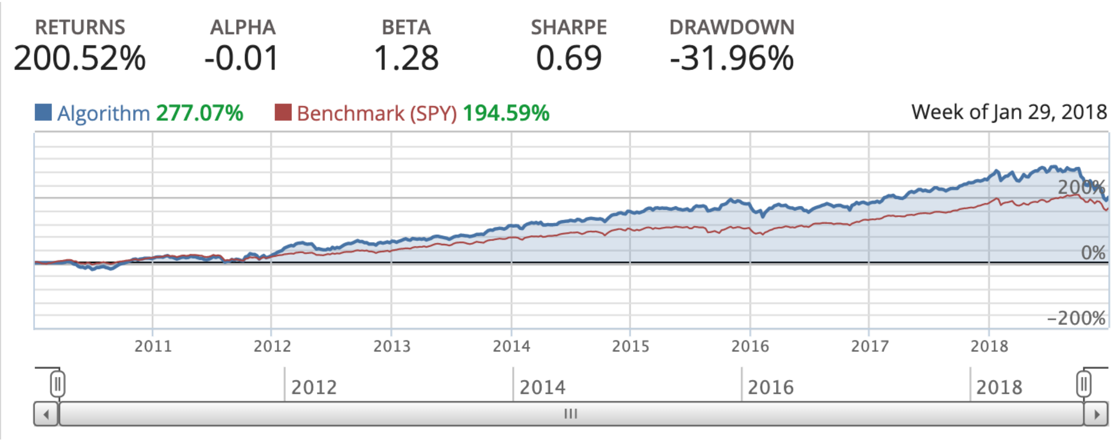


RETURNS	ALPHA	BETA	SHARPE	DRAWDOWN
77.68%	-0.01	0.70	0.58	-20.48%

🟦 Algorithm 104.67% 🟥 Benchmark (SPY) 197.45%

Week of Jan 22, 2018

Figure 5.7: Backtesting Industrials Sector 2010-2019 with No-normalization performed



RETURNS	ALPHA	BETA	SHARPE	DRAWDOWN
200.52%	-0.01	1.28	0.69	-31.96%

🟦 Algorithm 277.07% 🟥 Benchmark (SPY) 194.59%

Week of Jan 29, 2018

Figure 5.8: Backtesting Industrials Sector 2010-2019 with Min-Max performed

- This portfolio might not be sufficiently diversified in order to reduce correlation to the overall market (this was not considered because it requires advanced financial expertise, and is certainly something to improve in future work). We know this because of the Beta values of 0.70 and 1.28, which are high.
- This also may be a fact that the industrials sector is heavily weighted in the S&P 500. Although in order to clarify this assumption we would need to look at how **SPY**, the ETF in consideration, (that tracks the S&P 500) is built and allocated in terms of investment in this sector.

Looking at the test with no-normalization, overall returns are positive, with a value of 77.68%. As an example, this would mean that an investor allocating 100 000€ in the beggining of 2010, would have an overall return of 77680€ in 2019. However, an investment professional would mainly be interested in how much it outperform the S&P 500, by looking at *Alpha*, and in this case, this strategy has earned a return that has not compensated for the volatility risk assumed. We know this because the Alpha value is less than 0. Investing in an ETF like **SPY** that tracks the S&P 500 is considered a passive way of investing, because it does not require allocation strategy and does not require the investor to pick companies to invest in. This strategy will provide the investor with the same return as the overall market. So, the fact that our feature based investment strategy, which requires more effort, underperforms the **SPY** means that our strategy would be difficult to sell to potential investors. However, the fact the backtest with features ranked based on the results of importance with min-max normalization are very interesting. It seems to point to an improvement of performance when the data set is properly scalled, even though it is a monotonic transformation, which as we have seen, shoud not have an impact on the tree. However, this assumption that monotonic transformations do not have an impact might only apply to the predictions of the tree, which in this case we are not considering. We are only considering the ranking of features produced by the tree.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

For both the technology and industrials sector, the MSE and R^2 seem to point to an increased accuracy of the trees predictions when data is not scalled before being fitted to the regression tree. However, when comparing results from backtesting an investment strategy built upon ranking companies based on feature importances provided by the tree, we see a great improvement in performance when the data is scalled prior to being fitted. In order to answer to question of whether this, or other, monotonic transformation in data increases accuracy of predictions or accuracy of ranking features, there is further testing to be done. Nonetheless, the process of analysis presented in chapter 5 describes a type of workflow that provide additional information that can be used by an investor to make decisions regarding whether or not to invest in a particular company or sector of the market. Lastly, in order to answer to objectives proposed for this thesis:

1. **What are the benefits of automating the investment decision process?**
   As a general rule, automating any kind of process improves productivity and efficiency. Automating the investment decision process in particular, would also create a more thorough and complete analysis because investors could provide a larger amount of input data. It would also reduce the inherent emotional aspect of working in capital markets and investing.

2. **Is it helpful to design a schema that accommodates the collected data?** For a fundamental analysis based approach the amount of features seems to be manageable, so there wasn't a need to create a complex data model. Also, the fact that financial data is generally *Time-Series* data makes it suitable for relational databases. The data collected from this thesis was managed using two relational tables. However, if

a different user opted or had the necessity to work with more features, maybe using a technical analysis approach, there might be a need to create a more advanced data model.

3. **How should we deal with pre-processing of financial data?** Given the fact that announcing financial results to the public markets is a very regulated affair, it is uncommon to have companies not providing access to certain numbers. However, these results are public, and different data providers can then take them and organize them in different ways. With different feature names, with ratios or with empty fields. During the development of this thesis this was not very noticeable, which demonstrates the importance of using a reliable data provider like Morningstar. Nonetheless, common methods for dealing with empty fields (mean value for example) seem to be adequate to deal with this issue.

4. **How can regression trees be used for analysis of financial data provided by publicly traded companies?** Regression Trees can be used to work on financial data in a variety of ways. We could have used them to make predictions (even though there are algorithms that have shown a better prediction accuracy). But we can also used them like this thesis has shown, meaning we can use them to rank features by importance. This ranking can then be applied in a number of ways, one of which a ranking algorithm.

5. **How valuable is the information provided by a research and backtesting system like Quantopian?** As mentioned before, investing in the capital market is becoming a data science affair and as a result, financial data is becoming more valuable. However, the disruption in this field that now starts to happen has caused companies like Quantopian to provide free access to very valuable data, as well as their backtesting platform. To an investor, having this type of research tool, that allows him/her to quickly and easily test assumptions and help make decisions is crucial and therefore very valuable.

6. **How should we deal with the problem of scalling data when using decision trees?** Making decisions on what companies to invest in requires investors to analyse a lot of companies, not just one. Automating this process of analysis would allow the investor to increase the amount of companies in considerantion. However, it is commonly accepted that in orther to do so, companies need to be on the same scale to be interpretable. During this thesis, a lot of research was done regarding this particular topic, but the results obtained are not enough to draw final conclusions besides the fact that when fitting financial data to a tree model, we should at least scale the response variable, independently of the method to do it.

7. **What is the future of Decision Support Systems in the capital markets?** The fact that the financial industry is still somewhat closed oppens a lot of opportunities for

disruption. In writing this thesis project, and studying the state of the art, there is a lot of confidence in that decision support systems will become an integral part of every investment management company,bank and it will provide new tools for every professional working in this industry. Machine learning will also, increasingly be a very important resource for companies that want to create competitive advantages in capital markets.

6.2 Future Work

Particularly regarding the ranking algorithm, there are a few aspects that can be improved:

- This investment algorithm is conservative because it does not use leverage to make investments. Although the use of leverage increases risk and potential losses, properly managed it can also increase returns. This did not fall into the scope of this thesis, which is why it was not implemented.
- Currently, the raking model balances the portfolio in order to mantain 20 companies with equal weight (equal percentage of capital invested) in the portfolio, which is 5% of the capital for each company. However, perhaps this algorithm would benefit from having a stronger allocation in some companies and lower allocation in other. This is discussed in a field named **portfolio optimization**, which does not belong into the scope of this project. Nonetheless, it could be studied in order to further this field of knowledge.
- Regarding the regression tree, there are a few other scalling methods that could be implemented that could perhaps have different results.

6.2.1 Financial Data

As previously mentioned, in the development of this thesis, a lot of time was spent gathering the necessary financial data. Future studies in this field could avoid this problem by directly partnering with a data provider interested in further research in the field of computational finance and machine learning applied to capital markets. This would greatly improve the rate of improvements

6.2.2 Incorporating Economic Analysis

Public traded companies are part of the economy because they provide products, services, jobs, interest rates have a big impact on debt, for example, and many other aspects that *influence* and *are influenced* by the economy. As such, and since the goal of the study conducted in this theses was to present more information to investment professionals, one possible method for expanding this study is to incorporate economic data into the study. Unfortunately this was not possible during the thesis because the necessary data was not easily accessible and it would require a much deeper knowledge of how the economy and financial markets operate, which is not something this degree delves into.

6.2.3 Other methods of analysing data

Tree-based methods are simple and useful for interpretation. However, they typically are not competitive with the best supervised learning approaches. Hence, continuing with this approach of decision support systems could be even more interesting when using more accurate models and techniques such as **bagging**, **random forests**, **boosting**, **Gradient Boosting Machines** and many others. Each of these approaches involves producing multiple trees which are then combined to yield a single prediction. Some argue [29] that combining a large number of trees can often result in substantial improvements in prediction accuracy, at the expense of some loss in interpretation which is one of the advantages of using a single tree.

6.3 Final Remarks

6.3.1 Difficulties Encountered

The main difficulty encountered during the development phase of this thesis was definetely collecting financial data. As mentioned before investing in the capital market is becoming a data science affair and as a result, financial data is becoming increasingly expensive and difficult to get access to. Engineers are trained with a strong analytic emphasis and are also trained to make their calculations as precise as possible, which is why they are now sought after for this field of work. The fact that capital markets require much more than the ability to perform analysis of numbers, and demand part of human behavior understanding made this project challenging and thus very interesting. Starting this project also required a lot of research both in data analytics principles, machine learning and financial concepts and principles. This made the research quite long, and as a result, leaving less time to implement the desired solution, which as we learn, always keeps getting more ambicious. Never having worked with a language also required some practice and continuous improvement.

6.3.2 Lessons Learned

Every difficulty mentioned previously provided an opportunity for learning. Firstly, this project emphasized the importance of getting as much information from reliable sources as possible. Trying to make some type of advancement in any field is only possible if we first learn from the work of previous experts. Never having worked with a language like Python also opened a new way of thinking and approaching a problem. Being at its core a general-purpose languange, Python provides and easy and fast way of working with any kind of data, and using it for the development of this project created an opportunity for developing skills for working with Python in the future.

Bibliography

[1] Beecher Tuttle. *Ranking the most in-demand programming languages in banking technology*. [Online; accessed 15-February-2019]. 2018. URL: https://news.efinancialcareers.com/uk-en/137065/the-six-hottest-programming-languages-to-know-in-banking-technology.

[2] *ACATIS*. [Online; accessed 05-August-2019]. URL: https://www.acatis.de/en/investmentfunds/investmentfunds/.

[3] A. Andriyashin. "Financial Applications of Classification and Regression Trees A Master Thesis Presented." In: (Feb. 2019).

[4] *Azure for banking and capital markets*. [Online; accessed 13-January-2018]. URL: https://azure.microsoft.com/en-us/industries/financial/banking/usecases/.

[5] *Azure for Financial Services*. [Online; accessed 13-January-2018]. URL: https://azure.microsoft.com/en-us/industries/financial/.

[6] H. E.M. E. Badr HSSINA Abdelkarim MERBOUHA. "A comparative study of decision tree ID3 and C4.5." In: *International Journal of Advanced Computer Science and Applications, Special Issue on Advances in Vehicular Ad Hoc Networking and Applications* 241 (). URL: https://saiconference.com/Downloads/SpecialIssueNo10/Paper_3-A_comparative_study_of_decision_tree_ID3_and_C4.5.pdf.

[7] *Breaking the 80/20 rule: How data catalogs transform data scientists' productivity*. [Online; accessed 20-August-2019]. URL: https://www.ibm.com/cloud/blog/ibm-data-catalog-data-scientists-productivity.

[8] F. J.H.O.R.A. S. Breiman L. *Classification and Regression Trees*. 1984. ISBN: 9780534980542.

[9] Carnegie Mellon University. *Lecture 10: Regression Trees*. [Online; accessed 15-February-2019]. URL: http://www.stat.cmu.edu/~cshalizi/350-2006/lecture-10.pdf.

[10] CFI. *Types of financial models*. [Online; accessed 12-February-2019]. URL: https://corporatefinanceinstitute.com/resources/knowledge/modeling/types-of-financial-models/.

[11] CFI. *What is a DCF model?* [Online; accessed 12-February-2019]. URL: https://corporatefinanceinstitute.com/resources/knowledge/modeling/dcf-model-training-free-guide/.

[12] J. Chen. *The Dow 30.* [Online; accessed 11-February-2019]. URL: https://www.investopedia.com/terms/d/dow-30.asp.

[13] J. Chen. *Technical Analysis.* [Online; accessed 15-January-2019]. 2018. URL: https://www.investopedia.com/terms/t/technicalanalysis.asp.

[14] J. Chen. *Stock.* [Online; accessed 04-February-2019]. 2019. URL: https://www.investopedia.com/terms/s/stock.asp.

[15] Chris DeBrusk Emily Du. *Why Wall Street needs to make investing i Machine Learning a higher priority.* [Online; accessed 14-February-2019]. 2018. URL: https://www.oliverwyman.com/content/dam/oliver-wyman/v2/publications/2018/may/Machine-Learning-on-Wall-Street.pdf.

[16] D.J.Power. *A Brief History of Decision Support Systems.* URL: http://dssresources.com/history/dsshistory.html.

[17] N. R. Draper and H. Smith. "Checking the Straight Line Fit." In: *Applied Regression Analysis.* John Wiley Sons, Ltd, 2014. Chap. 2, pp. 47–77. ISBN: 9781118625590. DOI: 10.1002/9781118625590.ch2. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118625590.ch2.

[18] J. Emspak. *Alpha and beta for beginners.* [Online; accessed 29-January-2019]. 2018. URL: https://www.investopedia.com/articles/investing/092115/alpha-and-beta-beginners.asp.

[19] *ERAAM.* [Online; accessed 05-August-2019]. URL: https://www.eraam.com/en/eraam-long-short-equity.

[20] *ERAAM.* [Online; accessed 05-August-2019]. URL: https://www.eraam.com/en/funds.

[21] K. Faulkenberry. *Alpha and Beta: How Do They Relate to Investment Risk?* [Online; accessed 11-February-2019]. URL: http://www.arborinvestmentplanner.com/alpha-and-beta-investment-risk/.

[22] Y. Finance. *Tesla, Inc (TSLA) Financial Statements.* [Online; accessed 07-February-2019]. URL: https://finance.yahoo.com/quote/TSLA/financials?p=TSLA.

[23] FinTech News. *6 Best programming languages for FinTech.* [Online; accessed 15-February-2019]. 2018. URL: https://www.fintechnews.org/6-best-programming-languages-for-fintech/.

[24] W. E. Forum. *The computer revolution: how it's changed our world over 60 years.* [Online; accessed 09-February-2019]. URL: https://www.weforum.org/agenda/2016/06/the-computer-revolution-how-its-changed-our-world-over-60-years.

[25] W. E. Forum. *The Future of Financial Services, How disruptive innovations are reshaping the way financial services are structured, provisioned and consumed.* [Online; accessed 20-January-2019]. 2015. URL: http://www3.weforum.org/docs/WEF_The_future__of_financial_services.pdf.

[26] W. E. Forum. *The New Physics of Financial Services, Understanding how artificial intelligence is transforming the financial ecosystem.* [Online; accessed 20-January-2019]. 2018. URL: http://www3.weforum.org/docs/WEF_New_Physics_of_Financial_Services.pdf.

[27] Frederico Caeiro. *Regressão Linear Simples - Estatística (Slides da cadeira de Probabilidades e Estatística).* [Online; accessed 5-January-2019]. 2016.

[28] R. Furhmann. *Equity valuation: The comparables approach.* [Online; accessed 04-February-2019]. 2018. URL: https://www.investopedia.com/articles/investing/080913/equity-valuation-comparables-approach.asp.

[29] Gareth James Daniela Witten Trevor Hastie Robert Tibshirani. *An Introduction to Statistical Learning.* [Online; accessed 18-August-2019]. URL: http://faculty.marshall.usc.edu/gareth-james/ISL/ISLR%20Seventh%20Printing.pdf.

[30] *GitHub: Bloomberg/comdb2.* [Online; accessed 13-January-2018]. URL: https://github.com/bloomberg/comdb2.

[31] J. Glen. *Fundamental Analysis vs. Technical Analysis.* [Online; accessed 08-February-2019]. URL: http://www.businessdictionary.com/article/1104/fundamental-analysis-vs-technical-analysis-d1412/.

[32] J. Glen. *Fundamental Analysis vs. Technical Analysis.* [Online; accessed 15-January-2019]. 2018. URL: https://www.investopedia.com/ask/answers/difference-between-fundamental-and-technical-analysis/.

[33] *GoBusiness.* [Online; accessed 13-January-2018]. URL: https://gobusiness-seguros.pt/pt/.

[34] Google Cloud. *Preparing Data.* [Online; accessed 21-February-2019]. 2019. URL: https://cloud.google.com/ml-engine/docs/tensorflow/data-prep.

[35] J. Greenblatt. *The Little Book That Still Beats the Market.* John Wiley & Sons, 2010. ISBN: 0471733067. URL: https://www.amazon.com/Little-Book-Still-Beats-Market/dp/0470624159?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0470624159.

[36] Y. Hilpisch. *Python for Finance: Analyze Big Financial Data.* O'Reilly Media, 2014. ISBN: 9781491945285. URL: https://www.amazon.com/Python-Finance-Analyze-Financial-Data/dp/1491945281?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=1491945281.

[37] *How AI Will Invade Every Corner of Wall Street*. [Online; accessed 14-January-2018]. URL: https://www.bloomberg.com/markets/fixed-income.

[38] E. Ikonomovska, J. Gama, R. Sebastião, and D. Gjorgjevikj. "Regression Trees from Data Streams with Drift Detection." In: Oct. 2009, pp. 121–135. DOI: 10.1007/978-3-642-04747-3_12.

[39] C. F. Institute. *Three Financial Statements*. [Online; accessed 07-February-2019]. URL: https://corporatefinanceinstitute.com/resources/knowledge/accounting/three-financial-statements/.

[40] *Investopedia*. URL: https://www.investopedia.com/.

[41] Investopedia-Chris Seabury. *How Interest Rates Affect the U.S. Markets*. [Online; accessed 15-February-2019]. 2018. URL: https://www.investopedia.com/articles/stocks/09/how-interest-rates-affect-markets.asp.

[42] Kaggle. *Underfitting and Overfitting*. [Online; accessed 21-February-2019]. 2018. URL: https://www.kaggle.com/dansbecker/underfitting-and-overfitting.

[43] J. Kennon. *Past Performance Is No Guarantee of Future Results*. [Online; accessed 12-February-2019]. URL: https://www.thebalance.com/past-performance-is-no-guarantee-of-future-results-357862.

[44] W. Kenton. *Beta*. [Online; accessed 11-February-2019]. URL: https://www.investopedia.com/terms/b/beta.asp.

[45] W. Kenton. *SP 500 Index (Standard Poor's 500 Index)*. [Online; accessed 11-February-2019]. URL: https://www.investopedia.com/terms/s/sp500.asp.

[46] W. Kenton. *Asset*. [Online; accessed 07-February-2019]. 2018. URL: https://www.investopedia.com/terms/a/asset.asp.

[47] W. Kenton. *Capital Markets*. 2018. URL: https://www.investopedia.com/terms/c/capitalmarkets.asp (visited on 01/12/2019).

[48] W. Kenton. *Time Series*. [Online; accessed 29-January-2019]. 2018. URL: https://www.investopedia.com/terms/t/timeseries.asp.

[49] S. Kotsiantis, I Zaharakis, V Tampakas, and P Pintelas. "On Constructing a Financial Decision Support System." In: (Jan. 2019).

[50] S. Liu, A. Duffy, R. Whitfield, and I. M. Boyle. "Integration of decision support systems to improve decision support performance." In: *Knowl. Inf. Syst.* 22 (Mar. 2010), pp. 261–286. DOI: 10.1007/s10115-009-0192-4.

[51] R. H. U. of London. *Computational Finance: The Course*. [Online; accessed 09-February-2019]. URL: https://www.royalholloway.ac.uk/studying-here/postgraduate/computer-science/computational-finance/.

[52] P. Lynch. *One Up On Wall Street: How To Use What You Already Know To Make Money In The Market*. Simon & Schuster, 2000. ISBN: 0743200403. URL: https://www.amazon.com/One-Up-Wall-Street-Already/dp/0743200403?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0743200403.

[53] B. B. Mandelbrot. *The Mis Behaviour Of Markets A Fractal Views Of Risk Ruin And Reward*. Viva Books Private Limited, 2009. ISBN: 1846682622. URL: https://www.amazon.com/Behaviour-Markets-Fractal-Views-Reward/dp/B0072N9NSS?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=B0072N9NSS.

[54] T. M. Mitchell. *Machine Learning*. McGraw-Hill Education, 1997. ISBN: 0070428077. URL: https://www.amazon.com/Machine-Learning-Tom-M-Mitchell/dp/0070428077?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0070428077.

[55] D. Mondal, G. Maji, T. Goto, N. C. Debnath, and S. Sen. "A Data Warehouse Based Modelling Technique for Stock Market Analysis." In: *International Journal of Engineering Technology* 7 (July 2018), p. 165. DOI: 10.14419/ijet.v7i3.13.16344.

[56] NASDAQ. *FINTECH TRENDS 2018: How Technology is rewriting the capital markets*. [Online; accessed 10-February-2019]. URL: https://business.nasdaq.com/marketinsite/2018/MT/FinTech-Trends-2018-How-Technology-is-Rewriting-the-Capital-Markets.html?source=RSSfeed.

[57] J. Nguyen. *How to choose the best stock valuation method*. [Online; accessed 04-February-2019]. 2018. URL: https://www.investopedia.com/articles/fundamental-analysis/11/choosing-valuation-methods.asp.

[58] E. Norris. *What is the difference between fundamental and technical analysis?* [Online; accessed 15-January-2019]. 2018. URL: https://www.investopedia.com/ask/answers/difference-between-fundamental-and-technical-analysis/.

[59] U. of Oxford. *MSc in Mathematical and Computational Finance*. [Online; accessed 09-February-2019]. URL: https://www.ox.ac.uk/admissions/graduate/courses/msc-mathematical-and-computational-finance?wssl=1.

[60] C. Pareto. *A Deeper look at Alpha*. [Online; accessed 29-January-2019]. 2009. URL: https://www.investopedia.com/articles/financial-theory/08/deeper-look-at-alpha.asp.

[61] W. S. Prep. *Introduction to the DCF Model*. [Online; accessed 12-February-2019]. URL: https://www.wallstreetprep.com/knowledge/dcf-model-training-6-steps-building-dcf-model-excel/.

[62] *Quandl*. [Online; accessed 19-June-2019]. URL: https://https://www.quandl.com/.

[63] *Quantopian*. [Online; accessed 19-June-2019]. URL: https://www.quantopian.com/home.

[64] Ray Dalio. *How the Economic Machine Works*. [Online; accessed 14-February-2019]. 2018. URL: https://www.economicprinciples.org/wp-content/uploads/ray_dalio__how_the_economic_machine_works__leveragings_and_deleveragings.pdf.

[65] *Regression Analysis: How Do I Interpret R-squared and Assess the Goodness-of-Fit?* [Online; accessed 11-September-2019]. URL: https://blog.minitab.com/blog/adventures-in-statistics-2/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit.

[66] *Renaissance Technologies*. [Online; accessed 20-November-2018]. URL: https://www.rentec.com/Home.action?index=true.

[67] *Renaissance Technologies LLC - Latest 13F Holdings, Performance, AUM*. [Online; accessed 20-November-2018]. URL: https://fintel.io/i/renaissance-technologies-llc.

[68] *Renaissance Technologies Rebounds Across Hedge Funds in March*. [Online; accessed 20-November-2018]. URL: https://www.bloomberg.com/news/articles/2018-04-18/renaissance-technologies-rebounds-across-hedge-funds-in-march.

[69] W. S. Sarle. “Subject: What are the population, sample, training set, design set, validation set, and test set?” In: *Neural Network FAQ, part 1 of 7: Introduction*. 2002. URL: ftp://ftp.sas.com/pub/neural/FAQ.html#A_data.

[70] *Scikit Learn - R^2Score*. [Online; accessed 17-August-2019]. URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html#sklearn.metrics.r2_score.

[71] *Scikit-Learn Lasso Regularization*. URL: https://scikit-learn.org/stable/modules/linear_model.html#lasso.

[72] A. Scotti, M. Hannum, M. Ponomarenko, D. Hogea, A. Sikarwar, M. Khullar, A. Zaimi, J. Leddy, F. Angius, R. Zhang, and L. Deng. “Comdb2: Bloomberg’s Highly Available Relational Database System.” In: *PVLDB* 9.13 (2016), pp. 1377–1388. DOI: 10.14778/3007263.3007275. URL: http://www.vldb.org/pvldb/vol9/p1377-scotti.pdf.

[73] U. Securities and E. Commission. *Public Companies*. [Online; accessed 07-February-2019]. URL: https://www.investor.gov/introduction-investing/basics/how-market-works/public-companies.

[74] *Stopping Rules (CART algorithms)*. URL: https://www.ibm.com/support/knowledgecenter/en/SSLVMB_22.0.0/com.ibm.spss.statistics.algorithms/alg_tree-cart_stopping.htm.

[75] *This is how much a Bloomberg terminal costs*. [Online; accessed 20-November-2018]. URL: https://qz.com/84961/this-is-how-much-a-bloomberg-terminal-costs/.

[76] Wikipedia contributors. *Machine Learning*. https://en.wikipedia.org/wiki/Machine_learning. [Online; accessed 05-February-2019].

[77] Wikipedia contributors. *Economic model — Wikipedia, The Free Encyclopedia*. [Online; accessed 13-February-2019]. 2018. URL: https://en.wikipedia.org/w/index.php?title=Economic_model&oldid=863123474.

[78] Wikipedia contributors. *Valuation using discounted cash flows — Wikipedia, The Free Encyclopedia*. [Online; accessed 12-February-2019]. 2018. URL: https://en.wikipedia.org/w/index.php?title=Valuation_using_discounted_cash_flows&oldid=842758343.

[79] Wikipedia contributors. *Capital asset pricing model — Wikipedia, The Free Encyclopedia*. [Online; accessed 20-February-2019]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Capital_asset_pricing_model&oldid=883053788.

[80] Wikipedia contributors. *Coefficient of determination — Wikipedia, The Free Encyclopedia*. [Online; accessed 3-August-2019]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Coefficient_of_determination&oldid=908029066.

[81] Wikipedia contributors. *Cross-industry standard process for data mining — Wikipedia, The Free Encyclopedia*. [Online; accessed 21-February-2019]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Cross-industry_standard_process_for_data_mining&oldid=877336439.

[82] Wikipedia contributors. *Decision Support Systems*. https://en.wikipedia.org/wiki/Decision_support_system. [Online; accessed 04-February-2019]. 2019.

[83] Wikipedia contributors. *Decision tree learning — Wikipedia, The Free Encyclopedia*. [Online; accessed 21-February-2019]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Decision_tree_learning&oldid=881171846.

[84] Wikipedia contributors. *Linear regression — Wikipedia, The Free Encyclopedia*. [Online; accessed 20-February-2019]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Linear_regression&oldid=883726170.

[85] Wikipedia contributors. *New York Stock Exchange — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=New_York_Stock_Exchange&oldid=877477195. [Online; accessed 15-January-2019]. 2019.

[86] Wikipedia contributors. *Normalization (statistics) — Wikipedia, The Free Encyclopedia*. [Online; accessed 17-August-2019]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Normalization_(statistics)&oldid=907866084.

[87] Wikipedia contributors. *Training, validation, and test sets — Wikipedia, The Free Encyclopedia*. [Online; accessed 11-August-2019]. 2019. URL: https://en.wikipedia.org/w/index.php?title=Training,_validation,_and_test_sets&oldid=905841557.

[88] Wikiversity. *Quantitative Finance*. [Online; accessed 09-February-2019]. URL: https://en.wikiversity.org/wiki/Quantitative_finance.

[89] Yale University. *Multiple Linear Regression*. [Online; accessed 5-January-2019]. 1998. URL: http://www.stat.yale.edu/Courses/1997-98/101/linmult.htm.

[90] C. C. Yong and S. M. Taib. "Designing a decision support system model for stock investment strategy." In: *Proceedings of the World Congress on Engineering and Computer Science 2009 I*. 2009.

[91] J. Young. *Market Index*. [Online; accessed 11-February-2019]. URL: https://www.investopedia.com/terms/m/marketindex.asp.

[92] K. Zucchi. *Financial markets: Capital vs. Money Markets*. [Online; accessed 04-February-2019]. 2018. URL: https://www.investopedia.com/articles/investing/052313/financial-markets-capital-vs-money-markets.asp.

Appendix A

Appendix

The following piece of code refers to the process of reading data from SQLite3, cleaning, normalizing and building the decision tree:

Listing A.1: Initializing and fitting a Decision Tree Regressor

```
import sklearn as sk
import pandas as pd
import numpy as np
import matplotlib
import scipy
from sklearn import tree
import csv
from decimal import *

import sqlite3
from sqlite3 import Error
pd.options.mode.chained_assignment = None

features = ['cash','cost_of_revenue', 'current_assets', 'current_liabilities', 'free_cash_flow_yield', 'gross_profitability', 'investment_to_asset', 'sales', 'share_issuance', 'total_debt', 'total_equity']

def regression_tree(conn):

    #Ler todas as features
    cur = conn.cursor()
    cur.execute("SELECT␣distinct␣type␣FROM␣fundamentals")

    rows = cur.fetchall()
    features = []
    for row in rows:
        if row[0]!='Sector':
            features.append(row[0])

    cur.execute("SELECT␣distinct␣date,␣sid,␣stock,␣val␣FROM␣fundamentals␣where␣type='Sector'")

    rows = cur.fetchall()
    date = []
    sid = []
    stock = []
    sector = []
    for row in rows:
        date.append(row[0])
        sid.append(row[1])
        stock.append(row[2])
        sector.append(row[3])

    data = {'date' : date,
    'sid': sid,
    'stock': stock,
    'Sector': sector}

    data_ex = pd.DataFrame(data)

    for feature in features:
        val = []
        for row in data_ex.iterrows():
            cur.execute("SELECT␣val␣FROM␣fundamentals␣where␣type=?␣AND␣date=?␣AND␣stock=?",(feature,row[1]["date"], row[1]["stock"]))
            rows = cur.fetchone()
            val.append(rows[0])
        data_ex[feature] = val

    cur.execute('SELECT␣distinct␣stock␣from␣fundamentals')
    stocks_fundamentals = cur.fetchall()
    cur.execute('SELECT␣distinct␣stock␣from␣prices')
    stocks_prices = cur.fetchall()
    stocks = intersection(stocks_fundamentals,stocks_prices)

    #Replace Str 'NaN' for numpy.nan
    data_ex.replace("NaN", np.nan, inplace=True)

    #Create row "Prices" with default value 0
    data_ex['price'] = 0.0
    #Add "Price" values
    for stock in stocks:
        for row in range(0,len(data_ex['price'])):
            stock_ = data_ex['stock'].values[row]
            if stock[0] == stock_:
                date =  data_ex['date'].values[row]
                cur.execute("SELECT␣price␣FROM␣prices␣where␣date=?␣AND␣stock=?", (date,stock_))
                price = cur.fetchone()[0]
                if price!="NaN":
                    data_ex['price'].values[row] = price

    #Escolher sector(311- Tech, 310- Industrials)
    data_training = data_ex[(data_ex.Sector == 310.0)]
    data_testing = data_ex[(data_ex.Sector == 310.0)]
    data_training = data_training[(data_training.date > "2006-03-30") & (data_training.date < "2018-09-30")]
    data_testing = data_testing[data_testing.date == "2018-12-30"]

    #Deal with NaN values in Training Set
    for column in data_training[['cash','cost_of_revenue', 'current_assets', 'current_liabilities','free_cash_flow_yield', 'gross_profitability', 'investment_to_asset', 'sales', 'share_issuance', 'total_debt', 'total_equity', 'price']]:
        data_training[column].fillna(data_training[column].mean(),inplace=True)

    #Deal with NaN values in Testing Set
    for column in data_testing[['cash','cost_of_revenue', 'current_assets', 'current_liabilities','free_cash_flow_yield', 'gross_profitability', 'investment_to_asset', 'sales', 'share_issuance', 'total_debt', 'total_equity', 'price']]:
        data_testing[column].fillna(data_testing[column].mean(),inplace=True)

    for stock in data_testing['stock'].unique():
        if stock not in data_training['stock'].unique():
            toDelete = data_testing[data_testing['stock'] == stock].index.values
            data_testing.drop(toDelete, inplace=True)

    x_training = data_training[['stock','cash','cost_of_revenue', 'current_assets', 'current_liabilities','free_cash_flow_yield', 'gross_profitability', 'investment_to_asset', 'sales', 'share_issuance', 'total_debt', 'total_equity']]
    y_training = data_training[['stock','price']]

    #Predict
    x_testing = data_testing[['stock','cash','cost_of_revenue', 'current_assets', 'current_liabilities','free_cash_flow_yield', 'gross_profitability', 'investment_to_asset', 'sales', 'share_issuance', 'total_debt', 'total_equity']]
    y_testing = data_testing[['stock','price']]

    print("Dimension␣of␣the␣Training␣set:␣␣␣", np.shape(x_training))
    print(x_training.describe(), "\n")

    #Min-Max Normalization
    from sklearn import preprocessing

    for stock_name in x_training['stock'].unique():
        x_training_values = x_training[x_training['stock'] == stock_name]
        x_min_max_scaler = preprocessing.MinMaxScaler(copy=True, feature_range=(0, 1)).fit(x_training_values.drop('stock', axis=1).values)
        x_scaled_training = x_min_max_scaler.transform(x_training_values.drop('stock', axis=1).values)

        counter = 0
        for i_training in x_training[x_training['stock'] == stock_name].index.values:
            counter_column_index = 0
            for column in x_training_values.drop('stock', axis=1).columns:
                x_training.at[i_training, column] = x_scaled_training[counter][counter_column_index]
                counter_column_index += 1
            counter += 1

        if stock_name in x_testing['stock'].unique():
            x_testing_values = x_testing[x_testing['stock'] == stock_name]
            x_scaled_testing = x_min_max_scaler.transform(x_testing_values.drop('stock', axis=1).values)
            counter = 0
            for i_testing in x_testing[x_testing['stock'] == stock_name].index.values:
                counter_column_index = 0
                for column in x_testing_values.drop('stock', axis=1).columns:
                    if x_scaled_testing[counter][counter_column_index] > 1:
                        x_testing.at[i_testing, column] = 1
                    elif x_scaled_testing[counter][counter_column_index] < 0:
                        x_testing.at[i_testing, column] = 0
                    else:
                        x_testing.at[i_testing, column] = x_scaled_testing[counter][counter_column_index]
                    counter_column_index += 1
                counter += 1

    for stock_name in y_training['stock'].unique():
        y_training_values = y_training[y_training['stock'] == stock_name]
        y_min_max_scaler = preprocessing.MinMaxScaler(copy=True, feature_range=(0, 1)).fit(y_training_values.drop('stock', axis=1).values)
        y_scaled_training = y_min_max_scaler.transform(y_training_values.drop('stock', axis=1).values)

        counter = 0
        for i_training in y_training[y_training['stock'] == stock_name].index.values:
            y_training.at[i_training, 'price'] = y_scaled_training[counter][0]
            counter += 1

        if stock_name in y_testing['stock'].unique():
            y_testing_values = y_testing[y_testing['stock'] == stock_name]
            y_scaled_testing = y_min_max_scaler.transform(y_testing_values.drop('stock', axis=1).values)
            counter = 0
            for i_testing in y_testing[y_testing['stock'] == stock_name].index.values:
                if y_scaled_testing[counter][0] > 1:
                    y_testing.at[i_testing, 'price'] = 1
                elif y_scaled_testing[counter][0] < 0:
                    y_testing.at[i_testing, 'price'] = 0
                else:
                    y_testing.at[i_testing, 'price'] = y_scaled_testing[counter][0]
                counter += 1

    #Build Tree
    #clf_industrial = tree.DecisionTreeRegressor()
    clf_industrial = tree.DecisionTreeRegressor(max_depth = 20, min_samples_split= 10,min_samples_leaf=11, max_leaf_nodes= 25)
    #clf_industrial = tree.DecisionTreeRegressor(max_depth = 20, min_samples_split=10)
    #clf_industrial = tree.DecisionTreeRegressor(, min_samples_split="", min_samples_leaf="", max_leaf_nodes="")
    clf_industrial = clf_industrial.fit(x_training[['cash','cost_of_revenue', 'current_assets', 'current_liabilities','free_cash_flow_yield', 'gross_profitability', 'investment_to_asset', 'sales', 'share_issuance', 'total_debt', 'total_equity']],
    y_training[['price']])

    x_testing.drop('stock', axis=1, inplace=True)
    x_training.drop('stock', axis=1, inplace=True)
    y_training.drop('stock', axis=1, inplace=True)
    y_testing.drop('stock', axis=1, inplace=True)
    print("Dimension␣of␣X␣Training␣set␣After␣Norm:␣␣␣", np.shape(x_training))
    print(x_training.describe(), "\n")

    import matplotlib.pyplot as plt
    #pd.options.display.mpl_style = 'default'
    x_training.boxplot()

    print("Dimension␣of␣X␣Testing␣set␣After␣Norm:␣␣␣", np.shape(x_testing))
    print(x_testing.describe(), "\n")

    print("Dimension␣of␣Y␣Training␣set␣After␣Norm:␣␣␣", np.shape(y_training))
    print(y_training.describe(), "\n")

    print("Dimension␣of␣Y␣Testing␣set␣After␣Norm:␣␣␣", np.shape(y_testing))
    print(y_testing.describe(), "\n")



    print("------------------PREDICTIONS-------------------------------")
    y_pred = clf_industrial.predict(x_testing)
    print(y_pred)

    from sklearn.metrics import r2_score
    r2 = r2_score(y_testing, y_pred)
    print(r2)

    from sklearn.metrics import mean_squared_error
    mse = mean_squared_error(y_testing, y_pred)
    print(mse)

    from sklearn.tree import export_graphviz
    #Export as dot file

    export_graphviz(clf_industrial, out_file='test.dot',
                feature_names = features,
                class_names = ["Price"], label = "all",
                rounded = True, proportion = False,
                precision = 3, filled = True)

    #Convert to png
    from subprocess import call
    call(['dot', '-Tpng', 'test.dot', '-o', 'test.png'])


    #Display in python
    import matplotlib.pyplot as plt
    x = features
    importances = clf_industrial.feature_importances_

    print("IMPORTANCES␣␣", importances)


def adjust_min_max(y):
    #for column in y.columns:
    y[y > 1 ] = 1
    y[y < 0 ] = 0
    return y

def test_adjust_min_max():
    test = pd.DataFrame({'stock': ['a','b','c','d', 'e', 'f'], 'col1': [2.4,1.5,-2.2,-0.2, 0.5, 0.3],'col2': [-2.4,5.5,1.2,-0.2, 0.9, -0.3]})
    print(test['stock'].unique())
    print(test[test['stock'] == 'a'])
    #print(test)
```

```python
    test_a = test[['col1', 'col2']]
    #print("Negativos ",test_a[test_a<0.0].count())
    #print("Positivos ",test_a[test_a>1.0].count())


    test[['col1', 'col2']] = adjust_min_max(test[['col1', 'col2']])
    #print(test)
    test_b = test[['col1', 'col2']]
    #print("Negativos ",test_b[test_b<0.0].count())
    #print("Positivos ",test_b[test_b>1.0].count())

def intersection(lst1, lst2):
    lst3 = [value for value in lst1 if value in lst2]
    return lst3


def create_connection(db_file):
     try:
        conn = sqlite3.connect(db_file)
        return conn
     except Error as e:
        print(e)

     return None


def main():
    database = "./Dados/database.db"
    # create a database connection
    conn = create_connection(database)
    with conn:
        regression_tree(conn)
    #test_adjust_min_max()


if __name__ == '__main__':
    main()
```