



Enhancing a Pairs Trading strategy using Financial Indicators with an application of Machine Learning

João Nuno Costa dos Santos

Thesis to obtain the Master of Science Degree in
Electrical and Computer Engineering

Supervisor: Prof. Nuno Cavaco Gomes Horta

Examination Committee

Chairperson: Prof. Teresa Maria Sá Ferreira Vazão Vasques
Supervisor: Prof. Nuno Cavaco Gomes Horta
Member of the Committee: Prof. Helena Isabel Aidos Lopes Tomás

November 2021

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

Acknowledgments

I would first like to thank my thesis supervisor, Prof. Nuno Horta, for his guidance during this work. His feedback was fundamental in delineating the aspects this thesis should focus on.

Secondly, I would like to thank all my Microsoft colleges that gave me the possibility to take some time to focus on this work during work-hours.

Lastly, I would also like to thank my friends and my family, for their constant support during the course of this work, even when that implied spending less time with them.

Introduzir agradecimento ao IT

Resumo

Pairs trading é uma estratégia de investimento popular, utilizada mundialmente. Tem a particularidade de não se focar no preço individual de um ativo financeiro, mas antes no preço relativo de dois títulos. Desta forma, esta estratégia torna-se viável independentemente da tendência do mercado. Através de uma escolha meticulosa destes pares de ativos financeiros, os investidores procuram oportunidades espontâneas na divergência de preços, comprando o título subvalorizado e vendendo aquele que estiver sobrevalorizado. O retorno financeiro advém da eventual convergência da cotação par.

Nesta tese, é proposto um melhoramento da estratégia de investimento *Pairs Trading* através da utilização de *Long Short-Term Memory Networks* para previsão da evolução de um determinado título financeiro com a ajuda de alguns indicadores financeiros. Este modelo terá a função de adiantar ou atrasar possíveis decisões com base nas suas previsões. Duas novas funções de decisão, serão adicionadas com o objetivo de tornar o modelo de investimento tradicional menos sensível a flutuações anormais de mercado.

O modelo proposto, durante o período de testes, obtem um retorno 54% superior ao modelo tradicional. No entanto, este melhoramento no modelo não se deve ao modelo de previsão, mas sim às funções de decisão que não só reduzem potenciais perdas, como investem em oportunidades que o modelo tradicional não descobre.

Palavras Chave

Pairs Trading; Mercado Bolsista; Redes Neurais; Indicadores Financeiros; Aprendizagem Automática

Abstract

Trading is a popular market-neutral investment strategy used by investors worldwide. This strategy focuses on relative price, profiting both from increasing and decreasing prices, thus avoiding high market volatility. By carefully selecting the pairs and analysing their behaviour, the investors pursue market opportunities to sell a relatively overvalued security and simultaneously buying an undervalued one. These opportunities usually arise from a spontaneous divergence, and a profit is made from the eventual pair's price convergence. Due to the evolution of computing power and higher accessibility of data, over the last decades, more and more investigation has been made into new investment approaches.

In this work, it's proposed an enhanced model of Pairs Trading through the use of Long Short-Term Memory Long Short-Term Memory (LSTM) Networks to forecast the behaviour of stocks based on its financial indicators. These forecasts aim to either entering earlier or later (than the reference that is the simple threshold-based model) a certain opportunity to increase its profit. Also, two other decision functions were added to make the overall enhanced model less vulnerable to abnormal market fluctuations.

During the test period, the proposed model, had a 54% increase in profit, when compared with the regular threshold-based model. However, this increase in performance is not due to the forecasting itself, but rather due to the decision functions that not only mitigate potential losses but also invest in new opportunities that the traditional model doesn't.

Keywords

Pairs Trading; Stock Market; Neural Networks; Financial Indicators; Machine Learning

Contents

1	Introduction	1
1.1	Topic Overview	2
1.2	Objectives	4
1.2.1	Improve Forecasting with Financial Indicators	5
1.2.2	Increase Accuracy	5
1.3	Report Outline	6
2	Key Concepts and Background	7
2.1	Financial Concepts	8
2.1.1	Financial Market	8
2.1.2	Long and Short Market Position	8
2.1.3	Simple Moving Average	10
2.1.4	Stochastic Oscillator	11
2.1.5	Relative Strength Index	12
2.1.6	Risk Management	13
2.2	Pairs Selection	13
2.2.1	SSD measure	14
2.2.2	Cointegration	15
2.2.3	Correlation	16
2.2.4	Conclusion	16
2.3	Stock Price Prediction	17
2.3.1	Backpropagation Algorithm	17
2.3.2	RNN and LSTM	18
2.3.3	Conclusion	19
2.4	Trading Strategies	20
2.4.1	Day Trading	21
2.4.2	Swing Trading	21
2.4.3	Scalping	21

2.4.4	Position Trading	21
2.4.5	Conclusion	21
3	System Architecture	23
3.1	Pairs Selection	24
3.1.1	Range Selection	25
3.1.2	Filter selection	26
3.1.3	Historical Profit	26
3.2	Time Series Forecasting Model	27
3.2.1	Training Model	27
3.2.2	Feature Preparation	27
3.2.3	Five day forecasting	28
3.3	Trading and Portfolio Management Models	28
3.3.1	Problem Statement	28
3.3.2	Trading Model	28
3.3.3	Stop Loss-Function	29
3.4	Project's Structure Overview	30
4	Implementation	31
4.1	Pairs Selection	32
4.1.1	Filter Sensitivity	32
4.1.2	Historical Profit	33
4.2	Forecasting Model	33
4.2.1	Model Creation	33
4.2.2	Model Accuracy Evaluation	34
4.2.3	Feature Preparation	35
4.2.3.A	Data Collection	35
4.2.3.B	Feature Calculation	35
4.2.3.C	Normalization	36
4.2.3.D	Feature Combination	36
4.2.4	Single Model Approach	37
4.2.5	Multiple Model Approach	38
4.3	Trading Model	39
4.3.1	Data Structure	40
4.3.2	Threshold Based Model	41
4.3.2.A	Market Movement Calculation	41
4.3.2.B	Profit Calculation	42

4.3.2.C	Visual Presentation in Power BI	43
4.3.3	Enhanced Model	44
4.3.3.A	Minimum and Maximum Spread Days	44
4.3.3.B	New Market Position	45
4.3.3.C	Profit Calculation	46
4.3.3.D	Visual Presentation in Power BI	47
4.4	Power BI Platform	47
4.4.1	Model Behaviour Dashboard	47
4.4.2	Transactions Dashboard	49
4.5	Computational Cost	50
4.5.1	Pairs Selection	51
4.5.2	Forecasting Model	51
5	Validation	53
5.1	Pairs Selection	55
5.2	Model Behaviour	56
5.2.1	Portfolio Decline Days	56
5.2.2	Overall Profitability	56
5.2.2.A	New Opportunities	57
5.2.2.B	Stop Loss Function Activation	59
5.2.2.C	Different Entry/Exit Days	60
5.2.2.D	Overall Performance	61
6	Conclusions and Future Work	63
6.1	Conclusions	65
6.2	Future Work	65
A	List of Stocks in NASDAQ-100	68
B	User Guide	71
B.1	Pairs Selection	72
B.2	Forecasting Model	73
B.3	Trading Model	73

List of Figures

1.1	Price series from Pfizer Inc. (PFE) and Johnson & Johnson (JNJ)	2
1.2	Example of Pairs Trading applied to a pair of securities	3
1.3	Flowchart of the proposed work	5
2.1	Example of a profitable Long Trade	9
2.2	Example of a profitable Short Trade	10
2.3	Price evolution of Pfizer and its moving averages	11
2.4	Price evolution of Pfizer and its K and D lines	12
2.5	Price evolution of Pfizer and its Relative Strength Index	13
2.6	Two Examples of pairs with different SSD measures 0.07 (left) and 0.968 (right)	14
2.7	Cointegration method for pairs selection	15
2.8	Two Examples of pairs with different p-values	15
2.9	Two Examples of pairs with different correlation values	16
2.10	Illustration of the output of the Stock Price Prediction model	17
2.11	Structure of BP neural network	18
2.12	Recurrent Neural Network Structure	18
2.13	Long Short-Term Memory cell	19
3.1	Architecture of the project	24
3.2	Pairs selection process	25
3.3	NASDAQ-100 Index Tracking Stock	26
3.4	Division of Data into Training and Testing	27
3.5	Example of a high uncertainty period	28
3.6	Trading model decision flowchart	29
3.7	Behaviour comparison between the Spread and the Normalized Spread	29
4.1	Code used to train the model	34
4.2	Code used to run the model	34

4.3	Feature Preparation Process	35
4.4	Creation of column 'Area' in Power Query Editor	41
4.5	Creation of column 'Market Movement' using DAX	42
4.6	Creation of columns 'Price1' and 'Price2' using DAX	42
4.7	Creation of column 'Shares_Stock2' using DAX	43
4.8	Creation of column 'Profit' using DAX	43
4.9	Application of the Threshold based model in Power BI	44
4.10	Creation of column 'minSpread_ID' using DAX	45
4.11	Creation of column 'New Market Position' using DAX	46
4.12	Creation of column 'New Market Position Hold' using DAX	46
4.13	Application of the Enhanced Model in Power BI	47
4.14	Power BI card's creation process	48
4.15	Power BI graphs fields selection	49
4.16	Power BI transactions dashboard	50
5.1	New market position opened in ADBE-MSFT pair	57
5.2	New market position opened in ADP-INTU pair	58
5.3	New market position opened in IDXX-MCHP pair	59
5.4	Stop Loss function being activated in LRCX-MAR pair	59
5.5	Different Entry and Closing days for the same opportunity in LRCX-MAR pair	60
B.1	Folder containing the code and Excel files that support the Pairs Selection process	72
B.2	Folder containing the code and Excel files that support the Forecasting Model creation process	73

List of Tables

1.1	Application of the Threshold-based trading model	4
2.1	Literature exploring the application of Machine Learning in Stock Market Prediction	20
3.1	Example of Indexes that could be used as a range selection	25
4.1	Variation of the number of Pairs with different Correlation and Cointegration Values	32
4.2	Variation of the number of Pairs with different SSD measures	32
4.3	Simulation Profit in USD of the Threshold-based trading model on top 10 pairs	33
4.4	Input Feature Combinations tested to develop models	37
4.5	Average RMSE% for each feature window and feature combination	37
4.6	Average RMSE% comparison when using the stock pair closing price	38
4.7	Best performing models for each stock	38
4.8	Accuracy of models on each forecasting day	39
4.9	Structure of table "Real Values"	40
4.10	Structure of table "5Days Forecasting"	40
4.11	Structure of table "Pairs Table"	40
4.12	Base Structure of table "Models"	41
4.13	Machine configuration where all tests were run	50
4.14	Table generated by the python script with the cointegration values for all pairs	51
5.1	Pre-Covid Standard Deviation of Spread during Test Period	55
5.2	Covid Standard Deviation of Spread during Test Period	55
5.3	Portfolio Decline Days when using the STBM	56
5.4	Portfolio Decline Days when using the Enhanced Model	56
5.5	Profit comparison between both models	56
5.6	Number of positions opened comparison between both models	57
5.7	Key actions performed by both models on the same opportunity	60

5.8 Profit per Transaction comparison between the two models	61
A.1 First Half Stocks Considered for the NASDAQ-100	69
A.2 Second Half Stocks Considered for the NASDAQ-100	70

Acronyms

NN	Neural Network
BP	Backpropagation
RNN	Recurrent Neural Networks
LSTM	Long Short-Term Memory
SMA-k	Simple Moving Average
RSI	Relative Strength Index
ETF	Exchange-traded funds
LLE	Local Linear Embedding
AUC	Area under the curve
RMSE	Root-mean-square deviation
SSD	Sum of Squared Distances
STBM	Simple Threshold Based Model

1

Introduction

Contents

1.1 Topic Overview	2
1.2 Objectives	4
1.3 Report Outline	6

1.1 Topic Overview

Pairs Trading is a popular market-neutral¹ investment strategy developed in the 1980s. It is a simple but yet important long/short² equity investment tool that will be of fundamental understanding for the topic of this work. To implement this strategy, two marketable securities³ whose prices have historically moved together will be paired up. Usually, these pairs can be found in companies that have similar core-businesses, operate in the same geographical area or share the same owner or parent company. In Figure 1.1 it is visible that the price series from the two biggest pharmaceutical companies in the world tend to behave similarly.

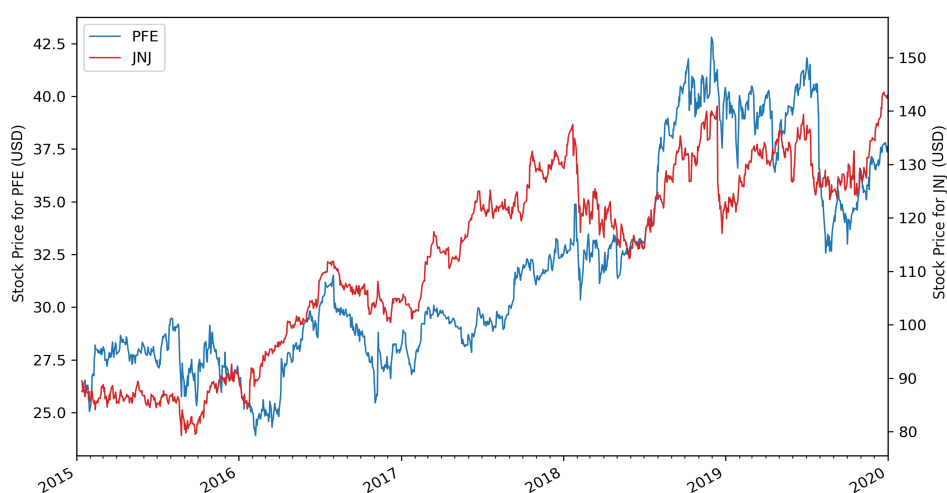


Figure 1.1: Price series from Pfizer Inc. (PFE) and Johnson & Johnson (JNJ)

A pair of securities whose prices behave identically are eligible for the second step of this strategy. The opportunities for investment in "Pairs Trading" rely on the premise that if the stock prices of the securities in the pair have followed each other, then it should continue in the future. Accordingly, if there is a divergence, it should mean that it is an attractive opportunity to invest assuming the prices will converge afterwards. These opportunities are found through the monitorisation of the spread⁴ of the pair. Whenever there is a spread anomaly, a market position is entered, then, after the prices converge, it is exited.

¹A market-neutral strategy seeks to profit both from increasing and decreasing prices in one or more markets, while trying to avoid market risk

²involves buying (long position) equities that are expected to increase in value and selling (short position) equities that are expected to decrease in value

³investments that can easily be bought, sold, or traded on public exchanges. I.e. stocks, ETF's, currencies etc.

⁴the spread is defined to be the ratio between the price of two securities

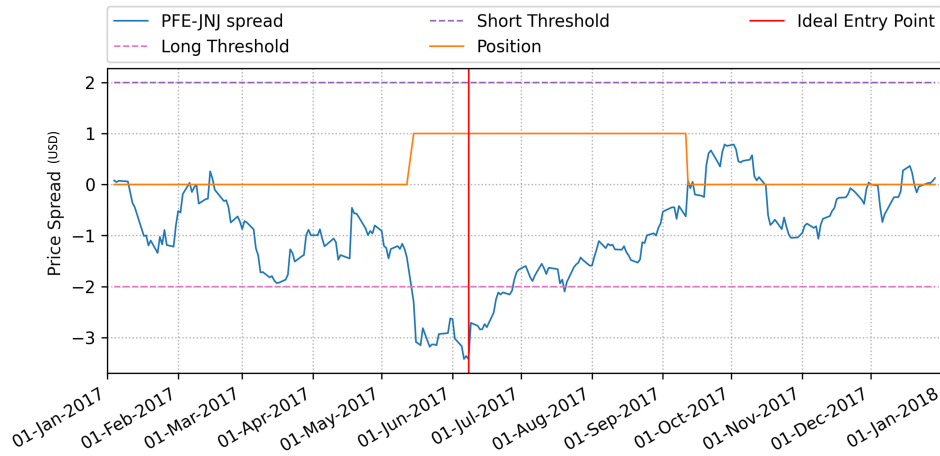


Figure 1.2: Example of Pairs Trading applied to a pair of securities

In Figure 1.1 it is visible that during the year of 2017 an anomaly occurs and both prices start diverging. Knowing that, Figure 1.2 was plotted where it is represented the normalized spread, defined as:

$$S_t = \frac{(PFE_t/JNJ_t) - \tilde{x}_{200}}{\sigma_{200}} \quad (1.1)$$

during the year of 2017. It is worth noting that \tilde{x}_{200} and σ_{200} represent the mean of the ratio and the standard deviation respectively, of the previous 200 days⁵. This value is arbitrary and it is used to prevent ever growing spreads that will incur in huge losses, a lower look-back period results in a more unstable spread. Figure 1.2 could have been plotted in real-time since all values rely on data from the past. Also in Figure 1.2 a Long and Short thresholds⁶ were plotted and will influence the market position⁷ (orange line) whenever the spread crosses them. In orange, it is plotted the market position, where a value of '1' means a Long position has been opened and a value of '-1' means that a Short position has been opened instead.

As depicted in Figure in 1.2 the position is entered whenever the spread crosses either the long or the short thresholds. During this period, one position is opened when the spread crosses the Short Threshold. This would have been a favourable transaction for the investor since the Spread reverted to its mean.

This model comprises four main steps:

I Calculate the mean \tilde{x}_{200} and the standard deviation σ_{200} of the ratio⁸, and the spread 1.1.

II Define both Long and Short Threshold that will define when a market position may be opened

III Monitor if the spread has crossed any threshold

⁵this number will be referred to as "look-back period"

⁶these values are arbitrary and as it gets closer to zero the risk the investor faces increases

⁷it can take values -1, 0 or 1 meaning the current position is short, on hold or long, respectively

⁸the look-back window of 200 days should be defined by the investor

IV In case any threshold is crossed, open the respective market position

In this example, the Long Position would have been opened on the 15th of May, Since it was a Long Position, the investor would buy PFE stocks and sell JNJ. Later that year, on the 10th of September the spread would go back to its mean and on that the day the investor would close its position. The application of this model can be summarized in Table 1.1 confirming that this was a favourable transaction.

Table 1.1: Application of the Threshold-based trading model

	PFE (USD)	JNJ (USD)	Observations:
# Shares	363	87	Investing 10,000 \$ in each stock
15-may-2017	27.57	114.85	Long position: Buying PFE and Selling JNJ
12-Sep-2017	29.73	121.5	Ending the position: Selling PFE and Buying JNJ
Profit per Share	$29.73 - 27.57 = 2.16$	$114.85 - 121.5 = -6.65$	Worth noticing that a profit is made in only one stock
Profit per Stock	$2.16 \times 363 = 784.08$	$-6.65 \times 87 = -570.55$	The profit in PFE counteracts the loss in JNJ
Total profit		$784.08 - 570.55 = 205.53$	Since the spread behaved as expected the investor profited from this transaction

Pairs trading always involves a long and a short position in each of the constituents of the pair. It is sometimes referred to as a market-neutral strategy since it profits from the pairs convergence instead of the security's price itself.

1.2 Objectives

Research in the field relies on purely statistical data to enhance this strategy. Even though Machine Learning applications have exponentially grown in the financial market, concerning Pairs Trading, there haven't been many improvements. This lack of research opens up a compelling opportunity to explore Intelligent Computation methods applied to the Pairs Trading strategy.

Out of the two main steps explained in 1.1 that compose this strategy, a big majority of the work done in this area, focuses on the pair selection problem, weather using Lead-Lag indicators [1], or even multi-asset trading [2].

The Pairs trading strategy profits from the mean reversion of the spread. However, in the example presented by Figure 1.2, the market entry point occurs whenever the threshold is crossed, missing the optimal entry point. This will lead to periods of high uncertainty, as the pair continues to diverge, and will not return the optimal profit.

1.2.1 Improve Forecasting with Financial Indicators

Through a forecasting method presented in 3.2, with the help of financial Indicators, as well as the evolution of the respective pair's price, the best entry point will be predicted. The high amount of data as well as the complexity that involves predicting stock prices, can only be solved by using state-of-the-art Deep Learning models explained in 2.

1.2.2 Increase Accuracy

With a more accurate forecasting model, this theses aims to increase the accuracy of the entry and closing points in the Pairs Trading strategy. Based on the forecast for the following five days, an algorithm will decide to either hold or open a market position. This will eventually lead to a better entry or exit points, providing the best profit out of each transaction.

In Figure 1.3 it is depicted an high-level flowchart of how this work proposes to enhance Pairs Trading. Each module presented in the picture will be explained in Chapter 3 and later on implemented.

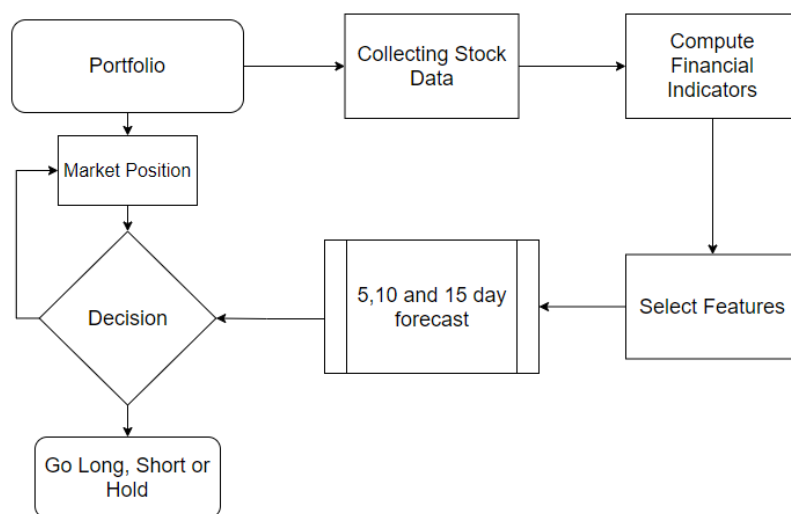


Figure 1.3: Flowchart of the proposed work

1.3 Report Outline

This work is composed by a total of 6 chapters. In chapter 1 an overview of Pairs trading is introduced, as well as, the problem that this work aims to solve. Chapter 2 focuses on the fundamental concepts of Pairs Trading and presents an overview of related works and findings. Finally, in chapter 3 it is described the proposed approach that aims to enhance the Pairs Trading investment strategy.

In the fourth chapter, the approach described in 3 will be further detailed and every step made will be explained. Following the implementation chapter, will be the validation one, where the results from the project will be assessed and the first conclusions will be taken. Lastly, there will be a full chapter dedicated to the main conclusions of the project as well as some suggestions for the following works done on this topic.

2

Key Concepts and Background

Contents

2.1 Financial Concepts	8
2.2 Pairs Selection	13
2.3 Stock Price Prediction	17
2.4 Trading Strategies	20

This chapter highlights several fundamental concepts of Pairs Trading as well as an overview of relevant findings on this topic.

2.1 Financial Concepts

In order to understand the purpose of some decisions throughout this work, it is fundamental to understand the following financial concepts.

2.1.1 Financial Market

The Financial Market can be broadly described as any marketplace where the trading of securities occurs, as in the stock market, bond market or money market. Financial markets play a vital role in facilitating the smooth operation of capitalist economies by allocating resources and creating liquidity for businesses and entrepreneurs.

2.1.2 Long and Short Market Position

In the financial market investors and analysts often use the terms long and short with many different meanings. When associated with the word "Position", instead of a reference to its length, long and short positions are a reference to the stocks the investor owns or needs to own.

The most traditional way of investing in the stock market is through long positions. If an investor has bought a share of stocks and owns them, it means that the investor has long positions. He may close this position afterwards and trade the ownership of those stocks for the current market value.

Long trade - Purchasing an asset and waiting for its value to rise so that it can be sold afterwards.

- Profit from: The price rising (Profit = Exit Price - Entry Price)
- Long Potential: Profit can be unlimited since the price of the asset can rise indefinitely
- Long Risk: The price of the asset can only drop to 0\$ so you can only lose what you invest

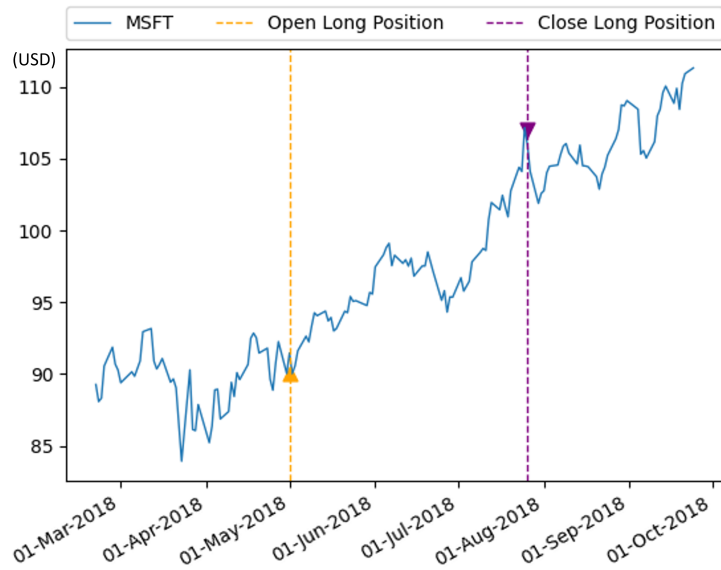


Figure 2.1: Example of a profitable Long Trade

On the other hand, if an investor has a short position, it means that he has sold a certain amount of shares without having its ownership. The short investor now owes those shares and must fulfill the commitment by buying them later on.

Short trade - Selling an asset and waiting for its value to go down so that it can be bought afterwards.

- Profit from: The price dropping (Profit = Entry Price - Exit Price)
- Potential: Profit is limited to the amount initially received on the sale
- Risk: The risk is unlimited since you will have to buy the stocks you initially sold so your losses can be unlimited

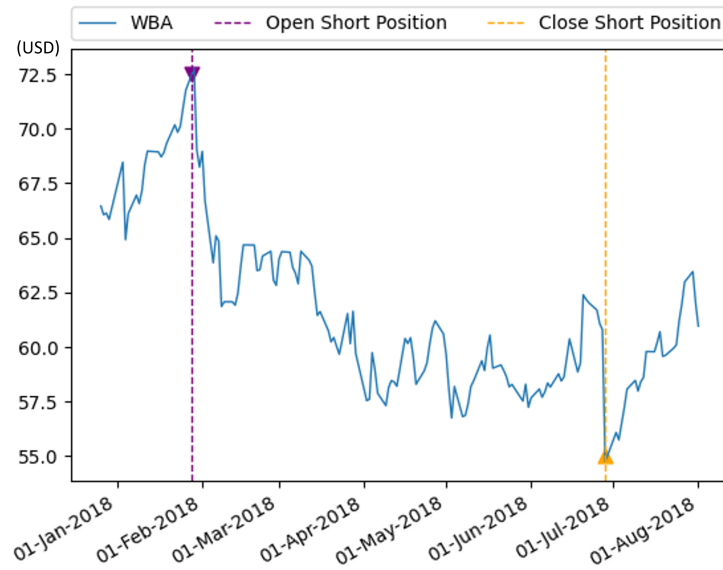


Figure 2.2: Example of a profitable Short Trade

2.1.3 Simple Moving Average

A Simple Moving Average (SMA-k) is an arithmetic moving average calculated by adding the most recent prices and then dividing that figure by the number of time periods in the calculation average. The 'k' is the amount of time periods added. The SMA-k is a lagging¹ indicator that can be used to confirm the a bullish or bearish trend. The SMA-k can be calculated as follows:

$$SMA(k)_t = \frac{P_t + P_{t-1} + \dots + P_{t-(k-1)}}{k} \quad (2.1)$$

where P_t is the price of an arbitrary stock for the time 't'. In Figure 2.3 it is possible to see the evolution of Pfizer stock's price as well as its 10 and 50 day moving average. Also, as referred above, it is clear to see the "lag" associated with the SMA, and that the bigger the value of 'k' is, the longer it takes to the SMA to react to a price variation.

¹A lagging technical indicator is one that trails the price action of an underlying asset, and traders use it to generate transaction signals or confirm the strength of a given trend

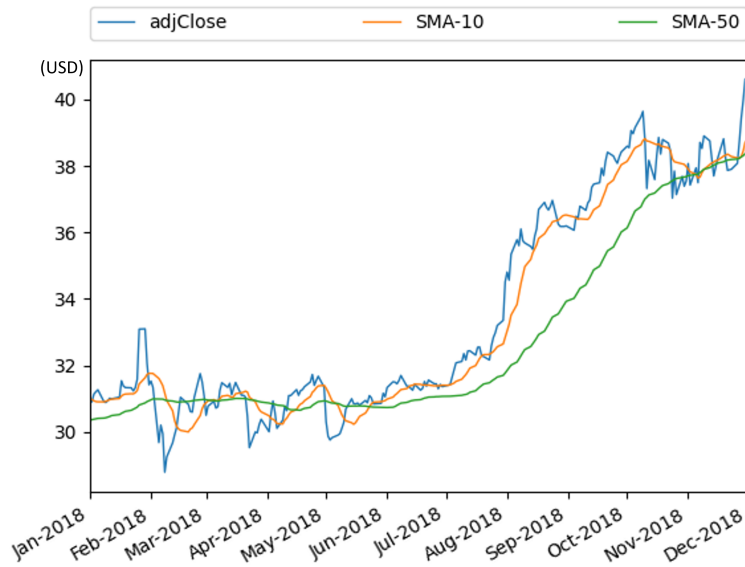


Figure 2.3: Price evolution of Pfizer and its moving averages

All the periods in the range have the same weight when calculating the Simple Moving Average. In order to have more emphasis in the latest periods an Exponential Moving Average can be used.

2.1.4 Stochastic Oscillator

The Stochastic displays, as a percentage, how a closing price compares to the high/low range over a defined period (14 days by default). It is some times referred to as Stochastic Oscillator, and even though it was developed in the 1950s, it is still one of the most used financial indicators today.

There are two measures in a Stochastic, the K and the D lines (or %K and %D). The k line can be computed as follows:

$$\%K_t = \frac{P_t - L_{14}}{H_{14} - L_{14}} \times 100 \quad (2.2)$$

where H_{14} and L_{14} are respectively the highest and lowest prices of the past 14 trading days. The value of %D can be calculated through a Simple Moving Average of the past 3 days of the value of %K.

The most conventional way of using the Stochastic Oscillator is to be aware of potential crossovers of the **D** line into the "Overbought"(above 80%) or "Oversold"(below 20%).



Figure 2.4: Price evolution of Pfizer and its K and D lines

2.1.5 Relative Strength Index

The Relative Strength Index (RSI) is one of many momentum indicators that measures the impact of recent price changes in the overall momentum of the time series. The RSI has a similar reading as the Stochastic since both can take values from 0 to 100 and can be calculated as:

$$RSI_t = 100 - \frac{100}{1 + R_{14}} \quad (2.3)$$

$$R_{14} = \frac{(Previous\ Average\ Gain \times 13) + Current\ Gain}{-((Previous\ Average\ Loss \times 13) + Current\ Loss)} \quad (2.4)$$

where for the calculation of the R_{14} the "Previous Averages" are calculated with the previous 13 trading days. An RSI value over 70 indicates that a security is "Overbought" and may be on the verge of a momentum shift. An RSI value under 30 is considered "Oversold".

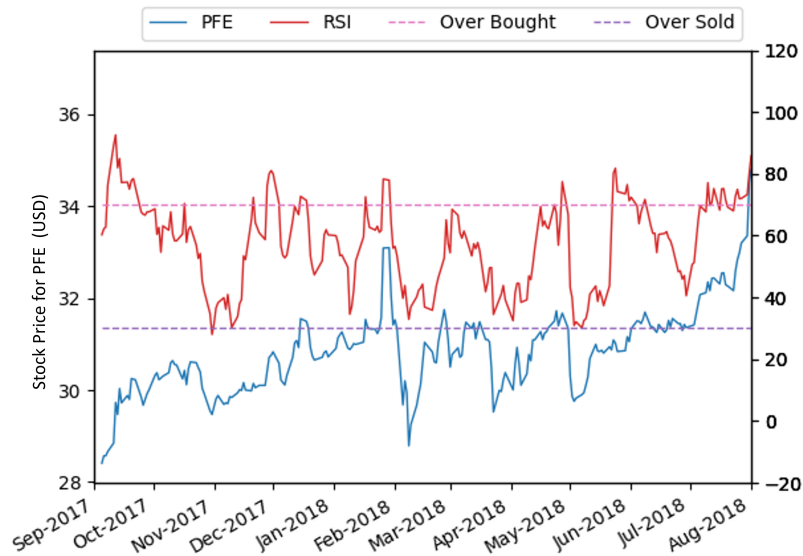


Figure 2.5: Price evolution of Pfizer and its Relative Strength Index

2.1.6 Risk Management

The stock market is known for having a strong risk-return relation. Usually, high risks mean greater returns. Risk Management is the process of identifying and estimating risks and develop strategies to minimize them, increasing overall profitability.

The most well-known risk management strategy **Portfolio Diversity**. In the financial market, there are several products, such as equities, Exchange-traded funds (ETF) or bonds that can be combined to diversify the portfolio. Usually, investors go one step further and combine these products across several companies, sectors and geographical areas. This protects the overall profit from specific market fluctuations.

The second most famous risk management strategy is the **Stop Loss**. This is a threshold at which the investor is willing to sell its securities in case of a very unfavourable scenario.

2.2 Pairs Selection

Selecting pairs for this strategy comprises two main steps: (i) selecting all eligible securities for the portfolio and (ii) pairing them up together in the most promising way possible.

Regarding the first step, there have been works using two different approaches. The first one is selecting specific industries, countries or another particular group [3, 4]. This method will result in more predictable pairs and will save a lot of computing time. The second approach consists in combining all

the possible combinations and may result in some unusual couples but will comprise a higher risk of both securities diverging indefinitely.

After selecting the group of eligible securities, the investor must define which ones to combine to form the most promising pairs. The most common procedures to select pairs involve the squared distance, cointegration and correlation approaches that will be explained in more detail in the following sections.

2.2.1 SSD measure

The SSD measure stands for "Sum of Squared Distances" of the price series of two stocks. However, since each security has a different price range, the most common way to mitigate these differences is to normalize the values. For each time series, the normalized price is determined by:

$$P'_t = \frac{P_t - \tilde{x}}{\sigma} \quad (2.5)$$

where P_t is the price of the asset at the moment t , \tilde{x} is the average price of the asset throughout the time series, σ is its standard deviation and lastly P'_t is the normalized price. After the normalization the calculation of the SSD measure can be done, using the following equation:

$$SSD_{x,y} = \frac{1}{n} \sum_{t=1}^n (x'_t - y'_t)^2 \quad (2.6)$$

For an optimal pair, the investor is looking to minimize this value since it would mean that both securities' price series have had similar behaviour in the past. However, if this value is too close to zero it would mean that this pair wouldn't offer many trade opportunities, hence why it shouldn't be used alone to select pairs.

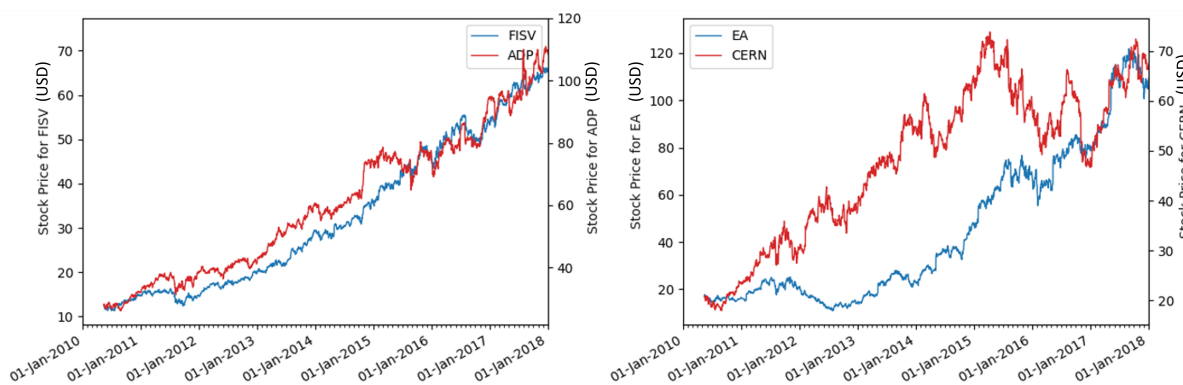


Figure 2.6: Two Examples of pairs with different SSD measures 0.07 (left) and 0.968 (right)

Figure 2.6 shows Two Examples of pairs with different SSD measures. It is clear to understand that in a pair with a lower SSD, both stocks have a more similar behaviour than in a pair with a higher SSD.

2.2.2 Cointegration

Two series (x and y) are said to be cointegrated if the linear combination $x_t - \beta_2 y_t$ is stationary. Firstly and using the Engle-Granger two-step approach, the static regression shall be estimated by:

$$x_t = \mu + \beta_2 y_t + u_t \quad (2.7)$$

where μ and β_2 are constant values, and u_t is a residual term that must be stationary in order for the series x and y to be cointegrated. Here the Dickey-Fuller [5] test is performed to test the null hypothesis of no cointegration.

After proving that there is a cointegration relationship, the 'p-value'² is obtained through regression surface approximation as explained in by MacKinnon in [6, 7]. Pairs with 'p-values' under 0.5 are considered mean-reverting³ stock pairs. Also, 'p-values' above 0.1 mean that the correspondent pair is likely to be non-stationary leaving the investor with the pairs with a 'p-value' lower than 0.1. Figure 2.7 represents how this approach is used to select the best pairs for Pairs trading. In this case, an arbitrary threshold is defined by 'i' which the investor determines based on his willingness to take risks.

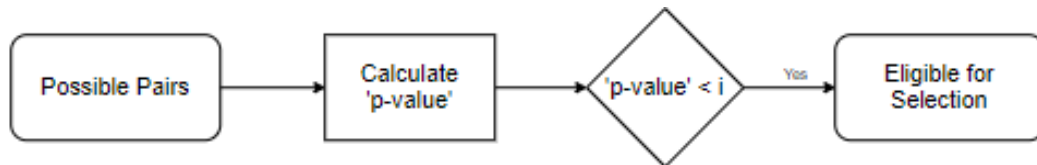


Figure 2.7: Cointegration method for pairs selection

Figure 2.8 shows a side-by-side comparison of a well cointegrated pair of stocks (on the left) with a p-value of 0.015, against a pair of two non-cointegrated stocks (on the right) with a p-value of 0.861.

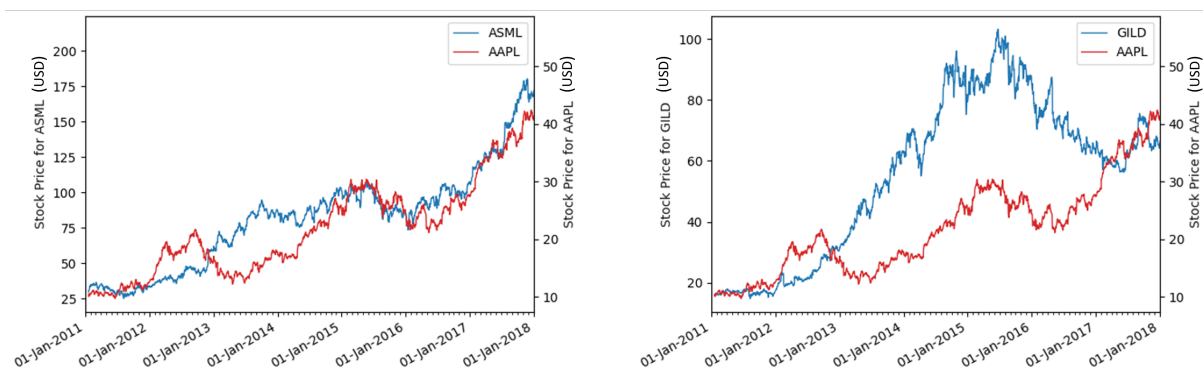


Figure 2.8: Two Examples of pairs with different p-values

²The p-value ranges from 0 to 1

³Mean reversion is a theory used in finance that suggests that asset prices and historical returns eventually will revert to the long-run mean or average level of the entire dataset

2.2.3 Correlation

Pairs trading has been used by investors primarily on highly correlated assets since their prices tend to have similar behaviours. For pairs selection it is usually used the Pearson correlation method that defines the correlation between two price time series x_t and y_t by:

$$CORR_{X,Y} = \frac{\sum_{t=1}^n (x_t - \tilde{x})(y_t - \tilde{y})}{\sqrt{\sum_{t=1}^n (x_t - \tilde{x})^2} \sqrt{\sum_{t=1}^n (y_t - \tilde{y})^2}} \quad (2.8)$$

where \tilde{x} and \tilde{y} are the average values of the time series x_t and y_t respectively. In [8] it is reported that the correlation measure is has a big impact on overall return and risk. In general stock pairs with higher correlation tend to be better candidates for pairs trading.

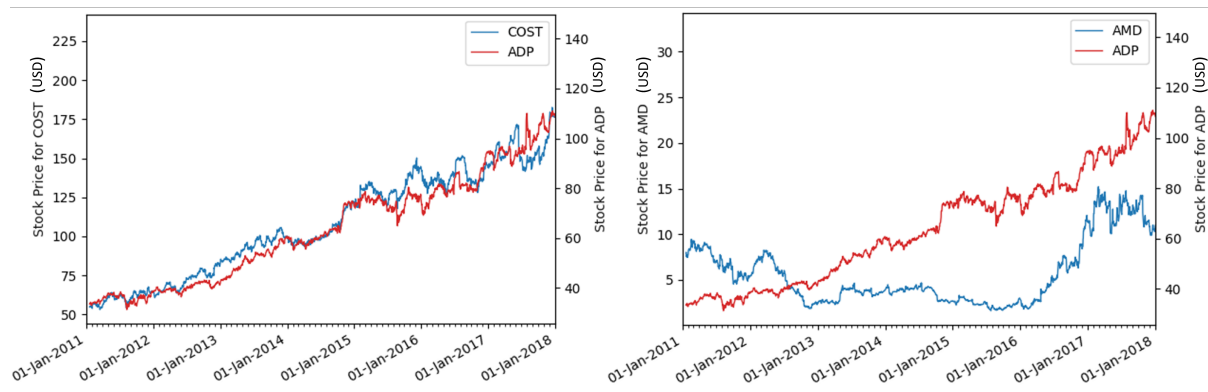


Figure 2.9: Two Examples of pairs with different correlation values

In Figure 2.9 it is clear that the Pearson Correlation Method is a powerful tool to select pairs which stock prices tend to behave accordingly. In the left picture it is plotted an example of a pair with a correlation value of 0.984 and on the right an example with a correlation value of 0.138.

2.2.4 Conclusion

The approaches just described are the most commonly used in the literature, however, they rely purely on statistical information. Some other approaches include specific pairs which tend to display an equilibrium that is explained by its geographical location [9] or a theoretical linkage between markets [10].

The main goal of this work is to enhance the Pairs Trading as a whole, hence why, in the next chapter, the methods to select the eligible pairs will be purely analytical.

2.3 Stock Price Prediction

As mentioned in 1.2 in this work Deep Learning will be used to predict the stock prices of each security of the Portfolio. Deep Learning is a branch of Machine Learning that uses different types of neural networks that are computing systems inspired by the biological neural networks in the human brain. These are composed of interconnected nodes that through different algorithms can classify data, recognise patterns or correlations and continuously learn and improve over time.

In the following subsections there are several references on how different investigations have used Deep Learning to predict the evolution of the stock market.

In the end, the main objective of the model is to return, as accurate as possible, a prediction for what the price of a certain stock will be on the following day. In Figure 2.10 it is illustrated the output of a forecasting model on an arbitrary stock.

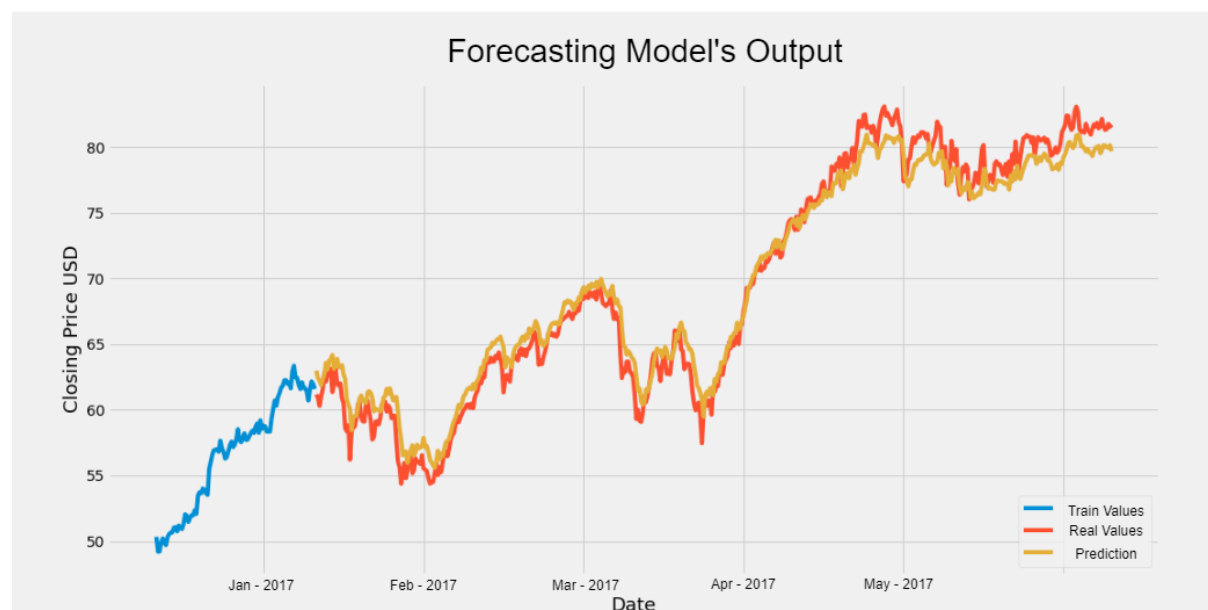


Figure 2.10: Illustration of the output of the Stock Price Prediction model

2.3.1 Backpropagation Algorithm

The Backpropagation algorithm was first introduced in 1960s and later popularized in [11] is one of the most fundamental pieces in a neural network. When training a Neural Network (NN) the backpropagation algorithm computes the weights and biases for each layer of the network to minimize the difference between the output vector and the desired one.

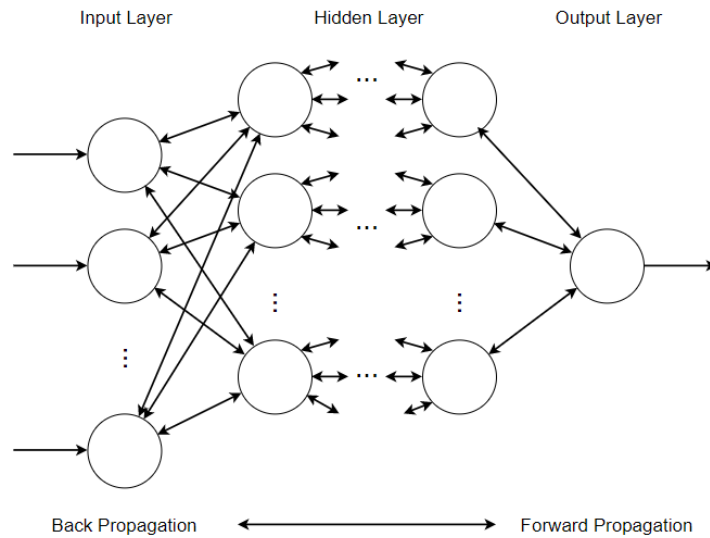


Figure 2.11: Structure of BP neural network

In [12] it is reported all the steps that the Backpropagation (BP) algorithm comprises when training the neural network, as well as, its accuracy on predicting stock market. Also in [13] it is used the BP algorithm onto the classification and prediction of stock market patterns.

2.3.2 RNN and LSTM

Recurrent Neural Networks (RNN) are a variance of NN where the output at each step is fed into the next one (Figure 2.12) whereas in regular NN's all the inputs and outputs are independent of each other. This makes them suitable to tasks such as text and speech recognition [14–16], or stock price prediction [17–19].

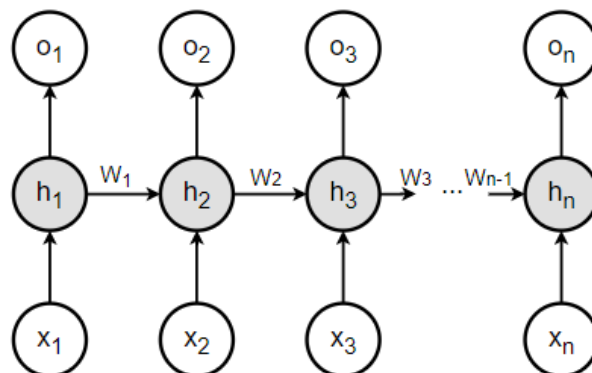


Figure 2.12: Recurrent Neural Network Structure

Considering the RNN structure, if every hidden unit (h_x) is replaced by a Long Short-Term Memory (LSTM) cell and between every cell there is another connection called "Cell State" the resulting structure

is what is called Long Short-Term Memory Network. Each LSTM cell defines its internal state as a function of the current state and input, through a gating mechanism. In Figure 2.13 it is depicted a scheme of an LSTM cell.

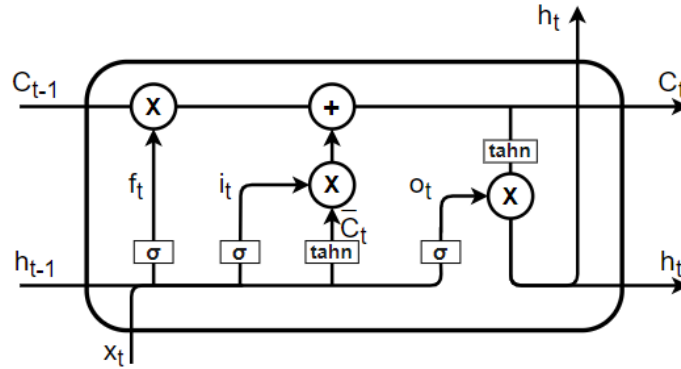


Figure 2.13: Long Short-Term Memory cell

Each cell has three binary gates: the Input Gate (i^t) controls whether the memory cell is updated or not. The Forget Gate (f^t) controls if the memory cell is reset to zero, and the Output Gate (o^t) defines which information of the current cell affects the output of the network. Each value presented in Figure 2.13 can be calculated as follows:

$$\begin{cases} i^t = \sigma(W^i x^t + W^i h_{t-1} + b_i) \\ f^t = \sigma(W^f x^t + W^f h_{t-1} + b_f) \\ o^t = \sigma(W^o x^t + W^o h_{t-1} + b_o) \\ \overline{C}^t = \tanh(W^c x^t + W^c h_{t-1} + b^c) \\ C^t = f^t C^{t-1} + i^t \overline{C}^t \\ h^t = \tanh(C^t) o^t \end{cases} \quad (2.9)$$

LSTMs are a variance of Recurrent Neural Networks RNN that have the capability of using data from the past for future predictions. This particular feature puts LSTM among the most popular deep learning models used to predict time series [20, 21].

2.3.3 Conclusion

With the exponential growth in computer power over the last years, it became easier and more affordable to train complex deep learning models capable of dealing with larger amounts of data, hence why there has been more and more research regarding the financial market. Many types of Neural Networks have been used on various case studies across different markets as stated in Table 2.1.

Starting with the Backpropagation Algorithm, in [13] it is tested its capabilities to predict stock price patterns. The authors achieved a 10% increase in accuracy when compared to the deep learning fuzzy algorithm model. Also, in [12] a Local Linear Embedding (LLE) dimensional reduction algorithm is se-

lected to reduce the dimension of the factors affecting the stock price, that lead to a higher accuracy than the traditional Backpropagation Algorithm.

With regard to RNN, Wang [17], through an ensemble learning (RNN-EL) framework combined trade-based features derived from trading records and characteristic features of the list companies and outperformed state-of-the-art approaches by an average of 29.8% in terms of Area under the curve (AUC) value. Another variant of RNN is the LSTM and Yadav [20] states that the best way of configuring these networks for price prediction is through the use of a single hidden layer since it provides better stability and a smaller Root-mean-square deviation (RMSE).

Lastly and fully applied to Pairs trading, Sarmiento [22], not only uses an Unsupervised Learning algorithm to increase the average portfolio Sharpe ratio to 3.79 (compared to 3.58 and 2.59 obtained by standard approaches), but also introduced a forecasting-based trading model capable of reducing the periods of portfolio decline by 75%.

Table 2.1: Literature exploring the application of Machine Learning in Stock Market Prediction

Work	Sample	Type of NN	Results
Yu et al. [12]	Pingtian Development, 01-01-2016 to 08-08-2017	LLE-BP	Higher prediction accuracy than the BP neural network model.
Zhang et al. [13]	Gree Electric, 10-12-2019 (5 min interval)	BP	BP algorithm neural network model has a better accuracy than the deep learning fuzzy algorithm model. (73.3% vs 62.1%)
Wang et al. [17]	Chinese Stock Market, 2012 to 2016	RNN-EL	RNN-EL outperforms state-of-the-art approaches by an average of 29.8% in AUC.
Yadav et al. [20]	Indian Stock Market (NIFTY-50), 29-12-2008 to 24-05-2019	LSTM	Using one hidden layer offers better accuracy, easier training and less risk of over-fitting.
Sarmiento et al. [22]	208 commodity-linked ETFs from Jan-2009 to Dec-2018	LSTM	Better average Sharpe ratio (3.79 vs 3.58 and 2.59). Reduce portfolio decline by 75%.

2.4 Trading Strategies

The act of buying and selling securities based on short-term movements is sometimes called "Active Trading". Active trading differs from the long-term "Buy-and-hold"⁴ strategy on the belief that short-term movements are where profits can be made. Active trading is a strategy that involves "beating the market"⁵ through identifying and timing profitable trades, often for short holding periods.

⁴Buy and hold is a passive investment strategy in which an investor buys securities and holds them for a long period regardless of fluctuations in the market.

⁵The phrase "beating the market" is a reference to an investor that has better results than the market standard

2.4.1 Day Trading

Day Trading is perhaps the most well-known active trading style. As the name implies, this strategy profits from buying and selling securities within the same day and no positions are held overnight. It is traditionally done by professional traders, however, with the rise of electronic trading, this practice has become more popular among novice traders.

2.4.2 Swing Trading

Swing traders benefit from the high price volatility whenever a trend breaks and the new one tries to establish itself. Swing trades are usually held for a couple of days. This strategy is complemented by a set of statistical analysis that are designed to identify the best entry and exit points. Swing trading can provide bigger profits when applied to markets that change trends somewhat frequently.

2.4.3 Scalping

Scalping is one of the quickest trading strategies, exploiting price gaps caused by the bid-ask spread⁶. The strategy works by buying the bid price⁷ and selling at the ask price⁸ and receive the difference between the two. To decrease the risk, Scalpers tend to hold their positions for a short period. Since the level of profits per trade is small, this strategy is usually applied to more liquid markets⁹ to increase the frequency of the trades.

2.4.4 Position Trading

Position Trading (or Trend Trading) is the active form of the "Buy-and-hold" strategy. Its trades usually can last for weeks, months and sometimes even more, depending on the trend. Trend traders aim to "ride the wave" by opening a position when the trend has been established and exiting the position when the trend breaks. This means that in high market volatility, the amount of market positions is generally reduced making trend trading less effective.

2.4.5 Conclusion

In Pairs trading, positions can have a duration of a couple of days or a couple of months depending on how fast prices converge or diverge. A particularity of pairs trading is also that (usually) the longer the investor has a position opened, the lower is its return. This might seem contradictory, but if we look at

⁶A bid-ask spread is the amount by which the ask price exceeds the bid price for an asset in the market

⁷A bid price is a price which is offered for a security.

⁸An ask price is the price a seller states they will accept for a security.

⁹Liquid markets have many available buyers and sellers, its prices change in small increments and it is easy to execute trades quickly and at the desired price.

the formula 1.1, regardless of the "look back period", it can happen that the spread stays roughly the same and due to the change in the moving average, the normalized spread converges to 0. This factor gives a false sense of profitability to the investor.

On the enhanced model presented in 4 it will be used a stop-loss function to avoid positions from being held for too long and having huge losses.

3

System Architecture

Contents

3.1 Pairs Selection	24
3.2 Time Series Forecasting Model	27
3.3 Trading and Portfolio Management Models	28
3.4 Project's Structure Overview	30

This work will comprise three main steps. The first will focus on the selection of the pairs since it is a key part of Pairs Trading. It will go through a series of filters and iterations based on some of the criteria described in 2.2. Also, during this stage, the application of the Threshold-based trading model mentioned in 1 is simulated, to have an empirical verification of the suitability of the chosen pairs for Pairs Trading. This stage isn't the main goal of the project, however it will serve as a starting point for the two following stages.

In a second stage, a Deep Learning approach is presented and aims to achieve the best possible accuracy when trying to predict the evolution of each of the stocks of the previously selected portfolio. To help increase the performance of the Neural Network a number of financial indicators will be used, as well as, data from the respective pair. There are some variables involving this process such as the number and type of financial indicators to use as features of the NN. For each stock that was selected in the first stage, five models were built to predict the stock's evolution in the following five days.

The last stage will use the predictions made in stage two and will try to optimize the strategy. It will start by trying to predict the spread's behaviour and select the best date to open and close a market position. The ultimate goal of this stage is to increase the profit of the "Threshold-based trading model".

In Figure 3.1 it is depicted the dependencies that each model has with its previous. Also, for each model, it is indicated the main tools that were used as well as the range of data that was used in order to avoid any incoherence.

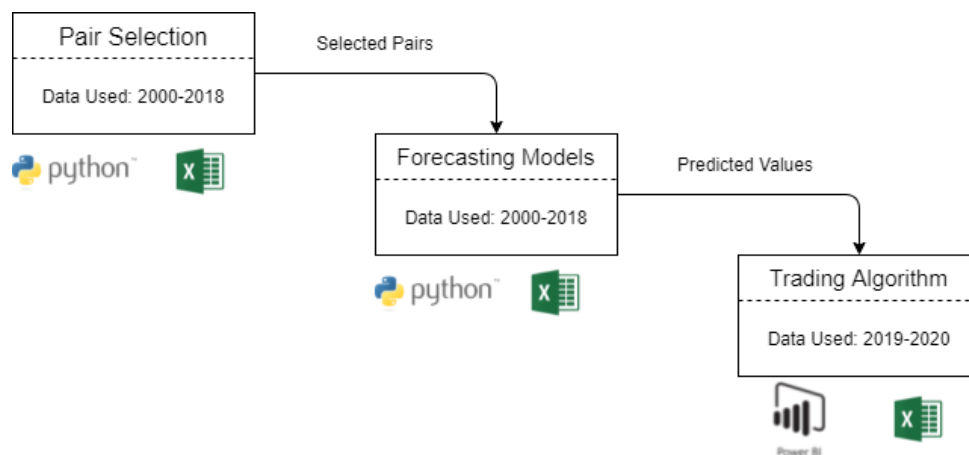


Figure 3.1: Architecture of the project

3.1 Pairs Selection

As stated in 1.2, the Pairs selection methodology isn't the main focus of this work but can by no means be skipped since it is vital for the success of this strategy. The pairs trading strategy highly relies on the assumption that the spread of the pair will revert to its historical mean. Skipping this step or having a dismissive approach to the pair's selection might lead to huge losses due to the Long/Short market

position's risks described in 2.1.2.

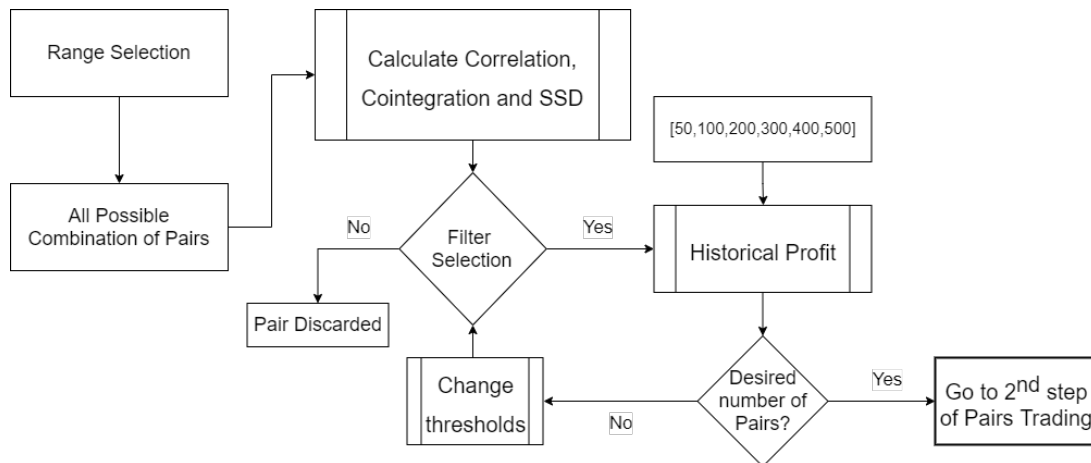


Figure 3.2: Pairs selection process

3.1.1 Range Selection

Pairs trading can be applied to any asset in the stock market like stocks, Exchange-traded funds ETF, currencies or indexes. The eligible securities for pairs trading don't need to come from the same country or industry. However, in order to reduce the possible range of electable pairs, it is common that investors opt to restrict the pool of securities within a certain industry, core business or country.

There are a couple of indexes that measure the value of a section of a country's stock market via a weighted average of selected stocks and can be used as a pool of stocks. The three most common types of indexes are the Global, Regional and National indexes, some of them are presented in the following table:

Table 3.1: Example of Indexes that could be used as a range selection

Global Indexes	Regional Indexes	National Indexes
S&P Global 100 Index	S&P Asia 50 Index	FTSE techMark 100 Index
S&P Global 1200 Index	Euro STOXX 50 Index	NSE of India Index
Dow Jones Global Titans 50	S&P Latin America 40 Index	NASDAQ 100

For this particular work, only the stocks listed in the Nasdaq-100 index will be evaluated. The Nasdaq-100 is one of the most preminent large-cap growth indexes, as it includes one hundred of the largest non-financial companies listed on the Nasdaq Stock Market, based on market capitalization.

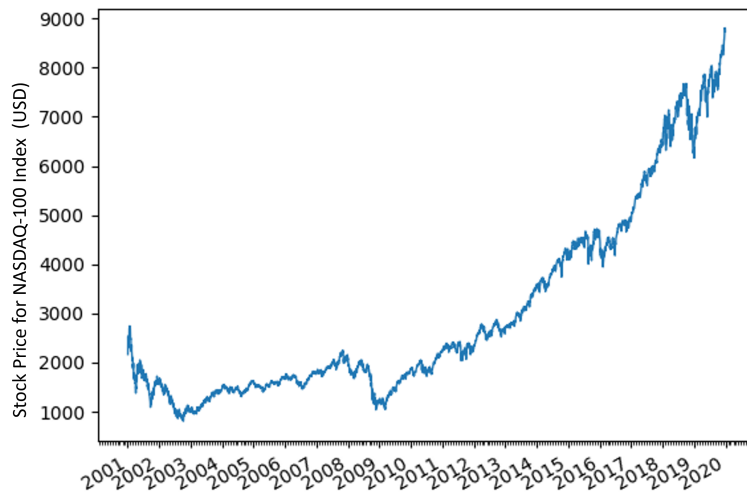


Figure 3.3: NASDAQ-100 Index Tracking Stock

3.1.2 Filter selection

To each one of these pairs, it is calculated both the cointegration, correlation, and Sum of Squared Distances (SSD) values. It is worth noting that, as explained in 2.2.2, 'p-values' closer to 0 means that there is a better cointegration between both stocks than the bigger 'p-values'. The correlation among two-time series also ranges from 0 to 1, however, the higher the value is, the better correlation exists among them. For a pair to be elected it has to have a 'p-value' below 'i' and a cointegration value above 'c', being 'i' and 'c' arbitrary thresholds.

Based on the filters selected by the investor, a different number of pairs will compliant and will be assessed in the following stage.

3.1.3 Historical Profit

After selecting a smaller group of pairs, the threshold-based trading model explained in 1 will be simulated and each pair's historical profits shall be analysed. This should give an empirical confirmation of which pairs suit the most this investment strategy. Also, for each pair, the lookback period described in the equation 1.1 was ranged from 50 to 500. Lower values of the lookback window, allow for more market positions opened, however, it reduces the percentage of profitable transactions. Studying how each pair would have performed in the past for each look back period value, should be a good indicator of how they will behave in the future.

3.2 Time Series Forecasting Model

Following the overall architecture of this project, after selecting the pairs that will compose the portfolio, the next step is to create a forecasting model to help predict each pair's spread. To achieve this goal, for every stock in the portfolio, a different combination of input features will be tested in a Neural Network. The models that offer the best performance will be used for each stock.

3.2.1 Training Model

As usual, the data from 2000 to 2018 will be separated into "training" and "test" data. For every stock within the portfolio, a new model for price prediction will be created, in order to maximize profit. This will also be an iterative process since some parameters can be changed to form new models. Once the desired performance is achieved, the model is saved.

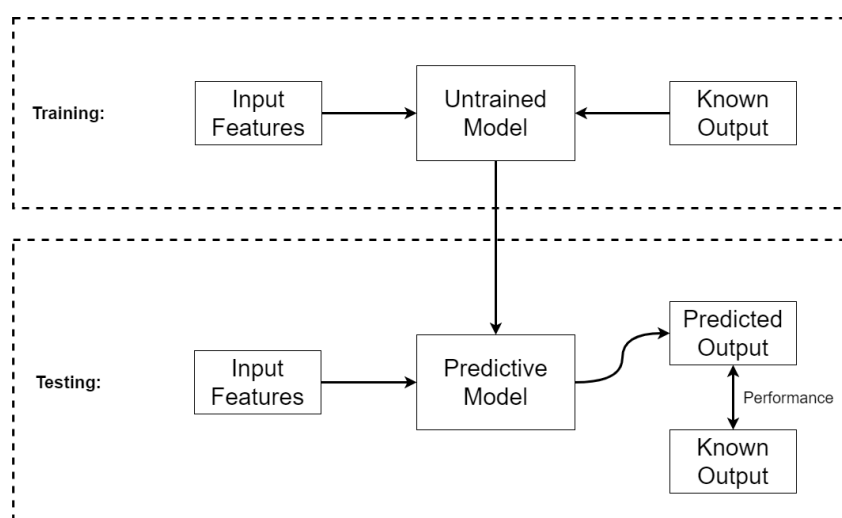


Figure 3.4: Division of Data into Training and Testing

3.2.2 Feature Preparation

As stated in 2.1 financial indicators are often used to support investors choices. Also, the price evolution of the respective stock's pair will be used as an input feature to train each model. All stocks were tested with all possible combinations of input features on two different train/test data divisions (80%-20% and 90%-10%). Also, different look-back windows¹ were tested (5, 10, 15 and 20 days).

All these parameters will be combined and all possibilities will be tested to find the best performing model for each one of the twelve stocks.

¹The look-back window represents how many days worth of information are fed into the model

3.2.3 Five day forecasting

The process explained in 3.2.1 and 3.2.2 will be performed once for each stock, and consecutively, twelve models with different combinations of input features and look-back windows will be saved. These are the models that achieved the best performance during the test phase. Afterwards, for each stock, four new models are going to be created with the same combination of parameters, to predict the next four days. At the end of this stage, for each stock, there are five models that aim to predict each one of the following five days.

3.3 Trading and Portfolio Management Models

3.3.1 Problem Statement

The Pairs trading strategy profits from the mean reversion of the spread. However, the threshold-based model described in 1 defines the market entry point whenever the threshold is crossed. This will lead to periods of high uncertainty as the pair continues to diverge as shown in Figure 3.5.

The proposed trading model aims to increase the accuracy of the optimal entry point. By doing so, periods of high uncertainty will be reduced and the overall profit should increase.

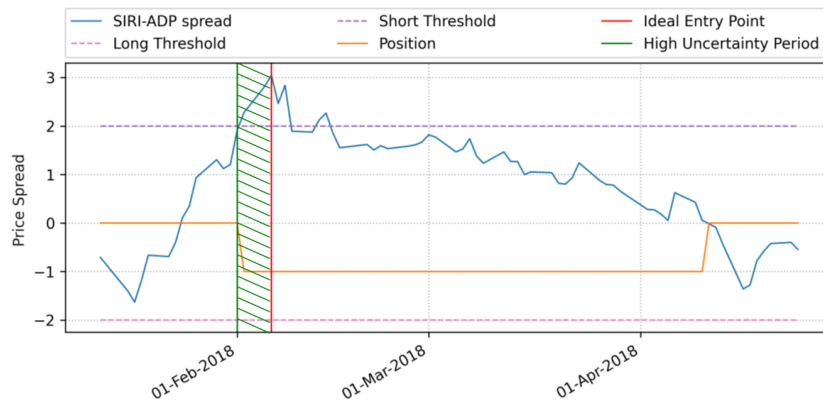


Figure 3.5: Example of a high uncertainty period

3.3.2 Trading Model

The proposed model will be triggered as the spread reaches near one of the two thresholds (short and long). From this point on, any time would be good to open a position, however, the ultimate goal is to enter on the "Ideal Entry Point" explained in Figure 3.5. Using the predicted prices obtained from 3.2, the algorithm will try and plot the evolution of the spread for the following couple of days. While the forecast for the following days keeps deviating the spread from the mean, the market position will remain on hold. As soon as the spread starts to revert back, the algorithm will open a position (long or short).

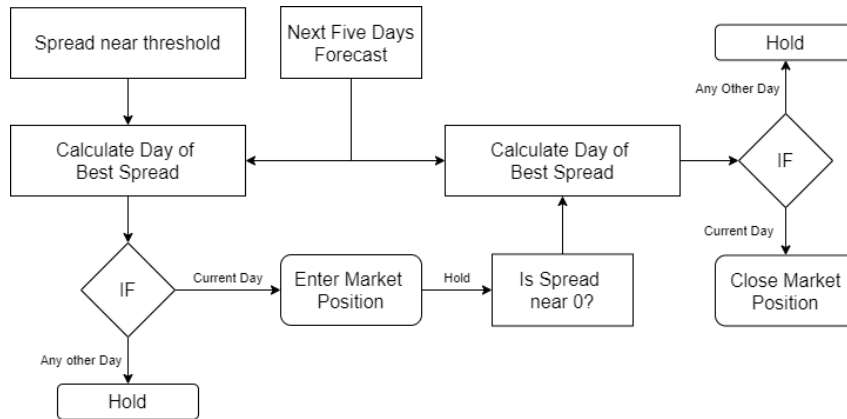


Figure 3.6: Trading model decision flowchart

As stated in Figure 3.6, the trading model is triggered whenever the spread is **near** the thresholds. This strategy aims to identify possibly good trading options before the thresholds are actually crossed. In order to achieve this, a margin was defined near both thresholds to open positions before the short and long thresholds, and also around zero to close positions earlier.

3.3.3 Stop Loss-Function

In Picture 3.7 it is shown a short position that was opened for a long period of time. The graph on the right indicates how the regular spread (a simple division of the pair's stock value) fluctuated during that period of time. On the left, it is depicted the exact same period of time and market position, but this time with the behaviour of the normalized spread. Remembering the fundamentals of Pairs Trading, profit is made from the convergence of the Spread to its historical average, hence why the normalized spread is used. However, if a short position is opened, the profit will be made if the regular spread decreases (and vice-versa for a long position).

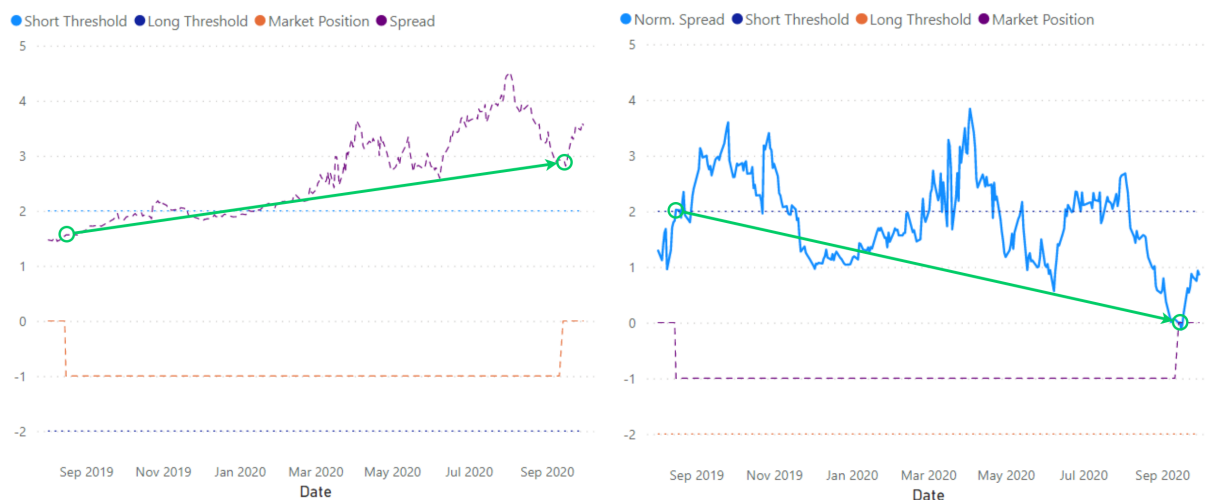


Figure 3.7: Behaviour comparison between the Spread and the Normalized Spread

In this particular case, a short position was opened on the 16th of August of 2019 as the normalized spread crossed the short threshold. However, the spread completely diverged from its historical mean and kept on growing and never returned. Since the premise for Pairs trading is that the pairs spread should always return to its historical average, this means that the investor is now losing money every day the spread keeps increasing. Even though the normalized spread eventually returns to zero (graph on the right), the investor lost a lot of money due to the regular spread increasing (graph on the left). In total, this position was opened for more than a year, and having a look at the formula 1.1 that calculates the normalized spread, we can see that the spread is normalized using the average and the standard deviation of the previous 200 days.

Given this information, to create a stop-loss function, there are two fundamental values that need to be looked at:

- Position "age": How many trading days has the position been opened for:
- Spread's deviation: How much has the spread increased or decreased

3.4 Project's Structure Overview

The overall architecture of the project is based on two main blocks: The first one is the creation block, which only uses data from a certain period (2000-2018) to select the pairs that will compose the portfolio as well as the models that best predict each stock's behaviour. The second block is the case study where every decision and model build from the data of the past, will be applied on the test period (2019-2020). During this period every bit of the enhanced pairs trading investment strategy will be performed as if it was in real-time. Also during the case study, the model shall be tested against the "regular" pairs trading strategy to assess its effectiveness.

4

Implementation

Contents

4.1 Pairs Selection	32
4.2 Forecasting Model	33
4.3 Trading Model	39
4.4 Power BI Platform	47
4.5 Computational Cost	50

4.1 Pairs Selection

Following the process described in Figure 3.1, and only using information from 2000 to 2018 the first step to do is to select the range of stocks. Out of the 100 stocks listed in the Nasdaq-100 index, we started by filtering out all those that didn't have at least 5000 trading days of information (leaving us with 69 stocks). With the remaining ones, for every possible pair it was calculated the cointegration (as explained by MacKinnon in [6, 7]), correlation (2.8) and the SSD (2.6). With this information, Tables 4.1 and 4.2 indicate how many of these pairs remain when the three thresholds are changed.

4.1.1 Filter Sensitivity

Out of the 2346^1 possible combinations, In Table 4.1 the amount of pairs that would comply with both the cointegration (rows) and correlation(columns) filters was calculated.

Table 4.1: Variation of the number of Pairs with different Correlation and Cointegration Values

		Correlation Value (c)								
		0.9	0.91	0.92	0.93	0.94	0.95	0.96	0.97	0.98
Cointegration Value (i)	0.01	62	56	53	51	45	41	31	19	5
	0.02	111	103	98	89	80	68	51	29	7
	0.03	142	135	125	112	102	87	61	34	9
	0.04	170	160	151	135	124	105	72	38	11
	0.05	190	179	169	152	140	117	77	38	11
	0.06	208	195	186	166	152	126	82	41	12
	0.07	221	206	194	176	159	131	84	42	13
	0.08	241	220	208	188	167	137	89	44	13
	0.09	256	235	222	200	177	145	93	47	13
	0.1	275	252	239	217	188	152	98	48	13

After combining Tables 4.1 and 4.2 and with the ultimate goal of having a small amount of pairs in the Portfolio, the chosen thresholds were $i = 0.03$, $c = 0.96$ and $s = 0.07$ leaving us with 47 pairs to enter the Historical Profit stage.

Table 4.2: Variation of the number of Pairs with different SSD measures

p-value	Corr. Value	SSD Measure (s)			
		0.08	0.07	0.06	0.05
0.02	0.95	51	39	29	13
0.03	0.96	61	47	34	16
0.04	0.96	72	54	38	19
0.05	0.96	77	56	38	19

${}^1_{69}C_2 = \frac{69!}{2!(69-2)!} = \frac{69!}{2! \times 67!} = 2346$

4.1.2 Historical Profit

For all 47 pairs that were selected in the previous section, the simple threshold-based model was run during the formation period (2000-2018) with different values for "look-back days".

Reducing the look-back period would increase the number of market openings, however, it would also reduce the profit of each transaction. Having it in mind, and after looking at the historical profits, the top pairs that had a more consistent profit with a satisfactory number of market positions per year were selected. In Table 4.3 it is presented the profit from a simulated Threshold-based trading model on the pairs that had the best performance with the different number of look-back days. In the simulation, ten thousand USD were invested for each stock each time a position was opened.

Table 4.3: Simulation Profit in USD of the Threshold-based trading model on top 10 pairs

Portfolio		# of Look Back Days					
		50	100	200	300	400	500
ADBE	MSFT	17419	17701	33932	27017	17999	14306
ADP	INTU	40440	32231	17866	15028	5054	10497
AMGN	CMCSA	28061	31469	19998	25175	20398	16863
HAS	PAYX	14861	19513	15810	19384	21200	13463
IDXX	MCHP	23688	23324	33903	36161	26213	25339
LRCX	MAR	50931	23348	18212	15847	24633	25192
Total Profit (USD)		175400	147586	139721	138613	115498	105660

4.2 Forecasting Model

The goal of this section is to create a forecasting model for each stock presented in table 4.3. As explained in 2.3.2, LSTM networks are well-suited to classifying, processing and making predictions based on time series data, hence why they were chosen for this project. Different combinations of input features were tested on the LSTM and its accuracy was accessed during the "test period" (2000 to 2018).

The model that presented the best accuracy was then saved and used during the "test period: (2019-2020).

4.2.1 Model Creation

To create the LSTM network, the "keras" library was used. As depicted in figure 4.1, the network was created with two LSTM layers of 50 neurons each, and then with a 25 neurons "Dense" layer. Then, the model is compiled with the "adam" optimizer and using the "mean squared error" as a loss function, that indicates how well the model is predicting. Lastly, the model is trained using as inputs the 'x_train'

and 'y_train' that are the vectors of the normalized input features and the normalized desired output, respectively, for each training day.

```
# Build LSTM model
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape=(x_train.shape[1], n_feat)))
model.add(LSTM(50, return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')

# train the model
model.fit(x_train, y_train, batch_size=1, epochs=epoch)
```

Figure 4.1: Code used to train the model

In figure 4.2 it is demonstrated the single line of code that takes on an array of input features and "transforms" it into an array of predictions that would need to be denormalized afterwards.

```
# get the models predicted price values
predictions = model.predict(x_test)
```

Figure 4.2: Code used to run the model

4.2.2 Model Accuracy Evaluation

It is important to understand the used measure to calculate the accuracy of any given model.

As explained earlier, it was only used data from 2000 to 2018. This period was divided into test and train data, where the first 90% of trading days were used to train each model and the last 10% to evaluate its performance. During the test period, the predicted closing price value was compared to the real closing value through the use of the Root mean Squared Error Function.

$$RMSE = \sqrt{\frac{\text{sum}(\text{predicted_value} - \text{real_value})^2}{\# \text{ trading days}}} \quad (4.1)$$

Since each prediction is made in USD, this value was then divided by the average closing price of the test period so that the evaluation of the performance of each model is made as a percentage.

$$RMSE\% = \frac{RMSE}{\text{average closing price}} \quad (4.2)$$

This adjustment allows us to better compare how each combination of input features behaves across the different stocks.

4.2.3 Feature Preparation

To train and test models, there is the need to collect and process data. As explained in picture 4.3, with the values collected in step 1, the input features are calculated. Afterwards, they are normalized to avoid big differences in the scale of the data. Lastly, for each model, different combinations of input features may be chosen.

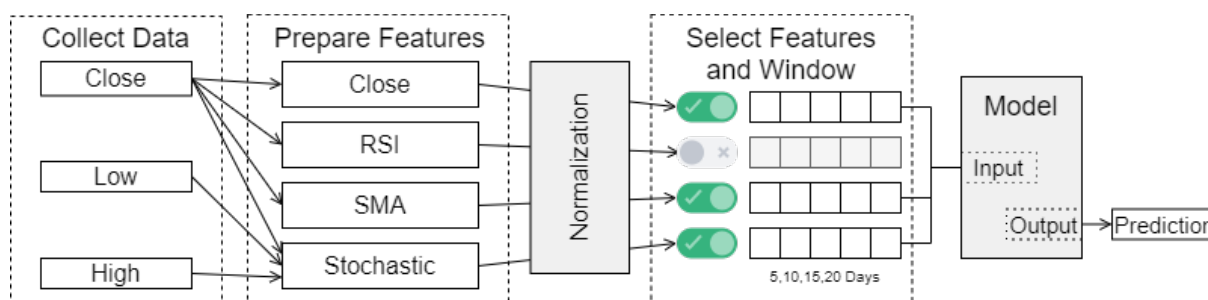


Figure 4.3: Feature Preparation Process

4.2.3.A Data Collection

Data collection, as the name suggests, is the process of collecting raw data that will be used in the following steps.

During this project, an API called Tiingo² was used to load the information of each stock into a 'python pickle file'. As the stock market only opens on work days, during our test period (from 2000 to 2018) there are more than 4000 days worth of data. In each day, for each stock it is loaded the following data:

- Open - value of the stock in the beginning of the day (in USD)
- High - highest value reached on the day (in USD)
- Low - lowest value reached on the day (in USD)
- Close - last value of the stock on that day (in USD)
- Adjusted Close - an amends to a stock's closing price to reflect that stock's value after accounting for any corporate actions
- Volume - Number of shares traded on that day (quantity)

4.2.3.B Feature Calculation

As input features, all three financial indicators explained in 2.1, will be used. Additionally, the closing price of both the stock itself and the respective stock pair will be used to create the models for each one of the stocks that make the portfolio.

²www.tiingo.com

Regarding the Simple Moving Average, as depicted in Figure 2.3, it is usually used by investors a "long" SMA (10 day SMA), and a "short" SMA (10 day SMA) to confirm market trends. To use this information as an input feature, the SMA-10 was subtracted by the SMA-50.

Both other financial indicators (the Stochastic Oscillator and the RSI), are calculated following the equations 2.2, 2.3, 2.4 and the resulting value is used directly into the Neural Network.

These financial indicators, may not be used all at the same time. In the following sections, it will be explained the selection process of the input features.

This process is made right after the data collection and three new arguments (RSI, SMA and Stochastic) are added to each day's information. This information is updated to the python pickle file to avoid repeating this process every time.

4.2.3.C Normalization

Following a procedure described in most works developed in this area, every input feature is normalized before entering the model. Data normalization is a set of techniques, usually mathematical formulas, that are applied especially when data comes in different scales.

One clear example of that is the closing price of the stocks we are dealing with in our portfolio. Some of them are way above 200 USD while others have never gone higher than 50 USD. For example, AMZN³ stock price is currently higher than 3000 USD.

There are several techniques, and in this project, the one used was the Min-Max Normalization that re-scales the range of data to [0,1],

$$\hat{X} = \frac{x - \min}{\max - \min} \quad (4.3)$$

where \hat{X} is the normalized value and x is the real value. Equation 4.3 was used separately for each input feature. Meaning that when normalizing the SMA, 'min' was the minimum SMA verified before we're calculating the same goes for 'max'.

This whole process is inverted to the model's output using the same formula (4.3).

4.2.3.D Feature Combination

To predict the following day, the model is trained using a combination of input features with its respective values for the previous days (further mentioned as "Feature Window"). Besides testing the accuracy of the model with different feature combinations, it was also tested different feature windows of 5, 10, 15 and 20 days.

³Stock Symbol for Amazon, biggest online retailer worldwide

For every combination tested, the closing price of the stock, was always used as input feature. Then, each one of the three financial indicators were combined with the closing price creating 8 different models.

Combining the three indicators that were prepared in 4.2.3.B with each stock's closing price⁴, eight different feature combinations are possible. In Table 4.4 it is depicted the different input combinations that were tested for each of the portfolio stocks.

Table 4.4: Input Feature Combinations tested to develop models

[Close]	[Close] [Stochastic]	[Close] [RSI]	[Close] [SMA]	[Close] [Stoch] [SMA]	[Close] [RSI] [SMA]	[Close] [Stochastic] [RSI]	[Close] [Stochastic] [RSI] [SMA]
---------	-------------------------	------------------	------------------	-----------------------------	---------------------------	----------------------------------	---

4.2.4 Single Model Approach

As a first approach, the intended result was to have the same model forecasting for every stock in our portfolio. For each stock, the combinations of input features presented in Table 4.4, were tested with feature windows of 5, 10, 15 and 20 days.

Every model was run with a single epoch due to the high computing time needed to perform all the tests. After running and evaluating every model, the table 4.5 was created with the average RMSE% of all stocks for each configuration.

Table 4.5: Average RMSE% for each feature window and feature combination

Window	[C]	[C,S]	[C,RSI]	[C,SMA]
5	2.29%	2.50%	1.97%	2.67%
10	1.70%	3.11%	2.83%	2.58%
15	1.45%	2.11%	3.04%	2.14%
20	1.14%	2.10%	2.36%	2.88%
Window	[C,S,SMA]	[C,RSI,SMA]	[C,S,RSI]	[C,S,RSI,SMA]
5	1.12%	1.21%	2.43%	1.93%
10	2.62%	3.47%	1.94%	1.61%
15	1.63%	1.90%	2.33%	3.14%
20	2.45%	2.12%	4.26%	2.15%

From Table 4.5 the configuration of input features [Close, Stochastic, SMA] with a feature window of 5 trading days, was selected as the best performing model on average for all stocks. This model was then used to evaluate the impact that the closing price of the respective pair would have when added as an input feature. Also, the simplest model (the one that only uses the closing price as an input feature) was used as a control to this experiment.

⁴to avoid sudden drops, due to corporate actions, we will always use the adjusted closing price

Table 4.6: Average RMSE% comparison when using the stock pair closing price

Window	[C]	[C,S]	[C,Pair]	[C,RSI]	[C,SMA]	[Close,S,SMA]	[Close,S,SMA,Pair]
5	2.29%	2.50%	2.13%	1.97%	2.67%	1.12%	3.32%
10	1.70%	3.11%	2.94%	2.83%	2.58%	2.62%	3.90%
15	1.45%	2.11%	2.56%	3.04%	2.14%	1.63%	3.96%
20	1.14%	2.10%	2.50%	2.36%	2.88%	2.45%	2.83%

As demonstrated through table 4.6, adding the pair to the input features didn't improve the accuracy of the model. It is clear to see that when using just two input features, there is always another model that performs better than the one using the pair's price. Also, when we had the pair's price to the best performing model yet, its RMSE% increases for every feature window tested. Due to these results, the pair's price will **no longer be tested** in further sections of the project.

Even though it would be possible to select a single model to forecast for every stock in our portfolio, it isn't feasible to do it with an RMSE% of 1.12%. This is because in the last 20 years, the stock market, as defined by the SP 500⁵, has moved on average between -1% and +1% per day on 70% of the trading days.

4.2.5 Multiple Model Approach

After testing every combination of input features, with all four possible feature windows, on a single epoch⁶. Having these models locked for each stock, the next step was to increase the number of epochs to try to achieve better accuracy. As such, each model was trained with 1, 2, 3, 4 and 5 epochs and the best-performing ones were the ones demonstrated in table 4.7.

Table 4.7: Best performing models for each stock

Stocks	Input Combo	Window	Epochs	RMSE %
ADBE	[Close,SMA]	20	3	0.56%
MSFT	[Close]	5	3	0.44%
ADP	[Close,SMA]	10	3	0.54%
INTU	[Close,Stoch]	15	3	0.04%
AMGN	[Close]	20	3	0.34%
CMCSA	[Close,RSI]	15	3	0.42%
HAS	[Close,Stoch,SMA]	5	3	0.13%
PAYX	[Close,RSI,SMA]	5	3	0.48%
IDXX	[Close,RSI]	20	5	0.03%
MCHP	[Close,Stoch,SMA]	20	4	0.16%
LRCX	[Close,Stoch]	15	5	0.36%
MAR	[Close,Stoch,SMA]	5	3	0.28%

As explained in 3.2.3 there is going to be one model for each forecasting day on every stock. To

⁵the SP500 is a market-capitalization-weighted index of the 500 largest U.S. publicly traded companies

⁶An epoch is a term used in machine learning and indicates the number of passes of the entire training dataset the machine learning algorithm has completed

keep things simple, in this project we will use the same input features and input window on all models of the same stock. To achieve this, during the testing phase, the model had as an expected output, not the following day but the day after, training it to forecast how much the closing price of that stock would cost after two trading days. The same process was applied to train and test models that would predict the third, fourth and fifth following days.

Table 4.8: Accuracy of models on each forecasting day

Stocks	Input Combo	Window	Day 1	Day 2	Day 3	Day 4	Day 5
			RMSE %				
ADBE	[Close,SMA]	20	0.56%	0.17%	0.40%	0.03%	0.23%
MSFT	[Close]	5	0.44%	0.50%	0.04%	0.21%	0.13%
ADP	[Close,SMA]	10	0.54%	0.16%	0.33%	0.03%	0.21%
INTU	[Close,Stoch]	15	0.04%	0.09%	0.27%	0.17%	0.46%
AMGN	[Close]	20	0.34%	0.13%	0.02%	0.25%	0.32%
CMCSA	[Close,RSI]	15	0.42%	0.00%	0.44%	0.06%	0.12%
HAS	[Close,Stoch,SMA]	5	0.13%	0.21%	0.24%	0.09%	0.30%
PAYX	[Close,RSI,SMA]	5	0.48%	0.20%	0.33%	0.33%	0.16%
IDXX	[Close,RSI]	20	0.03%	0.57%	0.02%	0.49%	0.45%
MCHP	[Close,Stoch,SMA]	20	0.16%	0.33%	0.16%	0.15%	0.47%
LRCX	[Close,Stoch]	15	0.36%	0.08%	0.16%	0.19%	0.18%
MAR	[Close,Stoch,SMA]	5	0.28%	0.18%	0.08%	0.26%	0.02%
Average			0.31%	0.22%	0.21%	0.19%	0.26%

In table 4.8 it is presented the results obtained during the testing phase for each stock's models when predicting the following five days. It is worth noticing that on average the RMSE% is **0.24%** which is a substantial improvement from the results presented in table 4.5.

4.3 Trading Model

As stated in Figure 3.1, this last step of the project was built using Microsoft's tool Power BI due to its capabilities of displaying data in a very clear and simple way. This section will be first demonstrated how the regular threshold model was created in Power BI. This model will be compared side by side with the proposed model during the test period of 2019 and 2020.

It is important to understand that in Power BI, all data is stored in tables. Similarly to Excel tables, these are composed of rows and columns and can have formulas to calculate values based on other cells and tables. One big difference from Excel is that a column can only have one formula, written in DAX (Data Analysis Expressions), that will define all its cells. Besides formulas, tables can be imported from API's, python scripts or Excel files etc. Between columns of different tables, there can be relations just like in a relational database (one-to-one, one-to-many, many-to-many).

Lastly, it is important to understand that in Power BI, there are two main ways of creating new columns. The first is through Power Query, the main data transformation engine in Power BI. The second

way is through DAX (Data Analysis Expressions) that resembles a coding language, similar to Excel's, that comprises a library of functions and data operations that when combined can create functions that define columns.

4.3.1 Data Structure

Before writing DAX code and creating models, some tables need to be imported from other sources. Firstly, the Tiingo API was used to import the adjusted close prices of the twelve stocks that compose the portfolio (chosen in 4.1) from the year of 2018⁷ to 2020, this table is called "Real Values" (Table 4.9).

Table 4.9: Structure of table "Real Values"

Real Values							
Date	Open	High	Low	Close	Adj Close	Volume	Stock
1/2/2018	175.85	177.8	175.26	177.7	177.7	2432800	ADBE
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Then the table "5Days Forecasting" (Table 4.10) was generated through a python script, using the models created in 4.2 were used to forecast the following five days of every day of the test period.

Table 4.10: Structure of table "5Days Forecasting"

5Days Forecasting			
Date	Predictions	Stock	Predicted Day
27/4/2018	158.0	AMGN	1
27/4/2018	157.7	AMGN	2
⋮	⋮	⋮	⋮

Finally, a small excel table called "Pairs Table" (Table 4.11) was imported with the name of the stocks of the portfolio and the name of the pair⁸ they belong.

Table 4.11: Structure of table "Pairs Table"

Pairs Table	
Pair Name	Stock
ADBE-MSFT	ADBE
⋮	⋮

The last table used on the project is called "Models" (Table 4.12) and to its base structure, will be added columns for each step of the creation of the models, which will be explained in the following sections.

⁷2018 data is only used to calculate all the indicators that need previous days to calculate its values

⁸the name of the pair is always the concatenation of the two stocks, however, this step makes it easier to create some graphs and filters in Power BI

Table 4.12: Base Structure of table "Models"

Models					
Date	Pair	Spread	Average	Std Dev	Norm. Spread
27/01/2019	IDXX-MCHP	2.477	2.768	0.343	-0.845
⋮	⋮	⋮	⋮	⋮	⋮

There is a many-to-many relationship between the 'Date' columns in "Real Values", "5Days Forecasting" and "Norm Spreads". Also, there is a one-to-many relationship between the 'Stock' column in the "Pairs Table" with "5Days Forecasting" and "Real Values".

4.3.2 Threshold Based Model

To create the simple threshold-based model, four columns were added to the "Models" table, one for each threshold ('Long Threshold' with a constant value of '-2' and a 'Short Threshold' with a constant value of '2'). The third auxiliary column is an index that indicates the number of each row (after being sorted by pair and afterwards by date).

4.3.2.A Market Movement Calculation

The first step towards calculating the market position is to understand where the 'Norm. Spread' is. Using the Power Query Editor, the column 'Area' was created as demonstrated in Figure 4.4. If the 'Norm. Spread' is between 'Long Threshold' and 'Short Threshold', the column 'Area' will return a blank.

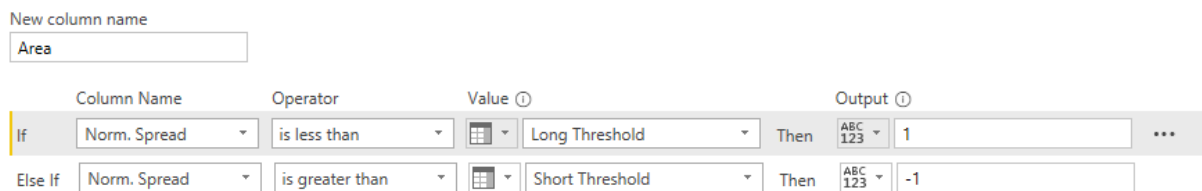


Figure 4.4: Creation of column 'Area' in Power Query Editor

The fact that the previous query returns blank is key for this next step that is to understand the Market Position. For that, the column 'Market Position' was created using the Power Query functionality to "Fill Down". This means that if a certain cell is blank, it will take the value of the cell above that was not blank.

To get the actual market movement, the column 'Market Movement' was created through the use of DAX syntax, as shown in picture 4.5.

```

1 Market Movement =
2 IF('Models'[Date].[Year]=2018 ), BLANK(),
3 IF(
4     AND(CALCULATE(
5         MAX('Models'[Market Position]),
6         FILTER('Models',
7             'Models'[Index] = EARLIER('Models'[Index]) - 1) // Selects the Market Position of the previous indexed row
8     ) < 'Models'[Market Position], 'Models'[Market Position]=1), "Open Long",
9
10    IF(
11        AND(CALCULATE(
12            MAX('Models'[Market Position]),
13            FILTER('Models',
14                'Models'[Index] = EARLIER('Models'[Index]) - 1) // Selects the Market Position of the previous indexed row
15        ) > 'Models'[Market Position], 'Models'[Market Position]=-1 ), "Open Short",
16
17        IF(
18            AND(CALCULATE(
19                MAX('Models'[Market Position]),
20                FILTER('Models',
21                    'Models'[Index] = EARLIER('Models'[Index]) - 1) // Selects the Market Position of the previous indexed row
22            ) > 'Models'[Market Position], 'Models'[Market Position]=0 ), "Close Long",
23
24            IF(
25                AND(CALCULATE(
26                    MAX('Models'[Market Position]),
27                    FILTER('Models',
28                        'Models'[Index] = EARLIER('Models'[Index]) - 1) // Selects the Market Position of the previous indexed row
29                ) < 'Models'[Market Position], 'Models'[Market Position]=0 ), "Close Short", ""
30            )
31        ))))

```

Figure 4.5: Creation of column 'Market Movement' using DAX

4.3.2.B Profit Calculation

To calculate the profit, it was followed the process described in 1.1. First, the column 'Pair' was split, into '1st.Stock' and '2nd.stock' using Power Query. For example, if the 'Pair' had the value, IDXX-MCHP, '1st.Stock' would be IDXX and '2nd.Stock' MCHP. Having these two columns created, the next two columns are created with a single line of DAX code. Column 'Price1' and 'Price2' indicate the adjusted closing price of stocks 1 and 2 respectively and are created as explained in figure 4.6:

```

1 Price1 = LOOKUPVALUE('Real Values'[Adj Close], 'Real Values'[Stock], 'Models'[1st.Stock])
1 Price2 = LOOKUPVALUE('Real Values'[Adj Close], 'Real Values'[Stock], 'Models'[2nd.Stock])

```

Figure 4.6: Creation of columns 'Price1' and 'Price2' using DAX

The following step is to know how many stocks can be purchased giving the available investment⁹. This process is quite simple and is demonstrated in picture 4.7. The same formula is used to compute 'Shares_Stock2'.

⁹For this case study, it was invested an amount of 10k USD per share

```

1 #Shares_Stock1 =
2 IF( ('Models'[Date].[Year]=2018 ), BLANK(),
3 IF( 'Models'[Market Movement] = "Open Long", (10000 / 'Models'[Price1] ),
4
5     IF( 'Models'[Market Movement] = "Open Short", (10000 / 'Models'[Price1] ) * -1 ,
6
7         IF( 'Models'[Market Movement] = "Close Long", (10000 / 'Models'[Price1] ) * -1,
8
9             IF( 'Models'[Market Movement] = "Close Short", (10000 / 'Models'[Price1] ), BLANK()
10
11 ))))

```

Figure 4.7: Creation of column 'Shares_Stock2' using DAX

Knowing the number of shares that are bought or sold at any given market movement, is enough information to calculate the profitability of each transaction. In picture 4.8 it is shown the DAX formula that calculates the profit whenever a certain position is closed. It is worth noticing that for this case study, every transaction is simulated with an investment of 10000 USD.

```

1 Profit =
2 // calculate the index of the last "Close Short" or "Close Long" of that pair
3 VAR lastCloseIndex = CALCULATE( LASTNONBLANK('Models'[Index],TRUE()),
4     FILTER( 'Models', OR('Models'[Market Movement] = "Close Long", 'Models'[Market Movement] = "Close Short") &&
5         'Models'[1st.Stock] = EARLIER('Models'[1st.Stock]) && 'Models'[Index] < EARLIER('Models'[Index]) ) )
6
7 //This sequence repeats four times, and gets the first non blank value of the desired column that has an index
8 //between the lastCloseIndex variable and the index of the row being calculated
9 VAR stock1 = CALCULATE( FIRSTNONBLANK('Models'[#Shares_Stock1],TRUE()),
10     FILTER( 'Models', 'Models'[#Shares_Stock1]<>0 && 'Models'[1st.Stock] = EARLIER('Models'[1st.Stock]) &&
11         'Models'[Index] < EARLIER('Models'[Index]) && 'Models'[Index]>lastCloseIndex ) )
12 VAR stock2 = CALCULATE( FIRSTNONBLANK('Models'[#Shares_Stock2],TRUE()),
13     FILTER( 'Models', 'Models'[#Shares_Stock1]<>0 && 'Models'[1st.Stock] = EARLIER('Models'[1st.Stock]) &&
14         'Models'[Index] < EARLIER('Models'[Index]) && 'Models'[Index]>lastCloseIndex ) )
15 VAR price1 = CALCULATE( FIRSTNONBLANK('Models'[Price1],TRUE()),
16     FILTER( 'Models', 'Models'[#Shares_Stock1]<>0 && 'Models'[1st.Stock] = EARLIER('Models'[1st.Stock]) &&
17         'Models'[Index] < EARLIER('Models'[Index]) && 'Models'[Index]>lastCloseIndex ) )
18 VAR price2 = CALCULATE( FIRSTNONBLANK('Models'[Price2],TRUE()),
19     FILTER( 'Models', 'Models'[#Shares_Stock1]<>0 && 'Models'[1st.Stock] = EARLIER('Models'[1st.Stock]) &&
20         'Models'[Index] < EARLIER('Models'[Index]) && 'Models'[Index]>lastCloseIndex ) )
21
22
23 VAR totalLong = ('Models'[Price1] - price1) * stock1 + (price2 - 'Models'[Price2]) * -stock2 //Profit Calculation for a Long Position
24 VAR totalShort = ('Models'[Price2] - price2) * stock2 + (price1 - 'Models'[Price1]) * -stock1 //Profit Calculation for a Short Position
25 RETURN
26 IF( 'Models'[Market Movement] == "Close Long",
27     totalLong,
28     IF( 'Models'[Market Movement] == "Close Short",
29         totalShort, BLANK()
30 ))

```

Figure 4.8: Creation of column 'Profit' using DAX

4.3.2.C Visual Presentation in Power BI

After completing the steps above, the "Models" table had more than 3000 rows and 21 columns. It is impossible to understand how the model performed or to get any insightful detail. Power BI, besides all the data processing capabilities that it offers, is an excellent tool to get clean and custom made

visualisations of our data. In figure 4.9 it is represented the threshold-based model's performance for all six pairs in the selected portfolio. For each pair, it is presented the normalized spread, both long and short thresholds and in a purple dashed line the market position¹⁰ taken throughout the period of 2019 and 2020.



Figure 4.9: Application of the Threshold based model in Power BI

Also, for all six pairs, it is presented what would have been the overall profit of implementing the threshold-based model during this period. out of the six, two of the pairs would induce a big loss for the investor. All transactions will be further analysed in section 5.

4.3.3 Enhanced Model

The proposed model will use some of the data that has already been processed in the calculation of the threshold-based model. Following the architecture depicted in figure 3.6, both decisions that need to be made (open and close a market position) have as a key indicator, the day of best spread.

4.3.3.A Minimum and Maximum Spread Days

Using the information from the table "5Days Forecasting", it is possible to calculate the spread of each one of the following 5 days. Note that in this formula it is being used the spread and not the normalized spread, given that it is a much more trustworthy indicator to the investor as depicted in picture 3.7. In picture 4.10 it is shown the DAX function that returns the day of minimum spread. A very similar function

¹⁰the market position line can only take three values: 0 (no position held), 1 (long position) and -1 (short position)

was created to return the day of maximum spread. These two values will be of huge importance when selecting the best day to open/close a position.

```

1 minSpread_ID =
2 VAR day1Spread = LOOKUPVALUE('5Days Forecasting'[Predictions],'5Days Forecasting'[Date], 'Models'[Date],
3   '5Days Forecasting'[Predicted Day],1,'5Days Forecasting'[Stock],'Models'[1st.Stock]) / LOOKUPVALUE('5Days Forecasting'[Predictions],
4   '5Days Forecasting'[Date], 'Models'[Date], '5Days Forecasting'[Predicted Day],1,'5Days Forecasting'[Stock],'Models'[2nd.Stock])
5 VAR day2Spread = LOOKUPVALUE('5Days Forecasting'[Predictions],'5Days Forecasting'[Date], 'Models'[Date],
6   '5Days Forecasting'[Predicted Day],2,'5Days Forecasting'[Stock],'Models'[1st.Stock]) / LOOKUPVALUE('5Days Forecasting'[Predictions],
7   '5Days Forecasting'[Date], 'Models'[Date], '5Days Forecasting'[Predicted Day],2,'5Days Forecasting'[Stock],'Models'[2nd.Stock])
8 VAR day3Spread = LOOKUPVALUE('5Days Forecasting'[Predictions],'5Days Forecasting'[Date], 'Models'[Date],
9   '5Days Forecasting'[Predicted Day],3,'5Days Forecasting'[Stock],'Models'[1st.Stock]) / LOOKUPVALUE('5Days Forecasting'[Predictions],
10  '5Days Forecasting'[Date], 'Models'[Date], '5Days Forecasting'[Predicted Day],3,'5Days Forecasting'[Stock],'Models'[2nd.Stock])
11 VAR day4Spread = LOOKUPVALUE('5Days Forecasting'[Predictions],'5Days Forecasting'[Date], 'Models'[Date],
12  '5Days Forecasting'[Predicted Day],4,'5Days Forecasting'[Stock],'Models'[1st.Stock]) / LOOKUPVALUE('5Days Forecasting'[Predictions],
13  '5Days Forecasting'[Date], 'Models'[Date], '5Days Forecasting'[Predicted Day],4,'5Days Forecasting'[Stock],'Models'[2nd.Stock])
14 VAR day5Spread = LOOKUPVALUE('5Days Forecasting'[Predictions],'5Days Forecasting'[Date], 'Models'[Date],
15  '5Days Forecasting'[Predicted Day],5,'5Days Forecasting'[Stock],'Models'[1st.Stock]) / LOOKUPVALUE('5Days Forecasting'[Predictions],
16  '5Days Forecasting'[Date], 'Models'[Date], '5Days Forecasting'[Predicted Day],5,'5Days Forecasting'[Stock],'Models'[2nd.Stock])
17 VAR todaySpread = 'Models'[Spread]
18 //Returns the minimum spread of the current and following 5 days
19 VAR minSpread = MIN(MIN(MIN(day1Spread,day2Spread),MIN(day3Spread,day4Spread)),MIN(day5Spread,todaySpread))
20 VAR minID = IF(minSpread=todaySpread,0,
21   IF(minSpread=day1Spread,1,
22     IF(minSpread=day2Spread,2,
23       IF(minSpread=day3Spread,3,
24         IF(minSpread=day4Spread,4,
25           IF(minSpread=day5Spread,5, BLANK()))))))
26
27 RETURN minID //returns the day of minimum spread

```

Figure 4.10: Creation of column 'minSpread.ID' using DAX

4.3.3.B New Market Position

There are five main steps when deciding when to open or close a certain market position. The first one presented in picture 4.11 is the stop-loss function which background is explained in 3.3.3. Here, if the spread is bigger (in absolute value) than 2 (value of both thresholds), with a divergence of the spread of over 75% and the position has been opened for over 100 days, it will automatically close avoiding further losses.

A second decision is made whenever the 'Norm. Spread' reaches near the threshold. For example, if it reaches near the 'Long Threshold' (using -1.95 as an example) if the 'minSpread_ID' has a value of 0, it means that the forecasting is predicting that the spread will revert to zero, indicating that a long position should be opened. A similar process is used to open short positions (line 20)

As the 'Norm. Spread' reverses to zero and crosses the "riskMargin" variable, and using the same logic regarding the 'minSpread_ID' and 'maxSpread_ID', the position may be closed. This clause aims to close the position whenever the 'Norm. Spread' is closest to zero, even though it may never cross it.

In the eventuality that the 'Norm. Spread' crosses the zero line, the position will be closed. In the same manner, if there isn't a position opened when the 'Norm. Spread' reverts back to zero and crosses any of the long or short thresholds, the position is opened to avoid losing out on an opportunity. This last case can happen if the forecasting model believes that the 'Spread' will keep diverging when in fact

it converges.

```

1 New Market Position =
2 // calculate the index of the last "Open Short" or "Open Long" of that pair
3 VAR lastOpenIndex = CALCULATE( LASTNONBLANK('Models'[Index],TRUE()),
4   FILTER( 'Models', OR('Models'[Market Movement] = "Open Long", 'Models'[Market Movement] = "Open Short") &&
5     'Models'[1st.Stock] = EARLIER('Models'[1st.Stock]) && 'Models'[Index] < EARLIER('Models'[Index]) ))
6 VAR margin = 0.15
7 VAR riskMargin = 0.15
8 VAR stoplossdays = 100
9 VAR stoploss = 1.5
10 return
11 // 0 = Close Position | 1 = Open Long | -1 = Open Short | -5 = Hold
12 IF('Models'[Date].[Year]=2018, 0, // Ignore every 2018 column (not part of case study)
13
14 //Stop Loss Function
15 IF( ABS('Models'[Norm. Spread] ) > 2 &&
16   MAX('Models'[Spread], 'Models'[Average])/ MIN('Models'[Spread], 'Models'[Average]) > 1.75
17   && 'Models'[Index]-lastOpenIndex > stoplossdays, 0 ,
18
19 //Margin to capture possible opening opportunities that don't cross the threshold
20 IF('Models'[Norm. Spread]>'Models'[Short Threshold]-margin && 'Models'[maxSpread_ID] = 0, -1,
21 IF('Models'[Norm. Spread]<'Models'[Long Threshold]+margin && 'Models'[minSpread_ID] = 0, 1,
22
23 //Margin to capture possible closing opportunities that don't cross the 0 line
24 IF('Models'[Previous Norm Spread]>riskMargin && 'Models'[Norm. Spread]<riskMargin && 'Models'[minSpread_ID] = 0, 0,
25 IF('Models'[Previous Norm Spread]<-riskMargin && 'Models'[Norm. Spread]>-riskMargin && 'Models'[maxSpread_ID] = 0, 0,
26
27 //Hard Close, as it crosses 0
28 IF( 'Models'[Previous Norm Spread] < 0.0 && 'Models'[Norm. Spread] > 0.0 , 0,
29 IF( 'Models'[Previous Norm Spread] > -0.0 && 'Models'[Norm. Spread] < -0.0, 0,
30
31 //Hard Open, as it converges back inside the thresholds
32 IF('Models'[Previous Norm Spread]>'Models'[Short Threshold] && 'Models'[Norm. Spread]<'Models'[Short Threshold], -1,
33 IF('Models'[Previous Norm Spread]<'Models'[Long Threshold] && 'Models'[Norm. Spread]>'Models'[Long Threshold], 1,
34
35 //If it doesn't enter any of the previous clauses, it holds position
36 -5
37
38 ))))))))

```

Figure 4.11: Creation of column 'New Market Position' using DAX

4.3.3.C Profit Calculation

Column 'New Market Position' indicates the action that needs to be taken, however, it is still needed to do the equivalent of the Power Query functionality to "Fill Down" used earlier in 4.3.2.A. Unfortunately, that functionality can only be used for columns created with Power Query, so in this case, an extra column needed to be created as explained in figure 4.12.

```

1 New Market Position Hold =
2 //get last index with an action different than old (-5)
3 VAR lastIndex = CALCULATE( LASTNONBLANK('Models'[Index],TRUE()),
4   FILTER( 'Models', 'Models'[New Market Position] <> -5 && 'Models'[1st.Stock] = EARLIER('Models'[1st.Stock])
5     && 'Models'[Index] <= EARLIER('Models'[Index]) ))
6 //get the action from 'Models'[New Market Position] and index=lastindex
7 VAR retValue = LOOKUPVALUE('Models'[New Market Position], 'Models'[Index], lastIndex)
8 RETURN IF(retValue = BLANK(), 0, retValue)

```

Figure 4.12: Creation of column 'New Market Position Hold' using DAX

The next step is to select the beginning and end of each transaction, and for that 'New Market movement' column was created, with the exact same syntax as the one presented in picture 4.5. From now on, the profit calculation follows the same steps like the ones presented in 4.3.2.B.

4.3.3.D Visual Presentation in Power BI

Similarly with the threshold-based model, for this new one, was created a dashboard (figure 4.13) to track the model's behaviour throughout the test period. The profit is shown in green indicates that it is better than the profit achieved in the previous model. In section 5, it will be made a direct comparison between all transactions to understand better where the new enhanced model outperformed the threshold-based one.



Figure 4.13: Application of the Enhanced Model in Power BI

4.4 Power BI Platform

As stated in Figure 3.1, Power BI was used on the later stage of the project to better analyze and present data.

4.4.1 Model Behaviour Dashboard

In order to create the dashboard presented in Figure 4.9, for each pair, a set of three Visualizations were put together. Two simple cards were created, one with the pair's name and the other with the

resulting profit of that pair's transactions. The latter was created by selecting the filters presented in Figure 4.14. On the filters tab, there are two options: the first indicates which filters are applied to the visual (in this case, since the goal is to calculate the profit of the ADBE-MSFT pair, the filter selects only the transactions that have ADBE as its first stock). The second option is to create a filter that affects the whole dashboard, which in this case it selects only dates after the 1st of January of 2019.

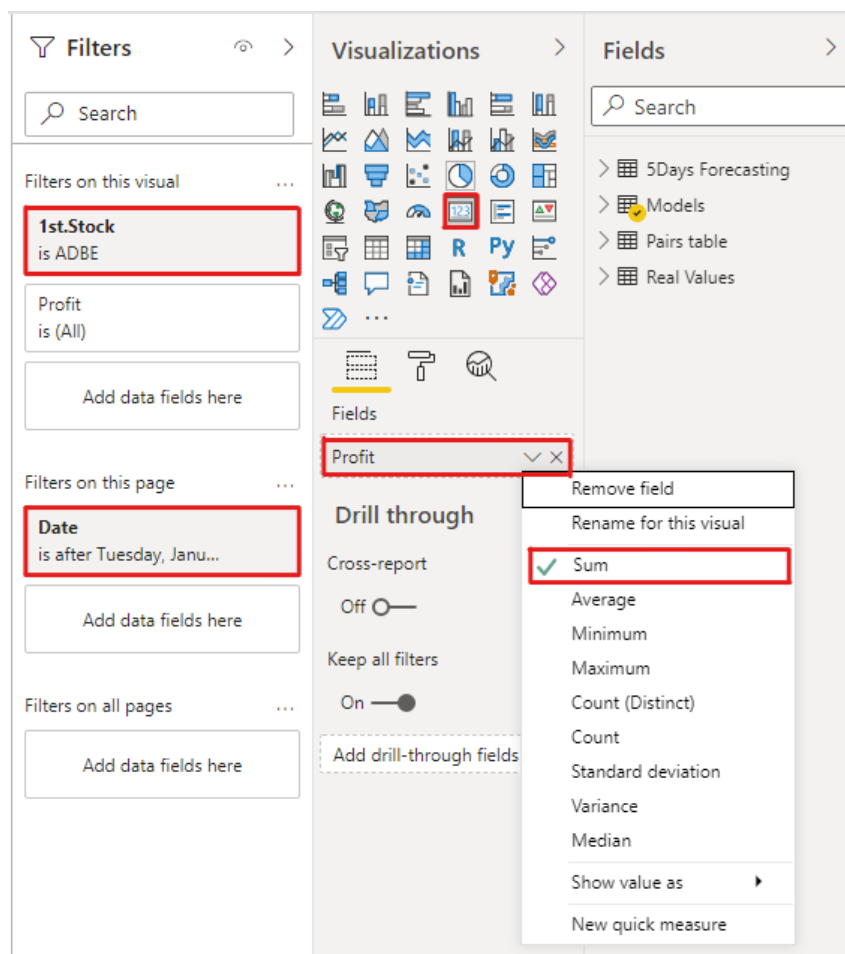


Figure 4.14: Power BI card's creation process

Adding to these two cards, a line chart was created. Even though the filters tab is exactly the same as the cards, on the middle tab, the fields chosen are a bit more complex as depicted in Figure 4.15.



Figure 4.15: Power BI graphs fields selection

The dashboard presented in Figure 4.13 follows the same process but swapping the 'Market Position' field by the 'New Market Position Hold' column. Lastly, instead of having a simple card indicating the profitability of the model, in this second dashboard, a KPI visualization was used, with the 'New profit' as the Indicator, and 'Profit' as the Target. This way, if the enhanced model outperforms the simple one, its profit will appear in green.

4.4.2 Transactions Dashboard

Aiming to have a deeper look at each transaction, a more complex and detailed dashboard was created (Figure 4.16). In the top right corner, a new object in Power BI is introduced, these are called slicers and allow to filter data on the dashboard. The first slicer selects the pair to be analyzed, and the second one, allows the user to select a period to "Zoom in". The main graph of the dashboard is created following the process described in Figure 4.15 but using both models market position and also adding the Spread.

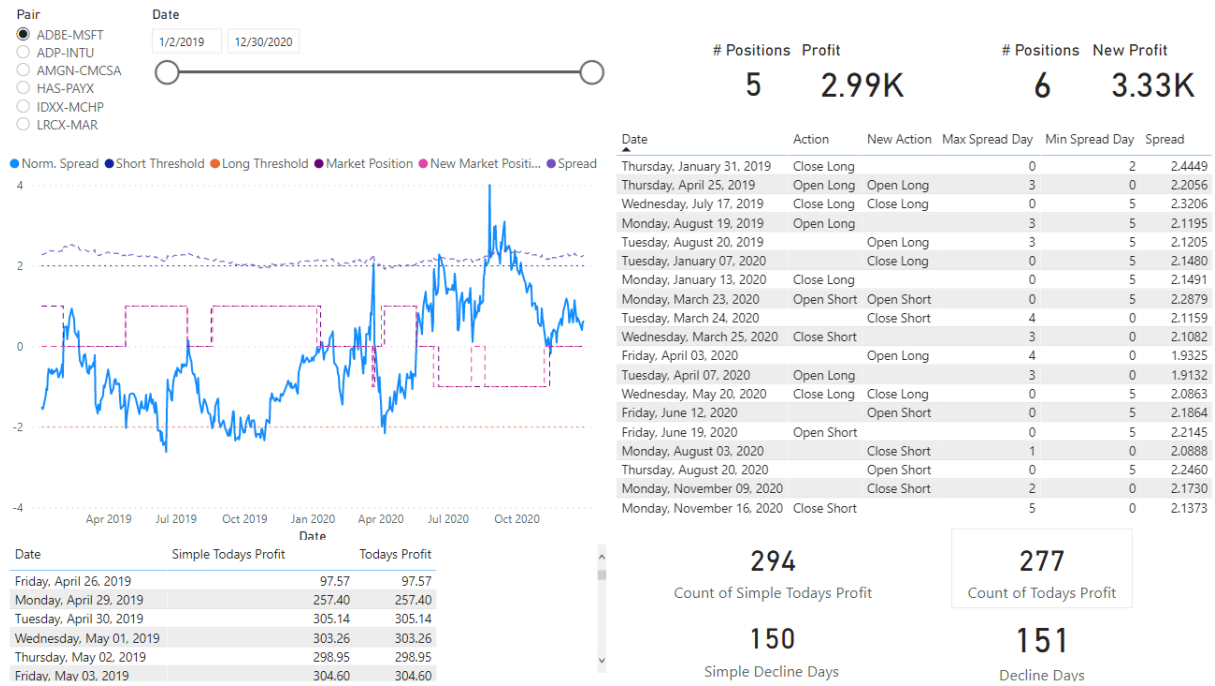


Figure 4.16: Power BI transactions dashboard

The small table below the graph indicates the daily profit evolution on both models and is useful to calculate the portfolio decline days (4 cards on the bottom right corner indicating how many days had a position opened as well as the portfolio decline days).

The table on the top right corner indicates the actions taken by both models, as well as the forecasted min and max spread days. This dashboard will be the main tool to compare both models in the next chapter.

4.5 Computational Cost

As this project had three main parts, they were all built as separate entities that run in sequence using the output file generated by the previous part. During this stage of the project, it will be assessed the computational cost needed to run all the algorithms.

For reference, all the computing was performed on a desktop with the following specifications:

Table 4.13: Machine configuration where all tests were run

Component	Specification
Processor:	Intel(R) Core(TM) i5-4690K CPU @ 3.50GHz
Operating System:	Windows 10
System Type	64-bit operating system, x64-based processor
Memory (RAM):	8.00 GB
Video/Graphics:	NVIDIA GeForce GTX 760

4.5.1 Pairs Selection

Regarding the pairs selection process, as stated in 4.1, 69 stocks were combined in pairs making a total of 2346 pairs. For each pair, 5000 days were compared using three different formulas (Correlation, Cointegration and the SSD measure). The python script writes on an excel spreadsheet the results of each calculation for all the possible pairs. To generate this file takes around one hour of computing power.

Table 4.14: Table generated by the python script with the cointegration values for all pairs

	AAPL	ADBE	...	XEL
ADBE	0.825	-	...	-
ADI	0.064	0.298	...	-
⋮	⋮	⋮	⋮	⋮
XLNX	0.073	0.119	...	0.071

After having all three tables, it is almost instantaneous, with a couple of Excel formulas, to have a list of pairs that comply with the selected filters. After having, in this case, the 47 pairs selected (Table 4.2), the historical profit calculation, is a simple process that took around one minute per pair to calculate.

4.5.2 Forecasting Model

From the previous step, a total of 12 stocks were selected and further used on this stage of the project. As explained in Figure 4.3, for all 12 stocks, a process of data collection and feature preparation is done before doing the actual model training. This initial process was only done once and took around a minute and a half to calculate and save the data of each stock during the test period (around 5000 days).

As stated in 4.2.3.D and 4.2.5 8 different models were tested for all 4 possible feature windows on the 12 pairs of the portfolio, which add up to a total of 384 models trained. Even though it depended on the number of input features, as well as the feature window's size, on average, per epoch the model would average 20 seconds to be trained.

5

Validation

Contents

5.1 Pairs Selection	55
5.2 Model Behaviour	56

This chapter aims to assess the quality of the results obtained during the previous section.

As this project had three main parts, each one will be individually assessed to validate some of the key decisions that were made during the implementation phase.

5.1 Pairs Selection

Regarding the pair's selection, the aim of the used process described in 3.1 was to select a short number of pairs that would allow the implementation of a pairs trading strategy. It is worth mentioning that during the test period, a worldwide pandemic crisis (Covid-19) started and it had a huge influence on the behaviour of the stock market. Due to this, the results will be evaluated in two periods of time: "Pre-Covid" (from Jan-2019 to Feb-2020, all-inclusive) and "Covid" (From Mar-2020 to Dec-2020 all-inclusive).

The following tables will compare the Standard Deviation of the spread with the Simple Threshold-Based model Simple Threshold Based Model (STBM).

Table 5.1: Pre-Covid Standard Deviation of Spread during Test Period

Pair Name	Standard Deviation	STBM Profit (USD)
ADBE-MSFT	0.14	717.73
ADP-INTU	0.02	2120
AMGN-CMCSA	0.48	-30.55
HAS-PAYX	0.12	1740
IDXX-MCHP	0.29	1910
LRCX-MAR	0.09	1470

Table 5.2: Covid Standard Deviation of Spread during Test Period

Pair Name	Standard Deviation	STBM Profit (USD)
ADBE-MSFT	0.11	2990
ADP-INTU	0.07	1280
AMGN-CMCSA	0.68	-842
HAS-PAYX	0.15	2310
IDXX-MCHP	0.42	9450
LRCX-MAR	0.93	-5490

In table 5.1 it is clear to see that in general, all pairs behaved as expected resulting in profit when applying the STBM. During the pandemic situation, the standard deviation generally increased as is expected, and in two of the pairs, this instability resulted in huge losses for the investor. During section 5.2 a closer look at the transactions will be made and the effect of the Covid-19 in the performance of this project will be once again assessed.

5.2 Model Behaviour

As briefly demonstrated in Figure 4.13 the overall results of the enhanced model were great. However, during this section, a deeper evaluation of the model's performance will be done.

5.2.1 Portfolio Decline Days

The first result to be evaluated is the percentage of Portfolio Decline Days, which means the number of days when the return on investment was negative.

Table 5.3: Portfolio Decline Days when using the STBM

Time Period	Simple Threshold Based Model		
	# of Opened Position Days	# of Decline Days	% of Decline Days
Pre-Covid	923	503	54%
Covid	840	553	65%
TOTAL	1763	1056	60%

Table 5.4: Portfolio Decline Days when using the Enhanced Model

Time Period	Enhanced Model		
	# of Opened Position Days	# of Decline Days	% of Decline Days
Pre-Covid	937	527	56%
Covid	772	449	58%
TOTAL	1709	976	57%

As expected, during the Covid period, the percentage of portfolio decline days increased when comparing to the Pre-Covid period. Also, and even though the improvement wasn't that big, with the enhanced model it was possible to decline a bit the amount of portfolio decline days.

5.2.2 Overall Profitability

Resuming Figures 4.9 and 4.13 in Table 5.5 it is clear to see that the profitability increased in all 6 pairs when applying the enhanced model.

Table 5.5: Profit comparison between both models

Pair Name	Simple Threshold Based Model	Enhanced Model
ADBE-MSFT	2990 \$	3330 \$
ADP-INTU	1280 \$	1590 \$
AMGN-CMCSA	-842 \$	-319 \$
HAS-PAYX	2310 \$	2650 \$
IDXX-MCHP	9450 \$	11080 \$
LRCX-MAR	-5490 \$	-2890 \$

To understand better this increase in profitability, we'll have a closer look at the circumstances that led to better transactions. After reviewing all transactions made in all six pairs, these were divided into 3 main categories.

5.2.2.A New Opportunities

The main difference between the behaviour of both models was the amount of opened positions. As stated in table 5.6, the enhanced model manages to "find" new opportunities to enter the market that in the end increases the profitability by a lot.

Table 5.6: Number of positions opened comparison between both models

Pair Name	Simple Threshold Based Model	Enhanced Model
ADBE-MSFT	5	6
ADP-INTU	4	5
AMGN-CMCSA	2	2
HAS-PAYX	2	3
IDXX-MCHP	4	5
LRCX-MAR	2	3

The first example occurs in the ADBE-MSFT pair. In figure 5.1 it is depicted how the normalized spread behaved, and it is interesting to analyse that there was a downwards spike (around the 3rd of August) where the normalized spread almost reaches 0. This would have been a great time to close this position, however, the simple model didn't do it. Due to the margins added to the enhanced model (explained in Figure 4.11) it managed to analyse the predictions for the following days and decided to close the position.

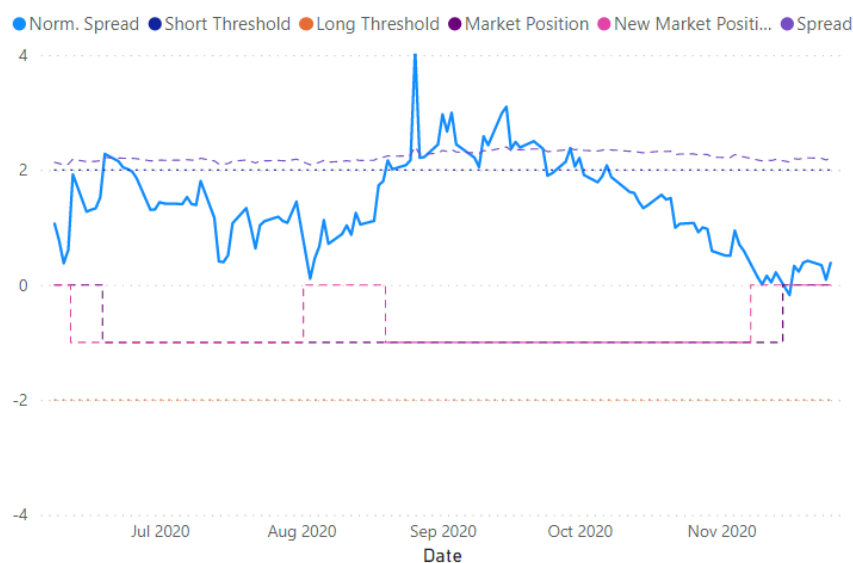


Figure 5.1: New market position opened in ADBE-MSFT pair

In this particular case, the predictions for the days following the 3rd of August indicated that there were no following days with a lower spread, which means that that day would be the best one to close the position. Also, and a couple of days later, the normalized spread had an upwards peak that led to another entering opportunity, allowing the enhanced model to have two different market positions while the simple model had only one. During this period, the enhanced model had more than twice as much profit as the simple one.

Under different circumstances, but following the same principle, in figure 5.2 it is depicted another case where the margins around the thresholds manage to capture a day where according to the forecast, was the best one to open a position.

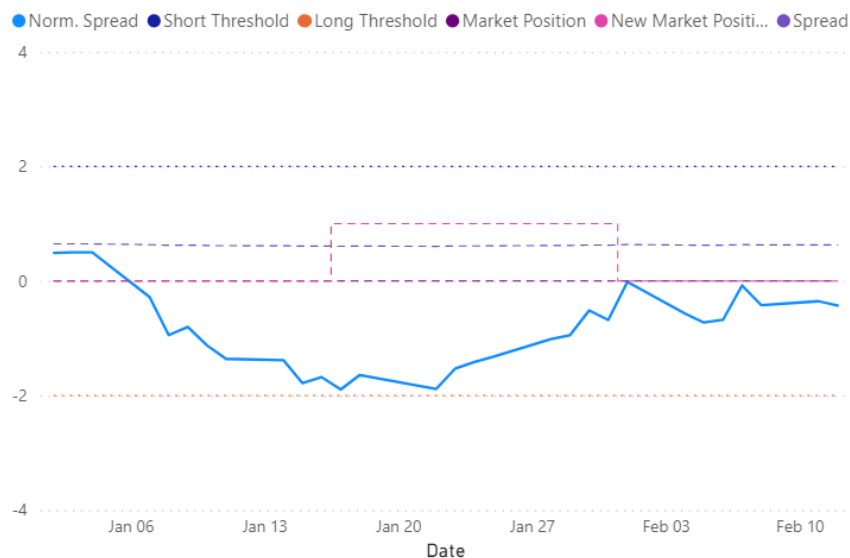


Figure 5.2: New market position opened in ADP-INTU pair

On the 17th of January, the normalized spread reached a value of -1.89 and for the following days, the predicted spread would always be higher than that (according to the forecasting model) which means that the spread should be converging back to its mean. It eventually happens, and the investor would have a profit of over 500\$ in less than 2 weeks. Similarly, in Figure 5.3, it is shown another new entry point that was made thanks to the margins added near the thresholds. This transaction had a 2k \$ profit in just 20 days.

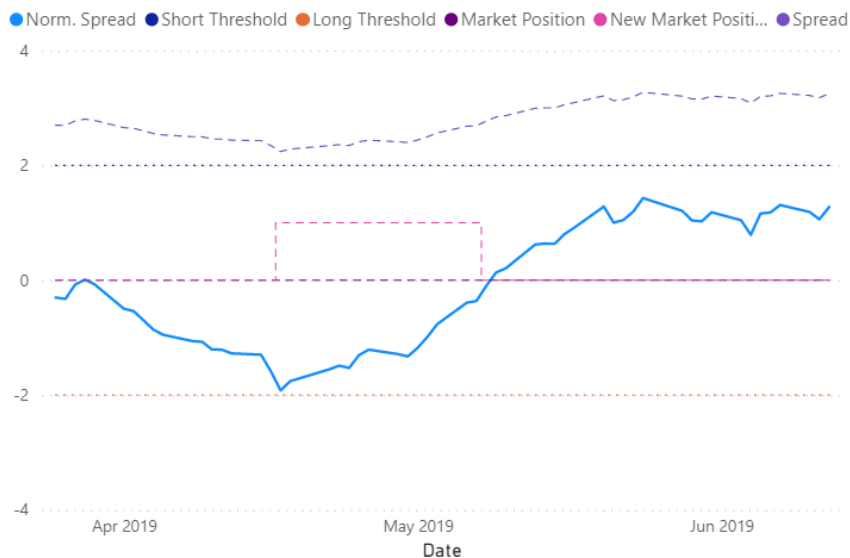


Figure 5.3: New market position opened in IDXX-MCHP pair

5.2.2.B Stop Loss Function Activation

As stated in Tables 5.2 and 5.5, and further confirmed in figure 3.3.3, the pair LRCX-MAR's spread diverged massively in the second half of the test period. This is something that the investor could not predict and for the trading algorithm, those are just market entry opportunities. It is for cases like this, that the stop-loss function was created.

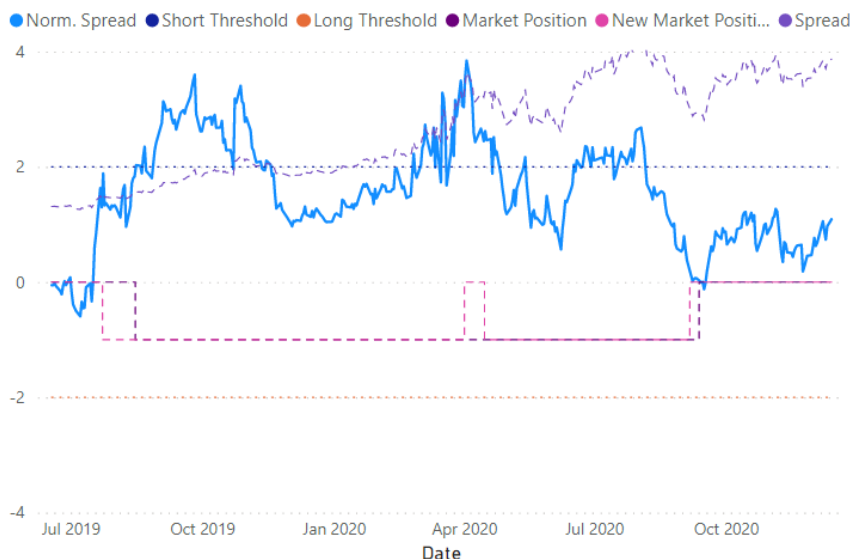


Figure 5.4: Stop Loss function being activated in LRCX-MAR pair

On the 2nd of April, the spread has had a 75% growth when comparing to the position opening day. Also, more than 100 days have passed since the opening which triggers the stop loss functions into closing it. Some days later, since the normalized spread is still above the short threshold, a new position

is reopened. The simple fact that a huge market position was split, prevented the investor from losing almost 7k \$ and would have lost 4k \$ instead.

5.2.2.C Different Entry/Exit Days

Apart from the new opportunities and the stop-loss function being activated, the two models may differ from one another in the entry and closing days of the same transaction. As explained in 3.3.2 and further detailed in Figure 4.11 the enhanced model, will use the predictions made by the forecasting model and try to calculate the best days to open or close a transaction.

In figure B.2 it is depicted an example of a transaction where each model had its entry and closing day for that same transaction.

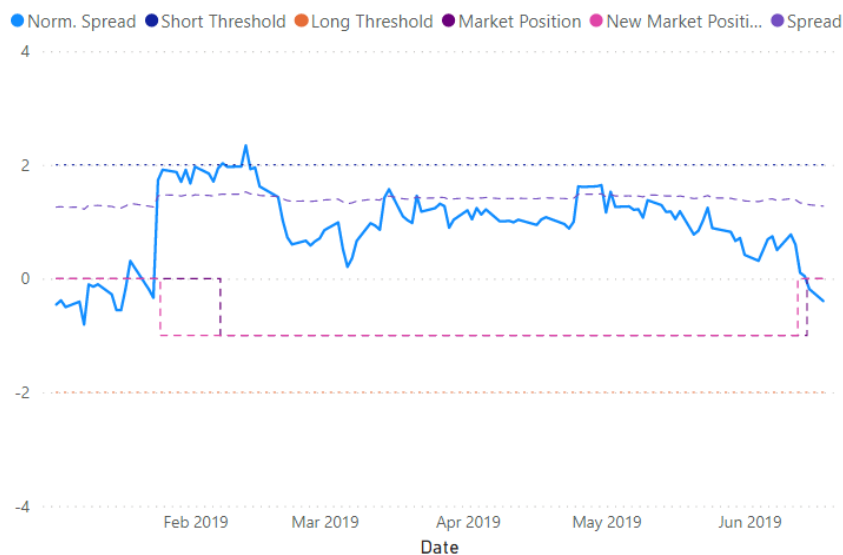


Figure 5.5: Different Entry and Closing days for the same opportunity in LRCX-MAR pair

In table 5.7 it is shown the result of the forecasting model (two columns on the right) as well as the actions taken by each model. On the 25th of January, as the Norm. Spread got closer to the short threshold, the whole process was activated and the min and max spread days were calculated. Given that according to the forecasting model the 25th was the day of maximum spread, the enhanced model opened a short position. The closing process was similar and also resulted in the enhanced model taking action a couple of days earlier than the simple model.

Table 5.7: Key actions performed by both models on the same opportunity

Date	Simple Model	Enhanced Model	Max Spread Day	Min Spread Day
25-Jan-2019		Open Short	0	3
07-Feb-2019	Open Short		0	3
12-Jun-2019		Close Short	4	0
14-Jun-2019	Close Short		4	0

Even though in both cases, the transaction was profitable, the fact that the spread kept growing to reach the threshold, means that the predicted max spread day was not right. The same happened with the closing day, meaning that the simple model had a better performance than the enhanced one. For this particular transaction, the enhanced model only achieved 80% of the profit achieved by the simple model.

5.2.2.D Overall Performance

In the end, the enhanced model outperformed the simple threshold-based one, however, that is due to the new opportunities that it found, and also due to the stop-loss function that prevented transactions lasting for too long and the spread from diverging a lot. The transactions that both models participated in, generally speaking, the enhanced model didn't perform better than the simple one. That led to the first and one of the main conclusions of this project: not only it is hard to build an accurate forecasting model to predict the market fluctuations, but it is also even harder to have a trustworthy forecast of the best entry/closing days because it requires a division of two forecasted values that increases a lot the error they may have.

Using the values presented in tables 5.5 and 5.6, table 5.8 was created. These values indicate that the profit per transaction also increased from one model to the other, and since the initial investment for each transaction is 10000 \$, it is safe to say, that on average, the profit will be around 6.4% per transaction.

Table 5.8: Profit per Transaction comparison between the two models

	Simple Threshold Based Model	Enhanced Model
Total Profit (USD)	9998	15441
Number of Transactions	19	24
Profit per Transaction (USD)	526	643

The major conclusion about the performance of the enhanced model is that it can indeed outperform the simple one. However, this outperformance comes from the finding of new trading opportunities as explained in 5.2.2.A or exiting some positions to avoid major losses (5.2.2.B). Even though these two segments may be influenced by the forecasting model, it is very minor when comparing the influence that it has whenever both models take on the same opportunity. As stated in 5.2.2.C, where the forecasting model is the only one responsible for the difference of both models, it can't out-perform the simple model due to the error that is propagated from the prediction of both stocks to the forecasted spread¹. In the end, both the new opportunities have a huge positive impact on the overall profit and whenever the enhanced model only relies on the forecasting model to make decisions it ends up losing when compared to the simple threshold model.

¹The forecasted values of the closing price of each stock naturally have an error associated, however, since the forecasted spread is the division of the two values, this error gets even bigger

6

Conclusions and Future Work

Contents

6.1	Conclusions	65
6.2	Future Work	65

6.1 Conclusions

The main goal of this project was to enhance an investment strategy using this forecasting model. Even though the enhanced model had a better performance than the simple one, the influence of the forecasting method was close to none. To predict market fluctuations, the model should be much more complex and there will always be an error. This error can increase a lot when divided by another stock prediction that also has an error associated with it. Due to this error propagation, whenever a decision relied solely on the forecasting results, its performance would be generally worse than the regular threshold model.

The simple threshold model has two main weaknesses, the first one is that it only enters on positions that cross the thresholds, wasting great profitable opportunities. The second weakness that can incur huge losses, is the fact that the simple model can hold on to positions for a long time. Tackling these two problems can increase profitability in such a way that reduces the impact of any eventual gains from entering earlier or later, based on the forecasting model. This approach had a significant boost in performance, increasing the profitability of the model by more than 54%.

6.2 Future Work

As a follow-up for this project, there are some different approaches that could be worth exploring.

On the pairs selection phase, there are some white-spaces, namely trying to mix and match individual stocks with indexes or ETF's (Exchange Trade Funds).

The forecasting model by training it to predict the spread directly using financial indicators of both stocks in the pair at the same time. The goal with this approach would be to decrease the error propagation and having a more trustworthy tool to reduce portfolio decline days, as well as, outperforming the standard model by entering/closing closer to the ideal entry/close point. One other approach would be to use sentiment analyses to predict the stock's price behaviour.

Regarding the trading model, the stop-loss function should be improved by keeping track of the spread's behaviour and in a worst-case scenario, leaving the pair to avoid reopening another position due to the high risk that it may have for the investor.

Bibliography

- [1] K. Gupta and N. Chatterjee, "Selecting stock pairs for pairs trading while incorporating lead–lag relationship," *Physica A: Statistical Mechanics and its Applications*, vol. 551, 2020.
- [2] T.-Y. Lin, C. W. Chen, and F.-Y. Syu, "Multi-asset pair-trading strategy: a statistical learning approach," *The North American Journal of Economics and Finance*, 9 2020.
- [3] J. P. Broussard and M. Vaihekoski, "Profitability of pairs trading strategy in an illiquid market with multiple share classes," *Journal of International Financial Markets, Institutions and Money*, vol. 22, pp. 1188–1201, 2012.
- [4] M. C. Blázquez, C. D. la Orden De la Cruz, and C. P. Román, "Pairs trading techniques: An empirical contrast," *European Research on Management and Business Economics*, vol. 24, pp. 160–167, 9 2018.
- [5] W. A. F. David A. Dickey, "Distribution of the estimators for autoregressive time series with a unit root," 1979.
- [6] J. G. Mackinnon, "Critical values for cointegration tests." [Online]. Available: <http://www.econ.queensu.ca/faculty/mackinnon/>
- [7] J. Mackinnon, "Approximate asymptotic distribution functions for unit root and cointegration tests," 1992.
- [8] Jieren Wang, C. Rostoker, and A. Wagner, "A high performance pair trading application," pp. 1–8, 2009.
- [9] K. Baltakys, M. Baltakienė, H. Kärkkäinen, and J. Kannianen, "Neighbors matter: Geographical distance and trade timing in the stock market," *Finance Research Letters*, vol. 31, 2019. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1544612318304367>
- [10] Y. Jiang, J. Lao, B. Mo, and H. Nie, "Dynamic linkages among global oil market, agricultural raw material markets and metal markets: An application of wavelet and copula approaches," *Physica*

A: Statistical Mechanics and its Applications, vol. 508, pp. 265 – 279, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0378437118306435>

- [11] Hinton and W. Rumelhart, "Learning representations by back propagagation."
- [12] Z. Yu, L. Qin, Y. Chen, and M. D. Parmar, "Stock price forecasting based on lle-bp neural network model," *Physica A: Statistical Mechanics and its Applications*, vol. 553, 9 2020.
- [13] D. Zhang and S. Lou, "The application research of neural network and bp algorithm in stock price pattern classification and prediction," *Future Generation Computer Systems*, vol. 115, pp. 872–879, 2 2021.
- [14] H. C. Wang, W. C. Hsiao, and S. H. Chang, "Automatic paper writing based on a rnn and the textrank algorithm," *Applied Soft Computing Journal*, vol. 97, 12 2020.
- [15] M. Dhyani and R. Kumar, "An intelligent chatbot using deep learning with bidirectional rnn and attention model," *Materials Today: Proceedings*, 6 2020.
- [16] W.-J. Wang, Y.-F. Liao, and S.-H. Chen, "Rnn-based prosodic modeling for mandarin speech and its application to speech-to-text conversion." [Online]. Available: www.elsevier.com/locate/specom
- [17] Q. Wang, W. Xu, X. Huang, and K. Yang, "Enhancing intraday stock price manipulation detection by leveraging recurrent neural networks with ensemble learning," *Neurocomputing*, vol. 347, pp. 46–58, 6 2019.
- [18] Z. Hajjaborabi, A. Kazemi, F. F. Samavati, and F. M. M. Ghaini, "Improving dwt-rnn model via b-spline wavelet multiresolution to forecast a high-frequency time series," *Expert Systems with Applications*, vol. 138, 12 2019.
- [19] A. S. Saud and S. Shakya, "Analysis of look back period for stock price prediction with rnn variants: A case study on banking sector of nepse," vol. 167. Elsevier B.V., 2020, pp. 788–798.
- [20] A. Yadav, C. K. Jha, and A. Sharan, "Optimizing lstm for time series prediction in indian stock market," vol. 167. Elsevier B.V., 2020, pp. 2091–2100.
- [21] A. Moghar and M. Hamiche, "Stock market prediction using lstm recurrent neural network," vol. 170. Elsevier B.V., 2020, pp. 1168–1173.
- [22] S. M. Sarmiento and N. Horta, "Enhancing a pairs trading strategy with the application of machine learning," *Expert Systems with Applications*, vol. 158, 2020.



List of Stocks in NASDAQ-100

Symbol	Name	Last Sale
AAPL	Apple Inc. Common Stock	141.875
ADBE	Adobe Inc. Common Stock	577.46
ADI	Analog Devices, Inc. Common Stock	166.44
ADP	Automatic Data Processing, Inc. Common Stock	204.03
ADSK	Autodesk, Inc. Common Stock	276.19
AEP	American Electric Power Company, Inc. Common Stock	82.78
ALGN	Align Technology, Inc. Common Stock	629.81
AMAT	Applied Materials, Inc. Common Stock	127.025
AMD	Advanced Micro Devices, Inc. Common Stock	105.065
AMGN	Amgen Inc. Common Stock	204.81
AMZN	Amazon.com, Inc. Common Stock	3245
ANSS	ANSYS, Inc. Common Stock	344.04
ASML	ASML Holding N.V. New York Registry Shares	724.24
ATVI	Activision Blizzard, Inc. Common Stock	75.3802
AVGO	Broadcom Inc. Common Stock	486.97
BIDU	Baidu, Inc. ADS	160.5
BIIB	Biogen Inc. Common Stock	288.13
BKNG	Booking Holdings Inc. Common Stock	2464.44
CDNS	Cadence Design Systems, Inc. Common Stock	150.46
CDW	CDW Corporation Common Stock	174.145
CERN	Cerner Corporation Common Stock	70.8
CHKP	Check Point Software Technologies Ltd. Ordinary Shares	120.49
CHTR	Charter Communications, Inc. Class A Common Stock New	693.34
CMCSA	Comcast Corporation Class A Common Stock	51.76
COST	Costco Wholesale Corporation Common Stock	451.37
CPRT	Copart, Inc. (DE) Common Stock	139.335
CRWD	CrowdStrike Holdings, Inc. Class A Common Stock	245.44
CSCO	Cisco Systems, Inc. Common Stock (DE)	54.36
CSX	CSX Corporation Common Stock	32.31
CTAS	Cintas Corporation Common Stock	404.93
CTSH	Cognizant Technology Solutions Corporation Class A Common Stock	75.04
DLTR	Dollar Tree Inc. Common Stock	98.82
DOCU	DocuSign, Inc. Common Stock	252.92
DXCM	DexCom, Inc. Common Stock	538.63
EA	Electronic Arts Inc. Common Stock	137.545
EBAY	eBay Inc. Common Stock	74.93
EXC	Exelon Corporation Common Stock	48.195
FAST	Fastenal Company Common Stock	54.35
FB	Facebook, Inc. Class A Common Stock	318.42
FISV	Fiserv, Inc. Common Stock	105.11
FOX	Fox Corporation Class B Common Stock	38.98
FOXA	Fox Corporation Class A Common Stock	42.105
GILD	Gilead Sciences, Inc. Common Stock	68.07
GOOG	Alphabet Inc. Class C Capital Stock	2741.65
GOOGL	Alphabet Inc. Class A Common Stock	2734.17
HON	Honeywell International Inc. Common Stock	215.73
HAS	Hasbro, Inc. (HAS)	92.47
IDXX	IDEXX Laboratories, Inc. Common Stock	617.6
ILMN	Illumina, Inc. Common Stock	405.41
INCY	Incyte Corp. Common Stock	64.985
INTC	Intel Corporation Common Stock	52.945
INTU	Intuit Inc. Common Stock	530.1

Table A.1: First Half Stocks Considered for the NASDAQ-100

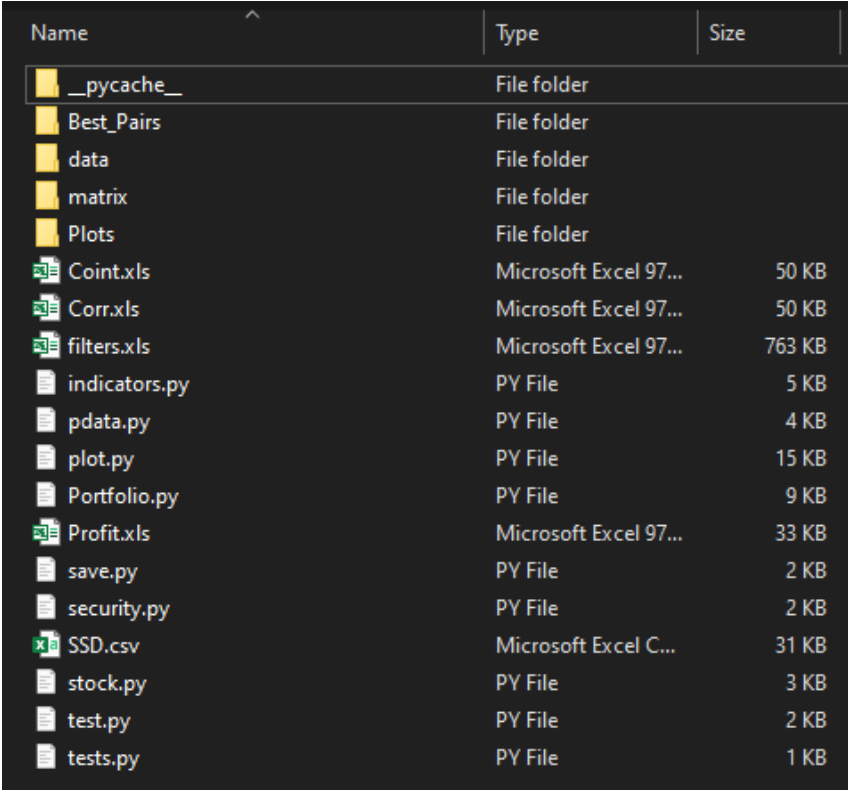
Symbol	Name	Last Sale
ISRG	Intuitive Surgical, Inc. Common Stock	323.07
JD	JD.com, Inc. American Depositary Shares	78.11
KDP	Keurig Dr Pepper Inc. Common Stock	35.085
KHC	The Kraft Heinz Company Common Stock	36.955
KLAC	KLA Corporation Common Stock	321.71
LRCX	Lam Research Corporation Common Stock	550.9841
LULU	lululemon athletica inc. Common Stock	386.89
MAR	Marriott International Class A Common Stock	156.83
MCHP	Microchip Technology Incorporated Common Stock	142.8
MDLZ	Mondelez International, Inc. Class A Common Stock	59.645
MELI	MercadoLibre, Inc. Common Stock	1470.51
MNST	Monster Beverage Corporation	88.39
MRNA	Moderna, Inc. Common Stock	306.655
MRVL	Marvell Technology, Inc. Common Stock	62.85
MSFT	Microsoft Corporation Common Stock	293.5831
MTCH	Match Group, Inc. Common Stock	157.425
MU	Micron Technology, Inc. Common Stock	66.8471
NFLX	Netflix, Inc. Common Stock	627.04
NTES	NetEase, Inc. American Depositary Shares	95.86
NVDA	NVIDIA Corporation Common Stock	207.49
NXPI	NXP Semiconductors N.V. Common Stock	184.16
OKTA	Okta, Inc. Class A Common Stock	233.22
ORLY	O'Reilly Automotive, Inc. Common Stock	608.42
PAYX	Paychex, Inc. Common Stock	117.9404
PCAR	PACCAR Inc. Common Stock	84.3
PDD	Pinduoduo Inc. American Depositary Shares	94.75
PEP	PepsiCo, Inc. Common Stock	157.5201
PTON	Peloton Interactive, Inc. Class A Common Stock	85.33
PYPL	PayPal Holdings, Inc. Common Stock	256.25
QCOM	QUALCOMM Incorporated Common Stock	123.55
REGN	Regeneron Pharmaceuticals, Inc. Common Stock	547.23
ROST	Ross Stores, Inc. Common Stock	106.71
SBUX	Starbucks Corporation Common Stock	111.68
SGEN	Seagen Inc. Common Stock	164.04
SIRI	Sirius XM Holdings Inc. Common Stock	6.03
SNPS	Synopsys, Inc. Common Stock	292.35
SPLK	Splunk Inc. Common Stock	153.08
SWKS	Skyworks Solutions, Inc. Common Stock	159.17
TCOM	Trip.com Group Limited American Depositary Shares	31.99
TEAM	Atlassian Corporation Plc Class A Ordinary Shares	394.67
TMUS	T-Mobile US, Inc. Common Stock	117.99
TSLA	Tesla, Inc. Common Stock	806.4177
TXN	Texas Instruments Incorporated Common Stock	191.119
VRSK	Verisk Analytics, Inc. Common Stock	207.94
VRSN	VeriSign, Inc. Common Stock	204.795
VRTX	Vertex Pharmaceuticals Incorporated Common Stock	179.99
WBA	Walgreens Boots Alliance, Inc. Common Stock	47.49
WDAY	Workday, Inc. Class A Common Stock	257.67
XEL	Xcel Energy Inc. Common Stock	62.915
XLNX	Xilinx, Inc. Common Stock	155.7918
ZM	Zoom Video Communications, Inc. Class A Common Stock	254.82

Table A.2: Second Half Stocks Considered for the NASDAQ-100

B

User Guide

B.1 Pairs Selection



Name	Type	Size
__pycache__	File folder	
Best_Pairs	File folder	
data	File folder	
matrix	File folder	
Plots	File folder	
Coint.xls	Microsoft Excel 97...	50 KB
Corr.xls	Microsoft Excel 97...	50 KB
filters.xls	Microsoft Excel 97...	763 KB
indicators.py	PY File	5 KB
pdata.py	PY File	4 KB
plot.py	PY File	15 KB
Portfolio.py	PY File	9 KB
Profit.xls	Microsoft Excel 97...	33 KB
save.py	PY File	2 KB
security.py	PY File	2 KB
SSD.csv	Microsoft Excel C...	31 KB
stock.py	PY File	3 KB
test.py	PY File	2 KB
tests.py	PY File	1 KB

Figure B.1: Folder containing the code and Excel files that support the Pairs Selection process

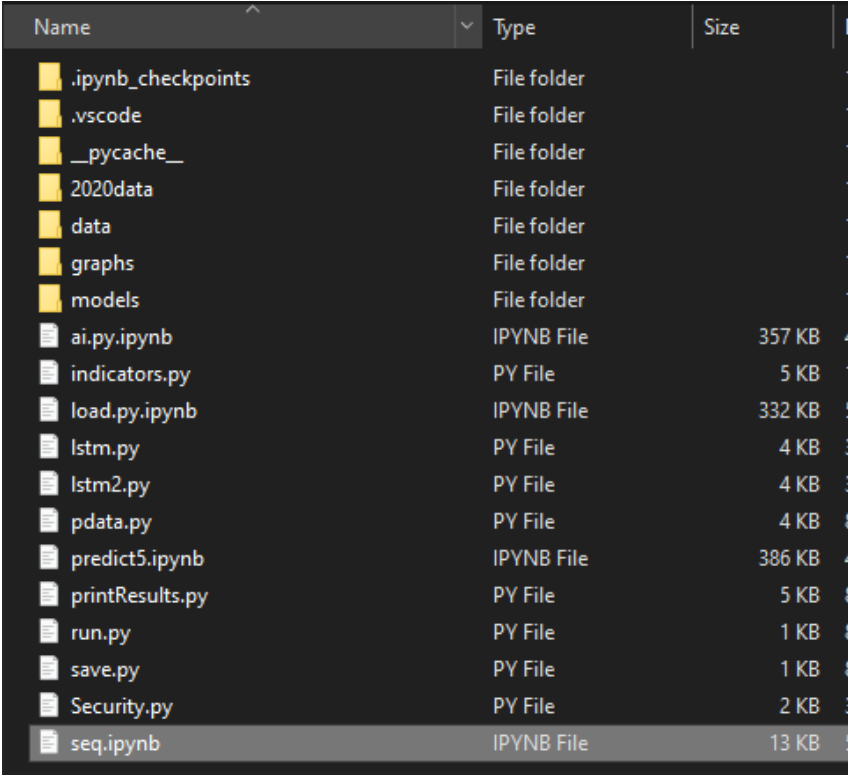
The first step is to run the "save.py" file that will get the data from the input stocks, and will also calculate their financial indicators throughout the selected time period. It will then save this information on a '.pkl' file in the "data" folder.

Running the "run.py" file will match all the saved stocks in pairs and will save on the "matrix" folder another '.pkl' file with the correlation, cointegration and ssd measures of each pair. Also, it will create this matrixes on 3 excel files that can then be copied to the "filters.xls" file. In this file, the user can select the filter for each measure and get the resulting pairs.

After getting the resulting pairs, the "stock.py" file shall be run and it will calculate the historical profit of each pair and write it on the "Profit.xls" file. On this file, and after some manual selection, the user shall choose only the pairs that provide him the best historical profit.

And this concludes the pairs selection process.

B.2 Forecasting Model



Name	Type	Size
.ipynb_checkpoints	File folder	
.vscode	File folder	
__pycache__	File folder	
2020data	File folder	
data	File folder	
graphs	File folder	
models	File folder	
ai.py.ipynb	IPYNB File	357 KB
indicators.py	PY File	5 KB
load.py.ipynb	IPYNB File	332 KB
lstm.py	PY File	4 KB
lstm2.py	PY File	4 KB
pdata.py	PY File	4 KB
predict5.ipynb	IPYNB File	386 KB
printResults.py	PY File	5 KB
run.py	PY File	1 KB
save.py	PY File	1 KB
Security.py	PY File	2 KB
seq.ipynb	IPYNB File	13 KB

Figure B.2: Folder containing the code and Excel files that support the Forecasting Model creation process

This process is easier for the user to do, since only one file needs to be run. The "seq.ipynb" is a jupyter notebook that creates models for each stock using all combinations of input features, epochs and feature windows. It will then save the best model on the "models" folder. In order to use these models on a real time scenario, some code needs to be written to allow the model to be run only for the "current" day.

B.3 Trading Model

After completing all the necessary setup described in section 4.4, there are two main things needed. The first one is to have the information up to date (assuming it is being applied in real time) such that, all the dashboards indicate how each pair has behaved in the previous days. The second thing is to be aware of what the model suggests the investor to do, either enter, close or hold. All the graphs in the powerBI can be plotted in real time enabling the investor to use them as long as all the data is up to date.