Summary

[1. Introduction 2](#_Toc140399630)

[1.1 Overview 4](#_Toc140399631)

[**1.1.1** **What is Pairs trading?** 4](#_Toc140399632)

[**1.1.2** **What is Machine Learning?** 6](#_Toc140399633)

[1.2 Research Question 7](#_Toc140399634)

[2. Data preparation 9](#_Toc140399635)

[2.1 Data collection 9](#_Toc140399636)

[**2.1.1** **Asset of choice** 9](#_Toc140399637)

[**2.1.2** **Market of choice** 11](#_Toc140399638)

[**2.1.3** **Time frame of choice** 13](#_Toc140399639)

[2.2 Data Cleaning 14](#_Toc140399640)

[3. Research Stage 1 16](#_Toc140399641)

[3.1 Pair space definition: clustering using unsupervised learning 16](#_Toc140399642)

[**3.1.1** **Dimensionality reduction** 16](#_Toc140399643)

[**3.1.2** **OPTICS** 18](#_Toc140399644)

[3.2 Pairs selection 23](#_Toc140399645)

[3.3 Trading performance 26](#_Toc140399646)

[**3.3.1** **Test Portfolios** 26](#_Toc140399647)

[**3.3.2** **Treshold-based trading model** 27](#_Toc140399648)

[**3.3.3** **Trading Simulation** 28](#_Toc140399649)

[**3.3.4** **Evaluation metrics** 30](#_Toc140399650)

[**3.3.5** **Trading performance** 31](#_Toc140399651)

# **Introduction**

The explosive growth of big data has transformed the finance industry by creating new opportunities to extract value from financial information. For global banking, McKinsey & Co. analysts estimate that AI technologies could potentially deliver up to $1 trillion of additional value each year. The use of machine learning techniques in finance has in fact shown promise in improving investment decisions, maximizing returns while minimizing risk, and developing more accurate trading strategies. Despite this, the use of such techniques in pairs trading is still scarce.

This thesis aims to contribute to the literature around the interaction between machine learning and trading by replicating the paper “A Machine Learning Based Pairs Trading Investment Strategy” by Sarmento and Horta (2020), and by exploring modifications that validate and extend their results. The paper presents a novel approach that involves using machine learning techniques to select pairs of assets to trade and a trading algorithm that uses a neural network to predict the price movements of the pairs.

The thesis will follow the same structure as the paper, which is divided into two research stages: the pairs selection stage and the trading stage. Research Stage 1, the pairs selection stage, encompasses two steps: (i) finding the appropriate candidate pairs and (ii) selecting the most promising ones. Here machine learning will be used in the form of unsupervised learning to identify pairs of assets suitable for pairs trading. In Research Stage 2, the trading stage, we will develop and test a trading algorithm that leverages supervised learning algorithms to predict the price movements of the selected pairs and generate buy/sell signals. To have a benchmark, the results obtained from the algorithms will be compared with those obtained with more traditional statistical techniques.

Each step necessary to complete these two research phases will be explained first from a theoretical point of view and then from a practical point of view, i.e. how to implement it in Python. All the codes developed for this thesis are fully reproducible and made publicly available on the author's GitHub page.

To the best of my knowledge, this thesis is one of the first to replicate and extend the original paper's approach. Firstly, we will increase the frequency of the data from 5-min to 1-min to further reduce the required formation period and consequently find more pairs. In doing so, we can also examine one of Horta and Sarmento’s propositions, which posits that shortening the training duration could enhance the predictive accuracy of the forecast-driven model. Secondly, we will add more features to predict the price variations, which could provide a more comprehensive view of the market. Thirdly, we will combine commodity-linked ETFs and currency-linked ETFs to increase the chances of finding profitable pairs. Finally, we will Experiment training the Artificial Neural Networks in a classification setting. While regression seems like the obvious choice when predicting real-valued outputs, it presents challenges in terms of optimization and robustness. The MSE loss, commonly used in regression, is more difficult to optimize and prone to disruptions from outliers, resulting in large gradients. On the other hand, classification, utilizing a stable loss function like Softmax, provides a more efficient optimization process and improved robustness. Additionally, classification offers the benefit of not only yielding a single regression output but also providing a distribution that signifies the confidence associated with the predictions.

Overall, this thesis aims to provide a comprehensive overview of the approach, including its theoretical and practical aspects, with a particular focus on the machine learning techniques used. The thesis and accompanying codes will be valuable resources for academics and practitioners interested in the application of machine learning in finance, and it is hoped that the results of this thesis will contribute to the growing body of literature around the interaction between machine learning and trading.

## **Overview**

Since this thesis has as its object the meeting of two worlds which, although superimposable, are intrinsically different, I thought it might be useful to briefly describe them both. In this way, those who are not familiar with one of the two can get an idea directly here without having to first consult other sources.

### **What is Pairs trading?**

Pairs trading is a market-neutral trading strategy that involves the simultaneous buying and selling of two highly related securities to profit from the price spread between them (Gatev et al., 2006). The strategy is based on the concept of mean reversion, which suggests that, over time, the prices of two related securities will tend to move towards their long-term average relationship (Chan, 2013). The goal is therefore to identify two stocks that have a stable relationship and initiate positions to profit from deviations from this relationship.

The relationship between two securities can be established by various methods, such as cointegration. As we will see later, cointegration is a statistical technique that involves regressing the prices of two securities on each other and testing for the presence of a long-term relationship. Other methods for establishing the relationship between two securities include fundamental analysis and technical analysis. Fundamental analysis involves examining the underlying financial and economic factors that affect the securities, such as earnings, revenue, and market share. Technical analysis involves studying price charts and identifying patterns and trends that can be used to predict future price movements.

Regardless of the chosen strategy, once a stable relationship is established, traders look for deviations from this relationship, which can be caused by various factors such as news events, changes in market sentiment, or fundamental shifts. Traders then initiate a long position in the underpriced security and a short position in the overpriced security, expecting the spread between the two securities to eventually revert to its mean value (Chan, 2013).

Let's look at a practical example. A trader has identified two stocks, BNO and UGA, that have a stable relationship.

Immagine che contiene grafico

Descrizione generata automaticamente

The trader has also found that historically, the price of stock BNO tends to be 1.5 times the price of stock UGA. However, due to a recent news event that negatively impacts stock UGA, its price has dropped significantly relative to stock BNO, causing the price spread between the two to widen. The trader believes that this deviation is temporary and that the price spread will eventually revert to its long-term average relationship. To profit from this mispricing, the trader initiates a long position in stock UGA and a short position in stock BNO.

Immagine che contiene grafico

Descrizione generata automaticamente

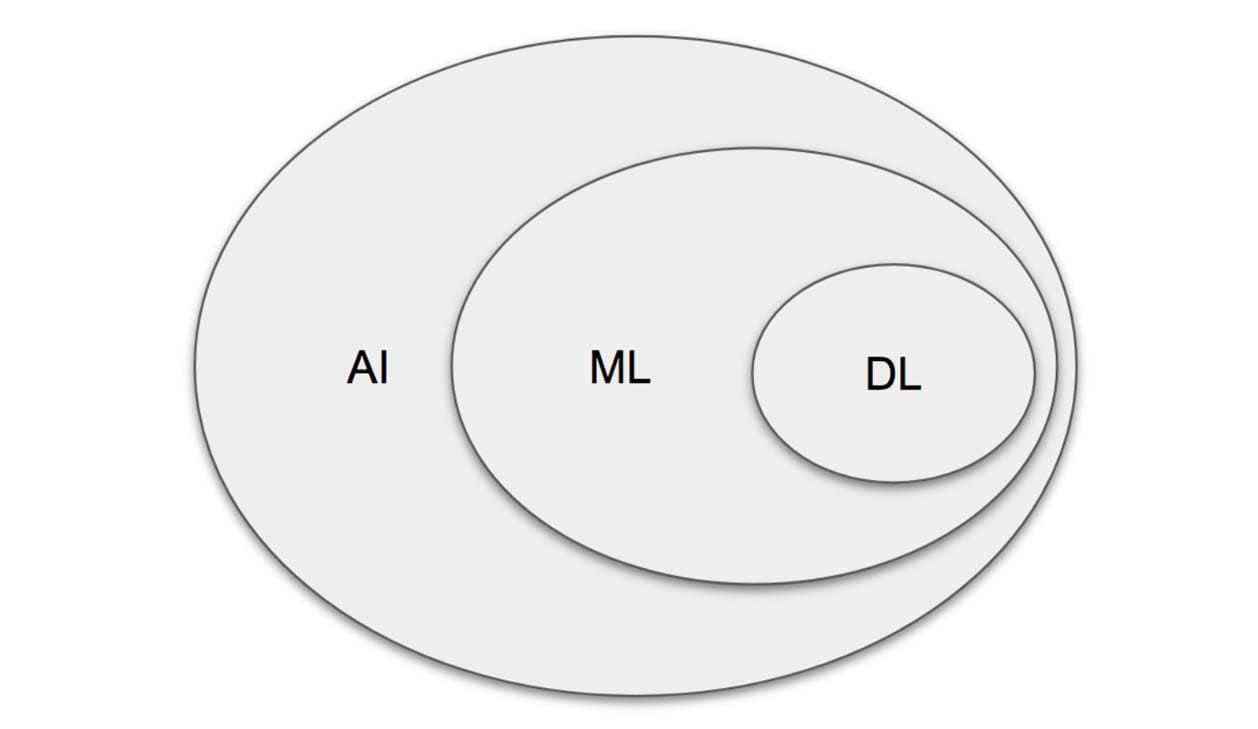
The trader profits from the price spread between the two stocks as it narrows towards its long-term average relationship, regardless of whether the broader market is going up or down.

As the price spread narrows, the trader can close the positions and lock in the profit. However, if the spread widens significantly, the trader may need to adjust the positions or close them to limit potential losses.

In summary, pairs trading is a market-neutral strategy that aims to profit from the price spread between two closely related securities. This strategy has been successfully applied in various financial markets and has been found to be profitable, as shown by studies such as Gatev et al. (2006).

### **What is Machine Learning?**

To facilitate understanding, the well-known scheme shown in the figure will be followed.



Artificial intelligence (AI) refers to the development of computer systems capable of performing tasks that typically require human intelligence, such as reasoning, learning, perception, and problem-solving (Russell & Norvig, 2016). AI can be classified into two categories: narrow AI, which is designed for specific tasks, and general AI, which aims to replicate human-like cognitive abilities across various domains (Bostrom, 2014).

A major subfield of AI is Machine learning (ML). ML is centered on the development of algorithms that allow computers to learn from data and make predictions or decisions without explicit programming (Samuel, 1959). ML techniques can be grouped into three categories:

1. Supervised learning: In this approach, an algorithm is trained using labeled data, where the input-output pairs are provided. The algorithm learns a mapping from input to output, enabling it to make predictions on new, unseen data. Common supervised learning algorithms include linear regression, logistic regression, support vector machines, and decision trees (Bishop, 2006).
2. Unsupervised learning: In contrast to supervised learning, unsupervised learning algorithms work with unlabeled data, identifying patterns or structures within the data. Clustering, a common unsupervised learning technique, groups similar data points together, while dimensionality reduction techniques, such as principal component analysis, reduce the number of variables in a dataset while preserving its structure (Bishop, 2006).
3. Reinforcement learning: This paradigm involves an agent learning optimal actions by interacting with its environment, receiving feedback in the form of rewards or penalties. The agent's goal is to maximize the cumulative reward over time. Q-learning and deep Q-networks are examples of reinforcement learning algorithms (Sutton & Barto, 2018).

Deep learning (DL) is a subfield of machine learning that employs artificial neural networks (ANNs) to model complex relationships within data (LeCun, Bengio, & Hinton, 2015). ANNs consist of interconnected nodes, or neurons, organized into layers, with each layer transforming the input data into higher-level representations. The depth of these networks, comprised of multiple layers, allows for the extraction of intricate features and patterns, giving deep learning its name.

Convolutional neural networks (CNNs) are a specific type of ANN particularly well-suited for image recognition tasks, as they can identify spatial patterns and hierarchies in data (Krizhevsky, Sutskever, & Hinton, 2012). Recurrent neural networks (RNNs), another ANN variation, have been instrumental in natural language processing and time series analysis due to their ability to model sequences and retain information from previous time steps (Hochreiter & Schmidhuber, 1997). More recently, transformer models, such as BERT and GPT, have advanced the state of the art in natural language understanding and generation, thanks to their attention mechanisms and self-supervised learning approaches (Vaswani et al., 2017; Devlin et al., 2018; Radford et al., 2018).

In the finance domain, machine learning, and more specifically, deep learning techniques have shown promising results in various applications. Recurrent neural networks, including Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures, have been employed for time-series forecasting, enabling more accurate predictions of stock prices, exchange rates, and other financial variables (Fischer & Krauss, 2018). CNNs have been utilized for sentiment analysis on financial news, extracting relevant information from text and images to inform trading strategies and risk management decisions (Ding et al., 2015). Other applications of machine learning in finance include credit scoring, fraud detection, algorithmic trading, and portfolio optimization (J.P. Morgan, 2016). By leveraging the predictive capabilities of these advanced models, financial institutions can increase efficiency, enhance risk management, and offer more personalized services to clients.

## **Research Question**

As the interest in Pairs Trading escalates, finding profitable pairs becomes more challenging. This scarcity prompts investors to explore wider security groups in hopes of increasing the probability of identifying a successful pair. A standard technique involves generating all potential pair combinations by comparing each security with every other security in the dataset. This results in n (n − 1) possible combinations, with n representing the number of available securities. Two issues emerge from this approach. Firstly, the computational expense of evaluating mean reversion for all combinations skyrockets as more securities enter the equation. This is particularly significant in situations where securities are continually monitored to create new pairs. Secondly, when conducting numerous hypothesis tests simultaneously, the multiple comparisons problem occurs. For instance, with 100 hypothesis tests executed at a 5% confidence level, the results will have a 5% false positive rate[[1]](#footnote-1). Considering a hypothetical scenario with 20 stocks, there would be 190 potential combinations, and around 9 results would be incorrect (5% of 190). This number is notably high since discovering genuine cointegrated pairs among randomly chosen securities is rare. While the multiple comparison problem cannot be completely eradicated, its influence can be lessened. Applying multiple correction tests, such as Bonferroni[[2]](#footnote-2), or conducting fewer statistical tests are two options. Bonferroni is frequently employed for multiple comparisons, but it is known to be overly conservative. In this regard, Harlacher (2016) discovered that the Bonferroni correction is excessively cautious for pairs selection, hindering the identification of genuinely cointegrated combinations. Instead, the author suggests efficiently partitioning the asset universe to decrease the number of viable combinations and, consequently, the number of statistical tests. This approach might lead investors to focus on comparing securities exclusively within the same sector. This significantly lowers the required number of statistical tests, thus reducing the chances of identifying false relationships. Additionally, securities within the same sector are more likely to exhibit correlated movements due to exposure to similar underlying factors. This method is also relatively simple to implement. However, this simplicity may prove to be a drawback, as the more traders know about the pairs, the more difficult it becomes to find pairs not already being heavily traded, leaving less room for profit.

The pursuit of a solution to this disparity involves seeking a technique that achieves equilibrium between two situations: effectively dividing the range of assets beforehand to avoid confining potential pairings to obvious solutions, all while minimizing the need for an excessive quantity of search combinations. Sarmento and Horta (2020) suggest employing an unsupervised learning technique on the expectation that it will deduce significant groupings of assets for choosing pairs. The primary objective is to enable the data to reveal its inherent patterns, as opposed to assigning predefined categories to each security individually.

The methods for identifying the candidate pair space described above will be compared in terms of profitability. Once selected, the pairs will in fact be fed to a threshold-based trading model in which, following the classical framework proposed by Gatev et al. (2016), trades are initiated based on the spread deviation. If the spread between two price series composing a pair diverges by more than two historical standard deviations, a trade is opened. The trade is closed upon convergence to the mean, at the end of the trading period or when delisting occurs.

A drawback of the threshold-based trading model is the imprecise definition of entry points. The sole criterion for entering a position is surpassing a predetermined threshold, regardless of the spread's present trajectory, which could lead to unfavorable portfolio downtrends if divergence persists. To validate the different clustering methods the threshold-based trading model is sufficient. However, it would be interesting to define a more robust strategy. To do this, Sarmento and Horta (2020) incorporate predicted future data in determining market entry points and propose the application of a forecast-driven trading approach. The model continuously track the percentage difference between the current spread and the anticipated spread in the following time-step. When the absolute value of the projected change exceeds a preset threshold, a position is initiated with the expectation of profiting from a sudden spread fluctuation.

There are several models we will explore to predict the spread, both parametric (e.g., ARMA) and non-parametric (e.g., ANN). Also in this case the implemented models will be compared in terms of profitability.

# **Data preparation**

Data preparation, also known as data preprocessing, is the process of collecting data and transforming it into a format suitable for analysis. It is a critical step in the machine learning pipeline, as the quality of the input data directly impacts the accuracy and reliability of the resulting models.

## **Data collection**

In the following section we will define the three dimensions that were followed to build the dataset on which the analysis was carried out: asset, market and time frame. Once this is done, we will move on to the cleaning phase and, finally, we will get to the heart of the analysis: the research stages.

### **Asset of choice**

As regards the choice of the asset to be traded, I followed the steps of the paper that I intend to replicate and choose Exchange Traded Founds (ETFs).

ETFs track an index, commodity, or collection of assets and trade like a stock. Chan (2013) highlights that trading ETF pairs, rather than stock pairs, offers increased stability when found to be cointegrated, as the fundamental economics of a basket of stocks evolves more slowly than individual stocks. This stability is particularly attractive for Pairs Trading, where the core assumption is that cointegrated security prices will continue to be cointegrated in the future.

To further corroborate the hypothesis that ETFs are useful in Pairs Trading, Chen et al. (2017) and Perlin (2017) provide significant references. However, before delving into the authors' findings, it is necessary to introduce some terminology. A univariate strategy is one in which both constituents of the pair are individual securities. On the other hand, if one side of the pair is a weighted average of comoving securities, the strategy is known as quasi-multivariate. Finally, when both sides of the spread are a weighted average of a group of securities, the strategy is multivariate, as illustrated in Fig.

Immagine che contiene diagramma

Descrizione generata automaticamente

a) Univariate strategy b) Quasi-multivariate strategy c) Multivariate strategy

Regarding the above-mentioned works, Chen et al. (2017) and Perlin (2017) attempt to create more stable mean-reverting time series by forming each leg of the spread with a linear combination of stocks. This is done in the hope that one outlier stock will not have such a significant impact on the spread. Chen et al. (2017) uses a Pairs Trading strategy based on Pearson correlation[[3]](#footnote-3) in both univariate and quasi-multivariate setups. In the former, a pair is formed by the two most correlated stocks. In the latter, a pair is created by a stock and an equally weighted portfolio of the 50 most correlated stocks. Results indicate that the quasi-multivariate trading strategy achieves better performance, whereas reducing the number of stocks in the weighted portfolio results in returns dropping up to two thirds. Perlin (2017) also evaluates the efficiency of a univariate strategy against a quasi-multivariate strategy for the 57 most liquid stocks in the Brazilian market. In the univariate scenario, the two most correlated stocks are paired. In the quasi-multivariate scenario, one stock is paired with five partner stocks. Once again, the results show that the quasi-multivariate strategy outperforms the univariate strategy. However, both studies are impacted by high transaction costs associated with purchasing multiple securities. Sarmento and Horta (2020) propose using ETFs as a substitute to realize the benefits of multivariate strategies more practically. This is because an ETF is essentially a weighted average of a set of securities, and therefore, trading ETFs may be considered a more practical way to implement a multivariate pairs trading strategy.

Additionally, the expanding universe of ETFs offers an interesting characteristic: the growing popularity of ETFs leads to the development of new, slightly different ETFs each year, which naturally create promising pairs for Pairs Trading.

Lastly, there is limited research on Pairs Trading applications within the ETF universe, making it an intriguing area for further exploration. This gap in the literature presents an opportunity to investigate the proposed approaches using ETFs in Pairs Trading.

### **Market of choice**

As regards the choice of the reference market, I decided to expand the one chosen by Sarmento and Horta (2020) by adding the sector of currencies to that of commodities, with the aim of increasing the chances of finding profitable pairs. To understand why, let's start by considering the relationship between commodity prices and currencies of commodity-exporting countries. According to a research paper by Chen, Rogoff, and Rossi (2010), there is a strong link between commodity prices and the exchange rates of commodity-exporting countries, such as Australia, Canada, and New Zealand. The authors found that changes in commodity prices can predict the future direction of these countries’ currencies. If commodity prices rise, the currencies of commodity-exporting countries typically strengthen because higher commodity prices increase these countries’ terms of trade. A stronger currency can then translate into higher returns for investors in currencies-linked ETFs that track these currencies. Secondly, consider the relationship between the U.S. dollar and commodities. Commodities are often priced in U.S. dollars. According to Frankel (2012), when the U.S. dollar depreciates, commodity prices usually increase. This is because when the dollar weakens, commodities become cheaper for holders of other currencies, which can increase demand for commodities and push up their prices. Thus, a negative shock to the U.S. dollar could lead to simultaneous increases in the prices of both commodities-linked ETFs and currencies-linked ETFs that track non-U.S. currencies (Frankel, Jeffrey A. "The effect of monetary policy on real commodity prices." Asset Prices and Monetary Policy, University of Chicago Press, 2008, pp. 291-327). However, it's crucial to note that many other factors can influence the prices of these ETFs, including changes in interest rates, economic growth, geopolitical events, and market sentiment. Additionally, different commodities and currencies can respond differently to the same factors. Therefore, while there can be a positive relationship between the prices of currencies-linked ETFs and commodities-linked ETFs, this relationship is not guaranteed and can vary over time and across different ETFs.

In the end, the set of securities based on which the analysis is carried out consists only of commodity-linked and currency-linked ETFs that are available for trading in January 2023. Here ETF.com has proved helpful: entering the preferred markets, the site returned 136 ETFs divided into the 7 categories (5 for commodities and 2 for currencies) listed below[[4]](#footnote-4).

|  |  |
| --- | --- |
| Category | Description |
| Agriculture | Plants or animals, or parts thereof, used primarily for producing food or fabrics |
| Broad Market | The commodity fund does not focus on a specific sector |
| Energy | Products or byproducts that relate to the generation of power |
| Industrial Metals | Metals whose primary function is in industrial use |
| Precious Metals | Gold, silver and platinum group metals |
| Basket | The fund targets a basket of currencies |
| Pair | The fund targets a pair of currencies |

The tickers obtained were then cross-referenced with a database that contained historical intraday bars of 1 minute, 5 minutes, 30 minutes, 1 hour (open, high, low, close, volume) for numerous liquid ETFs[[5]](#footnote-5).

In a context like this, a “database” is nothing more than a folder containing multiple files (usually in .txt or .csv format), one for each security. Each of these files contains a relational dataset with multiple rows, one for each time interval (in our case, one every minute), and for each row a series of information including the closing price of that interval. In other words, a time series. In this phase of the analysis, we will therefore look at several datasets, one for each ETF. Here the *locals()* function was very useful. If inserted in a loop, the function will easily allow you to create multiple variables, one for each ETF.

1. **import** pandas as pd
3. tickers **=** ['BNO', 'UGA']
4. **for** ticker **in** tickers:
5. locals()[ticker] **=** pd.read\_csv(f'{ticker}.csv')

This of course is not the only solution. In fact, we could store the datasets in a dictionary whose keys will be the names of the various ETFs (the tickers), and the corresponding historical series for values. Or we could create a single dataset with as many columns as there are ETFs. The only problem here is if you need more information than just closing prices, but a multi-level index will do the job.

Finally, other important information on the database from which the data on which the analysis is based is extracted is listed below:

* All prices and volumes are fully adjusted for both splits and dividends[[6]](#footnote-6)
* Timezone is US Eastern Time
* Timestamps run from the start of the period (e.g. 1min bars stamped 09:30 run from 09:30.00 to 09:30.59)
* Volume Numbers are in individual shares
* Times with zero volume are omitted (thus gaps in the data sequence are when there have been no trades)

The last one is also the one to which more attention should be paid since, as we will see later, it forces us to put forward an important hypothesis, namely that in the absence of a specific minute of the day the price is assumed unchanged.

### **Time frame of choice**

To better understand this section, it is helpful to first clarify some terminology. In every trading simulation, the data must be divided into two distinct periods: the formation period and the trading period. The formation period mimics the information accessible to an investor before partaking in any transaction, serving three primary functions. Firstly, it identifies the most attractive potential pairings. Secondly, a smaller subset of data, referred to as the validation set, is employed to approximate the strategy's recent performance. Thirdly, exclusive to forecasting-based models, a segment of the formation period, known as the training set, is utilized for training the prediction algorithm. The trading period is designed to emulate the performance of the adopted trading model when faced with previously unencountered data. Figure 5.4 visually represents the organization of these phases.

Immagine che contiene diagramma

Descrizione generata automaticamente

We can now describe the reference time frame. As anticipated, the chosen price frequency is 1 minute. In general, the rationale for using intraday data is threefold. First, finer granularity allows for more precise definition of entry and exit points, potentially increasing profit margins. Second, it could reveal intraday mean reversion patterns that would otherwise be undetectable. Finally, it offers a larger dataset, making it easier to train sophisticated forecasting models while reducing the risk of overfitting. In addition, we have decided to further increase the frequency, going from the 5 minutes of Horta and Sarmento to 1 minute.

This allows us to reduce the reference time frame while maintaining the same number of datapoints and, as we will see later, to find more pairs. As a result, we opted for a 6-month training period, a 1-year validation and a 1-year test period. We could have reduced these to six months too, however Do and Faff (2012) claims the profitability can be increased if the initial 6-month trading period proposed in Gatev et al. (2006) is extended to 1 year. Specifically, data will be extracted from June 2021 to December 2022.

## **Data Cleaning**

Before delving into the search of promising pairs, some data cleaning steps are implemented, as depicted in figure.

Immagine che contiene testo, lettera

Descrizione generata automaticamente

First, ETFs with insufficient data are excluded. An ETF has “insufficient data” if it is not active throughout the reference period. It is important to acknowledge that focusing solely on these ETFs introduces survivorship bias. By excluding delisted ETFs, potential positions that may have been abruptly terminated are not considered in the results. This occurs due to the unavailability of a survivorship-free dataset containing the desired ETFs. To mitigate this effect, recent periods are used whenever possible.

Regarding missing values, the issue becomes somewhat more intricate. As previously mentioned, data gaps occur when no trades have taken place. It follows that, at least theoretically, there shouldn't be any missing values. Thus, the challenge lies in determining how to fill these gaps. One possible approach is to presume a stable price and a trading volume of zero during these periods. This hypothesis may be bold but, in my opinion, it represents the optimal choice when faced with a database built in this way. Here the *fillna()* function with the “*ffill*” method was helpful.

|  |
| --- |
| 1. **for** ticker **in** tickers: 2. **try**: 3. locals()[ticker]["Close"].fillna(method**=**"ffill", inplace**=**True) 4. **except** Exception as e: 5. print(f"Error reading {ticker}: {e}") |

Next, we remove ETFs that don’t meet minimum liquidity requirements to ensure realistic transaction costs associated with bid-ask spreads[[7]](#footnote-7). The minimum liquidity criteria follow those used in Do and Faff (2012) and Gatev et al. (2006), which discard ETFs not traded for at least one day.

Ultimately, certain price series occasionally encompass outliers. Outliers are data points in a dataset that significantly differ from the others. They deviate from the general pattern and can impact data analysis. It's crucial to identify and handle them appropriately, as they could signal errors, anomalies, or important insights. Various techniques exist in the literature for anomaly detection, such as cluster analysis or other machine learning approaches. However, given the infrequency of these events in the dataset, I propose a simpler technique. For each price series, the return series is calculated and the points with percentage changes greater than 10% - an unlikely occurrence within 1 minute – are identified as outliers. If a single outlier exists in an ETF’s time series, it is corrected using Winsorization. If multiple outliers are present, the ETF is discarded.

Winsorization is a statistical technique used to minimize the influence of extreme values on the data. It can be performed in different ways, but the most common method involves replacing values that fall outside of a specified percentile range with the nearest value within that range. For example, a 5% Winsorization of a dataset involves replacing values that fall below the 5th percentile with the value at the 5th percentile, and values that fall above the 95th percentile with the value at the 95th percentile. If your data has a wide range of values and few outliers, you can use a higher percentile range, such as 99%, to preserve more of the original data.

Winsorization can be a good option when you have a small number of outliers that are skewing the distribution of your data, but you want to retain the original data scale and avoid removing the outliers completely. In this case, Winsorization can help to reduce the impact of the outliers on your statistical measures while still preserving the shape of the data.

The *winsorize()* function from the scipy.stats.mstats module provides a pre-built implementation of Winsorization that can be used to Winsorize a given dataset.

1. **from** scipy.stats.mstats **import** winsorize
3. outliers\_threshold **=** 0.1
5. **for** ticker **in** tickers:
6. **try**:
7. # Identify outliers based on returns exceeding the threshold
8. outliers **=** abs(locals()[ticker]['Return']) > outliers\_threshold
9. num\_outliers **=** outliers.sum()
11. # Remove tickers with multiple outliers
12. **if** num\_outliers > 1:
13. tickers.remove(ticker)
15. # Winsorize time series with one outlier
16. **elif** num\_outliers **==** 1:
17. winsorized\_close **=** winsorize(locals()[ticker]['Close'], limits**=**[0.1, 0.1])
18. outlier\_index **=** outliers.idxmax()
19. locals()[ticker].loc[outlier\_index, 'Close'] **=** winsorized\_close[outlier\_index]
21. **except** Exception as e:
22. print(f"Error reading {ticker}: {e}")

Finally, during the process of training forecasting-oriented models, it is essential to normalize the data using the following formula: St\_norm = (St - μs) / σs, where St\_norm signifies the standardized spread St. This normalization ensures that the series has a mean of zero and a variance of one. By normalizing the input data, neural network training can be expedited and the likelihood of encountering local optima is reduced.

# **Research Stage 1**

The following chapter consists of three sections: (i) pair space definition, (ii) pairs selection and (iii) performance evaluation. In the first section we will explain the three clustering methods we know to define the space of candidate pairs. Particular attention will be given to the one proposed by Horta and Sarmento (2020), i.e. the one that uses unsupervised learning. In the second we will explain the criteria by which certain pairs are chosen to be traded. Finally, we will implement a trading algorithm to understand which clustering method led to the choice of the most profitable pairs.

## **Pair space definition: clustering using unsupervised learning**

As anticipated, the simplest method to identify candidate pairs is to compare each security with every other security in the dataset and generate all potential pair combinations. Two drawbacks of this method are the of multiple comparisons problem and the high computational power requirements.

The second method consists in comparing each security exclusively with those belonging to the same sector. In our case, the sectors are represented by the seven categories that we have previously described. This method is relatively simple to implement, but it is precisely from this simplicity that its main disadvantage derives: the more traders know about pairs, the more difficult it becomes to find pairs that are not already heavily traded, leaving less room for profit.

This perceived imbalance fuels the quest for an approach that achieves a harmonious equilibrium between the two situations: an efficient pre-segmentation of the asset universe that avoids restricting pair combinations to apparent solutions, without necessitating an excessive number of search combinations. In the present study, we suggest employing an unsupervised learning technique on the expectation that it will deduce significant groupings of assets for choosing pairs. The main goal is to enable the data to reveal its inherent patterns, as opposed to assigning predefined categories to each security individually.

However, in order to implement this method, it is first necessary to reduce the dimensionality of each time series using principal component analysis (PCA).

### **Dimensionality reduction**

In the pursuit of identifying profitable pairings, our goal is to discover financial instruments with the same systematic risk exposure. As per the Arbitrage Pricing Theory (APT)[[8]](#footnote-8), such securities yield identical long-term expected returns. Discrepancies from these theoretical expected returns can be interpreted as mispricing, providing a basis for trading decisions. To uncover the shared underlying risk factors for each security, it is suggested to apply Principal Component Analysis (PCA) to the return series, as detailed in Jolliffe [9].

PCA is a statistical technique that employs an orthogonal transformation to change a set of possibly correlated variables into a collection of linearly uncorrelated variables, known as principal components. This transformation is designed so that the initial principal component accounts for the maximum possible variability in the data. Each subsequent component has the greatest variance possible, subject to the condition that it is orthogonal to the preceding components.

The PCA process is carried out as follows. Firstly, the return series for security i at time t, Ri, t, is derived from the security price series Pi. The return series must then be normalized, as PCA is sensitive to the relative scaling of the initial variables. This normalization is achieved by subtracting the mean, Ri, and dividing by the standard deviation σi, as in equation 2.2. The correlation matrix ρ is subsequently calculated from the normalized return series of all assets. The function of ρ will be elucidated shortly. However, it should be emphasized that the application of PCA to return series arises from the fact that a return correlation matrix offers more information to evaluate price co-movements. Instead, the use of price series could lead to the identification of false correlations due to underlying time trends. The next step involves extracting the eigenvectors and eigenvalues in order to construct the principal components. The eigenvectors identify the directions of maximum variance, while the eigenvalues quantify the variance along the corresponding direction. Singular value decomposition (SVD) can be employed to determine eigenvalues and eigenvectors. For this purpose, we arrange the normalized return series for all n securities into matrix A. By directly applying the SVD theorem, A is decomposed as A = USVT. Matrix U is orthogonal, with its columns representing the left singular vectors. Matrix S is diagonal and contains the singular values sorted in descending order along the diagonal, from the highest variance eigenvalue to the lowest. Matrix V is the transposed orthogonal matrix, with its rows comprising the right singular vectors.

Note that by multiplying ATA, we obtain VS2VT. However, ATA is precisely the correlation matrix ρ calculated earlier, so V can be determined. In this manner, it is possible to find the eigenvectors of the data matrix A. At this juncture, the k eigenvectors corresponding to the k directions of maximum variance are selected, where k represents the number of features to describe the transformed data. The more eigenvectors considered, the better the data is described. The matrix containing the selected eigenvalues in order of significance is referred to as the feature vector.

Ultimately, the new dataset is obtained by multiplying the original matrix A by the feature vector, resulting in an n × k matrix. It is important to note that the position (i, j) of the final matrix contains the product result between the normalized return series i and the eigenvector j. The resulting matrix is now reduced to the selected k features.

To conclude, the number of features, k, must be determined. A common approach involves analyzing the proportion of total variance explained by each principal component and using the number of components that explain a fixed percentage, as in Avellaneda and Lee [3]. However, this work adopts a different method. Since an Unsupervised Learning algorithm is applied using these data features, the data dimensionality should be taken into account. High data dimensionality presents a two-fold issue. The first issue is that the presence of more attributes increases the likelihood of discovering irrelevant features. The second issue is the problem of the curse of dimensionality, a term introduced by Bellman [4] to describe the problem caused by the exponential increase in volume associated with adding extra dimensions to Euclidean space. This has a significant impact when measuring the distance between seemingly similar data points that suddenly become very distant from each other. As a result, the clustering process becomes highly ineffective. Berkhin [5] notes that the effect starts to become severe for dimensions greater than 15. Bearing this in mind, the number of PCA dimensions is capped at this value and is chosen empirically.

The following table showcases the variations in clusters and identified pairs as the number of principal components fluctuates.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number of principal components | 3 | 5 | 8 | 12 |
| Number of clusters | 4 | 4 | 3 | 2 |
| Average cluster size | 7 | 7 | 7 | 7 |
| Pair selected | 21 | 13 | 6 | 2 |
| Pair selected from distinct categories | 11 | 11 | 2 | 2 |

The last line highlights the pairs made up of securities belonging to different categories, thus underlining the diversity offered by unsupervised learning.

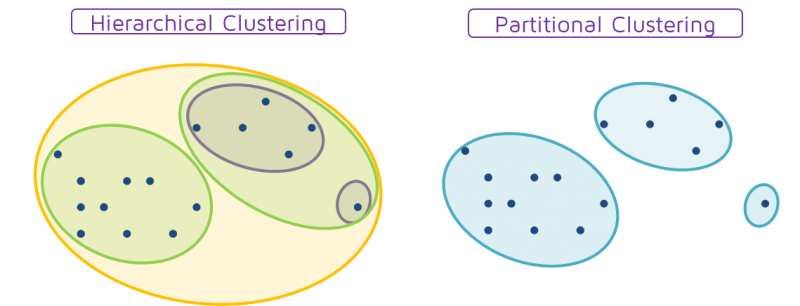
In the end I settled on three main components. Given the lack of evidence in favor of higher dimensions, I preferred to represent the ETFs in a lower one. As we will see in the next paragraph, as long as it falls within an acceptable size range, this parameter should not significantly influence the results.

### **OPTICS**

Unsupervised learning is a type of machine learning where the algorithm is not given any labeled data to learn from, but instead must discover the underlying structure or patterns in the data on its own. One common application of unsupervised learning is clustering, where the algorithm groups similar data points together based on some measure of similarity. According to the book “Hands-On Unsupervised Learning Using Python” by Ankur A. Patel, clustering “involves partitioning a dataset into groups, called clusters, such that data points within a cluster are similar to each other, while data points in different clusters are dissimilar.” Clustering methods can be divided into three broad categories: partition-based, hierarchical, and density-based clustering.

Partition-based clustering, also known as centroid-based clustering, attempts to partition the dataset into a predetermined number of clusters. The most well-known partition-based clustering algorithm is the K-means algorithm. K-means aims to minimize the sum of squared distances between each data point and the centroid of the cluster it belongs to (MacQueen, 1967). Partitioning techniques are generally regarded as inappropriate for what we are trying to do here. First, handling noisy data and outliers is a challenge for them. Secondly, partitioning algorithms impose convex cluster shapes and assuming that the data follows a normal distribution around the cluster center might be too rigorous, especially in the case of high data dimensionality. Third, determining the number of clusters beforehand is not optimal when aiming for models with minimal parameters. It may be contended that when performing a grid search to determine the optimal number of clusters, the resulting partition should conform to the configuration that achieves the lowest value of a designated criterion. Therefore, it would be satisfactory to indicate a cluster range. Nevertheless, the criteria utilized for identifying the most favorable configurations remain ambiguous. Although clustering integrity measures such as the silhouette coefficient proposed by Rousseeuw [13] might seem fitting, the empirical findings of Horta and Sarmento suggest that a strong correlation between greater cluster integrity and more promising pairs doesn't necessarily exist.

Hierarchical clustering, on the other hand, builds a tree-like structure of nested clusters, which can be represented as a dendrogram. This method can be further divided into two sub-categories: agglomerative and divisive. Agglomerative hierarchical clustering starts by treating each data point as a separate cluster and iteratively merges the closest clusters until only one cluster remains (Johnson, 1967). Divisive hierarchical clustering, as the name suggests, begins with one cluster containing all data points and recursively splits it until each data point forms its own cluster (Sneath, 1957). Hierarchical clustering techniques offer the benefit of providing a dynamic termination criteria. This enables investors to choose a cluster separation that matches their preferred level of detail. However, this could inadvertently introduce investor-related biases. Consequently, the chosen clusters may resemble outcomes achieved through conventional search approaches, which is what we seek to avoid. Therefore, employing hierarchical clustering is only suitable if an automated stopping criterion is present.



In conclusion, density-based clustering algorithms identify clusters by searching for areas of high data point density and separating them from areas with lower density (Ester et al., 1996). Density-based clustering techniques showcase certain advantages relevant to this research. Firstly, these techniques allow for the formation of clusters with various shapes, eliminating the need for assumptions about data conforming to Gaussian distributions. Additionally, such methods exhibit resilience to outliers, as they do not mandate the inclusion of all data points within a cluster. Furthermore, there is no need to predetermine the quantity of clusters. Two prominent density-based clustering algorithms are DBSCAN (Density-Based Spatial Clustering of Applications with Noise) and OPTICS (Ordering Points to Identify the Clustering Structure).

DBSCAN was introduced by Ester et al. in 1996 [1]. As anticipated, it operates on the premise that a cluster is a dense region of data points separated by areas of lower point density. The algorithm has two key parameters: a distance threshold (Eps) and the minimum number of points required to form a dense region (MinPts). It works by selecting an arbitrary unvisited data point and determining if it's part of a cluster by examining its neighborhood within the Eps distance. If there are at least MinPts points in the neighborhood, a new cluster is formed, and the process continues by expanding the cluster iteratively to include all reachable points within the Eps distance. If the neighborhood has fewer than MinPts points, the data point is marked as noise. This process is repeated until all data points have been visited.

OPTICS (Ordering Points To Identify the Clustering Structure), developed by Ankerst et al. in 1999 [2], is an extension of the DBSCAN algorithm that addresses some of its limitations. Unlike DBSCAN, OPTICS doesn't require specifying the Eps parameter, making it better suited for handling datasets with varying density. OPTICS generates an ordered list of data points, representing the density-based clustering structure of the dataset, with a reachability-distance value assigned to each point. It begins by selecting an unprocessed data point and calculating its core distance, which is the distance to the MinPts-th nearest neighbor. Then, it processes the data point's neighbors and calculates their reachability distances, updating them if a shorter distance is found. This process is repeated for each subsequent point in the dataset. The result is an ordered list of data points that can be visualized and analyzed to reveal the underlying cluster structure.

Immagine che contiene luce, scuro, notte

Descrizione generata automaticamente

By employing a reachability graph, one can effortlessly derive the hierarchical organization of various clusters. This graphical representation is two-dimensional, with the x-axis displaying the sequence of points as processed by OPTICS, while the y-axis indicates the reachability distance. Clusters are visible as troughs in the reachability graph, due to the fact that points within a cluster possess a reduced reachability distance to their closest neighbor. The more profound the trough, the greater the density of the cluster.

OPTICS offers some advantages over DBSCAN. Firstly, it can effectively handle datasets with varying densities, as it doesn't rely on a single Eps value [2]. Instead, it examines the dataset's intrinsic clustering structure. For instance, the results presented in Table 3.1.1 reveal that the impact of modifying the number of principal components within the tested range is minimal. The consistent number of clusters and their average size across different principal component counts can be attributed to the stability provided by the OPTICS algorithm. If the number of principal components is chosen within a reasonable range, the OPTICS algorithm automatically adjusts its parameters to best fit the data across various dimensions. Achieving the same outcome using DBSCAN is significantly more challenging, as investors would need to identify the optimal value for the ε parameter in each case. Secondly, OPTICS provides an ordered list of data points that can be easily visualized and analyzed to determine the clustering structure, whereas DBSCAN produces a set of clusters that may require additional interpretation.

Now let's see the results derived from the application of OPTICS on our dataset. Below is the composition of the clusters identified by OPTICS after reducing the dataset to three principal components[[9]](#footnote-9). Each visualization represents the logarithm of the price series for each ETF included in the cluster. The original price series has been adjusted by subtracting its average value for clarity.

|  |  |
| --- | --- |
| Immagine che contiene testo, linea, Diagramma, schermata  Descrizione generata automaticamente | Immagine che contiene testo, linea, Diagramma, diagramma  Descrizione generata automaticamente |
| Immagine che contiene testo, linea, Diagramma, diagramma  Descrizione generata automaticamente | Immagine che contiene testo, Diagramma, schermata, linea  Descrizione generata automaticamente |

Beginning with Figure a, it is fascinating to note the proximity of the series to one another, making them difficult to distinguish. This raises the question of whether a fundamental explanation exists for this pattern. Indeed, there is one. The four securities pinpointed are not only part of the Precious Metals category but also have a connection to Gold. This serves as initial proof that the OPTICS-driven method can delve further into categorizing ETFs, as it can discern not only general categories but also specific segments.

A comparable scenario can be found in Figure d. Here, the identified securities also fall within a single segment, Energy Crude Oil, which is a component of the broader Energy sector.

Figure b reveals that the OPTICS clustering abilities extend beyond merely identifying ETFs within the same segment. In cluster 2, we can find ETFs related to the Agriculture sector (COW), ETFs connected to funds focusing on currency pairs (FXA, FXB, FXC, FXE), and ETFs linked to funds targeting a collection of currencies (UDN). Despite not all belonging to the same category, a discernible relationship exists among the identified price series. The same holds true for cluster 3, which consists of two distinct categories: Agriculture (CANE, CORN, DBA, NIB, SOYB) and Broad Market (BCI, COMB, FTGC, GCC, USCI).

In conclusion, we can verify that the clusters formed achieve their goal of integrating the desired characteristics from the two other methods examined. Specifically, they display a tendency to group subsets of ETFs from the same category while still allowing for clusters containing ETFs from various categories.

## **Pairs selection**

Once the clusters have been defined, it remains essential to establish a series of criteria to select the pairs to be traded. Ensuring the stability of couples is essential. With this in mind, we suggest combining techniques that have been used independently in previous studies. In particulare, a pair is selected if it meets the following four conditions:

1. The components of the pair exhibit cointegration;
2. The spread Hurst exponent for the pair demonstrates a tendency towards mean reversion;
3. The divergence and convergence of the pair's spread occur within suitable timeframes;
4. The frequency of the pair's spread reverting to the mean is adequate.

We will now delve into each stage more comprehensively to elucidate the underlying rationale.

First, a pair a pair is only deemed eligible for trading if the two securities that form the pair are cointegrated.

To better understand the concept of cointegration and why it is crucial in this context, we must first introduce the notion of stationarity. A random process is considered stationary if its mean and variance remain constant over time, as stated by Adhikari and Agrawal [1]. Consequently, a time series that is stationary exhibits mean-reverting behavior, with oscillations around the average exhibiting similar magnitudes. A stationary process can also be described in terms of its order of integration, I(d). In this sense, a stationary time series is identified as an I(0) process, while a non-stationary one as I(1). The order of integration, d, serves as a summary statistic representing the minimum number of differences needed to obtain a stationary series. To ascertain whether a time series is stationary or not, a widely-used method is the Augmented Dickey-Fuller (ADF) test. Developed by Dickey and Fuller (1979), this test is a statistical technique designed to determine the presence of unit roots in a time series, which are indicative of non-stationarity[[10]](#footnote-10). The null hypothesis of the ADF test is that the time series contains a unit root (i.e., it is non-stationary), while the alternative hypothesis is that the series is stationary.

We can now move on to the concept of cointegration. Cointegration was first proposed by two econometricians, Engle and Granger (1987). A set of variables is said to be cointegrated if there exists a linear combination of those variables, of order d, which results in a lower order of integration, I(d − 1). In the context of this work, cointegration is verified if a set of I(1) variables (non-stationary) can be used to model an I(0) variable (stationary). Formally speaking, the concept of cointegration involves two time series, yt and xt, which are both characterized as I(1). This implies that there exist coefficients, denoted by μ and β, such that the equation yt − βxt = ut + μ holds, where ut represents a stationary time series. The significance of this relationship lies in the fact that it provides a formal means of generating a stationary time series for trading purposes. In order to examine this condition, we suggest utilizing the two-step Engle-Granger cointegration test, as it offers a straightforward approach. The test is carried out as follows:

1. Employ the Augmented Dickey-Fuller (ADF) test to determine if a unit root exists in the series yt and xt. If confirmed, move to step 2.
2. Execute the regression defined as above using Ordinary Least Squares, and store the residuals, ût.
3. Utilize the ADF test (or a similar test) to examine the residuals ût for a unit root.
4. Rejecting the null hypothesis of a unit root in the residuals (the null hypothesis of no cointegration) implies that the residual series is stationary and the two variables are cointegrated.

A significant drawback of the Engle-Granger method is that selecting the dependent variable may yield differing outcomes, as highlighted by Armstrong [2]. One solution to address this problem would be to perform the test for both potential choices of the dependent variable (i.e., for both securities that make up the pair), and then choose the option that yields the lowest t-statistic.

An additional validation step is proposed to increase confidence in the mean-reversion nature of pairs' spread, and to reduce false positives due to the multiple comparisons problem. This step involves enforcing a condition that the Hurst exponent associated with the spread of a given pair is less than 0.5. The Hurst exponent serves as an indicator of the long-term memory of time series. It relates to the autocorrelations present within the time series and the rate at which these diminish as the time lag between value pairs increases. The Hurst exponent ranges between 0 and 1, with three scenarios emerging based on its value. When H > 0.5, persistence is observed, suggesting that an increase (or decrease) in the time series values is more likely to be followed by a similar trend. If H < 0.5, the time series demonstrates anti-persistence or mean reversion, implying that an increase (or decrease) in the time series values is more likely to be followed by an opposing trend. Lastly, if H = 0.5, the time series follows a random walk and is considered to be uncorrelated (Peters, 1994). There are several methods for calculating the Hurst exponent, such as the rescaled range (R/S) analysis, the detrended fluctuation analysis (DFA), and the wavelet-based methods (Feder, 1988; Kantelhardt et al., 2002; Turiel et al., 2006). In the R/S analysis, the time series data is divided into non-overlapping segments, and for each segment, the range (R) and standard deviation (S) are calculated. The rescaled range is obtained by dividing R by S, and the Hurst exponent is estimated by analyzing the scaling behavior of the rescaled range as a function of the segment size (Hurst, 1951; Peters, 1994).

It is important to recognize that the Hurst exponent evaluates the extent of mean reversion in a time series, rather than its stationarity. In other words, it should not be considered a direct substitute for the cointegration test. While it is true that most stationary time series also exhibit mean reversion, there can be exceptions. For instance, ponder the process X that follows the rule Xt = Xt−1 for t > 0, and X0 is assigned the value 1 with a 50% probability and 0 otherwise. Although X is stationary, it does not display mean reversion, demonstrating that stationarity does not necessarily equate to mean reversion. However, such cases are infrequent, and thus, we disregard them in our approximation.

Thirdly, the stationarity of the spread is a sought-after attribute when selecting pairs. Nevertheless, a mean-reverting spread alone doesn't guarantee profits. The duration of mean-reversion and the trading timeframe must align. A significant concept in this context is the half-life, which can be viewed as an approximation of the anticipated time for the spread's mean-reversion (Bouchaud, Jean-Philippe, et al., "An Introduction to Statistical Finance," Cambridge University Press, 2018). Essentially, the half-life represents the time it takes for a spread to revert halfway back to its mean from its current position.

If a spread takes about 400 days for mean-reversion and the trading timeframe is one year, it's improbable that a profitable situation arises. Similarly, an extremely brief mean-reversion period is also unfavorable. As the half-life plays a crucial role in estimating the expected time for mean-reversion, we suggest eliminating pairs whose half-life doesn't align with the trading timeframe.

Furthermore, we require that each spread intersects its mean at least once every month, ensuring sufficient liquidity. It's important to acknowledge that while a (negative) correlation exists between the number of mean intersections and the half-life duration—more mean intersections naturally correspond to a shorter half-life—these characteristics are not always interchangeable (Cont, Rama, and Peter Tankov, "Financial Modelling with Jump Processes," Chapman and Hall/CRC, 2003). Incorporating this constraint might not only reinforce the prior condition but also exclude pairs that, despite meeting the mean-reversion timing requirements, fail to intersect the mean, offering no chances to exit a position.

Immagine che contiene diagramma

Descrizione generata automaticamente

After outlining the four steps for determining a suitable pair, we now detail the parameters employed in each phase. Figure 3.5 illustrates the four criteria a pair must satisfy to be eligible for trading. Firstly, pairs must exhibit cointegration with a 5% p-value. Next, the Hurst exponent of the spread, denoted by H, should be less than 0.5. Moreover, the half-life duration, represented by hl, should fall between one day and one year. Lastly, it is mandated that the spread intersects a mean at least 12 times per year, ideally equating to a minimum average of one intersection per month.

### **Pairs selection performance**

We commence by showcasing the quantity of pairs that the system chooses under the three distinct clustering techniques. These numbers are displayed in Table 3.2.1.

|  |  |  |  |
| --- | --- | --- | --- |
|  | No clustering | By category | OPTICS |
| Number of clusters | 1 | 7 | 4 |
| Possible combinations | 1035 | 155 | 96 |
| Pairs selected | 141 | 24 | 21 |

As anticipated, the single cluster method encompasses a wider range of ETFs to pick the pairs from. Naturally, this setup yields more pairs. Yet, it remains to be evaluated whether all the chosen pairs are similarly lucrative.

It can be seen that when we categorize the ETFs into seven groups (based on the classifications defined in Sect. 5.2.2), it leads to a decrease in the potential pair combinations.

Lastly, when utilizing OPTICS, the number of potential pair combinations is significantly less. These outcomes are achieved using three main components to depict the reduced data. Moreover, the only parameter for OPTICS to be determined, minPts, is assigned a value of three, as we have empirically confirmed that this value facilitates a reasonable distribution of clusters. Even though the quantity of clusters surpasses that when sorted by category, their smaller size leads to less combinations. This not only minimizes the computational load of conducting multiple statistical tests, but it also lowers the chance of detecting spurious pairs.

Let's now explore in depth the process of selecting pairs. The pair elimination at each stage of the process is shown in Table 6.3.

|  |  |
| --- | --- |
| *Pairs eliminated per stage* | |
| 1. Cointegration | 1004 |
| 2. Hurst exponent | 69 |
| 3. Half-life | 4 |
| 4. Mean-crosses | 0 |
| Not eliminated | 186 |
| Total Pairs | 1286 |

It's crucial to acknowledge that these steps occur in a sequential manner. This implies that each row exhibits the pairs that have been removed from the subset created by the prior selection criteria. Consequently, it's natural to assume that the largest portion of pairs are weeded out at the initial stage, and our observations align with this assumption. Furthermore, the function of the Hurst exponent in removing pairs that pass the cointegration test, yet their spread didn't record a Hurst exponent under 0.5, thereby not portraying a mean-reverting process, is substantiated.

The data in Table 6.3 also suggests that certain pairs are disqualified due to the incompatibility between their convergence period and the trading timeframe, which leads to the non-fulfillment of the half-life requirement. However, it's clear that the mean-crossing stipulation is satisfied by the entire subset of pairs that met the earlier requirements, signifying that applying this final regulation is superfluous in this instance.

To wrap up, we can assert that the Hurst exponent and the half-life constraints play a substantial role in sifting out pairs that would not have been ruled out if only cointegration was imposed.

## **Performance evaluation**

The objective of this last section is to evaluate the trading performance of the pairs that have been selected within each of the three cluster settings. By doing so we will be able to identify the most effective clustering approach to produce a high potential investment portfolio.

### **Test Portfolios**

Let's start by saying that three separate test portfolios are used to simulate all potential trading circumstances. The first portfolio includes all pairs identified during the training period. The second portfolio capitalizes on the insights gained from executing the strategy within the validation set, picking only the pairs with favorable outcomes. The third and final portfolio reflects a scenario where an investor has to commit to a fixed number of k pairs. In this case, we propose to select the top-k pairs based on the returns obtained in the validation set. We adopt k = 10, which is an intermediate value between Gatev et al. [12] choices of k = 5 and k = 20. Through the examination of various portfolio structures, we aim not only to identify the most effective clustering approach, but also to evaluate its ideal implementation conditions. The figure illustrates the three portfolios.

Immagine che contiene diagramma

Descrizione generata automaticamente

It should be highlighted that while choosing pairs according to their performance on the validation set may appear suitable, it is not certain that the top-performing pairs within the validation set will exhibit a similar pattern in the test set. The effectiveness of this guiding principle will be determined by the outcomes.

### **Treshold-based trading model**

As the primary objective of this research stage is to assess the outcomes of the pair selection methods in relation to one another, optimization of the trading model is not our focus. Consequently, we employ the conventional threshold-based model with the parameters specified in Table.

Within the framework suggested by Gatev at al. [15], the initiation of a trade is contingent on the divergence of the spread. When the spread between a pair's two price series exceeds twice the historical standard deviation, a trade is initiated. Trades are then closed when the spread converges to the average, at the conclusion of the trading period, or in the event of delisting. This model is likely to be familiar to readers, as it is the approach introduced in Section 1.2.1.

The model can be formally outlined as follows:

1. Establish the model thresholds: the long position-triggering threshold (αL), the short position-triggering threshold (αS), and the exit threshold (αexit) that determines when to close a position.
2. Track the spread (St) development and check for threshold crossings.
3. If αL is crossed, go long on the spread by purchasing Y and selling X. If αS is crossed, short the spread by selling Y and buying X.
4. Exit the position when αexit is crossed and a position is being held.

It is crucial to acknowledge that the definition of the spread, St, is closely associated with the method employed to identify the pairs. For instance, using the minimum distance approach, the spread formed by securities Y and X is straightforwardly expressed as St = Yt - Xt.[[11]](#footnote-11) Conversely, with cointegration, the spread takes on a slightly altered form, represented as St = Yt - β Xt. This nuance impacts how positions are established.

If the spread is defined as St = Yt - Xt, entering a long position involves purchasing Y and selling X in equal share quantities or allocating the same amount of capital to both, assuming the investor desires a dollar-neutral position. However, if the spread is expressed as St = Yt - β Xt, a decision must be made regarding the treatment of the cointegration factor. Various suggestions have been put forth in academic literature, but no consensus on the most suitable option has been reached.

|  |  |
| --- | --- |
| Parameters | Values |
| Long Threshold |  |
| Short Threshold |  |
| Exit Threshold |  |

Finally, it is important to note that there is existing research that specifically investigates the optimization of trading thresholds, such as the studies by Göncü and Akyıldırım [13] and Huang et al. [14]. Moreover, some researchers, like Dunis et al. [9], delve into the utilization of sliding windows to continuously update the mean (μs) and stardard deviation (σs). Nonetheless, we maintain this framework for the sake of simplicity.

### **Trading Simulation**

This segment provides a comprehensive account of the trading simulation. We delve into the various elements that were considered while constructing the portfolio, transaction costs, as well as the position settlement conditions.

We begin by observing that the portfolios analyzed in this study may differ in size. Regardless, we determine that all pairs within the portfolio should have equal weight. This allows the portfolio returns to be calculated by simply averaging the performance of all pairs, without worrying about the initial investment's relative proportion.

A logical question that arises is the allocation of capital for each pair. Theoretically, if equal values are maintained in the long and short positions of a pair, the capital gained from the short position can completely cover the cost of entering the long position, requiring no initial financing. This concept is known as a self-financing portfolio. While this notion is theoretically sound, it is not applicable in practice, as collateral is always necessary for borrowing the security being shorted, making zero initial investment impossible. Thus, the required investment corresponds to the collateral, which we assume to be the value of the security being shorted. This amount allows the investor to enter a long position of equal value. By using proceeds from short positions to finance long positions, the investor leverages their investment. This leverage strategy is common in hedge funds to increase the portfolio's absolute return and is therefore considered in this study.

For the sake of simplification, it is beneficial to assume a one-dollar investment in each pair. This method is commonly used by researchers in the field (Avellaneda and Lee [1], Caldeira and Moura [3], Dunis et al. [9], Gatev et al. [12], Rad et al. [19]), making it appropriate for comparison. It should be noted that this approximation assumes that an investor can buy fractional shares of trading ETFs since the security is likely not valued at exactly one dollar. In this study, we adopt this assumption based on the idea that the investor can always find the least common multiple between the two security prices, allowing them to take equal-sized positions in both ETFs.

In contrast to some studies that aim to be dollar-neutral, such as Gatev et al. [12] and Dunis et al. [9], which invest $1 in both long and short positions, our methodology adheres to the cointegration ratio β between the two securities. We suggest that the value invested in X is β times the value invested in Y. Under these conditions, to maintain a $1 initial investment, it must be ensured that neither the long nor the short position costs more than $1. Formally, the imposed condition is as follows:

max(leg1, leg2) = $1,

where leg1 and leg2 denote the capital invested in each leg of the pair. Based on this, we develop a framework depicted in Fig.

Immagine che contiene diagramma

Descrizione generata automaticamente

Throughout the trading process, we assume that all capital earned by a pair during the trading period is reinvested in the subsequent trade. For example, if a pair initially trades with a 5% return, the next time a trade is opened for that pair, the starting capital will be $1.05 instead of $1. This mechanism streamlines the calculation of the final return.

We confirm that all outcomes presented in this research take transaction expenses into account. The transaction fees taken into consideration are based on approximations from Do and Faff's study [7]. Their research delves into the effects of transaction costs on Pairs Trading and offers cautious calculations of all inherent costs. Other researchers in the field, like Huck and Afawubo [15], have also utilized these estimates. The costs can be separated into three components, as depicted in Table 5.4. The fee is presented in relation to the position's size.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Commission costs | Market impact | Rental costs |
| Description | Value charged when buying or selling a security | Indicator that reflects the cost faced by a trader due to market slippage | Constant loan fee for the short position, payable over the life of each trade |
| Charge | 8 bps | 20 bps | 1% per annum |

A crucial nuance in trading execution pertains to the price points considered when initiating a position. This applies to both threshold models and, as we will see later, forecast-based models, where the system constantly monitors price movements to see if a specific limit is violated, prompting you to establish a position. However, due to the unavoidable delay in a trading system's reaction, a slight variation in the entry price may arise, linked to the time taken to set a position. The way this issue is tackled in the proposed simulation setting merits discussion.

Drawing from the work of Do and Faff [6], Dunis et al. [9], and Gatev et al. [12], we adopt a cautious delay of five periods before entering a position. Therefore, if a position is activated at t = i, we only step into the position at time t = i + 5 min (considering the dataset in question has a 1-minute frequency). This measure is designed to ensure the strategy is implementable in real-world scenarios. If it turns out to be profitable under these conditions, the results could only be enhanced by assuming a less-than-five-minute delay, which is entirely plausible in practice.

In terms of specifying exit points, this study does not incorporate a stop-loss system in any situation. This implies a position is only relinquished if one of two conditions is met. Either the pair reaches convergence or the trading period concludes. In the latter situation, a recently established position might diminish the overall return, despite its potential profitability in the future. However, this closely mirrors how a hedge fund would declare its earnings at the conclusion of a specific period.

### **Evaluation metrics**

We will now discuss the most suitable financial evaluation metrics for assessing the suggested strategies, which include Return on Investment (ROI), Sharpe Ratio (SR), and Maximum Drawdown (MDD).

In its most basic form, return on investment can be determined by dividing net profit by the initial investment, as shown below:

ROI = Net Profit / Initial Investment × 100

It should be noted that the portfolio creation method used in this study ensures that the required initial investment is always $1. As a result, the return at any given time can also be viewed as the net profit.

When determining the returns for the entire portfolio, a question arises regarding whether the portfolio returns should be averaged across all chosen trading pairs or solely across those that opened positions during the trading period. Gatev et al. [12] referred to the first approach as return on committed capital (RCC) and the second as fully invested return (FII). This study adopts the first approach, as it is more conservative and takes into account the opportunity cost of hedge funds committing capital to a strategy even if no trades are made.

An additional detail should be highlighted. Dunis et al. [9] argue that in a realistic setting, the financial institution lending shares would charge interest on the collateral, which should be added to the strategy's net profit. However, we have chosen to disregard this aspect for the sake of simplicity, as it would only account for a small fraction of the net profit.

As explained in Section 5.5.1, the returns calculated are based on a leveraged position. In the case of an unleveraged position, the initial capital corresponds to the initial gross exposure (the sum of the long position and the short position), resulting in slightly lower returns.

The Sharpe Ratio is a well-established metric in financial analysis that helps investors understand the return of an investment compared to its risk. Named after William F. Sharpe, the ratio is a measure of the excess return (or risk premium) per unit of deviation in an investment asset or trading strategy, typically referred to as risk (Sharpe, 1966). The formula for calculating the Sharpe Ratio is:

Sharpe Ratio = (Rp - Rf) / σp

Rp represents the expected portfolio return, which is the return that an investor anticipates from a portfolio over a specific period. This value can be calculated by summing the products of the expected return of each asset and its weight in the portfolio.

Rf stands for the risk-free rate. The risk-free rate represents the projected rate of return for an hypothetical investment that carries no financial risk. When evaluating the returns generated by a specific strategy, it is necessary to subtract the risk-free rate to account for the potential interest that could have been earned on the same amount of cash with no risk involved. In this study, we adopt the common practice of setting the risk-free rate equal to the interest paid on the 3-month US government treasury bill. Table 5.5 demonstrates the risk-free annualized rates considered for each tested period. These rates are calculated by averaging the rates of the 3-Month Treasury bill during the corresponding periods. The data is sourced from the Board of Governors of the Federal Reserve System (US) [2].

σp, the denominator, represents the standard deviation of the portfolio's excess return, also known as the total risk. Standard deviation provides a measure of the dispersion of a set of values. In finance, it's used as an indicator of investment risk, or volatility. A low standard deviation suggests a tendency towards a mean, while a high standard deviation signals that data points are spread out over a large range of values. However, a significant issue with the Sharpe Ratio arises when considering the annualization factor. Standard practice is to annualize the Sharpe Ratio when the data frequency is not on an annual basis. This is typically accomplished by multiplying the Sharpe Ratio by the square root of the number of periods in a year. This process, though, is founded on the assumption that returns are independently and identically distributed, which is often not the case in the financial markets. To achieve a more precise approximation, we embrace the correction factor proposed by Lo [17]. This involves measuring the serial correlation of portfolio returns and subsequently applying a scale factor based on the findings. The specific details regarding the scale factors can be found in Appendix 2.

The Sharpe ratio, which normalizes returns by factoring in the associated risks, is contrasted in this study with the Maximum Drawdown. The latter metric signifies the largest noted downturn from a high point to a subsequent low in a data sequence, before a fresh peak is reached. Throughout the trading duration, this metric is specifically evaluated in relation to the balance of the account.

### **Trading performance**

1. In hypothesis testing a type I error is the rejection of a true null hypothesis (false positive), while a type II error is the non-rejection of a false null hypothesis (false negative). Formally, the probability (α) of committing at least one type I error when performing m independent comparisons is given by α = 1 − (1 − α) m. [↑](#footnote-ref-1)
2. The Bonferroni correction controls the Famili-Wise Error Rate, the probability of making a Type I error among a specified group (or "family") of tests in multiple comparisons, by adjusting the significance level (α). It divides the overall significance level (α) by the number of tests (m) to obtain an adjusted significance level (α\_adjusted). Each test is then performed using α\_adjusted as the threshold for statistical significance. [↑](#footnote-ref-2)
3. Pearson correlation is a statistical measure that quantifies the strength and direction of the linear relationship between two continuous variables. It ranges between -1 and +1, with +1 indicating a perfect positive relationship, -1 indicating a perfect negative relationship, and 0 indicating no relationship. [↑](#footnote-ref-3)
4. Please refer to https://www.etf.com/docs/FactSet\_ETF\_Classification\_System\_Rules\_and\_Methodology\_June2018.pdf for details of the ETFs classification system. [↑](#footnote-ref-4)
5. Please refer to <https://firstratedata.com/b/17/etf-complete-historical-intraday> for details on the database used [↑](#footnote-ref-5)
6. Please refer to <https://firstratedata.com/about/price_adjustment> for details of the adjustments. [↑](#footnote-ref-6)
7. Trading illiquid ETFs would result in higher bid-ask spreads which could dramatically impact the profit margins. [↑](#footnote-ref-7)
8. The APT is a model according to which the return of a share is expressed as a function of the returns of a series of risk factors (e.g. factors linked to macroeconomic variables such as the price of oil or GDP; but also factors of a different nature). [↑](#footnote-ref-8)
9. To identify which columns of the original dataset are within the clusters, it is first necessary to determine which original variables are most associated with each principal component. This can be done using principal component loadings, which are obtained by multiplying the original dataset by the eigenvectors of the principal components. For example, suppose you have a dataset with five variables (X1, X2, X3, X4, X5) and you have extracted three principal components. The loadings for the first principal component might be X1: 0.5, X2: 0.4, X3: 0.7, X4: 0.1, X5: 0.2. This means that variable X3 has the strongest association with the first principal component, followed by variables X1 and X2. Variables X4 and X5 have a weaker association with the first principal component. Once you have identified which variables are most strongly associated with each principal component, you can look at the cluster assignments for each observation in the dataset. The cluster assignments will indicate which observations are similar to each other in the principal component space. You can then examine the original variables associated with each observation to determine which variables are most commonly associated with each cluster. For example, suppose you have three clusters, labeled 1, 2, and 3. You can examine the observations in each cluster and count the number of times each variable appears. If you find that variables X1, X2, and X3 appear most frequently in cluster 1, you can say that these variables are inside cluster 1. Similarly, if you find that variables X4 and X5 appear most frequently in cluster 2, you can say that these variables are inside cluster 2. [↑](#footnote-ref-9)
10. In more technical terms, a unit root exists when the autoregressive parameter of a time series is equal to one (1). In such a scenario, the time series exhibits a random walk behavior, which results in the absence of mean reversion and constant variance over time (Enders, 2010). [↑](#footnote-ref-10)
11. The minimum distance approach is an alternative selection criterion to cointegration which is based on distance and has its foundations in Gatev et al. [15]. In their study, they create an aggregate total return index for each equity, which is then standardized to the initial day of a formation period spanning 12 months. This interval is utilized to compute the Sum of Squared Euclidean Distances (SSD) across the time series they created. Following this, they recommend ordering these pairs based on the least historical SSD. [↑](#footnote-ref-11)