RISK MANAGEMENT MAGAZINE

Vol. 19, Issue 1
January – April 2024

EXCERPT

https://www.aifirm.it/rivista/progetto-editoriale/

Portfolio optimization and risk management through Hierarchical Risk Parity and Logic Learning Machine: a case study applied to the Turkish stock market

Giacomo Gaggero, Pier Giuseppe Giribone, Marco Muselli, Erenay Ünal, Damiano Verda

Portfolio optimization and risk management through Hierarchical Risk Parity and Logic Learning Machine: a case study applied to the Turkish stock market

Giacomo Gaggero (University of Genoa – Department of Economics); Pier Giuseppe Giribone (University of Genoa – Department of Economics, BPER Banca – Financial Engineering); Marco Muselli (Rulex Innovation Labs); Erenay Ünal (Rulex Innovation Labs, Data Science Intern); Damiano Verda (Rulex Innovation Labs)

Corresponding Authors: Pier Giuseppe Giribone (piergiuseppe.giribone@bper.it) and Damiano Verda (damiano.verda@rulex.ai)

Article submitted to double-blind peer review, received on 7th March 2024 and accepted on 23rd April 2024

Abstract

This study explores an innovative approach to portfolio optimization, bridging traditional Modern Portfolio Theory (MPT) with advanced Machine Learning techniques. We start by recognizing the significance of Markowitz's model in MPT and quickly proceed to focus on the Hierarchical Risk Parity (HRP) method. HRP overcomes some of the limitations of Markowitz's model, particularly in managing complex asset correlations, by offering a more refined risk management strategy that ensures balanced risk distribution across the portfolio. The paper then introduces an innovative Machine Learning approach that employs the Logic Learning Machine (LLM) method to enhance the explainability of the Hierarchical Risk Parity strategy. Such integration is considered the core research part of the study, given that its application makes the output of the model more accessible and transparent. A case study based on the Turkish stock market has been provided as an example. The combination of traditional financial theories with modern Machine Learning tools marks a significant advancement in investment management and portfolio optimization, emphasizing the importance of clarity and ease of understanding in complex financial portfolio models.

Key Words: Hierarchical Risk Parity (HRP), Modern Portfolio Theory (MPT), Machine Learning (ML), Logic Learning Machine (LLM), Asset Allocation, Portfolio Optimization, Turkish Stock Market

JEL code: G11, C58, C63, F30

1) Introduction

The challenge of optimal asset allocation remains a fundamental pillar in the field of finance, driving investors through the uncertainties of continuously evolving financial markets. The quest for methodologies able of efficiently managing risk and maximizing returns has led to the progressive overcoming of the limitations encountered in classic theories, such as that proposed by (Markowitz, 1952), and later enriched by (Maillard *et al.*, 2010). While these theories have marked significant milestones in portfolio management development, they have revealed the need to adopt more dynamic and sophisticated approaches to address an increasingly complex and unpredictable market context.

In the current portfolio management landscape, our study starts from the premise that Hierarchical Risk Parity (HRP) allocation represents the state of the art Machine Learning technique for asset allocation. HRP, which surpasses the methodologies previously proposed by scholars like Markowitz, stands out for its ability to incorporate a more sophisticated understanding of asset correlation structures and for better adapting to the evolving market dynamics. However, despite HRP being widely recognized and used in the financial community, limitations related to its explainability, or the ability to understand and interpret the underlying decision-making processes, persist.

Our paper introduces an innovative approach aimed at overcoming these obstacles, proposing a portfolio optimization method that enhances explainability through the implementation of the Logic Learning Machine (LLM) developed through a low-code development platform. This approach leverages a rule-based Machine Learning method offering an alternative to traditional black-box models. With the LLM, we significantly move towards a clear-box approach, where portfolio allocation decisions are transparent and the underlying logics easily interpretable.

Hierarchical Risk Parity, introduced by (de Prado, 2016), has marked a turning point in portfolio risk management, offering an innovative approach beyond traditional methodologies. Using hierarchical analysis, HRP reduces portfolio volatility through a balanced risk distribution among assets, adjusting to market variations through advanced correlation analysis.

The evolution of HRP has seen significant contributions: (Raffinot, 2017) refined risk allocation by examining inter-asset correlations, improving the precision and effectiveness of HRP in risk management. (Lohre, Rother, Schafer, 2020) introduced tail-tail correlations, increasing portfolio resilience to market shocks through better management of extreme asset dependencies. (Molyboga, 2020) adapted HRP to offer more flexibility and customization, highlighting its applicability to various market contexts. Lastly, (Millea and Edalat, 2022) integrated Deep Reinforcement Learning into HRP, combining advanced Machine Learning techniques to optimize investment strategies.

Despite these notable advancements (Giudici, Polinesi and Spelta, 2022) and (Ahelegbey and Giudici, 2021), the literature has yet to deeply explore the decision-making mechanism underlying HRP, leaving explainability as a blind spot. Our study focuses on this aspect, investigating the underlying decisions in HRP risk allocation.

In the continuation of our study, we will begin by exploring the evolutionary path that led to the development of Hierarchical Risk Parity, highlighting how it represents not only the last piece but also the culmination of broader research in asset allocation; we will revise the technical mechanisms underlying HRP, outlining in detail how this methodology distinguishes itself for its ability to adapt to market dynamics. We will then introduce the algorithm of the Logic Learning Machine that has been developed using a low-code development platform. Finally, we will apply the LLM to a portfolio composed of major Turkish companies, using a historical dataset of stock prices from the Istanbul Stock Exchange to practically demonstrate the efficacy of our methodology. This application will not only highlight the capabilities of the LLM in the context of hierarchical risk allocation but will also offer valuable insights into interpreting the results in a real-market context.

Excluding this first paragraph, which includes a literary review, and the conclusions, the paper is structured into four sections. The first section briefly outlines the mathematical formulation that allows for the identification of optimal weights for each asset in a portfolio according to the traditional Markowitz approach, thus highlighting the pros and cons of this conventional method. The second section points out, among all the optimal portfolios constituting the efficient frontier, the one whose optimization is based on a risk parity allocation, to compare the results obtained with this classic approach to the modern HRP. The two formulations are described along with the operating principle of the Machine Learning technique proposed by de Prado in 2016, based on the following three steps: Hierarchical Tree Clustering, Matrix Seriation, and Recursive Bisection. Once the results are obtained, i.e., the weights to be assigned to each asset in the portfolio, the question was posed about which factors most significantly influenced the choice of these weightings. For this purpose, the low-code platform Rulex was used to efficiently implement the LLM algorithm, capable of defining in a highly intelligible manner the choice logics that led to the determination of the optimal portfolio weight configuration. The fourth section implements HRP on the same sample portfolio used for the previous analyses, but this time using the new block-based software architecture. After proving the exact equivalence of the results achieved with HRP (i.e., through the traditional Python coding approach and the low-code approach), the section concludes with a description of the Logic Learning Machine method, which allows for significant interpretability of the obtained results. The final section shows a real market case: the optimal allocation according to Hierarchical Risk Parity was studied using the stocks listed on the Istanbul Stock Exchange as an example. This choice was made because the open-source and downloadable dataset from Kaggle was used, allowing for easier scientific replicability of the results obtained in this study (for comparison, see: www.kaggle.com/datasets/gokhankesler/borsa-istanbul-turkish-stock-exchange-dataset). Not only is the optimal portfolio proposed, but also the rules summarized by the LLM that allow for a great explainability of the outputs.

2) Before the Hierarchical Risk Parity algorithm

When we deal with a portfolio optimization problem, it is impossible not to mention Harry Markowitz, universally recognized as the father of the "Modern Portfolio Theory" (MPT) and a fundamental figure in this field. Markowitz's model, presented in his influential article "Portfolio Selection" published in 1952 in "The Journal of Finance", revolutionized the approach to investment management. This model focuses on optimizing the allocation of financial resources among a variety of instruments, taking into account both expected returns and the associated risk of each investment.

The classic Markowitz model, also known as “the mean-variance portfolio”, is based on two key objectives: maximizing the expected return of the portfolio, calculated as the weighted sum of the expected returns of the assets, and minimizing the portfolio risk, expressed in terms of variance. This approach introduced the central concept of diversification, clearly demonstrating the benefits of constructing a portfolio composed of diversified assets.

In the context of the Markowitz model, the term "return" denotes the potential gain or loss that an investment can generate, while the concept of "risk" represents the degree of uncertainty or volatility associated with that investment. A fundamental pillar of MPT is the trade-off between risk and return, suggesting that investors are willing to accept higher risk in exchange for potentially higher returns. To address this trade-off, Markowitz introduced the concept of the "Efficient Frontier”, that is the curve generated by solving the minimization problem in (1), where the goal is to minimize variance, or risk, as returns vary (Markowitz, 1959).

$$\min_{\boldsymbol{w}} \ \sigma_p^2 = \boldsymbol{w}^{\boldsymbol{T}} \Sigma \boldsymbol{w}$$

$$\textbf{s.t.} \ r^T w = r_p, \ \sum_{i=1}^{N} w_i = 1, \ w_i \geq 0 \ \forall i \qquad (1)$$

The Efficient Frontier represents a series of optimal portfolios with minimal risk for a given level of return. The position along this curve depends on the individual investor's risk appetite. Risk-averse investors may prefer to position themselves in a low-risk portfolio, even if the return is more modest, while those willing to tolerate higher risk may opt for a portfolio with the potential for higher returns.

We now build a test portfolio specifically designed to provide a clear demonstration and explanation of the Hierarchical Risk Parity (HRP) algorithm working principles. The test portfolio serves not only as a proof of concept for the HRP methodology but also as a crucial step in aligning traditional programming with the capabilities of the advanced Machine Learning platform based on low-code paradigm. Furthermore, the usage of the test portfolio facilitates the subsequent step of checking the implementation of the algorithm in the logical-block oriented software, which is designed to enhance the HRP's interpretability and explainability of the algorithm. This alignment is essential to ensure that the HRP method can be effectively and transparently integrated into the Machine Learning environment, thereby leveraging the full potential of its data-driven insights for optimal asset allocation.

To highlight the differences between the Markowitz model and the Hierarchical Risk Parity (HRP) approach, we have constructed a portfolio consisting of 10 stock securities. This allows us to compare how the two methods allocate weights to diversify risk. We have selected the 10 companies with the highest market capitalization in the United States to analyze allocation strategies in real market contexts. The stocks included are Apple (AAPL), Microsoft (MSFT), Alphabet (GOOG), Amazon (AMZN), NVIDIA (NVDA), META (META), Berkshire Hathaway (BRK-B), Tesla (TSLA), Eli Lilly (LLY), and UnitedHealth Group (UNH). The historical data for these stocks was retrieved from Yahoo Finance and analyzed in software R, covering the period from January 2015 to December 2020. Table 1 provides summary statistics of the daily rate of return for each stock during the analyzed period.

Data reveals that tech stocks such as AAPL, MSFT, NVDA, and META have high volatility, indicating greater risks but also potential for substantial returns. Particularly within the tech sector, NVDA is notable for its pronounced volatility, which has recently typified the semiconductor industry. TSLA in the automotive sector also exhibits significant volatility.

In contrast, the financial and healthcare sectors, with stocks like BRK-B, LLY, and UNH, show lower volatility, signifying more stability. This contrast may attract investors looking for more predictable and stable long-term investment options.

Also, the correlation matrix reflects the relationships between the returns of the selected stocks. Looking at the color scheme of Figure 1, we can recognize that stocks from the same sector often exhibit higher correlations, as indicated by the redder shades among technology stocks like AAPL, MSFT, NVDA, and META. This is consistent with our previous analysis about the high volatility in

the technology sector, implying that these stocks might be affected by similar market factors and therefore exhibit more aligned price movements.
In contrast, sectors such as finance and healthcare, represented by BRK-B, LLY, and UNH, typically display a lower correlation with technology stocks.
The particularly low correlation between Eli Lilly and Tesla stands out, marking them as the least correlated pair in the portfolio and signaling a more pronounced risk diversification. The bluer shades denote a weaker correlation, suggesting that these stocks could respond differently to identical market events, thereby offering potential stability in a well-diversified portfolio.

Summary Statistics of Daily Rate of Return

Stocks	Mean	SD	Median	Min	Max
AAPL	0.13%	1.87%	0.09%	−12.86%	11.98%
AMZN	0.18%	1.95%	0.14%	−7.92%	14.13%
BRK-B	0.04%	1.31%	0.04%	−9.59%	11.61%
GOOG	0.09%	1.70%	0.10%	−11.10%	16.05%
LLY	0.08%	1.68%	0.06%	−10.51%	15.68%
META	0.10%	2.01%	0.11%	−18.96%	15.52%
MSFT	0.13%	1.75%	0.10%	−14.74%	14.22%
NVDA	0.26%	2.87%	0.25%	−18.76%	29.81%
TSLA	0.24%	3.47%	0.12%	−21.06%	19.89%
UNH	0.10%	1.76%	0.09%	−17.28%	12.80%

Table 1: Summary statistics of the test portfolio

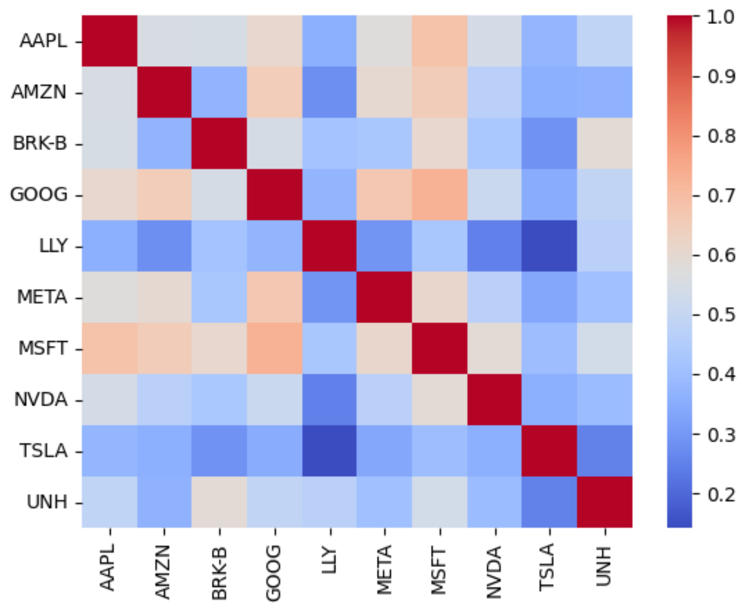


Figure 1: Correlation matrix of the test portfolio

Thus, after examining the historical trends of assets and their interrelationships, Markowitz posits that for selecting the appropriate position on the efficient frontier, grasping the investor's risk tolerance is crucial. Markowitz denotes the degree of risk aversion with lambda, λ: a higher λ value implies the investor has a strong risk aversion.
He is inclined toward more secure investments, even at the cost of missing higher potential returns.
On the other hand, a lower λ value indicates a propensity to embrace risk for the prospect of greater returns.
In the extreme case where λ equals zero, the investor is presumed to be exclusively focused on maximizing returns without any concern for the risk of substantial losses.
After reviewing the historical performance and correlations of assets, according to Markowitz, it is critical to ascertain an investor's risk tolerance to select the optimal point on the efficient frontier. This choice heavily depends on the investor's risk propensity, denoted by the parameter λ.
Once the investor's risk profile is established, the following optimization problem is addressed:

$$\max_{w} w^T r - \frac{\lambda}{2} w^T \Sigma w = r_p - \frac{\lambda}{2} \sigma_p \tag{2}$$

$$\text{s.t.} \sum_{i=1}^{N} w_i = 1\,,\ w_i \geq 0\ \forall i$$

Therefore, depending on the investor's risk preference, the construction of a portfolio will be oriented towards either higher or lower volatility. As illustrated in Figure 2, with a low λ, indicative of a risk-seeking investor, the tendency is to select a limited number of securities, typically those with the highest volatility.
Conversely, when λ is high, signifying a strong risk aversion, the goal is to form a more balanced and stable portfolio. In such a scenario, less volatile securities are favored, and the investment is spread across a wider array of assets to avoid excessive exposure to any single security and to mitigate the overall risk.

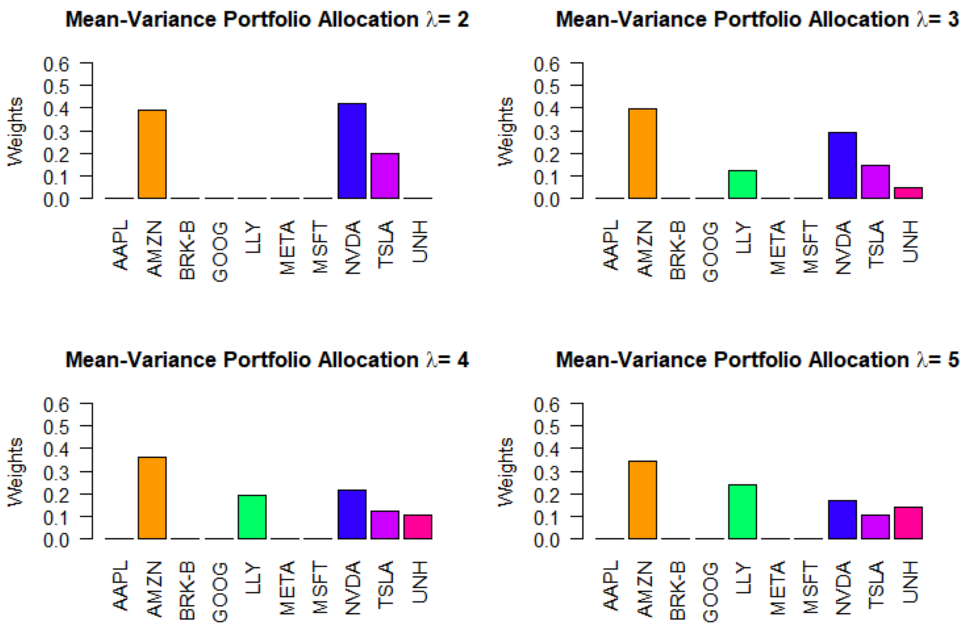


Figure 2: Portfolio allocation for the test portfolio

Since Harry Markowitz's groundbreaking work on portfolio theory, which laid the foundation for optimizing the balance between return and risk in investments, many other methods have been developed to construct portfolios that maximize returns while minimizing risk. A significant example in this context is the Maximum Sharpe Ratio Portfolio Allocation approach (Sharpe, 1966 and 1994).

3) Risk-Based Portfolio Optimization

Traditional portfolio optimization approaches, such as those proposed by Markowitz, entail certain limitations and drawbacks. Among these, the consideration of correlations between the returns of all assets in a portfolio leads to a highly complex scenario, as each asset is connected to the others, making the computational process challenging since the entire correlation matrix must be considered for the calculation. (Fabozzi et al., 2007) highlight this issue, emphasizing the complexity and computational difficulty of Markowitz's theory.

Another limitation is that not all assets in a portfolio are significantly correlated, making the consideration of all possible correlations sometimes superfluous (Vyas, 2020).
The sensitivity to estimation errors in expected returns and covariances can lead to potentially suboptimal allocation decisions (Michaud, 1989).
This model's rigidity also shows in its inability to adapt to market variations, making the allocation strategies less effective in dynamic financial contexts (Ilmanen & Kizer, 2012).
The assumption of normality of returns often does not reflect the real market dynamics, especially in the presence of abnormal behaviours like heavy tails and asymmetries in asset returns (Rocco, 2014).
The focus on minimizing variance neglects other types of risk, such as systemic or extreme event risks, a limitation pointed out by (Mandelbrot, 1963), who underscores the need to consider a distribution of returns that more accurately represents market reality.
The limitations of Markowitz's portfolio theory have led to the exploration of alternative methodologies for portfolio optimization.
We examine here the Risk Parity Portfolio and Hierarchical Risk Parity (HRP), which stand out for their ability to distribute risk more evenly, reducing reliance on expected return estimates and better managing diversification, even in volatile market conditions.
Risk Parity focuses on equalizing the risk contribution of each asset in the portfolio, while HRP utilizes hierarchical analysis to identify clusters of assets based on their correlation, thereby improving the effectiveness of risk allocation and diversification.
These methods offer a more resilient and adaptable portfolio composition, overcoming some of the criticisms associated with Markowitz's model.

3.1) Risk Parity Portfolio Allocation

The risk parity portfolio allocation method focuses on the higher risk-adjusted returns of safer assets. This means equalizing the risk allocation across assets and then overweighting safer assets and underweighting riskier assets (Asness, Frazzini and Pedersen, 2012). A safe asset indicates an asset with lower volatility, whereas a risky asset means it has high volatility. There is a difference between the Markowitz model and the risk parity approach.
The objective of Markowitz portfolios is to reach an expected return or to target ex-ante volatility, on the other hand, the risk parity approach goal is to assign a risk budget to each asset (Roncalli, 2013).
Each asset's individual risk contribution must be defined in a way that makes conceptual sense. As mentioned in the previous section, the volatility of a portfolio can be defined as follows:

$$\sigma_p = \sigma_p(w) = w^T \Sigma w \quad (3)$$

Since $\sigma_p(w)$ is homogeneous, it is possible to apply Euler's theorem for homogeneous functions,

$$\sigma(w) = \sum_{i=1}^{N} \sigma_i(w) \quad (4)$$

$$\sigma_i(w) = w_i \partial_{w_i} \sigma(w) = \frac{w_i (\Sigma w)_i}{\sqrt{w' \Sigma w}} \quad (5)$$

So, $\sigma_i(w)$ can be assumed as the contribution of asset i in the portfolio to the overall risk of the portfolio.
The goal of the risk parity portfolio is that each asset's contribution to the overall risk should be equal, therefore $\sigma_i(w) = \sigma_j(w)\ \forall\ i, j$ or in other terms, $\sigma_i(w) = \frac{\sigma(w)}{N}\ \forall\ i$ where N is the total number of assets.
This problem can be solved by looking at the fixed-point problem:

$$w_i = \frac{\sigma(w)^2}{(\Sigma w)_i N} \quad (6)$$

or by solving the minimization problem:

$$\min_{w} \sum_{i=1}^{N} \left[w_i - \frac{\sigma(w)^2}{(\Sigma w)_i N} \right]^2 \quad (7)$$

$$\textbf{s.t.} \sum_{i=1}^{N} w_i = 1,\ w_i \geq 0\ \forall i$$

As we can see from the chart in Figure 3, the more volatile and therefore riskier securities are assigned smaller weights, while those considered more stable and safer are given larger weights.
This allocation is deliberate: the aim is to balance the portfolio so that each security contributes equally to the total risk of the portfolio. Since more volatile securities are inherently riskier, even a small allocation to these securities can bring the same level of risk as a larger allocation to less volatile ones.
In this way, the strategy seeks to achieve a risk parity portfolio, where each asset contributes equally to the overall volatility of the portfolio.

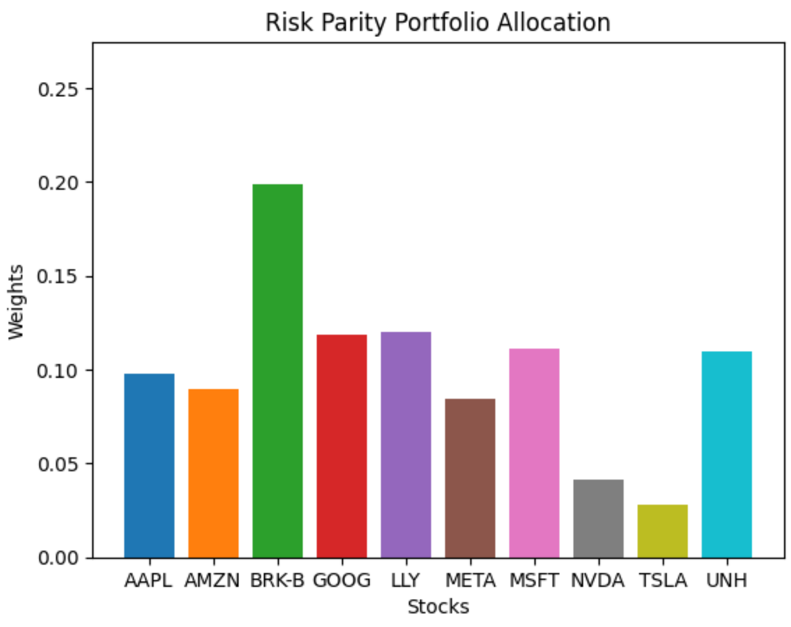


Figure 3: Risk parity allocation for the test portfolio.

3.2) Hierarchical Risk Parity Portfolio Allocation

This method addresses the issue arising from the lack of a hierarchical structure in the correlation matrix. Hierarchical Risk Parity (HRP), proposed by de Prado in 2016, takes an alternative approach.

HRP is an unsupervised Machine Learning algorithm that aims to improve the negative aspects that are typically associated with the traditional approach to portfolio optimization.
By using the hierarchical structure of clusters instead of the covariance structure, HRP fully capitalizes on the information contained within the covariance matrix and restores stability to the weights. (Burggraf, 2021).
The HRP algorithm has three major steps: Hierarchical Tree Clustering, Matrix Seriation and Recursive Bisection. We proceed to discuss these three steps in detail.

3.2.1) Hierarchical Tree Clustering

Hierarchical Tree Clustering divides investments into a hierarchical structure of clusters so that allocations can flow downstream through a tree graph. It is called Hierarchical Risk Parity because the first thing we will do is assign the financial assets into clusters (de Prado, 2016). The first step of the process is the main difference between this method and the classic ones. The first step aims to divide the assets of our portfolio into different hierarchical clusters, using the unsupervised Machine Learning algorithm known as Hierarchical Tree Clustering.
Subsequently, every cluster will be subdivided into smaller clusters, until we get down to the individual assets. This process will therefore define a hierarchy, i.e. a tree that represents the recursive process in which we divide the universe of assets into clusters that will then be divided too (Vyas, 2020).
To be more specific, we calculate the tree clusters based on the $T \times N$ matrix of stock returns where T represents the time series of the data and N represents the number of stocks in our portfolio.
We can analyse the process step-by-step to understand how the clusters are formed.

1) Given that $T \times N$ matrix of stock returns, we calculate the correlation of each stock's returns with the other stocks which gives us an $N \times N$ matrix of these correlations which is ρ.

2) The ρ matrix is converted into a distance matrix between correlations D, where each element is defined. The original formulation of D is:

$$D = \sqrt{0.5 \cdot (1 - \rho_{i,j})} \quad (8)$$

3) Subsequently, we calculate another distance matrix $\overline{D}$. In accordance with the original paper, its formulation is:

$$\overline{D}(i,j) = \sqrt{\sum_{k=1}^{N} \left(D(k,i) - D(k,j)\right)^2} \quad (9)$$

The calculation of the matrix $\overline{D}$ is basically formed by taking the Euclidean distance between all the columns in a pair-wise manner.
The matrix $D(i,j)$ represents the distance between two assets, while $\overline{D}(i,j)$ indicates the closeness in similarity of these assets with the rest of the portfolio.

4) After calculating the distance between the assets, we begin to compose asset clusters using distances recursively and denote the set of clusters by U. The first cluster i^*, j^* is calculated as (see Table 2):

$$U[1] = argmin_{(i,j)} \overline{D}(i,j) \quad (10)$$

	a	b	c	d	e
a	0	17	21	31	23
b	17	0	30	34	21
c	21	30	0	28	39
d	31	34	28	0	43
e	23	21	39	43	0

Table 2: Hierarchical Tree Clustering – Example of the working principle 1. Source: Vyas (2020)

5) At this point, we update the matrix D by calculating the distances of the other elements concerning the new cluster formed. This step is called linkage clustering and there are multiple ways of executing this process.
Hierarchical Risk Parity uses single linkage clustering which means that it evaluates the distance between two clusters that is defined by a pair of single elements of greatest proximity.
The columns and rows that correspond to the new cluster are removed: therefore, in our example, the rows and columns corresponding to assets a and b are omitted.
To calculate the distance of asset i outside this cluster, the "nearest point" method is used.

$$\overline{D}\,(i, U[1]) = min(\overline{D}(i,a), \overline{D}(i,b)) \quad (11)$$

Using the above formula (see Table 3), we calculate distances for c, d, and e from the cluster (a, b).

$$\bar{D}\,(c, U[1]) = min\big(\bar{D}(c,a), \bar{D}(c,b)\big) = \min(21{,}30) = 21$$

$$\bar{D}\,(d, U[1]) = min\big(\bar{D}(d,a), \bar{D}(d,b)\big) = \min(31{,}34) = 31$$

$$\bar{D}\,(e, U[1]) = min\big(\bar{D}(e,a), \bar{D}(e,b)\big) = \min(23{,}21) = 21$$

	(a,b)	c	d	e
(a,b)	0	21	31	21
c	21	0	28	39
d	31	28	0	43
e	21	39	43	0

Table 3: Hierarchical Tree Clustering – Example of the working principle 2. Source: Vyas (2020)

6) We proceed in this recursive way by combining assets in a cluster and updating the distance matrix until we are left with a single cluster of actions, as shown in Table 4, in which we combine d.

$$\bar{D}\,(d, U[2]) = min\left(\bar{D}\big(d,(a,b)\big), \bar{D}(d,c), \bar{D}(d,e)\right) = \min[31{,}28{,}43] = 28$$

	((a,b),c,e)	d
((a,b,)c,e)	0	28
d	28	0

Table 4: Hierarchical Tree Clustering – Example of the working principle 3. Source: Vyas (2020)

In hierarchical clustering, the clusters are always visualized in the form of a nice cluster diagram called a dendrogram. Below, Figure 4 shows the hierarchical clusters for our test portfolio.

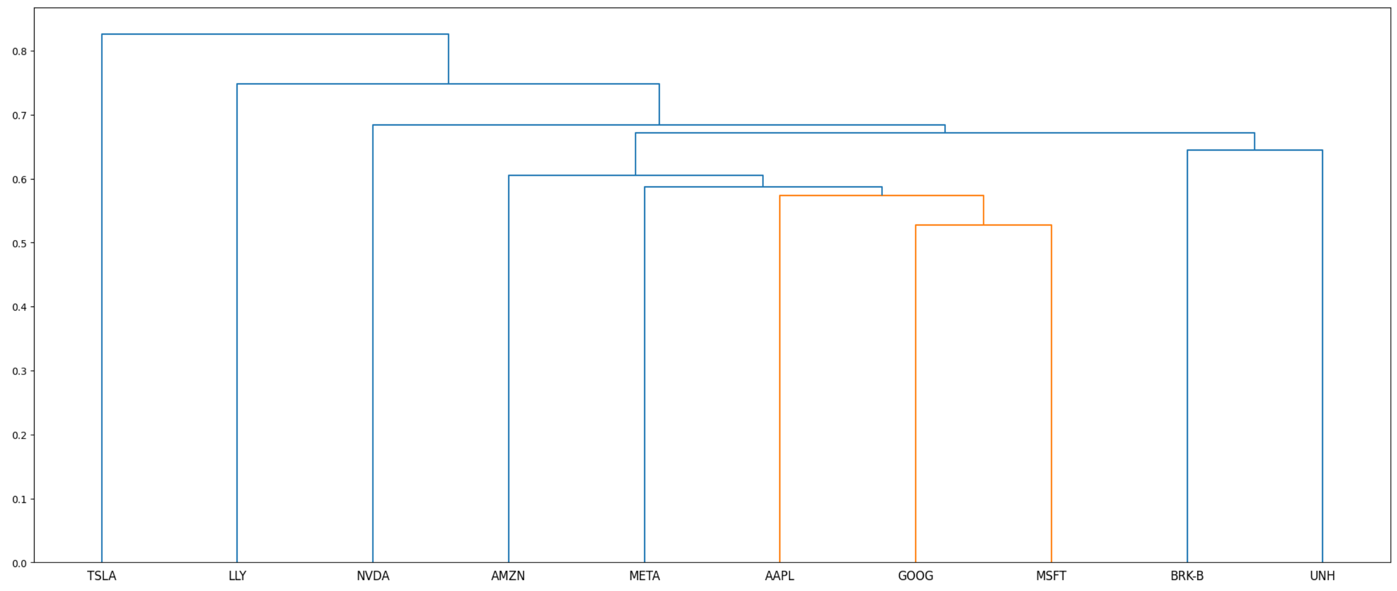


Figure 4: Dendrogram of the test portfolio

3.2.2) Matrix Seriation

This step is identified as Quasi-Diagonalization of a matrix, but in practice, it is a serialization algorithm.
The seriation of a matrix is a statistical technique, which is used to reorganize the data in order to show the inherent clusters clearly.
Using the order of the hierarchical clusters calculated in the previous step, we rearrange the rows and columns of the asset covariance matrix so that similar investments are positioned close together and dissimilar investments are positioned apart.
With this step, the original covariance matrix of the assets is reorganized so that the largest covariances are positioned along the diagonal.
The graph shows the correlation matrix of our selected portfolio after the quasi-diagonalization.
Figure 5 displays the seriated correlation matrix for our selected portfolio.
We observe that, prior to seriation, the asset clusters are broken down into small sub-sections. It is only after quasi-diagonalization that the clustering structure becomes more evident.

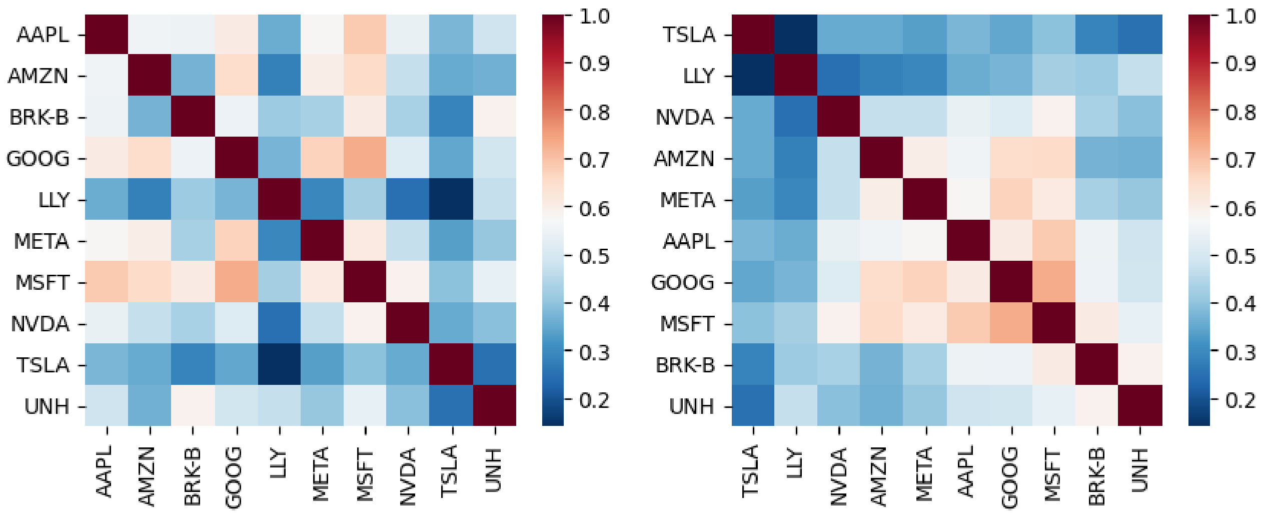


Figure 5: Quasi-Diagonalization of correlations

3.2.3) Recursive Bisection

This is the final and, in a sense, the most important step because weights are assigned to the assets in our portfolio.

1) First, we allocate the same weights for all the assets, $W_i = 1, \forall i = 1, \dots, N$

At the end of the tree clustering step, concerning clustering, there remained one maxi-cluster with subsequent clusters nested within each other. In this step with a top-down logic, we divided each cluster into two sub-clusters, starting from the highest grouping. The inverse-variance allocation is optimal for a diagonal covariance matrix. We can take advantage of this fact by allocating funds top-down through the tree structure, in which riskier clusters are given less funding (de Prado, 2016).

2) Hierarchical Tree clustering forms a binary tree in which each cluster has a left and a right child cluster L_1 and L_2. For each of these sub-clusters, we calculate its variance,

$$w = \frac{diag[V]^{-1}}{trace(diag[V]^{-1})} \quad (12)$$

Where V is the covariance matrix of the cluster members.

*3)*The variance is calculated for both clusters:

$$\tilde{V}_1 = w_1^T V_1 w_1 , \tilde{V}_2 = w_2^T V_2 w_2 \quad (13)$$

*4)*The two factors α_1 and α_2 are used to rescale the cluster weights already allocated.

$$\alpha_1 = 1 - \frac{\tilde{V}_1}{\tilde{V}_1 + \tilde{V}_2} , \alpha_2 = 1 - \alpha_1 \quad (14)$$

To understand how we reach the above formula for the weighting factor, we need to start from classic portfolio optimization theory, in which we have the following optimization objective:

$$\min \frac{1}{2} w^T \sigma w$$

subject to: (15)

$$s.t\ e^T w = 1; \quad e = 1^T$$

Here, w represents the portfolio weights and σ signifies the covariance matrix of the portfolio. Through this lens, we derive the minimum variance weights:

$$w = \frac{\sigma^{-1} e}{e^T \sigma^{-1} e} \quad (16)$$

When dealing with a diagonal σ, our weights simplify to:

$$w = \frac{\sigma_{n,n}^{-1}}{\sum_{i=1}^{N} N \sigma_{i,i}^{-1}} \quad (17)$$

Given that we are focusing on a scenario with N=2 (two subclusters), this equation refines our weights further:

$$w = \frac{\frac{1}{\sigma_1}}{\frac{1}{\sigma_1} + \frac{1}{\sigma_2}} = 1 - \frac{\sigma_1}{\sigma_1 + \sigma_2} \quad (18)$$

*5)*The sum of the final weights which are $w_1 = w_1\alpha_1$ and $w_1 = w_1\alpha_2$ should be equal to 1.

We applied HRP to our selected portfolio. Figure 6 and Table 5 show the weight allocations of the HRP portfolio.

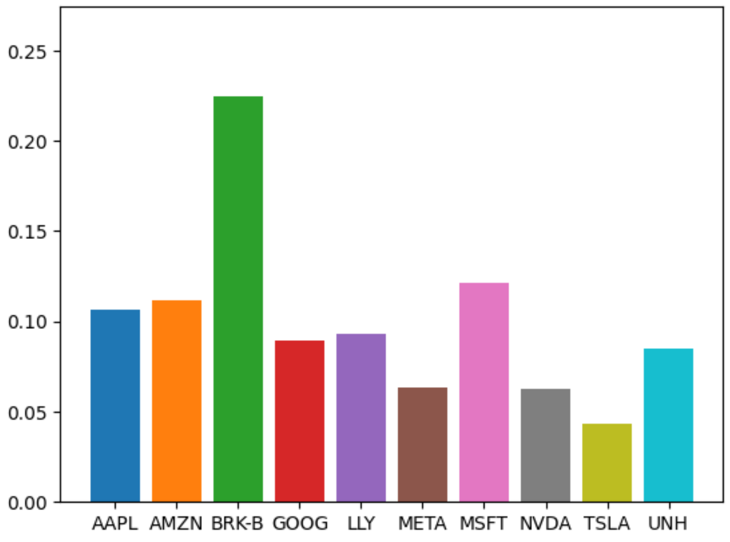


Figure 6: HRP allocation for the test portfolio

Weights of the HRP and RP Portfolios

Stocks	HRP	RP	Difference
AAPL	0.1066	0.0978	0.0088
AMZN	0.1116	0.0895	0.0222
BRK-B	0.2246	0.1987	0.0259
GOOG	0.0891	0.1185	-0.0294
LLY	0.0928	0.1204	-0.0276
META	0.0636	0.0846	-0.0210
MSFT	0.1209	0.1109	0.0100
NVDA	0.0629	0.0413	0.0216
TSLA	0.0430	0.0282	0.0148
UNH	0.0848	0.1100	-0.0252

Table 5: Differences between the HRP and RP portfolio weights allocation.

4) Implementation of the HRP algorithm in a self-coding platform

Self-coding is a method that combines the benefits of no-code platforms and custom code development, while overcoming the drawbacks of both approaches. Self-code platforms offer a user-friendly graphical interface, enabling users to construct multiple applications without writing code. This simplicity brings flexibility in dealing with specific business scenarios. The self-coding platforms autonomously generate the optimized underlying code in the background while users manipulate data through drag-and-drop build blocks and configure their tasks.

We have used the Rulex Platform for this model implementation. This software has two main parts: the Factory and the Studio, which provide a drag-and-drop interface from the initial building of logical flows. The Factory is the central component of the platform, where solutions are systematically developed in logical flows while offering the opportunity to check the underlying data at each phase. On the other hand, the Studio is the creative center of the Platform, where dashboards, reports, and executive summaries can be created. The Factory allows to examine the underlying data at each stage, eliminating the need for planning beforehand or waiting until the entire solution is completed. In fact, the visual structure of the self-coding platform allows to easily query the data in every step of the pipeline and analyze it, which ensures a better understanding and control of every step leading from the raw data to the desired outcome, even for end users who lack knowledge of programming languages. It also provides a wide range of Machine Learning algorithms, including explainable AI and advanced optimization tools. Among them, differently from any other tool, it also features the Logic Learning Machine (LLM) (Ferrari et al. 2023) classification and regression algorithm. This algorithm allows to extract a rule-based model and, unlike competing techniques such as Decision Trees, the extracted rules can be overlapping, ensuring a richer representation of the problem at hand: the scoring engine is equipped to solve conflicts among them and select a predicted outcome taking into account the contribution of each of them. In order to implement the HRP algorithm we need to follow the steps shown in the scheme (Figure 7).

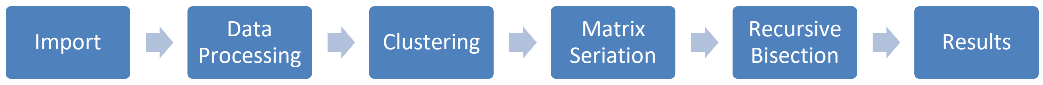


Figure 7: HRP algorithm block scheme for its implementation in the self-coding platform

The first step in implementing Hierarchical Risk Parity requires importing relevant financial data into the software. This data should include the trading date, ticker of the stock, and historical asset prices retrieved from Yahoo Finance for our selected portfolio consisting of the selected 10 stocks. Data quality is crucial in portfolio optimization. The data processing involves handling missing values, outliers, and any other inconsistencies in the dataset. The Factory's data preprocessing is possible with the Data Manager task (Figure 8). The task has the capability to facilitate the identification of the potential problem of the dataset. The Data Manager task enables various operations, including management of attributes calculation formulas, data querying, statistical calculations, and data visualization. For our HRP portfolio allocation, we need to calculate returns for each stock. Our dataset consists in the daily price of the stocks therefore calculation of the returns should be made daily. We can create a new attribute in which we can use the 'shift' function that gives us the opportunity to shift the prices one day by the group of the ticker. Then we can implement the classic percentage change variation.To calculate the statistical measure, which is covariance, we need to duplicate the existing Data Manager and subsequently link it with the original one using the Join task. Establishing an identical Data Manager is a straightforward process. To elucidate, connecting the current Data Manager with an empty one results in the replication. Following this, the Join task can be employed to merge these two identical Data Managers into a single one. After that, to have the covariances the Data Manager 'pearson' function allows us to compute the Pearson correlation coefficient which gives the same results for the ρ matrix as those estimated using traditional coding. After these operations, we can compute distance matrices to be used in the cluster loop.

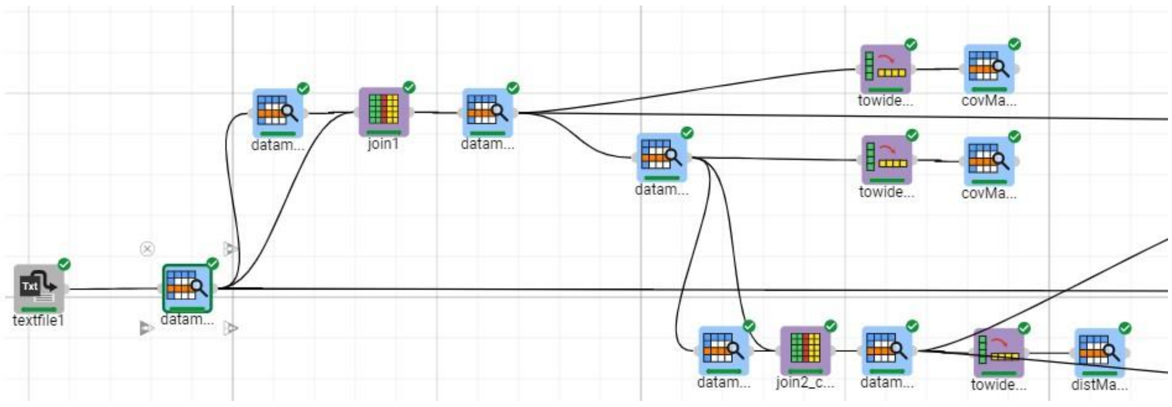


Figure 8: Data preparation flow

Clustering is a fundamental step in HRP that involves grouping assets with similar risk-return profiles. Software is able to employ advanced clustering algorithms to categorize assets based on their historical performance and correlation (Figure 9). In order to apply hierarchical tree clustering explained in part 3.2.1, another flow can be created, and this flow implements a loop for clustering. The factory allows us to embed new flows into our main flow. By identifying clusters of assets, the portfolio optimization process moves to the next step which is matrix seriation.

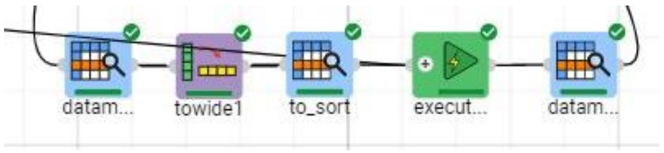


Figure 9: Clustering flow

Once the clusters are formed, there is the serialization algorithm step. In other words, the assets within each cluster are sorted based on their risk contributions. It is the same process that is demonstrated in section 3.2.2. The software provides blocks and tools for efficient sorting, allowing for the arrangement of assets within clusters in a way that aligns with the HRP methodology (Figure 10). We can build this sorting algorithm in another flow and then again embed it into our main flow.

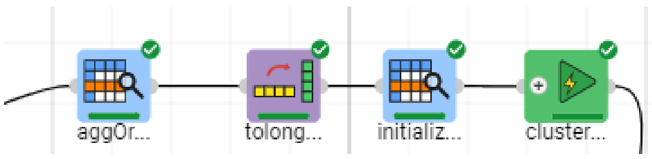


Figure 10: Matrix seriation flow

HRP portfolio employs a recursive bisection approach to build the hierarchical structure of the portfolio. The platform supports this recursive process by iteratively creating clusters and subclusters, refining the portfolio structure to achieve optimal hierarchical risk parity allocation (Figure 11). The final step of the HRP algorithm is applied, as described in section 3.2.3. This step ensures that the final portfolio is well-diversified and exhibits balanced risk contributions across assets.

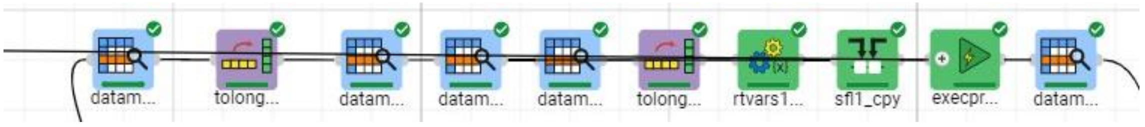


Figure 11: Recursive bisection flow

After the recursive bisection, the final output of the HRP portfolio implementation, i.e. the risk-efficient portfolio weights, can be viewed on the Data Manager. Figure 12 shows the flow of the implementation of the HRP algorithm using the blocks.

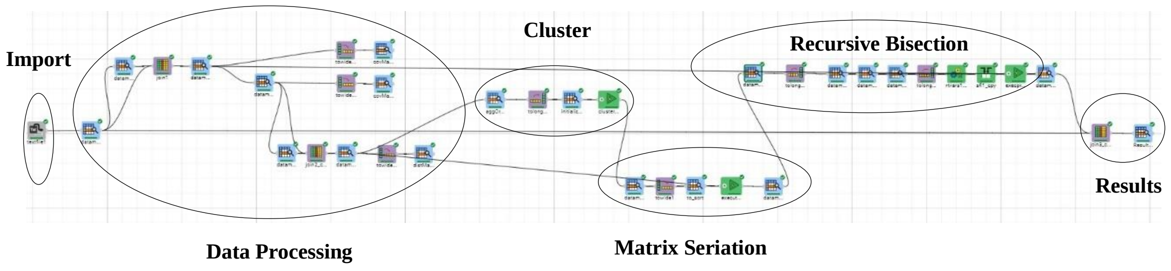


Figure 12: Hierarchical Risk Parity portfolio implementation flow

The results from the platform match those estimated using the traditional approach in Python (Figure 13).

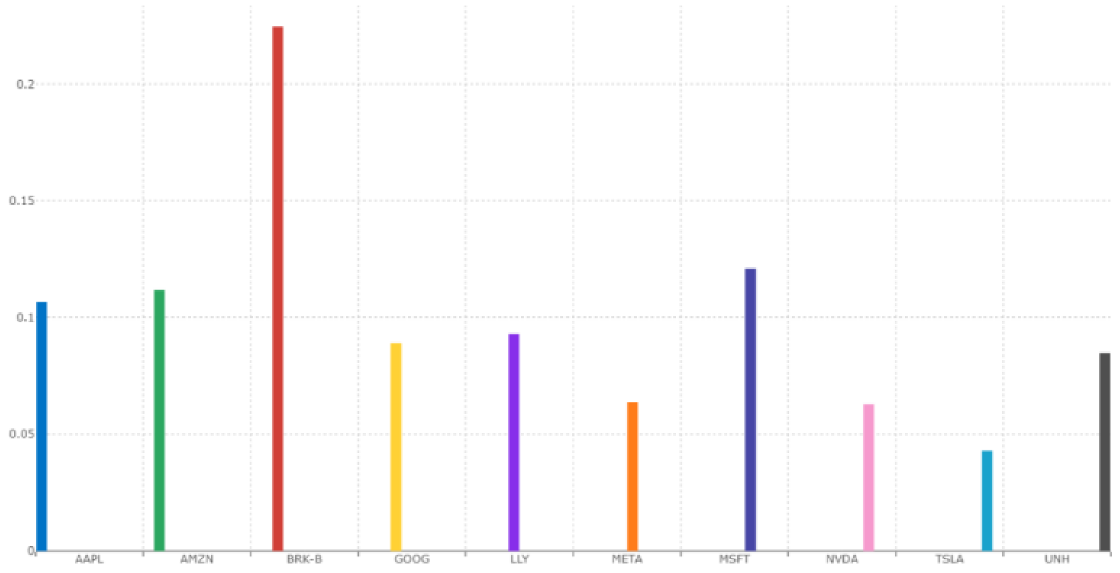


Figure 13: HRP portfolio allocation using the low-code platform

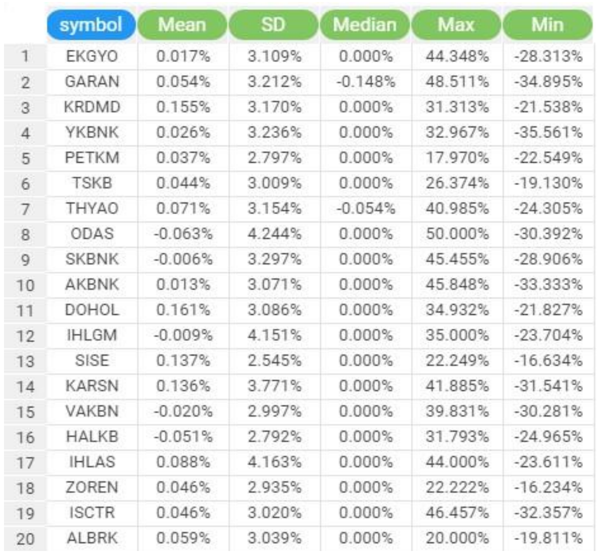
Having achieved the successful calibration of the portfolio, we can now consider its extension and, most importantly, take advantage of the Logic Learning Machine (LLM) method. This step adds significant value: whereas the explainability of the model remained constrained with the traditional approach, the integration of LLM allows us to overcome this barrier, ensuring greater transparency and understandability of the asset allocation decisions within the portfolio.

5) Case study: Logic Learning Machine application for the Turkish stock exchange market

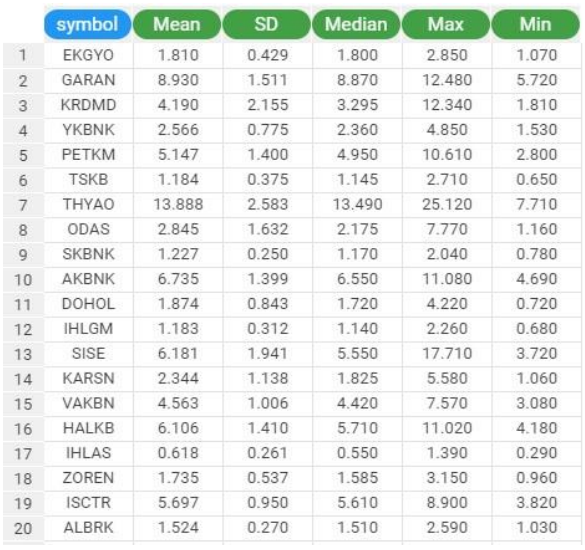
In this paragraph, the Logic Learning Machine (LLM) will be implemented to HRP portfolio allocation results from a new dataset. The new dataset was imported from Kaggle (Kesler, 2022). The dataset contains historical daily prices of the Borsa Istanbul Stock Exchange Market (BIST). BIST stands as a pivotal financial institution in Turkey, playing a crucial role in the country's economic development and providing a platform for businesses to raise capital through the issuance of stocks. The dataset includes the stock prices from 2015 until 2021. The dataset also contains detailed price information (open, close, high, and low), volume, market segment and other information about the stocks. The raw dataset had 507 stocks at the beginning. But to analyze it, each stock needed to have the same time period. In addition, some stocks were not operating on the market anymore. So, those stocks needed to be excluded from the dataset.

5.1) Descriptive Statistics of the Dataset

The total dataset consists of 507 stocks and after the data processing, 336 stocks remain in the dataset to be analyzed for the same time period. This reduction is attributed to the exclusion of stocks that were no longer operating in the market during the specified time period, ensuring the analysis is focused on actively traded stocks. Understanding the trading volume of stocks on BIST provides insights into market activity and investor sentiment. Highly liquid stocks often attract more attention from traders due to their ability to be easily bought or sold without significantly impacting the stock price. Figure 14.a shows the summary statistics of daily prices of 20 stocks which are sorted by average volume. The mean price of a stock within the top 20 highest average volume stocks represents the average closing price over the specified time period. It offers insights into the typical valuation of these high-volume securities. The standard deviation of prices for each stock in the top 20 indicates the degree of variability in their closing prices. Higher standard deviations suggest greater price volatility for these actively traded stocks. The summary statistics of daily prices indicated that all of the highly traded stocks experienced positive daily returns. But it is worth mentioning that Turkish Airlines (THYAO) has a 13.888 average return, and it is an extreme return compared to the rest of the stocks. Figure 14.b shows the summary statistics of the daily rate of return of 20 stocks which are sorted by average volume. The mean daily rate of return indicates the average percentage change in the stock price on a daily basis. It provides insights into the typical daily performance of these high-volume securities. The standard deviation of the daily rate of return measures the degree of variability in their daily price changes. Higher standard deviations indicate increased daily price volatility for these actively traded stocks. The median daily rate of return is the middle value in the distribution of daily percentage changes. It provides a robust measure that is less influenced by extreme values, offering insights into the central tendency of the stock daily performance. The maximum daily rate of return is the highest daily percentage change observed during the analyzed time period. It highlights peak daily performance and potential outliers in the rate of return for these high-volume stocks. The minimum daily rate of return is the lowest daily percentage change observed during the specified time period. It provides insights into the lowest daily performance of these stocks, aiding in risk assessment. The summary statistics of the daily rate of return showed that highly traded stocks performed mostly positive daily returns. Another fact about the daily return of these 20 stocks is that they ranged from 2 to 4 percent.



	symbol	Mean	SD	Median	Max	Min
1	EKGYO	0.017%	3.109%	0.000%	44.348%	-28.313%
2	GARAN	0.054%	3.212%	-0.148%	48.511%	-34.895%
3	KRDMD	0.155%	3.170%	0.000%	31.313%	-21.538%
4	YKBNK	0.026%	3.236%	0.000%	32.967%	-35.561%
5	PETKM	0.037%	2.797%	0.000%	17.970%	-22.549%
6	TSKB	0.044%	3.009%	0.000%	26.374%	-19.130%
7	THYAO	0.071%	3.154%	-0.054%	40.985%	-24.305%
8	ODAS	-0.063%	4.244%	0.000%	50.000%	-30.392%
9	SKBNK	-0.006%	3.297%	0.000%	45.455%	-28.906%
10	AKBNK	0.013%	3.071%	0.000%	45.848%	-33.333%
11	DOHOL	0.161%	3.086%	0.000%	34.932%	-21.827%
12	IHLGM	-0.009%	4.151%	0.000%	35.000%	-23.704%
13	SISE	0.137%	2.545%	0.000%	22.249%	-16.634%
14	KARSN	0.136%	3.771%	0.000%	41.885%	-31.541%
15	VAKBN	-0.020%	2.997%	0.000%	39.831%	-30.281%
16	HALKB	-0.051%	2.792%	0.000%	31.793%	-24.965%
17	IHLAS	0.088%	4.163%	0.000%	44.000%	-23.611%
18	ZOREN	0.046%	2.935%	0.000%	22.222%	-16.234%
19	ISCTR	0.046%	3.020%	0.000%	46.457%	-32.357%
20	ALBRK	0.059%	3.039%	0.000%	20.000%	-19.811%



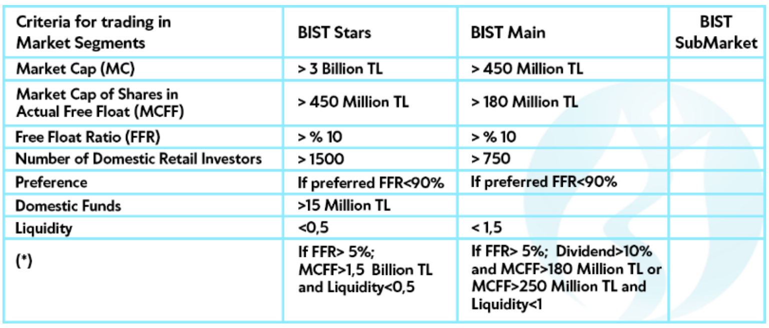
	symbol	Mean	SD	Median	Max	Min
1	EKGYO	1.810	0.429	1.800	2.850	1.070
2	GARAN	8.930	1.511	8.870	12.480	5.720
3	KRDMD	4.190	2.155	3.295	12.340	1.810
4	YKBNK	2.566	0.775	2.360	4.850	1.530
5	PETKM	5.147	1.400	4.950	10.610	2.800
6	TSKB	1.184	0.375	1.145	2.710	0.650
7	THYAO	13.888	2.583	13.490	25.120	7.710
8	ODAS	2.845	1.632	2.175	7.770	1.160
9	SKBNK	1.227	0.250	1.170	2.040	0.780
10	AKBNK	6.735	1.399	6.550	11.080	4.690
11	DOHOL	1.874	0.843	1.720	4.220	0.720
12	IHLGM	1.183	0.312	1.140	2.260	0.680
13	SISE	6.181	1.941	5.550	17.710	3.720
14	KARSN	2.344	1.138	1.825	5.580	1.060
15	VAKBN	4.563	1.006	4.420	7.570	3.080
16	HALKB	6.106	1.410	5.710	11.020	4.180
17	IHLAS	0.618	0.261	0.550	1.390	0.290
18	ZOREN	1.735	0.537	1.585	3.150	0.960
19	ISCTR	5.697	0.950	5.610	8.900	3.820
20	ALBRK	1.524	0.270	1.510	2.590	1.030

Figure 14: Summary statistics of daily prices (a) and returns (b) of 20 stocks from 2018 to 2021

5.2) Characteristic Features of the Stocks

Understanding the characteristics of BIST stocks is the core of the case study. By employing advanced methodologies to the HRP portfolio allocation results, specifically LLM, we aim to uncover concealed patterns within these characteristics. Thus, LLM will pave the way for the understanding of HRP portfolio allocation results in the dynamic environment of Borsa Istanbul. Some characteristics of stocks have been selected in order to apply LLM to the HRP portfolio allocation results. These characteristics are listed below:

- The *market segment of the stock* on Borsa Istanbul provides a detailed classification within the exchange, indicating whether the stock falls into categories. The segments are BIST Stars, BIST Main, BIST Sub-market, Watchlist (WL), Structured Products and Funds Market (SPFM), Commodity Market, Venture Capital Market, or the Pre-Market Trading Platform (PMTP). This segmentation is crucial for understanding the stock's positioning and dynamics within the broader market structure. In our analysis, we choose the stocks that are listed in BIST Stars, BIST Main, and BIST Sub-market, to be more specific. The conditions are listed in the Market Segments shown in Figure 15.



Criteria for trading in Market Segments	BIST Stars	BIST Main	BIST SubMarket
Market Cap (MC)	> 3 Billion TL	> 450 Million TL	
Market Cap of Shares in Actual Free Float (MCFF)	> 450 Million TL	> 180 Million TL	
Free Float Ratio (FFR)	> % 10	> % 10	
Number of Domestic Retail Investors	> 1500	> 750	
Preference	If preferred FFR<90%	If preferred FFR<90%	
Domestic Funds	>15 Million TL		
Liquidity	<0,5	< 1,5	
(*)	If FFR> 5%; MCFF>1,5 Billion TL and Liquidity<0,5	If FFR> 5%; Dividend>10% and MCFF>180 Million TL or MCFF>250 Million TL and Liquidity<1	

Figure 15: BIST market segment conditions. Source: BIST

- The *main sector of the stock*, as categorized on the Public Disclosure Platform in Turkey (KAP), represents the primary industry or business sector in which the company operates. This classification provides essential information about the core activities of the company, such as finance, technology, or manufacturing. The main sector classification allows investors to assess industry-specific challenges and opportunities in stocks.

- *Indexes* could be another explanatory attribute for stocks. Indexes are determined by their market value by regulatory authorities. Being part of these indices enhances the stock's visibility among investors. It means that the stock is among the top-performing companies in terms of market capitalization and liquidity, making it more attractive to a broader range of investors. BIST30 comprises the top 30 performing stocks, BIST50 includes the top 50, and BIST100 covers the top 100 stocks listed on the market. The BIST100 index serves as the primary benchmark for the Borsa Istanbul Equity Market. It comprises 100 stocks chosen from companies traded on the Stars Market.

- *Underlying Assets in the Derivatives Market*. Examining whether a stock is actively traded in the futures and options market can be another characteristic to consider for the portfolio. The availability of futures and options contracts enhances the overall liquidity of the stock. This increased liquidity can attract a broader spectrum of investors and contribute to smoother price formation. Another fact is that the availability of futures and options contracts allows market participants to protect their portfolios against adverse price movements. The futures and options of the stocks are not widely available in Turkey. There are 45 stocks in our dataset that are traded in the futures and options market.

- *Average Return and Standard Deviation*. The average return of the stock provides a quantitative measure of its historical performance. Stocks with positive average returns are generally viewed favourably, indicating that investors have experienced gains over the period but the risk of the stocks should be considered. Moreover, considering the risk of a stock, the standard deviation of daily returns assesses the stock's volatility. Higher standard deviation values indicate greater volatility, reflecting larger price fluctuations and increased risk. The average return and standard deviations of return of the stock are calculated on a daily basis from the beginning of 2018 to the end of 2021.

- The *average standard deviation ratio* is derived from the proportion of the average return to the standard deviation of the average return. The average standard deviation ratio quantifies the stock's risk relative to its average returns. A higher ratio may suggest a more favourable trade-off between potential return and risk, indicating that the investment may be more attractive in terms of its risk-adjusted performance. This metric is essential for investors employing risk-sensitive strategies, allowing them to assess the trade-off between potential returns and associated risks.

- The *Average volume* serves as a key indicator of market liquidity, representing the average number of shares traded daily. Investors often use average volume in their strategies for evaluating stock volatility. An increase in volume during periods of price volatility may indicate heightened market interest and the potential for more pronounced price movements. Another fact is that higher average volumes mean increased liquidity, making the stock more accessible to trade. The average volume is calculated on a daily basis for the time period from 2018 to 2021 for our portfolio.

- *Float Ratio of the Stock*. Floating stock means the number of shares available for trading of a particular stock. The float ratio of the stock measures the proportion of shares available for trading in the open market. A higher float ratio generally indicates greater liquidity, making the stock more attractive to investors. For our analysis, the float ratio of the stocks is the ratio on 31 December 2021 which is the last day of trading in our dataset. The float ratio information is retrieved from the IS Investment.

- The *foreign ratio of the stock* represents the percentage of shares owned by foreign investors. A higher foreign ratio may indicate increased exposure to international market dynamics. Stocks with a substantial foreign ratio may exhibit sensitivity to currency fluctuations and exchange rate movements. In our dataset foreign ratio of the stocks is the ratio on 31 December 2021 which is the last day of trading in our dataset. The foreign ratio information is retrieved from the IS Investment.

In short, looking into the characteristics of the stocks, we examined crucial attributes such as main sector classifications, index inclusions, derivatives trading involvement, historical returns, volatility metrics, liquidity indicators, and foreign ownership. These features will be the foundation for the next step of the case study which is the application of the Logic Learning Machine.

5.3) Logic Learning Machine

The Logic Learning Machine (LLM) is a rule-based method alternative to decision trees. In plain words, the LLM transforms the data into a Boolean domain where some Boolean functions (namely one for each output value) are reconstructed starting from a portion of their truth table with a method that is described in the paper of Muselli and Ferrari (2009).
The method creates a set of intelligible rules through Boolean function synthesis following 4 steps. These steps are:

1. Discretization
2. Latticization or Binarization
3. Positive Boolean function
4. Rule generation

In order to analyze the process, each step is applied to a bi-class toy problem, whose training set is demonstrated in Table 6.

X_1	0.07	0.11	0.22	0.14	0.23	0.08	0.12	0.21	0.26	0.24
X_2	a	a	c	a	d	c	d	a	a	b
Y	C_0	C_0	C_0	C_0	C_0	C_1	C_1	C_1	C_1	C_1

Table 6: Toy example for describing the LLM working principle

5.3.1) Discretization

In this step, each continuous variable domain is converted into a discrete domain by a mapping. $\psi_j : X_j \rightarrow I_M$ where X_j is the domain of the j-th variable and $I_M = 1, \dots, M$ is the set of positive integers up to M.
The mapping must preserve the ordering of the data. If $x_{ij} \le x_{kj}$ then $\psi_j(\boldsymbol{x}_i) \le \psi_j(\boldsymbol{x}_k)$, $\forall i = 1, \dots, d$. One way to describe ψ_j is that it consists of a vector $\boldsymbol{\gamma}_j = (\gamma_1, \dots, \gamma_{m-1})$ so that:

$$\psi_j(\boldsymbol{x}_i) = \begin{cases} 1, & x_{ij} < \gamma_{i1} \\ m, & \gamma_{jm-1} < x_{ij} < \gamma_{jm} \\ M_j, & x_{ij} > \gamma_{iM-1} \end{cases} \quad (19)$$

There are several strategies for discretization and the simplest one is creating M_j interval having the same length. Let $\boldsymbol{\rho}_j$ be the vector of all the α_j values for input variable j in ascending order $\left(p_{jl} \le p_{jl+1}\ \forall\ l = 1, \dots, \alpha_j\right)$, then the cutoff γ_{jm} is given by:

$$\gamma_{jm} = p_{j1} + \frac{p_{j\alpha_j} - p_{j1}}{M_j} m \ (20)$$

This method is referred to as Equal Width discretization. An alternative straightforward approach, known as Equal Frequency discretization, involves selecting cutoff points to ensure an equal distribution of examples from the training set S in each interval. However, these two discretization approaches are unsuitable for a classification problem, since they disregard the output value, whereas it is desired that: if $y_i \ne y_k$ then $\exists\ j$ **s.t.** $\psi_j(\boldsymbol{x}_i) \ne \psi_j(\boldsymbol{x}_k)$; otherwise, discretization brings on ambiguity in the training set.
The problem of determining cutoffs can be approached as an optimization problem, aiming to minimize the total number of cutoffs while considering the constraint introduced in the preceding equation. The cutoffs can be chosen among the values. In order to formalize the problem, τ_{jl} and X_{jik} for $j \in \{1, \dots, d\}$ and $i, k \in \{1, \dots, N\}$ and $l \in \{1, \dots, \alpha_j\}$ must be introduced. X_{jik} is the set of indexes l such that $x_{ij} < \rho_{jl} < x_{kj}$ and τ_{jl} are Boolean values that assert if ρ_{jl} is a cutoff j-th variable:

$$\tau_{jl} = \begin{cases} 1 & \text{if } \rho_{jl} \in \gamma_j \\ 0 & \text{otherwise} \end{cases} \ (21)$$

then the optimization problem is given by:

$$\min_{\boldsymbol{\tau}} \sum_{j=1}^{d} \sum_{l=1}^{\alpha_j} \tau_{jl}$$

with: (22)

$$\sum_{j=1}^{d} \sum_{l \in X_{jik}} \tau_{jl} \ge 1\ \forall\ i, k \quad \textbf{s.t. } i \ne k$$

If there are ambiguities in the training set, discretization must be able to minimize ambiguity and the number of cutoffs. The solution is to relax constraints by introducing a set of auxiliary variables, that control the violation of the starting constraints.

On the other hand, constraints may be enforced by imposing that examples belonging to the different classes are separated by at least $q > 1$ different variables. In any case, the problem stated above presents several variables and it could require excessive computational cost.

In order to find a near-optimal solution, the possible practical approach is to use a greedy approach method, such as the Attribute Driven Incremental Discretization (ADID) (Ferrari and Muselli, 2010; Cangelosi *et al.*, 2013). ADID uses the concept of effective distance d_{eff} that is the number of inputs that separate two examples:

$$d_{eff}(\mathbf{x}_i, \mathbf{x}_k) = \textstyle\sum_{j=1}^{d} \theta\left(|\ \psi(\mathbf{x}_i) - \psi(\mathbf{x}_k)\ |\right)$$

(23)

$$\text{where } \theta(u) = \begin{cases} 1 & \text{if } u > 0 \\ 0 & \text{otherwise} \end{cases}$$

given the effective distance, then two examples $\mathbf{x}_i, \mathbf{x}_k$ are considered separated in a determinant way if $d_{eff}(\mathbf{x}_i, \mathbf{x}_k) \ge q$. For each candidate cutoff, it is possible to define:

- The sets $S_{jl}^{(l)}, S_{jl}^{(r)}$ including examples separated by the cutoff ρ_{jl}

$S_{jl}^{(l)} = \{\mathbf{x}_i \mid (\mathbf{x}_i, y_i) \in S, x_{ij} \leq \rho_{jl}\}$ (24)

$S_{jl}^{(r)} = \{\mathbf{x}_i \mid (\mathbf{x}_i, y_i) \in S, x_{ij} > \rho_{jl}\}$ (25)

- The counter v_{ij} that counts the number of examples separated in a determinant way by ρ_{ij}

$v_{jl} = \sum_{\mathbf{x}_i \in S_{jl}^{(l)}} \sum_{\mathbf{x}_k \in S_{jl}^{(r)}} |y_i - y_k| \; \theta(q - d_{eff}(\mathbf{x}_i, \mathbf{x}_k))$ (26)

- The value w_{ij} that measures the total effective distance between examples belonging to different classes that are separated in a determinant way by ρ_{ij}

$w_{ij} = \sum_{\mathbf{x}_i \in S_{jl}^{(l)}} \sum_{\mathbf{x}_k \in S_{jl}^{(r)}} |y_i - y_k| \; \theta(q - d_{eff}(\mathbf{x}_i, \mathbf{x}_k)) \; d_{eff}(\mathbf{x}_i, \mathbf{x}_k)$ (27)

By using these definitions, ADID starts from an empty set, and at each step, it adds the cutoff that maximizes $v_{jl}$and in suborder that minimizes w_{jl}, v_{jl} and w_{jl} are recalculated according to the added cutoff and the process continues until $v_{jl} = 0 \; \forall \; \rho_{jl}$.
Once the cutoffs have been selected from $\boldsymbol{\rho}_j$, their positions can be refined by applying the midpoint between the two boundary examples as shown in the following equation.

$$\gamma_{jl} = \rho_{jl} \longrightarrow \gamma_{jl} = \frac{\rho_{jl} + \rho_{jl+1}}{2} \quad (28)$$

Prior to applying binarization, it is necessary to convert nominal attributes. In this instance, the transformation is simpler compared to continuous attributes, since it is sufficient to assign an integer to each possible nominal value.
For the example in Table 7 the mappings ψ_1 and ψ_2 found at this phase are the following:

$$\psi_1(\mathbf{x}_i) = \begin{cases} 1 & \text{if } x_{i1} \leq 0.17 \\ 0 & \text{if } x_{i1} > 0.17 \end{cases}; \; \psi_2(\mathbf{x}_i) = \begin{cases} 1 & \text{if } x_{i2} = a \\ 2 & \text{if } x_{i2} = b \\ 3 & \text{if } x_{i2} = c \\ 4 & \text{if } x_{i2} = d \end{cases} \quad (29)$$

At the end of this step, the output must also be codified with 0 and 1. The resulting discretized dataset for the toy problem is shown in Table 7.

X_1	X_2	Y
1	1	0
1	1	0
2	3	0
1	1	0
2	4	0

X_1	X_2	Y
1	3	1
1	4	1
2	1	1
2	1	1
2	2	1

Table 7: Toy Example discretized. Source: Muselli (2006)

5.3.2) Binarization

In this step each discretized domain is transformed into a binary domain through a mapping $\varphi : I_{M_j} \rightarrow \{0,1\}^{M_j}$, where I_{M_j} is the domain of the j-th variable and $\{0,1\}^{M_j}$ is a string having a bit for each possible value in I_{M_j}.
The mapping must maintain the ordering of data: $u < v$ if and only if $\varphi(u) < \varphi(v)$ where the standard ordering between z and $\boldsymbol{w} \in \{0,1\}^{M_j}$ is defined as follows:

$$\mathbf{z} < \mathbf{w} \text{ if and only if } \begin{cases} \exists \, j & \text{such that } z_j < w_j \\ \forall l \neq j & z_j \leq w_j \end{cases}$$

$$(30)$$

$$\mathbf{z} \leq \mathbf{w} \text{ if and only if } z_j \leq w_j \; \forall \, j = 1, \dots, M$$

if the relation in the equation above holds, then it is said that **z** covers **w**.

A suitable choice for φ_j is the inverse only coding, that for each $k \in I_M$ creates a string $\mathbf{h} \in \{0,1\}^{M_j}$ having all bits equal to 1 except the k-th bit which is set to 0.
For example, let $x_{ij} = 3$ with domain I_5, then $\varphi_j(\mathbf{x}_i) = 11011$. In this way $\varphi(\mathbf{x}_i) = \mathbf{z}_i$ where $\mathbf{z}_i$ is obtained by concatenating $\varphi_j(\mathbf{x})$ for $j = 1, \dots, d$.
As a result, the new training set is $S' = \{(\mathbf{z}_i, y_i)\}_{i=1}^{N}$, with $\mathbf{z}_i \in \{0,1\}^B$ where $B = \sum_{j=1}^{d} M_j$. The training set obtained after binarization for the toy problem is shown in Table 8.

Z	Y
01 0111	0
01 0111	0
10 1101	0
01 0111	0
10 1110	0

Z	Y
01 1101	1
01 1110	1
10 0111	1
10 0111	1
10 1011	1

Table 8: Toy Example after binarization. Source: Muselli (2006)

5.3.3) Synthesis of the Boolean function

The training set S', obtained after binarization, can be divided into different subsets according to the output class: T is the set containing $(\mathbf{z}_i, y_i)$ with $y_i = 1$ whereas F is the set containing the example for which $y_i = 0$.
T and F can be viewed as a portion of the truth table of a Boolean function f that must be reconstructed. Before proceeding with the method description, it is useful to give some definitions and notations.

- Each Boolean function can be written with operators AND, OR, and NOT that constitute the Boolean algebra; if NOT is not considered then a simpler structure, called Boolean lattice, is obtained. From now on, only the Boolean lattice is considered. It can be drawn by positioning $\mathbf{z}$ over $\mathbf{w}$ if $\mathbf{z} > \mathbf{w}$ and by linking all the couples $\mathbf{z}$, $\mathbf{w}$ for which an $\mathbf{a}$ exists such that $\mathbf{w} < \mathbf{a} < \mathbf{z}$. An example for $\{0,1\}^3$ is shown in Figure 16.b.

- The sum (OR) and product (AND) of m terms can be denoted as follows:

$$\bigvee_{j=1}^{m} z_j = z_1 + z_2 + \cdots + z_m = z_1 \text{ OR } z_2 \text{ OR} \ldots \text{OR } z_m$$

$$\bigwedge_{j=1}^{m} z_j = z_1 \cdot z_2 \cdot \ldots \cdot z_m = z_1 z_2 \ldots z_m = z_1 \text{ AND } z_2 \text{ AND} \ldots \text{ AND } z_m$$

- A logical product is called an implicant of a function f if the following relation holds: $\bigwedge_{j=1}^{m} z_j \leq f$, where each element z_j is called literal. The product is called prime implicant if the relation no longer holds when a literal is removed from the implicant.

- The ordering in a Boolean lattice is defined by the equations above, according to this ordering, a Boolean function $f\colon \{0,1\}^B \rightarrow \{0,1\}$ is called positive if $\mathbf{z} \leq \mathbf{w}$ implies $f(\mathbf{z}) \leq f(\boldsymbol{w})$ for each $\mathbf{z}, \mathbf{w} \in \{0,1\}^B$.

- A subset $A \subset I_B$ such that for each element $\mathbf{z}, \mathbf{w} \in A$, an ordering cannot be established (neither $\mathbf{z} < \mathbf{w}$, nor $\mathbf{w} < \mathbf{z}$), then A is called antichain.

- Given $\mathbf{a} \in \{0,1\}^B$, then the set $L(\mathbf{a}) = \{\mathbf{z} \in \{0,1\}^B \mid \mathbf{z} \leq \mathbf{a}\}$ is called lower shadow of $\boldsymbol{a}$**,** whereas the set $U(\mathbf{a}) = \{\mathbf{z} \in \{0,1\}^B \mid \mathbf{z} \geq \mathbf{a}\}$ is called an upper shadow of $\boldsymbol{a}$. The lower and upper shadows for $101 \in \{0,1\}^3$ are shown in Figures 16.a and 16.c.

- Given the subset $T, F \in \{0,1\}^B$, then T is lower separated from F, if there are no element $\mathbf{z} \in T$ belonging to the lower shadow of some element of F.

- Given the binary string $\mathbf{a}$, if there is a $\mathbf{z} \in T$ such that $\mathbf{a} \leq \mathbf{z}$ there is not a $\mathbf{w} \in F$ such that $\mathbf{a} \leq \mathbf{w}$ and for each $\mathbf{b} < \mathbf{a}$, there is $\mathbf{w} \in F$ such that $\mathbf{b} \leq \mathbf{w}$, then $\mathbf{a}$ is called bottom point for the pair (T, F).

- Every positive Boolean function can be written in its unique, not redundant Positive Disjunctive Normal Form (PDNF) as the sum of its prime implicants: $f(\mathbf{z}) = \bigvee_{a \in A} \bigwedge_{j \in P(\mathbf{a})} z_j$, where $P(\mathbf{a})$ is the subset I_B containing each i such that $a_i = 1$; A is an antichain of $\{0,1\}^B$ and each $\mathbf{a}$ is called the minimum true point.

For example, the not redundant PDNF $f(\mathbf{z}) = z_1 z_2 + z_4$ is obtained from antichain $A = \{1010, 0001\}$.

From these definitions, it follows that a method for finding f must retrieve the set of minimum true points to be used from T and F, in order to represent f in its irredundant PDNF and it follows that the set of all bottom points for (T, F) is an antichain, which elements are candidate minimum true points.

The algorithm employed by LLM to produce implicants is called Shadow Clustering (Muselli and Quarati, 2005). It generates implicants for f by the analysis of the Boolean lattice $\{0,1\}^B$.
The algorithm selects a node in the diagram and generates bottom points (T, F) by descending the diagram: the move down from a node to another node is equivalent to changing a component from 1 to 0 and a bottom point is added to A when any further move down leads to a node belonging to the lower shadow of some $\mathbf{w} \in F$.

In particular, the starting node is chosen between the $\mathbf{z} \in T \subset \{0,1\}^B$ that do not cover any point $\mathbf{a} \in A$ such that $\mathbf{a} \leq \mathbf{z}$ (in other words the algorithm ends when each element in T covers at least one element in A).
Once A has been found, it is possible that it contains redundant elements and consequently, it must be simplified in order to find A^* to which derives PDNF of the positive Boolean function.

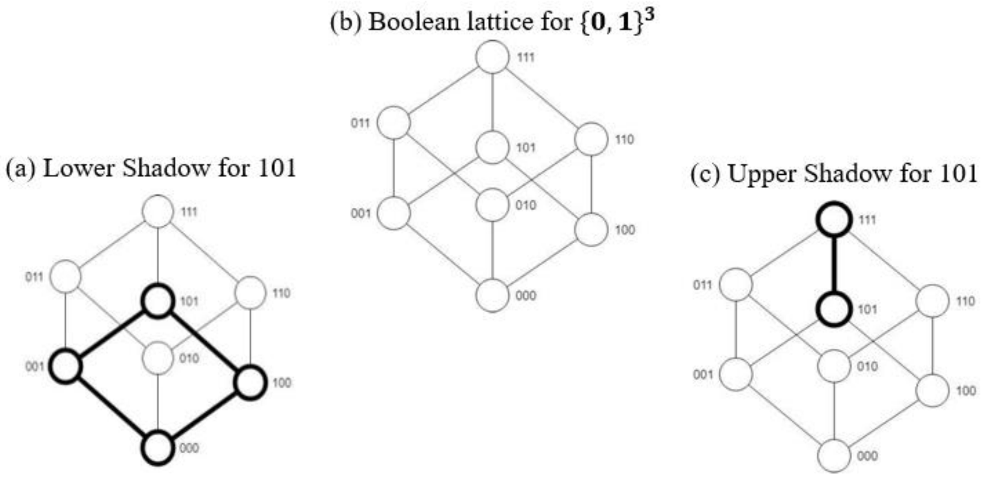


Figure 16: Boolean Lattice diagram. Source: Muselli and Ferrari (2009)

Different versions of Shadow Clustering exist depending on the choice of the element to be switched from 1 to 0 at each step of the diagram descent, for example, Exhaustive Shadow Clustering (ESC) retrieves all the bottom points deriving from the starting node. Since ESC has a high computational cost, it is usually better to use greedy approaches that at each step change the i-th element of the examined node in order to minimize the complexity of the final function, which is measured in terms of the number of implicants and number of literals in each implicant.
For example, Maximum-covering Shadow Clustering (MSC) at each step changes the element that maximizes the potential covering associated with the element index, where the potential covering associated with switching on index i from 1 to 0, is defined as the number of elements $\mathbf{z} \in T$ for which $z_i = 0$.
As concerns the selection of $A^* \subset A$, a possible choice is to subsequently add to A^* the element of A that covers the highest number of points in T that are not covered by any other element of A^*. The application of the Shadow Clustering algorithm to the dataset after binarization produces the implicants:
100011 which corresponds to $z_1 \wedge z_5 \wedge z_6$

011100 which corresponds to $z_2 \wedge z_3 \wedge z_4$

Then, the PDNF of the resulting Boolean function is the following: $f(\mathbf{z}) = (z_1 \wedge z_5 \wedge z_6) \vee (z_2 \wedge z_3 \wedge z_4)$

Algorithm 1: Shadow Clustering algorithm

```
Data: T
$Z = T$;
$A = \emptyset$
while $A \neq \emptyset$ do
    choose $\mathbf{z} \in Z$
    add to $A$ the bottom points for $(T, F)$ obtained by descending from $\mathbf{z}$ the
      diagram$\{0,1\}^B$
    remove from $Z$ the elements that cover a bottom point in $A$;
end
find $A^* \subset A$ with minimal complexity such that $\mathbf{x} \in T$ covers some $\mathbf{a} \in A^*$;
build from $A^*$ the not redundant PDNF of the positive boolean function $f$
```

5.3.4) Rule generation

In the last step, each implicant of the positive Boolean function f is transformed into an intelligible rule, where, as said before, a function is generated for each output value, and then the consequent of the rules only depends on f.
The transformation takes into account the coding applied during binarization. In particular, $\mathbf{z}$ was obtained by concatenating the results of the mapping $\varphi_j(\mathbf{x})$ for each $j = 1, \dots, d$ and consequently it can be split into substring $\mathbf{h}_j$ for each attribute, whose element $z_i \in \mathbf{h}_j$ corresponds to a nominal value if X_j is nominal, whereas it corresponds to an interval if X_j is ordered.
For each implicant, a rule in IF $-$ THEN form is generated by adding a condition for each attribute X_j as follows:
- if $z_i = 0$ for each $z_i \in \mathbf{h}_j$, then no condition relative to X_j is added to the rule;

- if X_j is nominal, then a condition $X_j \in V$ is added to the rule, where V is the set of values associated with each $z_i = 0 \in \mathbf{h}_j$;

- if X_j is ordered, then a condition $X_j \in V$ is added to the rule, where V is the union of the intervals associated with each $z_i = 0 \in \mathbf{h}_j$.

For the implicant 100011 obtained in the previous step, $\mathbf{h}_1 = 10$ leads to the condition $X_1 \in (0.17, \inf)$ or $X_1 > 0.17$ and $\mathbf{h}_2 = 0011$ leads to the condition $X_2 \in \{a, b\}$. Then the rule relative to 100011 is: IF $X_1 > 0.17$ AND $X_2 \in \{a, b\}$ THEN $Y = C1$

For the implicant 011100 obtained in the step described previously, $\mathbf{h}_1 = 01$ leads to the condition $X_1 \in (-\inf, 0.17]$ or $X_1 \leq 0.17$ and $\mathbf{h}_2 = 1100$ leads to the condition $X_2 \in \{c, d\}$. Then the rule relative to 011100 is: IF $X_1 \leq 0.17$ AND $X_2 \in \{c, d\}$ THEN $Y = C1$

Notice that conventionally the upper bound, if finite, is always included in the condition, whereas the lower bound is excluded.
In order to generate the rule for the other class it is sufficient to label $C0$ with 1 and $C1$ with 0. In the case of the multiclass problem, it is sufficient to decompose the problem into several bi-class problems according to the OVR strategies described so far, for each of the sub-problems the target class is labelled with 1, and all the remaining with 0.

5.3.5) Rule quality and class prediction

The process described in part 5.3.3 ensures that each element $\mathbf{x}_i$ of the training set only satisfies rules associated with the output class of $\mathbf{x}_i$, but since data are affected by noise, usually it is preferable to admit some errors in order to make the model able to generalize. In order to permit a fraction of error, the descent of the diagram does not stop when a further move down leads to the lower shadow of some $\mathbf{w} \in F$, but allows it to go on until a further move leads to a node belonging to the lower shadow of a percentage element $\mathbf{w} \in F$ greater than a regularization parameter *maxerror*. Then, it is usual that an element of the training set covers the rule of different classes. When it happens, the output class is established according to the relevance of the rules satisfied by it.
In order to present relevance, the following quantities relative to a rule $\mathbf{r}$ in the IF $< premise >$ THEN $< consequence >$ form are introduced:

- $TP(\mathbf{r})$ is the number of training set examples that satisfy both the premise and the consequence of the rule $\mathbf{r}$,

- $FP(\mathbf{r})$ is the number of training set examples that satisfy the premise but do not satisfy the consequence of the rule $\mathbf{r}$,

- $TN(\mathbf{r})$ is the number of training set examples that do not satisfy either the premise or the consequence of the rule $\mathbf{r}$,

- $FN(\mathbf{r})$ is the number of training set examples that do not satisfy the premise and satisfy the consequence of the rule $\mathbf{r}$,

Notice that an example $\mathbf{x}_i$ satisfies the premise of the rule $\mathbf{r}$ if it satisfies all its premise conditions, whereas $\mathbf{x}_i$ does not satisfy the premise of the rule $\mathbf{r}$ if it does not satisfy at least one among its premise conditions. Combining these quantities, it is possible to compute 2 quality measures for a rule $\mathbf{r}$:

- Covering: $C(\mathbf{r}) = \frac{TP}{TP+FN}$ (31) and Error: $E(\mathbf{r}) = \frac{FP}{TN+FP}$ (32)

It is evident that the greater the covering, and more relevant the rule is; on the other hand the smaller the error, the less relevant the rule is. Then relevance of a rule $\mathbf{r}$ is obtained by combining $C(\mathbf{r})$ and $E(\mathbf{r})$: $R(\mathbf{r}) = C(\mathbf{r})(1 - E(\mathbf{r}))$.

Once the relevance of the rule is defined, it is possible to use it in order to compute a score $S(\mathbf{x}_i, c)$ for each class c that measures how likely it is that $y_i = c$:

$$S(\mathbf{x}_i, c) = \sum_{\mathbf{r} \in \mathcal{R}_c^i} R(\mathbf{r}) \quad (33)$$

with $\mathcal{R}_c^i = \{\mathbf{r} \mid \mathbf{r} \in \mathcal{R}, \mathbf{r} \leq \mathbf{x}_i, O(\mathbf{r}) = c\}$, where $\mathcal{R}$ is the complete ruleset, $\mathbf{r} \leq \mathbf{x}_i$ denotes that $\mathbf{x}_i$ satisfies the premise of the $\mathbf{r}$ and $O(\mathbf{r}) = c$ denotes the consequence of $\mathbf{r}$ predict class c. Then $\mathcal{R}_c^i$ is the set of rules satisfied by $\mathbf{x}_i$ that predict class c.
From the scores of each output class, it is possible to define the probability that $y_i = c$:

$$P(c \mid \mathbf{x}_i) = \frac{S(\mathbf{x}_i, c)}{\sum_{k \in C} S(\mathbf{x}_i, k)} \quad (34)$$

Then the selected output is the one that maximizes the output probability: $\tilde{y}_i = \max_c P(c \mid \mathbf{x}_i)$. Since it is not guaranteed that $0 \leq S(r) \leq 1$ and in some cases it is preferable to have a single very relevant rule than a large set of little relevant rules, then it is possible to compute $S(\mathbf{x}_i, c)$ in an iterative way as shown in the algorithm below.

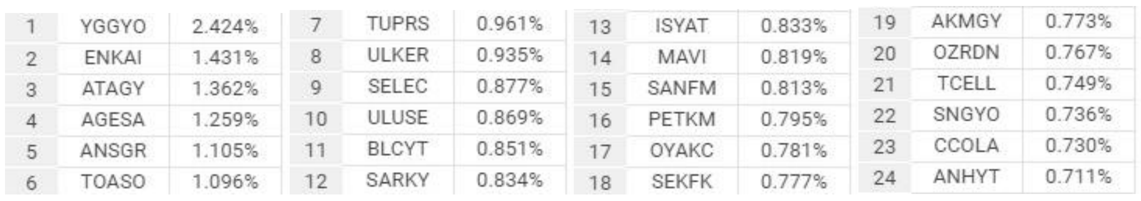
Algorithm 2: Scores computation

```
Data: $\mathcal{R}, \mathbf{x}_i$
$\mathcal{R}^* = \mathcal{R}$;
for $c \in C$ do
    $S(\mathbf{x}_i, c) = 0$;
end
while $\mathcal{R}^* \neq \emptyset$ do
    $\mathrm{r}^* = \mathrm{argmax}_{\mathbf{r}}\{R(\mathbf{r}) \mid \mathbf{r} \in \mathcal{R}^*\}$;
    $S(\mathbf{x}_i, O(\mathbf{r}^*)) = (1 - S(\mathbf{x}_i, O(\mathbf{r}^*)))R(\mathbf{r}^*)$;
    $\mathcal{R}^* = \mathcal{R}^* \setminus \{\mathbf{r}^*\}$;
end
```

By applying this algorithm, the score is incremented at each step by adding the relevance of the best rule normalized over the maximum value that can be added to the score without exceeding 1. In addition to the rule relevance, it is also possible to define the relevance of a condition r, which denotes the increase of error obtained by deleting condition r from rule **r**: $R(r) = E(\boldsymbol{r}') - E(\boldsymbol{r})$ where $\boldsymbol{r}'$ is the rule obtained by removing condition r from **r**.

5.4) Application of Hierarchical Risk Parity and Logic Learning Machine

This part delves into a comprehensive exploration of key attributes that define a stock's behavior, focusing on its characteristics which are explained in part 5.2. Before applying the LLM, results from HRP portfolio allocation for our dataset are shown in Figure 17.



1	YGGYO	2.424%	7	TUPRS	0.961%	13	ISYAT	0.833%	19	AKMGY	0.773%
2	ENKAI	1.431%	8	ULKER	0.935%	14	MAVI	0.819%	20	OZRDN	0.767%
3	ATAGY	1.362%	9	SELEC	0.877%	15	SANFM	0.813%	21	TCELL	0.749%
4	AGESA	1.259%	10	ULUSE	0.869%	16	PETKM	0.795%	22	SNGYO	0.736%
5	ANSGR	1.105%	11	BLCYT	0.851%	17	OYAKC	0.781%	23	CCOLA	0.730%
6	TOASO	1.096%	12	SARKY	0.834%	18	SEKFK	0.777%	24	ANHYT	0.711%

Figure 17: HRP allocation results for Turkish stocks (tickers versus weights)

After implementing the HRP allocation algorithm it can be merged with the characteristics explained in part 5.2. The first step in the LLM algorithm involves feature extraction from the dataset, encompassing both static characteristics features and dynamic historical performance metrics of the 336 stocks. The static features, such as market capitalization and industry sectors, offer a snapshot of the fundamental attributes of each stock. Simultaneously, the dynamic features, including historical returns and volatility, capture the temporal dynamics of the market. To facilitate supervised learning within the LLM framework, a set of labels is defined based on the performance of each stock. The first 24 stocks selected from the results which have a higher percentage than 0.7 are labelled as selected. The LLM is a symbolic Machine Learning model that combines decision trees and logic-based rule learning. It employs a hierarchical structure to create rules that capture the relationships between input features and output labels. The model's architecture allows it to generate human-understandable rules while maintaining the flexibility of traditional Machine Learning techniques. Figure 18 shows the Rule Generation workflow. The dataset is divided into training and test sets in order to train the model on the training set and perform the classification rules on the test set.

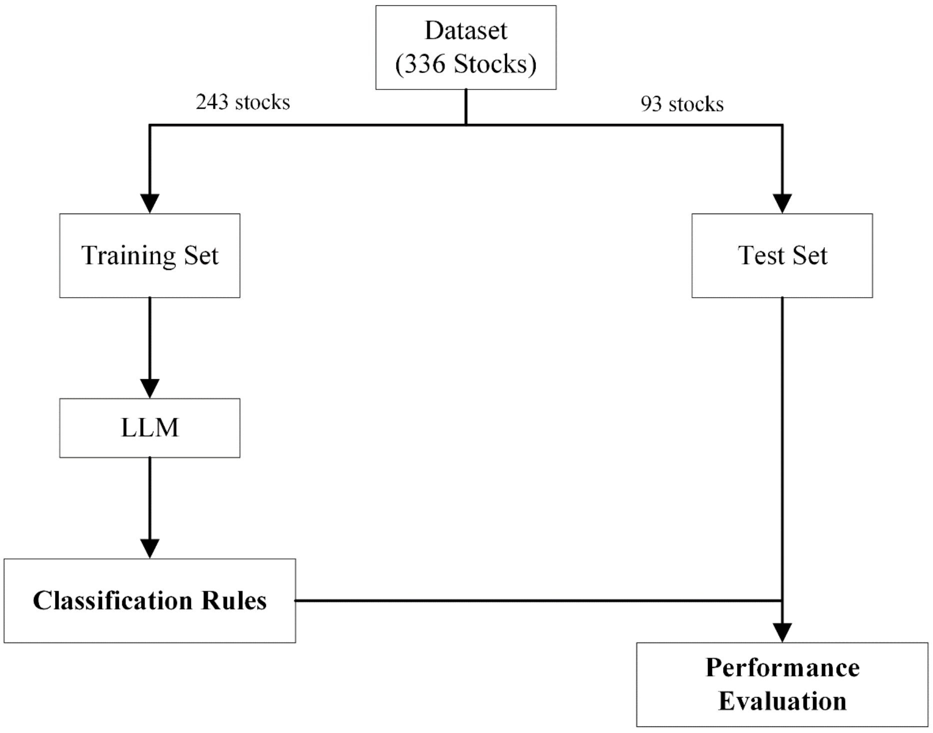


Figure 18: Rule generation workflow

The important thing about the dataset is that our selection criteria make it unbalanced. Unbalanced datasets tend to bias the performance towards the most represented class (Blagus and Lusa, 2010). In the context of unbalanced datasets, a Weighted Classification System is a technique used to address the issue of unequal class distribution. Unbalanced datasets occur when one class (the minority class) has significantly fewer instances than another class (the majority class).
In our dataset, the selected stocks were more frequent (7% Selected, 93% of Not Selected). Therefore, the Weighted Classification System (WCS) was applied to our dataset before the process started. We can imagine this system as if each sample of the selected class were repeated more times, to enhance the importance of that pattern in the learning phase. The more important and representative samples are the one with the highest allocation in the "selected class" and the one with the lowest allocation in the "Not Selected" one, that is the samples for which the decision (positive or negative) is clearer.
Classification weights, then, depend on the HRP portfolio allocation results: selected stocks with a higher allocation of capital have a higher classification weight compared to selected stocks with a lower allocation of capital. On the contrary, for the not selected class, lower weighted stocks have a higher classification weight than the higher weighted stocks.

As concerns the parametrization of the Logic Learning Machine algorithm, we set the maximum number of conditions to 3 and the maximum error allowed for each rule to 5%. LLM generated 9 rules, but 3 of them had a very low covering (< 20%). Table 10 shows the set of the 6 main rules (> 20 %) numbered from 1 to 6 (n).
Two rules predict selected stocks and 4 rules for not selected ones. Two parameters shown in Table 9 estimate the quality of the rules; 1) covering, measuring the generality, and 2) error measuring the ambiguity. The covering has two different ratios for the selected stocks which are 83.19% and 21.01%. On the contrary, covering ranged from 29.84% to 78.23% for the not selected stocks.
As mentioned previously, the maximum error rate should be 5%, therefore it ranges from 0 to 5% for all the rules.
As expected, all the rules are associated with standard deviation. The first rule for selected stocks says that SD should be lower or equal to 3.06.
This suggests that lower variability of the returns of a stock makes it a more stable and predictable investment. A standard deviation of 3.06 or lower indicates a tighter range of potential returns, signifying less risk and volatility in the stock's performance.
Investors often consider lower standard deviations desirable, reflecting a more consistent historical pattern of returns, providing confidence in the stock's future performance.
This rule can be read as a risk management tool, guiding investors towards stocks with lower price fluctuation and, consequently, a higher likelihood of meeting their investment objectives. The second rule for chosen stocks specifies that the average return must exceed 0.17. Additionally, the standard deviation of the stock's returns should be equal to or less than 4.12, and the Floating Stock Rate should surpass 22%.
This implies a preference for stocks that, on average, generate positive returns. Furthermore, a lower standard deviation suggests reduced volatility or variability in the stock's returns. Lastly, a significant Floating Stock rate is preferred as it enhances liquidity, ensuring a higher availability of shares in the market.
On the side of not selected stocks, rule 3 suggests that a stock will not be selected for inclusion in the portfolio if its SD exceeds 3.02, indicating a preference for stocks with lower volatility. Additionally, the condition for the AVG SD Ratio being less than or equal to 8.11 emphasizes the importance of consistency in the average return pattern.
Moreover, the requirement of an FRR greater than 0.19 suggests that foreign ownership of a stock can affect the investment decision in stocks with lower standard deviations. Rule 4 dictates that a stock will not be included in the portfolio if its SD exceeds 4.02, signalling a preference for avoiding stocks with very high volatility.
Notably, the error rate is zero, signifying that stocks meeting this condition are accurately identified and classified as “No”. This aligns with the objective of excluding stocks with excessively high standard deviation from the portfolio.

n	Outcome	1st condition	2nd condition	3rd condition	Covering (%)	Error (%)
1	Yes	SD ≤ 3.06			83.19	4.03
2	Yes	Mean > 0.17	SD ≤ 4.12	FLR > 22	21.01	5.00
3	No	SD > 3.02	AVG SD Ratio ≤ 8.11	FRR > 0.19	78.23	4.35
4	No	SD > 4.02			72.90	0
5	No	MSG in Stars or SubMarket	SD > 2.82	AVG SD Ratio ≤ 6.56	41.13	4.49
6	No	MSC in FI, WRT	SD > 3.62		29.84	0

SD = Standard Deviation, MSG = Market Segment, MSC = Main Sector, FI= Financial Institutions, WRT = Wholesale and Retail Trade, Tech = Technology, IC = Information and Communication, MFG = Manufacturing, FLR = Float Ratio, FRR = Foreign Ratio

Table 9: Classification rules for 336 stocks

A comparison with a competing, rule-based, technique (Decision Trees) has been inserted, highlighting that Logic Learning Machine achieves a better performance on the out-of-sample set.
The confusion matrix in Table 10 reflects the performance of the LLM model in predicting two outcomes which are “Yes” and “No”. The high True Positive rate of 95.83% indicates that the model accurately identified the majority of the selected stocks. On the not selected class, a True Negative rate of 90.38% suggests effective identification of the not selected stocks.
If we enter into more detail, the training set of the model demonstrated notable performance with a True Positive rate of 90.22% and a True Negative rate of 92.59% effective identification of the two outcomes. On the test set, the model accurately predicted all stocks with a rate of 100% for the selected stocks.
The True Negative rate of approximately 90.80% reveals the model's proficiency in correctly identifying not selected stocks from the actual 'No' cases. Figure 19 presents a plot of the overall performance of LLM.

Selected	*Forecast*	No	Yes
Actual	**336**	312	24
No	283	282 (90.38%)	30 (9.62%)
Yes	53	1 (4.17%)	23 (95.83%)

Table 10: Confusion matrix of LLM results

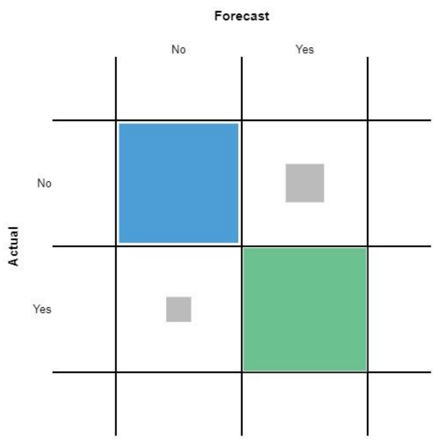


Figure 19: Confusion plot of LLM results

Overall, the model exhibited strength in accurately predicting both selected and not selected stocks. The confusion matrices for both the training and the test sets show the model's effectiveness in classifying selected and not selected stocks, with high True Positive and True Negative rates. Notably, the model achieved a perfect True Positive rate of 100% in the test set for selected stocks. If we set the continuous weight as an output and run a regression problem on the same split, we can verify that Logic Learning Machine also exhibits a quantitative advantage compared to a competing rule-based approach, the Decision Tree. More specifically, the out-of-sample median absolute error on the prediction of the (percentage) weight assigned by HRP to each stock is 0.3 for Logic Learning Machine and 0.7 for Decision Tree. After having considered each stock (selected or unselected) individually, let us also consider sub-groups of stocks and the degree to which each of these sub-groups is selected. To this end, clustering stands out as a primary approach to gaining a deeper understanding of the inherent patterns of data. The primary goal of clustering is to group a dataset into clusters where elements within each cluster share similarities and differ from those in other clusters. Among the commonly utilized clustering algorithms, K-means stands out due to its simplicity in result interpretation and ease of implementation (Pérez-Ortega *et al.*, 2020). For this reason, we use the K-means clustering method for our dataset. The attributes considered for clustering were Average Return, Standard Deviation of Average Return, Floating Stock Ratio, and Foreign Ownership Ratio. Figure 20 shows the elbow method which is employed to determine the optimal number of clusters. The results indicated that the dataset could be effectively distributed into 5 clusters.

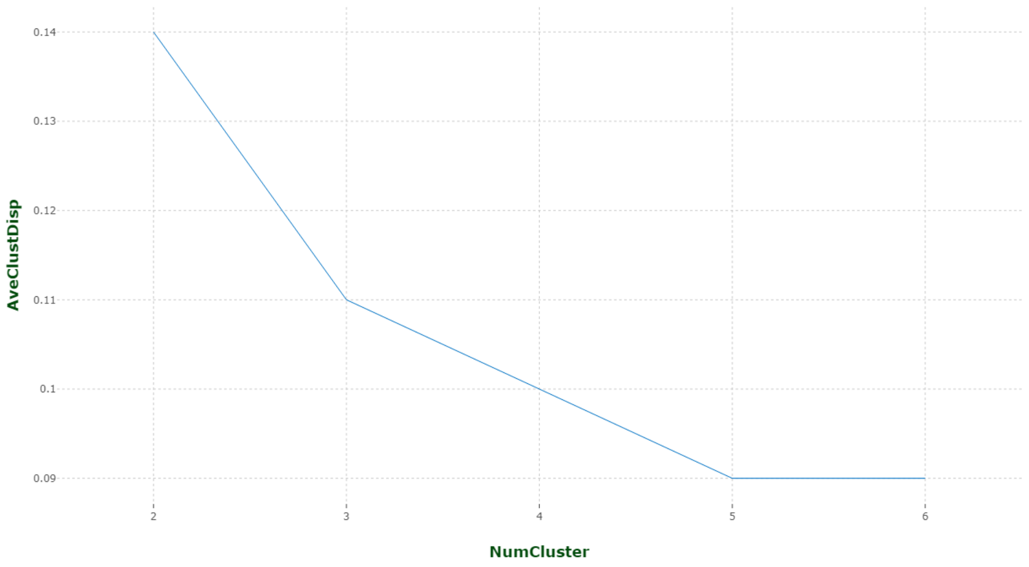


Figure 20: Elbow method for clustering 336 stocks

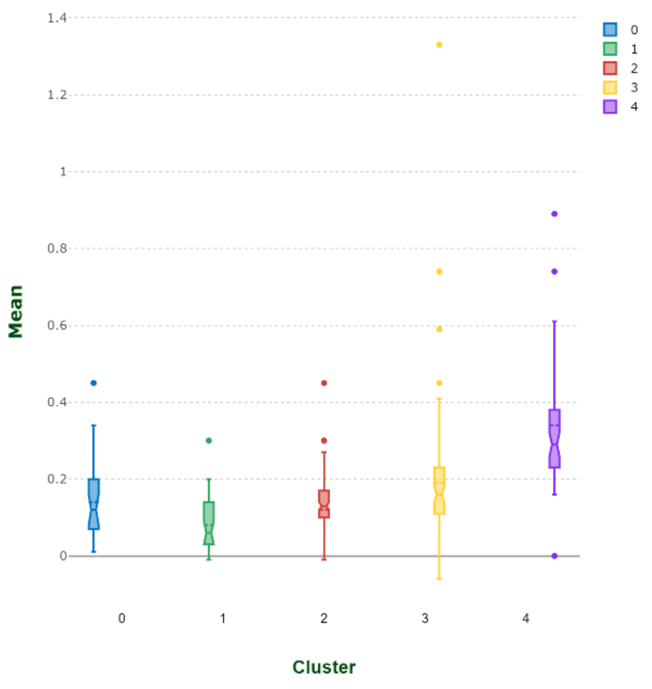
Table 11 shows the results of the clustering analysis, showing how stocks are distributed based on our selection criteria. The percentage column within the clustering results indicates the concentration of selected stocks within each cluster.
It is important to note that Cluster 0 consists of 8 selected stocks out of a total of 25 elements, representing 32.00% of the cluster.

Cluster	# Selected Stocks	# Elements	Percentage (%)
0	8	25	32.00
1	3	27	11.11
2	5	106	4.71
3	4	93	4.30
4	4	85	4.70

Table 11: Clustering results

Moreover, to understand the distinct characteristics of each cluster, boxplots were generated for each attribute within every cluster.

Figure 21 shows the average return and standard deviation of each cluster. Firstly, if we consider average returns, the wide range of average returns within Cluster 0, coupled with the absence of negative returns, signifies a diverse mix of modest and positive returns. Another fact, Cluster 0 distinguishes itself with a notably lower and tighter range in standard deviation compared to the other clusters. Figure 22 shows the boxplot for the attributes of foreign ownership and floating stock of each cluster. The observation that Cluster 0 has the highest range of Foreign Ratio indicates a pronounced level of international investor interest.
This not only points to the cluster's potential global appeal but also suggests that stocks within Cluster 0 may benefit from diverse market dynamics influenced by international trends and sentiments.
On the other hand, the lowest range of Floating Stock Ratio within Cluster 0 signifies lower liquidity levels.
This might pose challenges in terms of ease of trading positions, but from another point of view, it also implies a potential scarcity factor, which could be attractive to certain investors seeking unique investment opportunities.
In short, the distinctive features of Cluster 0, including its diverse range of positive average returns, lower standard deviation, high foreign ownership, and lower float ratio, collectively make it different than the other clusters.
It offers the potential for substantial gains with a stable risk profile, global market integration, and a unique investment landscape.



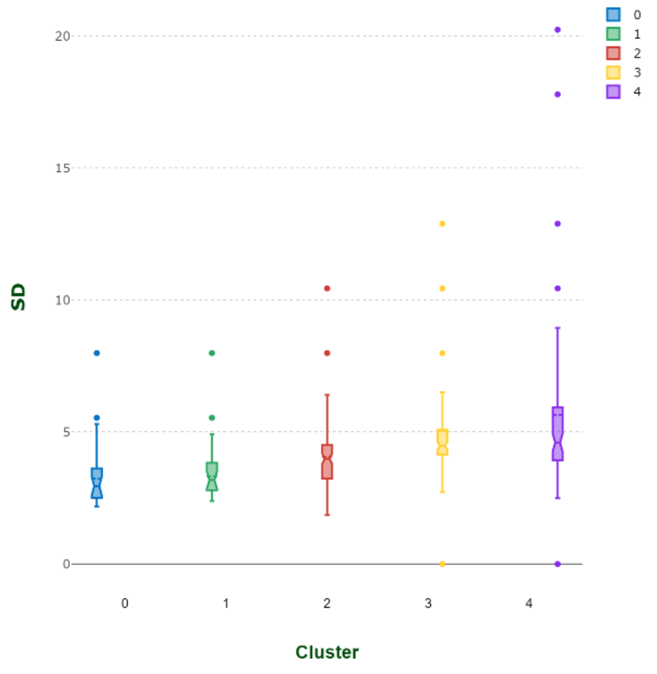
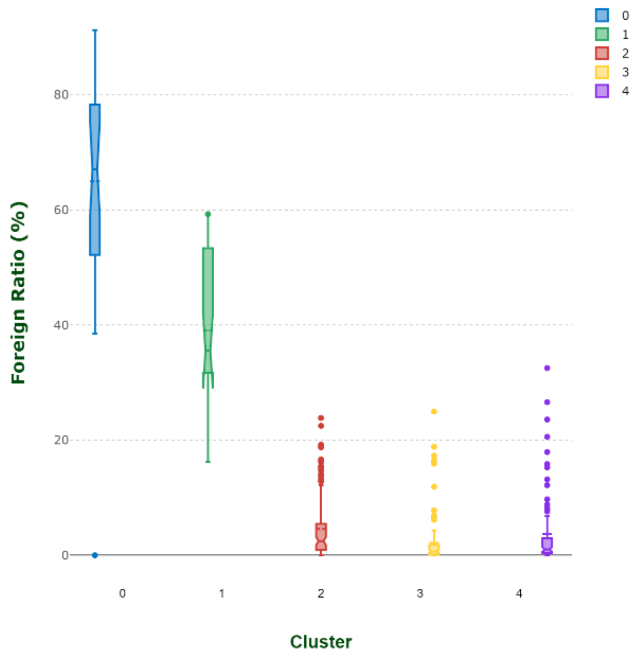


Figure 21: Boxplot of clusters for average return and standard deviation



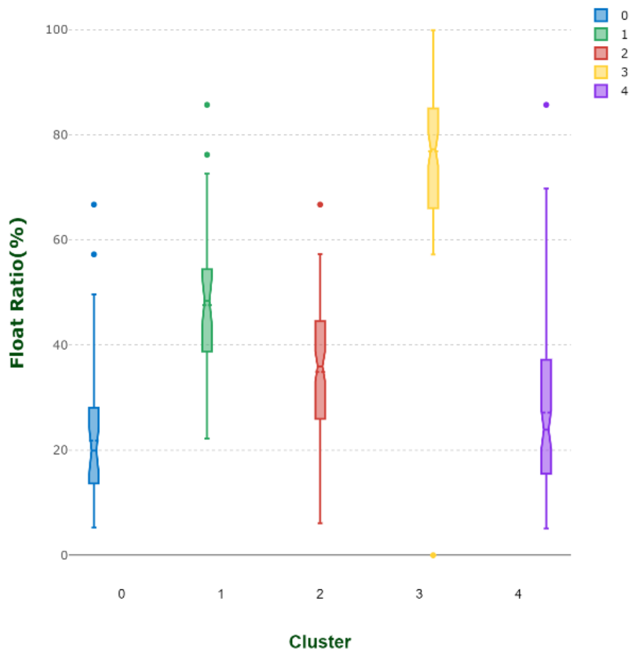


Figure 22: Boxplot of clusters for foreign and float ratio

6) Conclusions

Throughout this research, the application of the Logic Learning Machine (LLM) within the Hierarchical Risk Parity (HRP) framework has shown significant results in portfolio allocation analysis and understanding. Through the analysis of an extensive dataset related

to the Istanbul Stock Exchange, covering daily historical data of 336 stocks from 2015 to 2021, we have observed that the implementation of LLM has brought greater transparency and clarity to investment decisions.
In fact, HRP's unsupervised Machine Learning method allows the user to know the optimal allocation of weights among the assets of the analyzed portfolio by basing its aggregation criterion between clusters on the statistical value of correlation alone. In this way it is able to recognize the optimal configuration among assets but no indication is given as to which market factors may most significantly affect this choice. The addition of market features related to the Turkish stock market interpreted by the LLM allows not only an optimal level of integration between different Machine Learning methodologies, but also an optimal level of interpretability of the result thus provided.
The LLM's ability to identify and interpret significant patterns in asset characteristics, such as sector affiliation, index presence, activity in the derivatives market, and other financial features, has enabled a more informed and interpretable selection and weighting of assets in the portfolio. Looking ahead, there are several research directions that could further expand the field of portfolio allocation and enhance current methodologies:
- Alternative Methods for Estimating Correlations: Exploring the use of alternative techniques to calculate or estimate correlations between assets, such as applying neural networks or graph-based models, could offer new insights into portfolio construction and risk management.
- Dynamic Portfolio Optimization: Considering portfolio allocation in a dynamic context, where the portfolio composition can be adjusted periodically based on evolving market conditions, could provide a more realistic and applicable perspective in the long term.
- Robustness Testing in Market Stress Scenarios: Evaluating the resilience of proposed allocation strategies through the use of LLM in market stress scenarios, such as financial crises or economic shocks, could provide further evidence of their effectiveness and stability.
Exploring these themes could contribute to further enhancing portfolio allocation methodologies, making them more adaptive, robust, and capable of addressing the challenges of a continuously evolving market environment.

References

[1] Ahelegbey D. F., Giudici P. (2021). NetVIX – A network Volatility Index of Financial Markets. Physica A: statistical mechanics and its applications, 594, 127017.
[2] Asness C. A., Frazzini A., Pedersen H. L. (2012). Leverage Aversion and Risk Parity. Financial Analysts Journal, 68(1), pp. 47-59.
[3] Blagus R., Lusa L. (2010). Class prediction for high-dimensional class-imbalanced data. BMC Bioinformatics, 11(1).
[4] Burggraf T. (2021). Beyond risk parity – A machine learning-based hierarchical risk parity approach on cryptocurrencies. Finance Research Letters, 38, 101523.
[5] Cangelosi D., Muselli M., Parodi, S. Blengio F., Becherini P., Versteeg R., Conte M., Varesio, L. (2014). Use of Attribute Driven Incremental Discretization and Logic Learning Machine to build a prognostic classifier for neuroblastoma patients. BMC Bioinformatics, 15 (S5).
[6] de Prado M. L. (2016). Building Diversified Portfolios that Outperform Out of Sample. The Journal of Portfolio Management, 42(4), 59–69.
[7] Fabozzi F. J., Kolm N. P., Pachamanova A. D., Focardi M. S. (2007). Robust Portfolio Optimization and Management. 1st Edition. John Wiley & Sons.
[8] Ferrari E., Verda D., Pinna N., Muselli M. (2023). Optimizing Water Distribution through Explainable AI and Rule-Based Control. Computers, 12(6), 123.
[9] Ferrari E., Muselli M. (2010). Maximizing pattern separation in discretizing continuous features for classification purposes. The 2010 International Joint Conference on Neural Networks (IJCNN), pp. 1-8.
[10] Giudici P., Polinesi G., Spelta A. (2022). Network models to improve robot advisory portfolios. Annals of Operations Research, 313, 965-989.
[11] Ilmanen, A., Kizer, J. (2012). Dynamic allocation strategies for diversified portfolios. The Journal of Portfolio Management, 39(2), 48-60.
[12] Kesler G. (2022). Borsa Istanbul: Turkish Stock Exchange Dataset. Retrieved January 20, 2024, from: https://www.kaggle.com/datasets/gokhankesler/borsa-istanbul-turkish-stock-exchange-dataset/data.
[13] Lohre, H., Rother, C., Schafer, P. (2020). Hierarchical Risk Parity: Accounting for Tail Dependencies in Multi-Asset Multi-Factor Allocations. In E. Jurczenko (Ed.), Machine Learning and Asset Management (pp. 332-368). Iste and Wiley.
[14] Maillard, S., Roncalli, T., Teiletche, J. (2010). The Properties of Equally Weighted Risk Contribution Portfolios. The Journal of Portfolio Management, 36, 60-70.
[15] Mandelbrot, B. (1963). The Variation of Certain Speculative Prices. The Journal of Business, 36(4), 394-419.
[16] Markowitz H. M. (1952). Portfolio selection. The Journal of Finance, 7(1), pp. 77-91.
[17] Markowitz H.M. (1959). Portfolio Selection: Efficient Diversification of Investments. New York: John Wiley & Sons, Inc.
[18] Michaud, R. O. (1989). The Markowitz optimization enigma: Is 'optimized' optimal? Financial Analysts Journal, 45(1), 31-42.
[19] Millea, A., Edalat, A. (2022). Using Deep Reinforcement Learning with Hierarchical Risk Parity for Portfolio Optimization. International Journal of Financial Studies, 11(1), 1-16.
[20] Molyboga, M. (2020). A Modified Hierarchical Risk Parity Framework for Portfolio Management. The Journal of Financial Data Science, Summer 2020, 2(3), 128-139.
[21] Muselli M. (2006). Switching Neural Networks: a new connectionist model for classification. In Lecture Notes in Computer Science, pp. 23–30.
[22] Muselli M., Ferrari E. (2009). Coupling logical analysis of data and shadow clustering for partially defined positive Boolean function reconstruction. IEEE Transactions on Knowledge and Data Engineering, 23(1), pp. 37-50.
[23] Muselli M., Quarati A. (2005). Reconstructing positive Boolean functions with shadow clustering. Proceedings of the 2005 European Conference on Circuit Theory and Design, 3, pp. 377-380.
[24] Raffinot, T. (2017). Hierarchical clustering-based asset allocation. The Journal of Portfolio Management, 44(2), pp. 89-99.

[25] Rocco, D. (2014). Heavy tails and asymmetry in asset returns. Quantitative Finance, 14(12), 2189-2207.
[26] Roncalli T. (2013). Introduction to risk parity and budgeting. Social Science Research Network.
[27] Sharpe W. F. (1966). Mutual Fund Performance. The Journal of Business, 39(1), pp. 119–138.
[28] Sharpe W.F. (1994). The Sharpe Ratio. The Journal of Portfolio Management, 21, pp. 49-58.
[29] Vyas A. (2020, January 13). The Hierarchical Risk Parity Algorithm: An Introduction. In hudsonthames.org. Retrieved from: https://hudsonthames.org/an-introduction-to-the-hierarchical-risk-parity-algorithm/ (accessed 14 November 2023).