Implementing HDBSCAN in Nested Clustered Optimisation Algorithm

Nurbek Khazhimurat

Course Paper

Abstract

Optimal portfolio construction has always been a hot topic for investors and financial institutions. Recent developments in machine-learning discovered substantial opportunities in increasing the robustness and precision of existing portfolio optimisation models, and developing new ones. In this paper, the Nested Clustered Optimisation algorithm is examined and a new algorithm with certain alterations to it is presented. Then this algorithm is tested and compared to classical mean-variance approach by Markowitz.

Contents

1 Introduction

The question of optimal choice of weights for risky assets in portfolio has been raised since the beginning of financial theory. The foundation of modern portfolio theory was made in 1952 when Harry Markowitz presented his model of optimal portfolio construction [1], centering around quadratic optimisation problem involving mean returns and return variances of assets to be allocated.

Since then various alterations and improvements to Markowitz' approach were suggested, creating more and more deviations from original mean-variance approach. To this day, there are plenty of models that pursue the same idea of maximising expected returns given various other constraints. In Kalayci et. al [6], a total of 175 papers published between 1998 and 2018 were analyzed, and a detailed survey of main deterministic MVPO models were presented.

Recent developments in machine-learning discipline just made the common goal more intriguing as it brought a whole new number of opportunities to approach portfolio optimisation problem. In one of the fundamental studies in the field of application of machine learning algorithms in asset management, De Prado (2016), author pointed out main flaw of theoretical MVPO models, referring to the concept of condition number: he explained that despite mathematical soundness of MPT, the final weight allocation was subject to huge deviation from its intrinsic value, and introduced the new ML-based approach called Hierarchical Risk Parity, which was aimed to resolve the issue. However, as the author said himself, the idea behind algorithm was not in finding a better method of PO, rather in showing the potentials of implementing ML techniques in asset management tasks. Few years later he published a book named "Machine Learning for Asset Managers" (2019) which introduced a new algorithm called Nested Clustered Optimisation (NCO) based on unsupervised clustering ML algorithm.

In this paper the main focus will be on NCO. In addition, an attempt to improve it by using another clustering algorithm is made and empirical research to test the developed model is conducted. The main benchmarks for backtesting are the equal weighted portfolio and classical Markowitz minvariance portfolio.

The rest of the paper is divided as follows: Section 3 provides a theory behind all concepts and methods used in NCO algorithm, and description of NCO algorithm itself. Section 4 shows all steps of NCO applied to real-

world data; in addition, this Section shows the NCO backtesting procedure. Section 5 concludes the paper.

2 Literature Review

In 1952, when Markowitz firstly published his optimal portfolio weights computation model, the modern portfolio theory in its contemporary form was founded. In this work, he formulated the optimisation problem to be solved and later published the explicit solution for it. Many of advanced PO models are based on Markowitz classical MVPO technique, only resolving some of the issues associated with original approach.

To examine some of those advanced models, let us introduce the paper Kalayci et. al. (2019), called "A comprehensive review of deterministic models and applications for mean-variance portfolio optimization" which conducted a full review of 175 models associated with MVPO models. As the authors stated themselves, such comprehensive review was substantial to identify the pattern of historical models development, "to pave the way for future directions". For research particularly, this paper served as a beautiful introductory work, as not only it showed how versatile and different PO optimisation models can be in terms of instruments used, it also partially reveals the contribution of NCO model itself to the portfolio theory (see "clustering" type in Figure 1 tree). Figure 1 shows the tree taken from the paper showing how authors' view on portfolio optimisation models in general.

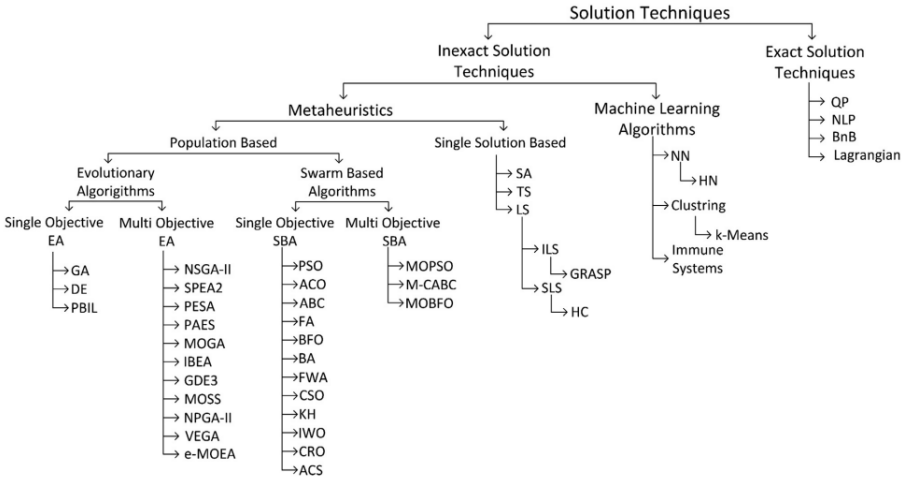

Figure 1: Classification of PO solution techniques

Before applying MVPO techniques to historical covariance matrix, it is necessary to preprocess it using a so-called denoising procedure. As it was shown in number of studies, such "Noise dressing of financial correlation matrices" by Laloux et al. (1999), who examined the eigenvalue density of historical covariance matrix under assumption that the covariance matrix was random following a Wishart distribution, pointed out that roughly 94% of eigenvalues of an empirical covariance matrix fell within theoretical ranges, meaning that only about 6% of eigenvalues contained useful information.

In the paper called "Between Nonlinearities, Complexity, and Noises: An Application of Portfolio Selection Using Kernel PCA", by Peng. et. al. (2019), authors proposed a so-called constant residual eigenvalue method to apply denoising procedure to an empirical covariance matrix. Assuming a Gaussian distribution of initial table of observations of some number of assets, authors showed the following from that assumption explicit probability density function of a covariance matrix constructed from the data. Then, using theoretical ranges of the aforementioned distribution eigenvalues are filtered and a new "denoised" covariance matrix is created.

In addition, the algorithm described in the paper called "A Well-Conditioned Estimator for Large-Dimensional Covariance Matrices", by Ledoit and Wolf (2004), is also substantial for our work, since it introduces another idea of creating a more stable estimation of covariance matrix.

In 2016, Lopez de Prado published his paper called Bulding diversified portfolios that outperform out-of-sample", where he firstly introduced his Hierarchical Risk Parity approach, and shed light on one of the major drawbacks of Markowitz model called Markowitz curse. Briefly, this algorithm included clustering assets using hierarchical clustering technique, then using dendrogram constructed, recomputing weights on every node of it, using recursive bisection technique, until one reaches the final node, where every asset will have its weight in the final portfolio.

After developing an algorithm, one needs to test its performance on financial data, comparing it to some bechnmark chosen by him. In the book called "Advances in Financial Machine Learning" by de Prado (2019), most popular portfolio backtesting algorithms were described and for each of them author provided clear instructions on avoiding operational mistakes, false strategy discoveries and etc. In addition, he explained what the main pros and cons of each of them were. WF-backtesting and k-folds cross-validation backtesting approaches described in this book were used in the

empirical part of this paper.

In 2019, de Prado published a book named "Machine-Learning for Asset-Managers", where not only did he introduce the main algorithm of the paper called NCO, but also dedicated 6 chapters before NCO introduction to various theoretical concepts and practical techniques to realise the full potential of NCO. As an example of those techniques, the aforementioned covariance matrix denoising procedure, Ledoit-Wolf shrinkage, correlation-based metric choice, k-means clustering and etc. Since almost all of the concepts covered in the book will also be explained in the paper, I will not review this book thoroughly.

In the original NCO model, the k-means clustering algorithm was used which for reasons described in Section 3.4 was replaced by HDBSCAN clustering algorithm. In the paper, called "Density-based clustering based on hierarchical density estimates", authors introduced new density-based clustering algorithm, which was an advancement of another popular density-based algorithm called DBSCAN, resolving some issues of the latter.

3 Theoretical Background

3.1 Markowitz Model

Before digging into ML methods applied in portfolio optimisation, let us provide a theoretical background of classical Markowitz' portfolio selection approach, a starting point to majority of MVPO models. First, let us introduce the following notations:

$$r_i = \frac{T_i - T_i(-1)}{T_i(-1)}$$

is the return on asset $i \in \{1, 2, 3, ..., n\}$, where T_i is the total market value of asset i at current period and $T_i(-1)$ is the total market value of the asset in the previous period. Let $r = (r_1 \ r_2 \ \dots \ r_n)^T$ be the vector of returns. Consider a portfolio consisting of n risky assets, each asset's weight is w_i, hence, $\sum_i w_i = 1$, and r_i is the return of asset $i \in \{1, 2, ..., n\}$. In this case, $w = (w_1 \ \dots \ w_n)^T$ will be the vector of weights on assets in the portfolio. Mean and variance of asset i are denoted as μ_i, σ_i^2, respectively, and covariance between returns on assets i, j is denoted as σ_{ij}. Considering r_p to be return on the whole portfolio, we have

$$\mu_p = E[r_p] = \sum_{i=1}^{n} r_i w_i = w^T r$$

$$\sigma_p^2 = Var[r_p] = \sum_{i=1}^{n} \sum_{j=1}^{n} \sigma_{ij} w_i w_j = w^T V w,$$

where V is the variance-covariance matrix for assets in the portfolio. So the main optimisation problem of the model is given by

$$\begin{cases} \frac{1}{2}\sigma_p^2 \to \min_{w_1, w_2, \dots, w_n} \\ \text{s.t. } w^T a = 1 \end{cases},$$

where choice of vector a reflects the goal of MVPO. From Markowitz portfolio theory we know that if $a = 1_n$, i.e. a is vector of ones of size n, the solution will be the min-variance portfolio, and if $a = r$, the solution maximizes the Sharpe Ratio of the portfolio and is a so-called tangent portfolio, and from two-fund separation theorem we know that the rest of the portfolios on the efficient frontier can be achieved via linear combination of any

two efficient portfolios, usually min-variance and tangent ones. Hence, in further empirical analyses the tangent and min-variance portfolios will be the main benchmarks in our tests.

Markowitz Curse

To explain the so-called Markowitz Curse issue, let us firstly recall some notions from numerical analysis and linear algebra. From numerical analysis, the condition number measures how much the output value of a function changes with a slight change in the input of the function. In our case the application of the notion of the condition number in matrix algebra is of importance: for example, the condition number of a SLE $Ax = b$ shows how inaccurate will the value of x be after approximation. Hence, if the condition number of a matrix is large, even a little error in b can cause a large error in x. This notion will help point out the fundamental flaw of Markowitz MVPO approach. The explicit solution of the MVPO depends on the covariance matrix, hence, let us show that the condition number of a covariance matrix tends to be large. Firstly, we know that correlation matrix can be calculated from covariance matrix in the following way (see Appendix for proof): let Σ be a covariance matrix, and ρ_M be the correlation matrix. Hence,

$$\rho_M = (diag(\Sigma))^{-1/2} \cdot \Sigma \cdot (diag(\Sigma))^{-1/2},$$

so if the condition number of the correlation matrix is high, then the condition number of the covariance matrix will be high as well, and vice versa. Condition number of symmetric matrix is the absolute value of the ratio between greatest and lowest eigenvalues of it. For simplicity, let us refer to example of 2x2 correlation matrix. Let us define matrix C as follows:

$$C = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}.$$

The eigenvalues of matrix C are then will be equal to $\lambda_1 = 1-\rho,\ \lambda_2 = 1+\rho$, hence, the condition number of matrix C is given by:

$$\kappa(C) = \frac{1 + |\rho|}{1 - |\rho|}.$$

It is clear that and the condition number increases in the absolute value of ρ. For $\rho \to 1^-$ or $\rho \to -1^+$, the condition number tends to infinity.

However, there is no formal way to prove that with increasing size of a correlation matrix the condition number will increase as well, since the latter is not always true. Nevertheless, consider the following approach to prove the concept that later will help explaining usefulness of NCO.

Definition. Consider some $n \times n$ symmetric matrix A. A *nested sequence of principal minors* of matrix A is a sequence of submatrices $\{A_k\}_{k=1,2,...,n}$, where each matrix contains first k rows and k columns of matrix A.

Theorem. Consider a positive definite symmetric matrix C. Let $C_1, C_2, ..., C_n$ be a nested sequence of principal minors of matrix C. Then, the condition numbers of principal minors of C satisfy the inequality: $\kappa(C_1) \leqslant \kappa(C_2) \leqslant ... \leqslant \kappa(C_n) = \kappa(C)$.

The prove of the theorem is written in the Appendix. However, the theorem gives us a lot of insight on relationship between condition number and the covariance matrix, as the latter is positive definite if and only if it is invertible, which is true for most cases.

In Markowitz MVPO problem, the value of vector of weights w depends on the so-called precision matrix, V^{-1}, thus, one can expect stable solutions for PO problem only when $\rho \approx 0$. However, the usefulness of Markowitz model (or diversification in general) reveals in the opposite situation, when assets are correlated ($\rho \not\approx 0$) so the more Markowitz model is needed, the less numerically stable answers it provides, which is the Markowitz Curse. Further it will be shown how NCO algorithm partially resolves this issue.

3.2 Covariance Matrix Denoising

3.2.1 Constant Residual Eigenvalue Method

Before introducing the NCO algorithm, several theoretical notions should be explained. In Peng et. al. (2019), the process called Covariance Matrix Denoising was shown, which referred to the concepts from Random Matrix Theory: under assumption that matrix of i.i.d. observations of asset returns X is subject to Gaussian distribution, authors showed the explicit form of p.d.f. of random eigenvalues, and those which did not fall into theoretical ranges of such p.d.f. were considered as either noise, or signal eigenvalues. However, in de Prado (2019), a slightly different approach was described: constructing an empirical covariance matrix of

i.i.d. observations of asset returns, author assumed the eigenvalues of a the matrix are distributed according to Marsenko-Pastur p.d.f. More formally, the eigenvalues λ of a $T \times N$ matrix asymptotically converge $(N \to \infty, T \to \infty,\ 1 < T/N < +\infty)$ to Marsenko-Pastur distribution:

$$f(\lambda) = \begin{cases} \frac{T}{N} \frac{\sqrt{(\lambda_+ - \lambda)(\lambda - \lambda_-)}}{2\pi\lambda\sigma^2}, & \text{if } \lambda \in (\lambda_-, \lambda_+) \\ 0, & \text{otherwise} \end{cases},$$

and if some empirical eigenvalue is less than λ_-, it belongs to the "noise" category (further referred to as "noisy eigenvalues"), and if it is greater than λ_+, it belongs to the "signals" category. Upper and lower bounds of Marsenko-Pastur distribution are given by

$$\lambda_+ = \sigma^2 \left(1 + \frac{N}{T}\right)^2,$$

$$\lambda_- = \sigma^2 \left(1 - \frac{N}{T}\right)^2$$

If that eigenvalue is in the range $[\lambda_-, \lambda_+]$, it is considered to be a result of randomness of price determination procedure (since it falls within theoretical range of eigenvalues of random matrix with certain variance). Covariance matrix denoising is the process in which all noisy eigenvalues in diagonal matrix $\Lambda = Q^T \Sigma Q$, with eigenvalues on diagonal entries, were replaced by their mean $\overline{\lambda}^{(\text{noise})}$, and then the filtered matrix Λ^* was used to compute the final denoised covariance matrix $\Sigma^* = Q\Lambda^* Q^T$. Another more radical way of covariance matrix denoising involves replacing all random and noisy eigenvalues with their mean, leaving only signal eigenvalues unchanged, however, this approach was not implemented in this paper.

In addition, this method guarantees condition number of denoised matrix to be lower or equal than the condition number of an initial matrix. To show that, consider the lowest noisy eigenvalue of the initial covariance matrix, and let us denote it as λ_{min}. After denoising procedure, the greatest eigenvalues of both matrices remain the same, however, the lowest eigenvalue of denoised matrix will change from λ_{min} to $\bar{\lambda}^{(\text{noise})}$. Since $\bar{\lambda}^{(\text{noise})} \geqslant \lambda_{min}$, we can ensure that

$$\kappa(\Sigma) = \frac{\lambda_{max}}{\lambda_{min}} \geqslant \kappa(\Sigma^*) = \frac{\lambda_{max}}{\bar{\lambda}^{(\text{noise})}}.$$

3.2.2 Ledoit-Wolf Shrinkage

Suppose X is a $p \times n$ matrix with n i.i.d. variables and p observations, with covariance matrix Σ. Considering Frubenius norm $||A|| = \sqrt{tr(AA^T)/p}$, the goal of the original paper was to find the algorithm to determine such linear combination $\Sigma^* = \alpha_1 I + \alpha_2 S$, where $S = XX^t/n$, for which the expected quadratic loss $E\big[||\Sigma^* - \Sigma||^2\big]$ is minimum. Mathematically, we can formulate the following optimisation problem:

$$E\big[||\Sigma^* - \Sigma||^2\big] \rightarrow \min_{\alpha_1, \alpha_2}$$

$$\text{s.t. } \Sigma^* = \alpha_1 I + \alpha_2 S.$$

The algorithm then returns Σ^* as an output. Although the theoretical part of the usefulness of Ledoit Wolf shrinkage will not be covered in the paper, it will be shown empirically that this procedure indeed leads to more stable results.

3.3 Proximity Matrix

To conduct Nested Clustered Optimisation algorithm, the main algorithm of the paper, the one needs to use unsupervised clustering algorithm on covariance matrix of a set of assets to be allocated. However, since clustering algorithms work with points on metric spaces and join *similar* points into clusters, the matrix to be clustered has to depict the dissimilarity between the returns on those assets. To achieve such result, each entry in empirical correlation matrix should be recomputed using a certain distance metric. In our case, we consider an isomorphic function $\phi : [-1, 1] \rightarrow [0, 1]$; denoting an entry in correlation matrix as $(ij) = \rho_{ij}$, each entry in the proximity matrix is then given by

$$(ij) = \sqrt{\frac{1}{2}(1 - \rho_{ij})}.$$

The metric ϕ is indeed a metric, as it satisfies all three metric axioms (see Appendix for a proof). In addition, this metric reveals dissimilarity between returns on two assets: if returns on assets are perfectly correlated, then the value of the distance metric will be $d(\rho) = d(1) = 0$, as the distance from point to itself needs to be zero, and is equal to $d(-1) = 1$ in the opposite case. Hence, having n assets, we will get a symmetric $n \times n$

matrix where each row (or each column) shows coordinates of a point of a certain asset in n-dimensional space. All assets close to each other will be joined in one cluster, and assets far from each other will belong to different clusters.

3.4 Unsupervised Clustering Algorithms

As the major alteration to original NCO introduced in this paper, HDBSCAN clustering algorithm is used, instead of classical k-means clustering. However, the theory behind HDBSCAN algorithm is complex and will not be covered in this paper. Nonetheless, HDBSCAN is just an advanced version of another density-based approach called DBSCAN, and to be able to interprete HDBSCAN results and not treat it as a complete "black-box" model, let us provide an intuiton behind its work. To pursue that goal it is necessary to explain a relatively robust theory behind DBSCAN algorithm first.

3.4.1 DBSCAN

The Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is an example of density-based clustering algorithms that detect distinctive groups in the data, assuming that a cluster in the underlying space is a contiguous region of high point density, separated from others by regions of low density. The algorithm uses two main parameters, epsilon (ϵ) and minimum number of points (MP) being located together in a region (whose size is determined by value of epsilon) for a region to be considered dense.

Generally, DBSCAN uses the following algorithm to cluster points on the data: firstly, algorithm creates an n-dimensional circle around each point in the data set, and classifies each point into three categories, (1) core point, which contains at least determined by the user (MP) number of points around it, (2) border point, which contains positive but less than determined minimum number of points (MP) and (3) noise if there are no other points in the neighborhood. For example, Figure 2 shows this step on the data in 2-dimensional space with MP parameter being set to 3. All red points on the figure are considered to be core points, as each of them contains at least 3 points in their neighborhood. Yellow points are classified as border points, and purple ones are deemed to be noise.

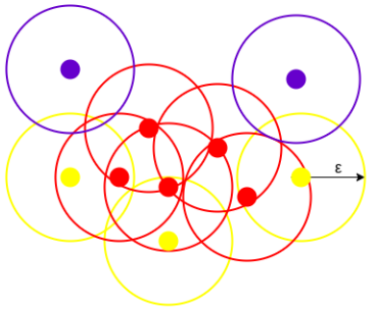

Figure 2: DBSCAN Algorithm with $MP = 3$

Before explaining the algorithm further, several definitions need to be provided. Consider two points x and y on the data space.

Definition. Point x is *directly density-reachable* from point y if x belongs to the neighborhood of y and y is a core point.

Definition. A point x is *density-reachable* from point y if there is a sequence of points $\{a_1, a_2, ..., a_k\}$, $k \in \mathbb{N}$, where $x = a_1$ and $y = a_k$, such that a_{i+1} is directly density-reachable from a_i, for all $i = 1, 2, ..., k$.

Definition. A point x is *density-connected* to point y if there exists a point C on the data space, such that both x, y are density-reachable from point C.

DBSCAN algorithm joins all density-connected pairs of points into one cluster, and separates points that are not density-connected. Following the definition, noisy points will not be density-connected to any other point in the data space and because of that algorithm forms a separate cluster of noises that includes all noisy points in the data space.

Despite its obvious advantages over k-means algorithm, DBSCAN suffers from the following major drawback: it performs poorly when clusters do not have homogeneous density. For example, suppose one has a dataset with two "true" clusters and wants to use DBSCAN to identify them, thus, both clusters are of interest. However, if points in one cluster are located densely and points in another cluster have less density than the former, the algorithm is more likely not to identify them both, as the any epsilon parameter (density measure) will only be suitable for one of clusters or to neither of them. To resolve this issue, Campello, Moulavi and Sander (2013) developed so-called Hierarchical Density-Based Spatial Clustering

of Applications with Noise (HDBSCAN) algorithm which became more suitable for datasets with varying density of clusters. Since there is no valid reason to assume that proximity matrix has only clusters with same density, in the PO algorithm developed in the paper HDBSCAN is used.

3.4.2 HDBSCAN

Now let us consider inputs and outputs of HBDSCAN algorithm. HDBSCAN requires only one parameter, which is the minimum number of points for a dense group of points to be considered a cluster.

As a result, HDBSCAN algorithm divides datapoints into some number of clusters and returns the clusters. However, since HDBSCAN uses hierarchical approach to clustering, in contrast to DBSCAN, it can also provide useful insights on the data structure constructing a dendrogram.

Basically, a dendrogram is the binary tree plot of the hierarchical structure of the data. For example, consider the following dataset in 2-dimensional space:

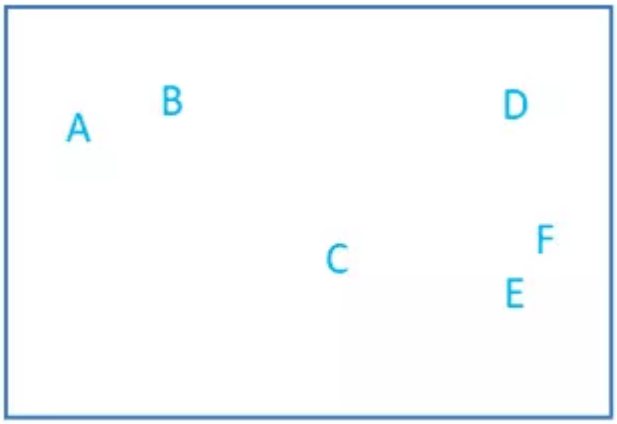

Figure 3: 2-dimensional set of points

Clearly, one can identify at least two clusters, {A, B} and {C, D, E, F}. However, inside the second cluster, we can see that E and F are located close to each other, however, C is further from them than D is. So one might identify {C}, {D, F, E} as two subclusters inside cluster {C, D, E, F}. Then continue the process for all subclusters until one reaches the final

step where all clusters have only one point. To visualize this procedure, one might use the dendrogram, shown in Figure 4:

Figure 4: Dendrogram of hierarchical clustering

Generally, this is the approach HDBSCAN uses for resolving issue of heterogeneous density among clusters in the data space by adjusting value of epsilon on every node. In the example introduced above, it is clear that epsilon parameter for cluster {D, E, F} is greater than for cluster {E, F}, since the latter cluster contains more densely located points than the former.

3.5 Nested Clustered Optimisation

This section shows how theoretical concepts described in previous subsections are applied in NCO algorithm. In addition, the justification of HDBSCAN algorithm usage is provided.

3.5.1 Proximity Matrix Clustering

In the paper, Lopez de Prado suggested to use simple k-means clustering algorithm to cluster the proximity matrix described in section 3.3 of the present paper. This algorithm is subject to several major issues: firstly, it requires its user to specify number of clusters, which is not a trivial task, if one does not possess any prior knowledge regarding the structure of the

data. In addition, as it was stated by de Prado (2019), ML techniques also serve the purpose of finding patterns in data that were not previously discovered. If one specifies number of clusters for the algorithm to find, clearly the possibility of unexpected discoveries decreases. Secondly, the algorithm poorly handles noise and outliers, however, as proximity matrix contains only values in the range from 0 to 1, the matrix to be clustered is not subject to neither noise, nor outliers, so this issue should not concern us.

However, determining number of clusters makes the algorithm extremely resource-intensive, since, for some $n \times n$ covariance matrix, in the original paper author finds optimal number of clusters via loop, where he iterates every number from 1 to $n/2$ (or $n/2-1/2$ for odd n), and computes silhouette score [1] for every iteration and chooses such number for which silhouette coefficient is the highest.

Out of all clustering algorithms, only density-based approaches allow clustering without specifying number of clusters. The most popular density-based clustering method is DBSCAN, which was explained in Section 3.4 of the paper. However, as DBSCAN has its own set of drawbacks and is not fully suitable for our case, to resolve the issue of resource intensiveness of original approach, in the paper the advanced algorithm based on DBSCAN called HDBSCAN was used.

Via clustering dense areas together, HDBSCAN divides set of assets into some number of clusters and the noise cluster. From theorem stated in Section 3.1 of the paper, we know that all principal minors in the sequence of nested principal minors of a covariance matrix have less or equal condition number than condition number of a whole covariance matrix. Since each cluster covariance matrix can be considered a principal minor of empirical covariance matrix, it is sufficient to say that condition number of every cluster covariance matrix will always be less or equal than the condition number of covariance matrix of all assets.

Since HDBSCAN has only one parameter for tuning, the minimum number of points for a group of points to have to be identified as a cluster, the number of iterations to determine "best" parameter will be much less than in the original approach. The altered NCO algorithm iterates all values for this parameter from 2 to 20 and then selects such clustering node (see Section 3.3) which satisfies minimum cluster size requirement

[1] One of the most popular metrics for evaluating clustering precision

and maximizes the silhouette score.

3.5.2 Intra-/Intercluster Weights

After clustering algorithm is performed, the following procedure on identified clusters is applied: firstly, we treat each cluster as a separate portfolio of assets in this cluster, and optimize it using Markowitz model. As a result, we get intracluster weights and values of expected return and variance of every portfolio consisting of assets from only one cluster. Then, we treat each of those portfolios as a separate asset and construct a so-called reduced covariance matrix that consists of those cluster assets. The next step is to optimize the portfolio using reduced covariance matrix, which by construction will be closer to diagonal tackling the source of Markowitz curse. As a result, we get intercluster weights. Suppose after clustering we get k clusters and $i = 1, 2, ..., k$ denotes index of a cluster.

Consider the $n \times k$ matrix of intracluster weights W_{intra}, where columns represent clusters and rows represent the assets; every entry then denotes what intracluster weight an asset j, $j \in \{1, 2, ..., n\}$, has in a cluster i, and if it does not belong to it, the weight is simply 0. Denote vector of size k, w_{inter} as the vector of intercluster weights. Then, the matrix multiplication $W_{\text{intra}} \times w_{\text{inter}}$ results in a vector of size n which shows the final allocation of assets in the portfolio. The entry j of this vector shows the weight of asset j in the resulting portfolio.

Since the condition numbers of a reduced covariance matrix and all cluster covariance matrices is less than the condition number of the initial covariance matrix of all assets, the weights estimated by NCO are expected to be much more stable than those which are achieved via applying Markowitz model to the empirical covariance matrix directly. Next section shows application of the whole procedure on 403 world ETF assets in python.

4 Empirical Research

This section shows empirical results of implementing altered NCO algorithm in python. Section 4.1 describes main steps during data collection. Section 4.2 examines in details every step of NCO applied to real-world data and Section 4.3 compares altered NCO results with equal weighted portfolio and Markowitz minvariance portfolio results using walk-forward and cross-validation backtesting approaches.

4.1 Data Collection

Since NCO algorithm is based on clustering, the main objective during data collection was to find such set of assets which would have natural clusters, for example, divided by countries, industries etc. After choosing such set, the time series of closed market prices for each of them will be collected and then formed into one dataset, which will be used for backtesting and model implementations.

The financial instruments used in the data for empirical research of the work were chosen to be world ETF-assets, containing ETF-instruments from above 40 countries and some world indices. Overall, the final sample contained 403 world ETF-instruments. The final dataset is the table consisting of 403 columns and 2460 rows, each showing closed market price of assets on a certain date from the first of january 2012 to the thirty first of may 2021.

The data preprocessing procedure included the following steps: (1) removing missed values from the data, (2) since the initial table consisted of daily prices on assets, the table was adjusted to show only weekly closed market prices on the assets, (3) then, the weekly returns table was computed from adjusted table in step (2), via calculating first differences.

Figure 5 shows the cut version of final dataset.

	Australia: ASX All Ordinaries	Australia: ASX Small Ordinaries	Australia: FTSE Australia	Australia: S&P/ASX 100	Australia: S&P/ASX 20	Australia: S&P/ASX 200	Australia: S&P/ASX 200 Accumulated	Australia: S&P/ASX 200 Banks	Australia: S&P/ASX 200 Captal Goods Size	Australia: S&P/ASX 200 Diversified Financials	...	Turkey: Basic Metal	Turkey: Chem Petrol Plastic	Turkey: Corporate Governance	Ele
2012-01-02	3259.15	1633.48	265.23	2687.57	2071.81	3245.20	24877.66	4901.50	2352.53	2387.29	...	416.91	259.90	335.66	
2012-01-09	3269.15	1643.82	268.43	2694.20	2074.74	3284.24	25177.47	4908.29	2400.32	2409.24	...	412.32	261.18	328.93	
2012-01-16	3306.36	1677.40	271.16	2719.39	2091.26	3317.68	25433.75	4952.15	2495.37	2361.90	...	432.12	271.84	347.85	
2012-01-23	3368.23	1738.81	275.73	2765.50	2119.42	3380.00	25911.80	4986.15	2663.75	2480.89	...	432.21	274.61	356.26	
2012-01-30	3404.95	1748.98	279.19	2798.11	2152.91	3418.08	26203.96	5099.21	2744.17	2502.81	...	459.45	274.14	369.58	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
2021-05-03	7286.80	3267.80	584.71	5806.90	3952.90	7028.80	79278.95	8208.70	3176.87	9760.83	...	5316.61	2096.69	1196.82	
2021-05-10	7419.80	3277.60	598.91	5938.30	4061.40	7172.80	80988.38	8299.99	3243.28	9661.01	...	5736.23	2118.90	1236.73	
2021-05-17	7255.80	3201.00	586.99	5815.30	3986.60	7023.60	79518.45	8240.02	3158.06	9406.52	...	5540.51	2139.40	1230.91	
2021-05-24	7276.00	3209.80	588.50	5835.70	4003.20	7045.90	79776.55	8398.68	3171.57	9379.27	...	5448.78	2142.60	1236.52	
2021-05-31	7406.70	3290.90	597.71	5926.50	4057.20	7161.60	81100.22	8512.08	3115.78	9646.04	...	5393.67	2032.19	1205.66	

492 rows × 403 columns

Figure 5: Cut version of the final dataset

4.2 Estimation of Weights

This subsection applies all theoretical concepts described in Section 3 to data described in the previous subsection and shows the results of this work.

4.2.1 Matrix Denoising and LW shrinkage

Firstly, after deriving dataset consisting of weekly returns on every week from january 2012 to may 2021, the algorithm forms the covariance and correlation matrices based on it. Figure 6 shows heatmap of raw empirical correlation matrix.

Figure 6: Heatmap of raw correlation matrix

After that both Constant residual eigenvalue denoising and Ledoit Wolf shrinkage procedures are applied to the covariance matrix. The process of estimation of theoretical maximum and minimum eigenvalues (λ_+, λ_-, Section 3.2) is described in Appendix. Figure 7 shows heatmap of denoised correlation matrix (calculated from denoised covariance matrix) and Figure 8 demonstrates result of Ledoit Wolf shrinkage. In Section 3.2 of the paper it was theoretically proven that condition number of a covariance matrix decreases (or, rarely, remains the same) after denoising procedure is used. Figure 9 shows output from Jupyter Notebook environment showing how condition numbers change after denoising and LW shrinkage procedures, which justifies aforementioned statement.

Figure 7: Heatmap of denoised correlation matrix

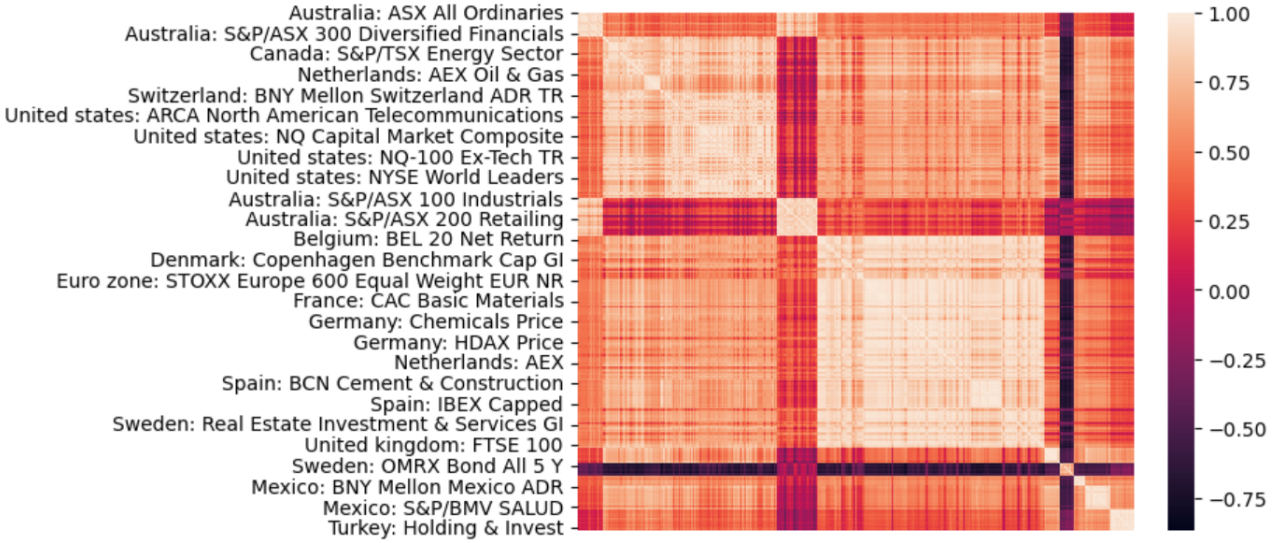

Figure 8: Heatmap of LW-shrinked correlation matrix

```
Condition number of historical covariance matrix : 3485826872800899072.000000
Condition number after denoising via Constant residual eigenvalue method : 22322.8449298109
Condition number after Ledoit Wolf shrinkage : 18034.466156118684
```

Figure 9: Condition numbers after denoising and LW-shrinkage

4.2.2 HDBSCAN Clustering

Then the algorithm applies HDBSCAN clustering to the matrix constructed from processes described in previous subsection. Since HDBSCAN uses only one parameter as a hyperparameter, which is minimum size of a cluster, and the most suitable value of it in terms of "goodness" of clustering is unknown, the algorithm iterates a set of values for this parameter to find such value of this hyperparameter that fits the data the most (maximises silhouette score). Overall, HDBSCAN identified 50 clusters and Figures 10, 11 show condition numbers of all clusters separately.

```
Condition number of cluster 0 : 47.465856301833036 ;  10 assets in total
Condition number of cluster 1 : 1259.2178800679 ;  17 assets in total
Condition number of cluster 2 : 424.88093550376203 ;  21 assets in total
Condition number of cluster 4 : 99.67182728397327 ;  4 assets in total
Condition number of cluster 6 : 139.54467023575066 ;  5 assets in total
Condition number of cluster 7 : 1076.5215704937007 ;  12 assets in total
Condition number of cluster 8 : 46.9631445507051 ;  3 assets in total
Condition number of cluster 9 : 21.49613082853204 ;  2 assets in total
Condition number of cluster 10 : 105.25247811689758 ;  5 assets in total
Condition number of cluster 11 : 145.11422991717458 ;  3 assets in total
Condition number of cluster 12 : 323.50946370972235 ;  12 assets in total
Condition number of cluster 13 : 51.69655765410633 ;  3 assets in total
Condition number of cluster 14 : 70.55897740365313 ;  3 assets in total
Condition number of cluster 15 : 438.87793221080574 ;  8 assets in total
Condition number of cluster 16 : 798.8559813124225 ;  11 assets in total
Condition number of cluster 17 : 46.987973265638004 ;  2 assets in total
Condition number of cluster 18 : 275.87358474744553 ;  6 assets in total
Condition number of cluster 19 : 65.47363436568595 ;  3 assets in total
Condition number of cluster 20 : 361.05217696964604 ;  10 assets in total
Condition number of cluster 21 : 222.26267688213284 ;  7 assets in total
Condition number of cluster 22 : 135.66959327251288 ;  5 assets in total
Condition number of cluster 23 : 243.40150099397582 ;  5 assets in total
Condition number of cluster 24 : 195.21583797458862 ;  8 assets in total
Condition number of cluster 25 : 180.78287915039536 ;  3 assets in total
Condition number of cluster 26 : 45.060862882759594 ;  2 assets in total
Condition number of cluster 27 : 83.15612545349133 ;  2 assets in total
Condition number of cluster 28 : 663.6113300393653 ;  10 assets in total
Condition number of cluster 29 : 216.44063866849967 ;  4 assets in total
```

Figure 10: Condition number of clusters from 0 to 29

```
Condition number of cluster 30 : 233.31923052716445 ;  5 assets in total
Condition number of cluster 31 : 165.63566139791578 ;  4 assets in total
Condition number of cluster 32 : 74.43386756635306 ;  3 assets in total
Condition number of cluster 33 : 168.01825991703117 ;  5 assets in total
Condition number of cluster 34 : 426.2581160073373 ;  9 assets in total
Condition number of cluster 35 : 171.8527504518416 ;  4 assets in total
Condition number of cluster 36 : 277.33795293759516 ;  7 assets in total
Condition number of cluster 37 : 208.5382221663394 ;  5 assets in total
Condition number of cluster 38 : 129.63901088653702 ;  3 assets in total
Condition number of cluster 39 : 598.5988428452476 ;  11 assets in total
Condition number of cluster 40 : 151.83774174413904 ;  4 assets in total
Condition number of cluster 41 : 245.98001536406065 ;  5 assets in total
Condition number of cluster 42 : 123.82305623862149 ;  3 assets in total
Condition number of cluster 43 : 178.59680171635 ;  4 assets in total
Condition number of cluster 44 : 154.0599762895517 ;  3 assets in total
Condition number of cluster 45 : 316.50496059488734 ;  6 assets in total
Condition number of cluster 46 : 140.3364346037381 ;  3 assets in total
Condition number of cluster 47 : 175.57971740185116 ;  4 assets in total
Condition number of cluster 48 : 426.5273252070457 ;  8 assets in total
Condition number of cluster 49 : 6205.718746715705 ;  108 assets in total
```

Figure 11: Condition numbers of clusters from 30 to 49

Figure 12 shows results of HDBSCAN clustering, in which rows and columns of denoised correlation matrix are rearranged to demonstrate clusters more explicitly. Figure 13 shows the dendrogram (see Section 3.3) showing the hierarchical structure of the data.

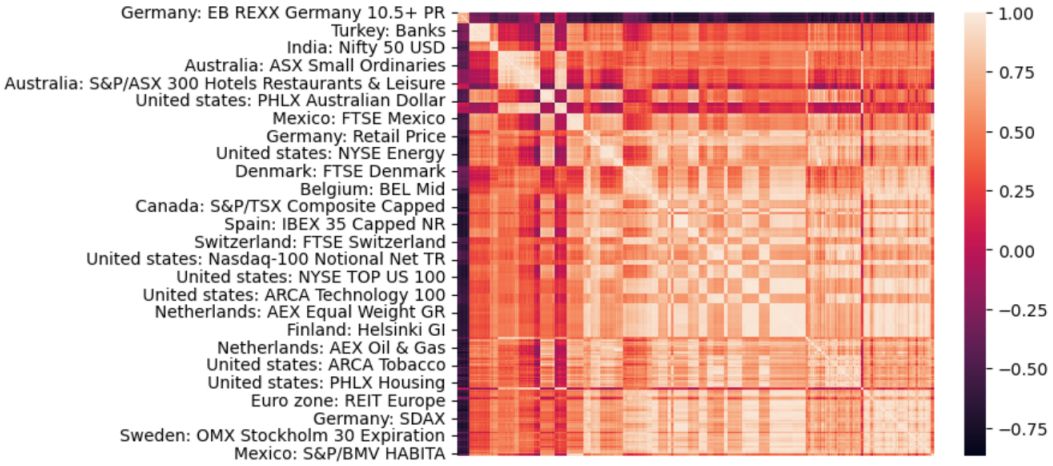

Figure 12: Heatmap of clustered correlation matrix

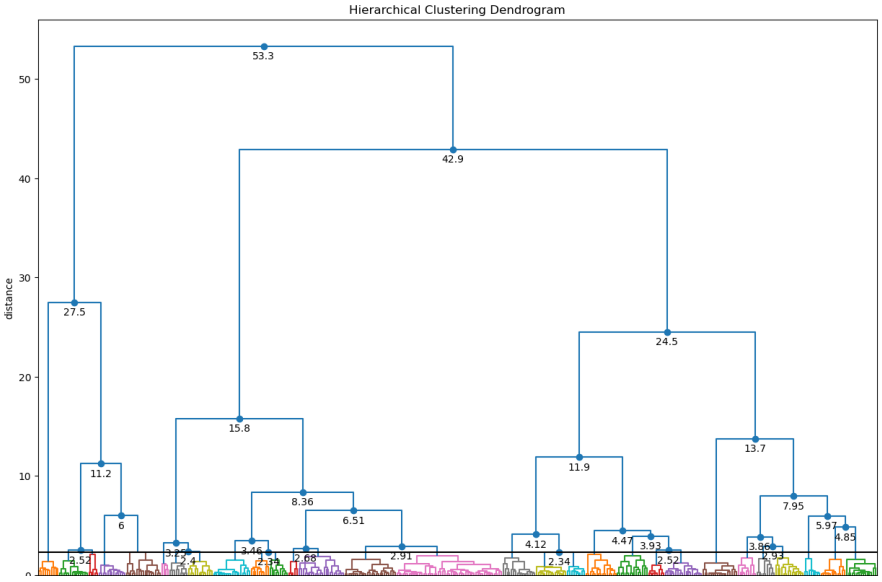

Figure 13: Dendrogram of HDBSCAN results

4.2.3 Final Allocation

After clustering algorithm is performed, the final weights are computed as described in Section 3.5 of the paper. Figure 14 shows the matrix of intracluster weights, sorted by the first column (cluster 0).

	0	1	2	3	4	5	6	7	8	9	...	40	41	42	43	44	45	46	47	48	49
United kingdom: FTSE Fixed Interest Up to 15 Years	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
United kingdom: FTSE Fixed Interest Up to 20 Years	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Germany: EB REXX Germany 5.5-10.5 PR	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
United kingdom: FTSE Fixed Interest 5 to 10 Years	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Sweden: OMRX Bond All 5 Y	0.19	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
United states: Nasdaq	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
United states: NYSE World Leaders Mini Value	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
United states: NYSE World Leaders	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Germany: EB REXX Germany 10.5+ PR	-0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
United kingdom: FTSE Fixed Interest Over 15 Years	-0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

403 rows × 50 columns

Figure 14: Matrix of intracluster weights

Figure 15 shows the horizontal bar chart of all assets whose weight in the final portfolio is greater or equal than 0.04.

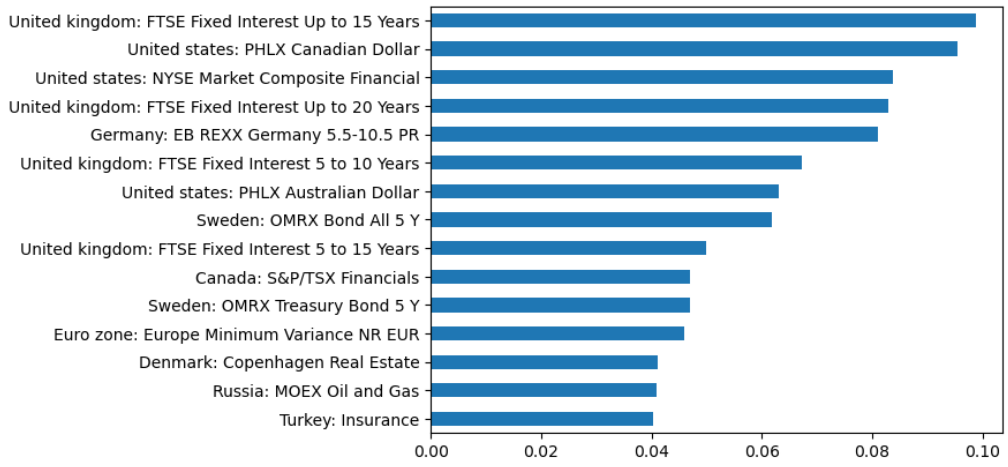

Figure 15: Top 15 weights of the portfolio

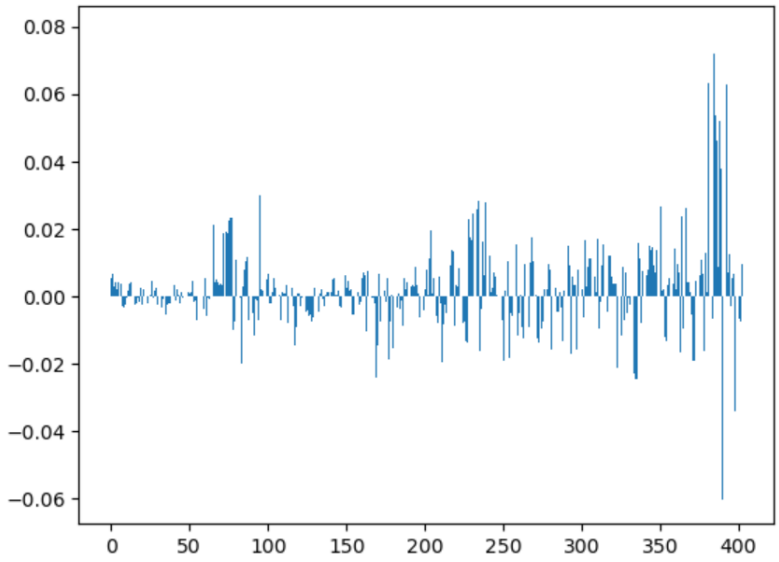

Figure 16: Bar chart of final weights allocation

The bar chart of all weights in the final portfolio are shown in Figure 16. Since there are 403 assets in the portfolio, x-axis shows numerical index of each.

4.3 Backtesting on Historical Data

Historically, backtesting trading strategies always was one of the most fundamental procedures in quantitative finance. Since the algorithm tested in this section differs from the original NCO, let us further refer to altered version of NCO as HDBSCAN NCO (HNCO). To evaluate the performance of HNCO, this section examines several approaches of backtesting.

4.3.1 Walk-Forward Backtesting

Walf-forward (WF) backtesting is a method that shows how a strategy would have performed in the past. In the real-world scenarios, portfolio strategies usually are recomputed every small fixed time interval, for example, week, to remain stable for changes in weekly returns.

To perform a backtest simulating close to real-world scenarios, given some number of weeks in the train set, on which the model is built, the test set will have only one week. As the table containing weekly returns has only 491 observations but covariance matrix denoising requires number of observations be greater than number of features, the length of all train sets is equal to 403. And the test sets are formed from the 404-th week to 491-th, having overall 82 iterations in which portfolio weights recomputed.

The performed backtesting had equal weighted and minvariance portfolios as benchmarks. The cumulative returns on all three portfolios are shown in Figure 17. As it can be seen from the diagram, HDBSCAN NCO outperforms minvariance and equal weighted portfolios in terms of cumulative returns on almost every point. In addition, let us pay attention on periods with falls in returns of all three portfolios, the interval from 17-th to 24-th weeks. Equal weighted portfolio suffered from this period the most, as it yielded the least returns and its cumulative returns even dropped below 1 after it. In contrast, minvariance portfolio, due to its conservative risk-minimising approach suffered the least loss among three portfolios. The HNCO portfolio also endured some losses, however, the magnitude of them was much less that that of equal weighted portfolio.

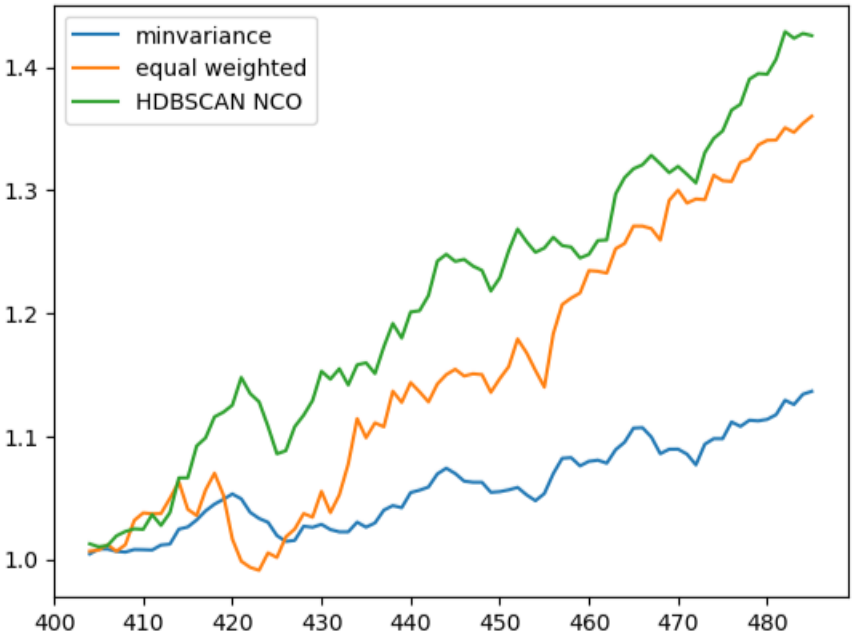

Figure 17: Cumulative returns of equal weighted, MinVariance and HDBSCAN NCO portfolios

The table in Figure 18 shows summary statistics of walk-forward backtesting. Minvariance portfolio yielded the least returns but also possesses the lowest std. deviation of returns among three portfolios. HNCO outperformed equal weighted portfolio in terms of both returns and risk, having higher net return on the portfolio over 82 weeks and lower volatility of it. To compute Sharpe-ratios of all portfolios the list of all weekly risk-free rates from july 2, 1926, to may 31, 2023, from "Kenneth R. French" data library [2] was used.

[2]The dataset contains values for factors SMB, HML and Market risk premia, and, additionally, weekly risk-free rates of return. Available via http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/Data_Library/f-f_factors.html

	Equal Weighted	MinVariance	HDBSCAN NCO
Net Return on Portfolio	0.3601995	0.1365364	0.4257348
Volatility (St.Dev.)	0.0136445	0.0062750	0.0120834
Sharpe Ratio	0.3153047	0.2509303	0.4221813

Figure 18: Metrics of portfolio performance

Walk-forward backtesting enjoys the following major advantages: firstly, it has a clear historical interpretation, since the train set always predates the test set; in addition, this approach ensures that the test set will always be out-of-sample, as it does not intersect with the training set.

However, several major drawbacks of this approach must be outlined: firslty, WF is carried on a single historical data, considers only one scenario, which can easily lead to model overfitting; additionally, backtest cannot be considered representative of future performance, as certain set of datapoints, such as anomalies, price movements due to local events and etc., are never guaranteed to be repeated in the future, making results biased towards these points.

Because of those issues, another backtesting approach called k-fold cross-validation backtesting was carried out which resolves the issue of single historical path being tested, however, has some drawbacks of its own.

4.3.2 K-Fold Cross-Validation Backtesting

The k-fold cross-validation backtesting procedure contains the following steps: firstly, some arbitrary or certain k number of weeks is selected; then, on every iteration of backtesting one of chosen weeks is treated as a test set, and some number of weeks in the data but the test set are chosen as the train set.

The idea of cross-validation backtesting is to test k different scenarios but only one of them would preserve the chronological order of the train and test sets.

The number of k-folds to be chosen is equal to the length of the returns table, to examine all possible testing sets. However, to preserve the training set properties as in WF-backtesting, the training sets on each iteration

will have a size of 403, being selected randomly from all the observations except the chosen test one. Figures 19, 20 and 21 show the results of cross-validation backtesting of three portfolios across 491 iterations using barcharts scaled to same y-axis limits.

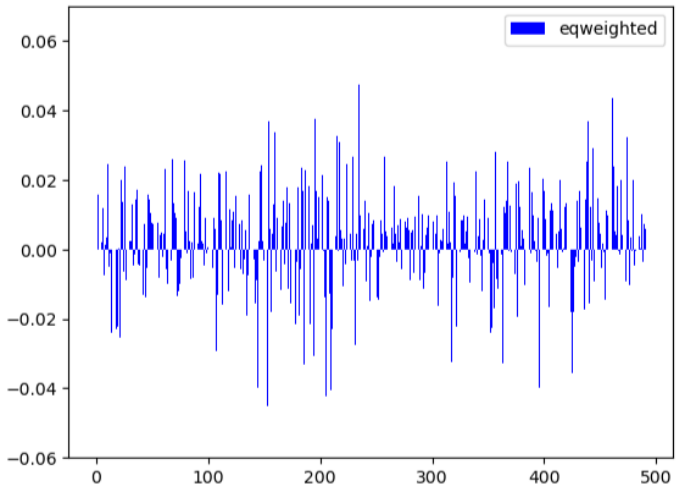

Figure 19: Bar chart of returns of equal-weighted portfolio across all iterations

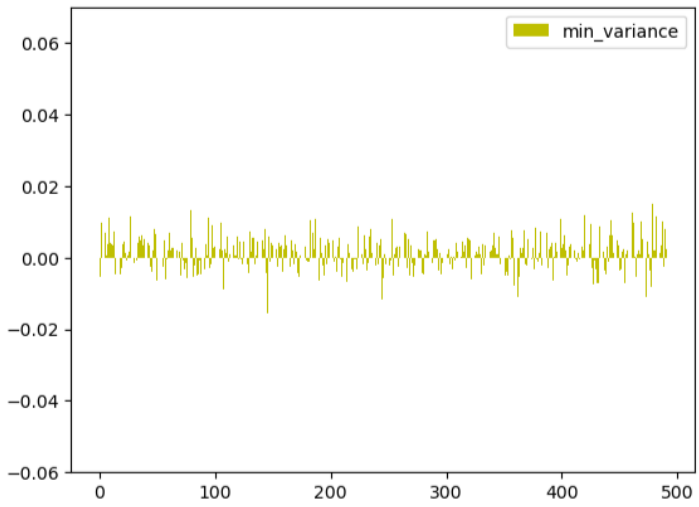

Figure 20: Bar chart of returns of minvariance portfolio across all iterations

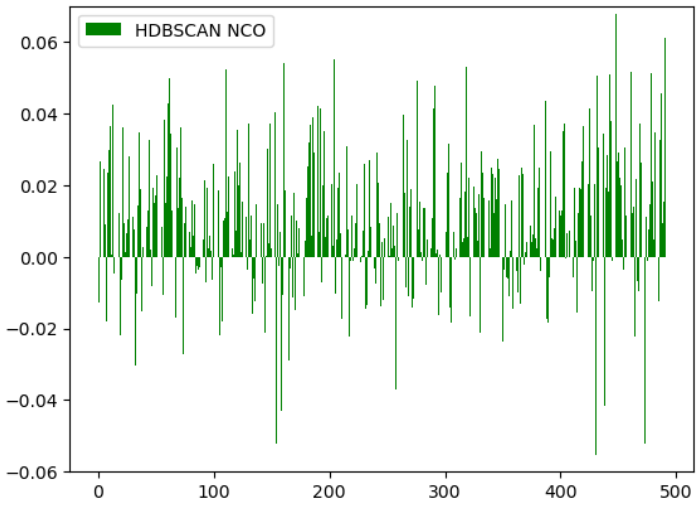

Figure 21: Bar chart of returns of HDBSCAN NCO portfolio across all iterations

Figure 22 shows the table where returns on every portfolio across all folds are shown. Rows in the table in Figure 22 shows the dates chosen to be the test week on cross-validation fold. As it is seen from the table,

HNCO portfolio yields the greatest mean return and sharpe ratio among three portfolios.

	eqweighted	minvariance	hnco	rf
2012-01-09	-0.0002169	-0.0054662	-0.0269182	0.0000000
2012-01-16	0.0156520	0.0096559	0.0159520	0.0000000
2012-01-23	0.0257426	-0.0025066	0.0236292	0.0000000
2012-01-30	-0.0011835	-0.0032906	-0.0094963	0.0000000
2012-02-06	0.0315657	0.0028439	0.0132830	0.0000100
...	...	...	...	...
2021-05-24	0.0073129	0.0078695	0.0214868	0.0000000
2021-05-31	0.0058332	0.0024258	0.0634678	0.0000000
Mean Return	0.0023265	0.0013462	0.0116655	NaN
Volatility (std.)	0.0142940	0.0045215	0.0182647	NaN
Sharpe Ratio	0.1542023	0.2706683	0.6319925	NaN

494 rows × 4 columns

Figure 22: Summary table of cross-validation backtesting results

5 Conclusion

In this paper, of the approaches to use the ML models in asset management problems called "nested clustered optimisation" was described. Since the clustering algorithm used in NCO had several drawbacks, affecting the theoretical performance of the algorithm, HDBSCAN algorithm was chosen instead of it. In addition, the explanation of several techniques to preprocessing covariance matrix were shown.

After that, the developed model was compared to two benchmarks, equal weighted and minvariance portfolios, firstly, in walk-forward backtesting procedure, which acknowledged the superiority of HNCO model over equally weighted portfolio; secondly, in cross-validation backtesting

procedure, which also resulted in HNCO yielding greater returns, although in this case, its volatility measure, standard deviation of returns was greater than those of benchmarks as well.

Nonetheless, two model extensions can be suggested:

1. HNCO and NCO use minvariance portfolio calculation for every cluster created and cluster assets. However, choosing any other portfolio optimisation model, for example, one of those shown in [5], might make the model perform significantly better.

2. Instead of $d(X,Y) = \sqrt{1/2 - \rho(X,Y)/2}$ distance metric, another correlation-based metric can be used. If such metric is chosen in a way to describe dissimilarity between two vectors (time series) of weekly returns better than aforementioned metric d, the clustering model will cluster assets more accurately.

6 References

1. Markowitz, H. (1952). Portfolio Selection. The Journal of Finance, 7(1), 77–91.

2. Laloux, L., Cizeau, P., Bouchaud, J. P., & Potters, M. (1999). Noise dressing of financial correlation matrices. Physical review letters, 83(7), 1467.

3. Peng, Y., Albuquerque, P. H. M., do Nascimento, I. F., & Machado, J. V. F. (2019). Between nonlinearities, complexity, and noises: An application on portfolio selection using kernel principal component analysis. Entropy, 21(4), 376.

4. Ledoit, O., & Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. Journal of multivariate analysis, 88(2), 365-411.

5. Kalayci, C. B., Ertenlice, O., & Akbay, M. A. (2019). A comprehensive review of deterministic models and applications for mean-variance portfolio optimization. Expert Systems with Applications, 125, 345-368.

6. De Prado, M. L. (2016). Building diversified portfolios that outperform out of sample. The Journal of Portfolio Management, 42(4), 59-69.

7. Campello, R. J., Moulavi, D., & Sander, J. (2013). Density-based clustering based on hierarchical density estimates. In Advances in Knowledge Discovery and Data Mining: 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Australia, April 14-17, 2013, Proceedings, Part II 17

(pp. 160-172). Springer Berlin Heidelberg.

8. De Prado, M. L. (2018). Advances in financial machine learning. John Wiley & Sons.

9. de Prado, M. M. L. (2020). Machine learning for asset managers. Cambridge University Press.

10. Hwang, S. G. (2004). Cauchy's interlace theorem for eigenvalues of Hermitian matrices. The American mathematical monthly, 111(2), 157-159. 2.

7 Appendix

7.1 Relationship between Correlation and Covariance Matrices

Consider some covariance matrix Σ and assets, X-s, indexed from 1 to n. The matrix $diag(\Sigma)^{-1/2}$ is then given by

$$\begin{pmatrix} 1/\sqrt{Var(X_1)} & 0 & \cdots & 0 \\ 0 & 1/\sqrt{Var(X_2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1/\sqrt{Var(X_n)} \end{pmatrix} = \begin{pmatrix} \sigma_1^{-1} & 0 & \cdots & 0 \\ 0 & \sigma_2^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \sigma_n^{-1} \end{pmatrix}.$$

Hence, the matrix multiplication

$$diag(\Sigma)^{-1/2} \times \Sigma \times diag(\Sigma)^{-1/2}$$

will result in some matrix ρ_M, where entry (ij) of it will be given by

$$(ij) = \frac{\sigma(X_i, X_j)}{\sigma_i \sigma_j} = \rho(X_i, X_j),$$

thus, ρ_M is the correlation matrix of the assets $X_1, ..., X_n$.

7.2 Proof of the Theorem

In this subsection I prove the theorem stated in Section 3.1.

Theorem. Consider a positive definite symmetric matrix C. Let $C_1, C_2, ..., C_n$ be a nested sequence of principal minors of matrix C. Then,

the condition numbers of principal minors of C satisfy the inequality: $\kappa(C_1) \leqslant \kappa(C_2) \leqslant ... \leqslant \kappa(C_n) = \kappa(C)$.

Proof. Let A_k be a $k \times k$ principal minor of A, where $k \leqslant n$. Let $0 < \lambda_1 \leqslant \lambda_2 \leqslant ... \leqslant \lambda_n$ be the eigenvalues of matrix A. Let $0 < \lambda'_1 \leqslant \lambda'_2 \leqslant ...\lambda'_k$ be the eigenvalues of matrix A_k. By Cauchy interlacing theorem [10], $\lambda_1 \leqslant \lambda'_1$ and $\lambda'_k \leqslant \lambda_n$. Hence, the following inequality holds:

$$\kappa(A_k) = \frac{\lambda'_k}{\lambda'_1} \leqslant \frac{\lambda_n}{\lambda_1} = \kappa(A).$$

Since the principal minor A_k was chosen arbitrarily, the inequality holds for all such principal minors where $0 < k \leqslant n$.

7.3 Estimation of Theoretical Maximum and Minimum Eigenvalues

Basically, to find theoretical minimum and maximum eigenvalues, one needs to apply the formulas for $\lambda_+, \lambda_- = (1 \pm \frac{N}{T})\sigma^2$ shown in Section 3.2. However, the value of σ is unknown for some empirical covariance matrix constructed. To find the theoretical magnitude of the variance of eigenvalues of covariance matrix, the following procedure is applied:

1. Firstly, one needs to compute the variance of eigenvalues of empirical covariance matrix use it to find the marsenko-pastur p.d.f. $p_{emp.}(\lambda)$ of empirical covariance matrix.

2. Then, letting $p(\lambda)$ be the analytical marsenko-pastur distribution of eigenvalues, one needs to fit it to the empirical p.d.f., i.e. find such value of σ which minimizes the function sum of squared standard errors $\sum_\lambda (p(\lambda) - p_{\text{emp.}}(\lambda))^2$.

Applying the same procedure to the empirical covariance matrix shown in Section 4.2, we get that $\sigma = 0.99999 \approx 1$. Figure 23 shows the plot of theoretical and empirical p.d.f.-s for $\sigma = 1$.

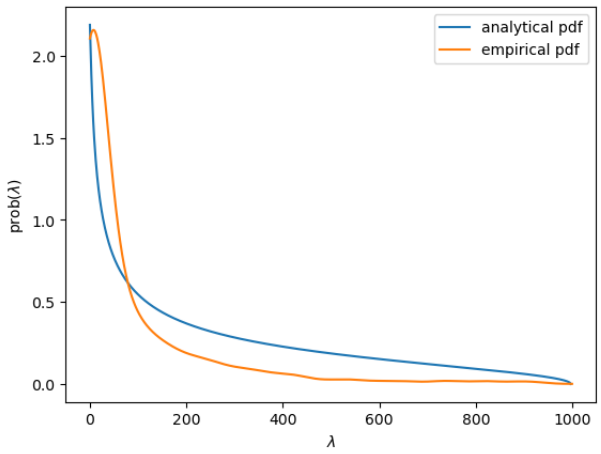

Figure 23: Theoretical and empirical eigenvalue probability density functions

7.4 Proximity Matrix Metric Justification

Let E be a set of all possible finite real-valued time-series. Considering function $d : E \times E \to [0, 1]$, $d(x, y) = \phi(\rho(x, y)) = \sqrt{1/2(1 - \rho(x, y))}$, where $x, y, z \in E$, we can show that d is a metric using a notion of a well-known metric called Euclidean distance.

The approach to prooving that $d(x, y)$ is a metric used in de Prado (2019) was the following:

The Euclidean distance between two vectors (two time series) x, y is given by $d(x, y) = ||x - y||_2$. Let $x' = \frac{x-\bar{x}}{\sigma_x}, y' = \frac{y-\bar{y}}{\sigma_y}$ be standardized vectors of x and y. Hence,

$$d(x', y') = ||x' - y'||_2 = \sqrt{\sum_{i=1}^{n}(x'_i)^2 + \sum_{i=1}^{n}(y'_i)^2 - 2\sum_{i=1}^{n} x'_i y'_i} =$$

$$= \sqrt{n + n - 2n \times \rho(x, y)}$$

Since correlation is not affected by standartization of two time series, $\rho(x, y) = \rho(x', y')$, so $d(x', y') = 4\sqrt{n} \times d(x, y)$. So as metric d between two time series is a product of some constant on euclidean distance between same but standardized time series, and euclidean distance is well-known a metric, we can conclude that d is also a metric.