Of the 28 projects done this semester, your project is the most directly relevant to actual portfolio management. Moreover, the care given to all steps in building the project, and the startling computational results, are unsurpassed in the 15 years of the course.

Clustering and Network-Based Portfolio Optimization with Shrinkage upon Risk Constraints

Kexin Deng(kd537), Olivia Guo(qg77), Weihang Lin(wl769)

A+

May 15, 2025

Abstract

This study develops and evaluates a set of portfolio construction strategies based on machine learning clustering and advanced optimization techniques for SP 500 stocks. We employ a rolling-window backtesting framework using weekly returns from resampled daily close prices. Stock selection is driven by clustering algorithms including Hierarchical Clustering, Partitioning Around Medoids (PAM), K-Means, DBSCAN, Gaussian Mixture Models (GMM), and Minimum Spanning Tree (MST). Within each cluster, top assets are selected based on historical Sharpe ratios.

Two portfolio optimization approaches are applied: Mean variance optimization using Ledoit-Wolf shrinkage estimators for robust covariance matrix estimation and expected shortfall (CVaR) minimization to capture downside risk. Each strategy is evaluated under both mean-variance and equally weighted schemes. The results are compared against the SP 500 benchmark in terms of cumulative returns and key performance metrics including annualized Sharpe ratio, Sortino ratio, volatility, CAGR, maximum drawdown, and Calmar ratio.

The framework highlights how clustering-enhanced stock selection and shrinkage-based optimization can improve portfolio risk-adjusted performance while adapting to market structure changes.

1 Introduction

This project enhances the classical Markowitz Mean-Variance Optimization (MVO) framework by integrating data-driven stock selection and advanced risk management techniques. Traditional MVO is known to suffer from estimation errors, particularly in high-dimensional settings with unstable covariance estimates. To address this, we introduce clustering-based methods that reduce dimensionality, improve estimation accuracy, and promote diversification. ✓

We evaluate a variety of unsupervised learning algorithms for stock selection, including K-Means, Gaussian Mixture Models (GMM), Partitioning Around Medoids (PAM), hierarchical clustering, and network-based techniques such as the Minimum Spanning Tree (MST). These clustering methods allow us to segment the investment universe into structurally similar groups, from which top-performing stocks are selected based on historical Sharpe ratios. Our backtesting framework implements dynamic portfolio construction using weekly returns and monthly rebalancing. We compare the performance of different stock selection strategies with the SP 500 Index. ✓

To further strengthen risk management, we incorporate Conditional Value-at-Risk (CVaR) minimization as an alternative to traditional variance-based optimization. This approach better captures downside tail risk and improves robustness in turbulent market conditions. In addition, we evaluate the effect of shrinkage estimators for more stable covariance estimation.

Overall, our study proposes a practical and machine-learning-based portfolio construction framework that adapts to changing market conditions and reduces estimation risk—offering improved risk-adjusted performance for managing large-scale equity portfolios.

2 Literature Review

Modern portfolio theory, pioneered by Markowitz (1952), introduced the mean-variance optimization (MVO) framework, a cornerstone of asset allocation strategies. However, its practical implementation often suffers due to sensitivity to input estimation errors, especially when dealing with high-dimensional covariance matrices. To mitigate this, recent research has emphasized dimensionality reduction and robust risk estimation, particularly through clustering and shrinkage methods. ✓

Clustering techniques have emerged as powerful tools for asset selection, enabling dimensionality reduction while preserving important structural information within financial markets. Approaches such as K-Means, Hierarchical Clustering, Partitioning Around Medoids (PAM), and Gaussian Mixture Models (GMM) have been employed to segment stocks into homogeneous groups based on return similarities or correlation structures. These clusters facilitate diversified selection by representing distinct market behaviors. For example, Cao et al. (2023) explored both clustering and network-based techniques—like Louvain and Minimum Spanning Tree (MST)—to enhance diversification and improve risk-adjusted performance through structured asset grouping. Quantpedia's analysis further supports this view, demonstrating that clustering-driven strategies, particularly those using PAM and hierarchical methods, outperform simple equal-weighted portfolios when applied with thoughtful intra- and inter-cluster allocation schemes. The silhouette method is often used to dynamically determine optimal cluster numbers, ensuring adaptability across varying market conditions.

The instability of the sample covariance matrix in MVO has led to the adoption of shrinkage estimators, particularly the Ledoit-Wolf estimator, which provides a well-conditioned and more stable covariance matrix by shrinking sample estimates toward a structured target. This significantly improves the out-of-sample performance of optimized portfolios, especially when combined with dimensionality reduction techniques. Studies show that shrinkage estimators help mitigate overfitting and enhance robustness in dynamic portfolio construction settings. Zhang et al. (2024) and DeMiguel et al. (2012) demonstrate that shrinkage methods, including those using option-implied information, reduce noise and enhance performance, particularly in large portfolios where estimation errors are most pronounced.

In response to the limitations of variance as a risk measure—particularly its symmetric treatment of upside and downside risk—Conditional Value-at-Risk (CVaR) has gained prominence as a coherent and downside-focused risk metric. CVaR captures expected losses in the tail of the return distribution and is especially valuable in stressed market environments. Rockafellar and Uryasev's framework laid the foundation for incorporating CVaR into optimization, enabling portfolio managers to directly control tail risk. However, CVaR optimization is not without its challenges. Lim et al. (2011) caution that CVaR-based portfolios can be fragile due to estimation errors in the tails of the return distribution, which are magnified by the optimization process. In response, Ban et al. (2018) introduce performance-based regularization (PBR), which improves stability by constraining estimation error propagation through convex approximations and k-fold cross-validation.

Recent efforts in portfolio construction combine clustering for selection and shrinkage/CVaR for optimization. For instance, nested clustering (e.g., hierarchical followed by K-means) can improve signal stability, while regularized optimization frameworks with CVaR constraints further reduce drawdowns. Moreover, using Gaussian Mixture Models (GMMs) not only aids in stock clustering but also in identifying market regimes, which enhances dynamic parameter estimation in rolling-window backtests. Collectively, these advancements provide a data-driven alternative to classical MVO. By integrating clustering, shrinkage, and CVaR, portfolio construction becomes more robust, diversified, and adaptive—better aligned with the complex and changing nature of financial markets.

3 Data Overview

This section describes the dataset and structural design of our asset universe, as well as the benchmark portfolios used to evaluate our strategies. The study begins with a broad and representative selection of 500 U.S. stocks to ensure market coverage and sectoral balance.

3.1 Raw Data Description

The asset universe consists of 500 individual U.S. equities chosen for their liquidity, economic diversity, and data reliability. The list of tickers was obtained from the publicly maintained S&P 500 constituents list on Wikipedia[1], reflecting the real-time composition of the index. The S&P 500 Index is a free-float market capitalization-weighted index that includes approximately 80% of the total value of the U.S. equity market. Maintained by S&P Dow Jones Indices, it comprises 500 leading publicly traded companies across all major sectors of the U.S. economy.

We retrieved historical weekly closing prices for each stock using the `yfinance` API. The time window for this study spans from January 1, 2020, to January 1, 2023. This three-year interval captures a variety

[1] `https://en.wikipedia.org/wiki/List_of_S%26P_500_companies`

of market conditions including pandemic-induced volatility, recovery, and inflationary pressures. Weekly data was chosen to balance noise reduction for portfolio rebalancing and statistical estimation.

To maintain quality in the dataset, stocks with excessive missing data were removed. Specifically, any stock missing more than 50% of weekly price records was excluded. The remaining dataset includes only stocks with consistent and reliable price histories suitable for portfolio optimization.

From the cleaned price data, we compute weekly returns via percentage change calculations. These return series form the foundation for future clustering, portfolio optimization, and performance analysis.

3.2 Sectoral Distribution

The data set we use reflects the current constituents of the S&P 500 and spans the 11 sectors of the Global Industry Classification Standard (GICS). Information Technology, Health Care, Financials, Consumer Discretionary, Communication Services, Industrials, Consumer Staples, Energy, Utilities, Real Estate, and Materials. This broad sectoral coverage ensures that our portfolio selection methods are tested against a structurally representative cross section of the US market.

Figure 1 presents a sector-level breakdown of the number of companies in the index. Although the number of constituents varies by sector, all major economic domains are represented, offering a diversified base for analysis.

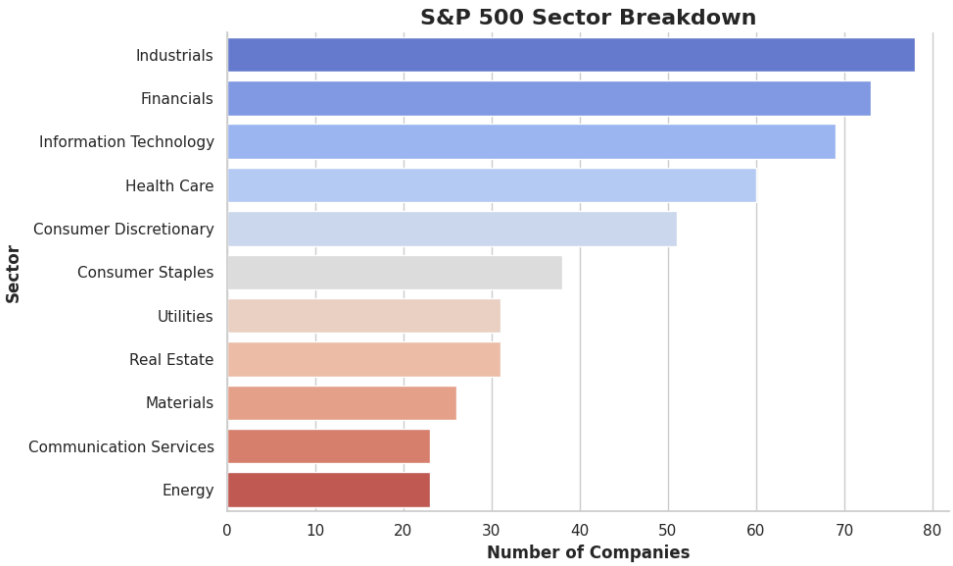


Figure 1: Number of S&P 500 constituent companies by GICS sector.

4 Methodology

4.1 Clustering and Network Algorithm for Stock Selection

To summarize, each clustering or network-based algorithm contributes a unique perspective on the structure of asset co-movements. These structured subsets of the asset universe serve as inputs to the portfolio optimization stage, where diversification and risk control are prioritized. Table 1 provides an overview of the distinguishing characteristics and motivations for each method.

informative table!

Table 1: Summary of Clustering and Network Algorithm and Motivations

Method	Key Characteristic	Motivation for Use
K-Means	Hard clustering via centroids	Fast, scalable for large datasets; simple to implement and interpret. Reduces universe based on return similarity.
PAM (Partitioning Around Medoids)	Robust clustering using real data points (medoids)	Resistant to outliers; provides actual stock representatives rather than synthetic centers. Suitable for volatile return data.
Gaussian Mixture Model (GMM)	Probabilistic clustering; soft assignment of stocks	Captures overlapping clusters and latent structures. Useful when stock relationships are not sharply separated.
DBSCAN Clustering	Density-based clustering; identifies noise and outliers	Detects clusters of arbitrary shape; does not require specifying number of clusters. Effective in handling noise and sparse data regions.
Hierarchical Clustering (Russian Doll)	Tree-based, multi-resolution structure	Reveals nested relationships. Allows flexible granularity in asset grouping and is adaptable for recursive selection.
Minimum Spanning Tree (MST)	Sparse topology preserving strong correlations	Highlights key structural dependencies among stocks while reducing dimensionality for optimization input.

4.1.1 K-Means Clustering

The K-Means clustering algorithm partitions a dataset into k distinct non-overlapping clusters by minimizing the within-cluster variance. It aims to assign each data point to the cluster whose centroid (mean vector) is closest, thereby reducing the total squared distance from each point to its assigned cluster center.

Let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^T$ be the set of return vectors for n stocks over T time periods (e.g., weekly returns). $k \in \mathbb{N}$ be the number of clusters. $\mu_j \in \mathbb{R}^T$ denote the centroid (mean vector) of cluster j, for $j = 1, \ldots, k$.

We aim to assign each point $\mathbf{x}_i$ to one of k clusters such that the total within-cluster squared Euclidean distance is minimized:

$$\min_{\{\mathcal{C}_1,\ldots,\mathcal{C}_k\}} \sum_{j=1}^{k} \sum_{\mathbf{x}_i \in \mathcal{C}_j} \|\mathbf{x}_i - \mu_j\|^2$$

where $\mu_j = \frac{1}{|\mathcal{C}_j|} \sum_{\mathbf{x}_i \in \mathcal{C}_j} \mathbf{x}_i$ is the mean of points assigned to cluster $\mathcal{C}_j$.

4.1.2 Partitioning Around Medoids

The Partitioning Around Medoids algorithm searches for k representative objects in a data set (k medoids) and then assigns each object to the closest medoid in order to create clusters. Its aim is to minimize the sum of dissimilarities between the objects in a cluster and the center of the same cluster (medoid).

Let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^T$ be the set of return vectors for n stocks over T time periods (e.g., weekly returns). $d(\mathbf{x}_i, \mathbf{x}_j)$ be a dissimilarity function (e.g., Euclidean or correlation distance). $k \in \mathbb{N}$ be the number of clusters.

We aim to choose a subset of k representative points (medoids) $\mathcal{M} = \{\mathbf{m}_1, \ldots, \mathbf{m}_k\} \subset \mathcal{X}$ such that the total within-cluster dissimilarity is minimized:

$$\min_{\mathcal{M} \subset \mathcal{X}, |\mathcal{M}|=k} \sum_{i=1}^{n} \min_{\mathbf{m} \in \mathcal{M}} d(\mathbf{x}_i, \mathbf{m})$$

Each data point $\mathbf{x}_i$ is assigned to its closest medoid:

$$C(\mathbf{x}_i) = \arg\min_{\mathbf{m} \in \mathcal{M}} d(\mathbf{x}_i, \mathbf{m})$$

4.1.3 Gaussian Mixture Model

The Gaussian Mixture Model (GMM) is a probabilistic clustering algorithm that assumes the data is generated from a mixture of several multivariate Gaussian distributions with unknown parameters.

Let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^T$ be the set of return vectors for n stocks over T time periods. $k \in \mathbb{N}$ be the number of Gaussian components (clusters). π_j be the mixture weight of component j such that $\sum_{j=1}^{k} \pi_j = 1$ and $\pi_j > 0$. $\mu_j \in \mathbb{R}^T$ and $\Sigma_j \in \mathbb{R}^{T \times T}$ be the mean and covariance matrix of component j, respectively.

The probability density of $\mathbf{x}_i$ under the mixture model is:

$$p(\mathbf{x}_i) = \sum_{j=1}^{k} \pi_j \cdot \mathcal{N}(\mathbf{x}_i \mid \mu_j, \Sigma_j)$$

The model parameters $\{\pi_j, \mu_j, \Sigma_j\}_{j=1}^{k}$ are estimated via the Expectation-Maximization algorithm. Each data point is then assigned to the cluster with the highest posterior probability:

$$C(\mathbf{x}_i) = \arg\max_j \gamma_{ij}$$

4.1.4 DBSCAN

The Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm is a non-parametric clustering method that groups stocks based on local density, making it particularly effective for discovering arbitrarily shaped clusters and identifying outliers. Given a set of return vectors $X = \{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^T$, DBSCAN requires two parameters: the neighborhood radius ε and the minimum number of points required to form a dense region, typically denoted as `min_samples`. A point is classified as a core point if it has at least `min_samples` neighbors within a distance ε; points within this radius of a core point are part of the same cluster. Points that do not meet either condition are treated as noise and excluded from clustering.

In our implementation, we dynamically determine the optimal value of ε for each training window using a k-nearest neighbors distance curve, followed by the `KneeLocator` method to identify the elbow point. This adaptive selection ensures that DBSCAN remains robust across varying market conditions and avoids reliance on fixed thresholds.

Once clustering is performed, we filter out noise points labeled as -1 and identify the top-performing assets within each cluster based on their historical Sharpe ratios. Specifically, we select the two stocks with the highest Sharpe values per cluster for inclusion in the portfolio. These selected assets are subsequently passed to the portfolio optimization stage, where either mean-variance optimization or expected shortfall minimization is applied.

DBSCAN's ability to handle noise and detect non-spherical, unevenly distributed clusters enhances the flexibility of our asset selection framework. This is especially valuable in financial markets, where asset relationships may not conform to traditional geometric assumptions. As a result, DBSCAN contributes meaningfully to diversification and tail-risk control within the overall portfolio construction pipeline.

4.1.5 Hierarchical Clustering

Let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^T$: set of return vectors for n stocks over T time periods. $d(\mathbf{x}_i, \mathbf{x}_j)$: dissimilarity between stocks $\mathbf{x}_i$ and $\mathbf{x}_j$, e.g.,

$$d(\mathbf{x}_i, \mathbf{x}_j) = 1 - \text{corr}(\mathbf{x}_i, \mathbf{x}_j)$$

or the Euclidean distance:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|_2$$

To build a hierarchy of nested clusters $\{\mathcal{C}^{(t)}\}_{t=0}^{n-1}$, where:

- $\mathcal{C}^{(0)} = \{\{\mathbf{x}_1\}, \ldots, \{\mathbf{x}_n\}\}$
- $\mathcal{C}^{(n-1)} = \{\mathcal{X}\}$

At each step t, merge the two closest clusters $A, B \in \mathcal{C}^{(t)}$:

$$(A^*, B^*) = \arg \min_{A \neq B \in \mathcal{C}^{(t)}} D(A, B)$$

$$\mathcal{C}^{(t+1)} = \Big(\mathcal{C}^{(t)} \setminus \{A^*, B^*\}\Big) \cup \{A^* \cup B^*\}$$

Choose a linkage method to define inter-cluster distance:

- **Ward's linkage:**
  Cluster Dissimilarity
  $$D(A, B) = \frac{|A||B|}{|A| + |B|} \|\boldsymbol{\mu}_A - \boldsymbol{\mu}_B\|^2$$
  where $\boldsymbol{\mu}_A$ is the centroid of cluster A.

4.1.6 Minimum Spanning Tree

The Minimum Spanning Tree (MST) provides a parsimonious, tree-structured summary of the full correlation network by retaining only the strongest connections that ensure connectivity. Starting from the same weighted graph $G = (V, E)$ with weights

$$w_{ij} = \sqrt{2\big(1 - \rho_{ij}\big)},$$

where ρ_{ij} is the Pearson correlation between weekly returns, we define a distance metric

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

Algorithm: Compute the full $n \times n$ distance matrix $D = [d_{ij}]$. Apply Kruskal's (or Prim's) algorithm to extract the MST $\mathcal{T} \subset E$, minimizing $\sum_{(i,j) \in \mathcal{T}} d_{ij}$ subject to connectivity and $|\mathcal{T}| = |V| - 1$. Represent $\mathcal{T}$ as a graph in NetworkX or via SciPy's `minimum_spanning_tree` routine.

Once the MST is built, we explore three heuristics to pick 11 assets from the MST for portfolio construction.

Method 1: Simple Average Dissimilarity In the first approach we compute, for each node i, the average distance to all other assets,

$$\bar{d}_i = \frac{1}{n-1} \sum_{j \neq i} d_{ij},$$

and then rank assets by decreasing $\bar{d}_i$. Implementation is trivial—simply take the row-mean of the $n \times n$ distance matrix—and runs in $O(n^2)$ time. This method excels in its transparency and computational speed, quickly highlighting those stocks that are most "peripheral" on average. However, by ignoring the MST's topology it tends to pick a cluster of extreme outliers that lie on a single branch, often yielding intra-set correlations above 0.5(see Fig.2). Its tree representation further reveals all red nodes confined to one sector of the MST (Fig.3), underscoring the method's limited coverage and suboptimal diversification.

Moreover, because this method does not respect network structure or contextual grouping, it can exhibit substantial flip-flopping of positions from one rebalancing period to the next. A stock that appears marginally more distant in one month may be supplanted by another equally extreme outlier in the next, resulting in high portfolio turnover. High turnover amplifies transaction costs and may erode net returns, especially when bid–ask spreads and market impact are factored in. Practitioners must therefore balance the benefit of capturing shifting dispersion patterns against the drag of trading frictions, possibly by incorporating drift-tolerance thresholds or minimum holding periods to mitigate excessive churn.

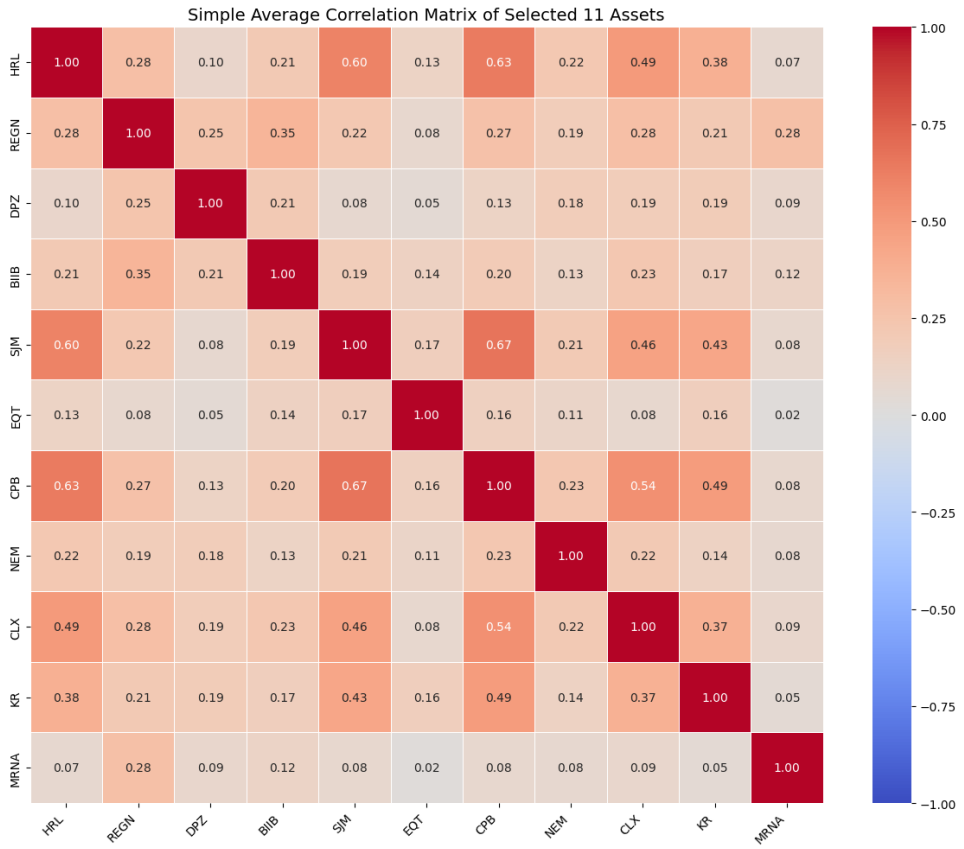


Figure 2: Simple Average Correlation Matrix of the 11 Selected Assets (Method 1).

Figure 3: MST of S&P 500 Stocks (Red Nodes: Top 11 Dissimilar Assets) using Simple Average (Method 1).

Method 2: MST-Degree-Weighted Dissimilarity The second heuristic respects the MST's sparse connectivity by averaging each node's incident edge weights in $\mathcal{T}$:

$$\bar{d}_i^{\text{MST}} = \frac{1}{\deg_{\mathcal{T}}(i)} \sum_{(i,j)\in\mathcal{T}} d_{ij}.$$

Because the MST has only $n-1$ edges, computing these per-node averages is $O(n)$ once the MST is built. This approach retains the strongest relationships while emphasizing nodes whose immediate neighbors are comparatively distant. Empirically, it yields an 11-asset set whose pairwise correlations all lie below 0.5 and whose red-node markers are scattered across multiple branches of the tree (see Fig.4). The corresponding edge-weighted correlation heatmap in Fig.5 further confirms low intra-cluster dependence. Its main strength lies in honoring the MST backbone and thereby capturing structurally significant outliers. Its drawback is that it discards potentially useful off-tree edges, and a single very large MST edge can unduly inflate one node's ranking.

From an financial interpretation standpoint, MST edges often correspond to sectoral or macro-factor linkages—stocks in the same industry or subject to the same macro drivers tend to cluster together. By weighting nodes according to the average dissimilarity of their MST connections, we implicitly favor assets that are under-represented in dominant clusters. For example, if technology names dominate one branch, degree-weighted picks may surface stocks in real estate, utilities, or emerging markets that would otherwise be overlooked. This geometric selection thus augments statistical diversification with economic breadth, ensuring that the portfolio spans distinct factor exposures rather than merely maximizing raw distance metrics.

Figure 4: MST of S&P 500 Stocks (Red Nodes: Top 11 Dissimilar Assets) using MST-Degree-Weighted Dissimilarity (Method 2).

Method 3: Greedy "Farthest-Point" Insertion The third method forsakes the tree entirely and applies a classic maximin rule: begin with the single asset having the highest $\bar{d}_i$, then iteratively add the stock k whose distance to the current selected set S,

$$\min_{j\in S} d_{kj},$$

is maximal. Each insertion costs $O(n)$, and selecting 11 assets takes $O(11\,n)$. This greedy farthest-point algorithm guarantees that every new asset is as far as possible from those already chosen, producing

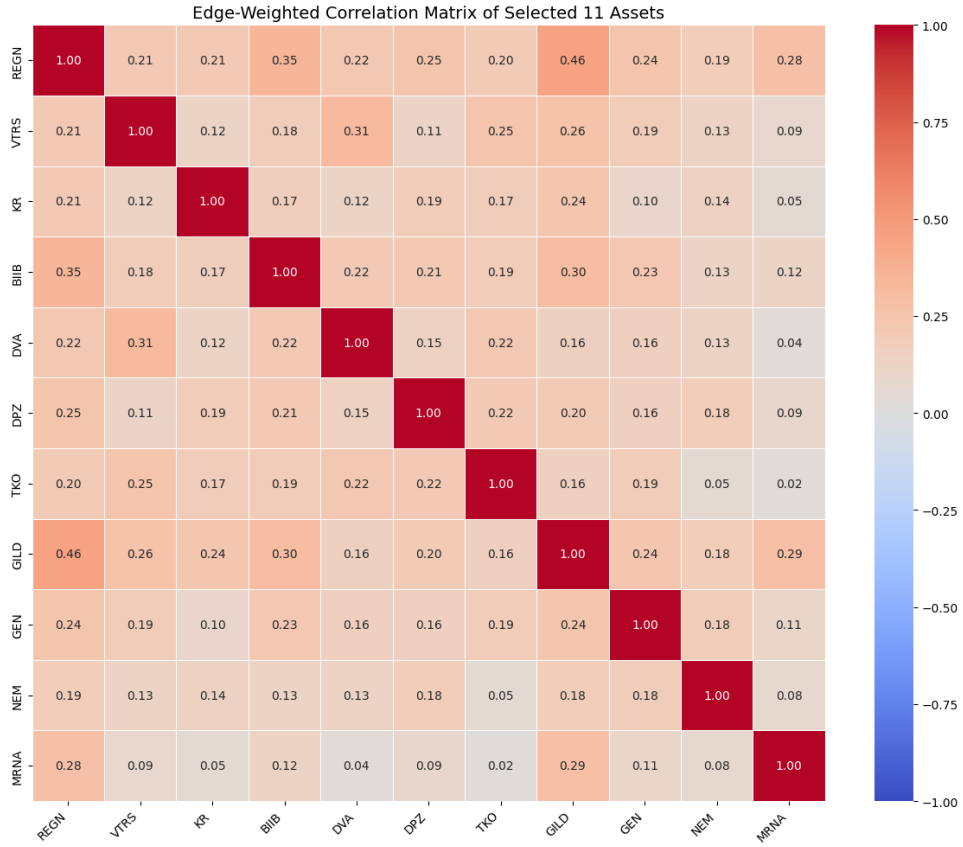


Figure 5: Edge-Weighted Correlation Matrix of the 11 Selected Assets (Method 2).

the lowest observed pairwise correlations across all three methods (Fig.13 and Fig.7). Yet because it ignores the MST's economic network, it may bypass moderately distant clusters that taken together would improve risk diversification.

In addition, another notable limitation of this approach is its disregard for common factor exposures. While it optimizes pairwise distances, it may inadvertently pick stocks that share a latent risk factor—such as value, momentum, or interest-rate sensitivity—if those dimensions do not manifest in the raw correlation metric. To guard against unintended factor concentration, one can augment the selection with a risk-budgeting overlay: after each greedy insertion, evaluate candidate factor loadings (e.g. via a Fama-French regression) and impose a constraint that the incremental factor exposure not exceed a predetermined threshold. This hybrid ensures that the portfolio achieves maximal statistical dispersion while maintaining balanced exposure across known systematic risks.

Comparative Discussion All three heuristics aim to reduce a 500-asset universe to a manageable 11 while maximizing diversification. The Simple Average method is the easiest to implement but suffers from branch clustering, higher intra-portfolio correlations, and potential estimation noise. MST-Degree-Weighted combines network structure with computational efficiency, delivering a well-scattered, low-correlation set by focusing on the MST backbone, though it ignores non-tree edges and may overweight single-edge extremes. The Greedy Farthest-Point insertion attains the lowest pairwise correlations by construction and has known approximation guarantees, yet it departs entirely from the MST topology and may neglect economic factor exposures. Table 2 summarizes these three heuristics—highlighting their focus, key strengths, and primary weaknesses—providing a clear reference to guide the choice of selection method.

Figure 6: MST of S&P 500 Stocks (Red Nodes: Top 11 Dissimilar Assets) using Greedy Farthest-Point Insertion (Method 3).

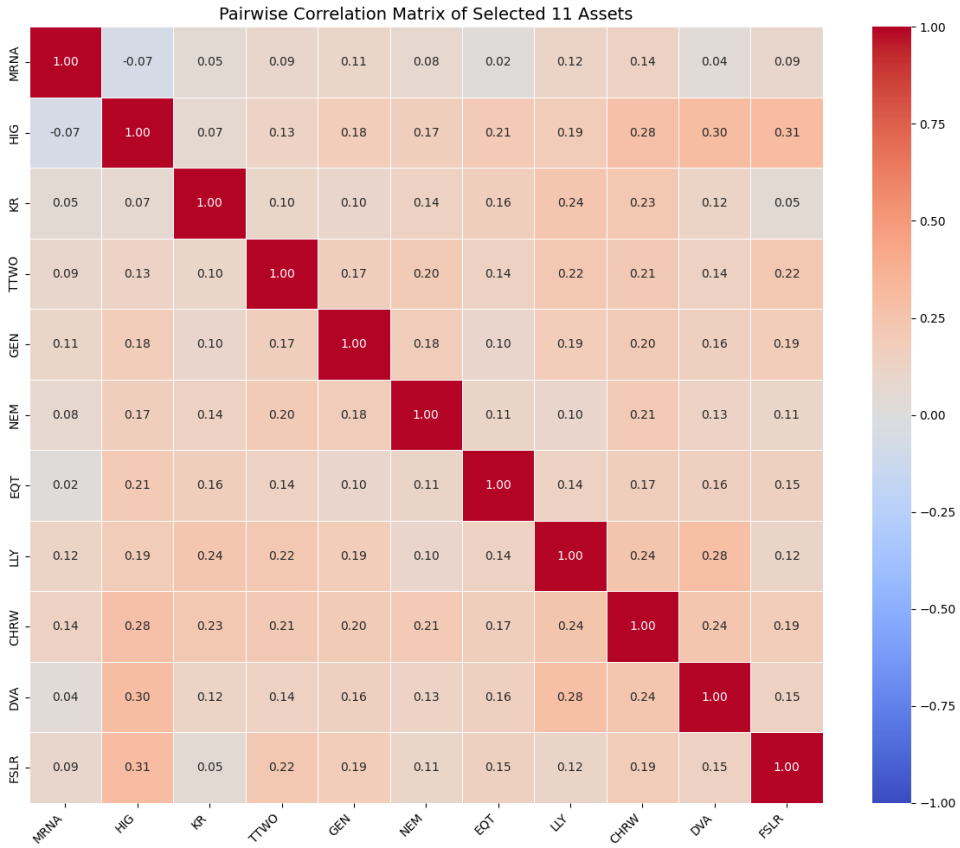


Figure 7: Pairwise Correlation Matrix of the 11 Selected Assets (Method 3).

Table 2: Comparison of MST-Based Selection Heuristics

Method	Focus	Key Strength	Key Weakness
Simple Average	Global distance mean	Ultra-simple; picks globally distant stocks	Clusters on one branch; higher correlations
MST-Degree-Weighted	Local MST edge averages	Honors tree topology; good branch coverage	Ignores non-MST edges; single-edge bias
Farthest-Point	Greedy maximin spread	Maximizes pairwise diversification	Greedy; neglects MST economic structure

In practice, the MST-Degree-Weighted approach strikes the best balance between respecting the underlying correlation network, controlling turnover, and achieving broad branch coverage—

4.2 Optimization Method

4.2.1 Mean-Variance Portfolio Optimization

The Mean-Variance Portfolio Optimization (MVPO) framework, introduced by Markowitz, forms the foundation of modern portfolio theory.

Let $\mathbf{r}_i \in \mathbb{R}^T$ be the vector of weekly returns for asset i, for $i = 1, \ldots, n$. $\mu \in \mathbb{R}^n$ be the vector of expected asset returns, estimated as the historical mean. $\Sigma \in \mathbb{R}^{n \times n}$ be the asset return covariance matrix, estimated using the Ledoit-Wolf shrinkage estimator to stabilize the sample covariance in the presence of noisy and limited data. $\mathbf{w} \in \mathbb{R}^n$ be the portfolio weight vector.

The investor seeks to maximize the trade-off between return and risk, leading to the following quadratic optimization problem:

$$\max_{\mathbf{w}} \quad \mu^\top \mathbf{w} - \lambda \cdot \mathbf{w}^\top \Sigma \mathbf{w}$$

subject to:

$$\sum_{i=1}^{n} w_i = 1, \quad w_i \geq 0 \quad \forall i$$

where λ is the risk-aversion parameter, controlling the balance between expected return and portfolio variance.

4.2.2 Shrinkage Estimation

To mitigate estimation error and enhance the numerical stability of portfolio optimization, we employ the Ledoit-Wolf shrinkage estimator for the covariance matrix Σ. The estimator constructs a convex combination of the sample covariance matrix S and a structured target matrix F, typically chosen as the identity matrix scaled by the average variance:

$$\hat{\Sigma} = \delta F + (1 - \delta) S,$$

where $\delta \in [0, 1]$ is the shrinkage intensity, optimally determined to minimize mean squared error. The shrinkage estimator balances the empirical accuracy of S with the stability of F, producing a more robust estimate of the covariance matrix, particularly when the number of observations is small relative to the number of assets.

In our implementation, $\hat{\Sigma}$ is used in the mean-variance optimization problem.

4.2.3 Expected Shortfall (CvaR) Incorporation

To better capture tail risk in portfolio construction, we implement an Expected Shortfall (ES) optimization framework—also known as Conditional Value-at-Risk (CVaR).

Given a return matrix $X \in \mathbb{R}^{T \times n}$ for n assets over T time periods, we denote the portfolio weights by $w \in \mathbb{R}^n$ and define the portfolio return vector as $R = Xw$. We formulate the empirical ES optimization at confidence level $\alpha \in (0, 1)$ as the following convex program:

$$
\begin{aligned}
\min_{w,\eta,z} \quad & \eta + \frac{1}{(1-\alpha)T} \sum_{t=1}^{T} z_t \\
\text{s.t.} \quad & z_t \geq 0, \quad z_t \geq -x_t^\top w - \eta, \quad \forall t \in \{1, \ldots, T\}, \\
& \sum_{i=1}^{n} w_i = 1, \quad w_i \geq 0 \quad \forall i.
\end{aligned}
$$

Here, η approximates the Value-at-Risk (VaR), and z_t are slack variables that absorb violations beyond this threshold. The objective function corresponds to the empirical estimate of the CVaR, which aggregates average tail losses.

5 Implementation Details

During each rolling-window period, we apply clustering to partition the asset universe, select top-performing stocks from each cluster based on historical Sharpe ratios, first incorporated shrinkage, then considered risk constraints of Expected Shortfall (CVaR).

5.1 Rebalancing Schedule and Rolling Window Framework

To closely mimic institutional asset management practices, we adopt a realistic rolling-window backtesting framework. Each portfolio is rebalanced at the end of every calendar month, with portfolio weights computed using data from the most recent three-month (13-week) training window. Following optimization, the resulting portfolio is held for one month before the rebalancing process is repeated. This setup ensures that all optimization relies solely on historical information, eliminating any possibility of look-ahead bias. Furthermore, it enables dynamic adjustment to shifting market conditions while preserving statistical robustness over medium-term horizons.

5.2 Clustering and Asset Selection Process

To reduce dimensionality and promote diversification, we segment the investment universe using several unsupervised clustering methods, including PAM, GMM, K-Means, DBSCAN, Hierarchical Clustering, and network based method - MST.

For most clustering algorithms, we manually set the number of clusters to four to maintain consistency across methods. An exception is made for Hierarchical, DBSCAN and GMM, where the number of clusters is automatically determined based on dendrogram, density- or likelihood-based criteria, respectively. Within each identified cluster, we select the top two assets based on their historical Sharpe ratios over the training period. These top performers are then used as inputs for later portfolio optimization.

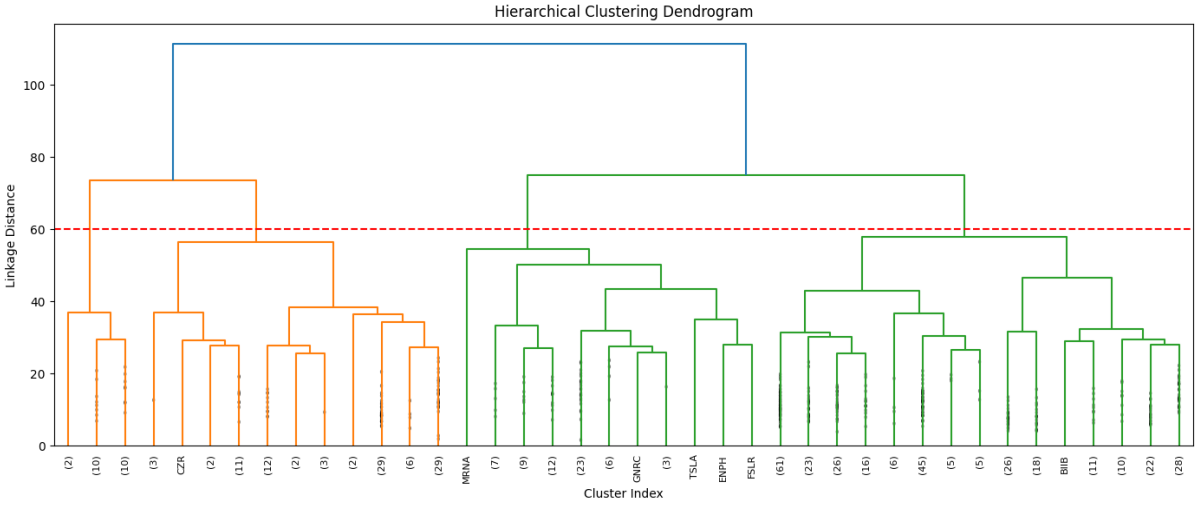


Figure 8: Hierarchical clustering dendrogram, which shows the distance between the clusters

In this method, we apply Ward's linkage criterion to form a nested tree structure of asset similarity. A fixed depth cutoff is used to extract four terminal clusters. The horizontal lines in the dendrogram represent the merging of clusters, while the height at which they are joined indicates the distance between them. By applying a uniform cutoff across rebalancing periods, we maintain temporal consistency in cluster formation while allowing room for economically meaningful intra-cluster variation.

Also, for MST, we artificially selected top 11 assets, aligning with number of sectors, sorted by node-weighted average that minimizes the overall correlation of all the assets within the portfolio, which in turn maximizes diversification and reduces volatility as a while.

6 Result and Discussion

6.1 Basic Clustering MVO and Equal Weighted without Shrinkage

6.1.1 Comprehensive Cumulative Return Comparison

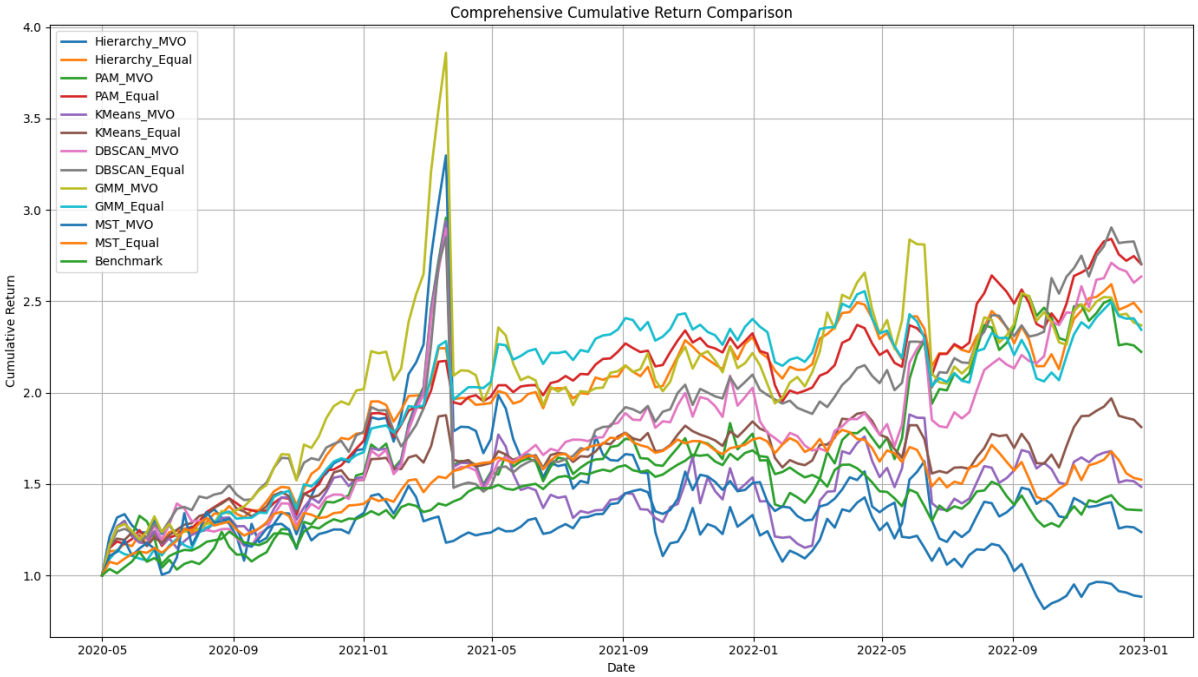


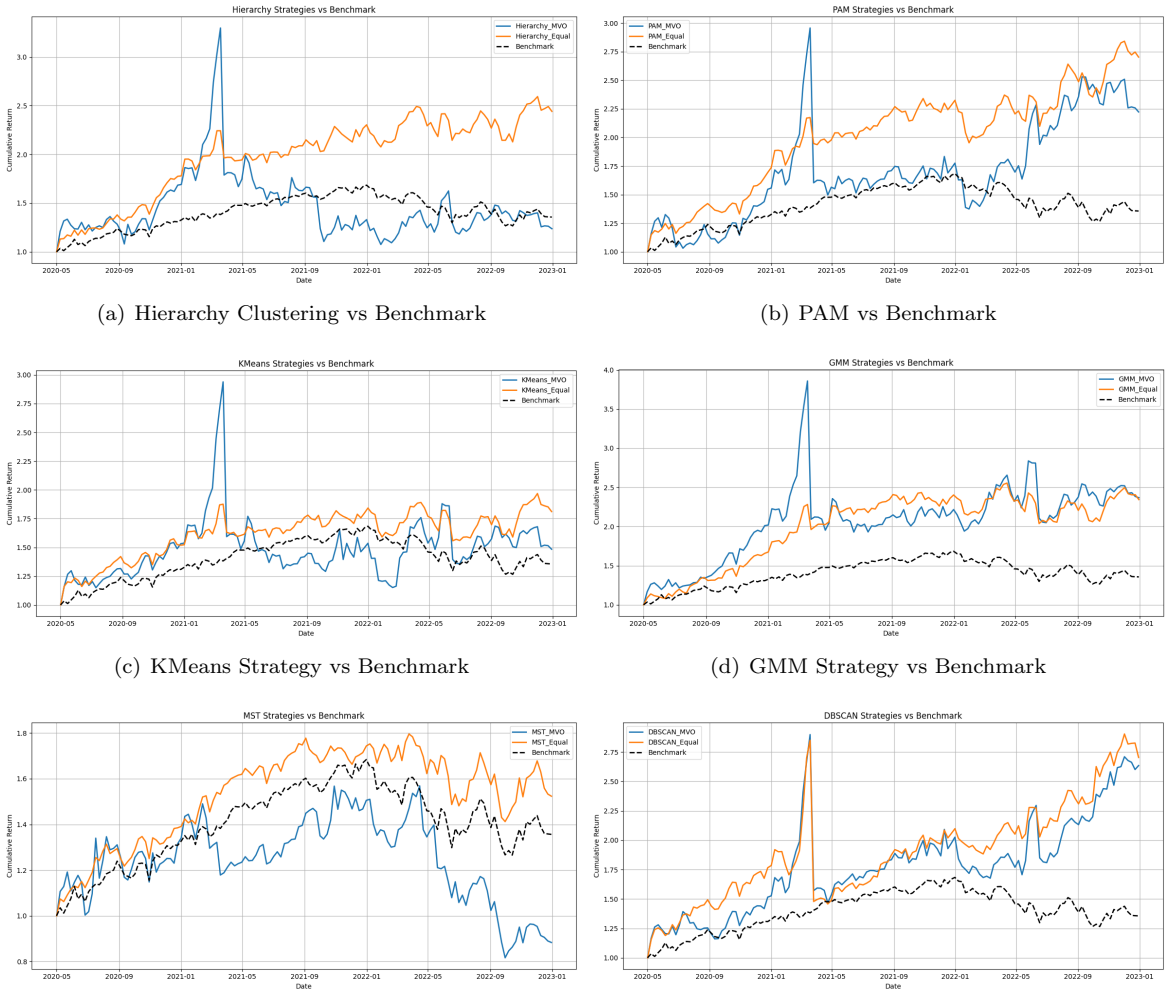
Figure 9: Comprehensive Cumulative Return Comparison

6.1.2 Performance Metrics Tabulation

Table 3: Strategy Performance Metrics

Strategy	Sharpe	Volatility	Max Drawdown	Sortino	CAGR	Calmar
PAM_Equal	1.565	0.252	-0.165	2.368	0.447	2.699
Hierarchy_Equal	1.383	0.259	-0.146	2.144	0.393	2.686
GMM_Equal	1.283	0.269	-0.203	1.852	0.372	1.836
DBSCAN_Equal	1.111	0.427	-0.489	0.996	0.447	0.915
DBSCAN_MVO	1.023	0.470	-0.494	1.048	0.433	0.877
KMeans_Equal	0.931	0.266	-0.176	1.389	0.247	1.400
GMM_MVO	0.915	0.497	-0.500	0.935	0.377	0.755
PAM_MVO	0.859	0.505	-0.535	0.914	0.345	0.645
MST_Equal	0.833	0.201	-0.214	1.349	0.169	0.793
Benchmark	0.646	0.188	-0.248	1.113	0.120	0.483
KMeans_MVO	0.557	0.543	-0.608	0.605	0.158	0.260
Hierarchy_MVO	0.435	0.570	-0.674	0.521	0.082	0.122
MST_MVO	0.036	0.375	-0.480	0.051	-0.045	-0.094

6.1.3 Different Strategies vs Benchmark



(a) Hierarchy Clustering vs Benchmark

(b) PAM vs Benchmark

(c) KMeans Strategy vs Benchmark

(d) GMM Strategy vs Benchmark

(e) MST Strategy vs Benchmark

(f) DBSCAN Strategy vs Benchmark

Figure 10: Comparison of Six Strategy Plots

6.2 With Tuned Shrinkage MVO Portfolio

6.2.1 Comprehensive Cumulative Return Comparison

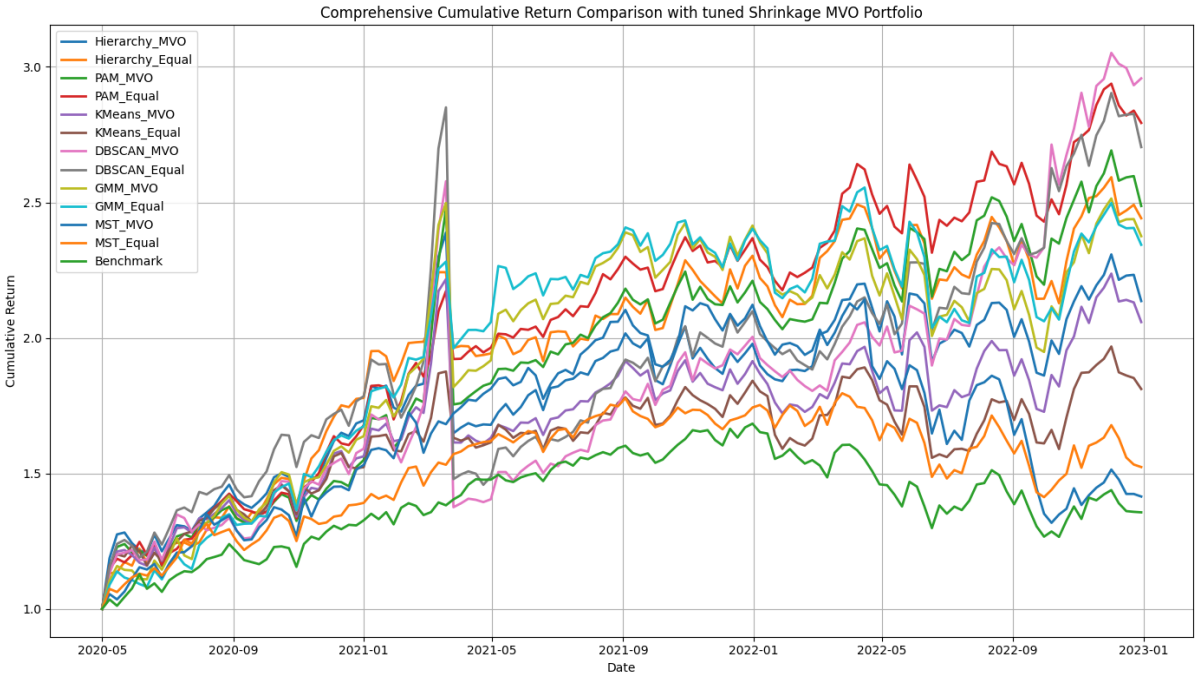


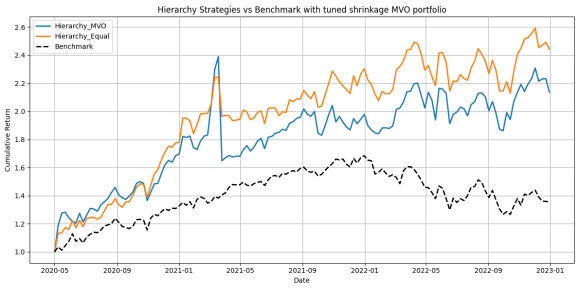
Figure 11: Comprehensive Cumulative Return Comparison - With Shrinkage

6.2.2 Performance Metrics Tabulation

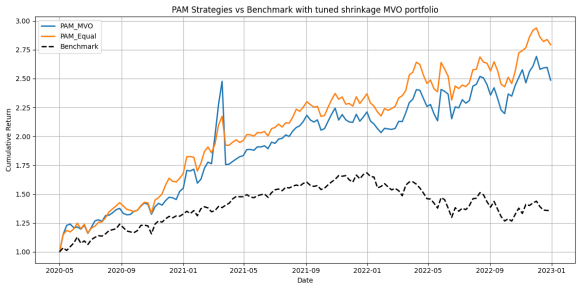
Table 4: Updated Strategy Performance Metrics - Shrinkage

Strategy	Sharpe	Volatility	Max Drawdown	Sortino	CAGR	Calmar
PAM_Equal	1.662	0.244	-0.124	2.618	0.465	3.750
Hierarchy_Equal	1.383	0.259	-0.146	2.144	0.393	2.686
GMM_Equal	1.283	0.269	-0.203	1.852	0.372	1.836
DBSCAN_MVO	1.204	0.417	-0.466	1.102	0.496	1.063
PAM_MVO	1.197	0.323	-0.291	1.298	0.403	1.382
GMM_MVO	1.143	0.323	-0.272	1.328	0.379	1.393
DBSCAN_Equal	1.111	0.427	-0.489	0.996	0.447	0.915
Hierarchy_MVO	0.989	0.340	-0.310	1.033	0.326	1.051
KMeans_MVO	0.978	0.321	-0.274	1.126	0.308	1.124
KMeans_Equal	0.931	0.266	-0.176	1.389	0.247	1.400
MST_Equal	0.833	0.201	-0.214	1.349	0.169	0.793
Benchmark	0.646	0.188	-0.248	1.113	0.120	0.483
MST_MVO	0.595	0.257	-0.385	0.859	0.138	0.357

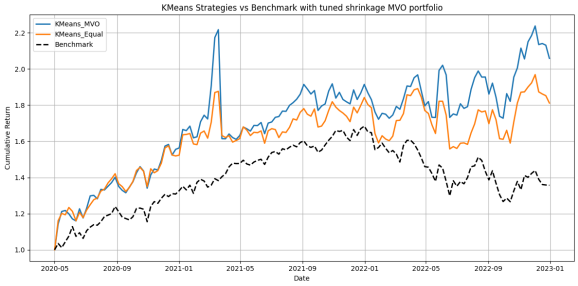
6.2.3 Different Strategies vs Benchmark



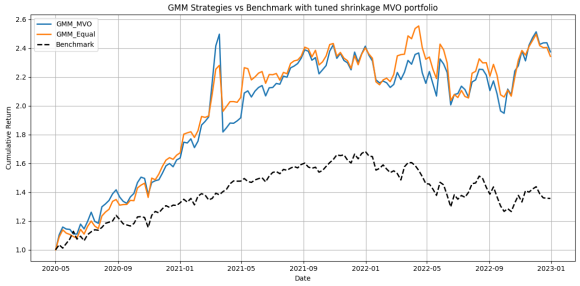
(a) Hierarchy vs Benchmark with Tuned Shrinkage



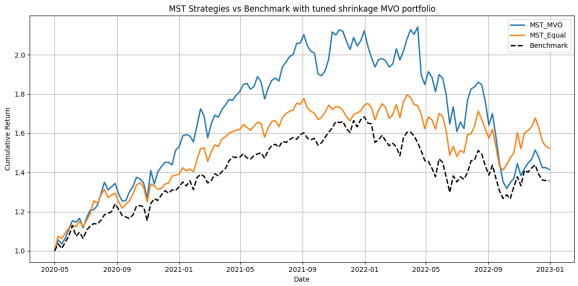
(b) PAM vs Benchmark with Tuned Shrinkage



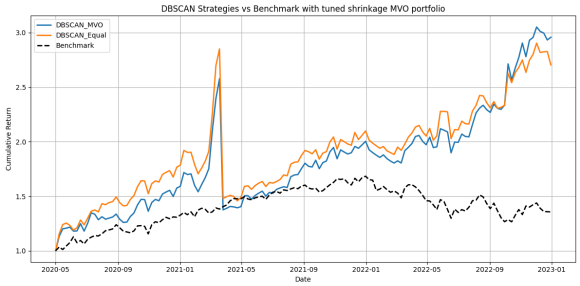
(c) KMeans vs Benchmark with Tuned Shrinkage



(d) GMM Strategy vs Benchmark with Tuned Shrinkage



(e) MST Strategy vs Benchmark with Tuned Shrinkage



(f) DBSCAN vs Benchmark with Tuned Shrinkage

Figure 12: Comparison of Six Strategy Plots

6.3 With Tuned Shrinkage-Expected Shortfall Portfolio (CVaR)

6.3.1 Comprehensive Cumulative Return Comparison

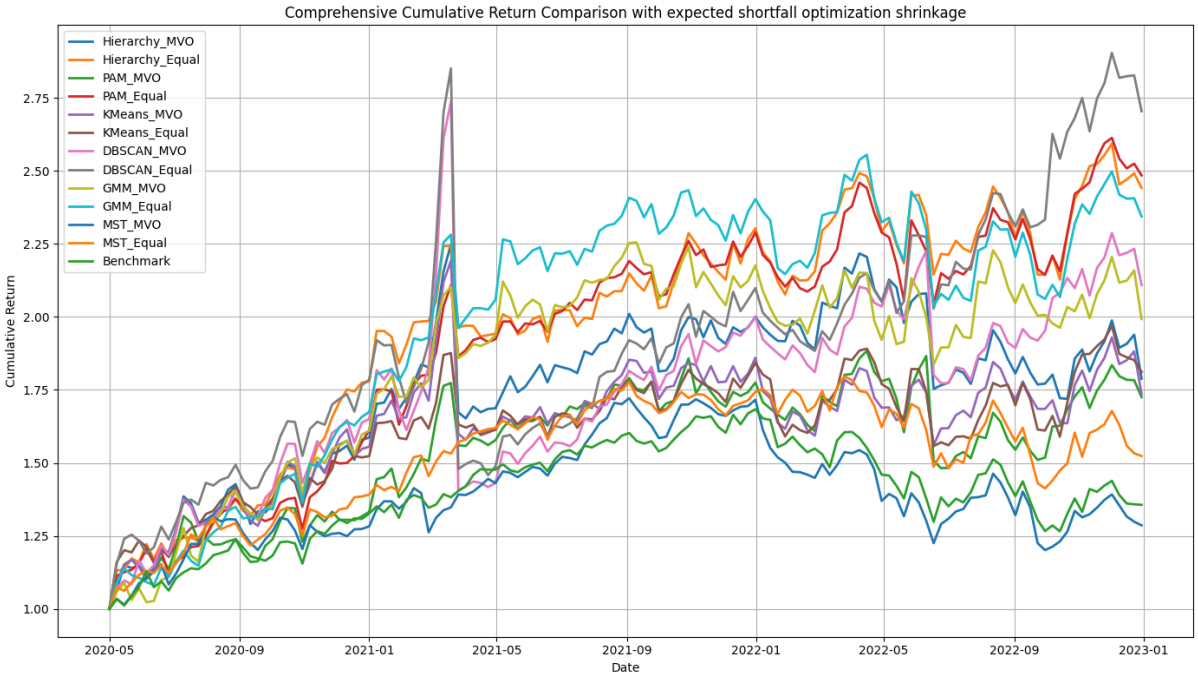


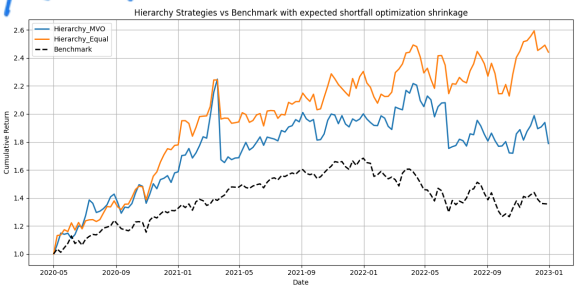
Figure 13: Comprehensive Cumulative Return Comparison - With Tuned Shrinkage-ES Portfolio (CVaR)

6.3.2 Performance Metrics Tabulation

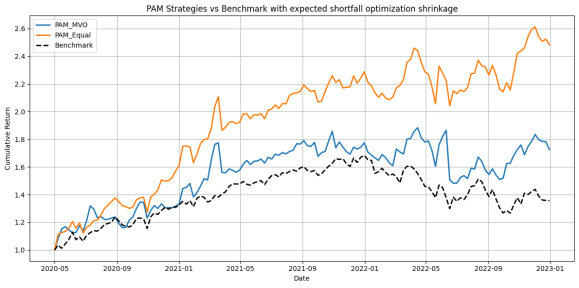
Table 5: Updated Strategy Performance Metrics - With Tuned Shrinkage-Expected Shortfall (CVaR)

Strategy	Sharpe	Volatility	Max Drawdown	Sortino	CAGR	Calmar
PAM_Equal	1.477	0.244	-0.170	2.169	0.402	2.371
Hierarchy_Equal	1.383	0.259	-0.146	2.144	0.393	2.686
GMM_Equal	1.283	0.269	-0.203	1.852	0.372	1.836
DBSCAN_Equal	1.111	0.427	-0.489	0.996	0.447	0.915
GMM_MVO	1.009	0.286	-0.185	1.667	0.292	1.575
KMeans_Equal	0.931	0.266	-0.176	1.389	0.247	1.400
DBSCAN_MVO	0.886	0.437	-0.487	0.800	0.319	0.656
PAM_MVO	0.843	0.277	-0.213	1.045	0.225	1.054
MST_Equal	0.833	0.201	-0.214	1.349	0.169	0.793
Hierarchy_MVO	0.830	0.312	-0.265	0.911	0.241	0.909
KMeans_MVO	0.787	0.317	-0.288	0.909	0.227	0.790
Benchmark	0.646	0.188	-0.248	1.113	0.120	0.483
MST_MVO	0.504	0.212	-0.302	0.700	0.098	0.324

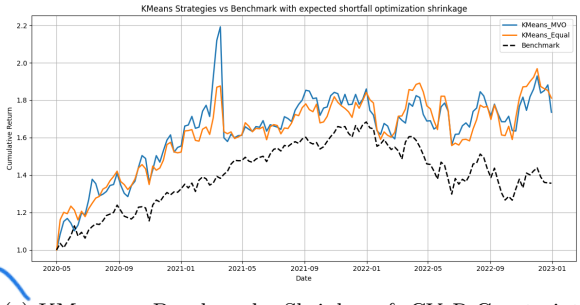
6.3.3 Different Strategies vs Benchmark



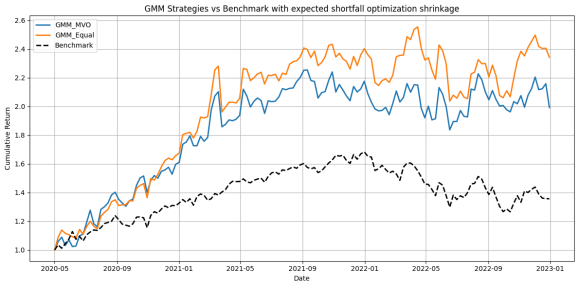
(a) Hierarchy vs Benchmark - Shrinkage & CVaR Constraint



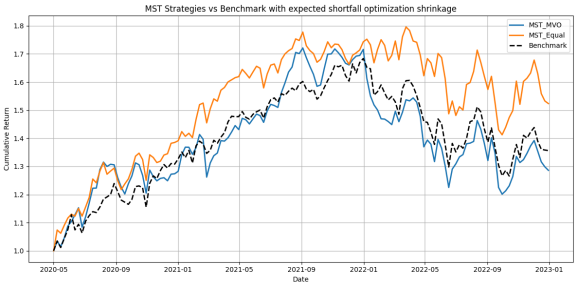
(b) PAM vs Benchmark - Shrinkage & CVaR Constraint



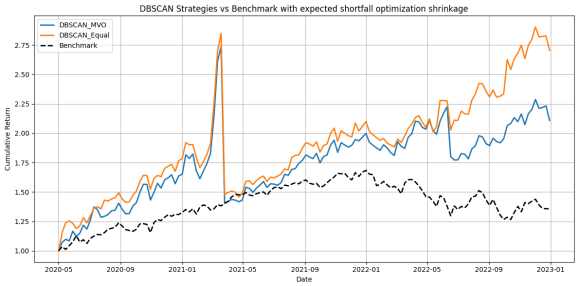
(c) KMeans vs Benchmark - Shrinkage & CVaR Constraint



(d) GMM Strategy vs Benchmark - Shrinkage & CVaR Constraint



(e) MST Strategy vs Benchmark - Shrinkage & CVaR Constraint



(f) DBSCAN vs Benchmark - Shrinkage & CVaR Constraint

Figure 14: Comparison of Six Strategy Plots

6.4 Result Summary and Analysis

6.4.1 Performance Metrics

To evaluate the effectiveness of different portfolio construction strategies, we compute a comprehensive set of performance metrics under the our different allocations, applied to out-of-sample returns generated from a rolling-window backtesting procedure.

The performance metrics include annualized Sharpe Ratio, volatility, maximum drawdown, Sortino Ratio, compound annual growth rate, and Calmar Ratio.

- **Sharpe Ratio**: Annualized mean excess return divided by annualized volatility, measuring overall risk-adjusted performance.
- **Volatility**: Annualized standard deviation of weekly portfolio returns, representing total return dispersion.
- **Maximum Drawdown (MDD)**: Largest peak-to-trough decline in cumulative returns, capturing downside risk severity.
- **Sortino Ratio**: Similar to Sharpe but considers only downside volatility, isolating harmful fluctuations.

- **Compound Annual Growth Rate (CAGR)**: Geometric average annual return over the backtest period.
- **Calmar Ratio**: Ratio of CAGR to maximum drawdown, representing return per unit of drawdown risk.

6.4.2 Logic of Model Evolution

Our framework evaluates portfolio construction across three successive models.

Model 1 (Section 6.1) uses clustering-based stock selection combined with traditional Mean-Variance Optimization (MVO) or equal-weighting. Covariance matrices are directly estimated from historical returns, without shrinkage or tail risk awareness. This setup establishes the initial performance.

Model 2 (Section 6.2) introduces Ledoit-Wolf shrinkage to stabilize covariance estimation. This adjustment is implemented in the `mean_variance_opt` function and is consistently applied across rolling windows. Comparing this section with Section 6.1 isolates the effect of shrinkage on MVO robustness and performance.

Model 3 (Section 6.3) further enhances the second by optimizing Conditional Value-at-Risk (CVaR) using a CVXPY-based convex program. The shift from variance to downside risk aligns better with investor aversion to large drawdowns. Section 6.3 presents this model, enabling comparison with the shrinkage-only setup in Section 6.2.

6.4.3 Model Comparison I: Non-Shrinkage vs. Shrinkage MVO (Section 6.1 vs Section 6.2)

Applying shrinkage estimation substantially improves performance metrics and stability. Cumulative return plots (Figures 9 vs. 11) show noticeable outperformance, especially in strategies like PAM_MVO and DBSCAN_MVO. For example, PAM_MVO's Sharpe ratio increases from 0.859 to 1.197, and Calmar ratio improves from 0.645 to 1.382.

In Table 3 (Section 6.1), many MVO portfolios underperform their equal-weight counterparts, highlighting the instability of naive covariance estimates. However, in Table 4 (Section 6.2), shrinkage flips this relationship—MVO strategies now consistently outperform equal-weighted versions across Sharpe, Sortino, and CAGR metrics.

Figure 10 and Figure 12 (strategy vs. benchmark comparisons) confirm this shift visually. Shrinkage-based portfolios trace smoother, more upward-trending paths, with reduced volatility and improved benchmark dominance.

6.4.4 Model Comparison II: Shrinkage MVO vs. Shrinkage-CVaR Optimization (Section 6.2 vs Section 6.3)

The CVaR-constrained model (Section 6.3) further enhances downside risk control. As seen in Table 5, while Sharpe ratios dip slightly for some strategies (e.g., PAM_MVO from 1.197 to 0.843), max drawdowns and Sortino ratios generally improve.

DBSCAN_MVO is particularly telling: although CAGR drops modestly from 0.496 to 0.319, Sortino and drawdown metrics indicate better resilience. MST_MVO and Hierarchy_MVO also demonstrate improved drawdown protection under CVaR (see Figure 13 vs. Figure 11).

The cumulative return curves (Figure 13) are visibly smoother, suggesting that CVaR models successfully contain tail risk and limit downside spikes. This robustness is especially valuable during volatile or turbulent periods, as captured in our rolling window backtest logic.

6.4.5 Key Takeaways Across Models

Our results highlight several important findings across the three model configurations evaluated in this study. First, the introduction of Ledoit-Wolf shrinkage consistently improves the performance of mean-variance optimization (MVO). By stabilizing the covariance matrix estimates, shrinkage enables optimized portfolios to outperform both their equal-weighted counterparts and the S&P 500 benchmark across a range of clustering methods. This effect is particularly evident in the substantial improvement in Sharpe and Calmar ratios observed in shrinkage-based models.

Second, incorporating Conditional Value-at-Risk (CVaR) constraints further enhances portfolio robustness, particularly in volatile market conditions. Although the addition of CVaR may slightly reduce Sharpe ratios in some cases, it delivers meaningful reductions in maximum drawdowns and improves

downside-adjusted metrics such as the Sortino ratio. This trade-off underscores CVaR's role in managing tail risk at a modest performance cost.

Among the clustering algorithms tested, PAM and DBSCAN emerge as the most effective across all model variants. These methods achieve strong diversification and clear cluster separation, which translates into superior asset selection and portfolio stability. Their performance remains robust regardless of the optimization objective or risk constraints applied.

Finally, the best-performing strategies—particularly PAM_MVO and DBSCAN_MVO under both shrinkage and CVaR—consistently dominate the benchmark in cumulative return plots. These strategies exhibit smoother return paths, lower volatility, and higher risk-adjusted returns, demonstrating the practical value of combining clustering-based stock selection with modern risk management techniques.

7 Conclusion

This research proposes a comprehensive portfolio optimization framework that combines clustering-based stock selection with robust risk modeling techniques, demonstrating its effectiveness across multiple market conditions. By structuring the asset universe using methods such as PAM, DBSCAN, GMM, and MST, and selecting top assets based on historical Sharpe ratios, we reduce dimensionality and enhance diversification prior to optimization. Our three-tiered approach—baseline mean-variance optimization, shrinkage-augmented optimization, and CVaR-constrained models—allows for a controlled assessment of how each enhancement contributes to portfolio performance.

Empirical results strongly support the role of shrinkage estimation in improving out-of-sample stability and returns. The introduction of Ledoit-Wolf shrinkage consistently elevates Sharpe ratios across strategies; for instance, PAM_MVO's Sharpe ratio rises from 0.859 (without shrinkage) to 1.197, while DBSCAN_MVO improves from 1.023 to 1.204. This effect is echoed across Sortino, Calmar, and CAGR metrics, confirming that shrinkage substantially mitigates covariance estimation noise and enhances mean-variance optimization. The final layer—CVaR-constrained optimization—adds further resilience by reducing maximum drawdowns and controlling tail risk, particularly in turbulent regimes.

Across all model tiers, the best-performing strategies, such as PAM_MVO and DBSCAN_MVO, achieve substantial risk-adjusted outperformance over the S&P 500 benchmark, with Sharpe ratios exceeding 1.2 compared to the benchmark's 0.65. The cumulative return plots further reveal smoother trajectories, lower volatility, and enhanced downside protection for these strategies. Notably, while MST-based strategies do not top the rankings in Sharpe or CAGR, they consistently deliver higher-than-benchmark returns with significantly lower volatility and modest drawdowns. This makes MST especially appealing for investors with low risk tolerance or those subject to mark-to-market constraints, where minimizing short-term fluctuations is paramount.

Although the inclusion of shrinkage and CVaR improves robustness, it also introduces additional modeling complexity. Calibrating parameters such as shrinkage intensity, CVaR confidence level, and clustering configuration requires care to avoid overfitting or performance decay. Nevertheless, the trade-offs are manageable for institutional investors and provide a high return on analytical effort.

Given its scalability, model transparency, and modular structure, the proposed framework offers a practical and deployable solution for asset managers seeking to enhance traditional long-only equity portfolios. Its dynamic rebalancing and structural adaptability make it particularly attractive in volatile or regime-shifting markets.

By unifying recent advances in clustering, shrinkage estimation, and tail-risk modeling, our study extends the literature beyond traditional MVO, aligning with the trends highlighted by Ban et al. (2018), Zhang et al. (2024), and Rockafellar and Uryasev (2000). In doing so, it contributes a coherent, well-tested methodology for constructing robust, risk-aware equity portfolios.

8 Future Work

8.1 Unsuccessful Attempt: Louvain Method for Stock Selection

In our exploration of network-based clustering techniques, we implemented the Louvain method—a popular community detection algorithm from network science—as a candidate for asset selection. The Louvain algorithm partitions graphs into communities by maximizing modularity, a quality metric that favors dense intra-group connections and sparse inter-group links.

In this application, individual stocks were treated as nodes in a fully connected graph, with edge weights derived from pairwise return correlations. The Louvain method was applied to detect latent clusters of co-moving stocks that may not be well captured by traditional distance-based clustering techniques. Theoretically, this approach offers enhanced flexibility and can better reflect non-linear market structures, as it does not require a pre-specified number of clusters and can uncover overlapping or nested communities.

Despite its conceptual appeal, the Louvain-based strategy ultimately underperformed. The detected communities were often highly imbalanced, with dominant clusters absorbing most stocks and leaving other clusters too sparse for diversified portfolio construction. Furthermore, the instability of community assignments across rolling windows led to erratic stock selections and poor out-of-sample consistency.

While the method was fully implemented using the `python-louvain` package, the empirical results did not justify its continued integration into our framework. Nonetheless, its potential remains promising under different formulations—e.g., thresholding weak correlations, applying modularity refinement heuristics, or integrating sectoral priors. These variants could be explored in future iterations to overcome the limitations we encountered.

8.2 Unexplored Yet Promising Extensions

Although our framework combines clustering, shrinkage estimation, and downside risk management into a robust portfolio strategy, several advanced directions remain unimplemented due to time and scope constraints.

A key area for future work is the incorporation of market sentiment and investor beliefs through models like Black-Litterman. This Bayesian approach enables the integration of subjective views—derived from analyst reports, social media sentiment, or macroeconomic indicators—into return expectations. Incorporating structured sentiment indices such as the RavenPack Sentiment Index, the SP 500 X Sentiment Index, or even large language model (LLM)-generated sentiment from financial news could inject forward-looking information into the optimization process and reduce exposure to estimation noise.

Another avenue lies in the application of reinforcement learning (RL) for adaptive portfolio rebalancing. Recent work in deep RL has demonstrated the potential to learn dynamic allocation policies that react to evolving market regimes. However, such models demand substantial training data, careful reward engineering, and hyperparameter tuning, which posed feasibility challenges for this project. Nonetheless, frameworks such as Proximal Policy Optimization (PPO) or Deep Q-Networks (DQN), implemented via TensorFlow or PyTorch, could offer a powerful alternative to static convex optimization, especially in non-stationary environments.

In conclusion, these directions—sentiment-aware optimization and RL-driven reallocation—could significantly augment the responsiveness and predictive strength of our framework. While technically demanding, they represent high-impact opportunities for future development.

References

[1] Ban, Gah-Yi, El Karoui, N., Lim, A. E. B. (2018). Machine Learning and Portfolio Optimization. *Management Science*, 64(3), 1136–1154.

[2] Cao, L., Shah, B., Xie, J. (2023). Network and Clustering-Based Portfolio Optimization. FE800 Project Report, Stevens Institute of Technology.

[3] DeMiguel, V., Plyakha, Y., Uppal, R., Vilkov, G. (2012). Improving Portfolio Selection Using Option-Implied Volatility and Skewness. SSRN Working Paper.

[4] Lim, A. E. B., Shanthikumar, J. G., Vahn, G.-Y. (2011). Conditional Value-at-Risk in Portfolio Optimization: Coherent but Fragile. *Operations Research Letters*, 39(2), 163–171.

[5] Quantpedia. (2021). Introduction to Clustering Methods in Portfolio Management – Part 3. *Quantpedia White Paper Series*. Retrieved from https://quantpedia.com

[6] Zhang, C., Tu, C., Wang, Y. (2024). Enhanced Portfolio Optimization: Shrinkage, Regularization, and Risk-Free Assets. ORIE 5370 Report, Cornell University.

[7] Renegar et al. (2024). Portfolio Optimization with Market Regime Classification Using Gaussian Mixtures. ORIE 5370 Report, Cornell University.

[8] Sood, S., Papasotiriou, K., Vaiciulis, M., Balch, T. (2023). Deep Reinforcement Learning for Optimal Portfolio Allocation: A Comparative Study with Mean-Variance Optimization. AAAI Conference on Artificial Intelligence.

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

[10] Rockafellar, R. T., Uryasev, S. (2000). Optimization of Conditional Value-at-Risk. *Journal of Risk*, 2(3), 21–41.