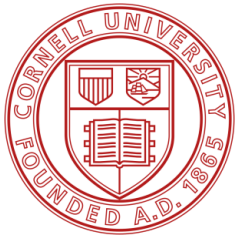


Clustering

Prof. Marcos López de Prado
Advances in Financial Machine Learning
ORIE 5256

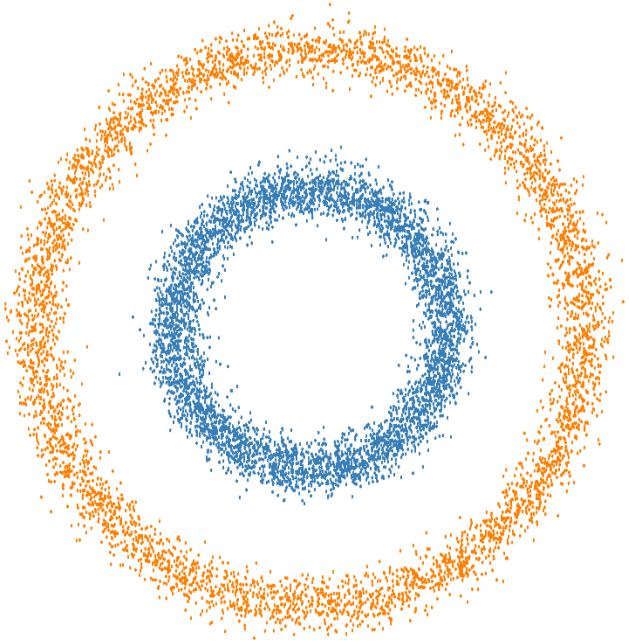
Key Points

- Many problems in finance require the clustering of variables or observations:
  - Factor investing, relative value analysis (e.g., forming quality minus junk portfolios)
  - Risk management, portfolio construction (e.g., deriving the efficient frontier)
  - Dimensionality reduction (e.g., decomposing bond return drivers)
  - Modelling of multicollinear systems (e.g., computing *p*-values)
- Despite its usefulness, clustering is almost never taught in Econometrics courses
  - None of the major Econometrics textbooks, and only a handful of academic journal articles, discuss the clustering of financial datasets
- In this seminar we review two general clustering approaches:
  - Partitional
  - Hierarchical
- Different features and/or similarity metrics will lead to different clusterings
  - **It is key to formulate the problem in a way that results have economic meaning and interpretability**

Introduction

What is Clustering?

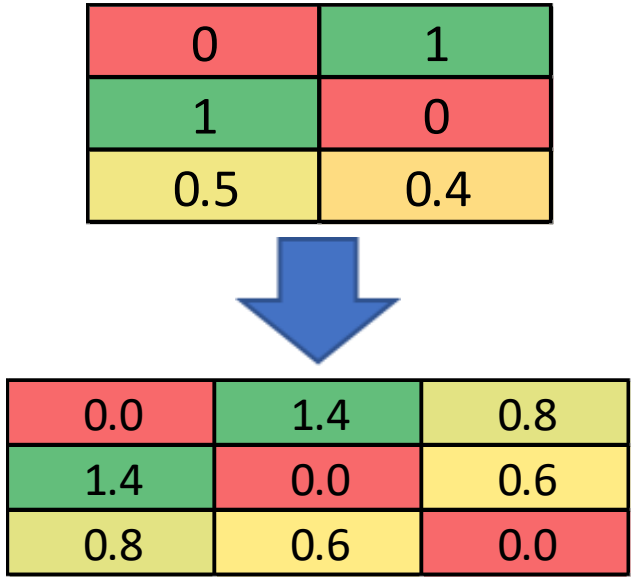
- A clustering problem is defined by a set of *objects* and a set of *features* associated with those objects
- Goal: Separate the objects into groups (called clusters) using the features, such that intra-group similarities are maximized, and inter-group similarities are minimized
- Clustering is a form of unsupervised learning
  - we do not provide examples to assist the algorithm in solving this task.
- Clustering problems appear naturally in finance, at every step of the investment process



In this example, an agglomerative clustering algorithms recognizes two types of points based on 2 features (their coordinates)

Proximity Matrix

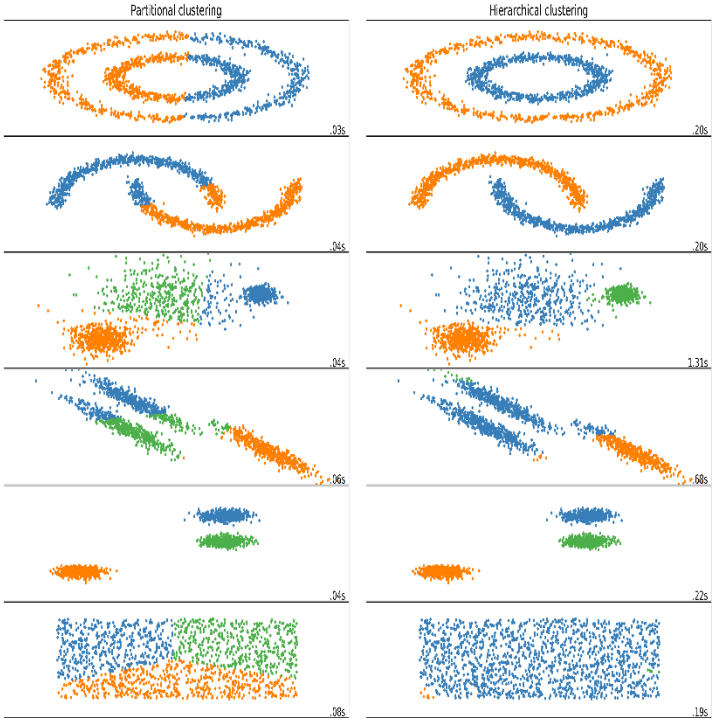
- Consider a data matrix X, of order NxF, where N is the number of objects and F is the number of features
- We use the F features to compute the proximity between the N objects, as represented by an NxN matrix
  - The proximity measure can indicate either *similarity* (e.g., correlation, mutual information) or *dissimilarity* (e.g., a distance metric).
  - It is convenient but not strictly necessary that dissimilarity measures satisfy the conditions of a metric
- When forming the proximity matrix, it is a good idea to standardize the input data, to prevent that one feature's scale dominates over the rest.



A data matrix with 3 objects described by 2 features implies a 3x3 proximity matrix. The 3rd object is slightly more similar to the 2nd than to the 1st

Main Types of Clustering Methods

- There are two main classes of clustering algorithms: partitional and hierarchical
  - **Partitional** techniques create a one-level (un-nested) partitioning of the objects (each object belongs to one cluster, and to one cluster only)
  - **Hierarchical** techniques produce a nested sequence of partitions, with a single, all-inclusive cluster at the top and singleton clusters of individual points at the bottom.
    - Hierarchical clustering algorithms can be divisive (top-down) or agglomerative (bottom-up).
- By restricting the growth of a hierarchical tree, we can derive a partitional clustering from any hierarchical clustering
  - However, one cannot generally derive a hierarchical clustering from a partitional one



Comparison of partitional vs hierarchical clustering on different datasets

A Partitional Algorithm: K-Means

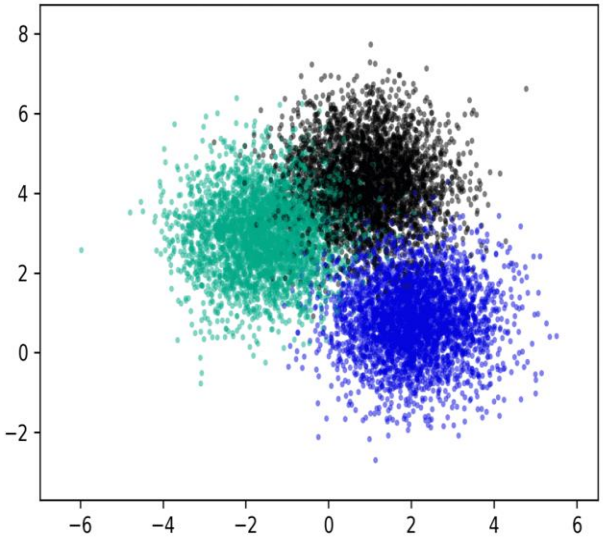
The Algorithm

- K-Means is a vector quantization model
  - It attempts to split the samples (rows) of X into a pre-determined number of clusters K

1. Initialize a random set of K centroids, $\{\mu_k\}_{k=1,\dots,K}$
2. Assign each sample X_n to one cluster, such that the within-clusters variance is minimized

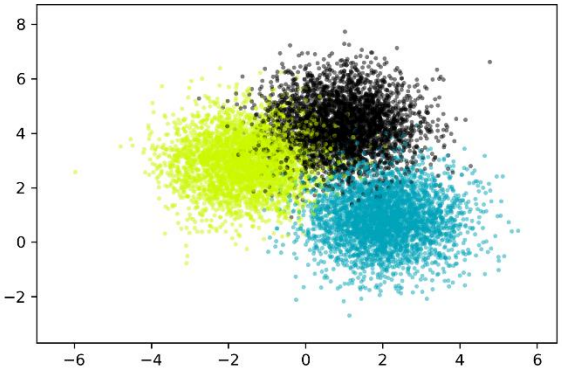
$$\sum_{n=1}^{N} \min_{k\in[1,K]} [\|X_n - \mu_k\|^2]$$

3. Update $\{\mu_k\}$, based on the clusters from (2)
4. Repeat steps (2)-(3) until convergence

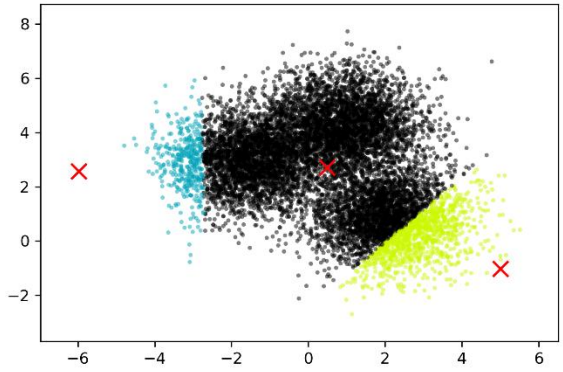


Convergence of K-Means on 3 blobs

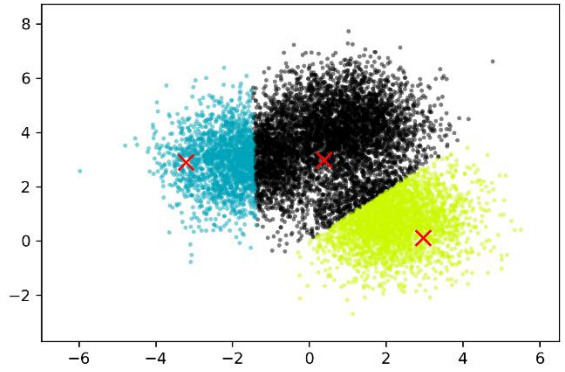
The Assignment-Update Cycle



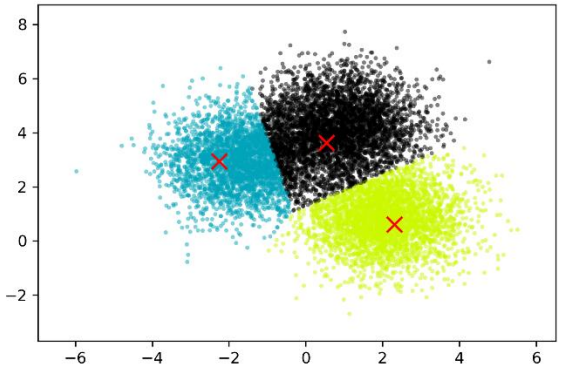
Original data



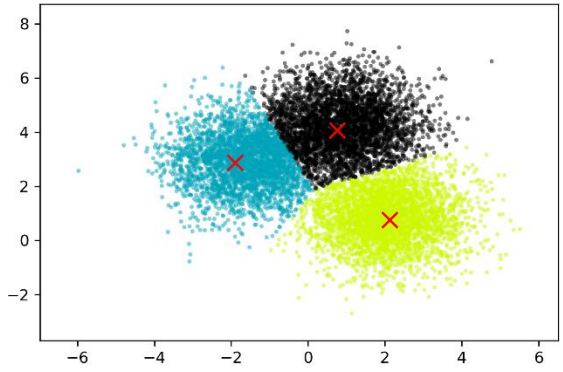
Iteration 1



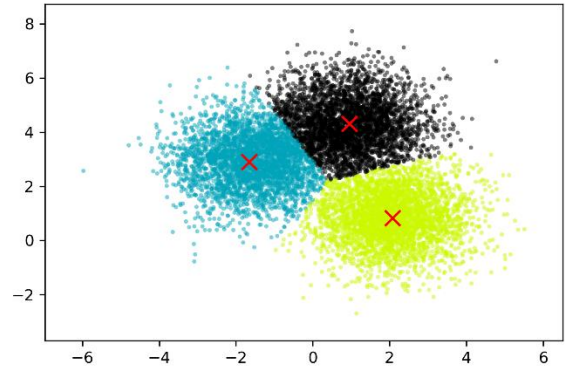
Iteration 2



Iteration 3



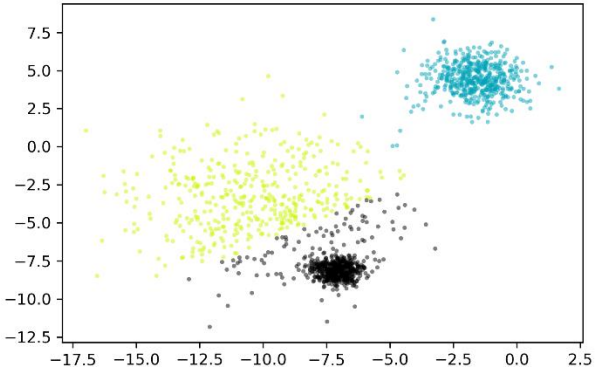
Iteration 4



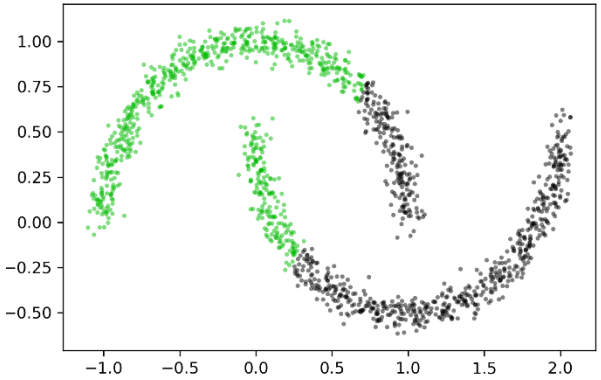
Iteration 5

A Few Considerations

- K-Means assumes that the clusters are convex, isotropic, and with similar variance
  - Features should be standardized prior to clustering
  - Other algorithms may perform better when clusters are elongated or irregular
- Within-cluster variance is not a normalized metric
  - Curse of dimensionality: When X has many columns, variances are inflated, and outcomes may be biased. One solution is to apply a dimensionality reductions technique (e.g., PCA) prior to clustering
- K-Means will always converge, however the outcome may be a local minimum
  - One solution is to run multiple instances in parallel, with different seed centroids



Three blobs with difference variance



Two anisotropic blobs

Python Implementation

```
import matplotlib.pyplot as mpl,matplotlib.cm as cm
from sklearn import datasets
from sklearn.cluster import KMeans
X,_=datasets.make_blobs(n_samples=10000,n_features=2,centers=3,random_state=0)
clusterer=KMeans(n_clusters=3).fit(X)
colors=cm.nipy_spectral(clusterer.labels_.astype(float)/n_clusters)
mpl.scatter(X[:,0],X[:,1],marker='.',s=30,lw=0,alpha=.5,c=colors,edgecolor='k’)
mpl.savefig('kmeans.png',dpi=300)
```

The above code generates a features matrix X, with values centered around 3 blobs. Then applies K-Means on the X matrix, targeting 3 clusters, and plots the predicted clusters.

A Hierarchical Algorithm: Agglomerative Clustering

Agglomerative Algorithm

1. Apply a distance metric to X
2. Combine into a cluster the pair with lowest distance
   - The pair can be composed of two items, two clusters, or one item and a cluster
3. Reduce the distance matrix
   a. Remove the 2 rows and columns associated with the pair
   b. Apply a **linkage criterion** to determine the distance between the new cluster and the rest of objects, e.g.:
      - Single linkage: minimum distance to any object in the pair
      - Complete linkage: maximum distance to any object in the pair
4. Repeat 2 and 3 until the distance matrix has been reduced to only one object

0.0	1.4	0.8
1.4	0.0	0.6
0.8	0.6	0.0

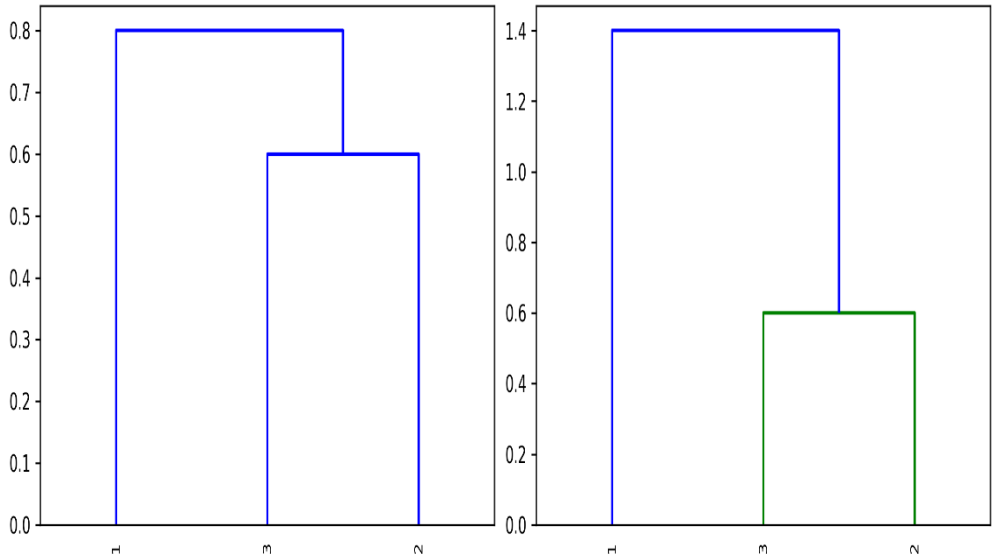
0.0	0.8
0.8	0.0

0.0

Reduction of the distance matrix: First we combine objects 2 and 3. Applying a single-linkage criterion, we determine that the distance between the cluster and object 1 is 0.8

Dendrogram

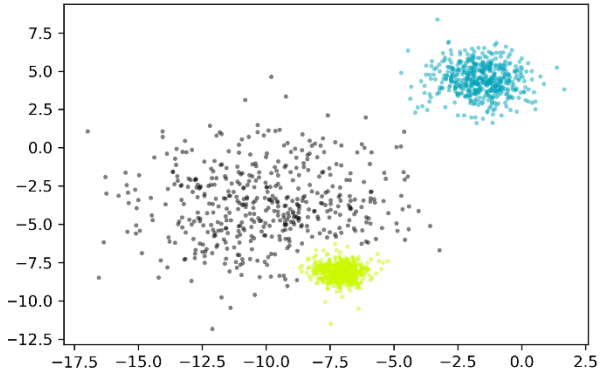
- A dendrogram is a tree graph that displays the hierarchical composition of the clusters
- The y-axis indicates the distance between the two objects that form a new cluster
- A linkage matrix characterizes a dendrogram
  - For N items, a linkage matrix has $N - 1$ rows (one row per cluster)
  - Three columns:
    - Integer identifying object 1
    - Integer identifying object 2
    - Distance between objects 1 and 2 (based on linkage criterion)



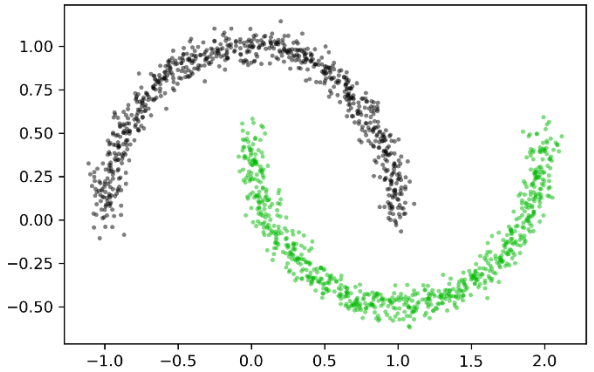
Dendrograms associated with the previous example, using the single-linkage (left) and complete-linkage (right) criteria. Note how the second clustering occurs at the distance of 0.8 for single-linkage, but 1.4 for complete-linkage

A Few Considerations

- Hierarchical algorithms can handle clusters that are non-convex, anisotropic, with unequal variance
  - This includes clusters within clusters
- Hierarchical algorithms allow connectivity constraints
  - Connectivity constraints cluster together only adjacent points. This links together points even if the centroid is not part of the cluster
- However, hierarchical algorithms may not handle properly elongated blobs
  - One solution is to orthogonalize the features (e.g., PCA without dimensionality reduction) prior to clustering
- The appropriate linkage method can be chosen via cross-validation, or [cophenetic correlation]



Three blobs with unequal variance



Two anisotropic blobs

Python Implementation

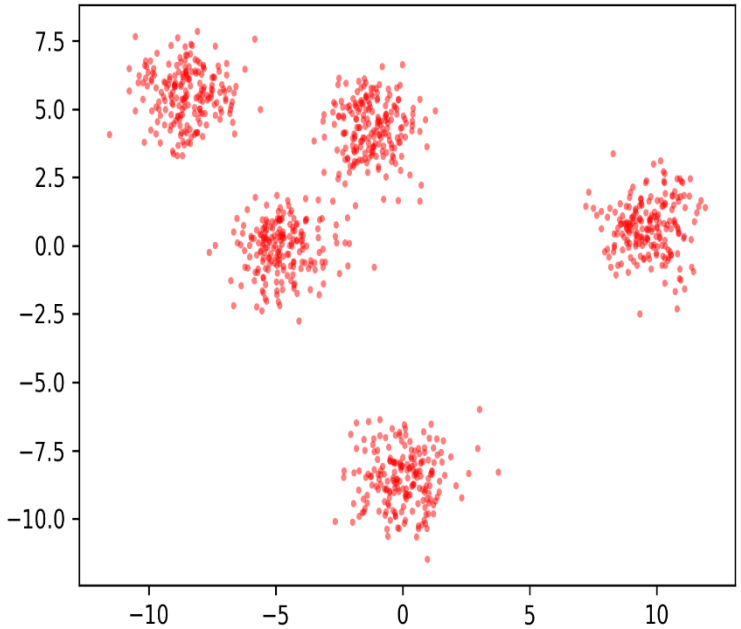
```
import matplotlib.pyplot as mpl,seaborn as sns
import scipy.cluster.hierarchy as sch
link=sch.linkage(ssd.squareform(dist0,force='tovector'),optimal_ordering=True,method='single')
dist1=dist0.iloc[sch.leaves_list(link),sch.leaves_list(link)] # apply clusters
sns.heatmap(dist1,cmap='viridis',xticklabels=False,yticklabels=False)
mpl.savefig('clusters.png',dpi=300)
mpl.clf();mpl.close() # reset pylab
dendro=sch.dendrogram(link,leaf_rotation=90.,leaf_font_size=8.,
                                labels=dist.columns,distance_sort=True)
mpl.savefig('dendro.png',dpi=300)
mpl.clf();mpl.close() # reset pylab
```

The above code takes a distance matrix `dist0` (a dataframe), derives the linkage matrix using a single-linkage criterion, applies the clusters to produce a reordered `dist1` distance matrix, plots `dist1`, and plots the associated dendrogram. See also Scikit-Learn's implementation.

Optimal Number of Clusters

Cluster Scoring

- In order to determine the optimal number of clusters, we first need to define a function that scores the output of a scoring algorithm
- In general, there are two types of clustering scoring functions:
  a) External: those that require ground-truth labels
  b) Internal: those that don't require it
- Because clustering is an unsupervised learning problem, internal scores are more natural. Three of the most used internal scoring functions are:
  - Calinski-Harabasz index (or variance ratio)
  - Davies-Boulding index
  - Silhouette scores



How many blobs are there? On 2-D, this is an easy question for a human. On higher dimensions, machines are more likely to win

Calinski-Harabasz Index (Variance Ratio)

Given N objects, centered around c, and grouped into K clusters

- The within-cluster dispersion (W_K) is

$$W_K = \sum_{k=1}^{K} \sum_{x \in C_k} (x - c_k)^T (x - c_k)$$

  where C_k is the set of objects in cluster k, c_k is the center of cluster k, and x is coordinates of an object.

- The between-cluster dispersion (B_K) is

$$B_K = \sum_{k=1}^{K} N_k (c_k - c)^T (c_k - c)$$

  where N_k is the number of objects in C_k.

- The Variance Ratio is defined as

$$s = \frac{B_K}{W_K} x \frac{N - K}{K - 1}$$

Davies-Bouldin Index

- This index is defined on K cluster as

$$s = \frac{1}{K}\sum_{i=1}^{K}\max_{i\neq j} R_{i,j}$$

  where $R_{i,j}$ is the (symmetric, non-negative) similarity between two clusters.

- A common choice is $R_{i,j} = \frac{\delta_i+\delta_j}{d_{i,j}}$, where
  - δ_i is the average distance between objects in cluster i and that cluster's centroid
  - $d_{i,j}$ is the distance between the centroids of clusters i and j
- Accordingly, this index can be interpreted as the average similarity between each cluster C_i and its most similar cluster C_j
  - Note: A lower index means better clustering. The more dissimilar the clusters, the better the clustering

Silhouette Scores

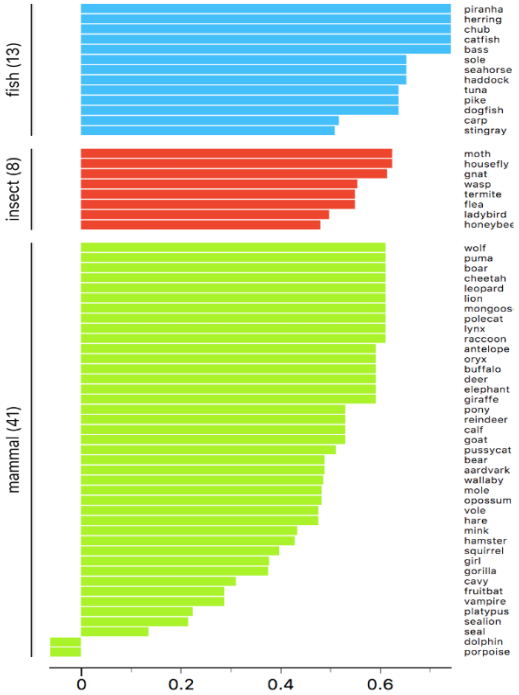
- Silhouette scores are defined for each sample, $\{s_n\}_{n=1,\ldots,N}$

$$s_n = \frac{b_n - a_n}{\max\{a_n, b_n\}}$$

where

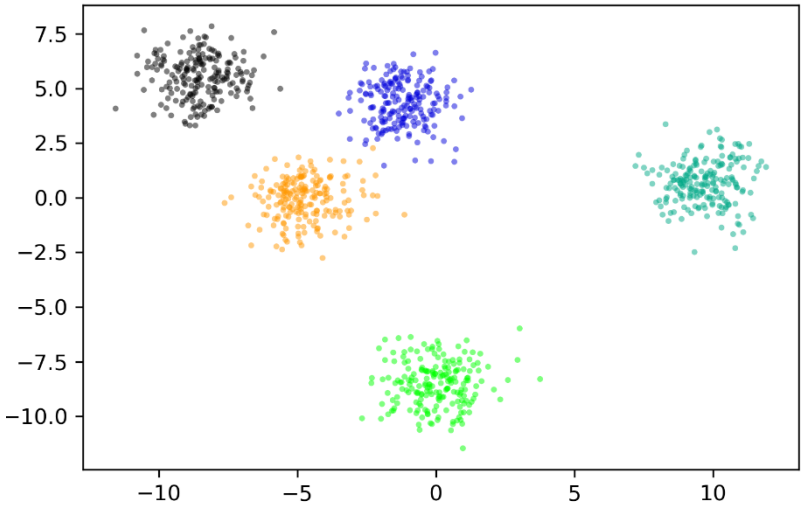
  - a_n: mean distance between object n and other objects in its cluster
  - b_n: mean distance between object n and objects in the nearest cluster

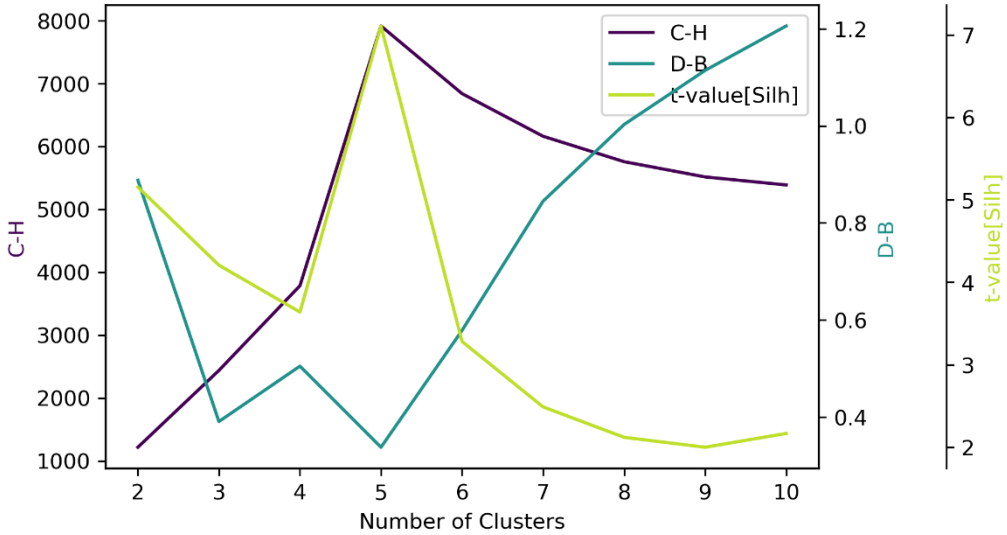
- Advantages:
  - The scores are bounded: $-1 \leq s_n \leq 1$
  - Because we have one score per sample, we can reallocate specific objects to better clusters
  - Clusters with average $s_n \approx 0$ are overlapping, and could be merged
  - We can use $\{s_n\}$ to derive a distribution of scores, and make inference (p-values). For example, we can compute the t-value, $s = \frac{\mathrm{E}[\{s_n\}]}{\sqrt{\mathrm{V}[\{s_n\}]}}$



Silhouette scores for dolphins highlight how unique they are among mammals

Optimal Number of Clusters



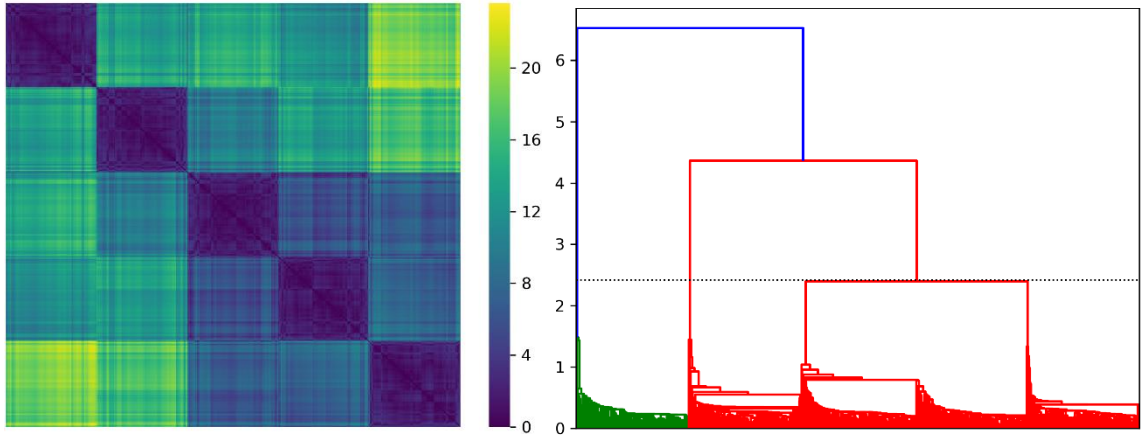


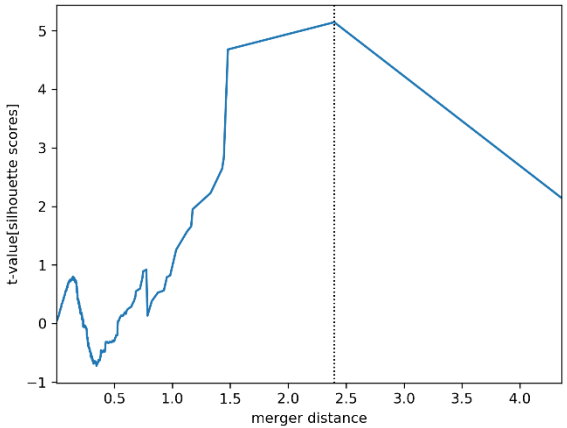
How many blobs are there? First, we applied the K-Means algorithm, setting as target 2,3,…,10 clusters. Second, we computed the three scores on each of the resulting groups for 2,…,10 clusters. Third, we plotted the scores for each number of clusters.

All scoring functions agree that the optimal number of clusters is 5. D-B points at the possibility of 3 clusters, which is not an unreasonable answer considering that 2 blobs are in close proximity.

Deriving Partitions from a Hierarchical Algorithm

- Dendrograms give us the distance at which each merger has taken place
- We can evaluate the scoring function at the distance of each merger, and derive the optimal merging distance
- That distance implies the number of clusters
- The procedure is fast, however not necessarily the most accurate





If we apply a hierarchical clustering method on the previous 5 blobs example, we recover 4 clusters (with the right answer, 5 clusters, being a close alternative)

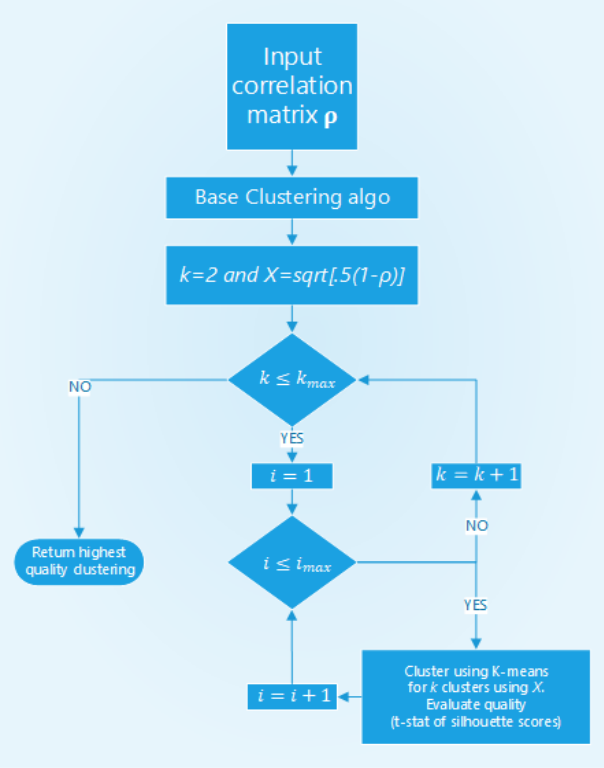
The ONC Algorithm (1/4)

- Clustering steps (find here an implementation):
  1. Base level clustering
  2. Recursive improvements to clustering

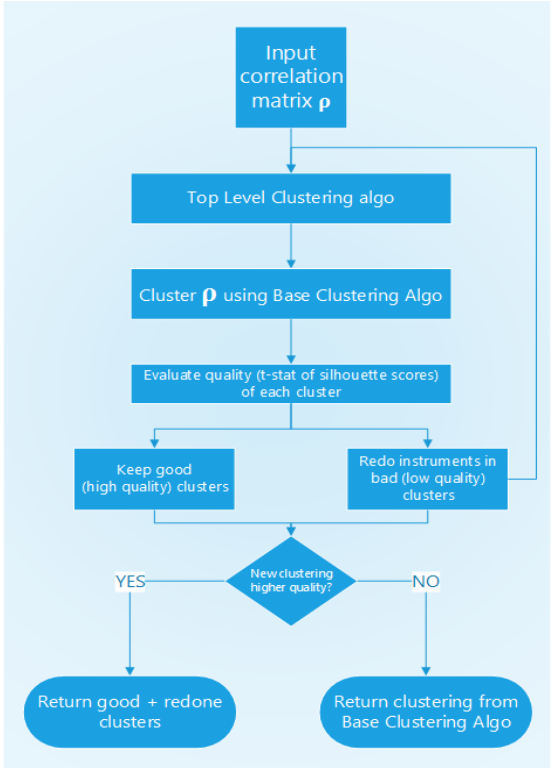
- Base Clustering:
  1. For each k in $2, \ldots, N-1$, and l in $1, \ldots, n_{init}$:
     - Apply K-Means to extract k clusters using distance matrix
     - Evaluate silhouette scores s_n, $n = 1, \ldots, N$, for the clustering
     - Evaluate quality score $q_{k,l} = \frac{\mathrm{E}[s_n]}{\sqrt{\mathrm{V}[s_n]}}$
  2. Choose optimal clustering, with $K = \mathrm{argmax}_k \left\{ \max_l \{q_{k,l}\} \right\}$

- Recursive improvement to clustering:
  1. Run Base Clustering to derive K
  2. Keep clusters where $q_i \geq \mathrm{E}[q_i]$
     - Recursively rerun Base Clustering for rest of clusters
     - Keep new clustering only if re-clustering improves average quality score

The ONC Algorithm (2/4)



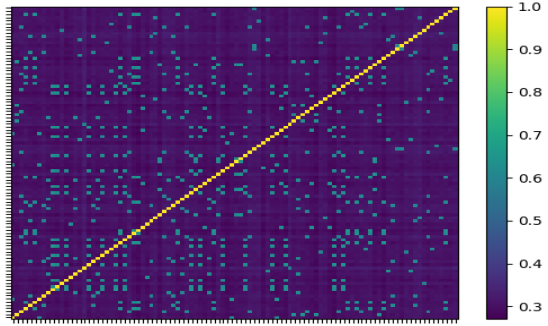
Base clustering



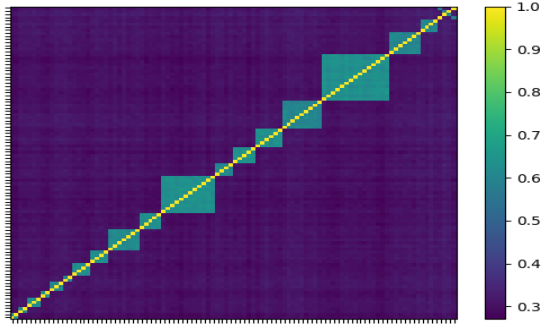
Higher-level clustering

The ONC Algorithm (3/4)

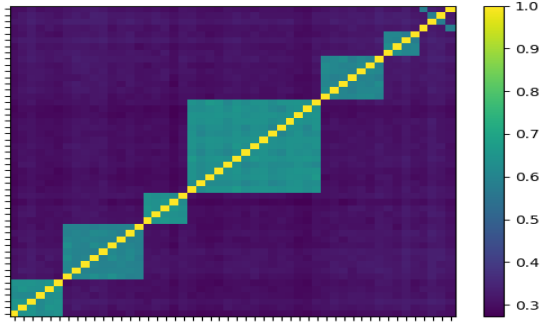
Initial Scrambled Random Block Covariance



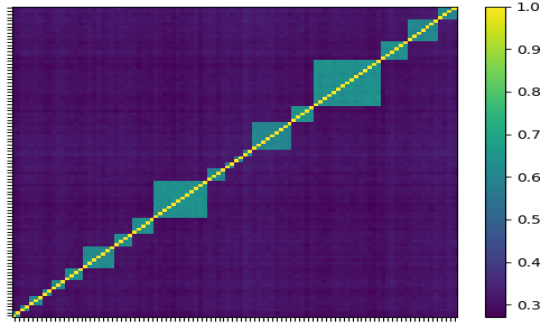
After Base Clustering



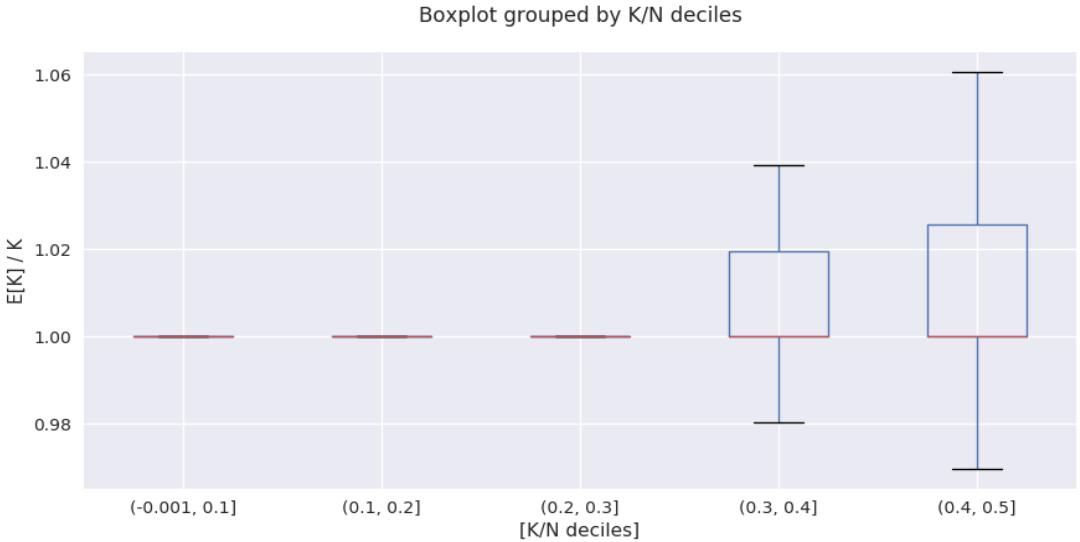
Recursively Re-evaluate Clustering



Final Clustering



The ONC Algorithm (4/4)



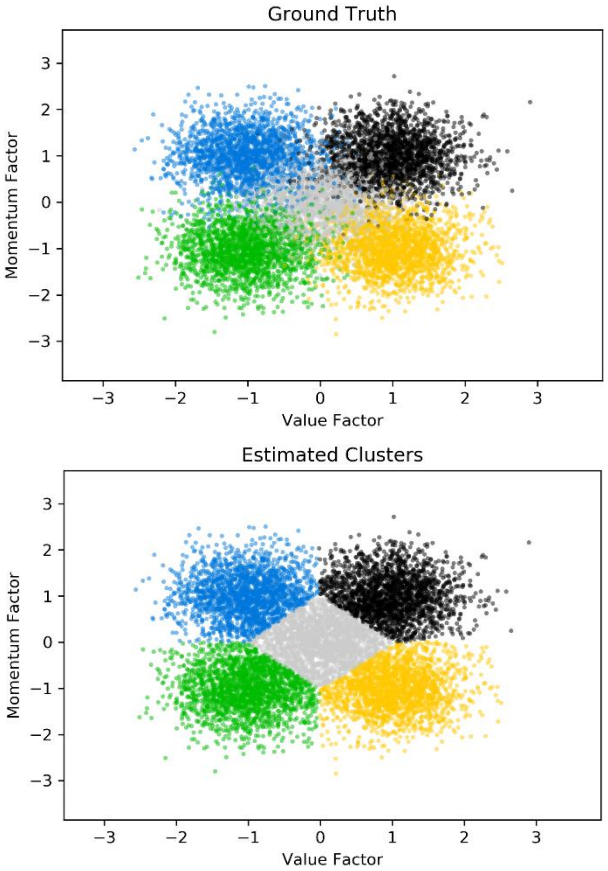
- Create a random block covariance matrix
  - Size NxN
  - K blocks of random size
- Add global noise
- Run Clustering
- Compare expected cluster count K to number of estimated clusters

The boxplots show the results from these simulations. In particular, for $\frac{K}{N}$ in a given decile, we display the boxplot of the ratio of K predicted by the clustering to the actual $\mathrm{E}[K]$ predicted by clustering. Ideally, this ratio should be near 1. Monte Carlo simulations confirm that ONC is effective at recovering the ground truth.

A Few Use Cases

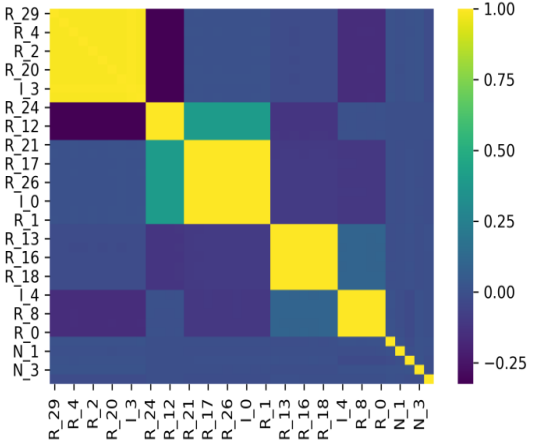
Factor Investing / Relative Value

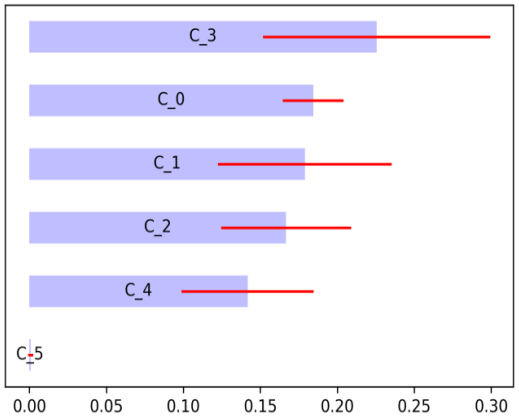
- Factor investing attempts to price assets that share some common characteristics
- Traditionally, economists group assets according to a single characteristic
  - E.g.: value, size, momentum, quality, liquidity, carry, etc.
- This misses known interaction effects, such as value vs. momentum, and hierarchical dependencies
- A natural solution is to cluster assets on multiple characteristics (features), and let the algorithm find the optimal number of clusters
  - We can then evaluate the performance of each cluster, and assess whether the risk-premium is statistically significant
  - This approach is also useful for relative value strategies



Feature Importance Analysis

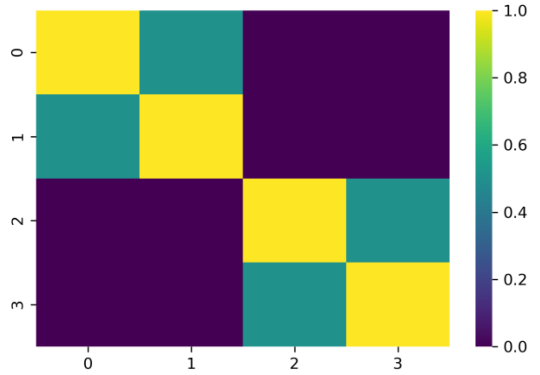
- Consider a binary random classification problem composed of 40 features, where 5 are informative, 30 are redundant, and 5 are noise
  - **Informative features** (marked with the “I_” prefix) are those used to generate labels
  - **Redundant features** (marked with the “R_” prefix) are those that are formed by adding Gaussian noise to a randomly chosen informative feature
  - **Noise features** (marked with the “N_” prefix) are those that are not used to generate labels
- A clustering algorithm prevents that *substitution effects* bias the MDA or MDI analyses, by
  - finding the optimal number of clusters (see top right plot)
  - bundling together the features that are redundant to an informative one (see bottom right plot)





Portfolio Construction

- When K securities form a correlation cluster, convex optimization methods (Markowitz, Black-Litterman, etc.) cannot distinguish between them
- This is an example of signal-induced instability
- One solution is to apply the NCO algorithm:
  1. cluster the correlation matrix,
  2. compute the optimal intra-cluster allocations,
  3. compute the optimal inter-cluster allocations,
  4. derive the optimal weights as the dot-product of (2) and (3)
- Steps (1) and (2) allow us to transform a "Markowitz-cursed" problem into a well-behaved problem

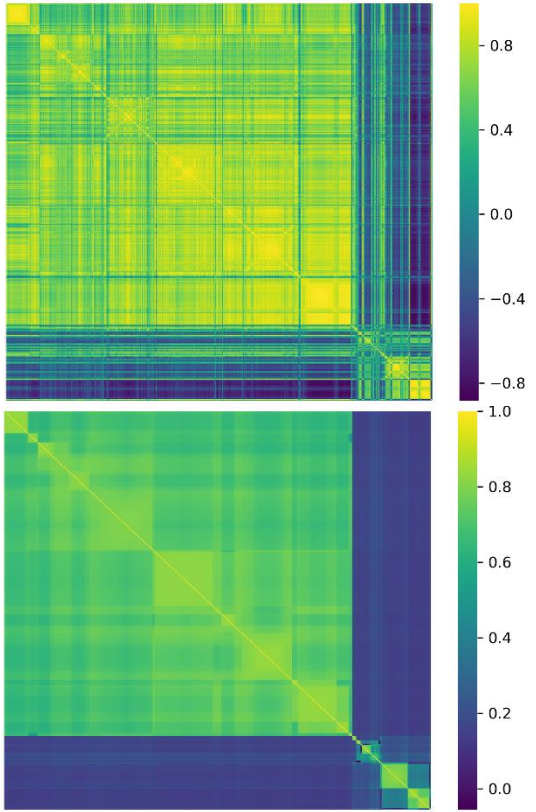


	Markowitz	NCO
Raw	7.02E-02	3.17E-02
Shrunk	6.54E-02	5.72E-02

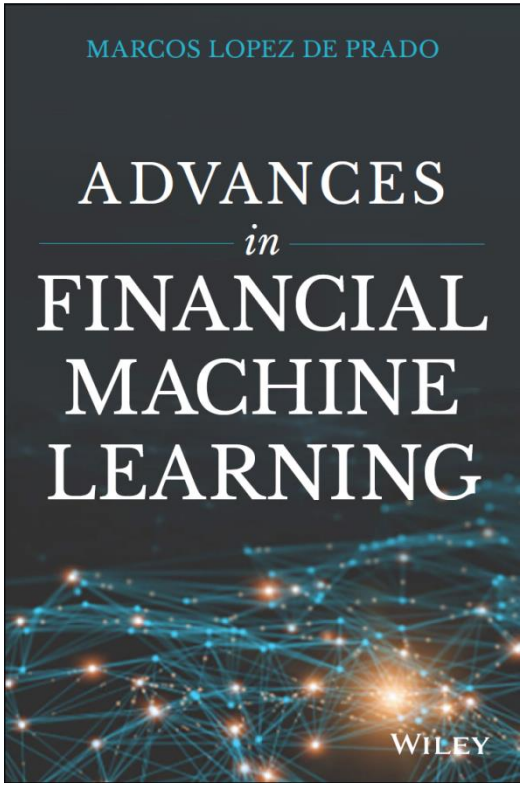
NCO computes the maximum Sharpe ratio portfolio with 45.17% of Markowitz's RMSE, i.e. a 54.83% reduction in RMSE

Forward-Looking Correlation Matrices

- It is widely acknowledged that empirical correlations have:
  a) poor numerical properties that lead to unreliable estimators; and
  b) poor predictive power
- Additionally, factor-based correlation matrices have their own caveats. In particular, estimated factors are typically:
  - non-hierarchical, and
  - do not allow for interactions at different levels
- We can derive a forward-looking correlation matrix from a knowledge graph
  - On the right, plots of a correlation matrix before (top) and after (bottom) imposing a theory-implied structure (GICS)
  - Adding signal through a theoretical dendrogram makes correlation patterns smoother and less noisy, while preserving the hierarchical structure



For Additional Details



The first wave of quantitative innovation in finance was led by Markowitz optimization. Machine Learning is the second wave and it will touch every aspect of finance. López de Prado's Advances in Financial Machine Learning is essential for readers who want to be ahead of the technology rather than being replaced by it.
— Prof. **Campbell Harvey**, Duke University. Former President of the American Finance Association.

Financial problems require very distinct machine learning solutions. Dr. López de Prado's book is the first one to characterize what makes standard machine learning tools fail when applied to the field of finance, and the first one to provide practical solutions to unique challenges faced by asset managers. Everyone who wants to understand the future of finance should read this book.
— Prof. **Frank Fabozzi**, EDHEC Business School. Editor of The Journal of Portfolio Management.

Disclaimer

- The views expressed in this document are the authors' and do not necessarily reflect those of the organizations he is affiliated with.
- No investment decision or particular course of action is recommended by this presentation.
-

www.QuantResearch.org