Guided Tour of Machine Learning in Finance

Week 3: Unsupervised Learning

Clustering algorithms

Igor Halperin

NYU Tandon School of Engineering, 2017

Why clustering?

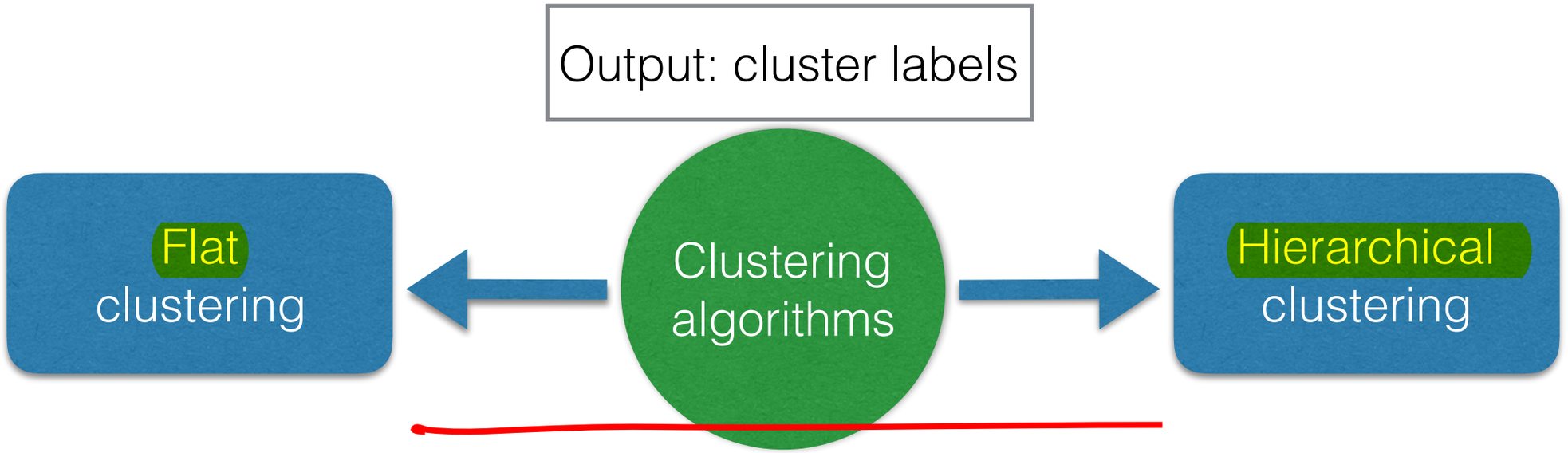
Aggregate a whole population of companies, stocks, credit card holders, mortgage holders etc. into a relatively homogeneous groups where intra-group variance is generally smaller than inter-group variance.

Purposes:

1. Visualization of data (when data is low-dimensional)
2. Conceptualization of clusters, model-building on clusters as homogeneous sets of data
3. Compact representation of data

Types of clustering

Segment a set of companies, stocks, credit card holders, mortgage holders etc. into a relatively homogeneous groups where intra-group variance is generally smaller than inter-group variance.

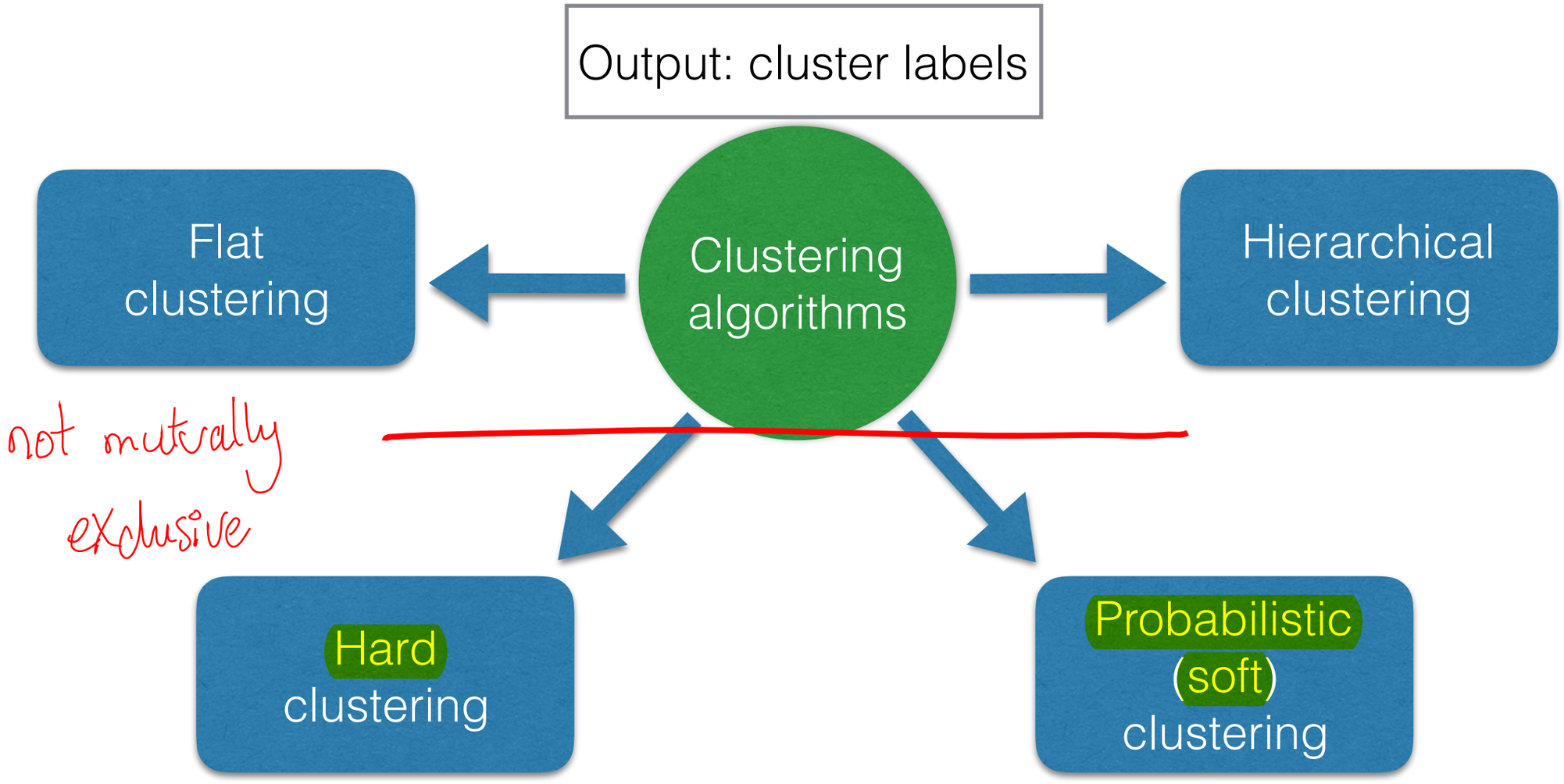


Flat : within each cluster all points are equal in terms of their info content

Some structure within each cluster

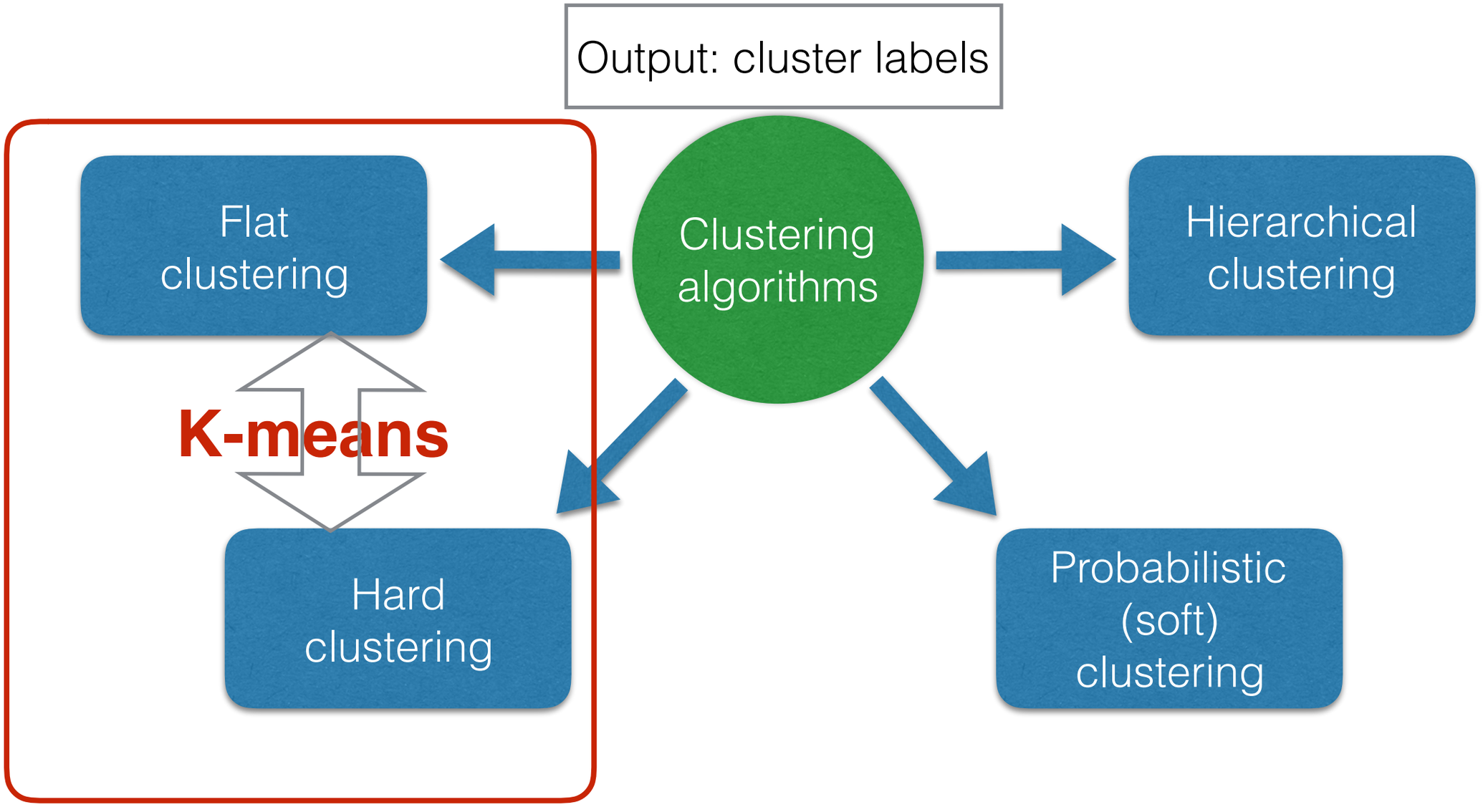
Types of clustering

Segment a set of companies, stocks, credit card holders, mortgage holders etc. into a relatively homogeneous groups where intra-group variance is generally smaller than inter-group variance.



K-means clustering

K-means is one of the most popular and scalable clustering algorithms



Control question

Select all correct answers

1. Clustering methods aggregate data points into relatively homogeneous groups of points that are more similar intra-cluster than inter-cluster.
2. With a Flat Clustering, all points within a given cluster are “equal”, but within a Hierarchical Clustering, some points within a cluster are “more equal” than the others, for example some points can form sub-clusters, etc.
3. A given clustering algorithm can be simultaneously Flat and Hard.
4. A given clustering algorithm can be simultaneously Hard and Soft.

Correct answers: 1, 2, 3

Guided Tour of Machine Learning in Finance

Week 3: Unsupervised Learning

3-2-2: K-means clustering

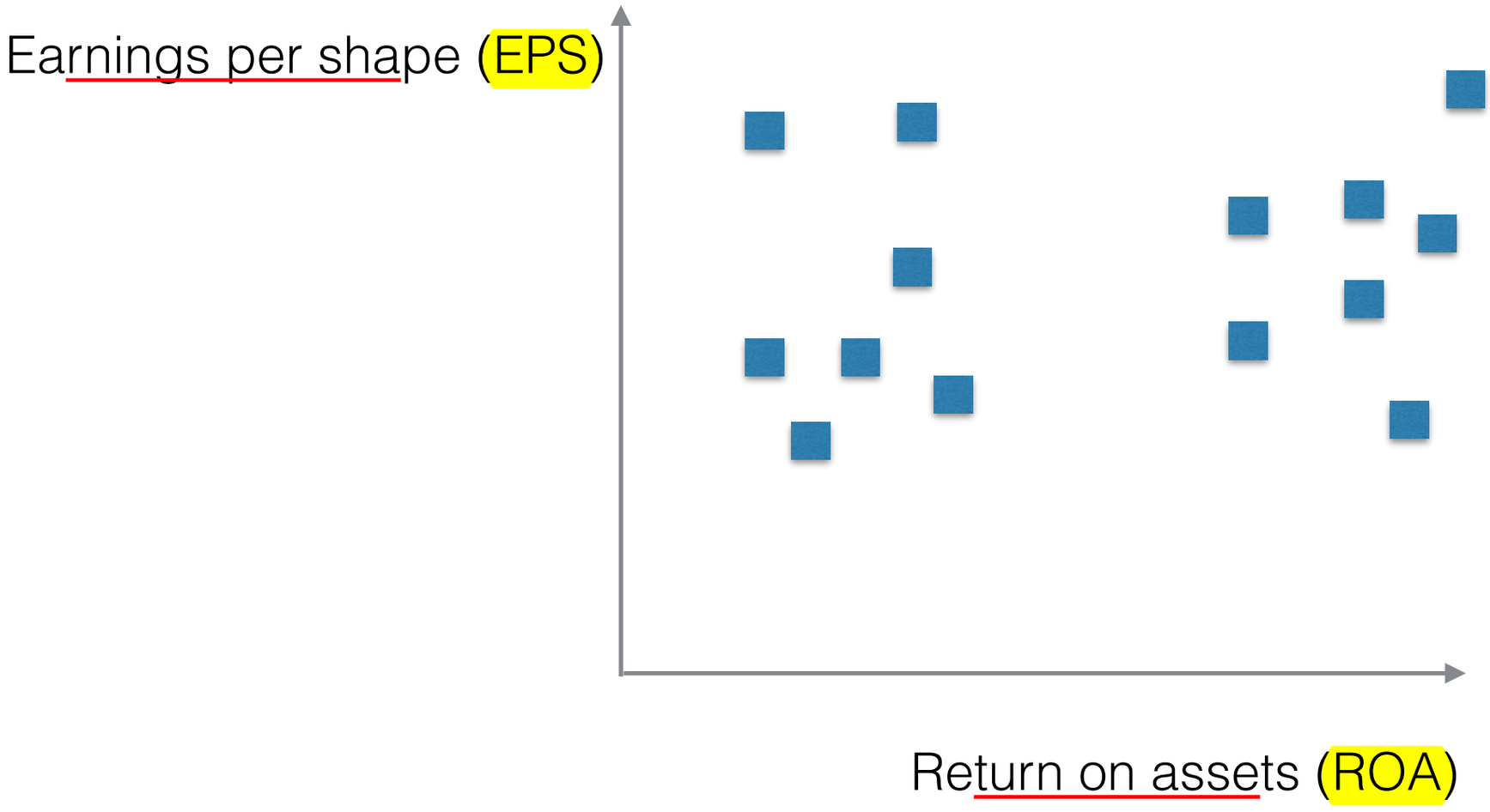
Igor Halperin

NYU Tandon School of Engineering, 2017

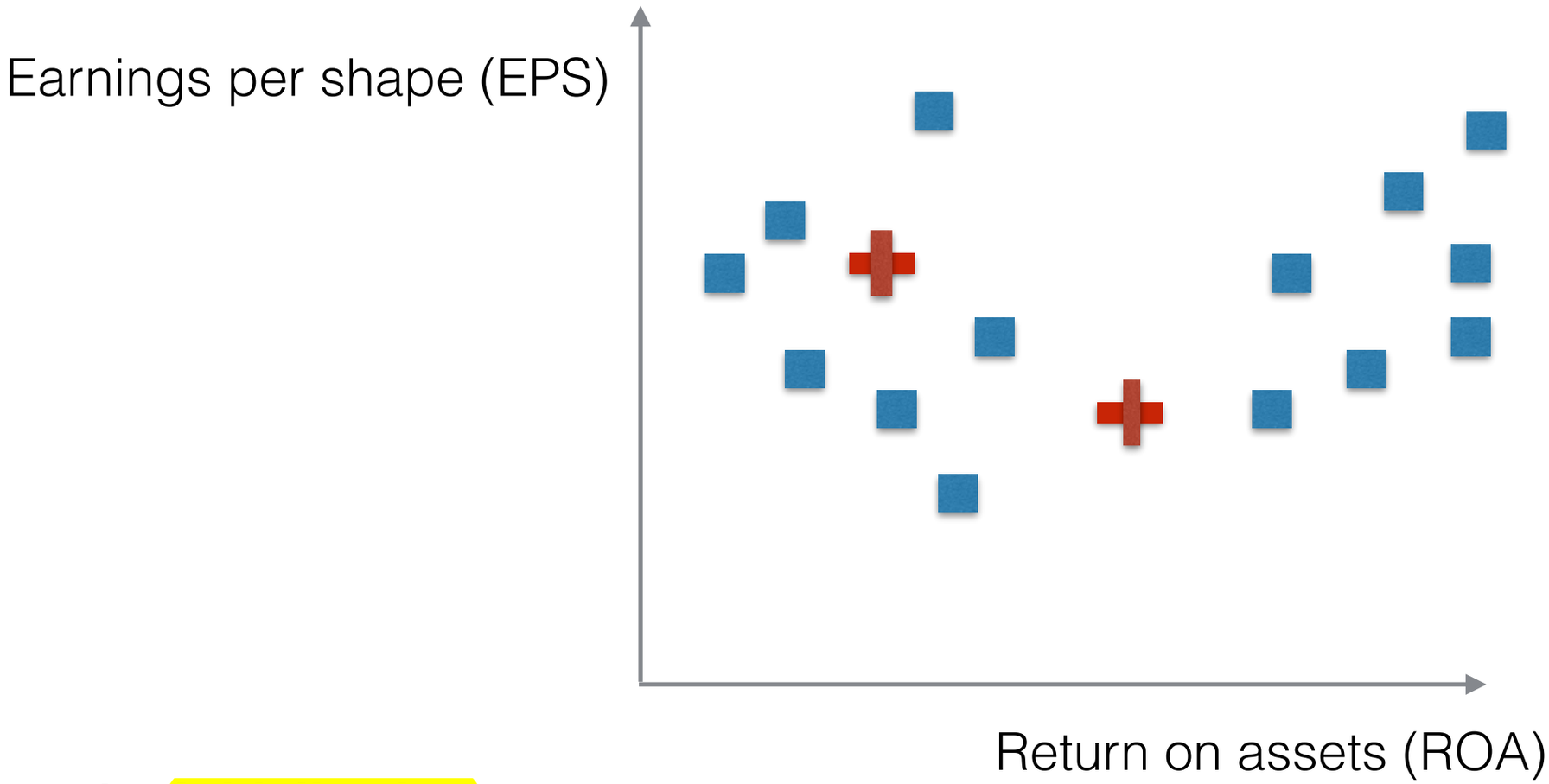
clusters identified by

K-means clustering

Cluster a two-dimensional view of a set of companies



K-means clustering

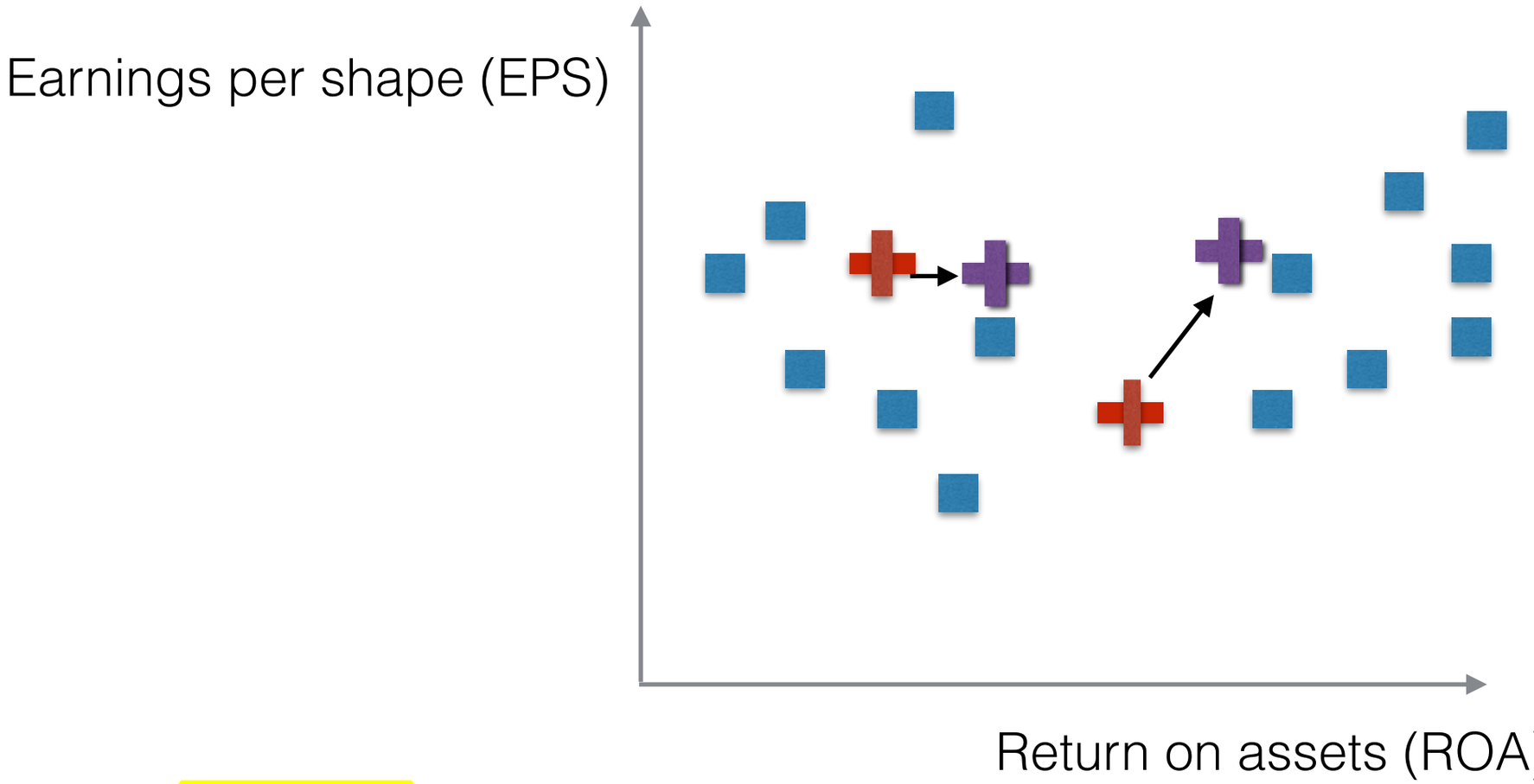


1. **Initialization:**

- choose $K: \ K = 2$ — 2 clusters
- choose K random points in the input space
- assign the cluster centers μ_k to these points

randomly

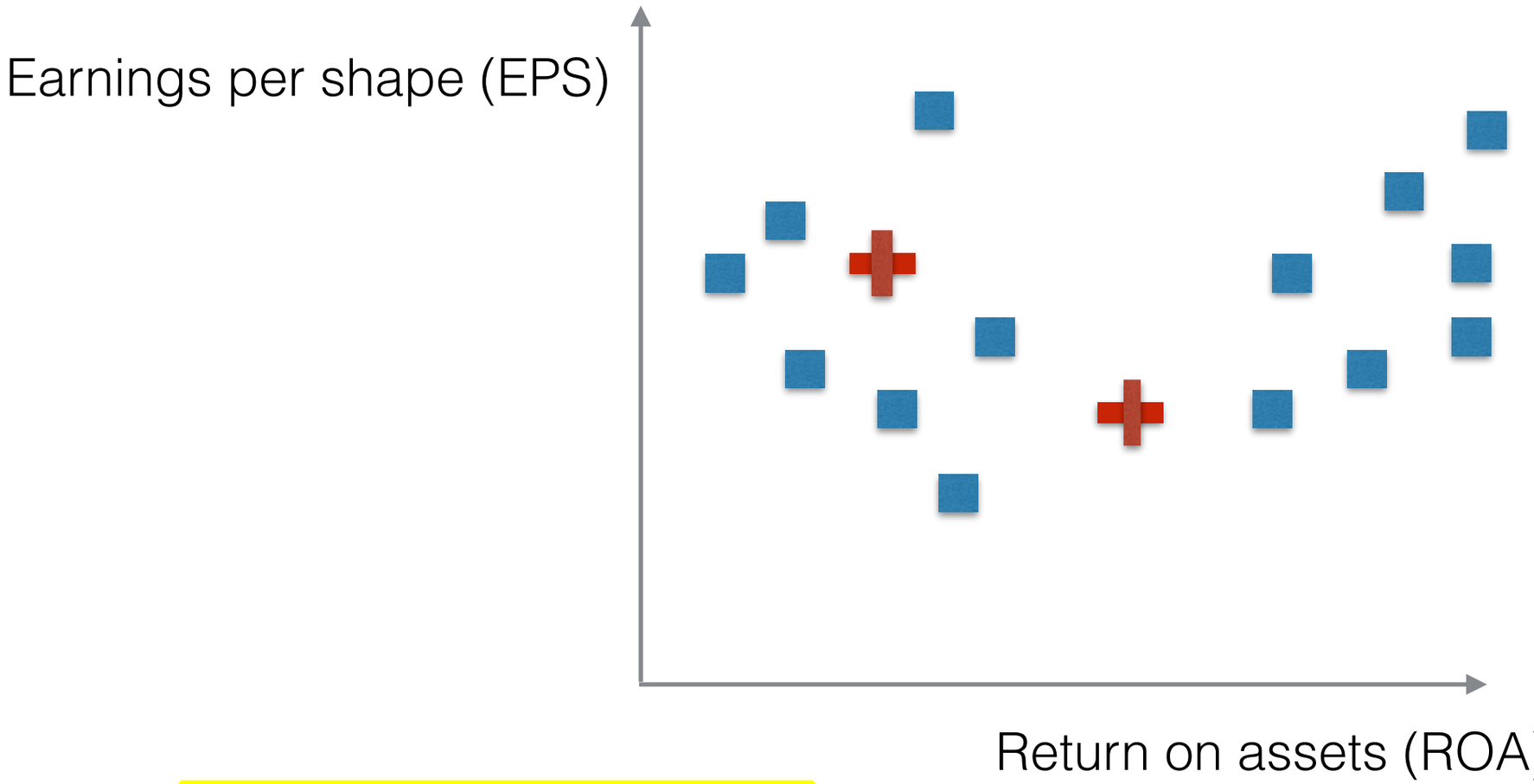
K-means clustering: learning



2. **Learning:**
   - repeat
     - for each datapoint $\mathbf{x}_i$:
       - compute the distance $d(\mathbf{x}_i, \mu_k)$ to each cluster center μ_k
       - assign the datapoint to cluster C_k with the smallest distance $d(\mathbf{x}_i, \mu_k)$
       - for each cluster C_k, re-compute its centroid

$$\mu_k = \frac{1}{N_{C_k}} \sum_{i \in C_k} \mathbf{x}_i$$

K-means clustering: usage

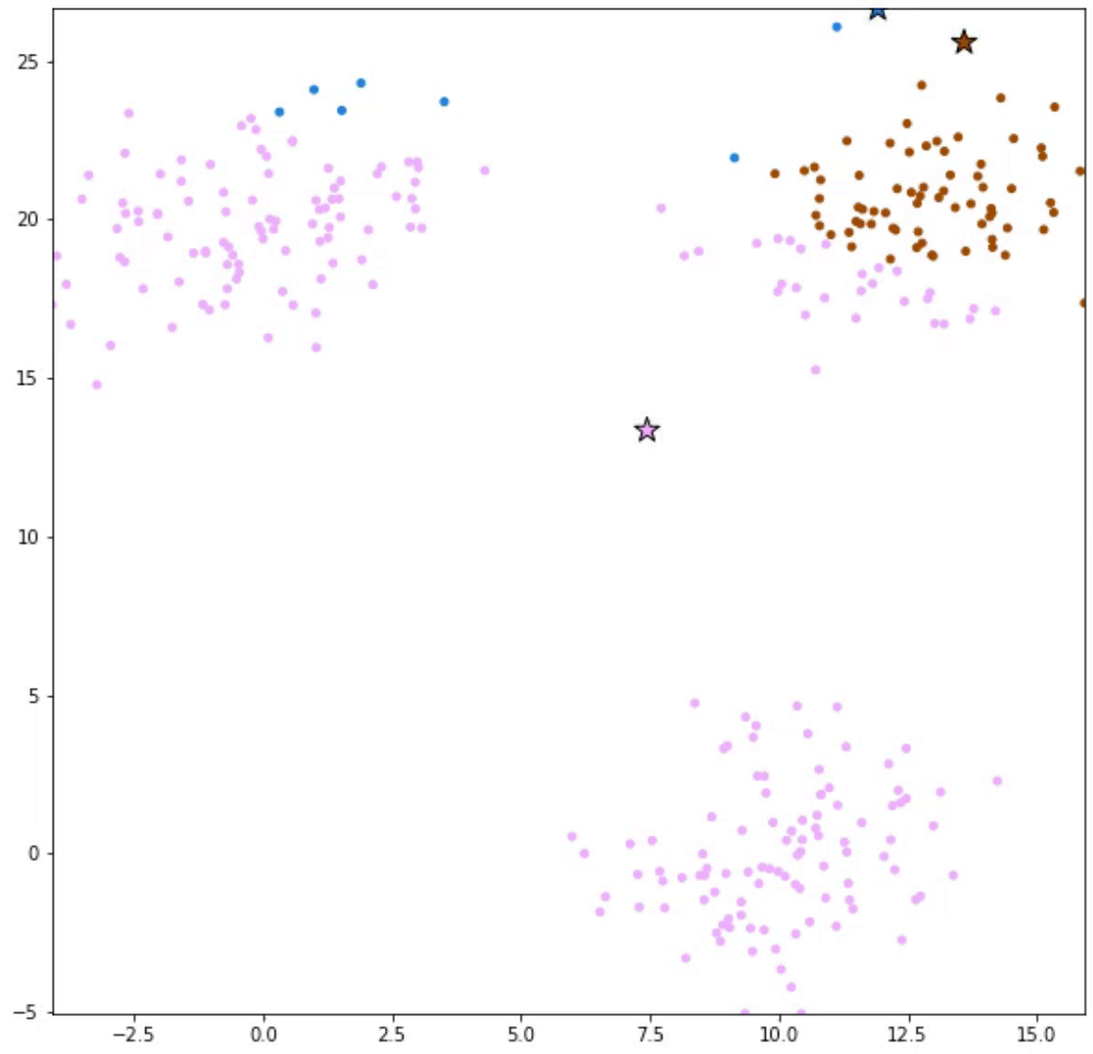


2. **Usage of a trained model:**

- for each test datapoint $\mathbf{x}_i$:
  - compute the distance $d(\mathbf{x}_i, \mu_k)$ to each cluster center μ_k
  - assign the datapoint to cluster C_k with the smallest distance $d(\mathbf{x}_i, \mu_k)$

Testing

K-means for analysis of companies



For a well-separated sets of points, the K-means finds clusters in a small number of steps.

K-means: energy-based formulation

We can alternatively re-formulate the K-means algorithm as the problem of minimization of the following loss function (distortion function):

$$L = \min_{r_{nk},\,\mu_k} \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\left\|\mathbf{x}_n - \mu_k\right\|^2$$

Optimization

non-convex

i.e multiple local minima

where $r_{nk} = 1 \; if \; \mathbf{x}_i \in C_k, else \; r_{nk} = 0$

Minimization of the loss function:

- choose initial values of $\{\mu_k\}$
- repeat:
  - Minimize w.r.t. $\{r_{nk}\}$ for fixed values of $\{\mu_k\}$
  - Minimize w.r.t. $\{\mu_k\}$ for fixed values of $\{r_{nk}\}$

K-means: energy-based formulation

We can alternatively re-formulate the K-means algorithm as the problem of minimization of the following loss function (distortion function):

$$L = \min_{r_{nk},\,\mu_k} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \left\| \mathbf{x}_n - \mu_k \right\|^2$$

where $r_{nk} = 1 \; if \; \mathbf{x}_i \in C_k, else \; r_{nk} = 0$

Expectation Maximization (**EM**) algorithm

Minimization of the loss function:
- choose initial values of centroids $\{\mu_k\}$
- repeat:
  - Minimize w.r.t. $\{r_{nk}\}$ for fixed values of $\{\mu_k\}$ (**M-step**)
  - Minimize w.r.t. $\{\mu_k\}$ for fixed values of $\{r_{nk}\}$ (**E-step**)

K-means: energy-based formulation

We can alternatively re-formulate the K-means algorithm as the problem of minimization of the following loss function (distortion function):

$$L = \min_{r_{nk},\,\mu_k} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \left\| \mathbf{x}_n - \mu_k \right\|^2$$

where $r_{nk} = 1 \; if \; \mathbf{x}_i \in C_k, else \; r_{nk} = 0$

Minimization of the loss function:
- choose initial values of centroids $\{\mu_k\}$
- repeat:
  - Minimize w.r.t. $\{r_{nk}\}$ for fixed values of $\{\mu_k\}$ (**M-step**)
  - Minimize w.r.t. $\{\mu_k\}$ for fixed values of $\{r_{nk}\}$ (**E-step**)

Expectation Maximization (**EM**) algorithm

M-step: $r_{nk} = 1 \; if \; k = \arg\min_j \left\| \mathbf{x}_n - \mu_j \right\|^2 ; and \; 0 \; otherwise, \forall n$

K-means: energy-based formulation

We can alternatively re-formulate the K-means algorithm as the problem of minimization of the following loss function (distortion function):

$$L = \min_{r_{nk},\,\mu_k} \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\left\|\mathbf{x}_n - \mu_k\right\|^2$$

where $r_{nk} = 1\ if\ \mathbf{x}_i \in C_k, else\ r_{nk} = 0$

Expectation Maximization (**EM**) algorithm

Minimization of the loss function:
- choose initial values of centroids $\{\mu_k\}$
- repeat:
  - Minimize w.r.t. $\{r_{nk}\}$ for fixed values of $\{\mu_k\}$ (**M-step**)
  - Minimize w.r.t. $\{\mu_k\}$ for fixed values of $\{r_{nk}\}$ (**E-step**)

M-step: $r_{nk} = 1\ if\ k = \arg\min_j \left\|\mathbf{x}_n - \mu_j\right\|^2; and\ 0\ otherwise, \forall n$

E-step: $\dfrac{\partial L}{\partial \mu_k} = 0 \quad \Rightarrow \mu_k = \dfrac{\sum_{n=1} r_{nk}\mathbf{x}_n}{\sum_{n=1} r_{nk}}, k = 1,...,K$

K-means: notes on the algorithm

- The K-means algorithm is **deterministic**: repeated experiments with the same datasets and the same initial positions of centroids give the same result
- The optimization problem of loss minimization in the K-means algorithm is **non-convex**: repeated experiments with the same datasets but different initial positions of centroids generally give different results
- **Cluster stability**: clustering results may vary for a different dataset from the same data-generating distribution. Centroid positions are sensitive to outliers
- Control of **convergence speed and local minima**: more advanced initialization algorithms, e.g. the K++ Means.
- **Inputs for K-means**: data normalization and/or whitening can help the algorithm
- For large datasets: **Mini-Batch** or **Online K-means** clustering
- **Fake clusters**: the K-means will find clusters even when there are no clusters (e.g. the data is white noise)
- What is the **right value of K** to choose?
- Why the **Euclidean distance** as a measure of similarity?

Control question

Select all correct answers

1. The name “K-means” means that the number of clusters in this algorithm is always set to be K times the mean of the data.
2. The K-means clustering is a Probabilistic (Soft) clustering, where cluster probabilities are estimated empirically using different initializations.
3. The K-means algorithm has a unique solution due to a strict convexity of its objective function.
4. The K-means is simultaneously a Flat and Hard deterministic clustering algorithm, whose solution is non-unique due to non-convexity of its loss function.

Correct answer: 4

Guided Tour of Machine Learning in Finance

Week 3: Unsupervised Learning

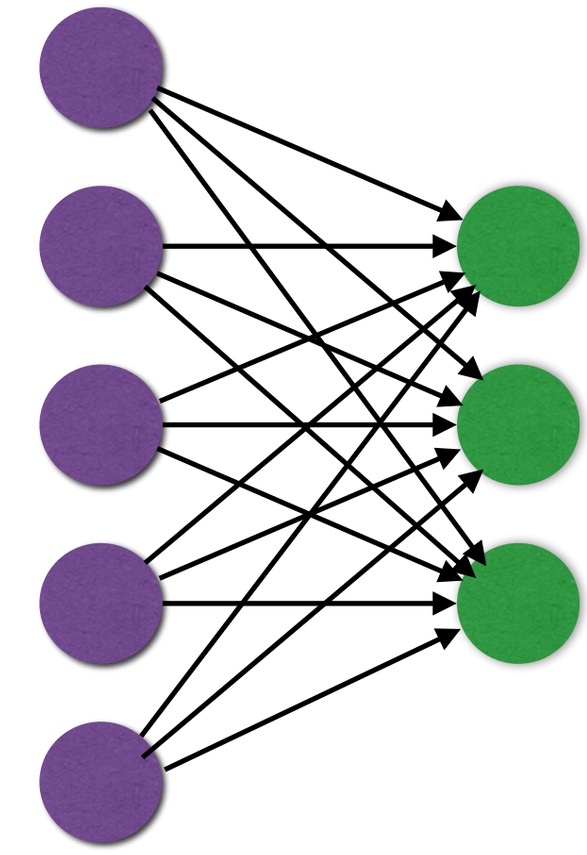
K-means neural algorithm

Igor Halperin

NYU Tandon School of Engineering, 2017

The K-means neural network

How to implement the K-means as a neural network:



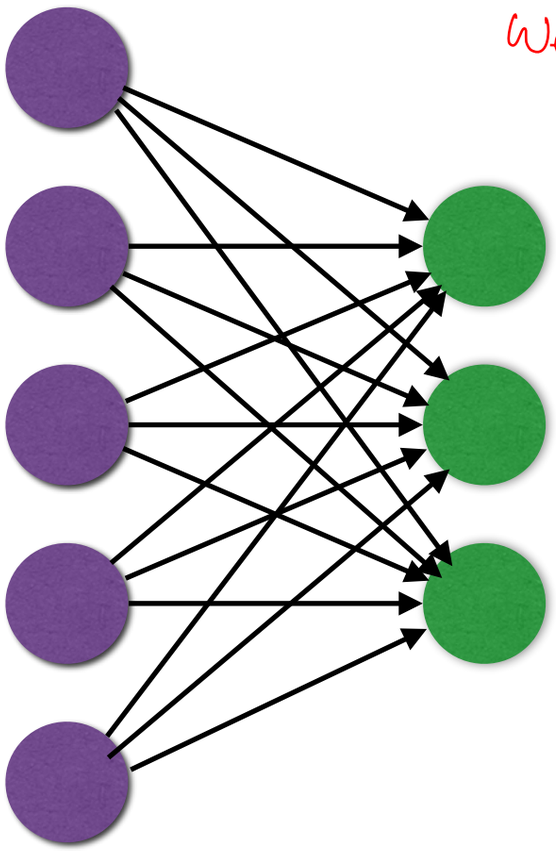
Inputs Hidden nodes

The K-means neural network

How to implement the K-means as a neural network:
Given an input vector $\mathbf{x}^{(i)} = (x_1^{(i)}, \ldots, x_N^{(i)})$, describe coordinates of K centroids relative to all component of $\mathbf{x}^{(i)}$ via a matrix $\mathbf{W}$ of "weights" of size $K \times N$ with elements w_{kn}

→ normalized distance
or
weight of cluster K in
n^{th} component of x_i

K^{th} row gives the weights of all components of x_i for a chosen cluster

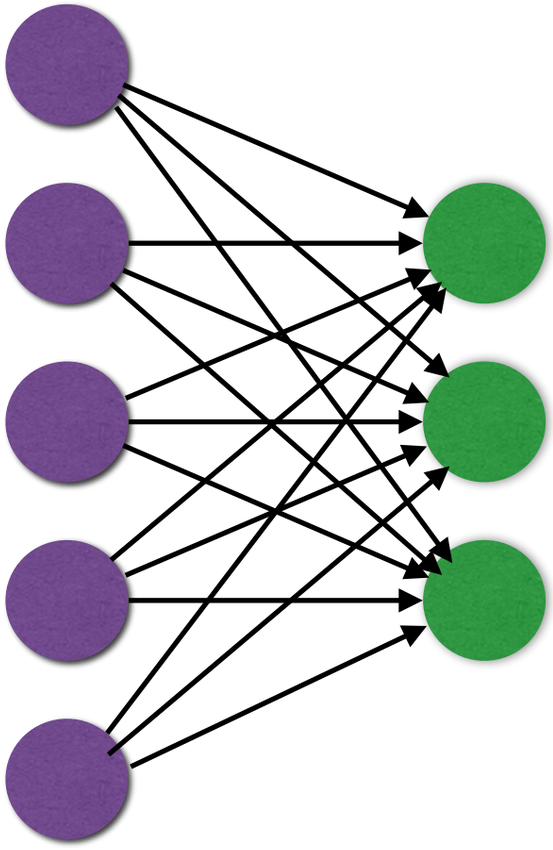


Inputs Hidden nodes

This is how cluster center positions are encoded as positions in a space of weights

The K-means neural network

How to implement the K-means as a neural network:
Given an input vector $\mathbf{x}^{(i)} = (x_1^{(i)},\ldots,x_N^{(i)})$, describe coordinates of K centroids relative to all component of $\mathbf{x}^{(i)}$via a matrix $\mathbf{W}$ of "weights" of size $K \times N$ with elements w_{kn}



Inputs Hidden nodes

Linear activation

Scalar product as a measure of similarity

$$h_k = \sum_n w_{kn} x_n = \mathbf{W}_\mathbf{k}^\mathbf{T} \cdot \mathbf{x}, \;\; k = 1,\ldots,K$$

The K-means neural network

How to implement the K-means as a neural network:
Given an input vector $\mathbf{x}^{(i)} = (x_1^{(i)}, \ldots, x_N^{(i)})$, describe coordinates of K centroids relative to all component of $\mathbf{x}^{(i)}$ via a matrix $\mathbf{W}$ of "weights" of size $K \times N$ with elements w_{kn}

→ Online setting

- new data points are added one-by-one
- mean is unknow till we finish scanning the whole dataset

Scalar product as a measure of similarity

Linear activation $h_k = \sum_n w_{kn} x_n = \mathbf{W_k^T} \cdot \mathbf{x}, \; k = 1, \ldots, K$

Neural Implementation

- repeat for a number of epochs:
  - for each datapoint:
    - compute all activations
    - choose the winning neuron with a maximal activation h_k
    - update the weights only for the winner neuron $\Delta w_{kn} = \eta(x_n - w_{kn})$

i.e adjust cluster centers towards true mean

Inputs Hidden nodes

Control question

Select all correct answers

1. The K-means algorithm can be implemented either in a batch mode (in one pass for the whole data), or in a mini-batch or online mode.
2. The mini-batch or online version of the K-means algorithm can be implemented using a single-layer Neural Network with competing neurons.
3. The online version of ANY Machine Learning algorithm, including yet unknown algorithms, can be implemented using a Neural Network with neurons that both compete and cooperate.

Correct answer: 1, 2

Guided Tour of Machine Learning in Finance

Week 3: Unsupervised Learning

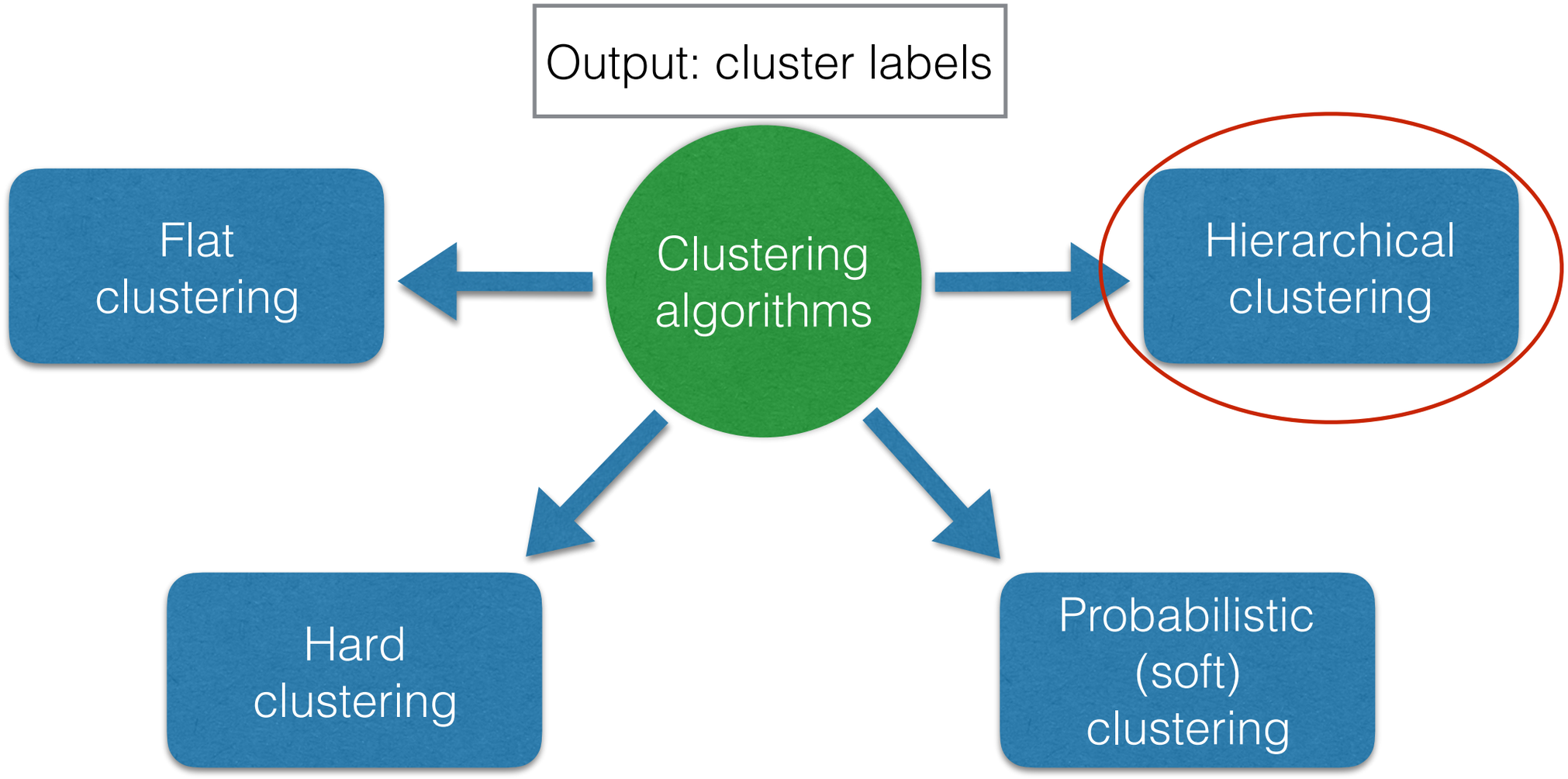
Hierarchical clustering algorithms

Igor Halperin

NYU Tandon School of Engineering, 2017

Types of clustering

Now consider hierarchical clustering methods



Why hierarchical clustering?

- Many complex physical, biological and social systems have nested hierarchical structures (sub-clusters within clusters, and so on)
- The hierarchical structure of interactions affects the dynamics of complex systems
- A quantitative description of hierarchies of the system is a key step in the modeling of complex systems, see P. Anderson (The Nobel Prize in Physics, 1977), “The More is Different”, Science 177 (1972), pp. 393-396.

Why hierarchical clustering?

- Many complex physical, biological and social systems have nested hierarchical structures (sub-clusters within clusters, and so on)
- The hierarchical structure of interactions affects the dynamics of complex systems
- A quantitative description of hierarchies of the system is a key step in the modeling of complex systems, see P. Anderson (The Nobel Prize in Physics, 1977), “The More is Different”, Science 177 (1972), pp. 393-396.

“In closing, I offer two examples from economics of what I hope to have said. Marx said that the quantitative differences become qualitative ones, but a dialogue in Paris in 1920’s sums it up even more clearly:

Why hierarchical clustering?

- Many complex physical, biological and social systems have nested hierarchical structures (sub-clusters within clusters, and so on)
- The hierarchical structure of interactions affects the dynamics of complex systems
- A quantitative description of hierarchies of the system is a key step in the modeling of complex systems, see P. Anderson (The Nobel Prize in Physics, 1977), “The More is Different”, Science 177 (1972), pp. 393-396.

“In closing, I offer two examples from economics of what I hope to have said. Marx said that the quantitative differences become qualitative ones, but a dialogue in Paris in 1920’s sums it up even more clearly:

FITZGERALD: The rich are different from us.

HEMINGWAY: Yes, they have more money.”

Hierarchical structures in finance

- Financial markets have hierarchical features (global indexes vs sector indexes vs geographic-based indexes).
- There are multi-factor asset pricing models with a hierarchical factor structure
- We follow a ML paradigm and try to learn a hierarchical structure directly from data!
- A hierarchical factor model with independent factors at different levels of the hierarchy can be extracted from a cluster found by a ML algorithm

* In classical Finance, we model hierarchical structures of risk factors via nested hierarchical factor models, where factors are observable & can be identified with market provided values of global & sector indices.

Agglomerative hierarchical clustering

1. Start with each data point forming its own cluster
2. Update clusters by adding to each point its closest neighbor
3. Merge the most "similar" cluster
4. Continue until convergence, or until a needed number of clusters is found

This is a "bottom-up" approach to clustering. Mergers are made in a greedy manner. Complexity in general for agglomerative clustering is $O(N^2 \log N)$, but is made $O(N^2)$ by two special cases: single linkage and complete linkage algorithms.

When N is large, one can first use K-means (complexity $O(KND)$), and then apply a hierarchical clustering to K-means centroids.

bottom-up → starts with old clusters & gradually builds more general clusters

greedy → each cluster mergers is only guaranteed to be optimal locally for this given step, but not necessarily for final objective

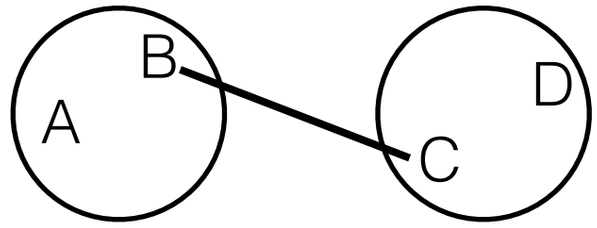
Choice of a metric

Metric	Definition
Euclidean distance	$\left\|\left\|X^{(1)} - X^{(2)}\right\|\right\|_2 = \sqrt{\sum_i \left(X_i^{(1)} - X_i^{(2)}\right)^2}$
Squared euclidean distance	$\left\|\left\|X^{(1)} - X^{(2)}\right\|\right\|_2^2 = \sum_i \left(X_i^{(1)} - X_i^{(2)}\right)^2$
Manhattan distance	$\left\|\left\|X^{(1)} - X^{(2)}\right\|\right\|_1 = \sum_i \left\|X_i^{(1)} - X_i^{(2)}\right\|$
Maximum distance	$\left\|\left\|X^{(1)} - X^{(2)}\right\|\right\|_\infty = \max_i \left\|X_i^{(1)} - X_i^{(2)}\right\|$
Mahalanobis distance	$\left\|\left\|X^{(1)} - X^{(2)}\right\|\right\|_2 = \sqrt{\left(X^{(1)} - X^{(2)}\right)^T \mathbf{S} \left(X^{(1)} - X^{(2)}\right)}$

Linkage methods

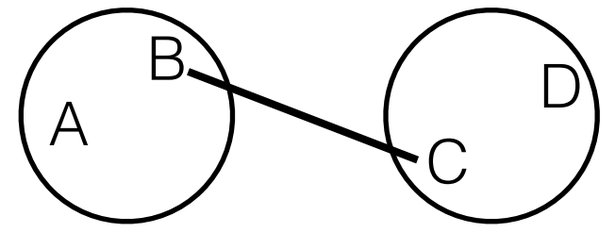
Name	Definition
Single linkage clustering	$\min\{d(a,b) : a \in A, b \in B\}$

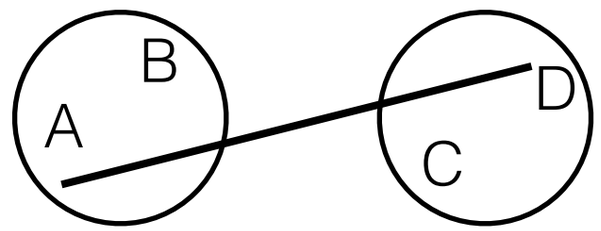
quadratic complexity in # points



Linkage methods

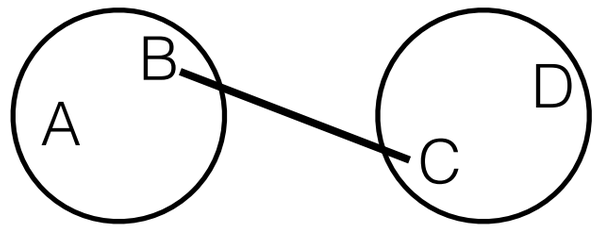
Name	Definition
Single linkage clustering	$\min\{d(a,b) : a \in A, b \in B\}$
Complete linkage clustering	$\max\{d(a,b) : a \in A, b \in B\}$

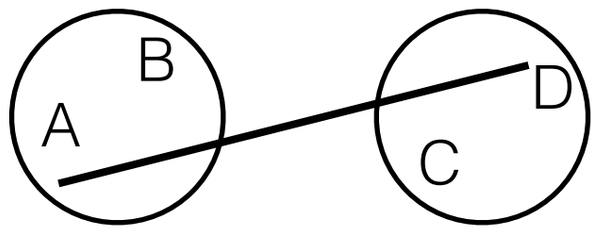


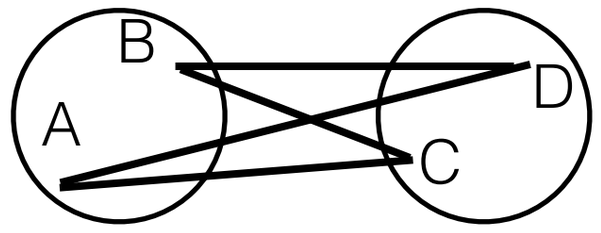


Linkage methods

Name	Definition
Single linkage clustering	$\min\{d(a,b) : a \in A, b \in B\}$
Complete linkage clustering	$\max\{d(a,b) : a \in A, b \in B\}$
Average linkage clustering	$\frac{1}{\lvert A\rvert \lvert B\rvert}\sum_{a\in A}\sum_{b\in B} d(a,b)$

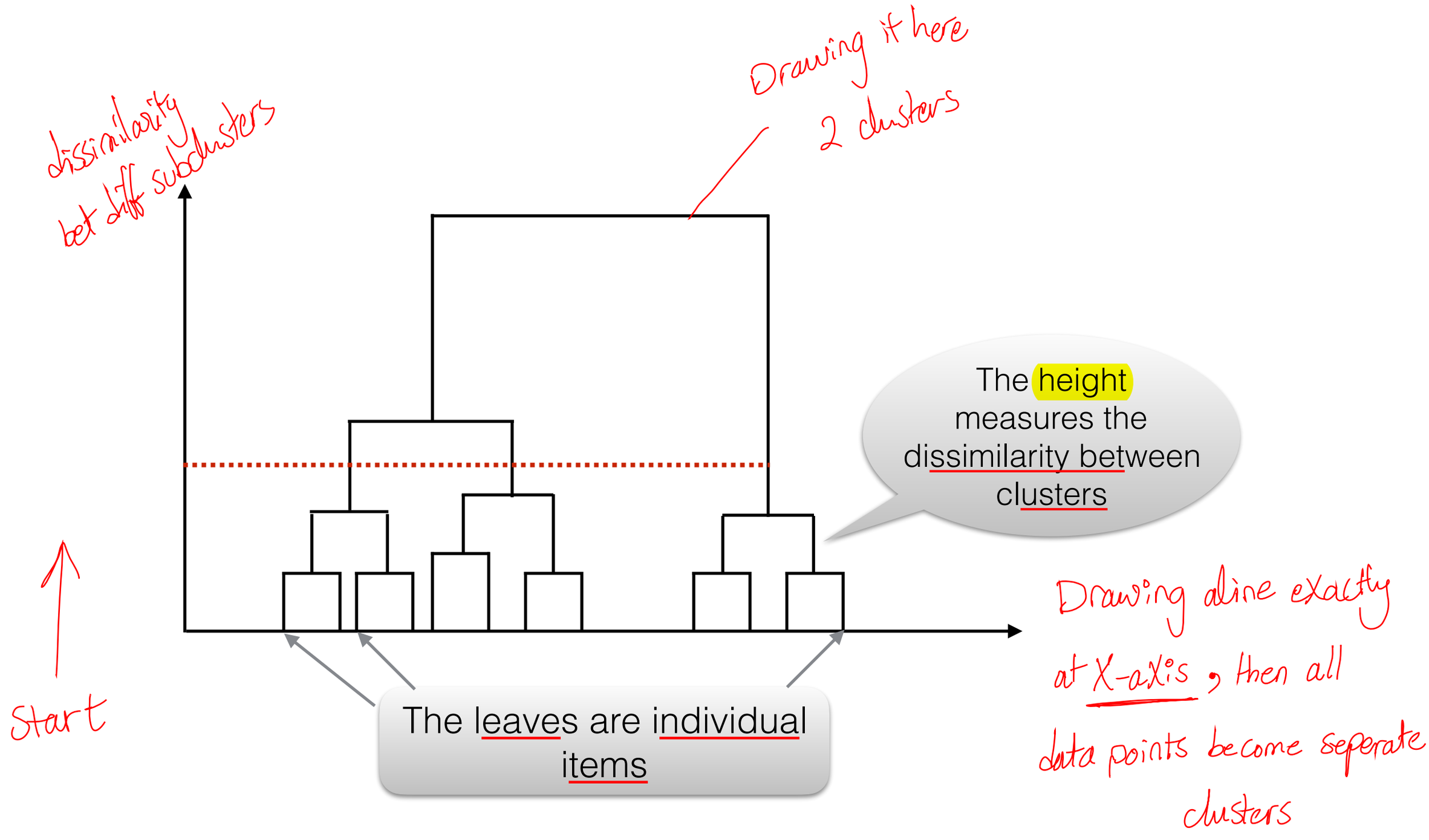






Dendrogram

Dendrogram is a binary tree representing the process of merging sub-clusters in a hierarchical cluster



Control question

Select all correct answers

1. Hierarchical clustering produces hierarchies of probability distributions for all cluster and sub-cluster labels.
2. Agglomerative clustering is a bottom-up type of hierarchical clustering that merges sub-clusters in a greedy manner.
3. While complexity of general agglomerative clustering methods is $O(N^2 \log N)$, it can be reduced to $O(N^2)$ for Single-Linkage and Complete Linkage methods.
4. For Average Linkage clustering, complexity is $O(N \log N)$, though so far it was observed only using GPU architectures.

Correct answers: 2,3

Guided Tour of Machine Learning in Finance

Week 3: Unsupervised Learning

Clustering and estimation of equity correlation matrix

Igor Halperin

NYU Tandon School of Engineering, 2017

Estimation of equity correlation matrix

Let $\mathbf{x}_i = \left\{ x_i^{(t)} \right\}_{t=1}^{T}$, $i = 1,\ldots,N$ be a vector of standardized log-returns of stock .

Empirical correlation matrix: $\hat{C}_{ij} = \frac{1}{T-1} \sum_{t=1}^{T} x_i^{(t)} x_j^{(t)}$

Estimation of equity correlation matrix

Let $\mathbf{x}_i = \left\{ x_i^{(t)} \right\}_{t=1}^{T}$, $i = 1,\ldots,N$ be a vector of standardized log-returns of stock .

Empirical correlation matrix: $\hat{C}_{ij} = \frac{1}{T-1} \sum_{t=1}^{T} x_i^{(t)} x_j^{(t)}$

The number of free parameters : $N(N-1)/2 \sim N^2/2$

The number of observations: NT

Estimation of equity correlation matrix

Let $\mathbf{x}_i = \left\{ x_i^{(t)} \right\}_{t=1}^{T}$, $i = 1,\ldots,N$ be a vector of standardized log-returns of stock .

Empirical correlation matrix: $\hat{C}_{ij} = \frac{1}{T-1} \sum_{t=1}^{T} x_i^{(t)} x_j^{(t)}$

The number of free parameters : $N(N-1)/2 \sim N^2/2$
The number of observations: NT

should have $T \gg N$
for a reliable estimation

of true $\hat{C}$ from emprical $\hat{C}$

Estimation of equity correlation matrix

Let $\mathbf{x}_i = \left\{ x_i^{(t)} \right\}_{t=1}^{T}$, $i = 1,\ldots,N$ be a vector of standardized log-returns of stock .

Empirical correlation matrix: $\hat{C}_{ij} = \frac{1}{T-1} \sum_{t=1}^{T} x_i^{(t)} x_j^{(t)}$

The number of free parameters : $N(N-1)/2 \sim N^2/2$
The number of observations: NT

should have $T \gg N$ for a reliable estimation

The "true" correlation matrix may differ from the empirical correlation matrix due to the "observational noise", whose amount depends on the ratio T/N

Estimation of equity correlation matrix

Let $\mathbf{x}_i = \left\{ x_i^{(t)} \right\}_{t=1}^{T}$, $i = 1,\ldots,N$ be a vector of standardized log-returns of stock .

Empirical correlation matrix: $\hat{C}_{ij} = \frac{1}{T-1} \sum_{t=1}^{T} x_i^{(t)} x_j^{(t)}$

The number of free parameters : $N(N-1)/2 \sim N^2/2$
The number of observations: NT

should have $T \gg N$ for a reliable estimation

The "true" correlation matrix may differ from the empirical correlation matrix due to the "observational noise", whose amount depends on the ratio T/N

Estimation of a "noise-filtered" equity correlation matrix is important for:
- trading
- portfolio management
- systemic risk monitoring

Estimation of equity correlation matrix

Some de-noising methods:

1. Methods based on the Random Matrix Theory (RMT) (E. Stanley et. al. 1999)

2. "Shrinkage" methods (Ledoit and Wold 2003) Bayesian-based

3. Clustering-based filtering: based on distances between individual points, and aggregation of linkages between sub-clusters

FORMALIZATION

Estimation of equity correlation matrix

Clustering:
Given N items with pair-wise distances d_{ij}, divide them into K groups so that minimum distance between items in different groups is maximized.

Maximum minimum distance:
-maintain clusters as a set of connected components of a graph
- iteratively combine the clusters containing the two closest items by adding an edge between them
- stop when there are K clusters

Estimation of equity correlation matrix

Clustering:
Given N items with pair-wise distances d_{ij}, divide them into K groups so that minimum distance between items in different groups is maximized.

Maximum minimum distance:
-maintain clusters as a set of connected components of a graph
- iteratively combine the clusters containing the two closest items by adding an edge between them
- stop when there are K clusters

This is the **Kruskal algorithm**, an example of a **single-linkage agglomerative clustering.**

operates on distances, while we are only given equity correlation

Estimation of equity correlation matrix

Clustering:
Given N items with pair-wise distances d_{ij}, divide them into K groups so that minimum distance between items in different groups is maximized.

Maximum minimum distance:
-maintain clusters as a set of connected components of a graph
- iteratively combine the clusters containing the two closest items by adding an edge between them
- stop when there are K clusters

This is the **Kruskal algorithm**, an example of a **single-linkage agglomerative clustering.**

Pair-wise distances: $$d_{ij} = \sqrt{2(1-\rho_{ij})}$$

This choice fulfills the three axioms of an Euclidean metric:
- $d_{ij} = 0 \; iff \; i = j$
- $d_{ij} = d_{ji}$
- $d_{ij} \le d_{ik} + d_{kj}, \quad \forall i,j,k$

symmetry
triangle inequality

Control question

Select all correct answers

1. Correlation matrices are most reliably estimated when $T \ll N$.
2. A clustering-based filtering of correlation matrices is based on distances between individual points, and aggregation of linkages between sub-clusters
3. The Kruskal algorithm belongs in the class of Complete Linkage methods.
4. Pair-wise distances in clustering-based methods for correlation matrices are determined in terms of pair-wise correlations.

Correct answers: 2, 4

Guided Tour of Machine Learning in Finance

Week 3: Unsupervised Learning

Minimum Spanning Trees, Kruskal algorithm, and equity correlation matrices

Igor Halperin

NYU Tandon School of Engineering, 2017

The Kruskal algorithm

Task:
Given a connected weighted graph with N nodes and connection weights d_{ij}, find a minimum spanning tree (MST) - a graph without loops connecting N nodes with $N-1$ links, such that the total weight of all edges is minimized.

Meaning: from $N(N-1)/2$ links, select $N-1$ shortest links that span all the nodes without forming loops.

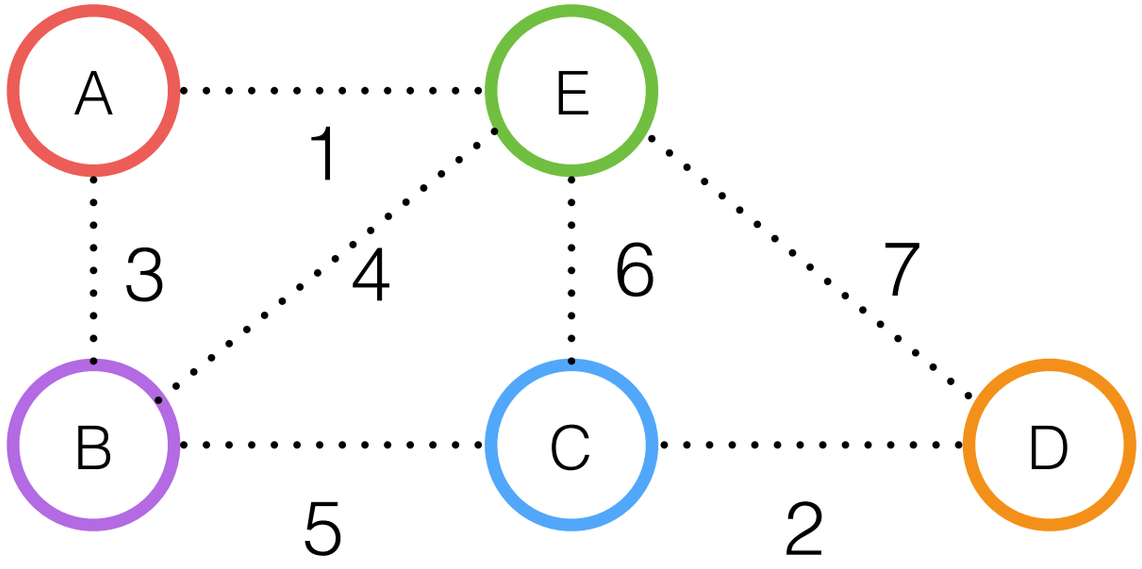
The Kruskal algorithm:

1. Find a minimum weight on a graph. Color it in any color (e.g. red)
2. Find the minimum uncolored edge that does not cross a colored or a red circuit. Mark this edge by a new color.
3. Repeat step 2 until connecting every vertex on the graph. The red edges form a MST.

The Kruskal algorithm

Wikipedia's example $(N = 5)$

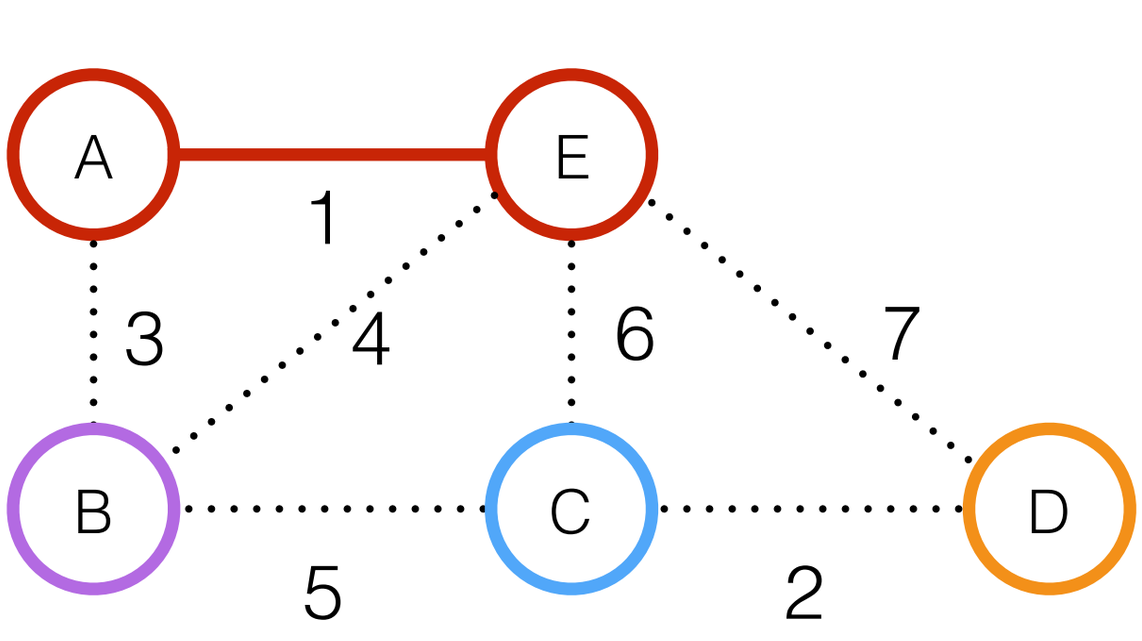
Edge	AE	CD	AB	BE	BC	EC	ED
Weight	1	2	3	4	5	6	7



The Kruskal algorithm

Wikipedia's example $(N = 5)$

Edge	AE	CD	AB	BE	BC	EC	ED
Weight	1	2	3	4	5	6	7

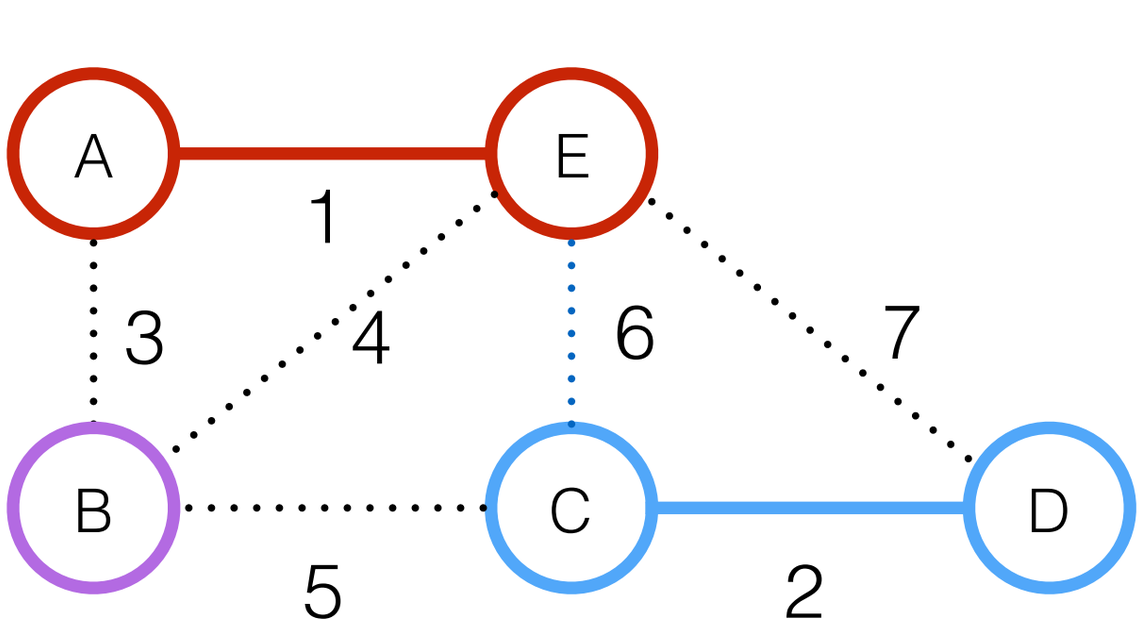


Select the smallest weight

The Kruskal algorithm

Wikipedia's example $(N = 5)$

Edge	AE	CD	AB	BE	BC	EC	ED
Weight	1	2	3	4	5	6	7

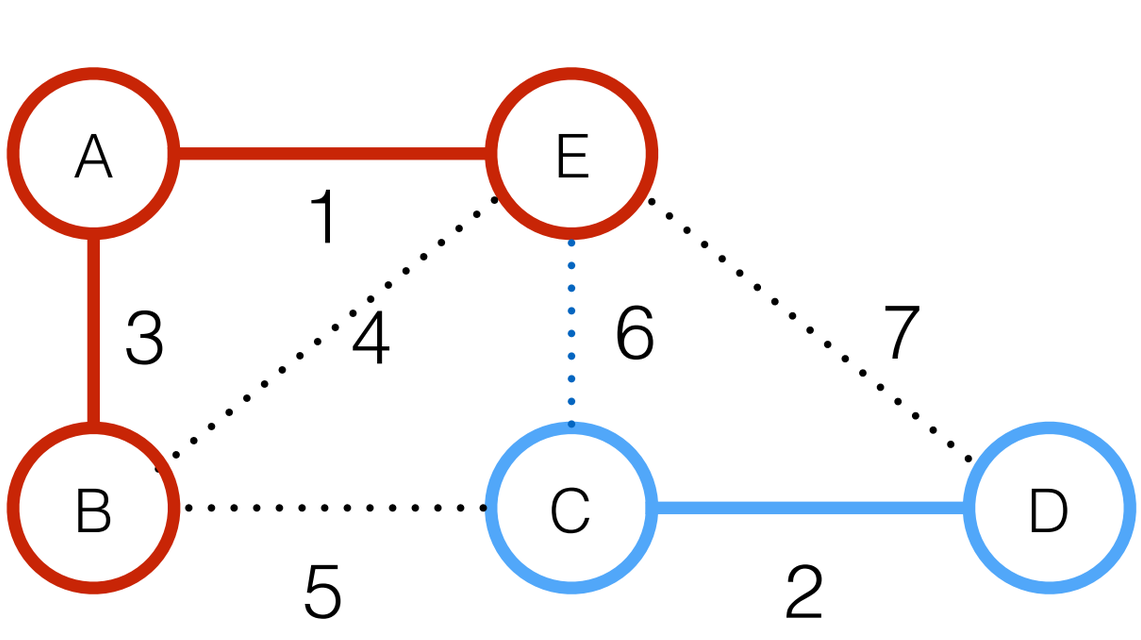


Select another smallest weight

The Kruskal algorithm

Wikipedia's example $(N = 5)$

Edge	AE	CD	AB	BE	BC	EC	ED
Weight	1	2	3	4	5	6	7

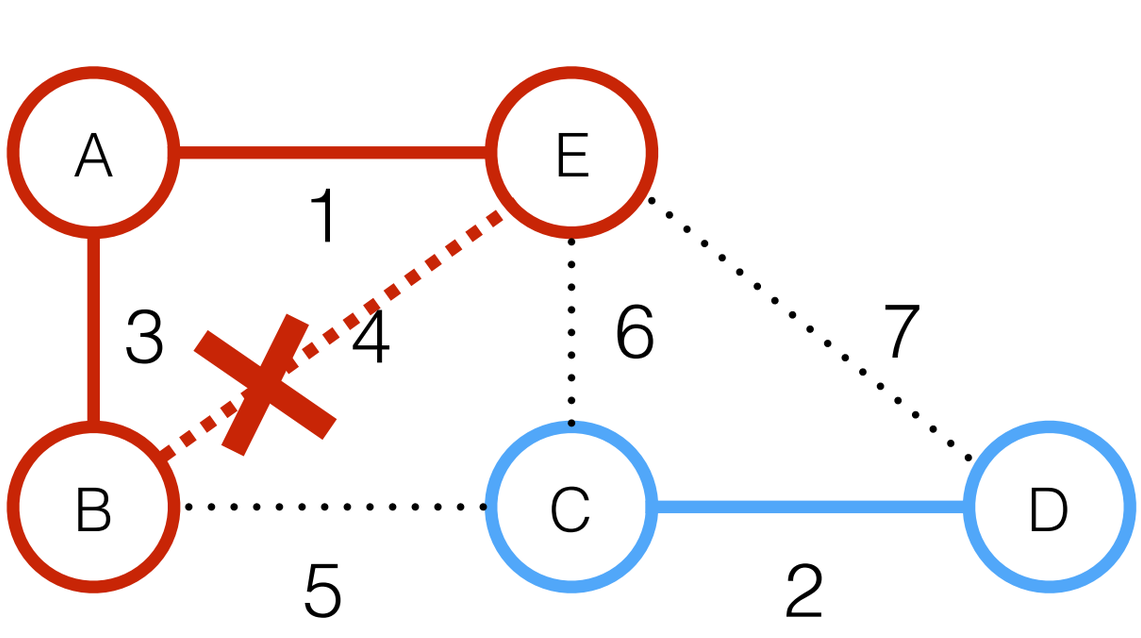


Select another smallest weight

The Kruskal algorithm

Wikipedia's example $(N = 5)$

Edge	AE	CD	AB	BE	BC	EC	ED
Weight	1	2	3	4	5	6	7

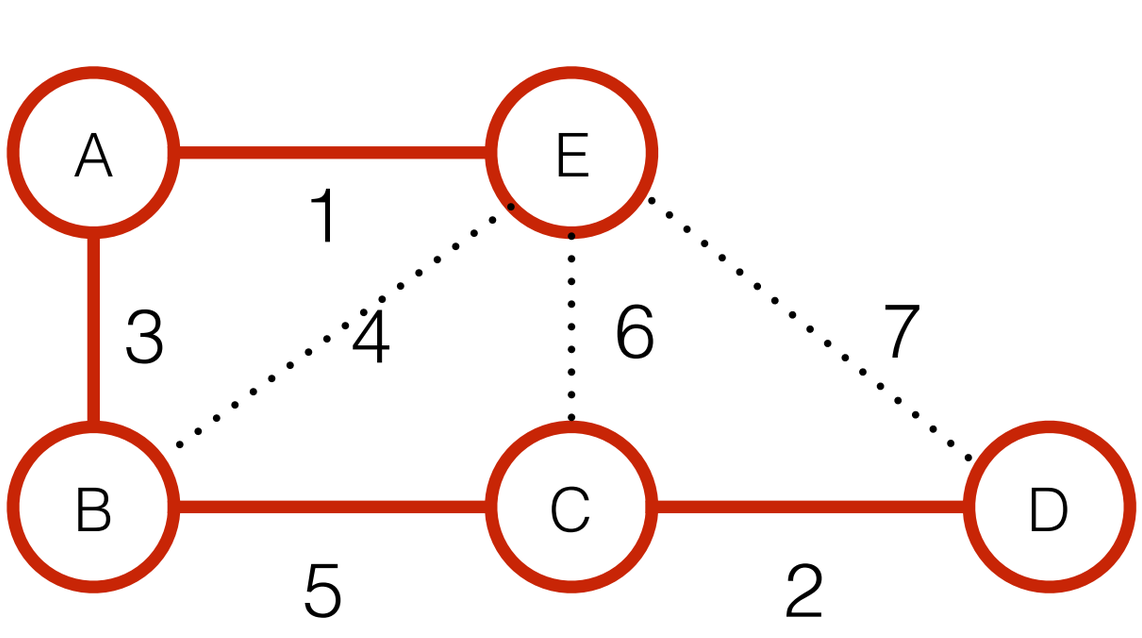


Selecting BE creates a loop!

The Kruskal algorithm

Wikipedia's example $(N = 5)$

Edge	AE	CD	AB	BE	BC	EC	ED
Weight	1	2	3	4	5	6	7

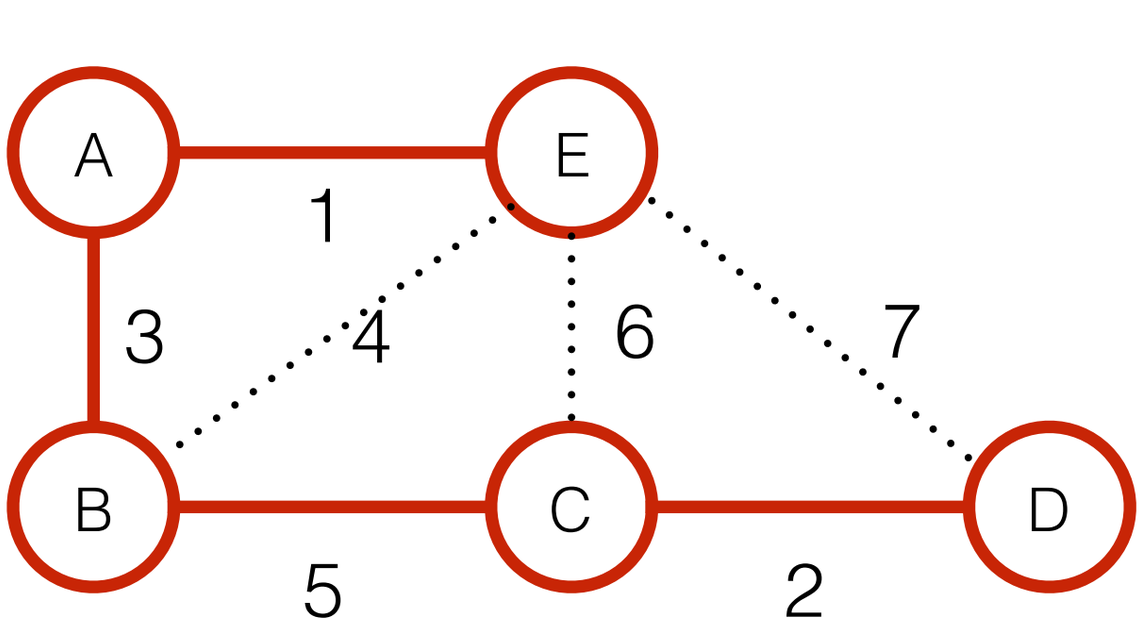


The minimum spanned tree is completed!

The Kruskal algorithm

Wikipedia's example $(N = 5)$

Edge	AE	CD	AB	BE	BC	EC	ED
Weight	1	2	3	4	5	6	7



The minimum spanned tree is completed!

The MST compresses the data representation from $N(N-1)/2$ parameters to $N-1$ parameters

Clustering-based filtering

Use the MTS to de-noise the empirical distance matrix $D(\Delta t) = \left\{ d_{ij}(\Delta t) \right\}_{i,j=1}^{N}$ (and the correlation matrix $\mathbf{C}(\Delta \mathrm{t})$!) matrix by replacing

$$d_{ij} \rightarrow d_{ij}^{<}$$

Maximum distance d_{kl} among single steps in a path from i to j in a MST

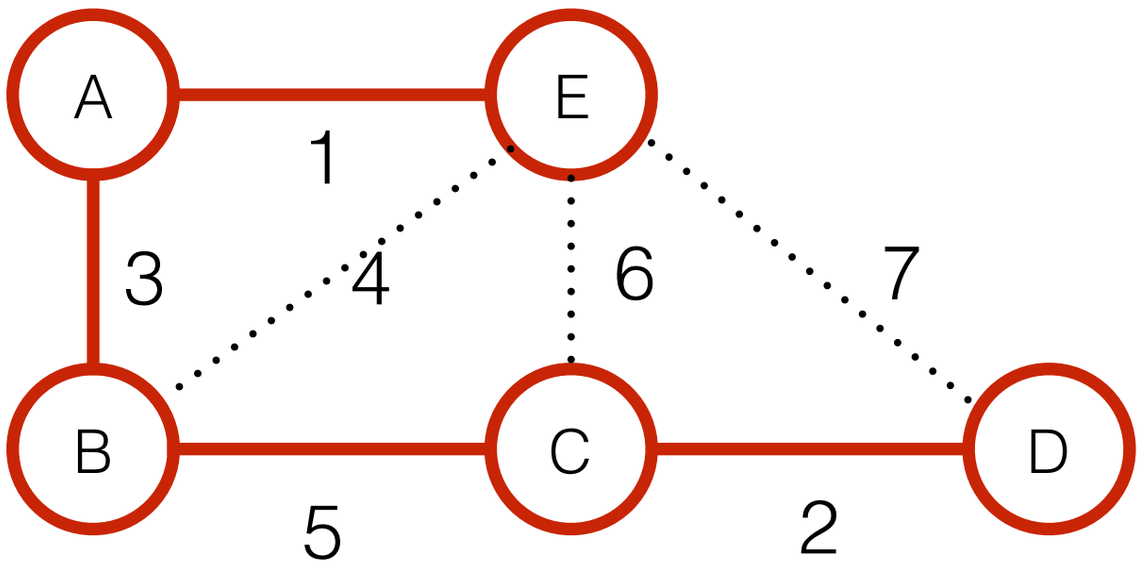
Clustering-based filtering

Use the MTS to de-noise the empirical distance matrix $D(\Delta t) = \left\{ d_{ij}(\Delta t) \right\}_{i,j=1}^{N}$ (and the correlation matrix $\mathrm{C}(\Delta \mathrm{t})$!) matrix by replacing

$$d_{ij} \rightarrow d_{ij}^{<}$$

Maximum distance d_{kl} among single steps in a path from i to j in a MST

Edge	AE	CD	AB	BE	BC	EC	ED
d_{ij}	1	2	3	4	5	6	7



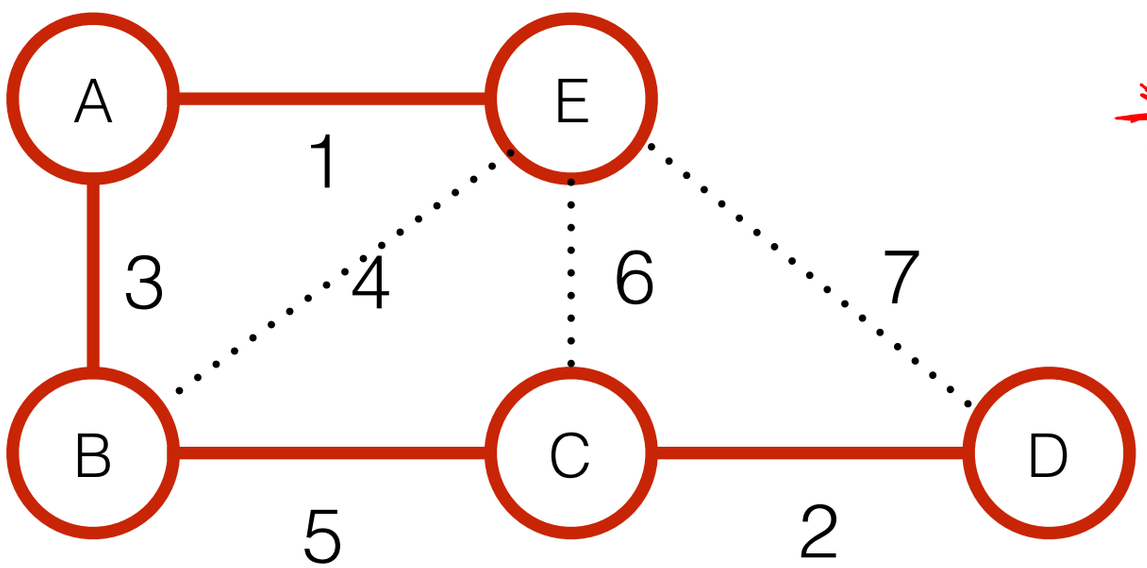
Clustering-based filtering

Use the MTS to de-noise the empirical distance matrix $D(\Delta t) = \left\{ d_{ij}(\Delta t) \right\}_{i,j=1}^{N}$ (and the correlation matrix $\mathrm{C}(\Delta \mathrm{t})$!) matrix by replacing

$$d_{ij} \rightarrow d_{ij}^{<}$$

Maximum distance d_{kl} among single steps in a path from i to j in a MST

Edge	AE	CD	AB	BE	BC	EC	ED
d_{ij}	1	2	3	4	5	6	7
$d_{ij}^{<}$	1	2	3	**3**	5	**5**	**5**



→ to be replaced by new weights computed from MST

Clustering-based filtering

Use the MTS to de-noise the empirical distance matrix $D(\Delta t) = \left\{ d_{ij}(\Delta t) \right\}_{i,j=1}^{N}$ (and the correlation matrix $\mathrm{C}(\Delta \mathrm{t})$!) matrix by replacing

$$d_{ij} \rightarrow d_{ij}^{<}$$

Maximum distance d_{kl} among single steps in a path from i to j in a MST (the Subdominant Ultrametric Distance)

Edge	AE	CD	AB	BE	BC	EC	ED
d_{ij}	1	2	3	4	5	6	7
$d_{ij}^{<}$	1	2	3	**3**	5	**5**	**5**

* New corrected distance $d_{ij}^{<}$ will be always equal or smaller than the original distance

Ultrametric distance: violates the triangle inequality of an Euclidean metric: satisfying the rest

$$d_{ij} \leq \max_k \left\{ d_{ik}, d_{kj} \right\}$$

Subdominant Ultrametric - the largest ultrametric among those that are less or equal to d_{ij}

Application for portfolio optimization

Application for portfolio optimization (E. Pantaleo et al, “When Do Improved Covariance Matrix Estimators Enhance Portfolio Optimization? An empirical study of nine estimators”, 2010):

- Applied different filtering techniques to a sample covariance matrix of US stocks
- Compared realized risk for Markowitz-optimal portfolios obtained with different filtered covariance matrices

Application for portfolio optimization

Application for portfolio optimization (E. Pantaleo et al, “When Do Improved Covariance Matrix Estimators Enhance Portfolio Optimization? An empirical study of nine estimators”, 2010):

- Applied different filtering techniques to a sample covariance matrix of US stocks
- Compared realized risk for Markowitz-optimal portfolios obtained with different filtered covariance matrices

- Alternatives used:
    (i) Sample covariance matrix
    (ii) Random Matrix Theory (RMT) based filtering
    (iii) MST and other graph-based filtering

Application for portfolio optimization

Application for portfolio optimization (E. Pantaleo et al, “When Do Improved Covariance Matrix Estimators Enhance Portfolio Optimization? An empirical study of nine estimators”, 2010):

- Applied different filtering techniques to a sample covariance matrix of US stocks
- Compared realized risk for Markowitz-optimal portfolios obtained with different filtered covariance matrices

- Alternatives used:
  (i) Sample covariance matrix
  (ii) Random Matrix Theory (RMT) based filtering
  (iii) MST and other graph-based filtering

Results:

- Substantially lower-risk optimal portfolios for the regime $T / N > 1$ when short sales are allowed

- No significant improvements over the sample covariance matrix when short sales are not allowed, and/or $T / N < 1$

Control question

Select all correct answers

1. A Minimum Spanning Tree (MTS) constructs a graph with N nodes without loops using $N-1$ links, such that the total weight of all links is **maximized**.
2. A Minimum Spanning Tree (MTS) constructs a graph with N nodes without loops using $N-1$ links, such that the total weight of all links is **minimized**.
3. The MST compresses the parametrization of a correlation matrix from $N(N-1)/2$ parameters to $N-1$ parameters
4. Graph-based filtering of correlation matrices produce lower-risk optimal portfolios for the regime $T/N<1$ when short sales are not allowed
5. Graph-based filtering of correlation matrices produce lower-risk optimal portfolios for the regime $T/N>1$ when short sales are allowed

Correct answers: 2, 3, 5

Guided Tour of Machine Learning in Finance

Week 3: Unsupervised Learning

Probabilistic clustering

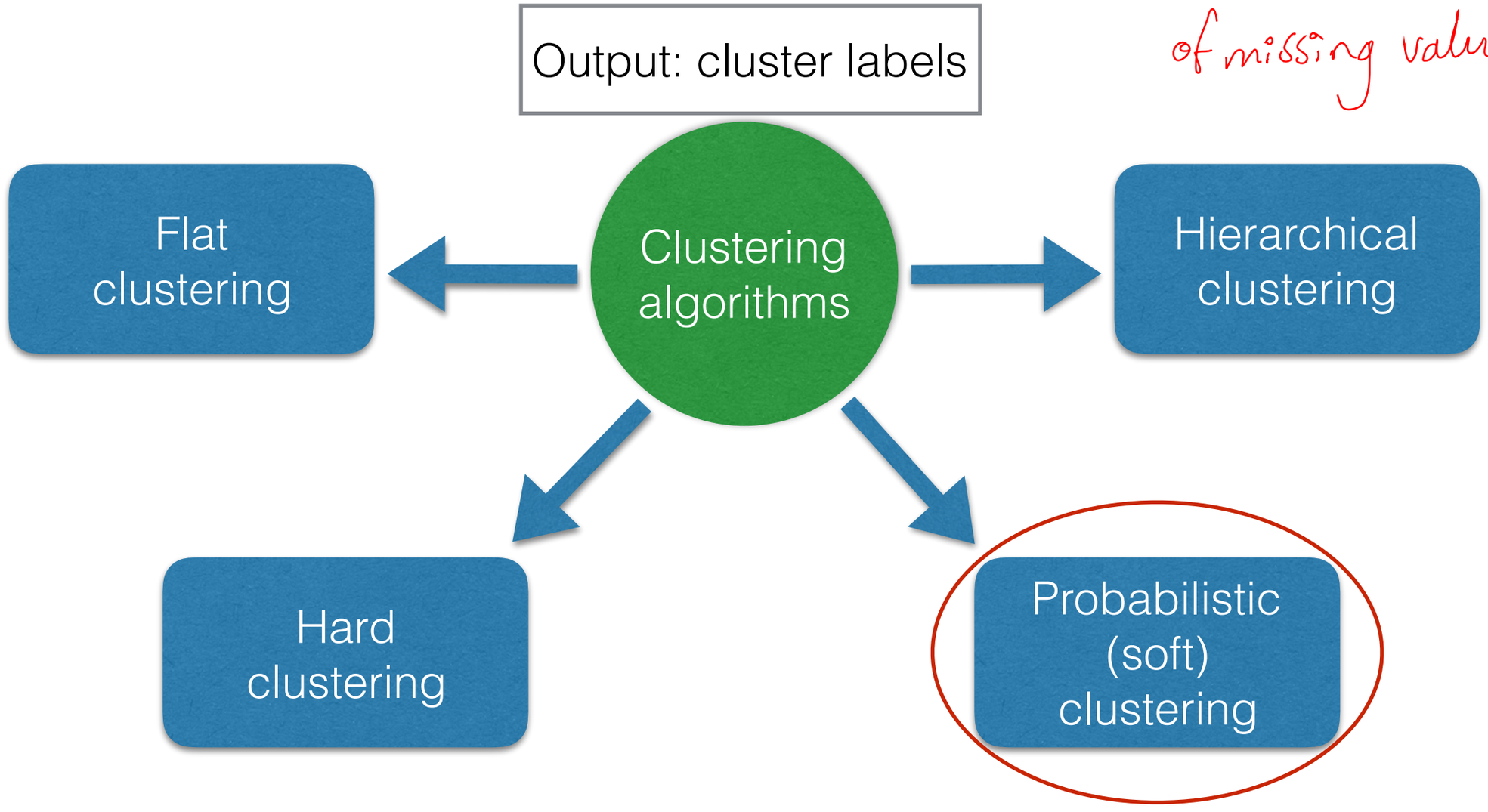
Igor Halperin

NYU Tandon School of Engineering, 2017

Types of clustering

Now consider Soft (Probabilistic) clustering methods

allowing simulating data, thus better handling of missing values



Mixtures of Gaussians

Mixture models are used to describe complex distributions. multi-model
A mixture of Gaussians is a popular common choice:

$$p(y \mid \Theta) = \sum_{k-1}^{K} \pi_k \, p(y \mid \theta_k)$$

Here each component is a Gaussian with $\theta_k = (\mu_k, \Sigma_k)$ for its mean and variance.
The component weight is $0 \le \pi_k \le 1$ with

$$\sum_{k-1}^{K} \pi_k = 1$$

Mixtures of Gaussians

Mixture models are used to describe complex distributions.
A mixture of Gaussians is a popular common choice:

$$p(y \mid \Theta) = \sum_{k-1}^{K} \pi_k \, p(y \mid \theta_k)$$

Here each component is a Gaussian with $\theta_k = (\mu_k, \Sigma_k)$ for its mean and variance.
The component weight is $0 \le \pi_k \le 1$ with

$$\sum_{k-1}^{K} \pi_k = 1$$

Weights can be described by a latent (hidden) variable s such that $s = k$ if the data point was generated by component k. Then we can write it as

$$p(y \mid \Theta) = \sum_{k-1}^{K} P(s = k \mid \pi) \, p(y \mid s = k, \theta)$$

Estimation of model now reduces to estimation of parameters $\theta_k = (\mu_k, \Sigma_k)$, as well as inference of the hidden variable s

Mixtures of Gaussians

Mixture models are used to describe complex distributions.
A mixture of Gaussians is a popular common choice:

$$p(y \mid \Theta) = \sum_{k-1}^{K} \pi_k \, p(y \mid \theta_k)$$

Here each component is a Gaussian with $\theta_k = (\mu_k, \Sigma_k)$ for its mean and variance.
The component weight is $0 \le \pi_k \le 1$ with

$$\sum_{k-1}^{K} \pi_k = 1$$

Weights can be described by a latent (hidden) variable s such that $s = k$ if the data point was generated by component k. Then we can write it as

$$p(y \mid \Theta) = \sum_{k-1}^{K} P(s = k \mid \pi) \, p(y \mid s = k, \theta)$$

Estimation of model now reduces to estimation of parameters $\theta_k = (\mu_k, \Sigma_k)$, as well as inference of the hidden variable s

This can be done using the **EM algorithm**!

The EM algorithm

The EM algorithm = Expectation Maximization.
Let $p(y,x \mid \theta)$ be the joint probability of observed data y and latent variables x
We have for the likelihood of data

$$L(\theta) = \log p(y \mid \theta) = \log \int p(x,y \mid \theta)\, dx = \log \int q(x) \frac{p(x,y \mid \theta)}{q(x)} dx$$

$$\geq \int q(x) \log \frac{p(x,y \mid \theta)}{q(x)} dx \triangleq F(q,\theta)$$

Here $q(x)$ is some arbitrary density over latent variables, and the last inequality is due to concavity of the logarithm (Jensen's inequality)

The EM algorithm: iterate between maximization of this low bound on the log-likelihood as a function of q, and as a function of θ

The EM algorithm

The EM algorithm = Expectation Maximization.

E-step: Optimize wrt $q(x)$ while keeping θ fixed:

$$q_k(x) = \arg\max_{q(x)} \int q(x) \frac{p(x,y \mid \theta_{k-1})}{q(x)} dx = \arg\max_{q(x)} \left[\log p(y \mid \theta_{k-1}) + \int q(x) \frac{p(x \mid y,\theta_{k-1})}{q(x)} dx \right]$$

$$\Rightarrow q_k(x) = p(x \mid y,\theta_{k-1})$$

most probable

The EM algorithm

The EM algorithm = Expectation Maximization.
E-step: Optimize wrt $q(x)$ while keeping θ fixed:

$$q_k(x) = \arg\max_{q(x)} \int q(x) \frac{p(x,y \mid \theta_{k-1})}{q(x)} dx = \arg\max_{q(x)} \left[\log p(y \mid \theta_{k-1}) + \int q(x) \frac{p(x \mid y,\theta_{k-1})}{q(x)} dx \right]$$

$$\Rightarrow q_k(x) = p(x \mid y,\theta_{k-1})$$

M-step: Optimize wrt θ, while holding the distribution $q(x)$ fixed:

$$\theta_k = \arg\max_{\theta} \int q_k(x) \frac{p(x,y \mid \theta)}{q(x)} dx$$

$$\Rightarrow \theta_k = \arg\max_{\theta} \int q(x) \log p(x,y \mid \theta) dx$$

The EM algorithm

The EM algorithm = Expectation Maximization.
E-step: Optimize wrt $q(x)$ while keeping θ fixed:

$$q_k(x) = \arg\max_{q(x)} \int q(x) \frac{p(x,y \mid \theta_{k-1})}{q(x)} dx = \arg\max_{q(x)} \left[\log p(y \mid \theta_{k-1}) + \int q(x) \frac{p(x \mid y,\theta_{k-1})}{q(x)} dx \right]$$

$$\Rightarrow q_k(x) = p(x \mid y,\theta_{k-1})$$

M-step: Optimize wrt θ, while holding the distribution $q(x)$ fixed:

$$\theta_k = \arg\max_{\theta} \int q_k(x) \frac{p(x,y \mid \theta)}{q(x)} dx$$

$$\Rightarrow \theta_k = \arg\max_{\theta} \int q(x) \log p(x,y \mid \theta) dx$$

- The EM algorithm is guaranteed to increase the likelihood or keep it constant at each iteration, and to find a local maximum of the log-likelihood.
- The EM algorithm can be applied to any model with hidden variables: Gaussian mixtures, Factor models, probabilistic PCA etc.
- For Gaussian mixtures, $q(x)$ reduces to a discrete distribution $P(s = k \mid \pi)$ over the mixture components.

Control question

Select all correct answers

1. Probabilistic Clustering focuses on data points that have reasonably high probability to be present in the data, and ignores the rest.
2. The Expectation Maximization algorithm expects that all data in a dataset is unlabelled, and maximizes this expectation by removing all labels, even when they are present in the data.
3. A Hard (non-probabilistic) Clustering can be obtained as a deterministic limit of Probabilistic Clustering.
4. The EM algorithm iterates between two steps: the E-step eliminates improbable points, and the M-step maximizes the probability to see the rest of points.
5. The EM algorithm iterates between two steps: the E-step estimates the probability distribution on hidden variables with model parameters fixed, and the M-step maximizes the low bound on the log-likelihood by adjusting model parameters while keeping the distribution over hidden variables fixed.

Correct answers: 3, 5