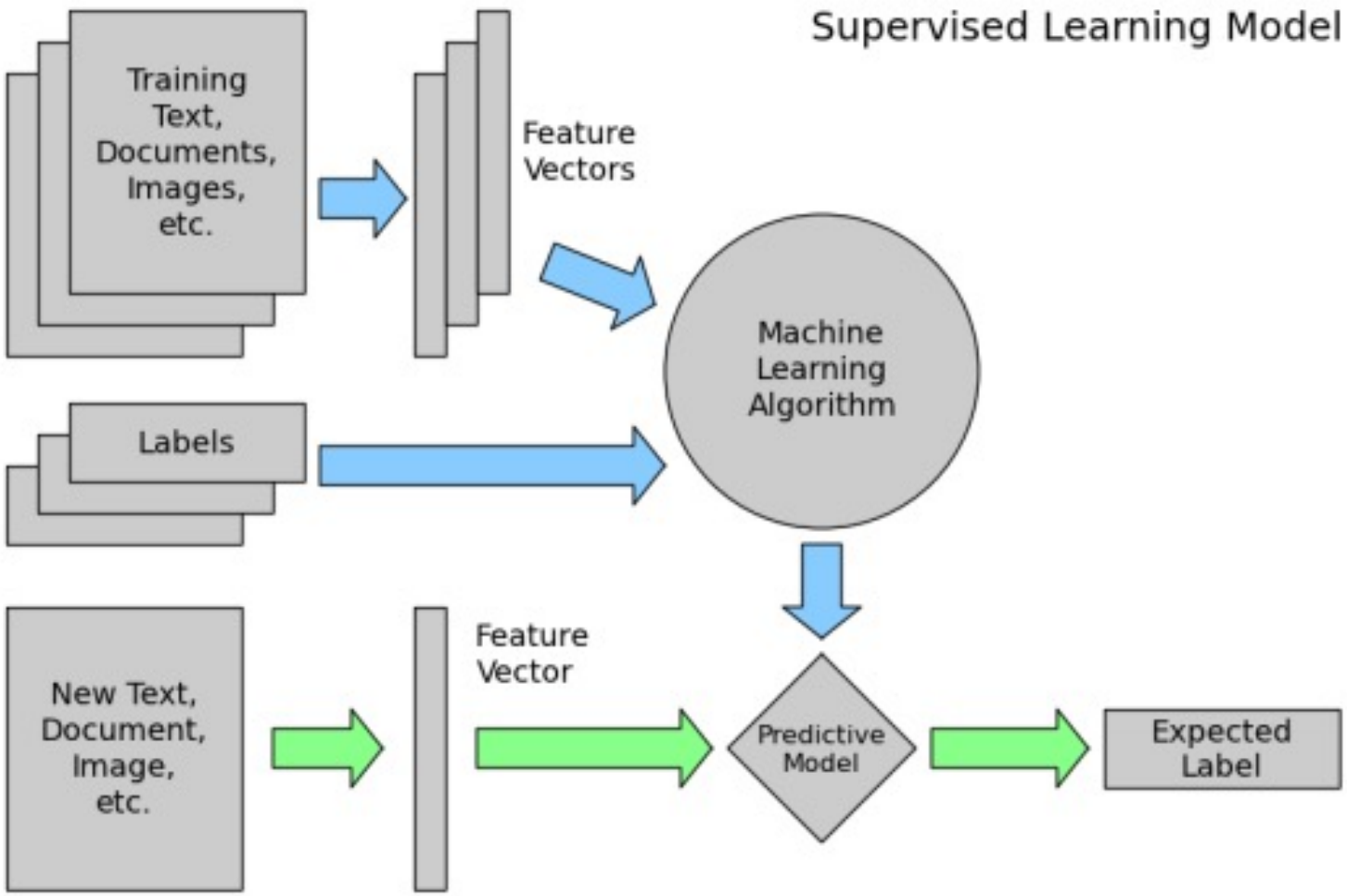
Unsupervised learning
Clustering

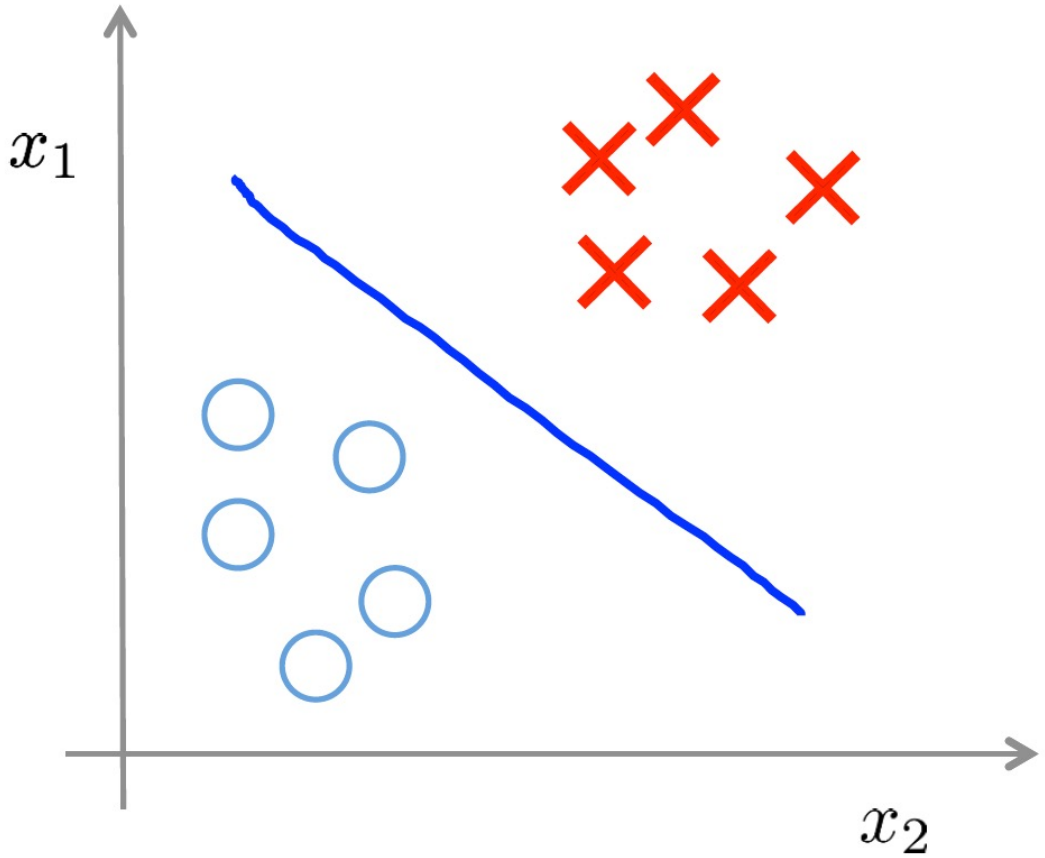
michel.riveill@univ-cotedazur.fr

Supervised learning



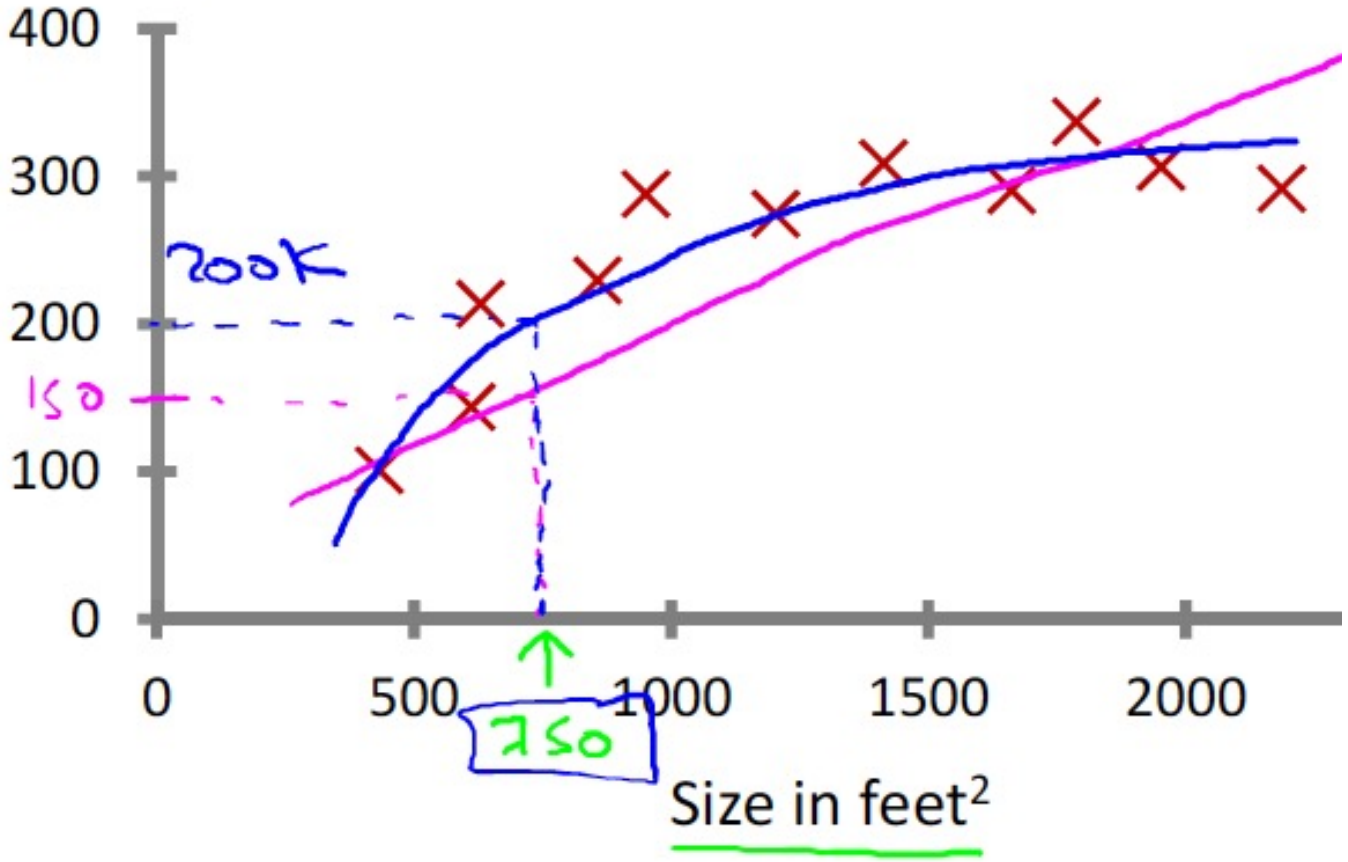
Supervised learning: classification

- Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots, (x^{(m)}, y^{(m)})\}$
- $x^{(i)}$ are featured samples
- $y^{(i)}$ are classes

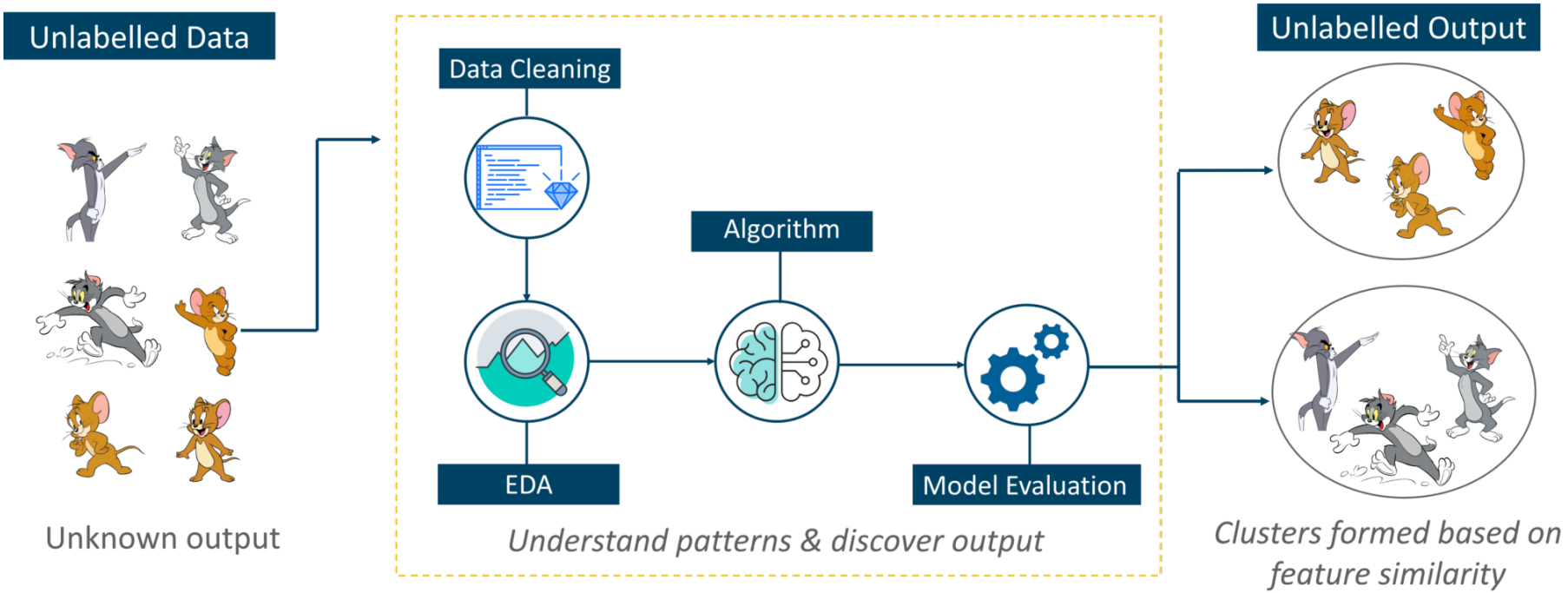


Supervised learning: regression

- Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots, (x^{(m)}, y^{(m)})\}$
- $x^{(i)}$ are featured samples
- $y^{(i)}$ are continuous value

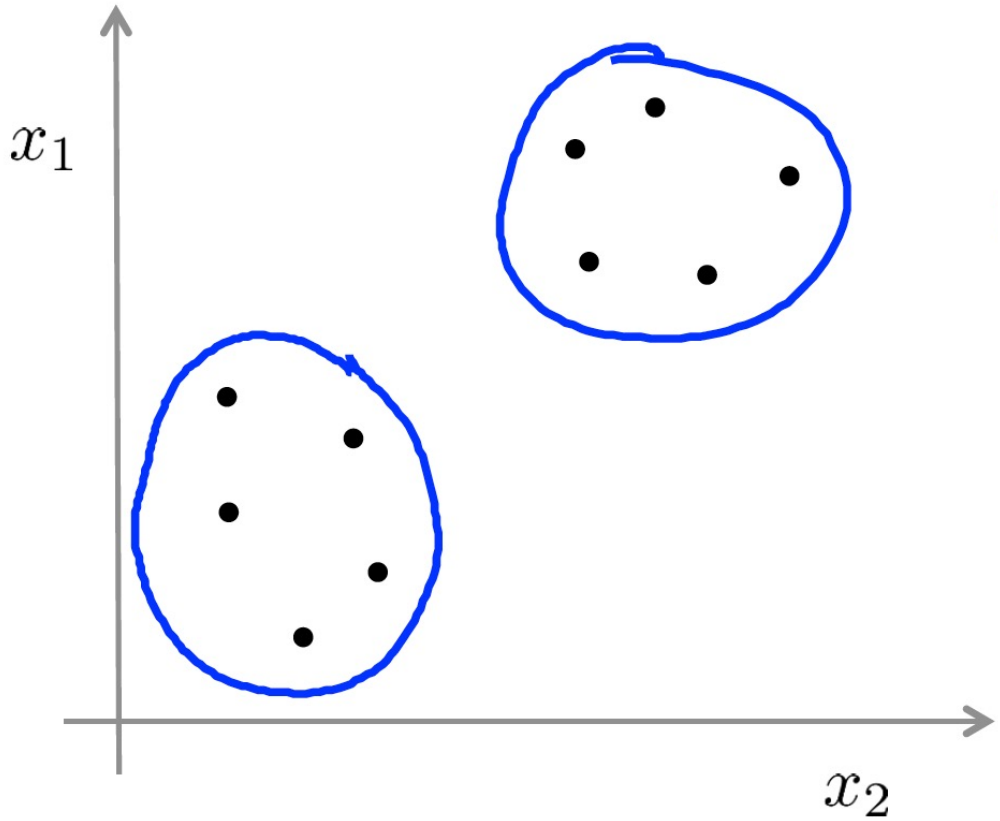


Unsupervised learning: classification



Unsupervised learning: clusterisation

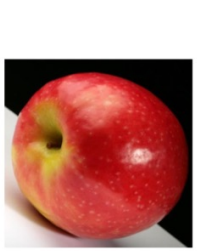
- Training set: $\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$
- $x^{(i)}$ are featured samples
- $y^{(i)}$ doesn't exist



Motivations

- Group similar items
- Detect outlier

Which cluster
Many possibility exists











Class by color






One color in each class



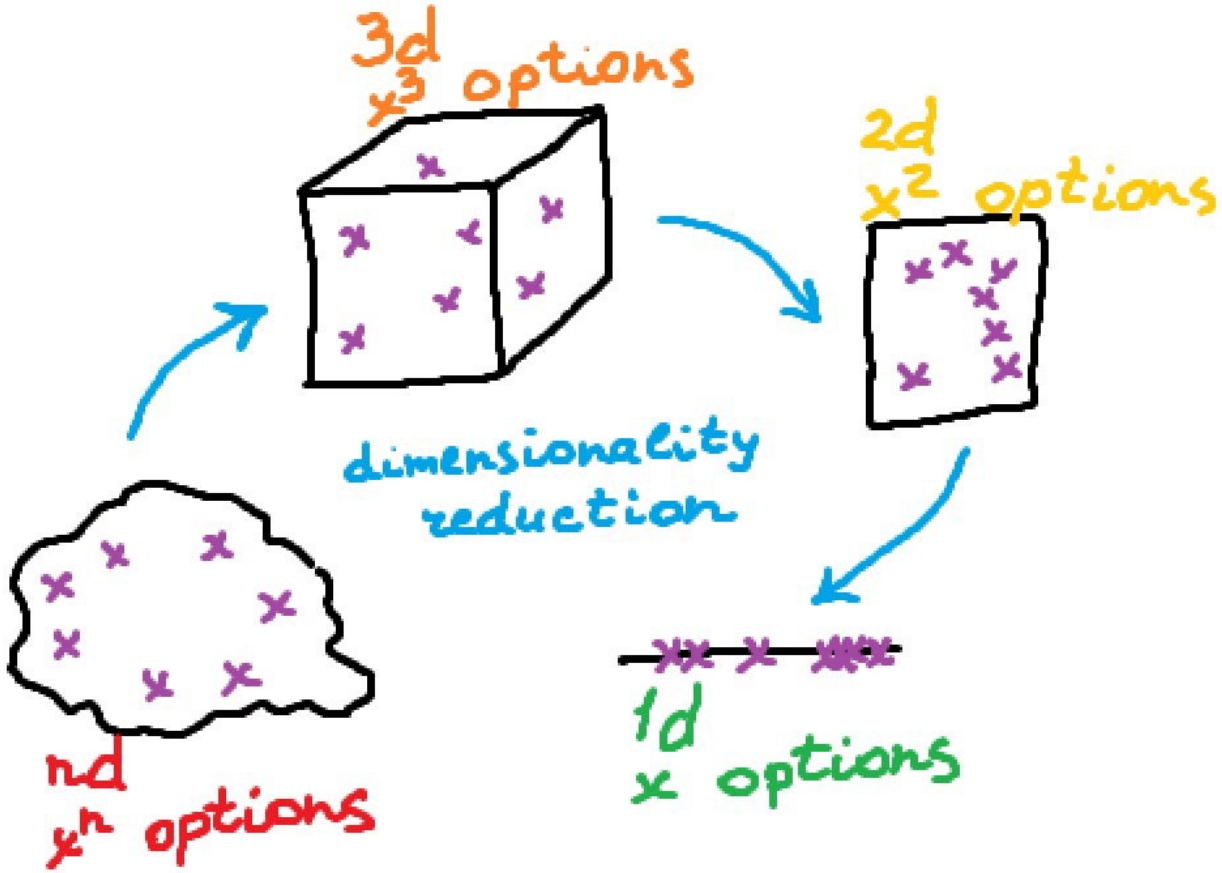



Class by form

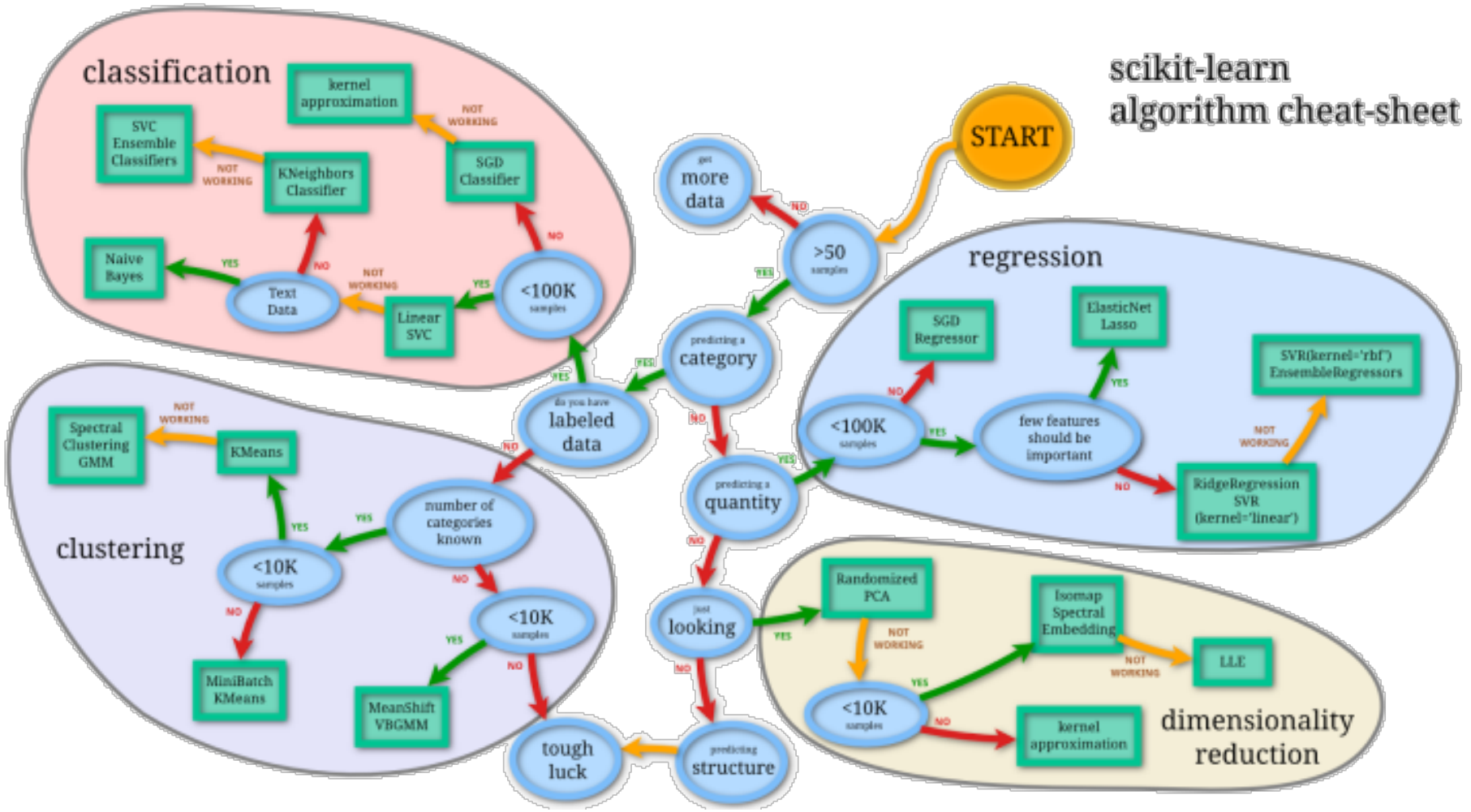
Unsupervised learning: reduction dimension

Motivations

- Reduce complexity
- Plot the data



The scikit learn models



Clustering

Clustering

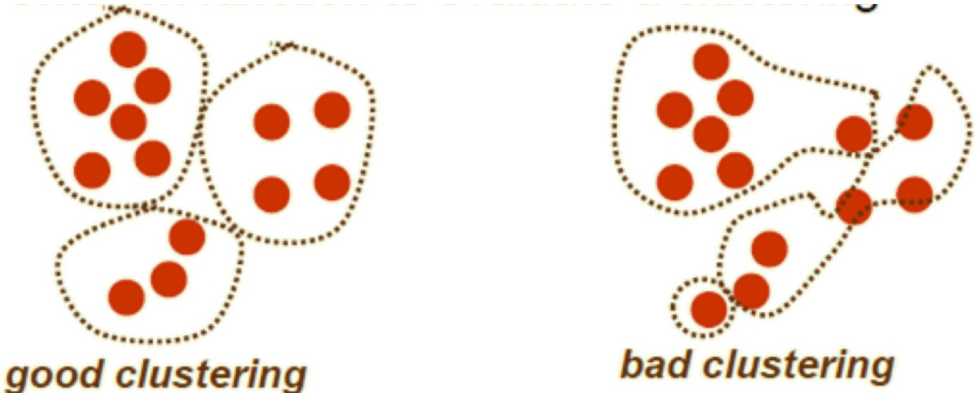
- The aim is now to find regularities, irregularities, relationships, similarities and associations in the input.

- What is Clustering?
  - Find K clusters (Bacher 1996)
    - the objects of one cluster are similar to each other
    - whereas objects of different clusters are dissimilar

- We can reach different goals:
  - Determine the intrinsic grouping in a set of unlabeled data.
  - There is no golden standard
  - Motivations
    - Group similar items
    - Detect outlier
  - Main algorithm: K-mean

What do we need for clustering

- Proximity measure to construct the cluster
  - Similarity measure – $s(x^i, x^j)$ – large if x^i and x^j are similar
  - Dissimilarity or distance – $d(x^i, x^j)$ – small if x^i and x^j are similar
- Criterion function to evaluate the result



- Algorithm to compute clustering

A first algorithm: K-means

A first algorithm: K-means in action

K-means: same dataset with another initialisation

A first algorithm: K-means

- Randomly initialize K cluster centroids: $\{\mu_1, \mu_2, \mu_3, \ldots, \mu_k\}$

```
do {
   # Step 1: Generate a new partition by assigning each pattern to its closest cluster centroids
   for i = 1 to m
      c(i) ← index (j from 1 to k) to the closest centroid of x(i)
      i.e. c(i) ← min_j d(x(i), μ_j)

   # Step 2: Compute new cluster centroids as the centroids of the clusters.
   for j = 1 to k
      μ_i = average of points assigned to cluster i

} stop when there is no change in the membership
   i.e. stop when cluster centroids remain the same
```

- Proximity measure → Euclidian distance
- Criterion function → distorsion = sum of squared distance to the closest centroid

16

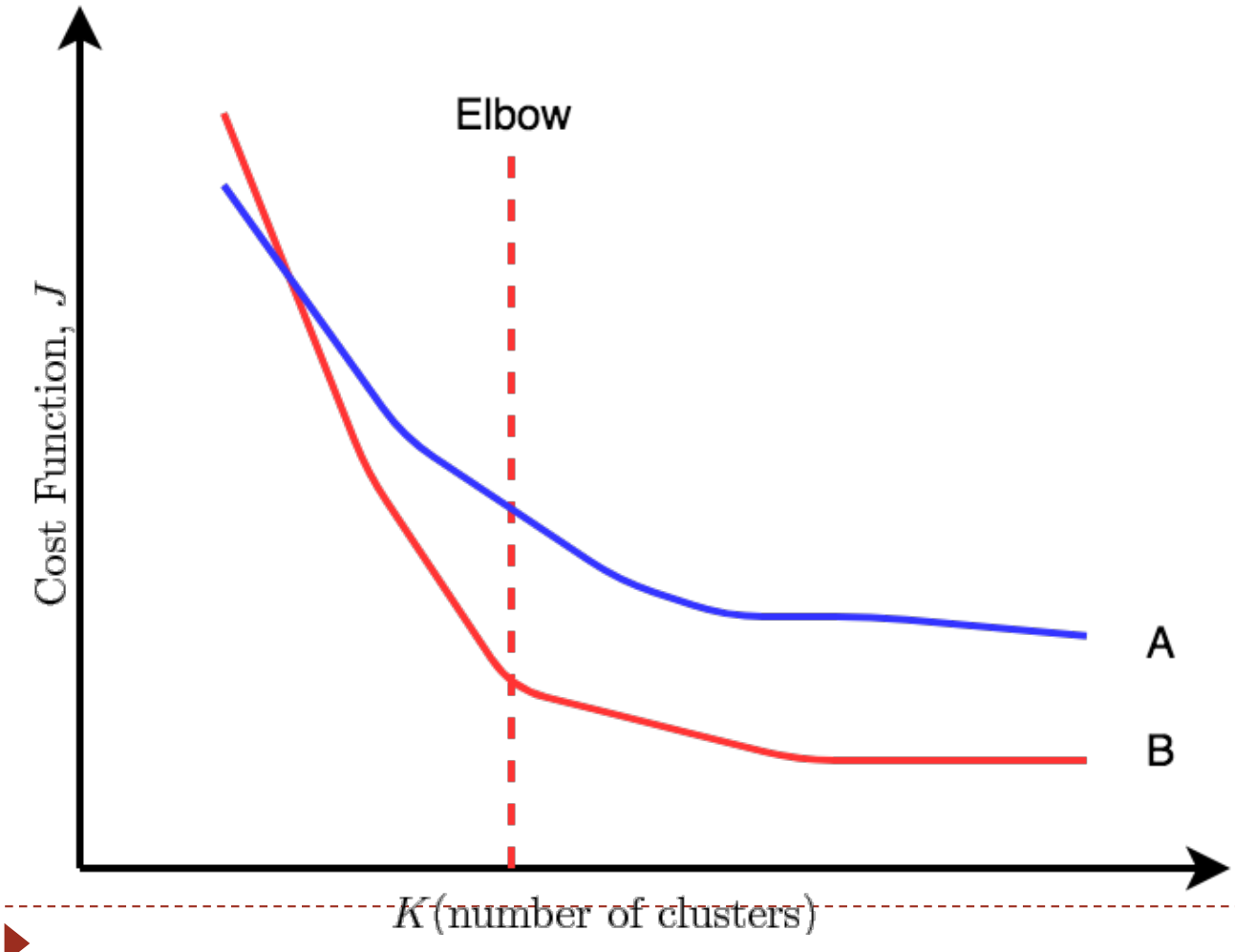
A first algorithm: K-means

- Distortion function (criterion function): J
  - $c^{(i)}$: index of cluster (1, 2, …, k) to which $x^{(i)}$ is currently assigned
  - μ_i : cluster centroid i
  - $\mu_c(i)$ = cluster centroid of cluster to which example $x^{(i)}$ has been assigned
  - $J(c^{(1)}, c^{(2)}, \ldots, c^{(m)}, \mu_1, \mu_2, \ldots, \mu_k) = 1/m * \Sigma_{i=1..m} d(x^{(i)} - \mu_c(i))2$
- How to choose the good clustering for k clusters
  - For I = 1 to N {
    - Apply K-means for k clusters
    - Compute distortion function
    - }
  - Choose the clustering that gave the lowest distortion cost
  - Random initialization → different clustering

A first algorithm: K-means

- How to chose the right value of K ?
- Elbow method
  1. Compute clustering algorithm (e.g., k-means clustering) for different values of k.
     - For instance, by varying k from 1 to 10 clusters.
  2. For each k, compute the distortion function
  3. Plot the curve of c distortion function to the number of clusters k.
  4. The location of a bend (elbow) in the plot is generally considered as an indicator of the appropriate number of clusters.
- Average silhouette method
  - Compute clustering algorithm (e.g., k-means clustering) for different values of k. For instance, by varying k from 1 to 10 clusters.
  - For each k, calculate the average silhouette of observations (*avg.sil*).
  - Plot the curve of *avg.sil* according to the number of clusters k.
  - The location of the maximum is considered as the appropriate number of clusters.

Elbow method



Silhouette method – calculate silhouette

- The silhouette analysis measures how well an observation is clustered and it estimates the **average distance between clusters**

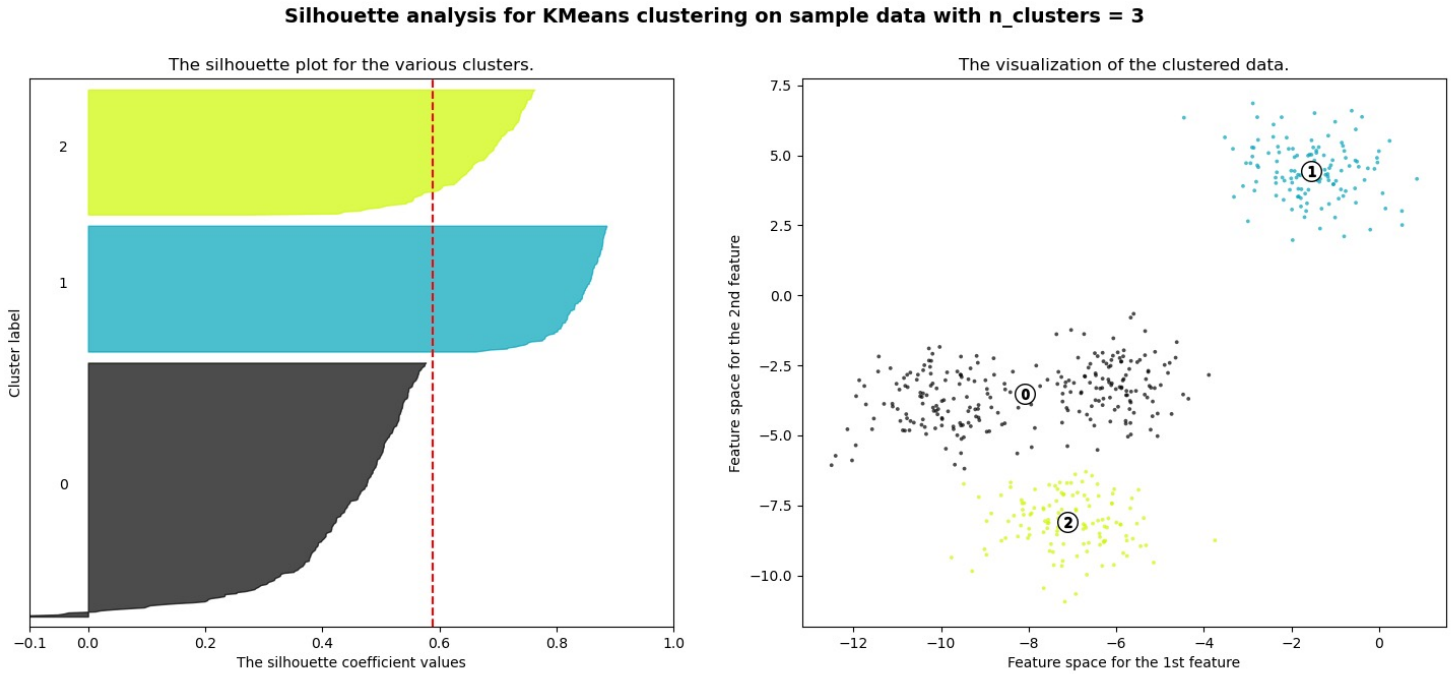
- The Silhouette Coefficient is defined for each sample and is composed of two scores:
  - a: The mean distance between a sample and all other points in the same class.
  - b: The mean distance between a sample and all other points in the next nearest cluster.
  - The Silhouette Coefficient for a single sample is then given as:
    - s = b-a / max(a, b)

- Score bounded between [-1, 1]
  - -1 incorrect clustering
  - +1 highly dense clustering
  - 0 indicates overlapping clusters.

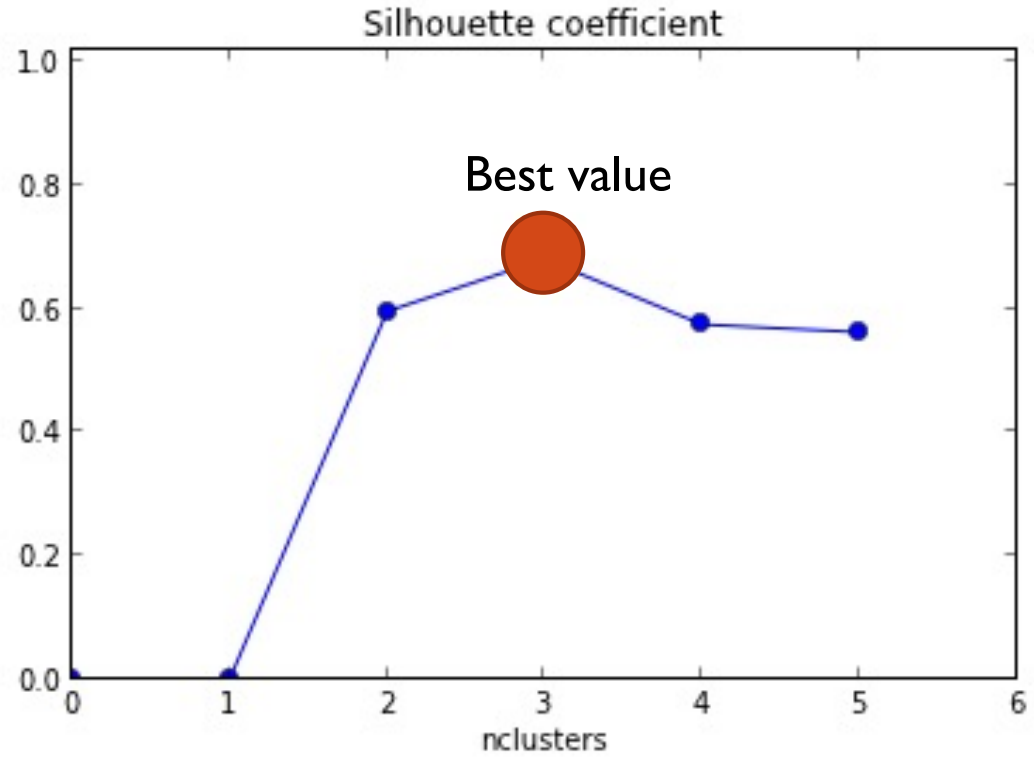
Silhouette interpretation

Score bounded between [-1, 1]

- -1 incorrect clustering
- +1 highly dense clustering
- 0 indicates overlapping clusters.



Silhouette interpretation



K-means in Python

```
from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=i).fit(X)

# get the cluster
labels = kmeans.predict(X)
# get the centroid
C = kmeans.cluster_centers_
# Evaluate the cluster :
I = kmeans.inertia_          # Sum of squared distances of
                             # samples to their centroid.
```

Silhouette Coefficient in Python

- `from sklearn.metrics import silhouette_score`

- ```
  metrics.silhouette_score(X,
                           labels,
                           metric='euclidean')
  ```

- Returns
  - **Silhouette:** *float*
  - Mean Silhouette Coefficient for all samples.

- What is "labels" ?
  ```

K-means - drawback

- Disadvantage 1:
  - An initial choice of different centroids may lead k-means to produce different final groupings, each corresponding to a different local minimum.

- Strategies for dealing with drawback 1:
  - Multiple run methods:
    - Create different initial choices of centroids, run the k-means for each, and select the best result.

K-means - drawback

- Disadvantage 2:
  - Knowledge of the number of clusters K is required a priori.

- Strategies to deal with disadvantage 2:
  - Run a sequential algorithm many times for different numbers m of clusters
    - Solution 1
      - Compute the cost function J
      - Plot J as a function of the number of clusters
      - And set K to the resulting elbow (Elbow method).
    - Solution 2
      - Compute average silhouette
      - Plot ave.sil as a function of the number of clusters
      - And set K to the max

K-means - drawback

- Disadvantage 3:
  - k-means is sensitive to outliers and noise.
- Strategies to deal with disadvantage 3:
  - Discard all "small" clusters (they are likely to be formed by outliers).
  - Use a k-medoids algorithm, where a cluster is represented by one of its points.
- Disadvantage 4:
  - k-means are not suitable for data with nominal (categorical) coordinates.
- Strategies for dealing with drawback 4:
  - Use a k-medoids algorithm.

K-medoids algorithms

- Each cluster is represented by a vector selected among the elements of X (medoid).
- A cluster contains
  - Its medoid
  - All vectors in X that
    - Are not used as medoids in other clusters
    - Lie closer to its medoid than the medoids representing other clusters.
- Obtaining the set of medoids Θ that best represents the data set, X is equivalent to minimizing the following cost function
  - $J(\Theta)=\sum_{i\in\Theta}\sum_{j\in\bar{\Theta}} u_{ij} d(x_i\ x_j)\ \ avec\ uij = \begin{cases} 1, if\ d(x_i\ x_j) = min_{q\in\Theta} d(x_i\ x_q) \\ 0, otherwise \end{cases}$

K-medoids algorithms

- Close to k-means algorithms

Choose arbitrarily k medoids

Repeat

assign each remaining object to the nearest medoid

Randomly choose a non-medoid O_r

For each medoid O_j

Compute the cost J of replacing O_j with O_r

(J = $J_{O_r} - J_{O_j}$ difference of cost function)

If J < 0 then

Replace O_j by O_r

Compute the new clusters

Until there are no more changes

K-medoids: Exemple

- Let A={1,3,4,5,8,9}, k=2 and M={1,8} be the set of medoids
  - C1={1,3,4} and C2={5,8,9}
  - $J_{\{1,8\}}=dist(3,1)^2+dist(4,1)^2+dist(5,8)^2+dist(9,8)^2=26$
- Try to swap 1 and 3 → M={3,8} → C1={1,3,4,5} and C2={8,9}
  - $J_{\{3,8\}}=dist(1,3)^2+dist(4,3)^2+dist(5,3)^2+dist(9,8)^2=10$
  - $J_{\{3,8\}} - J_{\{1,8\}}= -29$ <0 so the replacement is done.
- Try to swap 3 and 4 → M={4,8} → C1 and C2 unchanged
  - $J_{\{4,8\}}=dist(1,4)^2+dist(3,4)^2+dist(5,4)^2+dist(8,9)^2= 12$
  - $J_{\{4,8\}} - J_{\{3,8\}} = 2 >0$ so 3 is not replaced by 4
- Try to swap 3 and 5 → M={5,8} → C1 and C2 unchanged
  - $J_{\{5,8\}} > J_{\{3,8\}}$

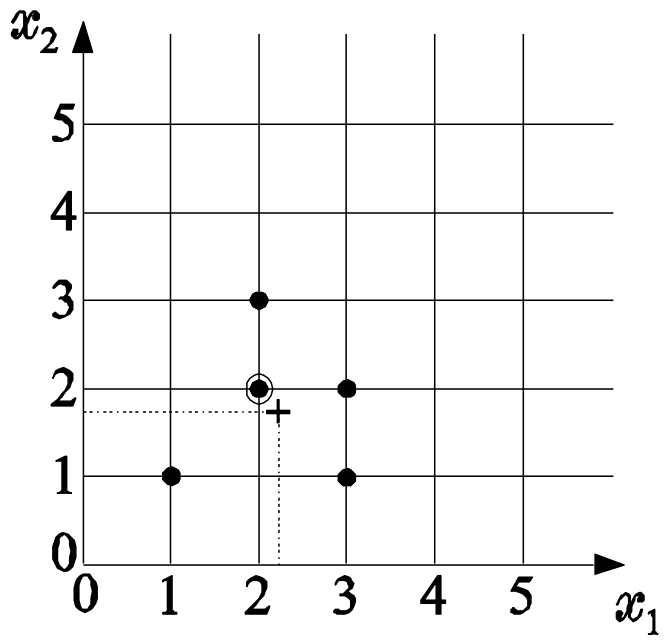
K-means vs K-medoids

K-means	K-medoids
1. Suited only for continuous domains	1. Suited for either cont. or discrete domains
2. Algorithms using means are sensitive to outliers	2. Algorithms using medoids are less sensitive to outliers
3. The mean possess a clear geometrical and statistical meaning	3. The medoid has not a clear geometrical meaning
4. Algorithms using means are not computationally demanding	4. Algorithms using medoids are more computationally demanding

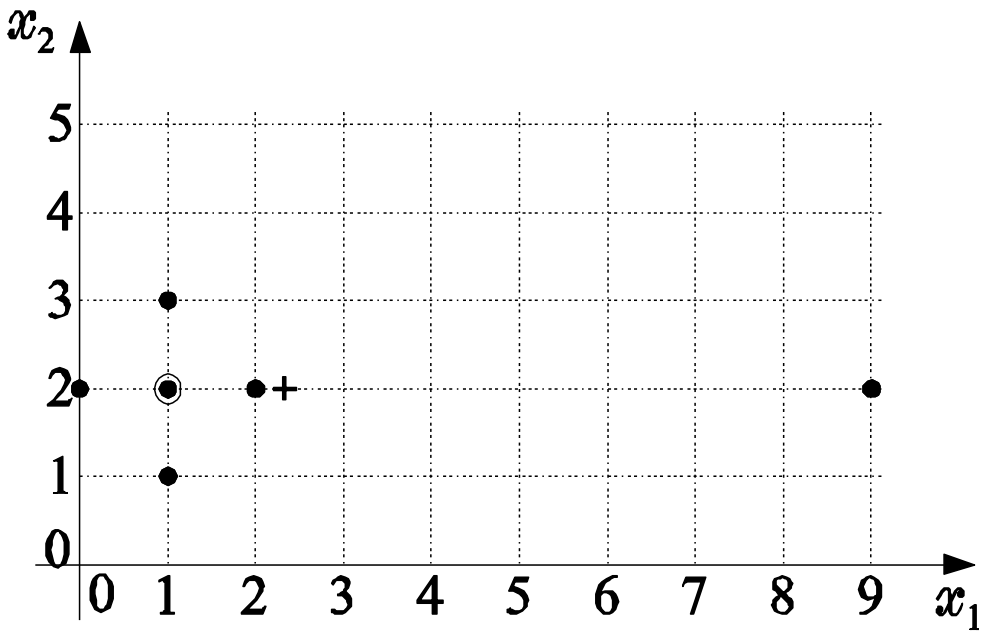
K-means vs K-medoids

- In pictures a and b, medoid is the circled point and mean is the "+" point

- In a) the mean does not belong to the initial data $D = \{1, 2, 3, 4, \ldots\}$.
- In b) the point (9, 2) can be considered as an outlier
  - While the outlier affects significantly the mean of the set, it does not affect its medoid.



(a)



(b)

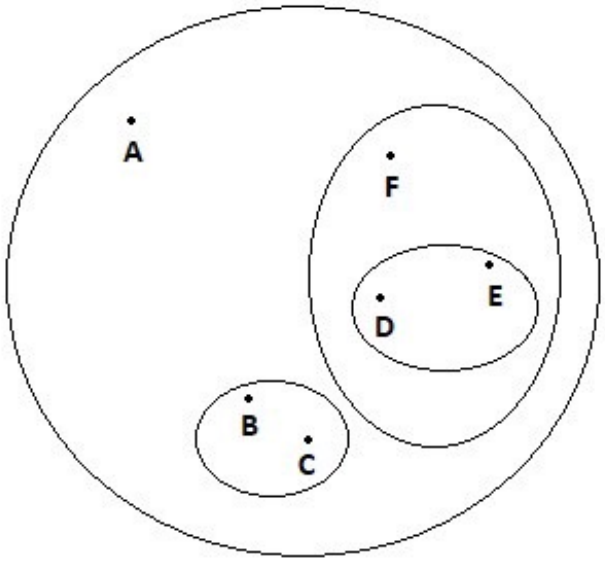
Hierarchical clustering (or agglomerative)

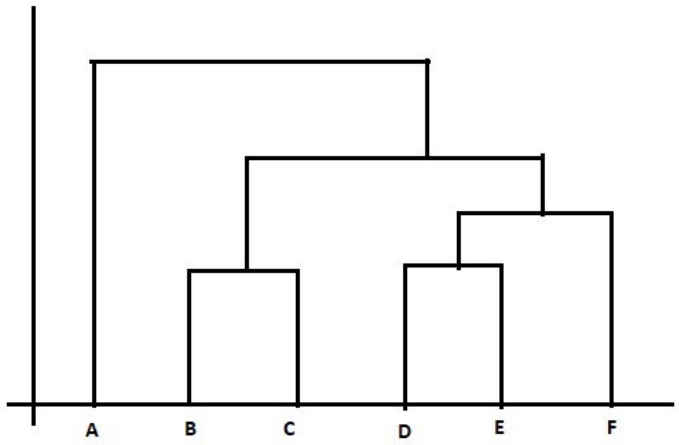
Hierarchical clustering

- Hierarchical clustering is one of the popular and easy to understand clustering technique. This clustering technique is divided into two types:
  - Agglomerative
    - Initially, each data point is considered as an individual cluster.
    - At each iteration, the similar clusters merge with other clusters until one cluster or K clusters are formed.
  - Divisive
    - Initially, all the data points as a single cluster
    - At each iteration, we separate the data points from the cluster which are not similar. Each data point which is separated is considered as an individual cluster.
    - In the end, we'll be left with n clusters.
  - **Divisive Hierarchical clustering** is exactly the opposite of the **Agglomerative Hierarchical clustering.**

Agglomerative hierarchical clustering

- The basic algorithm of Agglomerative is straight forward.
  - Compute the proximity matrix
  - Let each data point be a cluster
  - Repeat: Merge the two closest clusters and update the proximity matrix
  - Until only a single cluster remains
- It can be visualized by a dendogram





How to perform Hierarchical Clustering

- Suppose a teacher wants to divide her students into different groups.
- He has the marks scored by each student in an assignment and based on these marks, he wants to segment them into groups.

Student_ID	Marks
1	10
2	7
3	28
4	20
5	35

How to perform Hierarchical Clustering

- Step 1: Creating a Proximity Matrix
  - The diagonal elements of this matrix will always be 0 as the distance of a point with itself is always 0.
  - We will use the Euclidean distance formula to calculate the rest of the distances.
    - $D(S_1, S_2) = \sqrt{(10-7)^2} = 3$

Student_ID	Marks
1	10
2	7
3	28
4	20
5	35

ID	1	2	3	4	5
1	0	3	18	10	25
2	3	0	21	13	28
3	18	21	0	8	7
4	10	13	8	0	15
5	25	28	7	15	0

How to perform Hierarchical Clustering

- Step 2: Assign all the points to an individual cluster

- **Step 3:** look at the smallest distance in the proximity matrix and merge the points with the smallest distance
  - Here, the smallest distance is 3 and hence we will merge point 1 and 2:

ID	1	2	3	4	5
1	0	3	18	10	25
2	3	0	21	13	28
3	18	21	0	8	7
4	10	13	8	0	15
5	25	28	7	15	0

How to perform Hierarchical Clustering

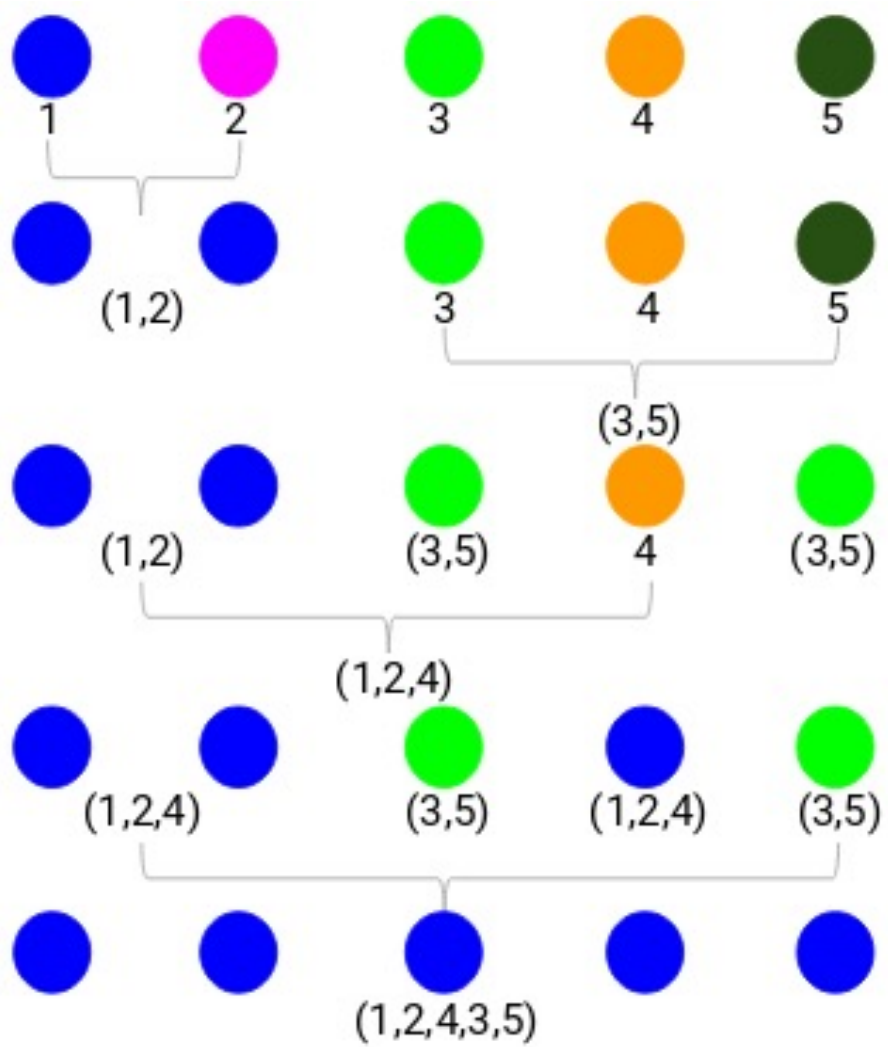
- Step 4: update cluster and accordingly update the proximity matrix
  - Here, we have taken the maximum of the two marks (7, 10) to replace the marks for this cluster.
  - Instead of the maximum, we can also take the minimum value or the average values as well.

Student_ID	Marks
(1,2)	10
3	28
4	20
5	35

ID	(1,2)	3	4	5
(1,2)	0	18	10	25
3	18	0	8	7
4	10	8	0	15
5	25	7	15	0

How to perform Hierarchical Clustering

- Repeat the process until to obtain only one cluster

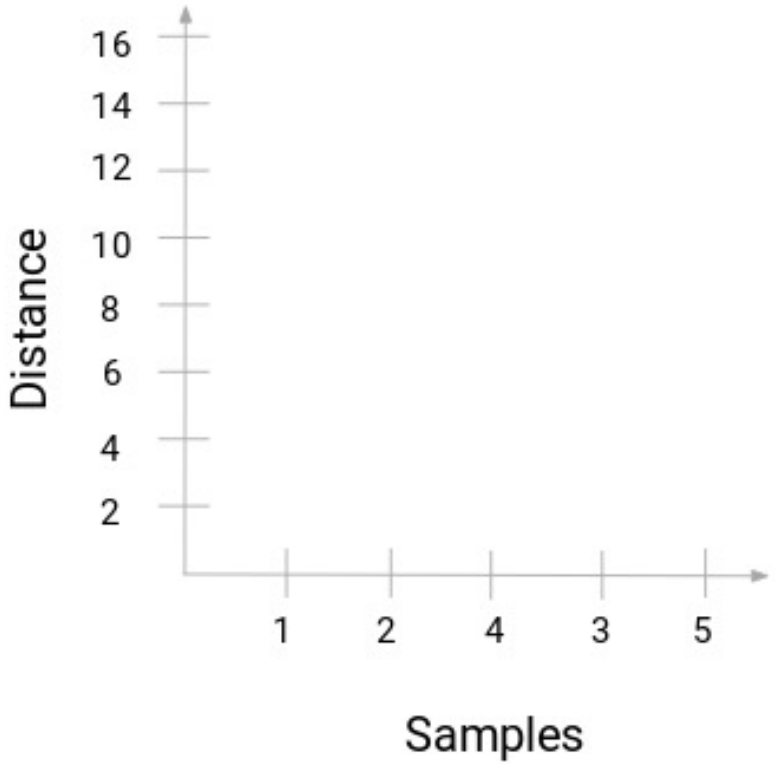


How should we Choose the Number of Clusters in Hierarchical Clustering?

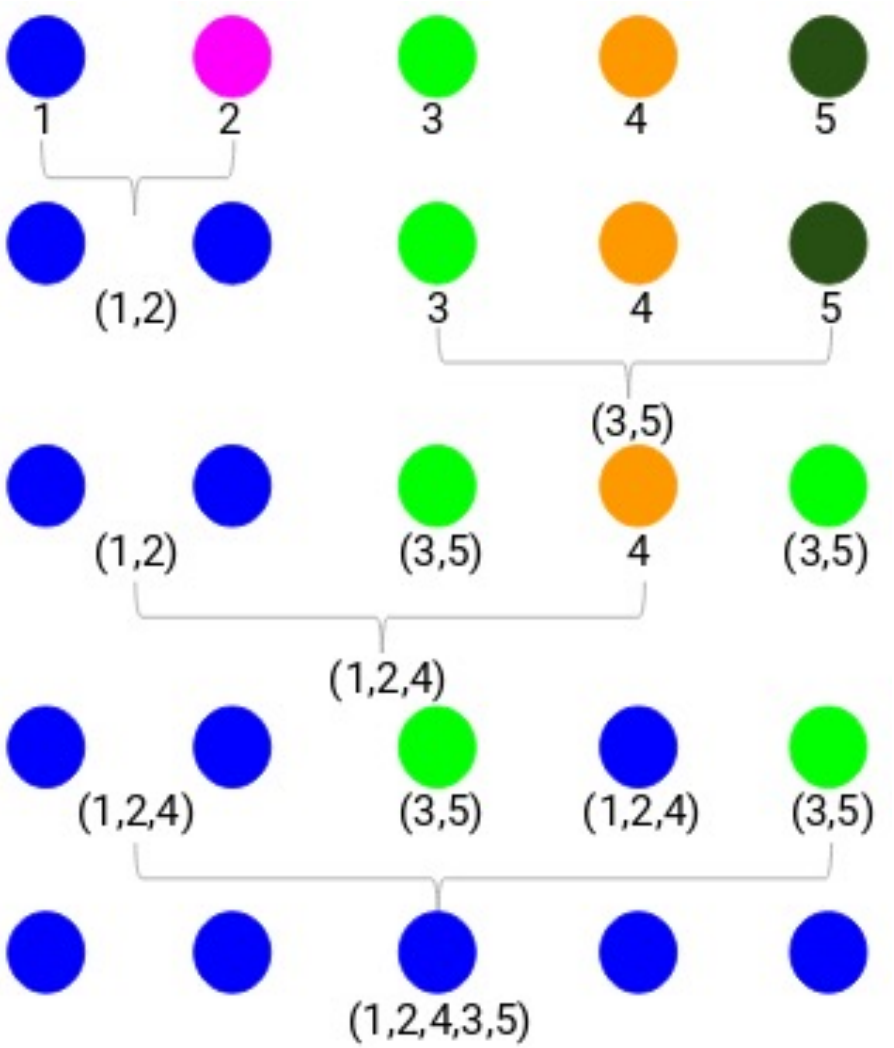
- To get the number of clusters for hierarchical clustering, we make use of an usefull concept called a Dendrogram.
  - It is a tree-like diagram that records the sequences of merges or splits.

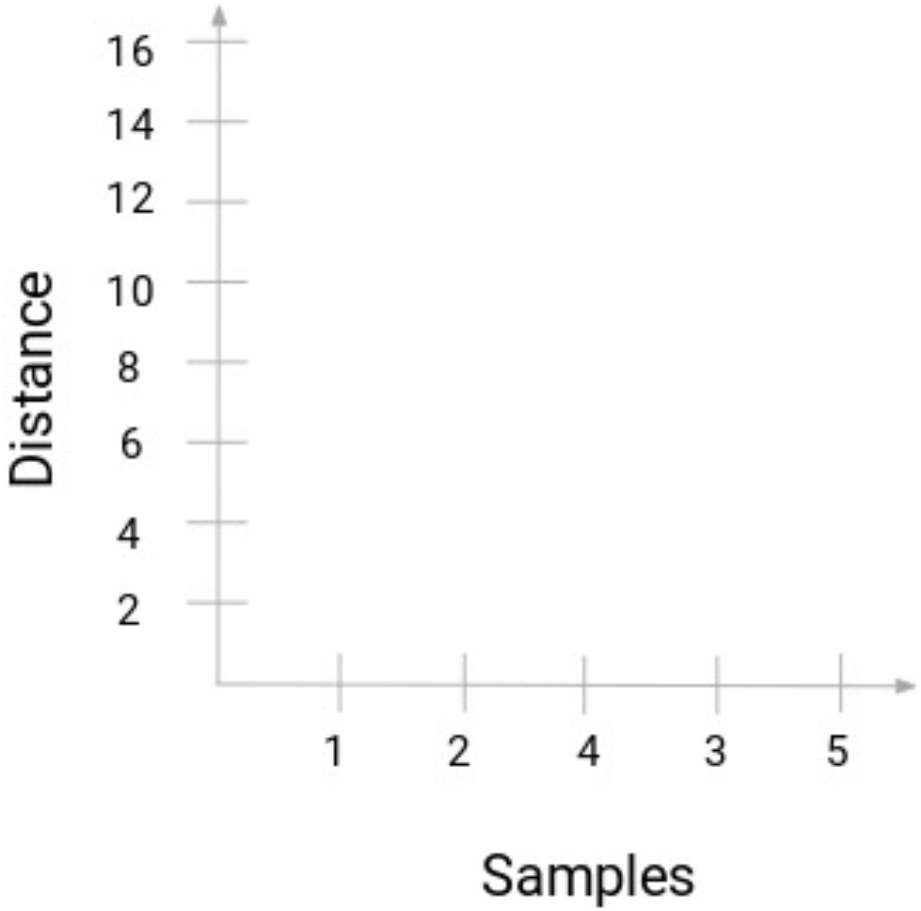
- X-axis: samples of the dataset
- Y-axis: distance between cluster

- When two clusters are merged
  - We will join them in the dendrogram
  - The height of the join will be the distance between these points.

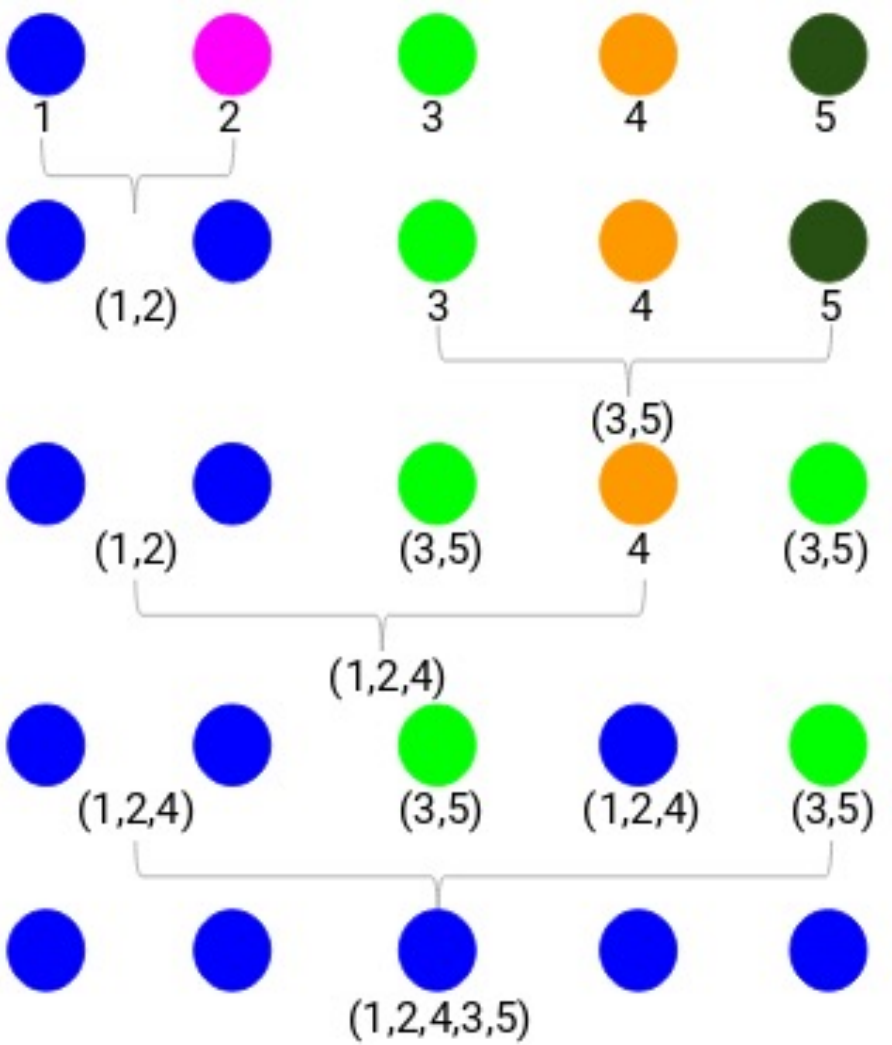


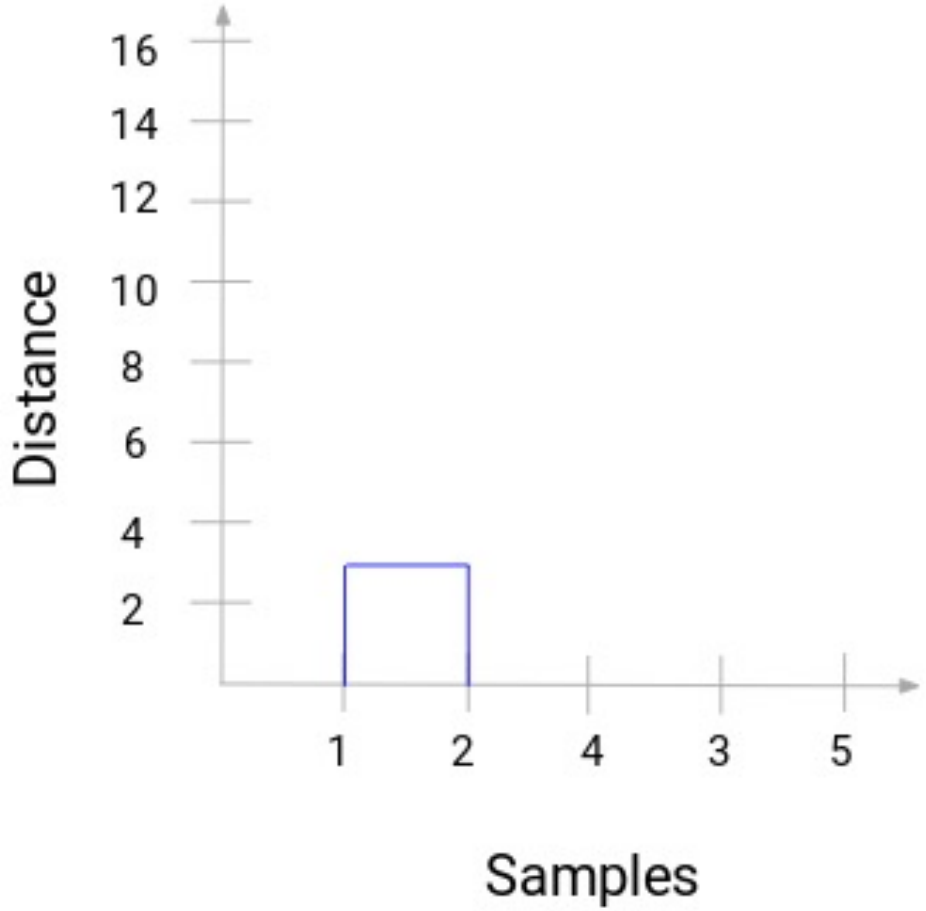
Dendogram representation



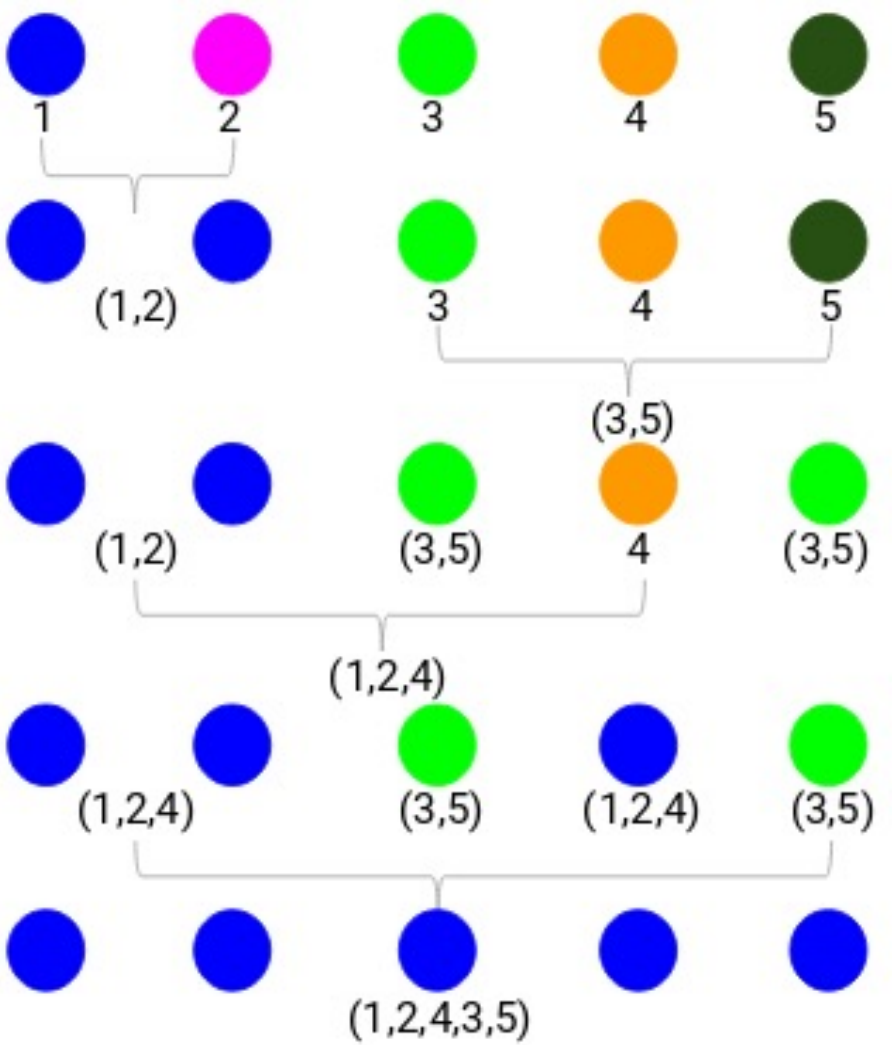


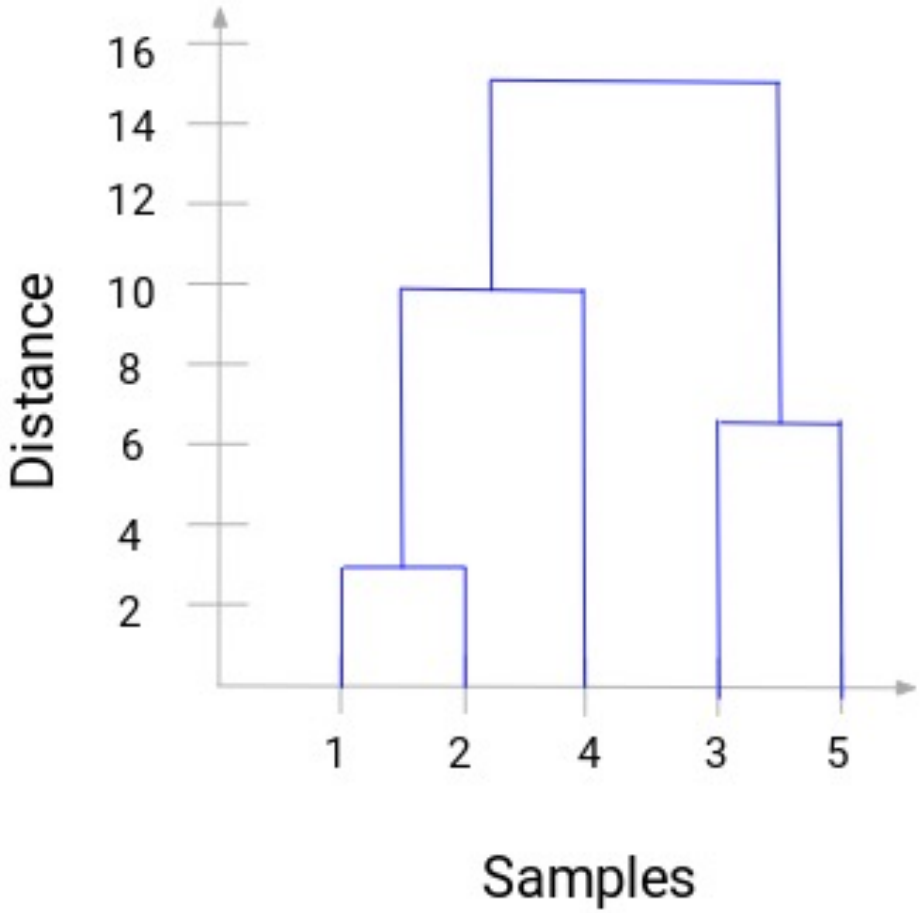
Dendogram representation



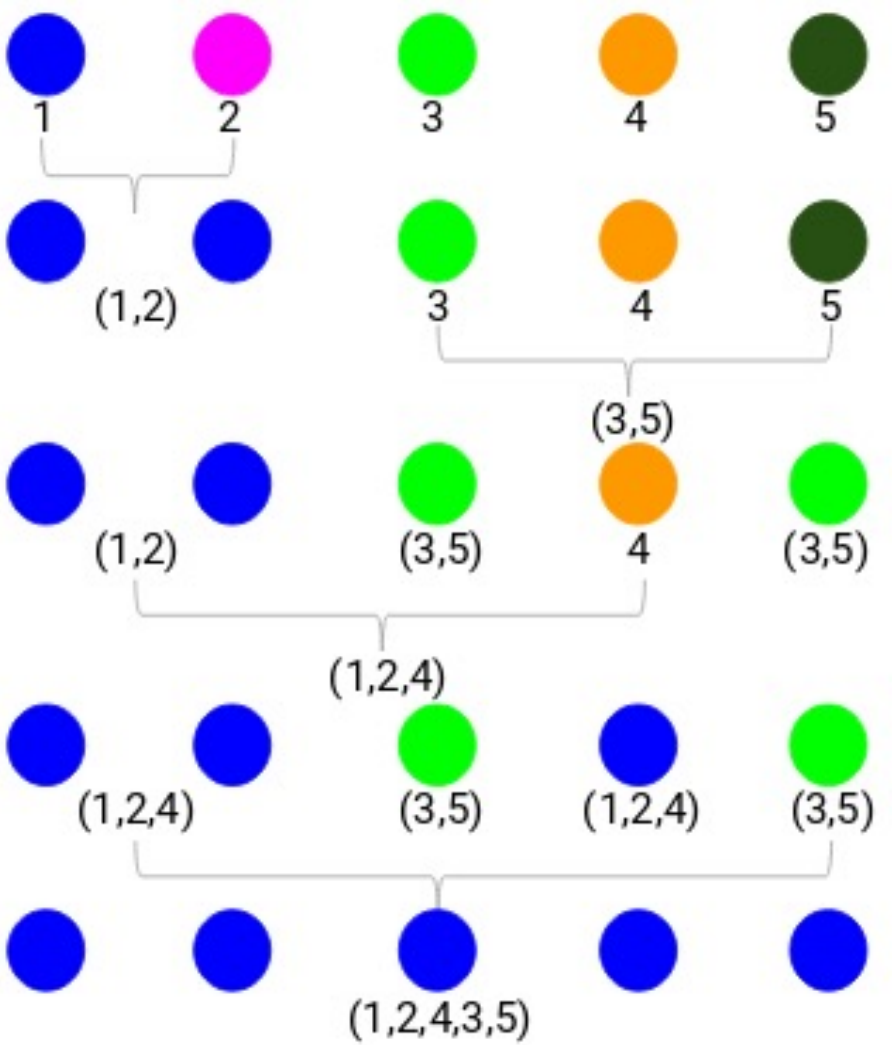


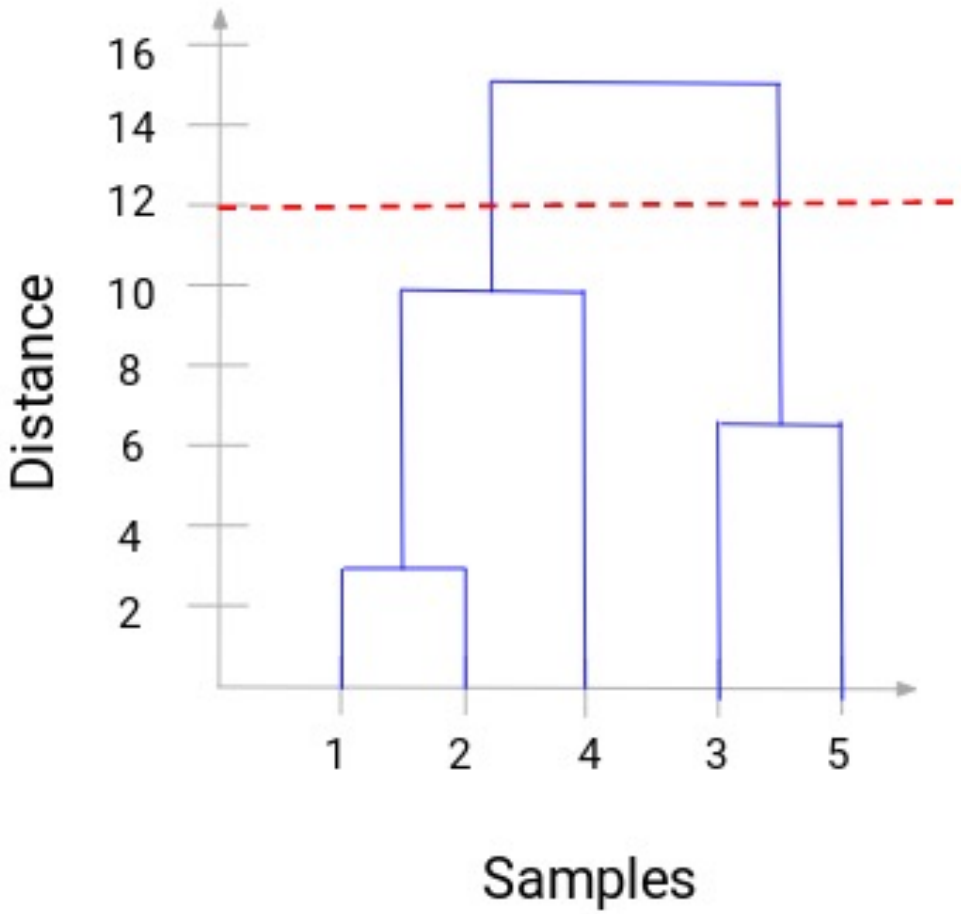
Dendogram representation





Dendogram representation

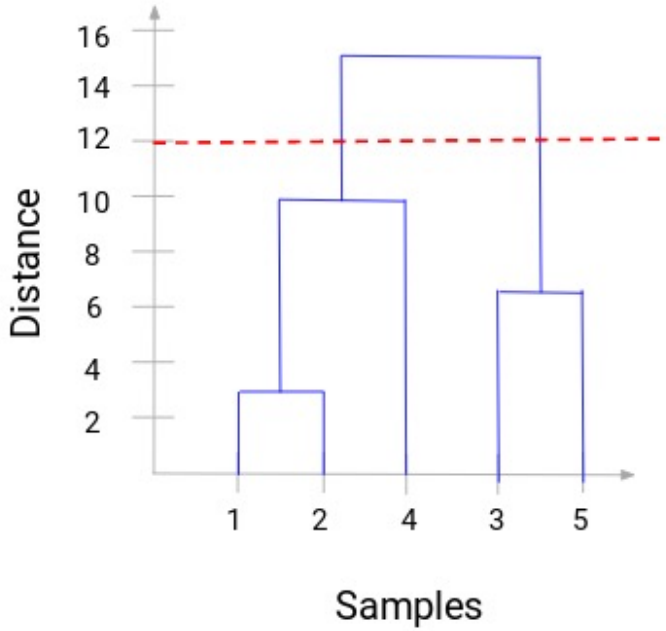




Dendogram representation

- How should we Choose the Number of Clusters in Hierarchical Clustering?
  - We can set a threshold distance and draw a horizontal line
    - *Generally, we try to set the threshold in such a way that it cuts the highest vertical line*).
    - Let's set this threshold as 12 and draw a horizontal line

The number of clusters will be the number of vertical lines which are being intersected by the line drawn using the threshold.



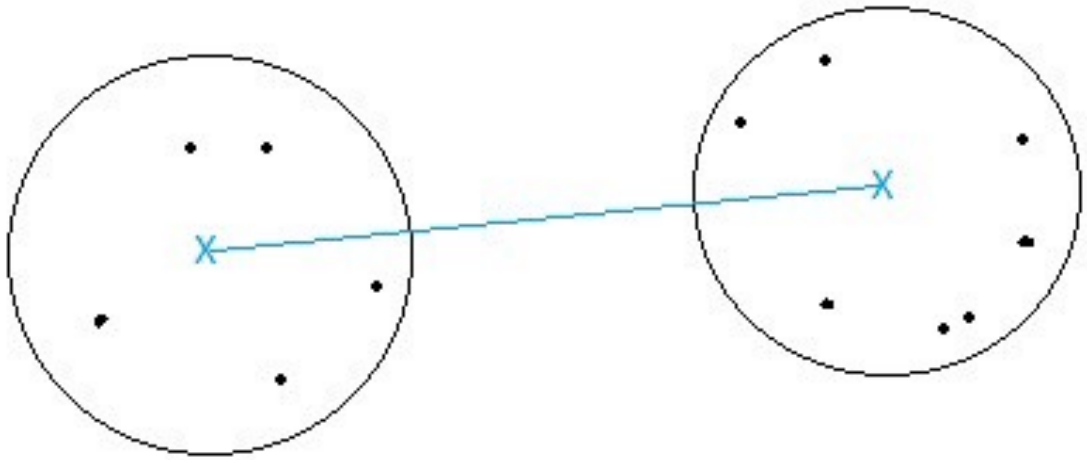
Hierarchical clustering

- Calculating the similarity between two clusters is important to merge or divide the clusters.

- Similarity approaches:
  - Distance Between Centroids
  - MIN
  - MAX
  - Group Average
  - Ward's Method

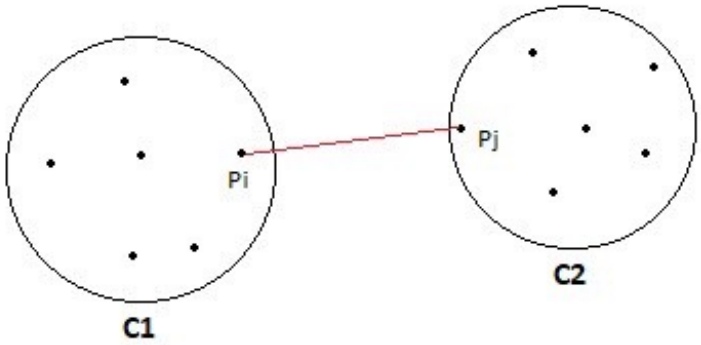
Distance between centroids

- Distance between centroids: Compute the centroids of two clusters C1 & C2 and take the similarity between the two centroids as the similarity between two clusters.
- This is a less popular technique in the real world.



MIN

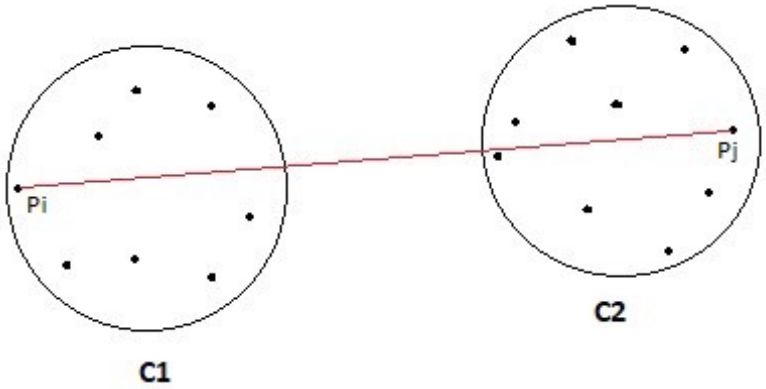
- Also known as single-linkage algorithm
- Sim(C1,C2) = Min Sim(P_i,P_j) such that $P_i \in C1$ & $P_j \in C2$
  - In simple words, pick the two closest points such that one point lies in cluster one and the other point lies in cluster 2 and takes their similarity and declares it as the similarity between two clusters.



- **Pros:** MIN approach can separate non-elliptical shapes as long as the gap between the two clusters is not small.
- **Cons:** MIN approach cannot separate clusters properly if there is noise between clusters.

MAX

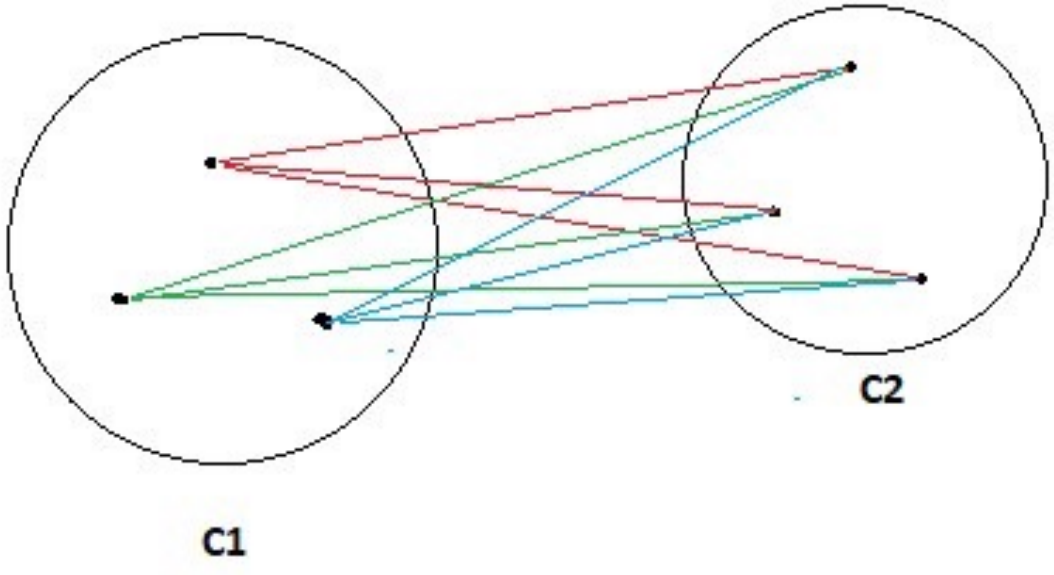
- It's the opposite of MIN
- Also known as complete-linkage algorithm
- Sim(C1,C2) = Max Sim(P_i,P_j) such that $P_i \in C1$ & $P_j \in C2$
  - In simple words, pick the two farthest points such that one point lies in cluster one and the other point lies in cluster 2 and takes their similarity and declares it as the similarity between two clusters.



- **Pros:** MAX approach does well in separating clusters if there is noise between clusters.
- **Cons:** Max approach is biased towards globular clusters and tends to break large clusters.

Group average

- Take all the pairs of points and compute their similarities and calculate the average of the similarities.
- $\text{sim}(C_1, C_2) = \frac{\sum sim(P_i, P_j)}{|C1| * |C2|}$ where, $P_i \in C_1 \& \ P_j \in C_2$



- **Pros:** The group Average approach does well in separating clusters if there is noise between clusters.
- **Cons:** The group Average approach is biased towards globular clusters.

Ward's Method

- Exactly the same as Group Average
- Replace similarity by the square of the distance
- $\text{sim}(C_1, C_2) = \frac{\sum dist(P_i, P_j)^2}{|C1| * |C2|}$ where, $P_i \in C_1 \& \ P_j \in C_2$

- **Pros & Cons same as group average**
  - **Pros:** Ward's method approach also does well in separating clusters if there is noise between clusters.
  - **Cons:** Ward's method approach is also biased towards globular clusters.

Space and Time Complexity

- **Space complexity**
  - The space required for the Hierarchical clustering Technique is very high when the number of data points are high as we need to store the similarity matrix in the RAM. The space complexity is the order of the square of n.
  - Space complexity = $O(n^2)$ where n is the number of data points.
- **Time complexity:**
  - Since we've to perform n iterations and in each iteration, we need to update the similarity matrix and restore the matrix, the time complexity is also very high. The time complexity is the order of the cube of n.
  - Time complexity = $O(n^3)$ where n is the number of data points.

Limitations

- There is no mathematical objective for Hierarchical clustering.
- All the approaches to calculate the similarity between clusters has its own disadvantages.
- High space and time complexity for Hierarchical clustering. Hence this clustering algorithm cannot be used when we have huge data.

Hierarchical clustering in sklearn

- Use : AgglomerativeClustering with support Ward, single (MIN), average, and complete (MAX) linkage strategies.

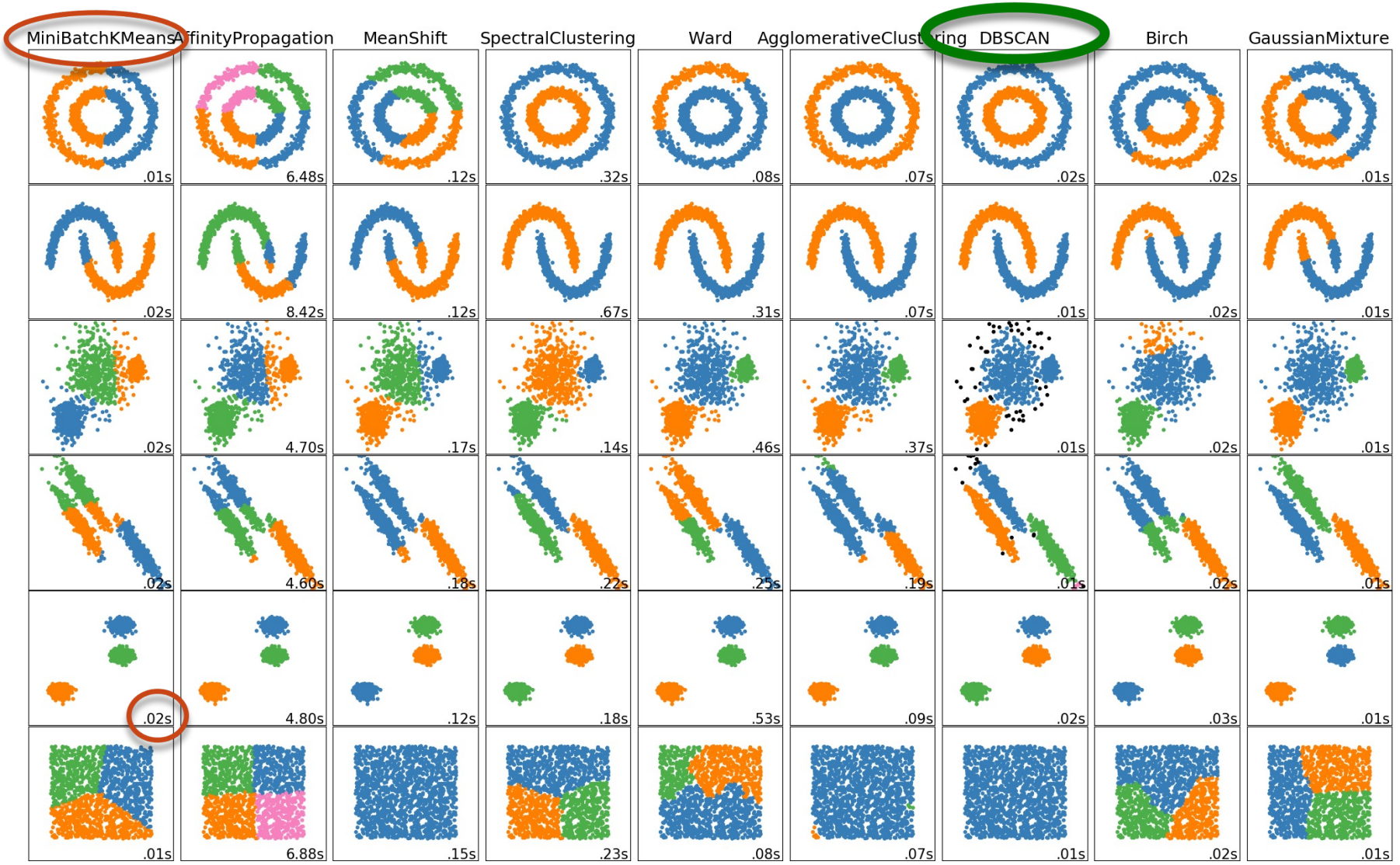
- Main parameters
  - **distance_threshold=None → Computer the full tree**
    - N_clusters=None
    - compute_full_tree=None
  - **affinity={**'euclidian', 'manhattan', 'cosine', …}
    - the distance use for similarity
  - **Linkage**={*'ward', 'complete', 'average', 'single'*}
    - 'ward' minimizes the variance of the clusters being merged.
    - 'average' uses the average of the distances of each observation of the two sets.
    - 'complete' uses the maximum of distances between all observations of the two sets.
    - 'single' uses the minimum of the distances between all observations of the two sets.

Visualization of cluster hierarchy¶

- It's possible to visualize the tree representing the hierarchical merging of clusters as a dendrogram.
  - https://scikit-learn.org/stable/auto_examples/cluster/plot_agglomerative_dendrogram.html

- But directly implemented in scipy or in seaborn

Others algorithms (try by yourself)



From: https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html

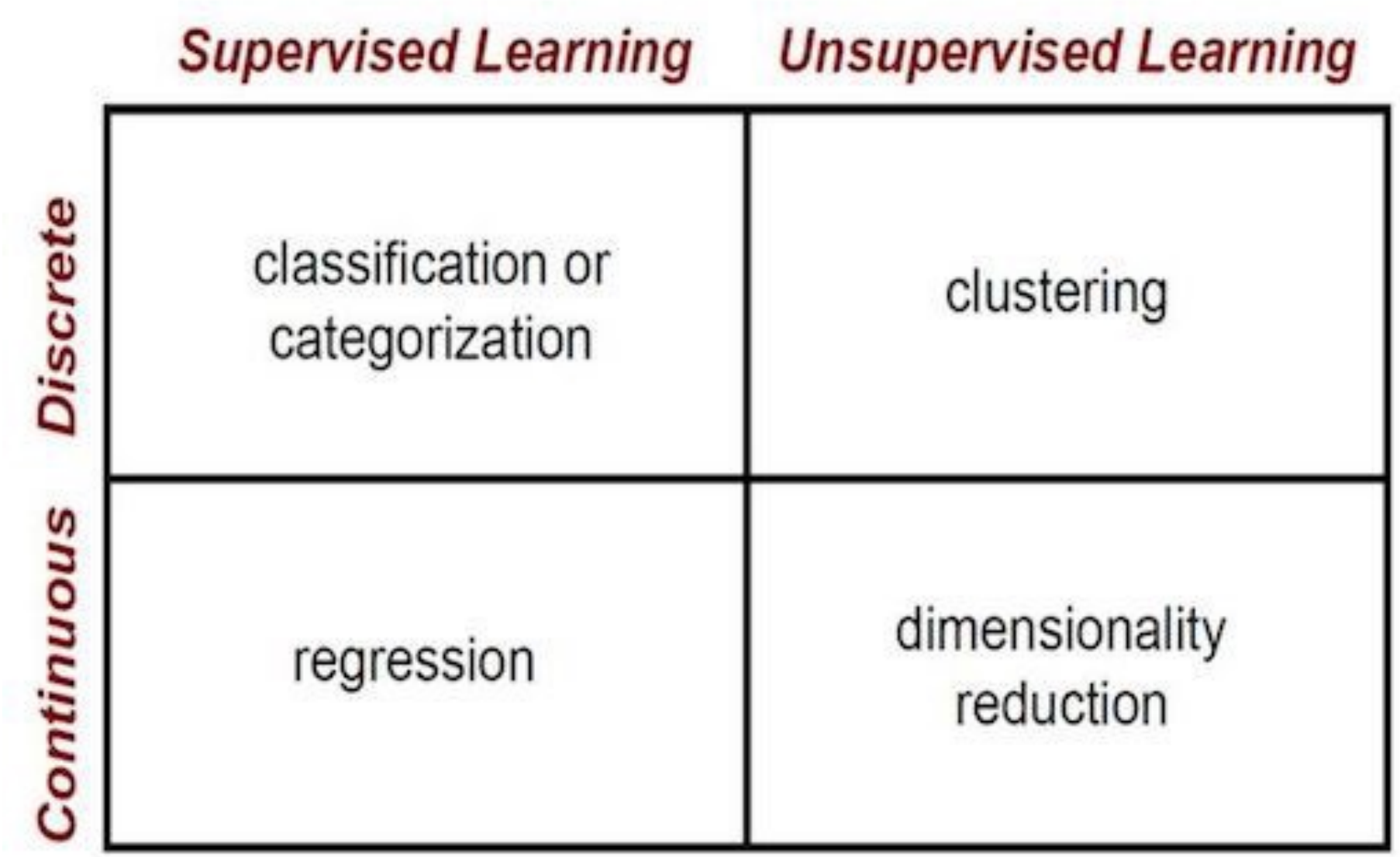
Summary: Unsupervised learning

- Clustering
  - Motivations
    - Group similar items
    - Detect outlier
  - Main algorithm: **K-mean**, K-medoids, Hierarchical clustering

- Dimensionality reduction
  - Motivations
    - data compression
    - data visualization
  - Main algorithm: **PCA**, t-SNE, LDA

- It is possible to use deep learning for unsupervised learning
  - Based on auto-encoder approach

Summary


Supervised Learning
Unsupervised Learning
Discrete
Continuous
classification or categorization
clustering
regression
dimensionality reduction