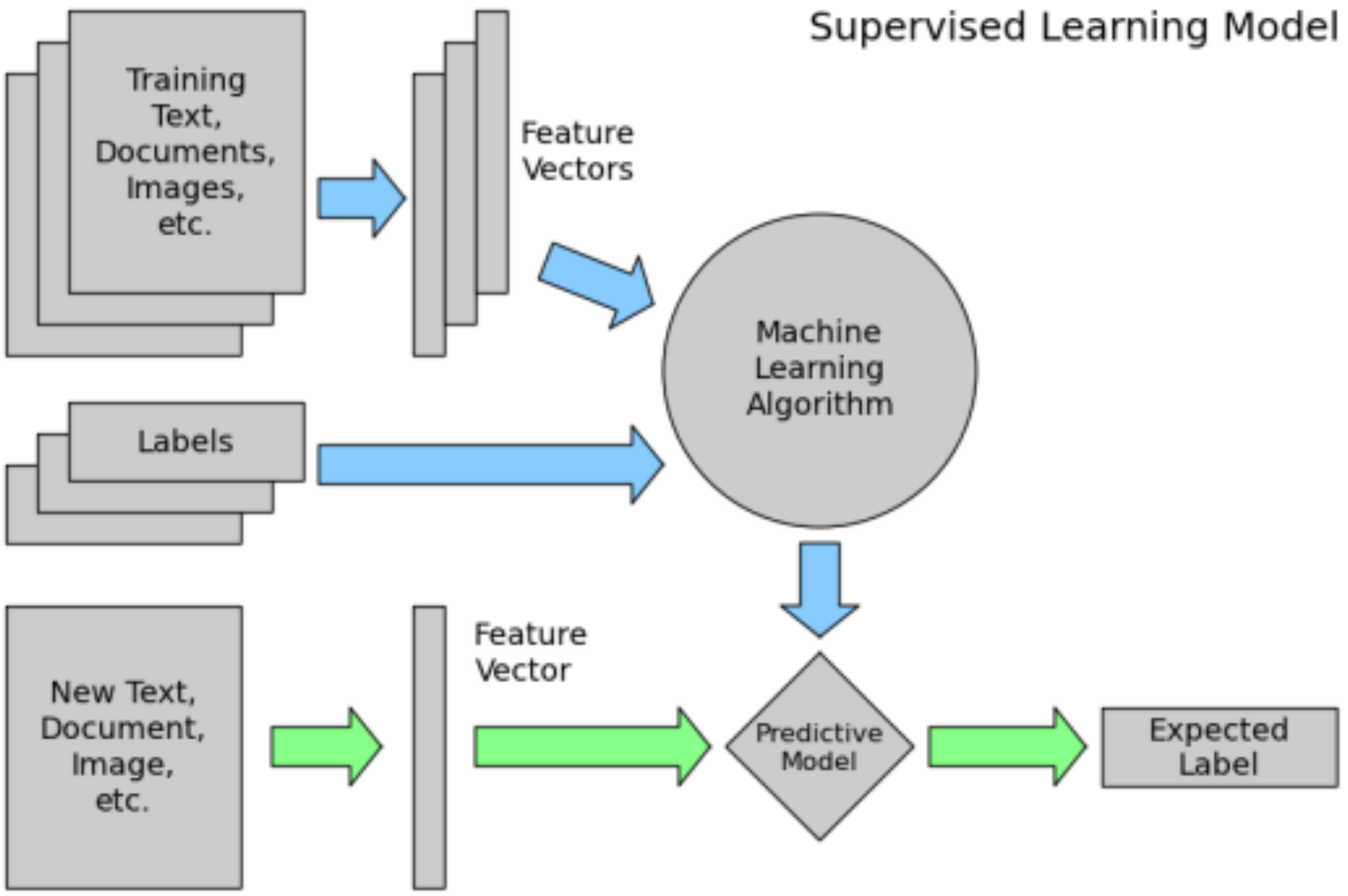
Unsupervised learning
Clustering

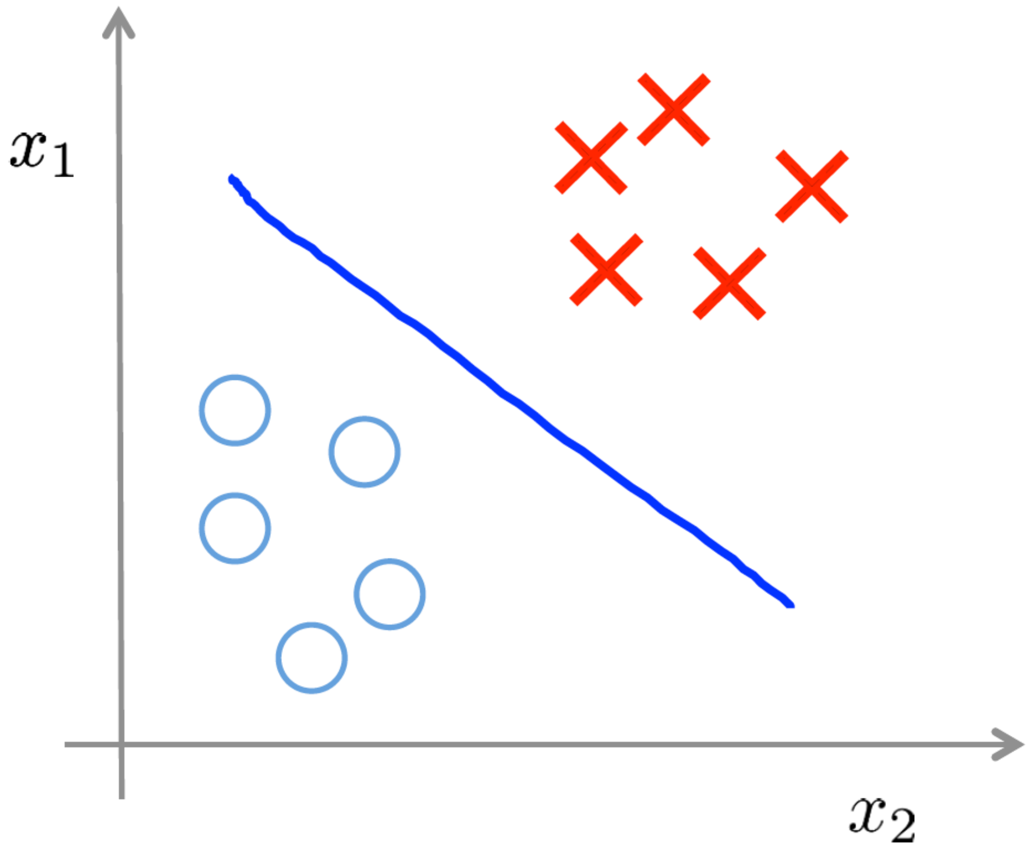
Michel.RIVEILL@univ-cotedazur.fr

Supervised learning


Supervised Learning Model
Training Text, Documents, Images, etc.
Feature Vectors
Machine Learning Algorithm
Labels
New Text, Document, Image, etc.
Feature Vector
Predictive Model
Expected Label

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



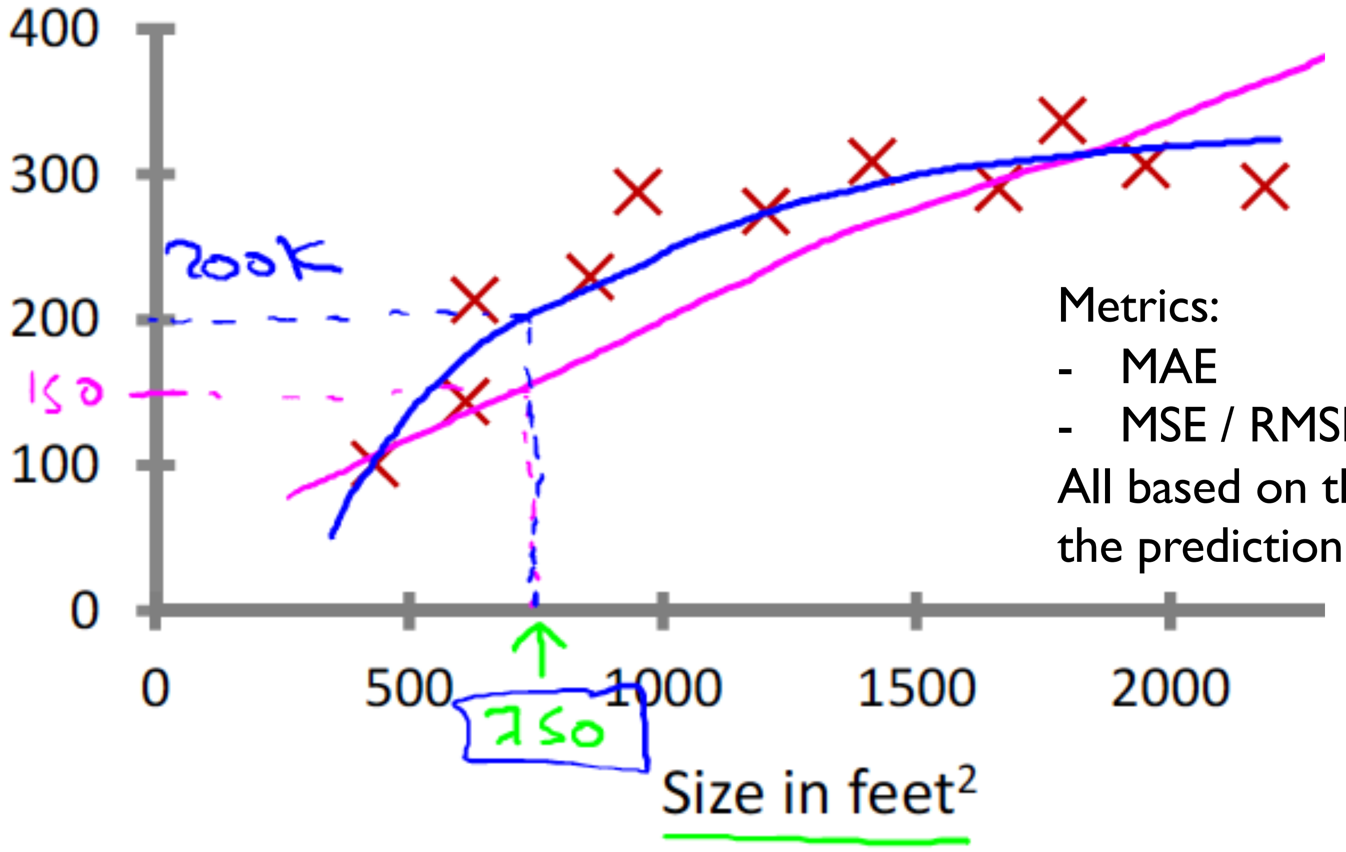
Metrics:

- Accuracy
- Precision / Recall / F1

All based on confusion matrix

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

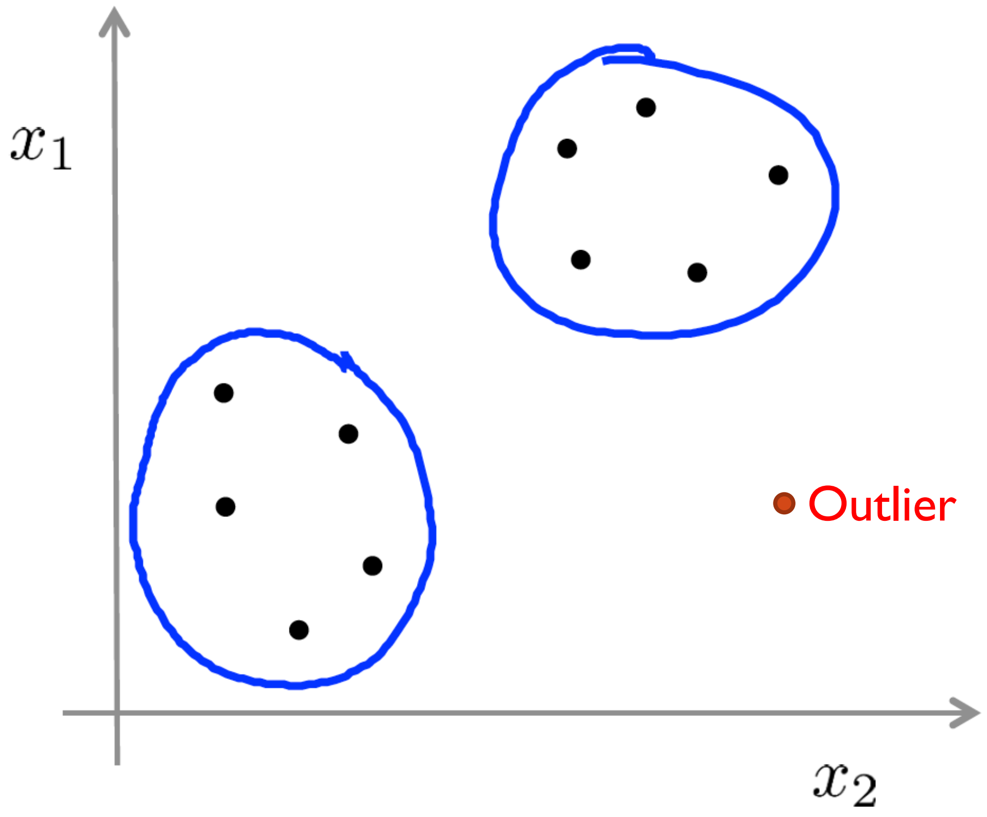


Metrics:
- MAE
- MSE / RMSE

All based on the distance between the prediction and the true value

Unsupervised learning: clusterisation

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



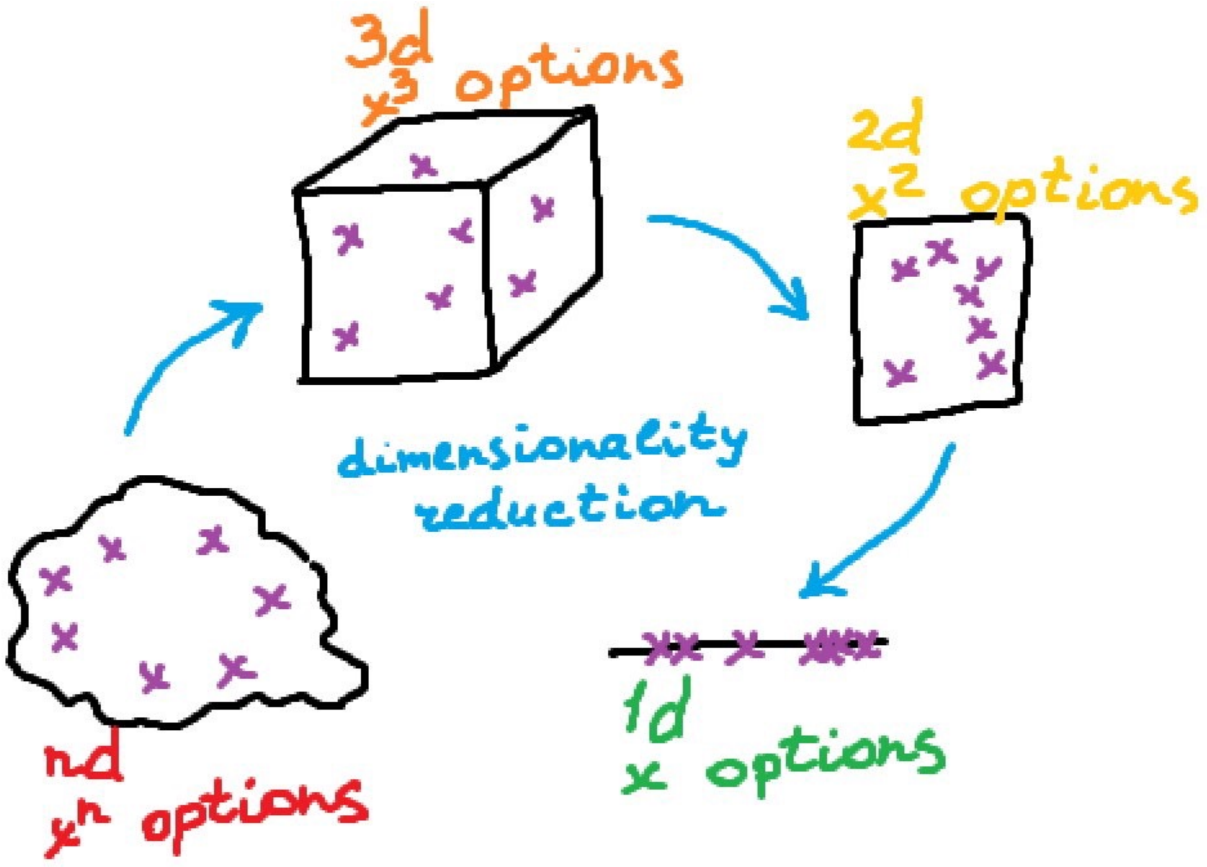
Motivations

- Group similar items
- Detect outlier

Unsupervised learning: reduction dimension

Motivations

- Reduce complexity
- Plot the data

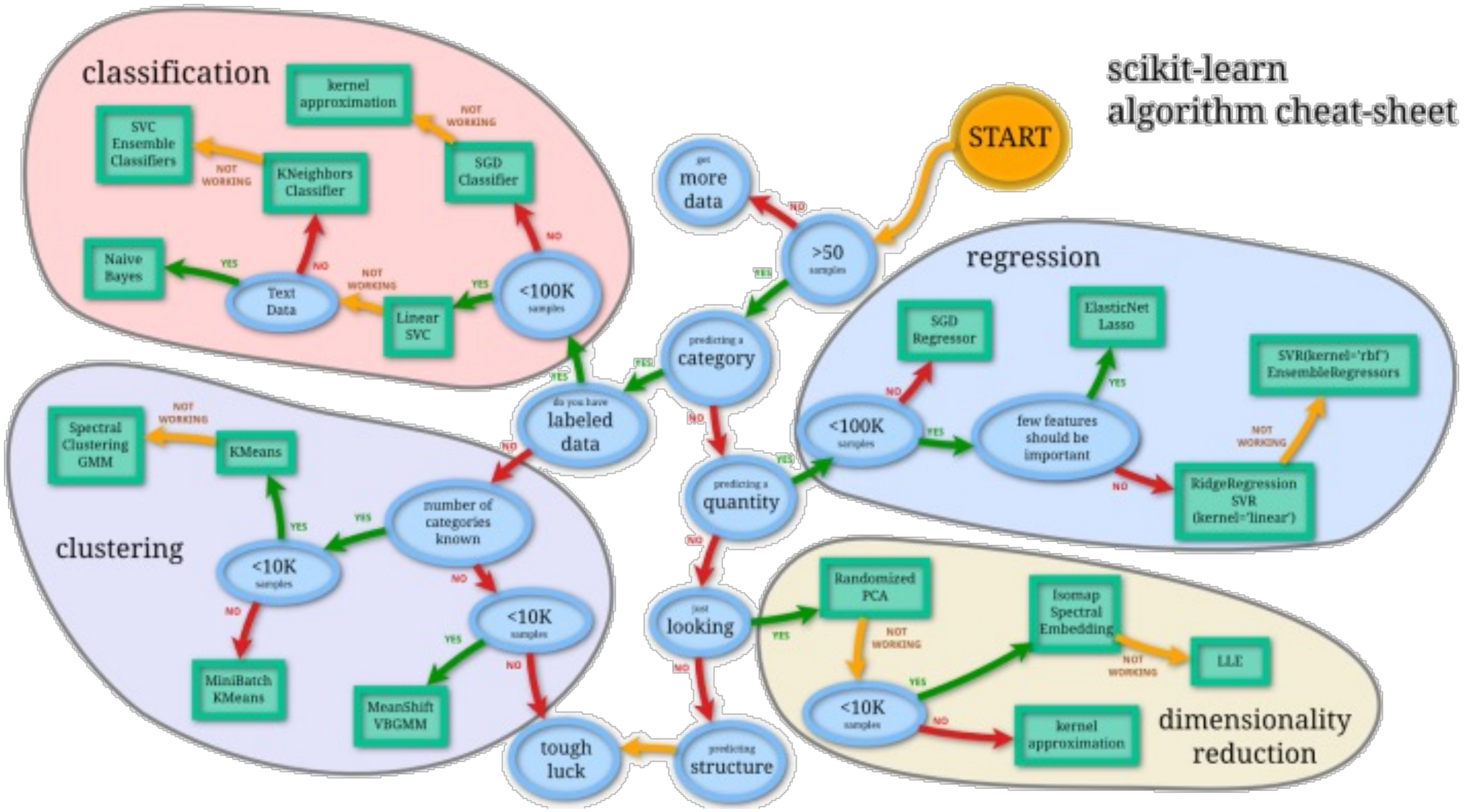


Unsupervised learning

- There is no supervisor and only input data is available.
  - We don't know the result

- Clustering
  - Motivations
    - Group similar items
    - Detect outlier
  - Main algorithm: **K-means**
- Dimensionality reduction
  - Motivations
    - data compression
    - data visualization
  - Main algorithm: **Principal Component Analysis (PCA)**

The scikit learn models



Clustering

Clustering

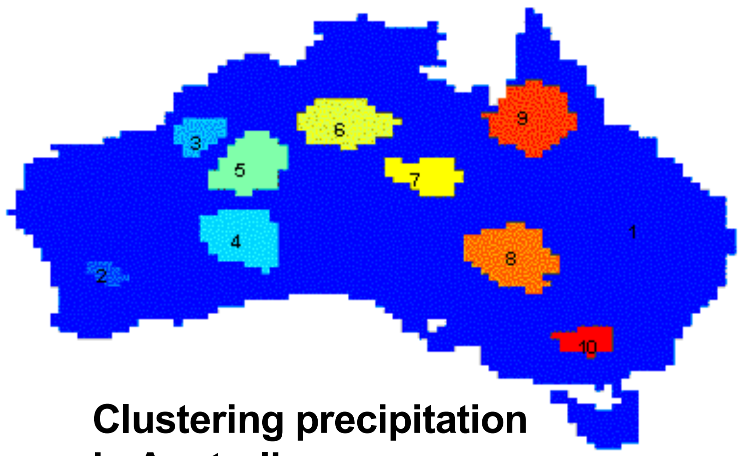
- The aim is now to find regularities, irregularities, relationships, similarities and associations in the input.

- 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 gold standard / no label
    - We need to take a look at clusters
  - Motivations
    - Group similar items
    - Detect outlier

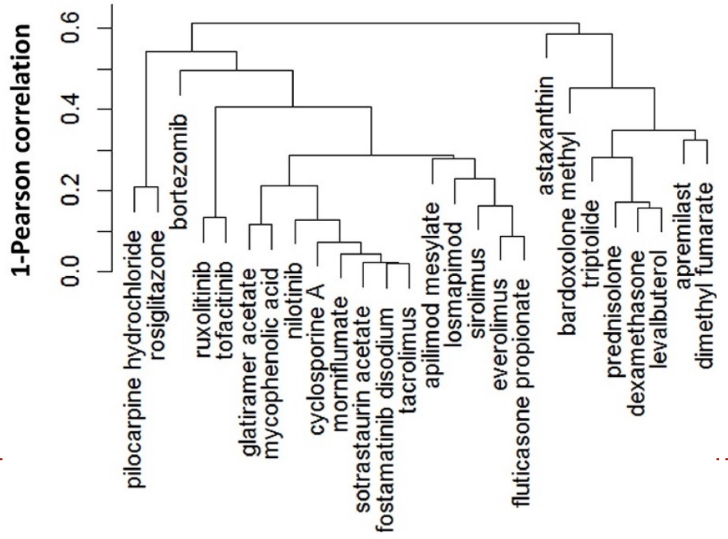
Applications of Cluster Analysis

- **Understanding**
  - Group related documents for browsing
  - Group genes and proteins that have similar functionality
  - Group stocks with similar price fluctuations



Clustering precipitation in Australia

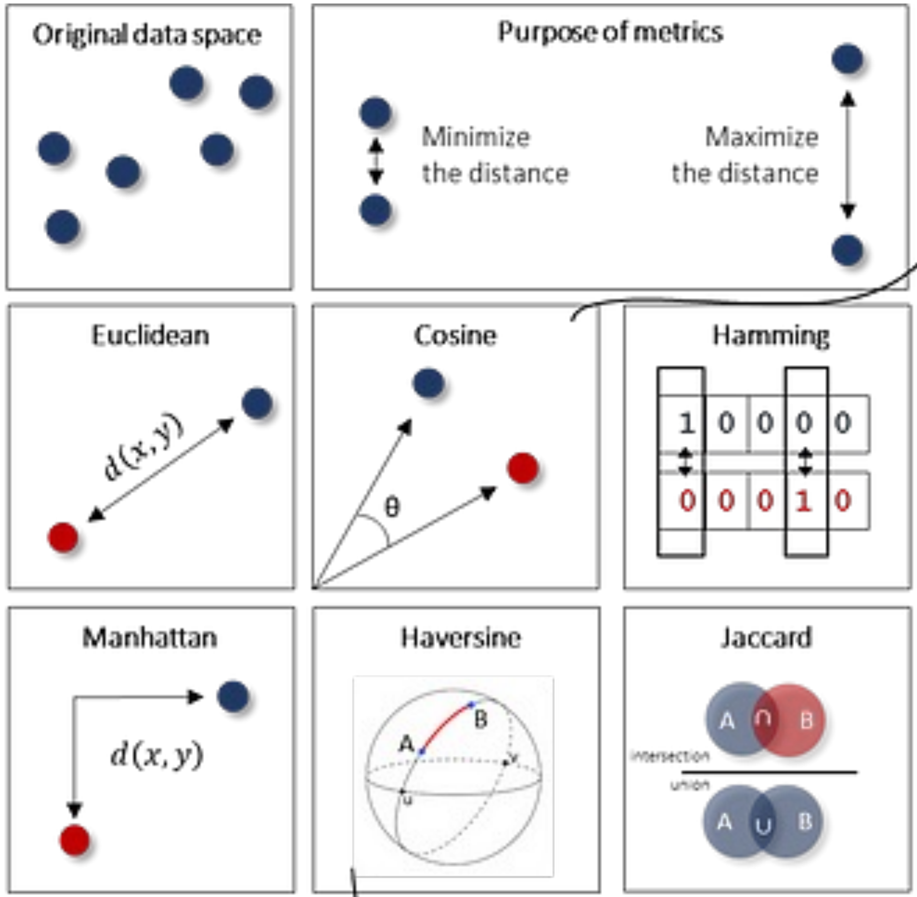
Clustering of drugs



- **Summarization**
  - Reduce the size of large data sets

What do we need for clustering

- A proximity measure to construct the cluster



$$D(x,y) = cos(\theta) = \frac{x \cdot y}{\|x\| \, \|y\|}$$

$$D(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

$$d^{HAD}(i,j) = \sum_{k=0}^{n-1}\left[y_{i,k} \neq y_{j,k}\right]$$

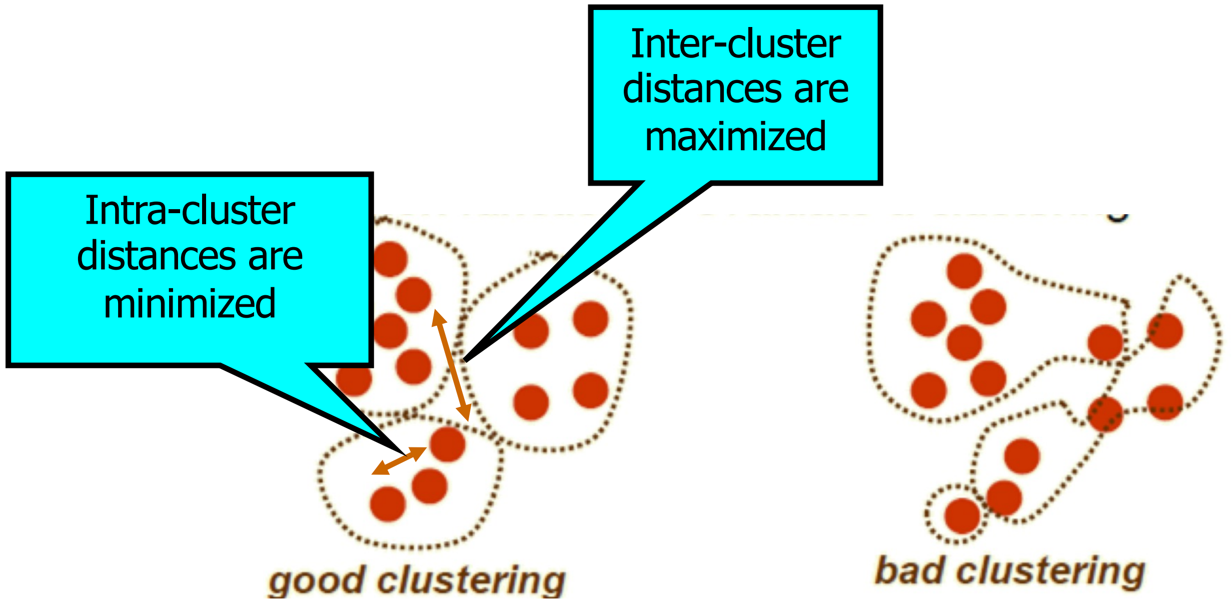
$$D(x,y) = \sum_{i=1}^{k}|x_i - y_i|$$

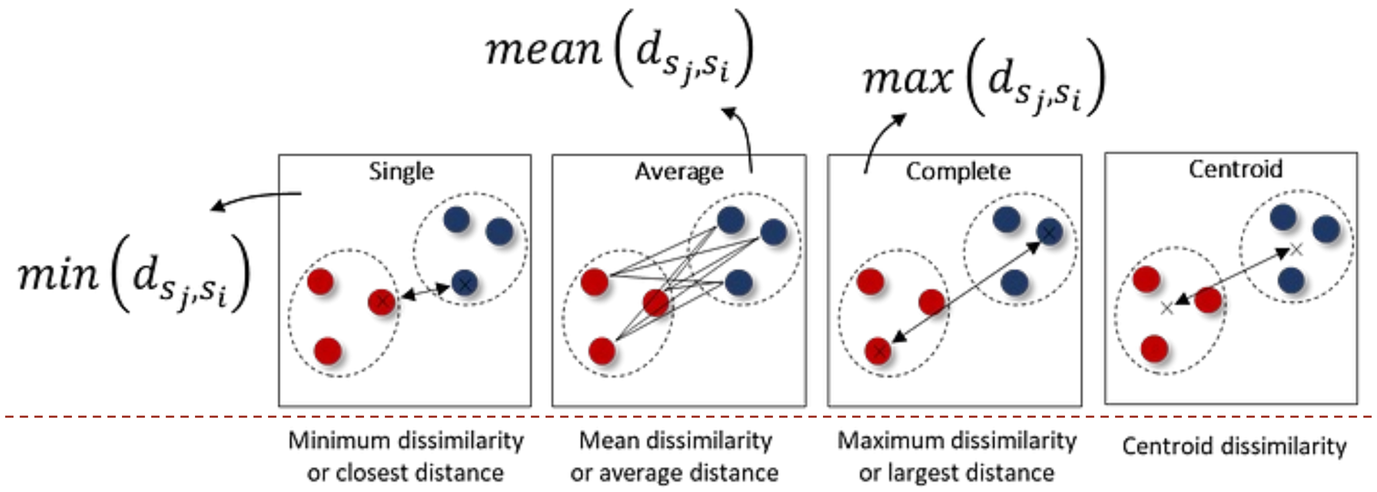
$$D(x,y) = 1 - \frac{|x \cap y|}{|y \cup x|}$$

$$d = 2r arcsin\left(\sqrt{sin^2\left(\frac{\varphi_2 - \varphi_1}{2}\right) + cos(\varphi_1)cos(\varphi_2)sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$$

What do we need for clustering

- Criterion function to evaluate the result


Inter-cluster distances are maximized
Intra-cluster distances are minimized
good clustering
bad clustering

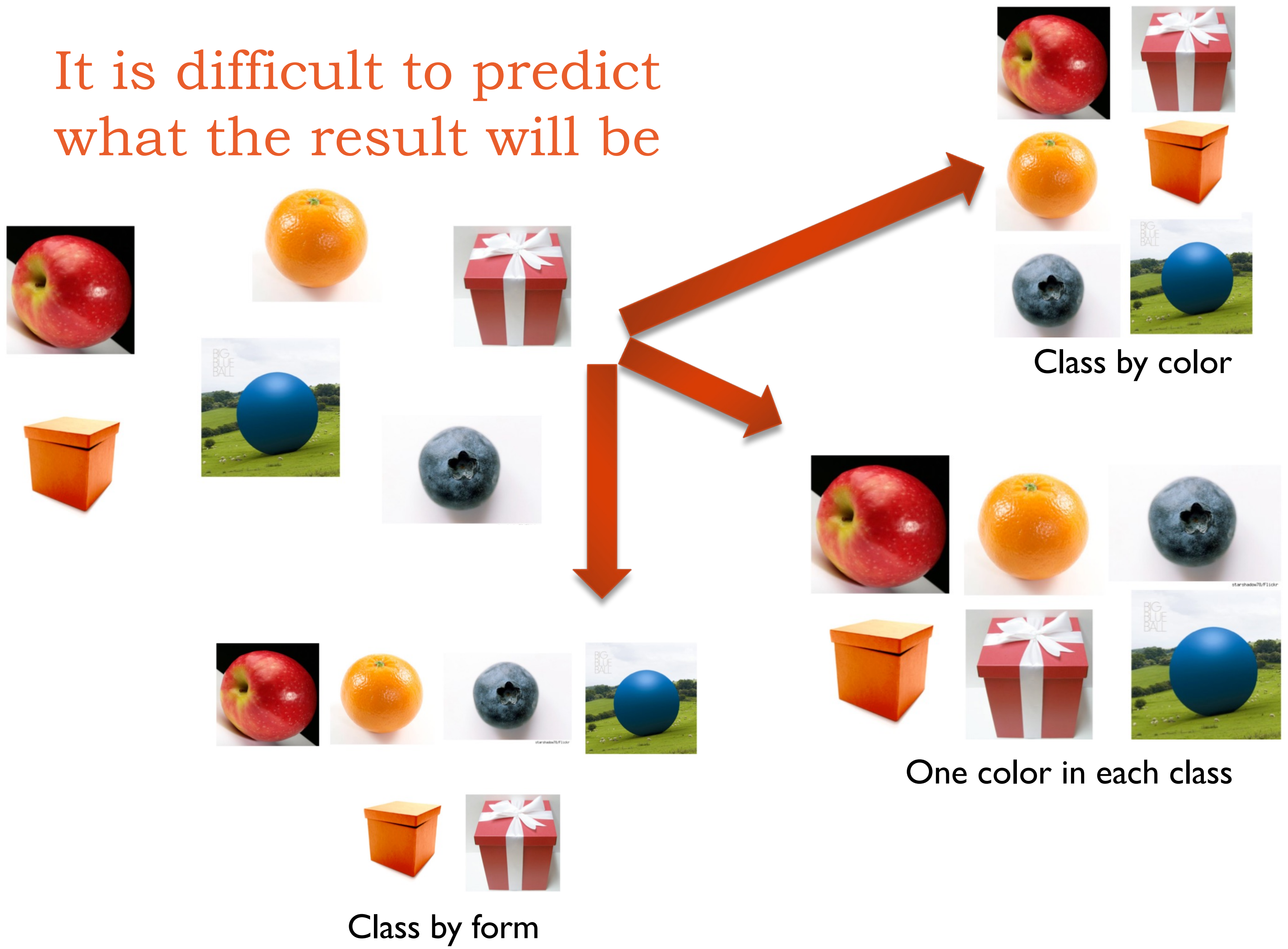

$mean\left(d_{s_j,s_i}\right)$
$max\left(d_{s_j,s_i}\right)$
$min\left(d_{s_j,s_i}\right)$
Single
Average
Complete
Centroid
Minimum dissimilarity or closest distance
Mean dissimilarity or average distance
Maximum dissimilarity or largest distance
Centroid dissimilarity

What do we need for clustering

- Algorithm to compute clustering
  - Based on ‘proximity measure’
    - Kmeans
    - Kmedoids
    - Hierarchical clustering
  - Based on density
    - DBScan

- A metric to evaluate the result
  - ARI index (with label)
  - Sihouette score (without label)

It is difficult to predict what the result will be



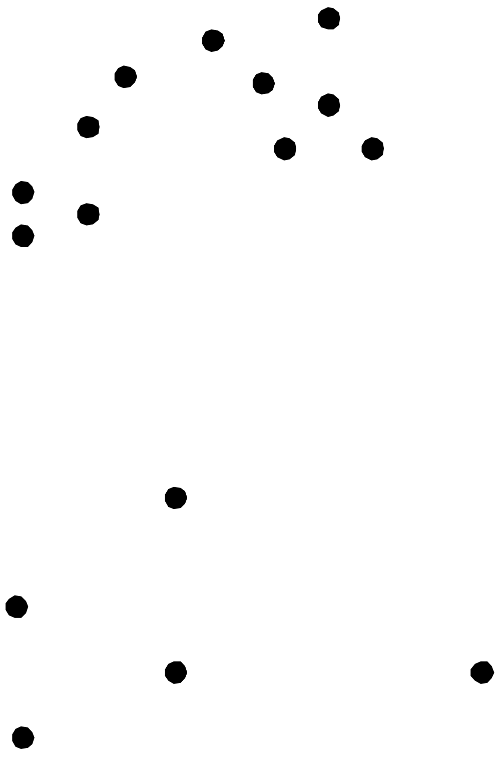
Preprocessing

- Each time the distance is used, it is necessary:
  - Delete the missing values or give them a value
  - Resize (normalize - standardize) the numerical variables so that they are comparable on a common scale.
    - Feature normalization
      - Scale input vectors individually to unit norm
      - Norm (l1, l2) equals one.
        - $l1 = \sum |x_i|$
        - $l2 = \sum x_i^2$
    - Feature standardization
      - Scale input vector : $z : \frac{x-\mu}{\sigma}$
    - Observation normalization
      - Scale each observation independently of other observations
      - Norm (l1, l2) equals one

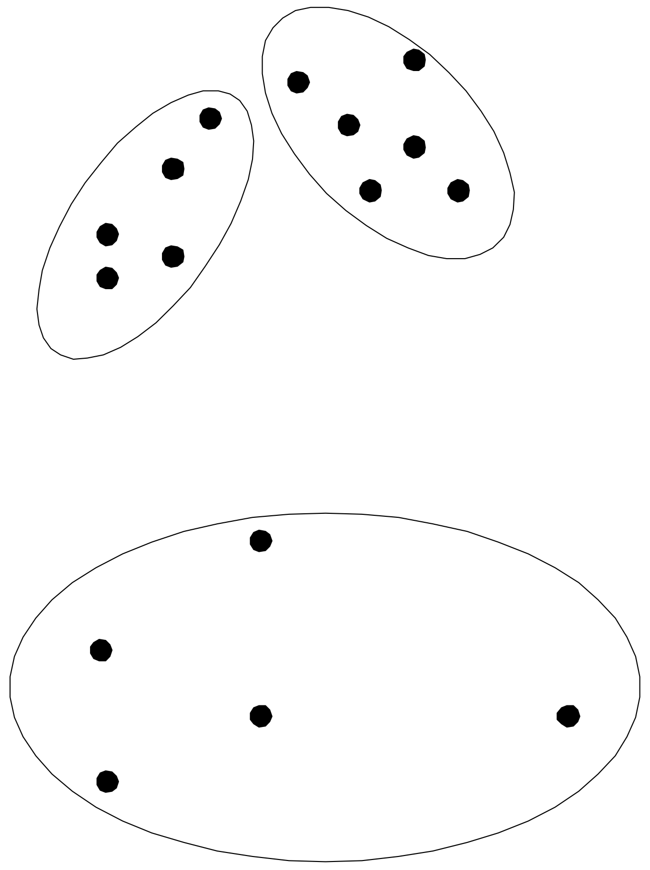
Types of Clusterings

- A clustering is a set of clusters
- Important distinction between hierarchical and partitional sets of clusters
- Partitional Clustering
  - A division of data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset
- Hierarchical clustering
  - A set of nested clusters organized as a hierarchical tree

Partitional Clustering

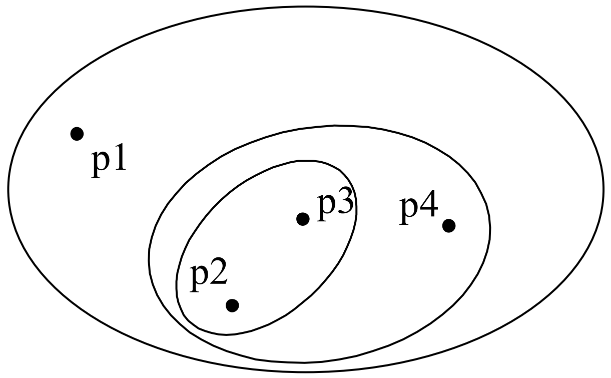


Original Points

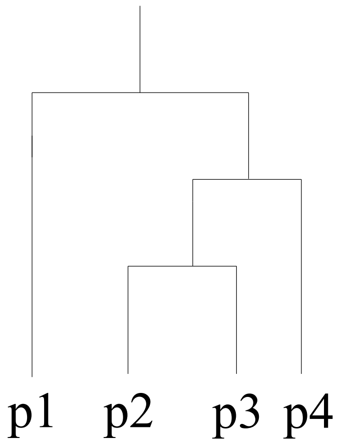


A Partitional Clustering

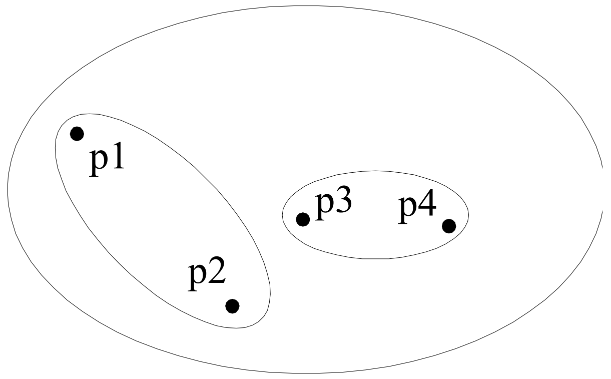
Hierarchical Clustering



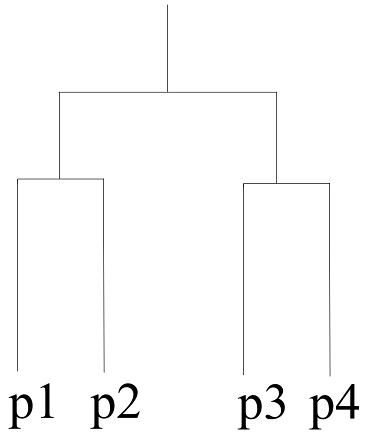
Traditional Hierarchical Clustering



Traditional Dendrogram

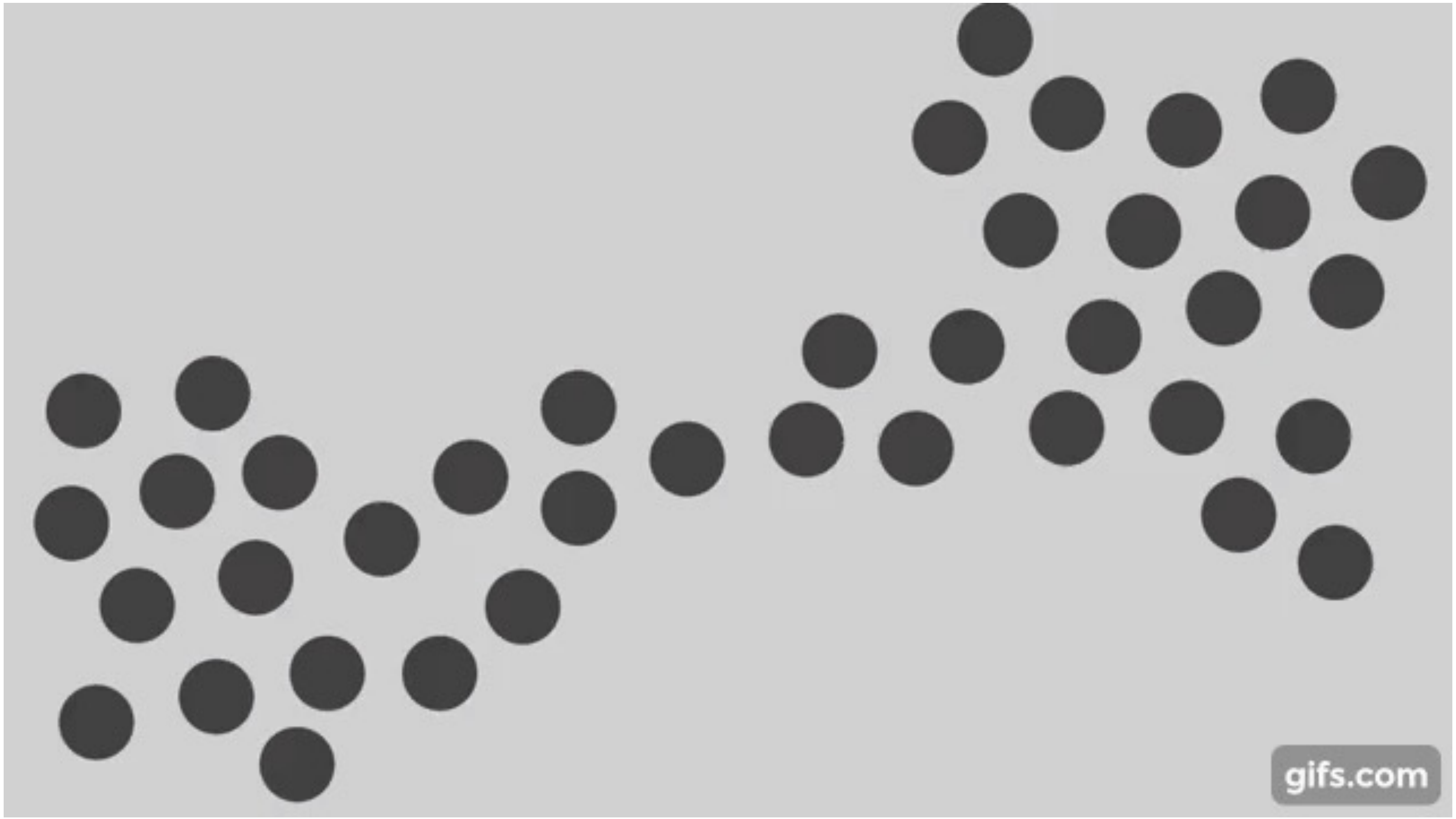


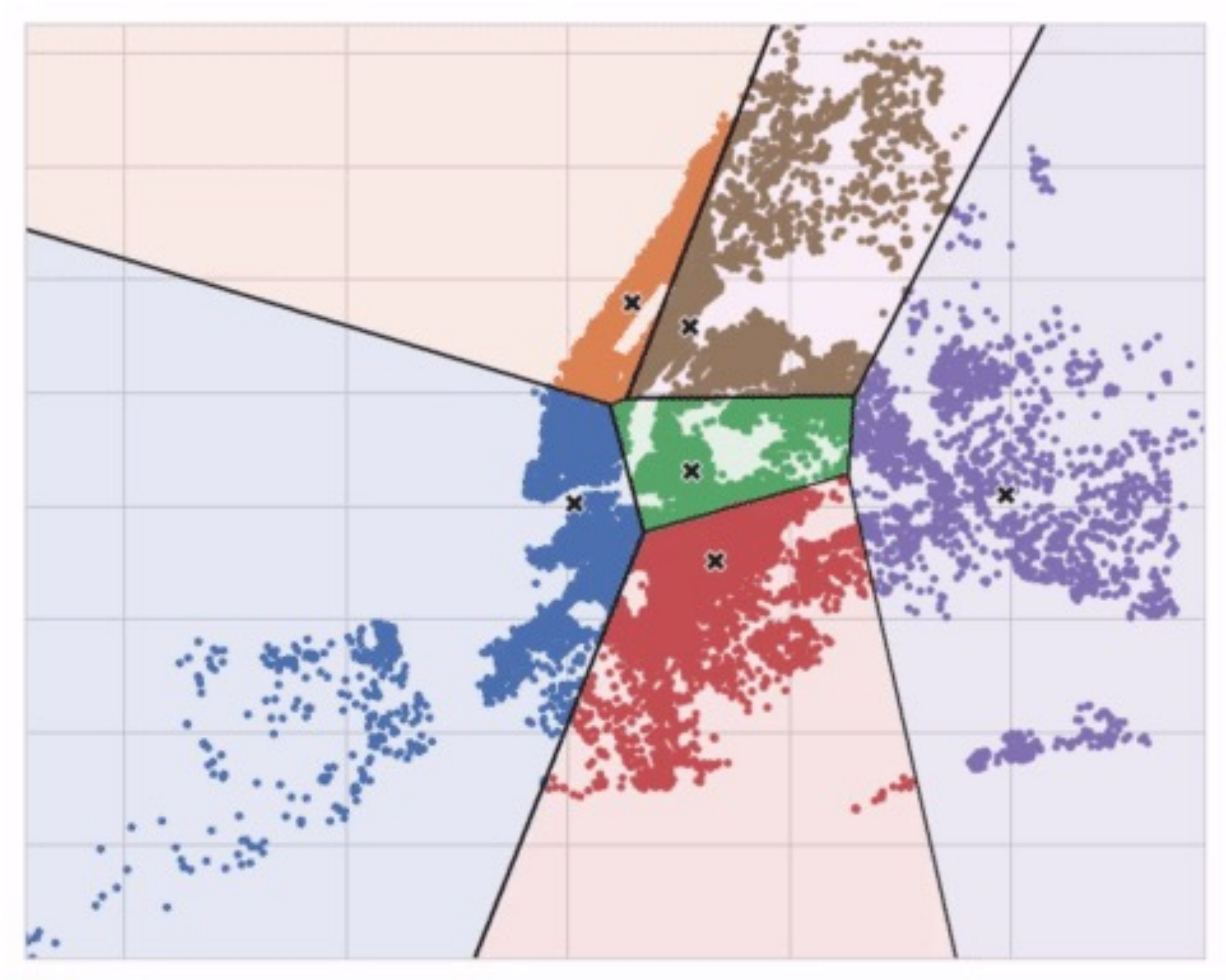
Non-traditional Hierarchical Clustering

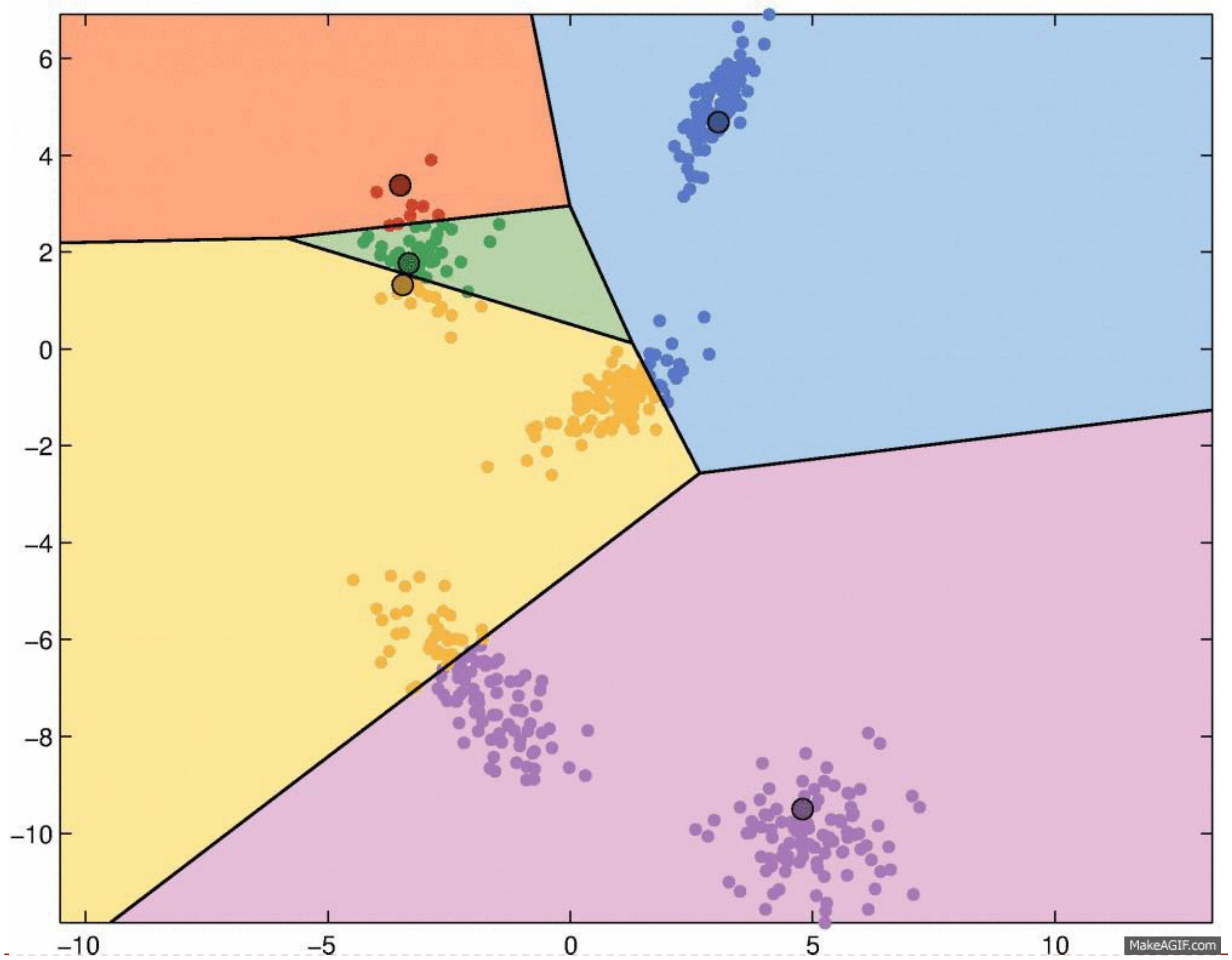


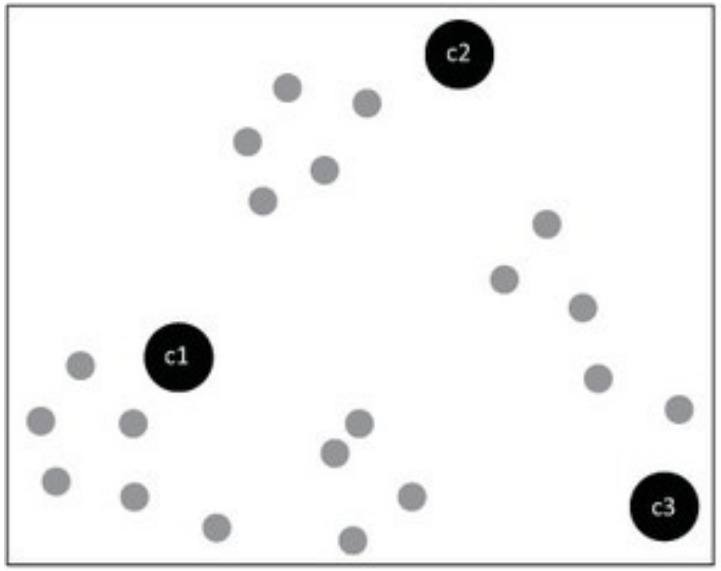
Non-traditional Dendrogram

A first algorithm: K-means

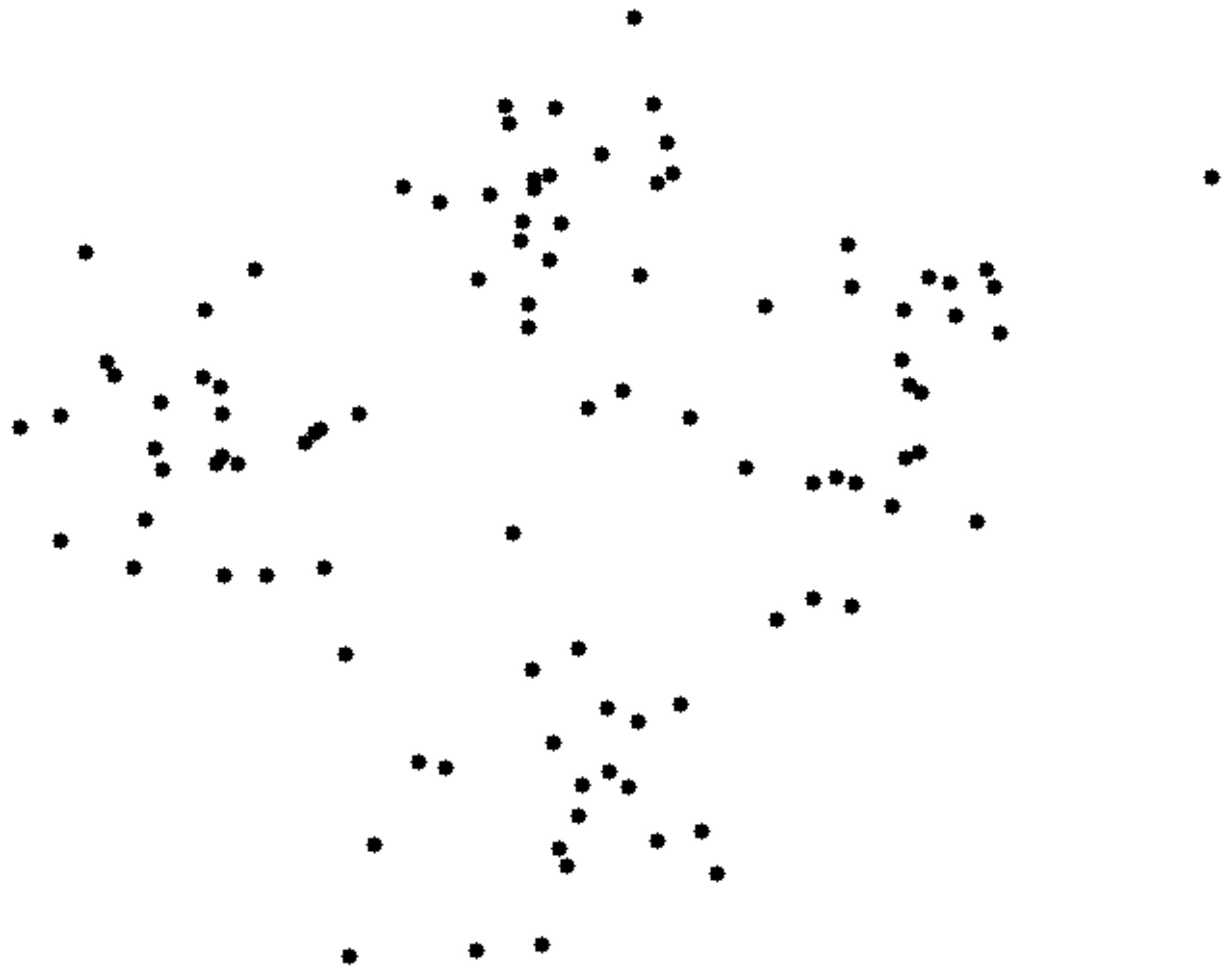

gifs.com



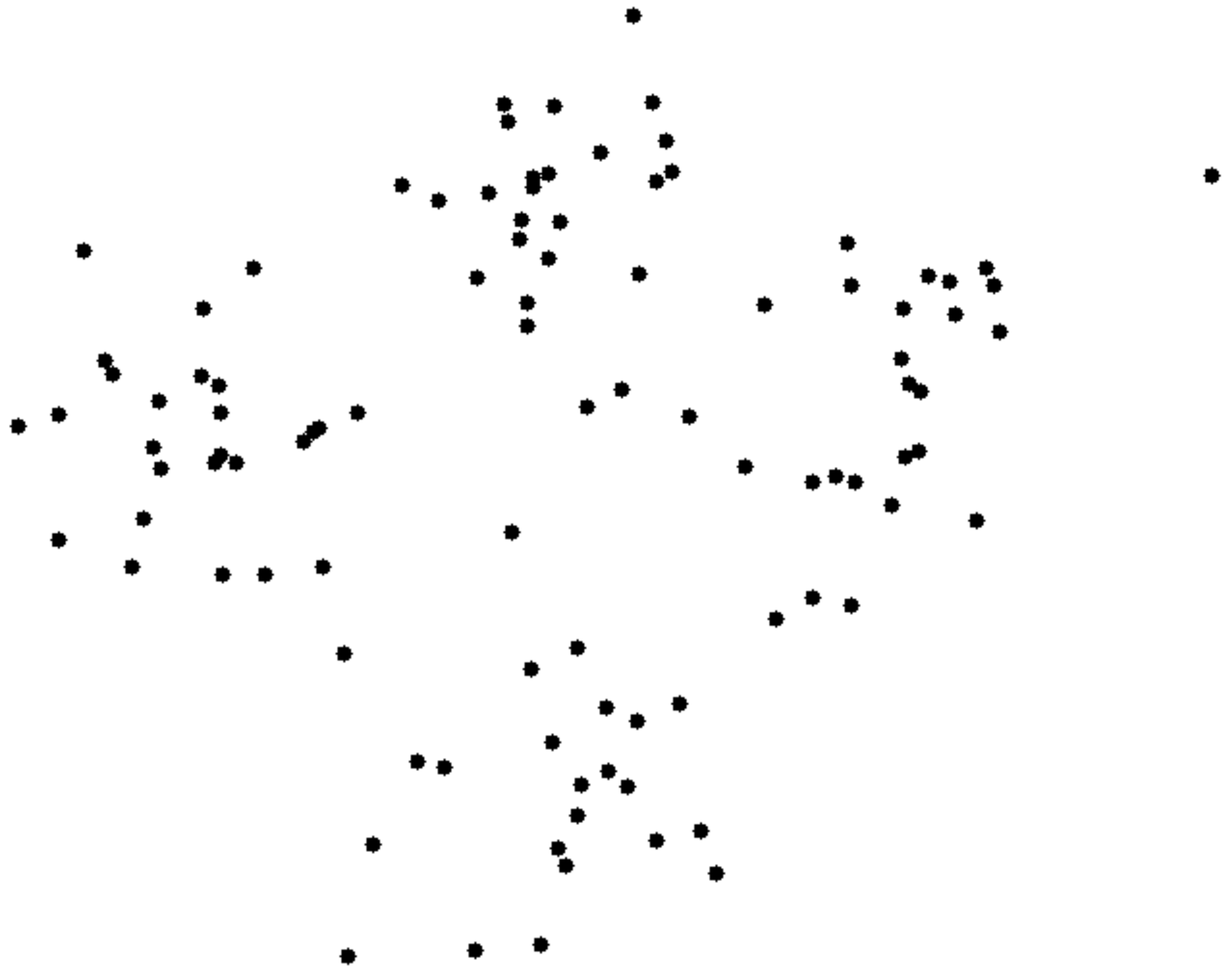

6
4
2
0
−2
−4
−6
−8
−10
−10
−5
0
5
10
MakeAGIF.com


c2
c1
c3

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) ← $\min_j d(x^{(i)}, \mu_j)$

   # Step 2: Compute new cluster centroids as the centroids of the clusters.
   for j = 1 to k
      $\mu_i$ = 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

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
  - $J(c^{(1)}, c^{(2)}, \dots, c^{(m)}, \mu_1, \mu_2, \dots, \mu_k) = 1/m * \Sigma_{i=1..m} d(x^{(i)} - \mu_c(i))2$
- How to choose the good clustering for k clusters

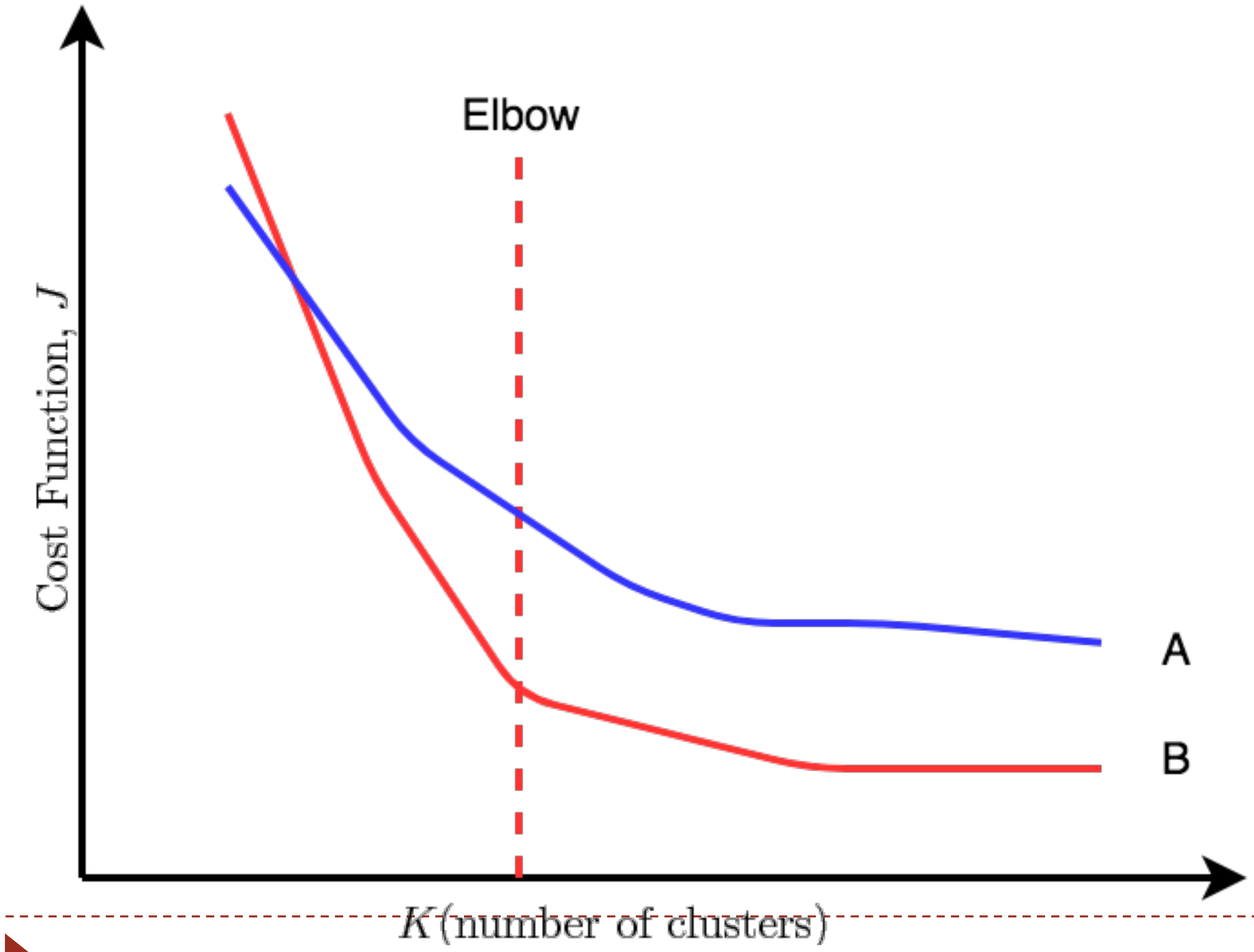
```
For I = 1 to N {
    Apply K-means for k clusters
    Compute distortion function
}
```

  - Choose the clustering that gave the lowest distortion cost
  - Random initialization $\rightarrow$ 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 coefficient

- 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 cluster.
    - Measure the cohesion
  - b: The mean distance between a sample and all other points in other cluster.
    - Measure the separation
- For a point : $silhouette = \frac{b-q}{\max(a,b)} \in [-1, 1]$
  - <0 incorrect cluster affectation
  - >0 correct cluster affectation
- For a clustering : Silhouette = average of all silhouette points $\in [-1, 1]$
  - -1 incorrect clustering
  - +1 highly dense clustering
  - 0 indicates overlapping clusters.

Silhouette method – calculate silhouette

- For each observation i, the silhouette width s_i is calculated as follows:

1. For each observation i, calculate the average dissimilarity ai between i and all other points of the cluster to which i belongs.

2. For all other clusters C, to which i does not belong, calculate the average dissimilarity d(i,C) of i to all observations of C.
   The smallest of these d(i,C) is defined as $b_i = \min_C d(i,C)$.

   The value of b_i can be seen as the dissimilarity between i and its "neighbor" cluster, i.e., the nearest one to which it does not belong.

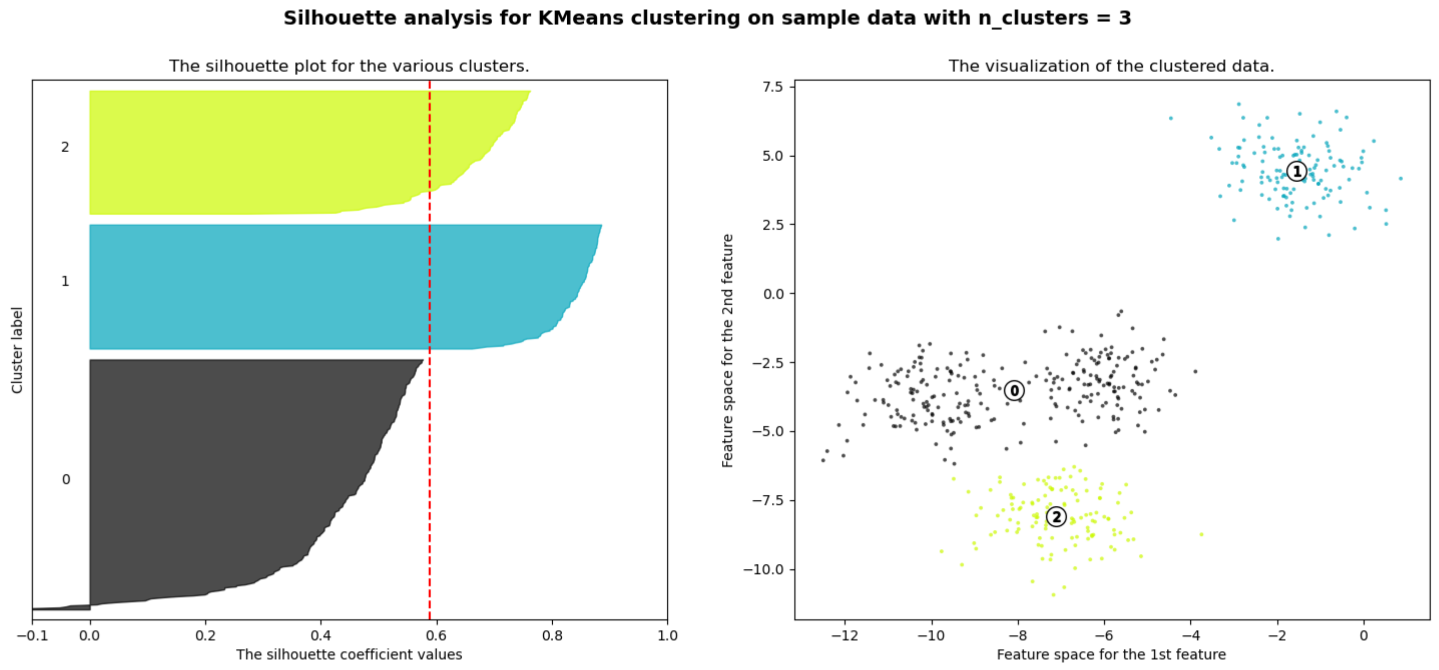
3. Finally, the silhouette width of the observation i is defined by the formula: $S_i = (b_i - a_i)/\max(a_i, b_i)$.

Silhouette interpretation

Score bounded between [-1, 1]

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

Silhouette score = mean(silhouette coefficient for all samples)

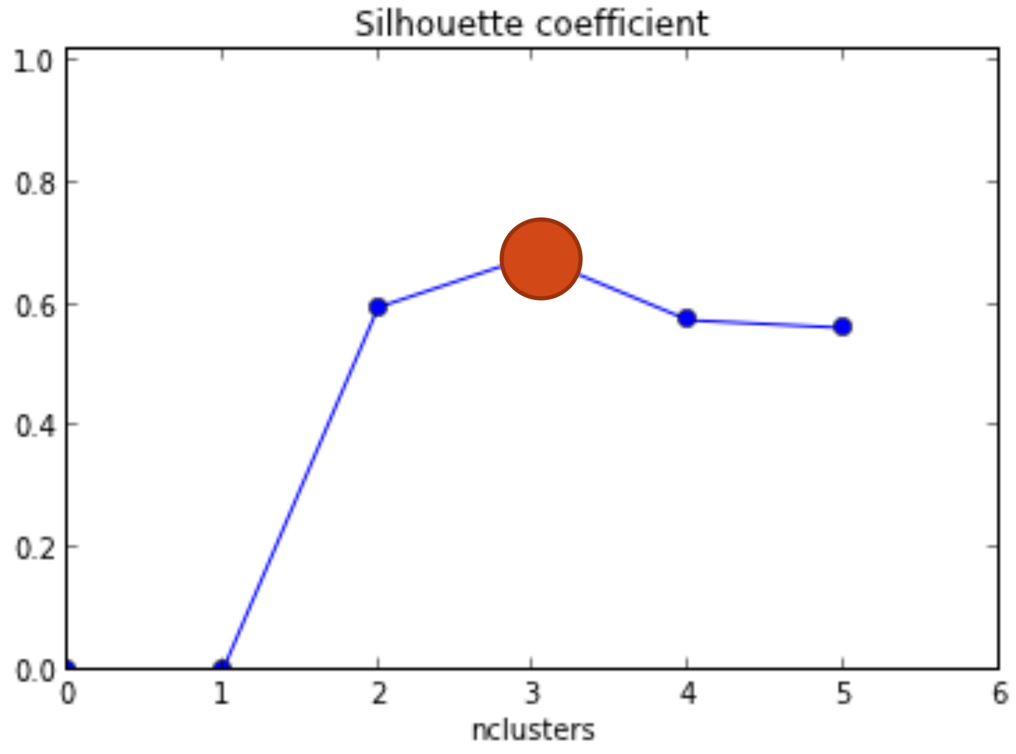


Silhouette interpretation

From sklearn.metrics import silhouette_score
# Silhouette is only defined if number of labels is
#   2 <= n_labels <= n_samples - 1.

score = silhouette_score(X, labels, metric=…, sample_size=…)
# which labels ?
# choose the metric
# choose a random subset of the data or all data

Silhouette score / nb clusters



K-means with sklearn

```
from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=I, n_init=…, …).fit(X)
# Sklearn Kmeans

# 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 with sklearn

- `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:
  - Single run methods:
    - Use a sequential algorithm to produce an initial estimate of centroids.
- Multiple run methods:
  - Create different initial choices of centroids, run the k-means for each, and select the best result.
  - Use tools from stochastic optimization techniques (simulated annealing, genetic algorithms, etc.).

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

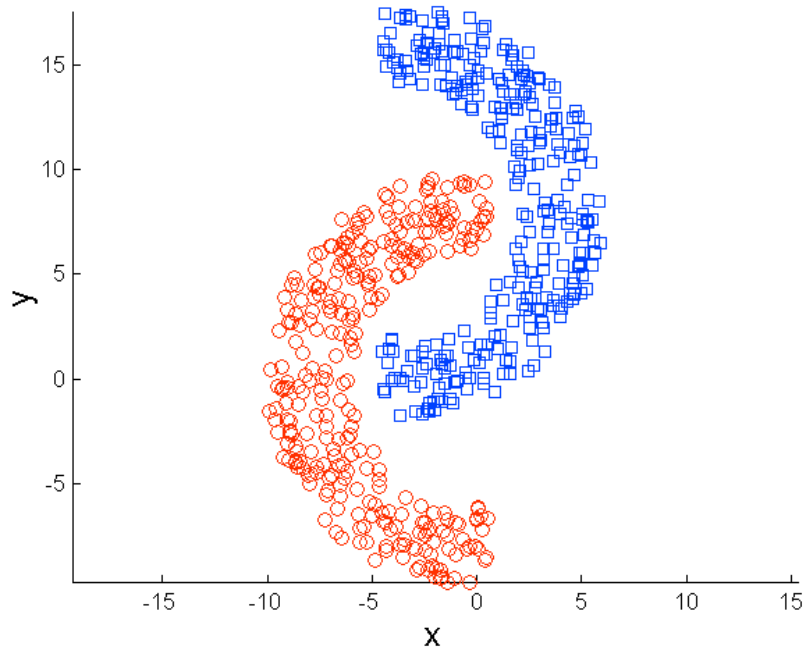
- Disadvantage 3:
  - k-means is sensitive to outliers and noise.

- 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.

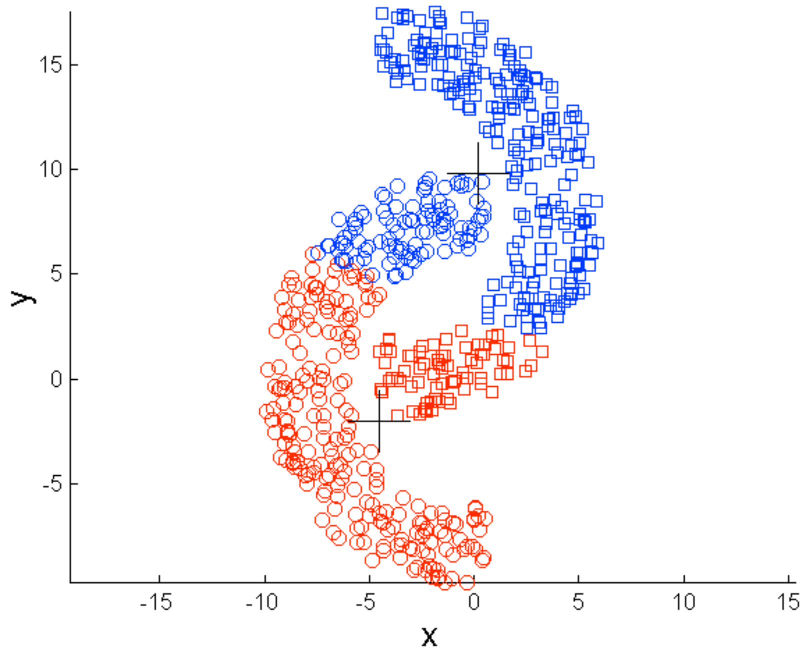
- Disadvantage 4:
  - k-means are not suitable for data with nominal (categorical) coordinates.

- Strategies for dealing with drawback 4:
  - Use a k-medoids algorithm.

K-means drawback: Non-globular Shapes



Original Points



K-means (2 Clusters)

Change approach: density-based clustering (not in this lecture)

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={$\underline{1}$,3,4} and C2={5,$\underline{8}$,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,$\underline{3}$,4,5} and C2={$\underline{8}$,9}
  - $J_{\{3,8\}} =dist(1,3)^2+dist(4,3)^2+dist(5,3)^2+dist(9,8)^2=10$
  - $J_{\{3,8\}} - J_{\{1,8\}}= -16$ <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

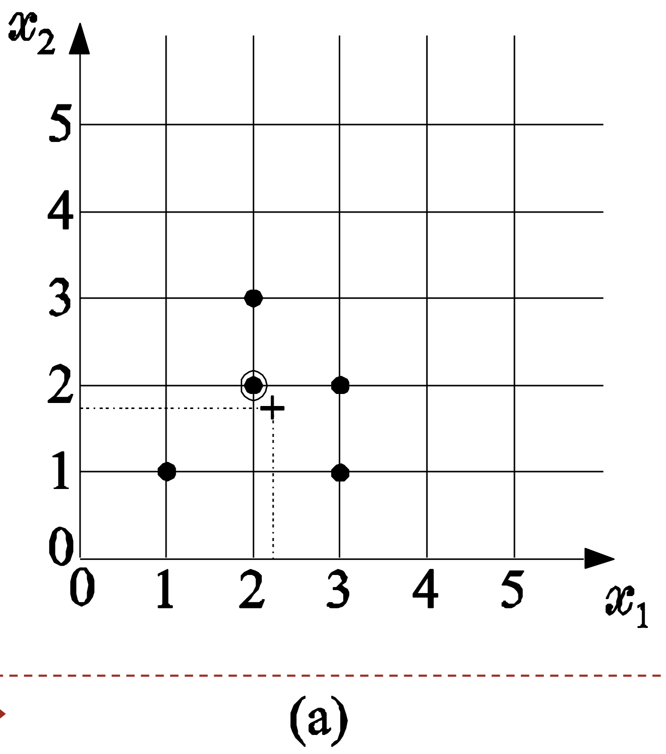
- Try to swap 3 and 5 → M={5,8} → C1 and C2 unchanged
  - $J_{\{5,8\}} > J_{\{3,8\}}$

K-means vs K-medoids

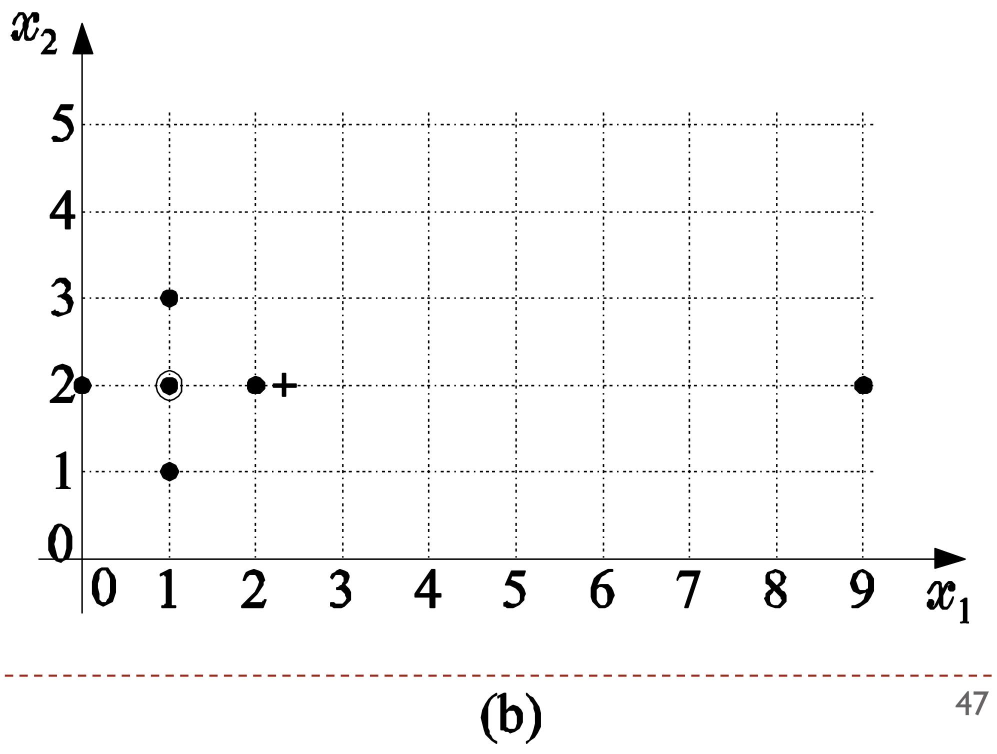
K-means	K-medoids
1. Suited only for continuous domains	1. Suited for either cont. or discrete domains
2. Algorithms using means are sensitive to outliers	2. Algorithms using medoids are less sensitive to outliers
3. The mean possess a clear geometrical and statistical meaning	3. The medoid has not a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

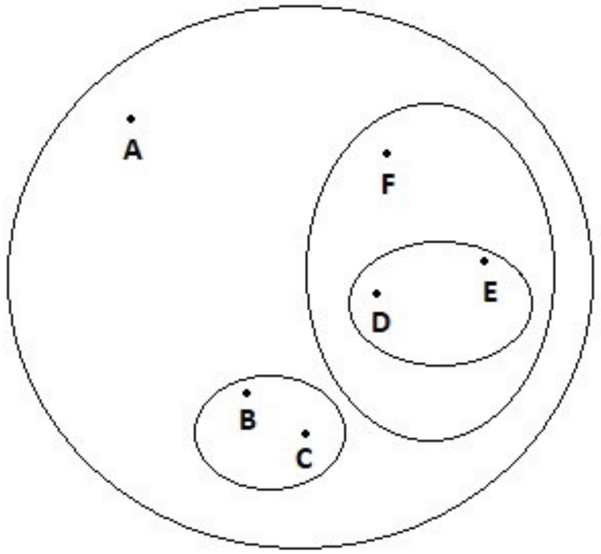
Hierarchical clustering (or agglomerative)

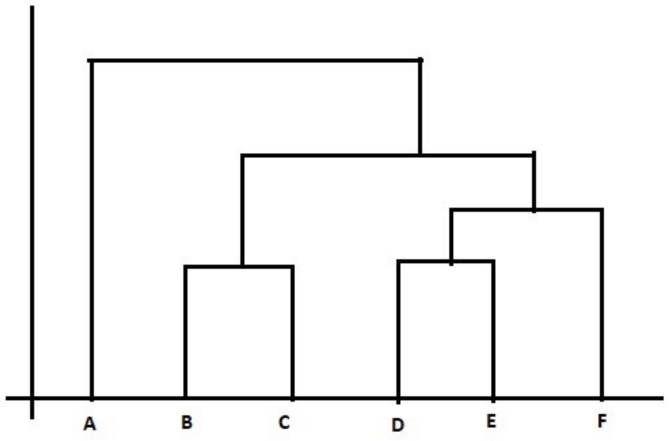
Hierarchical clustering

- 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

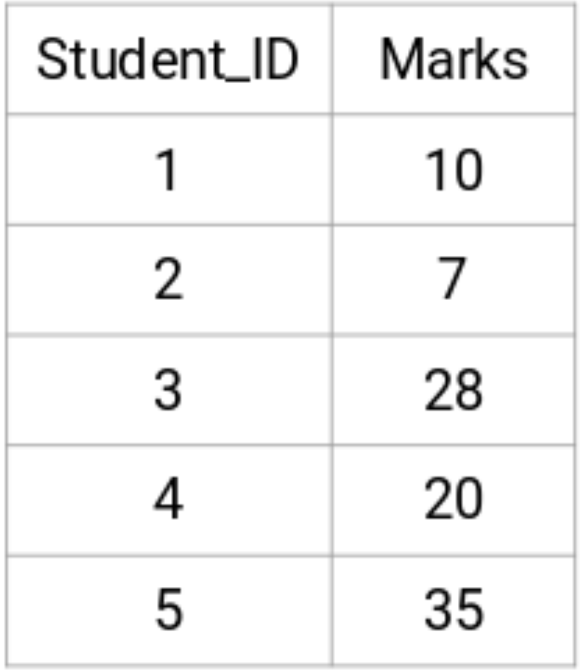
- The basic algorithm of Agglomerative is straight forward.
  - Compute the proximity matrix
  - Let each data point be a cluster
  - Repeat: Merge the two closest clusters and update the proximity matrix
  - Until only a single cluster remains
- It can be visualized by a dendogram





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.
    - $D(S_1, S_2) = \sqrt{(10-7)^2} = 3$

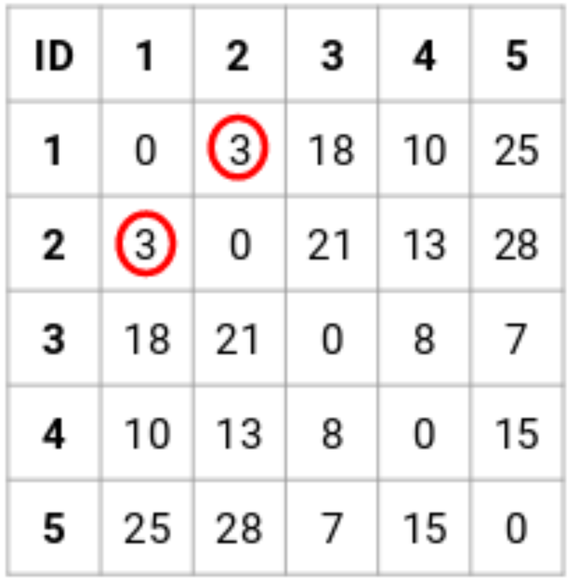
Student_ID	Marks
1	10
2	7
3	28
4	20
5	35

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

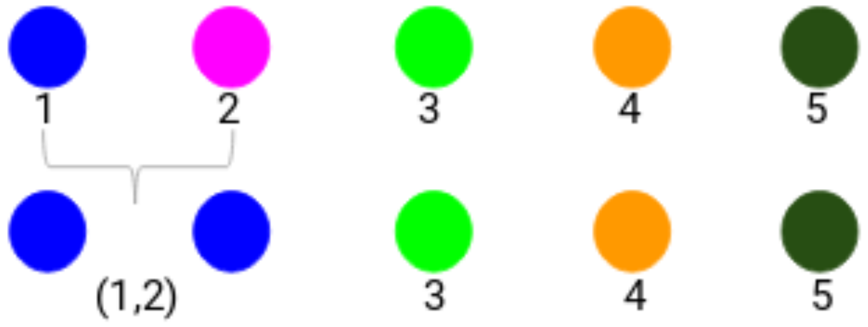
How to perform Hierarchical Clustering

- 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:

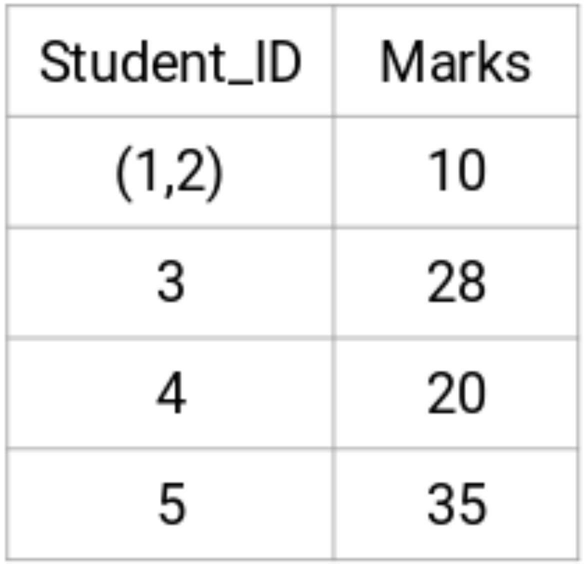


ID	1	2	3	4	5
1	0	3	18	10	25
2	3	0	21	13	28
3	18	21	0	8	7
4	10	13	8	0	15
5	25	28	7	15	0

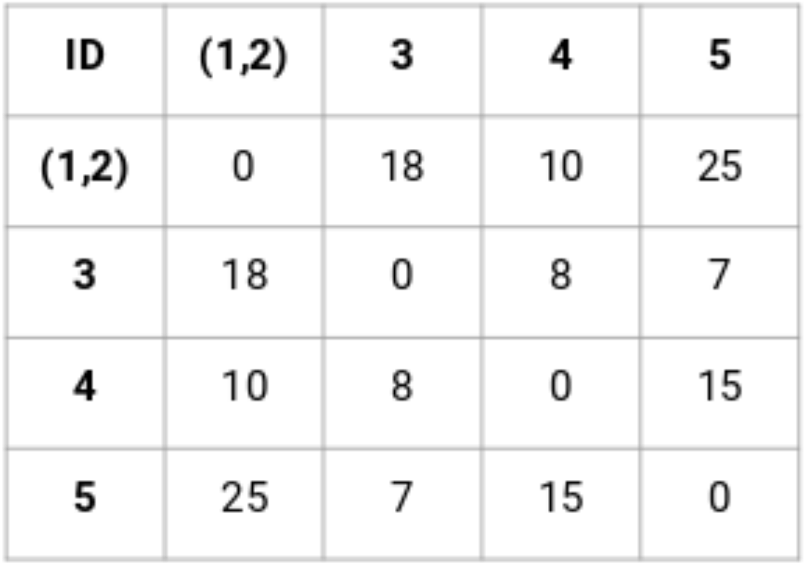


How to perform Hierarchical Clustering

- 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.



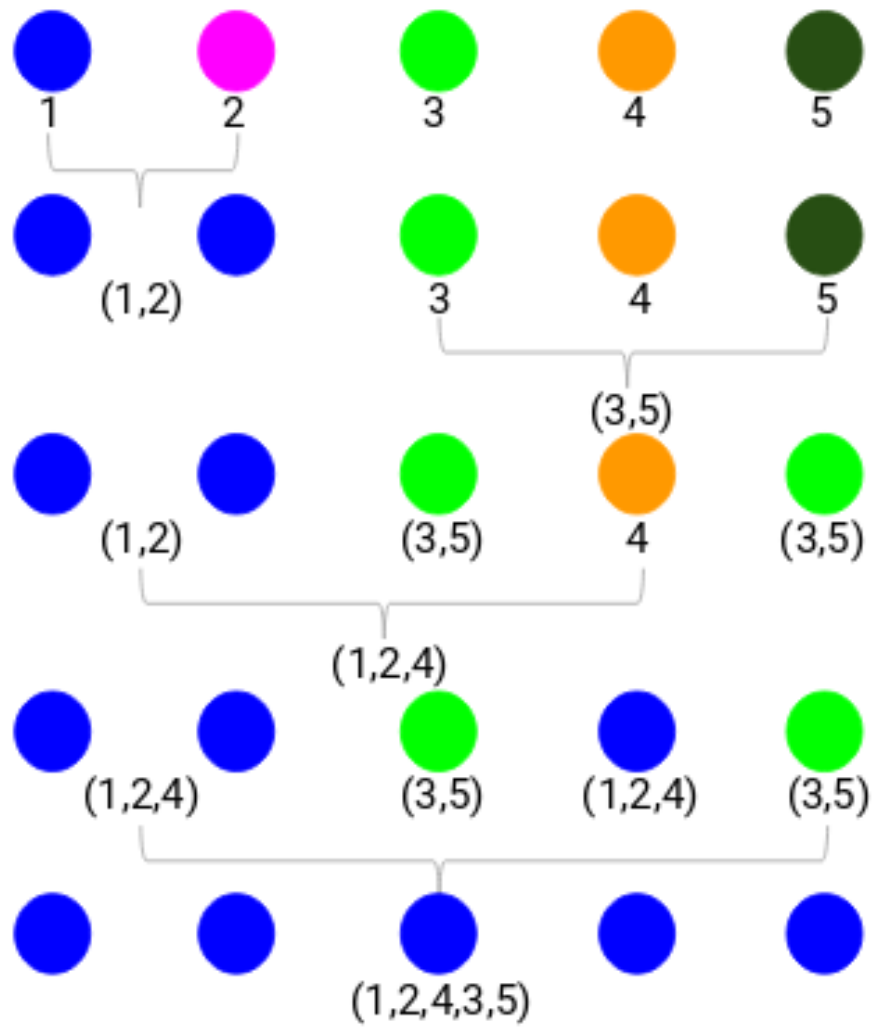
Student_ID	Marks
(1,2)	10
3	28
4	20
5	35



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

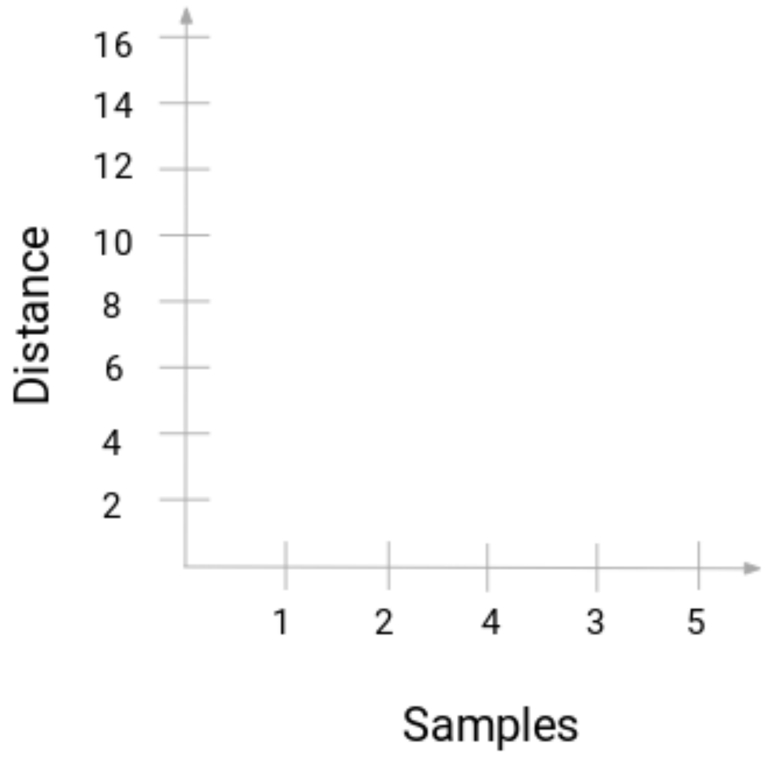


How should we Choose the Number of Clusters in Hierarchical Clustering?

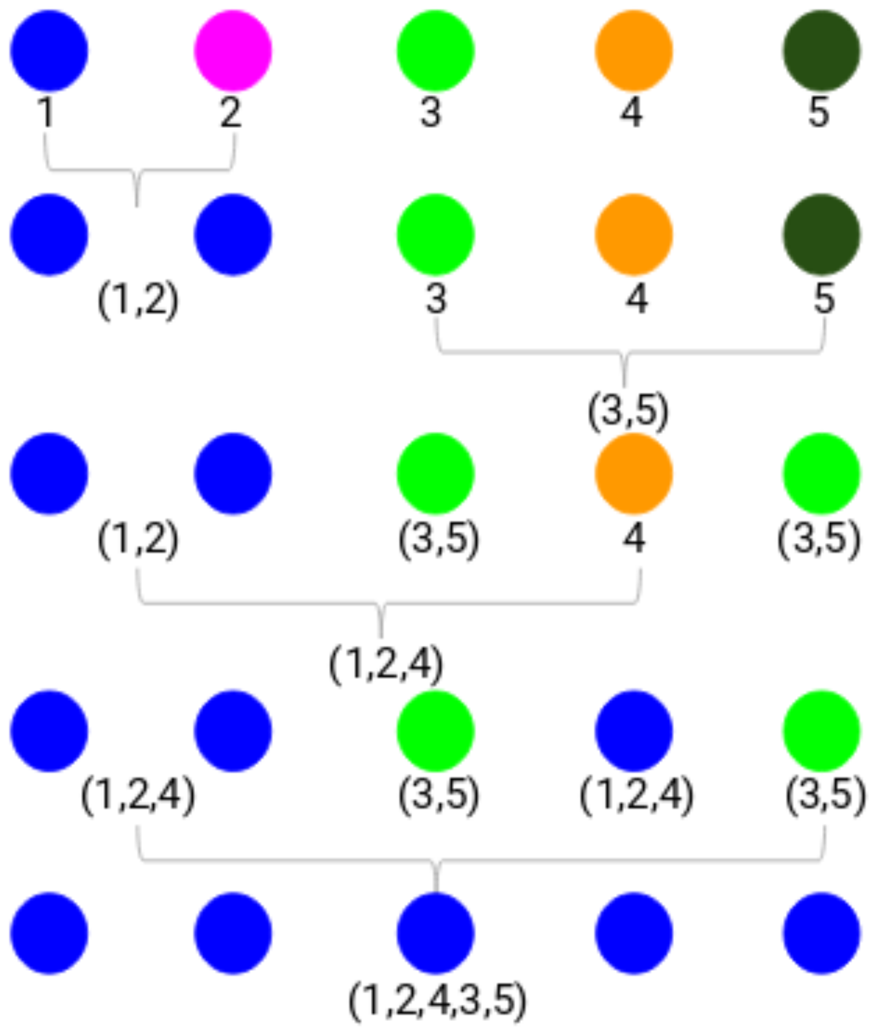
- To get the number of clusters for hierarchical clustering, we make use of an usefull concept called a Dendrogram.
  - It is a tree-like diagram that records the sequences of merges or splits.

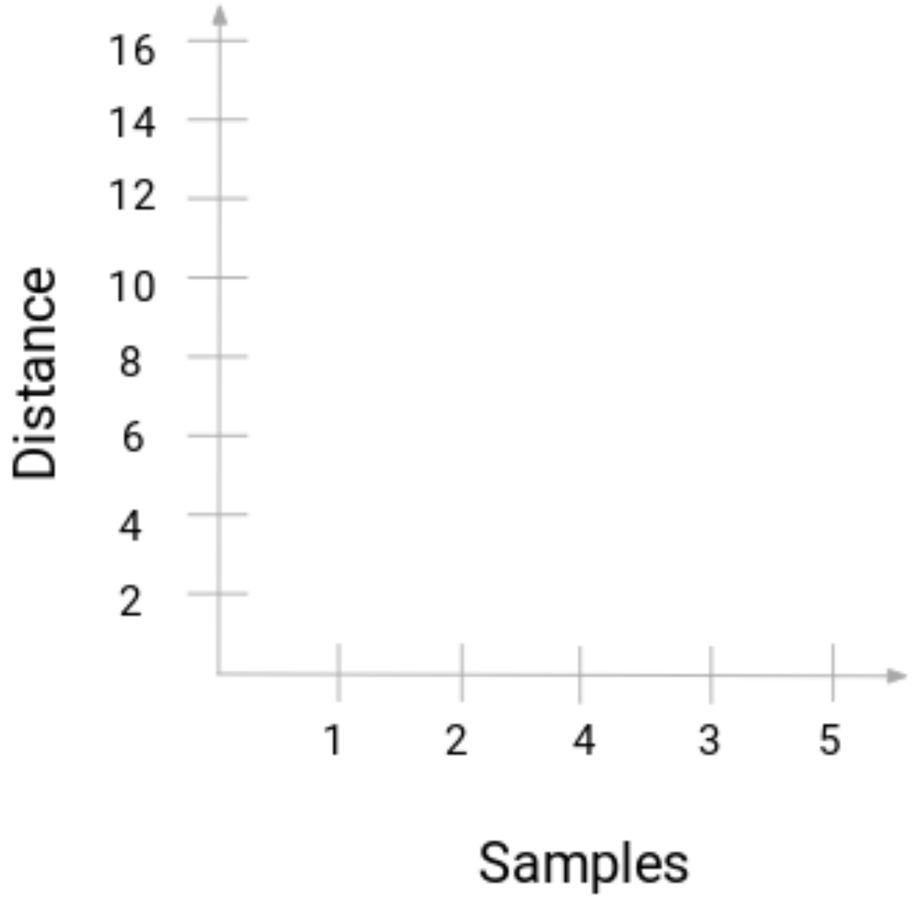
- X-axis: samples of the dataset
- Y-axis: distance between cluster

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

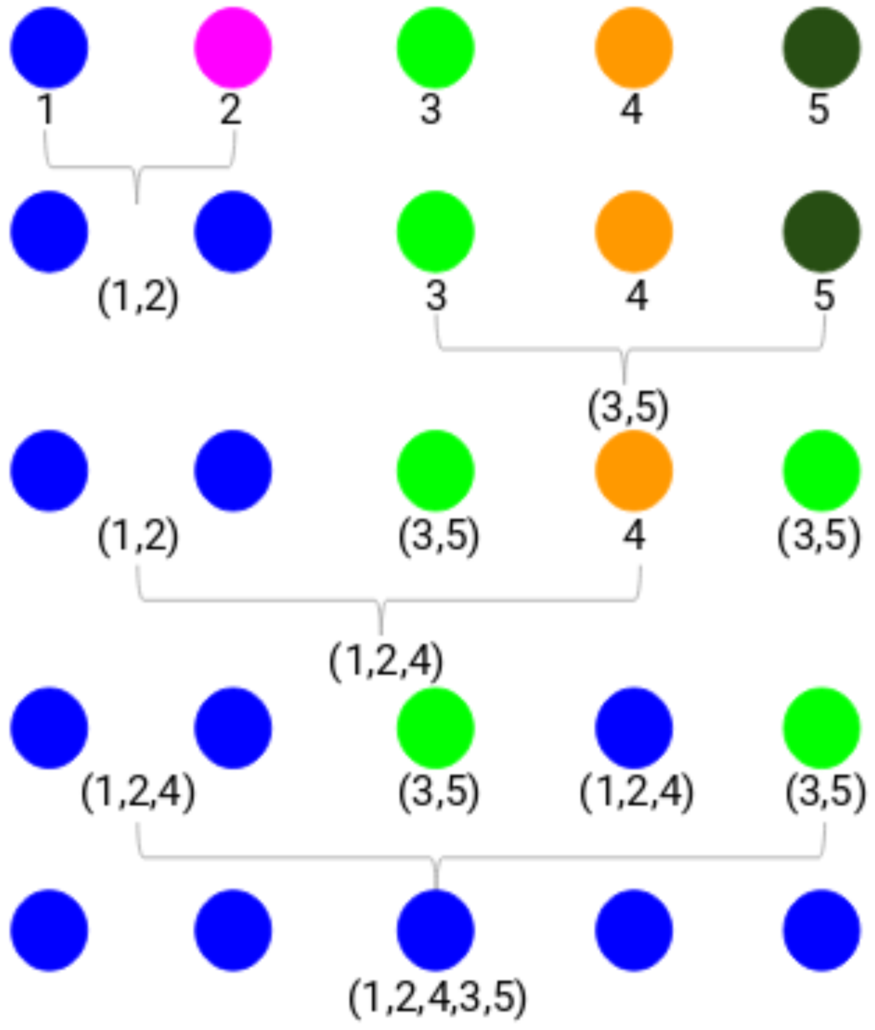


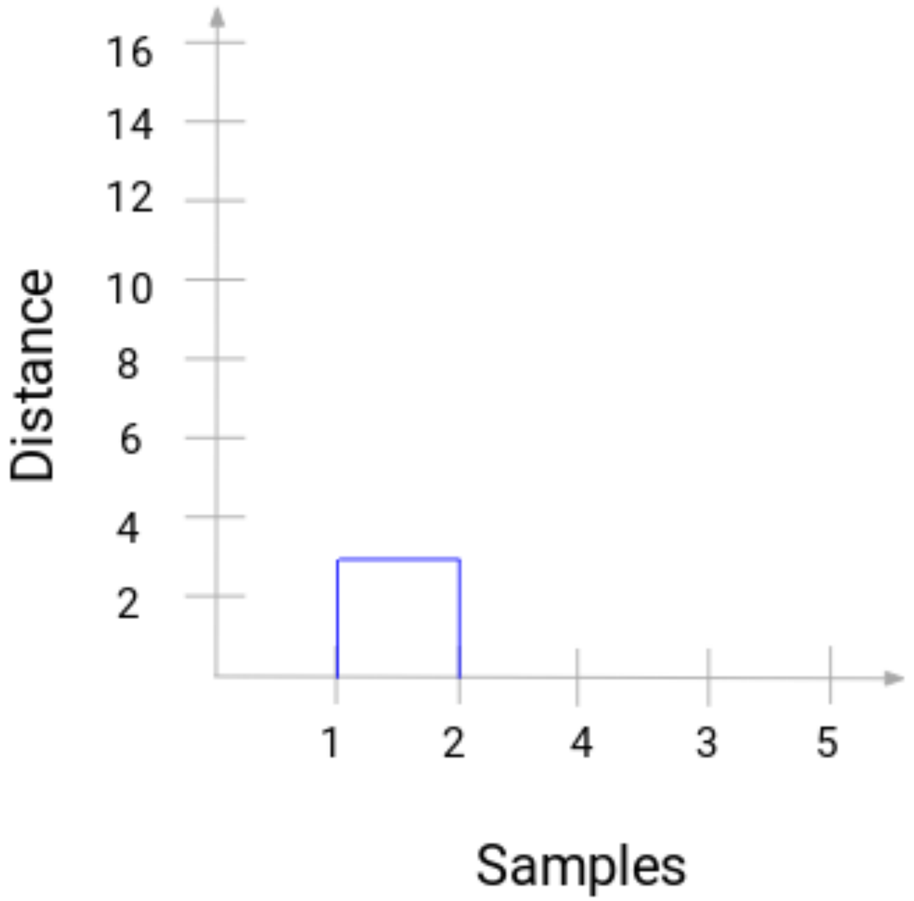
Dendogram representation



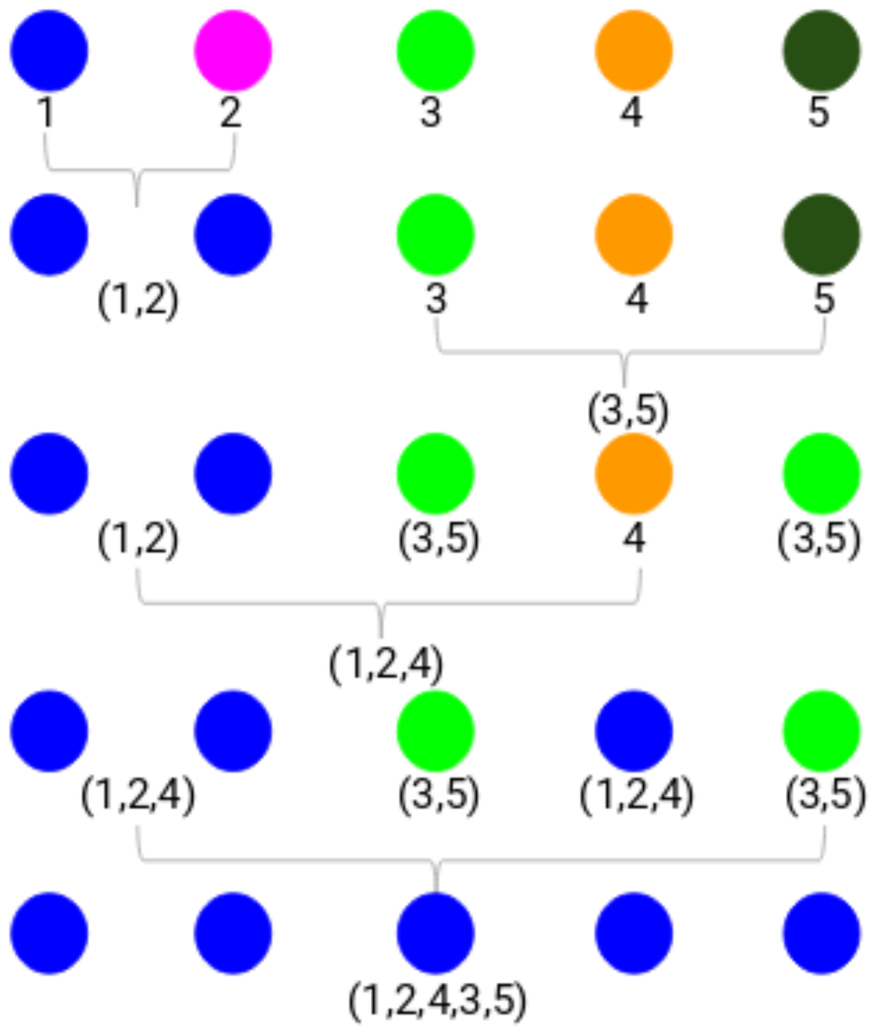


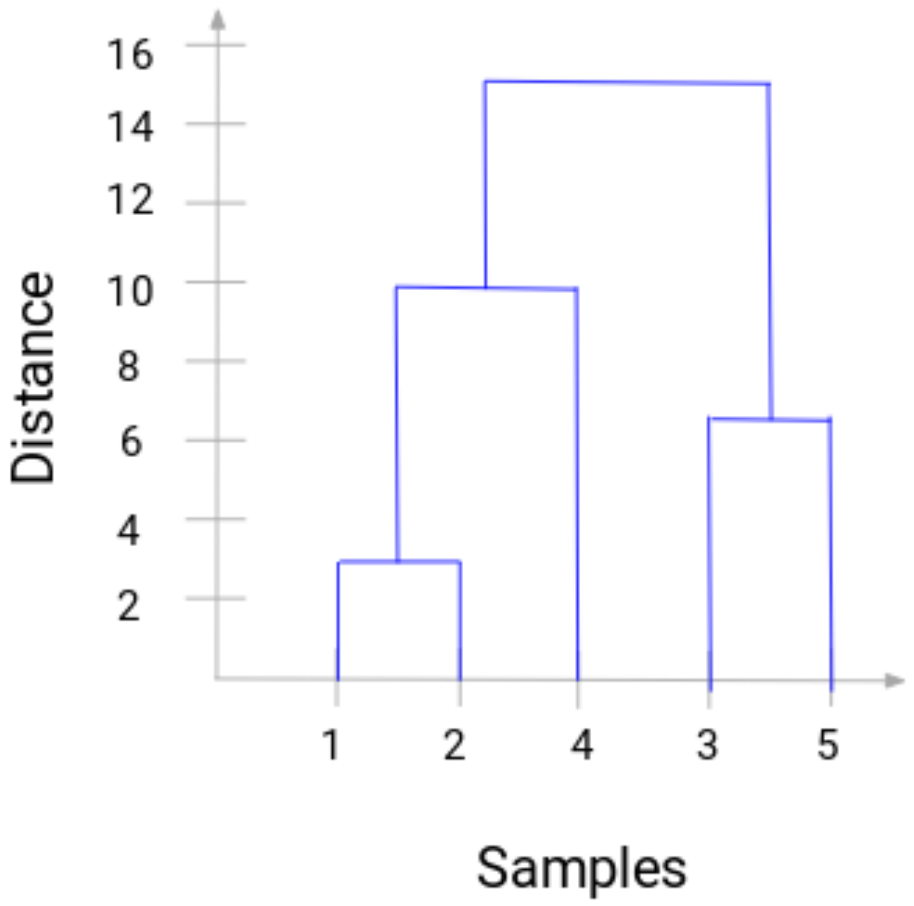
Dendogram representation



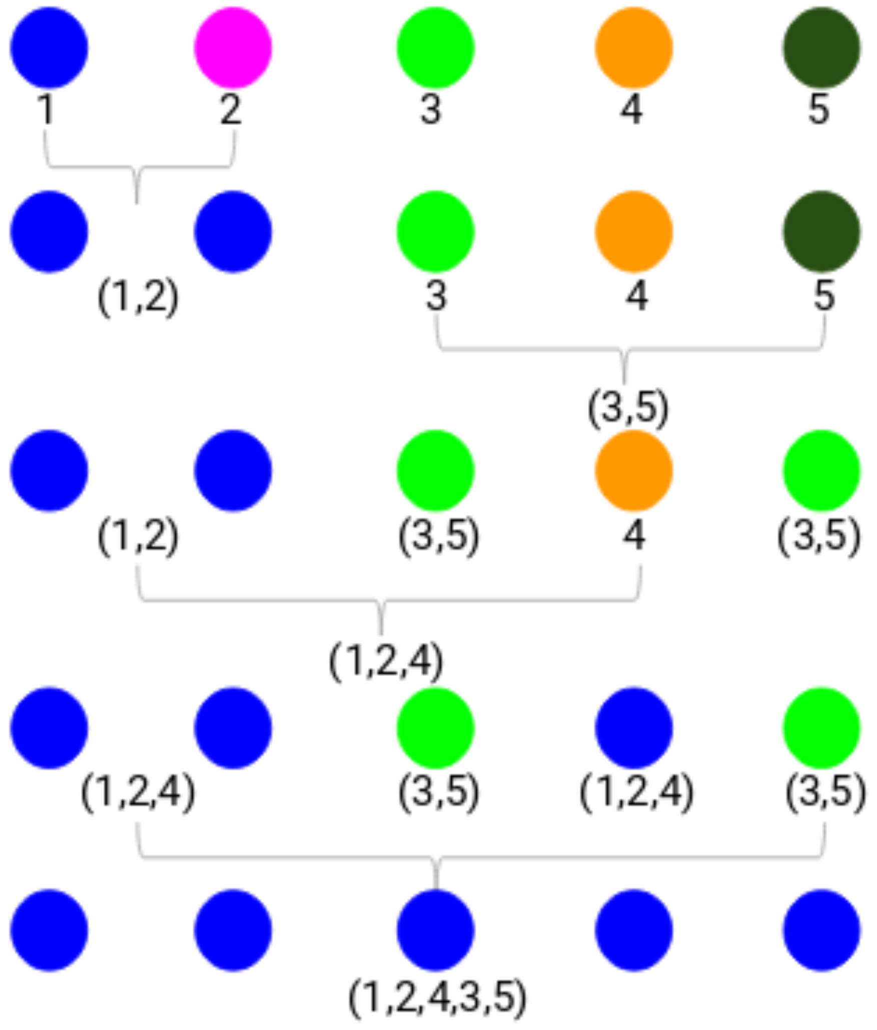


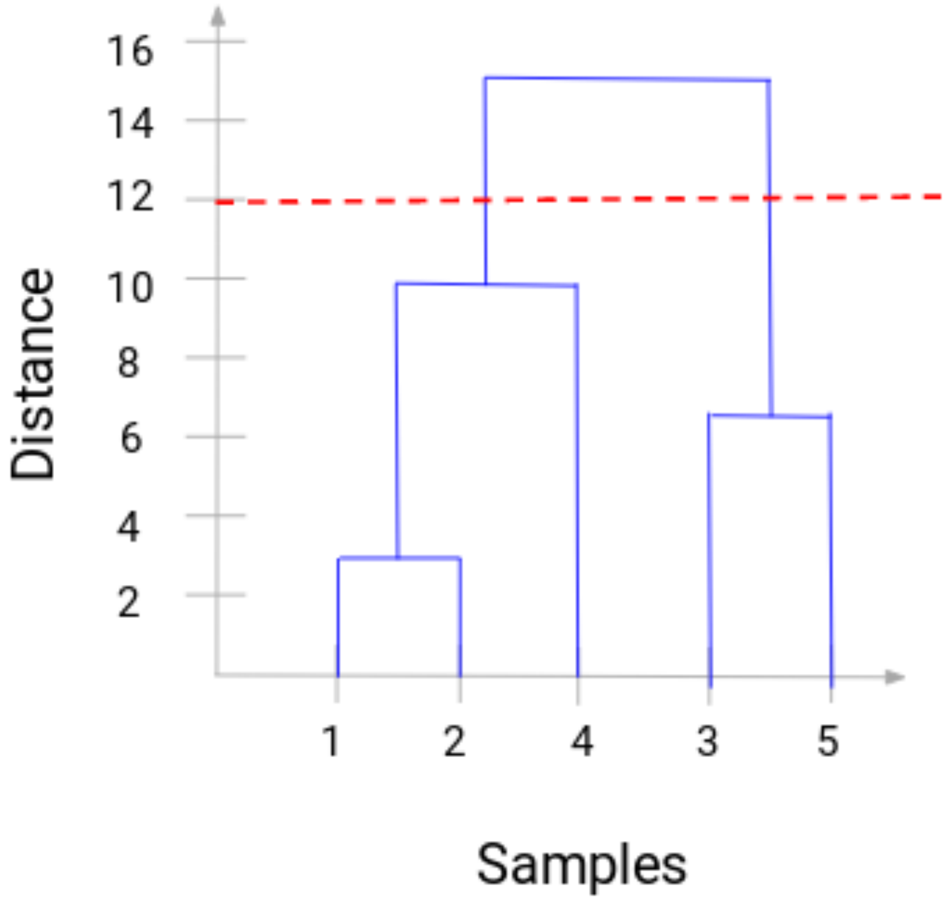
Dendogram representation





Dendogram representation

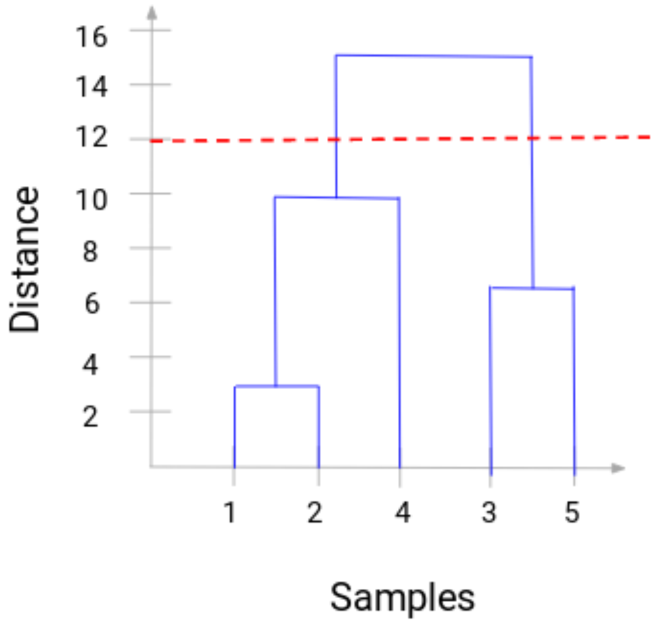




Dendogram representation

- 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.



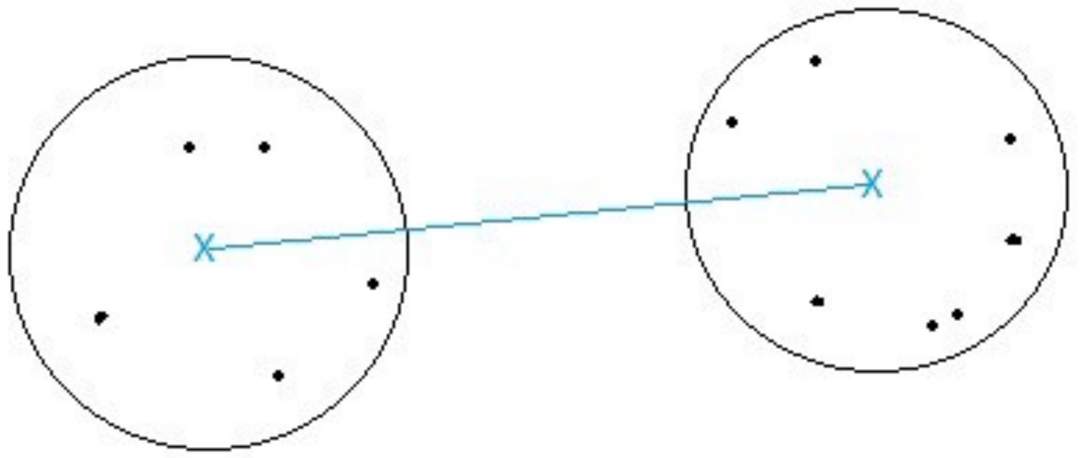
Hierarchical clustering

- 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

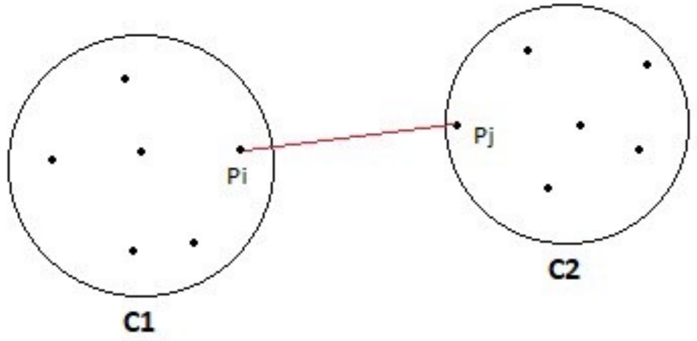
Distance between centroids

- 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

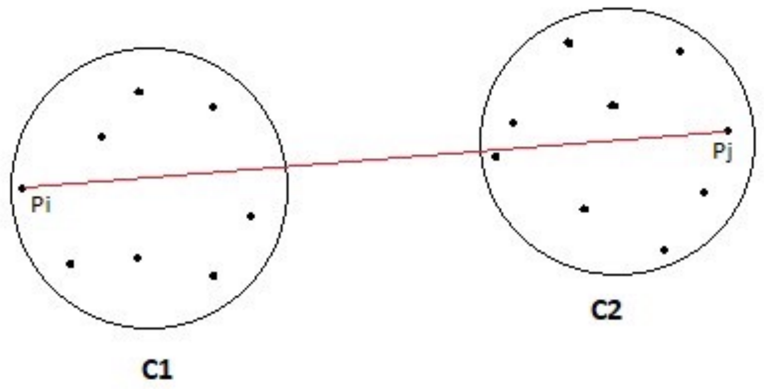
- Also known as single-linkage algorithm
- Sim(C1,C2) = Min Sim(P_i,P_j) such that $P_i \in C1$ & $P_j \in C2$
  - In simple words, pick the two closest points such that one point lies in cluster one and the other point lies in cluster 2 and takes their similarity and declares it as the similarity between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

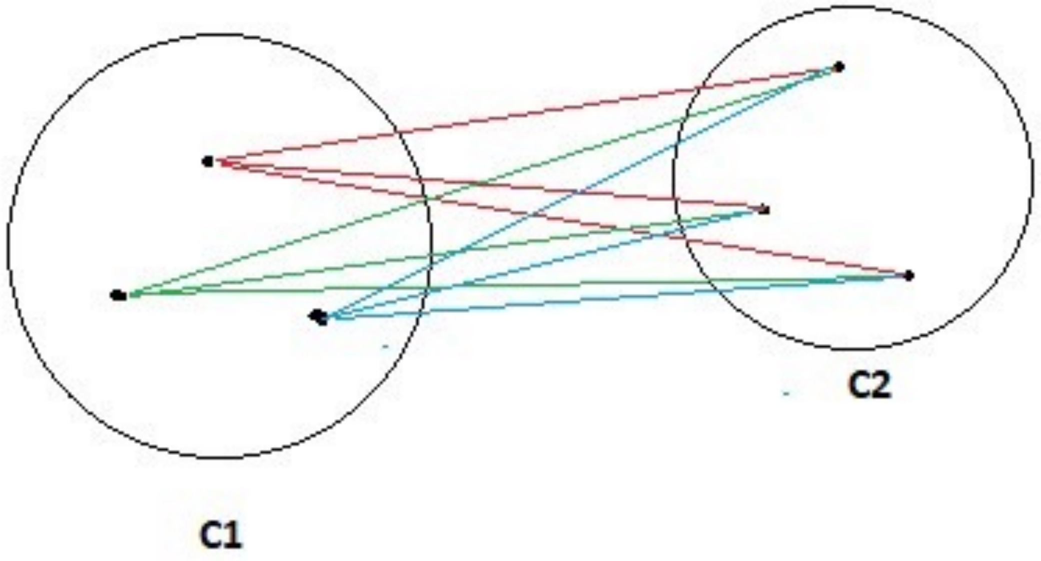
- It's the opposite of MIN
- Also known as complete-linkage algorithm
- Sim(C1,C2) = Max Sim(P_i,P_j) such that $P_i \in C1$ & $P_j \in C2$
  - In simple words, pick the two farthest points such that one point lies in cluster one and the other point lies in cluster 2 and takes their similarity and declares 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.
- $\mathrm{sim}(C_1, C_2) = \frac{\sum sim(P_i, P_j)}{|C1| * |C2|}$ 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

Density based clustering

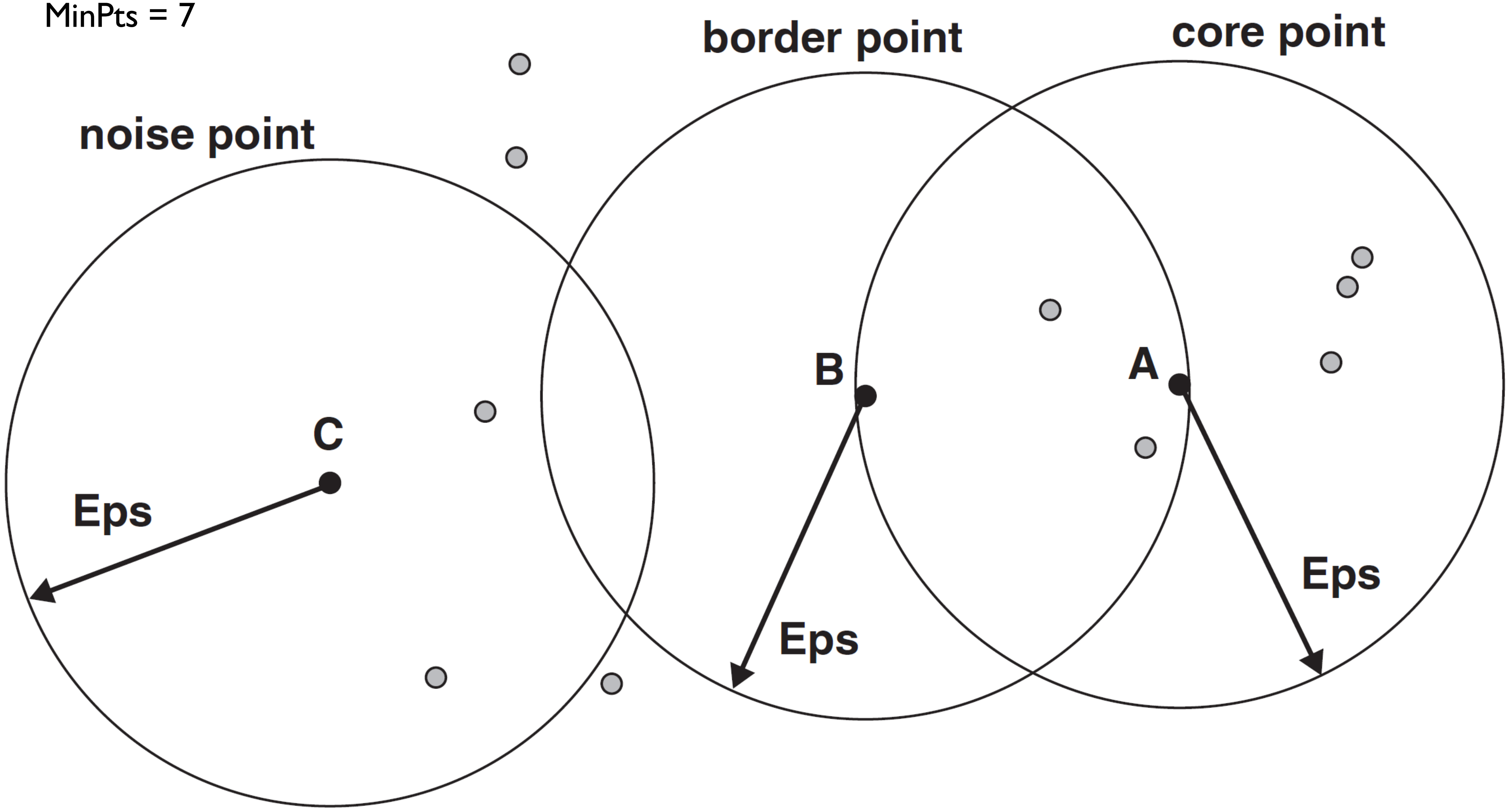
DBSCAN

- There are two key parameters of DBSCAN:
  - **eps**: The distance that specifies the neighborhoods. Two points are considered to be neighbors if the distance between them are less than or equal to eps.
  - **minPts:** Minimum number of data points to define a cluster.

- Based on these two parameters, points are classified as core point, border point, or outlier:
  - **Core point:** A point is a core point if there are at least minPts number of points (including the point itself) in its surrounding area with radius eps.
  - **Border point:** A point is a border point if it is reachable from a core point and there are less than minPts number of points within its surrounding area.
  - **Outlier:** A point is an outlier if it is not a core point and not reachable from any core points.

DBSCAN: Core, Border, and Noise Points

MinPts = 7

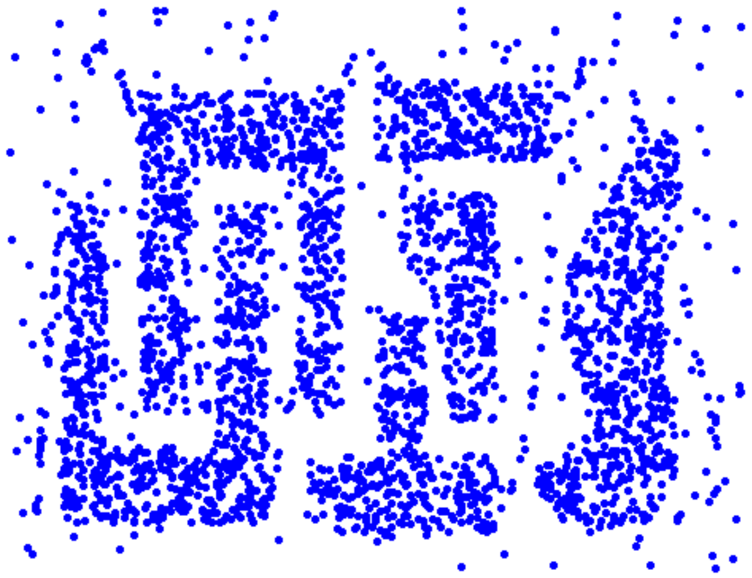


DBSCAN Algorithm

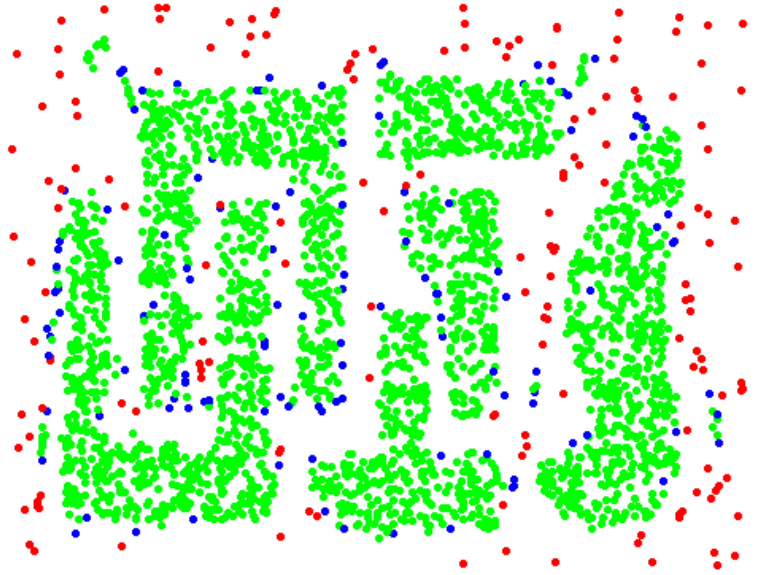
- Eliminate noise points
- Perform clustering on the remaining points

```
$current\_cluster\_label \leftarrow 1$
for all core points do
  if the core point has no cluster label then
    $current\_cluster\_label \leftarrow current\_cluster\_label + 1$
    Label the current core point with cluster label $current\_cluster\_label$
  end if
  for all points in the $Eps$-neighborhood, except $i^{th}$ the point itself do
    if the point does not have a cluster label then
      Label the point with cluster label $current\_cluster\_label$
    end if
  end for
end for
```

DBSCAN: Core, Border and Noise Points



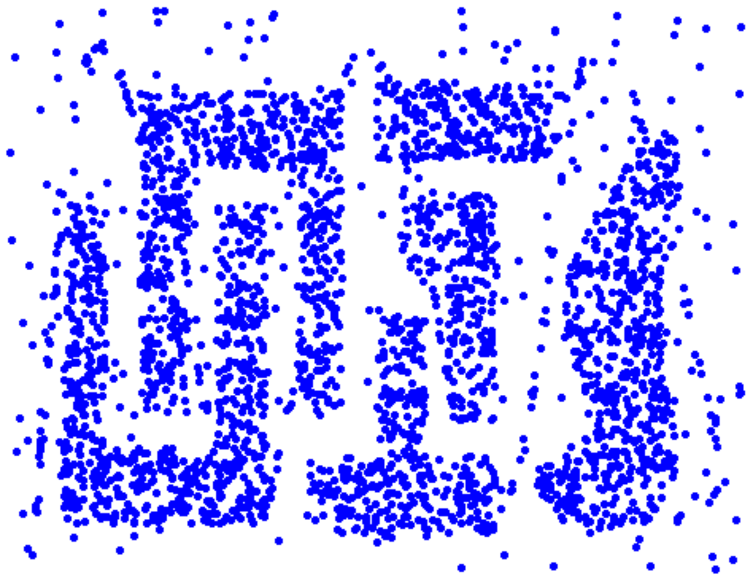
Original Points

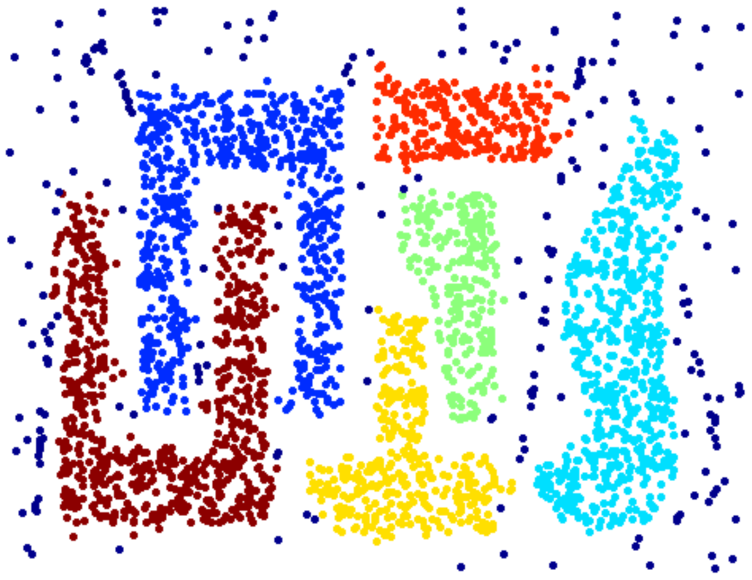


Point types: core, border and noise

Eps = 10, MinPts = 4

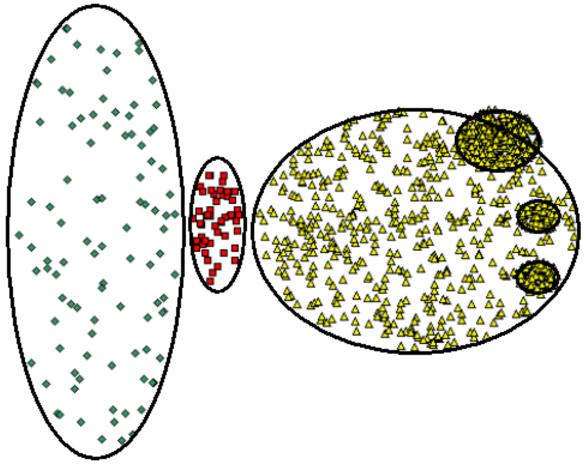
When DBSCAN Works Well


Original Points

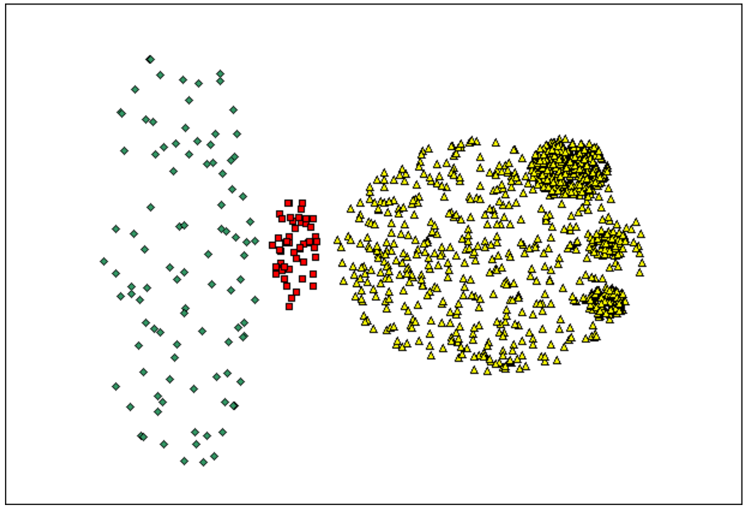

Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

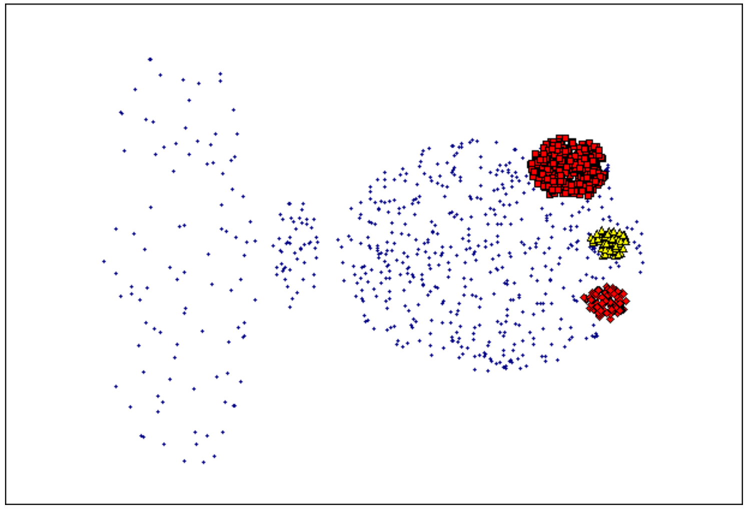
When DBSCAN Does NOT Work Well



Original Points



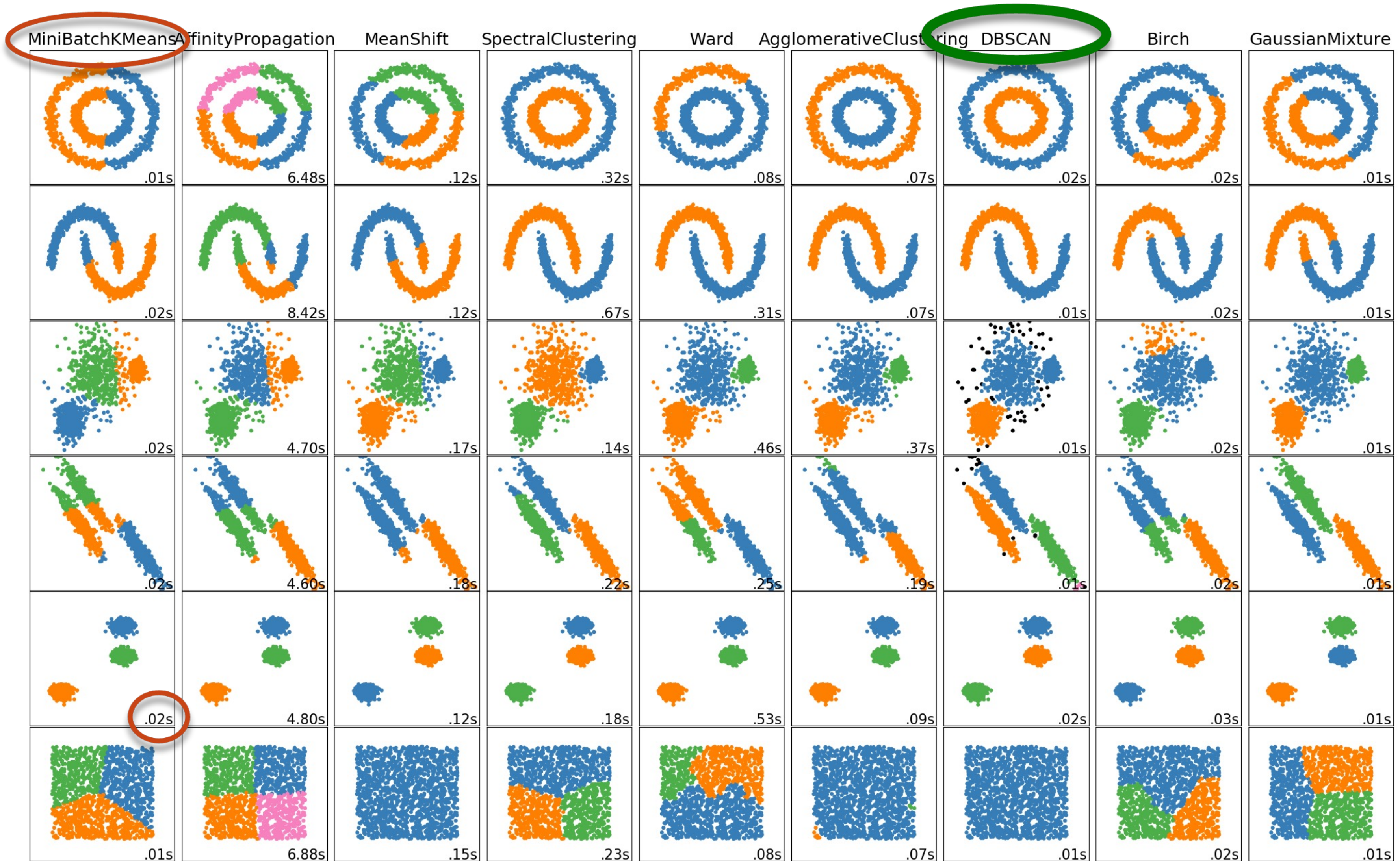
(MinPts=4, Eps=9.75).



(MinPts=4, Eps=9.92)

- **Varying densities**
- **High-dimensional data**

Others algorithms (try by yourself)



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

Clustering evaluation

If we **have the ground truth** class assignments

- The **adjusted Rand index** is a function that measures the similarity of the two assignments, ignoring permutations

```
from sklearn import metrics
metrics.adjusted_rand_score(labels_true, labels_pred)
```

- Advantages
  - Bounded range [-1, 1]:
    - Negative values are bad (independent labeling)
    - Similar clusterings have a positive score, 1.0 is the perfect match score.
  - No assumption is made on the cluster structure
- Drawbacks
  - Requires knowledge of the ground truth classes while is almost never available in practice

Clustering evaluation

If we **don't have the ground truth** class assignments

- 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)`

```
from sklearn import metrics
from sklearn.metrics import pairwise_distances
metrics.silhouette_score(X, pred_labels, metric='euclidean')
```

- **Advantages**
  - Score bounded between [-1, 1]
    - -1 incorrect clustering
    - +1 highly dense clustering
    - 0 indicates overlapping clusters.
  - The score is higher when clusters are dense and well separated
- **Drawbacks**
  - The Silhouette Coefficient is generally higher for convex clusters than other concepts of clusters, such as density based clusters like those obtained through DBSCAN.

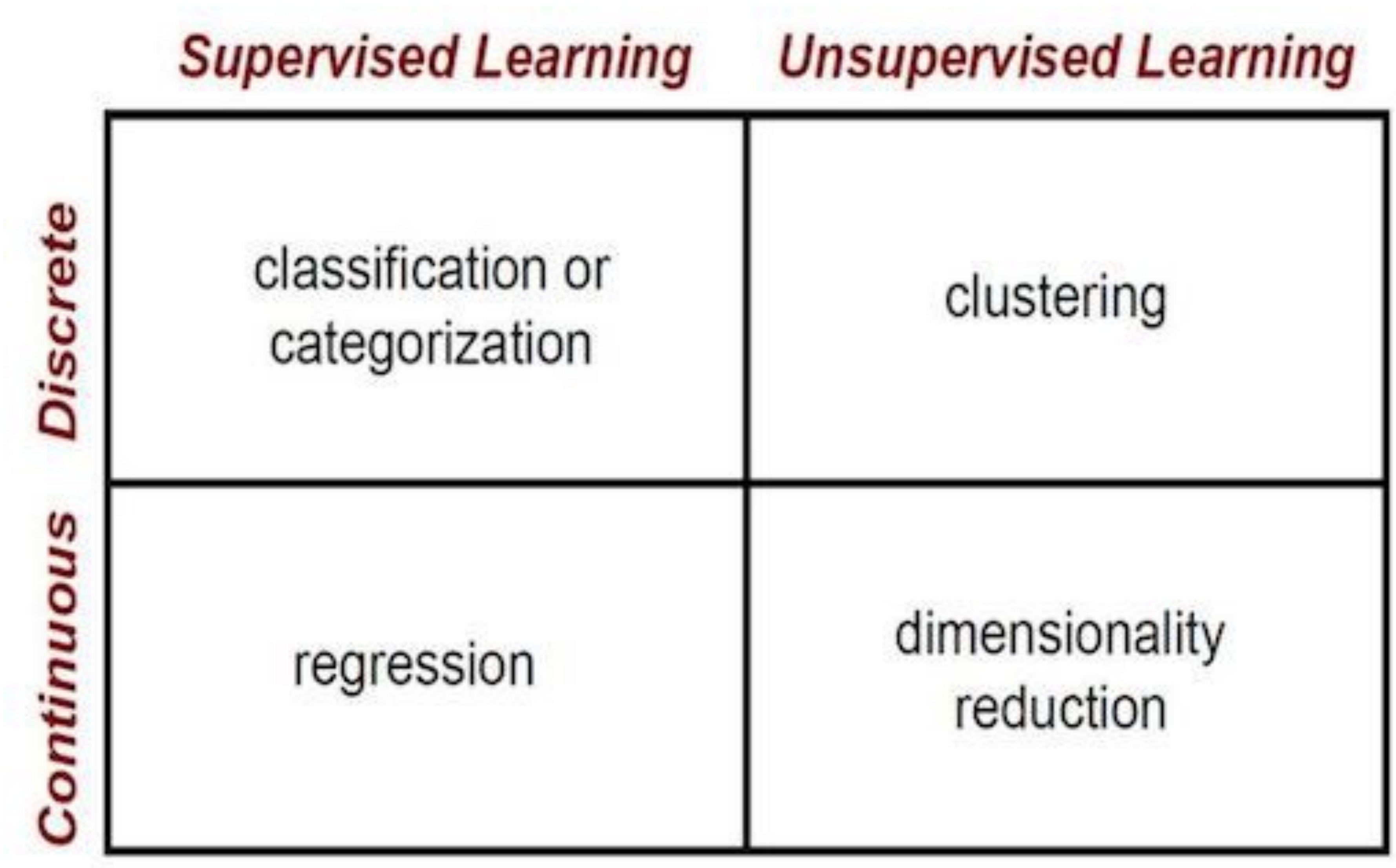
Summary: Unsupervised learning

- 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	classification or categorization	clustering
Continuous	regression	dimensionality reduction

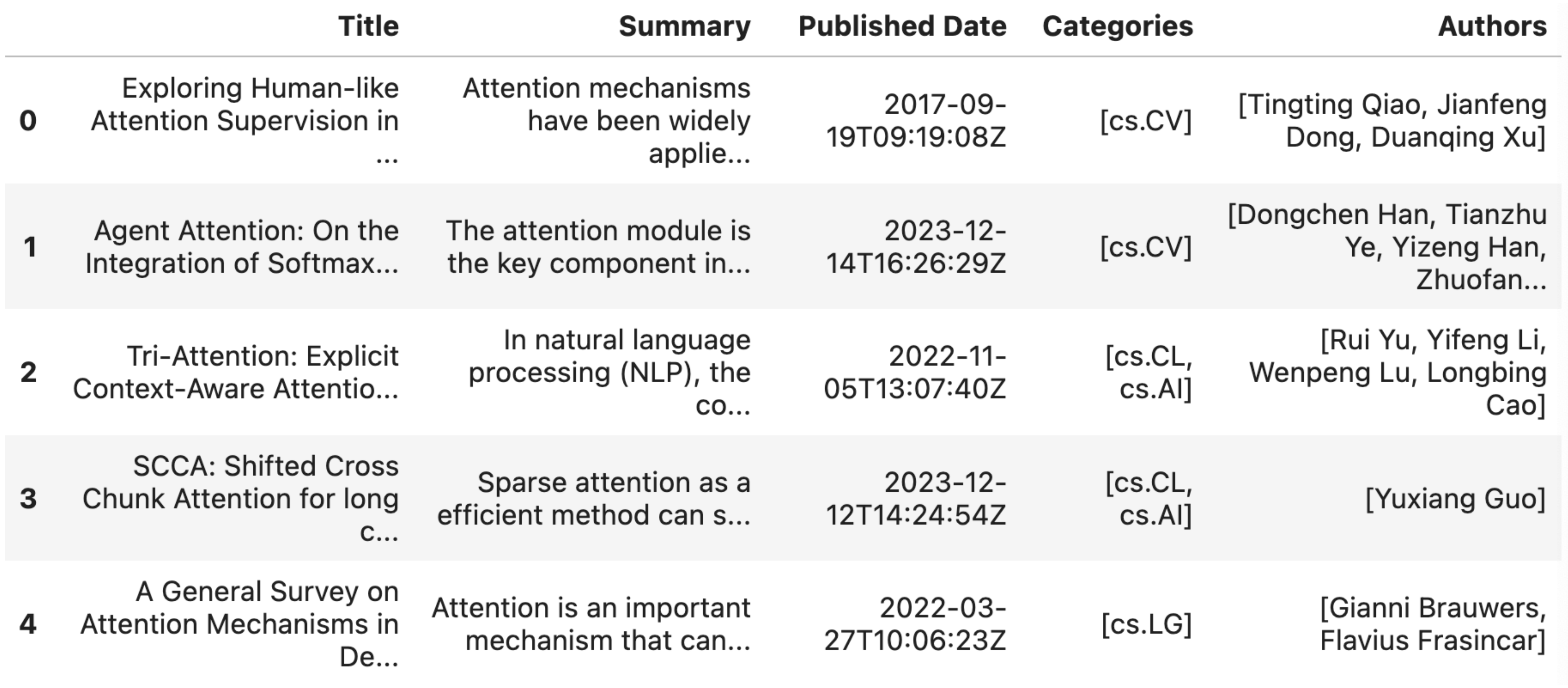
The lab of today – Text clusterisation

- When starting a research project on a given subject, it is advisable to carry out a bibliographical study
- Here's a suggestion for retrieving a set of documents and then using the content of their summaries to try and cluster them (clusterizer)

- It's a new subject… créé juste pour vous ☺

The lab of today – step 1

- Choose a topic related to Machine or Deep learning classes
- An using the code available on the notebook, download some documents (100 to 1000) from arxiv
- You obtain a DataFrame



	Title	Summary	Published Date	Categories	Authors
0	Exploring Human-like Attention Supervision in ...	Attention mechanisms have been widely applie...	2017-09-19T09:19:08Z	[cs.CV]	[Tingting Qiao, Jianfeng Dong, Duanqing Xu]
1	Agent Attention: On the Integration of Softmax...	The attention module is the key component in...	2023-12-14T16:26:29Z	[cs.CV]	[Dongchen Han, Tianzhu Ye, Yizeng Han, Zhuofan...
2	Tri-Attention: Explicit Context-Aware Attentio...	In natural language processing (NLP), the co...	2022-11-05T13:07:40Z	[cs.CL, cs.AI]	[Rui Yu, Yifeng Li, Wenpeng Lu, Longbing Cao]
3	SCCA: Shifted Cross Chunk Attention for long c...	Sparse attention as a efficient method can s...	2023-12-12T14:24:54Z	[cs.CL, cs.AI]	[Yuxiang Guo]
4	A General Survey on Attention Mechanisms in De...	Attention is an important mechanism that can...	2022-03-27T10:06:23Z	[cs.LG]	[Gianni Brauwers, Flavius Frasincar]

The lab of today – Step 2

- Clean the summary
  - Keep only nouns, proper nouns and adjectives
    - POS Tag: ["PROPN", "NOUN, "ADJ"]
  - Put words in lower case
  - Delete stop words
  - Applying a lemmatization (not a stemming)
- Piece of code

```
nlp = spacy.load('en_core_web_sm')
a = nlp("The attention module is one the key components in")
for d in a:
    print(d.text, d.pos_, d.lemma_)
```

```
>>> The DET the
>>> attention NOUN attention
>>> module NOUN module
>>> is AUX be
>>> one NUM one
>>> the DET the
>>> key ADJ key
>>> components NOUN component
>>> in ADP in
```

Select only

The lab of today - Step 3

- Feature extraction
  - Use all your background and be curious
- Use DTM approach with 1 and 2 gram, select less than 100 words
  - Binary
  - Count
  - TfIDF
- Or use an vectorized approach to build an embedding for the summary
  - Word2Vec
  - FastText
  - BERT (transformer)
  - SentenceBERT (transformer)

The lab of today - Step 3

- Clustering

- Use the previous embedding to group documents with one of the following algorithm (start by one and try others later)
  - K-means
  - K-medoid
  - DBSCAN
  - Hierarchical clustering

- Find the optimal number of cluster

The lab of today - Step 4

- Exploit the result:
  - Which document best represents the cluster?
    - Depending on the algorithm, this is more or less easy to find
      - K-medoid: the medoid is the document representing the centre of the cluster
      - K-means: you need to search for the document that is closest to the centre
        - by distance or similarity
      - DBSCAN or Hierarchical clustering:
        - No centroid
        - But: https://stackoverflow.com/questions/62215910/how-to-get-the-centroids-in-dbscan-sklearn
        - The centroids can be outside the cluster if it is not convex
  - For each cluster
    - What is the central document?
    - What is the list of documents?
    - Which is the oldest document
    - Which is the most recent document
  - What is the topics of each cluster?

The lab of today - Step 5

- Plot the interaction graph
  - Choose a cluster
  - Select the 20, 30 most frequent words inside the cleaned summary of all documents affected to this cluster
    - Create a node correspond for each most frequent words
  - For each document affected to this cluster
    - Create an edge between 2 nodes/words if the 2 words appear in the summary of the document
- You can use nx library in order to create the graph

```
import networkx as nx
G = nx.Graph() # Create an empty graph
G.add_nodes_from(cluster_terms) # Create a list of node from a list
G.add_edge(word1, word2) # Create an edge between word1 and word2
```

- Plot your graph

```
layout = nx.spring_layout(G)
nx.draw(G, layout, with_labels=True, node_size=10000,
              font_size=10, font_color="white")
```

Some reading

- https://towardsdatascience.com/from-data-to-clusters-when-is-your-clustering-good-enough-5895440a978a

Some complements to help you revise