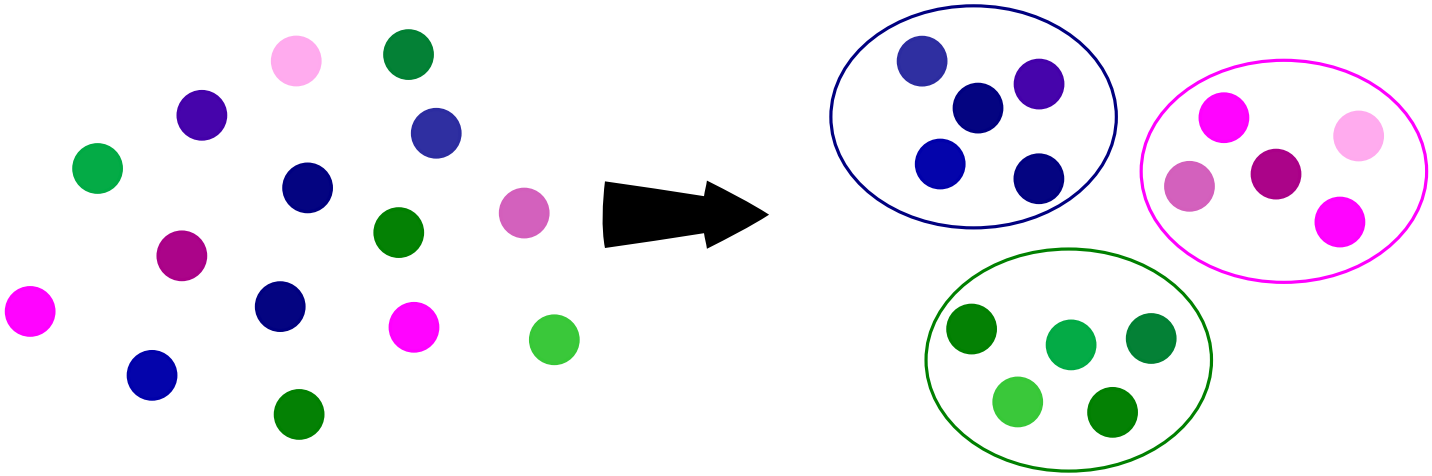
Clustering algorithms (*Algorithmes de partitionnement*)

Diane Lingrand and many contributors

Polytech SI

2022 - 2023

- **Unsupervised learning :** no class label needed.
- Find the class labels directly from the data.



- Cluster : a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
  - Need a way to calculate object similarity/distance
- Typical applications :
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

Examples of Clustering Applications

UNIVERSITÉ CÔTE D'AZUR

- Marketing : Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Land use : Identification of areas of similar land use in an earth observation database
- Insurance : Identifying groups of motor insurance policy holders with a high average claim cost
- City-planning : Identifying groups of houses according to their house type, value, and geographical location
- web : Cluster Web log data to discover groups of similar access patterns
- Earth-quake studies : Observed earth quake epicenters should be clustered along continent faults
- Representation of data (bag of words/features)

What Is Good Clustering ?

- A good clustering method will produce high quality clusters with
  - high intra-class similarity
  - low inter-class similarity
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation
- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns

Outline

Similarity : an example

- images from six classes :
  - muffin,
  - bagel,
  - fried chicken,
  - chihuahua,
  - labradoodle and
  - puppy
- How would you do a binary clustering of theses images ?

Similarity : an example

- images from six classes :
  - muffin,
  - bagel,
  - fried chicken,
  - chihuahua,
  - labradoodle and
  - puppy
- How would you do a binary clustering of theses images ?
  - food : muffin, bagel and fried chicken
  - pet : chihuahua, labradoodle and puppy

Similarity : an example

- food : muffin, bagel and fried chicken
- pet : chihuahua, labradoodle and puppy

Choosing the distance is of crucial importance !

Interval-scaled variables

- Standardize data
  - Calculate the mean absolute deviation : $s_j = \frac{1}{n} \sum_{i=1}^{n} |x_{ij} - \mu_j|$ where

$$\mu_j = \frac{1}{n} \sum_{i=1}^{n} x_{ij}$$

  - Calculate the standardized measurement (z-score) : $z_{ij} = \frac{x_{ij} - \mu_j}{s_j}$
- Using mean absolute deviation is more robust than using standard deviation
- A classical example ($\mu_{age} = 60$; $\mu_{salary} = 11074$; $s_{age} = 5$; $s_{salary} = 37$) :

	age	salary
alice	50	11000
bob	70	11100
charly	60	11122
dany	60	11074

$\Rightarrow$

	age	salary
alice	-2	-2
bob	2	0.7
charly	0	1.3
dany	0	0

Why standardizing data ?

	age	salary
alice	50	11000
bob	70	11100
charly	60	11122
dany	60	11074

Manhattan distance	alice	bob	charly	dany
alice	0	120	132	84
bob	120	0	32	36
charly	132	32	0	48
dany	84	36	48	0

Why standardizing data ?

	age	salary
alice	50	11000
bob	70	11100
charly	60	11122
dany	60	11074

	age	salary
alice	-2	-2
bob	2	0.7
charly	0	1.3
dany	0	0

Manhattan distance	alice	bob	charly	dany
alice	0	120	132	84
bob	120	0	32	36
charly	132	32	0	48
dany	84	36	48	0

Manhattan distance	alice	bob	charly	dany
alice	0	6.7	5.3	4
bob	6.7	0	2.6	2.7
charly	5.3	2.6	0	1.3
dany	4	2.7	1.3	0

Why standardizing data ?

	age	salary
alice	50	11000
bob	70	11100
charly	60	11122
dany	60	11074

	age	salary
alice	-2	-2
bob	2	0.7
charly	0	1.3
dany	0	0

Manhattan distance	alice	bob	charly	dany
alice	0	120	132	84
bob	120	0	32	36
charly	132	32	0	48
dany	84	36	48	0

Manhattan distance	alice	bob	charly	dany
alice	0	6.7	5.3	4
bob	6.7	0	2.6	2.7
charly	5.3	2.6	0	1.3
dany	4	2.7	1.3	0

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
Xscaled = scaler.fit_transform(X)
```

- Minkowski distance

$$d_q(i,j) = \sqrt[q]{\sum_{k=1}^{m} |x_{ik} - x_{jk}|^q}$$

- if $q = 1$: Manhattan distance

$$d_1(i,j) = \sum_{k=1}^{m} |x_{ik} - x_{jk}|$$

- if $q = 2$: Euclidean distance

$$d_2(i,j) = \sqrt{\sum_{k=1}^{m} |x_{ik} - x_{jk}|^2}$$

- if $q = \infty$: Max. distance

Typical alternatives for the distance between clusters

- Single linkage : smallest distance between an element in one cluster and an element in the other, i.e., $d(K_p, K_q) = \min_{x_i \in K_p, x_j \in K_q} d(x_i, x_j)$
- Complete linkage : largest distance between an element in one cluster and an element in the other, i.e., $d(K_p, K_q) = \max_{x_i \in K_p, x_j \in K_q} d(x_i, x_j)$
- Average linkage : average distance between an element in one cluster and an element in the other
- Centroid : distance between the centroids of two clusters, i.e., $d(K_p, K_q) = d(C_p, C_q)$
  - centroid = midpoint of a cluster

$$C_p = \frac{1}{N_p} \sum_{i=1}^{N_p} x_i^{(p)}$$

- Medoid : distance between the medoids of two clusters, i.e., $d(K_p, K_q) = d(M_p, M_q)$
  - Medoid : one chosen, centrally located object in the cluster

Major Clustering Approaches

- Partitioning (iterative construction of partitions)
  - K-Means, k-Medoids, etc.
- Hierarchical (construct a dendrogram of instances)
  - Diana, Agnes, BIRCH, ROCK, CAMELEON
- Density-Based (based on connectivity and density function)
  - DBSCAN, OPTICS, DenClue
- Grid-Based
  - STING, WaveCluster, CLIQUE
- Model-Based
  - Expectation Maximization
  - Self-organizing maps (SOM)
- Spectral clustering
- Frequent-Pattern-Based

Outline

- Partitioning method : Construct a partition of a dataset D of n objects into a set of k clusters, minimizing the sum of squared distances :

$$\sum_{p=1}^{k}\sum_{i=1}^{N_p}(x_i^{(p)} - C_p)^2$$

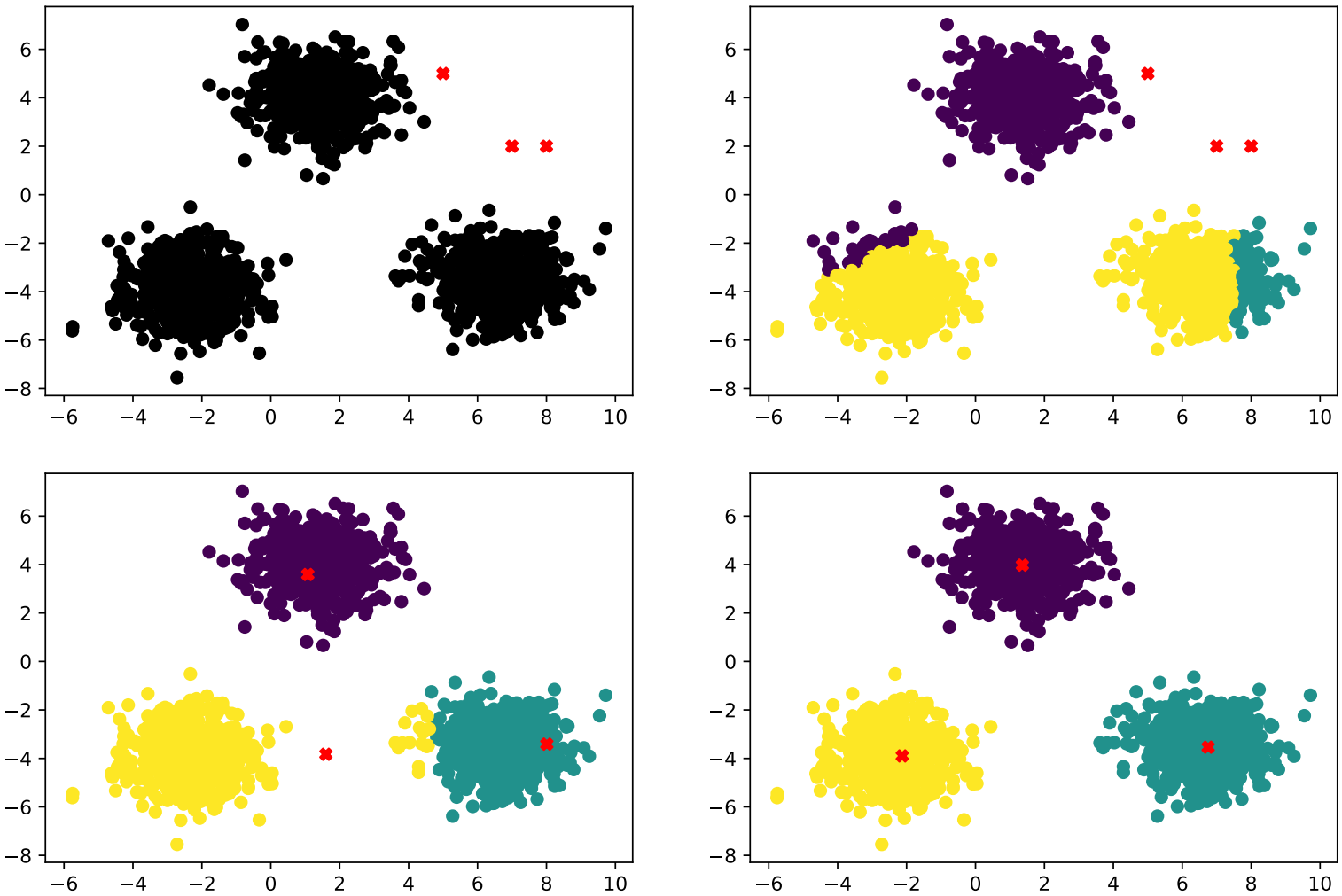
- Given a k, find a partition of k clusters that optimizes the chosen partitioning criterion
  - Global optimal : exhaustively enumerate all partitions
  - Heuristic methods : k-means and k-medoids algorithms
    - k-means (MacQueen '67) : Each cluster is represented by the centroid of the cluster
    - k-medoids or PAM (Partition around medoids) (Kaufman and Rousseeuw '87) : Each cluster is represented by one of the objects in the cluster

The k-means clustering algorithm

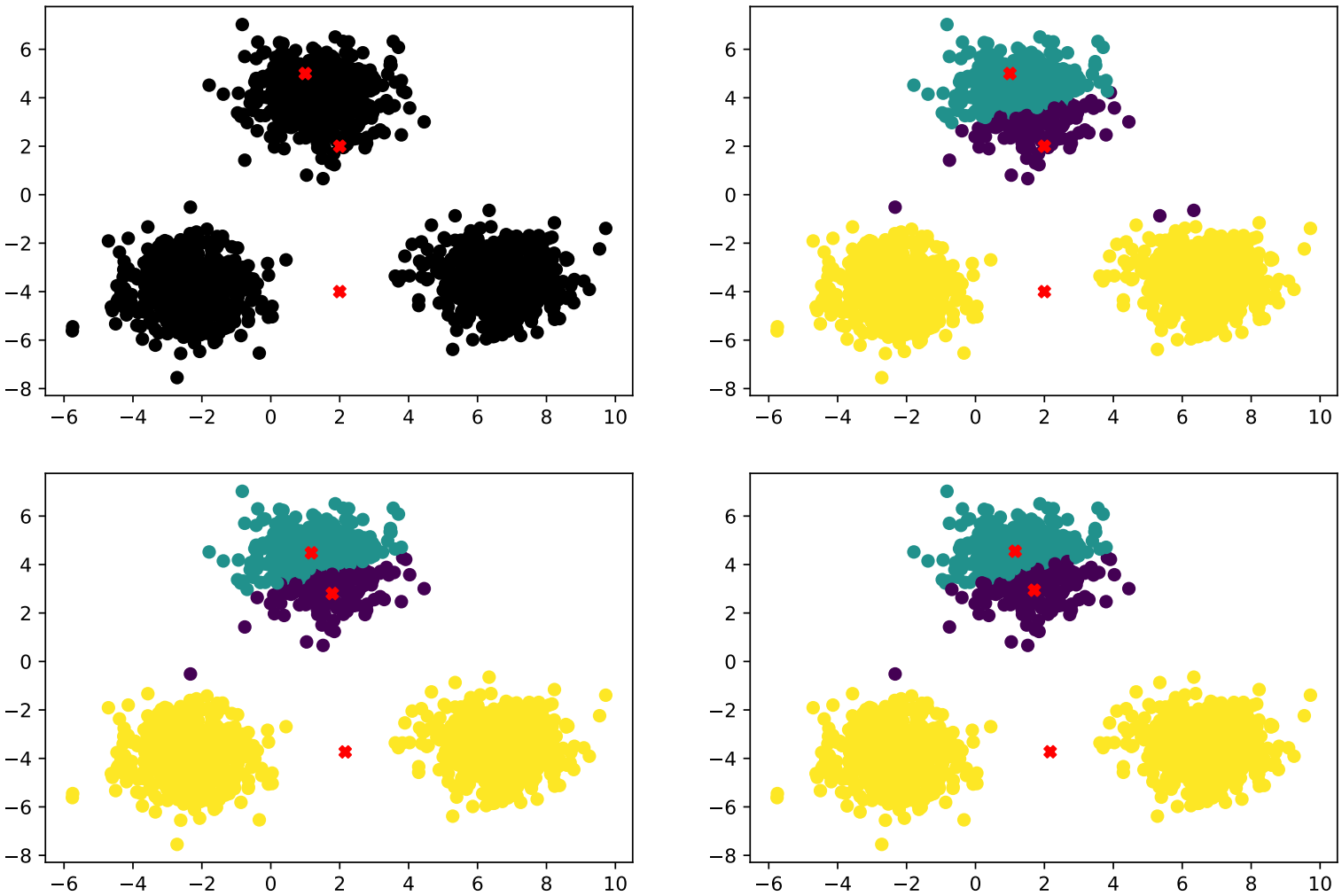
Given k, the k-means algorithm is implemented in four steps :

1. Choose k initial centroids (could be samples), each labeled as k
2. Assign each object to the cluster having the nearest centroid point
3. Compute centroids of the clusters of the current partition (the centroid is the center, i.e., mean point, of the cluster)
4. Go back to Step 2, stop when no more change

Example of k-means



Example of k-means : bad initialisation

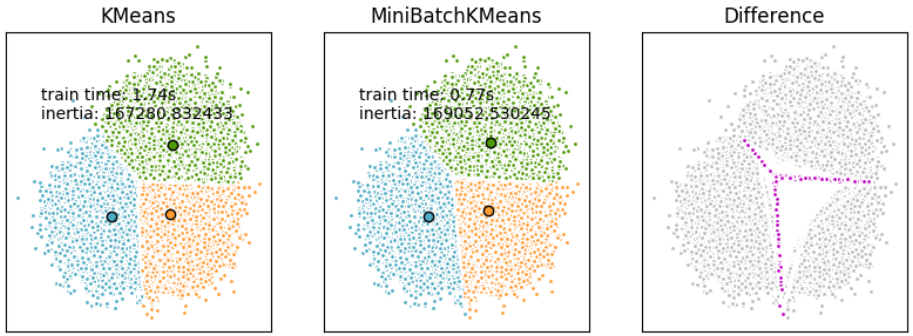


k-mean advantages and drawbacks

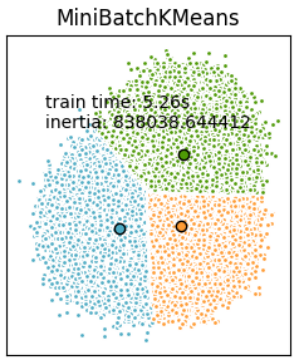
- Strength : Relatively efficient : $O(tkn)$, where n nb. of objects, k nb. of clusters, and t nb. of iterations. Normally, $k, t << n$.
  - For comparison : PAM : $O(k(n–k)^2)$, CLARA : $O(ks^2 + k(n–k))$
- Comment : Often terminates at a local optimum.
  - The global optimum may be found using optimization methods such as : simulated annealing and evolutionary algorithms.
  - In practice : several trials using aleatory initialisations and some heuristics.
- Weaknesses :
  - Applicable only when mean is defined, then what about categorical data ?
  - Need to specify k, the number of clusters, in advance
    - or use the Elbow method
  - Unable to handle noisy data and outliers
  - Not suitable to discover clusters with non-convex shapes

mini-batch k-mean

- a variant of k-mean for scalability improving
- mini-batch = subset of dataset
  - randomly sampled at each iteration of the training process
- results almost as good as the standard algorithm
  - code from `https://scikit-learn.org/stable/auto_examples/cluster/plot_mini_batch_kmeans.html#sphx-glr-auto-examples-cluster-plot-mini-batch-kmeans-py` with modification of the number of data to 200000 (upper limit to have k-means running).

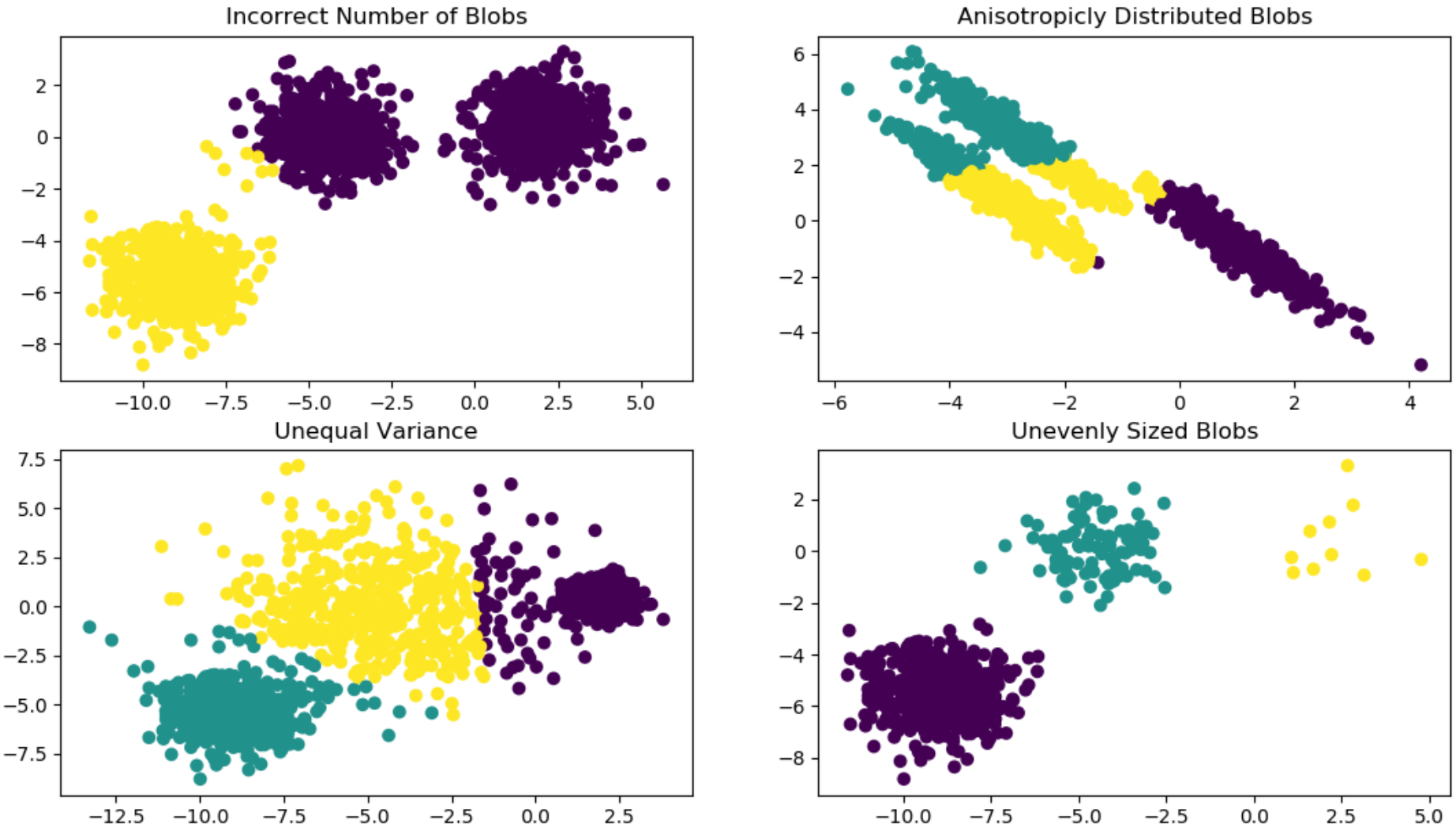


  - Only with mini-batch and using 1 million of data :



Demonstration of k-means assumptions

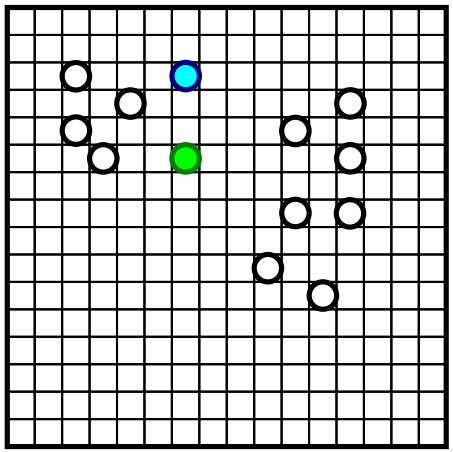
https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_assumptions.html



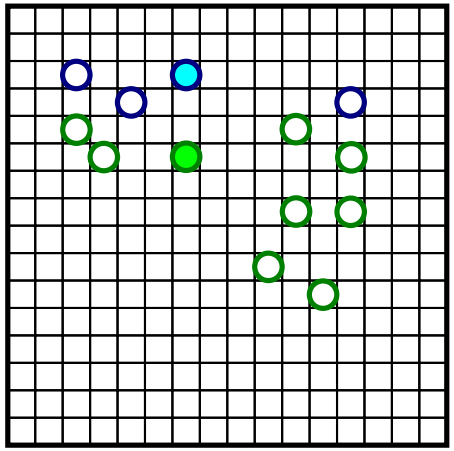
The k-Medoids Clustering Method

- Find representative objects, called medoids, in clusters
- PAM (Partitioning Around Medoids, 1987)
  - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
  - less sensitive to outliers than k-mean
  - PAM works effectively for small data sets, but does not scale well for large data sets
- CLARA (Clustering LARge Applications, Kaufmann & Rousseeuw, 1990) : PAM on several subsets of samples (take the best clustering)
- CLARANS (Clustering Large Applications based upon RANdomized Search, Ng & Han, 1994) : Randomized sampling. Graph of solutions (=set of centroids).

k-medoids : an example

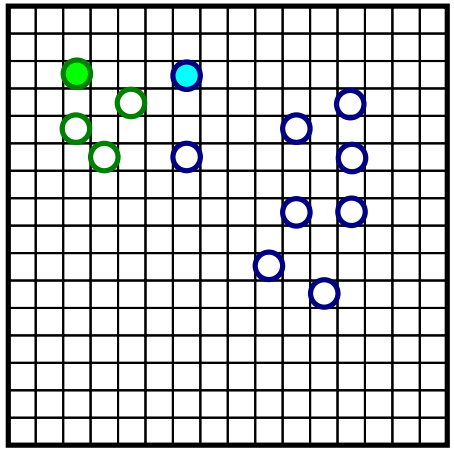


initialisation
choose 2 samples



label all data
compute the cost:
14+50=64

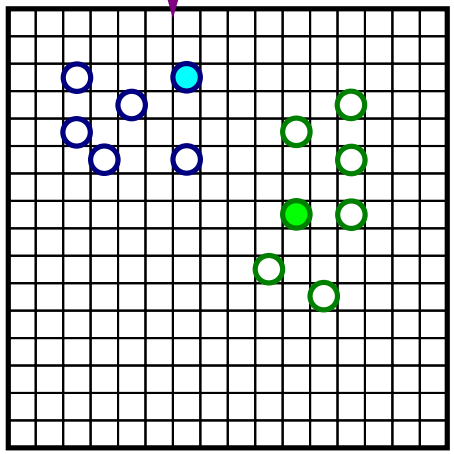
try to change
a medoid



new cost: 9+68=77 > 64

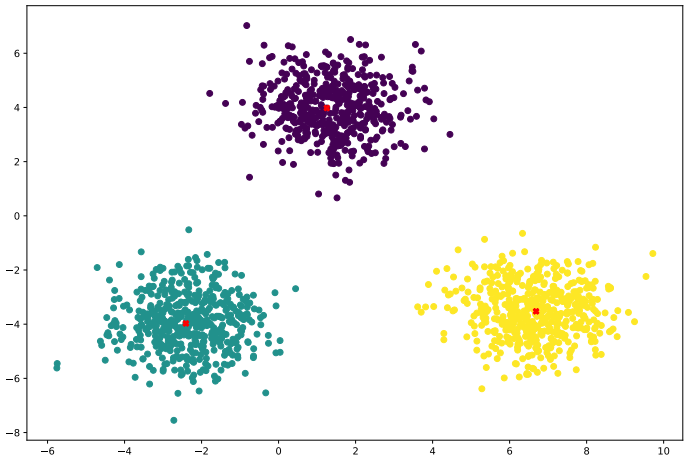
another try

keep this change
and try other changes
until no possible change

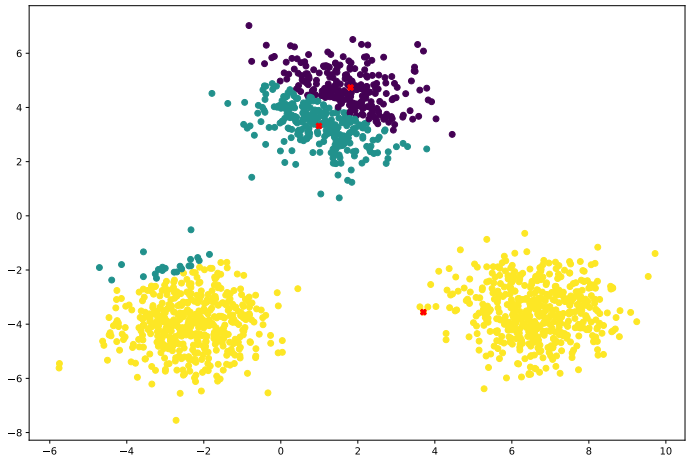


new cost: 22 + 22 = 44 < 64

k-medoids : previous example



good initialization



bad initialization

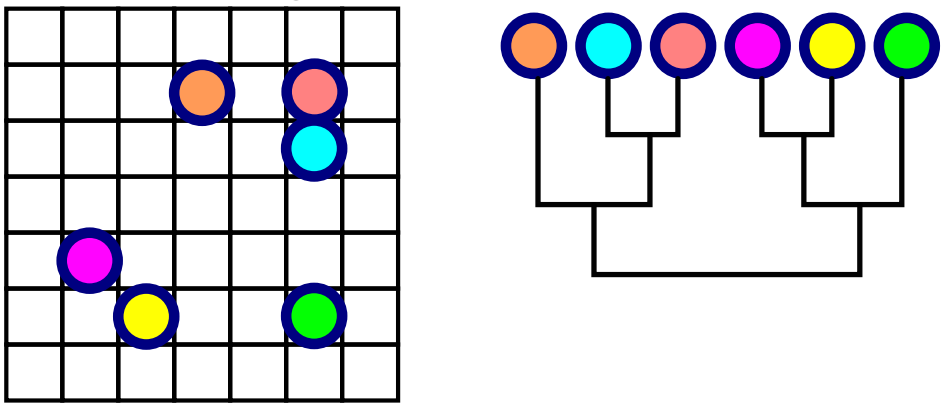
- A fuzzy extension of the k-means algorithm
- A record can belong to more than one cluster to a degree (soft vs hard assignment) :
  - how much x_i belongs to cluster k : $0 \leq \mu_k(x_i) \leq 1$
  - constraint (c the nb. of clusters) : $\sum_{k=1}^{c} \mu_k(x_i) = 1$
  - objective function (v_k represents the cluster k) :

$$\min \sum_{k=1}^{c} \sum_{i=1}^{N} \mu_k(x_i) d(x_i, v_k)$$

Outline

Hierarchical clustering methods

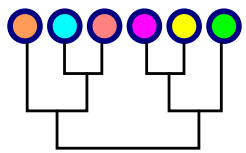
- Input : distance matrix - Output : a dendrogram (tree of clusters)
- This method does not require the number of clusters k as an input, but may need a termination condition
- Two approches :
  - Bottom up approach = agglomerative
    - Example using Manhattan distance and complete linkage



  - Top down approach (DIANA = DIvide ANAlysis)
    - start with all samples in one cluster. Hierarchical division of clusters related to a dispersion criterion (divide into clusters where closest samples are the more distant ones).

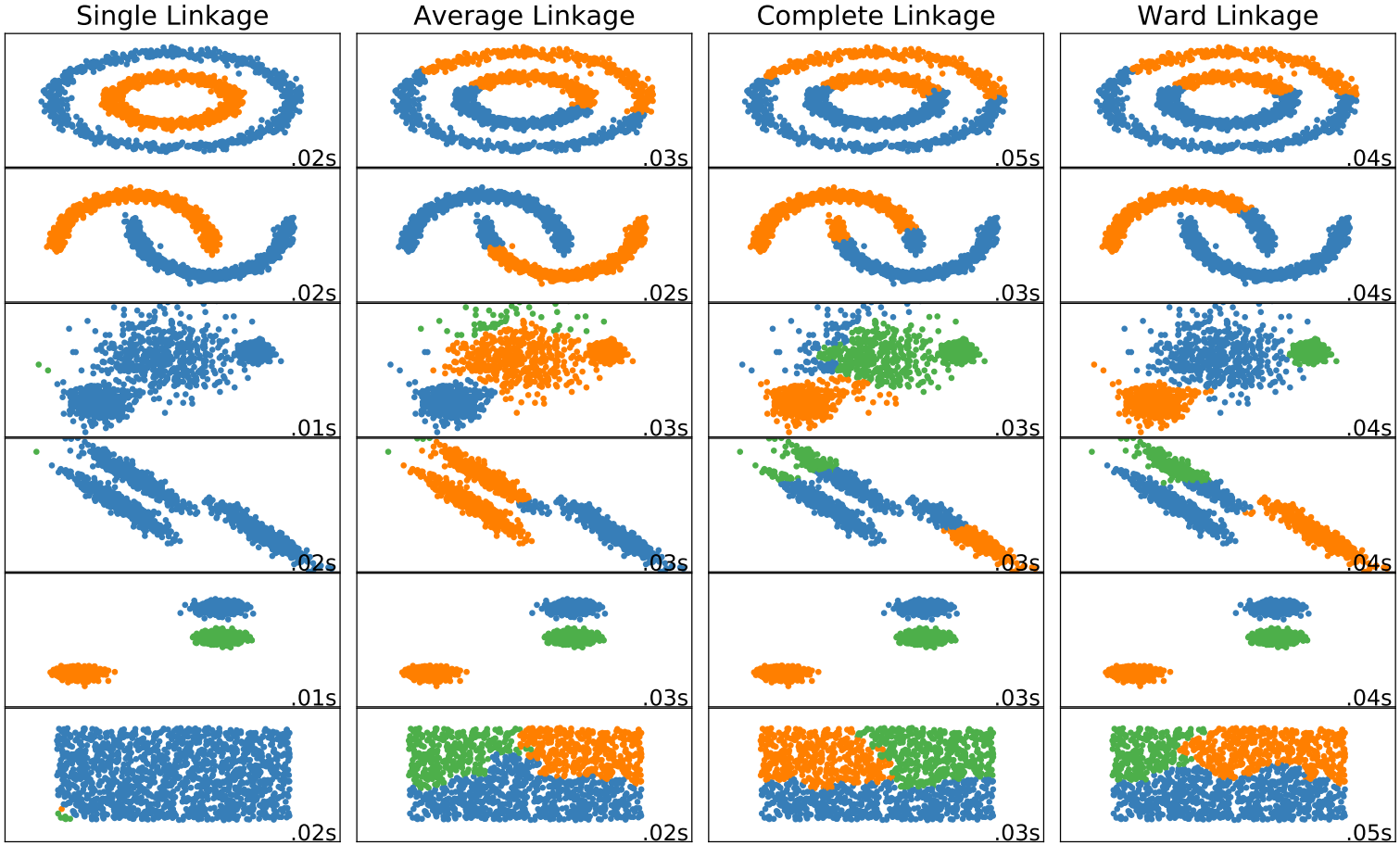
Agglomerative hierarchical clustering methods

- Algorithm :
  - starts with all single data as their own cluster
  - merged clusters together according to a linkage criteria
  - ends when all data are in the same cluster
- The linkage criteria determines the metric used for the merge strategy : (from `https://scikit-learn.org/stable/modules/clustering.html#hierarchical-clustering`)
  - **Ward** minimizes the sum of squared differences within all clusters. It is a variance-minimizing approach and in this sense is similar to the k-means objective function but tackled with an agglomerative hierarchical approach.
  - **Maximum or complete linkage** minimizes the maximum distance between observations of pairs of clusters.
  - **Average linkage** minimizes the average of the distances between all observations of pairs of clusters.
  - **Single linkage** minimizes the distance between the closest observations of pairs of clusters.



Impact of the linkage criteria

From `https://scikit-learn.org/stable/modules/clustering.html#different-linkage-type-ward-complete-average-and-single-linkage.`



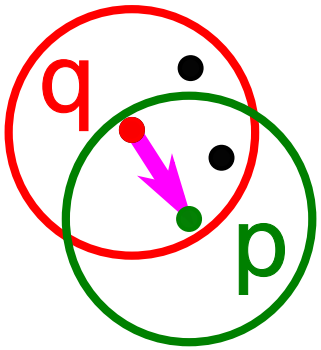
Outline

Density based clustering methods

- Clusters : areas of high density separated by areas of low density
- Clustering based on density (local cluster criterion), such as density-connected points
- Major features :
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters as termination condition
- Examples of density-based methods :
  - DBSCAN, OPTICS, DENCLUE, CLIQUE

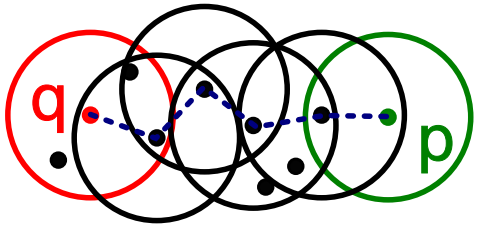
Density-Based Clustering : Basic Concepts

- Two parameters :
  - ϵ : Maximum radius of the neighbourhood
  - MinPts : Minimum number of points in an ϵ-neighbourhood of that point
- Neighborhood : $N_\epsilon(p) = \{q \in D | d(p,q) \leq \epsilon\}$
- Core point : if at least MinPts points are within distance ϵ of p (including p).
- Directly density-reachable : A point p is directly density-reachable from a point q w.r.t. ϵ, MinPts if :
  - p belongs to $N_\epsilon(q)$
  - core point condition : $|N_\epsilon(q)| \geq MinPts$

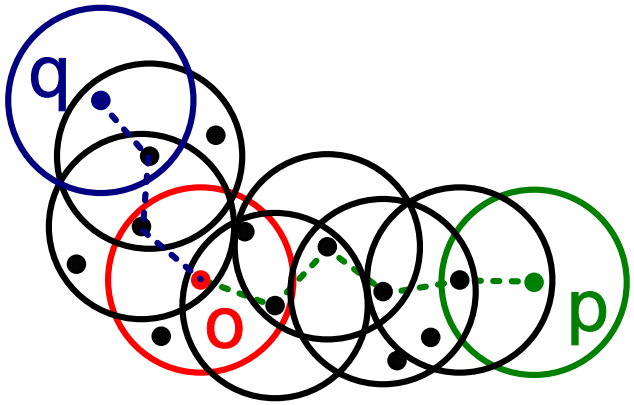


Density-Reachable and Density-Connected

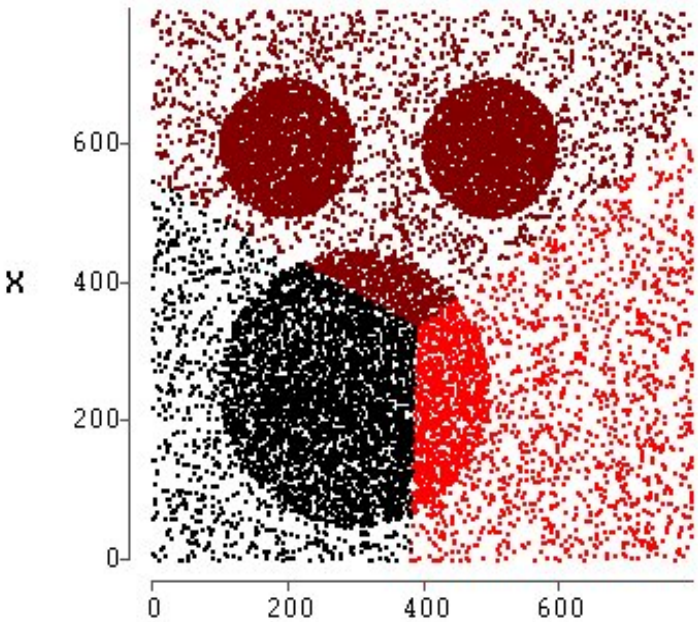
- Density-Reachable :
  - A point p is density-reachable from a point q w.r.t. ϵ, MinPts if there is a chain of points $p_1, \ldots, p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i



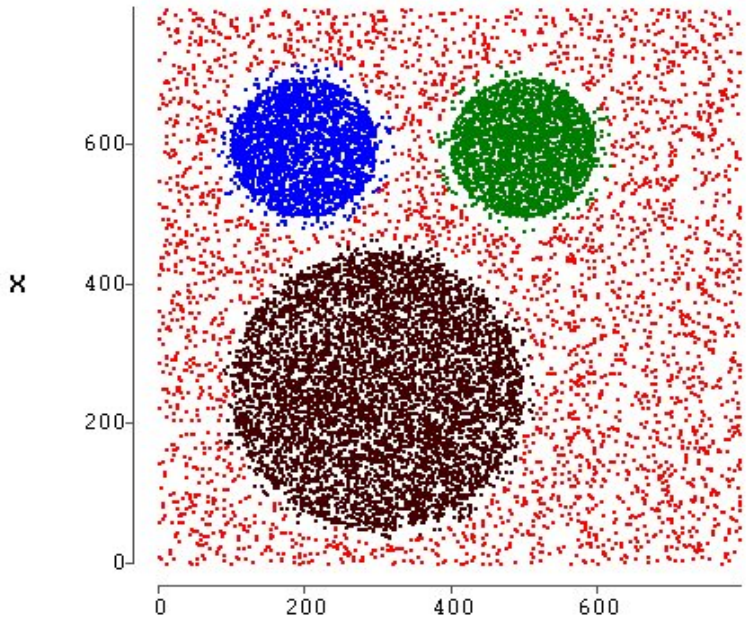
- Density-Connected :
  - A point p is density-connected to a point q w.r.t. ϵ, MinPts if there is a point o such that both, p and q are density-reachable from o w.r.t. ϵ and MinPts



- choose a core point p, not already assigned to a cluster
- add all density-reachable points from p to this cluster
- go back to step 1 until no more points without a cluster
- all points not reachable from any other point are outliers or noise points



using k-means

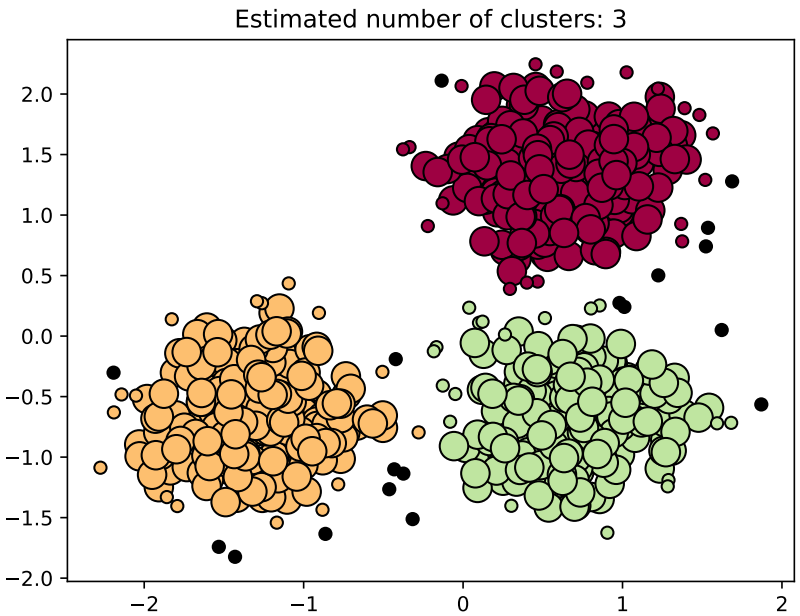


using DBSCAN

DBSCAN in scikit-learn

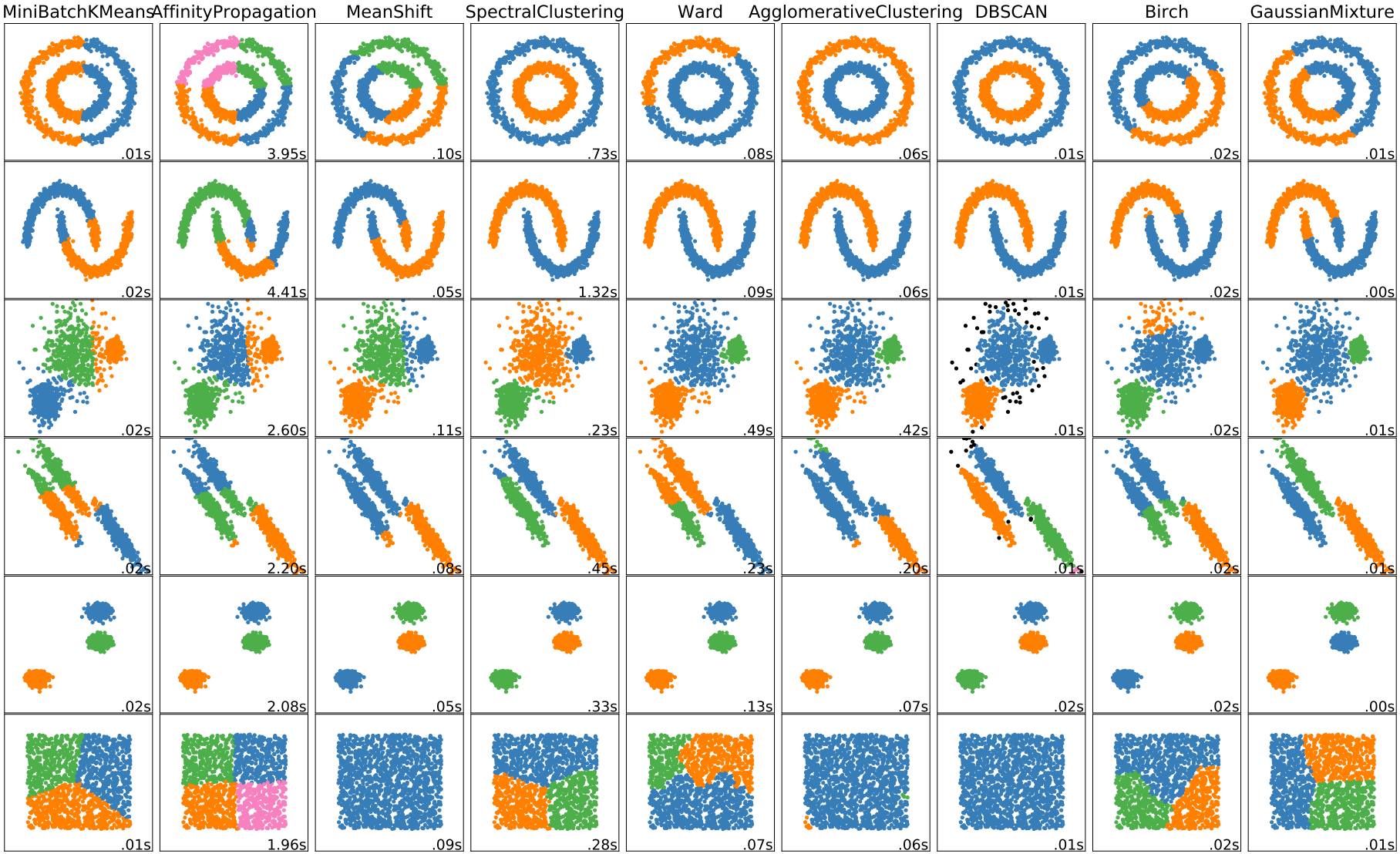
from `https://scikit-learn.org/stable/auto_examples/cluster/plot_dbscan.html#`

`sphx-glr-download-auto-examples-cluster-plot-dbscan-py`




```
from sklearn.cluster import DBSCAN
db = DBSCAN(eps=0.3, min_samples=10)
```

Outline


MiniBatchKMeans
AffinityPropagation
MeanShift
SpectralClustering
Ward
AgglomerativeClustering
DBSCAN
Birch
GaussianMixture
.01s
3.95s
.10s
.73s
.08s
.06s
.01s
.02s
.01s
.02s
4.41s
.05s
1.32s
.09s
.06s
.01s
.02s
.00s
.02s
2.60s
.11s
.23s
.49s
.42s
.01s
.02s
.01s
.02s
2.20s
.08s
.45s
.23s
.20s
.01s
.02s
.01s
.02s
2.08s
.05s
.33s
.13s
.07s
.02s
.02s
.00s
.01s
1.96s
.09s
.28s
.07s
.06s
.01s
.02s
.01s

Evaluation of clustering

- ground truth known
  - ARI (Adjusted Rand Index)
  - MI (Mutual Information)
  - V-measure
- ground truth not known
  - Davies-Bouldin index
  - Dunn index
  - Calinkski-Harabaz
  - silhouette

- Internal validation : Separation ? Compactness ?
  - E.g. Dunn, Davies-Bouldin, Calinski-Harabaz, and Silhouette indexes.
  - Problems :
    - Different performance WRT existence of noise, variable densities, and non well-separated clusters.
    - Overrate the algorithm that uses the same clustering model.

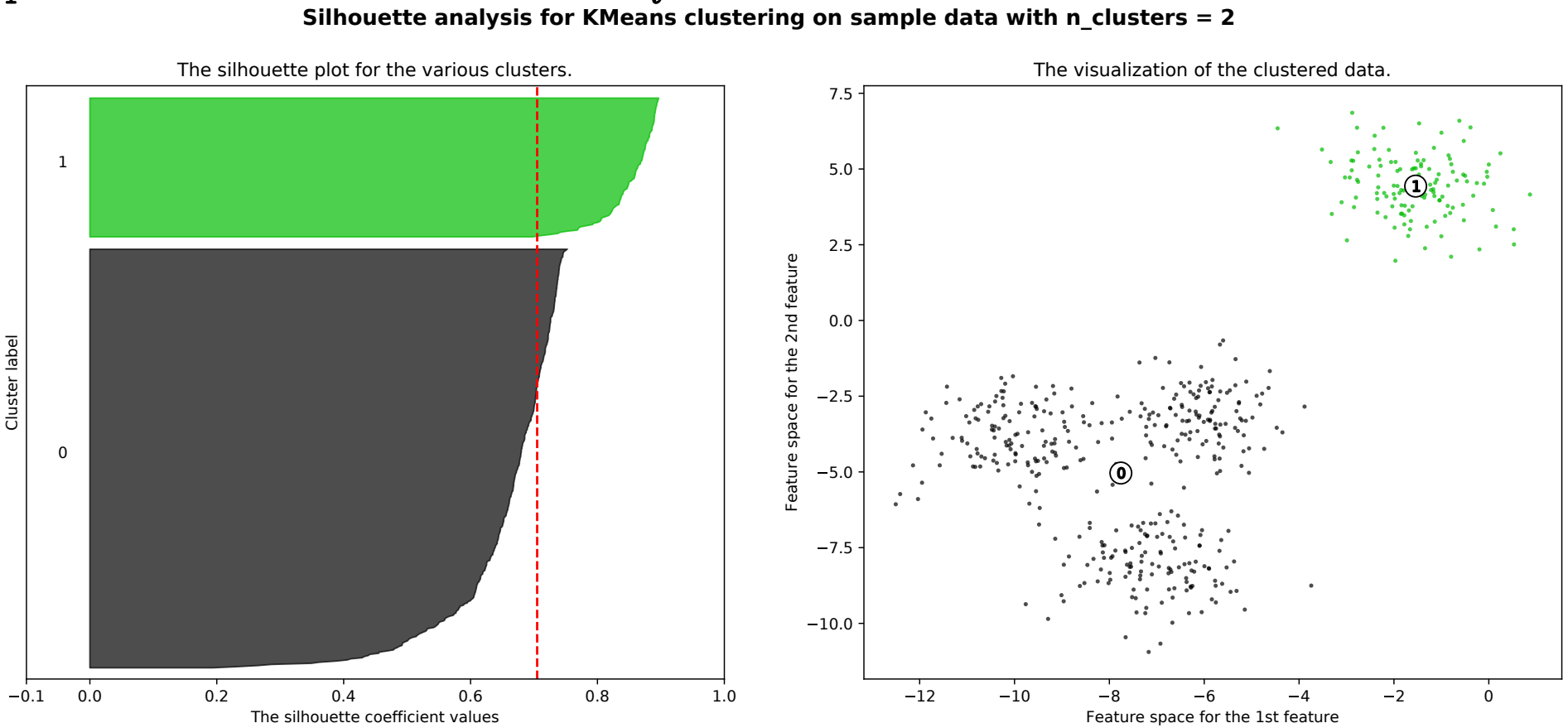
number of clusters	average silhouette score	Calinski Harabaz score
2	0.71	1604
3	0.59	1810
4	0.65	2704
5	0.56	2263
6	0.45	2043

Silhouette coefficients

- for a single sample : Silhouette coefficient $s = \frac{b-a}{\max(a,b)}$ with
  - a : mean distance between a sample and all other samples from the same cluster
  - b : mean distance between a sample and all other samples in the next nearest cluster
- for a dataset : mean of Silhouette coefficients for each sample
- sklearn : `metrics.silhouette_score`

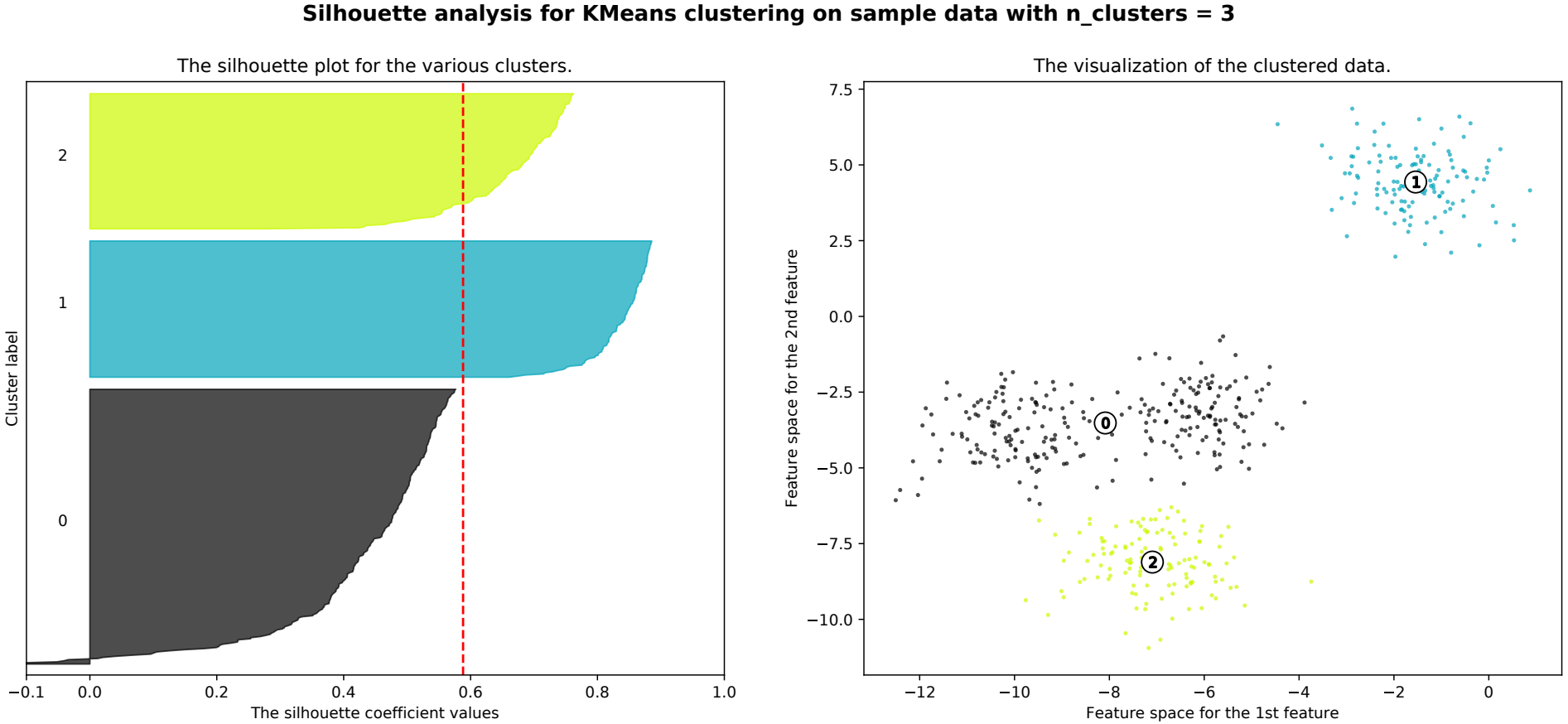
k-means with $k = 2$ and silhouette profiles

inspired by `https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html`



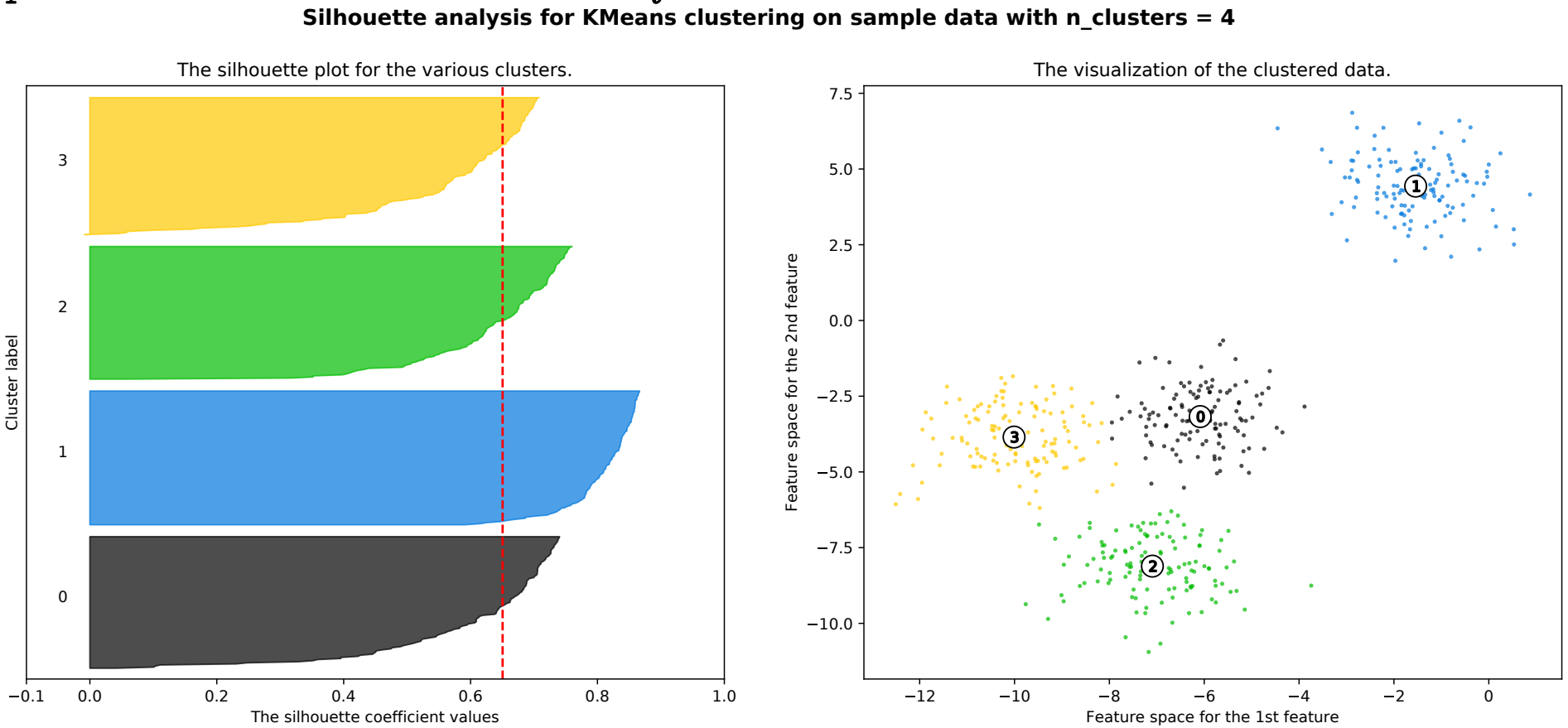
k-means with $k = 3$ and silhouette profiles

inspired by `https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html`



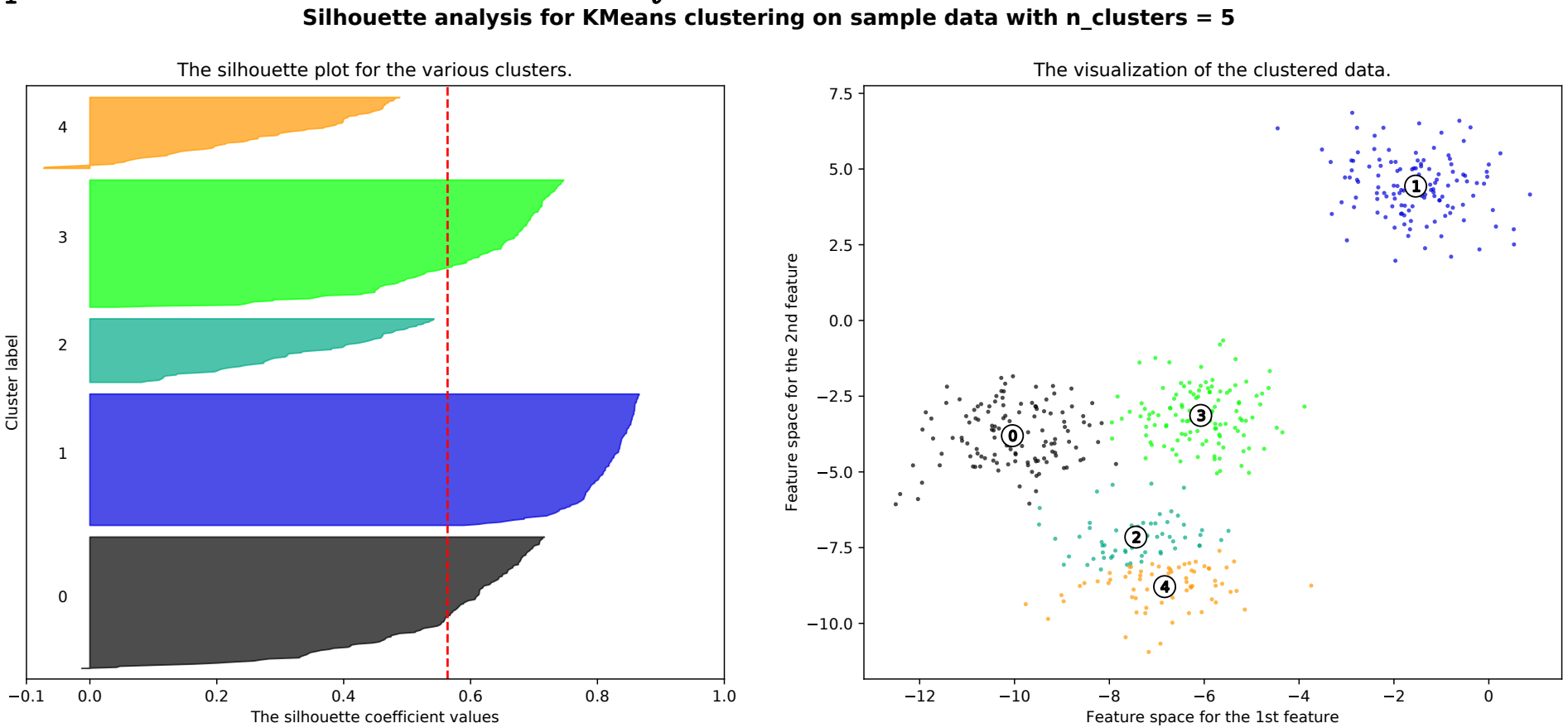
k-means with $k = 4$ and silhouette profiles

inspired by `https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html`



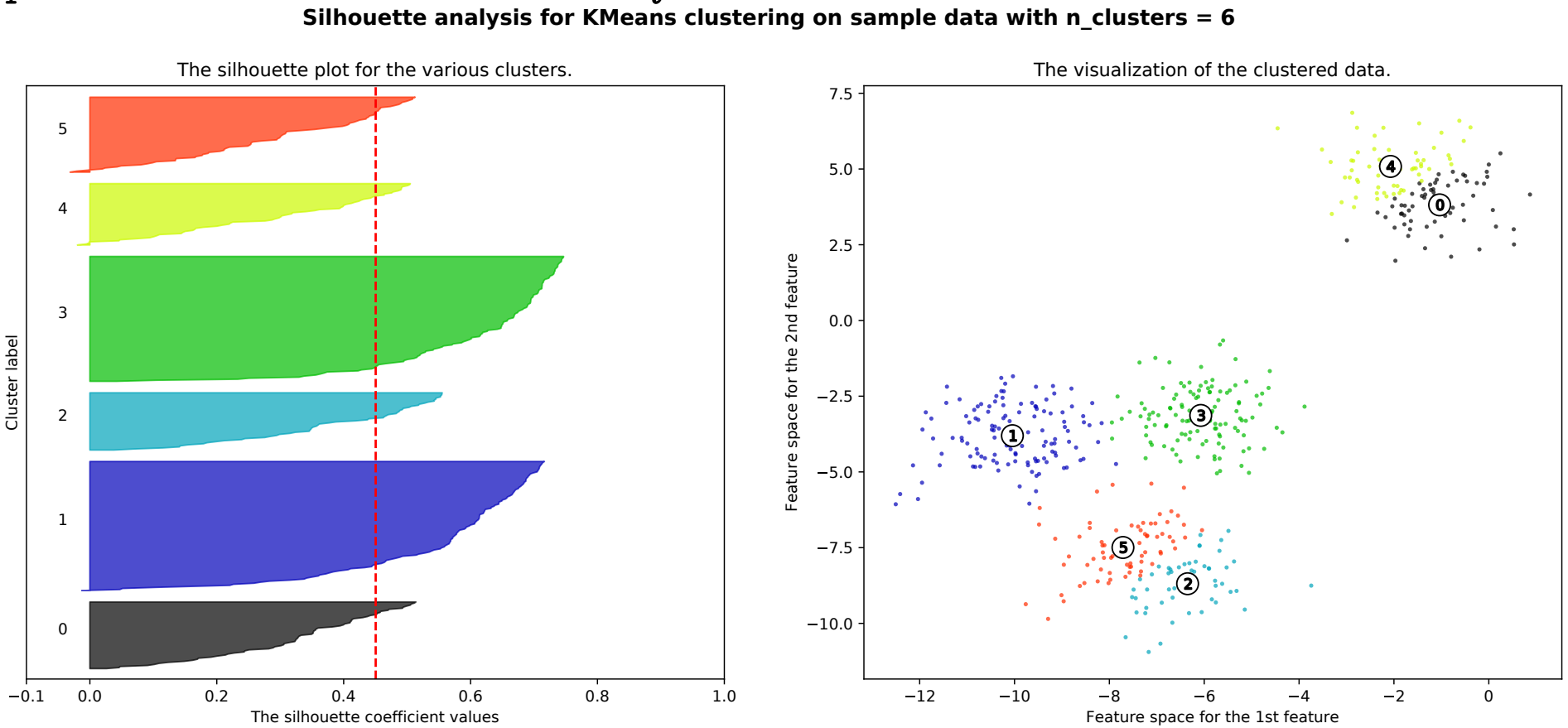
k-means with $k = 5$ and silhouette profiles

inspired by `https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html`



k-means with $k = 6$ and silhouette profiles

inspired by `https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html`



You need to remember

- For numerical data, standardisation is really important
- Many algorithms exists with different advantages and drawbacks
  - you need to find the one that fits you data
- Parameters of algorithms are also important
  - you need to tune them
- Evaluation metrics are usefull but should be considered carefully
  - no better than a dedicated metric for your problem (final goal of a pipeline)