



# **ANALYSE DE DONNÉES**

Nassim Laga  
nassimlaga@gmail.com  
Janvier 2019

# Plan du cours

- Introduction au data mining (1h)
- Apprentissage supervisé et non supervisé
- Focus sur l'apprentissage supervisé + mesures de performance et de validation (2h)
- **Focus sur l'apprentissage non supervisé + mesures de performance et de validation (2h)**
- Focus sur l'analyse de texte (2h)



# **APPRENTISSAGE NON SUPERVISÉ**

Nassim Laga  
nassimlaga@gmail.com  
Janvier 2019

# Plan du jour

- [Rappel] Apprentissage supervisé et non supervisé

- Motivation & Use cases
- Formulation mathématique [Rappel]
  - Apprentissage non supervisé

- Algorithmes d'apprentissage non supervisé

- Trois grandes famille d'algorithmes
- K-moyenne (kmeans)
- le clustering hiérarchique (HAC)
- les algorithmes à base de densité
  - dbscan
  - hdbscan

- Mesures de performances / détermination du nombre de clusters

- Davies – Bouldin
- Silhouette



# **[RAPPEL] APPRENTISSAGE NON SUPERVISÉ**

# [Rappel] Apprentissage non supervisé

## - Motivation & Use cases

- Marketing

- Segmentation client : construire des regroupements de profils client. Rassembler les clients qui se **ressemblent** dans le même groupe pour :
  - faire des propositions commerciale les plus susceptible de les intéresser ;
  - personnaliser le canal qu'on va utiliser pour communiquer avec eux ;
- Segmentation prospect : détecter les différents groupe de prospects qui se **ressemblent** dans le but :
  - de détecter les gammes produits proposées par l'entreprise couvre tous les groupes détectés



# [Rappel] Apprentissage non supervisé

## - Motivation & Use cases

- Usage de site Web
  - Clustering des logs laissé par les clients pour identifier les différents patterns
- Sécurité de site Web (ou d'applications)
  - Clustering des logs dans le but d'isoler les traces « anormale » potentiellement liées à des attaques
- Détection de fraudes (assurance, fiscale, etc.)
  - Clustering des profil et comportements clients (navigation web, type de demande, nombre d'appels, nombre de demande, nombre de réclamation , type de réclamation, ...etc.)

# [Rappel] Apprentissage non supervisé

## – Formulation mathématique

- Apprentissage non supervisé – modélisation mathématique
  - Soit  $K(X_1, \dots, X_n)$  les données d'apprentissage. Il s'agit d'un ensemble **d'observations** des variables **explicatives**  $X_i$  ( $\mathcal{X}$ )
  - But :
    - Créer des regroupements (clusters) **cohérents/homogène**
  - Que veut dire alors **cohérent et homogène** ?



# [Rappel] Apprentissage non supervisé

## – Formulation mathématique

- Apprentissage non supervisé – modélisation mathématique
  - Il s'agit de définir une fonction  $f: X \rightarrow D: \{1, \dots, k\}$ 
    - $k$  étant le nombre de groupes qu'on souhaite créer à partir des observations  $X$
  - Contrairement à un problème de classification
    - la cible n'est pas connue, pas de base de vérité
    - l'analyse se fait uniquement sur les données d'apprentissage

# [Rappel] Apprentissage non supervisé

## – Formulation mathématique

- Apprentissage non supervisé – modélisation mathématique
  - L'absence de base de vérité ➔ nécessite de définir des mesures de qualité, qui permettraient de valider ou pas le résultat
  - Un **bon** regroupement (clustering) est souvent défini
    - Homogénéité intra-cluster
    - Distance (inhomogénéité) inter-clusters

# [Rappel] Apprentissage non supervisé

## – Formulation mathématique

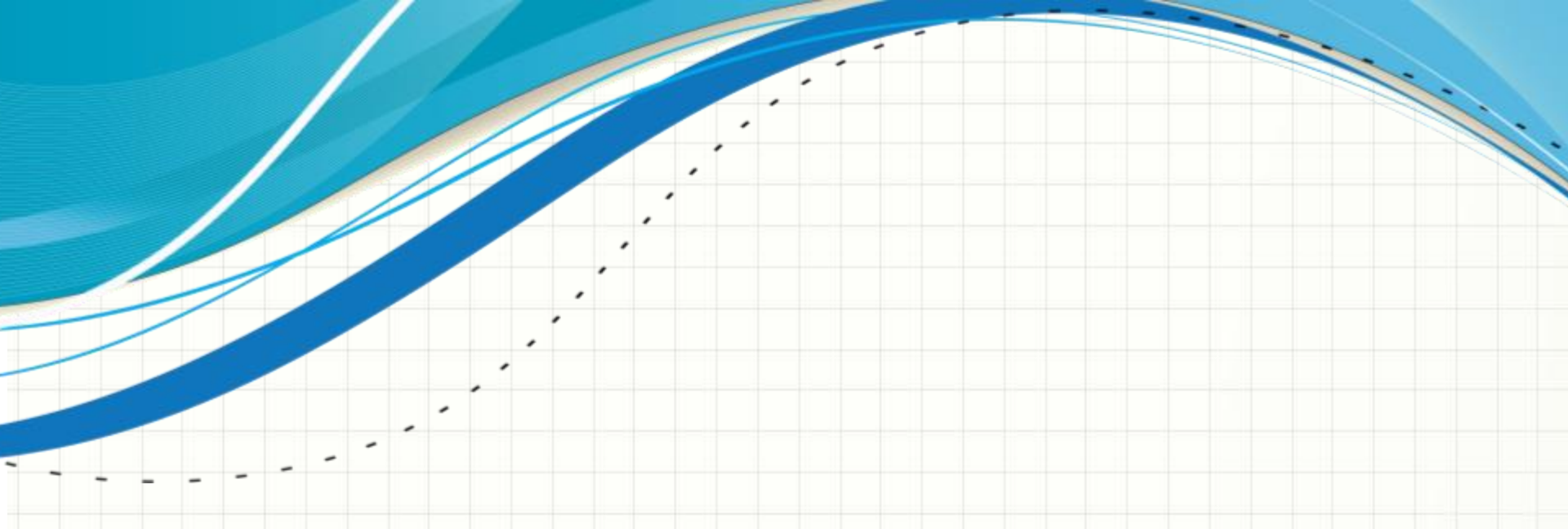
- Apprentissage non supervisé – modélisation mathématique
  - Ces métriques sont souvent basées sur la notion de distance entre les individus ou encore entre les clusters
  - Exemple de distance
    - variance intra-cluster
    - variance inter-cluster



# **ALGORITHMES D'APPRENTISSAGE NON SUPERVISÉ**

# Trois grandes familles d'algorithmes

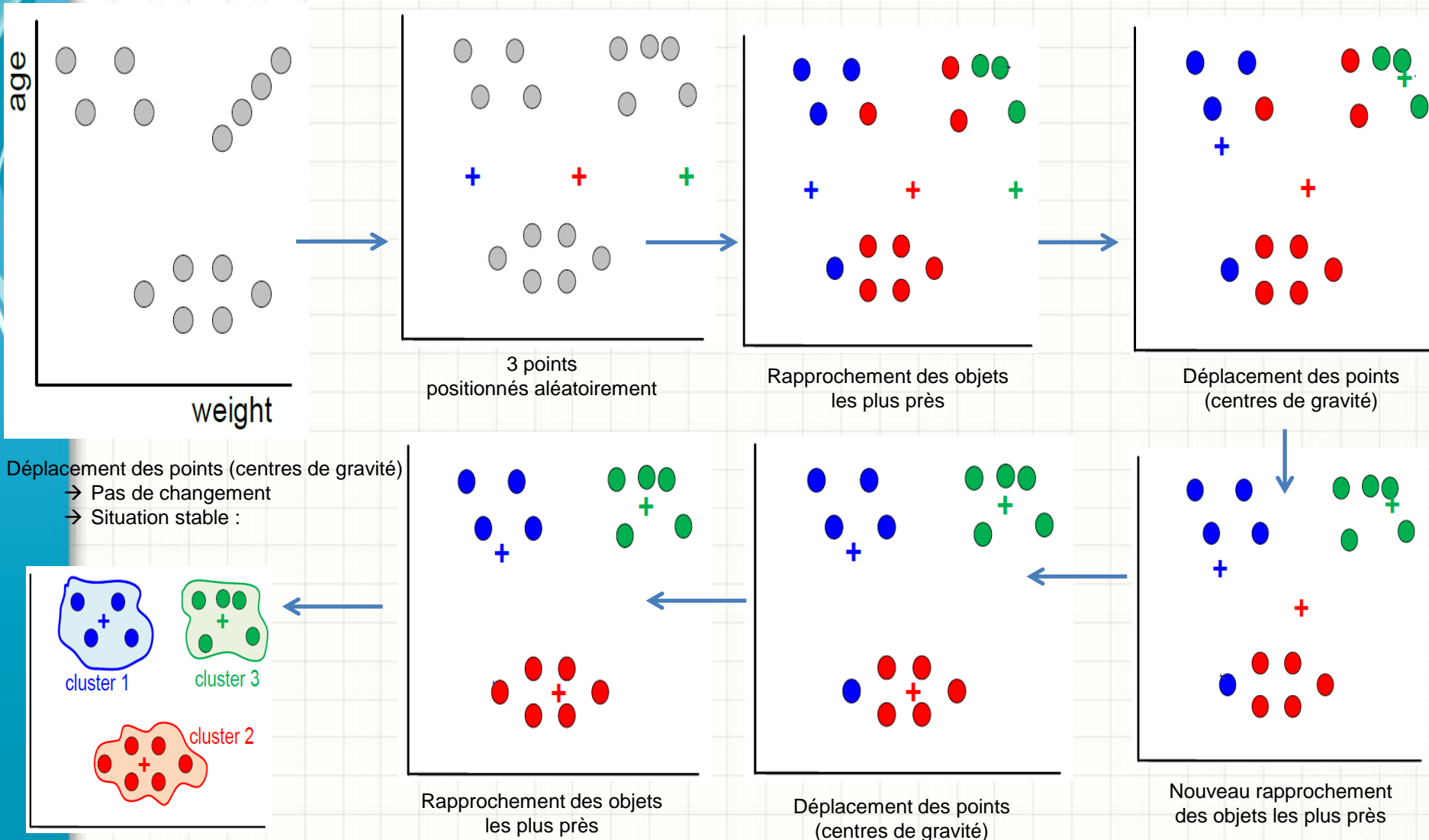
- Algorithmes à base de partitionnement
  - Intuition :
    - Créer K groupes qui minimisent un certain critère d'homogénéité.
    - À besoin du nombre k et du critère d'homogénéité.
  - e.g. k-means, k-medoid, etc.
- Algorithmes à base de densité
  - Intuition :
    - Créer K groupes, qui ont une densité forte à l'intérieur (intra-clusters), et une densité faible entre clusters (inter-clusters).
  - e.g. dbscan, hdbscan, etc.
- Algorithmes agglomératifs (« assembleur »)
  - Intuition :
    - Créer autant de groupes qu'il y'a d'individus
    - Regrouper itérativement les groupes les plus proches jusqu'à obtenir un seul groupe
    - Génère un arbre de clusters au lieu d'un ensemble de clusters
  - e.g. hiérarchique



# **ALGORITHMES D'APPRENTISSAGE NON SUPERVISÉ — K-MEANS**



# Algorithmes d'apprentissage non supervisé – K- means



# Algorithmes d'apprentissage non supervisé – K- means

- Chaque cluster est défini par son centroïde  $\mu_k$
- Chaque individu est associé (assigné) à son plus proche centroïde
- Chaque cluster est défini de sorte à minimiser la distance entre les individus et le centroïde correspondant

$$- \sum_{j=1}^k \sqrt{\sum_{i=1}^{n_j} \|X_{ji} - \mu_j\|^2}$$

# Algorithmes d'apprentissage non supervisé – K- means

- Algorithme détaillé
- K-means (K, X):
  - Placer aléatoirement  $K$  points (centroïdes)  $c_k$
  - Répéter les deux points suivants jusqu'à convergence
    1. Pour chaque point  $x_i$  dans  $X$   
trouver le plus proche centroïde  $c_k$  en se basant sur une formule de distance (e.g. distance euclidienne)
$$\sqrt{\sum_{i=1}^{n_j} \|X_{ji} - \mu_j\|^2}$$
    2. Pour chaque cluster, calculer le centre de gravité  $c_k = \frac{1}{|S_k|} \sum_{i=1}^{|S_k|} x_i$  ( $S_k$  représente l'ensemble des individus affectés au cluster  $k$ )

# Algorithmes d'apprentissage non supervisé – K- means

- Quelques remarques
  - Le choix de K est crucial dans la validité du résultat final
    - Le connaître à l'avance reste une question ouverte.
  - La convergence est détectée si on fait une itération sans aucun changement au niveau de l'affectation des points au clusters.
  - L'algorithme ne garantit pas la convergence → nécessité de définir le nombre d'itérations max.
  - Il n'est pas garanti non plus d'atteindre l'optimum global
  - L'initialisation joue un rôle crucial dans la performance de l'algorithme, et le résultat final
  - Complexité :  $O(K*N*I*D)$  → I représente le nombre d'itérations, et D représente la dimension des données.

# Algorithmes d'apprentissage non supervisé – K- means

- Quelques variantes

- Algorithme détaillé

- K-means (K, X):

- Placer aléatoirement  $K$  points (centroïdes)  $c_k$
- Répéter les deux points suivants jusqu'à convergence

1. Pour chaque point  $x_i$  dans  $X$

trouver le plus proche centroïde  $c_k$  en se basant sur une formule de distance (e.g. distance euclidienne

$$\sqrt{\sum_{i=1}^{n_j} \|X_{ji} - \mu_j\|^2})$$

2. Pour chaque cluster, calculer le centre de gravité  $c_k = \frac{1}{|S_k|} \sum_{i=1}^{|S_k|} x_i$  ( $S_k$  représente l'ensemble des individus affectés au cluster  $k$ )

k-means++ : placer les points les plus éloigner les uns des autres

D'autre distance peuvent être utilisée : [distances scipy](#)

k-medoid : Choisir un individu parmi les données comme centroïde. Choisir l'individu le plus proche du centre de gravité

# Algorithmes d'apprentissage non supervisé – K- means

- Le clustering k-means en python
- **sklearn.cluster.KMeans** : <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>

```
In [54]: #Clustering avec kmeans  
from sklearn.cluster import KMeans  
kmeans = KMeans(n_clusters=4)  
kmeans.fit(dataset)
```

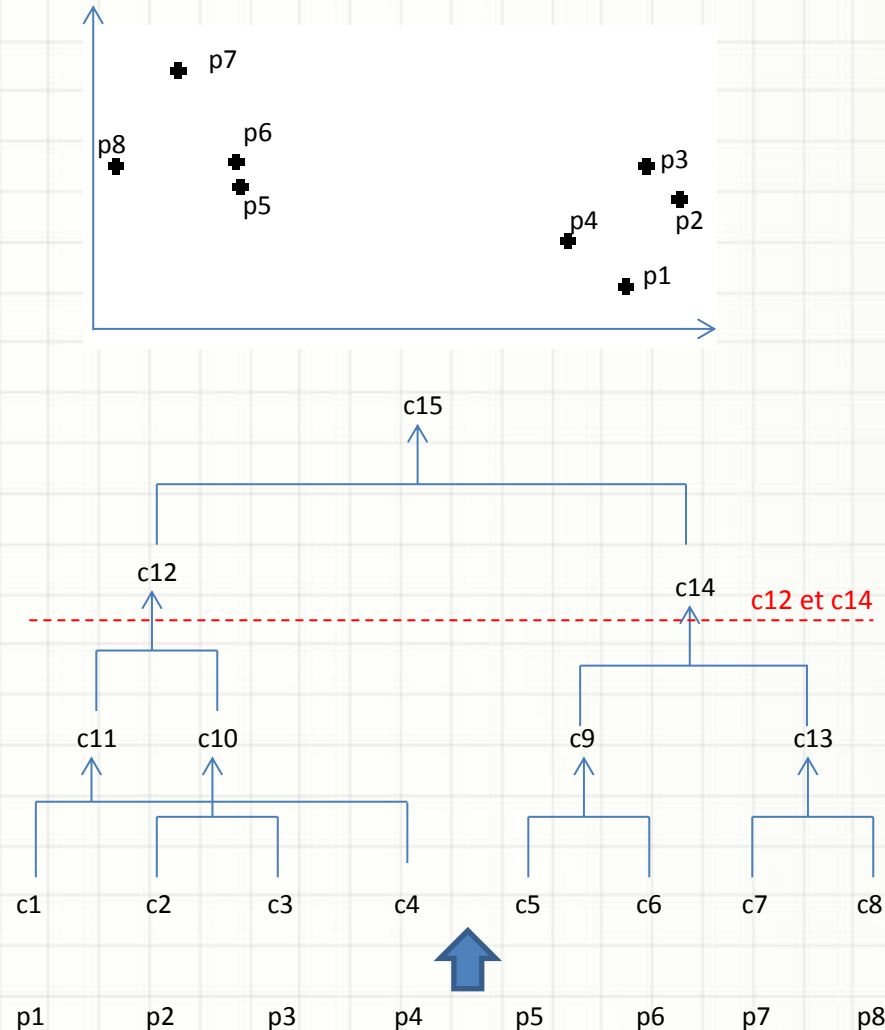




# **ALGORITHMES D'APPRENTISSAGE NON SUPERVISÉ — LE CLUSTERING HIÉRARCHIQUE**

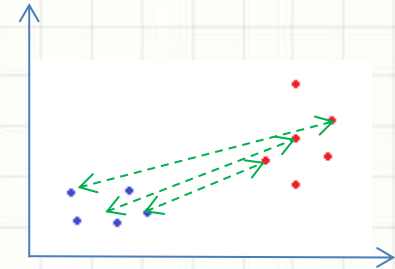
# Algorithmes d'apprentissage non supervisé – le clustering hiérarchique

- Associer à chaque point un cluster
- Répéter les étapes suivantes jusqu'à ce qu'il y'en ait un seul cluster
  - Calculer la matrice de distance entre les clusters
  - Détecter les deux clusters les plus proches
  - Fusionner ces deux points dans un cluster
- Couper horizontalement l'arbre pour avoir la liste des clusters à plat



# Algorithmes d'apprentissage non supervisé – le clustering hiérarchique

- Plusieurs questions ?
  - On connaît la distance entre points, mais qu'en est-il de la distance entre clusters ? ➔ average, median, complete, ...etc.
  - Où couper dans l'arbre pour avoir une liste de clusters ? ➔  $k$  (Number of clusters), max intra-cluster distance
  - Qu'en est-il de la complexité ?



# Algorithmes d'apprentissage non supervisé – le clustering hiérarchique

- Plusieurs questions ?
  - On connaît la distance entre points, mais qu'en est-il de la distance entre clusters ? → Linkage
    - **Average** : la distance entre deux clusters (c1 et c2) est la distance moyenne entre tous les points du premier cluster (c1) et ceux du deuxième (c2).
    - **Single** : la distance entre deux clusters (c1 et c2) est la distance minimale entre les points du premier cluster (c1) et ceux du deuxième (c2).
    - **Complete** : la distance entre deux clusters (c1 et c2) est la distance maximale entre les points du premier cluster (c1) et ceux du deuxième (c2).
    - **Centroid** : la distance entre deux clusters (c1 et c2) est la distance entre le centroid du premier cluster (c1) et celui du deuxième (c2).
    - ...etc

# Algorithmes d'apprentissage non supervisé – le clustering hiérarchique

- Ou couper dans l'arbre pour avoir une liste de clusters ? →  $k$  (Number of clusters), max intra-cluster distance
  - $k$  représente le nombre de cluster cible (difficile à évaluer à priori) → On descend dans l'arbre tant que  $k$  n'est pas atteint.
  - max intra-cluster distance définit le « rayon » de vos cluster. (difficile à évaluer à priori) → On descend dans l'arbre tant que la taille du cluster est supérieure à la distance max intra-cluster

# Algorithmes d'apprentissage non supervisé – le clustering hiérarchique

- Le clustering hiérarchique en python
- **sklearn.cluster.AgglomerativeClustering** : <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html#sklearn.cluster.AgglomerativeClustering>

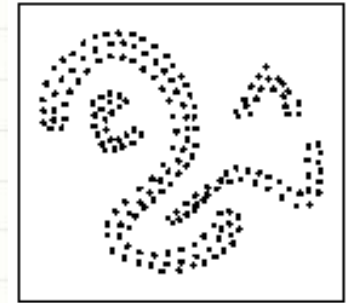
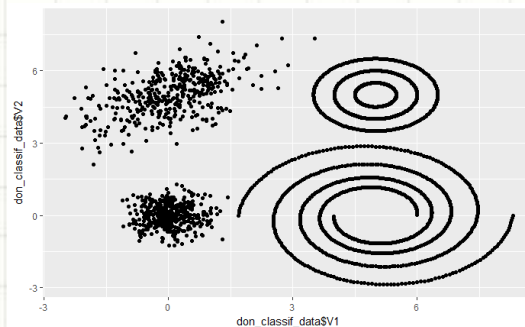
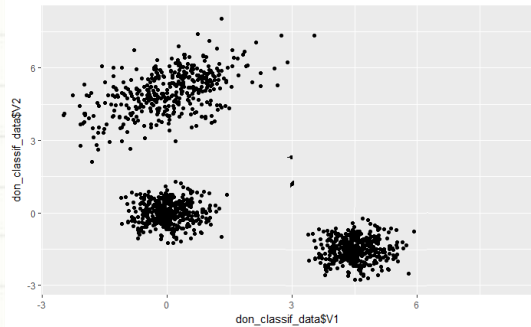
```
In [25]: #Le clustering hiérarchique  
from sklearn.cluster import AgglomerativeClustering  
hacClustering = AgglomerativeClustering(n_clusters=4, affinity="euclidean", linkage="ward")  
hacClustering.fit(dataset)|
```





# **ALGORITHMES D'APPRENTISSAGE NON SUPERVISÉ – DBSCAN**

# Algorithmes d'apprentissage non supervisé – DBSCAN



Autant qu'humain, quel seraient les cluster que vous créerait ?

Quel serait les cluster que k-means créerait ?

# Algorithmes d'apprentissage non supervisé – DBSCAN

- Principe de densité
  - Le nombre de points (individus) contenu dans un rayon (distance).
  - Pour un **rayon donné (eps)**, on peut comparer des zone en fonction de leur densité (nombre de points contenus dans ce rayon).
- Deux paramètres sont à donner à DBSCAN
  - **Le rayon (eps)**
  - **MinPts (Minimum Points)**: Nombre minimum point dans le rayon pour considérer une zone dense ou pas.



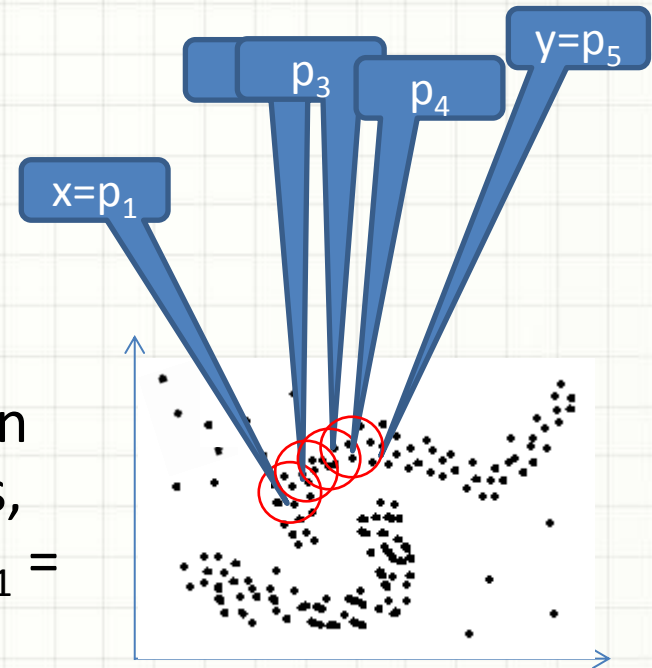
# Algorithmes d'apprentissage non supervisé – DBSCAN

- Soit  $x$  un point dans l'ensemble des individus  $X$
- Soit  $N_{eps}(x)$  l'ensemble des points qui sont dans le rayon  $eps$  de  $x$ 
  - $N_{eps}(x) = \{y \in X \mid dist(x, y) \leq eps\}$
- Soit  $DDR(x)$  (*directly density-reachable*) l'ensemble des points « directement atteignables par densité » à partir de  $x$ .
  - Un point  $y$  est **directement atteignable par densité**, sachant un rayon  $eps$  et un nombre minimum de point  $MinPts$ , ssi,  $y$  fait partie du voisinage de  $x$ , et le nombre de voisin de  $x$  est supérieur à  $minPts$  (la zone délimité par le rayon sur le point  $x$  est dense).
    - $y \in N_{eps}(x)$
    - et  $|N_{eps}(x)| \geq MinPts$



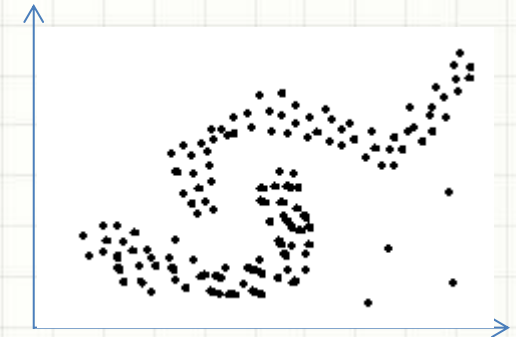
# Algorithmes d'apprentissage non supervisé – DBSCAN

- Soit  $DR$  (*density-reachable*) l'ensemble des points « atteignables par densité ».
  - Un point  $y$  est **atteignable par densité**, sachant un rayon  $\epsilon$  et un nombre minimum de point  $MinPts$ , ssi, il existe une chaîne  $p_1, \dots, p_n$  ( $p_1 = x$  et  $p_n = y$ ) telque  $p_{i+1} \in DDR(p_i)$
- Points connectés sont des points mutuellement atteignables par densité



# Algorithmes d'apprentissage non supervisé – DBSCAN

- Dans DBSCAN, on définit un cluster comme un ensemble maximal de points connectés par densité (mutuellement atteignables par densité).
- Algorithmes
  - Choisir un point  $x$  de l'ensemble  $X$ , de manière arbitraire
  - Calculer le voisinage de  $x$  ( $N_{\text{eps}}(x)$ )
  - Si  $|N_{\text{eps}}(x)| < \text{MinPts}$ , marquer  $x$  comme bruit (ne faisant partie d'aucun cluster)
  - Sinon, définir un cluster  $C_i$ ,
  - et pour chaque points  $p$  dans  $N_{\text{eps}}(x)$ 
    - Ajouter  $p$  à  $C_i$
    - Calculer le voisinage  $N_{\text{eps}}(p)$
    - $N_{\text{eps}}(x) = N_{\text{eps}}(x) \cup N_{\text{eps}}(p)$
  - répéter jusqu'à ce que tous les points soit visités





# Algorithmes d'apprentissage non supervisé – DBSCAN

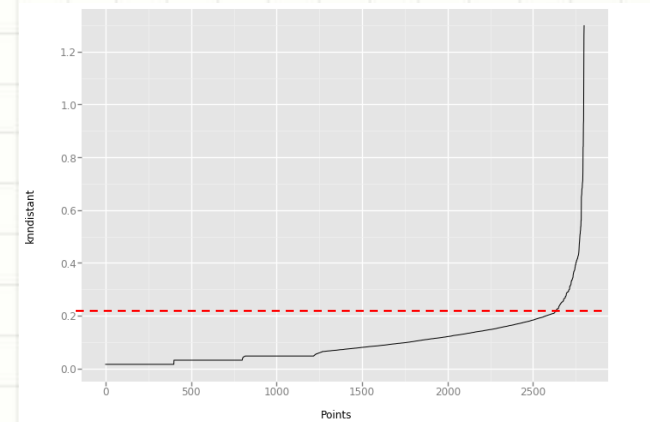
- Animation DBSCAN
  - <https://www.youtube.com/watch?v=h53WMllmUuc>

# Algorithmes d'apprentissage non supervisé – DBSCAN

- DBSCAN Avantages et inconvénients
  - Avantages
    - Permet de découvrir des clusters de différentes formes.
    - Permet de découvrir les outliers (les points isolés, qui souvent représentent du bruit).
  - Inconvénients
    - A besoin de deux paramètres **eps** et **MinPts** qui sont souvent difficiles à estimer en pratique
    - Ne permet de découvrir des clusters de différentes densités

# Algorithmes d'apprentissage non supervisé – DBSCAN

- Estimation du rayon (eps)
  - Estimer la densité des cluster (en supposant que les clusters sont de même densité !!)
    - Pour chaque point (individu), calculer la distance qui le sépare de  $k^{\text{ème}}$  voisin (knn, ou  $k^{\text{th}}$  nearest neighbor) → plus cette distance est petite, plus la zone est dense
    - Ordonner ces distances de manière ascendante, et les afficher sur un graphe
    - Repérer les points « vallée » ou « knee », qui représentent les changements de densité

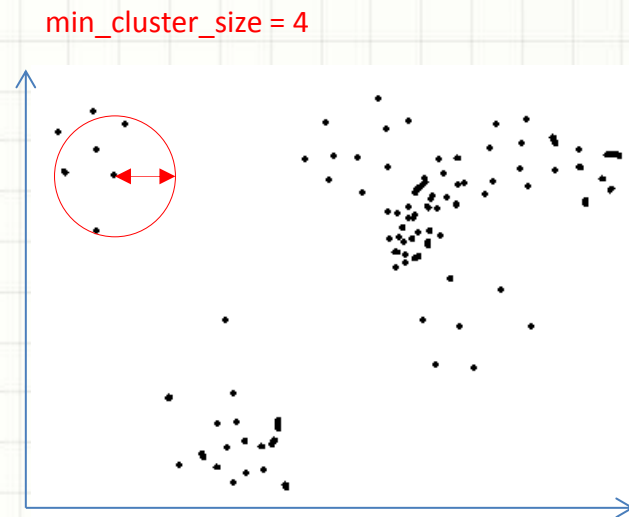


# Algorithmes d'apprentissage non supervisé – DBSCAN

- Estimation du rayon (minPts)
  - Se baser sur les connaissances métier de vos données
  - Tester différentes valeurs !

# Algorithmes d'apprentissage non supervisé – DBSCAN

- Découverte de clusters de différentes densités → HDBSCAN
- Suppose qu'ils y'a des zones de différentes densités, avec potentiellement des zones incluses dans d'autres
- à besoin d'un paramètre important *min\_cluster\_size*
- Pour chaque point  $p$  calculer la *core\_distance*. le rayon (eps) minimal pour avoir un nombre de voisin  $\geq$  à *min\_cluster\_size*
- Construire le « mutual reachability graph » : Un graphe complet: les nœuds sont les points, et le arcs représente la distances entre ces point
- Construire le « Minimum Spanning Tree » (MST) – ou en français « Arbre couvrant de poids minimal »
- Construire une hiérarchie de clusters, et choisir les clusters les plus stables
  - à l'élément racine on associe tous les points du dataset
  - Supprimer de manière itérative des liens dans le MST en commençant par celui de poids le plus grand (qui relie les deux points les plus éloignés)
    - Créer deux fils dans l'arbre de clusters, correspondant aux deux composants connectés dans le MST qu'on vient de déconnecter.



# Algorithmes d'apprentissage non supervisé – DBSCAN en python, qlqs liens utile

- **sklearn.neighbors.KDTree**: permet de calculer, en autre, la distance aux plus proches voisins, <https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KDTree.html>
- **sklearn.cluster.dbscan** : <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html>
- **hdbscan** : <https://hdbscan.readthedocs.io/en/latest/>

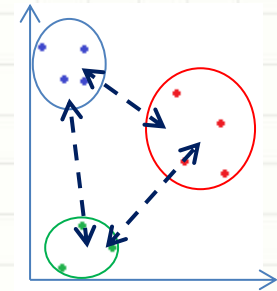




# **MESURES DE PERFORMANCES / DÉTERMINATION DU NOMBRE DE CLUSTERS**

# Mesures de performances / détermination du nombre de clusters

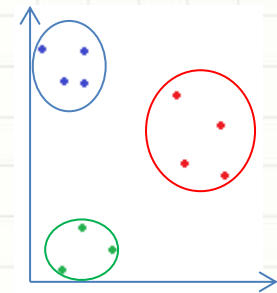
- Davies – Bouldin
  - Permet d'évaluer le résultat d'un regroupement (clustering)
  - Est basé sur la notion de similarité entre clusters (plus la similarité est faible mieux est le partitionnement), et l'inertie intra-clusters
  - Ce n'est pas une validation métier → une bonne valeur ne signifie pas un bon clustering du point de vu métier



# Mesures de performances / détermination du nombre de clusters

- Davies – Bouldin (formule)

- $DB = \frac{1}{N} \sum_{i=1}^N D_i$ 
  - $D_i$  reflète la distance entre le cluster  $i$  et le cluster le plus proche
- $D_i = \text{Max}_{i \neq j} R_{ij}$ 
  - $R_{ij}$  reflète la similarité entre un cluster  $i$  et un cluster  $j$  différent
- $R_{ij} = \frac{S_i + S_j}{M_{i,j}}$ 
  - $S_i$  est une mesure qui reflète la proximité des points d'un cluster à son centroid
  - $M_{i,j}$  est une mesure qui reflète la distance des points d'un cluster  $i$  à ceux du cluster  $j$
- $S_i = \left( \frac{1}{T_i} \sum_{j=1}^{T_i} |X_j - A_i|^p \right)^{1/p}$ 
  - $T_i$  est la taille du cluster  $i$
  - $A_i$  est le centroid du cluster  $i$
  - $X_j$  représente les points associé au cluster
  - Pour une distance euclidienne,  $p$  sera égal 2
- $M_{i,j} = d(c_i, c_j)$  est la distance entre le cluster  $i$  et le cluster  $j$  (cf. Linkage pour la distance entre clusters)



# Mesures de performances / détermination du nombre de clusters

- Silhouette

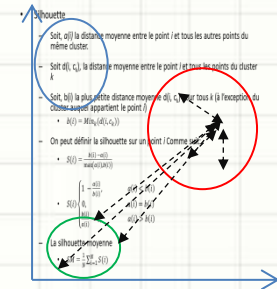
- Soit,  $a(i)$  la distance moyenne entre le point  $i$  et tous les autres points du même cluster.
- Soit  $d(i, c_k)$ , la distance moyenne entre le point  $i$  et tous les points du cluster  $k$
- Soit,  $b(i)$  la plus petite distance moyenne  $d(i, c_k)$  pour tous  $k$  (à l'exception du cluster auquel appartient le point  $i$ )
  - $b(i) = \min_k (d(i, c_k))$
- On peut définir la silhouette sur un point  $i$  Comme suit:

- $$S(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

- $$S(i) = \begin{cases} 1 - \frac{a(i)}{b(i)}, & a(i) < b(i) \\ 0, & a(i) = b(i) \\ \frac{b(i)}{a(i)} - 1, & a(i) > b(i) \end{cases}$$

- La silhouette moyenne

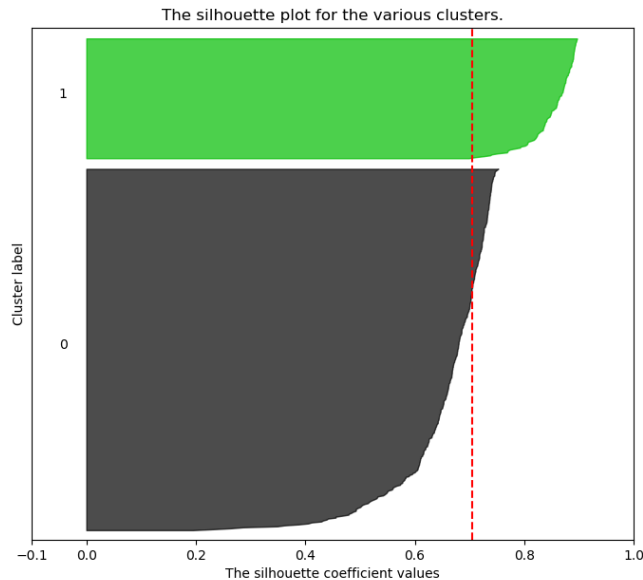
- $$SM = \frac{1}{N} \sum_{i=1}^N S(i)$$



# Mesures de performances / détermination du nombre de clusters

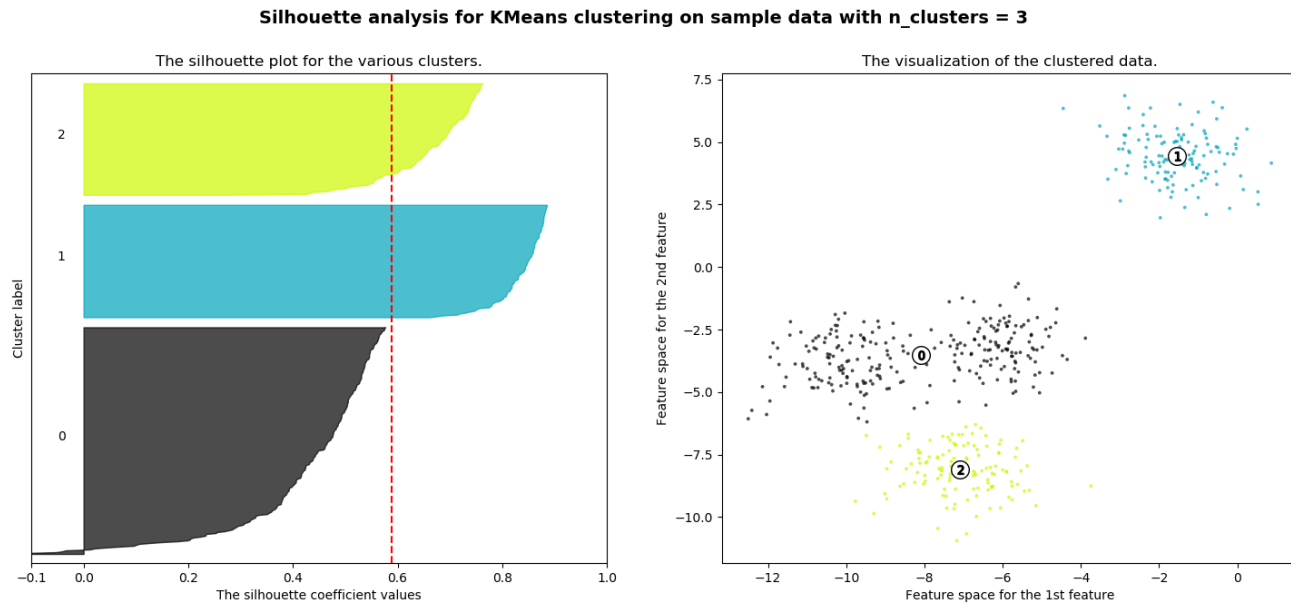
- Silhouette

**Silhouette analysis for KMeans clustering on sample data with  $n\_clusters = 2$**



# Mesures de performances / détermination du nombre de clusters

- Silhouette

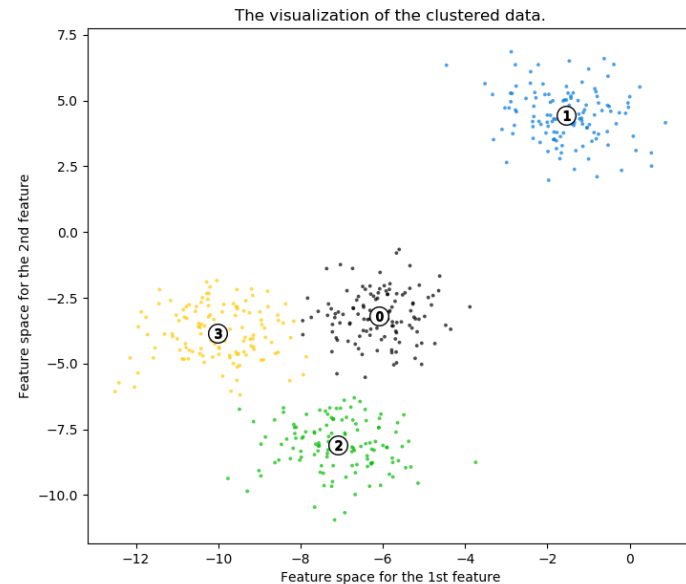
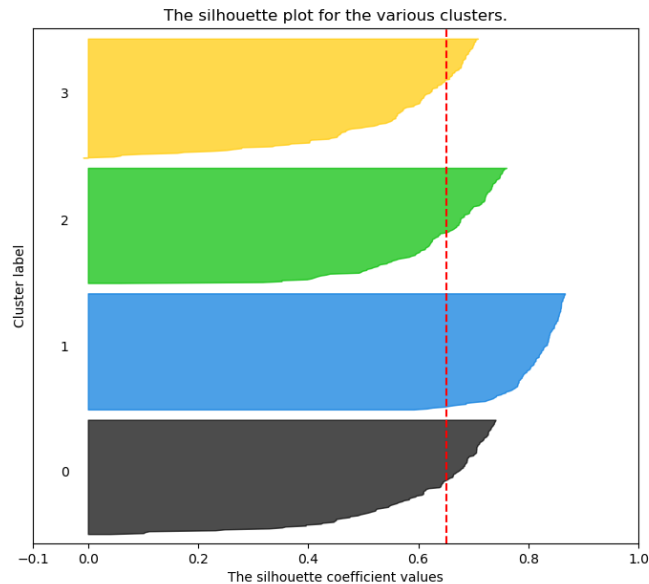




# Mesures de performances / détermination du nombre de clusters

- Silhouette

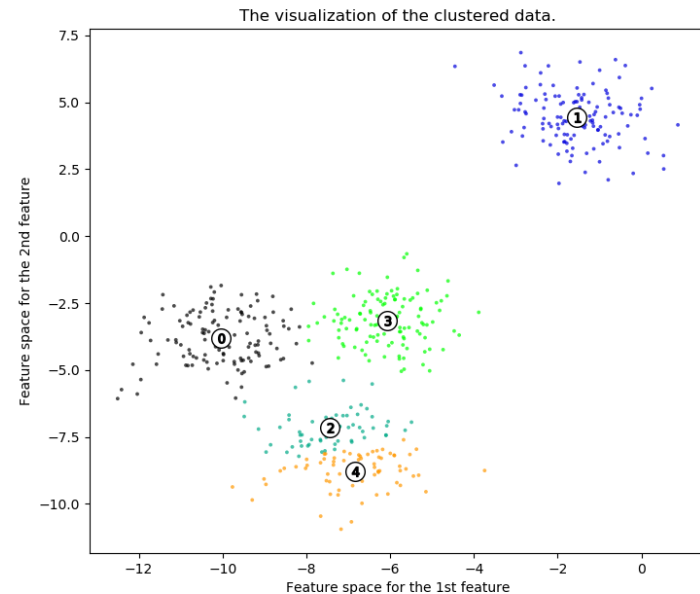
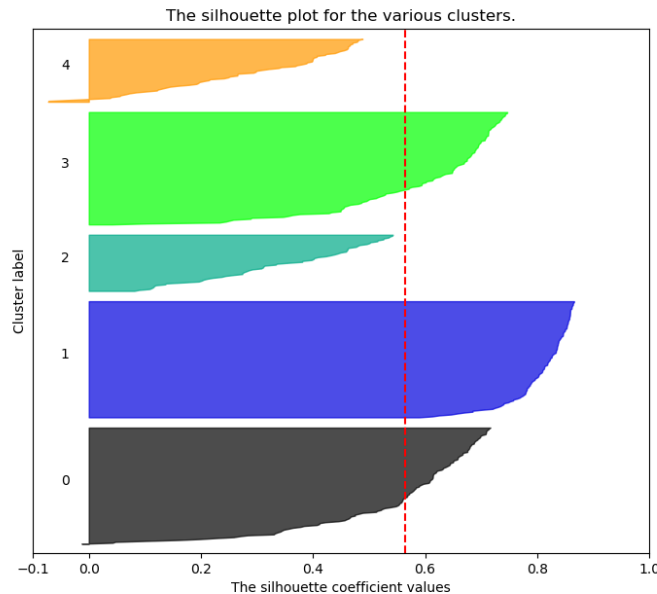
**Silhouette analysis for KMeans clustering on sample data with  $n\_clusters = 4$**



# Mesures de performances / détermination du nombre de clusters

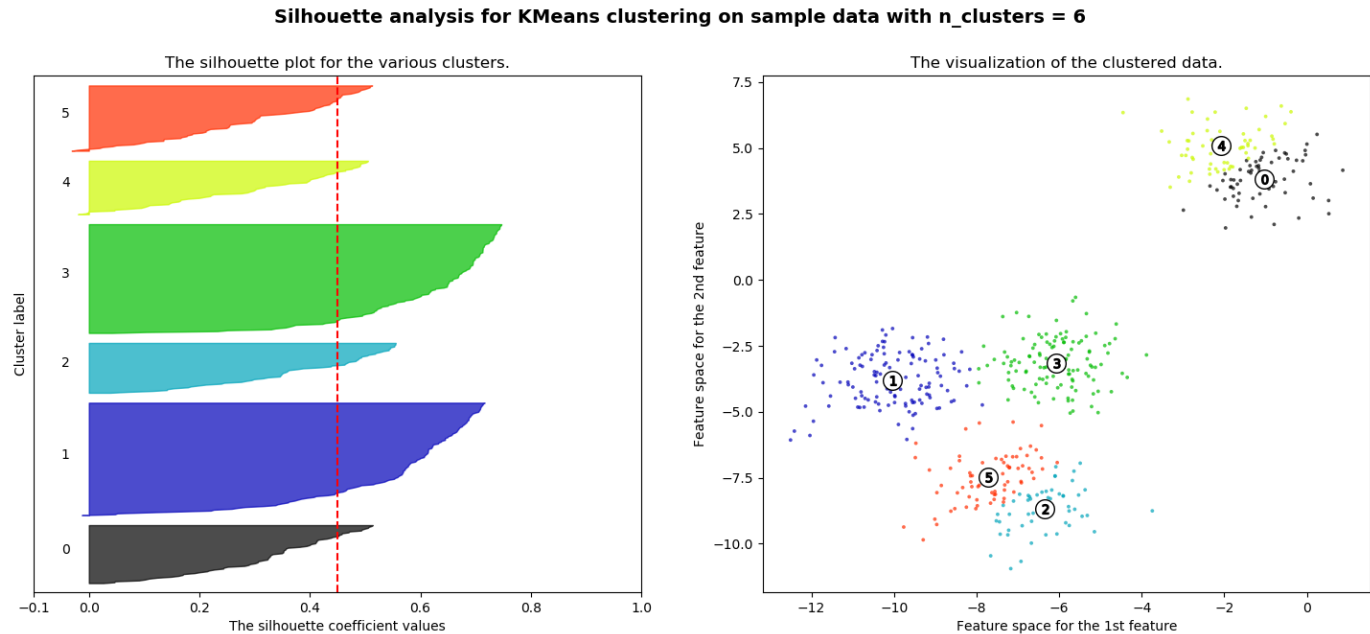
- Silhouette

**Silhouette analysis for KMeans clustering on sample data with  $n\_clusters = 5$**



# Mesures de performances / détermination du nombre de clusters

- Silhouette



# Davies-Bouldin et Silhouette en python

- **sklearn.metrics.davies\_bouldin\_values** : [https://scikit-learn.org/stable/modules/generated/sklearn.metrics.davies\\_bouldin\\_score.html](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.davies_bouldin_score.html)
- **sklearn.metrics.silhouette\_score** : calcule la silhouette moyenne. [https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette\\_score.html#sklearn.metrics.silhouette\\_score](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html#sklearn.metrics.silhouette_score)
- **sklearn.metrics.silhouette\_samples** : calcule la silhouette pour chaque point. [https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette\\_samples.html#sklearn.metrics.silhouette\\_samples](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_samples.html#sklearn.metrics.silhouette_samples)