Explication de K-means Clustering

K-means est un algorithme de clustering très populaire en **machine learning** et en **traitement du langage naturel (NLP)**. Il permet de regrouper des objets similaires en **K groupes** (ou clusters) en fonction de leurs caractéristiques.

Algorithme	Utilisation	Fonction
K-means	Machine learning, NLP	Regrouper des objets similaires en K groupes

Principe du K-means

1. Choix du nombre de clusters K

   a. L'utilisateur définit combien de groupes (K) il veut créer.
   b. Exemple : Si on analyse des avis clients, on peut choisir **K=3** pour regrouper les avis en **positifs, neutres et négatifs**.

2. **Initialisation des centroids**
   a. L'algorithme place **K points aléatoires** dans l'espace des données. Ces points sont appelés **centroïdes** et représentent le centre de chaque cluster.
3. **Assignation des points aux clusters**
   a. Chaque point (ex : un avis client) est attribué au **cluster le plus proche** en fonction d'une mesure de distance, souvent la **distance Euclidienne**.
4. **Recalcul des centroids**
   a. Une fois tous les points assignés, on **calcule le centre moyen** de chaque cluster et on **déplace le centroid** vers cette nouvelle position.
5. **Répétition du processus**
   a. On répète l'opération (étapes 3 et 4) **jusqu'à ce que les clusters deviennent stables** (c'est-à-dire que les centroïdes ne bougent plus significativement).

Étape	***Description***
Choix du nombre de clusters K	L'utilisateur définit combien de groupes (K) il veut créer. Exemple : K=3 pour

	regrouper les avis en positifs, neutres et négatifs.
Initialisation des centroids	L'algorithme place K points aléatoires dans l'espace des données. Ces points sont appelés centroïdes.
Assignation des points aux clusters	Chaque point est attribué au cluster le plus proche en fonction d'une mesure de distance, souvent la distance Euclidienne.
Recalcul des centroids	On calcule le centre moyen de chaque cluster et on déplace le centroid vers cette nouvelle position.
Répétition du processus	On répète les étapes 3 et 4 jusqu'à ce que les clusters deviennent stables.

Exemple concret : Segmentation des avis clients

Prenons un site e-commerce qui vend des chaussures. On collecte les commentaires des clients et on applique **K-means (K=3)** :

Avis client	*Cluster attribué*
"Super confortables !"	Cluster 1 : Confort
"Design élégant et tendance"	Cluster 2 : Esthétique
"Semelle trop fine, pas durable"	Cluster 3 : Qualité du produit

Résultat :

- On identifie les **thèmes clés des avis clients**.
- On peut améliorer la **qualité perçue du produit** en fonction des retours.

Avantages et Inconvénients

Avantages :

- Simple et rapide à exécuter.
- Fonctionne bien pour des **grands volumes de données**.
- Aide à **identifier des patterns** dans des données non structurées.

Inconvénients :

- Il faut choisir **K à l'avance** (souvent testé avec la méthode du coude).
- Sensible aux **valeurs aberrantes**.
- Suppose que les clusters sont **de forme sphérique et de taille similaire**.

Comment fonctionne DBSCAN ?

DBSCAN utilise deux **paramètres clés** :

- **ε (epsilon)** : distance maximale entre deux points pour être considérés comme voisins.
- **MinPts (Minimum Points)** : nombre minimum de points nécessaires pour former un cluster.

Paramètre	***Description***
ε (epsilon)	distance maximale entre deux points pour être considérés comme voisins
MinPts (Minimum Points)	nombre minimum de points nécessaires pour former un cluster

L'algorithme suit ces étapes :

1. **Définition des points** :
    a. Chaque point de données est soit :

i. Un **point central** (suffisamment de voisins dans un rayon ε).
ii. Un **point bordure** (appartient à un cluster mais n'a pas assez de voisins pour être un centre).
iii. Un **point bruit** (isolé, non assigné à un cluster).

2. **Formation des clusters** :
   a. DBSCAN explore chaque point non encore visité.
   b. Si un point est un **point central**, un **nouveau cluster** est créé.
   c. Tous les **points accessibles** dans un rayon ε sont ajoutés au cluster.
   d. Si un point est un **point bruit**, il est ignoré ou laissé comme **outlier**.
3. **Expansion des clusters** :
   a. DBSCAN continue d'ajouter des points voisins jusqu'à ce qu'aucun nouveau point ne puisse être ajouté.
   b. Une fois terminé, il passe au prochain point non visité.

Étape	***Description***
Définition des points	Chaque point de données est soit : un point central, un point bordure, un point bruit
Formation des clusters	DBSCAN explore chaque point non encore visité, crée un nouveau cluster pour un point central, ajoute les points

	accessibles dans un rayon ε, ignore ou laisse les points bruit comme outliers
Expansion des clusters	DBSCAN continue d'ajouter des points voisins jusqu'à ce qu'aucun nouveau point ne puisse être ajouté, passe au prochain point non visité

Exemple concret : Analyse des avis clients

Supposons que nous analysons les commentaires clients d'un site e-commerce.

Avis client	***Cluster attribué***
"Chaussures très confortables, top !"	Cluster 1 : Confort
"Design moderne et tendance"	Cluster 2 : Esthétique
"Semelle fragile, s'est usée vite"	Cluster 3 : Qualité
"Ce produit est une arnaque !"	Point bruit (outlier)

Pourquoi DBSCAN est utile ici ?

- Il **identifie des groupes naturels** sans imposer un nombre de clusters.
- Il **rejette les avis extrêmes** comme du bruit (ex. avis frauduleux ou isolés).
- Il gère bien des **groupes de tailles et formes différentes**.

Avantages et inconvénients de DBSCAN

Avantages :

✓ **Pas besoin de spécifier K** (contrairement à K-means).

✓ **Identifie automatiquement les anomalies** (*outliers*).

✓ Fonctionne bien avec **des clusters de formes irrégulières**.

✓ Idéal pour **des données bruitées et réelles** comme les avis clients.

Inconvénients :

✕ Choisir **ε et MinPts** peut être difficile.

✕ Moins performant si la densité des clusters est **très variable**.

✕ Peut être lent sur **de très grands datasets**.

📊 Comparaison entre DBSCAN et K-means

Critère	*DBSCAN*	*K-means*
Nombre de clusters	Détecté automatiquement	Doit être défini
Sensibilité aux outliers	Gère bien les outliers	Sensible aux outliers
Forme des clusters	Peut détecter des formes irrégulières	Suppose des formes sphériques
Performance sur grands datasets	Plus lent	Plus rapide
Facilité de paramétrage	Difficile	Facile (choix de K)

Comparaison avec DBSCAN

- **K-means** fonctionne bien quand **les clusters sont bien séparés**.
- **DBSCAN** est mieux adapté aux données où les **groupes ont des formes irrégulières** et où il y a du bruit.