ISM MP, Informatique
Année 2025/2026

TP Informatique 16

Exercice 1

L'algorithme des *k-moyennes* est une heuristique cherchant à partitionner un ensemble fini S de points de $\mathbb{R}^d$ en k sous-ensembles $\mathrm{S}_1, \mathrm{S}_2, \ldots, \mathrm{S}_k$ appelés *clusters* tels que la quantité

$$\mathrm{V}(\mathrm{S}_1, \mathrm{S}_2, \ldots, \mathrm{S}_k) = \sum_{i=1}^{k} \sum_{x \in \mathrm{S}_i} \|x - \bar{x}_i\|^2 \quad \text{avec} \quad \bar{x}_i = \frac{1}{|\mathrm{S}_i|} \sum_{x \in \mathrm{S}_i} x$$

soit minimale. Les isobarycentres $\bar{x}_i$ de chaque cluster sont appelés *centroïdes*.

Les états successifs des clusters et de leurs centroïdes dans l'algorithme des k-moyennes sont notés :

$$\forall \ell = 0, 1, 2 \ldots \qquad \mathrm{S}_1^{(\ell)}, \mathrm{S}_2^{(\ell)}, \ldots, \mathrm{S}_k^{(\ell)} \quad \text{et} \quad \bar{x_1}^{(\ell)}, \bar{x_2}^{(\ell)}, \ldots, \bar{x_k}^{(\ell)}$$

L'algorithme de classification non supervisé des k-moyennes consiste en les étapes suivantes :

- Initialisation (état $\ell = 0$) : chaque point de S est placé au hasard dans $\mathrm{S}_1^{(0)}, \mathrm{S}_2^{(0)}, \ldots, \mathrm{S}_k^{(0)}$, la variable $\mathrm{V}^{(0)}$ reçoît $+\infty$ et une variable booléenne de test de décroissance reçoit `True` ;
- Tant que le test de décroissance est vrai (on passe de l'état ℓ à $\ell + 1$) :
  - On calcule les centroïdes à savoir les $\bar{x_i}^{(\ell)} = \dfrac{1}{\left|\mathrm{S}_i^{(\ell)}\right|} \displaystyle\sum_{x \in \mathrm{S}_i^{(\ell)}} x$ pour $i \in [\![1 ; k]\!]$;
  - On calcule une nouvelle partition $\mathrm{S}_1^{(\ell+1)}, \mathrm{S}_2^{(\ell+1)}, \ldots, \mathrm{S}_k^{(\ell+1)}$ telle que chaque point $x \in \mathrm{S}$ est affecté dans $\mathrm{S}_i^{(\ell+1)}$ avec i indice pour lequel $\|x - \bar{x_i}^{(\ell)}\|$ est minimale ;
  - On calcule la nouvelle valeur de $\mathrm{V}^{(\ell+1)}$ variance intra-classe et la variable booléenne de test de décroissance reçoit la valeur de $\mathrm{V}^{(\ell+1)} < \mathrm{V}^{(\ell)}$.

Exécuter le fichier `TP16_EX01.py`. Celui-ci affiche le nuage de points obtenu par mélange de trois classes de points ayant été générés, au sein d'une même classe, selon des caractéristiques statistiques communes. Les lignes de code qui suivent sont la trame d'une implémentation de l'algorithme des k-moyennes.

1. Modifier l'une des affectations de l'initialisation (prévue pour éviter une boucle infinie lors du premier lancement).
2. Compléter l'initialisation de la partition. On utilisera la fonction `rd.randint(0,k)` qui renvoie la réalisation d'un tirage aléatoire uniforme dans $[\![0 ; k-1]\!]$.
3. Dans la boucle `while`, compléter le calcul des centroïdes, la mise à jour de la partition et la gestion du booléen `decrease`. On utilisera la fonction `alg.norm` pour le calcul de la norme euclidienne.
4. Tester le programme. On pourra modifier à son gré le choix de k, le choix des moyennes et écarts-types servant à générer le nuage de points, etc. ...

Exercice 2

Dans cet exercice, on se propose d'utiliser l'algorithme des k-moyennes pour réduire considérablement le nombre de couleurs dans une image. Chaque pixel est codé en trois nuances de rouge, vert, bleu avec 2^8 niveaux de nuance pour chaque couleur ce qui implique 2^{24} choix de couleurs par pixel.

Exécuter le fichier `TP16_EX02.py`. Celui-ci affiche l'image `Chateau_Sarran.jpg` dans sa configuration d'origine. L'image est ensuite stockée dans la variable `tab` qui est un tableau `ndarray` de taille $\mathrm{L} \times \mathrm{C} \times 3$ avec L le nombre de lignes et C et le nombre de colonnes. On choisit de considérer les coordonnées $(i, j) \in [\![0\,;\, \mathrm{L}-1]\!] \times [\![0\,;\, \mathrm{C}-1]\!]$ d'un pixel pour identifier un point de l'image. L'ensemble S est donc une partition de clusters où chaque cluster contient des couples repérant les points de l'image.

1. Modifier l'une des affectations de l'initialisation (prévue pour éviter une boucle infinie lors du premier lancement).
2. Dans la boucle `while`, compléter le calcul des centroïdes, la mise à jour de la partition et la gestion du booléen `decrease`. On tiendra compte de la nature des éléments de chaque cluster, à savoir des listes des deux coordonnées repérant un point de l'image. On utilisera la fonction `alg.norm` pour le calcul de la norme euclidienne.
3. Tester le programme. Expliquer le processus final de construction de l'image compressée.