Informatique pour tous - MP* - PC*

Partie 5 :
Apprentissage non supervisé et k-moyennes

Florent Pompigne
pompigne@crans.org

Lycée Buffon

année 2022/2023

Plan

Apprentissage non supervisé

Le principe de l'**apprentissage non supervisé** est d'écrire un programme identifiant la structure d'un ensemble de données non annotées.

Apprentissage non supervisé

Le principe de l'**apprentissage non supervisé** est d'écrire un programme identifiant la structure d'un ensemble de données non annotées.

Une tâche classique de l'apprentissage non supervisé est le partitionnement de données (*clustering*) : on souhaite identifier les groupes d'éléments proches apparaissant dans les données. À la différence de la tâche de classification dans l'apprentissage supervisé, on cherche donc à identifier les classes de l'échantillon, qui ne sont pas connues a priori.

Apprentissage non supervisé

Le principe de l'**apprentissage non supervisé** est d'écrire un programme identifiant la structure d'un ensemble de données non annotées.

Une tâche classique de l'apprentissage non supervisé est le partitionnement de données (*clustering*) : on souhaite identifier les groupes d'éléments proches apparaissant dans les données. À la différence de la tâche de classification dans l'apprentissage supervisé, on cherche donc à identifier les classes de l'échantillon, qui ne sont pas connues a priori.

Par exemple, si l'ensemble de données consiste en des images non annotées représentant des animaux, un programme de répartition de données devrait identifier un groupe d'images pour chaque espèce représentée.

Plan

Partitionnement de points d'un espace euclidien

On considère dans la suite un ensemble X de n points de $\mathbb{R}^d$ muni de la distance euclidienne, et un entier k. On souhaite en sortie obtenir la meilleure répartition de ces points en k groupes. On formalise cela en cherchant k parties disjointes $X_0, \ldots X_{k-1}$ de X, de barycentres (moyennes) respectifs $m_0, \ldots, m_{k-1}$, minimisant la quantité

$$\sum_{i=0}^{n-1} \sum_{x \in X_i} \|x - m_i\|^2$$

Partitionnement de points d'un espace euclidien

On considère dans la suite un ensemble X de n points de $\mathbb{R}^d$ muni de la distance euclidienne, et un entier k. On souhaite en sortie obtenir la meilleure répartition de ces points en k groupes. On formalise cela en cherchant k parties disjointes $X_0, \ldots X_{k-1}$ de X, de barycentres (moyennes) respectifs $m_0, \ldots, m_{k-1}$, minimisant la quantité

$$\sum_{i=0}^{n-1} \sum_{x \in X_i} \|x - m_i\|^2$$

Notons qu'on ne sait a priori rien de la structure ces n points, notamment s'ils admettent une répartition pertinente en k groupes.

Partitionnement de points d'un espace euclidien

On considère dans la suite un ensemble X de n points de $\mathbb{R}^d$ muni de la distance euclidienne, et un entier k. On souhaite en sortie obtenir la meilleure répartition de ces points en k groupes. On formalise cela en cherchant k parties disjointes $X_0, \ldots X_{k-1}$ de X, de barycentres (moyennes) respectifs $m_0, \ldots, m_{k-1}$, minimisant la quantité

$$\sum_{i=0}^{n-1} \sum_{x \in X_i} \|x - m_i\|^2$$

Notons qu'on ne sait a priori rien de la structure ces n points, notamment s'ils admettent une répartition pertinente en k groupes.

Puisqu'il y a $k^n/k!$ partitions possibles, ce problème est difficile à résoudre de façon exacte en complexité raisonnable. l'algorithme des **k-moyennes** est une méthode d'approximation se calculant en pratique en un temps raisonnable.

Plan

Pseudo-code de k-moyennes

```
k_moyennes(X, k):
    soit M une liste de $k$ moyennes
    tant que M est modifiée:
        soit P une liste de k parties vides
        pour chaque point p de X:
            soit i l'indice de la moyenne
                la plus proche de p
            on ajoute p à la partie d'indice i
        on recalcule la moyenne de chaque partie
    on renvoie P
```

Pseudo-code de k-moyennes

```
k_moyennes(X, k):
    soit M une liste de $k$ moyennes
    tant que M est modifiée:
        soit P une liste de k parties vides
        pour chaque point p de X:
            soit i l'indice de la moyenne
                la plus proche de p
            on ajoute p à la partie d'indice i
        on recalcule la moyenne de chaque partie
    on renvoie P
```

Remarques : Pour implémenter cet algorithme, il faudra donc écrire préalablement une fonction renvoyant la moyenne d'un ensemble de points, et une fonction renvoyant pour une liste de points l'indice du plus proche d'un point donné.

Propriétés de l'algorithme

Terminaison

On peut montrer qu'à chaque étape, la somme qu'on cherche à minimiser ne peut que diminuer. Comme il y a un nombre fini de partitionnements, cela garantit la terminaison de l'algorithme.

Propriétés de l'algorithme

Terminaison

On peut montrer qu'à chaque étape, la somme qu'on cherche à minimiser ne peut que diminuer. Comme il y a un nombre fini de partitionnements, cela garantit la terminaison de l'algorithme.

Correction

L'algorithme n'est pas correct, au sens où il n'y a aucune garantie que le minimum trouvé soit global. C'est un algorithme d'approximation.

Propriétés de l'algorithme

Terminaison

On peut montrer qu'à chaque étape, la somme qu'on cherche à minimiser ne peut que diminuer. Comme il y a un nombre fini de partitionnements, cela garantit la terminaison de l'algorithme.

Correction

L'algorithme n'est pas correct, au sens où il n'y a aucune garantie que le minimum trouvé soit global. C'est un algorithme d'approximation.

Complexité

La complexité est exponentielle en n dans le pire cas, mais polynomiale en moyenne.

Initialisation

Le résultat renvoyé par l'algorithme dépend fortement des valeurs initiales choisies pour les k moyennes.
Il existe différentes stratégies pour cette initialisation, en voici quelques exemples :

Initialisation

Le résultat renvoyé par l'algorithme dépend fortement des valeurs initiales choisies pour les k moyennes.
Il existe différentes stratégies pour cette initialisation, en voici quelques exemples :

- **uniforme** : on tire k points uniformément dans un ensemble fini. contenant X.

Initialisation

Le résultat renvoyé par l'algorithme dépend fortement des valeurs initiales choisies pour les k moyennes.
Il existe différentes stratégies pour cette initialisation, en voici quelques exemples :

- **uniforme** : on tire k points uniformément dans un ensemble fini. contenant X.
- **Forgy** : on tire uniformément k points de X.

Initialisation

Le résultat renvoyé par l'algorithme dépend fortement des valeurs initiales choisies pour les k moyennes.
Il existe différentes stratégies pour cette initialisation, en voici quelques exemples :

- **uniforme** : on tire k points uniformément dans un ensemble fini. contenant X.
- **Forgy** : on tire uniformément k points de X.
- **partition aléatoire** : pour chaque point de X, on tire uniformément dans quel groupe le placer, puis on calcule les moyennes des groupes ainsi formés.

Initialisation

Le résultat renvoyé par l'algorithme dépend fortement des valeurs initiales choisies pour les k moyennes.
Il existe différentes stratégies pour cette initialisation, en voici quelques exemples :

- **uniforme** : on tire k points uniformément dans un ensemble fini. contenant X.
- **Forgy** : on tire uniformément k points de X.
- **partition aléatoire** : pour chaque point de X, on tire uniformément dans quel groupe le placer, puis on calcule les moyennes des groupes ainsi formés.
- **k-moyennes++** : Un point de X est tiré uniformément pour la première moyenne, puis on tire chaque autre moyenne parmi les points restant, avec une probabilité proportionnelle au carré de la moyenne la plus proche.