Rapport du TP2 de MA 412

Paul Ezvan et Omar Givernaud

23 octobre 2009

Table des matières

1 Introduction

Le but de ce TP est d'étudier les algorithmes de classification vu en cours, première l'algorithme des K-means puis celui de construction ascendante.

2 Algorithme des k-means

2.1 Étude de l'algorithme

Nous utilisâmes la fonction kmeans de Matlab afin de réaliser une classification à l'aide de l'algorithme des k-means étudié en cours. Cette fonction permet de spécifier la manière dont sont choisis les centres initiaux. On peut également afficher le critère de la somme des inerties afin de déterminer quel choix de centre est le plus efficace. Les deux méthodes que nous pouvons utiliser choisissent les centres initiaux de manière aléatoire, la première parmi les points à classifier, la deuxième sur l'espace formé par ces points. Les deux ont une efficacité similaire. La méthode de preclustering n'est pas utile dans notre cas car elle utile dans le cas d'un grand nombre de points.

La fonction kmeans permet également d'être exécutée plusieurs fois en choisissant à chaque fois des centres initiaux différents, et de retourner le résultat optimum. Malheureusement elle ne revois pas le critère associé à ce résultat optimum, ce qui ne permet pas de comparer le gain obtenu.

Nous représentâmes la classification obtenue dans le plan principal à l'aide de la fonction fournie classr.

2.2 Choix du nombre de classes

Nous déterminâmes ensuite le nombre de classes le plus adapté à l'aide de la fonction silhouette. Celle-ci calcule pour chaque point la fonction suivante :

$S(i) = (min(b(i,:),2) - a(i))./max(a(i), min(b(i,:)))$

a(i) est la distance moyenne du point i aux autres points de sa classe, et b(i,k) est la distance moyenne du point i aux points des autres classes k.

Plus le résultat est proche de un plus la classification est pertinente, en effet dans le cas où S(i) = 1, a(i) = 0 et min(b(i, :)) > a(i).

La classification à trois classes est la plus homogène.

2.3 Calcul du critère

Nous écrivîmes ensuite une fonction permettant de calculer le critère pour une classification. Nous pûmes aisément en vérifier le résultat car la fonction kmeans permet également d'afficher ce critère.

```
% calcule le critère de la somme des inerties
% X : données
% L : classification
% k : nombre de classes
```

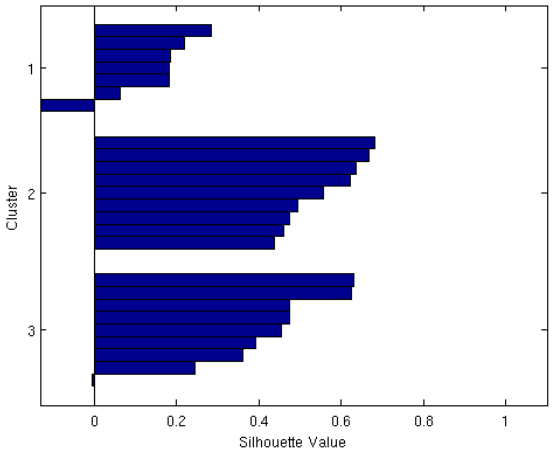

FIGURE 1 – Silhouette avec trois classes

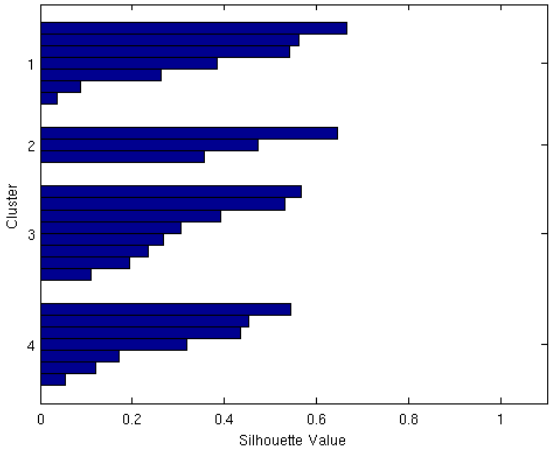

FIGURE 2 – Silhouette avec quatre classes

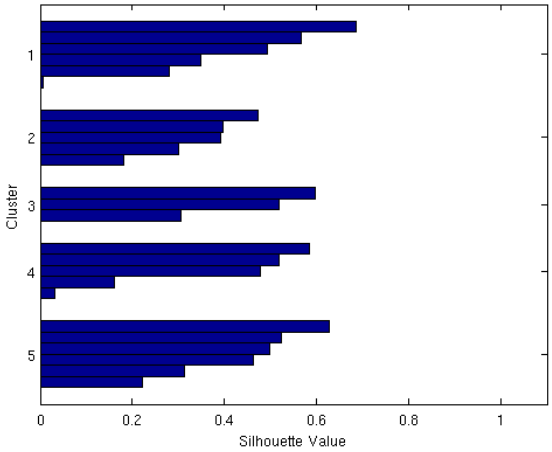

FIGURE 3 – Silhouette avec cinq classes

```
function [d] = critere(X,L,k)
% Centre de gravité G
G=zeros(k,6);
% Nombre de points dans chaque classe II
II=zeros(k,6);
d=0;
for i = 1 : size(X,1)
        G(L(i),:)=X(i,:) + G(L(i),:);
        II(L(i),:)=II(L(i),:) + 1;
end
G=G./II;
for i = 1 : size(X,1)
        d = d + sum((X(i,:)-G(L(i),:)).^2);
end
end
```

3 Construction ascendante hiérarchique

Nous étudiâmes ensuite une autre méthode de classification, la construction ascendante hiérarchique. La réalisation de cet algorithme est facilitée par la présence dans Matlab de fonctions correspondants au étapes de l'algorithme.

Premièrement la fonction *pdist* permet de générer la matrice de distance entre chaque point.

Ensuite la fonction *linkage* génère un arbre binaire représentant la classification hiérarchique.

La fonction *cluster* permet le vecteur de classes permettant d'associer chaque point à sa classe. Cela permet ensuite d'utilise notre fonction critere sur la classification réalisée afin de la comparer à ce qu'on obtient avec l'algorithme des k-means.

La fonction *dendrogram* permet de représenter le dendrogramme de la classification hiérarchique.

Voici le code que nous utilisâmes.

```
% génère la classification par
% construction ascendante hiérarchique
% X : données
% k : nombre de classes
% dt : type de distance
%

function [C, L] = casc(X,k,dt)
        D = pdist(X,'euclidean');
        L = linkage(D,dt);
        C = cluster(L,'maxclust',k);
end

% Classification par construction
% ascendante hiérarchique
[C,L]=casc(Y,3,'complete')
% affiche du dendrogramme
dendrogram(L)
```

Nous utilisâmes ensuite la fonction critere pour comparer les différents types de distance utilisés par la fonction linkage.

```
 % K-means avec un échantillon de centres initiaux.
critere(Y,L0,3)
ans = 80.3957
% K-means avec 1000 échantillons de centres initiaux.
critere(Y,L3,3)
ans = 77.3986
% Critère de distance du lien minimale
critere(Y,casc(Y,3,'single'),3)
ans = 106.5850
% Critère de distance du lien maximale
critere(Y,casc(Y,3,'complete'),3)
ans = 83.9056
% Critère de distance moyenne sans poids
critere(Y,casc(Y,3,'average'),3)
ans = 83.0781
% Critère de distance moyenne avec poids
critere(Y,casc(Y,3,'weighted'),3)
```

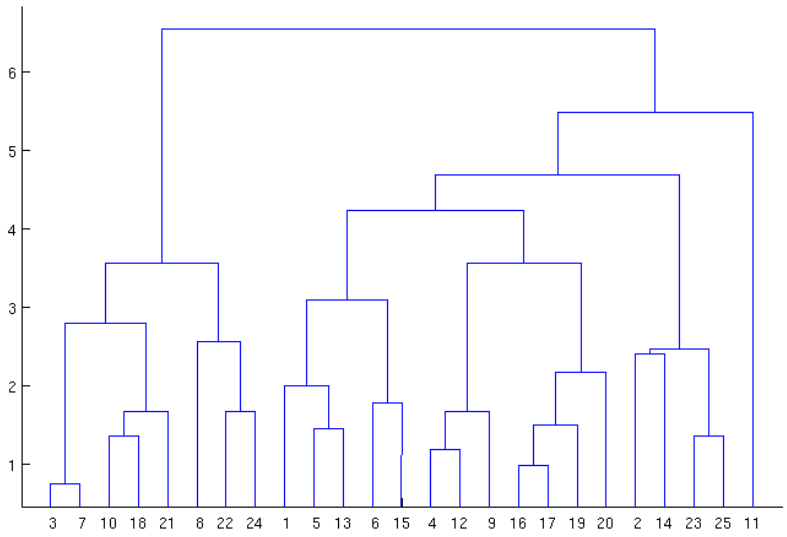

FIGURE 4 – Dendrogramme

```
ans = 83.0781
% Critère des centres de gravités sans poids
critere(Y, casc(Y,3,'centroid'),3)
ans = 126.4047
% Critère des centres de gravités avec poids
critere(Y, casc(Y,3,'median'),3)
ans = 84.9816
% Critère de variance minimale ou de Ward
critere(Y, casc(Y,3,'ward'),3)
ans = 79.1874
```

Les critères donnant le meilleur résultat sont ceux de la distance moyenne et de Ward. Le résultat obtenu par l'algorithme des k-means est meilleur.

Seul le critère de Ward est aussi efficace que l'algorithme des k-means avec un échantillon de centres initiaux.

4 Conclusion

Ce TP nous permit de comparer les différents algorithmes de classification. L'algorithme des K-means est efficace pour un nombre de classes donné. Les algorithmes par construction hiérarchique le sont moins mais à partir d'une construction hiérarchique on peut déterminer tous les classifications en k classes possibles.