Rapport INF 413
« Etude et implémentation de l'algorithme des k-means »

LIN Sheng
SHI Jinglei

TELECOM Bretagne
Institut Mines-Télécom

Sommaire

1. INTRODUCTION

L'algorithme k-means est une méthode d'origine de traitement du signal. Il est souvent utilisé dans le partitionnement de données (ou cluster analysis in data mining en anglais). L'objectif de k-means est de partitionne n observations dans k classes, avec la plus petite moyenne. Cette espace de données est partitionnée en cellules de Voronoi.
La complexité de ce genre de problème est NP difficile, cependant il y a des algorithmes heuristique qui sont efficaces et convergent rapidement. Le k-means a tendance de classifier vers une répartition d'étendue spatiale, mais dans la vraie vie, cela peut être de différents formes.

2. L'ALGORITHME DES K-MEANS

2.1 FONCTIONNEMENT

Le k-means emploie une technique itérative. Traditionnellement, on fixe une valeur k, qui signifie le nombre de groupes que l'on veut créer. Le problème est, que l'on ne peut pas prédire exactement le nombre de groupe sans connaissance de données. Ce problème peut être amélioré avec unu façon alternative, qui s'appelle k-means++(nous allons pas détailler dans cet article).
Le k-means consiste à 3 étapes :
D'abord, il faut initialiser l'espace en k groupes, avec k centres choisies aléatoirement ou de certaine manières.
L'étape 2, affecter les données au groupe le plus proche, pour que l'ensemble de distance de l'espace soit minimal.
L'étape 3, mettre à jour le centre (ou représentant) de chaque groupe.
Finalement, vérifier une condition d'arrêt, si la répartition ne suffit pas la condition d'arrêt, revenir à l'étape 2.

2.2 VARIANTES

Dans cet article, les variantes de l'algorithme sont : l'algorithme d'initialisation, les méthodes pour calculer les distances. Les paramètres de conditions d'arrêt. Nous allons préciser les méthodes et algorithmes dans la partie implémentation.

2.3 DIFFICULTES LIEES AUX K-MEANS

Les difficultés liées aux k-means est d'abord le façon d'initialiser. Afin que l'algorithme k-means converge rapidement et avec une meilleure performance, il est important de bien initialiser les centres.

Selon la distribution, la condition d'arrêt est aussi primordiale pour avoir une répartition convergente. Nous allons proposer une méthode mathématique (le choix de paramètre est ainsi plus important que l'algorithme).

3. IMPLEMENTATION

3.1 INITIALISATION

Dans cet articles, nous allons implémenter trois méthodes d'initialisation. Le premier est d'initialisation aléatoire. Dans un premier temps, c'est une méthode stupide et paresseux. Mais en fait, nous avons constaté que les données convergent rapidement vers un résultat acceptable, voire meilleure qu'avec une autre méthode qu'on va présenter.

Ensuite, nous avons décidé de trouver k centres en calculant le plus grand distance minimum (en anglais, Largest Mininum Distance). Le processus est d'abord choisir aléatoirement un point de départ, et le mettre dans un groupe où on mettrai tous les centres choisis. Ensuite, chercher dans l'ensemble de l'espace, un deuxième centre qui a le plus grand distance possible avec le premier point choisi. Une étape itérative, c'est de calculer les plus petites sommes de distances entre chaque point, aux centres choisies, et choisir celui qui a la plus grande sommes comme la prochaine centre. Faire l'étape itérative jusqu'à ce qu'on trouve tous les k centres. Nous avons constaté que le performance est relativement meilleur même si l'algorithme suivant paraît beaucoup plus sophistiqué !

Finallement, nous allons présenter un algorithme qui permet d'initialiser avec un nombre de centres variable. C'est l'algorithme de canopée (canopy algorithm en anglais) :

Pour commencer il faut choisir deux paramètres : la distance « lâche » T1(the loose distance) et la distance « serrée » T2(the tight distance), avec T1>T2 (ce sont ces deux paramètres qui vont influencer le performance).

Ensuite, on choisit d'abord aléatoirement un point de départ. Et on définit la première canopée et enlève le point dans l'espace initial.

Si la distance d'un point au centre de cette canopée, noté D, est inférieur à T1 (D<T1), rajouter le point dans cette canopée.

Si cette distance est aussi inférieur à T2 (D<T2), on l'enlève de l'espace initial.

Répète les étapes précédentes jusqu'à ce que l'espace initial soit vide ou avec une limite pour les itérations.

Cette méthode, théoriquement meilleur que les deux précédentes, est en pratique très dépendant des deux paramètres et la limite d'itération.

3.2 DISTANCES

Nous avons implémenté 3 méthodes pour calculer la distance. La première, est la valeur absolue, entre deux paramètres, (the City Block Distance). Le deuxième, est la distance Euclidéenne. Enfin, nous avons implémenté la distance Minkowski, c'est de prendre à la place de la puissance deux, on généralise avec un paramètre λ ; City Block est en fait le cas spécial quand λ=1 et Euclidéenneλ=2. Tous les algorithme prennent en compte le poids de chaque paramètres.

3.3 CONDITIONS D'ARRET

La condition d'arrêt est basé sur la convergeance de l'algorithme k-means. Elle consiste à deux partie :

Dans un premier temps, pour éviter la situation où l'algorithme arrive dans un état d'oscillation(qui ne converge plus). Nous avons mise en place un maximum nombre d'itération (Le choix de cette paramètre est très important).

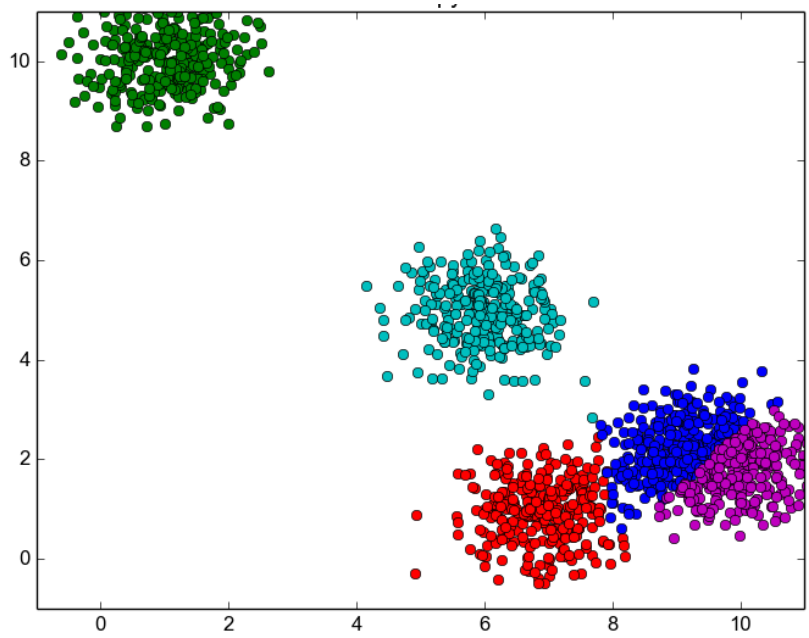
Ensuite, pour que la convergeance tend vers un état stable, on compare chaque fois la différence entre deux générations, et on met un seuil ε à sortir de la boucle.

4. EXPERIMENTATION

4.1 DESCRIPTION DES DONNEES

Pour avoir des expériences moins « artificielles » et plus convaincant, nous allons générer les données en deux dimentions(pour que tous s'affiche dans l'image), avec distributions gaussiennes, les centres étant choisis aléatoirement entre 0 et 10 (dans les deux axes).

Voisi un exemple des données en image:



4.2 DESCRIPTION DES EXPERIENCES

D'abord, nous avons testé les paramètres de l'algorithme de canopée en fixant autres variantes.

Ensuite, nous avons testé les différents façons pour calculer la distance (λ varie de 1 à 5).

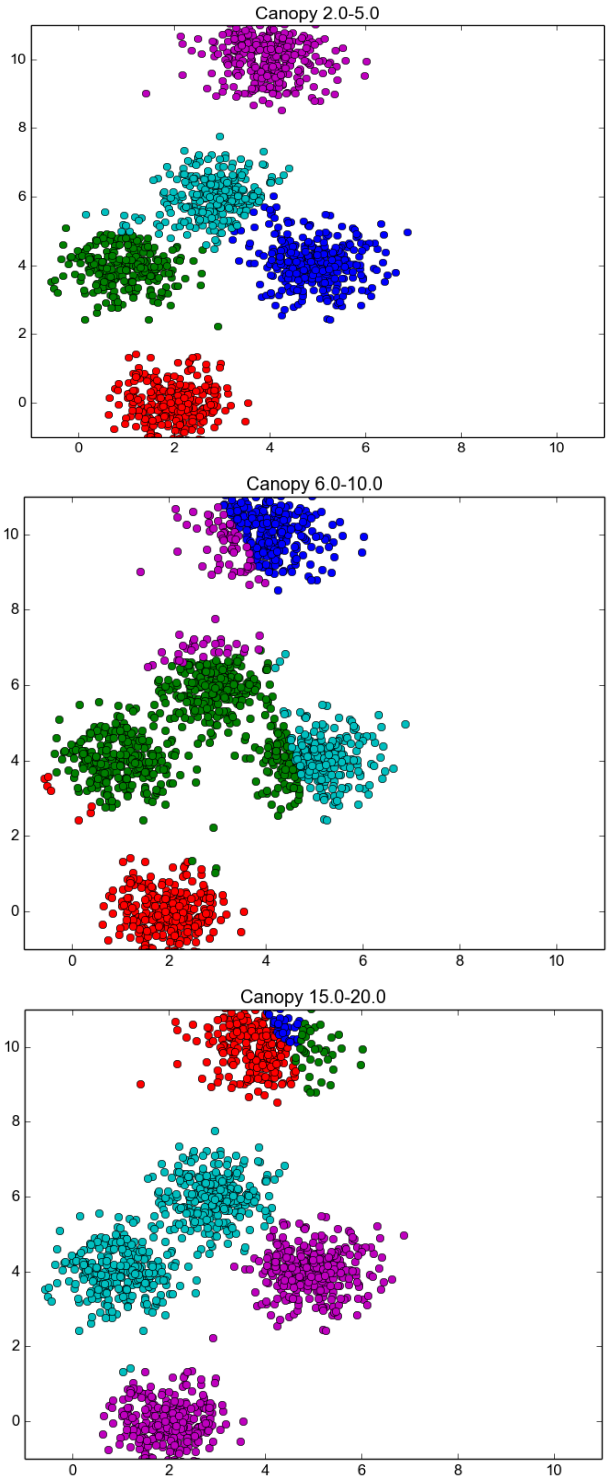
Et finallement, nous avons testé les trois méthodes d'initialisation, en prenant la meilleure méthode pour calculer la distance.

4.3 RESULTATS

4.3.1 Paramètres pour l'algorithme de canopée

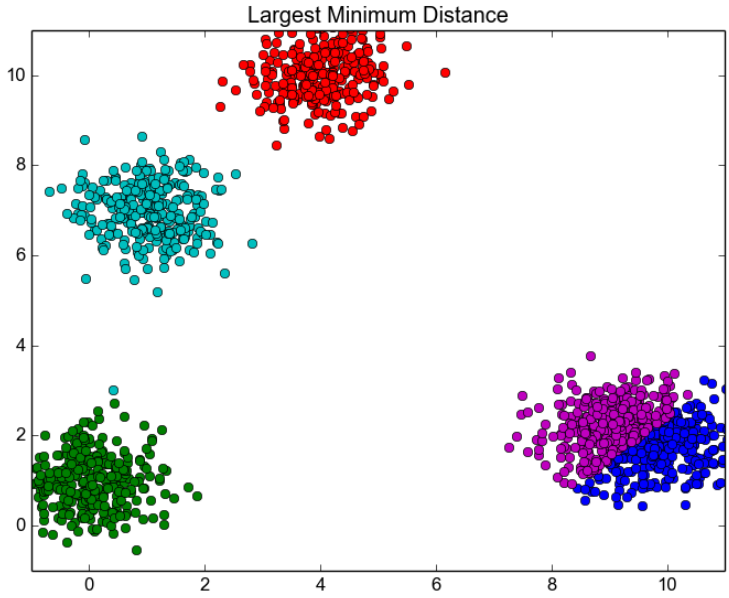
Nous pouvons constater que, dans une domaine acceptable, plus les paramètres sont petites, meilleure la performance. Mais si on les choisit trop petites, on va obtenir trop de canopées, ce qui n'est pas souhaitable. Si on choisit des paramètres

trop petites, et on limites le nombre de canopées, ce serait identique comme on choisit aléatoirement .

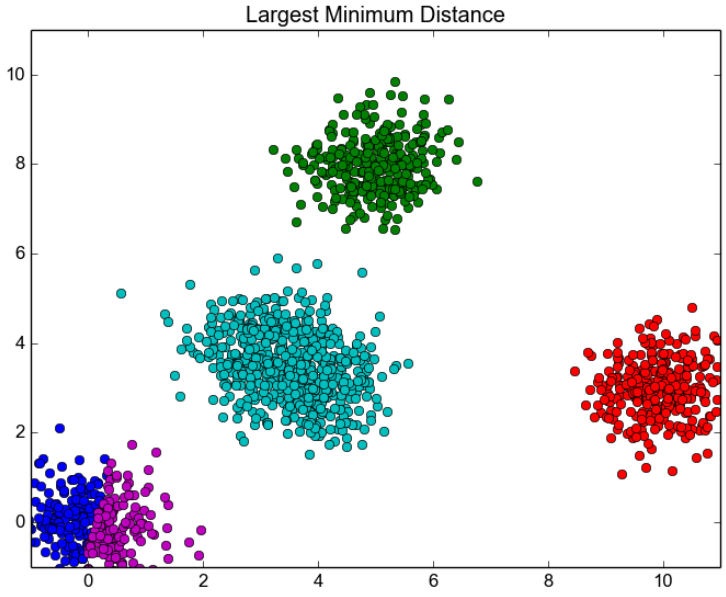


4.3.2 Choix pour calculer la distance

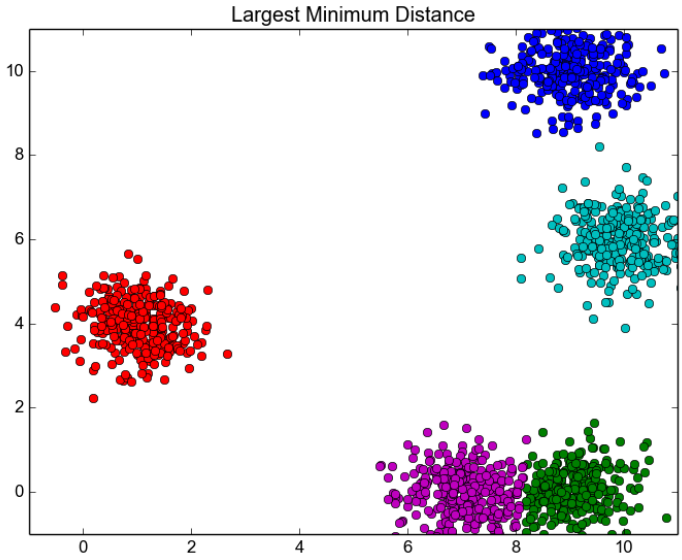
Nous avons constaté que, généralement, plus λ est grande, meilleur le résultat. Mais, à la fin d'expériences, nous avons trouvé que pour λ un nombre impair, la performance est légèrement meilleur qu'un nombre pair. Nous avons implémenté, pour différents λ, avec les trois algorithmes. Ici, on illustre avec les résultats de LMD, ce qui a un meilleur performance.



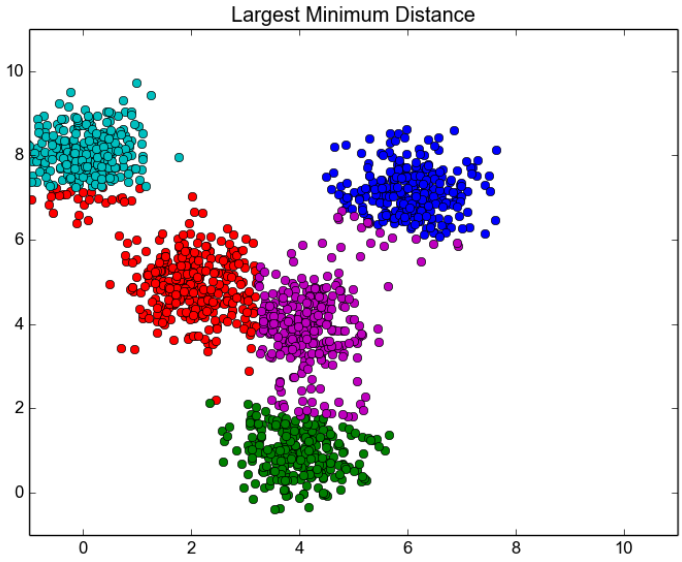
Quand λ =1, dans le cas de city block, on a constaté que pour les données bien distribuées, le City Block est efficace. Mais si les données se croisent dans l'espace, cela ne va pas fonctionner.



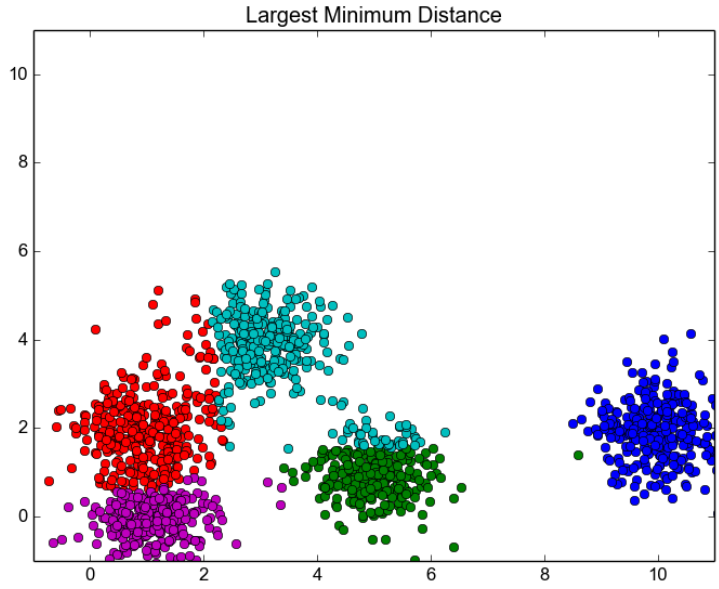
Quand λ = 2, la distance euclidéenne.



Quand λ = 3, on voit que la performance est beaucoup meilleur voire quasi parfaite, mais c'est aussi grâce aux données générées.



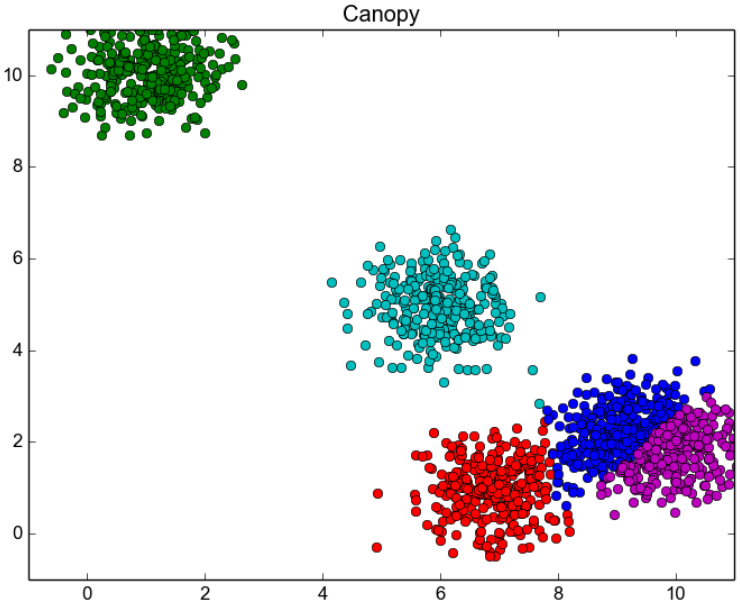
Quand λ =4, on a constaté des ambiguïtés sur les bords de groupes. C'est pourquoi on se demande s'il est commun que plus λ est grande, meilleur la performance.



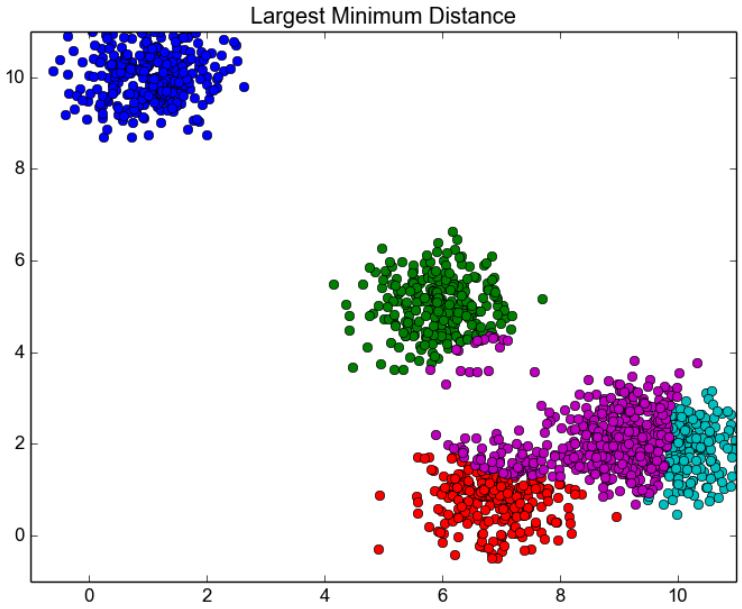
Quand λ =5, on voit bien que même si les données générées sont trop proches, voire fusionnées, l'algorithme arrive à apercevoir les groupes. Avec moins d'ambiguïté qu'avec λ =4.

4.3.3 Choix d'algorithme pour l'initialisation

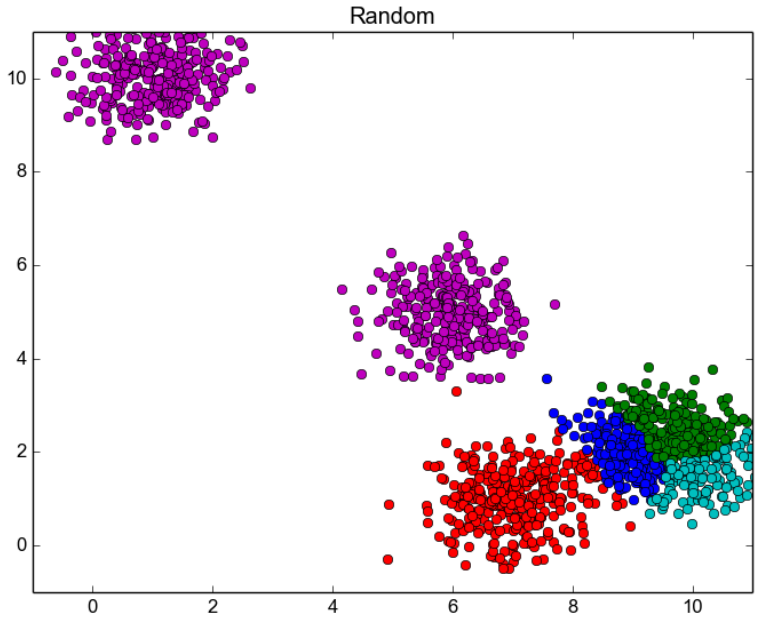
Nous avons choisi, pour le choix d'algorithme, λ =7 (qui est un nombre impair et plus grand). Ensuite, nous fixe les meilleurs paramètres pour l'initialisation .(2-5)



L'algorithme de canopée dépend beaucoup de paramètres et les données générées, si on ne peut pas prélire les données (comme dans notre cas), on ne pourrais pas améliorer la performance de l'algorithme. Mais généralement, il y a moins d'ambiguïté pour cet algorithme.



L'algorithme Largest Minimum est plus stable mais moins performant que l'algorithme de canopée. Il dépend de points de départ, la façon de calculer la distance, et le changement de centres. Il est très possible comme nous voyons dans l'image, de fusionner et séparer les données dans une zone floue.



L'algorithme de Random est, comme nous voyons, dépendant de la structure et les points d'initialisation. Si nous avons choisi des points d'initialisation dans un région voisinage, on pourrait y avoir plusieurs groupe. Et par contre, dans un autre région plus épars, où il y a moins de points de départ. C'est très possible d'avoir un centre entre deux groupes, et les deux groupes sont combiné en deux. (comme dans l'exemple).

5. CONCLUSION :

D’auprès les différents résultats que l’on obtient avec différents façons et paramètres, on peut constater que l’ algorithme de Random avec la moindre complexité , est moins stable que l’ algorithme de LMA, pour les mêmes données. D’habitude, l’algorithme LMA peut souvent mieux distinguer et séparer les données, l’algorithme Canopy,qui a plus de complexitè, a un avantage de décider le nombre de groupe automatiquement, mais le résultat de classfier est largement conditionné par les paramètres qu’on choisit. Le meuilleur cas pour le Canopy,même si il y a un peu d’ambiguité, le résultat peut aussi être acceptable,le pire cas, pour le groupe hyper simple, quelque fois cet algorithme rend un résultat stupid,d’ailleurs, ces trois algorithmes sont déterminé par le choix du mesure de distance, comme l’éconné précédent, pour λ un nombre impair, la performance est légèrement meilleur qu’un nombre pair,et pour un nombre pair, mais on ne peut pas dire exactement que le plus grand le nombre est, le meilleur résultat on obtient, généralement, on pense que pour λ=4, le résultat peut être acceptable.

Technopôle Brest-Iroise
CS 83818
29238 Brest Cedex 3
France
+33 (0)2 29 00 11 11
www.telecom-bretagne.eu