Rapport Technique GRAPPA
GRAPPA-0105

SSC : Statistical Subspace Clustering

Laurent Candillier[1,2], Isabelle Tellier[1], Fabien Torre[1], Olivier Bousquet[2]

[1] GRAppA - Université Charles de Gaulle - Lille 3
http://www.grappa.univ-lille3.fr

[2] Pertinence - 32 rue des Jeûneurs - 75002 Paris
http://www.pertinence.com

Résumé

Cet article se place dans le cadre du subspace clustering. Apparue récemment, la problématique du subspace clustering est double : identifier simultanément les clusters et le sous-espace spécifique dans lequel chacun est défini, et caractériser chaque cluster par un nombre minimal de dimensions, permettant ainsi une présentation des résultats compréhensible par un expert du domaine d'application. Les méthodes proposées jusqu'à présent pour cette tâche ont le défaut de se restreindre à un cadre numérique. L'objectif de cet article est de proposer un algorithme de subspace clustering capable de traiter des données décrites à la fois par des attributs continus et des attributs catégoriels. Nous présentons une méthode basée sur l'algorithme classique EM mais opérant sur un modèle simplifié des données et suivi d'une technique originale de sélection d'attributs pour ne garder que les dimensions pertinentes de chaque cluster. Les expérimentations présentées ensuite, et menées sur des bases de données aussi bien artificielles que réelles, montrent le comportement de notre algorithme : celui-ci présente des résultats robustes en termes de qualité de la classification et de compréhensibilité des clusters obtenus.

Abstract

In this paper, we focus on the task of subspace clustering, that has two goals : simultaneously identify the clusters and the subspaces in which each of them is defined, and describe each cluster with as few dimensions as possible, so that the results are easily interpretable for a human user. One default of existing methods is that they only consider numerical databases. The aim of this paper is to propose a new subspace clustering algorithm, able to tackle databases that may contain continuous as well as discretes attributes. We present a method based on the classical EM algorithm, but applied to a simplified model, and followed with an original technique of feature selection that only keeps dimensions that are relevant to each cluster. Experiments, conducted on artifical as well as real databases, show that our algorithm gives robust results, in terms of classification and interpretability of the output.

SSC : Statistical Subspace Clustering

Laurent Candillier[1,2], Isabelle Tellier[1], Fabien Torre[1], Olivier Bousquet[2]

Introduction

Face aux quantités d'informations qui ne cessent d'augmenter dans les bases de données du monde entier, l'extraction automatique de connaissances à partir de ces bases et les techniques de visualisation des résultats sont devenues indispensables. C'est la raison d'être de la *fouille de données*. Dans ce cadre, l'apprentissage non supervisé (ou *clustering*) est depuis longtemps utilisé pour identifier les groupes (ou *clusters*) d'éléments similaires (voir Berkhin 2002 pour un état de l'art). Une problématique supplémentaire apparaît face à des bases de données de grande dimensionnalité : dans ce cas, les groupes peuvent être caractérisés uniquement par certains sous-ensembles de dimensions et ces dimensions pertinentes peuvent être différentes d'un groupe à l'autre. Sur de tels problèmes, les techniques classiques de *clustering* fonctionnent mal car elles sont systématiquement fondées sur une distance entre objets définie globalement dans l'espace de description, et ces méthodes ne peuvent donc pas appréhender le fait que la notion de similarité varie d'un groupe à l'autre.

Une nouvelle problématique est donc apparue récemment, celle du *subspace clustering*, dont l'enjeu est de cibler les groupes d'objets et, pour chacun, le *sous-espace spécifique* dans lequel il est défini *. Et cet objectif s'accompagne d'un second : celui de fournir une description compréhensible des groupes identifiés. Les méthodes proposées jusqu'à présent pour cette tâche se sont focalisées sur le premier objectif, celui de trouver des groupes dans des sous-espaces différents, et ont négligé le second, celui de produire un résultat compréhensible. De plus, la partie expérimentale de ces travaux porte exclusivement sur des données numériques.

L'objectif de cet article est de proposer un algorithme de *subspace clustering* capable de traiter des données décrites à la fois par des attributs continus et des attributs catégoriels, demandant à l'utilisateur de régler le moins de paramètres possible, et fournissant en sortie une représentation simple des clusters identifiés. Nous nous basons pour cela sur l'algorithme EM adapté au *clustering* (Ye et al. 2003). Nous en proposons une version simplifiée en ajoutant l'hypothèse que les données sont générées selon des distributions indépendantes sur chaque dimension. Ceci nous permet d'en dériver une théorie compréhensible sous forme de règles puisque chaque dimension est caractérisée indépendemment des autres.

La suite de l'article est organisée comme suit : dans la section 1, nous présentons les méthodes existantes de *subspace clustering* et discutons de leurs performances ; puis nous détaillons, dans la section 2, la méthode que nous proposons ; les expérimentations présentées ensuite dans la section 3, menées sur des bases de données aussi bien artificielles que réelles, présentent les résultats de notre algorithme ; et nous terminons dans la section 4 par quelques conclusions et perspectives ouvertes par ce travail.

1 Le subspace clustering

La problématique du *subspace clustering* a été introduite pour la première fois dans (Agrawal et al. 1998). De nombreuses méthodes ont émergées depuis, parmi lesquelles deux familles peuvent être distinguées :

1. les méthodes *ascendantes sur les dimensions* (considérant des sous-espaces de dimensionnalité de plus en plus grande), créant un ensemble de clusters qui peuvent se chevaucher : (Agrawal

*la différence avec la problématique de la sélection d'attributs (ou *feature selection*) est que le sous-espace ciblé est local à chaque cluster, et non global à tous (Parsons et al. 2004).

et al. 1998), (Cheng et al. 1999), (Nagesh et al. 1999), (Procopiuc et al. 2002), (Kailing et al. 2004).

2. et les méthodes optimisant une partition de l'ensemble des objets : (Aggarwal et al. 1999), (Woo et Lee 2002), (Yip et al. 2003), (Sarafis et al. 2003), (Domeniconi et al. 2004).

Dans (Parsons et al. 2004), les auteurs ont étudié et comparé ces méthodes. Toutes sont capables de retrouver efficacement les clusters et leur sous-espace spécifique, mais elles nécessitent souvent des paramètres difficiles à régler par l'utilisateur et influant sur leurs performances (seuil de densité, nombre moyen de dimensions caractéristiques des clusters, distance minimale entre clusters, etc.) De plus, même si une adaptation aux données catégorielles a été proposée pour les méthodes ascendantes, tous les tests ont été effectués sur des bases de données exclusivement numériques.

Enfin, notons qu'aucune proposition aboutie de présentation simple des résultats n'a été effectuée. Ce point est pourtant crucial car même si la dimensionnalité des clusters a été réduite dans les sous-espaces qui leur sont propres, celle-ci peut encore être trop élevée pour qu'un expert du domaine d'application puisse appréhender le résultat. Nous verrons qu'il est pourtant souvent possible d'ignorer certaines de ces dimensions, tout en conservant le même partitionnement des objets.

La section suivante présente un nouvel algorithme de *subspace clustering*, s'apparentant à la famille des méthodes par partition mais fournissant en sortie un ensemble de clusters représentés sous forme de règles qui peuvent se chevaucher. Il suppose la donnée du nombre de clusters recherchés.

2 Algorithme SSC

Notre méthode est basée sur l'algorithme EM, que nous présentons d'abord. La suite traite de la modélisation des données utilisée dans notre algorithme. Puis nous détaillons les différentes étapes de cet algorithme, et finissons en présentant une technique originale de sélection d'attributs permettant une présentation compréhensible des résultats obtenus.

2.1 L'algorithme EM

Étant donné un modèle paramétrique, l'algorithme EM permet de trouver les paramètres du modèle qui correspondent le mieux aux données. Il utilise pour cela une répétition de deux phases, permettant d'améliorer le modèle au fur et à mesure :

1. *Expectation* : trouver les classes attendues pour chaque objet en fonction des paramètres courants du modèle.
2. *Maximization* : calculer les nouveaux paramètres du modèle en fonction des nouvelles attributions de classes aux objets.

Cet algorithme générique peut-être instancié de la manière suivante pour résoudre une tâche de clustering : les classes sont les clusters (il faudra fournir en entrée de l'algorithme le nombre de clusters recherchés) ; et le modèle utilisé classiquement est un mélange de gaussiennes avec l'hypothèse que chaque cluster a été généré selon une distribution gaussienne, modélisée par un centre et une matrice de covariance sur les dimensions. Dans ce cas, dans la première phase (*Expectation*), on calcule les paramètres cachés du modèle, qui correspondent aux probabilités d'appartenance de chaque objet à chacune des gaussiennes. Et dans la seconde phase (*Maximization*), les paramètres des gaussiennes sont réévalués en fonction de ces nouveaux paramètres cachés calculés. La fonction critère que l'on cherche à maximiser est la somme des log-probabilités de chaque objet $\overrightarrow{X_i}$:

$$E = \sum_i log(P(\overrightarrow{X_i}))$$

Cette fonction s'améliore à chaque étape de la méthode. L'algorithme converge donc, et un critère d'arrêt naturel est de stopper lorsque le modèle se stabilise, c'est-à-dire lorsque E ne subit pas de modification d'une itération à l'autre.

2.2 Notre modèle

Nous nous plaçons dans le cas où l'on cherche à fournir en sortie de notre algorithme de *subspace clustering* une description simple des clusters trouvés. Dans ce cadre, il est reconnu que la représentation sous forme de règles (hypercubes dans des sous-espaces de l'espace original) est tout à fait appropriée car elle est facilement interprétable. Pour intégrer cette contrainte dans l'algorithme EM classique, nous proposons d'ajouter l'hypothèse que les données sont générées selon des distributions indépendantes sur chaque dimension. Cette hypothèse a comme effet d'affaiblir le modèle classique, prenant également en compte les corrélations possibles entre dimensions, mais nous gagnons à l'intégrer directement dans l'algorithme au lieu d'utiliser le modèle classique pour le simplifier ensuite. En effet, ainsi, la modélisation est adaptée à la présentation sous forme de règles des clusters trouvés car chaque dimension est caractérisée indépendemment des autres. De plus, l'algorithme est alors plus rapide que l'algorithme classique, car le nouveau modèle nécessite moins de paramètres ($O(M)$ au lieu du $O(M^2)$ classique, pour M le nombre de dimensions), et les opérations matricielles sont évitées.

Dans notre modèle, nous supposons que les données ont été générées selon des distributions gaussiennes sur les dimensions continues, et selon des distributions multinomiales sur les dimensions discrètes. Le modèle est donc composé des paramètres suivants pour chaque cluster C_k : W_k son poids, μ_{kd} (moyenne) et σ_{kd} (variance) pour les dimensions d continues, et $Freqs_{kd}$ (fréquences de chacune des modalités) pour d discrète.

2.3 Initialisation du modèle

Notons N le nombre d'objets de la base, $Largeur_d$ la largeur de l'intervalle sur la dimension d continue (la différence entre le maximum et le minimum des valeurs des objets sur la dimension), $Modalites_d$ l'ensemble des modalités possibles sur la dimension d discrète, et $Frequences_d$ l'ensemble des fréquences de chacune de ces modalités sur l'ensemble de la base. Étant donné K le nombre de clusters recherchés, nous proposons d'initialiser notre modèle aléatoirement de la façon suivante :

- Tirer aléatoirement K objets de la base servant de centroïdes initiaux, et poser, pour $\overrightarrow{X_i}$ centroïde de C_k et $1 \leq d \leq M$:

$$\begin{cases} \mu_{kd} = X_{id} & \text{si } d \text{ continue} \\ Freqs_{kd}(X_{id}) = 1 & \text{si } d \text{ discrète} \end{cases}$$

- Associer chaque objet $\overrightarrow{X_i}$ au cluster C_k dont le centroïde est le plus proche, en calculant la somme des distances sur chaque dimension : distance au centroïde normalisée sur les dimensions continues, et pour les dimensions discrètes, 1 si la modalité de l'objet est différente de celle du centroïde, 0 sinon :

$$Distance(\overrightarrow{X_i}, C_k) = \sqrt{\sum_{d \text{ continue}} \left(\frac{X_{id} - \mu_{kd}}{Largeur_d}\right)^2 + |\{d \text{ discrète} | Freqs_{kd}(X_{id}) = 0\}|}$$

- Et initialiser les paramètres du modèle :

$$N_k = |\{\overrightarrow{X_i} \in C_k\}| \quad \text{et} \quad W_k = \frac{N_k}{N}$$

$$\mu_{kd} = \frac{\sum_{\overrightarrow{X_i} \in C_k} X_{id}}{N_k} \quad \text{et} \quad \sigma_{kd} = \sqrt{\frac{\sum_{\overrightarrow{X_i} \in C_k} (X_{id} - \mu_{kd})^2}{N_k}}$$

$$Freqs_{kd}(mod) = \frac{|\{\overrightarrow{X_i} \in C_k | X_{id} = mod\}|}{N_k} \; \forall \; mod \in Modalites_d$$

2.4 Mise à jour des paramètres du modèle

Nous utilisons ensuite l'algorithme EM classique sur notre modèle. Les paramètres cachés du modèle correspondent aux probabilités d'appartenance de chaque objet à chaque cluster. Dans notre cas, les dimensions étant supposées indépendantes, la probabilité d'appartenance d'un objet

à un cluster correspond au produit des probabilités sur chaque dimension. Et pour éviter qu'une probabilité nulle sur une dimension n'annule la probabilité globale, nous utilisons une constante positive très faible ϵ :

$$P(\overrightarrow{X_i}|C_k) = \prod_{d=1}^{M} max(P(X_{id}|C_{kd}), \epsilon)$$

$$P(X_{id}|C_{kd}) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma_{kd}} e^{-\frac{1}{2}\left(\frac{X_{id}-\mu_{kd}}{\sigma_{kd}}\right)^2} & \text{si } d \text{ continue} \\ Freqs_{kd}(X_{id}) & \text{si } d \text{ discrète} \end{cases}$$

$$P(\overrightarrow{X_i}) = \sum_{k=1}^{K} W_k \times P(\overrightarrow{X_i}|C_k) \quad \text{et} \quad P(C_k|\overrightarrow{X_i}) = W_k \times \frac{P(\overrightarrow{X_i}|C_k)}{P(\overrightarrow{X_i})}$$

Une fois les paramètres cachés calculés à partir des paramètres courants du modèle, celui-ci est à son tour mis à jour comme suit :

$$W_k = \frac{1}{N} \sum_i P(C_k|\overrightarrow{X_i})$$

$$\mu_{kd} = \frac{\sum_i X_{id} \times P(C_k|\overrightarrow{X_i})}{\sum_i P(C_k|\overrightarrow{X_i})} \quad \text{et} \quad \sigma_{kd} = \sqrt{\frac{\sum_i P(C_k|\overrightarrow{X_i}) \times (X_{id}-\mu_{kd})^2}{\sum_i P(C_k|\overrightarrow{X_i})}}$$

$$Freqs_{kd}(mod) = \frac{\sum_{\{i|X_{id}=mod\}} P(C_k|\overrightarrow{X_i})}{\sum_i P(C_k|\overrightarrow{X_i})} \; \forall \; mod \in Modalites_d$$

2.5 Algorithme

L'algorithme EM est connu pour converger lentement dans certains cas. Pour l'accélérer, nous proposons d'ajouter l'heuristique suivante : s'arrêter lorsque les attributions de clusters aux objets ne changent pas. Ce critère d'arrêt ressemble alors fortement au critère d'arrêt de K-means. À chaque itération, il faut donc également évaluer les attributions de clusters aux objets comme suit :

$$Cluster(\overrightarrow{X_i}) = ArgMax_{k=1}^{K} P(\overrightarrow{X_i}|C_k)$$

Finalement, l'algorithme est relancé un certain nombre de fois pour faire face aux difficultés d'initialisation inhérentes à toute méthode de type EM. Puis la partition maximisant la fonction E définie section 2.1 est conservée.

2.6 Présentation du résultat

Afin que le résultat soit le plus compréhensible possible, nous souhaitons nous donner une seconde vue sur chaque cluster, correspondant à sa représentation simplifiée sous forme de règle, décrite par le moins de dimensions possible. Dans un premier temps, chaque cluster est représenté par l'intervalle minimum contenant l'ensemble des valeurs des objets inclus dans le cluster sur les dimensions continues, et par la modalité la plus probable sur les dimensions discrètes. Ensuite, nous calculons le support de la règle, c'est-à-dire l'ensemble des objets compris dans la règle. Cette étape est nécessaire car il peut arriver que des objets n'appartenant pas au cluster appartiennent au support de la règle associée à ce cluster. Puis nous attribuons un poids W_{kd} à chacune des dimensions d du cluster C_k, en fonction de la dispersion relative des objets sur la dimension. Pour les dimensions continues, il s'agit du rapport entre variance locale et variance globale par rapport à μ_{kd}. Et pour les dimensions discrètes, il s'agit de la fréquence relative de la modalité la plus probable.

$$W_{kd} = \begin{cases} 1 - \frac{\sigma_{kd}^2}{\sigma_d^2} \text{ , avec } \sigma_d^2 = \frac{\sum_i (X_{id}-\mu_{kd})^2}{N} & \text{si } d \text{ continue} \\ \frac{Freqs_{kd}(mod) - Frequences_d(mod)}{1 - Frequences_d(mod)} & \text{si } d \text{ discrète} \\ \text{avec } mod = ArgMax_{\{m \in Modalites_d\}} Freqs_{kd}(m) & \end{cases}$$

Puis la sélection des dimensions pertinentes s'effectue comme suit : pour toutes les dimensions, présentées dans l'ordre croissant de leur poids, supprimer la dimension si sa suppression ne modifie pas le support de la règle. Il est également possible d'effectuer un pré-traitement à cette procédure, en supprimant directement les dimensions dont le poids est inférieur à un certain seuil. Ceci permet d'accélérer le traitement, et il semble raisonnable de supprimer les dimensions n'apportant que peu d'information sur l'appartenance des objets au cluster.

Pour illustrer le résultat de cette méthode, prenons l'exemple de la figure 1. Sur cet exemple, le cluster de droite est dense sur les deux dimensions X et Y. Son sous-espace de description est donc bien $X \times Y$. Par contre, il n'est pas utile de considérer Y pour le définir par rapport aux autres clusters. Le décrire par un intervalle élevé sur X est suffisant. Le même raisonnement s'applique pour le cluster du haut.

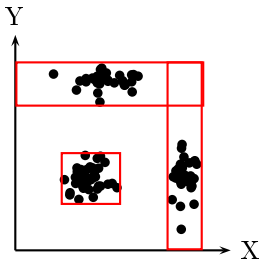


FIG. 1 – Exemple de description minimale.

3 Expérimentations

Des expériences ont été menées sur des bases de données artificielles ainsi que sur des bases de données réelles. Nous utilisons les premières pour tester la robustesse de notre algorithme face à différents types de bases. Un exemple d'application sur une base réelle permettra d'observer l'applicabilité réelle de notre méthode.

3.1 Données artificielles

Pour tester la robustesse de l'algorithme que nous proposons, nous produisons automatiquement des problèmes, un problème étant caractérisé par les paramètres suivants : N le nombre d'objets de la base, M_c et M_d le nombre de dimensions, respectivement continues et discrètes, décrivant les objets, K le nombre de clusters, C la dimensionnalité moyenne des sous-espaces dans lesquels les clusters sont définis, e_k l'écart type des coordonnées des objets appartenant à un même cluster C_k par rapport à son centre de gravité et sur ses dimensions caractéristiques continues, et p le pourcentage d'objets appartenant à un même cluster et partageant la même modalité sur ses dimensions caractéristiques discrètes.

K points d'ancrage $(\overrightarrow{O_1}, ..., \overrightarrow{O_K})$ sont tirés aléatoirement dans l'espace de description à $M = M_c + M_d$ dimensions, et sont utilisés comme centroïdes des clusters $(C_1, ..., C_K)$ à générer. L'intervalle de définition des dimensions continues est [0..100], et le nombre moyen de modalités sur les dimensions discrètes est de 5. À chacun de ces clusters est associée une partie des N objets, et un sous-ensemble des M dimensions constituant ses dimensions caractéristiques. Puis les coordonnées des objets appartenant à un cluster C_k sont générées selon une loi normale de centre O_{kd} et d'écart type e_k sur toute dimension continue d caractéristique de C_k, et avec une probabilité p d'être égal à la modalité O_{kd} sur toute dimension discrète d caractéristique de C_k ; elles sont générées selon une loi uniforme dans l'espace de description des dimensions non caractéristiques.

3.2 Visualisation graphique des résultats

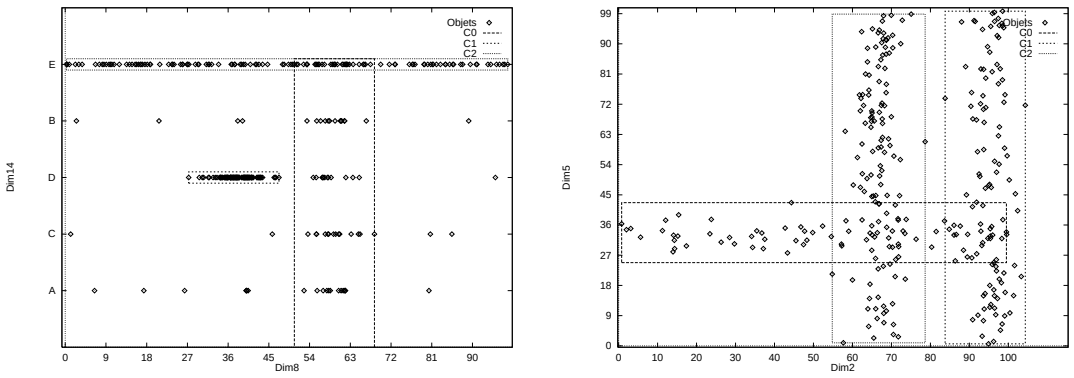
Afin de tester l'efficacité de notre méthode dans la tâche de génération d'une représentation simple des clusters identifiés, nous générons une base artificielle ayant les caractéristiques suivantes : $N = 300$, $M_c = 10$, $M_d = 5$, $K = 3$, $C = 4$, $e = 2 \pm 3$ et $p = 0.8$. Une base obtenue ainsi est décrite sous forme de règles comme suit :

- $C_0(N_0 = 104) \Rightarrow d_0 \in [64, 87], d_2 \in [84, 104], d_8 \in [27, 47], d_{13} = D$ et $d_{14} = D$
- $C_1(N_1 = 66) \Rightarrow d_5 \in [25, 43]$ et $d_8 \in [51, 68]$
- $C_2(N_2 = 130) \Rightarrow d_2 \in [55, 79], d_4 \in [75, 92], d_9 \in [69, 92]$ et $d_{14} = E$

Sur cette base, notre algorithme retrouve effectivement les clusters, et la représentation obtenue sous forme de règles est la suivante :

- $C_0 \Rightarrow d_2 \in [84, 104], d_{10} = A$ et $d_{14} = D$
- $C_1 \Rightarrow d_5 \in [25, 43]$ et $d_8 \in [51, 68]$
- $C_2 \Rightarrow d_2 \in [55, 79], d_4 \in [75, 92]$ et $d_{14} = E$

Notre méthode de description minimale des clusters nous permet de réduire efficacement le nombre de dimensions décrivant les clusters. Il nous est donc possible de calculer, avec un coût calculatoire réduit, un poids associé à chaque couple de dimensions présentes dans la description des clusters, poids qui reflète le pouvoir de visualisation graphique de ce couple de dimensions : $V_{ij} = \sum_{k=1}^{K} max(W_{ki}, W_{kj})$. Dans notre exemple, les visualisations graphiques correspondant aux deux couples de dimensions dont le poids est maximum sont fournies figure 2.



(a) Projections sur les dimensions d_8 et d_{14}.

(b) Projections sur les dimensions d_2 et d_5.

FIG. 2 – Visualisation graphique des clusters obtenus par SSC dans l'exemple.

3.3 Robustesse de l'algorithme

Afin de nous comparer aux méthodes existantes de *subspace clustering*, nous proposons de mener ces expériences face à des bases uniquement numériques. Notre méthode s'apparente à la famille des méthodes par partition. Parmi les plus récentes, LAC de (Domeniconi et al. 2004) est une méthode efficace qui, comme la nôtre, nécessite un seul paramètre utilisateur : le nombre de clusters recherchés. Nous proposons donc de nous comparer à cet algorithme. LAC est basé sur l'algorithme K-means et associe à chaque centroïde de cluster un vecteur de poids sur chaque dimension de l'espace. À chaque étape, ces vecteurs sont mis à jour en fonction de la dispersion des objets du cluster sur les dimensions : plus la dispersion est élevée, plus le poids associé à la dimension est faible.

Nous utilisons des bases artificielles pour évaluer les taux d'erreurs de notre algorithme et de LAC dans une tâche de classification. Pour toute base de données test, 70% des données sont utilisées pour l'apprentissage (le partitionnement de la base), et 30% forment l'ensemble T utilisé pour tester la classification de nouveaux exemples (leur association à un cluster créé). Pour chaque nouvel objet $\overrightarrow{X_i}$ à classer, la méthode adoptée est de l'associer au cluster le plus probable, comme défini section 2.5. Pour la méthode LAC, un nouvel objet est associé au cluster dont le centroïde est le plus proche, en utilisant une distance pondérée par les poids identifiés par l'algorithme sur chaque dimension. Nous utilisons la mesure d'entropie pour évaluer les taux d'erreurs en classification des méthodes. Soient, pour toute classe réelle i et tout cluster créé C_k, Nb_k le nombre d'éléments du cluster C_k dans T, et N_{ik} le nombre d'éléments de la classe i appartenant au cluster C_k,

$Precision(i,k) = N_{ik}/Nb_k$, $Ent_k = -\sum_i Precision(i,k) \times log(Precision(i,k))$, et $Entropie = \sum_k \frac{Nb_k \times Ent_k}{|T|}$. Plus l'entropie est faible, plus le taux d'erreurs de la méthode est important.

La figure 3 illustre les résultats de notre méthode est ceux de LAC face à une base générée artificiellement. Sur cet exemple, nous pouvons observer une limitation classique des méthodes de type K-means face aux méthodes de type EM : les premières recherchent des clusters de forme hypersphérique alors que les secondes sont capables de générer des clusters de forme allongée. En fait, de la même façon que EM généralise K-means, SSC généralise LAC (l'hypothèse spécifique de K-means est que les probabilités d'appartenance à chaque cluster sont égales, et que chaque distribution gaussienne a la même variance).

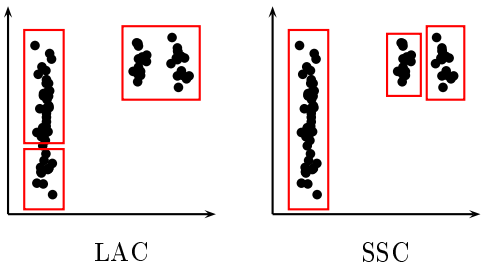


FIG. 3 – LAC versus SSC.

Les expériences menées en faisant varier les paramètres de génération des bases artificielles ont mis en avant la robustesse de notre méthode. En particulier, elle se révèle efficace pour faire face au bruit existant dans les données (cf. figure 4).

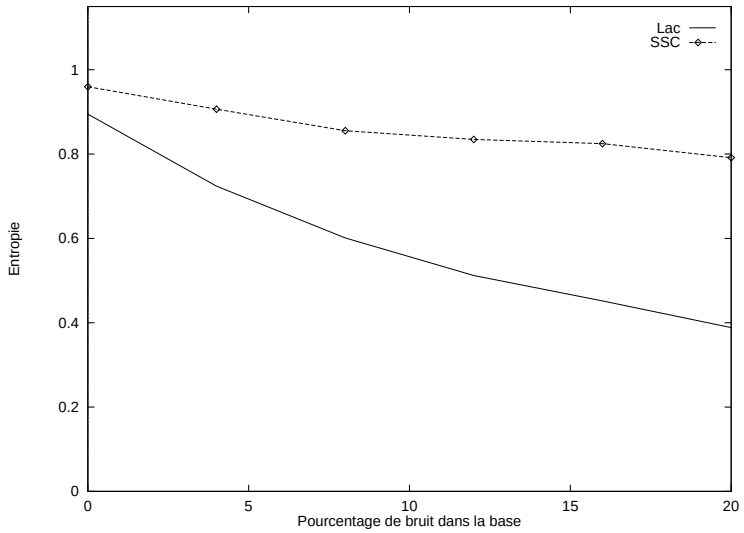


FIG. 4 – Résistance au bruit variant de 0 à 20% pour des bases artificelles avec $N = 600$, $M = 30$, $K = 5$, $C = 3$ et $e = 2 \pm 3$. Le nombre de clusters à trouver est fourni aux algorithmes.

Concernant le temps d'exécution de la méthode, l'heuristique que nous avons proposée permet d'obtenir, pour des résultats de qualité similaire, des temps de calcul plus proches de ceux de K-means (connu pour sa rapidité) que de ceux de EM (cf. figure 5).

Concernant le seul paramètre de l'algorithme, le nombre de clusters recherchés, lorsque celui fourni à l'algorithme est inférieur au nombre réel, alors quelques concepts sont fusionnés mais le résultat ne s'éloigne pas complètement de la solution réelle. Et lorsque le nombre de clusters fourni à l'algorithme est supérieur au nombre réel, alors plusieurs concepts se recouvrent (cf. figure 6).

Enfin, notons que les résultats de notre algorithme sont tout aussi robustes si les données ont

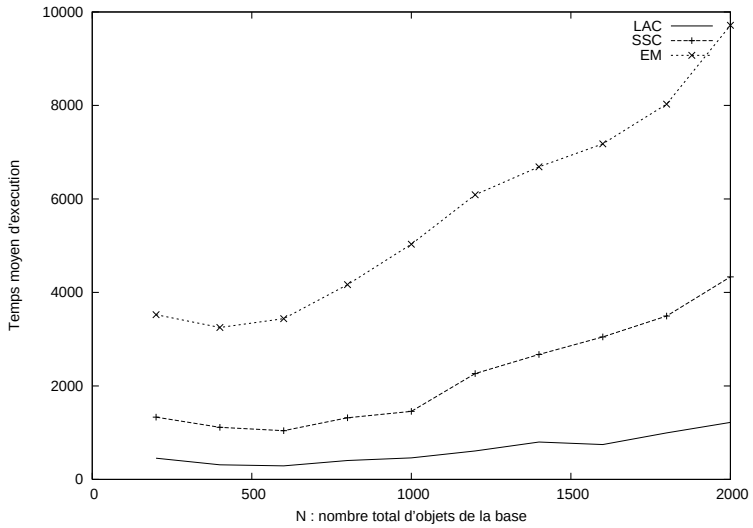


FIG. 5 – Temps d'exécution selon le nombre d'objets variant de 200 à 2000, sur des bases artificielles avec $M = 30$, $K = 5$, $C = 3$ et $e = 2 \pm 3$. Le nombre de clusters à trouver est fourni aux algorithmes.

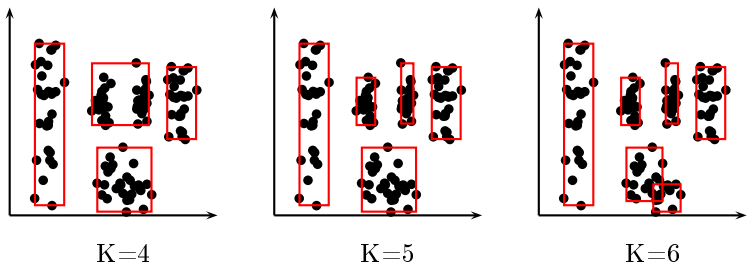


FIG. 6 – Influence du paramètre K de SSC.

été générées selon des lois uniformes dans les intervalles de définition de leurs dimensions caractéristiques, au lieu de lois gaussiennes.

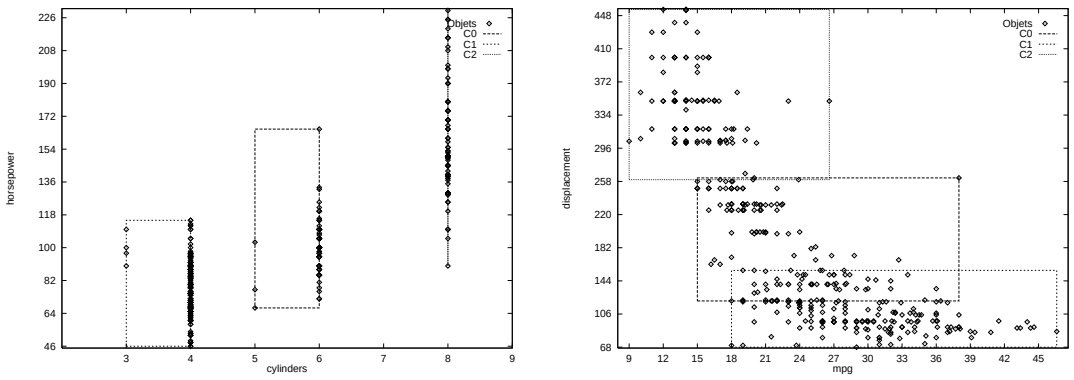
3.4 Tests sur données réelles

Des expériences ont également été menées sur des bases de données réelles. Parmi elles, la base *AutoMpg*, issue des bases de données de l'UCI (Blake et Merz 1998), contient la description d'un ensemble de moteurs de voitures (nombre de cylindres, de chevaux, accélération, etc.) Le tableau 1 présente les règles obtenues par notre algorithme, instancié avec le nombre de clusters recherchés égal à 3.

mpg	cylinders	displacement	horsepower	weight	acceleration	modelYear	origin
[18,47]	[3,4]	[68,156]	[46,115]	[1613,3270]	[12,25]	[70,82]	[1,3]
[15,38]	[5,6]	[121,262]	[67,165]	[2472,3907]	[11,21]	[70,82]	[1,3]
[9,27]	[8,8]	[260,455]	[90,230]	[3086,5140]	[8,22]	[70,81]	[1,1]

TAB. 1 – Règles obtenues par SSC sur *AutoMpg* avant simplification, pour K=3.

Après la phase de description minimale, seul l'attribut *cylinders* est conservé pour caractériser l'appartenance des objets aux clusters. Finalement, la figure 7 présente les visualisations graphiques associées aux clusters identifiés, mettant ainsi en avant l'applicabilité réelle de notre méthode.



(a) Projections sur *cylinders* et *horsepower*.

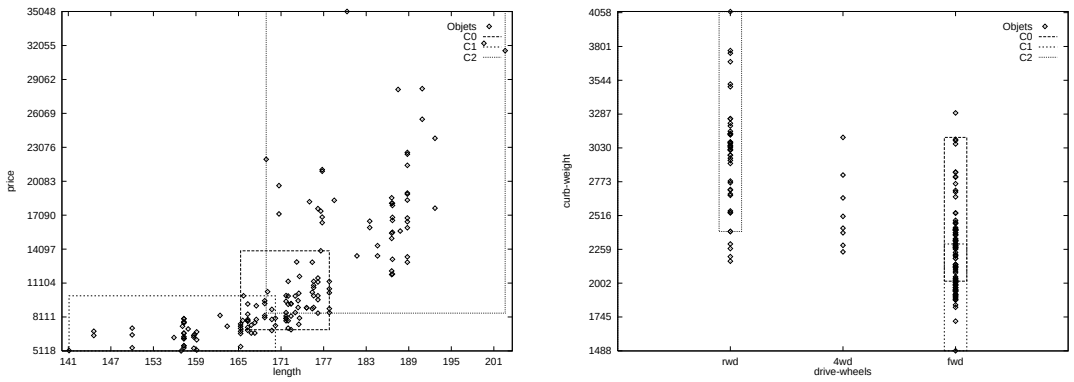
(b) Projections sur *mpg* et *displacement*.

FIG. 7 – Visualisation graphique des résultats de SSC sur la base *AutoMpg*, pour $K = 3$.

D'autres expériences ont été menées sur une autre base de l'UCI, contenant davantage d'attributs catégoriels : la base *Automobile*. Sur cette base, les visualisations graphiques correspondant à deux couples de dimensions de poids maximum sont fournies figure 8. L'algorithme met ainsi en avant que le prix des voitures augmente fortement lorsque leur longueur dépasse les 170 (figure 8(a)), que les voitures ayant une traction arrière (*rwd*) ont un poids à vide supérieur aux tractions avant et 4 roues motrices (figure 8(b)), et que la majorité des voitures les plus chères sont à traction arrière (correspondance entre les deux figures concernant le cluster C_2).

4 Conclusions et perspectives

Nous avons présenté dans cet article une nouvelle méthode de *subspace clustering* basée sur l'algorithme EM en ajoutant l'hypothèse que les données ont été générées selon des distributions indépendantes sur chaque dimension. Cette idée a déjà été étudiée dans (Pelleg et Moore 2000). Il existe plusieurs différences entre notre méthode et la leur. La première apparaît dans la modélisation : au lieu de supposer la distribution gaussienne sur une dimension continue, les auteurs



(a) Projections sur *longueur* et *prix*.

(b) Projections sur *traction* et *poids*.

FIG. 8 – Résultats de SSC sur la base *Automobile*, pour $K = 3$.

la supposent uniforme à l'intérieur d'un intervalle donné, et utilisent une *queue* de distribution aux bords de cet intervalle, dépendant d'un paramètre σ qui évolue au cours de l'algorithme. Cette différence se retrouve ensuite dans la méthode finale de clustering. En particulier, leur méthode n'est pas capable de mettre à jour son modèle de façon incrémentale, alors que la nôtre peut s'adapter à la présentation de nouveaux exemples. De plus, nous avons intégré effectivement la problématique catégorielle qui n'était évoquée qu'à titre de perspectives dans l'article et nous avons proposé une méthode originale de sélection d'attributs permettant de fournir en sortie un résultat compréhensible et visuel des clusters identifiés.

Nous avons également défini une heuristique originale pour accélérer l'algorithme. Pour poursuivre la recherche dans ce sens, il semble intéressant de s'inspirer de l'article de (Bradley et al. 1998) qui traite de l'accélération de l'algorithme EM dans le cas général. Une autre piste possible est d'éviter de considérer toutes les dimensions au cours de l'algorithme, en ne sélectionnant que les dimensions de poids maximum. De la même façon, nous pensons qu'il est possible de ne considérer qu'une partie des dimensions lors de la tâche de classification de nouveaux exemples.

Notons également que notre méthode ne nécessite qu'un seul paramètre de la part de l'utilisateur : K, le nombre de clusters recherchés. Une idée de poursuite de ces travaux est de définir automatiquement ce paramètre. Pour cela, il est classique d'utiliser le critère BIC (Ye et al. 2003). Dans notre cas, une autre piste originale serait d'utiliser le fait que lorsque K est supérieur au nombre réel de clusters recherchés, alors les règles associées aux clusters se chevauchent.

Enfin, il serait intéressant d'étudier l'efficacité de notre méthode dans une tâche de sélection d'attributs.

Références

Aggarwal C.C., Wolf J.L., Yu P.S., Procopiuc C. et Park J.S. Fast algorithms for projected clustering. In *ACM SIGMOD Int. Conf. on Management of Data*, pp 61-72, 1999.

Agrawal R., Gehrke J., Gunopulos D. et Raghavan P. Automatic subspace clustering of high dimensional data for data mining applications. In *ACM SIGMOD Int. Conf. on Management of Data*, pp 94-105, Seattle, Washington, 1998.

Berkhin P. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, California, 2002.

Blake C.L. et Merz C.J. (1998). UCI Repository of machine learning databases [http ://www.ics.uci.edu/∼mlearn/MLRepository.html].

Bradley P., Fayyad U., et Reina C. Scaling EM (Expectation-Maximization) Clustering to Large Databases. Microsoft Research Report, MSR-TR-98-35, Aug. 1998.

Cheng C.H., Fu A.W. et Zhang Y. Entropy-based subspace clustering for mining numerical data. In *Knowledge Discovery and Data Mining*, pp 84-93, 1999.

Domeniconi C., Papadopoulos D., Gunopolos D. et Ma S. Subspace clustering of high dimensional data. In *SIAM International Conference on Data Mining*, 2004.

Kailing K., Kriegel H.P. et Kröger P. Density-connected subspace clustering for high-dimensional data. In *SIAM Int. Conf. on Data Mining*, pp 246-257, Orlando, Florida, USA, 2004.

Nagesh H., Goil S. et Choudhary A. Mafia : Efficient and scalable subspace clustering for very large data sets. Technical report, Northwestern University, 1999.

Parsons L., Haque E. et Liu H. Evaluating subspace clustering algorithms. In *Workshop on Clustering High Dimensional Data and its Applications, SIAM Int. Conf. on Data Mining*, pp 48-56, 2004.

Pelleg D. et Moore A. Mixtures of rectangles : Interpretable soft clustering. In Carla Brodley and Andrea Danyluk, editors, *18th Int. Conf. on Machine Learning*, pp 401-408. Morgan Kaufmann, San Francisco, California, 2001.

Procopiuc C.M., Jones M., Agarwala P.K. et Murali T.M. A monte carlo algorithm for fast projective clustering. In *ACM SIGMOD Int. Conf. on Management of Data*, pp 418-427, 2002.

Sarafis I.A., Trinder P.W. et Zalzala A.M.S. Towards effective subspace clustering with an evolutionary algorithm. In *IEEE Congress on Evolutionary Computation*, Canberra, Australia, December 2003.

Wang H., Wang W., Yang J. et Yu P.S. Clustering by pattern similarity in large data sets. In *ACM SIGMOD Int. Conf. on Management of Data*, pp 394-405, 2002.

Woo K.G. et Lee J.H. *FINDIT : a fast and intelligent subspace clustering algorithm using dimension voting*. PhD thesis, Korea Advanced Institute of Science and Technology, Department of Electrical Engineering and Computer Science, 2002.

Ye L. et Spetsakis M.E. Clustering on Unobserved Data using Mixture of Gaussians. Technical Report, York University, Oct. 2003.

Yip K.Y., Cheung D.W. et Ng M.K. A highly-usable projected clustering algorithm for gene expression profiles. In *3rd ACM SIGKDD Workshop on Data Mining in Bioinformatics*, pp 41-48, 2003.