TER | UNIVERSITÉ PARIS DESCARTES

Machine Learning for Data Science

Interface pour l'*Ensemble clustering*

Lazhar Labiod et Mohamed Nadif

1 Contexte du sujet

La classification automatique ou *clustering* consiste à partionner un ensemble d'objets (instances) décrits par un ensemble de variables en groupes (classes) homogènes. Avec l'avènement du Big Data et de la science des données, le *clustering* est devenu une tâche encore plus importante. Cependant, les méthodes classiques présentent parfois des difficultés, devant la complexité des données, à détecter un partionnement profitable pour l'utilsateur. La méthode dite *Ensemble Clustering* (Strehl et Ghosh 2003, Jain 2005), également connue sous le nom de *clustering de consensus*, apparait comme une alternative efficace et robuste. Elle vise à fusionner plusieurs partitions en une *partition consensus*, où chaque partition est obtenue en appliquant une méthode de clustering sur le même jeu de données.

Des efforts considérables ont été déployés dans ce domaine. Une direction prometteuse consiste à dériver la similarité par paire des partitions de base puis à transformer le regroupement en un problème de partitionnement. Ces méthodes basées sur les graphes résument généralement les partitions de base en une matrice de co-association, qui calcule réellement la cooccurrence des instances appartenant à la même classe.

L' *ensemble clustering* s'est révélé être une bonne alternative face aux problèmes d'analyse des clusters (classes). Il consiste à générer un ensemble de regroupements à partir du même ensemble de données et à les combiner en un regroupement final. Le but de ce processus de combinaison est d'améliorer la qualité des regroupements de données individuels. Une grande variété d'algorithmes de clustering a été proposée: k-Means, EM (Expectation Maximization), classification basée sur la théorie des graphes spectraux, des algorithmes de clustering hiérarchiques comme Single-Link, Fuzzy c-Means, etc. Cependant, comme on le sait, il n'existe pas de méthode de regroupement capable de trouver correctement la structure sous-jacente pour tous les ensembles de données. Lorsque nous appliquons un algorithme de clustering à un ensemble d'instances, il impose une organisation aux données selon un critère interne, les caractéristiques de la fonction de (dis) similarité utilisée et l'ensemble de données. Par conséquent, si nous avons deux algorithmes de clustering différents et que nous les appliquons au même jeu de données, nous pouvons obtenir des résultats très différents. Mais, lequel est le bon? Comment pouvons-nous évaluer les résultats?.

L'idée de combiner différents résultats de regroupement (ensemble de clusters ou agrégation de clusters) est apparue comme une approche alternative pour améliorer la qualité des résultats des algorithmes de clustering. Il est basé sur le succès de la combinaison des classifieurs supervisés. Étant donné un ensemble d'objets, une méthode d'ensemble clustering se compose de deux étapes principales: (1) Génération, celle-ci consiste en la création d'un ensemble de partitions de instances, et (2) une partition consensus est déterminée à partir de toutes les partitions obtenues dans l'étape de génération.

2 Description de l'algorithme

L'algorithme à implémenter permet la combinaison de plusieurs partitions de base en une *partition consensus* finale, il s'agit d'un algorithme itératif simple qui est basé sur quatre formules de mise à jour :

- Mise à jour de la matrice W_l des poids associés aux classes de la partition de base P_l
- Mise à jour de la matrice de la partition consensus G
- Mise à jour de la matrice de visualisation B
- Mise à jour de la matrice des centres des classes, S.

Voici les détails sur l'algorithme à implementer ($\mathbf{K}_i$ est la matrice de partition associée à la partition de base P_i).

$$\min_{\mathbf{W},\mathbf{B},\mathbf{S},\mathbf{G}} \sum_{i=1}^{m} ||\mathbf{K}_i\mathbf{W}_i - \mathbf{B}||_F^2 + \lambda||\mathbf{B} - \mathbf{G}\mathbf{S}||_F^2 + \delta \sum_{i=1}^{m} ||\mathbf{W}||_{2,1} \tag{1}$$

$$\mathbf{S} = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{B} \tag{2}$$

$$\mathbf{W}_i = (\mathbf{K}_i^T\mathbf{K}_i)^{-1}\mathbf{K}_i^T\mathbf{B} \tag{3}$$

$$\mathbf{B} = eig[\mathbf{M}], \text{ where, } \mathbf{M} = \lambda\mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T + \sum_i \mathbf{K}_i(\mathbf{K}_i^T\mathbf{K}_i)^{-1}\mathbf{K}_i^T \tag{4}$$

$$\mathbf{G} = kmeans(\mathbf{B}, \mathbf{S}) \tag{5}$$

3 Travail à effectuer

Il existe aujourd'hui des implémentations de plusieurs algorithmes pour les méthodes d'ensemble clustering. Un grand nombre de ces logiciels sont développés en R, en Python, en C++ ou en Java, et sont exécutés en local, sur des machines où les interpréteurs et les librairies correspondant à ces différents langages sont installés. L'objectif du PPD est de développer une application Web client-serveur en utilisant le package Shiny développé par Rstudio et permettant d'exécuter une tâche de comparaison et d'analyse des différentes methodes d'ensemble clustering choisies. L'outil à utiliser pour construire l'interface sera le package Shiny, les packages et outils d'ensemble clustering à interfacer seront les librairies python 'Cluster_Ensembles', 'sickitlearn', 'rpy' qui ont été développées en python. Le projet comporte quatre étapes:

1. Prise en main de Shiny, les librairies Cluster_Esemble, rpython et rticule
2. Implemention de l'algorithme d'*ensemble clustrering* décrite plus haut.
3. Mettre en place un protocole experimental pour la comparaison des différentes méthodes d'ensemble.
4. Affichage interactive des résultats.

4 Intérêt pédagogique

Le projet doit permettre aux étudiants de mettre en œuvre et d'étendre leurs connaissances sur les points suivants :

- Découvrir l'outil Shiny de Rstudio pour le développement d'interface web interactive.
- Découvrir le langage python et les connexions R - Python.
- Découvrir l'intérêt des méthodes d'ensemble en apprentissage machine
- Définir une interface riche avec Javascript, Html, R et python.

5 Références

- Calling Python from R with rPython: `https://www.r-bloggers.com/calling-python-from-r-with-rpython/`
- Cluster_Ensembles: `https://pypi.python.org/pypi/Cluster_Ensembles/1.16`
- `https://github.com/rstudio/reticulate`
- `https://rstudio.github.io/reticulate/articles/introduction.html`
- `Shiny:http://shiny.rstudio.com/tutorial/`
- `http://scikit-learn.org/stable/`
- Alexander Strehl, Joydeep Ghosh: Cluster Ensembles — A Knowledge Reuse Framework for Combining Multiple Partitions. Journal of Machine Learning Research 3: 583-617(2002)