TP Fouille de données
Master 2 Datascale

K. ZEITOUNI

Objectif :

Ces exercices ont pour but la pratique d'un outil de data mining, en l'occurrence Weka. Il suppose que vous avez déjà installé le logiciel (version 3.8). cf. https://www.cs.waikato.ac.nz/ml/weka/index.html

Il vous est recommandé de suivre les WekaMOOC sur Youtube. Le premier niveau est une bonne introduction, mais limitée à l'apprentissage supervisé. Le second est plus complet (clustering, règles d'association, fouille de données complexes, etc.). Au-delà de l'utilisation de l'outil, vous devez vous documenter sur les différentes méthodes et tenter d'expliquer les résultats et les différences observées lors des tests.

Test 1 (IRIS) :

Données en entrée :

Un des jeux de test les plus connus est celui décrivant les **IRIS**.
Il est décrit par quatre attributs observables et la classe de la fleur. Le but est d'induire la classe d'Iris connaissant les attributs numériques observables.

a1. sepal length (cm)
a2. sepal width in cm
a3. petal length in cm
a4. petal width in cm
L'attribut label désigne la classe possède trois modalités : « Iris-Setosa », « Iris-Versicolour » et « Iris-Virginica ».

On va suivre la méthodologie standard CRISP-DM comme le préconise le cours. Ce processus commence par la compréhension du problème « métier », la compréhension des données, leur préparation, la modélisation proprement dite, l'évaluation du modèle et son déploiement.
Ici, le problème est clairement défini : apprentissage et prédiction de la classe de cette fleur à partir des quatre attributs décrits ci-dessus.

Dans un premier temps, il faudra préparer les données. Pour cela, dans la fenêtre « Explorer » de Weka, ouvrir le fichier iris.arff (format ascii spécifique utilisé par Weka) et repérez les différents éléments de l'interface (taille, nombre d'attributs, classes cibles, distribution des classes par attribut, etc.). Utilisez le bouton « Edit » pour afficher le contenu au format tabulaire (sans l'éditer). Ensuite, explorez les données en utilisant l'onglet « Visualize ». Zoomez sur quelques plots de la *ScatterMatrice*. Vous pouvez aussi visualiser les 4 dimensions sur le même graphique avec Projection plot. Peut-on déduire des connaissances par simple visualisation ?

Clustering

Appliquez une méthode de clustering (*segmentation*) K-means d'abord avec les options par défaut puis en indiquant le nombre de classe k=3. Essayez avec et sans la variable classe (utilisez « Class to cluster evaluation » plutôt que « Ignore attributes » pour ignorer la classe tout en l'utilisant pour vérifier s'il retrouve les bonnes classes[1]. Visualisez graphiquement les clusters (menu contextuel dans la zone « Results lists »).

Dans un deuxième temps, utilisez la méthode de clustering EM avec les options par défaut. Comparez la qualité des clusters par comparaison aux classes réelles. Puis changez le nombre de clusters et comparez à nouveau les résultats de EM avec K-means. Comparez avec X-means sensé trouver le nombre de classes automatiquement. Qu'en pensez-vous ?

Classification supervisée et arbre de décision

Construisez un modèle de classification supervisé en utilisant les options par défaut. Quelle méthode a été choisie ? Comment interpréter / expliquer son résultat ?
Testez cette fois-ci en J48 avec les options par défaut. Comparez les résultats.
Utilisez la cross-validation. Quelle précision obtenez-vous ?
Testez également avec RandomForest puis avec kNN (IBk Pour « Instance Based with k classes ») qui est de type « Lazy ».

A présent, ouvrez le fichier « iris.2D.arff » qui se limite aux deux dimensions les plus caractéristiques de la classe d'iris (vous pouvez le vérifier avec l'onglet Projection plot). Ensuite, utilisez le menu « Visualization -> Boundary Visualizer » dans le 1er menu de Weka (GUI Choser). Cela permet de mieux comprendre la manière dont ces classifieurs séparent les classes et affectent (parfois) des probabilités aux classes prédites. Testez la visualisation de plusieurs classifieurs : J48, IBk, Naive Bayes, Random Forest, LogitBoost, …

[1]Ici la classe sert comme vérité terrain pour évaluer le clustering. Le plus souvent, il n'y a pas de classe connue et le clustering sert à en générer une.

Test 2 (ABALONE) :

Commencez par ouvrir le fichier csv (abalone.csv fourni sur e-campus). Celui-ci décrit des spécimens d'ormeaux (petit coquillage). Il a été fourni par la division des ressources maritimes de Tasmanie. Il contient 4177 lignes (exemples annotés). Chaque exemple est décrit par neuf attributs :

1. Sexe : M,F,I (comme mâle, femelle, enfant)
2. Longueur : réel (le grand axe de l'animal en mm)
3. Diamètre : réel (le petit axe en mm)
4. Hauteur : réel (en mm)
5. Poids total : réel (en gr)
6. Poids de la chair : réel (gr)
7. Poids des organes : réel (gr)
8. Poids de la coquille : réel (gr)
9. Nombre d'anneaux : entier (+1.5 = Age de l'animal)

On souhaite analyser les données pour essayer de découvrir s'il est possible de calculer l'âge d'un ormeau à partir des autres caractéristiques, beaucoup plus faciles à observer.

Suivez le même processus que les tests précédents pour explorer puis construire un modèle de classification. Notez qu'ici, la variable à prédire est le nombre d'anneaux de type continue. Il faudra donc identifier et tester les modèles de classification adaptés à des variables continues (notez d'ailleurs que certains sont grisés). Notez la différence dans le coefficient de corrélation. Testez la régression linéaire, kNN (IBk) avec les options par défaut puis avec k=10. Testez enfin le « MultilayerPerceptron » (MLP)[2]. Comparez les qualités respectives mais aussi les temps d'exécution.

Une autre façon de procéder est de discrétiser la classe : jeune (< 7 anneaux), adulte (entre 7 et 13 anneaux) et vieux (> 13 anneaux). Essayez de le faire dans Weka ou sous un tableur normal. Refaites l'exploration puis la classification supervisée. Quelle conclusion pouvez-vous en tirer ?

[2] Cette méthode est un type réseau neuronal adapté aux problèmes non linéaires. Elle est basée sur la technique de rétro-propagation du gradient.

Test 3 (Market Basket) :

Ouvrez le fichier « supermarket » situé dans le répertoire data de Weka. Supprimez les attributs superflus (departmentxx) puis sauvegardez le résultat (pour ne pas écraser l'ancien, gardez-en une copie). Pour les autres attributs, on remarque qu'il y a beaucoup de valeurs nulles. Est-ce normal ? Faut-il les remplacer (justifiez) ?

Utilisez l'onglet « Associate » offrant différents algorithmes dont ceux vus en cours. Explorez les options de l'algorithme Apriori et essayez de les comprendre en lisant les explications données par l'outil. Activez l'affichage des itemset fréquents et utilisez cet algorithme. Vous verrez qu'il illustre bien le déroulement des différentes phases d'Apriori vues en cours. Interprétez les règles et leurs métriques.

Faites varier les paramètres, par exemple remplacer la confiance par le *lift*, la conviction, le *leverage*, augmentez le nombre de règles, variez les seuils, etc. Comparez les résultats et essayez d'expliquer les différences. Que pouvez-vous dire sur ces méthodes et leur usage en pratique ?

Test 4 (Churn) :

Idem pour le fichier « churn.csv » pour l'analyse des désabonnements (signifiant le départ chez les concurrents) des clients d'un opérateur téléphonique. Explorez les distributions des attributs et leur variation par rapport à la classe.

Comme vous pouvez le remarquer, il y a beaucoup d'attributs et d'instances et il y a beaucoup moins d'instances de churn (heureusement !). Pouvez-vous visuellement détecter les attributs pouvant contribuer à discriminer les churns des clients fidèles ?

Appliquez une classification par J48 et sauvegardez le modèle.

En réalité, ce qui intéresse l'analyse, c'est la qualité de prédiction des churns en priorité. Interprétez finement les résultats pour différentier entre la qualité globale de classement et la qualité de classement pour cette classe en particulier. Ensuite, essayez de revenir aux données pour rectifier le non balancement des classes : avec une méthode de prétraitement fournie « ClassBalancer ». Puis, réappliquez J48[3].

Une autre piste pour traiter ce jeu de données est de réduire sa dimensionnalité en utilisant la méthode « Select attributes ». Celle-ci mesure l'importance des attributs vis-a-vis de la classe afin de les filtrer par la suite. Cela permet de construire un modèle plus rapidement sans en réduire la qualité. Essayez différentes méthodes qui vous parlent : test de Chi2 (pensez à changer le seuil). Comparez les dimensions retenues ou générées. Qu'observez-vous ? Choisissez-en deux (projection sur 12 attributs pour l'un, puis sur 4 attributs pour l'autre) et relancer la classification avec J48 normal puis avec « FilteredClassifier » (en combinant le « ClassBalancer » et J48).

[3] Notez qu'il existe un méta-classifier spécifique « FilteredClassifier » pour les combiner n'importe quel Filter avec n'importe quel classifieur à la volée.