TP 2 - Panorama des méthodes de fouille de données

Introduction

L'objectif de cette séance est de passer en revue les différents types de méthodes utilisées en fouille de données, que ce soit en analyse exploratoire ou en analyse prédictive.

1 Analyse exploratoire

1.1 Classification / Clustering

Dans cette première partie, vous avez à étudier les données « eurostat » téléchargées depuis le site Web d'Eurostat [1], organisme attaché à la Commission européenne. Ces données fournissent des indicateurs économiques mesurés entre 2011 et 2013 pour les 28 pays de l'Union européenne, ainsi que la Suisse, le Japon et les États-Unis. Vous trouverez un descriptif des variables étudiées dans le fichier `description.txt` de l'archive `eurostat.zip` à télécharger.

1. Ouvrez le fichier `eurostat-2013.csv` à l'aide de la bibliothèque pandas.
   Certains attributs sont influencés par la taille démographique du pays ; quels sont ces attributs ? Divisez les valeurs de ces attributs par la population du pays. Supprimez ensuite les données correspondant à la population.
2. Il est souvent utile d'appliquer un filtre de normalisation sur tous les attributs avant d'utiliser des méthodes de *clustering*. Quel est l'effet du filtre StandardScaler sur les données ? A-t-on besoin ici d'employer ce filtre sur les données étudiées ?
3. Effectuez une analyse en composantes principales (ACP) sur les données à l'aide de la bibliothèque scikit-learn.
   Joignez au rapport l'affichage des instances étiquetées par le code du pays suivant les 2 facteurs principaux de l'ACP, puis suivant les facteurs 3 et 4 de l'ACP.
4. Que représentent les 4 premiers facteurs de l'ACP ? Vous pourrez répondre à cette question en calculant la variation des valeurs propres en fonction du rang et en traçant le cercle de corrélation entre les facteurs et les variables originales.
   Commentez les deux graphiques obtenus à la question précédente (affichage dans les deux premiers plans de l'ACP) en discutant la proximité entre les pays. Combien de groupes de pays définiriez-vous sur ces données ?
5. Reprenez les données telles qu'elles étaientt avant leur transformation par l'ACP. En utilisant KMeans de la bibiothèqe scikit-learn, réalisez un *clustering* avec la méthode des k-moyennes, où k représente le nombre souhaité de *clusters*. Tracez la variation du R^2 pour k variant de 2 à 8. D'après cette courbe, quelle valeur de k reteneriez-vous ?
6. En fixant k à la valeur déterminée à la question précédente, commentez les profils des groupes contenant la France et le Japon.

1. `http://epp.eurostat.ec.europa.eu/portal/page/portal/eurostat/home/`

7. Réalisez un *clustering* au moyen d'une méthode hiérarchique ascendante avec la classe dendogram de la bibliothèque scipy.

   Joignez au rapport l'affichage du dendogramme obtenu.

8. Commentez l'arbre obtenu précédemment en discutant des similitudes entre pays. À quel niveau de l'arbre feriez-vous une coupure ? Combien de groupes de pays seraient alors créés ?

1.2 Recherche de règles d'association

Cette partie aborde la recherche de règles d'association, en illustrant son principe par l'algorithme *Apriori*, disponible dans la bibliothèque mlxtend.

Apriori opère sur des données dont tous les attributs sont nominaux, les attributs numériques devant être discrétisés auparavant. L'algorithme génère des règles, sous la forme *prémisses* (appelées *antecedents* dans la bibliothèque utilisée) $\Rightarrow$ *conclusion* (appelée *consequents*). Ces règles sont ordonnées par défaut selon leur support, c'est-à-dire le nombre d'instances qui satisfont la règle.

La découverte de règles d'association est susceptible de produire un nombre très important de règles, dont la plupart sont peu pertinentes pour le problème. Afin de limiter le nombre de résultats obtenus, l'algorithme est basé sur un niveau de couverture minimal, c.-à-d. la proportion des instances couvertes par chaque règle. Pour des raisons d'efficacité algorithmique, la méthode *Apriori* commence par chercher des règles à un item (restreint à la classe à prédire), puis étend celles qui ont une couverture suffisante avec une valeur d'attribut pour obtenir des règles à deux items. Elle effectue ensuite de nouvelles itérations en ajoutant à chaque fois un nouvel item.

Dans cet exercice, il est demandé d'étudier le fichier `supermarket.arff`, qui fournit des informations sur les achats dans un supermarché de Nouvelle-Zélande. L'objectif est ici de découvrir des règles s'appliquant sur le panier de consommation.

1. Ouvrez les données à l'aide de la fonction loadarff() disponible dans la bibliothèque scipy. Quels sont les types des attributs ? Que représentent les valeurs associées aux ' ?' ? Comment agit les commandes du fichier squelette, permettant la discrétisation ?

2. Le niveau de couverture est très dépendant des données. Établissez l'ensemble des itemsets fréquents à l'aide de la fonction apriori() de mlxtend en fixant un support minimal à 0,1. Expliquez ce que signifie cette valeur de 0,1. Combien d'itemsets obtenez-vous ? Donnez leur fréquence en fonction du nombre d'items qu'ils contiennent.

3. Construisez les règles à partir de l'ensemble des itemsets précédemment obtenus en recourant à la fonction association_rules() de mlxtend. Pour ne pas avoir un nombre trop important de règles générées, fixez un seuil minimal de 0,7 pour la valeur de confiance.

   Combien de règles obtenez-vous ? Commentez la première règle obtenue.

4. Parmi les règles obtenues, combien impliquent 5 items (4 antécédents et 1 conséquence) ? [2] Que constatez-vous en comparant entre elles ces règles ?

5. *Apriori* possède plusieurs critères pour évaluer la pertinence d'une règle : *confidence*, *lift*, *leverage* et *conviction* [3]. Définissez chaque métrique pour une règle sous la forme $A \Rightarrow B$ à partir des probabilités de A, B, $!B$, $A \cap B$ et $A \cup B$.

   Quelle est la meilleure règle pour l'ensemble « supermarket » pour chacune des métriques *confidence*, *lift*, *leverage* et *conviction* ?

2. Pour sélectionner les règles respectant cette condition, vous pourrez utiliser la fonction loc() s'appliquant sur un dataframe avec la fonction map().

3. Vous pouvez vous référer au cours de l'UCE ou bien aux informations disponibles sur la page `https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/` .

6. Un dirigeant du supermarché souhaite améliorer ses ventes en réorganisant le positionnement des produits dans ses rayons. D'après votre analyse du panier de consommation fourni, quels produits sont souvent achetés conjointement avec du café et du lait-crème ? [4]

2 Analyse descriptive

2.1 Classement

Dans cette partie du TP, vous allez étudier et comparer le comportement de différents classifieurs sur les données « Labor ». Ces données concernent les résultats finaux de négociations du travail menées dans l'industrie au Canada en 1987 et 1988. Elles incluent tous les accords collectifs dans le secteur des affaires et des services à la personne pour des organisations locales d'au moins 500 membres (enseignants, infirmiers, employés d'université, police, etc.).

Les méthodes de classement que nous testerons sur ces données seront :

— le classifieur élémentaire associant chaque instance à la classe majoritaire (DummyClassifier avec la stratégie most_frequent),
— la méthode bayésienne naïve (GaussianNB),
— l'arbre de décision (DecisionTreeClassifier),
— la méthode de régression logistique (LogisticRegression),
— le modèle SVM (SVC).

1. Ouvrez les données à l'aide de la fonction loadarff() disponible dans la bibliothèque scipy. Combien y a-t-il d'attributs dans le ficher ? Quels sont leurs types ? Quelle est la variable à prédire ? Les classes sont-elles équilibrées dans le fichier ? Y a-t-il beaucoup de valeurs manquantes dans les données ? Comment ces valeurs sont-elles représentées dans le ficher ?

   Dans un premier temps, seules les données numériques sont considérées pour prédire la classe. Normalisez les valeurs numériques à l'aide de StandardScaler. Remplacez chaque valeur manquante par la moyenne obtenue par chaque attribut en utilisant SimpleImputer.

2. Expliquez en une phrase en quoi consiste la méthode élémentaire de classement DummyClassifier. Dans quel cas ce classifieur pourrait donner les meilleurs résultats parmi ceux testés ?

3. Calculez les performances des cinq méthodes de classement étudiées sur les données « Labor » en réalisant une validation croisée en 5 strates. Parmi les méthodes testées, quelle est la plus performante en terme de taux de classification ? Quel est l'intérêt d'utiliser une validation croisée plutôt qu'un découpage des données en $\frac{2}{3}$ entraînement et $\frac{1}{3}$ test pour l'évaluation ? Quelle est la classe pour laquelle la prédiction fait le plus d'erreurs ?

4. Dans un deuxième temps, on se propose de n'employer que les attributs catégoriels. Remplacez chaque valeur manquante par le mode (valeur la plus fréquente) de chaque attribut en recourant à SimpleImputer [5]. Réalisez ensuite une discrétisation des données à l'aide de la fonction get_dummies() de pandas.

   Calculez les performances de classement des cinq méthodes étudiants, en réalisant à nouveau une validation croisée en 5 strates. Observez-vous des performances similaires à ce que vous aviez à la question précédente ?

5. Utilisez enfin l'ensemble des données, en les normalisant et en remplaçant les valeurs manquantes par les moyennes ou modes des attributs. Calculez à nouveau les performances de classement en validation croisée à 5 strates et commentez les résultats obtenus.

4. Vous aurez peut-être besoin de revoir la valeur auparavant fixée de support minimal.
5. Suivant la version installée de scikit-learn il se peut que vous ayez à utiliser l'ancienne classe preprocessing.Imputer.

6. L'application de méthodes de normalisation et de transformation des données sur l'ensemble du fichier avant de faire la validation croisée introduit un biais dans les résultats. Quel est ce biais ? Discutez de l'utilisation de Pipeline disponible dans scikit-learn.

2.2 Régression linéaire

On s'intéresse ici au modèle de régression le plus simple : le modèle linéaire. La régression linéaire permet de modéliser la relation entre une valeur aléatoire y et un ensemble de variables aléatoires x_i, sous la forme $y = \sum_i a_i x_i + u$ où u désigne le terme d'erreur.

Les données utilisées ici, disponibles dans `real-estate-valuation.zip`, affichent le prix de vente de logements d'une ville de Taïwan selon plusieurs paramètres (vous pouvez consulter le fichier de descriptions pour avoir des informations sur ces données).

1. Chargez les données et appliquez le modèle linear_model.Ridge de la bibliothèque scikit-learn. Quel est le modèle de régression produit en sortie ? Quelle variable est la plus importante ? Commentez le coefficient attribué à l'âge de construction du logement (X_2 dans le fichier).
2. Prédisez la valeur de vente pour un logement possédant les caractéristiques suivantes : $(2013.5, 6.5, 90.45606, 9, 24.97433, 121.5431)$. En sachant que la valeur réelle est 63.9, commentez la valeur prédite.