Travaux pratiques n°3 — #Découverte des règles d'association

Matière : Fouille de données
Enseignants : M. Taoufik Ben Abdallah
Discipline : 2ème année Génie Informatique
Mme. Rahma Boujelben
Année Universitaire : 2023-2024 / S1

L'objectif de ce TP est d'extraire des règles d'association à partir du jeu de données décrit dans le fichier ***market.csv***. Ce jeu de données contient **13736** observations, chacune comportant deux informations: le numéro de ticket d'une transaction effectuée dans un magasin (**ticketNumber**) et la désignation de l'un des **10** produits associés à ce ticket (**itemDescription**) **["whole milk", "yogurt", "sausage", "bottled water", "other vegetables", "rolls/buns", "root vegetables", "tropical fruit", "soda", "citrus fruit"]**.

Le **Tableau 1** montre un extrait du jeu de données ***market.csv***

ticketNumber	itemDescription
1808	tropical fruit
2552	whole milk
1187	other vegetables
3037	whole milk
4941	rolls/buns

Tableau 1 : Extrait de jeu de données market.csv

Étant donnée la fonction **gener_regles(df_rules, cond="antecedents", res="consequents")** qui prend en paramètre un DataFrame **df_rules**, ainsi que les paramètres **cond="antecedents"** et **res="consequents"**, elle génère une règle d'association sous la forme item→item, items→item ou items→item. Les paramètres **cond="antecedents"** (respectivement **res="consequents"**) représentent le nom de la colonne contenant l'antécédent (respectivement le conséquent) de la règle à extraire à partir de **df_rules**

```
def gener_regles(df_rules, cond="antecedents", res="consequents"):
  ant=map(lambda x : list(map(lambda y: y, x)), df_rules[cond])
  conseq=map(lambda x : list(map(lambda y: y, x)), df_rules[res])
  return "\n".join(map(lambda x,y: ", ".join(x)+"-->"+", ".join(y), ant, conseq))
```

NB. Si vous utilisez Colaboratory, importez le fichier *titanic.csv* depuis Google Drive. Dans ce cas, veuillez connecter votre compte Google Drive à Colaboratory

1/ Charger le jeu de données ***market.csv*** dans une variable nommée **df_market**. Afficher les 5 premières lignes de **df_market**

2/ Transformer la colonne **itemDescription** du DataFrame **df_market** en **10** **colonnes binaires** distinctes **portant les mêmes noms que les articles correspondants**

3/ Créer un nouveau DataFrame **trans** à partir de **df_market**, où les lignes sont regroupées par numéro de ticket (**ticketNumber**), et **les valeurs dans les colonnes binaires sont additionnées pour chaque groupe**

4/ Supprimer la colonne **ticketNumber** à partir de **trans** et modifier les autres colonnes de manière à attribuer la valeur **1** si un article apparaît plus d'une fois dans le ticket, et **0** pour les autres occurrences

5/ Afficher le type de chaque attribut dans **`trans`** et les modifier en un type booléen (**`bool`**)

6/ Déterminer les **`k-items fréquents`** (**`k≥1`**) en appliquant l'algorithme **Apriori**, avec une valeur minimale de support égale à **`0.05`.** Enregistrer les résultats dans un DataFrame nommé **`f_items`**

7/ Afficher l'ensemble des **`k-items fréquents`** (**`k≥2`**) contenant l'item **`'whole milk'`** à partir de **`f_items`**. **Utiliser l'une des fonctions de comparaison vectorisées des DataFrames (`eq(==)`, `ge(≥)`, ou `gt(>)`)**

8/ Déterminer **toutes les règles d'association pertinentes** à partir de **`f_items`**, sachant que la confiance minimale est fixée à **`0.2`**. Enregistrer le résultat dans un DataFrame nommé **`rules_p`** et afficher-le

9/ Identifier et afficher les **`5`** règles d'association **les plus intéressantes** en termes de **`lift`** à partir de **`rules_p`**. Une règle doit avoir la forme `item→item`, `items→item` ou `items→item`. **Utiliser la fonction donnée `generer_regles()`**

10/ Identifier et afficher toutes les règles d'association qui **comportent dans l'antécédent l'article le plus fréquent**, à partir de **`rules_p`**, sous la forme `item→item`, `items→item` ou `items→item`

11/ Identifier toutes les règles d'association **présentant une forte association positive entre l'antécédent et le conséquent**, où **le conséquent comprend au moins deux articles**, en termes de **`levrage`**, qui doit être **strictement positif**, et une **`conviction`** supérieure ou égale à **`1.05`**, à partir de **`rules_p`**. Enregistrer le résultat dans un DataFrame nommé **`rules_f`**

12/ Étudier le **choix du support minimal** de manière à obtenir **une ou plusieurs règles d'association intéressante(s)** ayant un **`lift`** supérieur à **`1`**

Bon Travail ♣