Package 'diffman'

February 28, 2020

Type Package

Title Detect Differentiation Problems

Version 0.1.1

Date 2020-02-28

Maintainer Vianney Costemalle <vianney.costemalle@insee.fr>

Description An algorithm based on graph theory tools to detect differentiation problems. A differentiation problem occurs when aggregated data are disseminated according to two different nomenclatures. By making the difference for an additive variable X between an aggregate composed of categories of the first nomenclature and an other aggregate, included in that first aggregate, composed of categories of the second nomenclature, it is sometimes possible to derive X on a small aggregate of records which could then lead to a break of confidentiality. The purpose of this package is to detect the set of aggregates composed of categories of the first nomenclature which lead to a differentiation problem, when given a confidentiality threshold. Reference: Vianney Costemalle (2019) <doi: 10.3233/SJI-190564>.

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Suggests testthat

Imports tidyverse, igraph, progress, Rcpp, sf, dplyr, Matrix, data.table

LinkingTo Rcpp

NeedsCompilation yes

Author Vianney Costemalle [aut, cre], Arlindo Dos Santos [aut], Francois Semecurbe [aut]

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2020-02-28 16:10:09 UTC

R topics documented:

`agregate` *Fonction pour agreger le graphe*

Description

On agrege le graphe en utilisant la fonction agregate_one un grand nombre de fois, jusqu'a ce que plus aucune agregation ne soit possible où jusqu'a un threshold limite d'agregations.

Usage

```
agregate(
  m_crois,
  threshold,
  kmax = 0,
  pas_pb = floor(nrow(m_crois)/50),
  verbose = TRUE,
  ...
)
```

Arguments

m_crois	Matrice de croisement.
threshold	threshold de confidentialite.
kmax	Entier indiquant le nombre d'agregtion maximale a effectuer. Si kmax vaut 0, on agrege au maximum.
pas_pb	Entier indiquant a quelle frequence afficher la barre de progression (pb = progress bar). Par defaut, affiche tout les 2% environ.
verbose	Boolean. If TRUE, progress bar is displayed.
...	other parameters

Value

En sortie, on a la matrice de croisement apres agregation.

---

agregate_one *Fonction pour agreger une seule fois 2 zones de z1*

Description

On test toutes les paires de zones de z1 jusqu'a trouver deux zones qu'on peut fusionner. Alors on fusionne et on s'arrête la.

Usage

```
agregate_one(m_crois, threshold, shuffle = TRUE, verbose = TRUE, ...)
```

Arguments

m_crois	Matrice de croisement.
threshold	Entier indiquant le threshold de confidentialite.
shuffle	Booleen indiquant s'il faut melanger les lignes de m_crois apres avoir fusionne ou non. Vaut TRUE par defaut.
verbose	Boolean. If TRUE, displays messages.
...	other parameters

Value

En sortie on a une liste de deux elements : `matrice` donne la matrice de croisement avec une ligne en moins et `continue` indique si l'agregation peut être poursuivie.

---

comp_connexe_list *Desagreger en composantes connexes*

Description

Permet de separer la matrice de croisement en plusieurs sous-matrices de croisement connexes.

Usage

```
comp_connexe_list(m_crois)
```

Arguments

`m_crois`	Matrice de croisement.

Value

On retourne une liste de matrices de croisement etant chacune connexe.

---

decompose_m_crois *Decomposer une matrice de croisement en composantes connexes*

Description

Permet de decomposer totalement une matrice de croisement, qu'elle soit connexe ou non, en plusieurs sous-matrices de croisements connexes "incassables" ou ayant moins de lignes que `taille_max`

Usage

```
decompose_m_crois(m_crois, taille_max = 10)
```

Arguments

`m_crois`	Matrice de croisement.
`taille_max`	Entier indiquant la taille a partir de laquelle on decide de "casser" la matrice ou non. Vaut 10 par defaut.

desagregate_list *Desagreger la liste d'agregat*

Description

Permet de creer une liste de vecteur, chaque vecteur donnant la composition en termes de zones de z1 de l'agregat.

Usage

```
desagregate_list(list_agregat)
```

Arguments

list_agregat	Liste de chaînes de caractere. Chaque element de la liste est une chaîne de caracteres donnant les nomes des zones de z1 composant l'agregat. Chaque nom est separe dans la chaîne de caractere par le symbole ".".

Value

On retourne une liste de vecteur de chaînes de caracteres.

detect_central_node *Detecter un noeud central*

Description

Un noeud central est un noeud qui lorsqu'on le retire rend le graphe non connexe. Cette fonction permet de detecter un noeud parmi les noeud centraux.

Usage

```
detect_central_node(m_crois)
```

Arguments

m_crois	Matrice de croisement.

Value

En sortie on a un entier qui indique le numero du noeud central (c'est-a-dire le numero de la ligne). Si aucun noeud central n'a ete trouve on retourne 0.

`differencierRcpp` *Recherche exhaustive des problemes de differentiation*

Description

Fonction exposée permettant de détecter les différenciations territoire/carreau Elle prépare les paramètres et appelle la fonction récursive explorerRcpp

Usage

```
differencierRcpp(
  iTailleCible,
  iSeuil,
  vNbObsTerritoire,
  vNbObsCarreaux,
  mContiguiteT,
  mContiguiteTC
)
```

Arguments

`iTailleCible`	: entiers indiquant la taille des agrégats à tester
`iSeuil`	: entier indiquant le seuil en dessous duquel il y a rupture du secret statistique
`vNbObsTerritoire`	: vecteur d'entiers contenant le nombre d'observations pour chaque territoire
`vNbObsCarreaux`	: vecteur d'entiers contenant le nombre d'observations pour chaque carreau
`mContiguiteT`	: matrice d'entiers indiquant la contiguité territoire/territoire - la diagonale est supposée à zéro
`mContiguiteTC`	: matrice d'entiers indiquant la contiguité territoire/carreau - les territoires sont en ligne et les carreaux en colonne

Details

Remarque : Il est inutile d'explorer des tailles d'agrégats supérieures à nbTerritoires/2

Value

liste de vecteurs dont chaque élément est une différenciation. Chaque élément est un vecteur constitué - des indices des territoires de l'agrégat - du nombre d'observations différenciées à l'intérieur - et enfin du nombre d'observations différenciées à l'extérieur

Author(s)

Arlindo Dos Santos / PSAR Analyse urbaine

diffman *diffman package*

Description

Package to manage differentiation

Details

Differentiation is a technique to deduce information on small agregates from a source of information that has been disseminated according to two different nomenclatures. Knowing the value of additive variables on agregates of observations according to the nomenclatures, one can decude the value of these variables on new agregates. If those agregates are smaller than a given threshold then we can considerer there is a confidentiality break for the observations within those agregates. It order to respect statistical secrecy for every observations, one need to check for the possible problematic differentiations that a data user can compute. The objective of this package is to provide a tool for detecting all observations at risk of differentiation.

find_id_obs_risque *Recuperer les identifiants des observations a risque*

Description

Permet de recuperer les identifiants des observations a risque de differenciation. Permet egalement de donner le nombre d'observations qu'il y a sur les zones "internes" et "externes" lorsqu'on effectue la differenciation.

Usage

```
find_id_obs_risque(list_agregat, t_ind, threshold, verbose = TRUE)
```

Arguments

list_agregat	Liste de vecteurs de chaînes de caracteres donnant les noms des zones de z1 constituant les differents agregats. Ces agregats ont dû prealablement être identifies comme conduisant a un probleme de differenciation.
t_ind	La table individuelle donnant pour chaque observation la zone de z1 et la zonne de z2 auquelle elle appartient.
threshold	Entier indiquant le threshold de confidentialite.
verbose	Boolean. If TRUE, progress bar is displayed.

Details

Il se peut que certains agregats conduisent au même probleme de differenciation, c'est-a-dire qu'ils permettent de deduire de l'information sur le même groupe d'observation. Dans ce cas, cette fonction ne garde que l'agregat de plus petite taille (en terme de nombre de zones de z1 impliques dans l'agregat). Il se peut donc qu'en sortie, le nombre d'agregats soit inferieur au nombre d'agregats en entree.

Value

En sortie on obtient un data.frame/data.table donnant la liste des observations a risque, sans doublons, et indiquant pour chaque observation a risque, l'agregat sur lequel la differenciation est faite, le type de differenciation (interne ou externe), la taille de l'agregat (nombre de zones de z1) et le nombre d'observations a risque dans la même differenciation

---

`find_pbm_diff` *Perform all the process to detect risky observations*

Description

Allow from a table of observations for which there are two different nomenclatures (z1 and z2) to determine the observations at risk when using the differentiation technique

Usage

```
find_pbm_diff(
  t_ind,
  threshold,
  max_agregate_size,
  save_file = NULL,
  simplify = TRUE,
  verbose = TRUE
)
```

Arguments

`t_ind`	The table of observations (data.frame or data.table). Each row correspond to an observtion and for each observation we must know in which category of the z1 nomenclature it belongs and in which category of the z2 nomenclature.
`threshold`	Strictly positive integer indicating the confidentiality threshold. Observations are considered at risk if one can deduce information on a agregate of n observations where n < threshold.
`max_agregate_size`	Integer indicating the maximal size of agregates which are tested exhaustively. If that number is too large (greater than 30), the computations may not end because of the combinations number that can become very large. Also the RAM can be overloaded.

save_file	Character indicating the suffix of the name of the saved results. If is null, results are not writing on the hardware. The path root is taken from the working directory (getwd()).
simplify	Boolean. If TRUE then the graph simplification (merging + splitting) occures. Otherwise the exhaustive search is directly applied on the original graph.
verbose	Boolean. If TRUE (default), the different steps of the process are notified and progress bars provide an estimation of time left.

Details

Risky observations because of differentiation are the ones for which information can be deduced on agregates smaller than the confidentiality threshold. For example, considering the confidentiality threshold is 10 and if by making the difference between some categories of z1 and some categories of z2 one can deduce the value of a variable for 5 observations, then those 5 observations are considered as "risky".

Value

As an output there is a data.table or data.frame with five columns :

1. $id_obs for the observation at risk
2. $agregat for the agregate of categories from z1 nomenclature on which the differentiation is performed
3. $agregat_size indicating the number of categories composing the agregate
4. $nb_obs the number of observations on which information is deduced when the differentiation is computed (nb_obs must be stricly inferior to $threshold)
5. $type_diff the type of differentiation between "internal" or "external".

Examples

```
res_diff <- find_pbm_diff(t_ex,threshold = 5,max_agregate_size = 15)
```

---

fusion *Fonction qui fusionne les zones numero i et numero j*

Description

Fusion des zones i et j du zonage z1.

Usage

```
fusion(i, j, m_crois, col_to_suppress)
```

Arguments

i, j	Entiers indiquant les zonages de z1.
m_crois	Matrice de croisement.
col_to_suppress	Vecteur d'entiers indiquant les colonnes a supprimer.

Details

Les colonnes a supprimer col_to_suppress sont determinees lors du test de fusion. Si le test conclue qu'il faut fusionner deux zones, alors on retourne en plus les zones de z2 qui sont a l'interieur de cette zone fusionnee.

Value

En sortie, on a la matrice de croisement avec une ligne en moins, et potentiellement des colonnes en moins egalement.

---

is_connected_m2 *Tester s'il existe un chemin, selon la méthode 2*

Description

Fonction permettant de tester s'il existe un chemin entre les noeuds i et j où chaque arête a une valeur plus grande que le threshold de confidentialité moins la valeur du lien entre i et j.

Usage

```
is_connected_m2(i, j, m_graph, v_arete, threshold)
```

Arguments

i, j	Entiers indiquant deux sommets différents du graphe.
m_graph	Matrice carré d'adjacence. Si on note a_ij l'élément de cette matrice correspondant à la ième ligne et jème colonne, alors a_ij=0 si les sommets i et j ne sont pas connectés et sinon a_ij est un entier qui indique la valeur de l'arête entre i et j.
v_arete	valeur du lien entre i et j
threshold	threshold de confidentialité.

Value

Un booléen qui vaut TRUE si les deux sommets i et j sont connectés, selon la méthode 2, et FALSE sinon.

matrix_atomize *Completement decomposer un graphe.*

Description

Permet de decomposer un graphe, de façon iterative, jusqu'a ce que tous les sous-graphes soient "incassables".

Usage

```
matrix_atomize(m_crois_connexe, etat = "cassable", taille_max)
```

Arguments

m_crois_connexe	Matrice de croisement connexe.
etat	Caracteres parmi "cassable" ou "incassable" indiquant l'etat de la matrice de croisement connexe.
taille_max	Entier indiquant a partir de quelle taille de graphe on decide de le decomposer (le "casser").

Value

En sortie on a une liste de matrices de croisement connexes soient "incassables" soient ayant moins de lignes que taille_max.

matrix_break *Casser un graphe en plusieurs sous-graphes*

Description

Cette fonction permet de decomposer un graphe connexe en plusieurs sous-graphes connnexe apres avoir detecter un noeud central.

Usage

```
matrix_break(m_crois_connexe)
```

Arguments

m_crois_connexe	Matrice de croisement connexe.

Value

On retourne une liste de liste. Chaque sous-liste est composee d'une matrice m et d'un etat parmi "cassable" ou "incassable".

`matrix_crois` *Obtenir une matrice (sparse) qui croise les zones de z1 et celles de z2*

Description

La matrice de croisement est une matrice dont les lignes indiquent les zones du zoange z1 et les colonnes les zones du zonage z2. Chaque élément de la matrice donne le nombre d'observations situées à l'intersection entre une zone de z1 et une zone de z2.

Usage

```
matrix_crois(t_crois)
```

Arguments

`t_crois`	Table de croisement (1 colonne z1 et une colonne z2).

Value

En sortie on obtient une matrice sparse de croisement. Les noms des lignes correspondent aux noms des zones de z1 et les noms des colonnes aux noms des zones de z2.

`matrix_crois_from_tind`
Creation de la matrice de croisement

Description

Permet de creer directement la matrice de croisement a partir de la table individuelle des observations. Wrapper pour les fonctions simplify_z2_rem, tab_crois, simplify_z2_fus, matrix_crois et simplify_mcrois

Usage

```
matrix_crois_from_tind(t_ind)
```

Arguments

`t_ind`	La table individuelle (format data.table) Chaque ligne représente une observation et elle doit au moins contenir deux colonnes nommées 'z1' et 'z2' et indiquant à quels zonages appartient l'observation.

Value

matrice sparse de croisement

matrix_graphe *Constuire la matrice d'adjacence du graphe*

Description

Fonction permettant à partir de la matrice de croisement de déterminer la matrice d'adjacence du graphe. Cette matrice de graphe est pondérée et non symmétrique (ce qui correspond à un graphe orienté).

Usage

```
matrix_graphe(m_crois, multi = TRUE)
```

Arguments

m_crois	Matrice de croisement.
multi	Booléen indiquant s'il faut considérer les zones de z2 recouvrant trois zones ou plus de z1.

Details

L'option multi permet de choisir si on prend en compte ou non les zones de z2 recouvrant 3 zones de z1 ou plus. En effet si on les prend en compte, alors certaines observations sont comptées plusieurs fois dans le graphe, ce qui peut conduire à de mauvaises interprétations.

Value

En sortie on obtient une matrice carré d'adjacence.

matrix_liens *Créer la matrice de liens (ou matrice de contiguïté)*

Description

Cette fonction permet de créer une matrice carré de booléens de taille égale au nombre de zones du zonage z1.

Usage

```
matrix_liens(m_crois)
```

Arguments

m_crois	Matrice de croisement.

Details

Un élément (i,j) de cette matrice vaut TRUE (ou 1) si les zones i et j du zonage z1 sont contigues, c'est-à-dire s'il existe au moins une zone de z2 recouvrant à la fois i et j. Les élements de la diagonales portent la valeur FALSE.

Value

En sortie on a une matrice carré de booléens.

---

search_diff_agregate *Tester toutes les combinaisons possibles (jusqua une certaine taille)*

Description

Permet de tester, pour une liste de matrice de croisement, les agregats de zones de z1 conduisant a un probleme de differenciation.

Usage

```
search_diff_agregate(list_m_crois, threshold, max_agregate_size = 20)
```

Arguments

list_m_crois	Liste de matrices de croisement.
threshold	threshold de confidentialite.
max_agregate_size	Entier indiquant la taille maximale des agregats a tester

Value

On retourne une liste d'agregats.

---

simplify_mcrois *Simplifier la matrice de croisement*

Description

Simplifier la matrice de croisement revient à supprimer deux types de colones (et donc supprimer des zones du zonage z2). 1 - on supprime les colonnes vides, c'est-à-dire ne contenant que des 0. 2 - on supprime les colonnes ne contenant qu'un seul élément différent de 0.

Usage

```
simplify_mcrois(m_crois)
```

Arguments

m_crois	Une matrice de croisement.

Value

En matrice on a une matrice de croisement simplifiée, c'est-à-dire avec moins de colonnes.

---

simplify_z2_fus *Fusionne les zones de z2 recouvrant les mêmes zones de z1*

Description

Cette fonction permet de fusionner les zones de z2 dont les observations sont réparties sur les mêmes zones de z1. Cela permet de diminuer le nombre de zonages de z2.

Usage

```
simplify_z2_fus(t_crois)
```

Arguments

t_crois	Table de croisement (c'est la table de fréquences lorsqu'on croise les deux zonages z1 et z2).

Value

En sortie on obtient une table de croisement avec moins de classes pour le zonage z2.

---

simplify_z2_rem *Enlever les zones de z2 entièrement incluses dans une zone de z1*

Description

Cette fonction permet de retirer de la table individuelle, les observations contenues dans une zone de z2 entièrement incluse dans une seule zone de z1. Une zone de z2 est entièrement incluse dans une autre zone si toutes les observations qu'elle contient sont également contenue dans cette autre zone.

Usage

```
simplify_z2_rem(t_ind)
```

Arguments

t_ind	La table individuelle (format data.table) Chaque ligne représente une observation et elle doit au moins contenir deux colonnes nommées 'z1' et 'z2' et indiquant à quels zonages appartient l'observation.

Value

En sortie on récupère une table individuelle avec des observations en moins

---

`tab_crois` *Creer la matrice de croisement entre deux zonages*

Description

Compte le nombre d'observations aux intersections des deux zonages.

Usage

```
tab_crois(t_ind)
```

Arguments

`t_ind`	La table individuelle (format data.table) Chaque ligne représente une observation et elle doit au moins contenir deux colonnes nommées 'z1' et 'z2' et indiquant à quels zonages appartient l'observation.

Value

table de croisement

`test_fus` *Tester si on peut fusionner deux zones de z1*

Description

On teste selon la méthode 1, 2 ou les deux à la fois, si deux zones i et j de z1 peuvent être fusionnées ou non

Usage

```
test_fus(i, j, m_crois, threshold, methode = "both")
```

Arguments

`i, j`	Entiers indiquant les zones de z1.
`m_crois`	Matrice de croisement.
`threshold`	threshold de confidentialité.
`methode`	Méthode à choisir parmi "m1", "m2" et "both".

Value

En sortie, on a une liste contenant deux élements, fus et col. fus vaut TRUE ou FALSE et indique s'il faut fusionner les deux lignes. col est un vecteur d'entier qui indique les numéros de colonne à supprimer.

`test_fus_m1` *Tester, selon la méthode 1, si on peut fusionner deux sommets*

Description

Fonction permettant de tester si les sommets i et j peuvent être fusionnés. La méthode 1 consiste à regarder les liens entre les deux sommets et voir si ces deux liens ont une valeur plus grande que le threshold de confidentialité. Si c'est le cas, on peut fusionner les deux sommets.

Usage

```
test_fus_m1(i, j, m_crois, threshold)
```

Arguments

`i, j`	Entiers indiquant deux lignes différentes de `m_crois`.
`m_crois`	Matrice de croisement.
`threshold`	threshold de confidentialité.

Details

La fonction renvoie TRUE si on peut fusionner les deux sommets (c'est-à-dire qu'il n'y a aucun problème de différenciation en considérant un sommet sans considérer l'autre), et FALSE sinon.

Value

En sortie, on a une liste contenant deux élements, fus et col. fus vaut TRUE ou FALSE et indique s'il faut fusionner les deux lignes. col est un vecteur d'entier qui indique les numéros de colonne à supprimer.

`test_fus_m2` *Tester, selon la méthode 2, si on peut fusionner deux noeuds ou non.*

Description

Permet de tester si les deux sommets i et j du graphe peuvent être fusionnés.

Usage

```
test_fus_m2(i, j, m_crois, threshold)
```

Arguments

`i, j`	Entiers indiquant les noeuds du graphe à tester.
`m_crois`	Matrice de croisement.
`threshold`	threshold de confidentialité.

Details

La méthode 2 s'applique après la méthode 1. Si les sommets i et j sont reliés par une arête en-dessous du threshold de confidentialité la méthode 1 conclut qu'on ne peut pas les fusionner. Mais la méthode 2 regarde s'il existe un autre chemin entre i et j, avec des arêtes ayant des valeurs suffisamment élevées (plus grande que le threshold - la valeur du lien entre i et j). Si c'est le cas, on peut en déduire qu'il n'y aura pas de problème de différentiation en considérant i et j séparément. On peut donc fusionner ces deux zones.

Value

En sortie, on a une liste contenant deux élements, fus et col. fus vaut TRUE ou FALSE et indique s'il faut fusionner les deux lignes. col est un vecteur d'entier qui indique les numéros de colonne à supprimer.

---

t_ex *A simulated dataset of 34695 observations*

Description

A dataset containing the categories observations belongs to. Each observation is included in two categories : one from the nomenclature z1 and one from the nomenclature z2.

Usage

```
t_ex
```

Format

A data frame with 34695 rows and 3 variables:

id number to identify the observation

z1 first nomenclature. Each category is coded by an integer (between 5 and 363)

z2 second nomenclature. Each category is coded by an integer (between 1 and 406)

---

valeurs_lien *Valeurs des arêtes reliant i et j*

Description

i et j étant deux zones de z1, cette fonction donne la valeur de l'arête reliant i et j et de celle reliant j à i.

Usage

```
valeurs_lien(i, j, m_crois, ...)
```

Arguments

i, j	Entiers indiquant les zones de z1.
m_crois	Matrice de croisement.
...	d'autres paramètres hérités de zones_front.

Value

En sortie on obtient un vecteur de deux entiers donnant la valeur de l'arête de i vers j et de celle de j vers i.

---

zones_front *Détermine les zones de z2 reouvrant plusieurs zones de z1*

Description

Cette fonction permet de déterminer les colonnes de m_crois correspondant aux zones de z2 à la frontière entre les zones i et j de z1.

Usage

```
zones_front(i, j, m_crois, type = "all")
```

Arguments

i, j	Entiers indiquant les zones de z1.
m_crois	Matrice de croisement.
type	Character entre "all", "multi" et "unique", pour sélectionner le type de zones de z2 à prendre en compte.

Value

En sortie on a un vecteur d'entiers indiquant les numéros de colonnes de m_crois impliqués.

Index