ANALYSE TEXTE: identifier les comportements d'achats fréquents parmi les clients

Serigne Fallou MBacke NGOM

2023-12-14

Les techniques d'analyse de liens entre valeurs des variables seront mises en œuvre sur des données de transactions de ventes correspondant à des paniers d'achats dans le commerce de détail.

L'objectif final de cette application est d'identifier les comportements d'achats fréquents parmi les clients, c'est-à-dire les articles fréquemment achetés ensembles.

CHARGEMENT ET VISUALISATION DES DONNEES GROCERIES

L'ensemble de données Data Groceries contient 9 836 tickets de transactions de ventes de détail. Cet ensemble représenté sous forme d'une matrice binaire, décrit pour chaque transaction la liste des articles achetés ensemble.

```
groceries <- read.table("ANALYSE DE TEXTE/Data Groceries.csv", header=TRUE, dec=".", sep="\t", stringsAsFactors=T)
ncol(groceries)
```

```
## [1] 169
```

```
colnames(groceries)
```

```
##   [1] "Citrus.fruit"              "Semi.finished.bread"
##   [3] "Margarine"                 "Ready.soups"
##   [5] "Tropical.fruit"            "Yogurt"
##   [7] "Coffee"                    "Whole.milk"
##   [9] "Pip.fruit"                 "Cream.cheese"
##  [11] "Meat.spreads"              "Other.vegetables"
##  [13] "Condensed.milk"            "Long.life.bakery.product"
##  [15] "Butter"                    "Rice"
##  [17] "Abrasive.cleaner"          "Rolls.buns"
##  [19] "Uht.milk"                  "Bottled.beer"
##  [21] "Liquor..appetizer."        "Pot.plants"
##  [23] "Cereals"                   "White.bread"
##  [25] "Bottled.water"             "Chocolate"
##  [27] "Curd"                      "Flour"
##  [29] "Dishes"                    "Beef"
##  [31] "Frankfurter"               "Soda"
##  [33] "Chicken"                   "Sugar"
##  [35] "Fruit.vegetable.juice"     "Newspapers"
##  [37] "Packaged.fruit.vegetables" "Specialty.bar"
##  [39] "Butter.milk"               "Pastry"
##  [41] "Processed.cheese"          "Detergent"
##  [43] "Root.vegetables"           "Frozen.dessert"
##  [45] "Sweet.spreads"             "Salty.snack"
##  [47] "Waffles"                   "Candy"
##  [49] "Bathroom.cleaner"          "Canned.beer"
##  [51] "Sausage"                   "Brown.bread"
##  [53] "Shopping.bags"             "Beverages"
##  [55] "Hamburger.meat"            "Spices"
##  [57] "Hygiene.articles"          "Napkins"
##  [59] "Pork"                      "Berries"
##  [61] "Whipped.sour.cream"        "Artif..sweetener"
##  [63] "Grapes"                    "Dessert"
##  [65] "Zwieback"                  "Domestic.eggs"
##  [67] "Spread.cheese"             "Misc..beverages"
##  [69] "Hard.cheese"               "Cat.food"
##  [71] "Ham"                       "Turkey"
##  [73] "Baking.powder"             "Pickled.vegetables"
##  [75] "Oil"                       "Chewing.gum"
##  [77] "Chocolate.marshmallow"     "Ice.cream"
##  [79] "Frozen.vegetables"         "Canned.fish"
##  [81] "Seasonal.products"         "Curd.cheese"
##  [83] "Red.blush.wine"            "Frozen.potato.products"
##  [85] "Specialty.fat"             "Specialty.chocolate"
##  [87] "Candles"                   "Flower..seeds."
##  [89] "Sparkling.wine"            "Salt"
##  [91] "Frozen.meals"              "Canned.vegetables"
##  [93] "Onions"                    "Herbs"
##  [95] "White.wine"                "Brandy"
##  [97] "Photo.film"                "Sliced.cheese"
```

```
##  [99] "Pasta"                    "Softener"
## [101] "Cling.film.bags"          "Fish"
## [103] "Male.cosmetics"           "Canned.fruit"
## [105] "Instant.food.products"    "Soft.cheese"
## [107] "Honey"                    "Dental.care"
## [109] "Popcorn"                  "Cake.bar"
## [111] "Snack.products"           "Flower.soil.fertilizer"
## [113] "Specialty.cheese"         "Finished.products"
## [115] "Cocoa.drinks"             "Dog.food"
## [117] "Prosecco"                 "Frozen.fish"
## [119] "Make.up.remover"          "Cleaner"
## [121] "Female.sanitary.products" "Dish.cleaner"
## [123] "Cookware"                 "Meat"
## [125] "Tea"                      "Mustard"
## [127] "House.keeping.products"   "Skin.care"
## [129] "Potato.products"          "Liquor"
## [131] "Pet.care"                 "Soups"
## [133] "Rum"                      "Salad.dressing"
## [135] "Sauces"                   "Vinegar"
## [137] "Soap"                     "Hair.spray"
## [139] "Instant.coffee"           "Roll.products"
## [141] "Mayonnaise"               "Rubbing.alcohol"
## [143] "Syrup"                    "Liver.loaf"
## [145] "Baby.cosmetics"           "Organic.products"
## [147] "Nut.snack"                "Kitchen.towels"
## [149] "Frozen.chicken"           "Light.bulbs"
## [151] "Ketchup"                  "Jam"
## [153] "Decalcifier"              "Nuts.prunes"
## [155] "Liqueur"                  "Organic.sausage"
## [157] "Cream"                    "Toilet.cleaner"
## [159] "Specialty.vegetables"     "Baby.food"
## [161] "Pudding.powder"           "Tidbits"
## [163] "Whisky"                   "Frozen.fruits"
## [165] "Bags"                     "Cooking.chocolate"
## [167] "Sound.storage.medium"     "Kitchen.utensil"
## [169] "Preservation.products"
```

```
library(arules)
```

```
## Warning: le package 'arules' a été compilé avec la version R 4.2.3
```

```
## Le chargement a nécessité le package : Matrix
```

```
## Warning: le package 'Matrix' a été compilé avec la version R 4.2.3
```

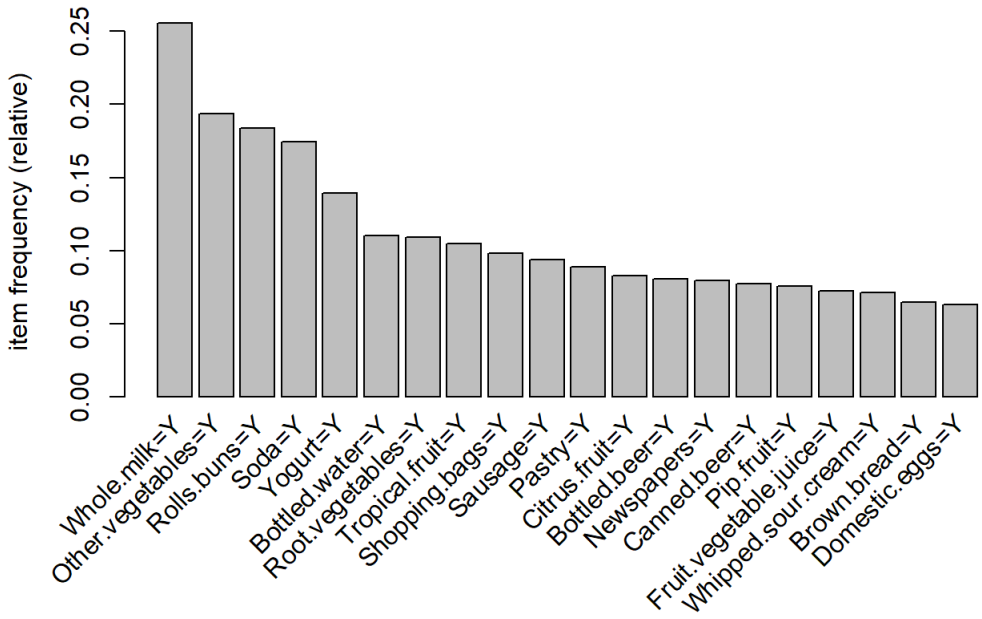
```
##
## Attachement du package : 'arules'
```

```
## Les objets suivants sont masqués depuis 'package:base':
##
##     abbreviate, write
```

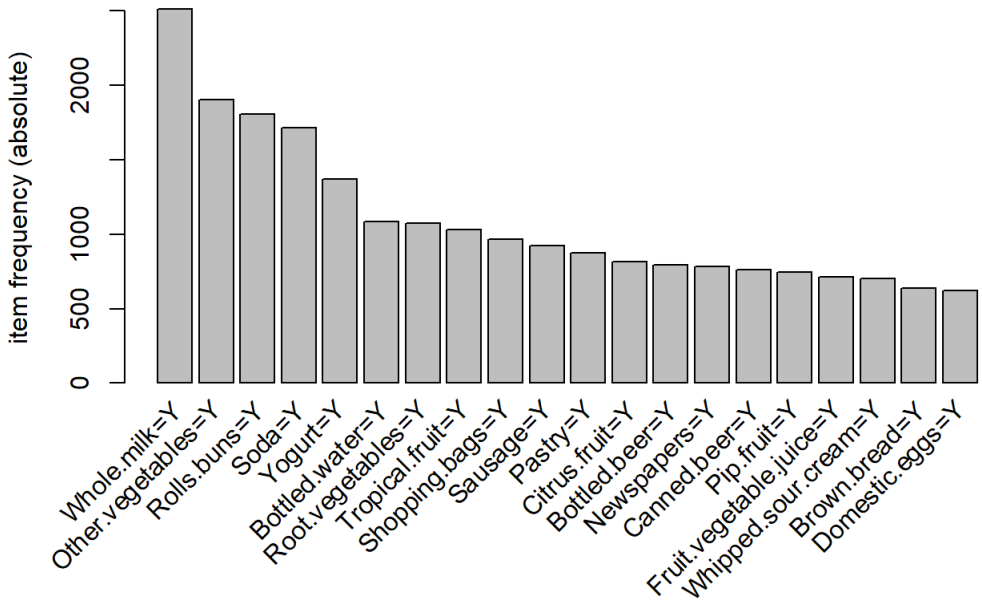
```
# Représentation des données au format transactionnel
groceries_tr <- as(groceries, "transactions")
summary(groceries_tr)
```

```
## transactions as itemMatrix in sparse format with
##  9835 rows (elements/itemsets/transactions) and
##  169 columns (items) and a density of 0.02609146
##
## most frequent items:
##       Whole.milk=Y Other.vegetables=Y       Rolls.buns=Y            Soda=Y
##               2513               1903               1809               1715
##          Yogurt=Y            (Other)
##               1372              34055
##
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 2159 1643 1299 1005  855  645  545  438  350  246  182  117   78   77   55   46
##   17   18   19   20   21   22   23   24   26   27   28   29   32
##   29   14   14    9   11    4    6    1    1    1    1    3    1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   4.409   6.000  32.000
##
## includes extended item information - examples:
##                  labels          variables levels
## 1        Citrus.fruit=Y       Citrus.fruit      Y
## 2 Semi.finished.bread=Y Semi.finished.bread      Y
## 3          Margarine=Y          Margarine      Y
##
## includes extended transaction information - examples:
##   transactionID
## 1             1
## 2             2
## 3             3
```

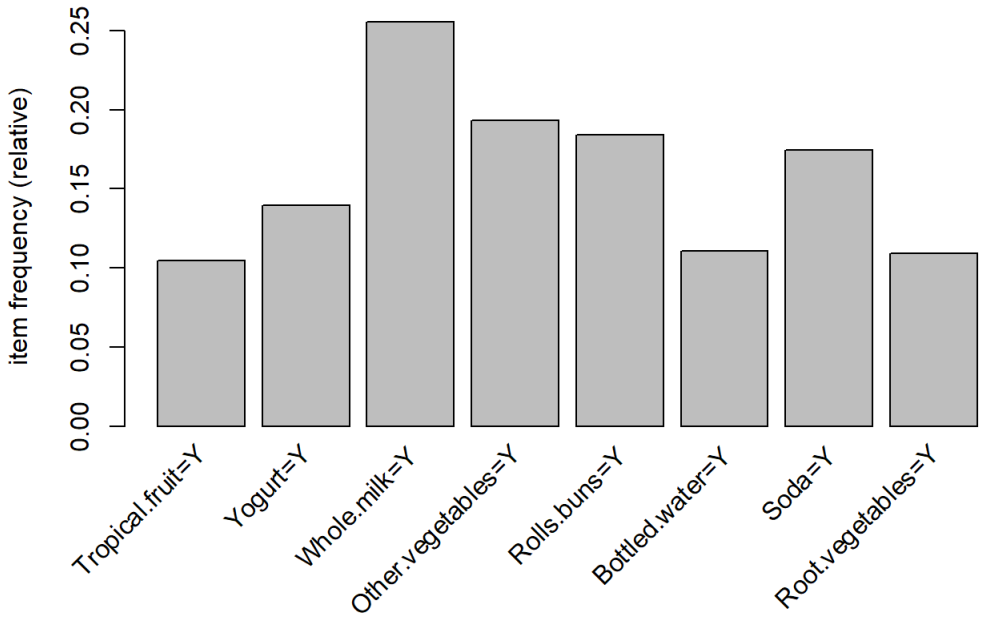
```
# Histogramme d'effectifs des 20 items les plus fréquents
itemFrequencyPlot(groceries_tr, topN=20)
```




```
itemFrequencyPlot(groceries_tr, topN=20,type="absolute")
```




```
itemFrequencyPlot(groceries_tr, support=0.10)
```




```
# Affichage des nombres réels avec 3 décimales
options(digits=3)
```

EXTRACTION DE REGLES D'ASSOCIATION

Nous souhaitons extraire les règles d'association les plus pertinentes montrant les liens entre les achats de deux articles.

```
# Extraction des règles d'association pour minsupport = 1% et minconfiance = 40%
rules1 <- apriori(groceries, parameter = list(supp = 0.01, conf = 0.4, target = "rules"))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.4    0.1    1 none FALSE            TRUE       5    0.01      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 98
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.01s].
## sorting and recoding items ... [88 item(s)] done [0.00s].
## creating transaction tree ... done [0.01s].
## checking subsets of size 1 2 3 4 done [0.08s].
## writing ... [62 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
inspect(rules1)
```

```
##      lhs                          rhs                    support confidence coverage lift count
## [1]  {Hard.cheese=Y}           => {Whole.milk=Y}        0.0101     0.411    0.0245 1.61    99
## [2]  {Butter.milk=Y}           => {Whole.milk=Y}        0.0116     0.415    0.0280 1.62   114
## [3]  {Ham=Y}                   => {Whole.milk=Y}        0.0115     0.441    0.0260 1.73   113
## [4]  {Sliced.cheese=Y}         => {Whole.milk=Y}        0.0108     0.440    0.0245 1.72   106
## [5]  {Oil=Y}                   => {Whole.milk=Y}        0.0113     0.402    0.0281 1.57   111
## [6]  {Onions=Y}                => {Other.vegetables=Y}  0.0142     0.459    0.0310 2.37   140
## [7]  {Hamburger.meat=Y}        => {Other.vegetables=Y}  0.0138     0.416    0.0332 2.15   136
## [8]  {Hamburger.meat=Y}        => {Whole.milk=Y}        0.0147     0.443    0.0332 1.74   145
## [9]  {Sugar=Y}                 => {Whole.milk=Y}        0.0150     0.444    0.0339 1.74   148
## [10] {Cream.cheese=Y}          => {Whole.milk=Y}        0.0165     0.415    0.0397 1.63   162
## [11] {Chicken=Y}               => {Other.vegetables=Y}  0.0179     0.417    0.0429 2.16   176
## [12] {Chicken=Y}               => {Whole.milk=Y}        0.0176     0.410    0.0429 1.60   173
## [13] {White.bread=Y}           => {Whole.milk=Y}        0.0171     0.406    0.0421 1.59   168
## [14] {Frozen.vegetables=Y}     => {Whole.milk=Y}        0.0204     0.425    0.0481 1.66   201
## [15] {Beef=Y}                  => {Whole.milk=Y}        0.0213     0.405    0.0525 1.59   209
## [16] {Curd=Y}                  => {Whole.milk=Y}        0.0261     0.490    0.0533 1.92   257
## [17] {Margarine=Y}             => {Whole.milk=Y}        0.0242     0.413    0.0586 1.62   238
## [18] {Butter=Y}                => {Whole.milk=Y}        0.0276     0.497    0.0554 1.95   271
## [19] {Domestic.eggs=Y}         => {Whole.milk=Y}        0.0300     0.473    0.0634 1.85   295
## [20] {Whipped.sour.cream=Y}    => {Other.vegetables=Y}  0.0289     0.403    0.0717 2.08   284
## [21] {Whipped.sour.cream=Y}    => {Whole.milk=Y}        0.0322     0.450    0.0717 1.76   317
## [22] {Tropical.fruit=Y}        => {Whole.milk=Y}        0.0423     0.403    0.1049 1.58   416
## [23] {Root.vegetables=Y}       => {Other.vegetables=Y}  0.0474     0.435    0.1090 2.25   466
## [24] {Root.vegetables=Y}       => {Whole.milk=Y}        0.0489     0.449    0.1090 1.76   481
## [25] {Yogurt=Y}                => {Whole.milk=Y}        0.0560     0.402    0.1395 1.57   551
## [26] {Yogurt=Y,
##       Curd=Y}                  => {Whole.milk=Y}        0.0101     0.582    0.0173 2.28    99
## [27] {Other.vegetables=Y,
##       Pork=Y}                  => {Whole.milk=Y}        0.0102     0.469    0.0217 1.84   100
## [28] {Whole.milk=Y,
##       Pork=Y}                  => {Other.vegetables=Y}  0.0102     0.459    0.0222 2.37   100
## [29] {Other.vegetables=Y,
##       Butter=Y}                => {Whole.milk=Y}        0.0115     0.574    0.0200 2.24   113
## [30] {Whole.milk=Y,
##       Butter=Y}                => {Other.vegetables=Y}  0.0115     0.417    0.0276 2.15   113
## [31] {Other.vegetables=Y,
##       Domestic.eggs=Y}         => {Whole.milk=Y}        0.0123     0.553    0.0223 2.16   121
## [32] {Whole.milk=Y,
##       Domestic.eggs=Y}         => {Other.vegetables=Y}  0.0123     0.410    0.0300 2.12   121
## [33] {Other.vegetables=Y,
##       Fruit.vegetable.juice=Y} => {Whole.milk=Y}        0.0105     0.498    0.0210 1.95   103
## [34] {Yogurt=Y,
##       Whipped.sour.cream=Y}    => {Other.vegetables=Y}  0.0102     0.490    0.0207 2.53   100
## [35] {Yogurt=Y,
##       Whipped.sour.cream=Y}    => {Whole.milk=Y}        0.0109     0.525    0.0207 2.05   107
## [36] {Other.vegetables=Y,
##       Whipped.sour.cream=Y}    => {Whole.milk=Y}        0.0146     0.507    0.0289 1.98   144
## [37] {Whole.milk=Y,
##       Whipped.sour.cream=Y}    => {Other.vegetables=Y}  0.0146     0.454    0.0322 2.35   144
## [38] {Pip.fruit=Y,
##       Other.vegetables=Y}      => {Whole.milk=Y}        0.0135     0.518    0.0261 2.03   133
```

```
## [39] {Whole.milk=Y,
##       Pip.fruit=Y}            => {Other.vegetables=Y}  0.0135      0.449   0.0301 2.32   133
## [40] {Other.vegetables=Y,
##       Pastry=Y}               => {Whole.milk=Y}        0.0106      0.468   0.0226 1.83   104
## [41] {Citrus.fruit=Y,
##       Root.vegetables=Y}      => {Other.vegetables=Y}  0.0104      0.586   0.0177 3.03   102
## [42] {Citrus.fruit=Y,
##       Yogurt=Y}               => {Whole.milk=Y}        0.0103      0.474   0.0217 1.86   101
## [43] {Citrus.fruit=Y,
##       Other.vegetables=Y}     => {Whole.milk=Y}        0.0130      0.451   0.0289 1.76   128
## [44] {Citrus.fruit=Y,
##       Whole.milk=Y}           => {Other.vegetables=Y}  0.0130      0.427   0.0305 2.21   128
## [45] {Other.vegetables=Y,
##       Bottled.water=Y}        => {Whole.milk=Y}        0.0108      0.434   0.0248 1.70   106
## [46] {Tropical.fruit=Y,
##       Root.vegetables=Y}      => {Other.vegetables=Y}  0.0123      0.585   0.0210 3.02   121
## [47] {Tropical.fruit=Y,
##       Root.vegetables=Y}      => {Whole.milk=Y}        0.0120      0.570   0.0210 2.23   118
## [48] {Tropical.fruit=Y,
##       Yogurt=Y}               => {Other.vegetables=Y}  0.0123      0.420   0.0293 2.17   121
## [49] {Tropical.fruit=Y,
##       Yogurt=Y}               => {Whole.milk=Y}        0.0151      0.517   0.0293 2.02   149
## [50] {Tropical.fruit=Y,
##       Rolls.buns=Y}           => {Whole.milk=Y}        0.0110      0.446   0.0246 1.75   108
## [51] {Tropical.fruit=Y,
##       Other.vegetables=Y}     => {Whole.milk=Y}        0.0171      0.476   0.0359 1.86   168
## [52] {Tropical.fruit=Y,
##       Whole.milk=Y}           => {Other.vegetables=Y}  0.0171      0.404   0.0423 2.09   168
## [53] {Yogurt=Y,
##       Root.vegetables=Y}      => {Other.vegetables=Y}  0.0129      0.500   0.0258 2.58   127
## [54] {Yogurt=Y,
##       Root.vegetables=Y}      => {Whole.milk=Y}        0.0145      0.563   0.0258 2.20   143
## [55] {Rolls.buns=Y,
##       Root.vegetables=Y}      => {Other.vegetables=Y}  0.0122      0.502   0.0243 2.59   120
## [56] {Rolls.buns=Y,
##       Root.vegetables=Y}      => {Whole.milk=Y}        0.0127      0.523   0.0243 2.05   125
## [57] {Other.vegetables=Y,
##       Root.vegetables=Y}      => {Whole.milk=Y}        0.0232      0.489   0.0474 1.91   228
## [58] {Whole.milk=Y,
##       Root.vegetables=Y}      => {Other.vegetables=Y}  0.0232      0.474   0.0489 2.45   228
## [59] {Other.vegetables=Y,
##       Soda=Y}                 => {Whole.milk=Y}        0.0139      0.425   0.0327 1.67   137
## [60] {Yogurt=Y,
##       Rolls.buns=Y}           => {Whole.milk=Y}        0.0156      0.453   0.0344 1.77   153
## [61] {Yogurt=Y,
##       Other.vegetables=Y}     => {Whole.milk=Y}        0.0223      0.513   0.0434 2.01   219
## [62] {Other.vegetables=Y,
##       Rolls.buns=Y}           => {Whole.milk=Y}        0.0179      0.420   0.0426 1.64   176
```

```
# Ordonnancement des règles par mesures de confiance et support respectivement
rules1 <- sort(rules1, by = c("confidence", "support"))
inspect(rules1)
```

```
##     lhs                     rhs                   support confidence coverage lift count
## [1]  {Citrus.fruit=Y,
##       Root.vegetables=Y}      => {Other.vegetables=Y}  0.0104      0.586   0.0177 3.03   102
## [2]  {Tropical.fruit=Y,
##       Root.vegetables=Y}      => {Other.vegetables=Y}  0.0123      0.585   0.0210 3.02   121
## [3]  {Yogurt=Y,
##       Curd=Y}                 => {Whole.milk=Y}        0.0101      0.582   0.0173 2.28    99
## [4]  {Other.vegetables=Y,
##       Butter=Y}               => {Whole.milk=Y}        0.0115      0.574   0.0200 2.24   113
## [5]  {Tropical.fruit=Y,
##       Root.vegetables=Y}      => {Whole.milk=Y}        0.0120      0.570   0.0210 2.23   118
## [6]  {Yogurt=Y,
##       Root.vegetables=Y}      => {Whole.milk=Y}        0.0145      0.563   0.0258 2.20   143
## [7]  {Other.vegetables=Y,
##       Domestic.eggs=Y}        => {Whole.milk=Y}        0.0123      0.553   0.0223 2.16   121
## [8]  {Yogurt=Y,
##       Whipped.sour.cream=Y}   => {Whole.milk=Y}        0.0109      0.525   0.0207 2.05   107
## [9]  {Rolls.buns=Y,
##       Root.vegetables=Y}      => {Whole.milk=Y}        0.0127      0.523   0.0243 2.05   125
## [10] {Pip.fruit=Y,
##       Other.vegetables=Y}     => {Whole.milk=Y}        0.0135      0.518   0.0261 2.03   133
## [11] {Tropical.fruit=Y,
##       Yogurt=Y}               => {Whole.milk=Y}        0.0151      0.517   0.0293 2.02   149
## [12] {Yogurt=Y,
```

```
##       Other.vegetables=Y}      => {Whole.milk=Y}        0.0223      0.513   0.0434 2.01   219
## [13] {Other.vegetables=Y,
##       Whipped.sour.cream=Y}    => {Whole.milk=Y}        0.0146      0.507   0.0289 1.98   144
## [14] {Rolls.buns=Y,
##       Root.vegetables=Y}       => {Other.vegetables=Y}  0.0122      0.502   0.0243 2.59   120
## [15] {Yogurt=Y,
##       Root.vegetables=Y}       => {Other.vegetables=Y}  0.0129      0.500   0.0258 2.58   127
## [16] {Other.vegetables=Y,
##       Fruit.vegetable.juice=Y} => {Whole.milk=Y}        0.0105      0.498   0.0210 1.95   103
## [17] {Butter=Y}                => {Whole.milk=Y}        0.0276      0.497   0.0554 1.95   271
## [18] {Curd=Y}                  => {Whole.milk=Y}        0.0261      0.490   0.0533 1.92   257
## [19] {Yogurt=Y,
##       Whipped.sour.cream=Y}    => {Other.vegetables=Y}  0.0102      0.490   0.0207 2.53   100
## [20] {Other.vegetables=Y,
##       Root.vegetables=Y}       => {Whole.milk=Y}        0.0232      0.489   0.0474 1.91   228
## [21] {Tropical.fruit=Y,
##       Other.vegetables=Y}      => {Whole.milk=Y}        0.0171      0.476   0.0359 1.86   168
## [22] {Citrus.fruit=Y,
##       Yogurt=Y}                => {Whole.milk=Y}        0.0103      0.474   0.0217 1.86   101
## [23] {Whole.milk=Y,
##       Root.vegetables=Y}       => {Other.vegetables=Y}  0.0232      0.474   0.0489 2.45   228
## [24] {Domestic.eggs=Y}         => {Whole.milk=Y}        0.0300      0.473   0.0634 1.85   295
## [25] {Other.vegetables=Y,
##       Pork=Y}                  => {Whole.milk=Y}        0.0102      0.469   0.0217 1.84   100
## [26] {Other.vegetables=Y,
##       Pastry=Y}                => {Whole.milk=Y}        0.0106      0.468   0.0226 1.83   104
## [27] {Onions=Y}                => {Other.vegetables=Y}  0.0142      0.459   0.0310 2.37   140
## [28] {Whole.milk=Y,
##       Pork=Y}                  => {Other.vegetables=Y}  0.0102      0.459   0.0222 2.37   100
## [29] {Whole.milk=Y,
##       Whipped.sour.cream=Y}    => {Other.vegetables=Y}  0.0146      0.454   0.0322 2.35   144
## [30] {Yogurt=Y,
##       Rolls.buns=Y}            => {Whole.milk=Y}        0.0156      0.453   0.0344 1.77   153
## [31] {Citrus.fruit=Y,
##       Other.vegetables=Y}      => {Whole.milk=Y}        0.0130      0.451   0.0289 1.76   128
## [32] {Whipped.sour.cream=Y}    => {Whole.milk=Y}        0.0322      0.450   0.0717 1.76   317
## [33] {Whole.milk=Y,
##       Pip.fruit=Y}             => {Other.vegetables=Y}  0.0135      0.449   0.0301 2.32   133
## [34] {Root.vegetables=Y}       => {Whole.milk=Y}        0.0489      0.449   0.1090 1.76   481
## [35] {Tropical.fruit=Y,
##       Rolls.buns=Y}            => {Whole.milk=Y}        0.0110      0.446   0.0246 1.75   108
## [36] {Sugar=Y}                 => {Whole.milk=Y}        0.0150      0.444   0.0339 1.74   148
## [37] {Hamburger.meat=Y}        => {Whole.milk=Y}        0.0147      0.443   0.0332 1.74   145
## [38] {Ham=Y}                   => {Whole.milk=Y}        0.0115      0.441   0.0260 1.73   113
## [39] {Sliced.cheese=Y}         => {Whole.milk=Y}        0.0108      0.440   0.0245 1.72   106
## [40] {Root.vegetables=Y}       => {Other.vegetables=Y}  0.0474      0.435   0.1090 2.25   466
## [41] {Other.vegetables=Y,
##       Bottled.water=Y}         => {Whole.milk=Y}        0.0108      0.434   0.0248 1.70   106
## [42] {Citrus.fruit=Y,
##       Whole.milk=Y}            => {Other.vegetables=Y}  0.0130      0.427   0.0305 2.21   128
## [43] {Other.vegetables=Y,
##       Soda=Y}                  => {Whole.milk=Y}        0.0139      0.425   0.0327 1.67   137
## [44] {Frozen.vegetables=Y}     => {Whole.milk=Y}        0.0204      0.425   0.0481 1.66   201
## [45] {Tropical.fruit=Y,
##       Yogurt=Y}                => {Other.vegetables=Y}  0.0123      0.420   0.0293 2.17   121
## [46] {Other.vegetables=Y,
##       Rolls.buns=Y}            => {Whole.milk=Y}        0.0179      0.420   0.0426 1.64   176
## [47] {Chicken=Y}               => {Other.vegetables=Y}  0.0179      0.417   0.0429 2.16   176
## [48] {Whole.milk=Y,
##       Butter=Y}                => {Other.vegetables=Y}  0.0115      0.417   0.0276 2.15   113
## [49] {Hamburger.meat=Y}        => {Other.vegetables=Y}  0.0138      0.416   0.0332 2.15   136
## [50] {Cream.cheese=Y}          => {Whole.milk=Y}        0.0165      0.415   0.0397 1.63   162
## [51] {Butter.milk=Y}           => {Whole.milk=Y}        0.0116      0.415   0.0280 1.62   114
## [52] {Margarine=Y}             => {Whole.milk=Y}        0.0242      0.413   0.0586 1.62   238
## [53] {Hard.cheese=Y}           => {Whole.milk=Y}        0.0101      0.411   0.0245 1.61    99
## [54] {Whole.milk=Y,
##       Domestic.eggs=Y}         => {Other.vegetables=Y}  0.0123      0.410   0.0300 2.12   121
## [55] {Chicken=Y}               => {Whole.milk=Y}        0.0176      0.410   0.0429 1.60   173
## [56] {White.bread=Y}           => {Whole.milk=Y}        0.0171      0.406   0.0421 1.59   168
## [57] {Beef=Y}                  => {Whole.milk=Y}        0.0213      0.405   0.0525 1.59   209
## [58] {Tropical.fruit=Y,
##       Whole.milk=Y}            => {Other.vegetables=Y}  0.0171      0.404   0.0423 2.09   168
## [59] {Tropical.fruit=Y}        => {Whole.milk=Y}        0.0423      0.403   0.1049 1.58   416
## [60] {Whipped.sour.cream=Y}    => {Other.vegetables=Y}  0.0289      0.403   0.0717 2.08   284
## [61] {Oil=Y}                   => {Whole.milk=Y}        0.0113      0.402   0.0281 1.57   111
## [62] {Yogurt=Y}                => {Whole.milk=Y}        0.0560      0.402   0.1395 1.57   551
```

```
# Règles d'association de taille maximale de 2 items pour minsupport = 1% et minconfiance = 40%
rules2 <- apriori(groceries, parameter = list(supp = 0.01, conf = 0.4, target = "rules", maxlen=2))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.4    0.1    1 none FALSE            TRUE       5    0.01      1
##  maxlen target  ext
##       2  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 98
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [88 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2
```

```
## Warning in apriori(groceries, parameter = list(supp = 0.01, conf = 0.4, :
## Mining stopped (maxlen reached). Only patterns up to a length of 2 returned!
```

```
##  done [0.00s].
## writing ... [25 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
inspect(rules2)
```

```
##      lhs                     rhs                  support confidence coverage
## [1]  {Hard.cheese=Y}      => {Whole.milk=Y}       0.0101  0.411      0.0245
## [2]  {Butter.milk=Y}      => {Whole.milk=Y}       0.0116  0.415      0.0280
## [3]  {Ham=Y}              => {Whole.milk=Y}       0.0115  0.441      0.0260
## [4]  {Sliced.cheese=Y}    => {Whole.milk=Y}       0.0108  0.440      0.0245
## [5]  {Oil=Y}              => {Whole.milk=Y}       0.0113  0.402      0.0281
## [6]  {Onions=Y}           => {Other.vegetables=Y} 0.0142  0.459      0.0310
## [7]  {Hamburger.meat=Y}   => {Other.vegetables=Y} 0.0138  0.416      0.0332
## [8]  {Hamburger.meat=Y}   => {Whole.milk=Y}       0.0147  0.443      0.0332
## [9]  {Sugar=Y}            => {Whole.milk=Y}       0.0150  0.444      0.0339
## [10] {Cream.cheese=Y}     => {Whole.milk=Y}       0.0165  0.415      0.0397
## [11] {Chicken=Y}          => {Other.vegetables=Y} 0.0179  0.417      0.0429
## [12] {Chicken=Y}          => {Whole.milk=Y}       0.0176  0.410      0.0429
## [13] {White.bread=Y}      => {Whole.milk=Y}       0.0171  0.406      0.0421
## [14] {Frozen.vegetables=Y} => {Whole.milk=Y}      0.0204  0.425      0.0481
## [15] {Beef=Y}             => {Whole.milk=Y}       0.0213  0.405      0.0525
## [16] {Curd=Y}             => {Whole.milk=Y}       0.0261  0.490      0.0533
## [17] {Margarine=Y}        => {Whole.milk=Y}       0.0242  0.413      0.0586
## [18] {Butter=Y}           => {Whole.milk=Y}       0.0276  0.497      0.0554
## [19] {Domestic.eggs=Y}    => {Whole.milk=Y}       0.0300  0.473      0.0634
## [20] {Whipped.sour.cream=Y} => {Other.vegetables=Y} 0.0289  0.403     0.0717
## [21] {Whipped.sour.cream=Y} => {Whole.milk=Y}     0.0322  0.450      0.0717
## [22] {Tropical.fruit=Y}   => {Whole.milk=Y}       0.0423  0.403      0.1049
## [23] {Root.vegetables=Y}  => {Other.vegetables=Y} 0.0474  0.435      0.1090
## [24] {Root.vegetables=Y}  => {Whole.milk=Y}       0.0489  0.449      0.1090
## [25] {Yogurt=Y}           => {Whole.milk=Y}       0.0560  0.402      0.1395
##      lift count
## [1]  1.61  99
## [2]  1.62 114
## [3]  1.73 113
## [4]  1.72 106
## [5]  1.57 111
## [6]  2.37 140
## [7]  2.15 136
## [8]  1.74 145
## [9]  1.74 148
## [10] 1.63 162
## [11] 2.16 176
## [12] 1.60 173
## [13] 1.59 168
## [14] 1.66 201
## [15] 1.59 209
## [16] 1.92 257
## [17] 1.62 238
## [18] 1.95 271
## [19] 1.85 295
## [20] 2.08 284
## [21] 1.76 317
## [22] 1.58 416
## [23] 2.25 466
## [24] 1.76 481
## [25] 1.57 551
```

```
# Règles d'association de taille maximale de 2 items pour minsupport = 0,1% et minconfiance = 50%
rules3 <- apriori(groceries, parameter = list(supp = 0.001, conf = 0.5, target = "rules", maxlen=2))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.5    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target  ext
##       2  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 9
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.01s].
## sorting and recoding items ... [157 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2
```

```
## Warning in apriori(groceries, parameter = list(supp = 0.001, conf = 0.5, :
## Mining stopped (maxlen reached). Only patterns up to a length of 2 returned!
```

```
##  done [0.00s].
## writing ... [11 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules3 <- sort(rules3, by=c("confidence", "support"))
inspect(rules3)
```

```
##      lhs                    rhs                 support confidence coverage
## [1]  {Honey=Y}              => {Whole.milk=Y}       0.00112 0.733      0.00153
## [2]  {Cereals=Y}            => {Whole.milk=Y}       0.00366 0.643      0.00569
## [3]  {Rice=Y}               => {Whole.milk=Y}       0.00468 0.613      0.00763
## [4]  {Cocoa.drinks=Y}       => {Whole.milk=Y}       0.00132 0.591      0.00224
## [5]  {Pudding.powder=Y}     => {Whole.milk=Y}       0.00132 0.565      0.00234
## [6]  {Jam=Y}                => {Whole.milk=Y}       0.00295 0.547      0.00539
## [7]  {Baking.powder=Y}      => {Whole.milk=Y}       0.00925 0.523      0.01769
## [8]  {Tidbits=Y}            => {Rolls.buns=Y}       0.00122 0.522      0.00234
## [9]  {Rice=Y}               => {Other.vegetables=Y} 0.00397 0.520      0.00763
## [10] {Cooking.chocolate=Y} => {Whole.milk=Y}       0.00132 0.520      0.00254
## [11] {Specialty.cheese=Y}  => {Other.vegetables=Y} 0.00427 0.500      0.00854
##      lift count
## [1]  2.87 11
## [2]  2.52 36
## [3]  2.40 46
## [4]  2.31 13
## [5]  2.21 13
## [6]  2.14 29
## [7]  2.05 91
## [8]  2.84 12
## [9]  2.69 39
## [10] 2.04 13
## [11] 2.58 42
```

```
# Ciblage de l'item 'Whole.milk=Y' en antécédent des règles
wm1 <- apriori(data=groceries, parameter=list(supp=0.01, conf=0.2), appearance=list(lhs="Whole.milk=Y"), control=list(verbose=F))
wm1 <- sort(wm1, by = c("confidence", "support"))
inspect(wm1)
```

```
##     lhs                rhs                   support confidence coverage lift
## [1] {Whole.milk=Y} => {Other.vegetables=Y} 0.0748  0.293      0.256    1.51
## [2] {Whole.milk=Y} => {Rolls.buns=Y}       0.0566  0.222      0.256    1.21
## [3] {Whole.milk=Y} => {Yogurt=Y}           0.0560  0.219      0.256    1.57
##     count
## [1] 736
## [2] 557
## [3] 551
```

```
# Ciblage de l'item 'Whole.milk=Y' en conséquence des règles
wm2 <- apriori(data=groceries, parameter=list(supp=0.01, conf=0.55), appearance=list(rhs="Whole.milk=Y"), control=list(verbose=F))
wm2 <- sort(wm2, by = c("confidence", "support"))
inspect(wm2)
```

```
##     lhs                                rhs            support confidence
## [1] {Yogurt=Y, Curd=Y}                 => {Whole.milk=Y} 0.0101  0.582
## [2] {Other.vegetables=Y, Butter=Y}     => {Whole.milk=Y} 0.0115  0.574
## [3] {Tropical.fruit=Y, Root.vegetables=Y} => {Whole.milk=Y} 0.0120  0.570
## [4] {Yogurt=Y, Root.vegetables=Y}      => {Whole.milk=Y} 0.0145  0.563
## [5] {Other.vegetables=Y, Domestic.eggs=Y} => {Whole.milk=Y} 0.0123  0.553
##     coverage lift count
## [1] 0.0173   2.28  99
## [2] 0.0200   2.24 113
## [3] 0.0210   2.23 118
## [4] 0.0258   2.20 143
## [5] 0.0223   2.16 121
```

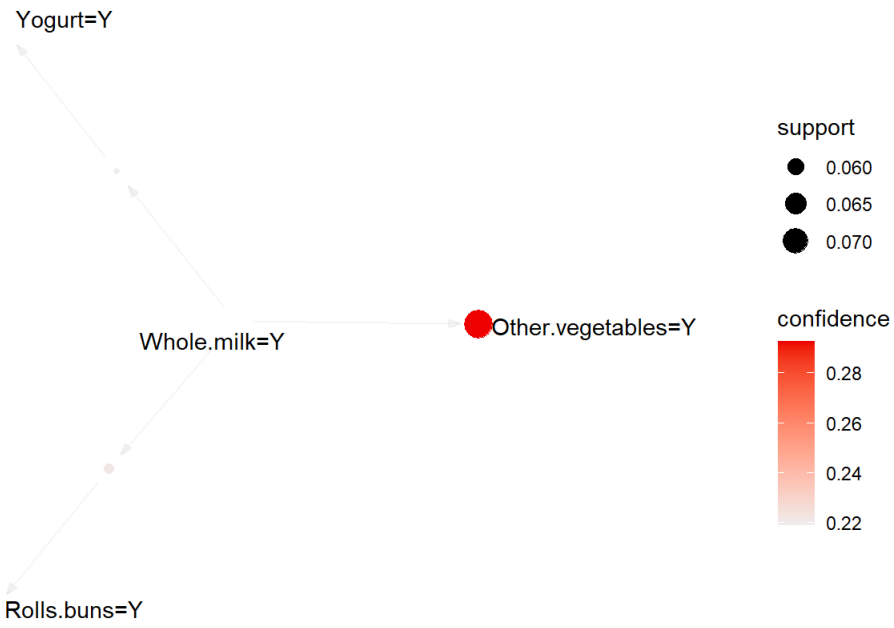
• lhs est l'itemset antécédent de la règle, • rhs est l'itemset conséquence de la règle, • support est la proportion d'instances contenant lhs et rhs, • confidence est la proportion d'instances contenant rhs parmi celles contenant lhs, • coverage est la proportion d'instances contenant lhs, • lift mesure la corrélation statistique entre lhs et rhs : lift(lhs → rhs) = P(lhs rhs) / (P(lhs).P(rhs)).

AFFICHAGES GRAPHIQUES DE REGLES D'ASSOCIATION

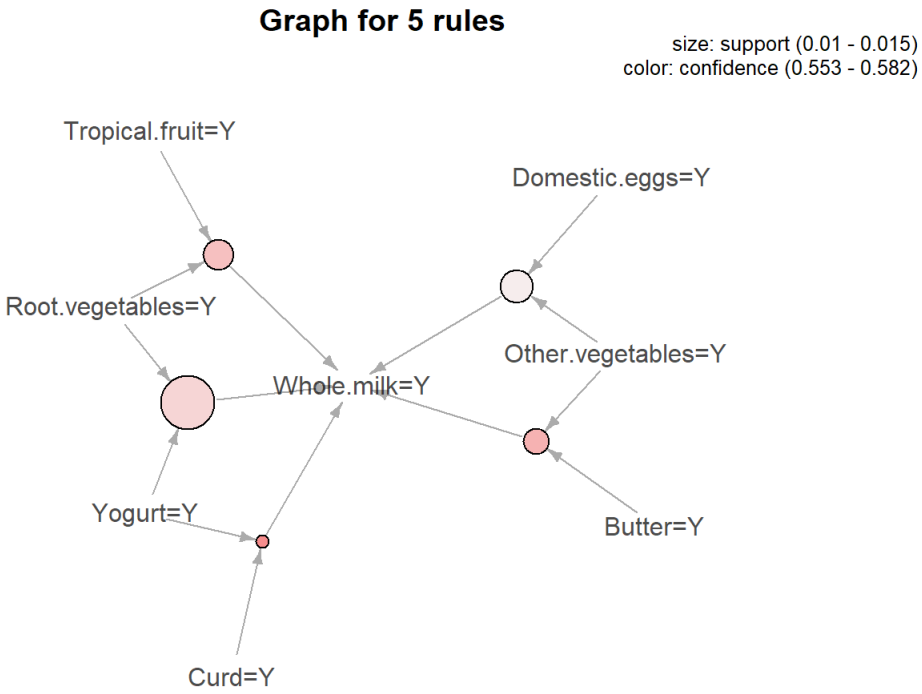
```
library(arulesViz)
```

```
## Warning: le package 'arulesViz' a été compilé avec la version R 4.2.3
```

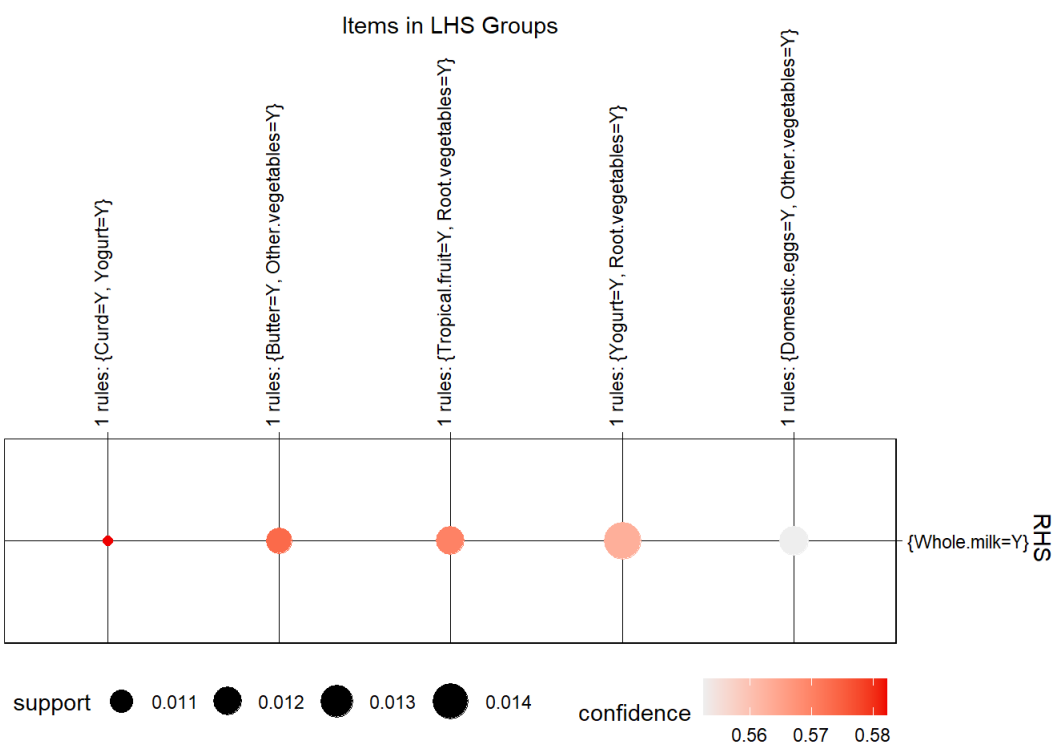
```
# Affichage des règles de 'wm1' sous forme de graphe
plot(wm1, method="graph", shading="confidence")
```




```
# Affichage des règles de 'wm2' sous forme de graphe interactif avec le moteur plotly
plot(wm2, method="graph", shading="confidence", engine='igraph')
```




```
# Affichage des règles de 'wm2' sous forme de boulier
plot(wm2, method="grouped", shading="confidence", engine='ggplot2')
```



Ces représentations permettent d'explorer de grands ensembles de règles de façon simplifiée.