Feuille de Travaux Dirigés n° 1
Règles d'association

1 La bibliothèque `arules`

```
> setwd("C:/Données/ESIEA/Module5A/DataMining_2011-2012/TDs_DM")
```

Installer la bibliothèque `arules` pour le langage **R**.

```
> install.packages("arules")
```

Charger cette bibliothèque.

```
> library("arules")
```

Pour lire le fichier des achats avec le format d'une "Transaction class".

- `file` : fichier au format "csv" ou "txt".
- `format` : single/basket (Pour le `format` "basket", chaque ligne dans le fichier de données des transactions represente une transaction où chacun des items (item labels) sur lesquels elle a porté sont séparés par le caractère spécifié par sep. Pour le `format` "single", chaque ligne correspond à un seul item, qui contient au moins un id pour la transaction et l'item.)
- `rm.duplicates` : "`TRUE`" ou "`FALSE`" (Suppression des doublons)
- `cols` : Pour le format "single", `cols` est un vecteur numeric de longueur deux donnant respectivement les numéros des colonnes (fields) avec la transaction et les ids de l'item. Pour le format panier "basket", `cols` est un nombre donnant le numéro de la colonne (field) avec les ids de la transaction.
- `sep` : "`,`" pour un fichier csv, "`\t`" pour un fichier avec des tabulations comme délimiteurs de colonnes.

```
> txn = read.transactions(file = "transaction.csv",
+     rm.duplicates = FALSE, format = "single", sep = ",",
+     cols = c(1, 2))
```

Pour contrôler si les données sont bien lues par le logiciel.

```
> inspect(txn)
```

```
  items       transactionID
1 {Choclates,
   Marker,
   Pencil}             1001
2 {Choclates,
```

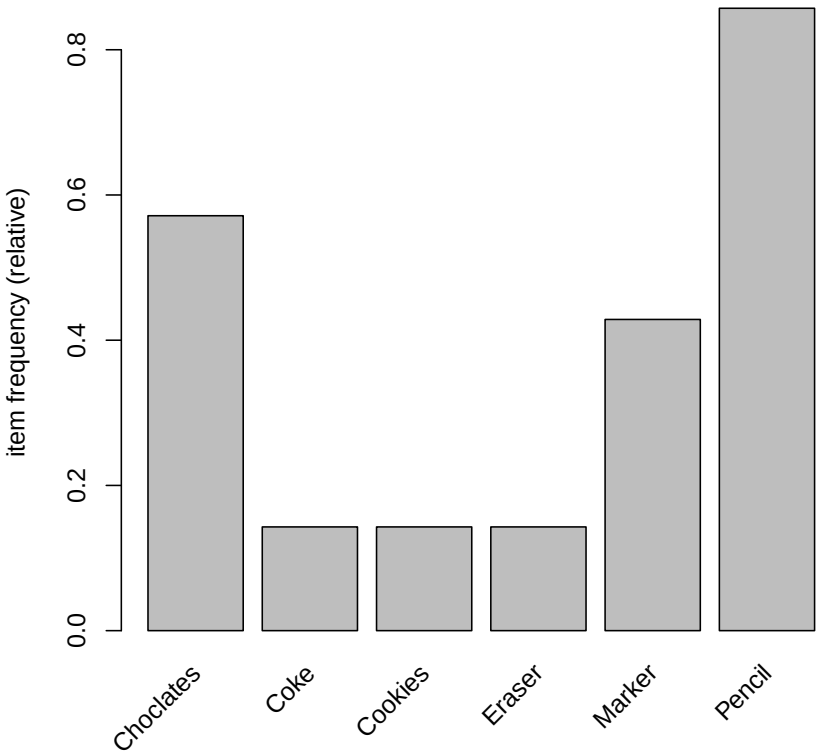
```
   Pencil}             1002
3 {Coke,
   Eraser,
   Pencil}             1003
4 {Choclates,
   Cookies,
   Pencil}             1004
5 {Marker}             1005
6 {Marker,
   Pencil}             1006
7 {Choclates,
   Pencil}             1007
```

Nous pouvons visualiser la répartition des objets entre les achats à l'aide des commandes image(txn) et itemFrequencyPlot(txn) pour visualiser les fréquences empiriques de chaque item dans une transaction de l'objet txn.

```
> inspect(txn)

  items       transactionID
1 {Choclates,
   Marker,
   Pencil}             1001
2 {Choclates,
   Pencil}             1002
3 {Coke,
   Eraser,
   Pencil}             1003
4 {Choclates,
   Cookies,
   Pencil}             1004
5 {Marker}             1005
6 {Marker,
   Pencil}             1006
7 {Choclates,
   Pencil}             1007

> itemFrequencyPlot(txn)
```



Exécuter l'algorithme apriori

```
> basket_rules <- apriori(txn, parameter = list(sup = 0.5,
+     conf = 0.9, target = "rules"))

parameter specification:
 confidence minval smax arem  aval originalSupport support minlen
        0.9    0.1    1 none FALSE            TRUE     0.5      1
 maxlen target   ext
     10  rules FALSE

algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

apriori - find association rules with the apriori algorithm
version 4.21 (2004.05.09)        (c) 1996-2004   Christian Borgelt
set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[6 item(s), 7 transaction(s)] done [0.00s].
sorting and recoding items ... [2 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
```

```
checking subsets of size 1 2 done [0.00s].
writing ... [1 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

Comme nous le constatons à l'aide de la sortie de la fonction `apriori`, le nombre de règles générées est égal à 1 pour un support de 50% et une confiance de 90%. La (ou plus généralement les) règles obtenues peuvent être étudiées avec la commande `inspect(basket_rules)` :

```
> inspect(basket_rules)

  lhs            rhs        support confidence     lift
1 {Choclates} => {Pencil} 0.5714286          1 1.166667
```

Si beaucoup de règles sont générées, certaines peuvent être sélectionnées indivuellement.

```
> inspect(basket_rules[1])

  lhs            rhs        support confidence     lift
1 {Choclates} => {Pencil} 0.5714286          1 1.166667
```

La règle ci-dessous signifie que si du chocolat est acheté alors il est vraisemblable à 90% qu'un stylo le sera aussi. Le support de 57% indique que 57% des achats dans le jeu de données impliquent l'achat de chocolat.

```
  lhs            rhs        support confidence     lift
1 {Choclates} => {Pencil} 0.5714286          1 1.166667
```

Points forts de la bibliothèque La bibliothèque `arules` fournit le cadre qui permet d'extraire des associations, les analyser puis organiser les résultats. Les caractéristiques principales de cette bibliothèque sont :

- Une implémentation efficace à l'aide de matrices "sparse" ;
- Une interace simple et intuitive pour manipuler et analyser les transactions, les ensembles d'items et les règles d'association ;
- Deux algorithmes d'extraction de connaissance rapides ;
- Une grande flexibilité puisqu'il est possible d'ajouter des mesures de la pertinence des règles ; des descriptions supplémentaires pour les items et les transactions dont il est possible de se servir pour sélectionner et analyser les associations ainsi construites ;
- Une structure de données extensible pour permettre une implémentation simple de nouveaux types d'associations et interfacer de nouveaux algorithmes.

2 Exemple 1 : Préparer puis analyser un jeu de données de transactions

Dans cet exemple, nous montrons comment un jeu de données peut être analysé et manipulé avant d'en extraire des règles d'association.

Cette étape est importante car elle permet de déceler d'éventuels problèmes dans le jeu de données qui pourraient rendre les associations trouvées inutiles ou au minimum inférieures à celles qui auraient été extraites d'un jeu de données bien préparé.

Comme exemple, nous nous intéresserons aux données de transactions de Epub qui est inclus dans la bibliothèque `arules`. Ce jeu de données contient les téléchargements des documents de la Electronic Publication platform de la Vienna University of Economics and Business accessibles depuis l'adresse internet `http://epub.wu-wien.ac.at`, de Janvier 2003 à Décembre 2008.

Commençons par charger la bibliothèque et le jeu de données.

```
> library("arules")
> data("Epub")
> Epub

transactions in sparse format with
 15729 transactions (rows) and
 936 items (columns)
```

Nous constatons que le jeu de données comporte 15729 transactions et est représenté par une matrice "sparse" de 15729 lignes et de 936 colonnes qui représentent les items. Ensuite, nous utilisons la fonction `summary` pour obtenir plus de détails sur le jeu de données.

```
> summary(Epub)

transactions as itemMatrix in sparse format with
 15729 rows (elements/itemsets/transactions) and
 936 columns (items) and a density of 0.001758755

most frequent items:
doc_11d doc_813 doc_4c6 doc_955 doc_698 (Other)
    356     329     288     282     245   24393

element (itemset/transaction) length distribution:
sizes
    1     2     3     4     5     6     7     8     9    10    11
11615  2189   854   409   198   121    93    50    42    34    26
   12    13    14    15    16    17    18    19    20    21    22
```

```
   12    10    10     6     8     6     5     8     2     2     3
   23    24    25    26    27    28    30    34    36    38    41
    2     3     4     5     1     1     1     2     1     2     1
   43    52    58
    1     1     1

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.000   1.000   1.646   2.000  58.000

includes extended item information - examples:
   labels
1 doc_11d
2 doc_13d
3 doc_14c

includes extended transaction information - examples:
      transactionID           TimeStamp
10792  session_4795 2003-01-02 02:59:00
10793  session_4797 2003-01-02 13:46:01
10794  session_479a 2003-01-02 16:50:38
```

`summary()` affiche les items apparaissant le plus fréquemment dans le jeu de données, des informations sur la distribution des tailles des transactions et que le jeu de données contient des informations supplémentaires sur les différentes transactions. Nous constatons que le jeu de données contient non seulement les IDs des transactions mais aussi des indications temporelles (utilisant la classe POSIXct) sur les moments où ces transactions ont été réalisées. Cette information peut être utilisée pour analyser le jeu de données.

```
> year <- strftime(as.POSIXlt(transactionInfo(Epub)[["TimeStamp"]]),
+     "%Y")
> table(year)

year
2003 2004 2005 2006 2007 2008 2009
 987 1375 1611 3012 4053 4690    1
```

En 2003, pour la première année représentée dans le jeu de données, il y a 987 transactions. Nous pouvons sélectionner les transactions correspondantes et inspecter leur structure à l'aide d'un level-plot (voir Figure 1).

```
> Epub2003 <- Epub[year == "2003"]
> length(Epub2003)

[1] 987

> image(Epub2003)
```

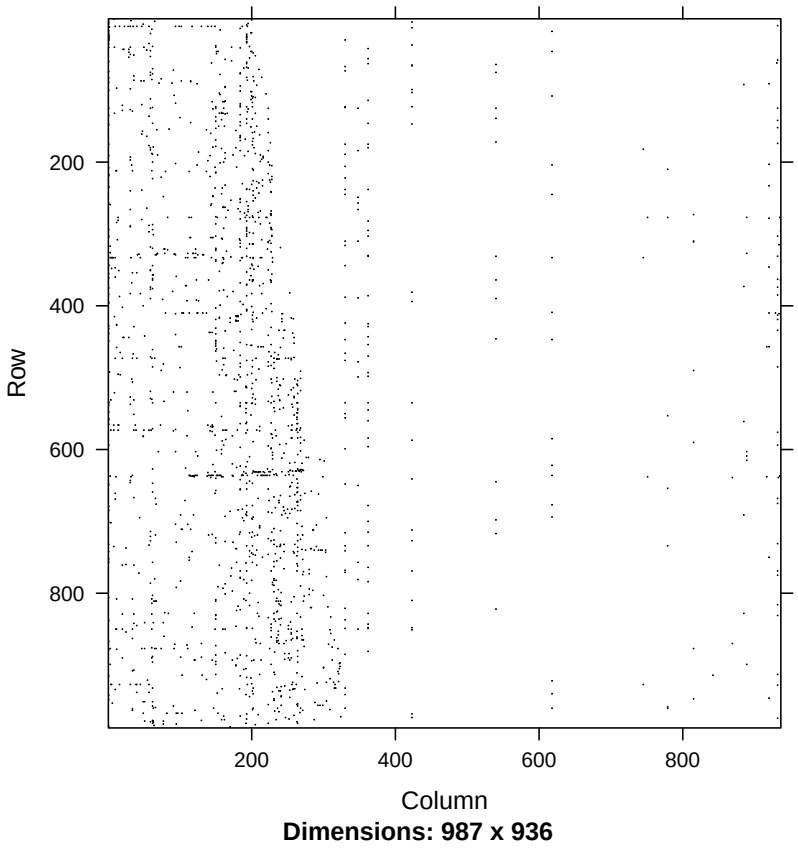


FIGURE 1 – Le jeu de données Epub (année 2003).

Ce graphique est une visualisation directe de la matrice binaire d'incidence où les points noirs représentent les 1 dans la matrices (c'est-à-dire les présences).
À partir de ce graphique nous voyons que les items dans le jeu de données ne sont répartis de manières équilibrée. En fait, la région majoritairement blanche située du côté droit du graphique suggère qu'au début de l'année 2003 suels quelques items étaient disponibles (moins de 50) et que durant l'année plus d'items ont été ajoutés jusqu'à un nombre total d'environ 300. Nous pouvons également constater que certaines des transactions dans le jeu de données contiennent un très grand nombre d'items (lignes horizontales les plus denses). Ces transactions nécessitent un questionnement spécifique puisqu'elles pourraient être dues à un problème de recueil de données (par exemple, un robot web téléchargeant systématiquement des documents depuis le site internet). Pour trouver les transactions dépassant une certaine longueur nous pouvons utiliser la fonction `size` et sélectionner ainsi les très longues transactions (contenant plus de 20 items).

```
> transactionInfo(Epub2003[size(Epub2003) > 20])

      transactionID            TimeStamp
```

```
11092  session_56e2 2003-04-29 19:30:38
11371  session_6308 2003-08-18 00:16:12
```

Nous trouvons trois longues transactions et imprimons les informations correspondantes. Naturellement, la longueur peut être utilisée d'une manière similaire pour supprimer les transactions trop longues ou trop courtes. Les transactions peuvent être étudiées à l'aide de la fonction `inspect`. Comme l'affichage des longues transactions identifiées ci-dessus résulterait par une sortie d'une très grande taille, nous inspecterons seulement les 5 premières transactions de l'année 2003.

```
> inspect(Epub2003[1:5])

  items     transactionID           TimeStamp
1 {doc_154}  session_4795 2003-01-02 02:59:00
2 {doc_3d6}  session_4797 2003-01-02 13:46:01
3 {doc_16f}  session_479a 2003-01-02 16:50:38
4 {doc_11d,
   doc_1a7,
   doc_f4}   session_47b7 2003-01-03 00:55:50
5 {doc_83}   session_47bb 2003-01-03 03:27:44
```

La plupart des transactions contiennent seulement un item. Seule la transaction 4 contient trois items. Pour un examen plus détaillé, les transactions peuvent être converties en une liste de la manière suivante :

```
> as(Epub2003[1:5], "list")

$session_4795
[1] "doc_154"

$session_4797
[1] "doc_3d6"

$session_479a
[1] "doc_16f"

$session_47b7
[1] "doc_11d" "doc_1a7" "doc_f4"

$session_47bb
[1] "doc_83"
```

Enfin, des transactions stockées horizontalement peuvent être converties dans un format vertical.

```
> EpubTidLists <- as(Epub, "tidLists")
> EpubTidLists
```

```
tidLists in sparse format with
 936 items/itemsets (rows) and
 15729 transactions (columns)
```

Pour des raisons de performance, la ID list est aussi stockée dans une matrice "sparse". Pour la transformer en liste, il suffit d'utiliser la commande suivante :

```
> as(EpubTidLists[1:3], "list")

$doc_11d
  [1] "session_47b7"    "session_47c2"    "session_47d8"
  [4] "session_4855"    "session_488d"    "session_4898"
  [7] "session_489b"    "session_489c"    "session_48a1"
 [10] "session_4897"    "session_48a0"    "session_489d"
 [13] "session_48a5"    "session_489a"    "session_4896"
 [16] "session_48aa"    "session_48d0"    "session_49de"
 [19] "session_4b35"    "session_4bac"    "session_4c54"
 [22] "session_4c9a"    "session_4d8c"    "session_4de5"
 [25] "session_4e89"    "session_5071"    "session_5134"
 [28] "session_51e6"    "session_5227"    "session_522a"
 [31] "session_5265"    "session_52e0"    "session_52ea"
 [34] "session_53e1"    "session_5522"    "session_558a"
 [37] "session_558b"    "session_5714"    "session_5739"
 [40] "session_57c5"    "session_5813"    "session_5861"
 [43] "session_wu48452" "session_5955"    "session_595a"
 [46] "session_5aaa"    "session_5acd"    "session_5b5f"
 [49] "session_5bfc"    "session_5f3d"    "session_5f42"
 [52] "session_5f69"    "session_5fcf"    "session_6044"
 [55] "session_6053"    "session_6081"    "session_61b5"
 [58] "session_635b"    "session_64b4"    "session_64e4"
 [61] "session_65d2"    "session_67d1"    "session_6824"
 [64] "session_68c4"    "session_68f8"    "session_6b2c"
 [67] "session_6c95"    "session_6e19"    "session_6eab"
 [70] "session_6ff8"    "session_718e"    "session_71c1"
 [73] "session_72d6"    "session_7303"    "session_73d0"
 [76] "session_782d"    "session_7856"    "session_7864"
 [79] "session_7a9b"    "session_7b24"    "session_7bf9"
 [82] "session_7cf2"    "session_7d5d"    "session_7dae"
 [85] "session_819b"    "session_8329"    "session_834d"
 [88] "session_84d7"    "session_85b0"    "session_861b"
 [91] "session_867f"    "session_8688"    "session_86bb"
 [94] "session_86ee"    "session_8730"    "session_8764"
 [97] "session_87a9"    "session_880a"    "session_8853"
[100] "session_88b0"    "session_8986"    "session_8a08"
[103] "session_8a73"    "session_8a87"    "session_8aad"
[106] "session_8ae2"    "session_8db4"    "session_8e1f"
```

```
[109] "session_wu53a42" "session_8fad"    "session_8fd3"
[112] "session_9083"    "session_90d8"    "session_9128"
[115] "session_9145"    "session_916e"    "session_9170"
[118] "session_919e"    "session_91df"    "session_9226"
[121] "session_9333"    "session_9376"    "session_937e"
[124] "session_94d5"    "session_9539"    "session_9678"
[127] "session_96a0"    "session_9745"    "session_97b3"
[130] "session_985b"    "session_9873"    "session_9881"
[133] "session_9994"    "session_9a20"    "session_9a2f"
[136] "session_wu54edf" "session_9af9"    "session_9b69"
[139] "session_9ba4"    "session_9c27"    "session_9c99"
[142] "session_9ce8"    "session_9de3"    "session_9e8a"
[145] "session_9ebc"    "session_a051"    "session_a16e"
[148] "session_a19f"    "session_a229"    "session_a24a"
[151] "session_a328"    "session_a340"    "session_a3ab"
[154] "session_a3ee"    "session_a43a"    "session_a4b2"
[157] "session_a515"    "session_a528"    "session_a555"
[160] "session_a5bb"    "session_a62d"    "session_a77a"
[163] "session_ab9c"    "session_abe9"    "session_ac0e"
[166] "session_ad30"    "session_adc9"    "session_af06"
[169] "session_af4a"    "session_af8d"    "session_b0b7"
[172] "session_b391"    "session_b6d3"    "session_b807"
[175] "session_b8c7"    "session_b91f"    "session_bb0b"
[178] "session_bb8a"    "session_bc3d"    "session_bc40"
[181] "session_bceb"    "session_bea7"    "session_bf9f"
[184] "session_c359"    "session_c3c2"    "session_c442"
[187] "session_c62d"    "session_c6ba"    "session_c936"
[190] "session_ca81"    "session_cad3"    "session_cbd4"
[193] "session_cbe1"    "session_cd63"    "session_d14f"
[196] "session_d370"    "session_d69f"    "session_d815"
[199] "session_d82e"    "session_d849"    "session_d8b5"
[202] "session_da68"    "session_db51"    "session_db75"
[205] "session_dbcd"    "session_dde2"    "session_deac"
[208] "session_dfb7"    "session_dfe9"    "session_e00a"
[211] "session_e2ad"    "session_e3c7"    "session_e7d2"
[214] "session_e7e5"    "session_e7f2"    "session_ea38"
[217] "session_edbc"    "session_edf9"    "session_edfc"
[220] "session_f0be"    "session_f2d9"    "session_f2fe"
[223] "session_f39b"    "session_f5e9"    "session_f650"
[226] "session_f853"    "session_f989"    "session_fab1"
[229] "session_fcef"    "session_fd0e"    "session_fe49"
[232] "session_fe4f"    "session_ffa0"    "session_10057"
[235] "session_1019a"   "session_1028a"   "session_10499"
[238] "session_10513"   "session_105e3"   "session_10b03"
[241] "session_10b53"   "session_10c0c"   "session_10cb2"
```

```
[244] "session_10e4d"   "session_10e67"   "session_10e92"
[247] "session_10fbd"   "session_10fcc"   "session_114f1"
[250] "session_116fb"   "session_11822"   "session_1185e"
[253] "session_118d0"   "session_11b0d"   "session_12182"
[256] "session_121af"   "session_121ee"   "session_12405"
[259] "session_126db"   "session_12825"   "session_12896"
[262] "session_12a0b"   "session_12c7c"   "session_12e21"
[265] "session_1346d"   "session_13622"   "session_13886"
[268] "session_13d33"   "session_140bd"   "session_14428"
[271] "session_14b8a"   "session_14e58"   "session_14fdc"
[274] "session_1517f"   "session_151b2"   "session_15549"
[277] "session_155a9"   "session_1571b"   "session_15b18"
[280] "session_15b99"   "session_15d2c"   "session_15e0c"
[283] "session_15f75"   "session_15fbf"   "session_16621"
[286] "session_16691"   "session_16f0d"   "session_17027"
[289] "session_173fe"   "session_17eaf"   "session_17ecd"
[292] "session_180dd"   "session_18641"   "session_187ae"
[295] "session_18a0b"   "session_18b18"   "session_18db4"
[298] "session_19048"   "session_19051"   "session_19510"
[301] "session_19788"   "session_197ee"   "session_19c04"
[304] "session_19c7a"   "session_19f0c"   "session_1a557"
[307] "session_1ac3c"   "session_1b733"   "session_1b76a"
[310] "session_1b76b"   "session_1ba83"   "session_1c0a6"
[313] "session_1c11c"   "session_1c304"   "session_1c4c3"
[316] "session_1cea1"   "session_1cfb9"   "session_1db2a"
[319] "session_1db96"   "session_1dbea"   "session_1dc94"
[322] "session_1e361"   "session_1e36e"   "session_1e91e"
[325] "session_wu6bf8f" "session_1f3a8"   "session_1f56c"
[328] "session_1f61e"   "session_1f831"   "session_1fced"
[331] "session_1fd39"   "session_wu6c9e5" "session_20074"
[334] "session_2019f"   "session_201a1"   "session_209f9"
[337] "session_20e87"   "session_2105b"   "session_212a2"
[340] "session_2143b"   "session_wu6decf" "session_218ca"
[343] "session_21bea"   "session_21bfd"   "session_223e1"
[346] "session_2248d"   "session_22ae6"   "session_2324d"
[349] "session_23636"   "session_23912"   "session_23a70"
[352] "session_23b0d"   "session_23c17"   "session_240ea"
[355] "session_24256"   "session_24484"

$doc_13d
 [1] "session_4809"  "session_5dbc"  "session_8e0b"
 [4] "session_cf4b"  "session_d92a"  "session_102bb"
 [7] "session_10e9f" "session_11344" "session_11ca4"
[10] "session_12dc9" "session_155b5" "session_1b563"
[13] "session_1c411" "session_1f384" "session_22e97"
```

```
$doc_14c
 [1] "session_53fb"  "session_564b"  "session_5697"
 [4] "session_56e2"  "session_630b"  "session_6e80"
 [7] "session_6f7c"  "session_7c8a"  "session_8903"
[10] "session_890c"  "session_89d2"  "session_907e"
[13] "session_98b4"  "session_c268"  "session_c302"
[16] "session_cb86"  "session_d70a"  "session_d854"
[19] "session_e4c7"  "session_f220"  "session_fd57"
[22] "session_fe31"  "session_10278" "session_115b0"
[25] "session_11baa" "session_11e26" "session_12185"
[28] "session_1414b" "session_14dba" "session_14e47"
[31] "session_15738" "session_15a38" "session_16305"
[34] "session_17b35" "session_19af2" "session_1d074"
[37] "session_1fcc4" "session_2272e" "session_23a3e"
```

Sous cette forme, chaque item est associé à un vecteur qui contient toutes les transactions dans lesquels il apparaît. Certains algorithmes de fouille de données s'appliquent à ces structures `tidLists` de données verticles.

3 Exemple 2 : Préparer et miner un questionnaire

Nous allons préparer puis fouiller des données de questionnaire. Nous nous servirons du jeu de données `Adult` disponible sur le UCI machine learning repository ([1]) et contenu dans le package `arules`. Ce jeu de données est semblable au jeu de données de données de marketing utilisé dans le chapitre sur l'extraction de règles d'association du livre de [2]. Il a été recueilli par le bureau du recensement des États-Unis et comporte 48842 réponses à 14 questions comme l'age, la catégorie socio-professionnelle, le niveau et le type d'études, etc. Les réponses à ces questions étaient utilisées à l'origine pour prédire le niveau de rémunération de chacun des 48882 individus. Nous avons ajouté la variable niveau de rémunération avec deux catégories, un faible niveau de rémunération qui correspond à des revenus $\leqslant$ USD 50000 et un niveau de rémunération élevé qui correspond à des revenus $>$ USD 50000. Ce jeu de données complété est également inclus dans le package `arules` sous le nom `AdultUCI`.

```
> data("AdultUCI")
> dim(AdultUCI)

[1] 48842    15

> AdultUCI[1:2, ]

  age        workclass fnlwgt education education-num
1  39        State-gov  77516 Bachelors            13
```

```
2  50 Self-emp-not-inc  83311 Bachelors           13
      marital-status      occupation  relationship  race  sex
1      Never-married    Adm-clerical Not-in-family White Male
2 Married-civ-spouse Exec-managerial       Husband White Male
  capital-gain capital-loss hours-per-week native-country income
1         2174            0             40  United-States  small
2            0            0             13  United-States  small
```

`AdultUCI` contient un mélange d'attributs qualitatifs et d'attributs quantitatifs mesurables. Il requiert un peu de préparation avant de pouvoir être mis la forme d'un ensemble de transactions susceptibles d'être fouillé à la recherche d'associations. En premier lieu, nous supprimons les deux attributs `fnlwgt` et `education-num`. Le premier est un poids calculé par les créateur du jeu de donnée à partir de données de référence fournies par la Population Division du bureau du recensement des États-Unis. Le second attribut supprimé est simplement un codage numérique de l'attribut education qui est déjà intégré dans le jeu de données.

```
> AdultUCI[["fnlwgt"]] <- NULL
> AdultUCI[["education-num"]] <- NULL
```

Puis, nous allons transformons les quatre attributs quantitatifs restants (age, hours-per-week, capital-gain et capital-loss) en attributs qualitatifs ordinaux en créant des catégories adéquates. Nous divisons les variables age et hours-per-week en catégories à l'aide de notre connaissance sur de valeurs seuils usuelles pour l'âge (15, 25, 45, 65, 100) et les temps de travail par semaine (0, 25, 40, 60, 168). Pour les deux attributs associés au revenus liés au patrimoine, nous créons une catégorie None pour ceux qui n'ont ni perte, ni gain. Puis nous divisons ceux qui restent en deux groupes, Low et High, en fonction de la position de leurs pertes ou gains par rapport à la valeur médiane des pertes ou des gains.

```
> AdultUCI[["age"]] <- ordered(cut(AdultUCI[["age"]],
+     c(15, 25, 45, 65, 100)), labels = c("Young", "Middle-aged",
+     "Senior", "Old"))
> AdultUCI[["hours-per-week"]] <- ordered(cut(AdultUCI[["hours-per-week"]],
+     c(0, 25, 40, 60, 168)), labels = c("Part-time",
+     "Full-time", "Over-time", "Workaholic"))
> AdultUCI[["capital-gain"]] <- ordered(cut(AdultUCI[["capital-gain"]],
+     c(-Inf, 0, median(AdultUCI[["capital-gain"]][AdultUCI[["capital-gain"]] >
+         0]), Inf)), labels = c("None", "Low", "High"))
> AdultUCI[["capital-loss"]] <- ordered(cut(AdultUCI[["capital-loss"]],
+     c(-Inf, 0, median(AdultUCI[["capital-loss"]][AdultUCI[["capital-loss"]] >
+         0]), Inf)), labels = c("none", "low", "high"))
```

Les données peuvent être maintenant enregistrées sous la forme d'une matrice d'incidence en transformant le jeu de données en un ensemble de transactions.

```
> Adult <- as(AdultUCI, "transactions")
> Adult
```

```
transactions in sparse format with
 48842 transactions (rows) and
 115 items (columns)
```

Les 115 modalités qualitatives restantes ont été automatiquement recodées en 115 variables binaires. Pendant la transcription, les noms des modalités qualitatives ont été créés sous la forme <nom de la variable>=<nom de la catégorie >. Pour les enregistrements comportant des valeurs manquantes, toutes les variables binaires associées à des modalités renseignées par des valeurs manquantes prennent la valeur 0.

```
> summary(Adult)
```

```
transactions as itemMatrix in sparse format with
 48842 rows (elements/itemsets/transactions) and
 115 columns (items) and a density of 0.1089939

most frequent items:
           capital-loss=none            capital-gain=None
                       46560                        44807
native-country=United-States                   race=White
                       43832                        41762
           workclass=Private                      (Other)
                       33906                       401333

element (itemset/transaction) length distribution:
sizes
    9    10    11    12    13
   19   971  2067 15623 30162

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   9.00   12.00   13.00   12.53   13.00   13.00

includes extended item information - examples:
           labels variables      levels
1       age=Young       age       Young
2 age=Middle-aged       age Middle-aged
3      age=Senior       age      Senior

includes extended transaction information - examples:
  transactionID
1             1
2             2
3             3
```

Le résumé du jeu de transactions donne un aperçu rapide qui montre les items les plus fréquents, la distribution de la longueur des transactions et des informations

supplémentaires qui indiquent quelles sont les variables et les modalités qui ont été utilisées pour créer chacune des variables binaires. Dans le premier exemple, nous constatons que l'item avec l'étiquette `age=Middle-aged` a été engendré par la variable `age` et la modalité `middle-aged`.
Pour déterminer quels sont les items importants dans le jeu de données, nous pouvons utiliser le graphique `itemFrequencyPlot`. Pour reduire le nombre des items, nous représentons seulement ceux pour lesquels le support est supérieur à 10% (à l'aide du paramètre support). Afin de pouvoir lire plus facilement les étiquettes, nous réduisons la taille des étiquettes avec le paramètre `cex.names`. Ce graphique est représenté à la Figure 2.

```
> itemFrequencyPlot(Adult, support = 0.1, cex.names = 0.8)
```

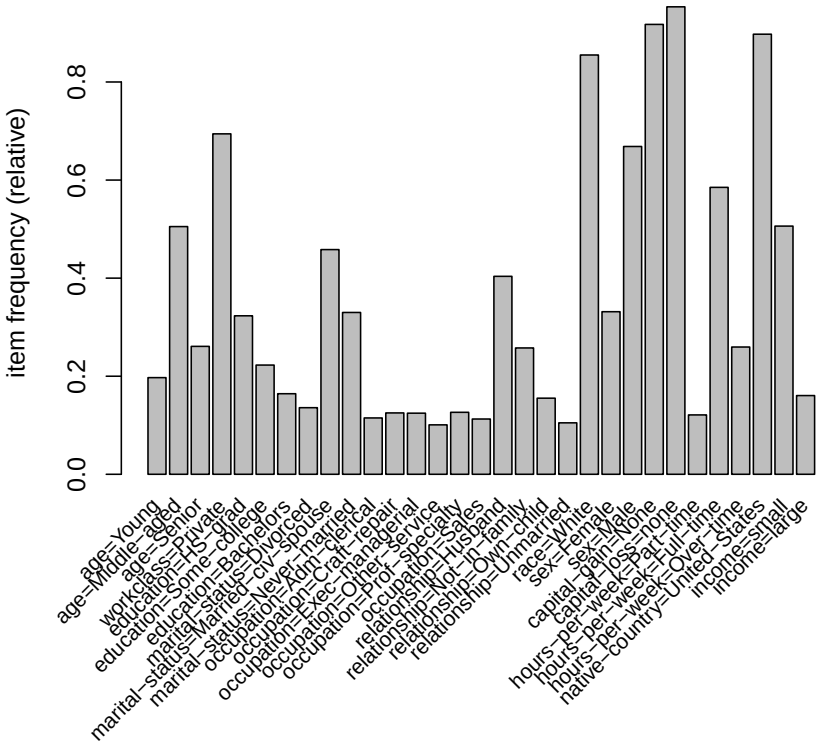


FIGURE 2 – Fréquences relatives des items dans le jeu de données `Adult` avec un support supérieur a 10%

Puis nous faisons appel à la fonction `apriori` pour trouver toutes les règles (le type d'association par défaut pour `apriori`) avec un support minimum de 1% et une confiance de 0.6.

```
> rules <- apriori(Adult, parameter = list(support = 0.01,
+     confidence = 0.6))

parameter specification:
 confidence minval smax arem  aval originalSupport support minlen
        0.6    0.1    1 none FALSE            TRUE    0.01      1
 maxlen target   ext
     10  rules FALSE

algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

apriori - find association rules with the apriori algorithm
version 4.21 (2004.05.09)        (c) 1996-2004   Christian Borgelt
set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[115 item(s), 48842 transaction(s)] done [0.05s].
sorting and recoding items ... [67 item(s)] done [0.00s].
creating transaction tree ... done [0.05s].
checking subsets of size 1 2 3 4 5 6 7 8 9 10 done [1.18s].
writing ... [276443 rule(s)] done [0.09s].
creating S4 object  ... done [0.17s].

> rules

set of 276443 rules
```

La fonction affiche en premier les paramètres utilisés. Sauf pour les valeurs du support minimum et de la confiance, tous les paramètres ont été utilisés avec leurs valeurs par défaut. Il est important de remarquer que le paramètre `maxlen`, la longueur maximale des règles d'association recherchées est par défaut égale à 5. Des associations plus longues ne seront donc recherchées que si `maxlen` est fixé à une valeur plus élévée.
Après l'affichage des paramètres de configuration de la fonction `apriori`, la sortie de l'implémenation en langage C de l'algorithme avec le temps qui a été utilisé pour l'exécution de celui-ci sont affichés. Le résultat de l'algorithme d'extraction est un ensemble de 276443 règles. La fonction `summary` peut être utilisée pour obtenir un aperçu des règles ainsi extraites. Elle montre le nombre de règles, les items les plus fréquents présents dans les membres de gauche (lhs) et dans les membres de droite (rhs), ainsi que la distribution de la longueur des règles, des résumés statistiques des mesures de qualité renvoyés par l'algorithme d'extraction.

```
> summary(rules)

set of 276443 rules
```

```
rule length distribution (lhs + rhs):sizes
    1     2     3     4     5     6     7     8     9    10
    6   432  4981 22127 52669 75104 67198 38094 13244  2588

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   5.000   6.000   6.289   7.000  10.000

summary of quality measures:
    support          confidence          lift
 Min.   :0.01001   Min.   :0.6000   Min.   : 0.7171
 1st Qu.:0.01253   1st Qu.:0.7691   1st Qu.: 1.0100
 Median :0.01701   Median :0.9051   Median : 1.0554
 Mean   :0.02679   Mean   :0.8600   Mean   : 1.3109
 3rd Qu.:0.02741   3rd Qu.:0.9542   3rd Qu.: 1.2980
 Max.   :0.95328   Max.   :1.0000   Max.   :20.6826

mining info:
  data ntransactions support confidence
 Adult         48842    0.01        0.6
```

Comme souvent lorsque nous cherchons des règles d'association, le nombre de règles extraites est considérablement élevé. Pour analyser ces règles, il est par exemple possible d'utiliser la fonction `subset` pour obtenir des sous-ensembles séparés de règles pour chacune des modalités qui provenait de la variable `income` lorqu'elle apparaît dans le membre de droite de la règle. Nous demandons de plus que la mesure du lift soit supérieure à 1,2.

```
> rulesIncomeSmall <- subset(rules, subset = rhs %in%
+     "income=small" & lift > 1.2)
> rulesIncomeLarge <- subset(rules, subset = rhs %in%
+     "income=large" & lift > 1.2)
```

Nous avons maintenant un ensemble de règles pour les individus ayant un faible revenu et un autre ensemble de règles pour ceux avec un revenu élevé. Nous comparons maintenant les trois règles avec la confiance la plus forte pour ces deux ensembles de règles en utilisant la fonction `sort`.

```
> inspect(head(sort(rulesIncomeSmall, by = "confidence"),
+     n = 3))
```

```
  lhs                                rhs                support confidence     lift
1 {workclass=Private,
   marital-status=Never-married,
   relationship=Own-child,
   sex=Male,
   hours-per-week=Part-time,
   native-country=United-States} => {income=small} 0.01074895  0.7104195 1.403653
2 {workclass=Private,
```

```
   marital-status=Never-married,
   relationship=Own-child,
   sex=Male,
   hours-per-week=Part-time}     => {income=small} 0.01144507  0.7102922 1.403402
3 {workclass=Private,
   marital-status=Never-married,
   relationship=Own-child,
   sex=Male,
   capital-gain=None,
   hours-per-week=Part-time,
   native-country=United-States} => {income=small} 0.01046231  0.7097222 1.402276
```

```
> inspect(head(sort(rulesIncomeLarge, by = "confidence"),
+     n = 3))
```

```
  lhs                                    rhs                support confidence     lift
1 {marital-status=Married-civ-spouse,
   capital-gain=High,
   native-country=United-States}      => {income=large} 0.01562180  0.6849192 4.266398
2 {marital-status=Married-civ-spouse,
   capital-gain=High,
   capital-loss=none,
   native-country=United-States}      => {income=large} 0.01562180  0.6849192 4.266398
3 {relationship=Husband,
   race=White,
   capital-gain=High,
   native-country=United-States}      => {income=large} 0.01302158  0.6846071 4.264454
```

De ces règles nous déduisons que les personnes travaillant dans le secteur privé à temps partiel ou dans les services ont tendance à avoir des revenus faibles tandis que ceux avec un patrimoine important qui sont nés aux États-Unis ont tendance à avoir des revenus élevés. Cet exemple montre qu'en utilisant les fonctions `subset` et `sort`, il est possible d'analyser un ensemble de règles d'association même si le nombre de celle-ci est très élevé.
Enfin, les règles que nous avons trouvées peuvent être enregistrées afin d'être partagées avec d'autres applications. La fontion `WRITE` permet de sauver ces règles au format texte. La fonction suivante enregistre un ensemble de règles dans le fichier `data.csv` au format CSV (comma separated value.

```
> WRITE(rulesIncomeSmall, file = "data.csv", sep = ",",
+     col.names = NA)
```

De manière alternative, avec le package `pmml` [3]) les règles peuvent être enregistrées au format PMML (Predictive Modelling Markup Language), une représentation standardisée basée sur le format XML et utilisée par de nombreux outils de data mining. Il faut noter que le package `pmml` a besoin du package `XML` pour fonctionner et que celui-ci peut ne pas être disponible pour tous les systèmes d'exploitation.

```
> install.packages("pmml")
> library("pmml")
> rules_pmml <- pmml(rulesIncomeSmall)
> saveXML(rules_pmml, file = "data.xml")
```

```
[1] "data.xml"
```

Les données ainsi sauvées peuvent maintenant être facilement partagées et utilisées par d'autres applications. Les Itemsets (avec `WRITE` de même que les transactions) peuvent être sauvegardés dans un fichier de la même manière.

Table des matières

Références

[1] Asuncion A, Newman DJ (2007). *UCI Repository of Machine Learning Databases*. University of California, Irvine, Deptartment of Information and Computer Sciences. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

[2] Hastie T, Tibshirani R, Friedman J (2001). *The Elements of Statistical Learning (Data Mining, Inference and Prediction)*. Springer Verlag.

[3] Williams G (2008). *pmml : Generate PMML for various models*. R package version 1.1.7.