

PLATFORM USED:

Programming Language : R

Tool Used for Database: Microsoft Excel 2010

Operating System: Windows

Hardware: Laptop

List of some common supermarket items: Filename – **Basket_set1**

itemDescription
milk
bread
beer
apple
diaper
yogurt
eggs
chocolate
juice
fruits

Database1: Filename- Transaction1

	itemList					
1	milk	bread	beer	apple	diaper	yogurt
2	milk	bread	beer	yogurt	eggs	apple
3	apple	yogurt	beer	chocolate		
4	chocolate	juice	yogurt	fruits		
5	milk	bread	chocolate	beer	yogurt	
6	milk	eggs	bread	beer	fruits	
7	chocolate	eggs	yogurt			
8	milk	chocolate	eggs	fruits		
9	beer	apple	chocolate	fruits		
10	yogurt	milk	fruits	beer		
11	yogurt	juice	bread	milk		
12	beer	fruits	milk	chocolate	juice	

13	milk	bread	beer	diaper		
14	diaper	milk	eggs	apples	chocolate	
15	yogurt	milk	bread	apple	fruits	
16	apple	diaper	chocolate	beer		
17	milk	juice	beer			
18	fruits	juice	eggs	milk	apples	
19	milk	bread	eggs	fruits		
20	milk	eggs	fruits	beer	apple	

```

df_basketset <- read.csv("Basket_set1.csv")

if(sessionInfo()['basePkgs']=="dplyr" | sessionInfo()['otherPkgs']=="dplyr"){
  detach(package:dplyr, unload=TRUE)
}

library(plyr)

colnames(df_itemList) <- c("itemList")

write.csv(df_itemList,"Transaction1.csv", row.names = TRUE, quote = FALSE)

library(arules)

txn1 = read.transactions(file="Transaction1.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)

txn1@itemInfo$labels <- gsub("\\"", "", txn1@itemInfo$labels)

s <- readline(prompt="Enter Support: ")

c <- readline(prompt="Enter confidence: ")

s <- as.numeric(unlist(strsplit(s, ",")))

c <- as.numeric(unlist(strsplit(c, ",")))

```

The screenshot shows the RStudio interface with several windows open:

- Code Editor:** The main window displays R code for reading a CSV file and creating a data frame. A red dot highlights line 21: `s <- as.numeric(unlist(strsplit(s, ",")))`. The console output below shows the execution of the script.
- Console:** Shows the execution of the R script, including the creation of a data frame named `basket1` and its inspection.
- Environment:** Shows the global environment with objects like `df_basket1`, `c`, `df_itemList`, `s`, and `txnl`.
- Help:** A search result for "R.Pattern nested data.frames" is shown, with a link to the "Flatten nested data frames" example.
- Output:** Displays the "Flatten nested data frames" example, which shows how to flatten a nested data frame structure.

Result for Database1 for frequent -2 itemsets

```
basket1<-apriori(txn1,parameter = list(supp = s, conf = c,maxlen=3,minlen=3))
```

```
if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){

  detach(package:tm, unload=TRUE)

}
```

```
inspect(basket1)
```

```
df_basket1 <- as(basket1,"data.frame")
```

```
View(df_basket1)
```

The screenshot shows the RStudio interface with the following details:

- File menu:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Script pane:** Shows the R code for generating frequent itemsets. It includes reading a CSV file, handling dependencies, and running the apriori function.
- Environment pane:** Shows the global environment with objects like df_basket, df_basket1, df_basketset, df_groceries, df_itemlist, df_sorted, basket1, basketo, txn, and txn1.
- Plots pane:** Empty.
- Packages pane:** Shows installed packages: acepack, arules, assertthat, backports, base64enc, BH, checkmate, colorspace, data.table, DBI, dichromat, digest, evaluate, Formula, ggridges, gridExtra, gtable, and hms.
- Help pane:** Empty.
- Viewer pane:** Shows the output of the R code, including the command-line arguments, the apriori call, and the resulting frequent itemsets table.

The screenshot shows an RStudio interface with multiple windows open. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help, and Admins. The left sidebar has tabs for Transaction.R, df_basket1, and kaita.R*. The main workspace displays a data frame named 'rules' with columns: rules, lhs, support, confidence, lift, and count. The rules table contains 19 rows of association rules, such as 'apples,eggs => (milk)' with support 0.0952381 and confidence 1.0000000. The bottom-left pane shows the console output of the apriori function, which prints the rule set and stops at the 10th rule due to memory constraints. The bottom-right pane shows the Global Environment, listing objects like df_basket, df_basket1, df_basket2, df_groceries, df_itemlist, df_sorted, and txn, each with their respective class and size.

lhs	rhs	support	confidence	lift	count
[1] {apples, eggs}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[2] {apples, milk}	=> {eggs}	0.0952381	1.0000000	2.6250000	2
[3] {eggs, milk}	=> {apples}	0.0952381	0.2857143	3.0000000	2
[4] {apple, diaper}	=> {beer}	0.0952381	1.0000000	1.7500000	2
[5] {beer, diaper}	=> {apple}	0.0952381	0.6666667	2.0000000	2
[6] {apple, beer}	=> {diaper}	0.0952381	0.3333333	1.7500000	2
[7] {bread, diaper}	=> {beer}	0.0952381	1.0000000	1.7500000	2
[8] {beer, diaper}	=> {bread}	0.0952381	0.6666667	1.7500000	2
[9] {beer, bread}	=> {diaper}	0.0952381	0.4000000	2.1000000	2
[10] {bread, diaper}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[11] {diaper, milk}	=> {bread}	0.0952381	0.6666667	1.7500000	2
[12] {bread, milk}	=> {diaper}	0.0952381	0.2500000	1.3125000	2
[13] {beer, diaper}	=> {milk}	0.0952381	0.6666667	0.9333333	2
[14] {diaper, milk}	=> {beer}	0.0952381	0.6666667	1.1666667	2
[15] {chocolate, juice}	=> {fruits}	0.0952381	1.0000000	2.1000000	2
[16] {fruits, juice}	=> {chocolate}	0.0952381	0.6666667	1.5555556	2
[17] {chocolate, fruits}	=> {juice}	0.0952381	0.5000000	2.1000000	2
[18] {fruits, juice}	=> {milk}	0.0952381	0.6666667	0.9333333	2
[19] {juice, milk}	=> {fruits}	0.0952381	0.5000000	1.0500000	2
[20] {fruits, milk}	=> {juice}	0.0952381	0.2500000	1.0500000	2
[21] {beer, juice}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[22] {juice, milk}	=> {beer}	0.0952381	0.5000000	0.8750000	2
[23] {apple, eggs}	=> {beer}	0.0952381	1.0000000	1.7500000	2
[24] {apple, beer}	=> {eggs}	0.0952381	0.3333333	0.8750000	2
[25] {beer, eggs}	=> {apple}	0.0952381	0.6666667	2.0000000	2
[26] {apple, eggs}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[27] {apple, milk}	=> {eggs}	0.0952381	0.5000000	1.3125000	2
[28] {eggs, milk}	=> {apple}	0.0952381	0.2857143	0.8571429	2
[29] {apple, chocolate}	=> {beer}	0.1428571	1.0000000	1.7500000	3
[30] {apple, beer}	=> {chocolate}	0.1428571	0.5000000	1.1666667	3
[31] {beer, chocolate}	=> {apple}	0.1428571	0.6000000	1.8000000	3
[32] {apple, bread}	=> {yogurt}	0.1428571	1.0000000	2.3333333	3

[33]	{apple,yogurt}	=> {bread}	0.1428571	0.7500000	1.9687500	3
[34]	{bread,yogurt}	=> {apple}	0.1428571	0.6000000	1.8000000	3
[35]	{apple,bread}	=> {beer}	0.0952381	0.6666667	1.1666667	2
[36]	{apple,beer}	=> {bread}	0.0952381	0.3333333	0.8750000	2
[37]	{beer,bread}	=> {apple}	0.0952381	0.4000000	1.2000000	2
[38]	{apple,bread}	=> {milk}	0.1428571	1.0000000	1.4000000	3
[39]	{apple,milk}	=> {bread}	0.1428571	0.7500000	1.9687500	3
[40]	{bread,milk}	=> {apple}	0.1428571	0.3750000	1.1250000	3
[41]	{apple,yogurt}	=> {beer}	0.1428571	0.7500000	1.3125000	3
[42]	{apple,beer}	=> {yogurt}	0.1428571	0.5000000	1.1666667	3
[43]	{beer,yogurt}	=> {apple}	0.1428571	0.6000000	1.8000000	3
[44]	{apple,yogurt}	=> {milk}	0.1428571	0.7500000	1.0500000	3
[45]	{apple,milk}	=> {yogurt}	0.1428571	0.7500000	1.7500000	3
[46]	{milk,yogurt}	=> {apple}	0.1428571	0.5000000	1.5000000	3
[47]	{apple,fruits}	=> {beer}	0.0952381	0.6666667	1.1666667	2
[48]	{apple,beer}	=> {fruits}	0.0952381	0.3333333	0.7000000	2
[49]	{beer,fruits}	=> {apple}	0.0952381	0.4000000	1.2000000	2
[50]	{apple,fruits}	=> {milk}	0.0952381	0.6666667	0.9333333	2
[51]	{apple,milk}	=> {fruits}	0.0952381	0.5000000	1.0500000	2
[52]	{fruits,milk}	=> {apple}	0.0952381	0.2500000	0.7500000	2
[53]	{apple,beer}	=> {milk}	0.1428571	0.5000000	0.7000000	3
[54]	{apple,milk}	=> {beer}	0.1428571	0.7500000	1.3125000	3
[55]	{beer,milk}	=> {apple}	0.1428571	0.3333333	1.0000000	3
[56]	{chocolate,eggs}	=> {milk}	0.0952381	0.6666667	0.9333333	2
[57]	{eggs,milk}	=> {chocolate}	0.0952381	0.2857143	0.6666667	2
[58]	{chocolate,milk}	=> {eggs}	0.0952381	0.5000000	1.3125000	2
[59]	{bread,eggs}	=> {fruits}	0.0952381	0.6666667	1.4000000	2
[60]	{eggs,fruits}	=> {bread}	0.0952381	0.4000000	1.0500000	2
[61]	{bread,fruits}	=> {eggs}	0.0952381	0.6666667	1.7500000	2
[62]	{bread,eggs}	=> {beer}	0.0952381	0.6666667	1.1666667	2
[63]	{beer,eggs}	=> {bread}	0.0952381	0.6666667	1.7500000	2
[64]	{beer,bread}	=> {eggs}	0.0952381	0.4000000	1.0500000	2
[65]	{bread,eggs}	=> {milk}	0.1428571	1.0000000	1.4000000	3
[66]	{eggs,milk}	=> {bread}	0.1428571	0.4285714	1.1250000	3
[67]	{bread,milk}	=> {eggs}	0.1428571	0.3750000	0.9843750	3
[68]	{eggs,fruits}	=> {beer}	0.0952381	0.4000000	0.7000000	2
[69]	{beer,eggs}	=> {fruits}	0.0952381	0.6666667	1.4000000	2
[70]	{beer,fruits}	=> {eggs}	0.0952381	0.4000000	1.0500000	2
[71]	{eggs,fruits}	=> {milk}	0.2380952	1.0000000	1.4000000	5
[72]	{eggs,milk}	=> {fruits}	0.2380952	0.7142857	1.5000000	5
[73]	{fruits,milk}	=> {eggs}	0.2380952	0.6250000	1.6406250	5
[74]	{beer,eggs}	=> {milk}	0.1428571	1.0000000	1.4000000	3
[75]	{eggs,milk}	=> {beer}	0.1428571	0.4285714	0.7500000	3
[76]	{beer,milk}	=> {eggs}	0.1428571	0.3333333	0.8750000	3
[77]	{chocolate,yogurt}	=> {beer}	0.0952381	0.5000000	0.8750000	2
[78]	{beer,chocolate}	=> {yogurt}	0.0952381	0.4000000	0.9333333	2
[79]	{beer,yogurt}	=> {chocolate}	0.0952381	0.4000000	0.9333333	2
[80]	{chocolate,fruits}	=> {beer}	0.0952381	0.5000000	0.8750000	2
[81]	{beer,chocolate}	=> {fruits}	0.0952381	0.4000000	0.8400000	2
[82]	{beer,fruits}	=> {chocolate}	0.0952381	0.4000000	0.9333333	2
[83]	{chocolate,fruits}	=> {milk}	0.0952381	0.5000000	0.7000000	2
[84]	{chocolate,milk}	=> {fruits}	0.0952381	0.5000000	1.0500000	2
[85]	{fruits,milk}	=> {chocolate}	0.0952381	0.2500000	0.5833333	2
[86]	{beer,chocolate}	=> {milk}	0.0952381	0.4000000	0.5600000	2
[87]	{chocolate,milk}	=> {beer}	0.0952381	0.5000000	0.8750000	2
[88]	{bread,yogurt}	=> {beer}	0.1428571	0.6000000	1.0500000	3
[89]	{beer,bread}	=> {yogurt}	0.1428571	0.6000000	1.4000000	3

[90]	{beer,yogurt}	=> {bread}	0.1428571	0.6000000	1.5750000	3
[91]	{bread,yogurt}	=> {milk}	0.2380952	1.0000000	1.4000000	5
[92]	{bread,milk}	=> {yogurt}	0.2380952	0.6250000	1.4583333	5
[93]	{milk,yogurt}	=> {bread}	0.2380952	0.8333333	2.1875000	5
[94]	{bread,fruits}	=> {milk}	0.1428571	1.0000000	1.4000000	3
[95]	{bread,milk}	=> {fruits}	0.1428571	0.3750000	0.7875000	3
[96]	{fruits,milk}	=> {bread}	0.1428571	0.3750000	0.9843750	3
[97]	{beer,bread}	=> {milk}	0.2380952	1.0000000	1.4000000	5
[98]	{bread,milk}	=> {beer}	0.2380952	0.6250000	1.0937500	5
[99]	{beer,milk}	=> {bread}	0.2380952	0.5555556	1.4583333	5
[100]	{fruits,yogurt}	=> {milk}	0.0952381	0.6666667	0.9333333	2
[101]	{milk,yogurt}	=> {fruits}	0.0952381	0.3333333	0.7000000	2
[102]	{fruits,milk}	=> {yogurt}	0.0952381	0.2500000	0.5833333	2
[103]	{beer,yogurt}	=> {milk}	0.1904762	0.8000000	1.1200000	4
[104]	{milk,yogurt}	=> {beer}	0.1904762	0.6666667	1.1666667	4
[105]	{beer,milk}	=> {yogurt}	0.1904762	0.4444444	1.0370370	4
[106]	{beer,fruits}	=> {milk}	0.1904762	0.8000000	1.1200000	4
[107]	{fruits,milk}	=> {beer}	0.1904762	0.5000000	0.8750000	4
[108]	{beer,milk}	=> {fruits}	0.1904762	0.4444444	0.9333333	4

Result for Database1 for frequent -3 itemsets

```
basket1<-apriori(txn1,parameter = list(supp = s, conf = c,minlen=4,maxlen=4))
```

```
if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){

  detach(package:tm, unload=TRUE)

}
```

```
inspect(basket1)
```

```
df_basket1 <- as(basket1,"data.frame")
```

```
View(df_basket1)
```

The screenshot shows the RStudio interface with the following details:

- File menu:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Source pane:** Transaction.R, df_basket1, kauits.R.
- Environment pane:** Shows objects like df_basket, df_basket1, df_basketset, df_groceries, df_itemlist, df_sorted, basket1, basket2, txn, and txn1.
- Values pane:** Shows formal class rules and transactions for basket1 and basket2.
- Files pane:** Shows installed packages: acepack, arules, assertthat, backports, base64enc, BH, checkmate, colorspace, data.table, DBI, dichromat, digest, dplyr, evaluate, Formula, ggplot2, gridExtra, gtable, and knitr.
- Plots pane:** Not visible.
- Packages pane:** Not visible.
- Help pane:** Not visible.
- Viewer pane:** Not visible.
- Console pane:** Displays the R script and its execution results. The results show the creation of the basket1 object from the df_basket1 data frame, and the output of inspect(basket1) which includes the frequent itemsets found in the dataset.
- System tray:** Shows the Windows taskbar with various icons and the date/time (10/21/2017, 1:28 PM).

RStudio Environment History

Data

- df_basket 710 obs. of 5 variables
- df_basket2 40 obs. of 5 variables
- df_basketset 10 obs. of 1 variable
- df_groceries 10 obs. of 1 variable
- df_itemslist 10 obs. of 1 variable
- df_sorted 10 obs. of 3 variables

Value

- basket1 Formal class rules
- basket2 Formal class rules
- txns Formal class transactions
- txnt Formal class transactions

Files Plots Packages Help Viewer

Console / - /

```

filter tree heap memopt load sort verbose
 0.1 TRUE TRUE FALSE TRUE 2 TRUE
Absolute minimum support count: 1

set item appearances ... [0 item(s)] done [0.00s].
set transactions... [21 item(s), 21 transaction(s)] done [0.00s].
sort transactions... [21 item(s), 21 transaction(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 & 4 done [0.00s].
writing ... [40 rule(s)] done [0.00s].
create 54 object ... done [0.00s].
warning message:
In apriori(..., parameter = list(supp = 0.05, conf = 0.25, minlen = 4, :
  minlen stopped (maxlen reached), only patterns up to a length of 4 returned!
> 
> if(sessionInfo()["basePkgs"]=="tcl" | sessionInfo()["otherPkgs"]=="cr")
+   detach(package:tcl, unload=TRUE)
+ )
> inspect(basket1)
  lhs          rhs      support  confidence    lift    count
[1] {beer,bread,diaper} => {milk} 0.0952381 1.0000000 1.4000000 2

```

12:46 PM 10/21/2017

	lhs	rhs	support	confidence	lift	count
[1]	{beer,bread,diaper}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[2]	{bread,diaper,milk}	=> {beer}	0.0952381	1.0000000	1.7500000	2
[3]	{beer,diaper,milk}	=> {bread}	0.0952381	1.0000000	2.6250000	2
[4]	{beer,bread,milk}	=> {diaper}	0.0952381	0.4000000	2.1000000	2
[5]	{apple,beer,eggs}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[6]	{apple,eggs,milk}	=> {beer}	0.0952381	1.0000000	1.7500000	2
[7]	{apple,beer,milk}	=> {eggs}	0.0952381	0.6666667	1.7500000	2
[8]	{beer,eggs,milk}	=> {apple}	0.0952381	0.6666667	2.0000000	2
[9]	{apple,bread,yogurt}	=> {beer}	0.0952381	0.6666667	1.1666667	2
[10]	{apple,beer,bread}	=> {yogurt}	0.0952381	1.0000000	2.3333333	2
[11]	{apple,beer,yogurt}	=> {bread}	0.0952381	0.6666667	1.7500000	2
[12]	{beer,bread,yogurt}	=> {apple}	0.0952381	0.6666667	2.0000000	2
[13]	{apple,bread,yogurt}	=> {milk}	0.1428571	1.0000000	1.4000000	3
[14]	{apple,bread,milk}	=> {yogurt}	0.1428571	1.0000000	2.3333333	3
[15]	{apple,milk,yogurt}	=> {bread}	0.1428571	1.0000000	2.6250000	3
[16]	{bread,milk,yogurt}	=> {apple}	0.1428571	0.6000000	1.8000000	3
[17]	{apple,beer,bread}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[18]	{apple,bread,milk}	=> {beer}	0.0952381	0.6666667	1.1666667	2
[19]	{apple,beer,milk}	=> {bread}	0.0952381	0.6666667	1.7500000	2
[20]	{beer,bread,milk}	=> {apple}	0.0952381	0.4000000	1.2000000	2
[21]	{apple,beer,yogurt}	=> {milk}	0.0952381	0.6666667	0.9333333	2
[22]	{apple,milk,yogurt}	=> {beer}	0.0952381	0.6666667	1.1666667	2
[23]	{apple,beer,milk}	=> {yogurt}	0.0952381	0.6666667	1.5555556	2
[24]	{beer,milk,yogurt}	=> {apple}	0.0952381	0.5000000	1.5000000	2
[25]	{bread,eggs,fruits}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[26]	{bread,eggs,milk}	=> {fruits}	0.0952381	0.6666667	1.4000000	2
[27]	{eggs,fruits,milk}	=> {bread}	0.0952381	0.4000000	1.0500000	2
[28]	{bread,fruits,milk}	=> {eggs}	0.0952381	0.6666667	1.7500000	2
[29]	{beer,bread,eggs}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[30]	{bread,eggs,milk}	=> {beer}	0.0952381	0.6666667	1.1666667	2
[31]	{beer,eggs,milk}	=> {bread}	0.0952381	0.6666667	1.7500000	2
[32]	{beer,bread,milk}	=> {eggs}	0.0952381	0.4000000	1.0500000	2

```
[33] {beer,eggs,fruits} => {milk} 0.0952381 1.0000000 1.4000000 2
[34] {eggs,fruits,milk} => {beer} 0.0952381 0.4000000 0.7000000 2
[35] {beer,eggs,milk} => {fruits} 0.0952381 0.6666667 1.4000000 2
[36] {beer,fruits,milk} => {eggs} 0.0952381 0.5000000 1.3125000 2
[37] {beer,bread,yogurt} => {milk} 0.1428571 1.0000000 1.4000000 3
[38] {bread,milk,yogurt} => {beer} 0.1428571 0.6000000 1.0500000 3
[39] {beer,bread,milk} => {yogurt} 0.1428571 0.6000000 1.4000000 3
[40] {beer,milk,yogurt} => {bread} 0.1428571 0.7500000 1.9687500 3
```

Result for Database1 for frequent -4 itemsets

```
basket1<-apriori(txn1,parameter = list(supp = s, conf = c,minlen=5,maxlen=5))
```

```
if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){

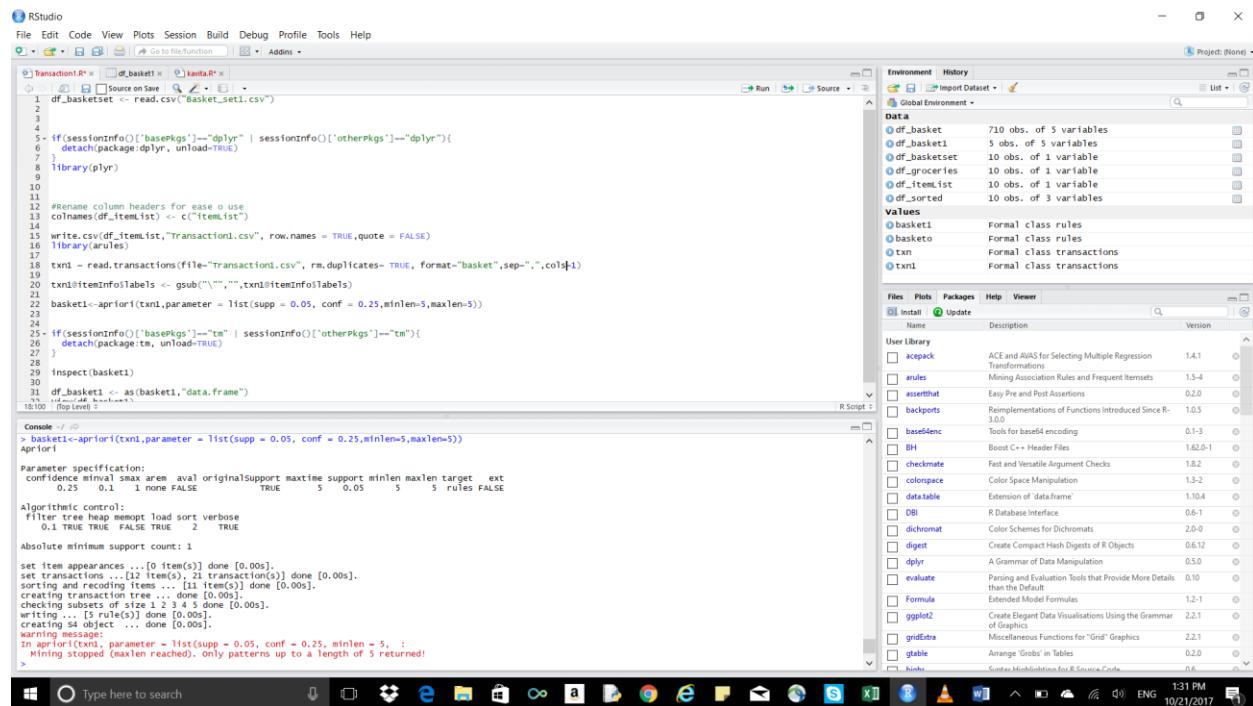
  detach(package:tm, unload=TRUE)

}
```

```
inspect(basket1)
```

```
df_basket1 <- as(basket1,"data.frame")
```

```
View(df_basket1)
```



The screenshot shows the RStudio interface with the following details:

- Environment pane:** Shows objects like df_basket, df_basket1, df_basketset, df_groceries, df_itemlist, df_sorted, basket1, basket2, txn, and txn1.
- Packages pane:** Shows installed packages including acepack, arules, assertthat, backports, base64enc, BH, checkmate, colorspace, data.table, DBI, dichromat, digest, dplyr, evaluate, Formula, ggplot2, gridExtra, gtable, and knitr.
- Code pane:** Displays the R script for generating frequent itemsets. It includes code for reading CSV files, setting parameters for the apriori function (minsup=0.05, minconf=0.25, maxlen=5), and inspecting the resulting basket object.
- Console pane:** Shows the output of the R script, including the parameter specification, algorithmic control, and a warning message about pattern length.
- System tray:** Shows standard Windows icons for file, copy, cut, paste, etc.
- Taskbar:** Shows various open applications including Microsoft Word, Excel, and a browser.

The screenshot shows the RStudio interface with the following details:

- Environment:**
 - df_basket: 710 obs. of 5 variables
 - df_basket1: 5 obs. of 5 variables
 - df_basketset: 10 obs. of 1 variable
 - df_groceries: 10 obs. of 1 variable
 - df_itemslist: 10 obs. of 1 variable
 - df_sorted: 10 obs. of 3 variables
- History:**
 - tx1: Formal class rules
 - tx2: Formal class rules
 - txn: Formal class transactions
 - tx1: Formal class transactions
- Console:**

```
> basket<-apriori(txn1,parameter = list(supp = 0.05, conf = 0.25,minlen=5,maxlen=5))
Apriori[1]

Parameter specification:
confidence minval maxval arm originalSupport maxtime support minlen maxlen target ext
0.25    0.1     1 none FALSE           TRUE      5   0.05      5      5 rules FALSE

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

Absolute minimum support count: 1

set Item.appearance ... [0 item(s)] done [0.00s].
set transactions ... [12 item(s), 21 transaction(s)] done [0.00s].
sorting and recoding items ... [11 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
check transaction tree ... done [0.00s].
writing ... [5 rule(s)] done [0.00s].
writing ... [5 rule(s)] done [0.00s].
creating s4 object ... done [0.00s].
warning: In apriori(txn1, parameter = list(supp = 0.05, conf = 0.25, minlen = 5, :
  'Mining stopped (maxlen reached); only patterns up to a length of 5 returned!
>
```

lhs	rhs	support	confidence	lift	count
[1] {apple,beer,bread,yogurt} => {milk}	0.0952381	1.0000000	1.400000	2	
[2] {apple,bread,milk,yogurt} => {beer}	0.0952381	0.6666667	1.166667	2	
[3] {apple,beer,bread,milk} => {yogurt}	0.0952381	1.0000000	2.333333	2	
[4] {apple,beer,milk,yogurt} => {bread}	0.0952381	1.0000000	2.625000	2	
[5] {beer,bread,milk,yogurt} => {apple}	0.0952381	0.6666667	2.000000	2	

Result for Database1 for frequent -5 itemsets

```
basket1<-apriori(txn1,parameter = list(supp = s, conf = c,minlen=6,maxlen=6))
```

```
if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){

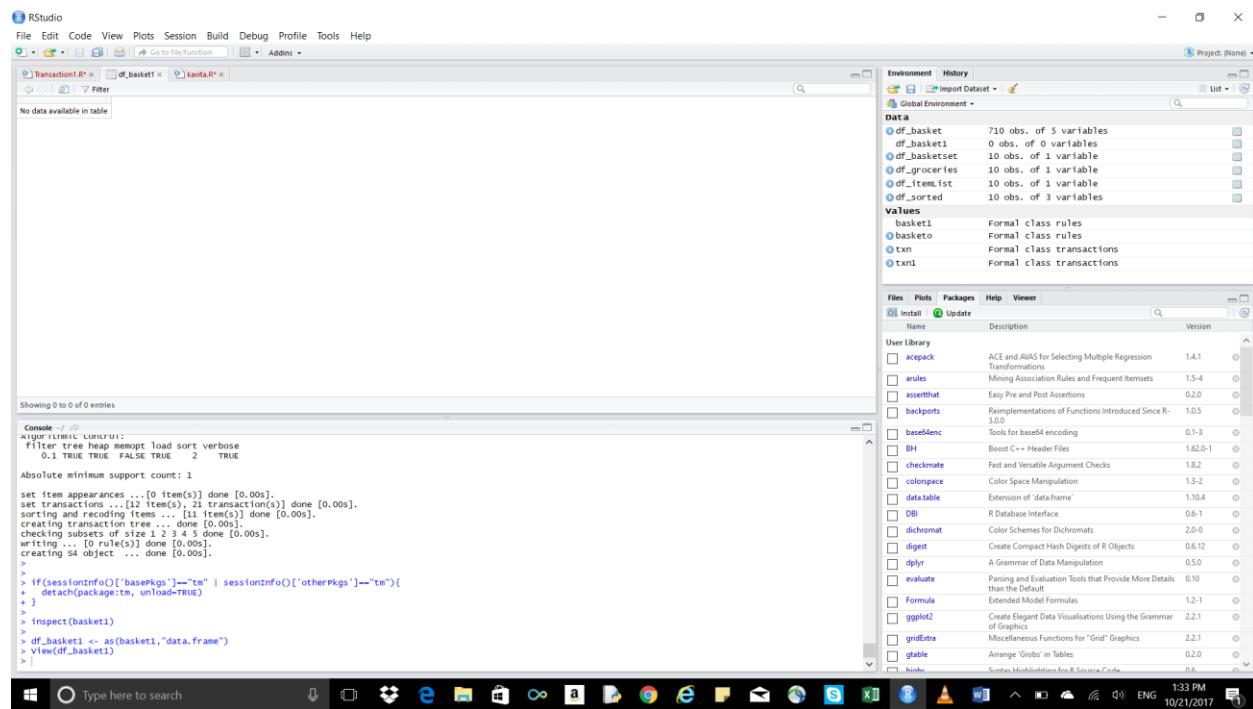
  detach(package:tm, unload=TRUE)

}
```

```
inspect(basket1)
```

```
df_basket1 <- as(basket1,"data.frame")
```

```
View(df_basket1)
```



The screenshot shows the RStudio interface with the following details:

- Environment pane:** Shows objects: df_basket (10 obs. of 5 variables), df_basket1 (0 obs. of 0 variables), df_basketset (10 obs. of 1 variable), df_groceries (10 obs. of 1 variable), df_itemlist (10 obs. of 1 variable), and df_sorted (10 obs. of 3 variables). It also lists values for basket1, basket2, txn, and txn1.
- Packages pane:** Shows the User Library with various packages installed, such as acepack, arules, assertthat, backports, base64enc, BH, checkmate, colorspace, data.table, DBI, dichromat, digest, dplyr, evaluate, Formula, ggplot2, gridExtra, gtable, and hms.
- Console pane:** Displays the R code and its execution output, including the creation of the transaction tree, frequent itemsets, and the final frequent itemsets object.
- Bottom status bar:** Shows the system tray and taskbar icons.

Database2: Filename- Transaction2

	itemList					
1	chocolate	milk	juice	apple	diaper	
2	beer	chocolate	yogurt	eggs	apple	
3	apple	eggs	beer	chocolate	diaper	
4	chocolate	juice	yogurt	fruits	milk	
5	milk	bread	chocolate	beer		
6	chocolate	eggs	bread	beer	fruits	
7	chocolate	eggs	yogurt	beer		
8	milk	chocolate	eggs	fruits	juice	
9	beer	apple	chocolate	fruits		
10	chocolate	beer	fruits	diaper		
11	chocolate	juice	bread	milk		
12	beer	fruits	milk	chocolate	juice	
13	milk	bread	beer	diaper		
14	diaper	beer	eggs	apples		
15	diaper	milk	bread	apple	fruits	
16	apple	diaper	chocolate	beer		
17	milk	juice	beer			
18	fruits	juice	beer	milk	apples	
19	milk	bread	diaper	fruits		
20	diaper	eggs	fruits	beer	apple	

```

df_basketset <- read.csv("Basket_set1.csv")

if(sessionInfo()['basePkgs']=="dplyr" | sessionInfo()['otherPkgs']=="dplyr"){
  detach(package:dplyr, unload=TRUE)
}

library(plyr)

library(arules)

txn2 = read.transactions(file="Transaction2.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)

txn2@itemInfo$labels <- gsub("\\\"","",txn2@itemInfo$labels)

```

```
s <- readline(prompt="Enter Support: ")
```

```
c <- readline(prompt="Enter confidence: ")
```

```
s <- as.numeric(unlist(strsplit(s, ",")))
```

```
c <- as.numeric(unlist(strsplit(c, ",")))
```

The screenshot shows the RStudio interface with the following details:

- File menu:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Project pane:** A tree view showing 'kanta.R' and 'main.R' under the 'Transactions2.R' project.
- Environment pane:** Shows objects in the global environment:
 - Data:** df_basket1 (5 obs. of 5 variables), df_basketset (10 obs. of 1 variable).
 - Values:** basket1, basket2, c, df_item.list, s, txm1, txm2.
- Global Environment pane:** Shows objects in the global environment:
 - Data:** df_basket1 (5 obs. of 5 variables), df_basketset (10 obs. of 1 variable).
 - Values:** basket1, basket2, Formal class rules, c, 0.9, df_item.list, "itemList", s, 0.9, txm1, txm2, Formal class transactions.
- Console pane:** Displays the R script and its execution results. The script reads 'Basket_set1.csv' and performs various operations like reading transaction files, splitting strings, and calculating support and confidence. It also shows the creation of data frames and the flattening of nested data frames.
- Help pane:** Shows the 'Flatten nested data frames' documentation.
- Status bar:** Shows the date (10/21/2017) and time (4:35 PM).

Result for Database2 for frequent -2 itemsets

```
basket2<-apriori(txn2,parameter = list(supp = s, conf = c,maxlen=3,minlen=3))
```

```
if(sessionInfo()['basePkgs']=='tm' | sessionInfo()['otherPkgs']=='tm'){


```

```
detach(package:tm, unload=TRUE)
```

```
}
```

```
inspect(basket2)
```

```
df_basket2 <- as(basket2,"data.frame")
```

```
View(df_basket2)
```

The screenshot shows the RStudio interface with the following details:

- File menu:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Project:** Project (None).
- Script Editor:** A large window containing the R script code.
- Environment:** Shows objects in the global environment:
 - df_basket1: 5 obs. of 5 variables
 - df_basket2: 18 obs. of 5 variables
 - df_basketset: 10 obs. of 1 variable
 - values: basket1, basket2, c, df_itemlist, s, txn1, txn2
- Help:** Shows the help page for "R: Flatten nested data frames".
- Console:** Displays the output of the R code execution, including the apriori function parameters and algorithmic control.
- Taskbar:** Shows various application icons.
- System Tray:** Shows the date and time (4:38 PM, 10/21/2017) and battery status.

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file function Addins

kanta.R final.R Transaction.R df_basket2 df_basket

rules support confidence lift count

1	{apples,eggs} => {milk}	0.0952381	1	1.400000	2
2	{apples,milk} => {eggs}	0.0952381	1	2.625000	2
3	{apple,diaper} => {beer}	0.0952381	1	1.750000	2
4	{bread,diaper} => {beer}	0.0952381	1	1.750000	2
5	{bread,diaper} => {milk}	0.0952381	1	1.400000	2
6	{chocolate,juice} => {fruits}	0.0952381	1	2.100000	2
7	{beer,juice} => {milk}	0.0952381	1	1.400000	2
8	{apple,eggs} => {beer}	0.0952381	1	1.750000	2
9	{apple,eggs} => {milk}	0.0952381	1	1.400000	2
10	{apple,chocolate} => {beer}	0.1428571	1	1.750000	3
11	{apple,bread} => {yogurt}	0.1428571	1	2.333333	3
12	{apple,bread} => {milk}	0.1428571	1	1.400000	3
13	{bread,eggs} => {milk}	0.1428571	1	1.400000	3
14	{eggs,fruits} => {milk}	0.2380952	1	1.400000	5
15	{beer,eggs} => {milk}	0.1428571	1	1.400000	3
16	{bread,yogurt} => {milk}	0.2380952	1	1.400000	5
17	{bread,fruits} => {milk}	0.1428571	1	1.400000	3
18	{beer,bread} => {milk}	0.2380952	1	1.400000	5

Showing 1 to 18 of 18 entries

Console /

```
Parameter specification:
confidence minimal maxrm arem aval originalsupport maxtime support minlen maxlen target ext
0.9 0.1 1 none FALSE TRUE 5 0.09 3 3 rules FALSE

Algorithmic control:
filter tree head memopt load sort verbose
0 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 1

set item appearances ... [0 items] done [0.00s].
set associations ... [0 items], 21 transaction(s) done [0.00s].
sorting and recording items ... [0 items] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done [0.00s].
writing frequent itemsets ... done [0.00s].
creating 54 object(s), ... done [0.00s].
warning message:
In apply {{itemInfo, parameter = list(supp = s, conf = c, maxlen = 3,
  minlen = 1, recursive = TRUE)}, only patterns up to a length of 3 returned!
> df_basket2 <- as(basket2,"data.frame")
> View(df_basket2)
> |
```

R Flattened nested data frames Find in Topic R Documentation

flatten (sonlite)

Flatten nested data frames

Description

In a nested data frame, one or more of the columns consist of another data frame. These structures frequently appear when parsing JSON data from the web. We can flatten such data frames into a regular 2 dimensional tabular structure.

Usage

```
flatten(x, recursive = TRUE)
```

Arguments

- x a data frame
- recursive flatten recursively

Examples

```
options(stringsAsFactors=FALSE)
x <- data.frame(driver = c("Bosser", "Peach"), occupation = c("Koops"
x$vehicle <- data.frame(model = c("Piranha Frouler", "Royal Racer"))
x$vehicleStats <- data.frame(speed = c(55, 34), weight = c(67, 24),
  st(x)
  print(x)
  print(x$vehicle)
  print(x$vehicleStats))
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file function Addins

kanta.R final.R Transaction.R df_basket2 df_basket

Source to search

5: if(sessionInfo()\$basePkgs=="dplyr" | sessionInfo()\$otherPkgs=="dplyr"){
6: detach(package:dplyr, unload=TRUE)
7: }
8: library(plyr)
9: library(arules)
11:
12: tx2n = read.transactions(file="Transaction2.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)
13: tx2n\$itemInfoLabels <- gsub("\\|\\|\\|", "",tx2n\$itemInfoLabels)
15:
16: s <- readline(prompt="Enter Support: ")
18: c <- readline(prompt="Enter confidence: ")
20:
21: s <- as.numeric(unlist(strsplit(s, ",")))
22: c <- as.numeric(unlist(strsplit(c, ",")))
23:
24: basket2<-apriori(tx2n,parameter = list(supp = s, conf = c,maxlen=3,minlen=3))
25:
26: if(sessionInfo()\$basePkgs=="tm" | sessionInfo()\$otherPkgs=="tm"){
28: detach(package:tm, unload=TRUE)
29: }
30:
31: inspect(basket2)
32:
33: df_basket2 <- as(basket2,"data.frame")
34: View(df_basket2)
35: | (open level)

Console /

```
[45] | (open level) >> {bread,yogurt} => {milk} U.2380952 1 1.400000 3
[47] | (bread,fruits) => {milk} 0.1428571 1 1.400000 3
[48] | (bread,bread) => {milk} 0.2380952 1 1.400000 5
> inspect(basket2)
lhs rhs support confidence lift count
[1] {apples,eggs} => {milk} 0.0952381 1 1.400000 2
[2] {apples,milk} => {eggs} 0.0952381 1 2.625000 2
[3] {apple,diaper} => {beer} 0.0952381 1 1.750000 2
[4] {bread,diaper} => {beer} 0.0952381 1 1.750000 2
[5] {bread,yogurt} => {milk} 0.0952381 1 1.400000 2
[6] {chocolate,juice} => {fruits} 0.0952381 1 2.100000 2
[7] {beer,juice} => {milk} 0.0952381 1 1.400000 2
[8] {apple,fruits} => {milk} 0.0952381 1 1.750000 2
[9] {apple,eggs} => {beer} 0.1428571 1 1.750000 3
[10] {apple,chocolate} => {beer} 0.1428571 1 2.333333 3
[12] {apple,bread} => {yogurt} 0.1428571 1 1.400000 3
[13] {bread,eggs} => {milk} 0.1428571 1 1.400000 3
[14] {eggs,fruits} => {milk} 0.2380952 1 1.400000 5
[15] {beer,fruits} => {milk} 0.1428571 1 1.400000 3
[16] {bread,yogurt} => {milk} 0.2380952 1 1.400000 5
[17] {bread,fruits} => {milk} 0.1428571 1 1.400000 3
[18] {beer,bread} => {milk} 0.2380952 1 1.400000 5
```

R Flattened nested data frames Find in Topic R Documentation

flatten (sonlite)

Flatten nested data frames

Description

In a nested data frame, one or more of the columns consist of another data frame. These structures frequently appear when parsing JSON data from the web. We can flatten such data frames into a regular 2 dimensional tabular structure.

Usage

```
flatten(x, recursive = TRUE)
```

Arguments

- x a data frame
- recursive flatten recursively

Examples

```
options(stringsAsFactors=FALSE)
x <- data.frame(driver = c("Bosser", "Peach"), occupation = c("Koops"
x$vehicle <- data.frame(model = c("Piranha Frouler", "Royal Racer"))
x$vehicleStats <- data.frame(speed = c(55, 34), weight = c(67, 24),
  st(x)
  print(x)
  print(x$vehicle)
  print(x$vehicleStats))
```

	lhs	rhs	support	confidence	lift	count
[1]	{apples,eggs}	=> {milk}	0.0952381	1	1.400000	2
[2]	{apples,milk}	=> {eggs}	0.0952381	1	2.625000	2
[3]	{apple,diaper}	=> {beer}	0.0952381	1	1.750000	2
[4]	{bread,diaper}	=> {beer}	0.0952381	1	1.750000	2
[5]	{bread,yogurt}	=> {milk}	0.0952381	1	1.400000	2
[6]	{chocolate,juice}	=> {fruits}	0.0952381	1	2.100000	2

```

[7] {beer,juice}      => {milk}    0.0952381 1           1.400000 2
[8] {apple,eggs}     => {beer}    0.0952381 1           1.750000 2
[9] {apple,eggs}     => {milk}    0.0952381 1           1.400000 2
[10] {apple,chocolate} => {beer}   0.1428571 1           1.750000 3
[11] {apple,bread}    => {yogurt}  0.1428571 1           2.333333 3
[12] {apple,bread}    => {milk}    0.1428571 1           1.400000 3
[13] {bread,eggs}     => {milk}    0.1428571 1           1.400000 3
[14] {eggs,fruits}    => {milk}    0.2380952 1           1.400000 5
[15] {beer,eggs}      => {milk}    0.1428571 1           1.400000 3
[16] {bread,yogurt}   => {milk}    0.2380952 1           1.400000 5
[17] {bread,fruits}   => {milk}    0.1428571 1           1.400000 3
[18] {beer,bread}     => {milk}    0.2380952 1           1.400000 5

```

Result for Database2 for frequent -3 itemsets

```
basket2<-apriori(txn2,parameter = list(supp = s, conf = c,maxlen=4,minlen=4))
```

```
if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){


```

```
detach(package:tm, unload=TRUE)
```

```
}
```

```
inspect(basket2)
```

```
df_basket2 <- as(basket2,"data.frame")
```

```
View(df_basket2)
```

The screenshot shows the RStudio interface with several windows open:

- Environment:** Shows objects like df_basket1, df_basket2, df_basketset, basket1, basket2, c, df_itemList, s, txn1, and txn2.
- Data:** Shows the structure of the data frames.
- Help:** The "Flatten nested data frames" help page is open, explaining how to flatten data frames.
- Console:** Displays the R code for generating the frequent itemsets and the resulting basket2 object.
- Script:** Shows the full R script for the analysis.
- Plots:** No plots are visible.
- Packages:** No packages are visible.
- Help:** The help page for the flatten function is visible.
- Viewer:** No output is visible.

```

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
kafka.R final.R Transaction2.R df_basket2 df_basket1
Source on Save Run Source
4
5 if(sessionInfo()['basePkgs']=="dplyr" | sessionInfo()['otherPkgs']=="dplyr"){
6   detach(package:dplyr, unload=TRUE)
7 }
8 library(plyr)
9
10 library(arules)
11
12 txn2 = read.transactions(file="transaction2.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)
13
14 txn2@itemInfo$labels <- gsub("\r","",txn2@itemInfo$labels)
15
16
17 s <- readline(prompt="Enter Support: ")
18 c <- readline(prompt="Enter confidence: ")
20
21 s <- as.numeric(unlist(strsplit(s, ",")))
22 c <- as.numeric(unlist(strsplit(c, ",")))
23
24 basket2<-apriori(txn2,parameter = list(supp = s, conf = c,maxlen=4,minlen=4))
25
26
27 if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){
28   detach(package:tm, unload=TRUE)
29 }
30
31 inspect(basket2)
32
33 df_basket2 <- as(basket2,"data.frame")
34 view(df_basket2)
35
36
37 (Top Level :)

Console > 
> basket2<-apriori(txn2,parameter = list(supp = s, conf = c,maxlen=4,minlen=4))
Apriori()

Parameter specification:
confidence minval maxval arm originalSupport maxtime support minlen maxlen target ext
0.9      0.1      1  none FALSE          TRUE      5  0.09      4      4 rules FALSE
Algorithmic control:
filter tree head memopt load sort verbose
 0.1 TRUE TRUE FALSE TRUE   2    TRUE
Absolute minimum support count: 1

set Item appearances ... [0 item(s) done [0.00s];
set transactions ... [12 item(s), 21 transaction(s)] done [0.00s].
sorting and recoding items ... [11 item(s)] done [0.00s].
create transaction environment ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [14 rule(s)] done [0.00s].
creating S4 object ... done [0.00s].
warning message:
In apriori(txn2, parameter = list(supp = s, conf = c, maxlen = 4, :
  mining stopped (maxlen reached). only patterns up to a length of 4 returned!
>

```

The screenshot shows the RStudio interface with the following details:

- Environment pane:** Shows a list of objects:
 - Data**: df_basket1 (5 obs. of 5 variables), df_basket2 (14 obs. of 5 variables), df_basketset (10 obs. of 1 variable)
 - values**: basket1, basket2, df_itemList, s, O, txnl, txn2
- Global Environment pane:** Shows the same objects as the Environment pane.
- R console output:**

```

Showing 1 to 14 of 14 entries

Consoles / R
> library(arules)
> library(arulesViz)
> library(arulesViz)
+ detach(package:arulesViz, unload=TRUE)
>
> inspect(basket2)
lhs rhs support confidence lift count
[1] {beer,bread,diaper} => {milk} 0.0952381 1 1.400000 2
[2] {bread,diaper,milk} => {beer} 0.0952381 1 1.750000 2
[3] {beer,diaper,milk} => {bread} 0.0952381 1 2.625000 2
[4] {apple,beer,eggs} => {milk} 0.0952381 1 1.400000 2
[5] {apple,eggs,milk} => {beer} 0.0952381 1 1.750000 2
[6] {apple,beer,bread} => {yogurt} 0.0952381 1 2.333333 2
[7] {apple,bread,yogurt} => {milk} 0.1428571 1 1.400000 3
[8] {apple,bread,milk} => {yogurt} 0.1428571 1 2.333333 3
[9] {apple,milk,yogurt} => {bread} 0.1428571 1 2.625000 3
[10] {apple,beer,bread} => {milk} 0.0952381 1 1.400000 2
[11] {bread,eggs,fruits} => {milk} 0.0952381 1 1.400000 2
[12] {beer,bread,eggs} => {milk} 0.0952381 1 1.400000 2
[13] {beer,eggs,fruits} => {milk} 0.0952381 1 1.400000 2
[14] {beer,bread,yogurt} => {milk} 0.1428571 1 1.400000 3
  
```

lhs	rhs	support	confidence	lift	count
[1] {beer,bread,diaper}	=> {milk}	0.0952381	1	1.400000	2
[2] {bread,diaper,milk}	=> {beer}	0.0952381	1	1.750000	2
[3] {beer,diaper,milk}	=> {bread}	0.0952381	1	2.625000	2
[4] {apple,beer,eggs}	=> {milk}	0.0952381	1	1.400000	2
[5] {apple,eggs,milk}	=> {beer}	0.0952381	1	1.750000	2
[6] {apple,beer,bread}	=> {yogurt}	0.0952381	1	2.333333	2
[7] {apple,bread,yogurt}	=> {milk}	0.1428571	1	1.400000	3
[8] {apple,bread,milk}	=> {yogurt}	0.1428571	1	2.333333	3
[9] {apple,milk,yogurt}	=> {bread}	0.1428571	1	2.625000	3
[10] {apple,beer,bread}	=> {milk}	0.0952381	1	1.400000	2
[11] {bread,eggs,fruits}	=> {milk}	0.0952381	1	1.400000	2
[12] {beer,bread,eggs}	=> {milk}	0.0952381	1	1.400000	2
[13] {beer,eggs,fruits}	=> {milk}	0.0952381	1	1.400000	2
[14] {beer,bread,yogurt}	=> {milk}	0.1428571	1	1.400000	3

Result for Database2 for frequent -4 itemsets

```
basket2<-apriori(txn2,parameter = list(supp = s, conf = c,maxlen=5,minlen=5))
```

```
if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){


```

```
detach(package:tm, unload=TRUE)
```

```
}
```

```
inspect(basket2)
```

```
df_basket2 <- as(basket2,"data.frame")
```

```
View(df_basket2)
```

The screenshot shows the RStudio interface with the following details:

- R Script:** The main workspace contains the R code for generating frequent itemsets and handling package dependencies.
- Environment:** The environment pane shows objects like `df_basket1`, `df_basket2`, `df_basketset`, `basket1`, `basket2`, `c`, `df_itemList`, `s`, `txn1`, and `txn2`.
- Help:** The help pane is open to the 'flatten' function, providing documentation on flattening nested data frames.
- System:** The bottom taskbar shows various system icons and the date/time (10/21/2017, 4:45 PM).

The screenshot shows the RStudio interface with the following details:

- Environment pane:** Shows a table named "rules" with columns: support, confidence, lift, and count. It contains three rows of data:

	rhs	support	confidence	lift	count
[1]	{apple,beer,bread,yogurt} => {milk}	0.0952381	1	1.400000	2
[2]	{apple,beer,bread,milk} => {yogurt}	0.0952381	1	2.333333	2
[3]	{apple,beer,milk,yogurt} => {bread}	0.0952381	1	2.625000	2
- Help pane:** Displays the documentation for the "flatten" function from the "jsonlite" package. It includes sections for Description, Usage, Arguments, and Examples.

```
    lhs                      rhs      support  confidence   lift      count
[1] {apple,beer,bread,yogurt} => {milk} 0.0952381 1      1.400000 2
[2] {apple,beer,bread,milk}   => {yogurt} 0.0952381 1      2.333333 2
[3] {apple,beer,milk,yogurt} => {bread} 0.0952381 1      2.625000 2
```

Result for Database2 for frequent -5 itemsets

```
basket2<-apriori(txn2,parameter = list(supp = s, conf = c,maxlen=6,minlen=6))
```

```
if(sessionInfo()['basePkgs']=='tm' | sessionInfo()['otherPkgs']=='tm'){


```

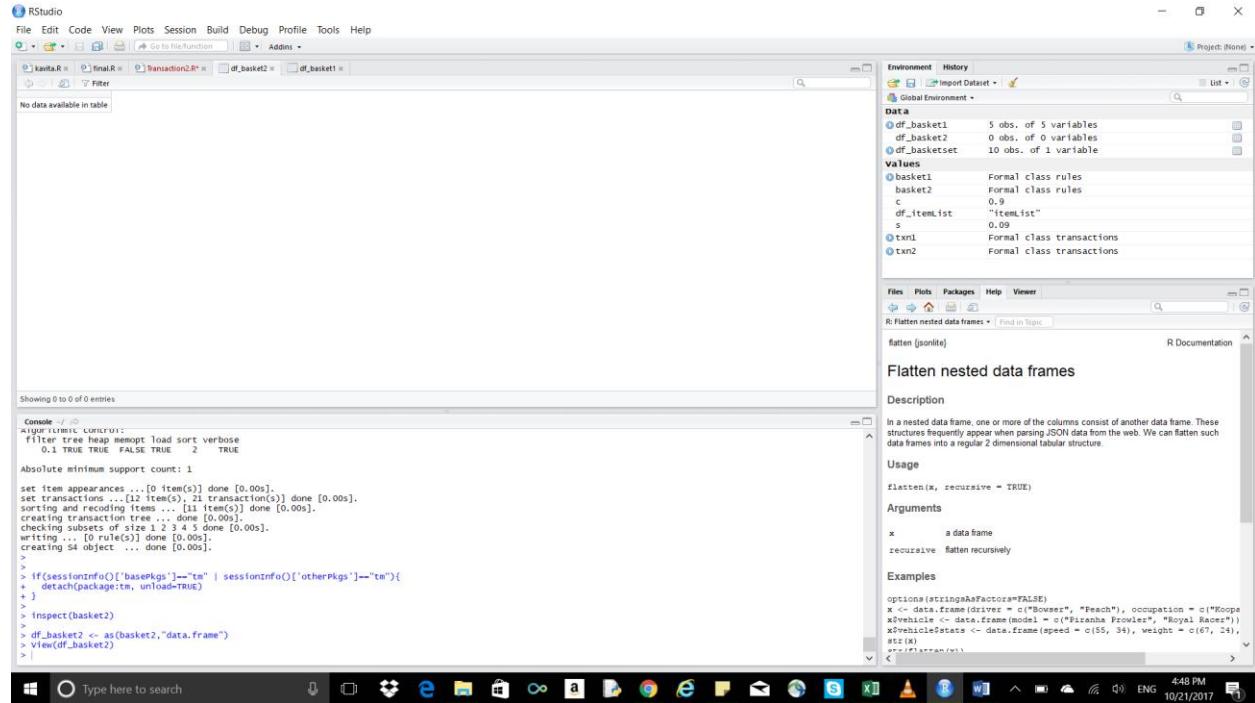
```
detach(package:tm, unload=TRUE)
```

```
}
```

```
inspect(basket2)
```

```
df_basket2 <- as(basket2,"data.frame")
```

```
View(df_basket2)
```



Database3: Filename- Transaction3

	itemList					
1	chocolate	milk	juice	apple		
2	beer	chocolate	yogurt	eggs		
3	apple	eggs	beer	chocolate		
4	chocolate	juice	yogurt	fruits		
5	milk	bread	chocolate	beer	juice	
6	chocolate	eggs	bread	beer	juice	
7	chocolate	eggs	yogurt	beer	fruits	
8	beer	chocolate	eggs	fruits	yogurt	
9	beer	apple	chocolate	fruits		
10	chocolate	beer	fruits	diaper		
11	chocolate	juice	bread	milk		
12	beer	fruits	milk	chocolate	juice	
13	milk	chocolate	beer	diaper	apple	
14	diaper	beer	chocolate	apple		
15	diaper	milk	bread	apple	juice	
16	apple	diaper	chocolate	beer		
17	milk	juice	chocolate	yogurt	diaper	
18	fruits	juice	beer	milk	eggs	
19	milk	chocolate	diaper	fruits	eggs	
20	diaper	eggs	fruits	beer	juice	

```

df_basketset <- read.csv("Basket_set1.csv")

if(sessionInfo()['basePkgs']=="dplyr" | sessionInfo()['otherPkgs']=="dplyr"){
  detach(package:dplyr, unload=TRUE)
}

library(plyr)
library(arules)

txn3 = read.transactions(file="Transaction3.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)

txn3@itemInfo$labels <- gsub("\\"","\"",txn3@itemInfo$labels)

s <- readline(prompt="Enter Support: ")
c <- readline(prompt="Enter confidence: ")
s <- as.numeric(unlist(strsplit(s, ",")))
c <- as.numeric(unlist(strsplit(c, ",")))

```

The screenshot shows the RStudio interface with the following details:

- Top Bar:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help, Help
- Code Editor:** The left pane displays R code for reading a CSV file, creating a basket set, and flattening item labels. The code includes comments and uses functions like `read.csv`, `df_basketset`, and `str`.
- Environment Tab:** Shows the global environment with objects like `df_basket1` (5 obs., 5 variables), `df_basket2` (0 obs., 0 variables), and `df_basketset` (10 obs., 1 variable).
- Data Tab:** Shows the structure of `df_basket1` as a formal class rules object.
- Plots Tab:** Not visible in the screenshot.
- Packages Tab:** Not visible in the screenshot.
- Help Tab:** Shows the help page for `flatten` from the `sonite` package, which describes flattening nested data frames.
- Documentation Tab:** Shows the R Documentation page for `flatten`.
- Console Tab:** Displays the R console output, including the execution of the provided R code and the resulting flattened data frame.
- Bottom Status Bar:** Shows the date (10/21/2017), time (502 PM), and system icons.

Result for Database3 for frequent -2 itemsets

```
basket3<-apriori(txn3,parameter = list(supp = s, conf = c,maxlen=3,minlen=3))
```

```
if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){
```

```
detach(package:tm, unload=TRUE)
```

}

```
inspect(basket3)
```

```
df_basket3 <- as(basket3,"data.frame")
```

```
View(df_basket3)
```

```
File Edit Code View Plots Session Build Debug Profile Tools Help
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/folder Addins ...
kanta.R Final.R Transaction3.R df_basket2 df_basket1
Source on Save Run Source ...
Run Environment History
Global Environment ...
df_basket1 5 obs. of 5 variables
df_basket2 0 obs. of 0 variables
df_basket3 10 obs. of 1 variable
values
df_basket1 Formal class rules
basket2 Formal class rules
basket3 Formal class rules
c 0.7
df_itemList
itemList 0.05
twn1 Formal class transactions
twn2 Formal class transactions
twn3 Formal class transactions
df_basket3
Files Plots Packages Help Viewer
R Flatten nested data frames Find in file ...
flatten (jsonlite)
R Documentation ...
Description
In a nested data frame, one or more of the columns consist of another data frame. These structures frequently appear when parsing JSON data from the web. We can flatten such data frames into a regular 2 dimensional tabular structure
Usage
flatten(x, recursive = TRUE)
Arguments
x a data frame
recursive flatten recursively
Examples
options(stringsAsFactors=FALSE)
x <- data.frame(driver = c("Bosuer", "Feach"), occupation = c("Koops
Xevehicle", "Fruit Vendor", "Royal Racer"))
x$vehicle$cstats <- data.frame(speed = c(55, 34), weight = c(67, 11))
str(x)
x %>% flatten(v1)
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file function Addins

rules

	support	confidence	lift	count
1 {apples,eggs} => {milk}	0.0952381	1.0000000	1.400000	2
2 {apples,milk} => {eggs}	0.0952381	1.0000000	2.625000	2
3 {apple,diaper} => {beer}	0.0952381	1.0000000	1.750000	2
4 {bread,diaper} => {beer}	0.0952381	1.0000000	1.750000	2
5 {bread,diaper} => {milk}	0.0952381	1.0000000	1.400000	2
6 {chocolate,juice} => {fruits}	0.0952381	1.0000000	2.100000	2
7 {beer,juice} => {milk}	0.0952381	1.0000000	1.400000	2
8 {apple,eggs} => {beer}	0.0952381	1.0000000	1.750000	2
9 {apple,eggs} => {milk}	0.0952381	1.0000000	1.400000	2
10 {apple,chocolate} => {beer}	0.1428571	1.0000000	1.750000	3
11 {apple,bread} => {yogurt}	0.1428571	1.0000000	2.333333	3
12 {apple,yogurt} => {bread}	0.1428571	0.7500000	1.968750	3
13 {apple,bread} => {milk}	0.1428571	1.0000000	1.400000	3
14 {apple,milk} => {bread}	0.1428571	0.7500000	1.968750	3
15 {apple,yogurt} => {beer}	0.1428571	0.7500000	1.312500	3
16 {apple,yogurt} => {milk}	0.1428571	0.7500000	1.050000	3
17 {apple,yogurt} => {yogurt}	0.1428571	0.7500000	1.750000	3
18 {apple,milk} => {beer}	0.1428571	0.7500000	1.312500	3
19 {apple,yogurt} => {beer}	0.1428571	0.7500000	1.050000	3
20 {eggs,fruits} => {milk}	0.2380952	1.0000000	1.400000	5
21 {eggs,milk} => {fruits}	0.2380952	0.7142857	1.500000	5
22 {bread,yogurt} => {milk}	0.2380952	1.0000000	1.400000	5
23 {milk,yogurt} => {bread}	0.2380952	0.8333333	2.187500	5
24 {milk,yogurt} => {milk}	0.2380952	0.8333333	1.400000	5
25 {bread,yogurt} => {milk}	0.2380952	1.0000000	1.400000	5
26 {beer,bread} => {milk}	0.2380952	1.0000000	1.400000	5
27 {beer,yogurt} => {milk}	0.1904762	0.8000000	1.120000	4
28 {beer,fruits} => {milk}	0.1904762	0.8000000	1.120000	4

Showing 1 to 20 of 28 entries

```

Console / 
[1] {apple,eggs} => {milk} 0.0952381 1.0000000 1.400000 2
[2] {apple,milk} => {eggs} 0.0952381 1.0000000 2.625000 2
[3] {apple,diaper} => {beer} 0.0952381 1.0000000 1.750000 2
[4] {bread,diaper} => {beer} 0.0952381 1.0000000 1.750000 2
[5] {bread,diaper} => {milk} 0.0952381 1.0000000 1.400000 2
[6] {chocolate,juice} => {fruits} 0.0952381 1.0000000 2.100000 2
[7] {beer,juice} => {milk} 0.0952381 1.0000000 1.400000 2
[8] {apple,eggs} => {beer} 0.0952381 1.0000000 1.750000 2
[9] {apple,eggs} => {milk} 0.0952381 1.0000000 1.400000 2
[10] {apple,chocolate} => {beer} 0.1428571 1.0000000 1.750000 3
[11] {apple,bread} => {yogurt} 0.1428571 1.0000000 2.333333 3
[12] {apple,yogurt} => {bread} 0.1428571 0.7500000 1.968750 3
[13] {apple,bread} => {milk} 0.1428571 1.0000000 1.400000 3
[14] {apple,milk} => {bread} 0.1428571 0.7500000 1.968750 3
[15] {apple,yogurt} => {beer} 0.1428571 0.7500000 1.312500 3
[16] {apple,yogurt} => {milk} 0.1428571 0.7500000 1.050000 3
[17] {apple,milk} => {yogurt} 0.1428571 0.7500000 1.750000 3
[18] {apple,milk} => {beer} 0.1428571 0.7500000 1.312500 3
[19] {apple,yogurt} => {beer} 0.1428571 0.7500000 1.050000 3
[20] {eggs,fruits} => {milk} 0.2380952 1.0000000 1.400000 5
[21] {eggs,milk} => {fruits} 0.2380952 0.7142857 1.500000 5
[22] {bread,yogurt} => {milk} 0.2380952 1.0000000 1.400000 5
[23] {milk,yogurt} => {bread} 0.2380952 0.8333333 2.187500 5
[24] {bread,fruits} => {milk} 0.2380952 1.0000000 1.400000 5
[25] {bread,yogurt} => {milk} 0.2380952 1.0000000 1.400000 5
[26] {beer,bread} => {milk} 0.2380952 1.0000000 1.400000 5
[27] {beer,yogurt} => {milk} 0.1904762 0.8000000 1.120000 4
[28] {beer,fruits} => {milk} 0.1904762 0.8000000 1.120000 4
> df_basket3 <- as(basket3,"data.frame")
> View(df_basket3)
| 
```

lhs	rhs	support	confidence	lift	count
[1] {apples,eggs}	=> {milk}	0.0952381	1.0000000	1.400000	2
[2] {apples,milk}	=> {eggs}	0.0952381	1.0000000	2.625000	2
[3] {apple,diaper}	=> {beer}	0.0952381	1.0000000	1.750000	2
[4] {bread,diaper}	=> {beer}	0.0952381	1.0000000	1.750000	2
[5] {bread,diaper}	=> {milk}	0.0952381	1.0000000	1.400000	2
[6] {chocolate,juice}	=> {fruits}	0.0952381	1.0000000	2.100000	2
[7] {beer,juice}	=> {milk}	0.0952381	1.0000000	1.400000	2
[8] {apple,eggs}	=> {beer}	0.0952381	1.0000000	1.750000	2
[9] {apple,eggs}	=> {milk}	0.0952381	1.0000000	1.400000	2
[10] {apple,chocolate}	=> {beer}	0.1428571	1.0000000	1.750000	3
[11] {apple,bread}	=> {yogurt}	0.1428571	1.0000000	2.333333	3
[12] {apple,yogurt}	=> {bread}	0.1428571	0.7500000	1.968750	3
[13] {apple,bread}	=> {milk}	0.1428571	1.0000000	1.400000	3
[14] {apple,milk}	=> {bread}	0.1428571	0.7500000	1.968750	3
[15] {apple,yogurt}	=> {beer}	0.1428571	0.7500000	1.312500	3
[16] {apple,yogurt}	=> {milk}	0.1428571	0.7500000	1.050000	3
[17] {apple,milk}	=> {yogurt}	0.1428571	0.7500000	1.750000	3
[18] {apple,milk}	=> {beer}	0.1428571	0.7500000	1.312500	3
[19] {apple,yogurt}	=> {beer}	0.1428571	0.7500000	1.050000	3
[20] {eggs,fruits}	=> {milk}	0.2380952	1.0000000	1.400000	5
[21] {eggs,milk}	=> {fruits}	0.2380952	0.7142857	1.500000	5
[22] {beer,eggs}	=> {milk}	0.2380952	1.0000000	1.400000	3
[23] {bread,yogurt}	=> {milk}	0.2380952	1.0000000	1.400000	5
[24] {milk,yogurt}	=> {bread}	0.2380952	0.8333333	2.187500	5
[25] {bread,fruits}	=> {milk}	0.2380952	1.0000000	1.400000	3
[26] {beer,bread}	=> {milk}	0.2380952	1.0000000	1.400000	5
[27] {beer,yogurt}	=> {milk}	0.1904762	0.8000000	1.120000	4
[28] {beer,fruits}	=> {milk}	0.1904762	0.8000000	1.120000	4

Result for Database3 for frequent -3 itemsets

```
basket3<-apriori(txn3,parameter = list(supp = s, conf = c,maxlen=4,minlen=4))
```

```
if(sessionInfo()['basePkgs']=='tm' | sessionInfo()['otherPkgs']=='tm'){


```

```
    detach(package:tm, unload=TRUE)

}
```

```
inspect(basket3)
```

```
df_basket3 <- as(basket3,"data.frame")
```

```
View(df_basket3)
```

The screenshot shows the RStudio interface with the following details:

- File menu:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Session tab:** kanta.R, final.R, Transaction3.R, df_basket1, df_basket2, df_basket3.
- Code pane:** Contains the R script for generating frequent itemsets. It includes imports for plyr, arules, and dplyr, reads a transactions file, and performs filtering based on session packages. It then calls the apriori function with parameters: parameter = list(supp = s, conf = c, maxlen=4, minlen=4). The output of this command is captured in the console.
- Environment pane:** Shows the global environment with objects like basket1, basket2, basket3, df_basket1, df_basket2, df_basket3, df_basketset, and txn1.
- Values pane:** Shows the values for each object, such as basket1 having 5 obs. of 5 variables.
- Help pane:** Shows the help for the flatten function.
- Console pane:** Displays the R command and its output. The output shows the parameter specification, algorithmic control, and a warning message indicating the search was stopped due to maxlen reached.
- System tray:** Shows the date and time as 10/21/2017 5:10 PM.

Screenshot of RStudio showing the environment and code editor.

Environment:

- Data:
 - df_basket1: 5 obs. of 5 variables
 - df_basket2: 0 obs. of 0 variables
 - df_basket3: 16 obs. of 5 variables
 - df_basketset: 10 obs. of 1 variable
- Values:
 - basket1: Formal class rules
 - basket2: Formal class rules
 - basket3: Formal class rules
 - c: 0.7
 - df_itemlist: "itemlist"
 - s: 0.07
 - txnl: Formal class transactions

Code Editor:

```

kanta.R *  Final.R *  Transaction.R*  df_basket3 *  df_basket2 *  df_basket1 *
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file function  Addins *
rules
support confidence lift count
1 {bread,chocolate,milk} => {juice} 0.0952381 1.00 2.100000 2
2 {beer,bread,juice} => {chocolate} 0.0952381 1.00 1.235294 2
3 {beer,bread,chocolate} => {juice} 0.0952381 1.00 2.100000 2
4 {eggs,fruits,yogurt} => {beer} 0.0952381 1.00 1.500000 2
5 {beer,fruits,yogurt} => {eggs} 0.0952381 1.00 2.625000 2
6 {eggs,fruits,yogurt} => {chocolate} 0.0952381 1.00 1.235294 2
7 {beer,eggs,yogurt} => {chocolate} 0.1428571 1.00 1.235294 3
8 {chocolate,eggs,yogurt} => {beer} 0.1428571 1.00 1.500000 3
9 {beer,chocolate,yogurt} => {eggs} 0.1428571 1.00 2.625000 3
10 {beer,fruits,yogurt} => {chocolate} 0.0952381 1.00 1.235294 2
11 {apple,beer,diaper} => {chocolate} 0.1428571 1.00 1.235294 3
12 {apple,chocolate,diaper} => {beer} 0.1428571 1.00 1.500000 3
13 {apple,fruits,juice} => {beer} 0.1428571 0.75 2.250000 3
14 {eggs,fruits,juice} => {beer} 0.0952381 1.00 1.500000 2
15 {fruits,juice,milk} => {beer} 0.0952381 1.00 1.500000 2
16 {beer,fruits,milk} => {juice} 0.0952381 1.00 2.100000 2
  
```

Showing 1 to 16 of 16 entries

Console

```

> ??
> inspect(basket3)
  In:
  [1] {bread,chocolate,milk}    rhs support confidence lift count
  [2] {beer,bread,juice}        rhs support confidence lift count
  [3] {beer,bread,chocolate}   rhs support confidence lift count
  [4] {eggs,fruits,yogurt}     rhs support confidence lift count
  [5] {beer,fruits,yogurt}     rhs support confidence lift count
  [6] {eggs,fruits,yogurt}     rhs support confidence lift count
  [7] {beer,fruits,yogurt}     rhs support confidence lift count
  [8] {chocolate,eggs,yogurt} rhs support confidence lift count
  [9] {beer,chocolate,yogurt}  rhs support confidence lift count
  [10] {beer,fruits,yogurt}    rhs support confidence lift count
  [11] {apple,beer,diaper}     rhs support confidence lift count
  [12] {apple,chocolate,diaper}rhs support confidence lift count
  [13] {apple,fruits,juice}    rhs support confidence lift count
  [14] {eggs,fruits,juice}     rhs support confidence lift count
  [15] {fruits,juice,milk}     rhs support confidence lift count
  [16] {beer,fruits,milk}      rhs support confidence lift count
  
```

R: Flatten nested data frames Find in Topic

Flatten nested data frames R Documentation

Flatten nested data frames

Description

In a nested data frame, one or more of the columns consist of another data frame. These structures frequently appear when parsing JSON data from the web. We can flatten such data frames into a regular 2 dimensional tabular structure.

Usage

```

flatten(x, recursive = TRUE)

```

Arguments

```

x             a data frame
recursive    flatten recursively

```

Examples

```

options(everask=FALSE)
x <- data.frame(driver = c("Bossz", "Peach"), occupation = c("Kops
x$vehicle <- data.frame(model = c("%Sirana Fowler", "Royal Racer"))
x$vehicle$stats <- data.frame(speed = c(55, 34), weight = c(67, 24),
size = c(1.2, 1.1)))
print(x)
  
```

lhs	rhs	support	confidence	lift	co
[1] {bread, chocolate, milk}	=> {juice}	0.0952381	1.00	2.100000	2
[2] {beer, bread, juice}	=> {chocolate}	0.0952381	1.00	1.235294	2
[3] {beer, bread, chocolate}	=> {juice}	0.0952381	1.00	2.100000	2
[4] {eggs, fruits, yogurt}	=> {beer}	0.0952381	1.00	1.500000	2
[5] {beer, fruits, yogurt}	=> {eggs}	0.0952381	1.00	2.625000	2
[6] {eggs, fruits, yogurt}	=> {chocolate}	0.0952381	1.00	1.235294	2
[7] {beer, eggs, yogurt}	=> {chocolate}	0.1428571	1.00	1.235294	3
[8] {chocolate, eggs, yogurt}	=> {beer}	0.1428571	1.00	1.500000	3
[9] {beer, chocolate, yogurt}	=> {eggs}	0.1428571	1.00	2.625000	3
[10] {beer, fruits, yogurt}	=> {chocolate}	0.0952381	1.00	1.235294	2
[11] {apple, beer, diaper}	=> {chocolate}	0.1428571	1.00	1.235294	3
[12] {apple, chocolate, diaper}	=> {beer}	0.1428571	1.00	1.500000	3
[13] {apple, fruits, juice}	=> {beer}	0.1428571	0.75	2.250000	3
[14] {eggs, fruits, juice}	=> {beer}	0.0952381	1.00	1.500000	2
[15] {fruits, juice, milk}	=> {beer}	0.0952381	1.00	1.500000	2
[16] {beer, fruits, milk}	=> {juice}	0.0952381	1.00	2.100000	2

Result for Database3 for frequent -4 itemsets

```
basket3<-apriori(txn3,parameter = list(supp = s, conf = c,maxlen=5,minlen=5))
```

```
if(sessionInfo()['basePkgs']=='tm' | sessionInfo()['otherPkgs']=='tm'){


```

```
detach(package:tm, unload=TRUE)
```

```
}
```

```
inspect(basket3)
```

```
df_basket3 <- as(basket3,"data.frame")
```

```
View(df_basket3)
```

The screenshot shows the RStudio interface with the following details:

- File Menu:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Project:** Project (None).
- Script Editor:** Displays the R script with the following code:

```
4 if(sessionInfo()['basePkgs']=='dplyr' | sessionInfo()['otherPkgs']=='dplyr'){
5   detach(package:dplyr, unload=TRUE)
6 }
7 library(plyr)
8 library(arules)
9
10 txn3 = read.transactions(file="transaction3.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)
11 txn3@itemInfo$labels <- gsub("\n","",txn3@itemInfo$labels)
12
13 s <- readline(prompt="Enter Support: ")
14 c <- readline(prompt="Enter confidence: ")
15
16 s <- as.numeric(unlist(strsplit(s, ",")))
17 c <- as.numeric(unlist(strsplit(c, ",")))
18
19 basket3<-apriori(txn3,parameter = list(supp = s, conf = c,maxlen=5,minlen=5))
20
21 if(sessionInfo()['basePkgs']=='tm' | sessionInfo()['otherPkgs']=='tm'){
22   detach(package:tm, unload=TRUE)
23 }
24
25 inspect(basket3)
26
27 df_basket3 <- as(basket3,"data.frame")
28 View(df_basket3)
29
30 df_basket3
```

- Environment View:** Shows the following objects:

 - df_basket1: 5 obs. of 5 variables
 - df_basket2: 0 obs. of 0 variables
 - df_basket3: 16 obs. of 5 variables
 - df_basketset: 10 obs. of 1 variable

- Values View:** Shows the following values:

 - df_basket1: Formal class rules
 - df_basket2: Formal class rules
 - df_basket3: Formal class rules
 - c: 0.7
 - df_basketset: 0.67
 - s: "1.00"
 - df_basket1: Formal class transactions

- Help View:** Shows the documentation for `flatten`.
- Bottom Status Bar:** Type here to search, system tray icons, and timestamp: 5:12 PM 10/21/2017.

```

Showing 1 to 4 of 4 entries

Console -> 
Absolute minimum support count: 1

set item appearances ... [0 items] done [0.00s].
set transactions ... [11 items], 21 transaction(s) done [0.00s].
sorting and recoding items ... [10 items] done [0.00s].
creating 4 object(s) ... done [0.00s].
checking subsets of size 1, 2, 3, 4, 5 done [0.00s].
writing ... [4 rule(s)] done [0.00s].
creating 54 object(s) ... done [0.00s].
> if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){
+   detach(package:tm, unload=TRUE)
+ }
> inspect(basket3)
#> lhs                                rhs      support  confidence    lift
#> count
[1] {beer,eggs,fruits,yogurt}      => {chocolate} 0.0952381 1      1.2352
94 2
[2] {chocolate,eggs,fruits,yogurt} => {beer}       0.0952381 1      1.5000
00 2
[3] {beer,chocolate,fruits,yogurt} => {eggs}       0.0952381 1      2.6250
00 2
[4] {beer,chocolate,eggs,fruits}   => {yogurt}    0.0952381 1      4.2000
00 2
> df_basket3 <- as(basket3,"data.frame")
> View(df_basket3)
>

```

lhs	rhs	support	confidence	lift
count				
[1] {beer,eggs,fruits,yogurt}	=> {chocolate}	0.0952381	1	1.2352
94 2				
[2] {chocolate,eggs,fruits,yogurt}	=> {beer}	0.0952381	1	1.5000
00 2				
[3] {beer,chocolate,fruits,yogurt}	=> {eggs}	0.0952381	1	2.6250
00 2				
[4] {beer,chocolate,eggs,fruits}	=> {yogurt}	0.0952381	1	4.2000
00 2				

Result for Database3 for frequent -5 itemsets

```

basket3<-apriori(txn3,parameter = list(supp = s, conf = c,maxlen=6,minlen=6))

if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){

  detach(package:tm, unload=TRUE)

}

inspect(basket3)

df_basket3 <- as(basket3,"data.frame")

View(df_basket3)

```

```

4
5 > if(sessionInfo()$basePkgs=="dplyr" | sessionInfo()$otherPkgs=="dplyr"){
6   detach(package:dplyr, unload=TRUE)
7 }
8 library(arules)
9
10 library(arules)
11
12 txm3 = read.transactions(file="Transaction3.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)
13 txm3$itemInfoLabels <- gsub("\r", "",txm3$itemInfoLabels)
14
15
16
17 s <- readline(prompt="Enter Support: ")
18 c <- readline(prompt="Enter confidence: ")
19
20 s <- as.numeric(unlist(strsplit(s, ",")))
21 c <- as.numeric(unlist(strsplit(c, ",")))
22
23 basket3<-apriori(txm3,parameter = list(supp = s, conf = c,maxlen=6,minlen=6))
24
25
26
27 if(sessionInfo()$basePkgs=="tm" | sessionInfo()$otherPkgs=="tm"){
28   detach(package:tm, unload=TRUE)
29 }
30
31 inspect(basket3)
32
33 df_basket3 <- as(basket3,"data.frame")
34 View(df_basket3)
29.2 (Top Level)

```

Console / R Script

```

> 
> df_basket3 <- as(basket3,"data.frame")
> View(df_basket3)
> basket3<-apriori(txm3,parameter = list(supp = s, conf = c,maxlen=6,minlen=6))

Parameter specification:
confidence minval maxc arcm aval originalSupport maxtime support minlen maxlen target ext
0.7 0.1 1 none FALSE TRUE 5 0.07 6 6 rules FALSE

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

Absolute minimum support count: 1

set item appearances ... [0 item(s)] done [0.00s].
set transactions ... [1 item(s), 21 transaction(s)] done [0.00s].
sorting and recoding items ... [0 item(s)] done [0.00s].
creating itemsets ... [0 item(s)] done [0.00s].
checking subsets of size 1:2:3:4:5 done [0.00s].
writing ... [0 rule(s)] done [0.00s].
creating S4 object ... done [0.00s].
|>


```

In a nested data frame, one or more of the columns consist of another data frame. These structures frequently appear when parsing JSON data from the web. We can flatten such data frames into a regular 2 dimensional tabular structure.

Usage

```

flatten(x, recursive = TRUE)

```

Arguments

```

x
  a data frame
recursive
  flatten recursively

```

Examples

```

options(stringsAsFactors=FALSE)
x <- data.frame(driver = c("Bosser", "Peach"), occupation = c("Koops"
x$vehicle <- data.frame(model = c("Kirby Frouler", "Royal Racer"))
x$vehicle$stats <- data.frame(speed = c(55, 34), weight = c(67, 24),
st(x)
print(x)

```

```

4
5 > if(sessionInfo()$basePkgs=="dplyr" | sessionInfo()$otherPkgs=="dplyr"){
6   detach(package:dplyr, unload=TRUE)
7 }
8 library(arules)
9
10 library(arules)
11
12 txm3 = read.transactions(file="Transaction3.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)
13 txm3$itemInfoLabels <- gsub("\r", "",txm3$itemInfoLabels)
14
15
16
17 s <- readline(prompt="Enter Support: ")
18 c <- readline(prompt="Enter confidence: ")
19
20 s <- as.numeric(unlist(strsplit(s, ",")))
21 c <- as.numeric(unlist(strsplit(c, ",")))
22
23 basket3<-apriori(txm3,parameter = list(supp = s, conf = c,maxlen=6,minlen=6))
24
25
26
27 if(sessionInfo()$basePkgs=="tm" | sessionInfo()$otherPkgs=="tm"){
28   detach(package:tm, unload=TRUE)
29 }
30
31 inspect(basket3)
32
33 df_basket3 <- as(basket3,"data.frame")
34 View(df_basket3)
29.2 (Top Level)

Showing 0 to 0 of 0 entries

Console / R Script

> 
> df_basket3 <- as(basket3,"data.frame")
> View(df_basket3)
> basket3<-apriori(txm3,parameter = list(supp = s, conf = c,maxlen=6,minlen=6))

Parameter specification:
confidence minval maxc arcm aval originalSupport maxtime support minlen maxlen target ext
0.7 0.1 1 none FALSE TRUE 5 0.07 6 6 rules FALSE

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

Absolute minimum support count: 1

set item appearances ... [0 item(s)] done [0.00s].
set transactions ... [1 item(s), 21 transaction(s)] done [0.00s].
sorting and recoding items ... [0 item(s)] done [0.00s].
creating itemsets ... [0 item(s)] done [0.00s].
checking subsets of size 1:2:3:4:5 done [0.00s].
writing ... [0 rule(s)] done [0.00s].
creating S4 object ... done [0.00s].
|> inspect(basket3)
|> df_basket3 <- as(basket3,"data.frame")
|> View(df_basket3)
|> 
```

In a nested data frame, one or more of the columns consist of another data frame. These structures frequently appear when parsing JSON data from the web. We can flatten such data frames into a regular 2 dimensional tabular structure.

Usage

```

flatten(x, recursive = TRUE)

```

Arguments

```

x
  a data frame
recursive
  flatten recursively

```

Examples

```

options(stringsAsFactors=FALSE)
x <- data.frame(driver = c("Bosser", "Peach"), occupation = c("Koops"
x$vehicle <- data.frame(model = c("Kirby Frouler", "Royal Racer"))
x$vehicle$stats <- data.frame(speed = c(55, 34), weight = c(67, 24),
st(x)
print(x)

```

Database4: Filename- Transaction4

	itemList				
1	chocolate	milk	juice	apple	bread
2	beer	bread	yogurt	eggs	milk
3	apple	milk	beer	chocolate	juice
4	chocolate	milk	yogurt	fruits	
5	milk	diaper	chocolate	beer	
6	chocolate	diaper	bread	beer	
7	chocolate	diaper	yogurt	beer	
8	beer	milk	eggs	fruits	
9	beer	milk	chocolate	fruits	
10	chocolate	diaper	fruits	diaper	
11	chocolate	diaper	bread	milk	
12	beer	fruits	milk	chocolate	
13	milk	fruits	beer	diaper	
14	diaper	milk	chocolate	apple	
15	diaper	beer	bread	apple	
16	apple	fruits	chocolate	beer	
17	milk	beer	chocolate	yogurt	
18	fruits	diaper	beer	milk	
19	milk	beer	diaper	fruits	
20	diaper	milk	fruits	beer	

```

df_basketset <- read.csv("Basket_set1.csv")

if(sessionInfo()['basePkgs']=="dplyr" | sessionInfo()['otherPkgs']=="dplyr"){
  detach(package:dplyr, unload=TRUE)
}

library(plyr)
library(arules)

txn4 = read.transactions(file="Transaction4.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)

txn4@itemInfo$labels <- gsub("\\\"","",txn4@itemInfo$labels)

s <- readline(prompt="Enter Support: ")

c <- readline(prompt="Enter confidence: ")

s <- as.numeric(unlist(strsplit(s, ",")))

c <- as.numeric(unlist(strsplit(c, ",")))

```

```

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file function Addins
kafka.R  final.R  Transaction4.R*  df_basket3  df_basket2  df_basket1
Source on Save  Run  Source  Environment History  Global Environment
df_basketset <- read.csv("basket_set1.csv")
if(sessionInfo()["basePkgs"]=="dplyr" | sessionInfo()["otherPkgs"]=="dplyr"){
  detach(package:dplyr, unload=TRUE)
}
library(arules)
txn4 = read.transactions(file="transaction4.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)
txn4@itemInfo$labels <- gsub("^","",txn4@itemInfo$labels)
s <- readline(prompt="Enter Support: ")
c <- readline(prompt="Enter confidence: ")
s <- as.numeric(unlist(strsplit(s, ",")))
c <- as.numeric(unlist(strsplit(c, ",")))
basket4<-apriori(txn4,parameter = list(supp = s, conf = c,maxlen=6,minlen=6))
if(sessionInfo()["basePkgs"]=="tm" | sessionInfo()["otherPkgs"]=="tm"){
  detach(package:tm, unload=TRUE)
}
inspect(basket4)
}

```

Environment

Data

- df_basket1 5 obs. of 5 variables
- df_basket2 0 obs. of 0 variables
- df_basket3 0 obs. of 0 variables
- df_basketset 10 obs. of 1 variable

Values

- basket1 Formal class rules
- basket2 Formal class rules
- basket3 Formal class rules
- c 0.5
- df_itemList "itemList"
- s 0.95
- txm1 Formal class transactions
- txm2 Formal class transactions

Files Plots Packages Help Viewer

R: Flatten nested data frames Find in Topic

Flatten (sonlite) R Documentation

Flatten nested data frames

Description

In a nested data frame, one or more of the columns consist of another data frame. These structures frequently appear when parsing JSON data from the web. We can flatten such structures into a regular 2 dimensional tabular structure.

Usage

```
flatten(x, recursive = TRUE)
```

Arguments

- x a data frame
- recursive flatten recursively

Examples

```
options(everyNamespace=FALSE)
x <- data.frame(driver = c("Bossy", "Peach"), occupation = c("Kopps", "Princess Peach"))
x@vehicle <- data.frame(model = c("Kirby", "Peach"))
x@vehicle$stats <- data.frame(speed = c(55, 34), weight = c(67, 24),
size = c(10, 15),
strength = c(10, 15))
```

Result for Database4 for frequent -2 itemsets

```
basket4<-apriori(txn4,parameter = list(supp = s, conf = c,maxlen=3,minlen=3))
```

```
if(sessionInfo()["basePkgs"]=="tm" | sessionInfo()["otherPkgs"]=="tm"){

}
```

```
detach(package:tm, unload=TRUE)
```

```
}
```

```
inspect(basket4)
```

```
df_basket4 <- as(basket4,"data.frame")
```

```
View(df_basket4)
```

The screenshot shows the RStudio interface with the following components:

- Top Bar:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Left Panel:** A script editor containing R code for basket analysis. The code includes imports for knitr, dplyr, plyr, and arules, reads a CSV file, performs transactions, and creates a frequent itemset list. It then defines a function `apriori` for generating association rules, which is used to find rules with support > 0.5 and confidence > 0.1. The resulting rules are printed to the console.
- Console:** Displays the output of the R code, including the generated rules:

```
> basket4<-apriori(txn,parameter = list(supp = s, conf = c,maxlen=3,minlen=3))
Apriori

Parameter specification:
confidence minval maxitem ave item辰al originalSupport maxtime support minlen maxlen target ext
0.5 0.1 1 0.1 FALSE TRUE 5 0.05 3 3 rules FALSE

Algorithmic control:
filter tree heap minsupp load sort verbose
0.1 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 1

set item appearances ... [0 items] done [0.00s].
set transactions ... [0 items], 21 transaction(s) done [0.00s].
sorts and condensation ... [0 items] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done [0.00s].
writing frequent itemsets ... done [0.00s].
creating S4 object ... done [0.00s].
Warning message:
In apriori(txn, parameter = list(supp = s, conf = c, maxlen = 3, :
  mining stopped (maxlen reached). only patterns up to a length of 3 returned!
```

- Right Panel:** A help page titled "Flatten nested data frames". It explains what flatten does, provides usage examples, and lists arguments and examples for the `flatten` function.

A screenshot of an RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help, and Addins. A search bar at the bottom says "Type here to search". The main workspace shows a console window with R code and output, an environment browser with a tree view of variables, and a help viewer for the 'flatten' function. A status bar at the bottom right shows the date and time.

	lhs	rhs	support	confidence	lift	count
[1]	{beer, eggs}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[2]	{eggs, milk}	=> {beer}	0.0952381	1.0000000	1.4000000	2
[3]	{apple, juice}	=> {chocolate}	0.0952381	1.0000000	1.6153846	2
[4]	{chocolate, juice}	=> {apple}	0.0952381	1.0000000	4.2000000	2
[5]	{apple, chocolate}	=> {juice}	0.0952381	0.5000000	5.2500000	2
[6]	{apple, juice}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[7]	{juice, milk}	=> {apple}	0.0952381	1.0000000	4.2000000	2

[8]	{apple,milk}	=> {juice}	0.0952381	0.6666667	7.0000000	2
[9]	{chocolate,juice}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[10]	{juice,milk}	=> {chocolate}	0.0952381	1.0000000	1.6153846	2
[11]	{chocolate,yogurt}	=> {beer}	0.0952381	0.6666667	0.9333333	2
[12]	{beer,yogurt}	=> {chocolate}	0.0952381	0.6666667	1.0769231	2
[13]	{chocolate,yogurt}	=> {milk}	0.0952381	0.6666667	0.9333333	2
[14]	{milk,yogurt}	=> {chocolate}	0.0952381	0.6666667	1.0769231	2
[15]	{beer,yogurt}	=> {milk}	0.0952381	0.6666667	0.9333333	2
[16]	{milk,yogurt}	=> {beer}	0.0952381	0.6666667	0.9333333	2
[17]	{apple,chocolate}	=> {beer}	0.0952381	0.5000000	0.7000000	2
[18]	{apple,beer}	=> {chocolate}	0.0952381	0.6666667	1.0769231	2
[19]	{apple,chocolate}	=> {milk}	0.1428571	0.7500000	1.0500000	3
[20]	{apple,milk}	=> {chocolate}	0.1428571	1.0000000	1.6153846	3
[21]	{bread,diaper}	=> {chocolate}	0.0952381	0.6666667	1.0769231	2
[22]	{bread,chocolate}	=> {diaper}	0.0952381	0.6666667	1.2727273	2
[23]	{bread,diaper}	=> {beer}	0.0952381	0.6666667	0.9333333	2
[24]	{beer,bread}	=> {diaper}	0.0952381	0.6666667	1.2727273	2
[25]	{bread,chocolate}	=> {milk}	0.0952381	0.6666667	0.9333333	2
[26]	{bread,milk}	=> {chocolate}	0.0952381	0.6666667	1.0769231	2
[27]	{diaper,fruits}	=> {beer}	0.1904762	0.8000000	1.1200000	4
[28]	{beer,fruits}	=> {diaper}	0.1904762	0.5000000	0.9545455	4
[29]	{beer,diaper}	=> {fruits}	0.1904762	0.5000000	1.0500000	4
[30]	{diaper,fruits}	=> {milk}	0.1904762	0.8000000	1.1200000	4
[31]	{fruits,milk}	=> {diaper}	0.1904762	0.5000000	0.9545455	4
[32]	{diaper,milk}	=> {fruits}	0.1904762	0.5714286	1.2000000	4
[33]	{chocolate,fruits}	=> {beer}	0.1428571	0.6000000	0.8400000	3
[34]	{chocolate,fruits}	=> {milk}	0.1428571	0.6000000	0.8400000	3
[35]	{beer,fruits}	=> {milk}	0.3333333	0.8750000	1.2250000	7
[36]	{fruits,milk}	=> {beer}	0.3333333	0.8750000	1.2250000	7
[37]	{beer,milk}	=> {fruits}	0.3333333	0.6363636	1.3363636	7
[38]	{chocolate,diaper}	=> {beer}	0.1428571	0.5000000	0.7000000	3
[39]	{chocolate,diaper}	=> {milk}	0.1428571	0.5000000	0.7000000	3
[40]	{beer,diaper}	=> {milk}	0.2380952	0.6250000	0.8750000	5
[41]	{diaper,milk}	=> {beer}	0.2380952	0.7142857	1.0000000	5
[42]	{beer,chocolate}	=> {milk}	0.2380952	0.6250000	0.8750000	5
[43]	{chocolate,milk}	=> {beer}	0.2380952	0.5555556	0.7777778	5

Result for Database4 for frequent -3 itemsets

```
basket4<-apriori(txn4,parameter = list(supp = s, conf = c,maxlen=4,minlen=4))

if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){

  detach(package:tm, unload=TRUE)

}

inspect(basket4)

df_basket4 <- as(basket4,"data.frame")

View(df_basket4)
```

The screenshot shows the RStudio interface with the following details:

- File menu:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Session tab:** Shows scripts kanta.R, final.R, and Transaction4.R.
- Code pane:** Displays the R code for generating frequent 3-itemsets from the txn4 dataset.
- Environment pane:** Shows the global environment with objects df_basket1 through df_basket4, c, and df_itemslist.
- Values pane:** Shows the values for each object, such as df_basket1 being a formal class rules object with 5 observations and 5 variables.
- Plots pane:** Empty.
- Packages pane:** Empty.
- Help pane:** Shows the documentation for the flatten function.
- Description pane:** Provides information about flattening nested data frames.
- Usage pane:** Shows the usage of the flatten function.
- Arguments pane:** Shows arguments x and recursive.
- Examples pane:** Shows examples of using flatten on a data frame.
- Console pane:** Shows the command history and output, including the execution of apriori and the resulting frequent itemsets.
- System status bar:** Shows the date (10/21/2017), time (5:35 PM), and system status (ENG).

```

library(arules)
#> Warning message:
#> In apriori(x=x4, parameter = list(supp = s, Conf = c, maxlen = 4, :
#>   With 'maxlen', at most maxlen records with patterns up to a length of 4 returned!
> if(sessionInfo()["basePkgs"]=="tm" | sessionInfo()["otherPkgs"]=="tm"){
+   detach(package:tm, unload=TRUE)
+ }
> inspect(basket4)
lhs          rhs          support      confidence      lift      co
[1] {apple, chocolate, juice} => {milk} 0.0952381 1.0000000 1.4000000 2
[2] {apple, juice, milk}      => {chocolate} 0.0952381 1.0000000 1.6153846 2
[3] {chocolate, juice, milk} => {apple} 0.0952381 1.0000000 4.2000000 2
[4] {apple, chocolate, milk}  => {juice} 0.0952381 0.6666667 7.0000000 2
[5] {beer, diaper, fruits}   => {milk} 0.1904762 1.0000000 1.4000000 4
[6] {diaper, fruits, milk}   => {beer} 0.1904762 1.0000000 1.4000000 4
[7] {beer, fruits, milk}     => {diaper} 0.1904762 0.5714286 1.0909091 4
[8] {beer, diaper, milk}    => {fruits} 0.1904762 0.8000000 1.6800000 4
[9] {beer, chocolate, fruits}=> {milk} 0.0952381 0.6666667 0.9333333 2
[10] {chocolate, fruits, milk}=> {beer} 0.0952381 0.6666667 0.9333333 2
> df_basket4 <- as(basket4, "data.frame")
> View(df_basket4)

```

lhs	rhs	support	confidence	lift	co
[1]	{apple, chocolate, juice} => {milk}	0.0952381	1.0000000	1.4000000	2
[2]	{apple, juice, milk} => {chocolate}	0.0952381	1.0000000	1.6153846	2
[3]	{chocolate, juice, milk} => {apple}	0.0952381	1.0000000	4.2000000	2
[4]	{apple, chocolate, milk} => {juice}	0.0952381	0.6666667	7.0000000	2
[5]	{beer, diaper, fruits} => {milk}	0.1904762	1.0000000	1.4000000	4
[6]	{diaper, fruits, milk} => {beer}	0.1904762	1.0000000	1.4000000	4
[7]	{beer, fruits, milk} => {diaper}	0.1904762	0.5714286	1.0909091	4
[8]	{beer, diaper, milk} => {fruits}	0.1904762	0.8000000	1.6800000	4
[9]	{beer, chocolate, fruits} => {milk}	0.0952381	0.6666667	0.9333333	2
[10]	{chocolate, fruits, milk} => {beer}	0.0952381	0.6666667	0.9333333	2

Result for Database4 for frequent -4 itemsets

```
basket4<-apriori(txn4,parameter = list(supp = s, conf = c,maxlen=5,minlen=5))
```

```
if(sessionInfo()['basePkgs']=='tm' | sessionInfo()['otherPkgs']=='tm'){


```

```
  detach(package:tm, unload=TRUE)

}
```

```
}
```

```
inspect(basket4)
```

```
df_basket4 <- as(basket4,"data.frame")
```

```
View(df_basket4)
```

The screenshot shows the RStudio interface with the following details:

- R Script:** The code is displayed in the top-left pane, starting with `basket4<-apriori(txn4,...` and ending with `View(df_basket4)`.
- Global Environment:** The top-right pane shows the environment structure:
 - Data:** df_basket1, df_basket2, df_basket3, df_basket4, df_basket5 (all 0 obs. of 0 variables).
 - values:** basket1, basket2, basket3, basket4, basket5, c, df_item.list, s.
- Help Browser:** The bottom-right pane displays the help page for `flatten`:
 - Description:** In a nested data frame, one or more of the columns consist of another data frame. These structures frequently appear when parsing JSON data from the web. We can flatten such data frames into a regular 2 dimensional tabular structure.
 - Usage:** `flatten(x, recursive = TRUE)
 - Arguments:** `x` – a data frame; `recursive` – flatten recursively.
 - Examples:** A code example using `options(stringsAsFactors=FALSE)` and creating data frames for vehicles.

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Transactions4.R* df_basket4* df_basket3* df_basket2* df_basket1* Filter

No data available in table

Environment History Global Environment

Data

- df_basket1 5 obs. of 5 variables
- df_basket2 0 obs. of 0 variables
- df_basket3 0 obs. of 0 variables
- df_basket4 0 obs. of 0 variables
- df_basketset 10 obs. of 1 variable

Values

- basket1 Formal class rules
- basket2 Formal class rules
- basket3 Formal class rules
- basket4 Formal class rules
- c 0.5
- df_items.list "items.list"
- s 0.05

Files Plots Packages Help Viewer

R: Flatten nested data frames Find in Topic

flatten (jsonlite) R Documentation

Flatten nested data frames

Description

In a nested data frame, one or more of the columns consist of another data frame. These structures frequently appear when parsing JSON data from the web. We can flatten such data frames into a regular 2 dimensional tabular structure.

Usage

```
flatten(x, recursive = TRUE)
```

Arguments

- x a data frame
- recursive flatten recursively

Examples

```
options(stringsAsFactors=FALSE)
x <- data.frame(driver = c("Bosser", "Peach"), occupation = c("Kops"
x$vehicle <- data.frame(model = c("Kirana Fowler", "Royal Racer"))
x$vehicle$stats <- data.frame(speed = c(55, 34), weight = c(67, 24),
size)
x$vehicle$stats$w1
x$vehicle$stats$w2)
```

Console

Algoirthmic control:
filter heap memory load sort verbose
0.1 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 1

```
set item appearances [0 item(s)] done [0.00s].
set transaction size [1 item(s)] done [0.00s].
sorting and recording items [10 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing frequent items [done [0.00s].
creating S4 object ... done [0.00s].
> if(sessionInfo()['basePkgs']=='tm' | sessionInfo()['otherPkgs']=='tm'){
+ }
> inspect(basket4)
> df_basket4 <- as(basket4,"data.frame")
> View(df_basket4)
> |
```

Type here to search

Database5: Filename- Transaction5

```

df_basketset <- read.csv("Basket_set1.csv")

if(sessionInfo()['basePkgs']=="dplyr" | sessionInfo()['otherPkgs']=="dplyr"){

  detach(package:dplyr, unload=TRUE)

}

library(plyr)

library(arules)

txn5 = read.transactions(file="Transaction5.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)

txn5@itemInfo$labels <- gsub("\","",txn5@itemInfo$labels)

s <- readline(prompt="Enter Support: ")

c <- readline(prompt="Enter confidence: ")

s <- as.numeric(unlist(strsplit(s, ",")))

c <- as.numeric(unlist(strsplit(c, ",")))

```

The screenshot shows the RStudio interface with the following details:

- File Menu:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Code Editor:** A script named "Basket_set1.R" is open, containing the R code provided above.
- Environment Pane:** Shows the global environment with objects like df_basket1 through df_basket5, s, c, and df_item.list.
- Values Pane:** Shows the values for each object, such as df_basket1 having 5 obs. of 5 variables.
- Help Pane:** The "Flatten nested data frames" help page is displayed, explaining how to flatten nested data frames.
- Console:** The console output shows the execution of the R code, including user input for support and confidence values.
- System Tray:** Shows the date (10/21/2017), time (5:44 PM), and battery status.

Result for Database5 for frequent -2 itemsets

```
basket5<-apriori(txn5,parameter = list(supp = s, conf = c,maxlen=3,minlen=3))
```

```
if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){


```

```
    detach(package:tm, unload=TRUE)

}
```

```
inspect(basket5)
```

```
df_basket5 <- as(basket5,"data.frame")
```

```
View(df_basket5)
```

The screenshot shows the RStudio interface with the following details:

- Script Editor:** Contains the R code for generating frequent itemsets and inspecting the results.
- Environment View:** Shows the global environment with objects like df_basket1 through df_basket5, basket1 through basket5, and df_items.list.
- Help View:** Displays the documentation for the flatten function, which is used to flatten nested data frames.
- Console:** Shows the command history and output, including the execution of the apriori function and its parameters.
- Taskbar:** Shows various open files and the current workspace.
- System Tray:** Shows the date and time (10/21/2017, 5:45 PM) and system status.

```
4
5: if(sessionInfo()['basePkgs']=="dplyr" | sessionInfo()['otherPkgs']=="dplyr"){
6:   detach(package:dplyr, unload=TRUE)
7: }
8: library(plyr)
9: library(arules)
10: txm5 = read.transactions(file="Transactions5.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)
11: txm5$itemInfo$labels <- gsub("\\" , "",txm5$itemInfo$labels)
12: 
13: s <- readline(prompt="Enter Support: ")
14: c <- readline(prompt="Enter confidence: ")
15: 
16: 
17: s <- as.numeric(unlist(strsplit(s, ",")))
18: c <- as.numeric(unlist(strsplit(c, ",")))
19: 
20: 
21: basket5.apriori(txm5,parameter = list(supp = s, conf = c,maxlen=3,minlen=3))
22: 
23: 
24: basket5.apriori(txm5,parameter = list(supp = s, conf = c,maxlen=3,minlen=3))
25: 
26: 
27: if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){
28:   detach(package:tm, unload=TRUE)
29: }
30: 
31: inspect(basket5)
32: 
33: df_basket5 <- as(basket5,"data.frame")
34: view(df_basket5)
35: 
36: 
37: 
38: 
39: 
40: 
41: (top Level :)
```

```
Parameter specification:
confidence maxval maxarm  aval originalSupport maxtime support minlen maxlen target ext
          0.6      0.1     1 none FALSE           TRUE      5  0.06     3     3 rules FALSE

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

Absolute minimum support count: 1

set Item appearances: [0 item(s)] done [0.00s].
set Transaction tree: [1 item(s)] done [0.00s].
sorting and recoding items... [9 item(s)] done [0.00s].
creating transaction tree... done [0.00s].
check for single item patterns... done [0.00s].
writing... [15 rule(s)] done [0.00s].
creating set object ... done [0.00s].
warning message:
In apriori(txm5, parameter = list(supp = s, conf = c, maxlen = 3, :
  Mining stopped (maxlen reached). only patterns up to a length of 3 returned!
> |
```

```

Showing 1 to 15 of 15 entries

Console / / 
+ devtools::package_load("arules")
> 
> inspect(baskets)
lhs          rhs          support  confidence   lift    count
[1] {chocolate,yogurt} => {milk} 0.0952381 1.0000000 1.4000000 2
[2] {milk,yogurt}        => {chocolate} 0.0952381 0.6666667 1.2727273 2
[3] {bread,juice}         => {chocolate} 0.0952381 1.0000000 1.9090909 2
[4] {bread,chocolate}    => {juice}   0.0952381 0.6666667 2.0000000 2
[5] {bread,diaper}        => {chocolate} 0.0952381 0.6666667 1.2727273 2
[6] {bread,diaper}        => {diaper}   0.0952381 0.6666667 1.4000000 2
[7] {beer,fruits}         => {milk}   0.0952381 0.6666667 0.9333333 2
[8] {diaper,fruits}       => {milk}   0.0952381 0.6666667 0.9333333 2
[9] {beer,juice}          => {chocolate} 0.0952381 1.0000000 1.9090909 2
[10] {diaper,juice}        => {chocolate} 0.0952381 1.0000000 1.9090909 2
[11] {chocolate,juice}     => {milk}   0.1904762 0.8000000 1.1200000 4
[12] {juice,milk}          => {chocolate} 0.1904762 0.6666667 1.2727273 4
[13] {beer,chocolate}      => {milk}   0.0952381 1.0000000 1.4000000 2
[14] {chocolate,diaper}    => {milk}   0.1904762 0.6666667 0.9333333 4
[15] {diaper,milk}         => {chocolate} 0.1904762 0.6666667 1.2727273 4
> df_basket5 <- as(basket5, "data.frame")
> View(df_basket5)
> 

```

lhs	rhs	support	confidence	lift	count
[1] {chocolate,yogurt}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[2] {milk,yogurt}	=> {chocolate}	0.0952381	0.6666667	1.2727273	2
[3] {bread,juice}	=> {chocolate}	0.0952381	1.0000000	1.9090909	2
[4] {bread,chocolate}	=> {juice}	0.0952381	0.6666667	2.0000000	2
[5] {bread,diaper}	=> {chocolate}	0.0952381	0.6666667	1.2727273	2
[6] {bread,diaper}	=> {diaper}	0.0952381	0.6666667	1.4000000	2
[7] {beer,fruits}	=> {milk}	0.0952381	0.6666667	0.9333333	2
[8] {diaper,fruits}	=> {milk}	0.0952381	0.6666667	0.9333333	2
[9] {beer,juice}	=> {milk}	0.1428571	1.0000000	1.4000000	3
[10] {diaper,juice}	=> {chocolate}	0.0952381	1.0000000	1.9090909	2
[11] {chocolate,juice}	=> {milk}	0.1904762	0.8000000	1.1200000	4
[12] {juice,milk}	=> {chocolate}	0.1904762	0.6666667	1.2727273	4
[13] {beer,chocolate}	=> {milk}	0.0952381	1.0000000	1.4000000	2
[14] {chocolate,diaper}	=> {milk}	0.1904762	0.6666667	0.9333333	4
[15] {diaper,milk}	=> {chocolate}	0.1904762	0.6666667	1.2727273	4

Result for Database5 for frequent -3 itemsets

```
basket5<-apriori(txn5,parameter = list(supp = s, conf = c,maxlen=4,minlen=4))
```

```
if(sessionInfo()['basePkgs']=="tm" | sessionInfo()['otherPkgs']=="tm"){


```

```
detach(package:tm, unload=TRUE)
```

```
}
```

```
inspect(basket5)
```

```
df_basket5 <- as(basket5,"data.frame")
```

```
View(df_basket5)
```

The screenshot shows the RStudio interface with the following details:

- File Menu:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Session Bar:** kanta.R, final.R, Transactions.R, df_basket1, df_basket2, df_basket3, df_basket4, df_basket5, df_basket6.
- Code Editor:** Contains the R script for generating frequent itemsets and creating a data frame from the basket5 object.
- Environment View:** Shows the global environment with objects like df_basket1 through df_basket5 and c.
- Data View:** Shows the structure of the data frames, indicating they have 5 variables each.
- Plots View:** Empty.
- Packages View:** Empty.
- Help View:** Empty.
- Viewer View:** Shows the output of the 'View(df_basket5)' command, displaying the contents of the data frame.
- Bottom Status Bar:** Shows the date and time (5:47 PM, 10/21/2017) and system information (ENG).

