R 言語による相関ルール分析

- 今回は，arules ライブラリを利用する．R 言語上で下記のコマンドを実行し，ライブラリをインストールすること．なお，インストール中に幾つか質問がされるが，内容を確認し，問題がなければ，yes と回答すれば良い．

  # ダウンロード& インストールには，それなりの時間（60 分以上）が必要となることがあります．

```
> install.packages("arules", dependencies=TRUE)
```

1 データの読み込み

● データの準備と読み込み

- 「1 行 1 トランザクション，かつ，トランザクション中の各アイテムをカンマ (",") 区切りで並べる」形式を採用する．
- トランザクションデータの読み込みには，ライブラリ"arules"に含まれる"read.transactions()"を

```
変数名 <-read.transactions("ファイル名", format="basket", sep=",")
```

の形式で利用する．

● データの表示

- 読み込んだデータは，下記の書式で内容を確認できる．

```
as(変数名, "data.frame")
```

- データ数が多いときは，"head"コマンドを利用し，上から n 数行だけ表示することもできる．
- 一方，"summary"を使って，データの情報を確認することも可能である．

```
> library(arules)  #ライブラリに読み込み
 要求されたパッケージ Matrix をロード中です
  <略>

> alcohol <- read.transactions("alcohol.dat", format="basket", sep=",")  #データの読み込み
>
> alcohol
transactions in sparse format with
 245 transactions (rows) and
 5 items (columns)
>
> as(alcohol,"data.frame")
                               items
1            {beer,distilled_spirit}
2                           {wine}
3                {sake,whiskey,wine}
4  {beer,distilled_spirit,sake,wine}
5            {distilled_spirit,wine}
6                           {wine}
7           {beer,sake,whiskey,wine}
<略>
>
> head(as(alcohol,"data.frame"),5) #5 行分だけ表示
                               items
1           {beer,distilled_spirit}
2                          {wine}
3               {sake,whiskey,wine}
4 {beer,distilled_spirit,sake,wine}
5           {distilled_spirit,wine}
>
> summary(alcohol) #データ数とアイテム種数を表示
transactions as itemMatrix in sparse format with
 245 rows (elements/itemsets/transactions) and
 5 columns (items) and a density of 0.5134694
<略>
```

2 頻度に基づく棒グラフの作成

- データを扱う際には，まずデータの概観を把握することも重要である．
- "arules"には，各アイテムの出現回数（頻度）を棒グラフで表示するコマンドとして，"itemFrequencyPlot"が準備されている．

itemFrequencyPlot(変数名, type="absolute or relative")

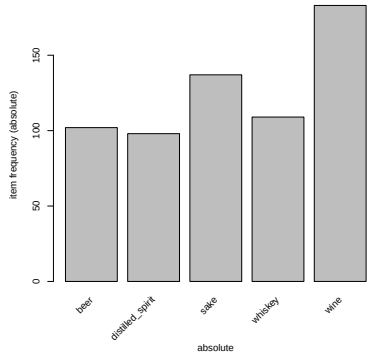
"type"には，"absolute"または"relative"のいずれかを指定する．

- type = "absolute"：絶対度数（出現数）
- type = "relative"：相対度数（出現率）

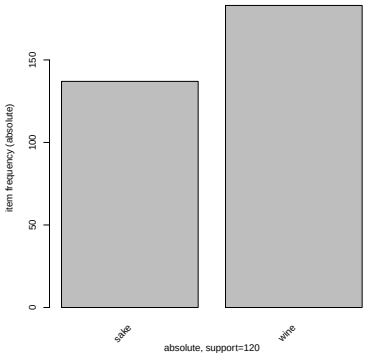
- また，オプション support や topN を用いて度数が低いアイテムを削除することが可能である．
  - support = 数値：数値以上の出現数（or 出現率）を持つアイテムのみを表示
  - topN = 数値：出現数上位 N アイテムのみ表示

- 以下に，具体例を示す．

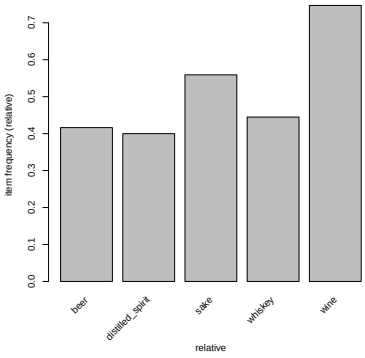
```
>
> itemFrequencyPlot(alcohol, type = "absolute") #出現数で棒グラフを作る⇒ (a)
> itemFrequencyPlot(alcohol, type = "absolute", support = 120) #出現数 120 以上のアイテムのみ出力⇒ (b)
> itemFrequencyPlot(alcohol, type = "relative") #出現率で棒グラフを作成⇒ (c)
> itemFrequencyPlot(alcohol, type = "relative", support=0.5) #出現率 0.5 以上のアイテムのみ出力⇒ (d)
> itemFrequencyPlot(alcohol, type = "relative", topN = 3) #出現数上位 3 件のみを表示⇒ (e)
>
```



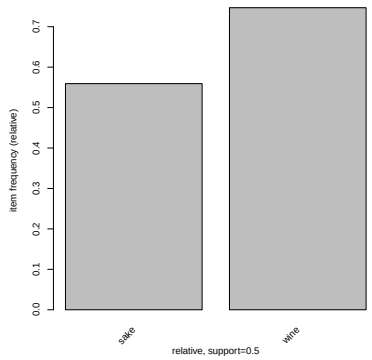
(a)



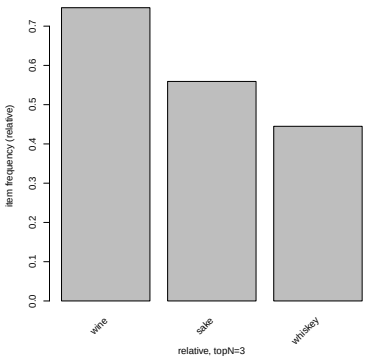
(b)



(c)



(d)



(e)

3 頻出アイテム集合の抽出

頻出アイテム集合の発見には，(1) 深さ優先探索に基づく Eclat，もしくは (2) 階層的探索（幅優先探索）に基づく Apriori が利用できる．ここでは，Eclat による頻出アイテム集合の抽出手順を示す．

● Eclat による頻出アイテム集合の抽出

- eclat による頻出パターン発見の書式は，以下の通り．

変数名<- eclat(データ, parameter = list(パラメタ))

- パラメタには，以下の 4 種が指定可能である．なお，複数のパラメタを設定する場合はカンマ (",") で区切る．また，指定しないパラメタはデフォルト値が利用される．

  最小支持度： `support = 最小支持度` の形式で指定する．デフォルト値は 0.1 である．例えば，最小支持度を 0.02 にする場合は，`support = 0.02` とする．

  最小サイズ： `minlen = 最小サイズ` の形式で指定する．これにより，最小サイズ 以上の（頻出）アイテム集合が抽出対象となる．デフォルト値は 1 である．例えば，サイズ 2 以上の頻出アイテム集合を抽出するには，`minlen = 2` とする．

  最大サイズ： `maxlen = 最大サイズ` の形式で指定する．これにより，最大サイズ 以下の（頻出）アイテム集合が抽出対象となる．デフォルト値は 10 である．例えば，サイズ 20 以下の頻出アイテム集合を抽出するには，`maxlen = 20` とする．※デフォルト値が 10 なので，10 を超えるサイズのアイテム集合を獲得するには必ずこのパラメタを設定する必要がある．

  ターゲット： `target = ターゲット` の形式で，抽出する頻出アイテム集合の種類を指定する．ターゲット には，`"frequent itemsets"`(頻出アイテム集合)，`"maximally frequent itemsets"`（極大頻出アイテム集合），`"closed frequent itemsets"`（飽和頻出アイテム集合）のいずれかを指定する．

- 以下に，頻出パターン抽出例を示す．同じデータを対象にしても，頻出パターン，飽和パターン，極大パターンで，得られるパターン数が異なることを確認しよう．

```
> library(arules) #ライブラリの読み込み
>
> retail <- read.transactions("retail.dat",format="basket",sep=",")
> as(retail, "data.frame")
                        items
1        {bread,butter,milk}
2           {bread,jam,milk}
3           {margarine,milk}
4             {bread,butter}
5    {bread,butter,jam,milk}
6                {margarine}
7 {bread,jam,margarine,milk}
8                       {jam}
>
>
> retail.freq <- eclat(retail, parameter = list(support=3/8, target="frequent itemsets"))
  #最小支持度 3/8 で頻出アイテム集合を発見
> <略>
> retail.freq  #結果を表示　（具体的な内容の表示は，後述）
> set of 10 itemsets  #10 の頻出アイテム集合が獲得された
>
> retail.closed <- eclat(retail, parameter = list(support=3/8, target="closed frequent itemsets"))
 #最小支持度 3/8 で飽和頻出アイテム集合を発見
> <略>
> retail.closed  #結果を表示
set of 7 itemsets  #7 の頻出アイテム集合が獲得された
>
> retail.max <- eclat(retail, parameter = list(support=3/8, target="maximally frequent itemsets"))
 #最小支持度 3/8 で極大頻出アイテム集合を発見
> <略>
> retail.max  #結果を表示
set of 3 itemsets  #3 の極大頻出アイテム集合が獲得された
```

● 結果の表示

- 結果の表示には，関数”inspect”を書式 inspect(抽出されたパターン集合) に従い利用する．

```
>
> inspect( retail.max )    #先ほど抽出した極大パターンの表示
    items            support count
[1] {bread,butter}   0.375   3
[2] {bread,jam,milk} 0.375   3
[3] {margarine}      0.375   3
>
> inspect( retail.closed )  #先ほど抽出した飽和パターンの表示
    items            support count
[1] {bread,butter}   0.375   3
[2] {bread,jam,milk} 0.375   3
[3] {bread,milk}     0.500   4
[4] {bread}          0.625   5
[5] {milk}           0.625   5
[6] {jam}            0.500   4
[7] {margarine}      0.375   3
```

● 支持度による並べ替えとフィルタリング

- 関数”sort”を利用することで，抽出されたパターン集合 を，支持度（頻度）の大きい順（もしくは小さい順）に並べ替えることができる．加えて，関数”head”を用いることで,（並べかえたパターン集合の）上から n 件のみを抽出することができる．”sort”と”head”の書式はそれぞれ以下の通りである．

sort(パターン集合, by="support" [, decreasing=FALSE])　　head(パターン集合, n)

- 並べかえたパターン集合や，抽出結果を，”inspect”へ渡すことで，実際の結果を見ることが可能である．
- 以下に例を示す．

```
>
> inspect(  sort(retail.closed, by="support")  ) #先ほど抽出した飽和パターンを頻度の大きい順に表示
    items            support count
[1] {bread}          0.625   5
[2] {milk}           0.625   5
[3] {bread,milk}     0.500   4
[4] {jam}            0.500   4
[5] {bread,butter}   0.375   3
[6] {bread,jam,milk} 0.375   3
[7] {margarine}      0.375   3
>
> inspect(  sort(retail.closed, by="support", decreasing=FALSE)  ) #頻度の小さい順位並べ替えることも可能
    items            support count
[1] {bread,butter}   0.375   3
[2] {bread,jam,milk} 0.375   3
[3] {margarine}      0.375   3
[4] {bread,milk}     0.500   4
[5] {jam}            0.500   4
[6] {bread}          0.625   5
[7] {milk}           0.625   5
>
> inspect( head( sort(retail.closed, by="support"), 3)  ) #飽和パターンを，頻度の大きい順に 3 つだけ抽出＆表示
    items        support count
[1] {bread}      0.625   5
[2] {milk}       0.625   5
[3] {bread,milk} 0.500   4
>
```

● 高度なフィルタリング

- ”subset”関数を利用することで，(1) 支持度，(2) サイズ，(3) 含まれるアイテムに基づいたフィルタリングが実現できる．

- ”subset”の書式は以下の通りである．

変数名<- subset(ルール集合, subset = 条件式)

条件式 の指定は，条件

  - items %in% "アイテム"：パターンが”アイテム” を含む
  - support > 数値：支持度が 数値 を超える
  - size(items) > 数値:パターンのサイズが 数値 を超える

の&(アンド) や|（オア）が指定可能である．加えて，数値との比較に関しては，(”>”(超える) だけでなく）”>=”, や”<”なども指定可能である．

- 以下に具体例を示す．

```
>
> inspect( subset( retail.freq, subset = items %in% "milk" ) )
  #"milk"を含むパターンのみを抽出
    items             support count
[1] {bread,jam,milk} 0.375   3
[2] {jam,milk}       0.375   3
[3] {bread,milk}     0.500   4
[4] {milk}           0.625   5
>
>
> inspect( subset( retail.freq, subset = items %in% "milk" & size(items) == 2) )
  #milk を含む，サイズ 2 のパターンのみを抽出
    items        support count
[1] {jam,milk}   0.375   3
[2] {bread,milk} 0.500   4
>
> inspect( subset( retail.freq, subset = items %in% "milk" | items %in% "jam") )
  #"milk" または "jam" を含むパターンのみを抽出
    items             support count
[1] {bread,jam,milk} 0.375   3
[2] {bread,jam}      0.375   3
[3] {jam,milk}       0.375   3
[4] {bread,milk}     0.500   4
[5] {milk}           0.625   5
[6] {jam}            0.500   4
>
> inspect( subset( retail.freq, subset = size(items) >= 2) )
  #サイズ 2 以上のパターンを抽出
    items             support count
[1] {bread,butter}   0.375   3
[2] {bread,jam,milk} 0.375   3
[3] {bread,jam}      0.375   3
[4] {jam,milk}       0.375   3
[5] {bread,milk}     0.500   4
>
> inspect( subset( retail.freq, subset = size(items) >= 2 & support < 0.5) )
  #サイズ 2 以上でかつ支持度 0.5 未満のパターンのみ抽出
    items             support count
[1] {bread,butter}   0.375   3
[2] {bread,jam,milk} 0.375   3
[3] {bread,jam}      0.375   3
[4] {jam,milk}       0.375   3
```

4 相関ルールの抽出

相関ルールの抽出には，関数”apriori”を利用する．

● Apriori による相関ルールの抽出

- apriori による相関ルール発見の書式は，以下の通りである．

```
変数名<- apriori(データ, parameter = list(パラメタ))
```

パラメタには，以下の 4 種が指定可能である．eclat の場合と同様，複数のパラメタを設定する場合はカンマ（”,”）で区切る．

最小支持度： `support = 最小支持度` の形式で指定する．デフォルト値は 0.1 である．

最小確信度： `confidence = 最小確信度` の形式で指定する．デフォルト値は 0.8 である．

最大サイズ： `maxlen = 最大サイズ` の形式で指定する．ルールの前提部と帰結部を合わせたサイズが 最大サイズ 以下の相関ルールが抽出対象となる．デフォルト値は 5 である．

最小サイズ： `minlen = 最小サイズ` の形式で指定する．ルールの前提部と帰結部を合わせたサイズが 最小サイズ 以上の相関ルールが抽出対象となる．

● 結果の表示

- 得られた相関ルールの表示には，頻出パターン同様，”inspect”を用いる．
- ”inspect”による表示は，以下の項目で構成される．
  - lhs：ルールの前提部（left hand side）
  - rhs：ルールの帰結部（right hand side）
  - support：ルールの支持度
  - confidence：ルールの確信度
  - lift：ルールのリフト値
- 以下に具体例を示す..

```
>
> library(arules)
> retail.rules <- apriori(retail, parameter=list(support=3/8, confidence=0.8 ) )
  #最小支持度 3/8，最小確信度 0.8，最大サイズ 5 で相関ルールを抽出
>
> inspect(retail.rules) #結果の表示
    lhs            rhs     support confidence lift count
[1] {butter}    => {bread} 0.375   1.0        1.60 3
[2] {milk}      => {bread} 0.500   0.8        1.28 4
[3] {bread}     => {milk}  0.500   0.8        1.28 4
[4] {jam,milk}  => {bread} 0.375   1.0        1.60 3
[5] {bread,jam} => {milk}  0.375   1.0        1.60 3
>
> retail.rules2 <- apriori(retail, parameter=list(support=3/8, confidence=0.8,minlen=2,maxlen=2))
# 最小支持度 3/8，最小確信度 0.8 を満たす，サイズ 2 の相関ルールを抽出（minlen=maxlen=2 とすることでサイズを限定）
>
> inspect(retail.rules2)
    lhs         rhs     support confidence lift count
[1] {butter} => {bread} 0.375   1.0        1.60 3
[2] {milk}   => {bread} 0.500   0.8        1.28 4
[3] {bread}  => {milk}  0.500   0.8        1.28 4
>
```

● 並べ替えとフィルタリング

- eclat の場合と同様，"sort"を用いてデータを並べかえることが可能である．なお，並べかえの基準として，(1)"support"（支持度），(2)"confidence"（確信度），(3)"lift"（リフト値）を用いることができる．また，"head"と合わせて利用することで，評価値の高いルールを n 件だけ抽出することも可能である．

```
>
> inspect( sort(retail.rules, by="support") ) #支持度で並べ替え
    lhs            rhs     support confidence lift count
[1] {milk}      => {bread} 0.500   0.8        1.28 4
[2] {bread}     => {milk}  0.500   0.8        1.28 4
[3] {butter}    => {bread} 0.375   1.0        1.60 3
[4] {jam,milk}  => {bread} 0.375   1.0        1.60 3
[5] {bread,jam} => {milk}  0.375   1.0        1.60 3
>
> inspect( sort(retail.rules, by="confidence") ) #確信度で並べ替え
    lhs            rhs     support confidence lift count
[1] {butter}    => {bread} 0.375   1.0        1.60 3
[2] {jam,milk}  => {bread} 0.375   1.0        1.60 3
[3] {bread,jam} => {milk}  0.375   1.0        1.60 3
[4] {milk}      => {bread} 0.500   0.8        1.28 4
[5] {bread}     => {milk}  0.500   0.8        1.28 4
>
> inspect( head( sort(retail.rules, by="lift" ),  3)  )  #リフトで並べ替え + 上位 3 件のみ
    lhs            rhs     support confidence lift count
[1] {butter}    => {bread} 0.375   1          1.6  3
[2] {jam,milk}  => {bread} 0.375   1          1.6  3
[3] {bread,jam} => {milk}  0.375   1          1.6  3
```

● フィルタリング

- フィルタリングには，"subset"を用いる．
- 頻出アイテム集合のときは，"items"を利用したが，相関ルールの場合は，items（ルールを構成するアイテム集合），lhs（ルールの条件部），rhs（ルールの帰結部），support（支持度），confidence（確信度），lift（リフト値），size（大きさ）を利用することが可能である．
- また，集合内にあるアイテムが含まれていることを要請する%in%に加え，ある文字列を 含む アイテムが含まれていることを要請する%pin%も利用できる．（%in%は完全一致，%pin%は「部分文字列」ということである）．
- 以下に例を示す．

```
>
> inspect( subset(retail.rules, subset = size(rhs)==1) ) ルールの帰結部 (rhs) の要素数が 1 のルールのみを抽出する
    lhs            rhs     support confidence lift count
[1] {butter}    => {bread} 0.375   1.0        1.60 3
[2] {milk}      => {bread} 0.500   0.8        1.28 4
[3] {bread}     => {milk}  0.500   0.8        1.28 4
[4] {jam,milk}  => {bread} 0.375   1.0        1.60 3
[5] {bread,jam} => {milk}  0.375   1.0        1.60 3
>
> inspect(subset(retail.rules, subset = lhs %in% "milk" | rhs %in% "bread" ) )
  #ルールの前提部に milk を持つか，帰結部に bread を持つルールのみを抽出する
    lhs           rhs     support confidence lift count
[1] {butter}   => {bread} 0.375   1.0        1.60 3
[2] {milk}     => {bread} 0.500   0.8        1.28 4
[3] {jam,milk} => {bread} 0.375   1.0        1.60 3
>
> inspect(subset(retail.rules, subset = support >= 0.5 & lift > 1.5) )
  #支持度 0.5 以上かつリフト値 1.5 を超えるルールを抽出する
  <該当するルールがないので，何も表示されない>
>
> inspect(subset(retail.rules, subset = items %in% "milk" & confidence >= 0.9) )
  #ルール内に milk を持つ，確信度 0.9 以上のルールを抽出する
    lhs            rhs     support confidence lift count
[1] {jam,milk}  => {bread} 0.375   1          1.6  3
[2] {bread,jam} => {milk}  0.375   1          1.6  3
>
```

- "sort"と"head"，"subset"を組み合わせることで，条件を満たすルールのうち，評価値の高いものを上位 n 件だけ抽出することも可能である．
- 例えば以下の例では,「最も Lift 値が高い"milk"と"bread"を含むルール」を抽出している．

```
>
>inspect( head( sort( subset(retail.rules, subset = items %in% "milk" & items %in% "bread"), by="lift"), 1)  )
    lhs           rhs     support confidence lift count
[1] {jam,milk} => {bread} 0.375   1          1.6  3
>
>
> #途中の結果を変数に入れてることで，同じ操作を実現
> retail.milkbread <- subset(retail.rules, subset=items %in% "milk" & items %in% "bread")
> inspect( head( sort( retail.milkbread, by="lift" ), 1)  )
    lhs           rhs     support confidence lift count
[1] {jam,milk} => {bread} 0.375   1          1.6  3
>
```

● 参考：Apriori による頻出パターンの発見

- "apriori"は，実行時に target 属性を parameter に指定することで，相関ルールの他にも，頻出パターンを求めることに利用することができる．
- 指定可能な target は，
  - "frequent itemsets"(頻出アイテム集合)
  - "maximally frequent itemsets"（極大頻出アイテム集合）
  - "closed frequent itemsets"（飽和頻出アイテム集合）
  - "rules" (相関ルール)

  のいずれかである．
- 以下に，具体例を示す．

```
>
> inspect( apriori(retail, parameter=list(target="maximally frequent itemsets", support=3/8)) ) #最小支
持度 3/8 で極大パターンを抽出
<略>
    items            support count
[1] {margarine}      0.375   3
[2] {bread,butter}   0.375   3
[3] {bread,jam,milk} 0.375   3
>
> inspect( apriori(retail, parameter=list(target="rules", support=3/8,confidence=0.8)) )
   #最小支持度 3/8, 最小確信度 0.8 で相関ルールを抽出
<略>
    lhs             rhs     support confidence lift count
[1] {butter}    => {bread} 0.375   1.0        1.60 3
[2] {milk}      => {bread} 0.500   0.8        1.28 4
[3] {bread}     => {milk}  0.500   0.8        1.28 4
[4] {jam,milk}  => {bread} 0.375   1.0        1.60 3
[5] {bread,jam} => {milk}  0.375   1.0        1.60 3
>
```