DEPARTMENT OF

CHANDIGARH UNIVERSITY

Discover. Learn. Empower.

Experiment 1.3

Student Name: Mayank Kumar **UID:20BCS1353**
Branch: BE CSE **Section/Group:DM_705-A**
Semester:6th **Date ofPerformance:27/02/2023**
Subject Name: Data Mining **Subject Code: 20CSP-376**

Aim:

To demonstrate association rule mining using apriory algorithm on supermarket data.

Objective:

To determine the association rule mining of file using R Studio and displaying the pattern on Weka Tool for further extraction and analysis of knowledge.

Code:

```
# Apriori Algorithm library(arules)

library(arulesViz) library(RColorBrewer)


# import dataset data("Groceries")


# using apriori() function rules <-

apriori(Groceries,

        parameter = list(supp = 0.01, conf = 0.2))
```

```
# using inspect() function

inspect(rules[1:10])


# using itemFrequencyPlot() function

arules::itemFrequencyPlot(Groceries, topN = 20,

                          col = brewer.pal(8, 'Pastel2'),

                          main = 'Relative Item Frequency Plot',

                          type = "relative",

                          ylab = "Item Frequency (Relative)")
```

Output:

```
> # Apriori Algorithm
>
> library(arules)
> library(arulesViz)
> library(RColorBrewer)
>
> # import dataset
> data("Groceries")
>
> # using apriori() function
> rules <- apriori(Groceries,
+                  parameter = list(supp = 0.01, conf = 0.2))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.2    0.1    1 none FALSE            TRUE       5    0.01      1     10  rules TRUE

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

Absolute minimum support count: 98

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
sorting and recoding items ... [88 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [232 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
>
```

```
> # using inspect() function
> inspect(rules[1:10])
     lhs                rhs                support    confidence coverage   lift     count
[1]  {}              => {whole milk}       0.25551601 0.2555160  1.00000000 1.000000 2513
[2]  {hard cheese}   => {whole milk}       0.01006609 0.4107884  0.02450432 1.607682   99
[3]  {butter milk}   => {other vegetables} 0.01037112 0.3709091  0.02796136 1.916916  102
[4]  {butter milk}   => {whole milk}       0.01159126 0.4145455  0.02796136 1.622385  114
[5]  {ham}           => {whole milk}       0.01148958 0.4414062  0.02602949 1.727509  113
[6]  {sliced cheese} => {whole milk}       0.01077783 0.4398340  0.02450432 1.721356  106
[7]  {oil}           => {whole milk}       0.01128622 0.4021739  0.02806304 1.573968  111
[8]  {onions}        => {other vegetables} 0.01423488 0.4590164  0.03101169 2.372268  140
[9]  {onions}        => {whole milk}       0.01209964 0.3901639  0.03101169 1.526965  119
[10] {berries}       => {yogurt}           0.01057448 0.3180428  0.03324860 2.279848  104
>
> # using itemFrequencyPlot() function
> arules::itemFrequencyPlot(Groceries, topN = 20,
+                           col = brewer.pal(8, 'Pastel2'),
+                           main = 'Relative Item Frequency Plot',
+                           type = "relative",
+                           ylab = "Item Frequency (Relative)")
> # Apriori Algorithm
>
> library(arules)
> library(arulesViz)
> library(RColorBrewer)
> # import dataset
> data("Groceries")
> # using apriori() function
> rules <- apriori(Groceries,
+                  parameter = list(supp = 0.01, conf = 0.2))
Apriori


Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.2    0.1    1 none FALSE            TRUE       5    0.01      1     10  rules TRUE

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

Absolute minimum support count: 98

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
sorting and recoding items ... [88 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [232 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
> # using inspect() function
> inspect(rules[1:10])
     lhs                rhs                support    confidence coverage   lift     count
[1]  {}              => {whole milk}       0.25551601 0.2555160  1.00000000 1.000000 2513
[2]  {hard cheese}   => {whole milk}       0.01006609 0.4107884  0.02450432 1.607682   99
[3]  {butter milk}   => {other vegetables} 0.01037112 0.3709091  0.02796136 1.916916  102
[4]  {butter milk}   => {whole milk}       0.01159126 0.4145455  0.02796136 1.622385  114
[5]  {ham}           => {whole milk}       0.01148958 0.4414062  0.02602949 1.727509  113
[6]  {sliced cheese} => {whole milk}       0.01077783 0.4398340  0.02450432 1.721356  106
[7]  {oil}           => {whole milk}       0.01128622 0.4021739  0.02806304 1.573968  111
[8]  {onions}        => {other vegetables} 0.01423488 0.4590164  0.03101169 2.372268  140
[9]  {onions}        => {whole milk}       0.01209964 0.3901639  0.03101169 1.526965  119
[10] {berries}       => {yogurt}           0.01057448 0.3180428  0.03324860 2.279848  104
```

```
> # using itemFrequencyPlot() function
> arules::itemFrequencyPlot(Groceries, topN = 20,
+                           col = brewer.pal(8, 'Pastel2'),
+                           main = 'Relative Item Frequency Plot',
+                           type = "relative",
+                           ylab = "Item Frequency (Relative)")
>
```