Assignment No :B9

Roll No.4431

1 Title:

BAI tool for shopping mall.

2 Problem Definition

A Mall has number of items for sale. Build a required Database to develop BAI tool for considering one aspect of growth to the business Such as organization of products based on demand and patterns use R Programming or other equivalent latest tools used in Industry.

3 Learning Objectives

1. To understand the concept of Business Analytics and Intelligence.
2. To identify BAI opportunities in real world businesses.

4 Learning Outcomes

1. Ability to apply BAI tools to real world business problems.
2. Acquire proficiency over BAI and its theoretical aspects by applying them in practical.

5 Related Mathematics

mathematical model is given as below,

S=X,Y,Mem-shared,DD,NDD,Fme,D,Dsupp,conf,lk,ck,subset(),apriori-gen(),rules

Where,
X=input Datasets of Groceries.
Supp= frequency count of itemset in D dataset support=support(A+B)/N
support=support(A+B)/N
conf=support(A+B)/support(A)
lk=large item set.
ck=set of k itemset
Y=output Bar chart of frequency distribution of grocery items.
Rules:Association rules

Mem-shared=memory shared by the applications.
DD=Deterministic Data Grocery data.
NDD=Non-Deterministic Data.
Fme= subset() To check whether subsets are large item sets or not

6 Concepts related theory

6.1 Algorithm used (Apriori) :

1.Find all itemsets with a specified minimal support (coverage).
An itemset is just a specific set of items, e.g. apples, cheese. The Apriori algorithm can efficiently find all itemsets whose coverage is above a given minimum.

2.Use these itemsets to help generate interersting rules.
Having done stage 1, we have considerably narrowed down the possibilities, and can do reasonably fast processing of the large itemsets to generate candidate rules

Find all large 1-itemsets
2: For (k = 2 ; while Lk-1 is non-empty; k++)
3 Ck = apriori-gen(Lk-1)
4 For each c in Ck, initialise c.count to zero
5 For all records r in the DB Cr = subset(Ck, r); For each c in Cr , c.count++
7 Set Lk := all c in Ck whose count ¿= minsup
8 /* end – return all of the Lk sets.

R programming Introduction:
It is common for today's scientific and business industries to collect large amounts of data, and the ability to analyze the data and learn from it is critical to making informed decisions. Familiarity with software such as R allows users to visualize data, run statistical tests, and apply machine learning algorithms. Even if you already know other software, there are still good reasons to learn R:
1. R is free. If your future employer does not already have R installed, you can always download it for free, unlike other proprietary software packages that require expensive licenses. No matter where you travel, you can have access to R on your computer.
2. R gives you access to cutting-edge technology. Top researchers develop statistical learning methods in R, and new algorithms are constantly added to the list of packages you can download.
3. R is a useful skill. Employers that value analytics recognize R as useful and important. If for no other reason, learning R is worthwhile to help boost your resume.
Note that R is a programming language, and there is no intuitive graphical user interface with buttons you can
click to run different methods. However, with some practice, this kind of environment makes it easy to quickly code scripts and functions for various statistical purposes. To get the most out of this tutorial, follow the examples by typing them out in R on your own computer. A line that begins with ¿ is input at the command prompt. We do not include the output in most cases, but you should try out the commands yourself and see what happens. If you type something at the command line and decide not to execute, press the down arrow to clear the line; pressing the up arrow gives you the previous executed command.

7 Algorithm :

Apriori Algorithm:
1.Find all itemsets with a specified minimal support (coverage). An itemset is just a specific set of items, e.g. apples, cheese. The Apriori algorithm can efficiently find all itemsets whose coverage is above a given minimum.2.Use these itemsets to help generate interersting rules. Having done stage 1, we have considerably narrowed down the possibilities, and can do reasonably fast processing of the large itemsets to generate candidate rules

: Find all large 1-itemsets
2: For (k = 2 ; while Lk-1 is non-empty; k++)

3 Ck = apriori-gen(Lk-1)

4 For each c in Ck, initialise c.count to zero

5 For all records r in the DB Cr = subset(Ck, r); For each c in Cr , c.count++

7 Set Lk := all c in Ck whose count ¿= minsup

8 /* end – return all of the Lk sets.

Figure 1: association rule

the following association rules: (Rules are just for illustrations and understanding of the concept. They might not represent the actuals).
Rule 1: If Milk is purchased, Then Sugar is also purchased.
Rule 2: If Sugar is purchased, Then Milk is also purchased.

Rule 3: If Milk and Sugar is Purchased, Then Coffee powder is also purchased in 60percent of the transactions.
Generally association rules are written in "IF-THEN" format. We can also use the term "antecedent" for IF and "Consequent" for THEN.

From the above rules, we understand the following explicitly:
1. Whenever Milk is purchased, Sugar is also purchased or vice versa.
2. If Milk and Sugar is purchased then coffee powder is also purchased. This is true in 3 out of the 5 transactions. In other words we can say that we have a support of 3 out of 5 transactions for this rule. (60percent possibility).

- Frequent item set :- Item set occurring in high frequency. For example in our Coffee dataset, Milk and sugar combinations occurred in 100percent of the transactions.

- Support:- The support for the rule indicates its impact in terms of overall size. If only a small number of transactions are affected, the rule may be little use. For example, the support of "IF Milk and Sugar THEN Coffee powder" is 3/5 transactions or 60percent of the total transactions.

- Confidence :- It determines the operational usefulness of a rule. Transactions with confidence with more than 50percent will be selected. For example, the confidence of milk, sugar and coffee powder given milk, coffee can be written as Number of transactions that include Milk and Sugar (Antecedent) and Coffee Powder (Consequent) is 3 Number of transactions that contains only Milk and Sugar (Antecedent)) is 5.
  P(Milk and Sugar AND Coffee Powder)/ P (Milk and Sugar) = 3/5 = 60percent

8 Output

Output : R program Introduction: R version 3.0.2 (2013-09-25) – "Frisbee Sailing"
Copyright (C) 2013 The R Foundation for Statistical Computing Platform: x86 - 64-pc-linux-gnu (64-bit) R is free software and comes with ABSOLUTELY NO WARRANTY. You are welcome to redistribute it under certain conditions. Type 'license()' or 'licence()' for distribution details. Natural language support but running in an English localeR is a collaborative project with many contributors. Type 'contributors()' for more information and 'citation()' on how to cite R or R packages in publications.Type 'demo()' for some demos, 'help()' for on-line help, or'help.start()' for an HTML browser interface to help. Type 'q()' to quit R.
: 2+2
[1] 4
: 3+4
[1] 7
: 3/5
[1] 0.6
: log(100,base=10)
[1] 2
: exp(12)

[1] 162754.8
: runif(10)
[1] 0.33166502 0.22091722 0.28724449 0.61601346 0.13121878 0.27327015
[7] 0.22955652 0.09091556 0.04072079 0.05192992
: runif(2) [1] 0.2017608 0.1082742
: plot(runif(10))
: x¡-runif(15)
: plot(x)
: x[4] [1] 0.492467
: plot(x[4])
: x¡-2+2
: y¡-3+3
: s¡-"Hello"
: s [1] "Hello"
: y [1] 6
: x [1] 4
: v¡-c(2,3,4,5)
: v [1] 2 3 4 5
: v1¡-seq(0,1,length=11)

: v1 [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
: v1¡-seq(1,10,length=11)
: v1
[1] 1.0 1.9 2.8 3.7 4.6 5.5 6.4 7.3 8.2 9.1 10.0
: v1¡-seq(1,100,length=100)
: v1
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
17 18
[19] 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
35 36
[37] 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52
53 54
[55] 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70
71 72
[73] 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88
89 90
[91] 91 92 93 94 95 96 97 98 99 100
: 1:10
[1] 1 2 3 4 5 6 7 8 9 10
: v¡-height/weight
Error: object 'height' not found
: v¡-height
Error: object 'height' not found
: v¡-height=1
Error in (v ¡- height) = 1 : could not find function "¡-¡-"
: v¡-v1/v2
Error: object 'v2' not found
: sum(v)
[1] 14
: length(v)

[1] 4
: w¡-mean(v)
: w
[1] 3.5
: a¡-median(v)
: a
[1] 3.5
: sd(v)
[1] 1.290994
: var(v)
[1] 1.666667
: IQR(v)
[1] 1.5
: fivenum(v)
[1] 2.0 2.5 3.5 4.5 5.0
: summery(v)
Error: could not find function ”summery”
: summary(v)

Min.	1st Qu.	Median	Mean	3rd Qu.	Max
2.00	2.75	3.50	3.50	4.25	5.00

: v1¡-c(1,2,3,4)
: v2¡-c(5,6,7,8)
: cbind(v1,v2)
v1 v2
[1,] 1 5
[2,] 2 6
[3,] 3 7
[4,] 4 8
: rbind(v1,v2)
[,1] [,2] [,3] [,4]
v1 1 2 3 4
v2 5 6 7 8
: v¡-seq(from=1,to=10,by=1)
: matrix(v,nrow=3,ncol=4)

[,1] [,2] [,3] [,4]

[1,] 1 4 7 10

[2,] 2 5 8 1

[3,] 3 6 9 2

: matrix(v,nrow=3,ncol=4, byrow=TRUE)

[,1] [,2] [,3] [,4]

[1,] 1 2 3 4

[2,] 5 6 7 8

[3,] 9 10 1 2
: v¡-seq(from=1,to=20,by=1)
: m1¡-matrix(v,nrow=3,ncol=4)
: m1
[,1] [,2] [,3] [,4]
[1,] 1 4 7 10
[2,] 2 5 8 11
[3,] 3 6 9 12
: colnames(m1)¡-c("c1","c2","c3","c4")
: rownames(m1)¡-c("row1","row2","row3")
: m1
c1 c2 c3 c4
row1 1 4 7 10
row2 2 5 8 11
row3 3 6 9 12
: m1[,]
c1 c2 c3 c4
row1 1 4 7 10
row2 2 5 8 11
row3 3 6 9 12
: m1[,"c2"]
row1 row2 row3
4 5 6
: m1[,2]
row1 row2 row3
4 5 6
: m1[2,4]
[1] 11
: length(v)
[1] 20
: nrow(m1)
[1] 3
: ncol(m1)
[1] 4
: : data(Bfox)
: Bfox
partic tfr menwage womwage debt parttime
1946 25.3 3748 25.35 14.05 18.18 10.28
1947 24.4 3996 26.14 14.61 28.33 9.28
1948 24.2 3725 25.11 14.23 30.55 9.51
1949 24.2 3750 25.45 14.61 35.81 8.87
1950 23.7 3669 26.79 15.26 38.39 8.54
1951 24.2 3682 26.33 14.58 26.52 8.84
1952 24.1 3845 27.89 15.66 45.65 8.60

1953 23.8 3905 29.15 16.30 52.99 5.49
1954 23.6 4047 29.52 16.57 54.84 6.67
1955 24.3 4043 32.05 17.99 65.53 6.25
1956 25.1 4092 32.98 18.33 72.56 6.32
1957 26.2 4168 32.25 17.64 69.49 7.30
1958 26.6 4073 32.52 18.16 71.71 8.65
1959 26.9 4100 33.95 18.58 78.89 8.80
1960 27.9 4119 34.63 18.95 84.99 9.39
1961 29.1 4159 35.14 18.78 87.71 10.23
1962 29.9 4134 34.49 18.74 95.31 10.77
1963 29.8 4017 35.99 19.71 104.40 10.84
1964 30.9 3886 36.68 20.06 116.80 11.70
1965 32.1 3467 37.96 20.94 130.99 12.33
1966 33.2 3150 38.68 21.20 135.25 12.18
1967 34.5 2879 39.65 21.95 142.93 13.67
1968 35.1 2681 41.20 22.68 155.47 13.82
1969 36.1 2563 42.44 23.75 165.04 14.91
1970 36.9 2571 42.02 25.63 164.53 15.52
1971 37.0 2503 45.32 26.79 169.63 15.47
1972 37.9 2302 45.61 27.51 190.62 15.85
1973 40.1 2931 45.59 27.35 209.60 15.40
1974 40.6 1875 48.06 29.64 216.66 16.23
1975 42.2 1866 46.12 29.33 224.34 16.71:

:dataset¡-read.csv(”/home/pccoe/Desktop/IRIS.csv”)

Program:
* Shopping Mall Market Basket Analysis for Grocery data set

* Load the libraries
library(arules)
library(arulesViz)
library(datasets)

* Load the data set
data(Groceries)

* show Groceries data set details
Groceries

* Create an item frequency plot for the top 20 items
itemFrequencyPlot(Groceries,topN=20,type=”absolute”)

* Get the rules
rules ¡- apriori(Groceries, parameter = list(supp = 0.001, conf = 0.8))

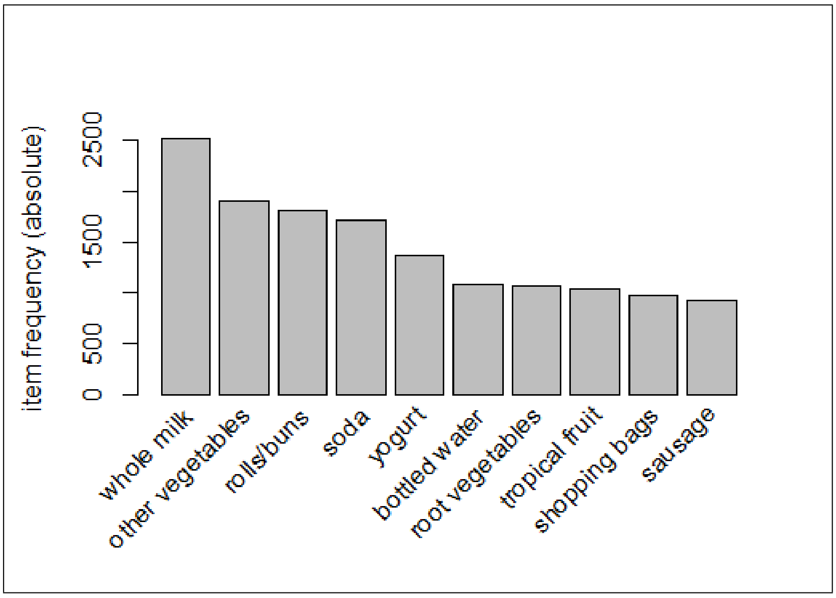
* Show the top 5 rules, but only 2 digits
options(digits=2)
inspect(rules[1:5])
output:

: library(arules)
: library(datasets)
: data("Groceries")
: Groceries transactions in sparse format with 9835 transactions (rows) and 169 items (columns)
: Groceries transactions in sparse format with 9835 transactions (rows) and 169 items (columns)
: data("Groceries")
: :
: Groceries transactions in sparse format with 9835 transactions (rows) and 169 items (columns)
: itemFrequencyPlot(Groceries)
: itemFrequencyPlot(Groceries,topN=20)
: itemFrequencyPlot(Groceries,topN=20,type="absolute")
:
:
: itemFrequencyPlot(Groceries,topN=20,type="absolute")
: itemFrequencyPlot(Groceries,topN=50,type="absolute")
: itemFrequencyPlot(Groceries,topN=120,type="absolute")
: rules¡-apriori(Groceries,parameter = list(support(0.001),conf(80))) Error in (function (classes, fdef, mtable) : unable to find an inherited method for function support for signature "numeric"
: inspect(rules[1:5])
lhs rhs support confidence
1 liquor,red/blush wine =: bottled beer 0.001931876 0.9047619
2 curd,cereals =: whole milk 0.001016777 0.9090909
3 yogurt,cereals =: whole milk 0.001728521 0.8095238
4 butter,jam =: whole milk 0.001016777 0.8333333
5 soups,bottled beer =: whole milk 0.001118454 0.9166667
lift
1 11.235269
2 3.557863
3 3.168192
4 3.261374
5 3.587512
: rules¡-apriori(Groceries,parameter = list(support=0.001,conf=80))
Error in validObject(.Object) :
invalid class APparameter object: confidence is not in [0,1]
: rules¡-apriori(Groceries,parameter = list(support=0.001,conf=0.80))
Apriori Parameter specification:
confidence minval smax arem aval originalSupport support minlen maxlen target ext
0.8 0.1 1 none FALSE TRUE 0.001 1 10 rules FALSE

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

Absolute minimum support count: 9

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[169 item(s), 9835 transaction(s)] done [0.01s].
sorting and recoding items ... [157 item(s)] done [0.00s].
creating transaction tree ... done [0.01s].
checking subsets of size 1 2 3 4 5 6 done [0.02s].
writing ... [410 rule(s)] done [0.00s].
creating S4 object ... done [0.00s].
: inspect(rules[1:5])
lhs rhs support confidence lift
1 liquor,red/blush wine = bottled beer 0.001931876 0.9047619 11.235269
2 curd,cereals = whole milk 0.001016777 0.9090909 3.557863
3 yogurt,cereals = whole milk 0.001728521 0.8095238 3.168192
4 butter,jam = whole milk 0.001016777 0.8333333 3.261374
5 soups,bottled beer = whole milk 0.001118454 0.9166667 3.587512
: inspect(rules[1:10])
lhs rhs support confidence lift
1 liquor,red/blush wine = bottled beer 0.001931876 0.9047619 11.235269
2 curd,cereals = whole milk 0.001016777 0.9090909 3.557863
3 yogurt,cereals = whole milk 0.001728521 0.8095238 3.168192
4 butter,jam = whole milk 0.001016777 0.8333333 3.261374
5 soups,bottled beer = whole milk 0.001118454 0.9166667 3.587512
6 napkins,house keeping products = whole milk 0.001321810 0.8125000 3.179840
7 whipped/sour cream,house keeping products = whole milk 0.001220132 0.9230769 3.612599
8 pastry,sweet spreads = whole milk 0.001016777 0.9090909 3.557863
9 turkey,curd = other vegetables 0.001220132 0.8000000 4.134524
10 rice,sugar = whole milk 0.001220132 1.0000000 3.913649



9 CONCLUSION :

Hence, we have Build a required Database to develop BAI tool for considering one aspect of growth to the business Such as organization of products based on demand and patterns use R Programming.