FALL SEMESTER 2022-23

Data Warehousing and Data Mining

Course Code: CSI3010

Slot: L43+L44

Digital Assessment-4

Submitted by: Mayank Raj (20MIC0018)

Submitted to: Hiteshwar Kumar Azad

1. Create R programmes to implements the Apriori algorithm for Market Basket Analysis and computes the strong rules through Association Rule Mining.

Code:

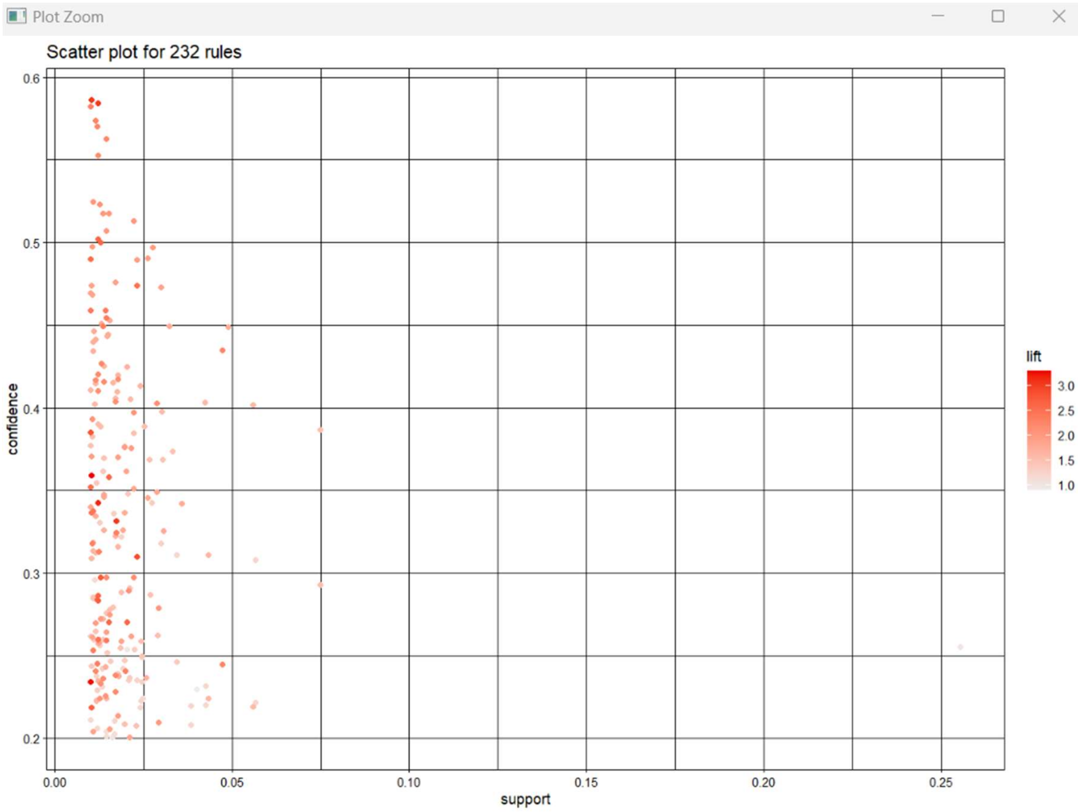
```
#Q1
library(arules)
library(arulesViz)
library(RColorBrewer)

#importing of the dataset
data("Groceries")

#apriori function
rules <- apriori(Groceries, parameter = list(supp=0.01,conf=0.2))

#inspect function
inspect(rules[1:10])

#Graph plot
plot(rules)
```

Output:

```
> library(arules)
> library(arulesViz)
> library(RColorBrewer)
>
> #importing of the dataset
> data("Groceries")
>
> #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
        0.2    0.1    1 none FALSE            TRUE       5    0.01      1     10  rules
  ext
 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].
>
> #inspect function
> inspect(rules[1:10])
     lhs                rhs                support    confidence coverage   lift
[1]  {}              => {whole milk}       0.25551601 0.2555160  1.00000000 1.000000
[2]  {hard cheese}   => {whole milk}       0.01006609 0.4107884  0.02450432 1.607682
[3]  {butter milk}   => {other vegetables} 0.01037112 0.3709091  0.02796136 1.916916
[4]  {butter milk}   => {whole milk}       0.01159126 0.4145455  0.02796136 1.622385
[5]  {ham}           => {whole milk}       0.01148958 0.4414062  0.02602949 1.727509
[6]  {sliced cheese} => {whole milk}       0.01077783 0.4398340  0.02450432 1.721356
[7]  {oil}           => {whole milk}       0.01128622 0.4021739  0.02806304 1.573968
[8]  {onions}        => {other vegetables} 0.01423488 0.4590164  0.03101169 2.372268
[9]  {onions}        => {whole milk}       0.01209964 0.3901639  0.03101169 1.526965
[10] {berries}       => {yogurt}           0.01057448 0.3180428  0.03324860 2.279848
```

```
     count
[1]  2513
[2]    99
[3]   102
[4]   114
[5]   113
[6]   106
[7]   111
[8]   140
[9]   119
[10]  104
>
> #Graph plot
> plot(rules)
```



2. Create R programmes to implement the decision trees and test both native speakers and non-native speakers. The "reading Skills" dataset can be used to create a decision tree and test its accuracy.

<u>Code:</u>

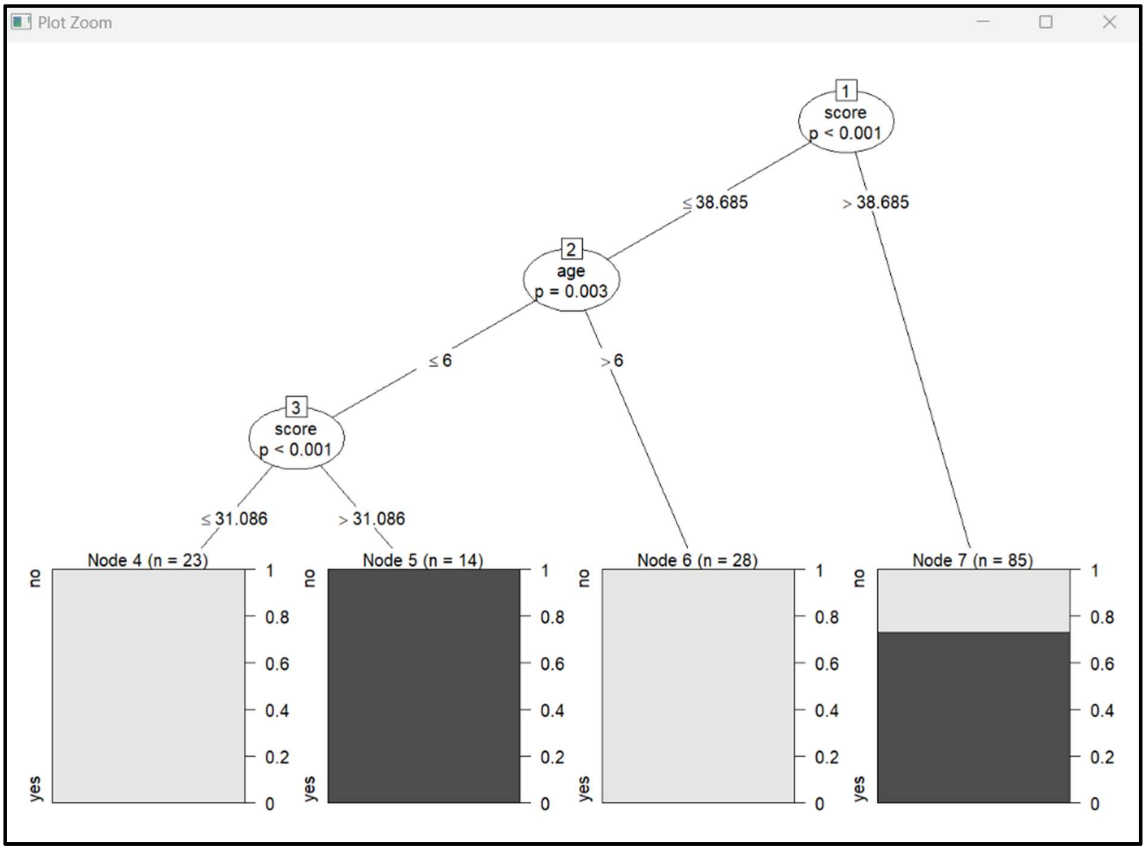
```
#Q2

library(datasets)

library(caTools)

library(dplyr)

library(magrittr)

library(party)

data("readingSkills")

head(readingSkills)


#Splitting Dataset into 4:1
```

```
sample_data = sample.split(readingSkills, SplitRatio = 0.8)
train_data <- subset(readingSkills, sample_data == TRUE)
test_data <- subset(readingSkills, sample_data == FALSE)

#Creating the decision tree and plotting the model
model<- ctree(nativeSpeaker~., train_data)
plot(model)

#Making Prediction
predict_model <- predict(model, test_data)
print(predict_model)

#create a table to count how many as classified
m_at <- table(test_data$nativeSpeaker, predict_model)
m_at

#determining the accuracy
ac_Test <- sum(diag(m_at))/sum(m_at)
ac_Test
```

Output:

```
> #Q2
> library(datasets)
> library(caTools)
> library(dplyr)
> library(magrittr)
> library(party)
> data("readingSkills")
> head(readingSkills)
  nativeSpeaker age shoeSize    score
1           yes   5 24.83189 32.29385
2           yes   6 25.95238 36.63105
3            no  11 30.42170 49.60593
4           yes   7 28.66450 40.28456
5           yes  11 31.88207 55.46085
6           yes  10 30.07843 52.83124
> #Splitting Dataset into 4:1
> sample_data = sample.split(readingSkills, SplitRatio = 0.8)
> train_data <- subset(readingSkills, sample_data == TRUE)
> test_data <- subset(readingSkills, sample_data == FALSE)
>
> #Creating the decision tree and plotting the model
> model<- ctree(nativeSpeaker~., train_data)
> plot(model)
> #Making Prediction
> predict_model <- predict(model, test_data)
> print(predict_model)
 [1] yes yes yes yes no  no  yes yes yes no  no  yes yes yes yes yes yes no  yes yes yes
[22] yes yes no  yes yes yes no  yes no  yes yes yes yes no  no  yes yes yes yes yes no
[43] no  yes yes no  yes yes yes yes
Levels: no yes
```

```
>
> #create a table to count how many as classified
> m_at <- table(test_data$nativeSpeaker, predict_model)
> m_at
     predict_model
      no yes
  no  13  13
  yes  0  24
>
> #determining the accuracy
> ac_Test <- sum(diag(m_at))/sum(m_at)
> ac_Test
[1] 0.74
```



Plot Zoom
1
score
p < 0.001
≤ 38.685
> 38.685
2
age
p = 0.003
≤ 6
> 6
3
score
p < 0.001
≤ 31.086
> 31.086
Node 4 (n = 23)
Node 5 (n = 14)
Node 6 (n = 28)
Node 7 (n = 85)
no
yes
1
0.8
0.6
0.4
0.2
0