Gabi Rivera

ADS502-01

26Nov2022

Introduction to Data Mining: Exercises 5.10

6. Consider the market basket transactions shown in Table 5.21 .

Table 5.21. Market basket transactions.

Transaction ID	Items Bought
1	{Milk, Beer, Diapers}
2	{Bread, Butter, Milk}
3	{Milk, Diapers, Cookies}
4	{Bread, Butter, Cookies}
5	{Beer, Cookies, Diapers}
6	{Milk, Diapers, Bread, Butter}
7	{Bread, Butter, Diapers}
8	{Beer, Diapers}
9	{Milk, Diapers, Bread, Butter}
10	{Beer, Cookies}

a. What is the maximum number of association rules that can be extracted from this data (including rules that have zero support)?

Answer: There is 206 maximum number of association rules that can be extracted from this data.

$R=3^d-2^{d+1}+1 = 3\text{^}(6) -2\text{^}(6+1)+1 = 206$

b. What is the maximum size of frequent itemsets that can be extracted (assuming minsup>0)?

Answer: There maximum size of a frequent itemset is 4. (Most transactions from ID 6 and 9)

c. Write an expression for the maximum number of size-3 itemsets that can be derived from this data set.

Answer: $\binom{6}{3} = \frac{6!}{3!3!} = \frac{6 \cdot 5 \cdot 4}{3 \cdot 2 \cdot 1} = 20$

d. Find an itemset (of size 2 or larger) that has the largest support.

Answer: {Bread, Butter} with instances of 5 for support when sizes 0 and 1 are ignored.

e. Find a pair of items, a and b, such that the rules {a}→{b} and {b}→{a} have the same confidence.

Answer: Beer and cookies both have a confidence of 2/4 = 0.5. The same with bread and butter having a confidence of 5/5 = 1.

8. Consider the following set of frequent 3-itemsets:

{1, 2, 3}, {1, 2, 4}, {1, 2, 5}, {1, 3, 4}, {1, 3, 5}, {2, 3, 4}, {2, 3, 5}, {3, 4, 5}.

Assume that there are only five items in the data set.

a. List all candidate 4-itemsets obtained by a candidate generation procedure using the Fk−1×F1 merging strategy.

   Answer:

   {1, 2, 3} = {1, 2, 3, 4}, {1, 2, 3, 5}

   {1, 2, 4} = {1, 2, 4, 5}

   {1, 3, 4} = {1, 3, 4, 5}

   {2, 3, 4} = {2, 3, 4, 5}

b. List all candidate 4-itemsets obtained by the candidate generation procedure in Apriori.

   Answer: {1, 2, 3, 4}, {1, 2, 3, 5}, {1, 2, 4, 5}, {1, 3, 4, 5}, and {2, 3, 4, 6}.

   Min-sup is 4 from 4 and 5 counts. Question (a) is from 3-itemsets so the same candidates are used for Apriori asking for 4-itemsets.

c. List all candidate 4-itemsets that survive the candidate pruning step of the Apriori algorithm.

Answer: {1, 2, 3, 4} and {1, 2, 3, 5} because their 3-itemset subsets are frequent.

15. Answer the following questions using the data sets shown in Figure 5.34 . Note that each data set contains 1000 items and 10,000 transactions. Dark cells indicate the presence of items and white cells indicate the absence of items. We will apply the Apriori algorithm to extract frequent itemsets with minsup=10% (i.e., itemsets must be contained in at least 1000 transactions).

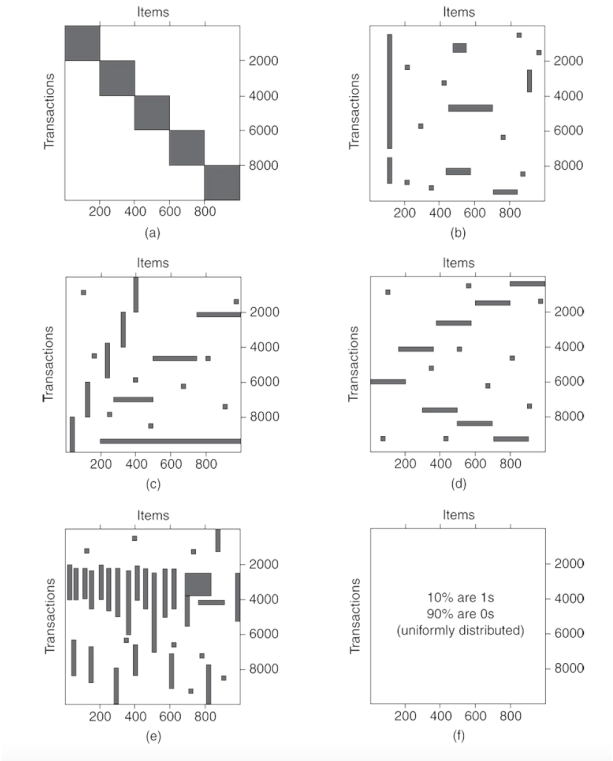

Figure 5.34.

Figures for Exercise 15.

a. Which data set(s) will produce the most number of frequent itemsets?

   Answer: Plot (e) because it has lengthy frequent occurrences or itemsets.

b. Which data set(s) will produce the fewest number of frequent itemsets?

   Answer: Plot (d) because it's opposite Plot (e). It has no frequent occurrences.

c. Which data set(s) will produce the longest frequent itemset?

   Answer: Plot (e) has the lengthiest frequency occurrences.

d. Which data set(s) will produce frequent itemsets with highest maximum support?

   Answer: Plot (b) has the longest single frequent itemset occurrence.

e. Which data set(s) will produce frequent itemsets containing items with wide-varying support levels (i.e., items with mixed support, ranging from less than 20% to more than 70%)?

   Answer: Plot (e) is the only plot that has varying frequency occurances.

Module5_Gabi Rivera

Gabi Rivera

2022-11-27

Data Science Using Python and R: Chapter 14

For the following exercises, work with the Churn_Training_File data set. Use R to solve each problem.

11. Subset the variables VMail Plan, Int'l Plan, CustServ Calls, and Churn into their own data frame. Change CustServ Calls into an ordered factor.

Libraries:

```
library(skimr)
library(arules)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'arules'
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

Import dataset:

```
c_train = read.csv("Churn_Training_File", sep = ",")
head(c_train)
```

```
##   State Account.Length Area.Code    Phone Intl.Plan VMail.Plan VMail.Message
## 1    KS            128       415 382-4657        no        yes            25
## 2    OH            107       415 371-7191        no        yes            26
## 3    NJ            137       415 358-1921        no         no             0
## 4    OH             84       408 375-9999       yes         no             0
## 5    OK             75       415 330-6626       yes         no             0
## 6    MA            121       510 355-9993        no        yes            24
##   Day.Mins Day.Calls Day.Charge Eve.Mins Eve.Calls Eve.Charge Night.Mins
## 1    265.1       110      45.07    197.4        99      16.78      244.7
## 2    161.6       123      27.47    195.5       103      16.62      254.4
## 3    243.4       114      41.38    121.2       110      10.30      162.6
## 4    299.4        71      50.90     61.9        88       5.26      196.9
## 5    166.7       113      28.34    148.3       122      12.61      186.9
## 6    218.2        88      37.09    348.5       108      29.62      212.6
##   Night.Calls Night.Charge Intl.Mins Intl.Calls Intl.Charge CustServ.Calls
## 1          91        11.01      10.0          3        2.70              1
## 2         103        11.45      13.7          3        3.70              1
## 3         104         7.32      12.2          5        3.29              0
## 4          89         8.86       6.6          7        1.78              2
## 5         121         8.41      10.1          3        2.73              3
## 6         118         9.57       7.5          7        2.03              3
##   Churn
## 1 False
## 2 False
## 3 False
## 4 False
## 5 False
## 6 False
```

Subset VMail Plan, Int'l Plan, CustServ Calls, and Churn:

```
min_churn = subset(c_train,
                   select = c("Churn",
                              "Intl.Plan",
                              "VMail.Plan",
                              "CustServ.Calls"))
```

Change Customer data type to factor:

```
min_churn$CustServ.Calls = as.factor(min_churn$CustServ.Calls)
skim(min_churn)
```

Data summary

Name	min_churn

Number of rows	3000
Number of columns	4
_____	
Column type frequency:	
character	3
factor	1
_____	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
Churn	0	1	4	5	0	2	0
Intl.Plan	0	1	2	3	0	2	0
VMail.Plan	0	1	2	3	0	2	0

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
CustServ.Calls	0	1	FALSE	10	1: 1068, 2: 679, 0: 626, 3: 383

```
summary(min_churn)
```

```
##     Churn            Intl.Plan          VMail.Plan        CustServ.Calls
##  Length:3000        Length:3000        Length:3000        1      :1068
##  Class :character   Class :character   Class :character   2      : 679
##  Mode  :character   Mode  :character   Mode  :character   0      : 626
##                                                           3      : 383
##                                                           4      : 149
##                                                           5      :  61
##                                                           (Other):  34
```

12. Create tables for each of the four variables. Include both counts and proportions in each table. Use the tables to discuss the “baseline” distribution of each variable.

Table: Intl.Plan

```
t1 =  table(min_churn$Intl.Plan)
t1
```

```
##
##   no  yes
## 2705  295
```

```
t11 = rbind(t1, round(prop.table(t1), 3))
colnames(t11) = c("Intl.Plan = no", "Intl.Plan = yes")
rownames(t11) = c("Count", "Proportion")
t11
```

```
##            Intl.Plan = no Intl.Plan = yes
## Count            2705.000         295.000
## Proportion          0.902           0.098
```

Table: VMail.Plan

```
t2 =  table(min_churn$VMail.Plan)
t2
```

```
##
##   no  yes
## 2170  830
```

```
t21 = rbind(t2, round(prop.table(t2), 3))
colnames(t21) = c("VMail.Plan = no", "VMail.Plan = yes")
rownames(t21) = c("Count", "Proportion")
t21
```

```
##            VMail.Plan = no VMail.Plan = yes
## Count             2170.000          830.000
## Proportion           0.723            0.277
```

Table: CustServ.Calls

```
t3 =  table(min_churn$CustServ.Calls)
t3
```

```
##
##    0    1    2    3    4    5    6    7    8    9
##  626 1068  679  383  149   61   22    8    2    2
```

```
t31 = rbind(t3, round(prop.table(t3), 3))
colnames(t31) = c("CSC = 0", "CSC = 1", "CSC = 2", "CSC = 3", "CSC = 4",
                  "CSC = 5", "CSC = 6", "CSC = 7", "CSC = 8", "CSC = 9")
rownames(t31) = c("Count", "Proportion")
t31
```

```
##            CSC = 0  CSC = 1 CSC = 2 CSC = 3 CSC = 4 CSC = 5 CSC = 6 CSC = 7
## Count      626.000 1068.000 679.000 383.000  149.00   61.00  22.000   8.000
## Proportion   0.209    0.356   0.226   0.128    0.05    0.02   0.007   0.003
##            CSC = 8 CSC = 9
## Count        2.000   2.000
## Proportion   0.001   0.001
```

Table: Churn

```
t4 =  table(min_churn$Churn)
t4
```

```
##
## False  True
##  2564   436
```

```
t41 = rbind(t4, round(prop.table(t4), 3))
colnames(t41) <- c("Churn = False", "Churn = True")
rownames(t41) <- c("Count", "Proportion")
t41
```

```
##            Churn = False Churn = True
## Count           2564.000      436.000
## Proportion         0.855        0.145
```

Answer: The baseline distribution of Intl.Plan is predominantly no at 90.2% and yes at only 9.8%. The baseline distribution of Vmail.Plan is majority no at 72.3% and yes at 27.7%. The baseline distribution of CustServ.Calls is highest at 1 value with 35.6% and lowest at 0.1% from values 8 & 9. The baseline distribution of Churn is majority false at 85.5% and yes at 14.5%.

13. Obtain the association rules using the settings outlined in Section 14.4. Mine association rules:

```
all_rules = apriori(data = min_churn,
                    parameter = list(supp = 0.01,
                                     target = "rules",
                                     conf = 0.4,
                                     minlen = 2,
                                     maxlen = 2))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.4    0.1    1 none FALSE            TRUE       5    0.01      2
##  maxlen target  ext
##       2  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 30
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[16 item(s), 3000 transaction(s)] done [0.00s].
## sorting and recoding items ... [12 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [32 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

Inspect top 5 rules (sorted by lift values):

```
inspect(head(all_rules, by = "lift", n = 10))
```

```
##      lhs                    rhs             support    confidence coverage
## [1]  {CustServ.Calls=5} => {Churn=True}     0.01200000 0.5901639  0.02033333
## [2]  {CustServ.Calls=4} => {Churn=True}     0.02266667 0.4563758  0.04966667
## [3]  {Intl.Plan=yes}    => {Churn=True}     0.04233333 0.4305085  0.09833333
## [4]  {Churn=True}       => {VMail.Plan=no} 0.12066667 0.8302752  0.14533333
## [5]  {CustServ.Calls=3} => {VMail.Plan=no} 0.09933333 0.7780679  0.12766667
## [6]  {VMail.Plan=yes}   => {Churn=False}    0.25200000 0.9108434  0.27666667
## [7]  {CustServ.Calls=1} => {Churn=False}    0.32000000 0.8988764  0.35600000
## [8]  {CustServ.Calls=3} => {Churn=False}    0.11433333 0.8955614  0.12766667
## [9]  {CustServ.Calls=4} => {VMail.Plan=no} 0.03733333 0.7516779  0.04966667
## [10] {CustServ.Calls=2} => {Churn=False}    0.20066667 0.8865979  0.22633333
##      lift     count
## [1]  4.060761  36
## [2]  3.140201  68
## [3]  2.962214 127
## [4]  1.147846 362
## [5]  1.075670 298
## [6]  1.065729 756
## [7]  1.051727 960
## [8]  1.047849 343
## [9]  1.039186 112
## [10] 1.037361 602
```

Identify which rules have Churn in the antecedent, lhs: *To work with lhs, we need our rules to be formatted as a data frame. However, the apriori() algorithm does not return output formatted as a data frame. To convert the format of lhs to a data frame, we use two as() commands.

```
all_rules_ant_df = as(as(attr(all_rules, "lhs"),"transactions"), "data.frame")
head(all_rules_ant_df, 15)
```

```
##                  items
## 1  {CustServ.Calls=5}
## 2  {CustServ.Calls=5}
## 3  {CustServ.Calls=5}
## 4  {CustServ.Calls=4}
## 5  {CustServ.Calls=4}
## 6  {CustServ.Calls=4}
## 7  {CustServ.Calls=4}
## 8     {Intl.Plan=yes}
## 9     {Intl.Plan=yes}
## 10    {Intl.Plan=yes}
## 11 {CustServ.Calls=3}
## 12 {CustServ.Calls=3}
## 13 {CustServ.Calls=3}
## 14       {Churn=True}
## 15       {Churn=True}
```

With isolated antecedents, examine to see which contain either Churn = True or Churn = False.

```
a1 = all_rules_ant_df$items == "{Churn=True}"
a2 = all_rules_ant_df$items == "{Churn=False}"
non_churn_ant = abs(a1+a2-1)
non_churn_ant
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1
```

14. Subset the rules from the previous exercise so none of the antecedents contain the Churn variable. Display the rules, sorted by descending lift value. 212 Chapter 14 A SSOCIAT ION RULES

Subset from datagrame all_rules only the rules that have non.churn.ant equal to one.

```
good_rules = all_rules[non_churn_ant == 1]
```

Inspect top 5 rules (sorted by lift values):

```
inspect(head(good_rules, by = "lift", n = 5))
```

```
##     lhs                    rhs             support    confidence coverage
## [1] {CustServ.Calls=5} => {Churn=True}     0.01200000 0.5901639  0.02033333
## [2] {CustServ.Calls=4} => {Churn=True}     0.02266667 0.4563758  0.04966667
## [3] {Intl.Plan=yes}    => {Churn=True}     0.04233333 0.4305085  0.09833333
## [4] {CustServ.Calls=3} => {VMail.Plan=no} 0.09933333 0.7780679  0.12766667
## [5] {VMail.Plan=yes}   => {Churn=False}    0.25200000 0.9108434  0.27666667
##     lift     count
## [1] 4.060761  36
## [2] 3.140201  68
## [3] 2.962214 127
## [4] 1.075670 298
## [5] 1.065729 756
```

Contingency table of Churn and Customer Service Calls:

```
t_csc_churn = table(min_churn$Churn, min_churn$CustServ.Calls)
t_csc_churn
```

```
##
##           0   1   2   3   4   5   6   7   8   9
##   False 540 960 602 343  81  25   8   4   1   0
##   True   86 108  77  40  68  36  14   4   1   2
```

```
colnames(t_csc_churn) = c("CSC = 0", "CSC = 1", "CSC = 2", "CSC = 3", "CSC = 4",
                          "CSC = 5", "CSC = 6", "CSC = 7", "CSC = 8", "CSC = 9")
rownames(t_csc_churn) <- c("Churn = False", "Churn = True")
addmargins(A = t_csc_churn, FUN = list(Total = sum), quiet = TRUE)
```

```
##
##                CSC = 0 CSC = 1 CSC = 2 CSC = 3 CSC = 4 CSC = 5 CSC = 6 CSC = 7
##   Churn = False     540     960     602     343      81      25       8       4
##   Churn = True       86     108      77      40      68      36      14       4
##   Total             626    1068     679     383     149      61      22       8
##
##                CSC = 8 CSC = 9 Total
##   Churn = False       1       0  2564
##   Churn = True        1       2   436
##   Total               2       2  3000
```

15. Obtain association rules using the confidence difference criterion outlined in Section 14.6.

Apply confidence difference criterion:

```
rules_confdiff = apriori(data = min_churn,
                         parameter = list(arem = "diff", aval = TRUE, minval = 0.4,
                                          supp = 0.01, target = "rules", conf = 0.05,
                                          minlen = 2, maxlen = 2))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem aval originalSupport maxtime support minlen maxlen
##        0.05    0.4    1 diff TRUE            TRUE       5    0.01      2      2
##  target  ext
##   rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 30
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[16 item(s), 3000 transaction(s)] done [0.00s].
## sorting and recoding items ... [12 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [1 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

Inspect top 5 rules (sorted by lift values):

```
inspect(head(rules_confdiff, by = "lift", n = 5))
```

```
##     lhs                   rhs          support confidence diff      coverage
## [1] {CustServ.Calls=5} => {Churn=True} 0.012   0.5901639  0.4448306 0.02033333
##     lift     count
## [1] 4.060761 36
```