Student Name: Brandon Russell

Submission Date: 6/10/18

DBST 667 – Data Mining

Dr. Irene Tsapara

Week 3 Individual Exercise

Deliverables: Two Files: (1) Submit this lab report with answers to all questions including output screenshots into the 'Individual Exercises Week 3' assignment folder. (2) Submit an R script that contains all commands with comments that briefly describe each commands purpose.

Grading: This exercise is worth 2% of the course grade. All questions must be answered in your own words with any paraphrased references properly cited using in-text citations and a reference list as needed. In addition, grammatical and spelling errors may affect the grade.

Part 2 – Run an exercise on the credit approval dataset you used for the week 2 exercise last week, completing this report and providing the commands, output screenshots, and discussion/interpretation as requested. Ensure that all commands are saved in this report and in an R script.

a. **Introduction: Describe the expected output and behavior of the Apriori method and what it will accomplish for the credit approval data. Use the knowledge gained through the week 3 tutorial and week 3 lectures. Provide a one-paragraph, masters-level response.**

The Apriori algorithm was designed to provide predictive capabilities through the analysis of k-itemsets, looking for items frequented together to formulate predictions within specific parameters. This method iteratively scans a given set of data and creates k number of itemsets, depending on what is specified. While this method could become computationally expensive depending on the size of the dataset, there are techniques employed to reduce the number of transactions being scanned. For example, the Apriori property will ensure a specified minimum threshold is met by a given itemset, otherwise it will not be scanned. Additionally, antimonotonicity will ensure all supersets of that itemset are not scanned either. Once frequent itemsets are collected, rules can be created based on a few different metrics, namely support, confidence, and lift. The support percentage threshold highlights the frequency of the antecedent itemset. The confidence percentage threshold tells us the frequency percentage of the antecedent and consequent itemsets together. Lift tells us the correlation between both itemsets. If lift is exactly 1, there is no correlation. If lift is above 1, there is a positive correlation, and below 1 means a negative correlation (Han, Kamber, & Pei, 2011).

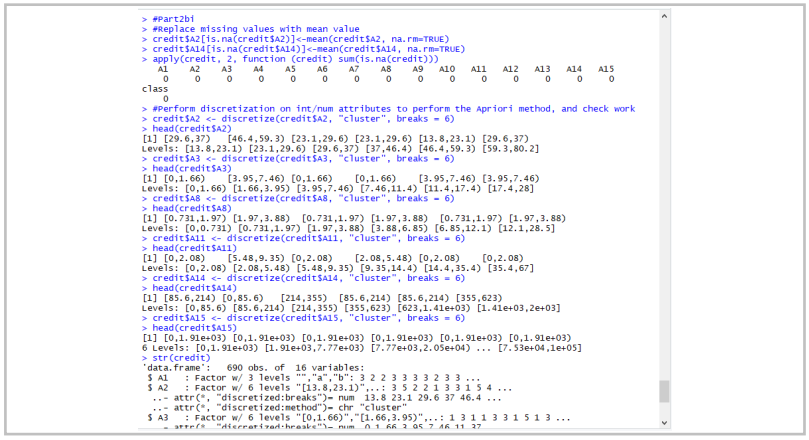
b. Data Pre-Processing: Load the credit approval data into R Studio using the read.csv command.

i. What data pre-processing does the Apriori method require for the credit approval data? Include the commands you ran and the output screenshots.

Command: >

```
#Part2bi
#Replace missing values with mean value
credit$A2[is.na(credit$A2)]<-mean(credit$A2,
na.rm=TRUE)
credit$A14[is.na(credit$A14)]<-mean(credit$A14,
na.rm=TRUE)
apply(credit, 2, function (credit) sum(is.na(credit)))
#Perform discretization on int/num attributes to perform the
```

Output:

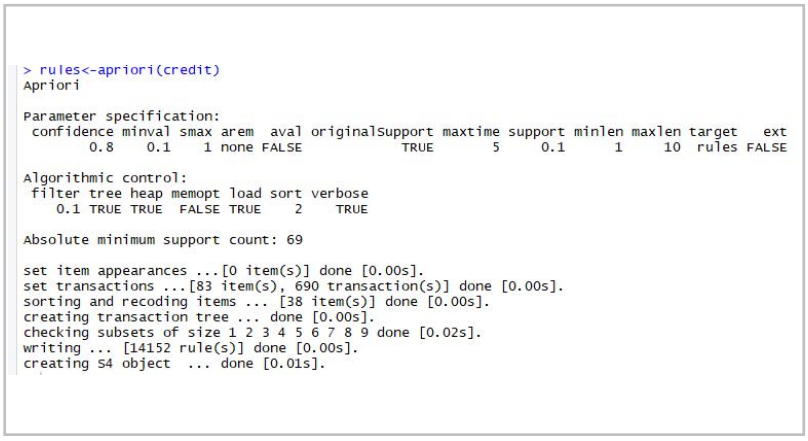


c. Apriori Method – Default Parameters:

i. Run the Apriori method with the default parameters and store the generated rules in a variable called 'rules'. Include the command, the output screenshot, and provide a one-paragraph, masters-level discussion of the returned rules and the default arguments.

Command: > rules<-apriori(credit)

Output:



Discussion:

In the above command we ran the Apriori algorithm on our credit data with the default parameters. The default parameters, in this instance, are a confidence level of 0.8, support of 0.1, minimum length of 1, maximum length of 10, and rules is the target. In the case of our credit data, 690 transactions were processed producing 14,152 rules.

ii. Run the inspect command to display the first 10 rules. Include the command, the output screenshot, and interpretation of the returned rules and metrics.

Command: > inspect(rules[1:10])

Output:

```
> inspect(rules[1:10])
     lhs                    rhs                    support   confidence lift      count
[1]  {}                  => {A15=[0,1.91e+03)} 0.8956522 0.8956522  1.0000000 618
[2]  {}                  => {A13=g}            0.9057971 0.9057971  1.0000000 625
[3]  {A11=[1.27,3.76)}   => {A10=t}            0.1057971 1.0000000  2.3389831  73
[4]  {A11=[1.27,3.76)}   => {A13=g}            0.1057971 1.0000000  1.1040000  73
[5]  {A6=q}              => {A5=g}             0.1043478 0.9230769  1.2272121  72
[6]  {A6=q}              => {A4=u}             0.1043478 0.9230769  1.2272121  72
[7]  {A6=q}              => {A15=[0,1.91e+03)} 0.1000000 0.8846154  0.9876774  69
[8]  {A6=q}              => {A13=g}            0.1101449 0.9743590  1.0756923  76
[9]  {A3=[7.46,11.4)}    => {A5=g}             0.1028986 0.8068182  1.0726484  71
[10] {A3=[7.46,11.4)}    => {A4=u}             0.1028986 0.8068182  1.0726484  71
>
```

Interpretation:

An example of one rule which was returned, if the antecedent were blank, then the consequent is predicted to that A13 will equal "g" with a support of approximately 0.9, confidence of 0.9, lift of 1.0 with a count of 625. This highlights a need to modify the default settings to retrieve more interesting results.

d. Apriori Method – Two Runs with Non-Default Parameters using different combinations of confidence, support, and minimum length values. For each run, specify the input parameters used (include the command, output screenshot, and discuss the rules returned) and then run the inspect command to preview the first 10 rules (include the command, output screenshot, and discuss which of the returned rules is the *strongest* and why):

i. 1st Non-Default Parameter Run:

Command: > rules<-apriori(credit, parameter = list(minlen=2, supp=0.2, conf=0.9))

Output:

```
> rules<-apriori(credit, parameter = list(minlen=2, supp=0.2, conf=0.9))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target   ext
        0.9    0.1    1 none FALSE            TRUE       5     0.2      2     10  rules FALSE

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

Absolute minimum support count: 138

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[83 item(s), 690 transaction(s)] done [0.00s].
sorting and recoding items ... [29 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 done [0.01s].
writing ... [1281 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

Discussion:

In the above command I have changed the minimum length to 2, support to 0.2, and confidence to 0.9. The purpose of this was to remove the rules with a blank antecedent, and also to narrow the resulting rules to those with higher levels of support and confidence in hopes of discovering more interesting rules. This command generated 1,281 rules.

Inspect Command: > inspect(rules[1:10])

Output:

```
> inspect(rules[1:10])
     lhs                    rhs                    support   confidence lift     count
[1]  {A2=[13.8,22)}      => {A15=[0,1.91e+03)} 0.2028986 0.9459459  1.056153 140
[2]  {A3=[1.66,3.95)}    => {A13=g}            0.2000000 0.9200000  1.015680 138
[3]  {A8=[0.731,1.97)}   => {A13=g}            0.2014493 0.9205298  1.016265 139
[4]  {A4=y}              => {A5=p}             0.2362319 1.0000000  4.233129 163
[5]  {A5=p}              => {A4=y}             0.2362319 1.0000000  4.233129 163
[6]  {A4=y}              => {A15=[0,1.91e+03)} 0.2231884 0.9447853  1.054857 154
[7]  {A4=y}              => {A13=g}            0.2188406 0.9263804  1.022724 151
[8]  {A5=p}              => {A15=[0,1.91e+03)} 0.2231884 0.9447853  1.054857 154
[9]  {A5=p}              => {A13=g}            0.2188406 0.9263804  1.022724 151
[10] {A14=[196,337)}     => {A15=[0,1.91e+03)} 0.2275362 0.9345238  1.043400 157
>
```

Discussion:

Look at the first 10 rules in our new rules set, we can see one the strongest rules are rules [4] and [5]. In rule [4] we see that the the antecedent of value "A4=y" is predicting a value of "A5=p" in the consequent. The support of 0.23, confidence of 1.0, lift of 4.23, and count of 163 are the highest across the board in these first 10 rules. This tells us there is a high support of both "A4=y" and "A5=p" in the transactions, and there is a high correlation between the two due to the high lift. Rule [5] simple has the antecedent and consequent reversed.

ii. 2nd Non-Default Parameter Run:

Command: > rules<-apriori(credit, parameter = list(minlen=2, supp=0.2, conf=0.9), appearance = list(rhs=c("A5=p", "A9=t")

Output:

```
> rules<-apriori(credit, parameter = list(minlen=2, supp=0.2, conf=0.9), appearance = list(rhs=c("A5=p", "A9
=t"), default="lhs"))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target   ext
        0.9    0.1    1 none FALSE            TRUE       5     0.2      2     10  rules FALSE

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

Absolute minimum support count: 138

set item appearances ...[2 item(s)] done [0.00s].
set transactions ...[83 item(s), 690 transaction(s)] done [0.00s].
sorting and recoding items ... [29 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 done [0.01s].
writing ... [44 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

Discussion: In the following command we leave the minimum length, confidence, and support values the same as before and now adjust the appearance setting to only show values in the

Inspect Command: > inspect(rules[1:10])

Output:

```
> inspect(rules[1:10])
     lhs                          rhs     support   confidence lift     count
[1]  {A4=y}                    => {A5=p} 0.2362319 1.0000000  4.233129 163
[2]  {class=+}                 => {A9=t} 0.4115942 0.9250814  1.768161 284
[3]  {A4=y,A15=[0,1.91e+03)}   => {A5=p} 0.2231884 1.0000000  4.233129 154
[4]  {A4=y,A13=g}              => {A5=p} 0.2188406 1.0000000  4.233129 151
[5]  {A10=t,class=+}           => {A9=t} 0.3000000 0.9904306  1.893067 207
[6]  {A12=f,class=+}           => {A9=t} 0.2130435 0.9130435  1.745152 147
[7]  {A7=v,class=+}            => {A9=t} 0.2289855 0.9349112  1.786949 158
[8]  {A1=b,class=+}            => {A9=t} 0.2739130 0.9174757  1.753624 189
[9]  {A4=u,class=+}            => {A9=t} 0.3521739 0.9492188  1.814296 243
[10] {A5=g,class=+}            => {A9=t} 0.3521739 0.9492188  1.814296 243
>
```

Discussion:

In the above results we can see the same rule from the previous section is still the strongest here, with the antecedent being "A4=y" and then consequent being "A5=p", now showing as rule [1]. Rule [3] is also showing potentially interesting results, with the antecedent being "A4=y,A15=[0,1.91e+03]" and the consequent being equal to "A5=p". The support is 0.22, confidence of 1.0, and lift of 4.23. Rule [3], however, might be considered redundant since the confidence of rule [1] is 100% and these two rules are very similar.

iii. How does changing the confidence, support, and minimum length values affect the returned rules? Provide a one-paragraph, masters-level response.

From the above commands, we can see changing the confidence, support, and minimum length settings can cause a significant change in the resulting rules. For example, if we adjust the minimum length to 2, versus the default of 1, we can immediately remove the uninteresting rules with blank item sets. Adjusting the support and confidence values will help narrow the rules to a more usable set. For instance, we might want to make sure our rules only showing items having a support of at least 50%, so we would need to adjust this value to 0.5. The confidence of a set is also an important attribute for finding interesting predictions, so we would want to adjust it accordingly to remove any rules that are not interesting. For example we might only want to see rules that have 100% confidence, so we would need to set the value to 1.000.

iv. What are the differences between the confidence, support, and lift metrics for identifying the strongest rules?

Confidence, support, and lift are different metrics we can use within the Apriori algorithm to narrow our parameters and discover more interesting predictive patterns in the datasets. The support threshold is a metric for the antecedent itemset, or the left-hand side. In this case the metrics is telling us the percentage of how often k-itemset is seen in the entire dataset. Confidence, on the other hand, tell us how often the antecedent itemset and consequent itemset are seen together. While these can be useful metrics, they do not always tell the whole story. If there happens to just be a lot of the consequent itemset, there is not necessarily a correlation. The lift metric takes into consideration the support metric for both itemsets and determines if there is a positive, negative, or zero correlation between the specified itemsets (Ng,

e. Apriori Method – Include only rules that have class='+' or class='-' on the right-hand side. Store the output in the variable 'rules'. (Hint: See the generating rules for the specified item sets section of the week 3 tutorial. You may need to adjust the confidence and support values). Include the command and output screenshot:

Command: > rules<-apriori(credit, appearance = list(rhs=c("class=+", "class=-"), default="lhs"))

Output:

```
> rules<-apriori(credit, appearance = list(rhs=c("class=+", "class=-"), default="lhs"))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target   ext
        0.8    0.1    1 none FALSE            TRUE       5     0.1      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: 69

set item appearances ...[2 item(s)] done [0.00s].
set transactions ...[83 item(s), 690 transaction(s)] done [0.00s].
sorting and recoding items ... [38 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 8 9 done [0.02s].
writing ... [1394 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
>
```

i. Run the inspect command to preview the first 10 rules. Include the command, output screenshot, a discussion identifying the strongest rules.

Command: > inspect(rules[1:10])

Output:

```
> inspect(rules[1:10])
     lhs                                rhs       support   confidence lift     count
[1]  {A9=f}                          => {class=-} 0.4434783 0.9300912  1.675621 306
[2]  {A7=h,A9=t}                     => {class=+} 0.1231884 0.8673469  1.949412  85
[3]  {A2=[13.8,22),A8=[0,0.731)}     => {class=-} 0.1014493 0.8045977  1.449536  70
[4]  {A2=[13.8,22),A9=f}             => {class=-} 0.1188406 0.9213483  1.659870  82
[5]  {A3=[1.66,3.95),A9=f}           => {class=-} 0.1217391 1.0000000  1.801567  84
[6]  {A3=[1.66,3.95),A10=f}          => {class=-} 0.1159420 0.8695652  1.566580  80
[7]  {A3=[1.66,3.95),A11=[0,1.27)}   => {class=-} 0.1289855 0.8165138  1.471004  89
[8]  {A2=[28.4,35.9),A9=f}           => {class=-} 0.1086957 0.9615385  1.732276  75
[9]  {A2=[28.4,35.9),A10=f}          => {class=-} 0.1130435 0.8571429  1.544200  78
[10] {A5=p,A9=f}                     => {class=-} 0.1362319 0.9591837  1.728033  94
>
```

Discussion:

In the above command I adjusted the appearance setting of the apriori command, including a list for the right hand itemsets to only include transactions containing a class attribute value of "+" or "-". Leaving the default support, confidence, and lift we found 1,394 rules. Rule [2], only looking at the first 10, had the highest positive correlation between the itemset {A7=h,A9=t} and itemset {class=+}.

ii. What do the returned rules suggest about the credit approval decision? Even though the attribute names are abstracted, provide a discussion about how the rules with abstracted attribute names, the strongest rules, and any other factors you wish to include positively or negatively impacted the credit approval decision (i.e. class='+' and class='-'). Provide a one-paragraph, masters-level response.

Examining the first 10 rules created in the previous step provides interesting results. Although the attributes are masked, we can still infer some meaning. As mentioned in the previous post, rule [2] has the strongest positive correlation between the itemset {A7=h,A9=t} and itemset {class=+}. The support was 0.12, confidence at 0.86, and lift was 1.94. After examining A7 and A9, it was noted there are only two possible values for A9, either "f" or "t", and A7 has 10 possible values. For this argument, we could assume A7 is relating to some type of credit code assigned to individuals based on their credit rating, A9 might refer to the person's gender, and class might be the result of whether the credit application was approved or denied. Looking at the rule [2]. we might deduce there is approximately 12% support that the applicants have a code "h", these applicants are male, and there is an 86% confidence level these applicants have an approved credit application. The lift value of 1.94 indicates there is a positive

f. Apriori Method – Prune the Returned Rules:

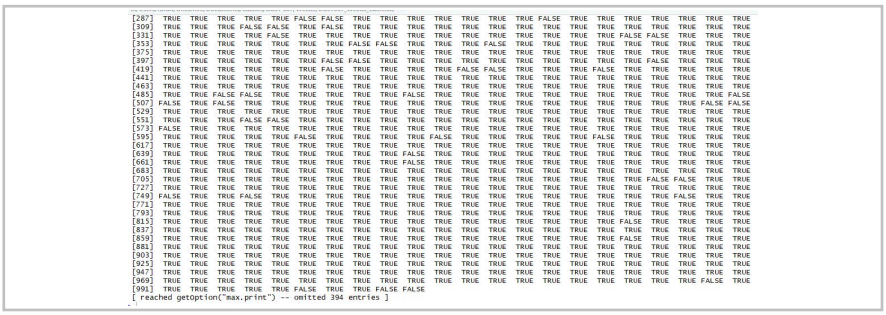
i. Why do we as data analysts prune the rules returned by the Apriori method?

Pruning is an important function for efficient utilization of the Apriori algorithm. This is referring to the process of removing transactions that are not interesting for any association rules. The Apriori property, mentioned in a previous answer, plays a part in this ensuring our minimum thresholds are maintained. Other actions taken to properly prune the dataset includes item merging, sub-itemset pruning, and item skipping. Item merging will form frequent closed itemsets if there is no superset of the consequent itemset. Sub-itemset pruning works by removing descendants of already discovered supersets. Item skipping involves pruning itemsets that have equivalent support at varying levels of different header tables (Han, Kamber, & Pei, 2011).

ii. Run the following commands on the variable 'rules' (generated from 2.e above) to find the redundant rules. Include the output screenshot from the commands and provide a discussion about the rules that were removed.

> rules.sorted<-sort(rules, by = "lift")
> inspect(rules.sorted)
> subset.matrix<-is.subset(rules.sorted, rules.sorted)
> subset.matrix[lower.tri(subset.matrix, diag=T)]<-NA
> redundant<-colSums(subset.matrix, na.rm=T)>=1
> which(redundant)

Output:

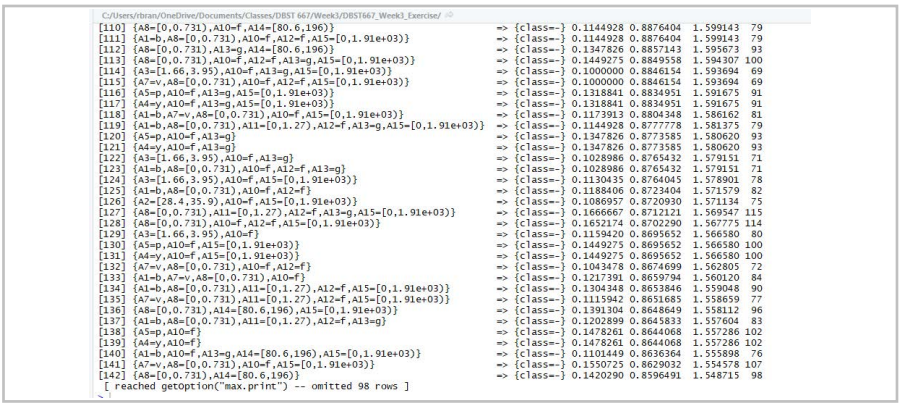


Discussion: I had to run the following code block to get this section to work properly:
rules.sorted<-sort(rules, by="lift")
inspect(rules.sorted)
redundant <- is.redundant(rules.sorted)
redundant
which(redundant)

iii. Run the following commands to remove the redundant rules and display the remaining rules. Include the commands, the output screenshot, and a discussion on which rules remain.

> rules.pruned<-rules.sorted[!redundant]
> inspect(rules.pruned)

Output :



Discussion : Now that we have sorted our rules by the lift value, and since we have pruned redundant rules we can now determine more interesting results about the credit dataset. For example, the rule with the highest lift value, rule [1], tells us the antecedent itemset {A4=u,A9=t,A10=t,A14=[0,80.6)} has approximately 11% support in the dataset, along with 97% confidence of being matched up with the consequent value of {class=+}, and an overall positive correlation of 2.19.

g. Rules Visualization – Choose any visualization method discussed in the tutorial to visualize the pruned rules in the previous step. Provide the command, output screenshot of the visualization, and a discussion about how effectively the plot represents the rules and the metrics for ranking rules.

Command: >

Output:

Discussion:

h. Summary

i. Why do we consider more than one metric to identify the strongest rules? Provide a one-paragraph, masters-level response.

ii. (Not graded) Which part of this exercise did you find the most challenging and why? What approach did you take the resolve the challenge?

iii. (Not graded) An example of page 13-14 of the week 3 tutorial shows how to display additional rule interest measurements including leverage, conviction, and coverage. What additional information do these metrics provide about the rules and why would we consider them? Check this link out for more information: Additional Metrics for Association Rules

References