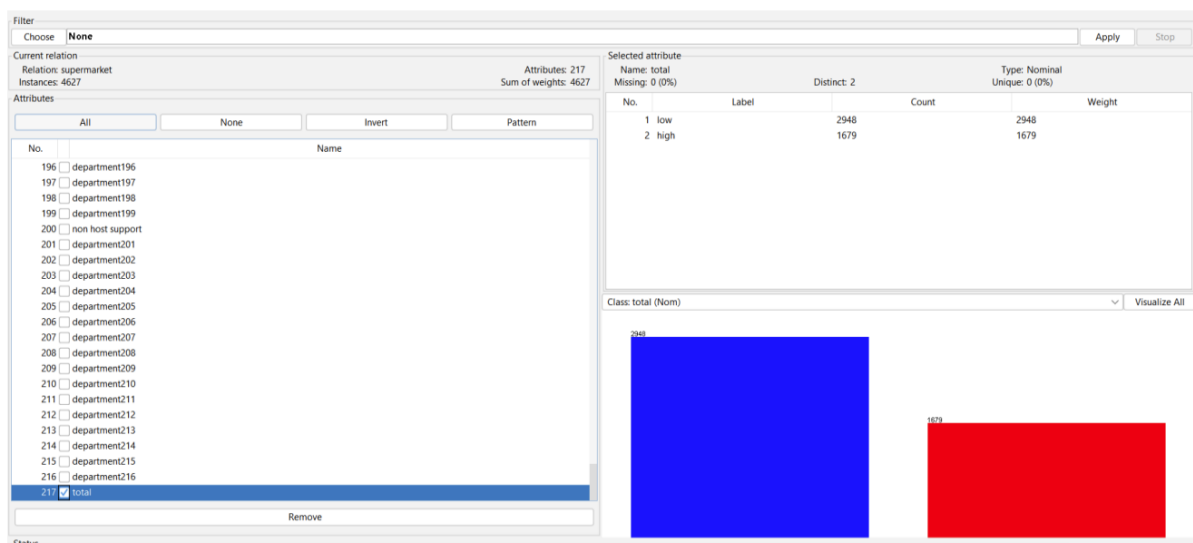


CO544 : Machine Learning and Data Mining  
Lab 05: Classification, Predictions, Clustering and Association  
Learning  
Nawarathna K.G.I.S.  
E/17/219

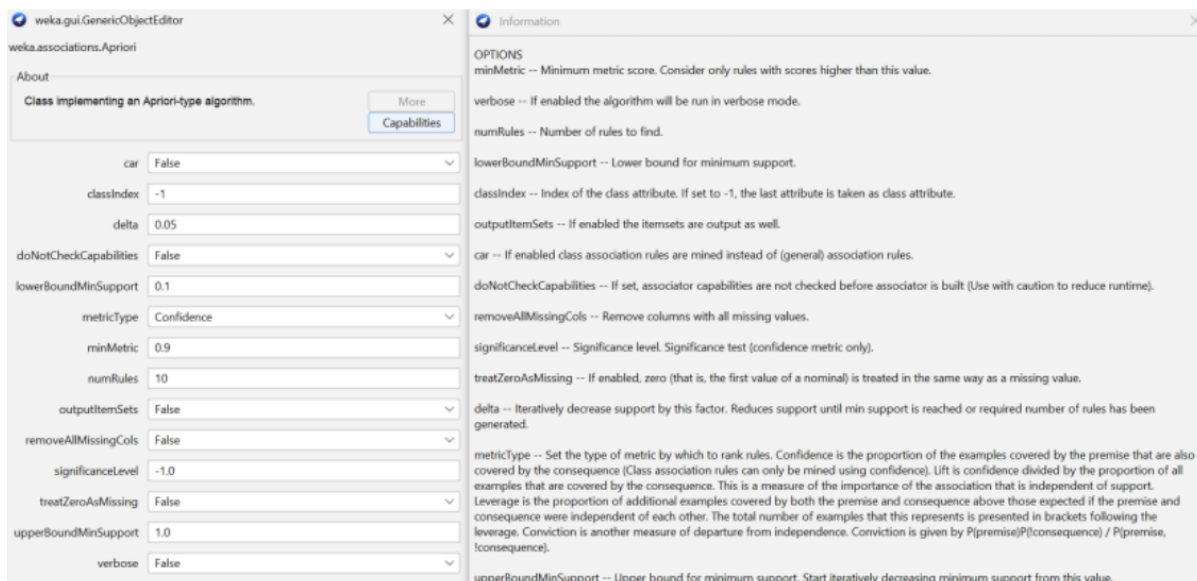
---

## Part 4 : Association Rule Learning

1. Load the `mini\_supermarket.arff` dataset into the WEKA tool and observe each transaction and its respective attribute values.



2. Go to the 'Associate' tab and choose 'Apriori' as the algorithm under the 'Associator' panel. Left click on the 'Apriori' label and obtain the 'Generic Object Editor' window. Click the 'More' button and identify what is indicated by each parameter in the 'Generic Object Editor'.



3. Briefly describe lowerBoundMinSupport, upperBoundMinSupport, delta, numRules and Confidence metricType parameters in the 'Generic Object Editor'.

lowerBoundMinSupport – Lower bound for minimum support

upperBoundMinSupport – Upper bound for minimum support. Start iteratively decreasing minimum support from this value

delta – Iteratively decrease support by this factor. Reduces support until min support is reached or required number of rules has been generated.

numRules – Number of rules to find.

Confidence metricType – Set the type of metric by which to rank rules.

Confidence is the proportion of the examples covered by the premise that are also covered by the consequence

4. Apply the 'Apriori' learner with the default values. Describe the meaning of each rule with its corresponding support and confidence.

```

Apriori
=====

Minimum support: 0.15 (694 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17

Generated sets of large itemsets:

Size of set of large itemsets L(1): 44

Size of set of large itemsets L(2): 380

Size of set of large itemsets L(3): 910

Size of set of large itemsets L(4): 633

Size of set of large itemsets L(5): 105

Size of set of large itemsets L(6): 1

Best rules found:

1. biscuits=t frozen foods=t fruit=t total=high 788 ==> bread and cake=t 723    <conf:(0.92)> lift:(1.27) lev:(0.03) [155] conv:(3.35)
2. baking needs=t biscuits=t fruit=t total=high 760 ==> bread and cake=t 696    <conf:(0.92)> lift:(1.27) lev:(0.03) [149] conv:(3.28)
3. baking needs=t frozen foods=t fruit=t total=high 770 ==> bread and cake=t 705    <conf:(0.92)> lift:(1.27) lev:(0.03) [150] conv:(3.27)
4. biscuits=t fruit=t vegetables=t total=high 815 ==> bread and cake=t 746    <conf:(0.92)> lift:(1.27) lev:(0.03) [159] conv:(3.26)
5. party snack foods=t fruit=t total=high 854 ==> bread and cake=t 779    <conf:(0.91)> lift:(1.27) lev:(0.04) [164] conv:(3.15)
6. biscuits=t frozen foods=t vegetables=t total=high 797 ==> bread and cake=t 725    <conf:(0.91)> lift:(1.26) lev:(0.03) [151] conv:(3.06)
7. baking needs=t biscuits=t vegetables=t total=high 772 ==> bread and cake=t 701    <conf:(0.91)> lift:(1.26) lev:(0.03) [145] conv:(3.01)
8. biscuits=t fruit=t total=high 954 ==> bread and cake=t 866    <conf:(0.91)> lift:(1.26) lev:(0.04) [179] conv:(3)
9. frozen foods=t fruit=t vegetables=t total=high 834 ==> bread and cake=t 757    <conf:(0.91)> lift:(1.26) lev:(0.03) [156] conv:(3)
10. frozen foods=t fruit=t total=high 969 ==> bread and cake=t 877    <conf:(0.91)> lift:(1.26) lev:(0.04) [179] conv:(2.92)

```

-Rule 1: If the customer buys biscuits, frozen foods, fruits, and the total bill is high, then the customer will be more likely to buy bread and cake with 0.92 confidence (723/788). : Support = 0.156

- Rule 2: If the customer buys baking needs, biscuits, fruit, and the total bill is high, then the customer will be more likely to buy bread and cake with 0.92 confidence (696/760). : Support = 0.150

- Rule 3: If the customer buys baking needs, frozen foods, fruits, and the total bill is high, then the customer will be more likely to buy bread and cake with 0.92 confidence (705/770). : Support = 0.152

- Rule 4: If the customer buys biscuits, vegetables, fruits, and the total bill is high, then the customer will be more likely to buy bread and cake with 0.92 confidence (746/815). : Support = 0.161

- Rule 5: If the customer buys party snack foods, fruits, and the total bill is high, then the customer will be more likely to buy bread and cake with 0.92 confidence (779/854). : Support = 0.168

- Rule 6: If the customer buys biscuits, frozen foods, vegetables, and the total bill is high, then the customer will be more likely to buy bread and cake with 0.92 confidence (725/797). : Support = 0.157

- Rule 7: If the customer buys baking needs, biscuits, vegetables, and the total bill is high, then the customer will be more likely to buy bread and cake with 0.92 confidence (701/772). : Support = 0.152

- Rule 8: If the customer buys biscuits, fruits, and the total bill is high, then the customer will be more likely to buy bread and cake with 0.92 confidence (866/954). : Support = 0.187

- Rule 9: If the customer buys frozen foods, fruits, vegetables, and the total bill is high, then the customer will be more likely to buy bread and cake with 0.92 confidence (757/834). : Support = 0.164

- Rule 10: If the customer buys frozen foods, fruits, and the total bill high, then the customer will be more likely to buy bread and cake with 0.92 confidence (877/969).: Support = 0.190

5. Apply the 'Apriori' learner with the two different numbers of rules: one less and one greater than the default number of rules. Comment on the three different sets of rules.

using only 5 rules :

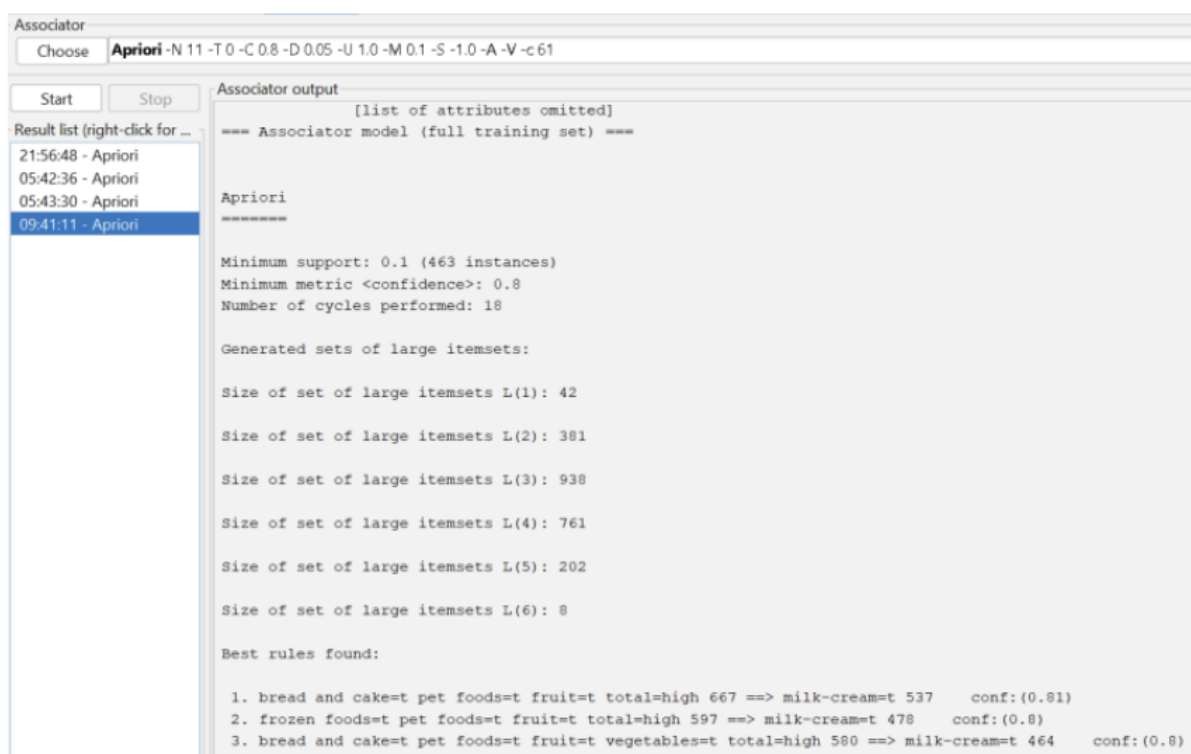
```
Best rules found:
1. biscuits=t frozen foods=t fruit=t total=high 788 ==> bread and cake=t 723 <conf:(0.92)> lift:(1.27) lev:(0.03) [155] conv:(3.35)
2. baking needs=t biscuits=t fruit=t total=high 760 ==> bread and cake=t 696 <conf:(0.92)> lift:(1.27) lev:(0.03) [149] conv:(3.28)
3. baking needs=t frozen foods=t fruit=t total=high 770 ==> bread and cake=t 705 <conf:(0.92)> lift:(1.27) lev:(0.03) [150] conv:(3.27)
4. biscuits=t fruit=t vegetables=t total=high 815 ==> bread and cake=t 746 <conf:(0.92)> lift:(1.27) lev:(0.03) [159] conv:(3.26)
5. party snack foods=t fruit=t total=high 854 ==> bread and cake=t 779 <conf:(0.91)> lift:(1.27) lev:(0.04) [164] conv:(3.15)
6. biscuits=t frozen foods=t vegetables=t total=high 797 ==> bread and cake=t 725 <conf:(0.91)> lift:(1.26) lev:(0.03) [151] conv:(3.06)
7. baking needs=t biscuits=t vegetables=t total=high 772 ==> bread and cake=t 701 <conf:(0.91)> lift:(1.26) lev:(0.03) [145] conv:(3.01)
```

using 17 rules :

```
Best rules found:
1. biscuits=t frozen foods=t party snack foods=t fruit=t vegetables=t total=high 510 ==> bread and cake=t 478 <conf:(0.94)> lift:(1.3) lev:(0.02) [110] conv:(4.33)
2. biscuits=t frozen foods=t cheese=t fruit=t total=high 495 ==> bread and cake=t 463 <conf:(0.94)> lift:(1.3) lev:(0.02) [106] conv:(4.2)
3. biscuits=t cheese=t fruit=t vegetables=t total=high 513 ==> bread and cake=t 479 <conf:(0.93)> lift:(1.3) lev:(0.02) [109] conv:(4.11)
4. baking needs=t biscuits=t party snack foods=t fruit=t total=high 557 ==> bread and cake=t 520 <conf:(0.93)> lift:(1.3) lev:(0.03) [119] conv:(4.11)
5. baking needs=t cheese=t fruit=t vegetables=t total=high 519 ==> bread and cake=t 483 <conf:(0.93)> lift:(1.29) lev:(0.02) [109] conv:(3.93)
6. frozen foods=t party snack foods=t tissues=paper prd=t fruit=t total=high 518 ==> bread and cake=t 482 <conf:(0.93)> lift:(1.29) lev:(0.02) [109] conv:(3.92)
7. juice=sat=ord=ms=t biscuits=t party snack foods=t fruit=t total=high 529 ==> bread and cake=t 492 <conf:(0.93)> lift:(1.29) lev:(0.02) [111] conv:(3.9)
8. biscuits=t cheese=t fruit=t total=high 584 ==> bread and cake=t 543 <conf:(0.93)> lift:(1.29) lev:(0.03) [122] conv:(3.9)
9. biscuits=t party snack foods=t fruit=t vegetables=t total=high 596 ==> bread and cake=t 554 <conf:(0.93)> lift:(1.29) lev:(0.03) [125] conv:(3.89)
10. baking needs=t biscuits=t frozen foods=t fruit=t vegetables=t total=high 561 ==> bread and cake=t 521 <conf:(0.93)> lift:(1.29) lev:(0.03) [117] conv:(3.84)
11. biscuits=t frozen foods=t party snack foods=t fruit=t total=high 589 ==> bread and cake=t 547 <conf:(0.93)> lift:(1.29) lev:(0.03) [123] conv:(3.84)
12. baking needs=t frozen foods=t party snack foods=t fruit=t total=high 558 ==> bread and cake=t 518 <conf:(0.93)> lift:(1.29) lev:(0.03) [116] conv:(3.81)
13. biscuits=t fruit=t department137=t total=high 502 ==> bread and cake=t 466 <conf:(0.93)> lift:(1.29) lev:(0.02) [104] conv:(3.8)
14. biscuits=t party snack foods=t tissues=paper prd=t fruit=t total=high 515 ==> bread and cake=t 478 <conf:(0.93)> lift:(1.29) lev:(0.02) [107] conv:(3.8)
15. baking needs=t cheese=t fruit=t total=high 584 ==> bread and cake=t 542 <conf:(0.93)> lift:(1.29) lev:(0.03) [121] conv:(3.81)
16. baking needs=t frozen foods=t tissues=paper prd=t fruit=t vegetables=t total=high 513 ==> bread and cake=t 476 <conf:(0.93)> lift:(1.29) lev:(0.02) [106] conv:(3.78)
17. biscuits=t canned vegetables=t fruit=t total=high 523 ==> bread and cake=t 485 <conf:(0.93)> lift:(1.29) lev:(0.02) [108] conv:(3.76)
```

Rules 7 case will get the best rules with less confidence when comparing the rules 17 case.

6. Suppose you are working in a supermarket which sells the above set of products and your company wants to increase the sales of the product milk. Using 'Apriori' builds a set of rules that predict in which situations the product milk will be sold or not. From the rule set select the best way to arrange the placement of the products in the supermarket to increase the sales of the milk. (Hint: set appropriate parameters for car and classindex) Note: In the supermarket.arff dataset an attribute called "milk" doesn't exist. The rules were therefore generated for the attribute "milk-cream".



The screenshot shows the 'Associator' software window. The 'Choose' dropdown is set to 'Apriori'. The command line below it reads: 'Apriori -N 11 -T 0 -C 0.8 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -A -V -c 61'. The 'Start' button is highlighted. On the left, a 'Result list (right-click for ...)' shows four entries: '21:56:48 - Apriori', '05:42:36 - Apriori', '05:43:30 - Apriori', and '09:41:11 - Apriori' (which is selected). The main 'Associator output' pane displays the following text:

```
[list of attributes omitted]
=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.1 (463 instances)
Minimum metric <confidence>: 0.8
Number of cycles performed: 18

Generated sets of large itemsets:

Size of set of large itemsets L(1): 42
Size of set of large itemsets L(2): 381
Size of set of large itemsets L(3): 938
Size of set of large itemsets L(4): 761
Size of set of large itemsets L(5): 202
Size of set of large itemsets L(6): 8

Best rules found:

1. bread and cake=t pet foods=t fruit=t total=high 667 ==> milk-cream=t 537    conf:(0.81)
2. frozen foods=t pet foods=t fruit=t total=high 597 ==> milk-cream=t 478    conf:(0.8)
3. bread and cake=t pet foods=t fruit=t vegetables=t total=high 580 ==> milk-cream=t 464    conf:(0.8)
```

car parameter = True

classindex = 61

confidence = 0.8 (min value)

It can be observed that, when bread, cake, fruits, vegetables, pet foods, frozen foods are kept closer together, and also if the customer will buy the combinations and also if the total bill is high, the customer will more likely buy milk-cream.

7. Now let us examine a real-world dataset supermarket.arff, which is provided with the WEKA tool. This dataset has been collected from an actual New Zealand supermarket. Take a look at this file using a text editor to verify that you understand the structure. The main point of this section is to show you how difficult it is to find any interesting patterns in this type of data. Experiment with 'Apriori' and learn a suitable set of association rules. Write a brief description of the main findings of your investigation.

```

Associator output
Attributes: 217
           [list of attributes omitted]
=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.4 (1851 instances)
Minimum metric <confidence>: 0.7
Number of cycles performed: 12

Generated sets of large itemsets:

Size of set of large itemsets L(1): 18

Size of set of large itemsets L(2): 16

Best rules found:

1. biscuits=t 2605 ==> bread and cake=t 2083    <conf:(0.8)> lift:(1.11) lev:(0.04) [208] conv:(1.4)
2. milk-cream=t 2939 ==> bread and cake=t 2337    <conf:(0.8)> lift:(1.1) lev:(0.05) [221] conv:(1.37)
3. fruit=t 2962 ==> bread and cake=t 2325    <conf:(0.78)> lift:(1.09) lev:(0.04) [193] conv:(1.3)
4. baking needs=t 2795 ==> bread and cake=t 2191    <conf:(0.78)> lift:(1.09) lev:(0.04) [179] conv:(1.29)
5. frozen foods=t 2717 ==> bread and cake=t 2129    <conf:(0.78)> lift:(1.09) lev:(0.04) [173] conv:(1.29)
6. vegetables=t 2961 ==> bread and cake=t 2298    <conf:(0.78)> lift:(1.08) lev:(0.04) [167] conv:(1.25)
7. juice-sat-cord-ms=t 2463 ==> bread and cake=t 1869    <conf:(0.76)> lift:(1.05) lev:(0.02) [96] conv:(1.16)
8. vegetables=t 2961 ==> fruit=t 2207    <conf:(0.75)> lift:(1.16) lev:(0.07) [311] conv:(1.41)
9. fruit=t 2962 ==> vegetables=t 2207    <conf:(0.75)> lift:(1.16) lev:(0.07) [311] conv:(1.41)
10. bread and cake=t 3330 ==> milk-cream=t 2337    <conf:(0.7)> lift:(1.1) lev:(0.05) [221] conv:(1.22)

```

- Most of the time the customer will be likely to buy bread and/or cake ; this depends on the fact that 70% of rules include those two food items.
- Another major fact is that fruits and vegetables are better to keep together since they are often bought together.
- Also bread, milk-cream will be more likely to be bought together by the customer.