OEIT 6, Data Analytics, 2022
Lab 8: Apriori Algorithm and Association rule mining with WEKA

SWAPNIL MANDAVILLI

2019120037

BE-EXTC

BATCH-B

Objective: Apply Apriori Algorithm to given dataset

Association Mining is defined as finding patterns, associations, correlations, or casual structures among sets of items or objects in transaction dataset, relational database, and other information repositories. The association rule takes the form of if … then… statement of the form:

A => B (read as, if A then B)

Performance measures for association rules:

Support:

support (A => B)= P(A ∩ B)

The minimum percentage of instances in the database that contain all items listed in a given association rule.

$$\textbf{support (A => B)=} \frac{\textbf{number of instances containing both A and B}}{\textbf{Total Number of instances}}$$

Example:

5000 transaction contain milk and bread in a set of 50000
➔ Support=> 5,000/50,000=10%

Confidence:

confidence (A=> B) = P(B|A)

Given a rule of the form “if A then B”, rule for confidence is the conditional probability that B is true when A is known to be True.

$$\textbf{confidence (A => B)=} \frac{\textbf{number of instances containing both A and B}}{\textbf{number of instances containing A}}$$

Example:

IF Customer purchases milk THEN they also purchase bread:
In a set of 50,000, there are 10,000 transactions that contain milk, and 5,000 of these contain also bread.
➔ Confidence => 5,000/10,000=50%

Association Rule Mining with WEKA

Task

Consider dataset “Groceries” and apply apriori algorithm on it. What are the first 5 rules generated when the min support is 0.001 (0.1%) and min confidence is 0.9 (90%) .

Exercise 1: Basic association rule creation manually

The 'database' below has four transactions. What association rules can be found in this set, if the minimum support (i.e coverage) is 60% and the minimum confidence (i.e. accuracy) is 80% ?

Trans_id	Itemlist
T1	{K, A, D, B}
T2	{D, A C, E, B}
T3	{C, A, B, E}
T4	{B, A, D}

Hint: Make a tabular and binary representation of the data in order to better see the relationship between Items. First generate all item sets with minimum support of 60%. Then form rules and calculate their confidence base on the conditional probability P(B|A) = |B∩A| / |A|. Remember to only take the item sets from the previous phase whose support is 60% or more.

ANSWER:

Tabular and binary representation of the data:

Transaction	A	B	C	D	E	K
T1	1	1	0	1	0	1
T2	1	1	1	1	1	0
T3	1	1	1	0	1	0
T4	1	1	0	1	0	0
TOTAL	4	4	2	3	2	1

THEREFORE,

A => 4/4 => 100%
B => 4/4 => 100%
C => 2/4 => 50%
D => 3/4 => 75%
E => 2/4 => 50%
K => 1/4 => 25%

The minimum support percent given is 60%.
Therefore, only A, B & D are selected.

Form the item sets containing 2 items.
A B 4, 100%
A D 3, 75%
B D 3, 75%

Form the item sets containing 3 items.
A B D 3

Form the rules and calculate their confidence (c).
Rules:

A -> B P(B|A) = |B∩A| / |A| = 4/4, |c: 100%
B -> A c: 100%
A -> D c: 75%
D -> A c: 100%
B -> D c: 75%
D -> B c: 100%
AB -> D c: 75%
D -> AB c: 100%
AD -> B c: 100%
B - > AD c: 75%
BD -> A c: 100%
A -> BD c: 75%

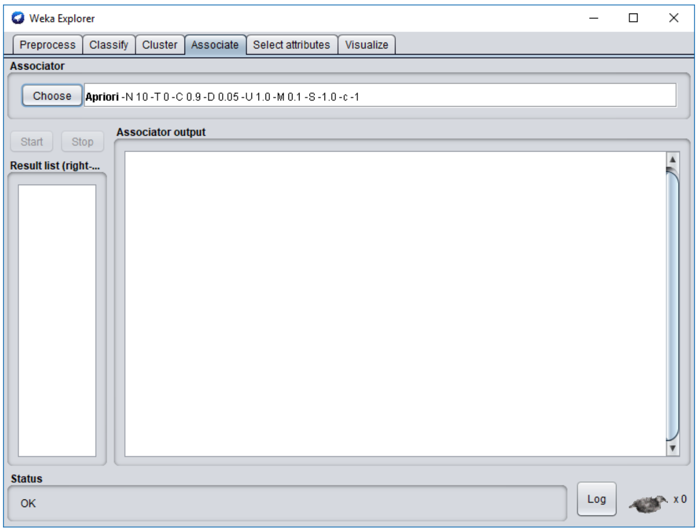
The rules with a confidence measure of 75% are pruned, and we are left with the following rule set:

A -> B
B -> A
D -> A
D -> B
D -> AB
AD-> B
DB-> A

<u>Exercise 2</u>: Input file generation and Initial experiments with Weka's association rule discovery.

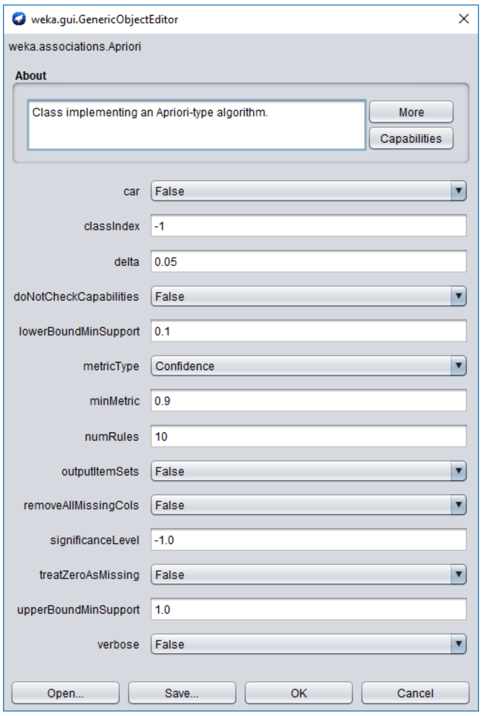
1. Launch Weka and try to do the calculations you performed manually in the previous exercise.Use the apriori algorithm for generating the association rules.

Once Data is loaded Click Associate Tab on top of the window.



Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

OK



weka.associations.Apriori

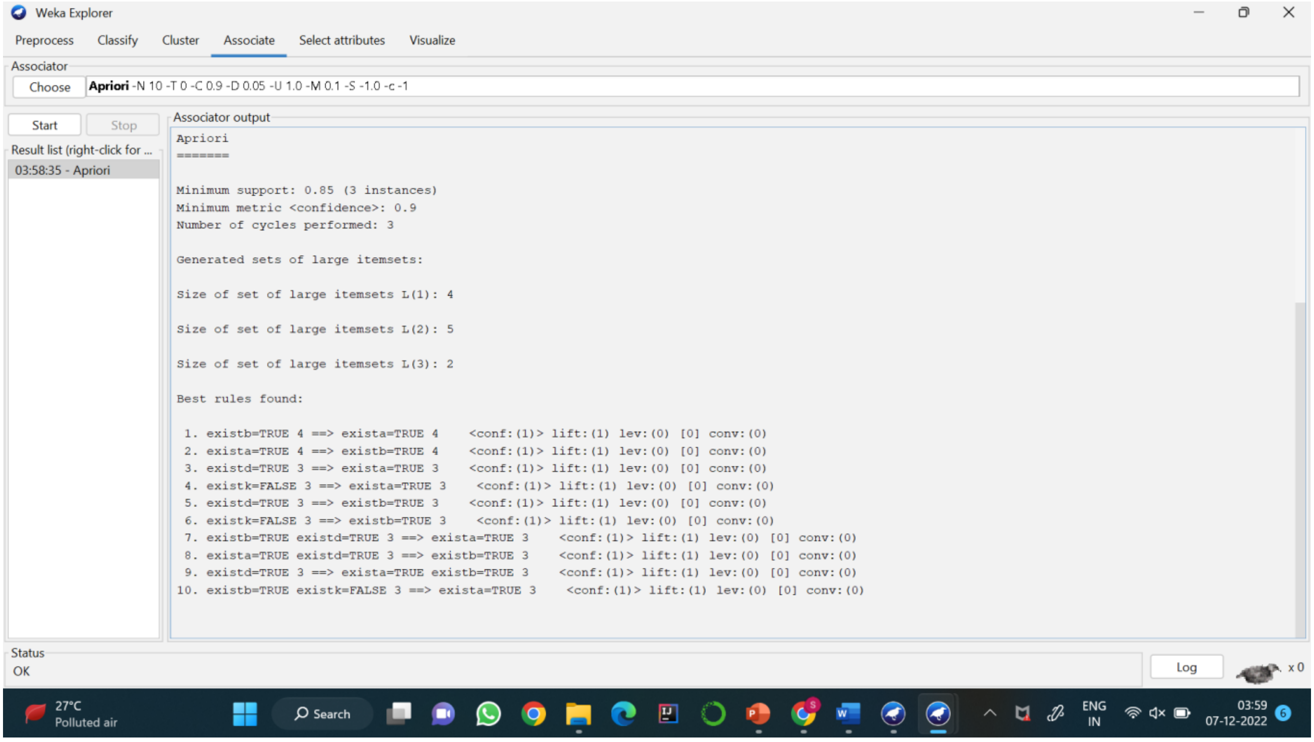
Class implementing an Apriori-type algorithm.

car	False
classIndex	-1
delta	0.05
doNotCheckCapabilities	False
lowerBoundMinSupport	0.1
metricType	Confidence
minMetric	0.9
numRules	10
outputItemSets	False
removeAllMissingCols	False
significanceLevel	-1.0
treatZeroAsMissing	False
upperBoundMinSupport	1.0
verbose	False

2. Left click the field of Associator, choose Show Property from the drop down list. The property window of Apriori opens.
3. Weka runs an Apriori-type algorithm to find association rules, but this algorithm is not exact the same one as we discussed in class.
    a. The min. support is not fixed. This algorithm starts with min. support as **upperBoundMinSupport** (default 1.0 = 100%), iteratively decrease it by **delta** (default 0.05 = 5%). Note that *upperBoundMinSupport* is decreased by delta before the basic Apriori algorithm is run for the first time.
    b. The algorithm stops when **lowerBoundMinSupport** (default 0.1 = 10%) is reached, or required number of rules – **numRules** (default value 10) have been generated.
    c. c. Rules generated are ranked by **metricType** (default Confidence). Only rules with score higher than **minMetric** (default 0.9 for Confidence) are considered and delivered as the output.
    d. If you choose to show the all frequent itemsets found, **outputItemSets** should be set as True.
4. Click Start button on the left of the window, the algorithm begins to run. The output is showing in the right window.

Did you succeed? Are the results the same as in your calculations? What kind of file did you use as input?

CSV format:
exista,existb,existc,existd,existe,existk
TRUE,TRUE,FALSE,TRUE,FALSE,TRUE
TRUE,TRUE,TRUE,TRUE,TRUE,FALSE
TRUE,TRUE,TRUE,FALSE,TRUE,FALSE
TRUE,TRUE,FALSE,TRUE,FALSE,FALSE



Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
Apriori
=======

Minimum support: 0.85 (3 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 3

Generated sets of large itemsets:

Size of set of large itemsets L(1): 4

Size of set of large itemsets L(2): 5

Size of set of large itemsets L(3): 2

Best rules found:

 1. existb=TRUE 4 ==> exista=TRUE 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 2. exista=TRUE 4 ==> existb=TRUE 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 3. existd=TRUE 3 ==> exista=TRUE 3    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 4. existk=FALSE 3 ==> exista=TRUE 3    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 5. existd=TRUE 3 ==> existb=TRUE 3    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 6. existk=FALSE 3 ==> existb=TRUE 3    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 7. existb=TRUE existd=TRUE 3 ==> exista=TRUE 3    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 8. exista=TRUE existd=TRUE 3 ==> existb=TRUE 3    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 9. existd=TRUE 3 ==> exista=TRUE existb=TRUE 3    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
10. existb=TRUE existk=FALSE 3 ==> exista=TRUE 3    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
```

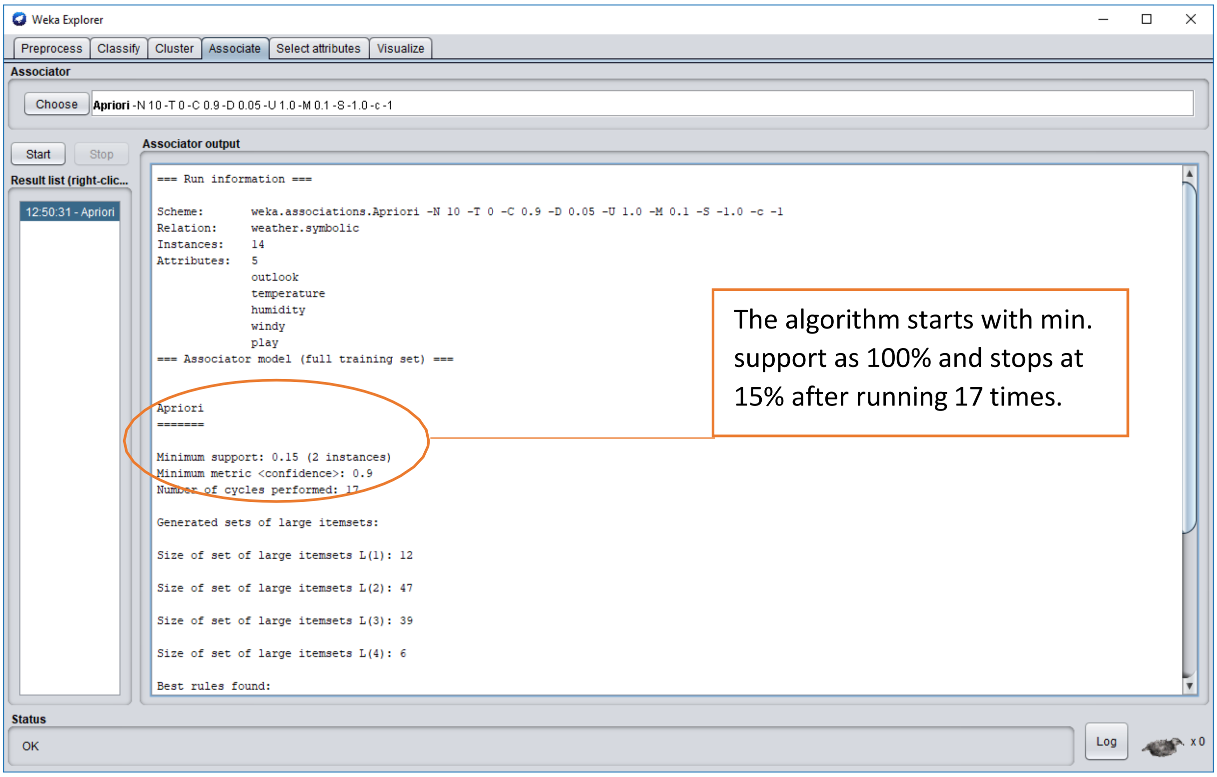
The results are not the same as the software has taken k=false into consideration. Whereas this wasn't a part of the manual calculations.

Exercise 3: Mining Association Rule with WEKA Explorer – Weather dataset

1. To get a feel for how to apply Apriori to prepared data set, start by mining association rules

from the weather.nominal.arff data set of Lab One. Note that Apriori algorithm expects **data that is purely nominal: If present, numeric attributes must be discretized first.**

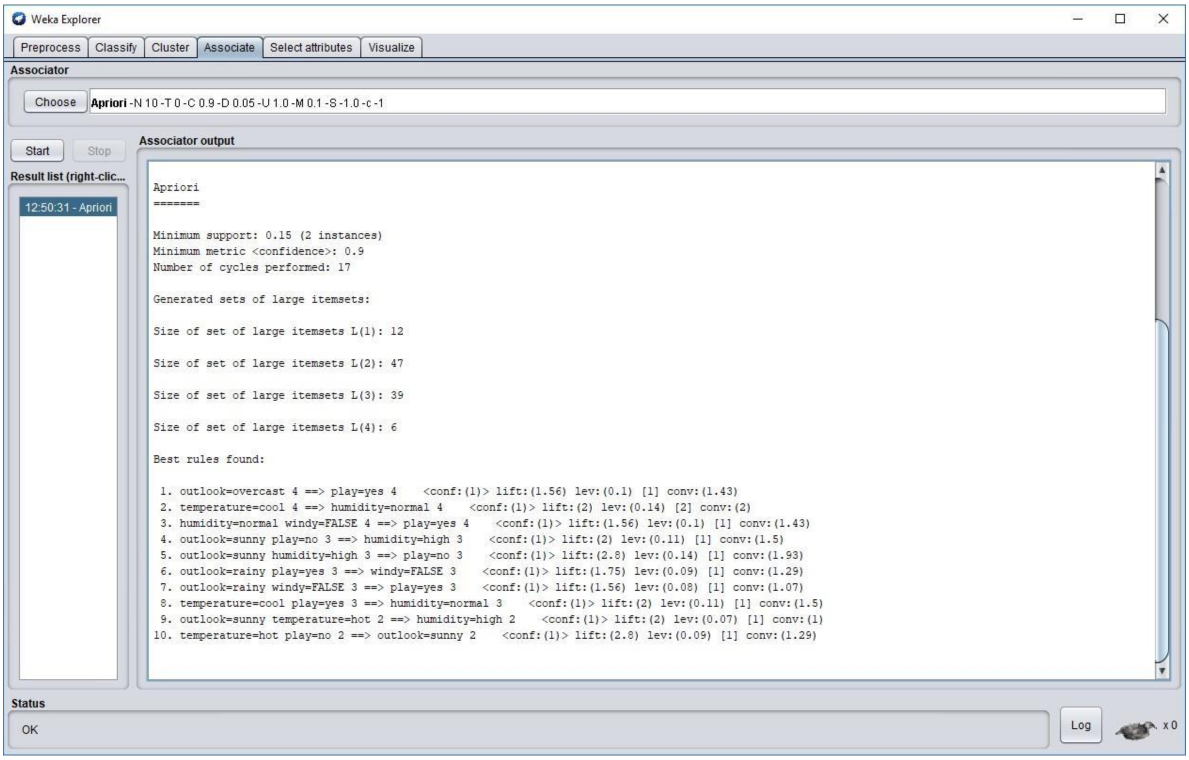
2. Like in the previous example choose Associate and Click Start button on the left of the window, the algorithm begins to run. The output is showing in the right window.
3. You could re-run Apriori algorithm by selecting different parameters, such as lowerBoundMinSupport, minMetric (min. confidence level), and different evaluation metric (confidence vs. lift), and so on.



Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
=== Run information ===

Scheme:       weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation:     weather.symbolic
Instances:    14
Attributes:   5
              outlook
              temperature
              humidity
              windy
              play
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.15 (2 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17

Generated sets of large itemsets:

Size of set of large itemsets L(1): 12

Size of set of large itemsets L(2): 47

Size of set of large itemsets L(3): 39

Size of set of large itemsets L(4): 6

Best rules found:
```

The algorithm starts with min. support as 100% and stops at 15% after running 17 times.



Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

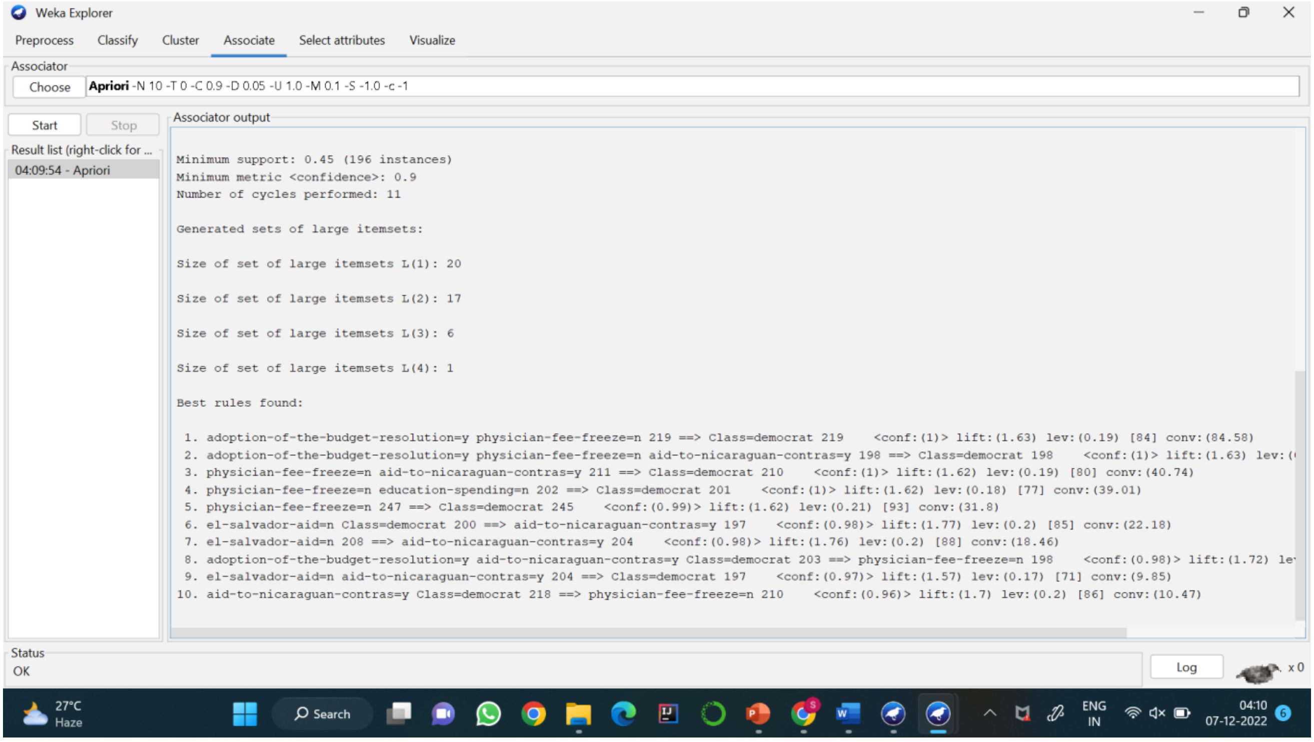
```
Apriori
=======

Minimum support: 0.15 (2 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17

Generated sets of large itemsets:

Size of set of large itemsets L(1): 12

Size of set of large itemsets L(2): 47

Size of set of large itemsets L(3): 39

Size of set of large itemsets L(4): 6

Best rules found:

 1. outlook=overcast 4 ==> play=yes 4    <conf:(1)> lift:(1.56) lev:(0.1) [1] conv:(1.43)
 2. temperature=cool 4 ==> humidity=normal 4    <conf:(1)> lift:(2) lev:(0.14) [2] conv:(2)
 3. humidity=normal windy=FALSE 4 ==> play=yes 4    <conf:(1)> lift:(1.56) lev:(0.1) [1] conv:(1.43)
 4. outlook=sunny play=no 3 ==> humidity=high 3    <conf:(1)> lift:(2) lev:(0.11) [1] conv:(1.5)
 5. outlook=sunny humidity=high 3 ==> play=no 3    <conf:(1)> lift:(2.8) lev:(0.14) [1] conv:(1.93)
 6. outlook=rainy play=yes 3 ==> windy=FALSE 3    <conf:(1)> lift:(1.75) lev:(0.09) [1] conv:(1.29)
 7. outlook=rainy windy=FALSE 3 ==> play=yes 3    <conf:(1)> lift:(1.56) lev:(0.08) [1] conv:(1.07)
 8. temperature=cool play=yes 3 ==> humidity=normal 3    <conf:(1)> lift:(2) lev:(0.11) [1] conv:(1.5)
 9. outlook=sunny temperature=hot 2 ==> humidity=high 2    <conf:(1)> lift:(2) lev:(0.07) [1] conv:(1)
10. temperature=hot play=no 2 ==> outlook=sunny 2    <conf:(1)> lift:(2.8) lev:(0.09) [1] conv:(1.29)
```

Exercise 4: Mining Association Rule with WEKA Explorer – Vote

Now consider a real-world dataset, **vote.arff**, which gives the votes of 435 U.S. congressmen on 16 key issues gathered in the mid-1980s, and also includes their party affiliation as a binary attribute. Association-rule mining can also be applied to this data to seek interesting associations.

Load data at Preprocess tab. Click the Open file button to bring up a standard dialog through which you can select a file. Choose the **vote.arff** file. To see the original dataset, click the **Edit** button, a viewer window opens with dataset loaded. This is a purely nominal dataset with some missing values (corresponding to abstentions).

> **Task 1**. Run Apriori on this data with default settings. Comment on the rules that are generated. Several of them are quite similar. How are their support and confidence values related?
>
> **Task 2.** It is interesting to see that none of the rules in the default output involve Class = republican. Why do you think that is?



Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
Minimum support: 0.45 (196 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 11

Generated sets of large itemsets:

Size of set of large itemsets L(1): 20

Size of set of large itemsets L(2): 17

Size of set of large itemsets L(3): 6

Size of set of large itemsets L(4): 1

Best rules found:

 1. adoption-of-the-budget-resolution=y physician-fee-freeze=n 219 ==> Class=democrat 219    <conf:(1)> lift:(1.63) lev:(0.19) [84] conv:(84.58)
 2. adoption-of-the-budget-resolution=y physician-fee-freeze=n aid-to-nicaraguan-contras=y 198 ==> Class=democrat 198    <conf:(1)> lift:(1.63) lev:(
 3. physician-fee-freeze=n aid-to-nicaraguan-contras=y 211 ==> Class=democrat 210    <conf:(1)> lift:(1.62) lev:(0.19) [80] conv:(40.74)
 4. physician-fee-freeze=n education-spending=n 202 ==> Class=democrat 201    <conf:(1)> lift:(1.62) lev:(0.18) [77] conv:(39.01)
 5. physician-fee-freeze=n 247 ==> Class=democrat 245    <conf:(0.99)> lift:(1.62) lev:(0.21) [93] conv:(31.8)
 6. el-salvador-aid=n Class=democrat 200 ==> aid-to-nicaraguan-contras=y 197    <conf:(0.98)> lift:(1.77) lev:(0.2) [85] conv:(22.18)
 7. el-salvador-aid=n 208 ==> aid-to-nicaraguan-contras=y 204    <conf:(0.98)> lift:(1.76) lev:(0.2) [88] conv:(18.46)
 8. adoption-of-the-budget-resolution=y aid-to-nicaraguan-contras=y Class=democrat 203 ==> physician-fee-freeze=n 198    <conf:(0.98)> lift:(1.72) le
 9. el-salvador-aid=n aid-to-nicaraguan-contras=y 204 ==> Class=democrat 197    <conf:(0.97)> lift:(1.57) lev:(0.17) [71] conv:(9.85)
10. aid-to-nicaraguan-contras=y Class=democrat 218 ==> physician-fee-freeze=n 210    <conf:(0.96)> lift:(1.7) lev:(0.2) [86] conv:(10.47)
```

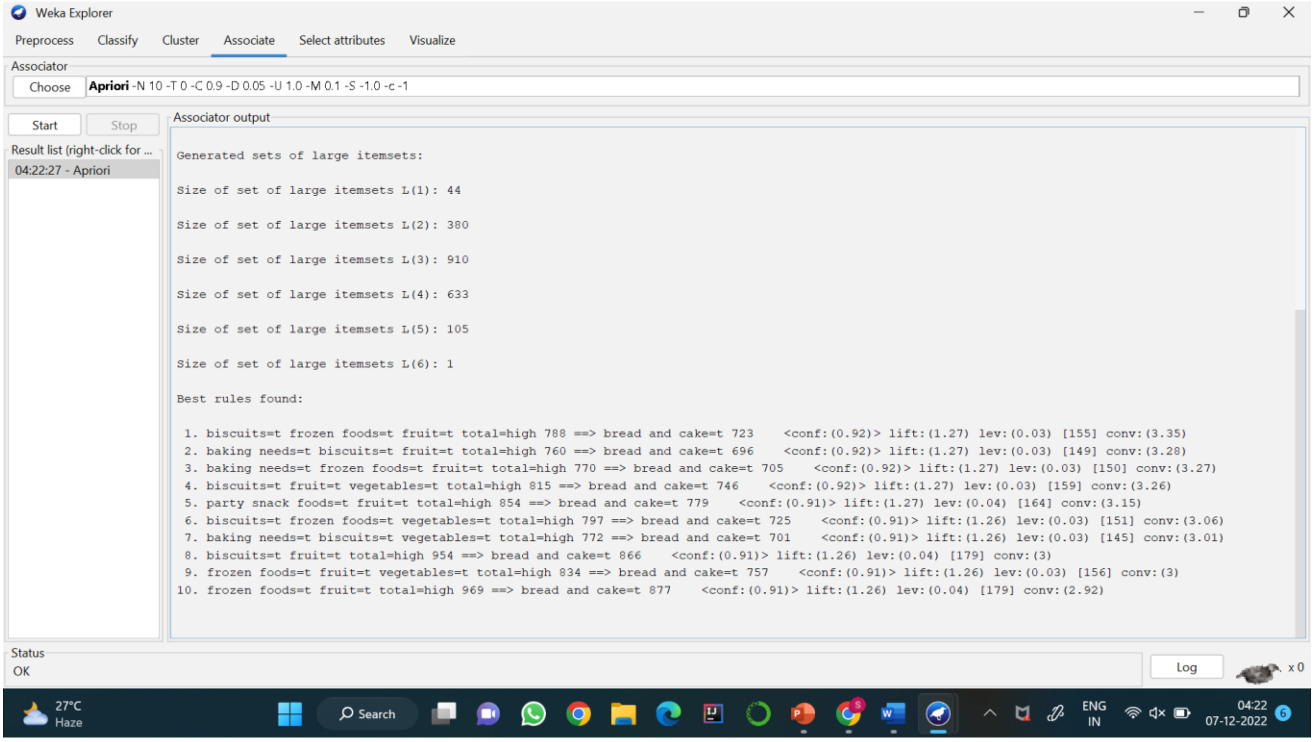
Class= republican did not appear in any of the rules since Class=democrat were 268 in no, whereas Class= republican were 167 in no, creating a lower support percentage.

Exercise 5: Let’s run Apriori on another real-world dataset.

Load data at Preprocess tab. Click the Open file button to bring up a standard dialog through which you can select a file. Choose the file. To see the original dataset, click the **Edit** button, a viewer window opens with dataset loaded.

To do market basket analysis in Weka, each transaction is coded as an instance of which the attributes represent the items in the store. Each attribute has only one value: If a particular transaction does not contain it (i.e., the customer did not buy that item), this is coded as a missing value.

Task 1. Experiment with Apriori and investigate the effect of the various parameters described before. Prepare a brief oral presentation on the main findings of your investigation.



Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
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)
```

From looking at the "*Associator output*" window, you can see that the algorithm presented 10 rules learned from the supermarket dataset. The algorithm is configured to stop at 10 rules by default, you can click on the algorithm name and configure it to find and report more rules if you like by changing the "*numRules*" value.

The rules discovered where:

1. biscuits=t frozen foods=t fruit=t total=high 788 ==> bread and cake=t 723 conf:(0.92)
2. baking needs=t biscuits=t fruit=t total=high 760 ==> bread and cake=t 696 conf:(0.92)
3. baking needs=t frozen foods=t fruit=t total=high 770 ==> bread and cake=t 705 conf:(0.92)
4. biscuits=t fruit=t vegetables=t total=high 815 ==> bread and cake=t 746 conf:(0.92)
5. party snack foods=t fruit=t total=high 854 ==> bread and cake=t 779 conf:(0.91)
6. biscuits=t frozen foods=t vegetables=t total=high 797 ==> bread and cake=t 725 conf:(0.91)
7. baking needs=t biscuits=t vegetables=t total=high 772 ==> bread and cake=t 701 conf:(0.91)
8. biscuits=t fruit=t total=high 954 ==> bread and cake=t 866 conf:(0.91)
9. frozen foods=t fruit=t vegetables=t total=high 834 ==> bread and cake=t 757 conf:(0.91)
10. frozen foods=t fruit=t total=high 969 ==> bread and cake=t 877 conf:(0.91)

You can see rules are presented in antecedent => consequent format. The number associated with the antecedent is the absolute coverage in the dataset (in this case a number out of a possible total of 4,627). The number next to the consequent is the absolute number of instances that match the antecedent and the consequent. The number in brackets on the end is the support for the rule (number of antecedent divided by the number of matching consequents). You can see that a cutoff of 91% was used in selecting rules, mentioned in the “Associator output” window and indicated in that no rule has a coverage less than 0.91.

I don’t want to go through all 10 rules, it would be too onerous. Here are few observations:

- We can see that all presented rules have a consequent of “bread and cake”.
- All presented rules indicate a high total transaction amount.
- “biscuits” an “frozen foods” appear in many of the presented rules.

You have to be very careful about interpreting association rules. They are associations (think correlations), not necessary causally related. Also, short antecedent are likely more robust than long antecedent that are more likely to be fragile.

If we are interested in total for example, we might want to convince people that buy biscuits, frozen foods and fruit to buy bread and cake so that they result in a high total transaction amount (Rule #1). This may sound plausible, but is flawed reasoning. The product combination does not cause a high total, it is only associated with a high total. Those 723 transactions may have a vast assortment of random items in addition to those in the rule.

What might be interesting to test is to model the path through the store required to collect associated items and seeing if changes to that path (shorter, longer, displayed offers, etc) have an effect on transaction size or basket size.