

Experiment 5

Aim: Apply Apriori Algorithm to a given dataset Association Rule Mining with WEKA

Exercise 1:

The 'database' below has four transactions. What can association rules 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 $P(B|A) = |B \cap A| / |A|$. Remember to only take the item sets from the previous phase whose support is 60% or more.

Exercise 2:

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.

The file may be given to Weka in e.g. two different formats. They are called ARFF (attribute-relation file format) and CSV (comma separated values). Both are given below:

ARFF:

@relation exercise

@attribute exista {TRUE, FALSE}

... @data

TRUE,TRUE,FALSE,TRUE,FALSE,TRUE

...

... CSV:

exista,existb,existc,existd,existe,existk TRUE,TRUE,FALSE,TRUE,FALSE,TRUE

...

...

2. Once Data is loaded Click Associate Tab on top of the window.
3. Left click the field of Associator, choose Show Property from the drop down list. The property window of Apriori opens.
4. 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.

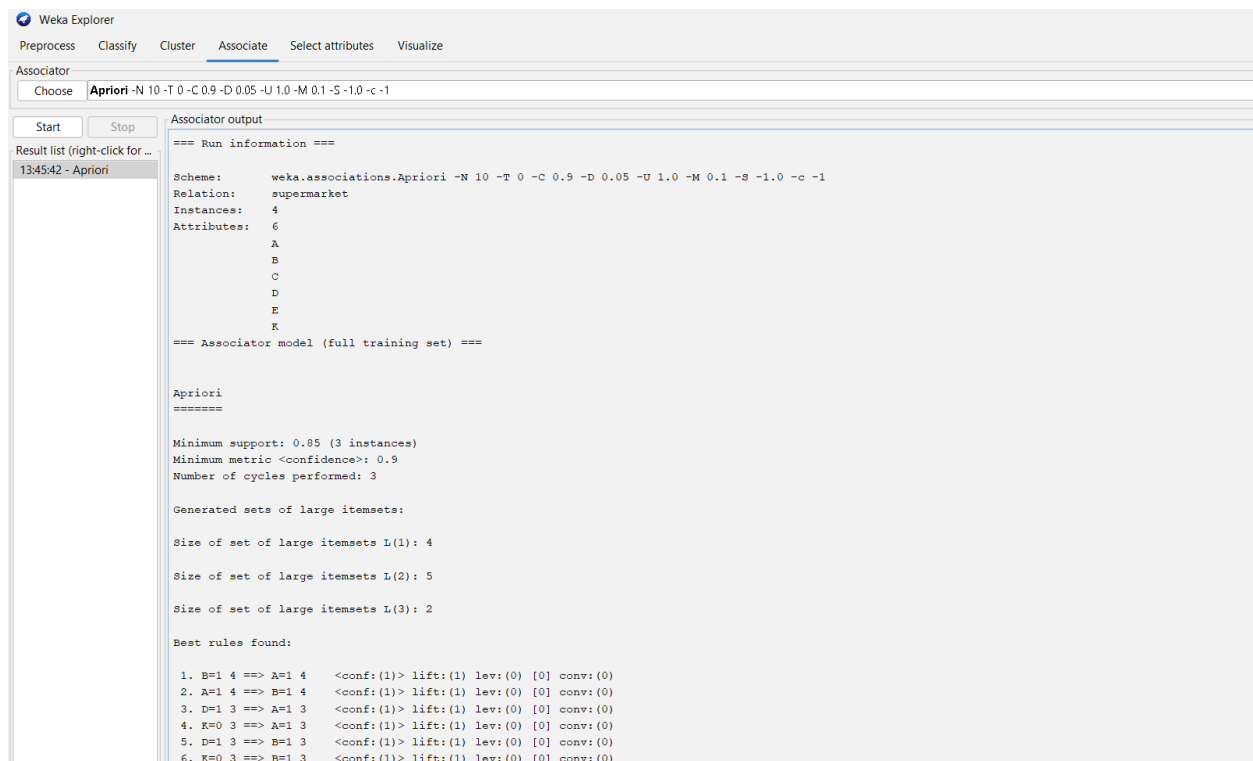
b. The algorithm stops when lowerBoundMinSupport (default 0.1 = 10%) is reached, or required number of rules – numRules (default value 10) have been generated.

c. 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.

5. 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?



The screenshot shows the Weka Explorer interface with the 'Associate' tab selected. The 'Apriori' algorithm is chosen, and its command line is visible: `-N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1`. The 'Start' button has been clicked, and the output is displayed in the 'Associator output' pane. The output includes run information, scheme details, and the generated rules.

```
==== 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:    supermarket
Instances:    4
Attributes:   6
              A
              B
              C
              D
              E
              K

==== Associator model (full training set) ====

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. B=1 4 ==> A=1 4 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
2. A=1 4 ==> B=1 4 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
3. D=1 3 ==> A=1 3 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
4. K=0 3 ==> A=1 3 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
5. D=1 3 ==> B=1 3 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
6. K=0 3 ==> B=1 3 <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
```



weka.gui.GenericObjectEditor



weka.associations.Apriori

About

Class implementing an Apriori-type algorithm.

More

Capabilities

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



Open...

Save...

OK

Cancel

Associator output

```
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:

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 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.



weka.gui.GenericObjectEditor



weka.associations.Apriori

About

Class implementing an Apriori-type algorithm.

More

Capabilities

car False



classIndex -1

delta 0.05

doNotCheckCapabilities False



lowerBoundMinSupport 0.1

metricType Confidence



minMetric 0.7

numRules 10

outputItemSets False



removeAllMissingCols False



significanceLevel -1.0

treatZeroAsMissing False



upperBoundMinSupport 1.0

verbose False



Open...

Save...

OK

Cancel

```

Associator output
Attributes:  5
            outlook
            temperature
            humidity
            windy
            play
=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.25 (3 instances)
Minimum metric <confidence>: 0.7
Number of cycles performed: 15

Generated sets of large itemsets:

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

Size of set of large itemsets L(2): 26

Size of set of large itemsets L(3): 4

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. humidity=normal 7 ==> play=yes 6    <conf:(0.86)> lift:(1.33) lev:(0.11) [1] conv:(1.25)
10. play=no 5 ==> humidity=high 4    <conf:(0.8)> lift:(1.6) lev:(0.11) [1] conv:(1.25)

```

Performed association rule mining on the weather dataset that I found online. Unlike decision tree, it has no target attribute. Instead, it tries to associate all the columns. In decision tree, depending upon values of outlook, temp, humidity and windy columns, values of play column is predicted. In association rule, values of play column is also considered and rest columns can also be predicted. For example, in rule 1, it states that if outlook is overcast then we can play. But in rule 4, it states that, if outlook is sunny and you are playing then humidity must be high.

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?

```
Scheme:      weka.associations.Apriori -N 10 -T 0 -C 0.7 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation:    vote
Instances:   435
Attributes:  17
             handicapped-infants
             water-project-cost-sharing
             adoption-of-the-budget-resolution
             physician-fee-freeze
             el-salvador-aid
             religious-groups-in-schools
             anti-satellite-test-ban
             aid-to-nicaraguan-contras
             mx-missile
             immigration
             synfuels-corporation-cutback
             education-spending
             superfund-right-to-sue
             crime
             duty-free-exports
             export-administration-act-south-africa
             Class
=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.5 (217 instances)
Minimum metric <confidence>: 0.7
Number of cycles performed: 10

Generated sets of large itemsets:

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

Size of set of large itemsets L(2): 4
```

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 supermarket.arff 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.

```
Scheme:      weka.associations.Apriori -N 10 -T 0 -C 0.7 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation:    supermarket
Instances:    4
Attributes:   6
              A
              B
              C
              D
              E
              K
=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.85 (3 instances)
Minimum metric <confidence>: 0.7
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. B=1 4 ==> A=1 4   <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
2. A=1 4 ==> B=1 4   <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
3. D=1 3 ==> A=1 3   <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
4. K=0 3 ==> A=1 3   <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
5. D=1 3 ==> B=1 3   <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
6. K=0 3 ==> B=1 3   <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
```

 weka.gui.GenericObjectEditor

✕

weka.associations.Apriori

About

Class implementing an Apriori-type algorithm.

More

Capabilities

car

False

▼

classIndex

-1

delta

0.05

doNotCheckCapabilities

False

▼

lowerBoundMinSupport

0.1

metricType

Confidence

▼

minMetric

0.7

numRules

10

outputItemSets

False

▼

removeAllMissingCols

False

▼

significanceLevel

-1.0

treatZeroAsMissing

False

▼

upperBoundMinSupport

1.0

verbose

False

▼

Open...

Save...

OK

Cancel

Conclusion:

Association rule mining finds interesting associations and relationships among large sets of data items. This rule shows how frequently a itemset occurs in a transaction. Given a set of transactions we can find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction.

Apriori algorithm allows us to mine the frequent itemset in order to generate association rule between them. The main limitation is time required to hold a vast number of candidates sets with much frequent item sets, low minimum support or large item sets i.e. it is not an efficient approach for large number of datasets.