**Practical: Association Rules**

In this lesson we perform a market basket analysis of a data set containing shopping information. The file *Shopping.csv* contains fields that indicate whether or not a customer, during a single visit, purchased a particular product. Each record represents a store visit in which at least one product was purchased. The file also contains basic demographics data, such as gender and age group.

# Introduction

When people buy cigarettes do they tend to buy chocolate or beer? If people have high cholesterol, do they also tend to have high blood pressure? If people buy car insurance, do they also buy house insurance?

Answers to such questions can form the basis of brand positioning, advertising and even direct marketing. But how do we find whether *associations* such as these exist, and how can we begin to search for them when our databases have tens of thousands of records and many fields?

Association detection algorithms provide rules describing the values of fields that typically occur together. They can therefore be used as an approach to this area of data understanding.

An association rule has two parts, an antecedent (if) and a consequent (then). An antecedent is an item found in the data. A consequent is an item that is found in combination with the antecedent. In RapidMiner, the term “premises” is used for antecedent and “conclusion” is used for consequent.

Below are some common terms used when working on association rules:

“Instances” indicates the number of records in the data set that match the antecedents.

“Support” refers to the percentage of records that match the antecedents. (Same as “Instances” but in percentage)

“Confidence” is the percentage of all records matching the antecedents that also match the consequent.

“Rule Support” is the percentage of records that match the entire rule (both the antecedents and consequent).

“Lift” refers to the expected return using a model or rule. In this context it is the ratio of the rule confidence to the overall percentage occurrence of the consequent in the data. Think of it as a measure of how much better the model is compared with a random-choice model.

In RapidMiner, you can select a criterion (confidence, lift, conviction, gain, lapace, and ps) to use to generate association rules. In this practical, we will be looking only at using the *confidence* criterion.

# Frequent Itemsets Generation

We will first see how to generate the frequent itemsets from a set of input data about purchases made at a supermarket.

1. Start RapidMiner and create a new process.
2. Obtain a copy of *shopping.csv* file (download a copy from the learning platform or check with the lab supervisor).
3. Insert a “Read CSV” operator into the process and use the *Import Configuration Wizard* to import the file. NOTE the following:
4. Use comma “,” as the delimiter for separating the columns.
5. In the file, we used value 1 to denote customer bought an item and 0 to denote did not buy the item. Since there are only two possible values, the data type should be binomial. However, RapidMiner will set the attributes for shopping items as *integer*.
6. Change the shopping item attributes data type to **binomial**. The attributes to change are “Ready Made”, “Frozen foods”, “Alcohol”, “Fresh Vegetables”, “Milk”, “Bakery goods”, “Fresh meat”, “Toiletries”, “Snacks” and “Tinned Goods”.

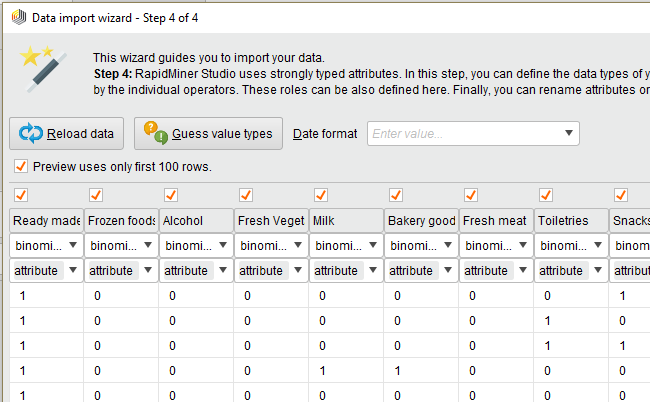


Figure : Setting attributes as binomial

Note that If you fail to do that and proceed to connect up the operators later, you will get an error message as shown below:

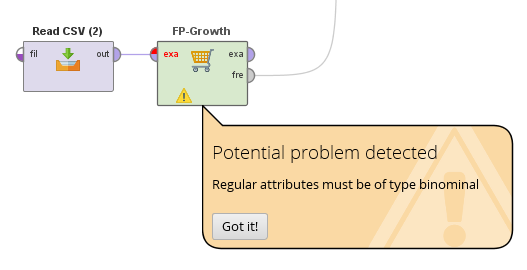


Figure : Error message when attibutes are not set as binomial

1. Use a “Select Attribute” operator to remove the demographic attributes like “Age”, “CHILDREN”, “GENDER”, “MARITAL” and “WORKING”. See figure below.

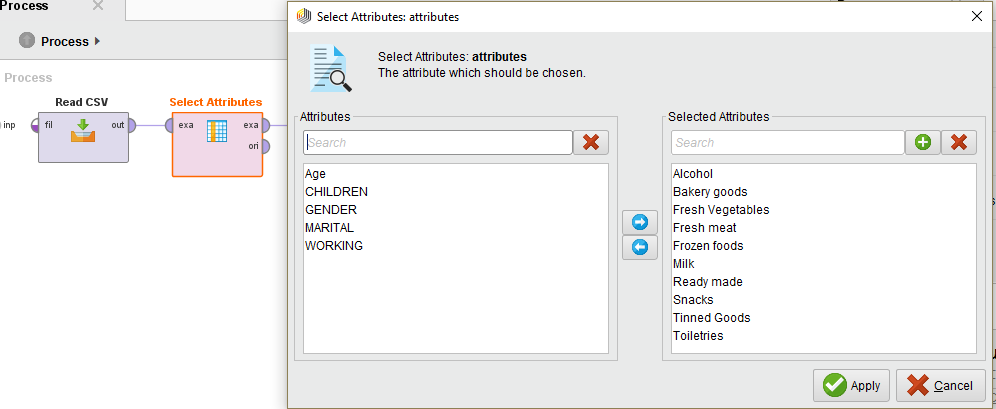


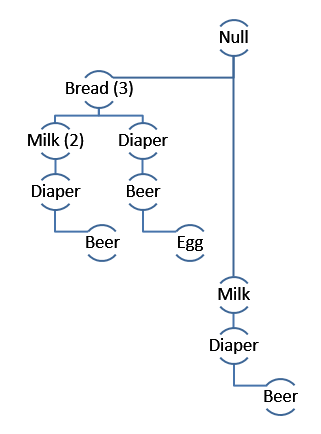
Figure : FIltering out the unwanted attributes.

We need to remove the demographic attribute because the frequent item set generation operator only takes in attributes of the type **binomial**.

With the data imported, we now see how we can generate frequent itemsets. We will use the “FP-Growth” operator provided by RapidMiner.

1. Insert a “FP-Growth” operator in the process and connect it to the “Select Attribute” operator.

The “FP-Growth” operator takes in an example sets with attributes of binomial type and generate frequent itemsets. The FP-Growth operator works by building a FP-tree data structure from the example set and derive frequent itemsets from the tree. It uses memory much more efficiently and executes faster compared to Apriori algorithm. *Detailed FP-Growth algorithm is not within the scope of this course.*



{Apple, Coke}, {Bread, Coke, Egg}

Figure : Using FP-Growth to generate frequent itemsets. (The tree in the figure is NOT the actual FP-tree solution for the table).

1. Connect the “FP-Growth” operator to the “res” port (see Figure 5 below).

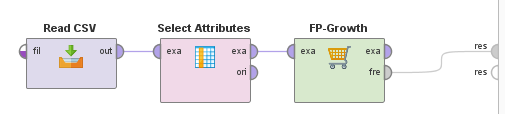


Figure : Connecting up the FP-Growth operator.

1. Click on the FP-Growth operator to look at the parameters. Note that by default, the “find min number of itemsets” checkbox is checked and the “min support” value is 0.95 (95%)
2. Run the process and look at the results. You should be able to see that a total of 12 frequent itemsets were discovered using the FP-Growth operator.

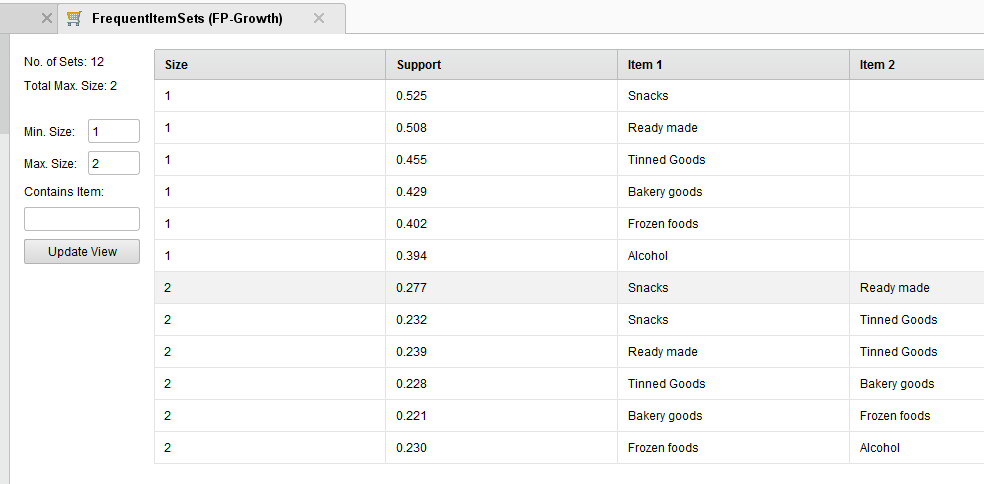


Figure : Frequent itemsets generated using the FP-Growth operator.

As can be seen from the results, we have 6 single-item and 6 2-item itemsets found. The single-item itemsets are pretty useless to us as there are no associations with other items. The other 6 itemsets indicates the associations between other items. For example, “*Snacks”* with “*Ready made”* and “Tinned Goods”, “Ready made” with “Tinned Goods” and so on.

Note also that the support of each itemset is listed under the “Support” column. Recall that *support* is the fraction of all the transactions that contains the items in the itemset. It is an indication of how frequently the itemset appears in all the transactions. So we see that about 27.7% of transactions contains the itemset {Snacks, Ready made}.

Although we indicated the minimum support (min support) to be at least 95%, the results still show itemsets of support less than 95%. This is because of the parameter settings for the FP-Growth operator.

1. Return to the process view, click on the FP-Growth operator and take a look at its parameters. Notice that the parameter “find min number of itemsets” is checked.

if the “find min number of itemsets” parameter is **checked**, the operator will ignore the “min support” value (even though we set it to 0.95) and generate a list of frequent itemsets sorting from highest to lowest support values. It will not generate all the itemsets though, the number of itemsets to generate is specified by the “min number of itemsets” parameters (see **Error! Reference source not found.** below)

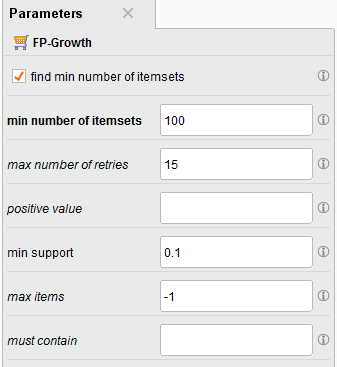


Figure : find min number of itemsets parameter.

On the other hand, if the “find min number of itemsets” box is unchecked, the system will only generate itemsets with support above that specified in the “min support” parameter.

The “max items” parameter will limit the number of items in an itemset, that is, the size of the itemset. It does NOT refer to the total number of frequent itemsets to be generated. The default value is -1, indicating that there is no limit to the number of frequent itemsets generated.

The “must contain” parameter allows us to specify the items that must be included in an itemset. In other words, itemsets without the specified items will be excluded from the results.

1. Uncheck the “find min number of itemsets” check box and leave the “min support” value as the default 0.95. Run the process again.

You will notice that no frequent itemsets are found as no frequent itemsets meet the criteria of having 95% support.

1. Change the “min support” to 0.1 (10%) and run the process again.

You should be able to finally see some frequent itemset being generated. How many itemsets are found? What is the maximum size of the itemsets? What is the itemset with the highest support value?

Of course 10% support is a very low percentage number to use. Some of the itemsets generated will be pretty useless as they only cover a small percentage of the transaction database.

You have seen how to use the FP-Growth operator to generate frequent itemsets from a set of transactions. As in any association rule mining, after generating frequent itemsets, we can start mining for association rules from the generated frequent itemsets.

# Mining Association Rules

We will now see how to mine association rule from our generated frequent itemsets.

1. Insert a “Create Association Rules” operator into the process.
2. Connect the “fre” output ports from the FP-Growth to the “ite” port of the newly inserted “Create Association Rules” operator (see Figure 8)

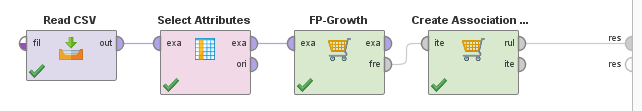


Figure : INserting the Create Association Rules operator.

As can be seen from the figure, the generated frequent itemsets will now be sent to a “Create Association Rules” operator for it to mine association rules.

1. Before you run the process, make sure the follow parameters are set

FP-Growth Operator

1. *find min number of itemsets*: unchecked
2. *min support*: 0.1 (10%)

Create Association Rules Operator

1. *criterion*: confidence
2. *min confidence*: 0.7 (70%)

RapidMiner will produce rules that have a minimum support of 10% of the sample, and a minimum confidence of 70%. In practice there are some trials and errors involved in finding useful values (too high and there are no rules generated; too low and there are too many rules generated).

1. Run the process and you should see the results as follows:

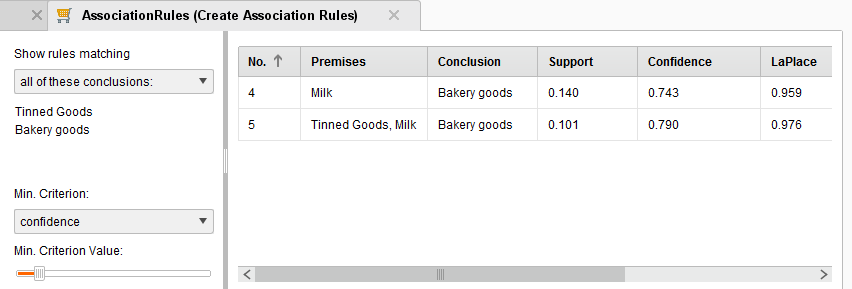


Figure : Rules generated with 10% support and 70% confidence.

The initial results displayed only 2 rules and it is a little misleading, there are actually 5 rules found that matches our specified support of 10% and confidence of 70%, however, the rules displayed are those that fulfils the filters criterion shown on the left of the screen (see Figure 10 below).

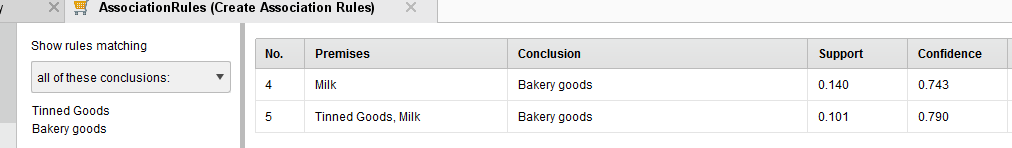


Figure : The left hand side shows the Filters for displaying rules while the right hand side shows details the rules.

As can be seen from the figure, rules 4 and 5 are displayed with the Premises, Conclusions, Support and Confidence. We can sort the rules by clicking on the headers. The first rule shown tells us that on 14% of the records (shopping visits), milk was purchased. Of this group, 74.3% also bought bakery goods.

The results view also displays values for the LaPlace, Gain, Lift and ps. These measure gives us some indication of the interestingness of the rule. For example, LaPlace is a function that is calculated based on the support and confidence. It ranges from 0 to 1 and the higher the value, the better the result. These measure will not be discussed in this course. If you are interested, you can explorer more on *association rule interestingness measures*.

1. To see all the 5 rules, change the filter from “all of these conclusions” to “any of these conclusion” and select all the items (Tinned Goods, Bakery goods). You can select them by holding down the Ctrl-Key and clicking the items. Also ensure that the “Min. Criterion” slider is slide all the way to the left.

Once you have done that, you should be able to see all the 5 rules (see Figure 11 below).

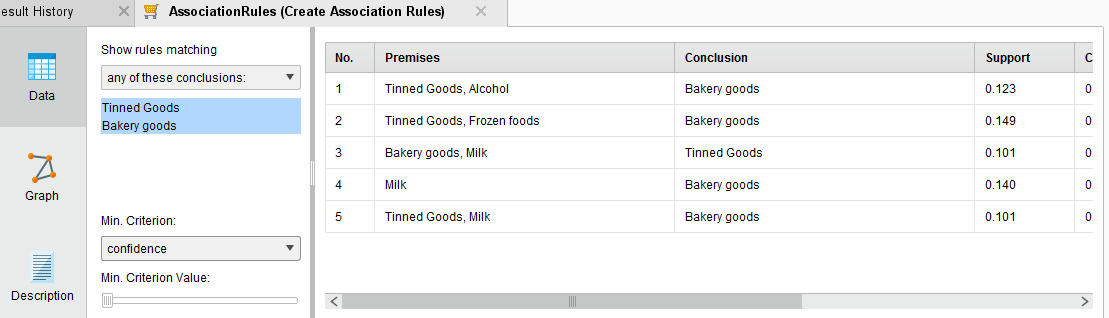


Figure : Displaying all the 5 rules generated for support of 10% and 70% confidence.

The “Min. Criterion Value” slider is used to vary the confidence criterion and controls how many rules are generated. This is useful for us if we have generated too many rules and wishes to increase the confidence level and thus reducing the number of rules.

1. Slide the slider to the right to reduce number of rules. Note that sliding the slider all the way to the right will always set the confidence level such that there is only one rule left.
2. Click on the “Graph” tab on the left and you should see a graph providing an overview of the rules (support/confidence) and the related items.

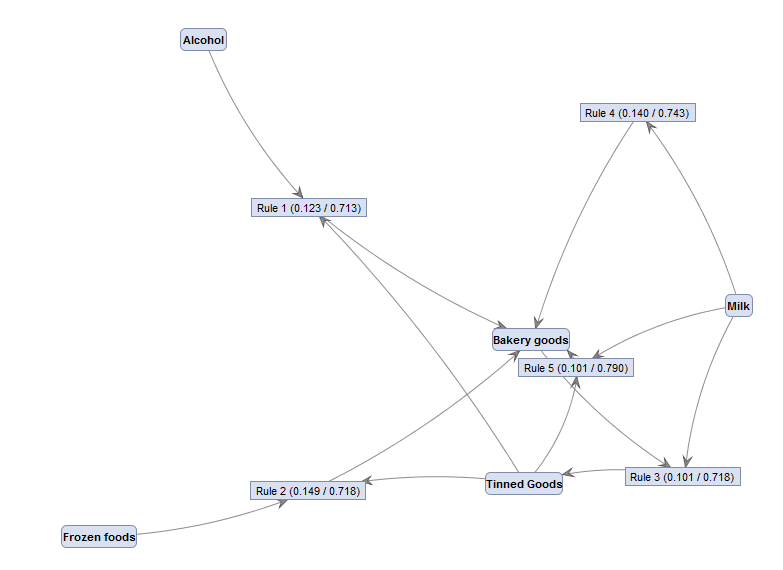


Figure : Graph of the generated rules and the related items.

The graph shows the relationship among the rules and shopping items. The arrows pointing into a rule indicates antecedent (premises) and arrows coming out of a rule indicates consequent (conclusion). For example, for Rule 1, we see that we have arrows from “Alcohol” and “Tinned Goods” pointing in and an arrow pointing out towards “Bakery goods”, hence the rule is {Alcohol, Tinned Goods} => {{Bakery goods}. The Support and Confidence of the rules is indicated in the box (0.123 / 0.713).

1. Click on the “Description” tab on the left and you should be able to see the rules in text form.

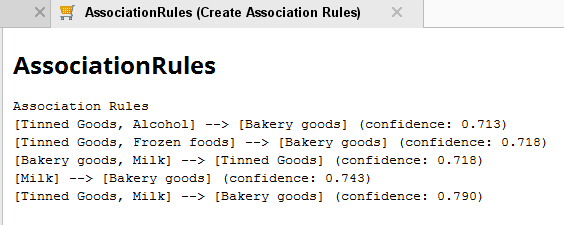


Figure : Description for Generated Association Rules.

The text uses pretty standard notation and should be self-explanatory. The confidence of each of the rules are displayed at the end of each of the rules.

# Using the Associations

Once we have the model (rules) generated, we can apply the rules to new shopping transactions and if any of the transactions matches the antecedent (premises), RapidMiner can calculates the confidence value for that transaction.

Let us now see how we can apply the rules we have generated on a new set of shopping transactions.

1. Download a copy of “shopping2.csv” file.
2. Add another “Read CSV” operator to the process and use the “Import Configuration Wizard” to import the shopping2.csv file. Remember to set the attributes correctly as you have done previously.

The shopping2.csv file contains 10 examples (records) as shown below:



Figure : Shopping2.csv

Notice that we have added the data such that all the rows meets one or more of the 5 rules except the last row. Let us see if the model can pick that up.

1. Add an “Apply Association Rules” operator and connect the process as shown in the figure below:

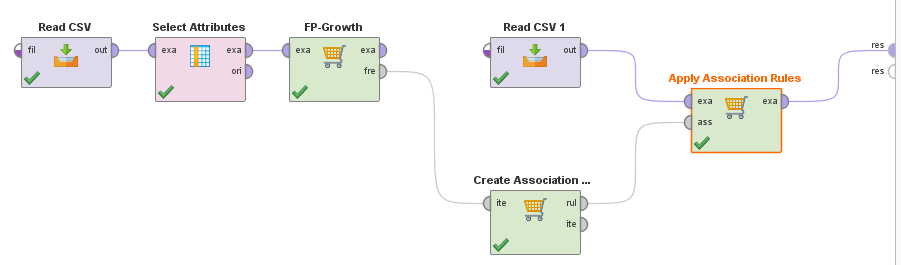


Figure : Applying association rules to new examples.

As can be seen from the figure, the “Apply Association Rules” operator reads in the new examples (records) as well as our generated association rules. It then applies the rules to the examples and generate a confidence for each of the examples.

1. Set the parameters of the “Apply Association Rules” operator as shown below:

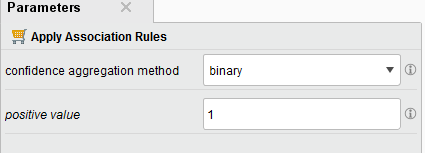


Figure : Parameters for the "APPLY ASSOCIATION Rules" operator.

By setting the “confidence aggregation method” to binary, the output results will indicate either a 0 or 1 with a value of 1 indicating that the example does match the premises for at least one of the rules. A value 0 indicates that is no match found for any of the rules.

The “positive value” parameter allows us to define the value used to indicate the item is bought. Recall that in our CSV file, value of 1 is used to indicate the item is bought and 0 is used to indicate the item is not bought.

1. Run the process to see the output results.

You should see a result as shown below:

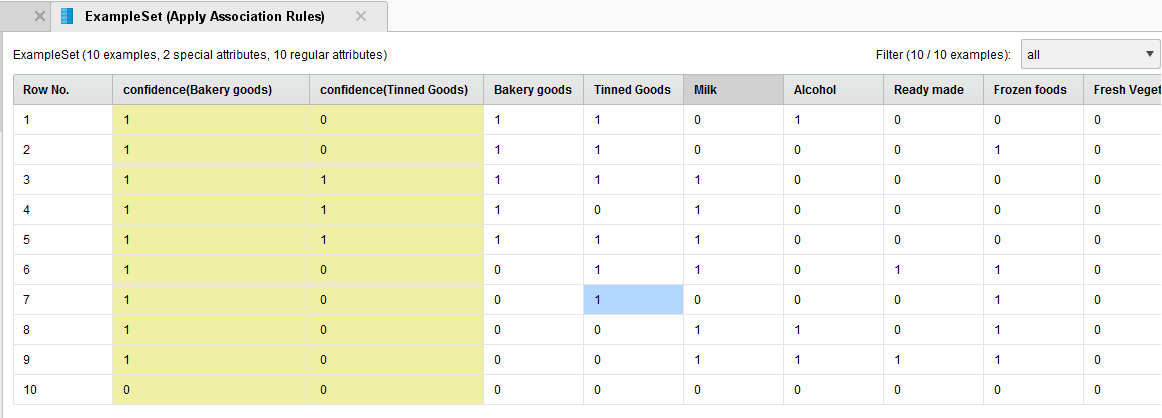


Figure : REsults from applying association rules.

Notice the two columns are added to our data – confidence(Bakery goods) and confidence(Tinned Goods). The columns are added based on the conclusions of the rules. Of the five rules we have generated, four of them have “Bakery goods” as the conclusion and one has “Tinned Goods” as conclusion (refer to Figure 13).

A vale of “1” in the confidence(Bakery goods) column indicates that the row matches premises of one or more of the four rules with Bakery goods as a conclusion. Similarly, a “1” in the confidence(Tinned Goods) column indicates that the row matches the premises of the rule with Tinned Goods as the conclusion.

As expected, all the rows except the last matches the premises for confidence(Bakery goods) and that row 3, 4 and 5 matches the rule for Tinned Goods.

We can get the operator to calculate the confidence value for us instead of simply displaying “1”. To do that we need to modify the “confidence aggregation method” parameter.

1. Change the “confidence aggregation method” parameter to “aggregated confidence” and run the process again. You should see the following result:

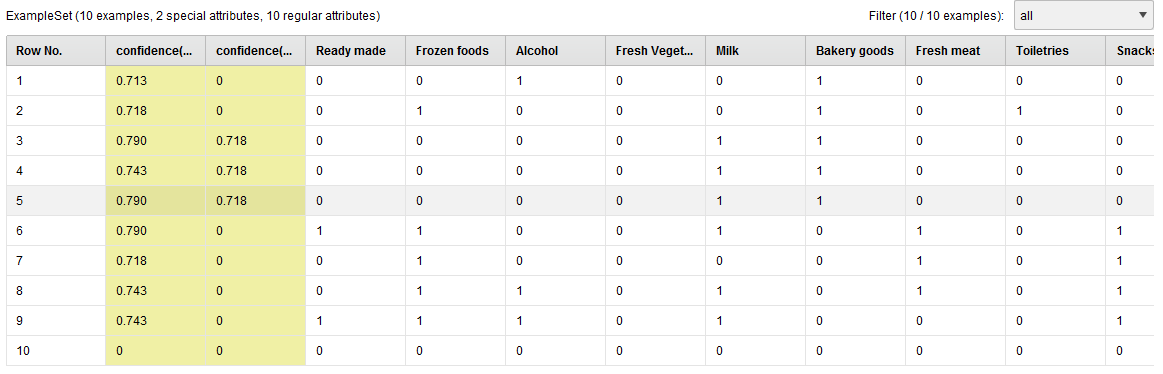


Figure : Results with Confidence values

As can be seen from the results, the confidence values are calculated and displayed instead of 1s and 0s.

# Summary

In this lesson you have been introduced to association rule generation using RapidMiner.

After this practical, you should now know how to:

• Generate frequent itemsets using FP-Growth operator.

• Mine association rules from the frequent itemsets using the Create Association Rules operator.

• View the resulting rules generated by the operator and modify filters to control the number of rules generated.

• Explain the meaning of rule confidence, support, rule support and lift

• Create a rule set and use this to identify those records whose conditions are related to a selected conclusion using the Apply Association Rules operator.

--- The End ----