1 Subject

Association Rules mining with Tanagra, R (arules package), Orange, RapidMiner, Knime and Weka.

This document extends a previous tutorial dedicated to the comparison of various implementations of association rules mining[1]. We had analyzed Tanagra, Orange and Weka. We extend here the comparison to R, RapidMiner and Knime.

We handle an attribute-value dataset. It is not the usual data format for the association rule mining where the "native" format is rather the transactional database. We see in this tutorial than some of tools can automatically recode the data. Others require an explicit transformation. Thus, we must find the right components and the correct sequence of treatments to produce the transactional data format. The process is not always easy according to the software.

2 Dataset and tools

We use the CREDIT-GERMAN.TXT[2] dataset. The characteristics of 1000 customers of a finance company are described. It comes from the UCI server[3]. The continuous attributes are discretized. We show below a sample of the dataset.

credit-german.xls

	A	B	C	D	E	F	G	H
1	checking_status	disc_duration	credit_history	purpose	disc_amount	savings_status	employment	personal_status
2	<0	lo_1_year	critical/other existing	radio/tv	1000_2000	no known savings	>=7	male single
3	0<=X<200	up_2_years	existing paid	radio/tv	up_2000	<100	1<=X<4	female div/dep/mar
4	no checking	lo_1_year	critical/other existing	education	up_2000	<100	4<=X<7	male single
5	<0	up_2_years	existing paid	furniture/equipment	up_2000	<100	4<=X<7	male single
6	<0	1_2_years	delayed previously	new car	up_2000	<100	1<=X<4	male single
7	no checking	up_2_years	existing paid	education	up_2000	no known savings	1<=X<4	male single
8	no checking	1_2_years	existing paid	furniture/equipment	up_2000	500<=X<1000	>=7	male single
9	0<=X<200	up_2_years	existing paid	used car	up_2000	<100	1<=X<4	male single
10	no checking	lo_1_year	existing paid	radio/tv	up_2000	>=1000	4<=X<7	male div/sep
11	0<=X<200	up_2_years	critical/other existing	new car	up_2000	<100	unemployed	male mar/wid
12	0<=X<200	lo_1_year	existing paid	new car	1000_2000	<100	<1	female div/dep/mar
13	<0	up_2_years	existing paid	business	up_2000	<100	<1	female div/dep/mar
14	0<=X<200	lo_1_year	existing paid	radio/tv	1000_2000	<100	1<=X<4	female div/dep/mar
15	<0	1_2_years	critical/other existing	new car	1000_2000	<100	>=7	male single
16	<0	1_2_years	existing paid	new car	1000_2000	<100	1<=X<4	female div/dep/mar
17	<0	1_2_years	existing paid	radio/tv	1000_2000	100<=X<500	1<=X<4	female div/dep/mar
18	no checking	1_2_years	critical/other existing	radio/tv	up_2000	no known savings	>=7	male single
19	<0	up_2_years	no credits/all paid	business	up_2000	no known savings	<1	male single
20	0<=X<200	1_2_years	existing paid	used car	up_2000	<100	>=7	female div/dep/mar
21	no checking	1_2_years	existing paid	radio/tv	up_2000	500<=X<1000	>=7	male single
22	no checking	lo_1_year	critical/other existing	new car	up_2000	<100	1<=X<4	male single
23	<0	lo_1_year	existing paid	radio/tv	up_2000	500<=X<1000	1<=X<4	male single

credit-german

All the tools analyzed in this tutorial can handle parameters on the confidence (0.75) and support (0.25). We can set also the maximum number of items in a rule (10). The tools studied in this tutorial

[1] http://data-mining-tutorials.blogspot.com/2008/10/association-rule-learning.html

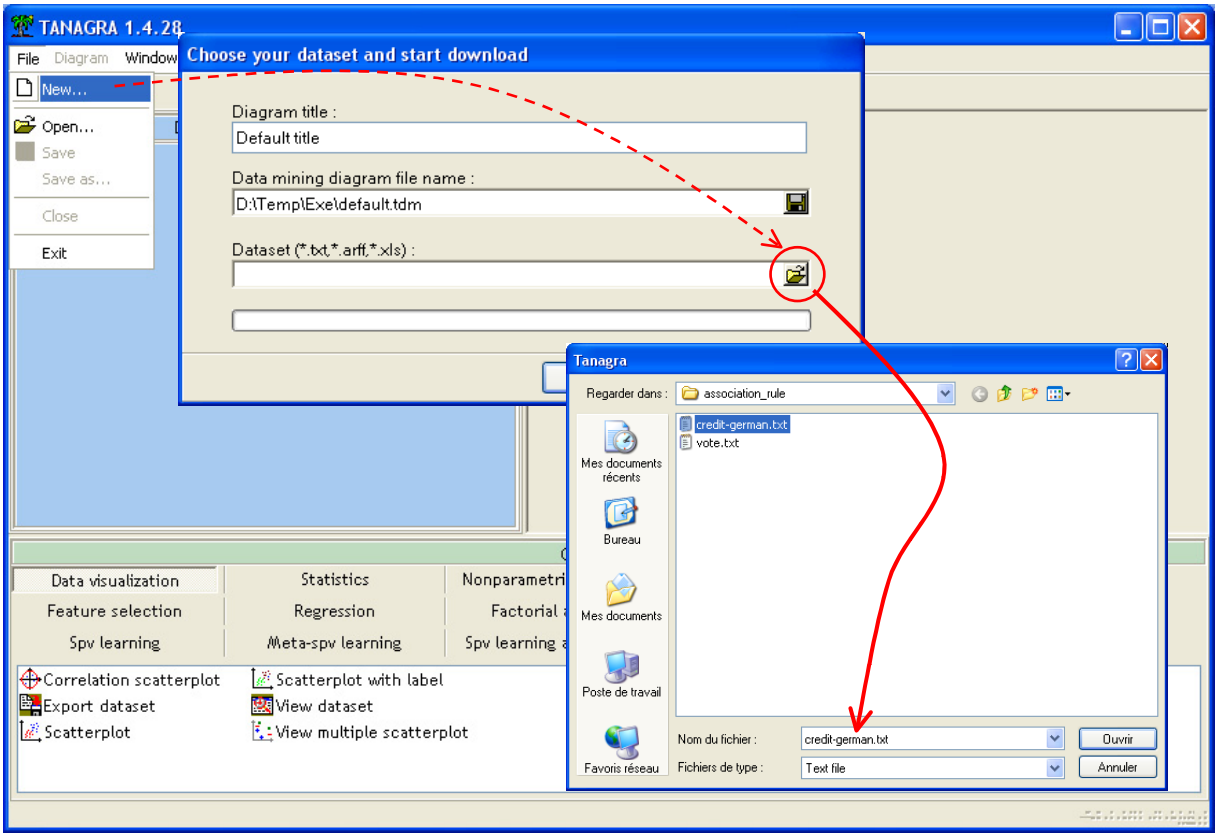
[2] http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/credit-german.zip

[3] http://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data)

are: Tanagra 1.4.28, R 2.7.2 (arules package 0.6-6), Orange 1.0b2, RapidMiner Community Edition, Knime 1.3.5 and Weka 3.5.6. These programs load the data and perform the calculations in memory. When the size of the database increases, the real bottleneck is the memory available on our personal computer.

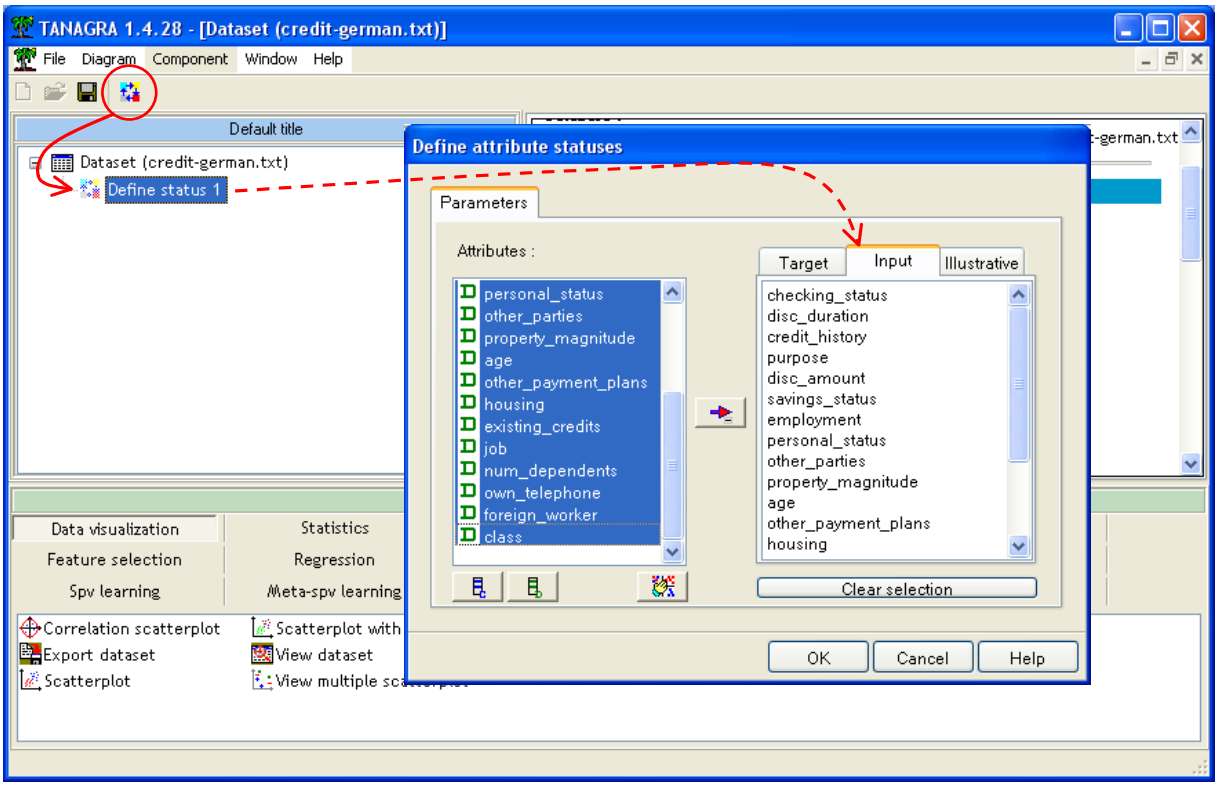
3 Tanagra (A Priori component)

Data importation and diagram initialization. After launching Tanagra, we activate the FILE / NEW menu. We select the CREDIT-GERMAN.TXT data file.

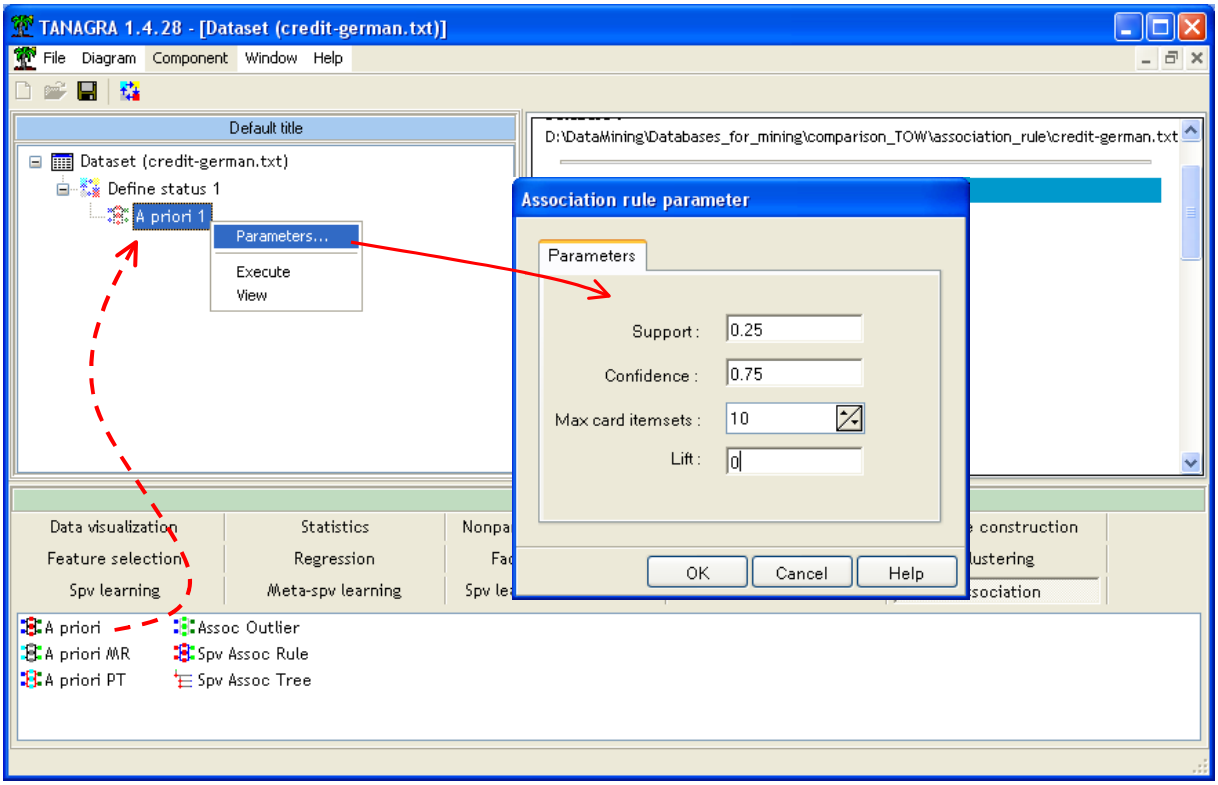


The dataset contains 19 variables and 1000 instances.

Specifying the status of the variables. In order to specify the status of each descriptor in the analysis, we use the DEFINE STATUS component. We set all the variables as INPUT.

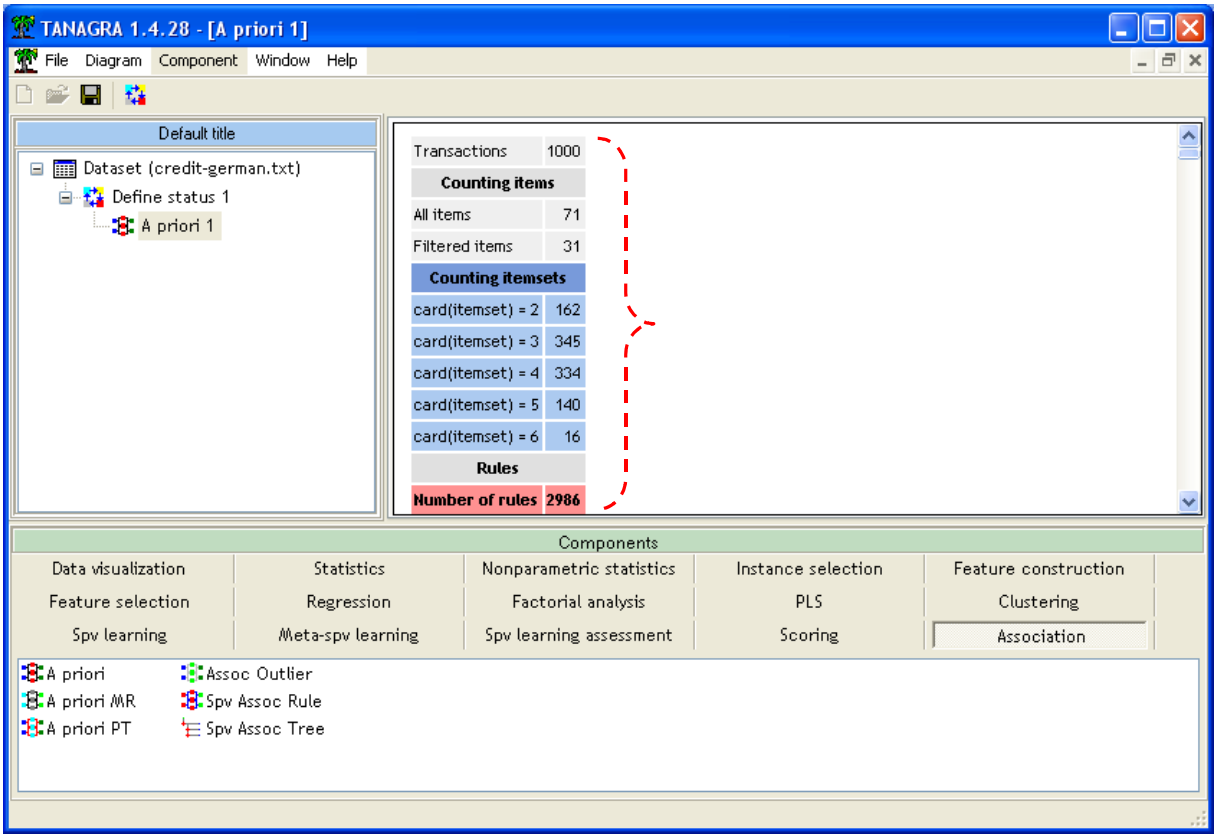


Extraction of association rules. We can insert the A PRIORI component now (ASSOCIATION tab). We activate the PARAMETERS menu in order to specify the parameters of the analysis.



The LIFT criterion is set to 0, thus it does not influence the extraction process. Our results will be directly comparable to the results of other tools which do not handle this parameter.

We click on VIEW menu.



Various indications are available: there are 71 items (attribute-value pair) into the dataset; 31 of them have a support >= 0.25 ; we see the number of itemsets of same length i.e. 162 itemsets with length = 2, etc.; thus, 2986 rules are extracted.

The rules are enumerated in the low part of the report. They are ranked according a decreasing value of the LIFT criterion.

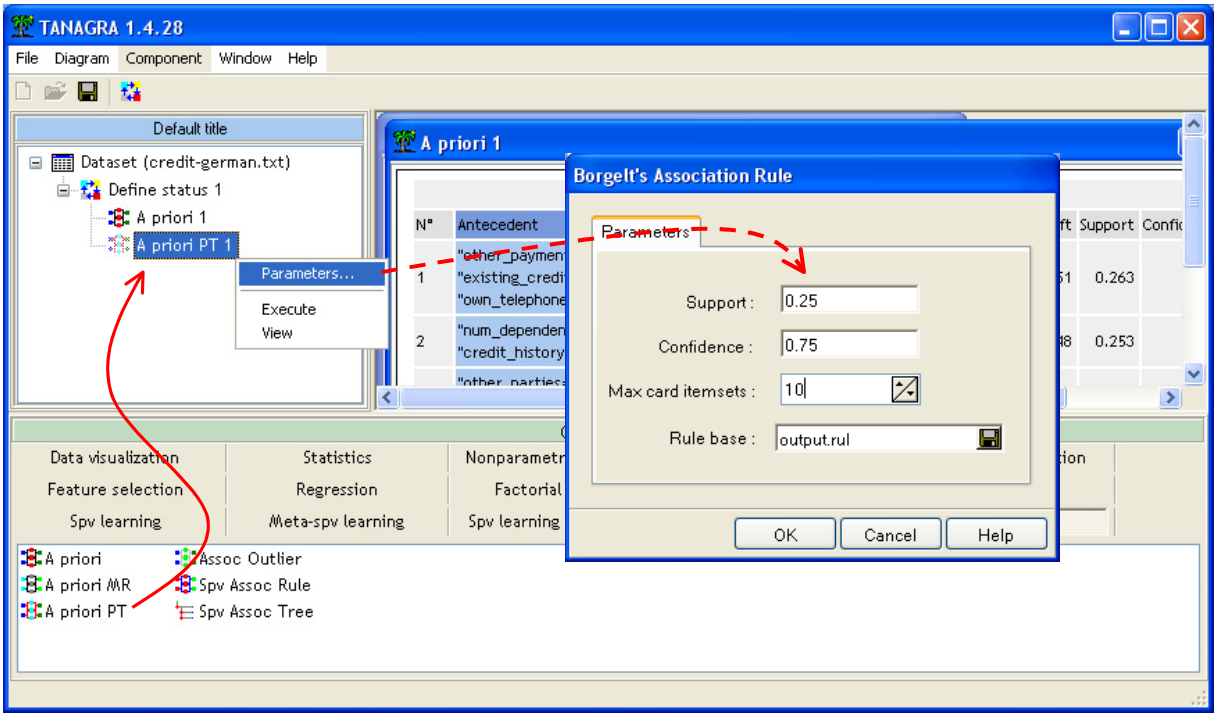
Number of rules : 2986

N°	Antecedent	Consequent	Lift	Support	Confidence
1	"other_payment_plans=none" - "existing_credits=one" - "own_telephone=none"	"credit_history=existing paid"	1.551	0.263	0.822
2	"num_dependents=one" - "class=good" - "credit_history=existing paid"	"other_payment_plans=none" - "existing_credits=one"	1.548	0.253	0.808
3	"other_parties=none" - "num_dependents=one" - "credit_history=existing paid"	"foreign_worker=yes" - "other_payment_plans=none" - "existing_credits=one"	1.534	0.319	0.769
4	"class=good" - "credit_history=existing paid"	"other_payment_plans=none" - "existing_credits=one"	1.534	0.289	0.801
5	"own_telephone=none" - "credit_history=existing paid"	"other_payment_plans=none" - "existing_credits=one"	1.527	0.263	0.797

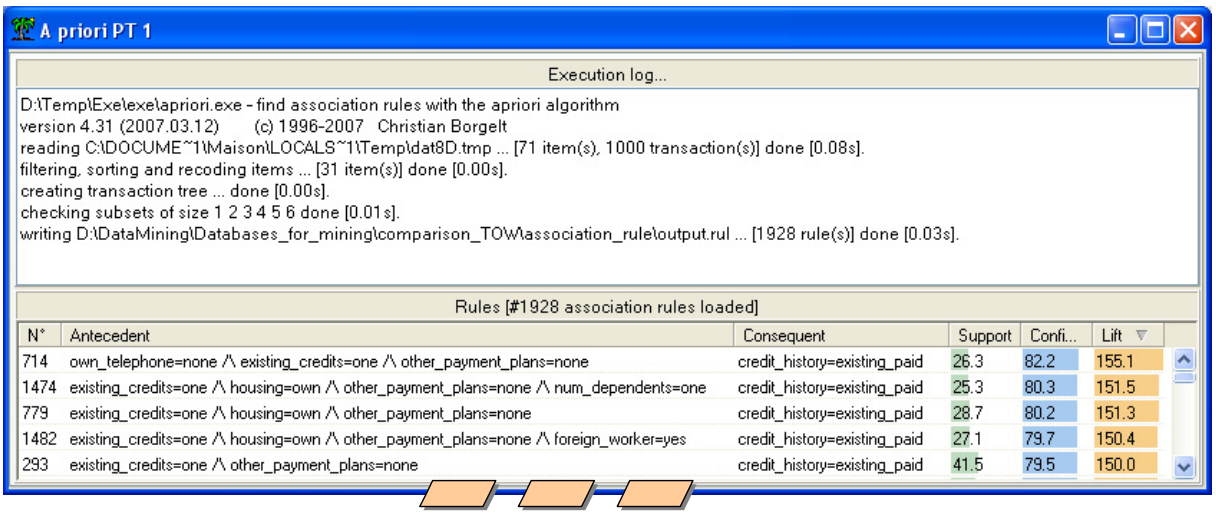
4 Tanagra (A Priori PT component)

There is a second component dedicated to the association rule extraction in Tanagra. This is an external program "apriori.exe" of Christian BORGELT (http://www.borgelt.net/apriori.html). We have already written a first tutorial where we show the functioning of this tool previously (http://data-mining-tutorials.blogspot.com/2008/11/association-rule-learning-using-prefix.html). BORGELT's program is really fast. But there is a slight limitation: only the rules with one item into the consequent can be generated. Therefore, for the same parameters above we obtain fewer rules.

In the previous diagram into Tanagra, we insert the A PRIORI PT component (ASSOCIATION tab). We set the parameters as follows.



We can click now on the VIEW menu.



In the upper part of the report, we can see the output of the BORGELT's program (version 4.31). In the lower part, the rules are enumerated. We can rank the rules according various numeric indicators by clicking on the column header. Because the component generates the rules with one item into the consequent, we obtain "only" 1928 rules.

5 R (arules package)

The « arules » package (http://cran.univ-lyon1.fr/web/packages/arules/index.html) allows extracting association rules with R (http://www.r-project.org/). This is also a version of the BORGELT's program, with the same limitation. In comparison with Tanagra, we must explicitly prepare the dataset before. The attribute-value representation must be transformed into a transactional data format. The operation is easy... if we read carefully the documentation.

Loading the « arules » package. We use the **library(.)** command in order to load the package

```
#charger le package
library(arules)
```

Data file importation and transformation. We import the dataset with the **read.table(.)** command, **summary(.)** gives some indications about the data characteristics.

```
#charger le fichier de données
setwd("D:/DataMining/Databases_for_mining/comparison_TOW/association_rule")
german <- read.table(file="credit-german.txt",header=T,dec=".",sep="\t")
summary(german)
```

We cannot extract rules from a data.frame, we must transform the internal format in "transactions".

```
#transformer les données attributs-variables
#en données transactionnelles
german.trans <- as(german,"transactions")
summary(german.trans)
```

We have always 71 items. R gives indications about the density of the dataset. We obtain a large number of rules if the density of the database is high.

```
> summary(german.trans)
transactions as itemMatrix in sparse format with
 1000 rows (elements/itemsets/transactions) and
 71 columns (items) and a density of 0.2676056

most frequent items:
      foreign_worker=yes       other_parties=none       num_dependents=one other_payment_plans=none
                     963                      907                      845                      814
            housing=own                  (Other)
                     713                    14758

element (itemset/transaction) length distribution:
sizes
  19
1000

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
     19      19      19      19      19      19

includes extended item information - examples:
                   labels       variables   levels
1       checking_status=<0 checking_status       <0
2    checking_status=>=200 checking_status    >=200
3 checking_status=0<=X<200 checking_status 0<=X<200

includes extended transaction information - examples:
  transactionID
1             1
2             2
3             3
```

Extraction of rules. The following instructions extract the association rules.

```
#extraction des règles
german.regles <- apriori(german.trans,parameter=
                  list(supp=0.25,conf=0.75,minlen=2,maxlen=10,target="rules"))
summary(german.regles)
```

We obtain again the BORGELT's program output.

```
> #extraction des règles
> german.regles <- apriori(german.trans,parameter=
+                   list(supp=0.25,conf=0.75,minlen=2,maxlen=10,target="rules"))

parameter specification:
 confidence minval smax arem  aval originalSupport support minlen maxlen target   ext
       0.75    0.1    1 none FALSE            TRUE    0.25      2      10  rules FALSE

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

apriori - find association rules with the apriori algorithm
version 4.21 (2004.05.09)        (c) 1996-2004   Christian Borgelt
set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[71 item(s), 1000 transaction(s)] done [0.00s].
sorting and recoding items ... [31 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 done [0.01s].
writing ... [1928 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

Some characteristics of the generated rule base are also available.

```
> summary(german.regles)
set of 1928 rules

rule length distribution (lhs + rhs):sizes
  2   3   4   5   6
112 487 783 476  70

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.000   3.000   4.000   3.951   5.000   6.000

summary of quality measures:
    support         confidence          lift
 Min.   :0.2500   Min.   :0.7500   Min.   :0.8894
 1st Qu.:0.2700   1st Qu.:0.8397   1st Qu.:0.9972
 Median :0.2990   Median :0.8906   Median :1.0121
 Mean   :0.3294   Mean   :0.8859   Mean   :1.0437
 3rd Qu.:0.3580   3rd Qu.:0.9485   3rd Qu.:1.0475
 Max.   :0.8800   Max.   :0.9937   Max.   :1.5507

mining info:
         data ntransactions support confidence
 german.trans          1000    0.25       0.75
```

Visualization of the rules. The inspect(.) command enables to visualize the details of rules. We show only the first 10 rules here.

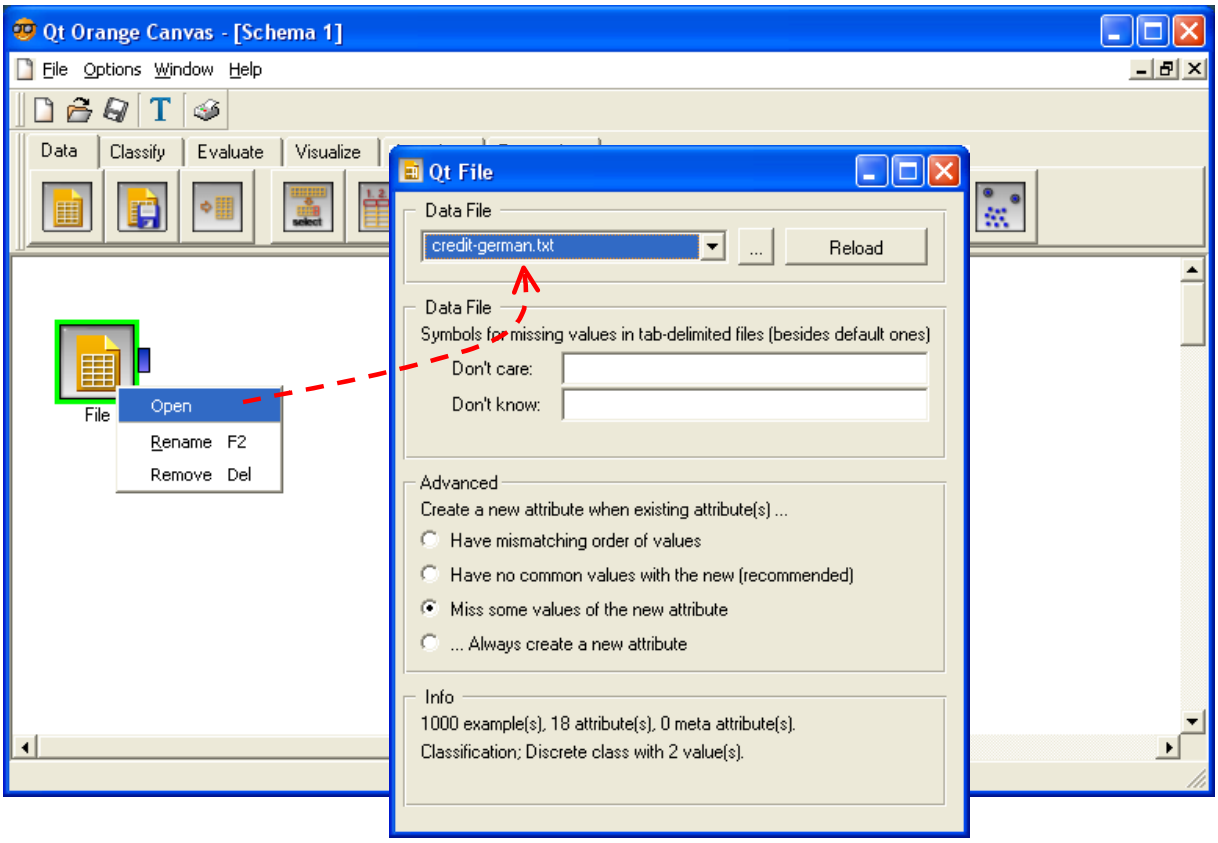
```
> #afficher les 10 premières règles trouvées
> inspect(german.regles[1:10])
   lhs                                      rhs                          support confidence      lift
1  {checking_status=0<=X<200}            => {foreign_worker=yes}           0.264  0.9814126 1.0191201
2  {checking_status=<0}                  => {foreign_worker=yes}           0.259  0.9452555 0.9815737
3  {purpose=radio/tv}                    => {num_dependents=one}           0.250  0.8928571 1.0566357
4  {purpose=radio/tv}                    => {foreign_worker=yes}           0.275  0.9821429 1.0198784
5  {property_magnitude=real estate}      => {foreign_worker=yes}           0.262  0.9290780 0.9647747
6  {credit_history=critical/other existing } => {other_payment_plans=none}   0.251  0.8566553 1.0524021
7  {credit_history=critical/other existing } => {other_parties=none}         0.268  0.9146758 1.0084628
8  {credit_history=critical/other existing } => {foreign_worker=yes}         0.279  0.9522184 0.9888042
9  {class=bad}                           => {num_dependents=one}           0.254  0.8466667 1.0019724
10 {class=bad}                           => {other_parties=none}           0.272  0.9066667 0.9996325
```

We can also rank the rules according to a rule quality indicator. We show here the first 5 rules according to the LIFT criterion. We obtain the same rules as the A PRIORI PT component of Tanagra.

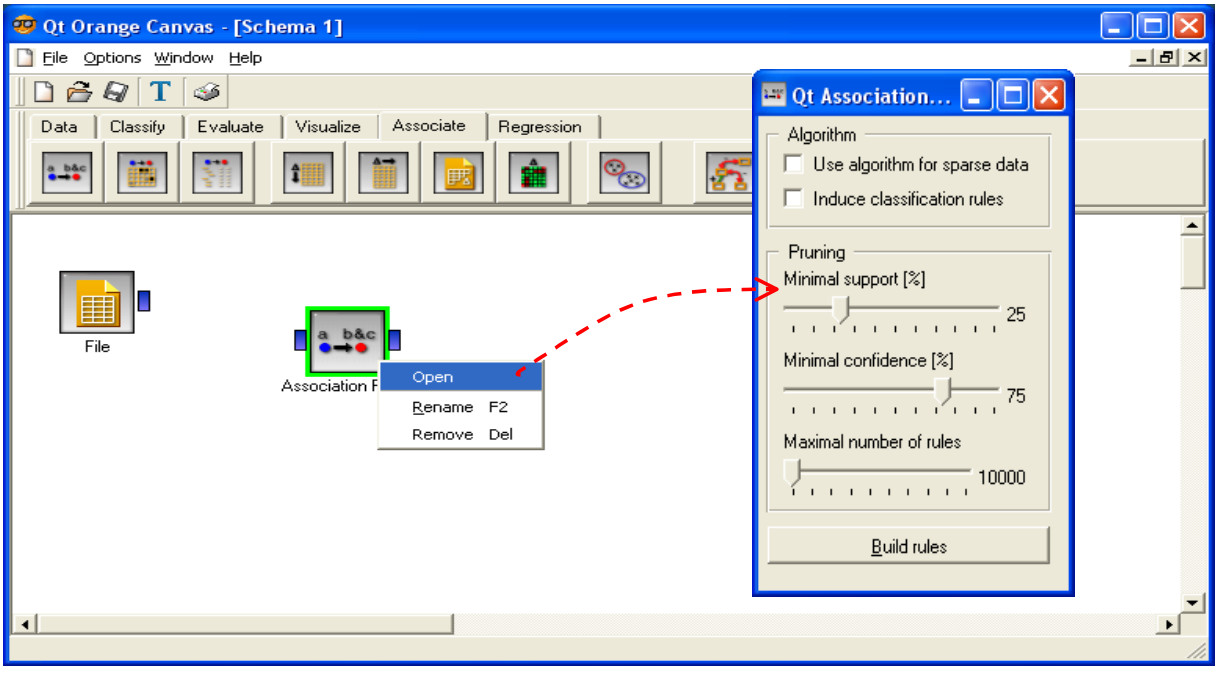
```
> #afficher les 5 règles avec le lift le + élevé
> regles.triees <- sort(german.regles,by="lift")
> inspect(regles.triees[1:5])
  lhs                                rhs                                support confidence     lift
1 {other_payment_plans=none,
   existing_credits=one,
   own_telephone=none}            => {credit_history=existing paid}    0.263  0.8218750 1.550708
2 {other_payment_plans=none,
   housing=own,
   existing_credits=one,
   num_dependents=one}            => {credit_history=existing paid}    0.253  0.8031746 1.515424
3 {other_payment_plans=none,
   housing=own,
   existing_credits=one}          => {credit_history=existing paid}    0.287  0.8016760 1.512596
4 {other_payment_plans=none,
   housing=own,
   existing_credits=one,
   foreign_worker=yes}            => {credit_history=existing paid}    0.271  0.7970588 1.503885
5 {other_payment_plans=none,
   existing_credits=one}          => {credit_history=existing paid}    0.415  0.7950192 1.500036
```

6 Orange

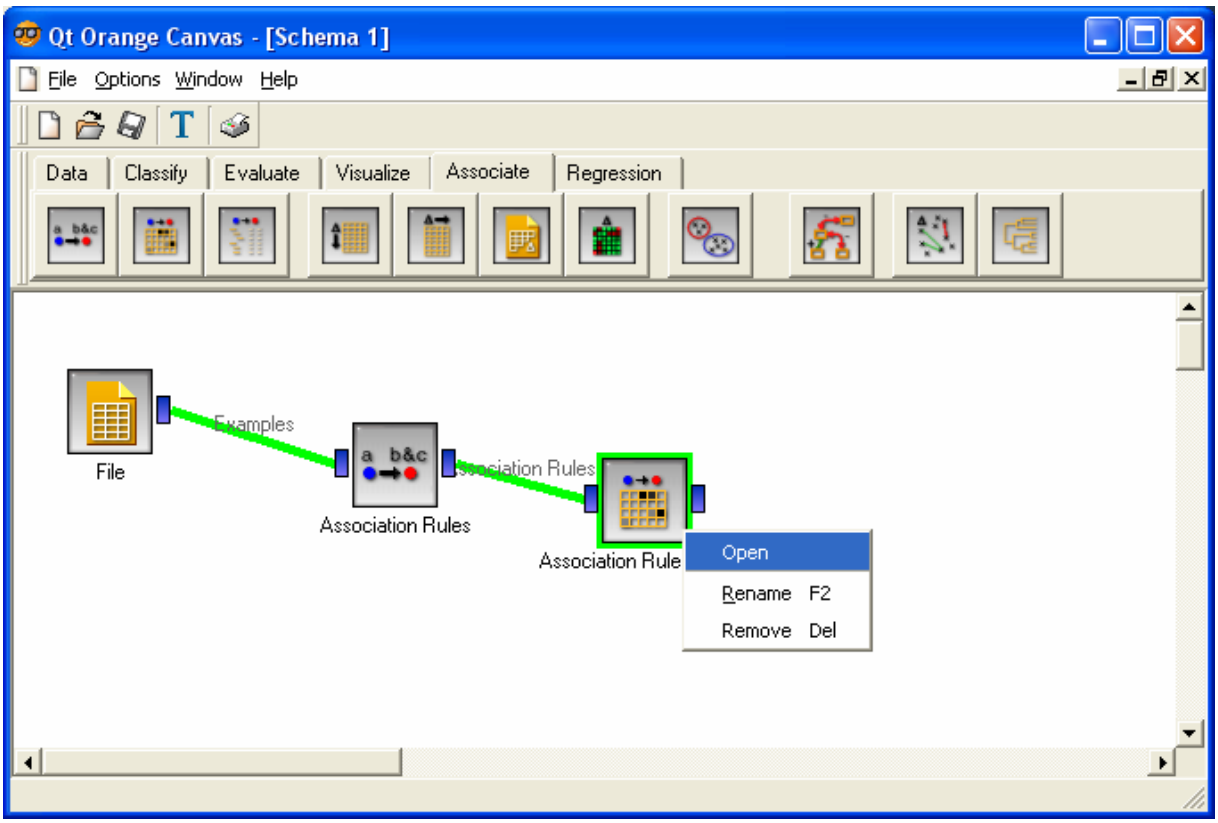
Creation of a "schema" and data importation. When we launch Orange, a new empty schema is available. We add the FILE component (DATA tab). We select the data file.



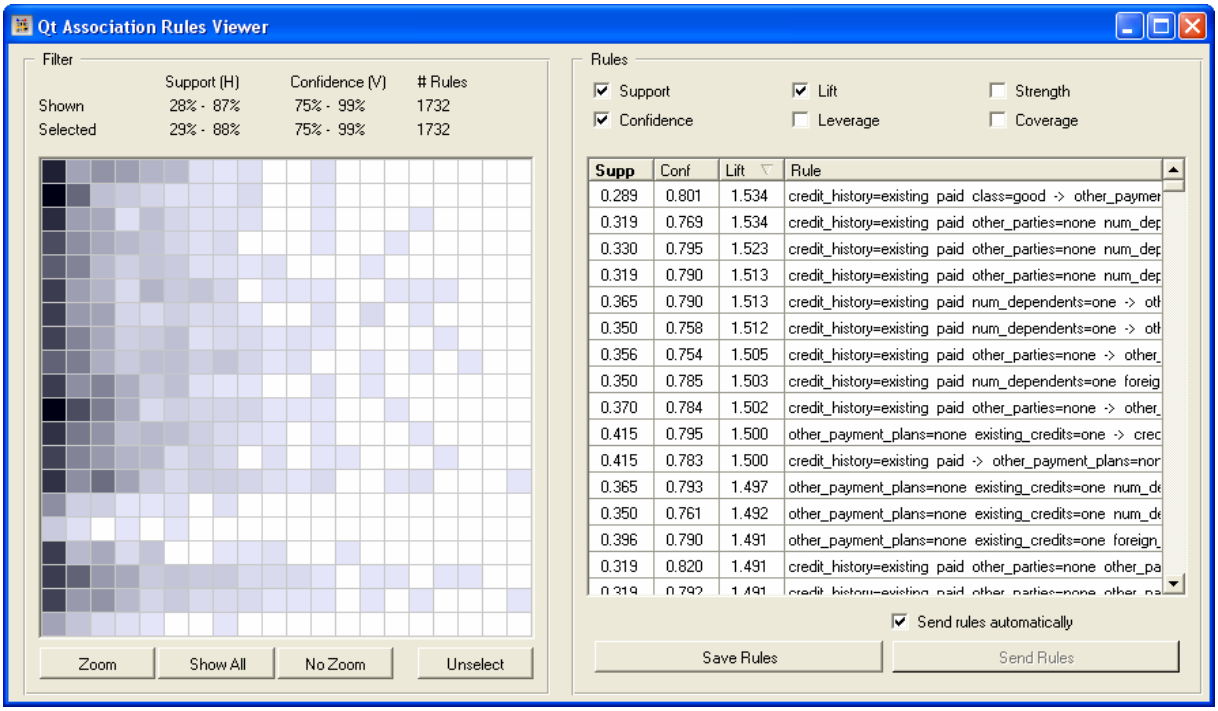
Rule extraction and visualization. We insert the ASSOCIATION RULES component (ASSOCIATE tab). We set the parameters of our analysis.



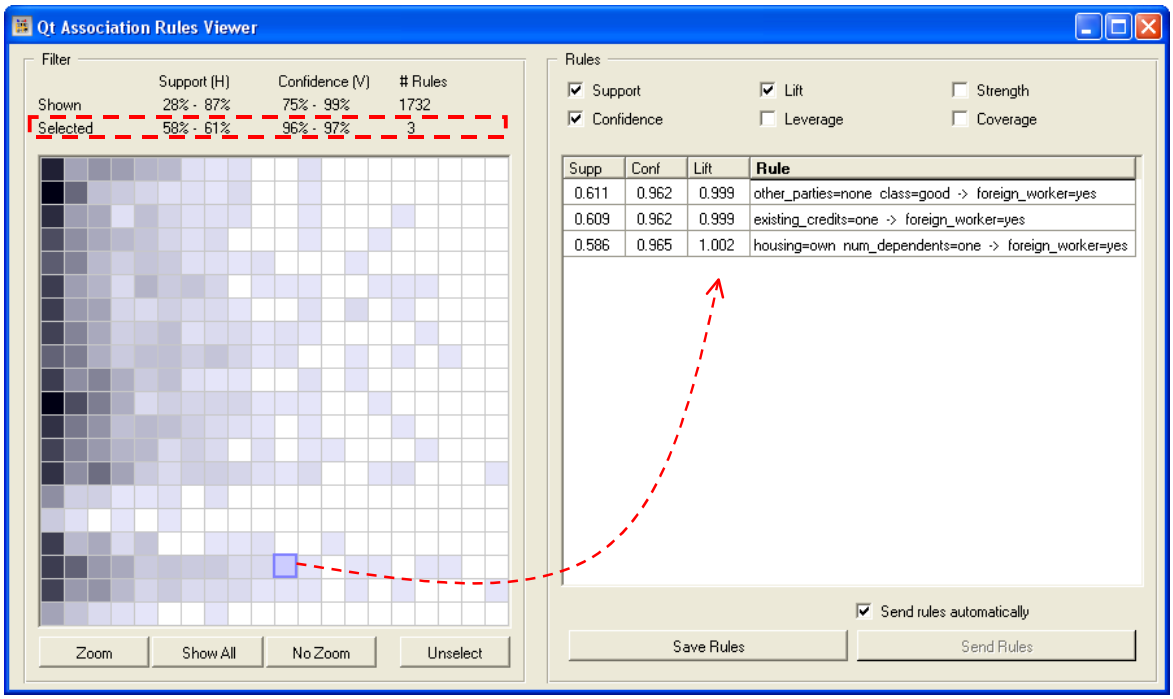
Then we add the ASSOCIATION RULES VIEWER component, we connect the components. We can now set the connection between FILE and ASSOCIATION RULES. The calculation is launched.



To see the rules, we click on the VIEW menu of the ASSOCIATION RULES VIEWER component.

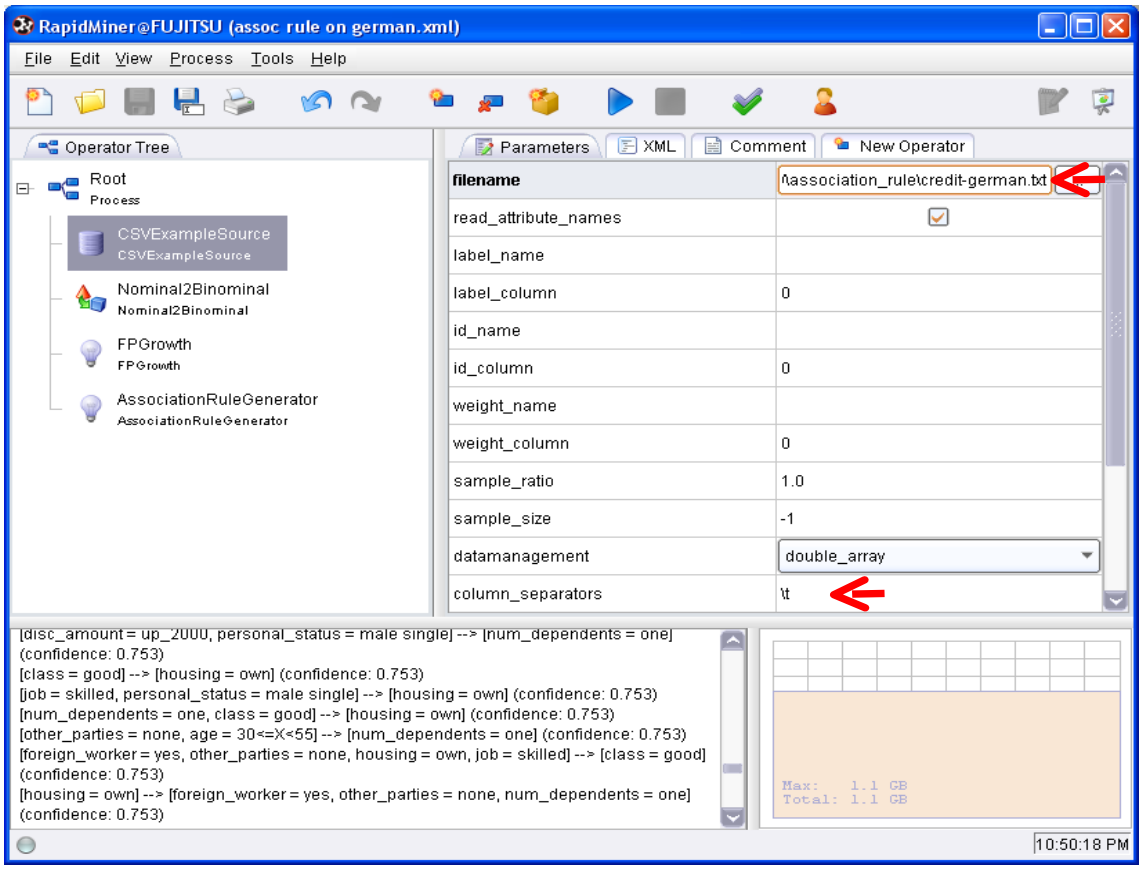


The visualization window is really original. We can graphically select a group of rules according to a range of the value of two criteria. Various criteria can be used; we can also rank the rules here.

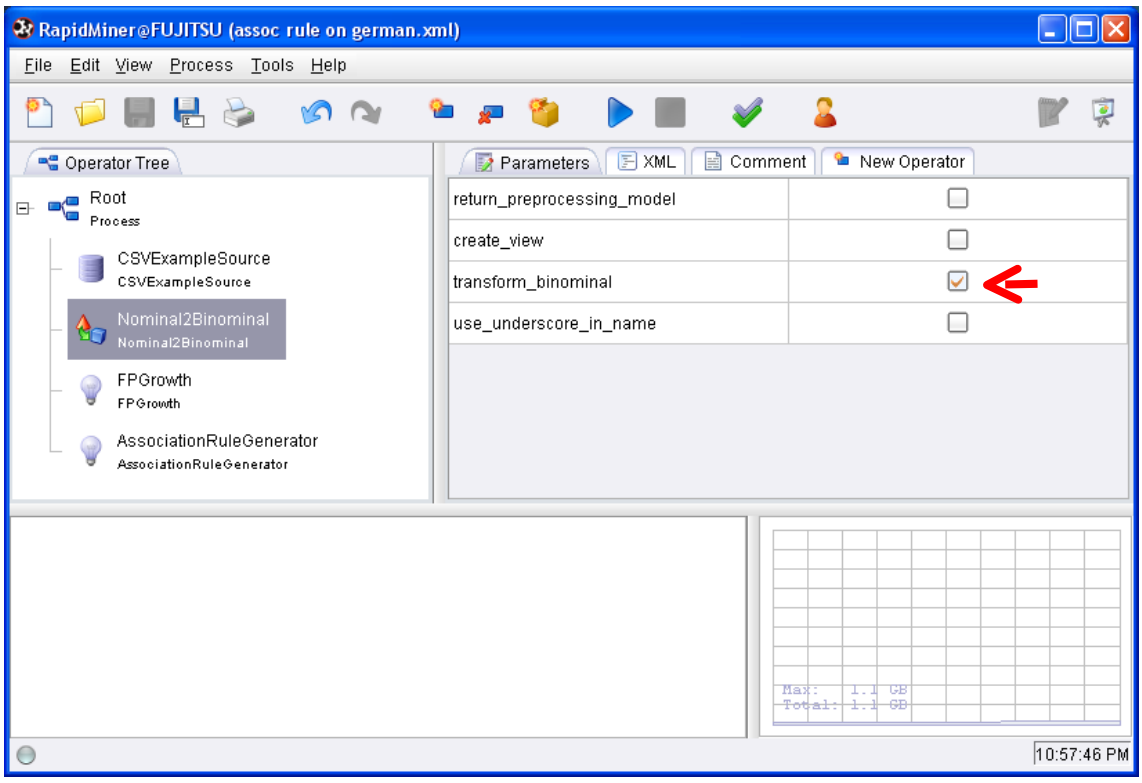


7 RapidMiner

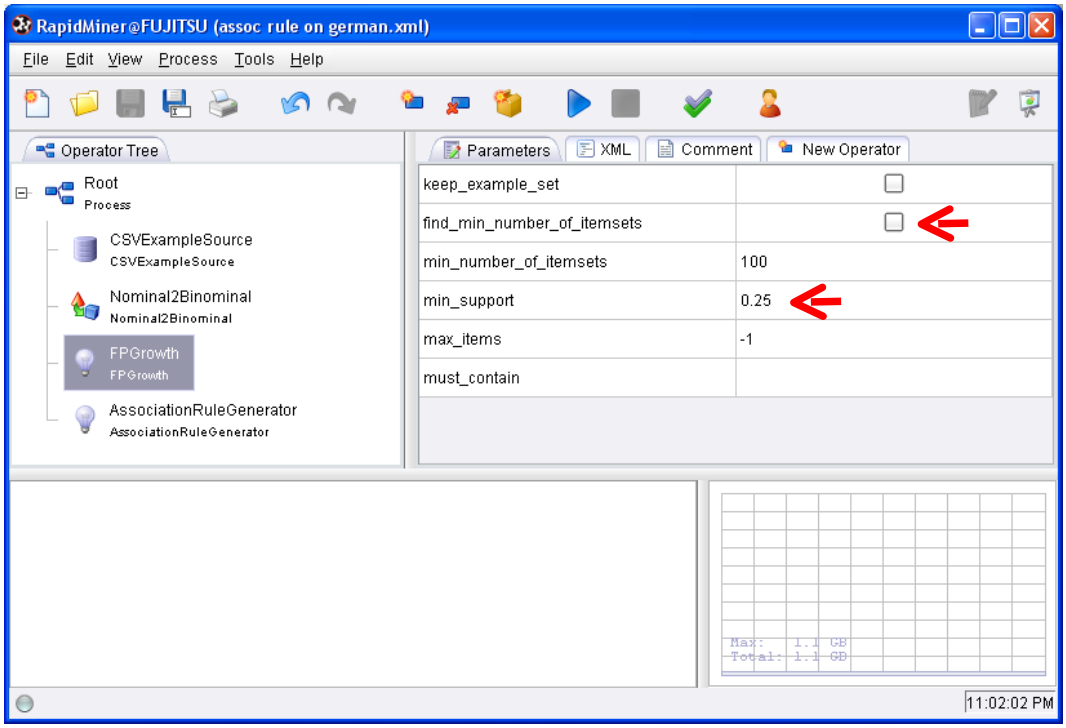
Creating a new operator tree. With RapidMiner, it is more convenient to define before all the sequences of operations before starting the computation. The first operator is **CSVEXAMPLESOURCE,** we set the file name and the column separator.



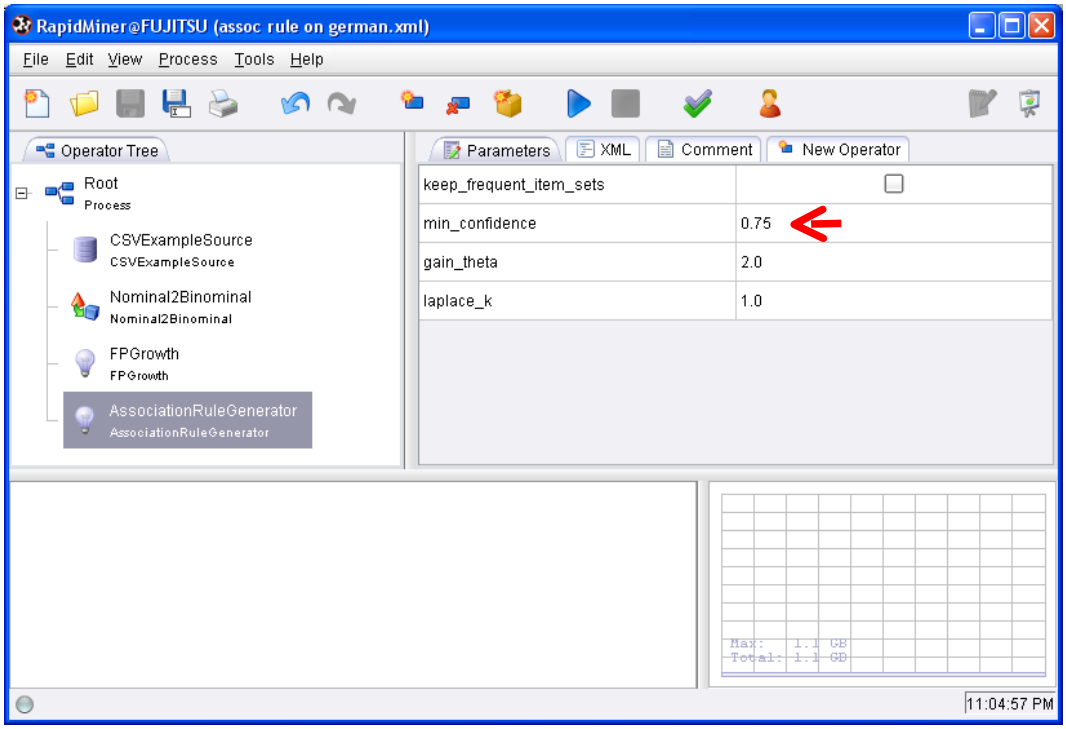
RapidMiner cannot create association rules from an attribute-value dataset. We must recode the variable into a set of binary columns with the **NOMINAL2BINOMIAL** component.



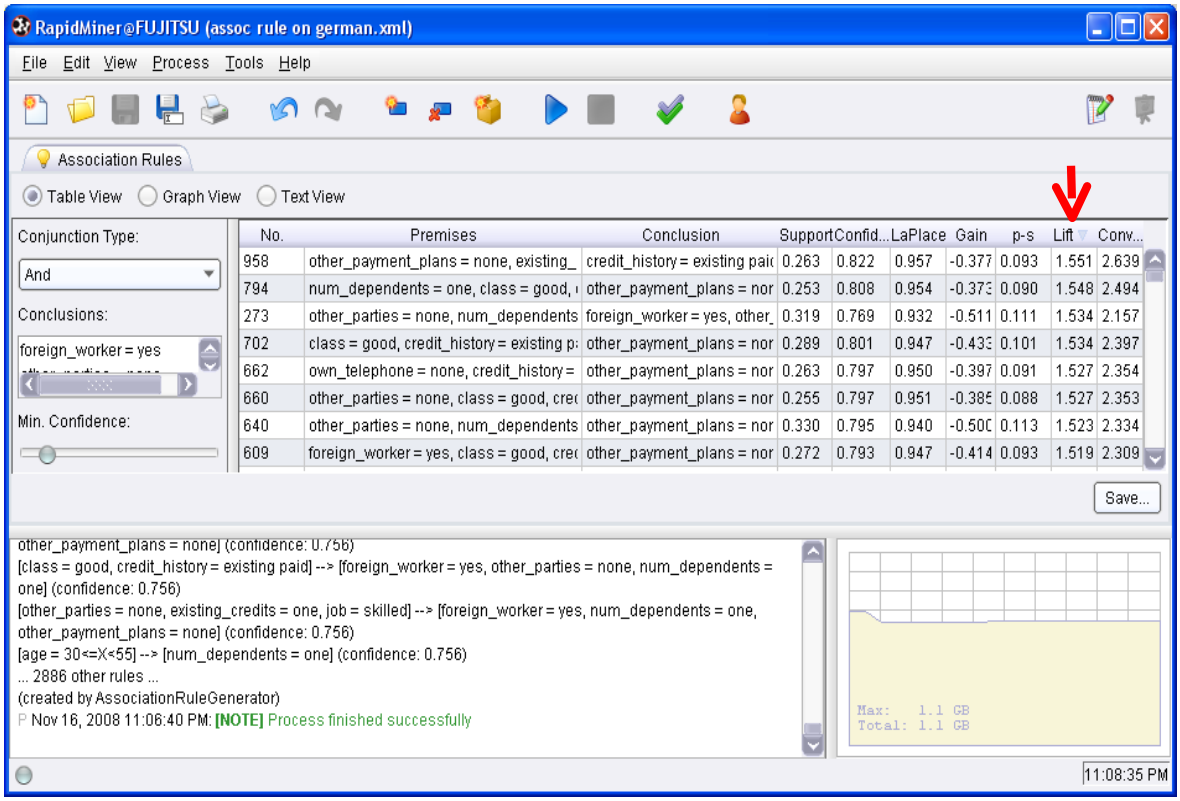
The extraction is carried out in 2 steps. First, with the **FPGROWTH** component, we generate the frequent itemsets. The settings must be defined carefully.



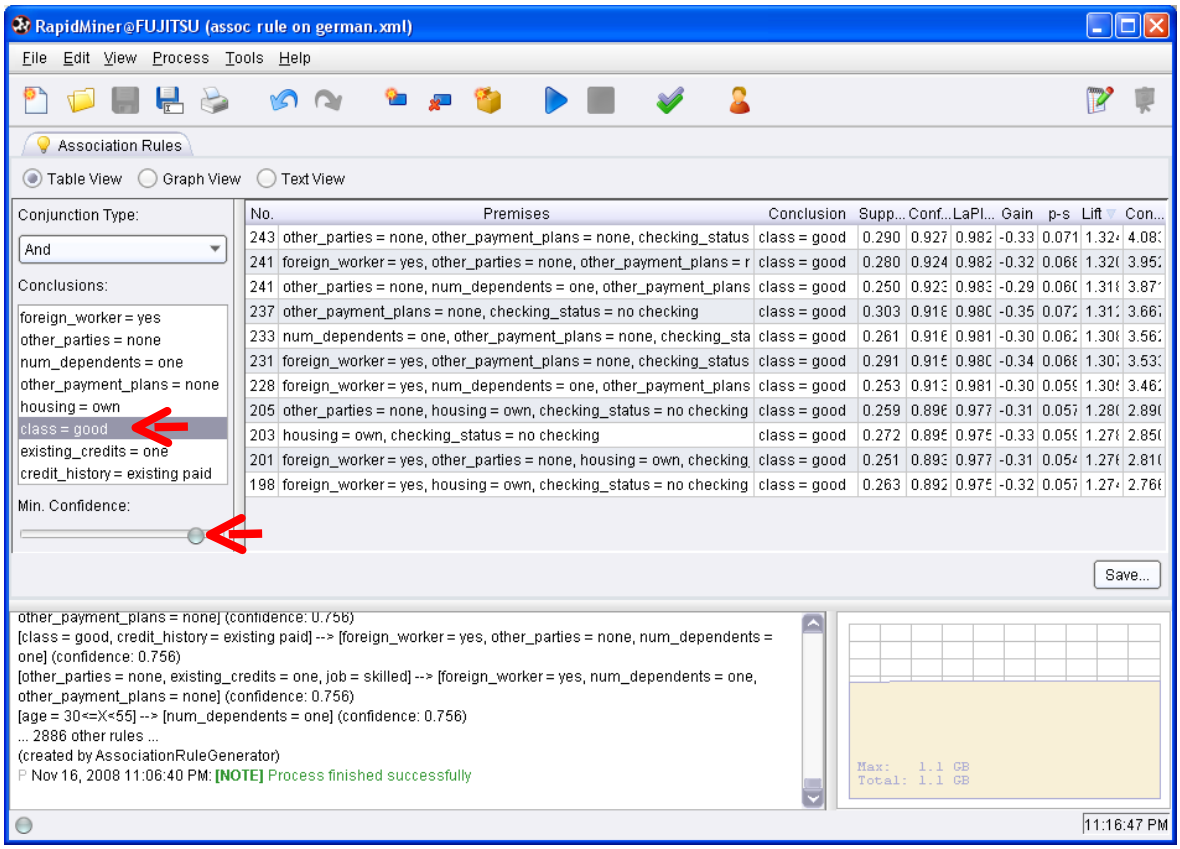
Then, with the **ASSOCIATIONRULEGENERATOR**, we generate the rules from the itemsets.



Launching the computation and visualization of rules. We launch the calculations by clicking on the PLAY menu into the toolbar.



We obtain 2986 rules, the same as the A PRIORI component of Tanagra. Various indicators enable to rank rules. Below, we sort the rules according to the LIFT criterion.

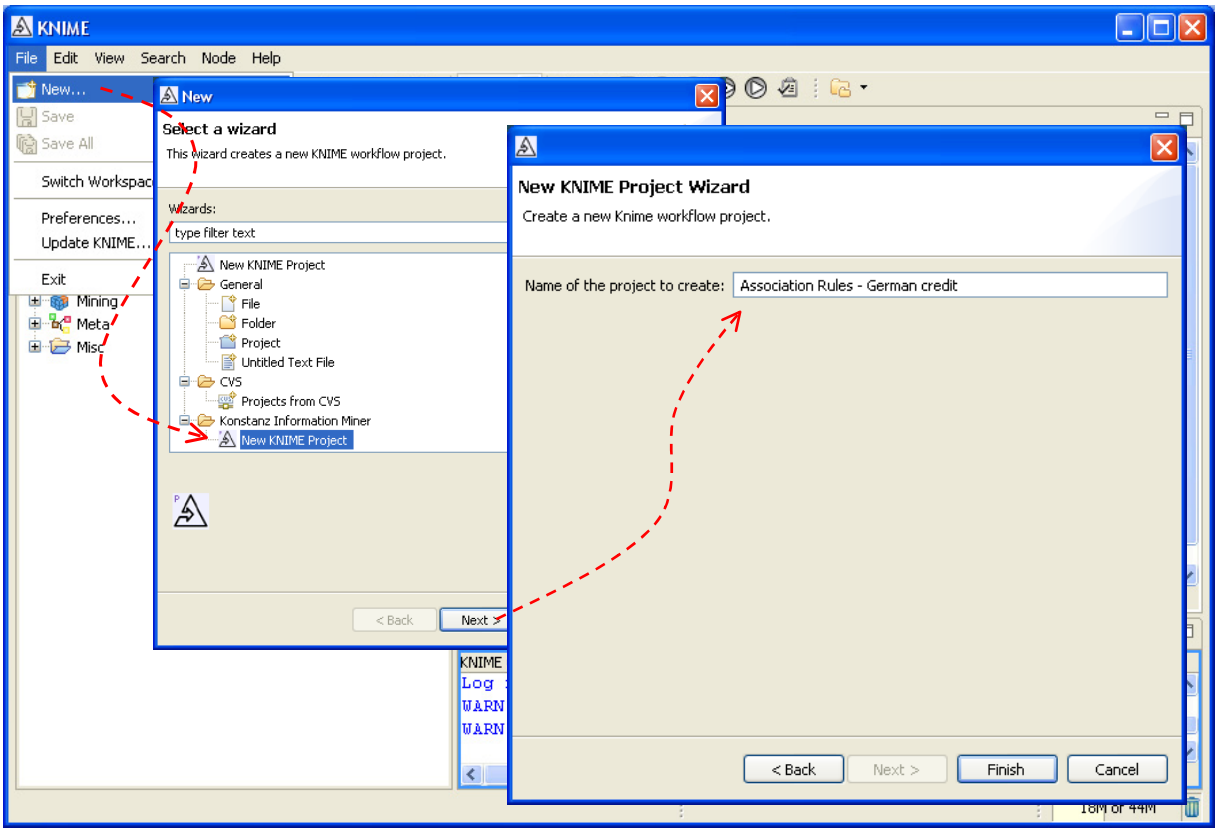


Another option is available. We can filter out the rules according to the presence or absence of an item or a set of items. This is very useful.

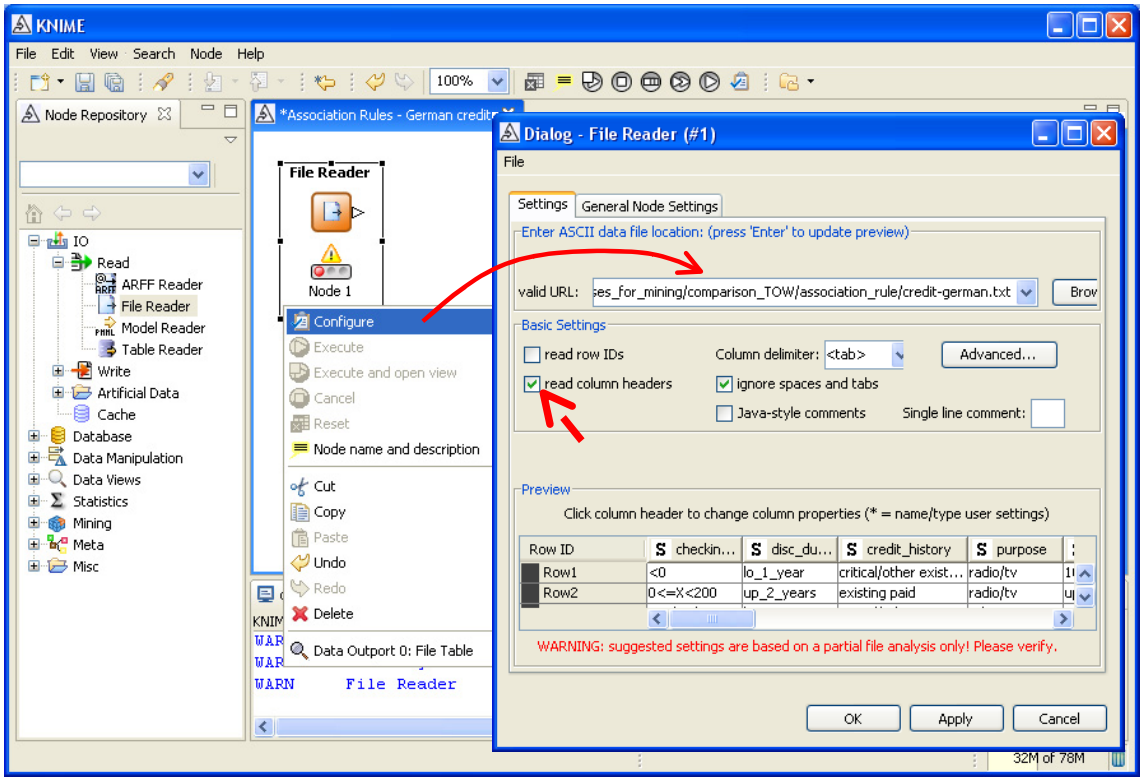
8 Knime

A double data preparation is necessary for KNIME before launching the learning algorithm: a coding 0/1 of the attribute-value dataset, followed by a transformation into transactional data.

Workflow creation and data importation. We create a new project by clicking on the FILE / NEW menu. We choose a KNIME project and we specify the name of the project.

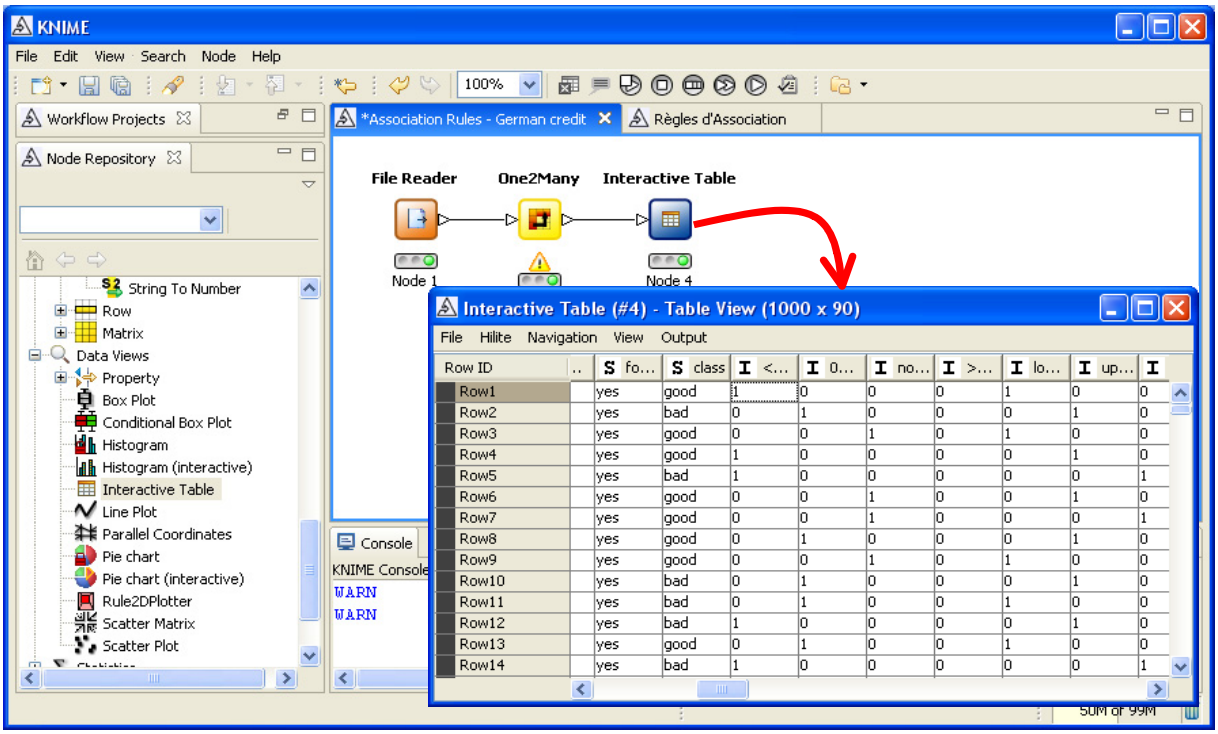


An empty workflow appears. We insert the file access component (FILE READER). We select our data file (CONFIGURE menu). We click on the EXECUTE menu in order to load the dataset.



First coding step. First we must recode the attribute-value dataset into a binary dataset. We use the ONE2MANY component (DATA MANIPULATION / COLUMN). We connect this component to the

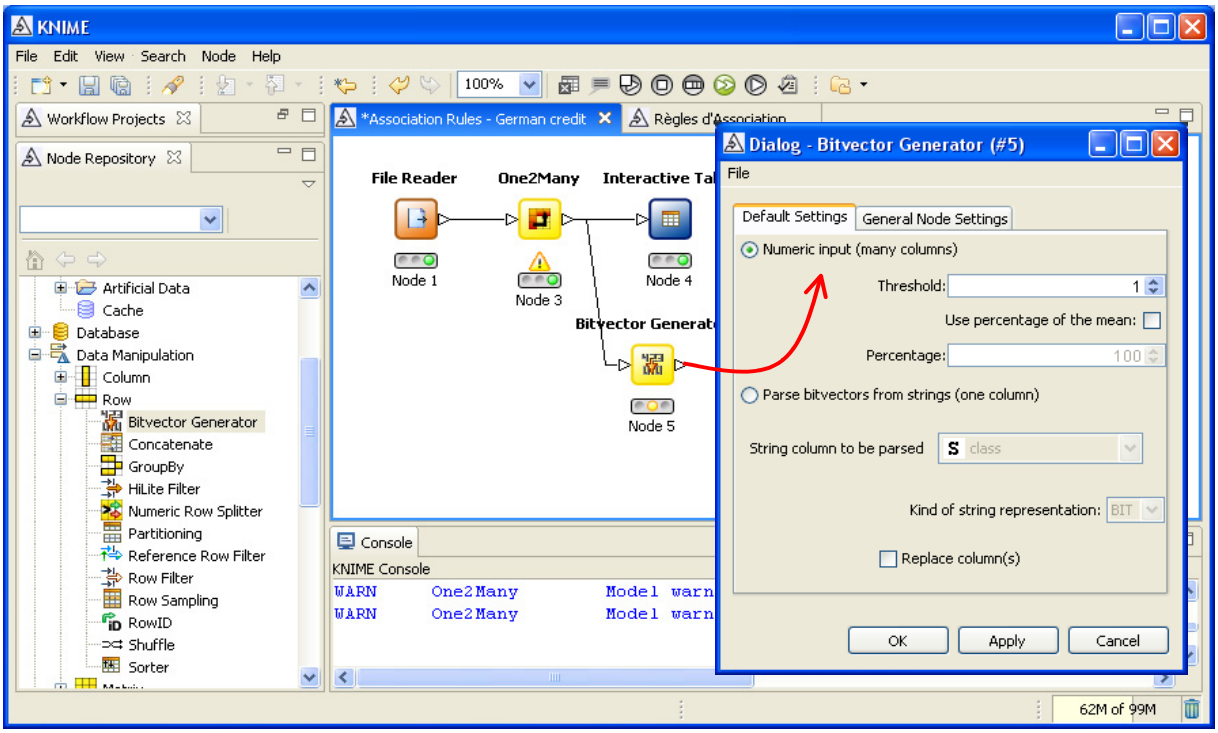
previous one. We add an INTERACTIVE TABLE component (DATA VIEWS), that we connect to ONE2MANY, in order to view the new dataset.



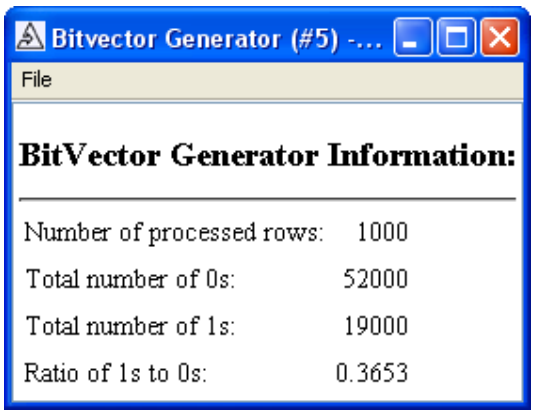
Row ID	..	S fo...	S class	I <...	I 0...	I no...	I >...	I lo...	I up...	I
Row1		yes	good	1	0	0	0	1	0	0
Row2		yes	bad	0	1	0	0	0	1	0
Row3		yes	good	0	0	1	0	1	0	0
Row4		yes	good	1	0	0	0	0	1	0
Row5		yes	bad	1	0	0	0	0	0	1
Row6		yes	good	0	0	1	0	0	1	0
Row7		yes	good	0	0	1	0	0	0	1
Row8		yes	good	0	1	0	0	0	1	0
Row9		yes	good	0	0	1	0	1	0	0
Row10		yes	bad	0	1	0	0	0	1	0
Row11		yes	bad	0	1	0	0	1	0	0
Row12		yes	bad	1	0	0	0	0	1	0
Row13		yes	good	0	1	0	0	1	0	0
Row14		yes	bad	1	0	0	0	0	0	1

There are 90 columns now. To the previous 19 variables are added 71 binary variables.

Second coding step. The first transformation is not enough. We must go through an internal format specific using the BITVECTOR component (DATA MANIPULATION / COLUMN). We connect it to ONE2MANY. We click on the CONFIGURE menu, we select only the binary variables.

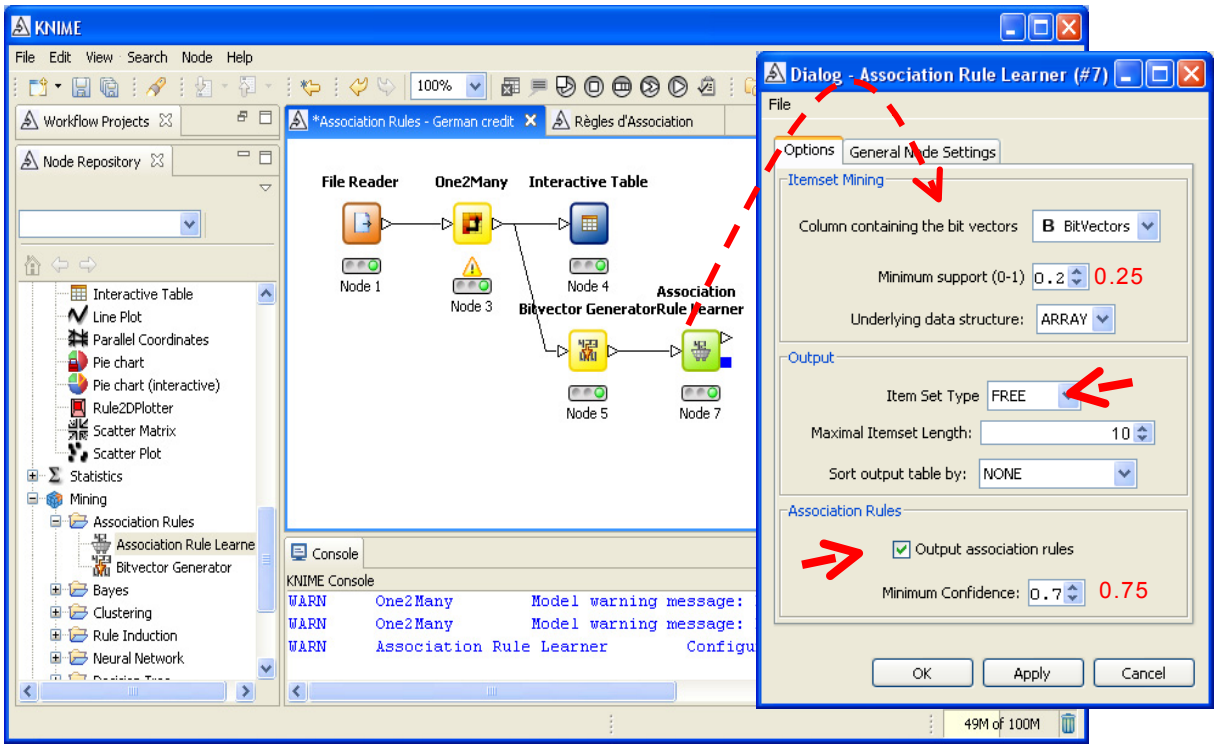


By clicking on the EXECUTE AND OPEN VIEW menu, we obtain a description of the generated transactional dataset.

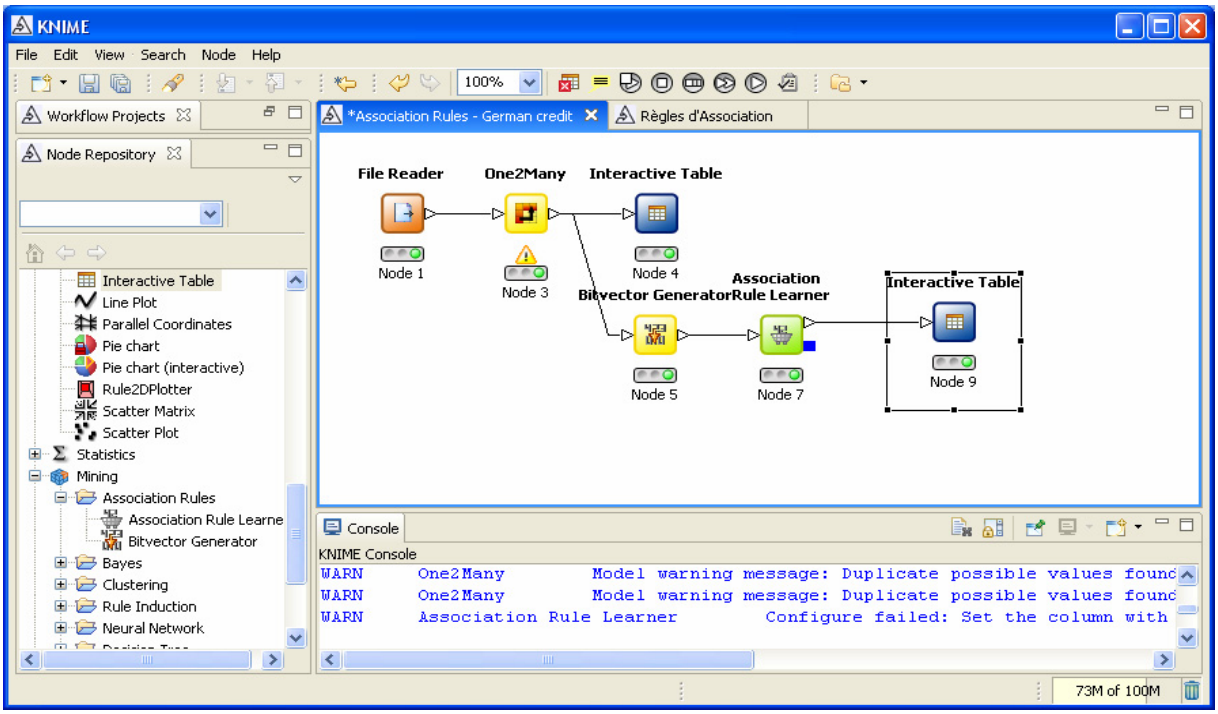


To obtain the information of density of R, we make 19000/(19000+52000) = 26.76%.

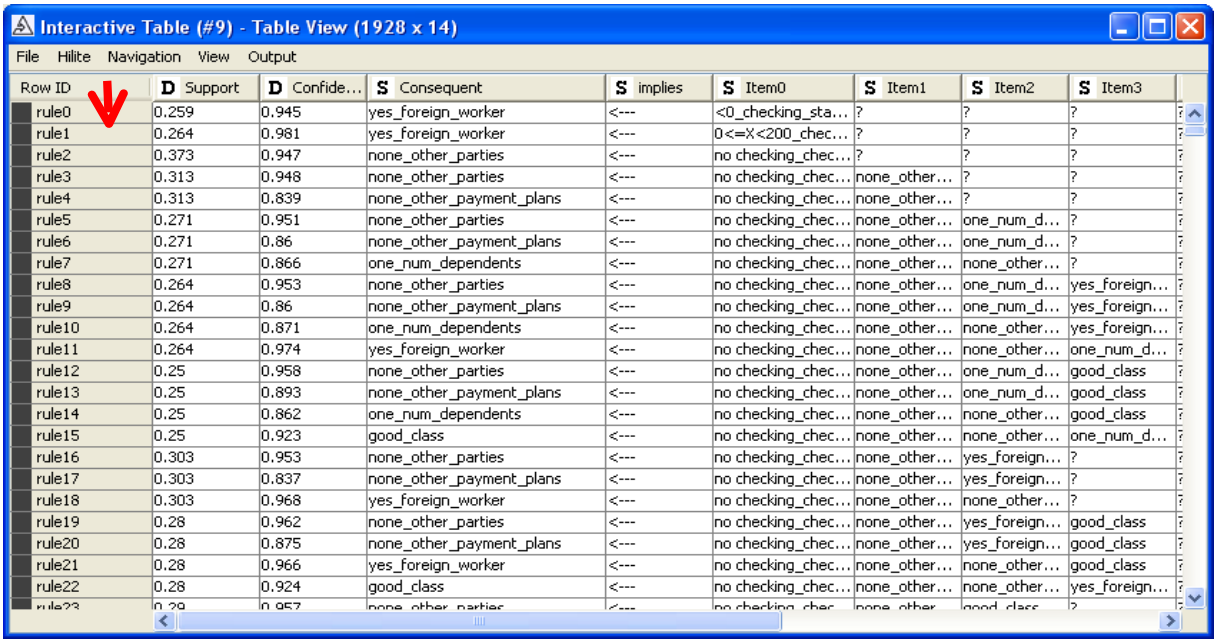
Extraction of rules. We use the ASSOCIATION RULE LEARNER component in order to extract rules. We set the following parameters.



The INTERACTIVE TABLE component enables to visualize the rules.



KNIME extracts only the rules with on item into the consequent. 1928 rules are generated. The visualization tool has not interactive functionalities (ranking or filtering rules).

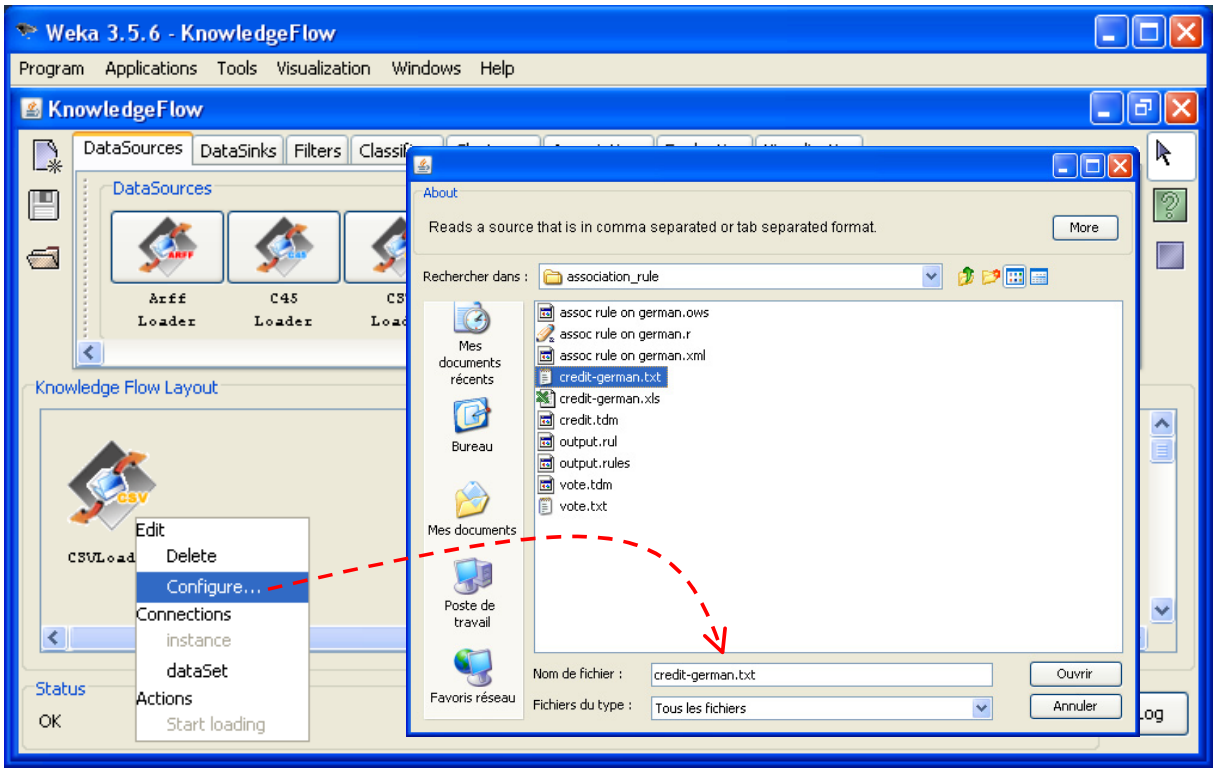


Row ID	Support	Confide...	Consequent	implies	Item0	Item1	Item2	Item3
rule0	0.259	0.945	yes_foreign_worker	<---	<0_checking_sta...	?	?	?
rule1	0.264	0.981	yes_foreign_worker	<---	0<=X<200_chec...	?	?	?
rule2	0.373	0.947	none_other_parties	<---	no checking_chec...	?	?	?
rule3	0.313	0.948	none_other_parties	<---	no checking_chec...	none_other...	?	?
rule4	0.313	0.839	none_other_payment_plans	<---	no checking_chec...	none_other...	?	?
rule5	0.271	0.951	none_other_parties	<---	no checking_chec...	none_other...	one_num_d...	?
rule6	0.271	0.86	none_other_payment_plans	<---	no checking_chec...	none_other...	one_num_d...	?
rule7	0.271	0.866	one_num_dependents	<---	no checking_chec...	none_other...	none_other...	?
rule8	0.264	0.953	none_other_parties	<---	no checking_chec...	none_other...	one_num_d...	yes_foreign...
rule9	0.264	0.86	none_other_payment_plans	<---	no checking_chec...	none_other...	one_num_d...	yes_foreign...
rule10	0.264	0.871	one_num_dependents	<---	no checking_chec...	none_other...	none_other...	yes_foreign...
rule11	0.264	0.974	yes_foreign_worker	<---	no checking_chec...	none_other...	none_other...	one_num_d...
rule12	0.25	0.958	none_other_parties	<---	no checking_chec...	none_other...	one_num_d...	good_class
rule13	0.25	0.893	none_other_payment_plans	<---	no checking_chec...	none_other...	one_num_d...	good_class
rule14	0.25	0.862	one_num_dependents	<---	no checking_chec...	none_other...	none_other...	good_class
rule15	0.25	0.923	good_class	<---	no checking_chec...	none_other...	none_other...	one_num_d...
rule16	0.303	0.953	none_other_parties	<---	no checking_chec...	none_other...	yes_foreign...	?
rule17	0.303	0.837	none_other_payment_plans	<---	no checking_chec...	none_other...	yes_foreign...	?
rule18	0.303	0.968	yes_foreign_worker	<---	no checking_chec...	none_other...	none_other...	?
rule19	0.28	0.962	none_other_parties	<---	no checking_chec...	none_other...	yes_foreign...	good_class
rule20	0.28	0.875	none_other_payment_plans	<---	no checking_chec...	none_other...	yes_foreign...	good_class
rule21	0.28	0.966	yes_foreign_worker	<---	no checking_chec...	none_other...	none_other...	good_class
rule22	0.28	0.924	good_class	<---	no checking_chec...	none_other...	none_other...	yes_foreign...

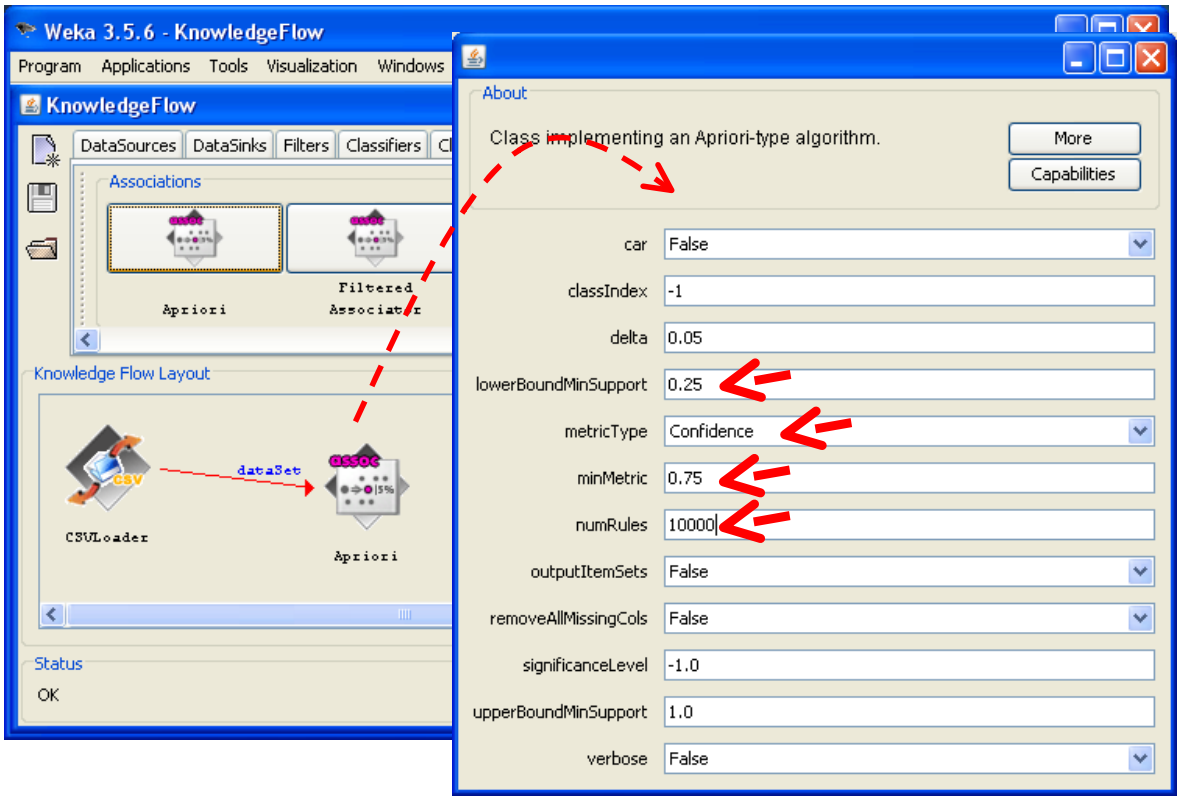
9 Weka

Like RapidMiner, it is more convenient to define the whole treatment before launching the computation with Weka. We use the Knowledge Flow in this tutorial.

Specifying the operations. When we launch Weka, we click on the APPLICATIONS / KNOWLEDGEFLOW menu. First, we must insert the data access component CSV LOADER (DATASOURCES tab). We select the data file.

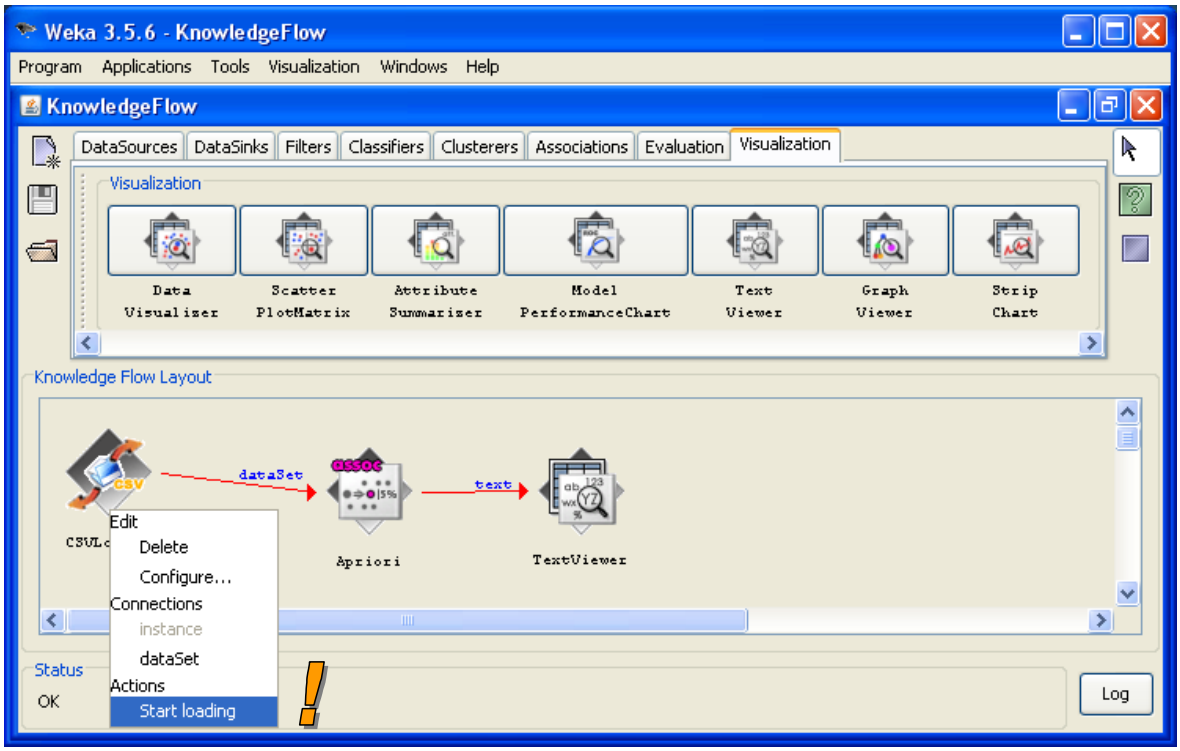


We add the APRIORI component (ASSOCIATIONS tab). We connect the previous component to this one. We click on the CONFIGURE menu in order to define the parameters.

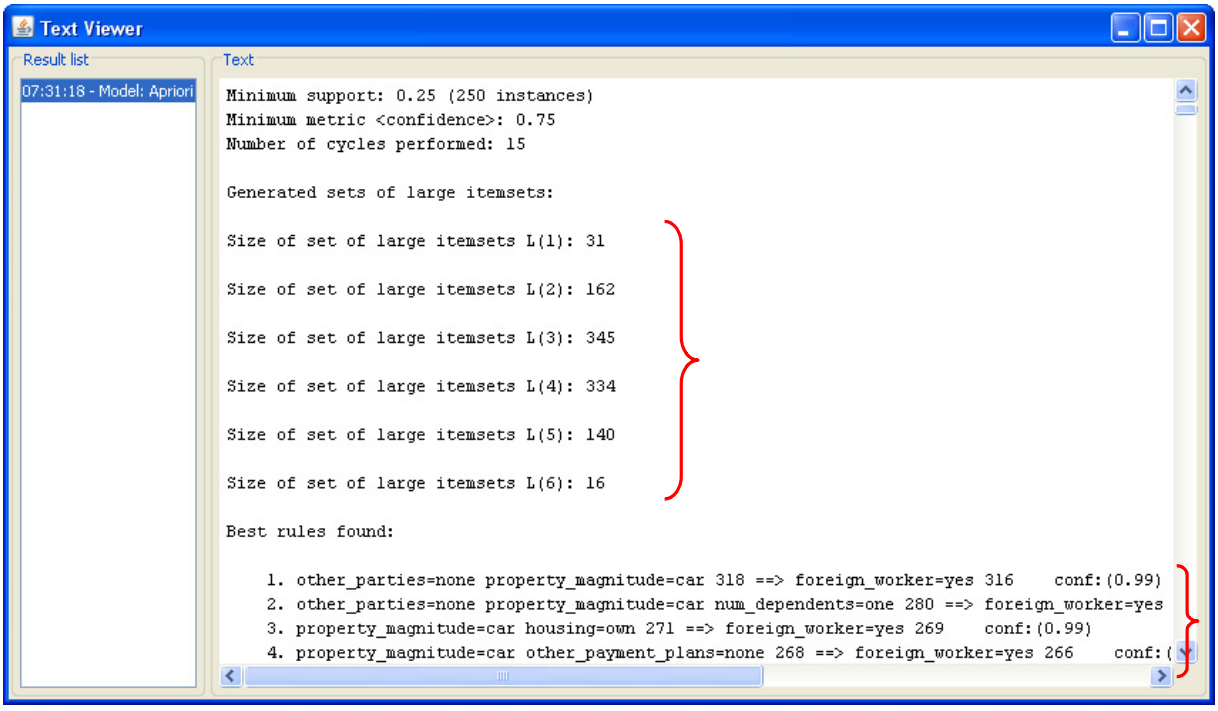


Setting NUMRULES to 1000 removes the restriction on the number of generated rules. We add then the TEXT VIEWER component in order to visualize the rules.

We launch the calculations by clicking on the START LOADING menu of... the CSV LOADER component into the knowledge flow.



To visualize the results, we click on the SHOW RESULTS menu of the TEXTVIEWER component. Like the A PRIORI component of Tanagra, we obtain 2986 rules.



10 Conclusion

All software presented in this tutorial can extract association rules from a data file in an "individuals x variables" format. For some of them, a data preparation is required before to produce a transactional data format. This is not always obvious, especially when the software is not well documented. I admit to having groping a bit. But finally, once the data are correctly generated, the software produced similar results. This is what matters.