1 Topic

Discretization of continuous features as preprocessing for supervised learning process.

The discretization transforms a continuous attribute into a discrete one. To do that, it partitions the range into a set of intervals by defining a set of cut points. Thus we must answer to two questions to lead this data transformation: (1) how to determine the right number of intervals; (2) how to compute the cut points. The resolution is not necessarily in that sequence.

The best discretization is the one performed by an expert domain. Indeed, he takes into account other information than those only provided by the available dataset. Unfortunately, this kind of approach is not always feasible because: often, the domain knowledge is not available or it does not allow to determine the appropriate discretization; the process cannot be automated to handle a large number of attributes. So, we are often forced to found the determination of the best discretization on a numerical process.

Discretization of continuous features as preprocessing for supervised learning process. First, we must define the context in which we perform the transformation. Depending on the circumstances, it is clear that the process and criteria used will not be the same. In this tutorial, we are in the supervised learning framework. We perform the discretization prior to the learning process i.e. we transform the continuous predictive attributes into discrete before to present them to a supervised learning algorithm. In this context, the construction of intervals in which one and only one of the values of the target attribute is the most represented is desirable. The relevance of the computed solution is often evaluated through an impurity based or an entropy based functions.

There are multiple motivations for performing discretization as a preprocessing step[1]: some learning methods do not handle continuous attributes; the data transformed in a set of intervals are more cognitively relevant for a human interpretation; the computation process goes faster with a reduced number of data, particularly when some attributes are suppressed from the representation space of the learning problem if it is impossible to find a relevant cutting; the discretization allows to discover non-linear relations – e.g. the infants and the elderly people are more sensitive to the illness, the relation between age and illness is then not linear – and that is why many authors propose to discretize the data even if the learning method can handle continuous attributes; lastly a discretization can harmonize the nature of the data if there are heterogeneous e.g., in text categorization, the attributes are a mix of numerical values and occurrence terms.

There are several ways to distinguish the discretization algorithms.

[1] F. Muhlenbach, R. Rakotomalala, « Discretization of Continuous Attributes », in Encyclopedia of Data Warehousing and Mining, John Wang (Ed.), pp. 397-402, 2005 (http://hal.archives-ouvertes.fr/hal-00383757/fr/).

Univariate vs. Multivariate. The multivariate approaches discretize the continuous descriptors simultaneously. These are the best solutions because the descriptors are often correlated. However, they are rarely used because they are time consuming, and above all, they are unavailable in the popular tools. On the other hand, the univariate approaches are very popular because they are fast and simple to use. But the cut points are defined individually for each continuous descriptor. We do not take into account the eventual interdependence between them.

Unsupervised vs. Supervised. The unsupervised approaches make use only of the attribute to discretize to define the cut points. Often, but not always[2], the number of intervals is a parameter of the algorithm. Their main advantage is that we can use them in any context, including the supervised learning context. Their main drawback is that they are not especially intended to a supervised learning, they do not use the information provided by the target attribute. The supervised approaches use explicitly the target attribute during the discretization process. They often provide at the same time both the number of intervals and the cut points. They are the most interesting ones in our context.

In this tutorial, we use only the univariate approaches. We compare the behavior of the supervised and the unsupervised algorithms on an artificial dataset. We use several tools for that: Tanagra 1.4.35, Sipina 3.3, R 2.9.2 (package dprep), Weka 3.6.0, Knime 2.1.1, Orange 2.0b and RapidMiner 4.6.0. We highlight the settings and the reading of the results.

About Tanagra, we show how to utilize the transformed variables after the discretization process. Almost all the tools discussed in this tutorial provide the same functionality.

[2] There are various formulas to determine automatically the number of intervals. They are mainly based on the sample size, and sometimes on the range of the values and the standard deviation (http://www.info.univ-angers.fr/~gh/wstat/discr.php) :

Approach	Formula
Brooks-Carruthers	$5\times\log_{10}(n)$
Huntsberger	$1+3.332\times\log_{10}(n)$
Sturges	$\log_2(n+1)$
Scott	$\frac{b-a}{3.5\times\sigma\times n^{(-1/3)}}$
Freedman-Diaconis	$\frac{b-a}{2\times q\times n^{(-1/3)}}$

Where n is the sample size ; b the maximum ; a the minimum ; σ the standard deviation ; q the interquartile range.

2 Dataset

We use an artificial dataset in this tutorial. The target attribute is binary (positive vs. negative). The 4 continuous predictive variables are generated according various conditional distributions.

Such experimentation is usual in the publications[3]. Indeed, knowing the characteristics of the data, we know the proper solutions. First, we give here the distribution of the attributes without taking into account the class membership (Figure 1).

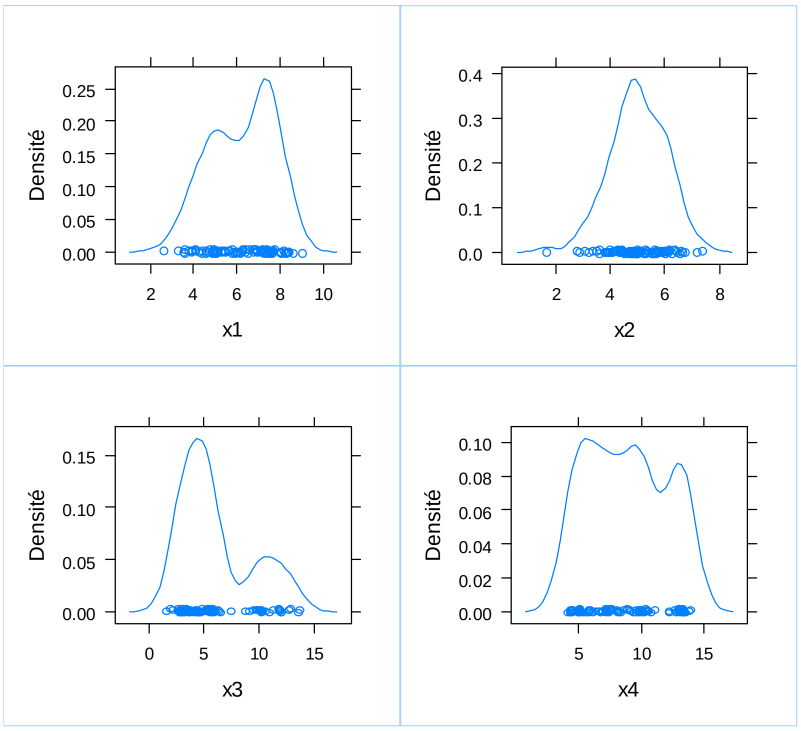


Figure 1 – Unconditional distribution of the predictive attributes

It is difficult to visually detect the cut points using these density functions. Even if it seems there are several modes in some cases, we do not know if these modes are associated to different class values or not.

For the same variables, we draw the density function according the class membership (Figure 2). The conclusions are very different. Excluding the variable X2 for which the conditional densities functions are not discernible, for the others, we note that we can found some cut points in order to define intervals associated to one of the values of the target attribute.

[3] E.g. M. Ismail, V. Ciesielski, "An Empirical Investigation of the Impact of Discretization on Common Data Distributions", in Design and Application of hybrid intelligent systems, 2003; http://portal.acm.org/citation.cfm?id=998117&dl=GUIDE&coll=GUIDE&CFID=88985182&CFTOKEN=88220030

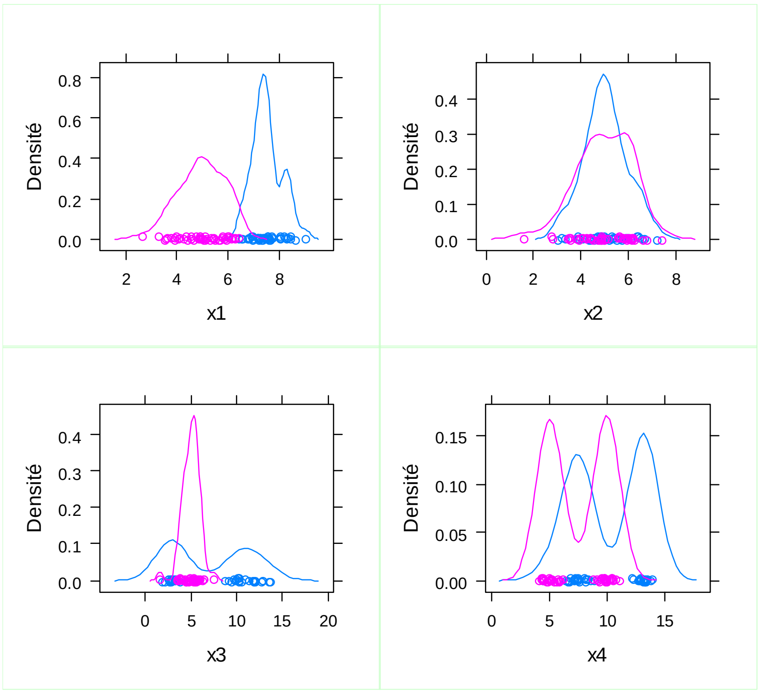


Figure 2 - Conditional distributions of the predictive attributes

Thus, in the supervised learning context, the utilization of the information provided by the target attribute during the discretization process is essential.

In what follows, we will use both the unsupervised and supervised techniques implemented in several data mining tools. For the former, we focus mainly on equal width intervals and equal frequency intervals approaches[4]. Both require the indication of the number of intervals. We will set it to 5. We set a fairly large number of intervals in order to obtain relatively pure intervals according to the target variable, although we have no guarantee on that. About the supervised approaches, we use the default settings. The algorithm detects automatically the appropriate number of intervals and the cut points.

The data file is in the Weka file format (ARFF[5]), it is available on line[6].

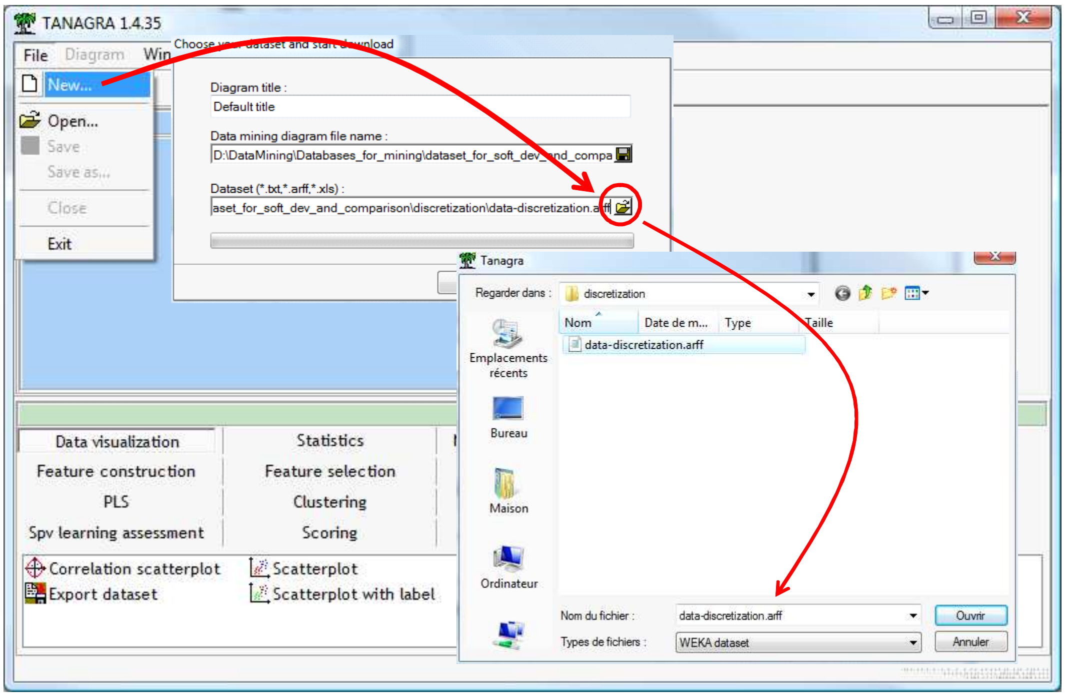
3 Discretization in Tanagra

Importing the data file. We launch TANAGRA. We create a new diagram by clicking on the FILE / NEW menu. We import the data file "**data-discretization.arff**".

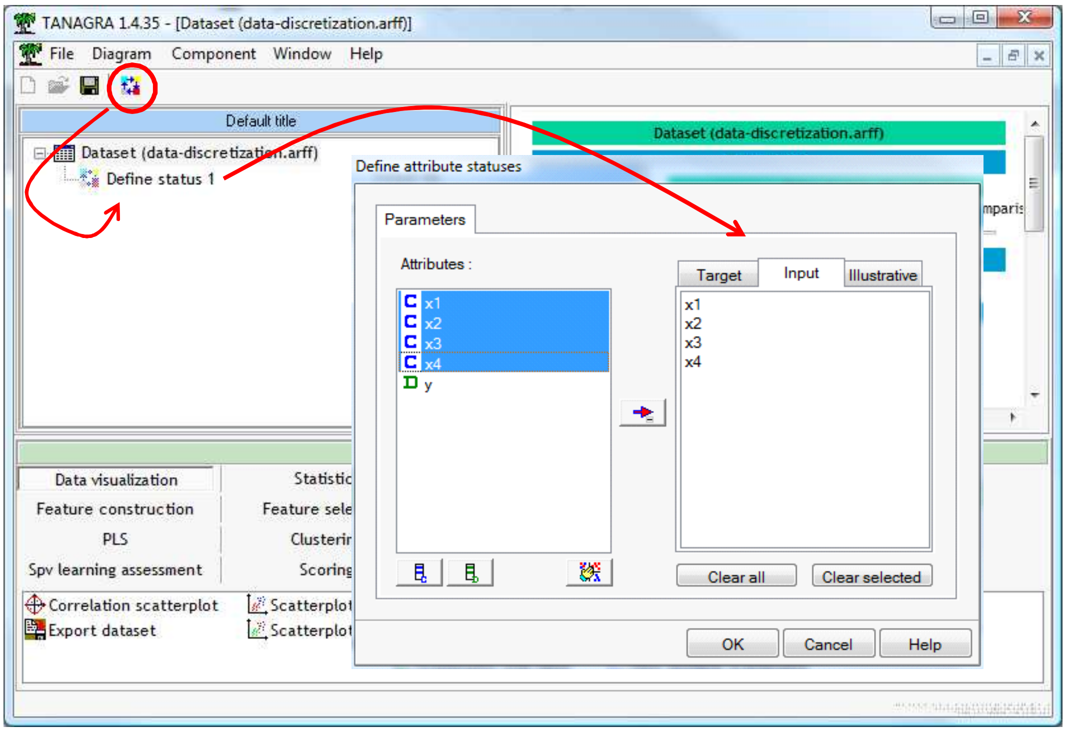
[4] http://robotics.stanford.edu/users/sahami/papers-dir/disc.pdf

[5] http://tutoriels-data-mining.blogspot.com/2008/03/importer-un-fichier-weka-dans-tanagra.html

[6] http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/data-discretization.arff

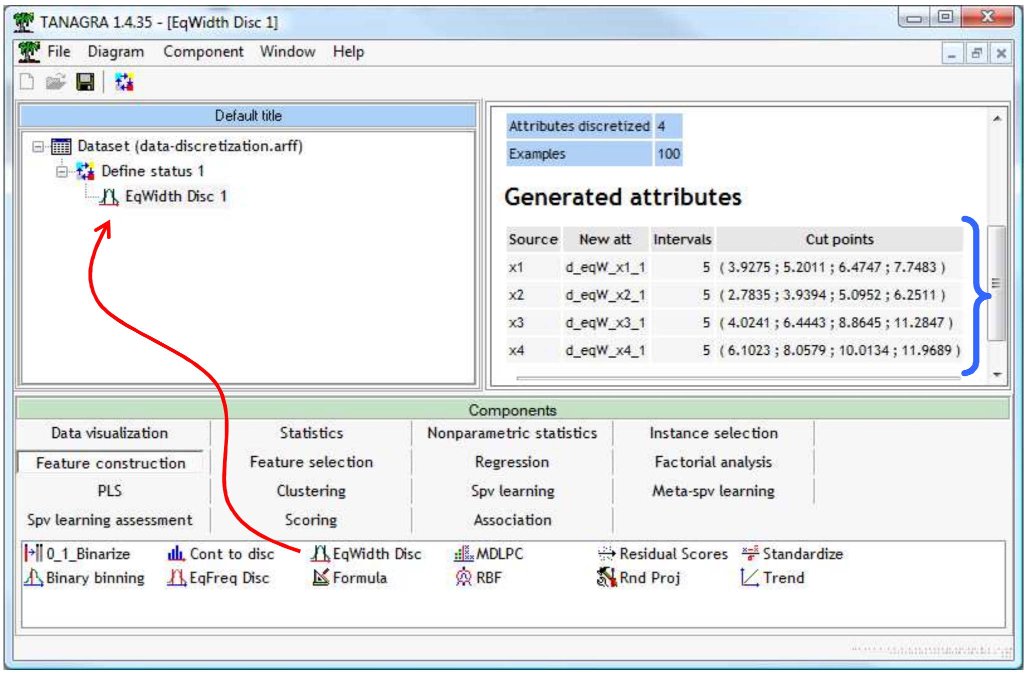


Unsupervised discretization. To discretize with the “equal width intervals” approach, we must specify the variables to analyze with the DEFINE STATUS component.



Then, we insert the **EqWidthDisc** tool (FEATURE CONSTRUCTION)[7]. With the default settings, the variables are discretized into 5 intervals. We can modify this parameter. We click on the VIEW menu to obtain the computed cut points.

[7] The other available tool is **EqFreqDisc**, equal frequency intervals discretization.



Attributes discretized	4
Examples	100

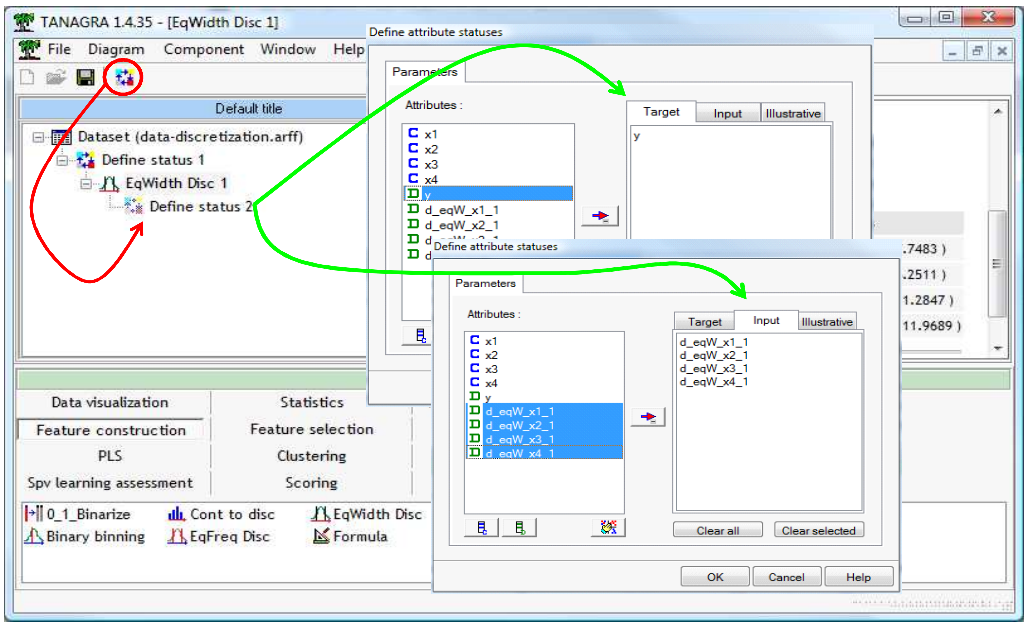
Generated attributes

Source	New att	Intervals	Cut points
x1	d_eqW_x1_1	5	(3.9275 ; 5.2011 ; 6.4747 ; 7.7483)
x2	d_eqW_x2_1	5	(2.7835 ; 3.9394 ; 5.0952 ; 6.2511)
x3	d_eqW_x3_1	5	(4.0241 ; 6.4443 ; 8.8645 ; 11.2847)
x4	d_eqW_x4_1	5	(6.1023 ; 8.0579 ; 10.0134 ; 11.9689)

Of course, if we consider the data characteristics (Figure 2), the discretization into 5 intervals is not appropriated in the majority of cases. We note however than some cut points are, a little by chance, suitable in some situations. For instance, the cut point 6.47 is not too bad for the variable X1. But, on the other hand, the others cut points are irrelevant. This is the main drawback of this approach. If we set a large number of intervals, some cut points are maybe interesting, but we increase also the risk of data fragmentation, increasing at the same time the variance of the induced classifier.

Conversely, it is clear that there is no discretization possible for X2. The procedure provides us four cut points that are totally irrelevant.

Supervised learning process from the transformed variables. To use the transformed variables, we insert the DEFINE STATUS tool, we set them as INPUT. The target attribute is Y.

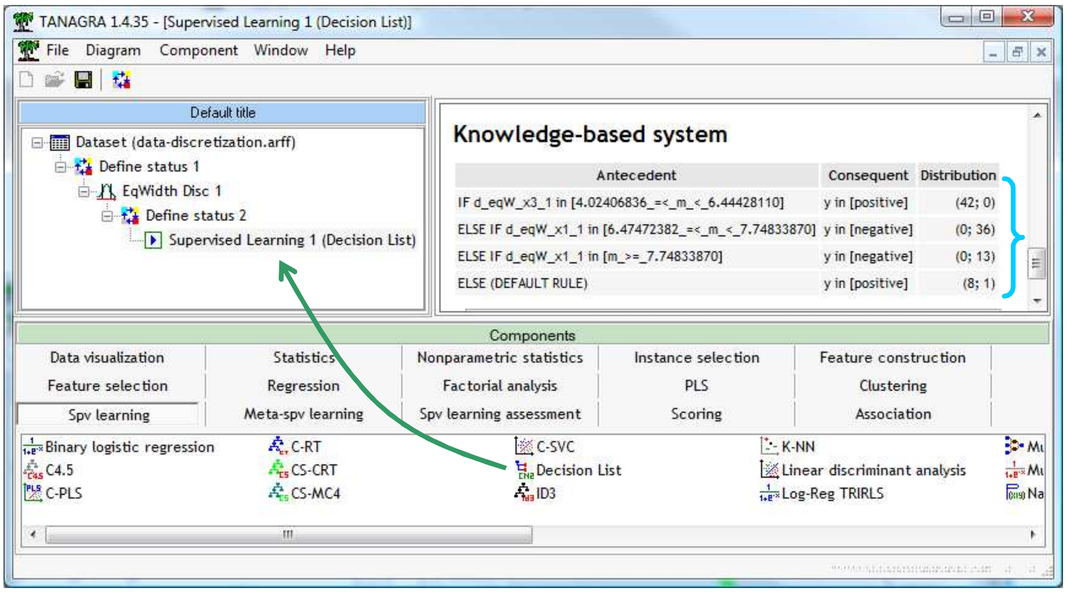


Define attribute statuses — Attributes: x1, x2, x3, x4, y, d_eqW_x1_1, d_eqW_x2_1, d_eqW_x3_1, d_eqW_x4_1

Target: y

Input: d_eqW_x1_1, d_eqW_x2_1, d_eqW_x3_1, d_eqW_x4_1

Then we add the decision list algorithm which induces ordered rules (SPV LEARNING – see http://data-mining-tutorials.blogspot.com/2010/02/supervised-rule-induction-software.html). It cannot handle directly continuous descriptors. The discretization preprocessing step is needed here.



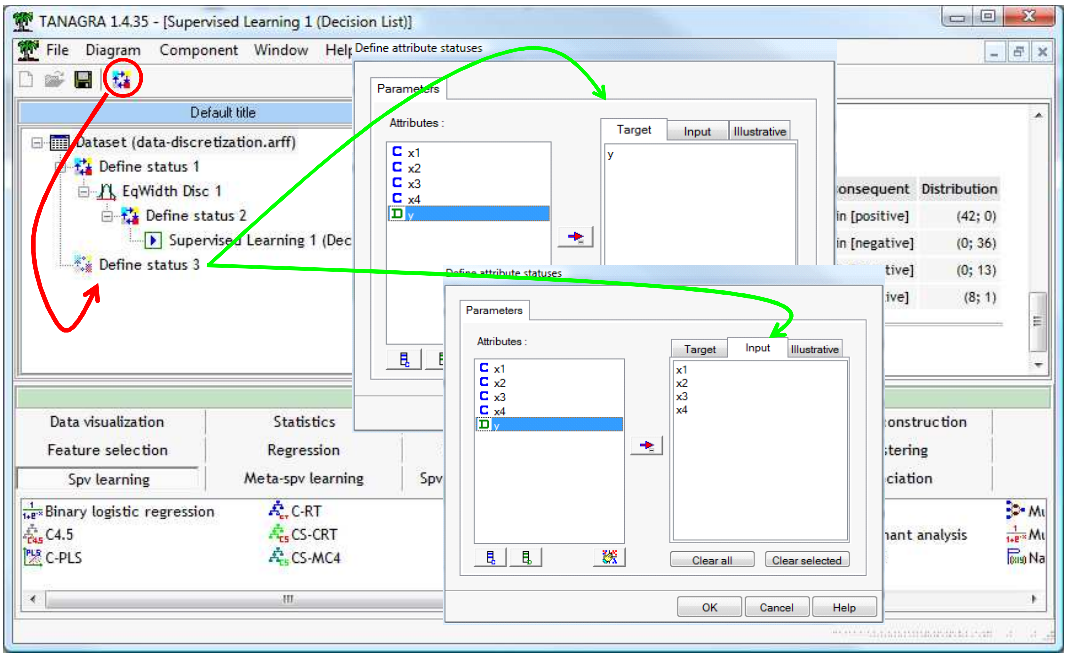
TANAGRA 1.4.35 - [Supervised Learning 1 (Decision List)]
File Diagram Component Window Help
Default title
Dataset (data-discretization.arff)
Define status 1
EqWidth Disc 1
Define status 2
Supervised Learning 1 (Decision List)
Components
Data visualization Statistics Nonparametric statistics Instance selection Feature construction
Feature selection Regression Factorial analysis PLS Clustering
Spv learning Meta-spv learning Spv learning assessment Scoring Association
Binary logistic regression C-RT C-SVC K-NN
C4.5 CS-CRT Decision List Linear discriminant analysis
C-PLS CS-MC4 ID3 Log-Reg TRIRLS

Knowledge-based system

Antecedent	Consequent	Distribution
IF d_eqW_x3_1 in [4.02406836_=<_m_<_6.44428110]	y in [positive]	(42; 0)
ELSE IF d_eqW_x1_1 in [6.47472382_=<_m_<_7.74833870]	y in [negative]	(0; 36)
ELSE IF d_eqW_x1_1 in [m_>=_7.74833870]	y in [negative]	(0; 13)
ELSE (DEFAULT RULE)	y in [positive]	(8; 1)

The classifier is rather good according to the resubstitution error rate. It is not surprising because we see before that in some situations, the discretization can find by chance some good cut points if we set a large number of intervals. But, when we consider carefully the ruleset, we note that we obtain unnecessary rules in the classifier. For instance, we observe that we need two rules (n°2 and n°3) to assign an instance to the negative class value, whereas one rule would have been enough e.g. « ELSE IF X1 ≥ 6.47 THEN Y = NEGATIVE ». The cut point “X1 = 7.748” is irrelevant.

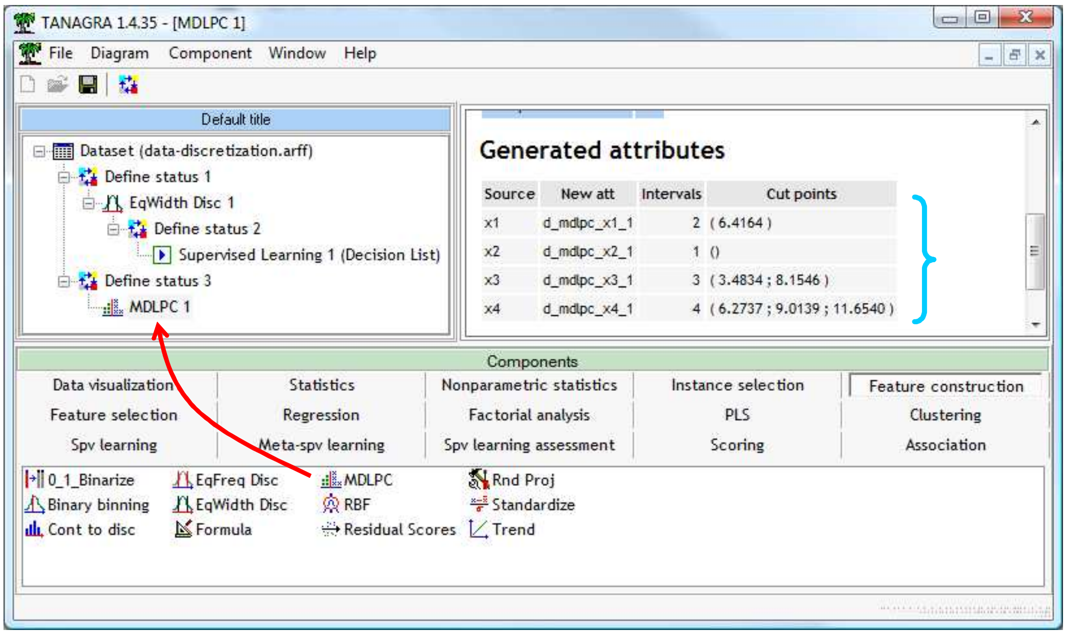
Supervised discretization. To perform a supervised discretization, we set X1…X4 as INPUT, and Y as TARGET with the DEFINE STATUS tool.



TANAGRA 1.4.35 - [Supervised Learning 1 (Decision List)]
File Diagram Component Window Help
Default title
Dataset (data-discretization.arff)
Define status 1
EqWidth Disc 1
Define status 2
Supervised Learning 1 (Dec
Define status 3
Define attribute statuses
Parameters
Attributes :
x1
x2
x3
x4
y
Target Input Illustrative
y
x1
x2
x3
x4
Clear all Clear selected
OK Cancel Help

We specify the target attribute here because the descriptors will be discretized with respect to Y.

We add the MDLPC component (FEATURE CONSTRUCTION) which implements a very popular approach (*U. Fayyad et K. Irani, « Multi-interval discretization of continuous-valued attributes for classification learning », in Proc. of IJCAI, pp.1022-1027, 1993*). We click on the VIEW menu.

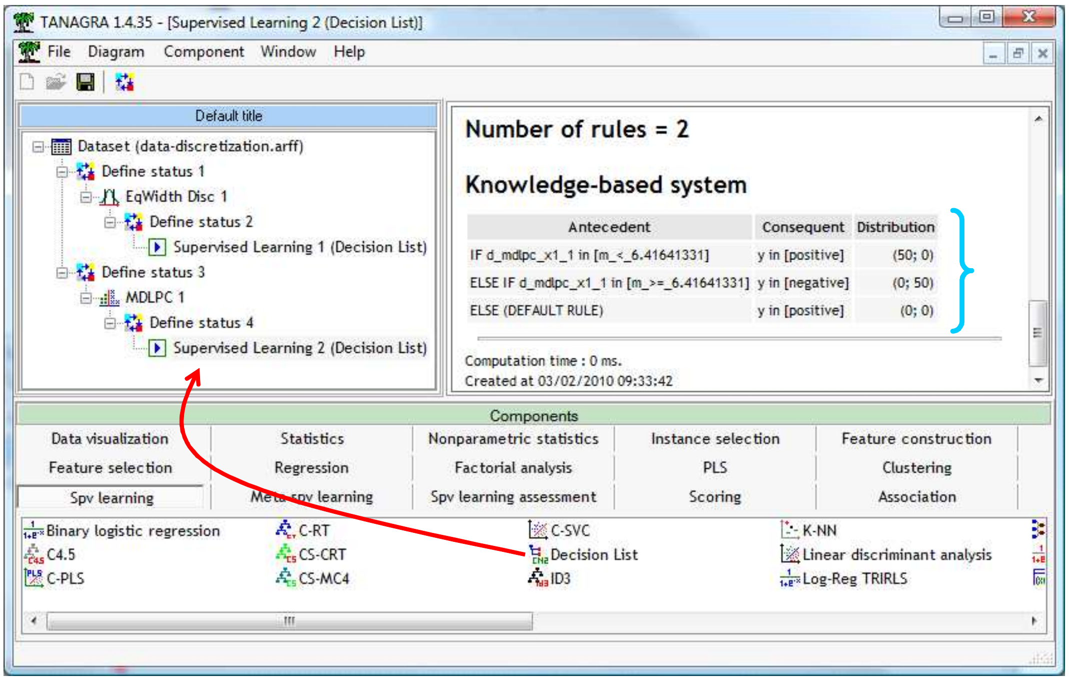

TANAGRA 1.4.35 - [MDLPC 1]
File Diagram Component Window Help
Default title
Dataset (data-discretization.arff)
Define status 1
EqWidth Disc 1
Define status 2
Supervised Learning 1 (Decision List)
Define status 3
MDLPC 1
Components
Data visualization Statistics Nonparametric statistics Instance selection Feature construction
Feature selection Regression Factorial analysis PLS Clustering
Spv learning Meta-spv learning Spv learning assessment Scoring Association
0_1_Binarize EqFreq Disc MDLPC Rnd Proj
Binary binning EqWidth Disc RBF Standardize
Cont to disc Formula Residual Scores Trend

Generated attributes

Source	New att	Intervals	Cut points
x1	d_mdlpc_x1_1	2	(6.4164)
x2	d_mdlpc_x2_1	1	()
x3	d_mdlpc_x3_1	3	(3.4834 ; 8.1546)
x4	d_mdlpc_x4_1	4	(6.2737 ; 9.0139 ; 11.6540)

The approach detects the good number of intervals according the variables. Sometimes, the discretization is not performed if the processing is not relevant. The method is singularly efficient on our artificial dataset. Among others, it detects rightly that the discretization is not feasible for X2.

Supervised learning process from the transformed variables. Here again, we perform the induction of decision list on the transformed descriptors.


TANAGRA 1.4.35 - [Supervised Learning 2 (Decision List)]
File Diagram Component Window Help
Default title
Dataset (data-discretization.arff)
Define status 1
EqWidth Disc 1
Define status 2
Supervised Learning 1 (Decision List)
Define status 3
MDLPC 1
Define status 4
Supervised Learning 2 (Decision List)
Components
Data visualization Statistics Nonparametric statistics Instance selection Feature construction
Feature selection Regression Factorial analysis PLS Clustering
Spv learning Meta-spv learning Spv learning assessment Scoring Association
Binary logistic regression C-RT C-SVC K-NN
C4.5 CS-CRT Decision List Linear discriminant analysis
C-PLS CS-MC4 ID3 Log-Reg TRIRLS

Number of rules = 2

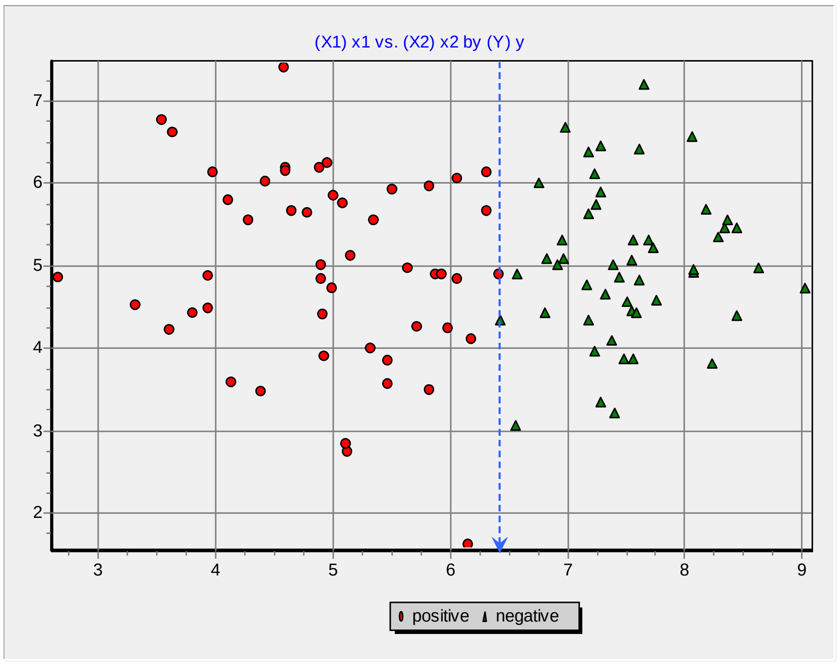
Knowledge-based system

Antecedent	Consequent	Distribution
IF d_mdlpc_x1_1 in [m_<_6.41641331]	y in [positive]	(50; 0)
ELSE IF d_mdlpc_x1_1 in [m_>=_6.41641331]	y in [negative]	(0; 50)
ELSE (DEFAULT RULE)	y in [positive]	(0; 0)

Computation time : 0 ms.
Created at 03/02/2010 09:33:42

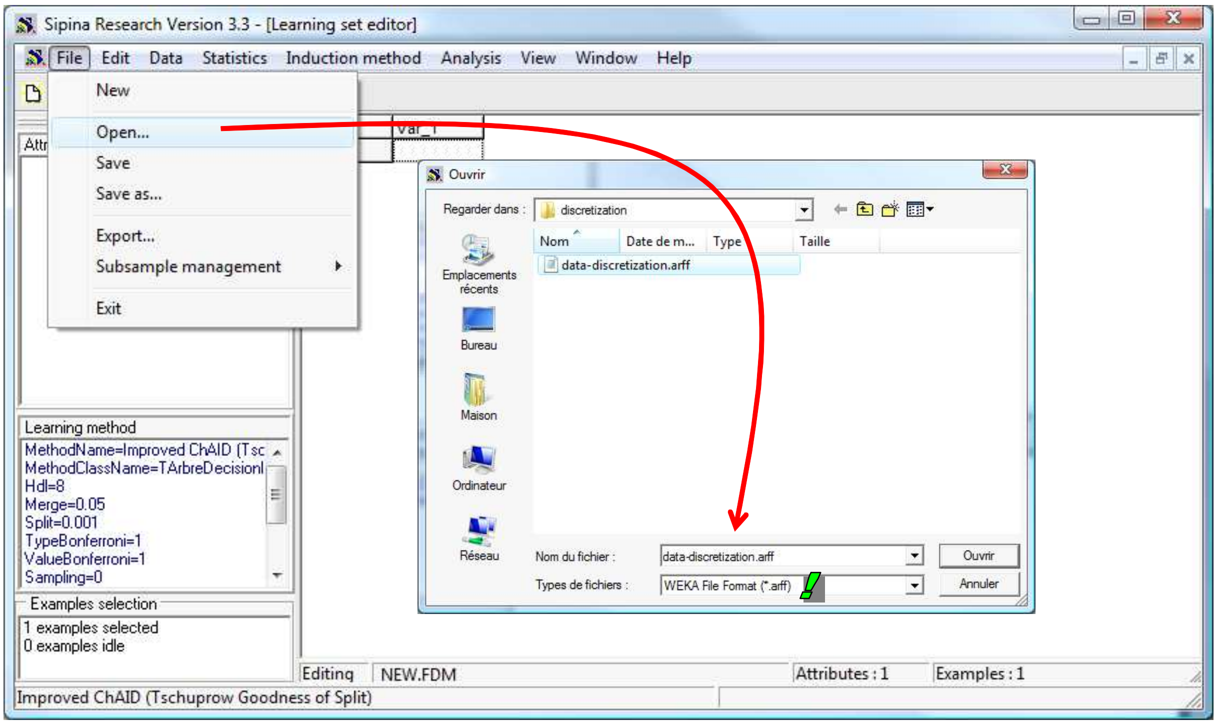
The obtained classifier is as good as (resubstitution error rate) the preceding one, but it is simpler with fewer rules. In our dataset, the descriptor X1 is enough to build an efficient classifier.

Geometrical interpretation. If we plot the points in a scatter diagram defined by (X1, X2), we observe indeed that X1 is enough to separate perfectly the individuals from different groups.



4 Discretization in SIPINA

Importing the data file. We launch SIPINA (http://eric.univ-lyon2.fr/~ricco/sipina.html). We import the data file by clicking on the FILE / OPEN menu. We select **data-discretization.arff**.

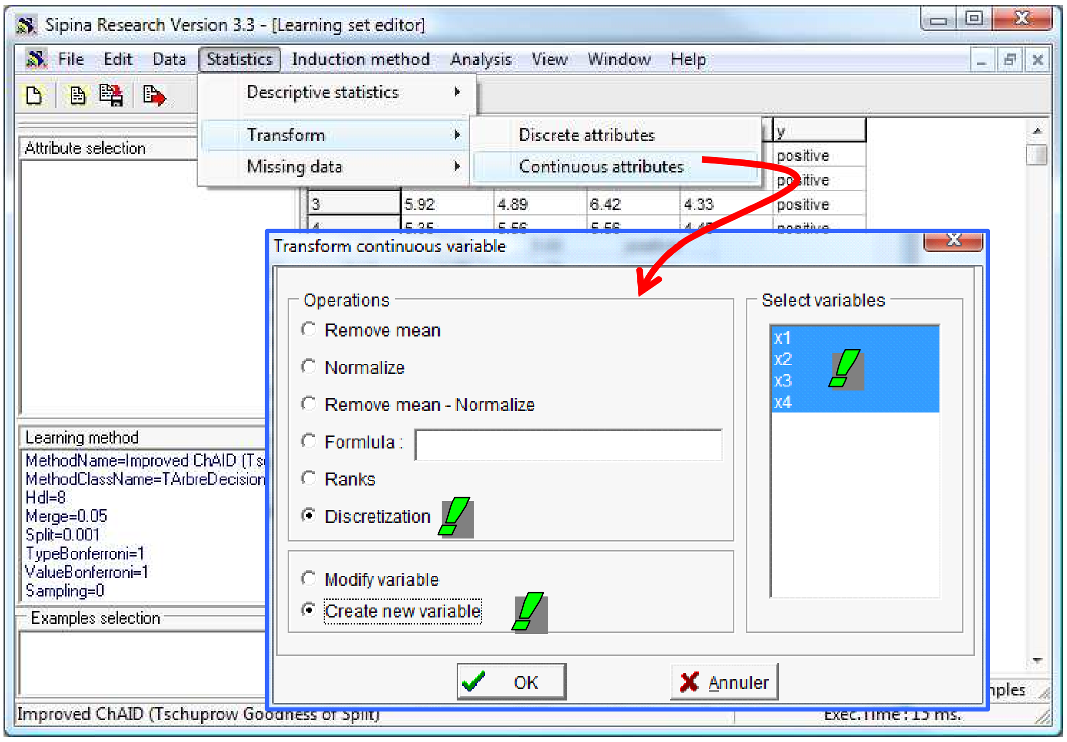


Learning method
MethodName=Improved ChAID (Tsc
MethodClassName=TArbreDecisionI
Hdl=8
Merge=0.05
Split=0.001
TypeBonferroni=1
ValueBonferroni=1
Sampling=0

Examples selection
1 examples selected
0 examples idle

Improved ChAID (Tschuprow Goodness of Split)

Discretization of the variables. The discretization tools are available from the STATISTICS / TRANSFORM / CONTINUOUS ATTRIBUTES menu. Into the dialog box, we select the continuous attributes. We ask a discretization. The new variables are created and displayed into the DataGrid.


Sipina Research Version 3.3 - [Learning set editor]
File Edit Data Statistics Induction method Analysis View Window Help
Descriptive statistics
Transform
Missing data
Discrete attributes
Continuous attributes
Attribute selection
Learning method
Examples selection
Transform continuous variable
OK
Annuler

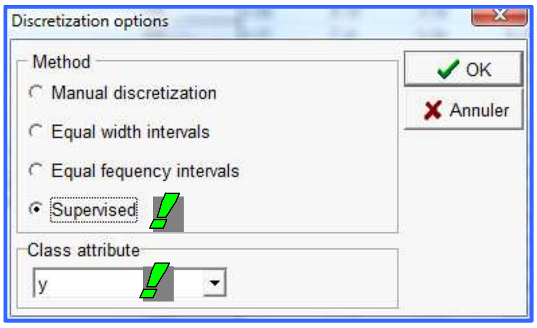
Operations: Remove mean / Normalize / Remove mean - Normalize / Formlula : / Ranks / [✓ Discretization]

Modify variable / [✓ Create new variable]

Select variables: x1, x2, x3, x4

MethodName=Improved ChAID (Ts
MethodClassName=TArbreDecision
Hdl=8
Merge=0.05
Split=0.001
TypeBonferroni=1
ValueBonferroni=1
Sampling=0

Improved ChAID (Tschuprow Goodness of Split)

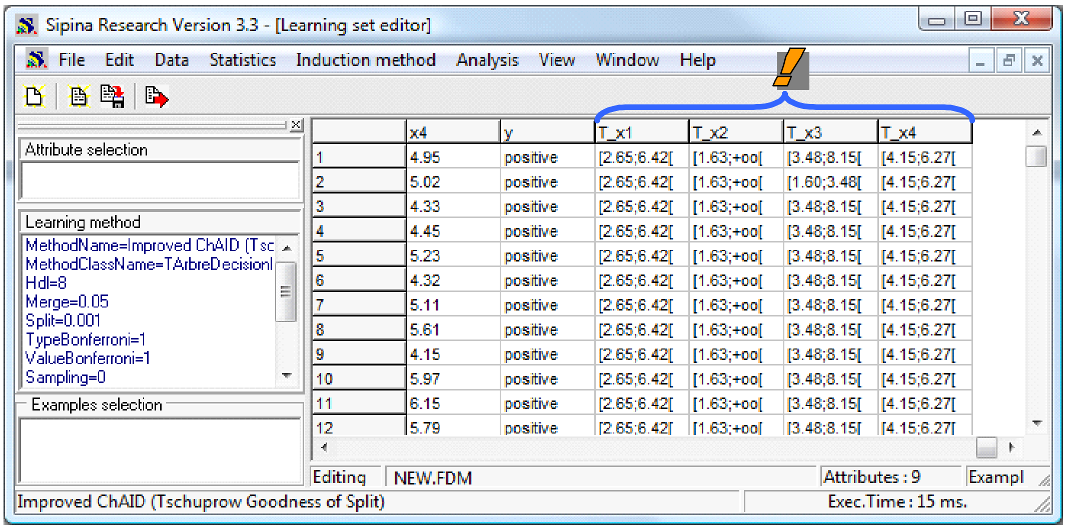


Method: Manual discretization / Equal width intervals / Equal fequency intervals / [✓ Supervised]

Class attribute: y

We click on the OK button. A second dialog box appears. We can choose the method to use. If we want a supervised approach (MDLPC), we select the SUPERVISED option and we specify the target attribute Y into the drop down list.

We confirm our settings by clicking on the OK button.


Sipina Research Version 3.3 - [Learning set editor]
File Edit Data Statistics Induction method Analysis View Window Help
Attribute selection
Learning method
Examples selection

MethodName=Improved ChAID (Tsc
MethodClassName=TArbreDecisionI
Hdl=8
Merge=0.05
Split=0.001
TypeBonferroni=1
ValueBonferroni=1
Sampling=0

	x4	y	T_x1	T_x2	T_x3	T_x4
1	4.95	positive	[2.65;6.42[	[1.63;+oo[	[3.48;8.15[	[4.15;6.27[
2	5.02	positive	[2.65;6.42[	[1.63;+oo[	[1.60;3.48[	[4.15;6.27[
3	4.33	positive	[2.65;6.42[	[1.63;+oo[	[3.48;8.15[	[4.15;6.27[
4	4.45	positive	[2.65;6.42[	[1.63;+oo[	[3.48;8.15[	[4.15;6.27[
5	5.23	positive	[2.65;6.42[	[1.63;+oo[	[3.48;8.15[	[4.15;6.27[
6	4.32	positive	[2.65;6.42[	[1.63;+oo[	[3.48;8.15[	[4.15;6.27[
7	5.11	positive	[2.65;6.42[	[1.63;+oo[	[3.48;8.15[	[4.15;6.27[
8	5.61	positive	[2.65;6.42[	[1.63;+oo[	[3.48;8.15[	[4.15;6.27[
9	4.15	positive	[2.65;6.42[	[1.63;+oo[	[3.48;8.15[	[4.15;6.27[
10	5.97	positive	[2.65;6.42[	[1.63;+oo[	[3.48;8.15[	[4.15;6.27[
11	6.15	positive	[2.65;6.42[	[1.63;+oo[	[3.48;8.15[	[4.15;6.27[
12	5.79	positive	[2.65;6.42[	[1.63;+oo[	[3.48;8.15[	[4.15;6.27[

Editing NEW.FDM Attributes : 9 Exampl

Improved ChAID (Tschuprow Goodness of Split) Exec.Time : 15 ms.

The transformed attributes are now displayed into the data grid. The results (number of intervals and cut points) are the same as those of Tanagra.

Note: We can manually discretize the variables in Sipina (MANUAL DISCRETIZATION). We enumerate the cut points by separating them with ";" in the dialog settings.

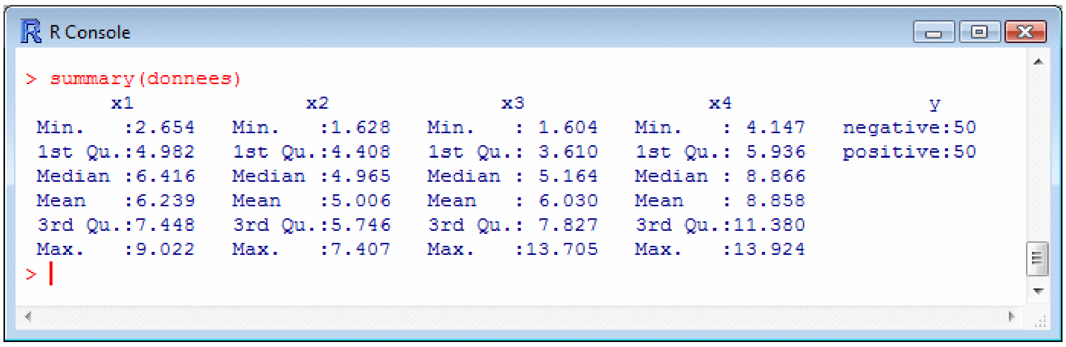
5 Discretization in R (« dprep » package)

The **dprep** package for R (http://www.r-project.org/) incorporates various approaches for the discretization process.

We set the following commands for the importation of the data file.

```
#load the dataset
library(RWeka)
donnees <- read.arff(file="data-discretization.arff")
summary(donnees)
```

Some descriptive statistics indicators allows to check the data integrity.

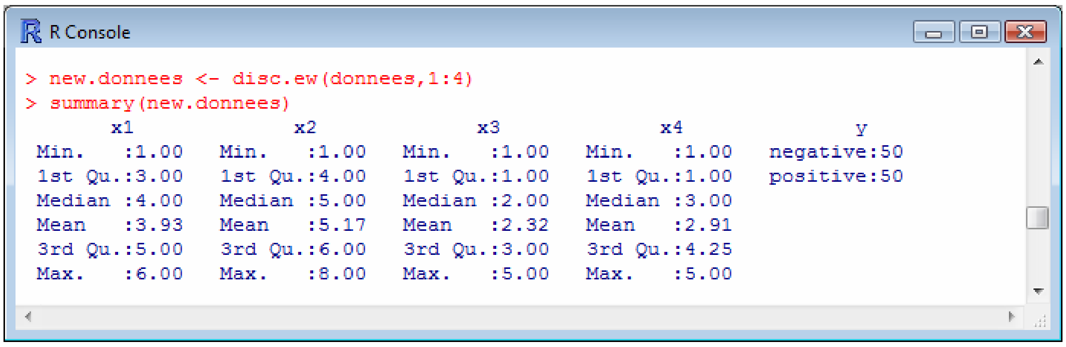


```
> summary(donnees)
       x1              x2              x3               x4                 y
 Min.   :2.654   Min.   :1.628   Min.   : 1.604   Min.   : 4.147   negative:50
 1st Qu.:4.982   1st Qu.:4.408   1st Qu.: 3.610   1st Qu.: 5.936   positive:50
 Median :6.416   Median :4.965   Median : 5.164   Median : 8.866
 Mean   :6.239   Mean   :5.006   Mean   : 6.030   Mean   : 8.858
 3rd Qu.:7.448   3rd Qu.:5.746   3rd Qu.: 7.827   3rd Qu.:11.380
 Max.   :9.022   Max.   :7.407   Max.   :13.705   Max.   :13.924
> 
```

Equal width discretization. For the equal width discretization process, we set the following commands.

```
#discretization
library(dprep)
#equal-width discretization
new.donnees <- disc.ew(donnees,1:4)
summary(new.donnees)
```

The number of intervals is automatically determined with the Scott's formula (see above).



```
> new.donnees <- disc.ew(donnees,1:4)
> summary(new.donnees)
       x1             x2             x3             x4              y
 Min.   :1.00   Min.   :1.00   Min.   :1.00   Min.   :1.00   negative:50
 1st Qu.:3.00   1st Qu.:4.00   1st Qu.:1.00   1st Qu.:1.00   positive:50
 Median :4.00   Median :5.00   Median :2.00   Median :3.00
 Mean   :3.93   Mean   :5.17   Mean   :2.32   Mean   :2.91
 3rd Qu.:5.00   3rd Qu.:6.00   3rd Qu.:3.00   3rd Qu.:4.25
 Max.   :6.00   Max.   :8.00   Max.   :5.00   Max.   :5.00
```

E.g. For X1, we have $n = 100$; $b = 9.022$; $a = 2.654$; $\sigma = 1.484$. We obtain

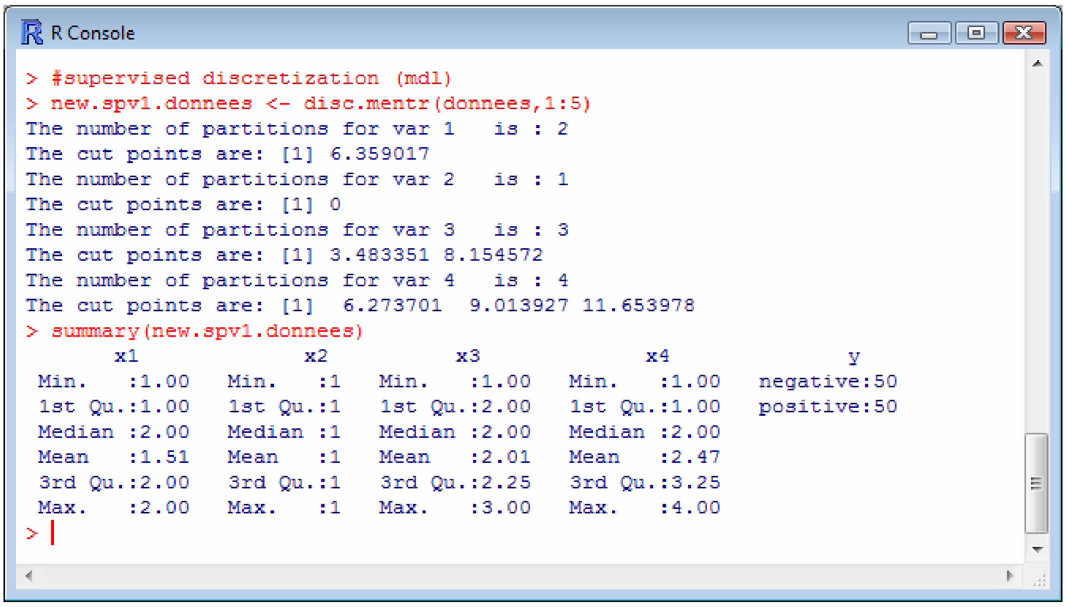
$$\frac{b-a}{3.5\times\sigma\times n^{(-1/3)}}=5.69$$

The value is rounded to 6.

Supervised discretization with MDLPC. Even if the help file does not quote explicitly the Fayyad and Irani's paper, it seems that the supervised algorithm relies on a very similar approach.

```
#supervised discretization (mdl)
new.spv1.donnees <- disc.mentr(donnees,1:5)
summary(new.spv1.donnees)
```

A new *data.frame* is generated. The function displays the number of intervals and the cut points.



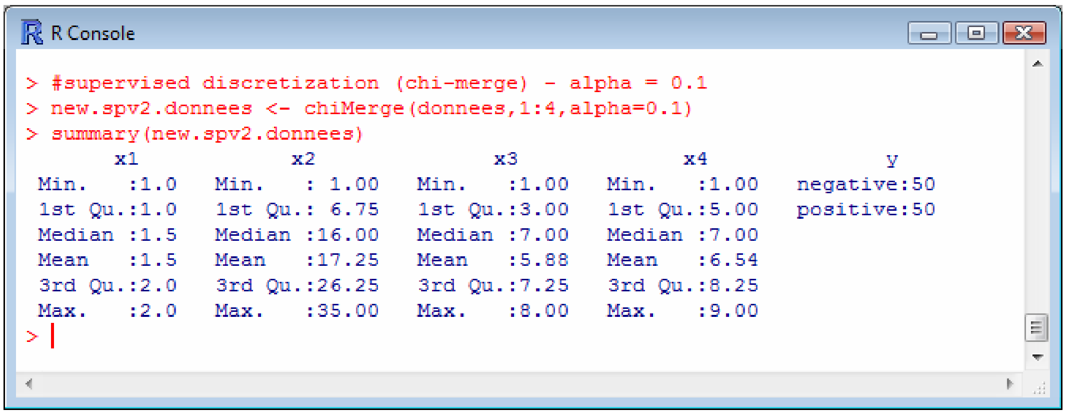
```
> #supervised discretization (mdl)
> new.spv1.donnees <- disc.mentr(donnees,1:5)
The number of partitions for var 1   is : 2
The cut points are: [1] 6.359017
The number of partitions for var 2   is : 1
The cut points are: [1] 0
The number of partitions for var 3   is : 3
The cut points are: [1] 3.483351 8.154572
The number of partitions for var 4   is : 4
The cut points are: [1]  6.273701  9.013927 11.653978
> summary(new.spv1.donnees)
       x1              x2            x3              x4                y     
 Min.   :1.00   Min.   :1   Min.   :1.00   Min.   :1.00   negative:50  
 1st Qu.:1.00   1st Qu.:1   1st Qu.:2.00   1st Qu.:1.00   positive:50  
 Median :2.00   Median :1   Median :2.00   Median :2.00                
 Mean   :1.51   Mean   :1   Mean   :2.01   Mean   :2.47                
 3rd Qu.:2.00   3rd Qu.:1   3rd Qu.:2.25   3rd Qu.:3.25                
 Max.   :2.00   Max.   :1   Max.   :3.00   Max.   :4.00                
> 
```

Supervised discretization with CHI-MERGE. The “dprep” package incorporates another supervised discretization technique. This is the CHI-MERGE method (Kerber, 1992). In comparison with MDLPC, it is a bottom-up approach. It starts with the purest discretization (in the worst case, we have an individual into each interval). It tries to merge the adjacent intervals if the distribution of the target attributes is not significantly different. The process is stopped when none merging is possible. Unfortunately, this approach relies heavily on the parameter alpha (the significance level of the test). To define the good value according the domain or the data characteristics is difficult.

We set the following commands.

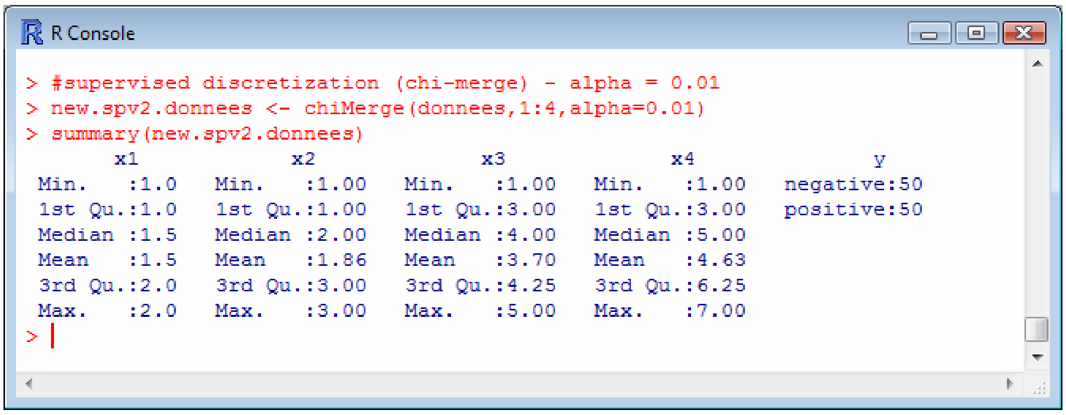
```
#supervised discretization (chi-merge)
new.spv2.donnees <- chiMerge(donnees,1:4,alpha=0.1)
summary(new.spv2.donnees)
```

The variable X2, for which it is not possible to find an efficient solution, is discretized into 35 intervals (!); the variable X4 is discretized into 9 intervals.



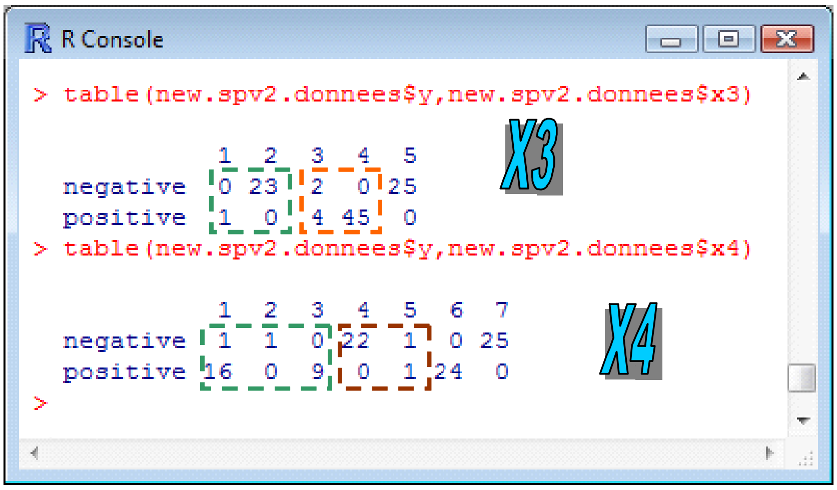
```
> #supervised discretization (chi-merge) - alpha = 0.1
> new.spv2.donnees <- chiMerge(donnees,1:4,alpha=0.1)
> summary(new.spv2.donnees)
       x1              x2               x3              x4                y
 Min.   :1.0    Min.   : 1.00    Min.   :1.00    Min.   :1.00    negative:50
 1st Qu.:1.0    1st Qu.: 6.75    1st Qu.:3.00    1st Qu.:5.00    positive:50
 Median :1.5    Median :16.00    Median :7.00    Median :7.00
 Mean   :1.5    Mean   :17.25    Mean   :5.88    Mean   :6.54
 3rd Qu.:2.0    3rd Qu.:26.25    3rd Qu.:7.25    3rd Qu.:8.25
 Max.   :2.0    Max.   :35.00    Max.   :8.00    Max.   :9.00
>
```

We try to decrease the significance level to favor the merging. The solution for X2 is improved. But the number of intervals remains too high for several variables.



```
> #supervised discretization (chi-merge) - alpha = 0.01
> new.spv2.donnees <- chiMerge(donnees,1:4,alpha=0.01)
> summary(new.spv2.donnees)
       x1              x2              x3              x4                y
 Min.   :1.0    Min.   :1.00    Min.   :1.00    Min.   :1.00    negative:50
 1st Qu.:1.0    1st Qu.:1.00    1st Qu.:3.00    1st Qu.:3.00    positive:50
 Median :1.5    Median :2.00    Median :4.00    Median :5.00
 Mean   :1.5    Mean   :1.86    Mean   :3.70    Mean   :4.63
 3rd Qu.:2.0    3rd Qu.:3.00    3rd Qu.:4.25    3rd Qu.:6.25
 Max.   :2.0    Max.   :3.00    Max.   :5.00    Max.   :7.00
>
```

The contingency tables between the target variable and recoded descriptors (X3, X4) show that merging operations are still necessary.



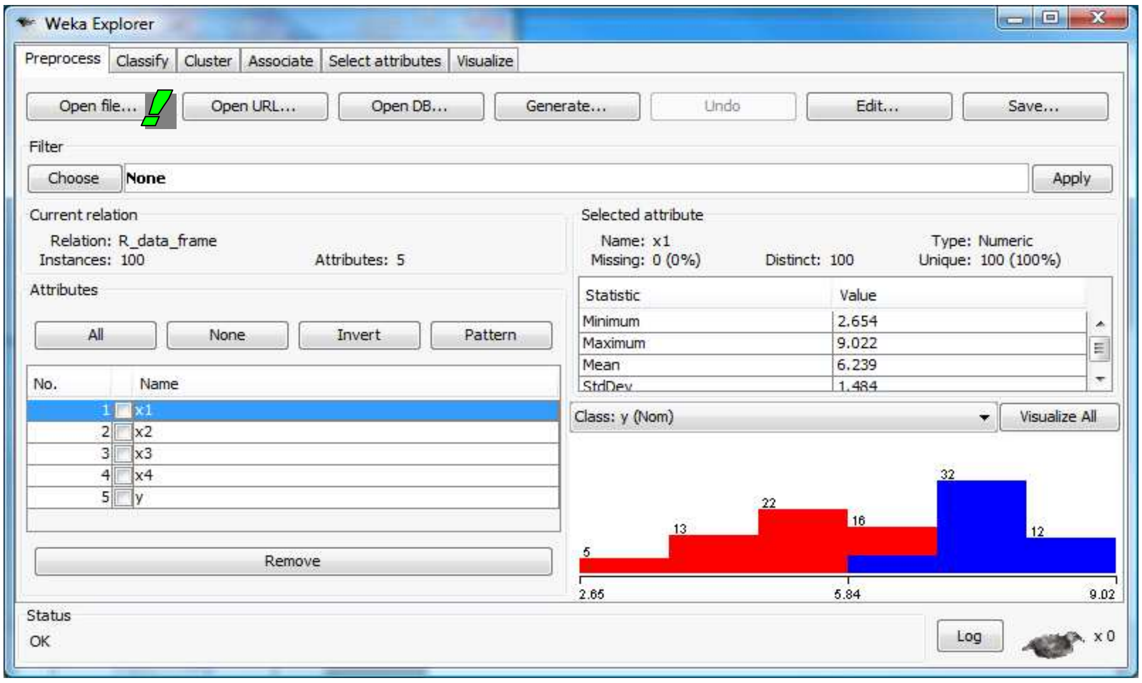
```
> table(new.spv2.donnees$y,new.spv2.donnees$x3)

            1  2  3  4  5
  negative  0 23  2  0 25
  positive  1  0  4 45  0
> table(new.spv2.donnees$y,new.spv2.donnees$x4)

            1  2  3  4  5  6  7
  negative  1  1  0 22  1  0 25
  positive 16  0  9  0  1 24  0
>
```

6 Discretization in Weka

We use Weka (http://www.cs.waikato.ac.nz/ml/weka/) in EXPLORER mode. Into the PREPROCESS tab, we load the data file by clicking on the OPEN FILE button.



Relation: R_data_frame
Instances: 100 Attributes: 5

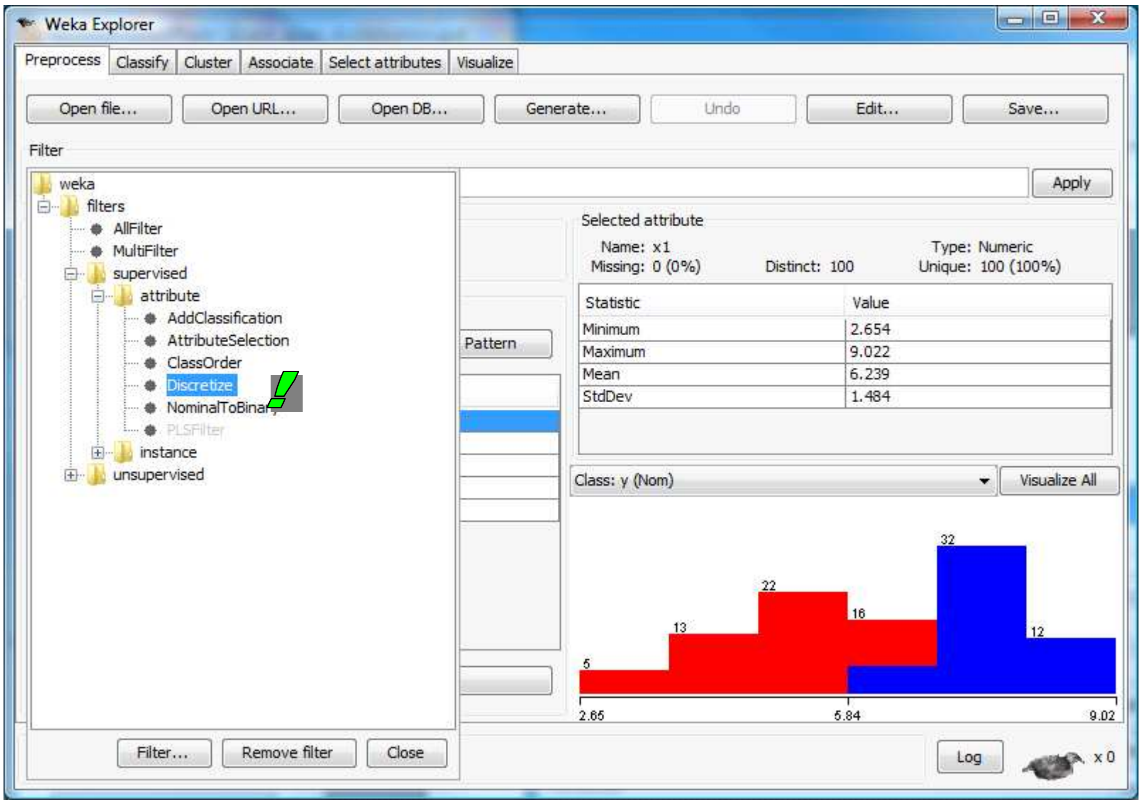
Name: x1 Type: Numeric
Missing: 0 (0%) Distinct: 100 Unique: 100 (100%)

No.	Name
1	x1
2	x2
3	x3
4	x4
5	y

Statistic	Value
Minimum	2.654
Maximum	9.022
Mean	6.239
StdDev	1.484

Weka displays automatically the distributions of the variables conditionally to the target attribute (the last column of the data file).

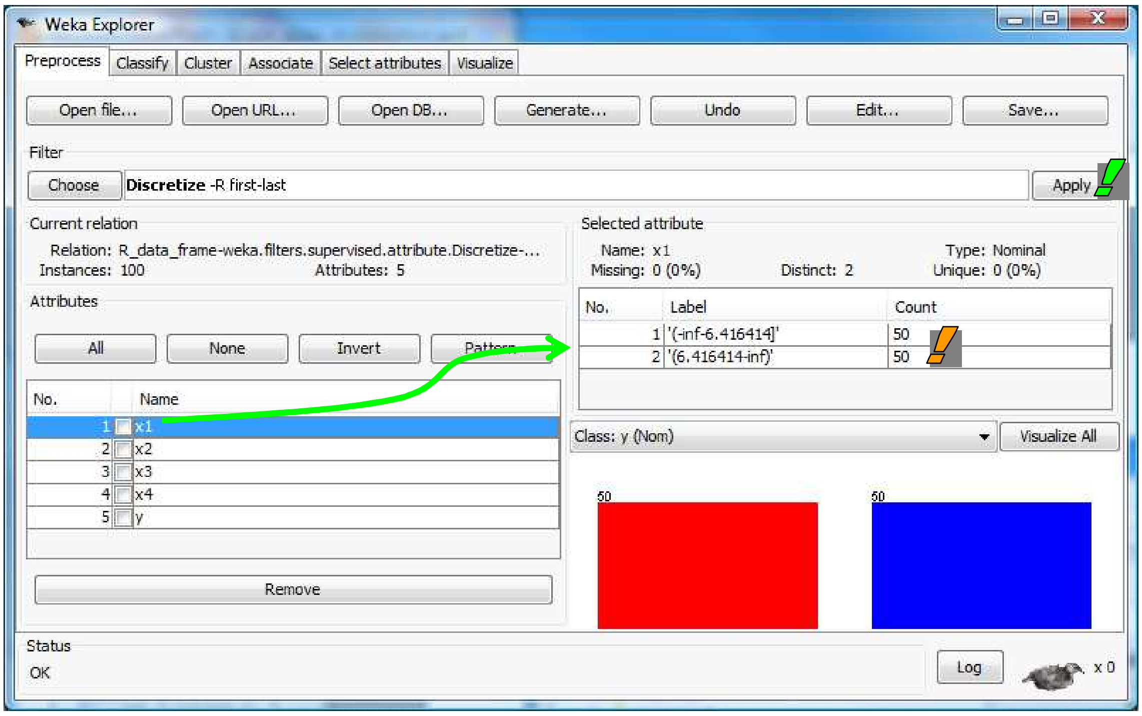
To perform the discretization process, we click on the CHOOSE button (FILTER), we select the DISCRETIZE tool in the SUPERVISED branch.



Name: x1 Type: Numeric
Missing: 0 (0%) Distinct: 100 Unique: 100 (100%)

Statistic	Value
Minimum	2.654
Maximum	9.022
Mean	6.239
StdDev	1.484

The tool implements the MDLPC approach (Fayyad and Irani, 1993). We click on the APPLY button. All the continuous variables (FIRST → LAST) are discretized.



Discretize -R first-last

Relation: R_data_frame-weka.filters.supervised.attribute.Discretize-...
Instances: 100 Attributes: 5

Name: x1 Type: Nominal
Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)

No.	Label	Count
1	'(-inf-6.416414]'	50
2	'(6.416414-inf)'	50

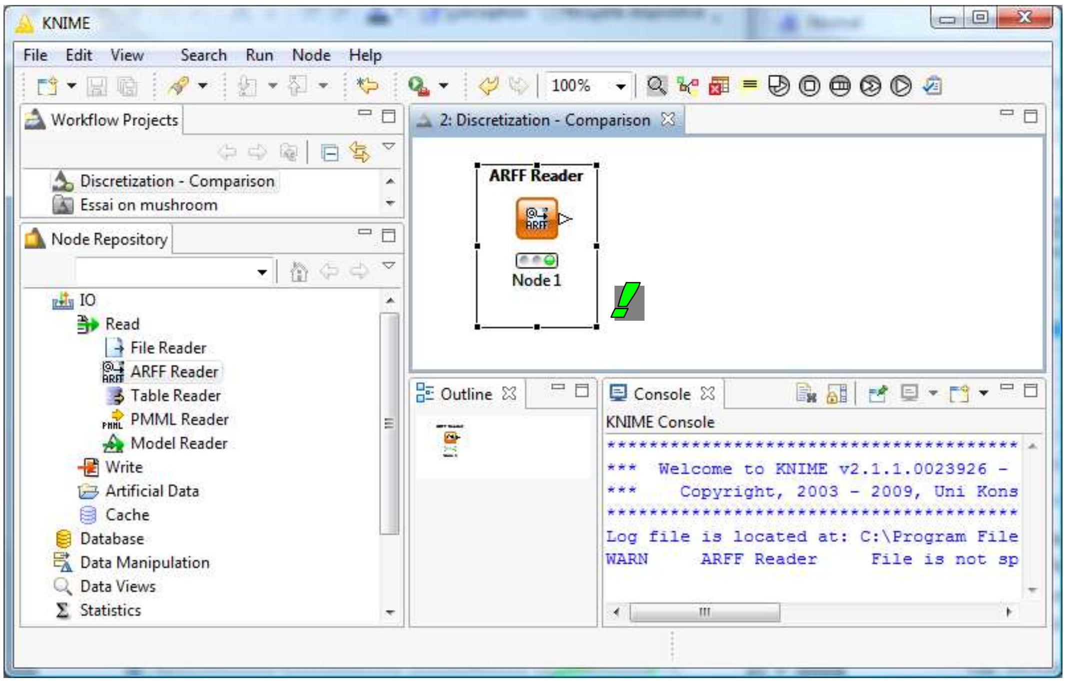
No.	Name
1	x1
2	x2
3	x3
4	x4
5	y

OK

The discretized variables are substituted to the previous ones. For each interval, we obtain both the number of intervals, the cut points, the number of instances, the classes distribution. E.g. for X1 ≤ 6.416414, we have 50 instances, they are all positives.

7 Discretization in Knime

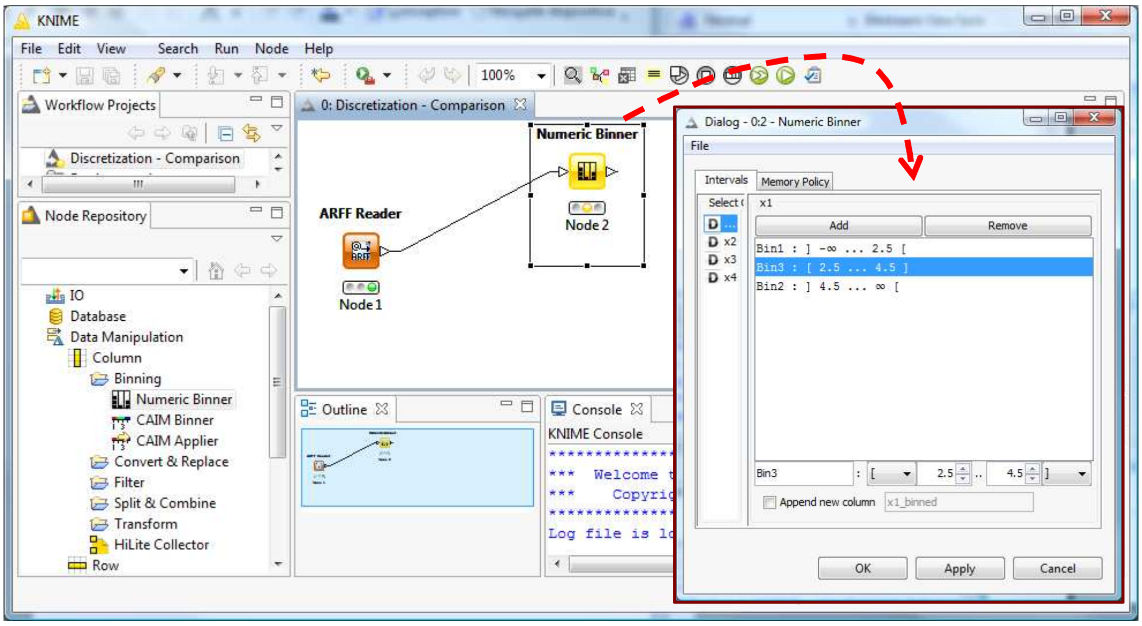
We create a new « Workflow » under Knime (http://www.knime.org/).



```
*****************************************
***   Welcome to KNIME v2.1.1.0023926 -
***     Copyright, 2003 - 2009, Uni Kons
*****************************************
Log file is located at: C:\Program File
WARN      ARFF Reader      File is not sp
```

We import the data file with the ARFF READER component. The CONFIGURE contextual menu allows to select the file. We click on the EXECUTE menu to read the file.

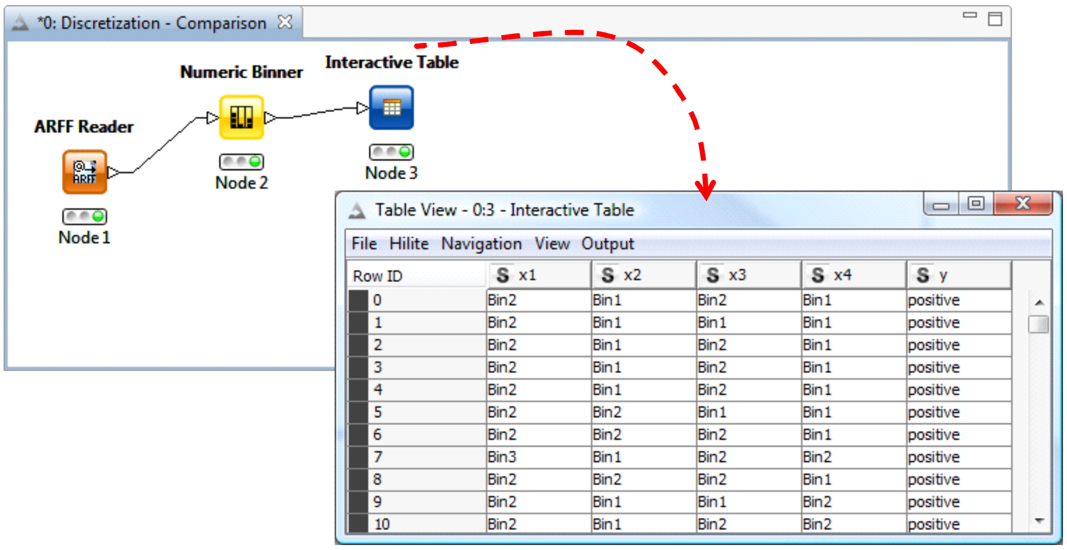
Manual discretization. The NUMERIC BINNER tool allows to specify manually the cut points (and thus the number of intervals). We add the component into the workflow, we click on the CONFIGURE menu.



```
Select (   x1
D ...
D x2
D x3
D x4
Bin1 : ] -∞ ... 2.5 [
Bin3 : [ 2.5 ... 4.5 ]
Bin2 : ] 4.5 ... ∞ [
Bin3  : [  2.5 .. 4.5 ]
```

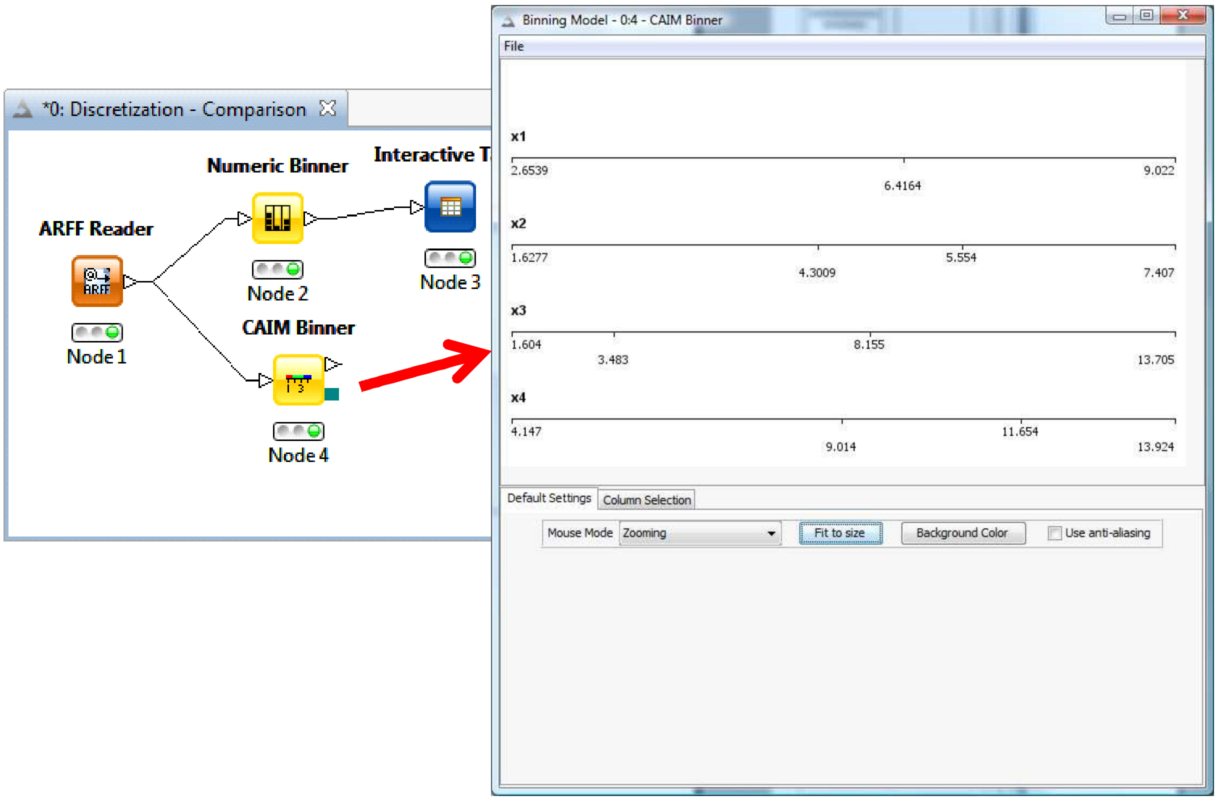
```
**************
***  Welcome t
***    Copyrig
**************
Log file is lo
```

For each variable, we can define the number of intervals and the cut points. The transformed variables are available behind the component into the workflow. We can use them with the other tools, especially with the machine learning algorithms. Here, we visualize the new columns using the INTERACTIVE TABLE component.


*0: Discretization - Comparison
ARFF Reader
Node 1
Numeric Binner
Node 2
Interactive Table
Node 3
Table View - 0:3 - Interactive Table
File Hilite Navigation View Output

Row ID	S x1	S x2	S x3	S x4	S y
0	Bin2	Bin1	Bin2	Bin1	positive
1	Bin2	Bin1	Bin1	Bin1	positive
2	Bin2	Bin1	Bin2	Bin1	positive
3	Bin2	Bin1	Bin2	Bin1	positive
4	Bin2	Bin1	Bin2	Bin1	positive
5	Bin2	Bin2	Bin1	Bin1	positive
6	Bin2	Bin2	Bin2	Bin1	positive
7	Bin3	Bin1	Bin2	Bin2	positive
8	Bin2	Bin2	Bin2	Bin1	positive
9	Bin2	Bin1	Bin1	Bin2	positive
10	Bin2	Bin1	Bin2	Bin2	positive

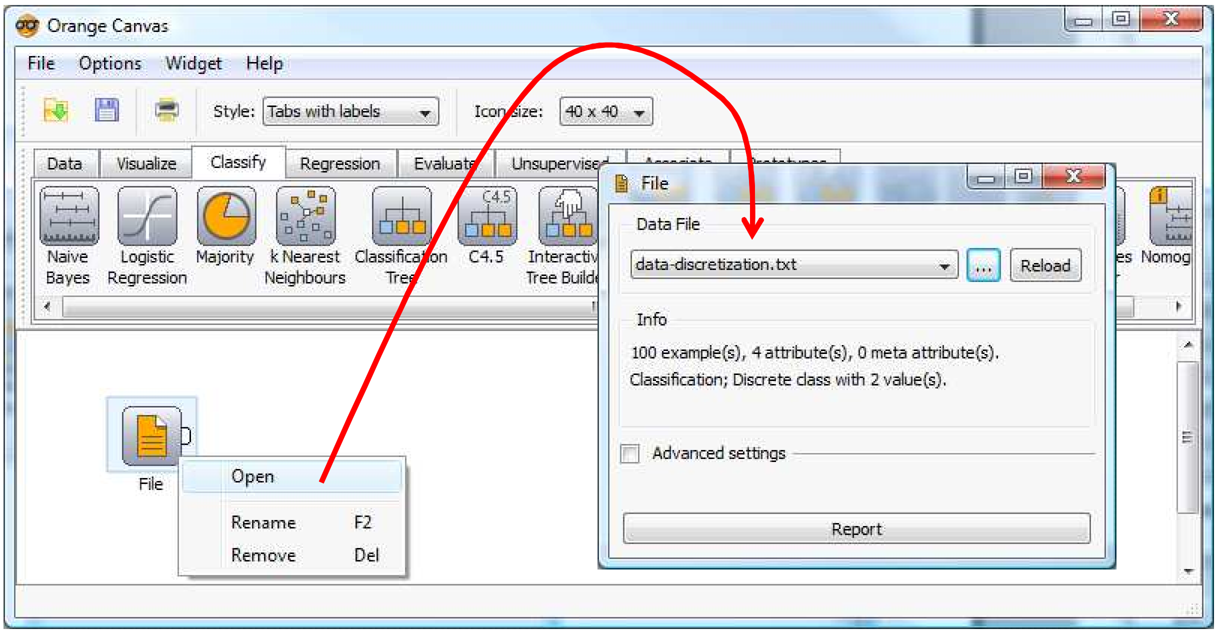
Supervised discretization. The CAIM BINNER supervised discretization tool of Knime is based on the CAIM algorithm (Kurgan et Cios, 2004; http://citeseer.ist.psu.edu/kurgan04caim.html). Let us analyze the behavior of this approach on our dataset.



Compared to MDLPC which found the good solutions for each variable, CAIM provides wrongly 3 intervals for the variable X2 (instead of 0) and 3 for X4 (instead of 4). But this is just one example among many others. It would require large-scale experiments to really assess the behavior of the algorithm.

8 Discretization in Orange

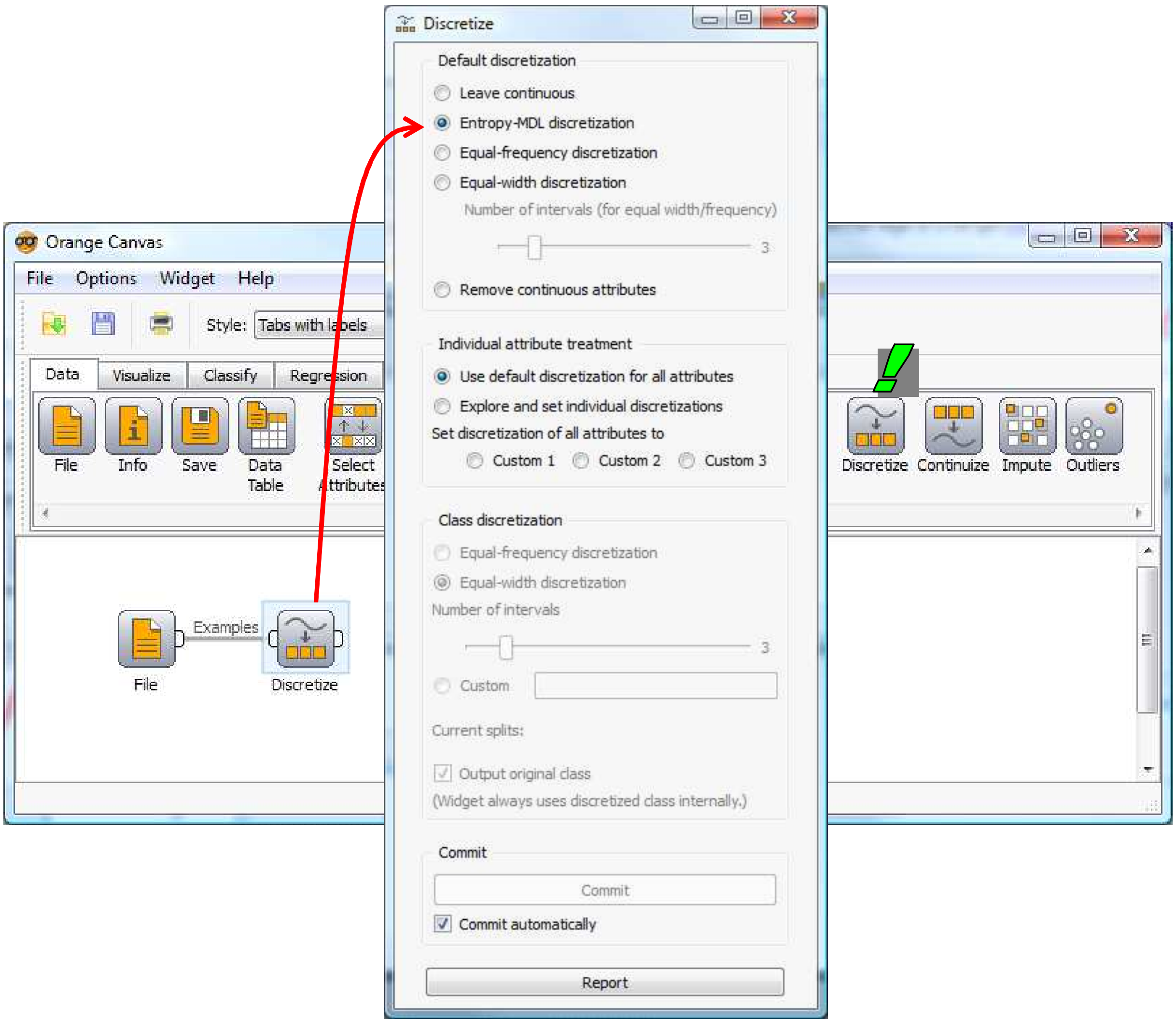
We import the data file with the FILE tool under Orange (http://www.ailab.si/orange/).



100 example(s), 4 attribute(s), 0 meta attribute(s).
Classification; Discrete class with 2 value(s).

***Note**: Orange does not handle the ARFF file, I do not why. Thus we used the tab delimited file format.*

Discretization. The DISCRETIZE tool (DATA tab) allows to perform a discretization process according various methods. The tool is rich with many options. We select the ENTROPY-MDL DISCRETIZATION which corresponds to the Fayyad and Irani's MDLPC approach according the user's guide (http://www.ailab.si/orange/doc/ofb/o_categorization.htm).


Discretize

Default discretization
- Leave continuous
- [✓ Entropy-MDL discretization]
- Equal-frequency discretization
- Equal-width discretization

Number of intervals (for equal width/frequency) 3

- Remove continuous attributes

Individual attribute treatment
- [✓ Use default discretization for all attributes]
- Explore and set individual discretizations

Set discretization of all attributes to: Custom 1 / Custom 2 / Custom 3

Class discretization
- Equal-frequency discretization
- Equal-width discretization

Number of intervals 3
- Custom

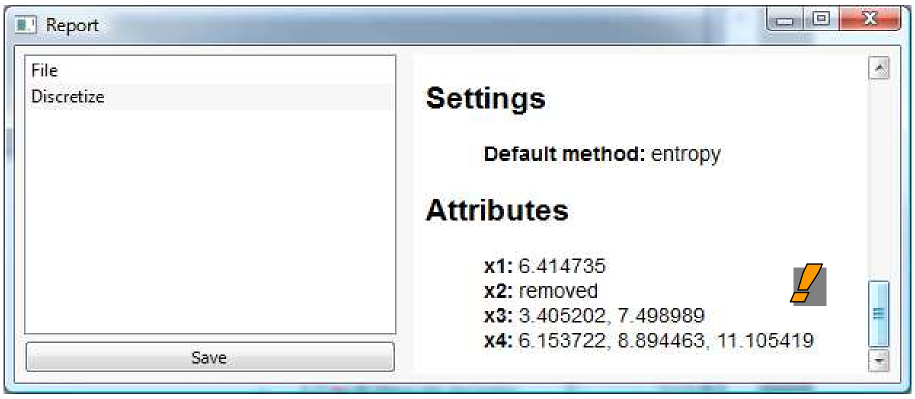
Current splits:

[✓] Output original class
(Widget always uses discretized class internally.)

Commit
[✓] Commit automatically

Orange Canvas
File Options Widget Help
Style: Tabs with labels
Data Visualize Classify Regression
File Info Save Data Table Select Attributes
Discretize Continuize Impute Outliers
Examples
File
Discretize
Commit
Report

We click on the REPORT button to obtain the results.


Report

File
Discretize

Settings

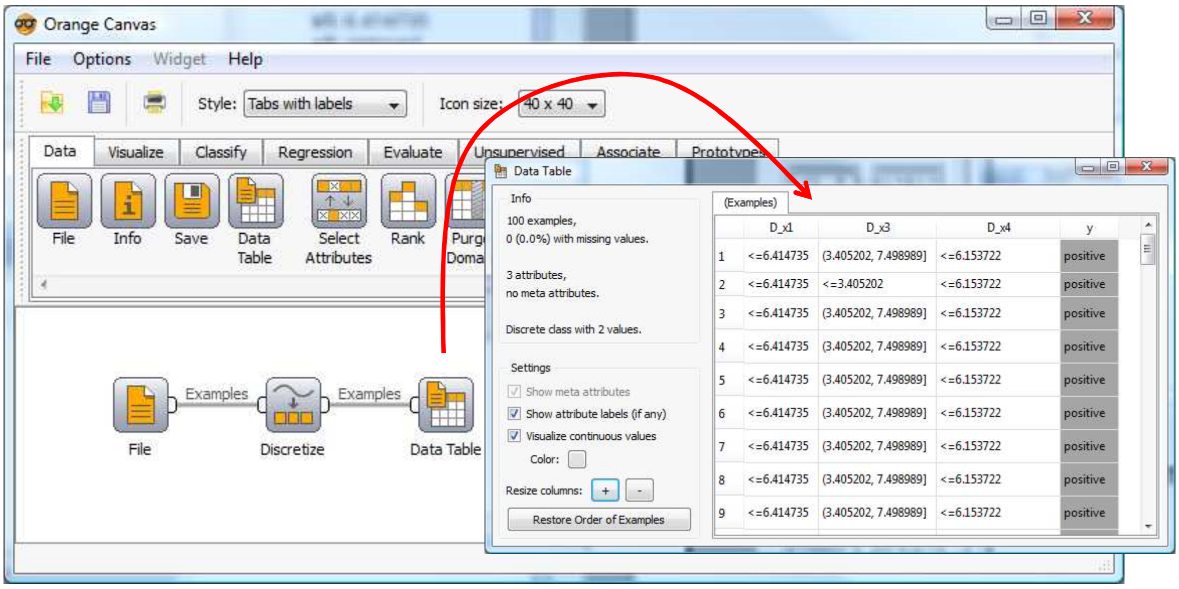
Default method: entropy

Attributes

x1: 6.414735
x2: removed
x3: 3.405202, 7.498989
x4: 6.153722, 8.894463, 11.105419

The results are roughly the same than those of Tanagra or Weka. But the cut points are slightly different. This has no incidence on the learning method implemented subsequently.

Of course, the transformed variables are available behind the component.


Orange Canvas
File Options Widget Help
Style: Tabs with labels
Icon size: 40 x 40
Data Visualize Classify Regression Evaluate Unsupervised Associate Prototypes
File Info Save Data Table Select Attributes Rank Purge Domain
Examples Examples
File Discretize Data Table
Data Table
Show meta attributes
Show attribute labels (if any)
Visualize continuous values
Color:
Resize columns: + -
Restore Order of Examples
(Examples)

Info

100 examples,
0 (0.0%) with missing values.

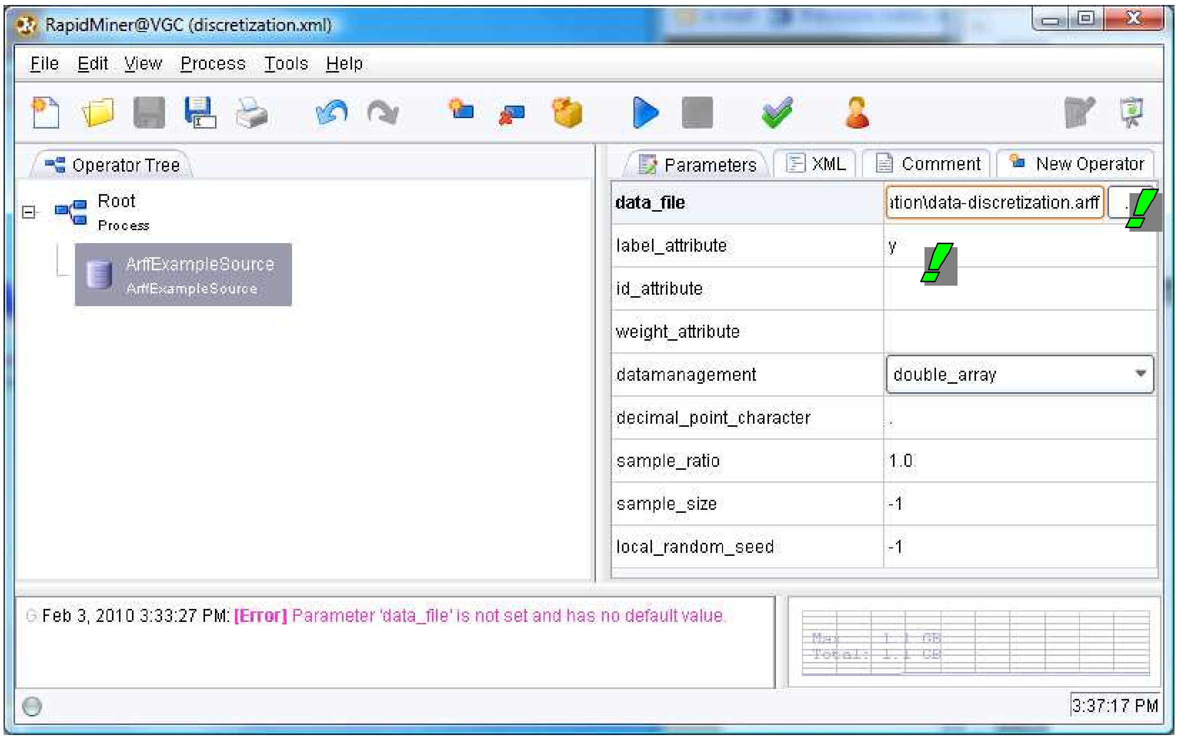
3 attributes,
no meta attributes.

Discrete class with 2 values.

	D_x1	D_x3	D_x4	y
1	<=6.414735	(3.405202, 7.498989]	<=6.153722	positive
2	<=6.414735	<=3.405202	<=6.153722	positive
3	<=6.414735	(3.405202, 7.498989]	<=6.153722	positive
4	<=6.414735	(3.405202, 7.498989]	<=6.153722	positive
5	<=6.414735	(3.405202, 7.498989]	<=6.153722	positive
6	<=6.414735	(3.405202, 7.498989]	<=6.153722	positive
7	<=6.414735	(3.405202, 7.498989]	<=6.153722	positive
8	<=6.414735	(3.405202, 7.498989]	<=6.153722	positive
9	<=6.414735	(3.405202, 7.498989]	<=6.153722	positive

9 Discretization in RapidMiner

We create a new diagram (FILE / NEW) with RapidMiner (http://rapid-i.com/content/view/181/196/). We insert the ARFF EXAMPLES SOURCE tool.


RapidMiner@VGC (discretization.xml)
File Edit View Process Tools Help
Operator Tree
Root
Process
ArffExampleSource
ArffExampleSource
Parameters XML Comment New Operator

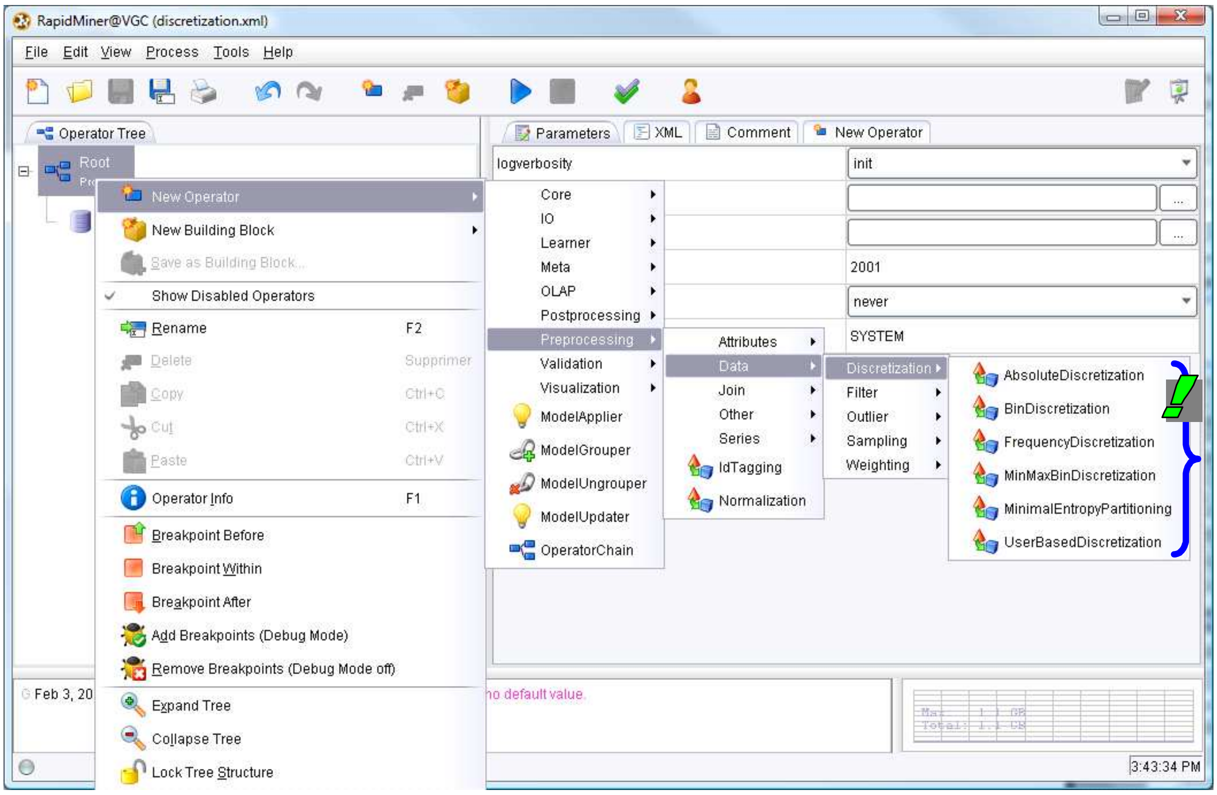
data_file	ation\data-discretization.arff
label_attribute	y
id_attribute	
weight_attribute	
datamanagement	double_array
decimal_point_character	.
sample_ratio	1.0
sample_size	-1
local_random_seed	-1

```
Feb 3, 2010 3:33:27 PM: [Error] Parameter 'data_file' is not set and has no default value.
```

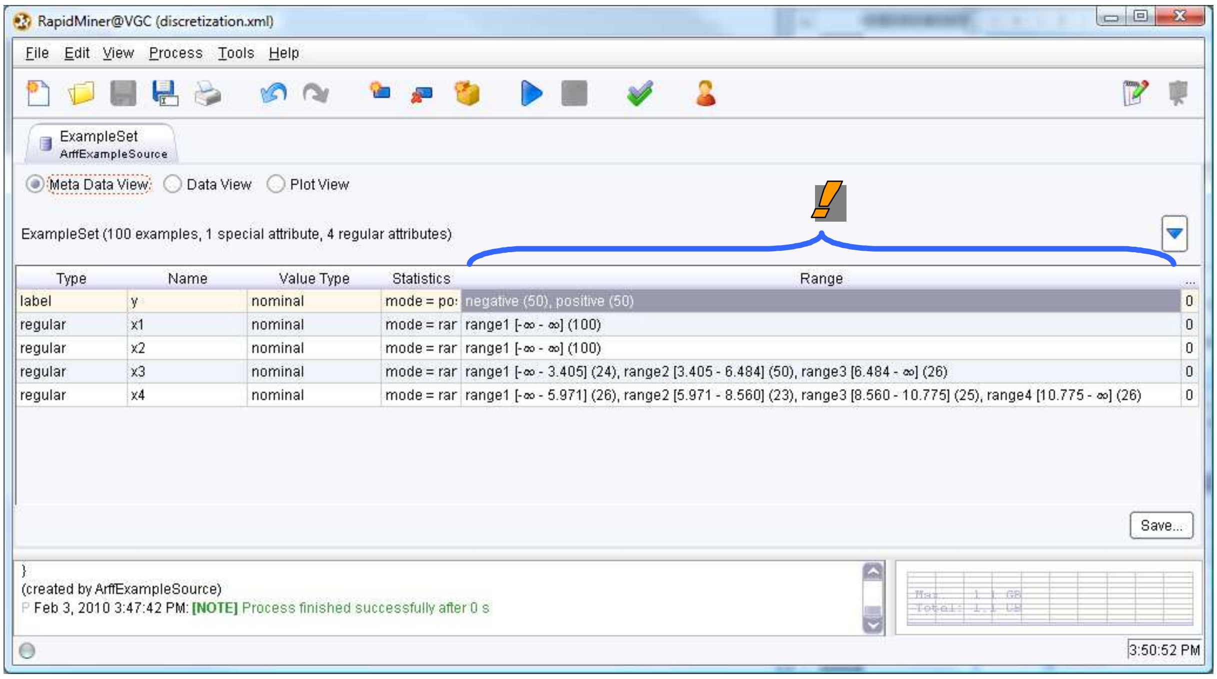
3:37:17 PM

We specify the data file name **data-discretization.arff** and the target variable **y**.

We must search in the submenus to find the tools for discretization.



We select the MINIMAL ENTROPY PARTITIONNING. The documentation gives two references (Fayyad and Irani, 1993; Dougherty et al., 1995). But we have not the details of the used algorithm. We click on the RUN button into the toolbar to perform the discretization.



ExampleSet (100 examples, 1 special attribute, 4 regular attributes)

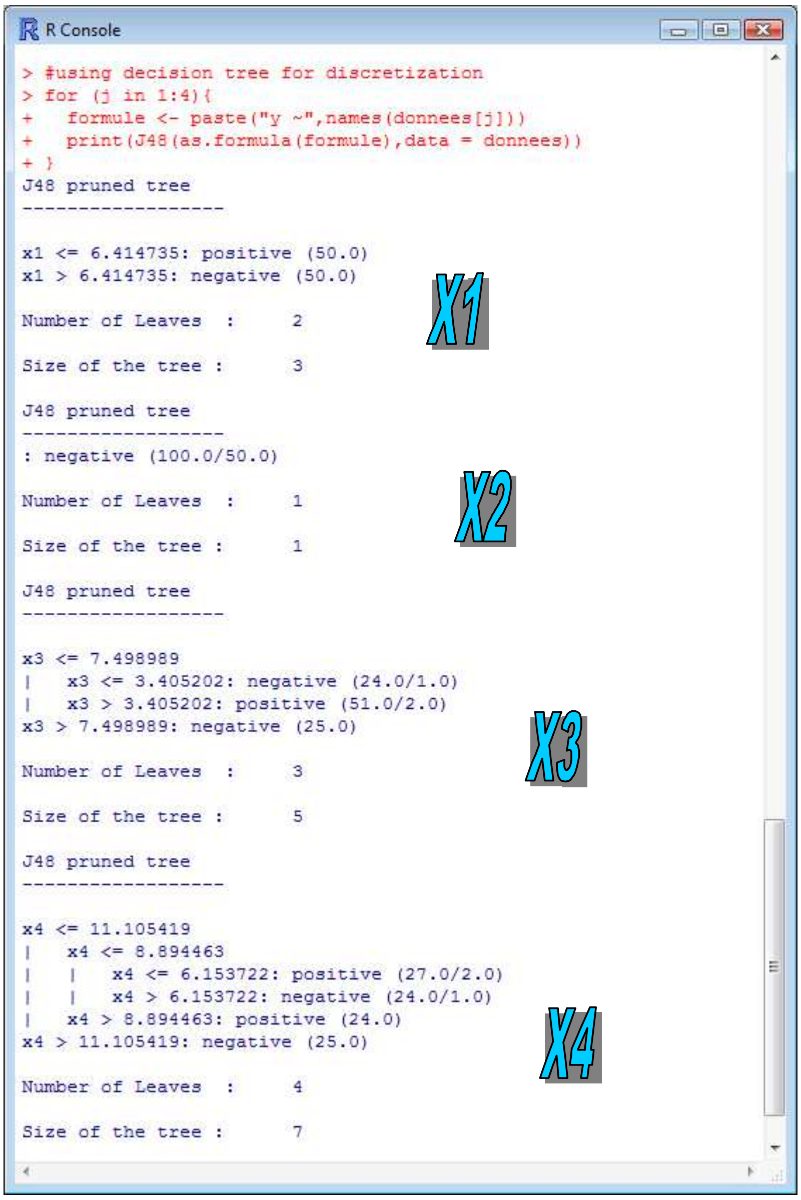
Type	Name	Value Type	Statistics	Range	...
label	y	nominal	mode = po:	negative (50), positive (50)	0
regular	x1	nominal	mode = rar	range1 [-∞ - ∞] (100)	0
regular	x2	nominal	mode = rar	range1 [-∞ - ∞] (100)	0
regular	x3	nominal	mode = rar	range1 [-∞ - 3.405] (24), range2 [3.405 - 6.484] (50), range3 [6.484 - ∞] (26)	0
regular	x4	nominal	mode = rar	range1 [-∞ - 5.971] (26), range2 [5.971 - 8.560] (23), range3 [8.560 - 10.775] (25), range4 [10.775 - ∞] (26)	0

```
}
(created by ArffExampleSource)
P Feb 3, 2010 3:47:42 PM: [NOTE] Process finished successfully after 0 s
```

RapidMiner provides some indications about the process. Into the RANGE column, we obtain the number of intervals and the cut points. We note that the software has rejected the discretization of X1 (wrongly) and X2 (rightly). About X3 and X4, the number of intervals corresponds to those of Weka and Tanagra. But some cut points are quite different, however.

10 Conclusion

The discretization of continuous variables is a usual task in the data mining process. If the unsupervised methods are fairly well known by practitioners, it is not the case for the supervised approaches. Yet, they exist and they provide relevant results in the vast majority of cases.



```
> #using decision tree for discretization
> for (j in 1:4){
+   formule <- paste("y ~",names(donnees[j]))
+   print(J48(as.formula(formule),data = donnees))
+ }
J48 pruned tree
------------------

x1 <= 6.414735: positive (50.0)
x1 > 6.414735: negative (50.0)

Number of Leaves  :     2

Size of the tree :      3

J48 pruned tree
------------------
: negative (100.0/50.0)

Number of Leaves  :     1

Size of the tree :      1

J48 pruned tree
------------------

x3 <= 7.498989
|   x3 <= 3.405202: negative (24.0/1.0)
|   x3 > 3.405202: positive (51.0/2.0)
x3 > 7.498989: negative (25.0)

Number of Leaves  :     3

Size of the tree :      5

J48 pruned tree
------------------

x4 <= 11.105419
|   x4 <= 8.894463
|   |   x4 <= 6.153722: positive (27.0/2.0)
|   |   x4 > 6.153722: negative (24.0/1.0)
|   x4 > 8.894463: positive (24.0)
x4 > 11.105419: negative (25.0)

Number of Leaves  :     4

Size of the tree :      7
```

Finally, there is a very simple alternative for cutting a continuous variable into classes in a supervised framework: we build a decision tree by specifying a single predictive variable, the one you want to discretize. We note indeed that the popular MDLPC (Fayyad and Irani, 1993) approach which performs a top down partitioning of the range of the values operates like a decision tree algorithm.

To check this idea, we use the J48 algorithm of the RWeka package under R. We create a decision tree by using in turn each continuous attribute.

The results are very similar to those of the MDLPC algorithm. We obtain the same number of intervals. The cut points however are slightly different. This is because the decision tree algorithm uses another goodness of split criterion (Quinlan's Gain ratio, 1993) during the process.

Whatever, we obtained valid solutions with the decision tree algorithm.