

1 Objectif

Discrétisation des descripteurs continus en apprentissage supervisé.

La discrétisation consiste à découper une variable quantitative en intervalles. Il s'agit d'une opération de recodage. De quantitative, la variable est transformée en qualitative ordinale. Nous devons répondre à deux questions pour mener à bien l'opération : (1) comment déterminer le nombre d'intervalles à produire ; (2) comment calculer les bornes de discrétisation à partir des données. La résolution ne se fait pas forcément dans cet ordre.

J'ai coutume de dire que le découpage d'expert est le meilleur possible. En effet, lui seul peut fournir une discrétisation raisonnée tenant compte des connaissances du domaine, tenant compte de tout un tas de contraintes dont on n'a pas idée si on se base uniquement sur les données, et en adéquation avec les objectifs de l'étude. Malheureusement, la démarche s'avère délicate parce que : d'une part, les connaissances ne sont pas toujours au rendez vous ou sont difficilement quantifiables ; d'autre part, elle n'est pas automatisable, le traitement d'une base comportant des centaines de variables se révèle rapidement ingérable. Souvent donc, nous sommes obligés de nous baser uniquement sur les données pour produire un découpage qui soit un tant soit peu pertinent.

Discrétisation comme prétraitement des variables en apprentissage supervisé. Tout d'abord, il faut situer le canevas dans lequel nous réalisons l'opération. Selon le cas, il est évident que la démarche et les critères utilisés ne seront pas les mêmes. Dans ce didacticiel, nous nous plaçons dans le cadre de l'apprentissage supervisé. Les variables quantitatives sont préalablement recodées avant d'être présentées à un algorithme d'apprentissage supervisé. La variable à prédire, elle, est naturellement qualitative. Lors de la discrétisation, il est par conséquent souhaitable que les groupes soient le plus purs possibles c.-à-d. les individus situés dans le même intervalle doivent appartenir majoritairement à l'une des modalités de la variable à prédire.

Les motivations d'une telle opération de prétraitement sont multiples¹ :

- Certaines méthodes supervisées ne savent manipuler que les descripteurs qualitatifs.
- Même si la technique ne le requiert pas, les variables découpées en intervalles sont parfois plus faciles à appréhender lors de l'interprétation des résultats.
- Certains algorithmes s'avèrent nettement plus rapides lorsqu'ils manipulent des descripteurs discrets. C'est le cas par exemple des arbres de décision. Même s'ils peuvent découper à la volée les variables lors de la segmentation, la discrétisation locale requiert toute une série de calculs que l'on peut s'épargner si l'on discrétisait préalablement les descripteurs continus.
- La discrétisation permet d'appréhender les relations de nature non linéaire. Par exemple, les jeunes enfants et les personnes âgées sont les plus vulnérables à certaines maladies. Une équation linéaire de prédiction ne permet pas de traduire ce type de relation.
- Enfin, la discrétisation est une approche simple pour harmoniser les bases hétérogènes, contenant des descripteurs de nature différente.

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/>).

Il y a plusieurs manières de distinguer les algorithmes de discrétisation².

Approche multivariée vs. Approche univariée. Les approches multivariées consistent à découper une variable en tenant compte des autres variables. Séduisante a priori, en effet les descripteurs sont rarement indépendants deux à deux dans une base, elles sont très peu utilisées dans la pratique. Parce qu'elles reposent souvent sur des calculs itératifs compliqués et très gourmands en ressources machines. Et surtout, parce qu'elles ne sont pas disponibles dans les logiciels. Les approches univariées consistent à découper chaque variable indépendamment des autres. Elles sont, par la force des choses (simplicité, disponibilité), les méthodes les plus utilisées, malgré les réserves que l'on pourrait émettre quant à leurs propriétés théoriques.

Approche non supervisée vs. Approche supervisée. L'approche non supervisée s'appuie exclusivement sur les informations fournies par la variable à découper pour déterminer les bornes de discrétisation. Les méthodes les plus connues (discrétisation en intervalles de largeur égales, en intervalles de fréquence égales, algorithme de Fisher) requièrent le nombre d'intervalles en paramètres³. Leur principal avantage est leur popularité, ce sont des approches « passe-partout » que l'on peut mettre en œuvre dans tout contexte, autres que l'analyse supervisée. Leur principal inconvénient est d'ignorer les informations en provenance de la variable cible pour définir le bon découpage. Ce que font justement les approches supervisées qui, pour la plupart, savent déterminer à la fois le nombre d'intervalles et les bornes de découpage.

Dans ce didacticiel, nous nous en tiendrons aux approches univariées. Nous comparerons le comportement des techniques supervisées et non supervisées implémentées dans les logiciels **Tanagra 1.4.35**, **Sipina 3.3**, **R 2.9.2** (package dprep), **Weka 3.6.0**, **Knime 2.1.1**, **Orange 2.ob** et **RapidMiner 4.6.0**. Comme nous pouvons le constater, tout logiciel de Data Mining se doit de proposer ce type d'outils. Nous mettrons en avant le paramétrage et la lecture des résultats.

² R. Rakotomalala, « Graphes d'Induction », Thèse de doctorat ; Chapitre 9, « Discrétisation des attributs continus », pp. 209-244 ; 1997 (<http://eric.univ-lyon2.fr/~ricco/publications.html>).

³ Ce n'est pas tout à fait vrai. Il existe dans la littérature des formules pour déterminer automatiquement le « bon » nombre d'intervalles pour les techniques univariées non supervisées. En voici quelques unes (<http://www.info.univ-angers.fr/~gh/wstat/dscr.php>) ; « Découpages en classes et discrétisation », Gilles Hunault, Université d'Angers) :

Appellation	Formule
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	$\frac{b - a}{2 \times q \times n^{(-1/3)}}$

Où n est le nombre d'observations ; b le maximum ; a le minimum ; σ l'écart type ; q l'intervalle interquartile.

Concernant Tanagra, nous montrons comment exploiter les variables transformées en aval du composant de discrétisation. Quasiment tous les logiciels étudiés dans ce didacticiel proposent les mêmes fonctionnalités.

2 Données

Nous utilisons des données synthétiques. La variable cible comporte 2 modalités (positif vs. négatif). Nous avons généré 4 descripteurs continus avec des distributions conditionnelles différentes.

Le principal intérêt de ce fichier est que nous connaissons à l'avance les résultats que nous devrions obtenir. Et, selon que nous appréhendons les données de manière supervisée ou non, la nature des résultats peut être différente. Pour s'en persuader, nous traçons dans un premier temps les fonctions de densité empiriques sans tenir compte des classes d'appartenance.

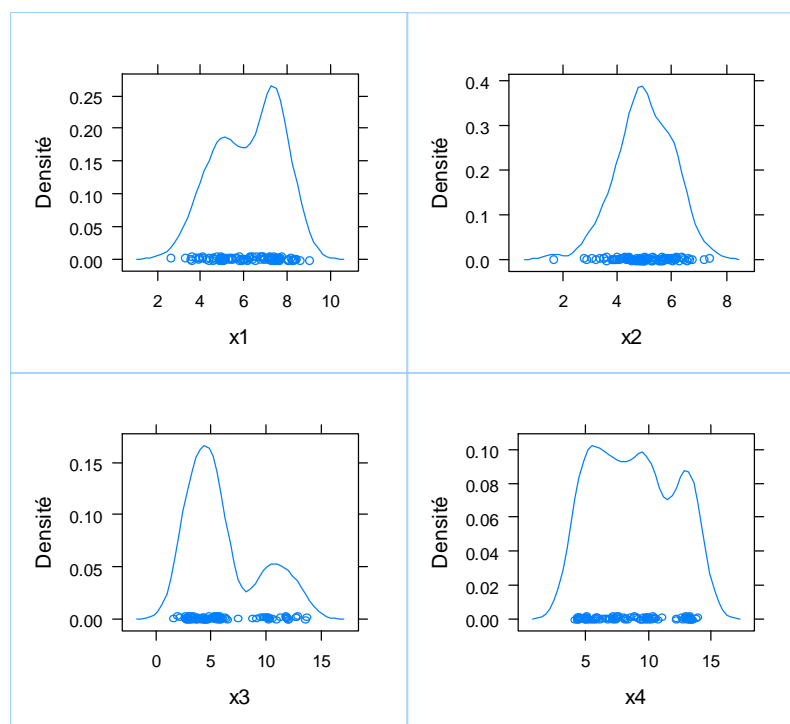


Figure 1 - Distribution des variables prédictives

Difficile de parier sur des découpages pertinents à la vue de ces seuls graphiques. On devine bien ici et là des « creux » semblant séparer des groupes d'observations. Mais en l'absence d'informations sur la variable cible, nous ne savons pas s'ils séparent les positifs des négatifs.

Avec les mêmes variables, traçons maintenant les fonctions de densités conditionnellement aux classes d'appartenance.

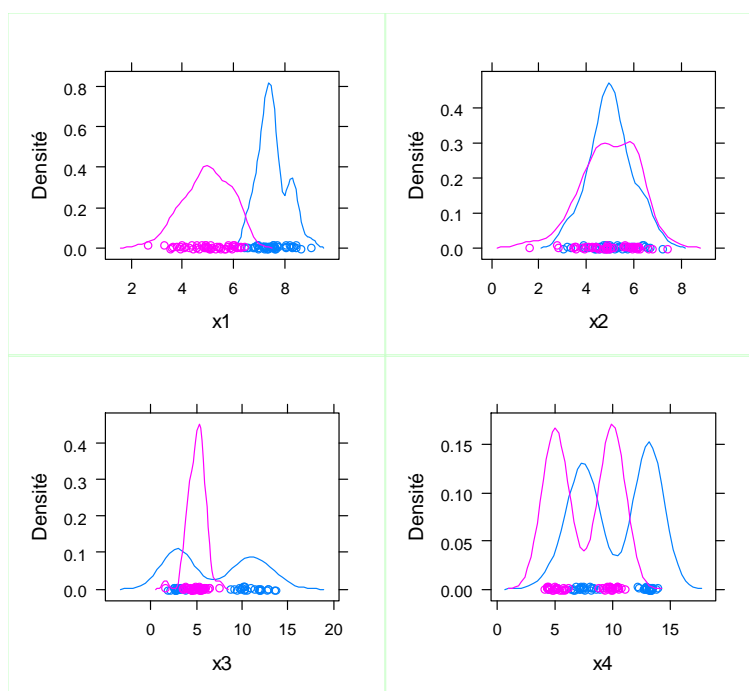


Figure 2 - Distribution des variables prédictives conditionnellement à la classe d'appartenance

Les conclusions sont différentes. Mis à part la variable x_2 où manifestement les distributions conditionnelles se superposent, toutes les autres peuvent être discrétisées avec plus ou moins de bonheur, le nombre d'intervalles approprié est différent d'une situation à l'autre. Nous nous en souviendrons lorsqu'il faudra analyser les résultats.

Moralité de tout ceci : en apprentissage supervisé, sans les informations liées aux classes, il n'est pas possible de déceler correctement les points de coupure adéquats en se basant uniquement sur les caractéristiques de la variable à découper.

Dans ce qui suit, nous utiliserons tour à tour les techniques non supervisées et supervisées proposés par les différents logiciels. Pour les premières, nous nous focaliserons principalement sur le découpage en intervalles de largeur égales et en intervalles de fréquences égales. Toutes deux requièrent l'indication du nombre d'intervalles à produire. Nous le fixerons arbitrairement à 5. L'idée étant d'en spécifier suffisamment pour espérer obtenir quelques groupes plus ou moins purs. Concernant les techniques supervisées, nous laisserons le soin à l'algorithme de choisir à la fois le nombre de classes et les bornes de discrétisation.

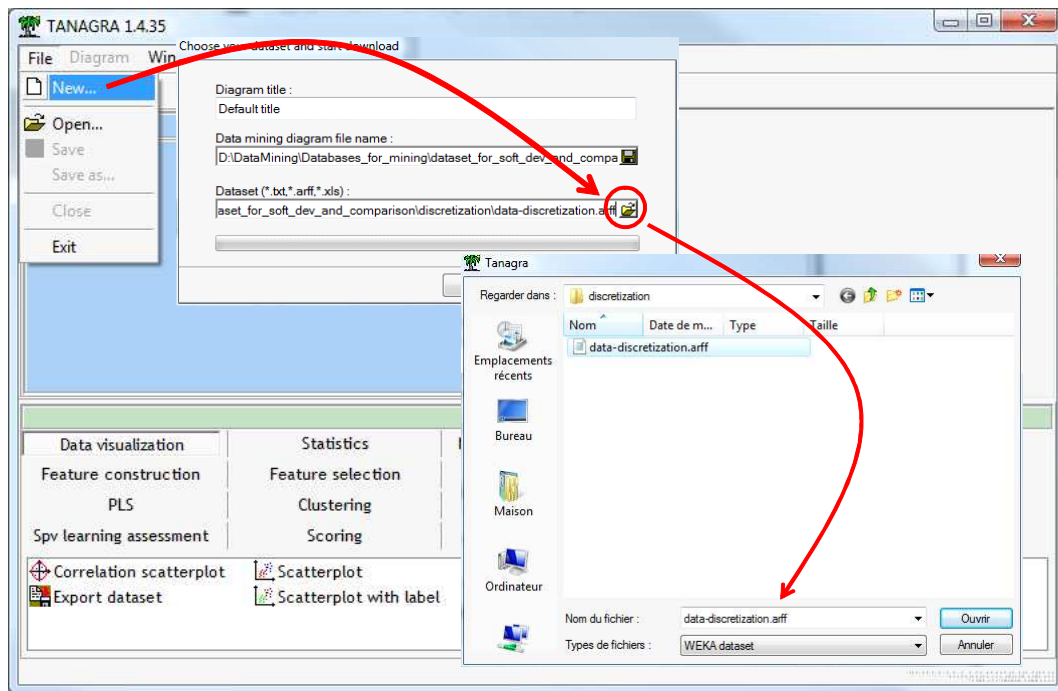
Le fichier de données au format ARFF⁴ (Weka) est accessible en ligne⁵.

3 Discrétisation avec Tanagra

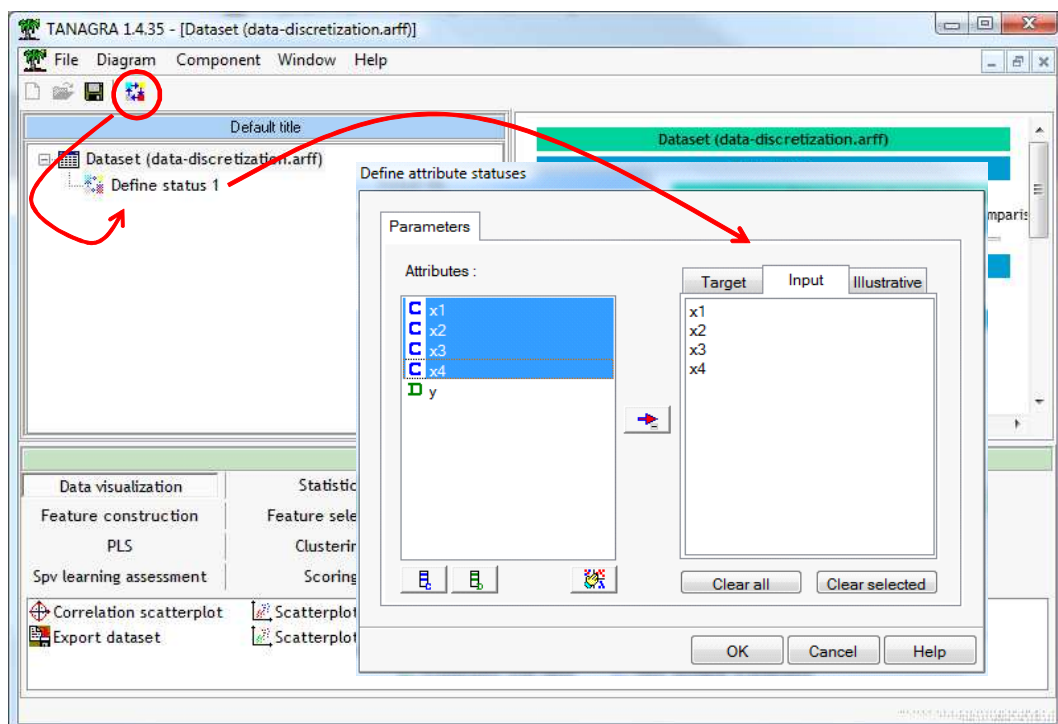
Importation des données. Après avoir démarré Tanagra, nous créons un nouveau diagramme (FILE / NEW) et nous importons le fichier de données.

⁴ <http://tutoriels-data-mining.blogspot.com/2008/03/importer-un-fichier-weka-dans-tanagra.html>

⁵ <http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/data-discretization.arff>

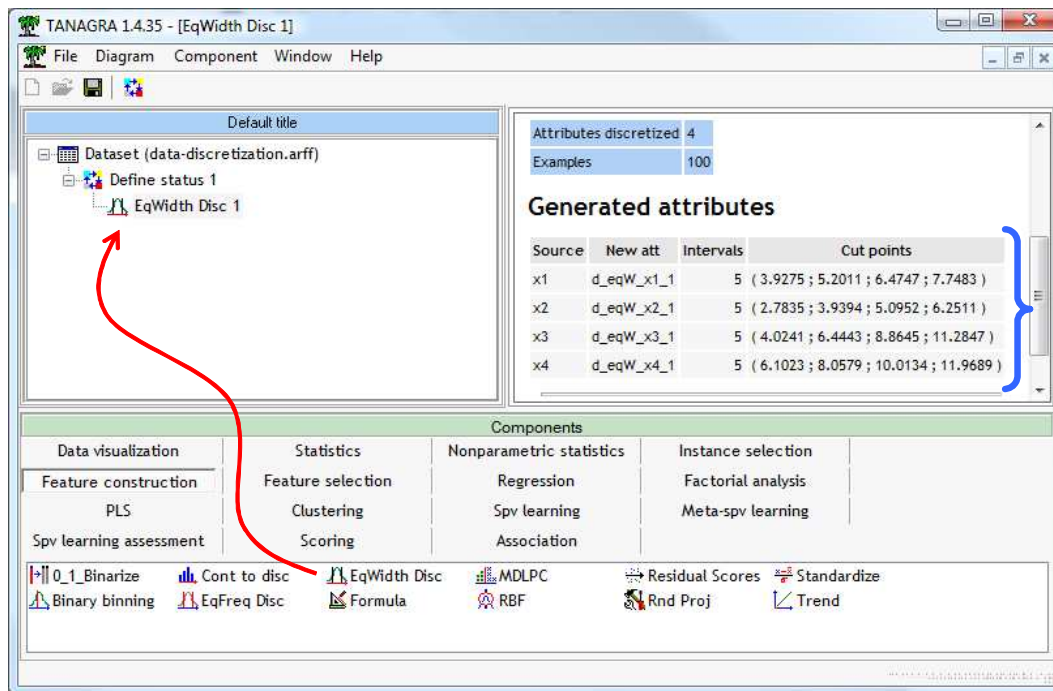


Discrétisation non supervisée. Pour discrétiser avec la méthode des intervalles de largeurs égales, nous devons tout d'abord spécifier les variables à traiter à l'aide du composant DEFINE STATUS.



Nous introduisons alors le composant **EqWidthDisc** (onglet FEATURE CONSTRUCTION)⁶. Par défaut, il initie un découpage en 5 intervalles de largeurs égales. Nous actionnons directement le menu contextuel VIEW.

⁶ L'autre composant disponible est **EqFreqDisc**, la discrétisation avec des intervalles de fréquences égales.



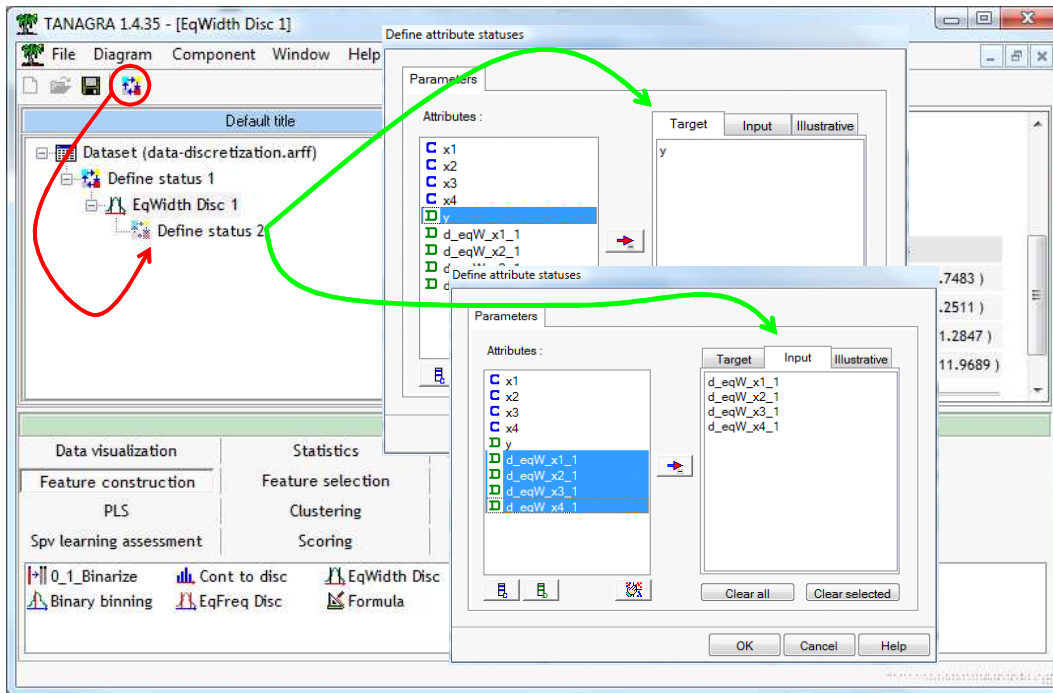
Bien évidemment, par rapport aux configurations décrites par les distributions conditionnelles (Figure 2), demander systématiquement 5 intervalles est exagéré.

Néanmoins, on se rend compte que certaines bornes « trouvent » un peu par hasard les bonnes séparations. Si l'on considère le découpage de la variable X_1 , il semble ainsi que la valeur 6.47 ne soit pas si mauvaise finalement. En revanche, les autres seuils découpent indûment l'espace de représentation. Et c'est bien là le danger. Accroître le nombre d'intervalles peut améliorer les chances de tomber sur les bornes appropriées. Mais, dans le même temps, nous fragmentons inutilement les données, avec le risque d'hypothéquer la méthode d'apprentissage supervisé placées en aval.

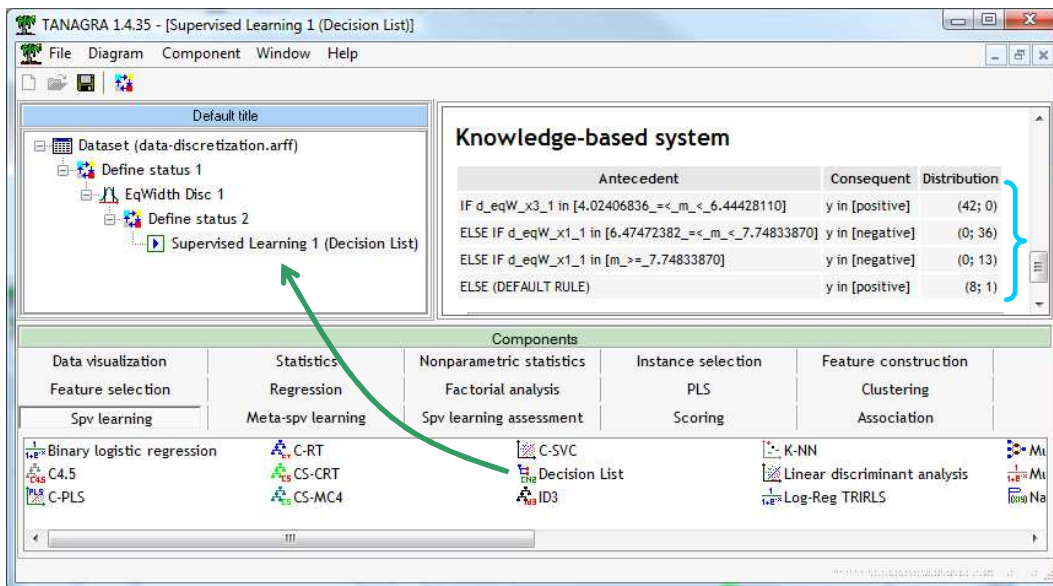
Il en est de même pour la variable X_3 . Les seuils 4.02 et 6.44 tombent fort à propos. Les autres en revanche n'induisent pas un découpage pertinent du point de vue de la variable cible.

A contrario, pour la variable X_2 , il est évident qu'il n'y a pas de découpage possible. La procédure, purement mécanique, nous fournit 4 seuils qui ne correspondent strictement à rien.

Apprentissage à partir des descripteurs discrétisés. Voyons comment intégrer ces nouveaux descripteurs dans le processus de modélisation. Nous insérons de nouveau le composant DEFINE STATUS. Nous plaçons en INPUT les variables discrétisées, en TARGET la variable cible.



Nous insérons la méthode d'induction des listes de décision (onglet SPV LEARNING)⁷.



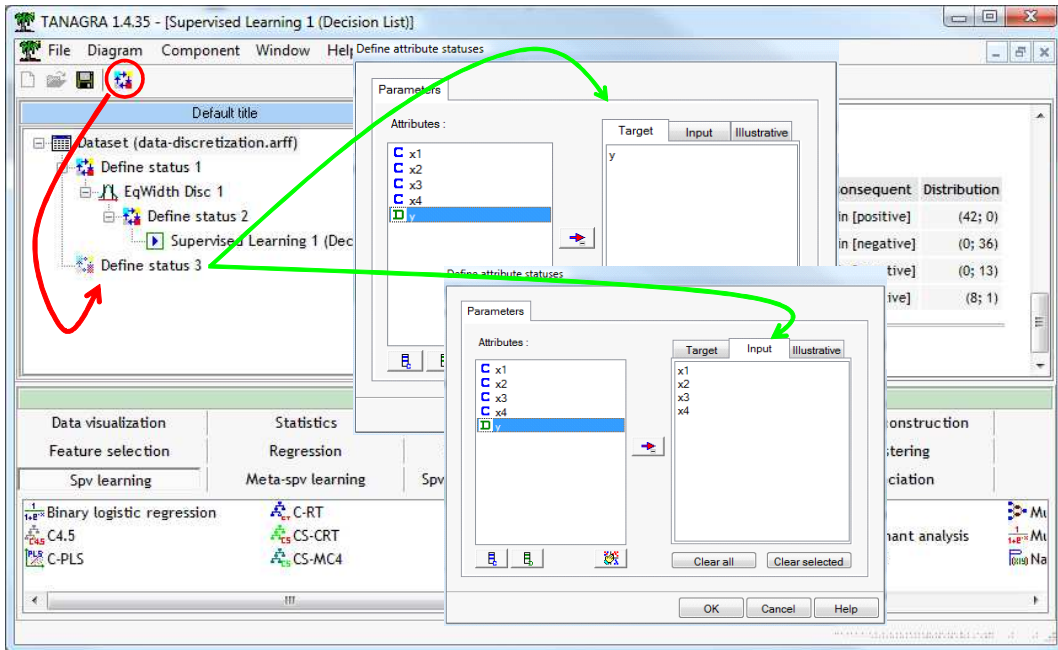
Le modèle est d'excellente qualité avec un taux d'erreur très faible (**Note** : il s'agit d'un fichier spécialement conçu pour ce tutoriel, ne le perdons pas de vue). Ce n'est guère étonnant compte tenu de ce que nous disions plus haut. La procédure a permis quand même de mettre à jour certaines bornes de discrétisation.

Lorsque nous analysons les règles, nous constatons l'inconvénient d'un nombre exagéré d'intervalles. Nous avons besoin de 2 règles (n°2 et n°3) pour désigner les négatifs, alors qu'une seule aurait suffi : « ELSE IF X1 ≥ 6.47 THEN Y = NEGATIVE ». La borne de découpage « 7.748 » de

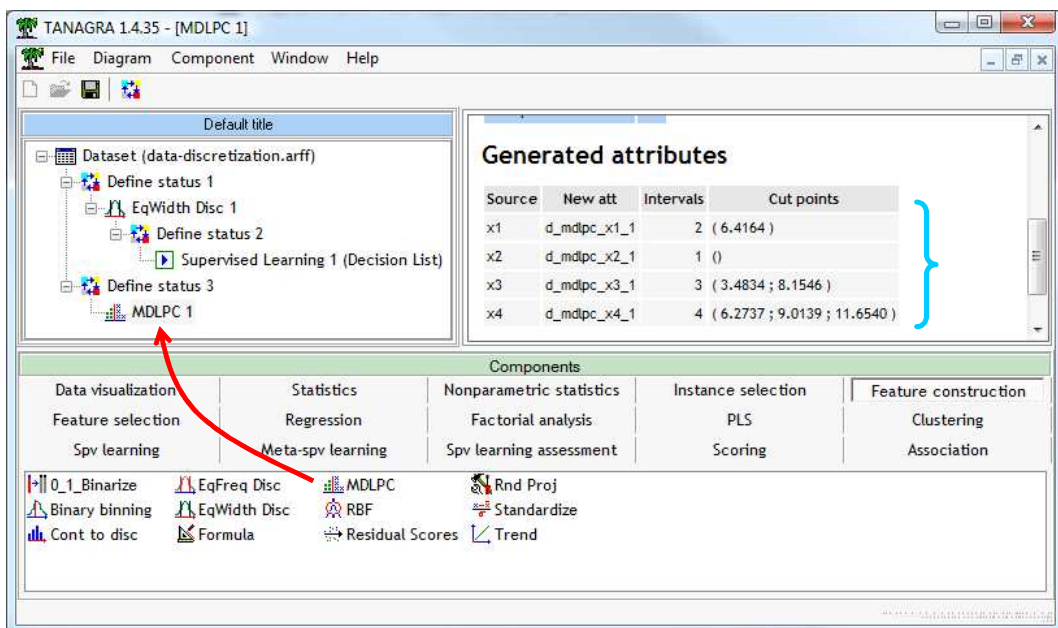
⁷ Voir <http://tutoriels-data-mining.blogspot.com/2009/11/induction-de-regles-predictives.html> et <http://tutoriels-data-mining.blogspot.com/2008/03/listes-de-dcision-vs-arbres-de-dcision.html>

X1 est superflue, elle engendre une fragmentation de la règle. Quand le concept à apprendre est complexe, et les données rares, ce phénomène est préjudiciable à l'exploration des solutions.

Discrétisation supervisée. Nous passons directement à la discrétisation supervisée maintenant. Nous insérons un nouveau composant DEFINE STATUS à la racine du diagramme. Nous devons indiquer au logiciel que les variables X1...X4 (INPUT) doivent être découpées en fonction de la cible Y (TARGET).



Nous ajoutons dans le diagramme le composant MDLPC (onglet FEATURE CONSTRUCTION) qui implémente une méthode de discrétisation supervisée descendante (*U. Fayyad et K. Irani, « Multi-interval discretization of continuous-valued attributes for classification learning », in Proc. of IJCAI, pp.1022-1027, 1993*). Nous actionnons directement le menu contextuel VIEW, il n'y a pas de paramètres à spécifier.



Tanagra affiche le nombre d'intervalles et les bornes de discrétisation pour chaque variable

Schématiquement, l'algorithme essaie de détecter une première borne permettant de subdiviser les observations en 2 parties. Si le gain d'information est significatif, il opère le découpage et poursuit récursivement sur les deux sous-ensembles ainsi constitués. La détermination du nombre final d'intervalles est donc intimement liée avec la détection des bornes.

Force est de constater que la méthode est particulièrement efficace, du moins sur nos données. Pour chaque situation ($X_1 \dots X_4$; Figure 2), elle détermine à la fois le nombre d'intervalles et les bornes adéquates. Concernant la variable X_2 , elle indique, à juste titre, qu'aucun découpage de cette variable n'est pertinent.

Apprentissage supervisé consécutivement à MDLPC. Nous initions de nouveau un apprentissage avec les listes de décision. Nous obtenons un classifieur au moins aussi précis que précédemment, mais avec moins de règles. Les innombrables expérimentations réalisées dans la littérature confirment souvent ce type de résultat⁸.

Dans notre exemple, la seule Learning variable X_1 suffit à produire un classifieur performant.

The screenshot shows the TANAGRA 1.4.35 interface. The main window displays a workflow diagram on the left and a 'Knowledge-based system' window on the right. The workflow includes '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', and 'Supervised Learning 2 (Decision List)'. The 'Knowledge-based system' window shows the following rules:

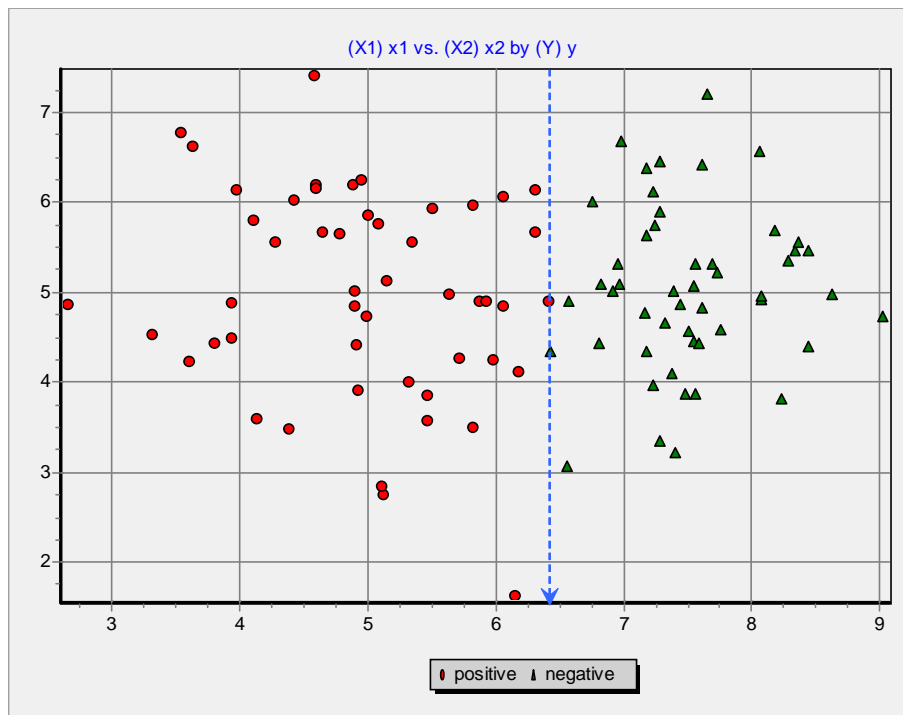
Antecedent	Consequent	Distribution
IF d_mdpc_x1_1 in [m_<_6.41641331]	y in [positive]	(50; 0)
ELSE IF d_mdpc_x1_1 in [m_>=6.41641331]	y in [negative]	(0; 50)
ELSE (DEFAULT RULE)	y in [positive]	(0; 0)

Below the rules, it indicates 'Computation time : 0 ms.' and 'Created at 03/02/2010 09:33:42'. A red arrow points from the 'Supervised Learning 2 (Decision List)' component in the workflow to the 'Decision List' component in the 'Components' panel at the bottom of the interface.

Interprétation géométrique. Si l'on projette les observations dans le plan (X_1, X_2), on note que la frontière (bleu pointillé) entre les positifs (rond rouge) et les négatifs (triangle vert) définie sur X_1 les sépare parfaitement.

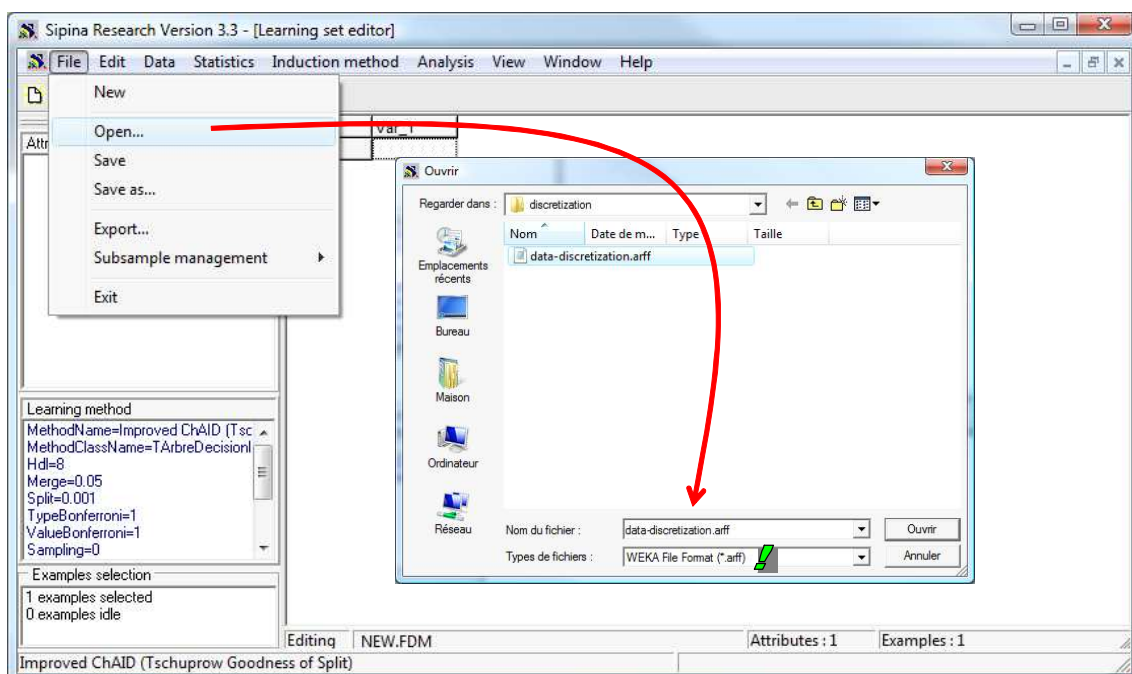
Deux points sont à la limite ($+ : X_1 = 6.4147$; $- : X_1 = 6.4181$), mais ils sont respectivement du bon côté de la frontière.

⁸ J. Dougherty, R. Kohavi, M. Sahami, « Supervised and Unsupervised Discretization of Continuous Features », 1995 (<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.47.6141>).

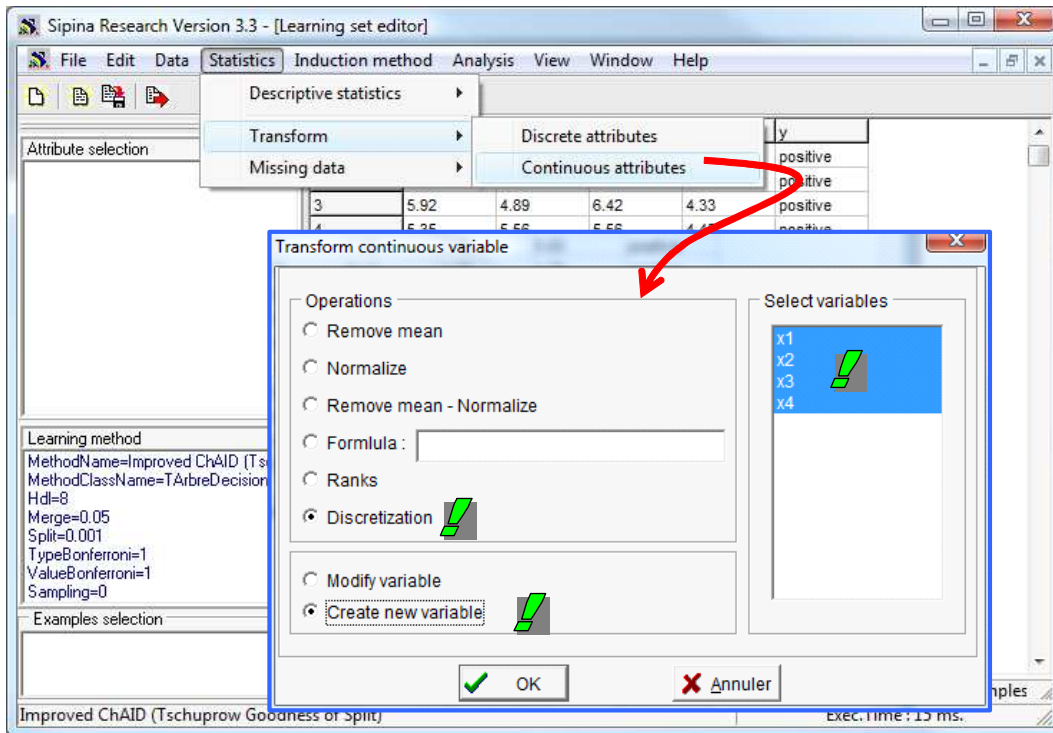


4 Discrétisation avec SIPINA

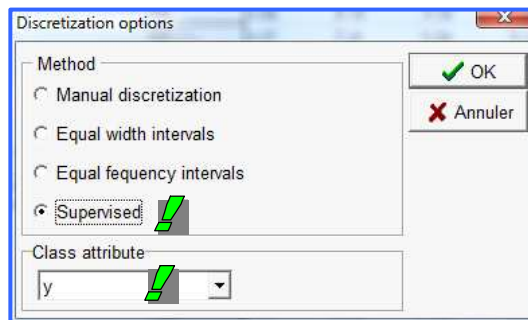
Importation des données. Après avoir démarré SIPINA (<http://eric.univ-lyon2.fr/~ricco/sipina.html>), nous chargeons les données en actionnant le menu FILE / OPEN et en sélectionnant le fichier **data-discretization.arff**.



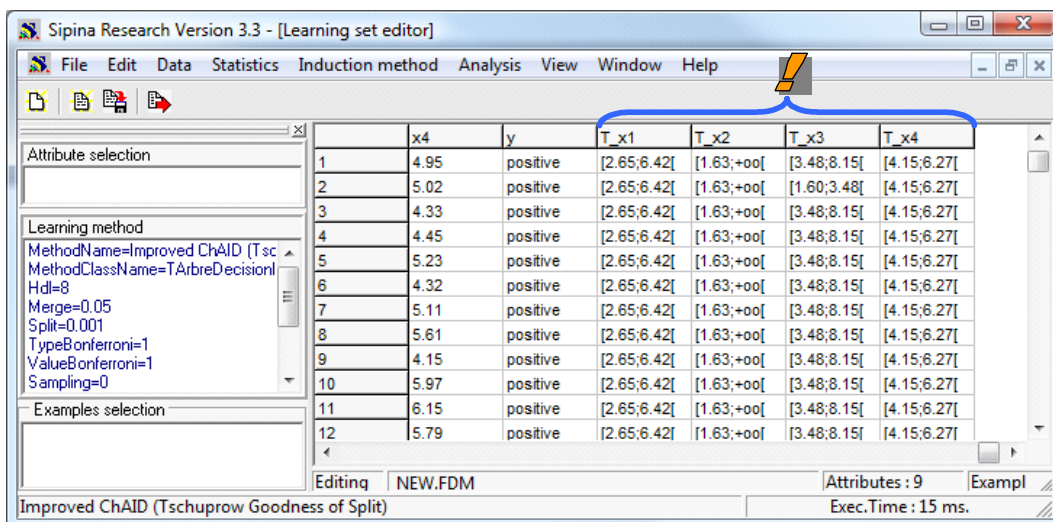
Discrétisation des variables. Les outils de discrétisation sont situés dans le menu STATISTICS / TRANSFORM / CONTINUOUS ATTRIBUTES. Dans la boîte de dialogue qui apparaît, nous sélectionnons tous les descripteurs continus et nous demandons une discrétisation. Nous souhaitons que de nouvelles variables soient créées et affichées dans la grille de données.



Après validation, une seconde boîte de paramétrage apparaît. Nous pouvons choisir la technique à mettre en œuvre. Si nous voulons utiliser une méthode supervisée (MDLPC en l'occurrence), nous spécifions SUPERVISED, puis nous indiquons la variable cible (Y) dans la liste déroulante.



Il ne reste plus qu'à valider (bouton OK).



Dans la grille sont maintenant affichées les nouvelles variables discrétisées. Les résultats (nombre d'intervalles et bornes) sont identiques à ceux de Tanagra.

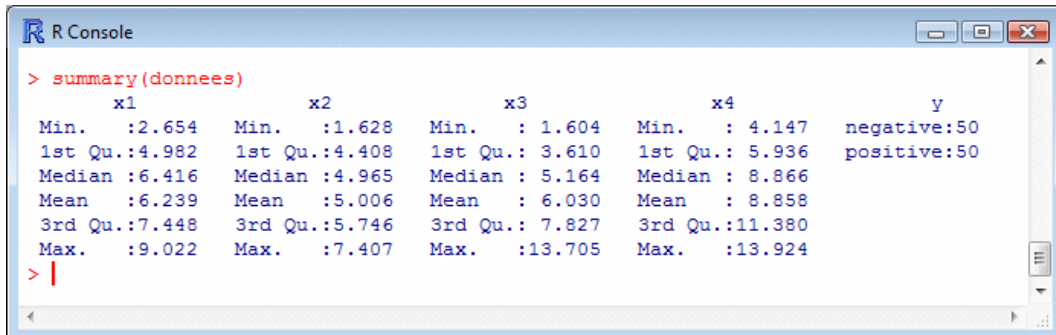
Remarque : Notons qu'il est possible de découper manuellement les variables (MANUAL DISCRETIZATION) en spécifiant explicitement les bornes de discrétisation (séparés par des « ; ») dans la boîte de paramétrage de SIPINA.

5 Discrétisation avec R (package « dprep »)

Le package **dprep** de R (<http://www.r-project.org/>) propose une série de fonctions pour le découpage d'une variable continue. Nous insérons les commandes suivantes pour charger le fichier.

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

Quelques statistiques descriptives permettent de vérifier l'intégrité des données.

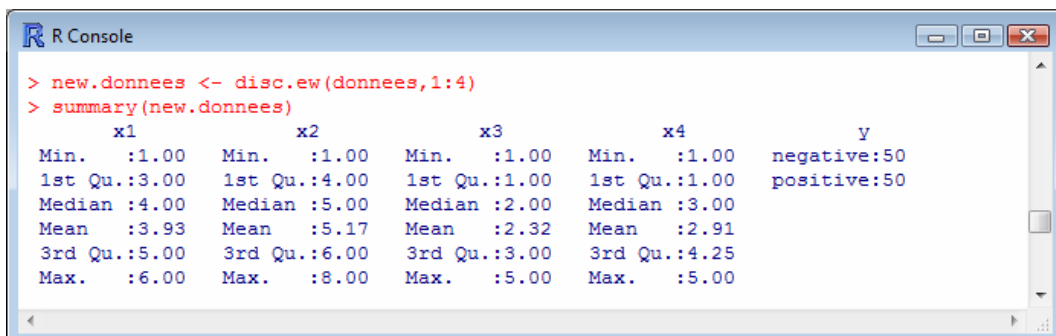


```
R Console
> summary(donnees)
      x1          x2          x3          x4          y
Min.   :2.654   Min.   :1.628   Min.   : 1.604   Min.   : 4.147   negative:50
1st Qu.:4.982   1st Qu.:4.408   1st Qu.: 3.610   1st Qu.: 5.936   positive:50
Median :6.416   Median :4.965   Median : 5.164   Median : 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
> |
```

Discrétisation avec des intervalles de largeur égales. Nous introduisons les commandes suivantes pour découper les variables X1...X4 en intervalles de largeur égales.

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

La procédure produit les variables recodées dans un nouveau data.frame. Le nombre d'intervalles est déterminé avec la formule de Scott (cf. plus haut).



```
R Console
> 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
```

Vérifions la formule pour la variable X_1 . Nous avons $n = 100$; $b = 9.022$; $a = 2.654$; $\sigma = 1.484$. Nous obtenons

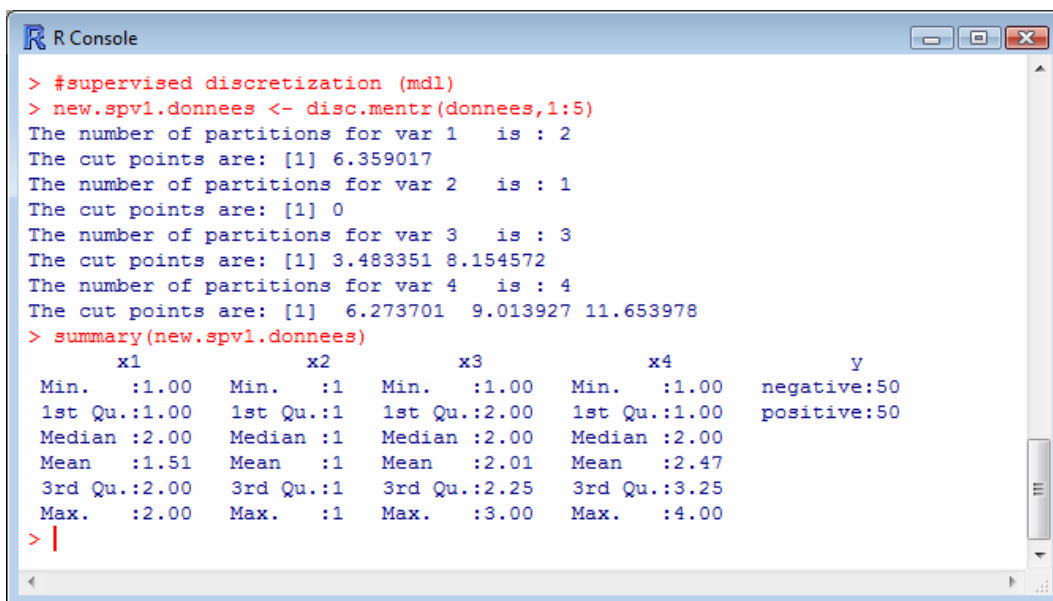
$$\frac{b - a}{3.5 \times \sigma \times n^{(-1/3)}} = 5.69$$

Qui a été arrondi à 6 semble-t-il.

Discrétisation supervisée avec MDLPC. Même si la documentation n'y fait pas explicitement référence, il semble que la technique descendante supervisée s'appuie sur l'algorithme de Fayyad et Irani (1993). Le gain d'entropie est utilisé pour déterminer les bornes de discrétisation. La règle d'arrêt est basée sur la théorie de la description minimale des messages.

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

Nous obtenons à la fois la description des bornes et les données recodées dans un nouveau data.frame.



```
R Console
> #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
> |
```

Discrétisation supervisée avec CHI-MERGE. Le package « dprep » intègre une autre technique supervisée, il s'agit de la méthode chi-merge (Kerber, 1992). C'est une méthode hiérarchique ascendante. Schématiquement, elle essaie de trouver les deux intervalles adjacents dont le profil, au regard de la variable cible, est le plus proche. Elle effectue alors un test d'équivalence distributionnelle. Si la p-value du test est plus grand qu'un seuil prédéfini « alpha » (risque du test), les deux intervalles sont fusionnés. Le rôle du seuil est considérable. Lorsque nous le diminuons, nous favorisons la fusion, nous obtiendrons peu d'intervalles. Dans le cas contraire, lorsque nous l'augmentons, nous favorisons l'éclatement, nous obtiendrons plus d'intervalles. En pratique, trouver la bonne valeur de « alpha » est très compliqué.

Nous avons entré le code suivant dans R :

```
#supervised discretization (chi-merge)
```

```
new.spv2.donnees <- chiMerge(donnees,1:4,alpha=0.1)
summary(new.spv2.donnees)
```

La variable X2, celle qui pose le plus problème, est codée en 35 (!) intervalles ; la variable X4, elle, est codée en 9 intervalles.

```
R Console
> #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
> |
```

Nous essayons de réduire le seuil pour favoriser la fusion. Pour $\alpha = 0.01$, la situation pour X2 est améliorée, mais pas résolue ; nous avons encore trop d'intervalles pour X3 (5 au lieu de 3) et X4 (7 au lieu de 4).

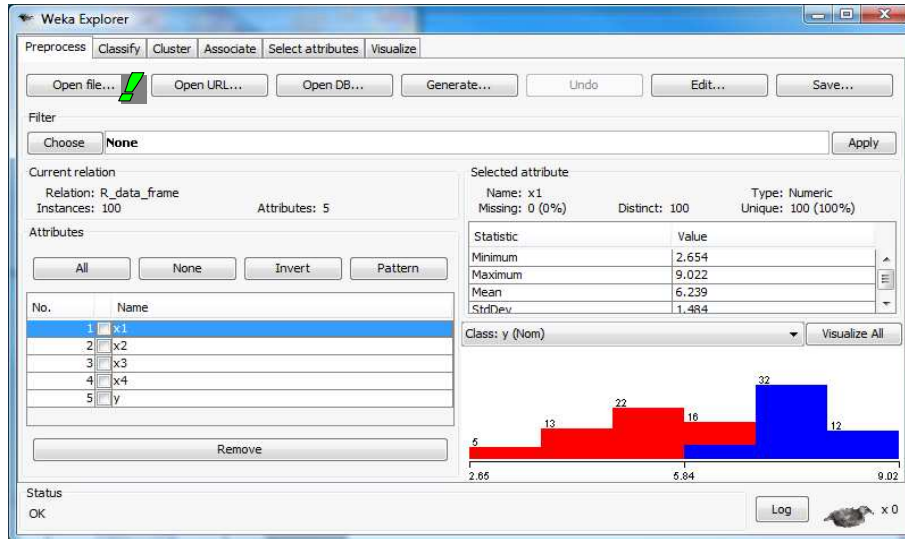
```
R Console
> #supervised discretization (chi-merge) - alpha = 0.01
> new.spv2.donnees <- chiMerge(donnees,1:4,alpha=0.01)
> summary(new.spv2.donnees)
      x1      x2      x3      x4      y
Min.  :1.0  Min.  :1.00 Min.  :1.00 Min.  :1.00 negative:50
1st Qu.:1.0  1st Qu.:1.00 1st Qu.:3.00 1st Qu.:3.00 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
> |
```

Les tableaux de contingence entre la variable cible et les descripteurs recodés (X3 ; X4) montrent que des fusions supplémentaires auraient été le bienvenu.

```
R Console
> 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
>
```

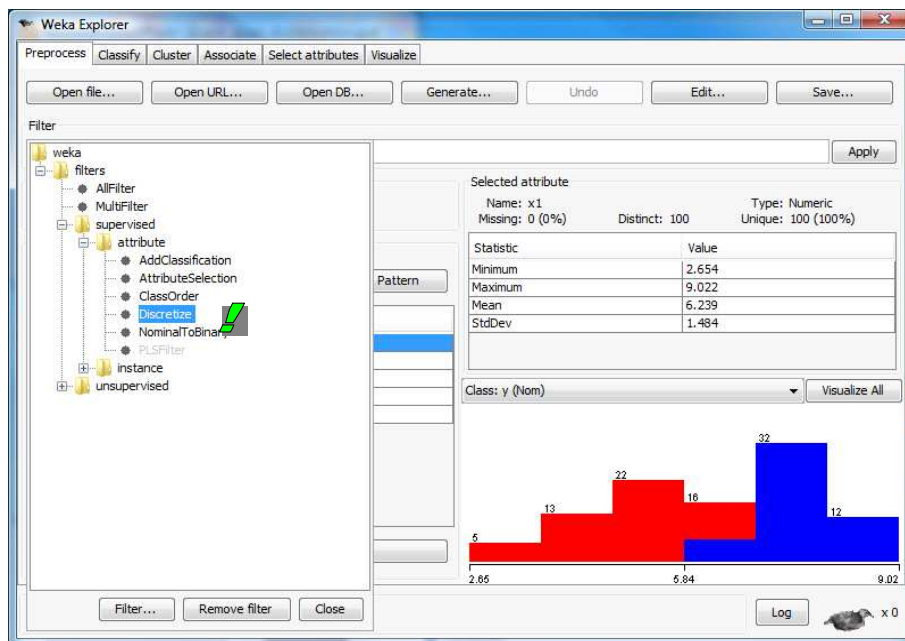
6 Discrétisation avec Weka

Nous utilisons Weka (<http://www.cs.waikato.ac.nz/ml/weka/>) en mode EXPLORER. Dans l'onglet PREPROCESS, nous chargeons le fichier en actionnant le bouton OPEN FILE.

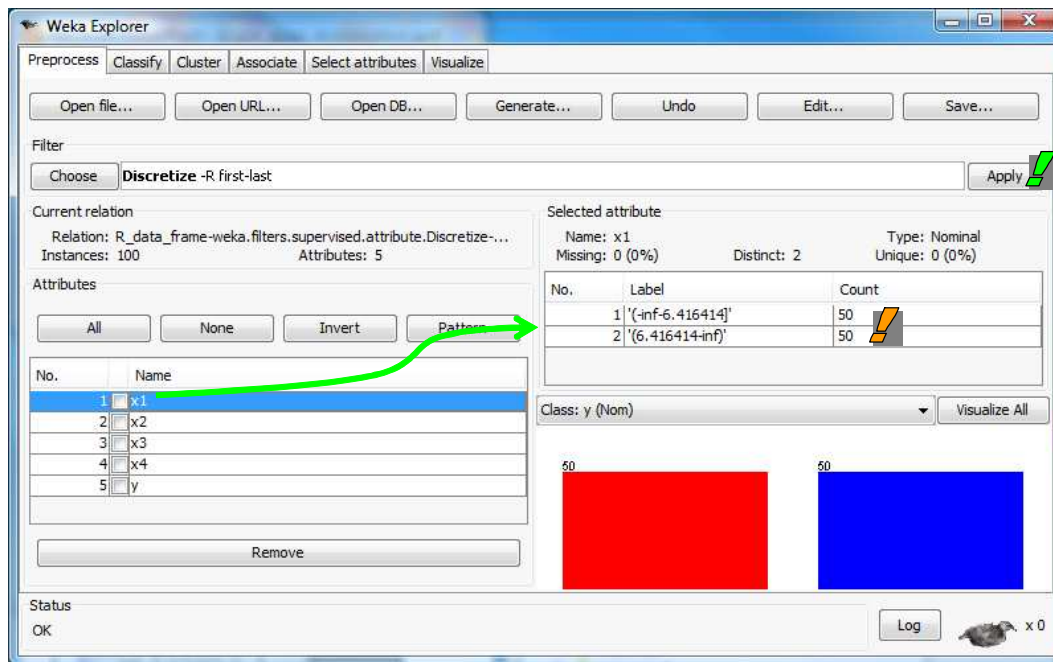


Weka considère que la dernière colonne du fichier correspond à la variable cible. Ce qui est le cas en ce qui nous concerne. Il affiche alors la distribution des autres variables conditionnellement aux classes. Nous avons dans la copie d'écran ci-dessus, la distribution de X1 où l'on observe le décalage entre les positifs (rouge) et les négatifs (bleu).

Pour accéder aux outils de transformation de variables, nous actionnons le bouton CHOOSE de la section FILTER. Nous choisissons l'option DISCRETIZE dans la branche SUPERVISED.



L'outil repose sur l'algorithme MDLPC de Fayyad et Irani (1993). Nous cliquons sur le bouton APPLY, toutes les variables continues (FIRST → LAST) sont découpées en fonction de Y.

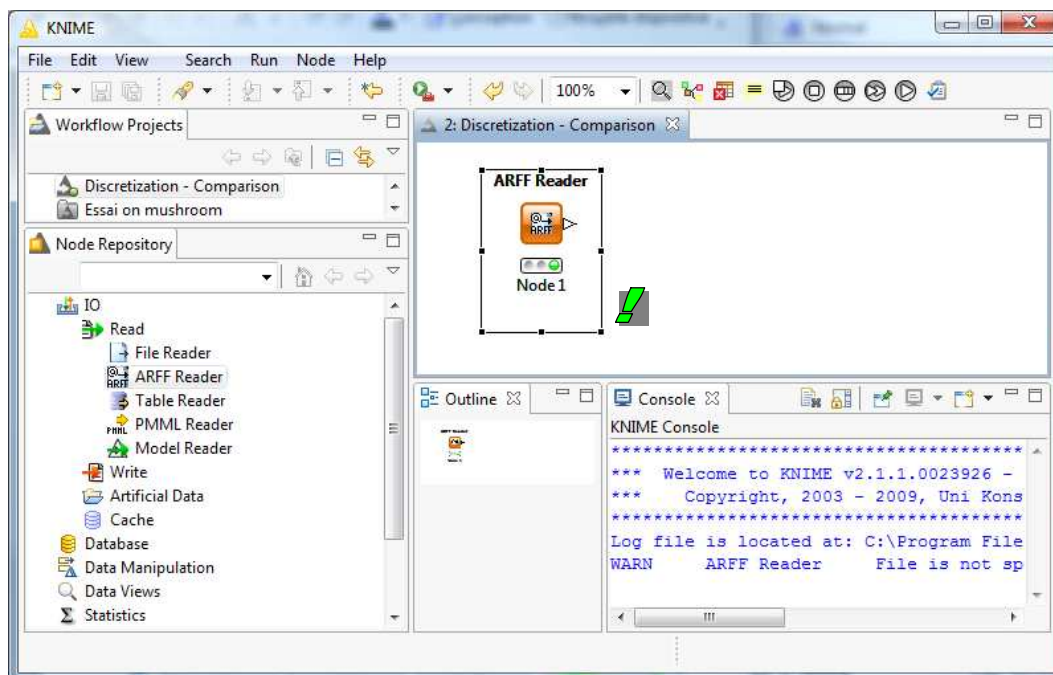


Les variables discrétisées remplacent les précédentes. Nous disposons des bornes de discrétisation et des effectifs par intervalles. Dans la partie en bas à droite, nous avons les distributions conditionnellement aux classes. Par exemple, 50 observations correspondent à $X_1 \leq 6.416414$, ils sont tous positifs.

Les résultats correspondent en tout point à ceux de Tanagra. Ce n'est pas étonnant, ils s'appuient sur les mêmes algorithmes.

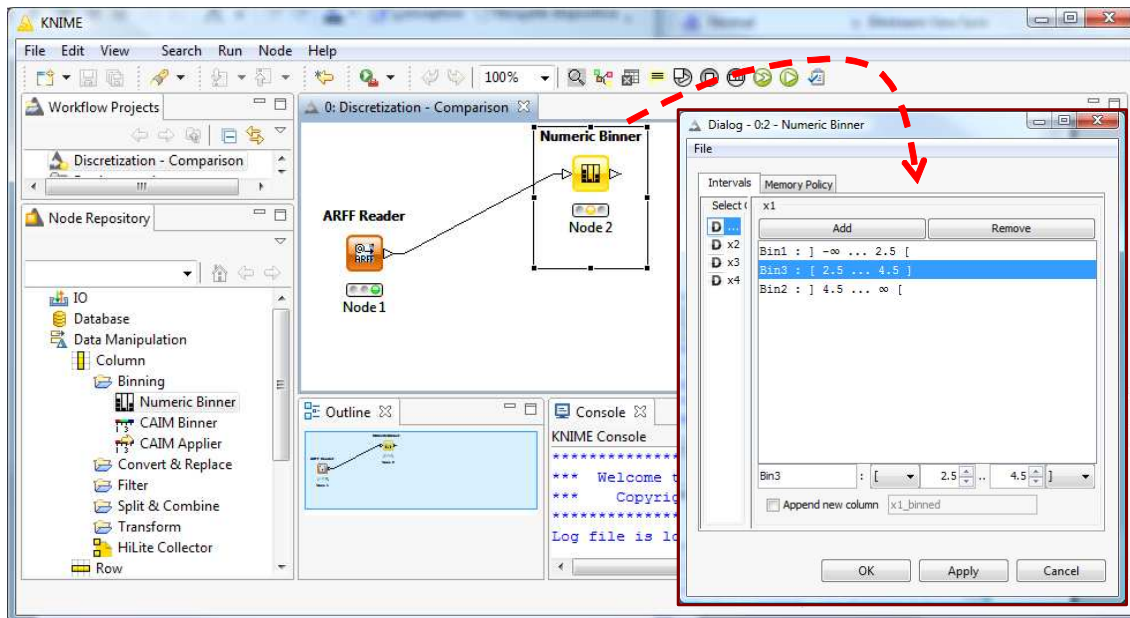
7 Discrétisation avec Knime

Nous créons un nouveau « Workflow » après avoir démarré Knime (<http://www.knime.org/>).

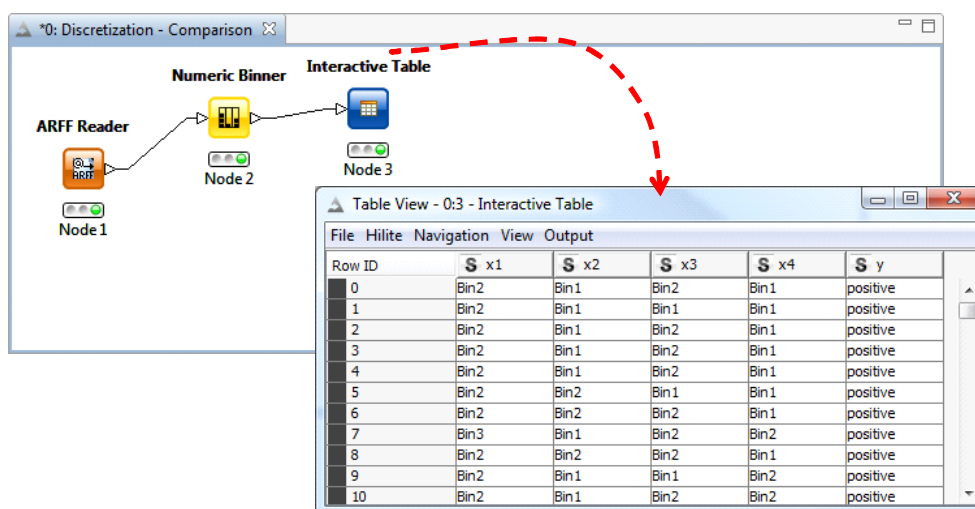


Nous accédons aux données avec le composant ARFF READER. Le menu contextuel CONFIGURE permet de sélectionner le fichier, EXECUTE de le charger.

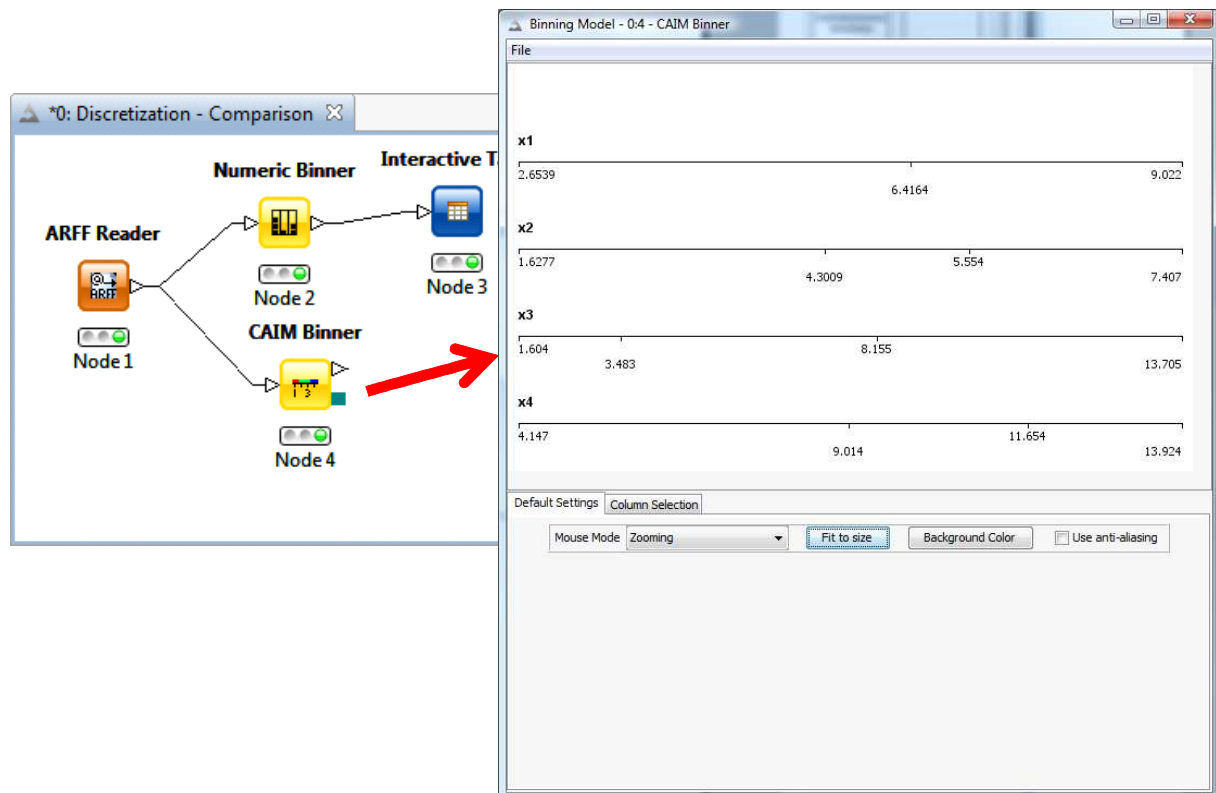
Discrétisation manuelle. Le premier outil que nous présentons est la discrétisation interactive. Un tel outil existe dans Sipina disions nous, mais force est de constater qu'il est particulièrement bien réalisé dans Knime. Nous insérons le composant NUMERIC BINNER dans le workflow, nous le paramétrons en actionnant le menu CONFIGURE.



Pour chaque variable, nous pouvons définir le nombre d'intervalles et les seuils. Les variables transformées sont disponibles en aval du composant lorsque nous avons validé le paramétrage et que nous actionnons le menu contextuel EXECUTE. Nous pouvons les explorer avec les composants de fouille de données. En ce qui nous concerne, nous nous contentons de les visualiser à l'aide de l'outil INTERACTIVE TABLE dans la copie d'écran ci-dessous.



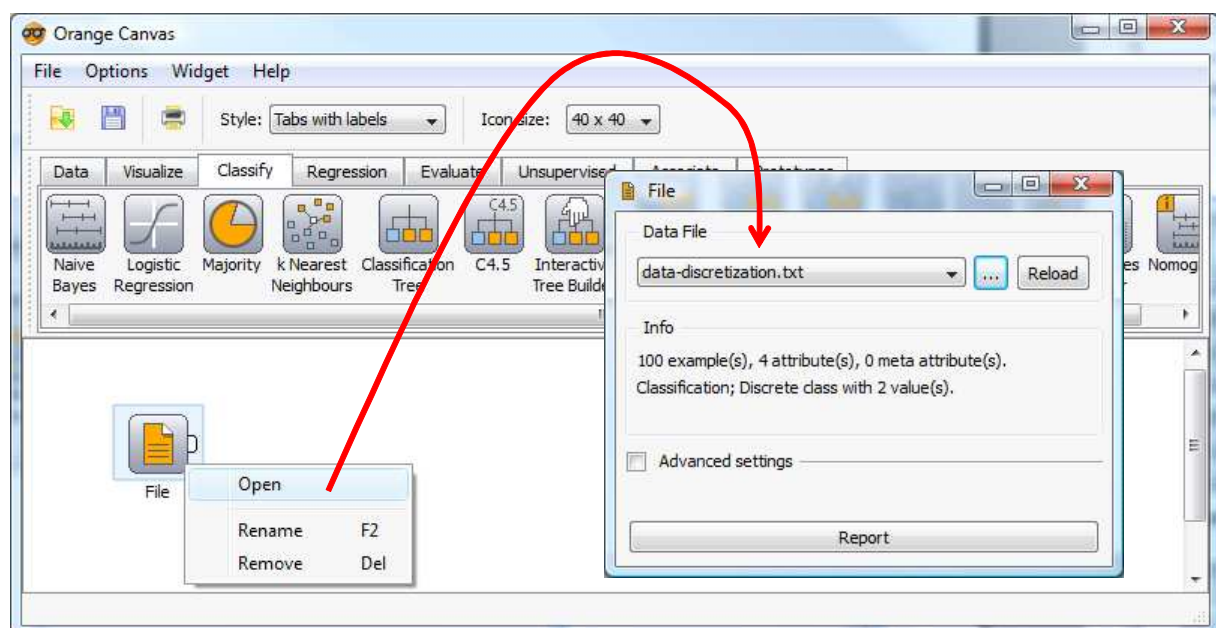
Discrétisation supervisée. Knime propose un outil de discrétisation supervisée avec le composant CAIM BINNER. Il repose sur l'algorithme de Kurgan et Cios (2004; <http://citeseer.ist.psu.edu/kurgan04caim.html>). Voyons ce que ça donne sur nos données.



Par rapport à MDLPC qui a trouvé les bonnes solutions pour chaque variable, CAIM propose, à tort, 3 intervalles pour la variable X_2 (au lieu de 0) et 3 pour X_4 (au lieu de 4). Mais ce n'est qu'un exemple parmi tant d'autres. Il faudrait réaliser des expérimentations à grande échelle pour cerner le comportement de l'algorithme.

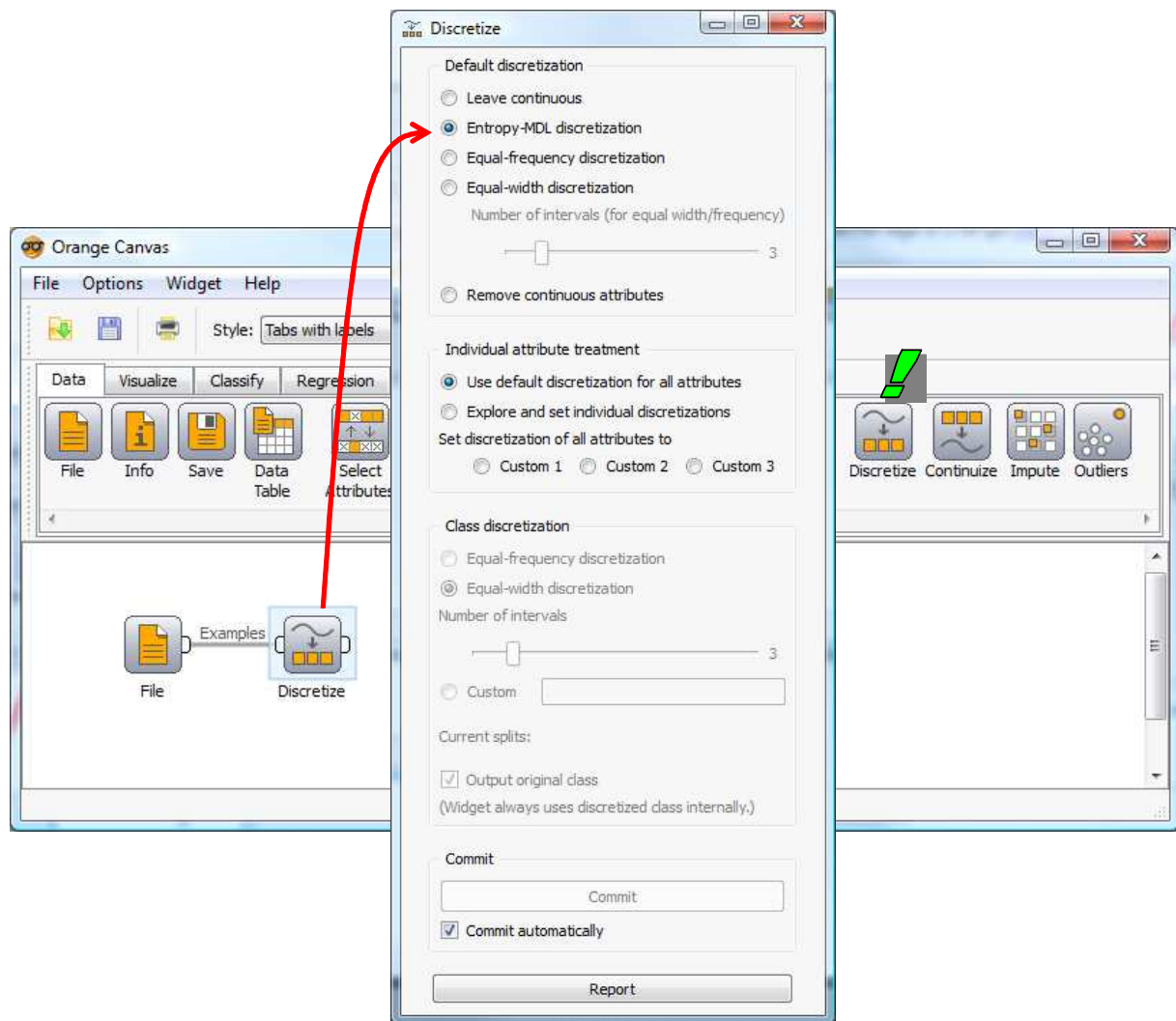
8 Discrétisation avec Orange

Au démarrage d'Orange (<http://www.aillab.si/orange/>), nous chargeons le fichier avec l'outil FILE.

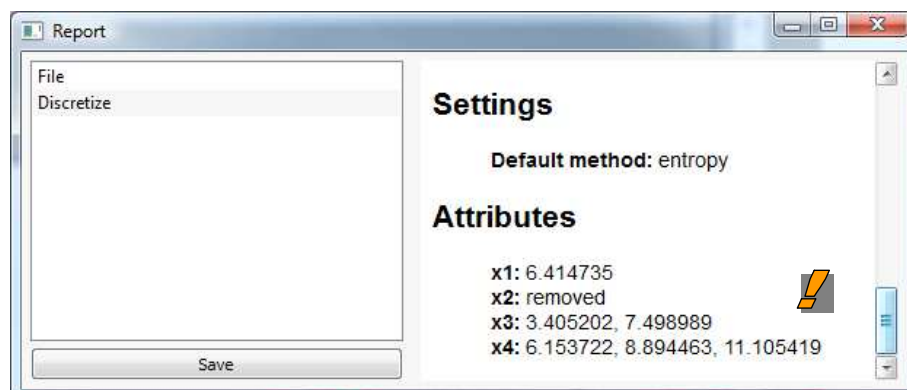


Remarque : Pour une raison qui m'échappe totalement, Orange n'a pas voulu charger le fichier au format ARFF, j'ai donc du me rabattre sur le format texte avec séparateur tabulation.

Discrétisation. L'outil DISCRETIZE (onglet DATA) permet de définir la discrétisation. On se rend compte de l'extrême richesse de l'outil lorsque nous voulons le paramétrer. Nous nous en tiendrons à l'option ENTROPY-MDL DISCRETIZATION qui correspond à l'algorithme de Fayyad et Irani (http://www.ailab.si/orange/doc/ofb/o_categorization.htm).

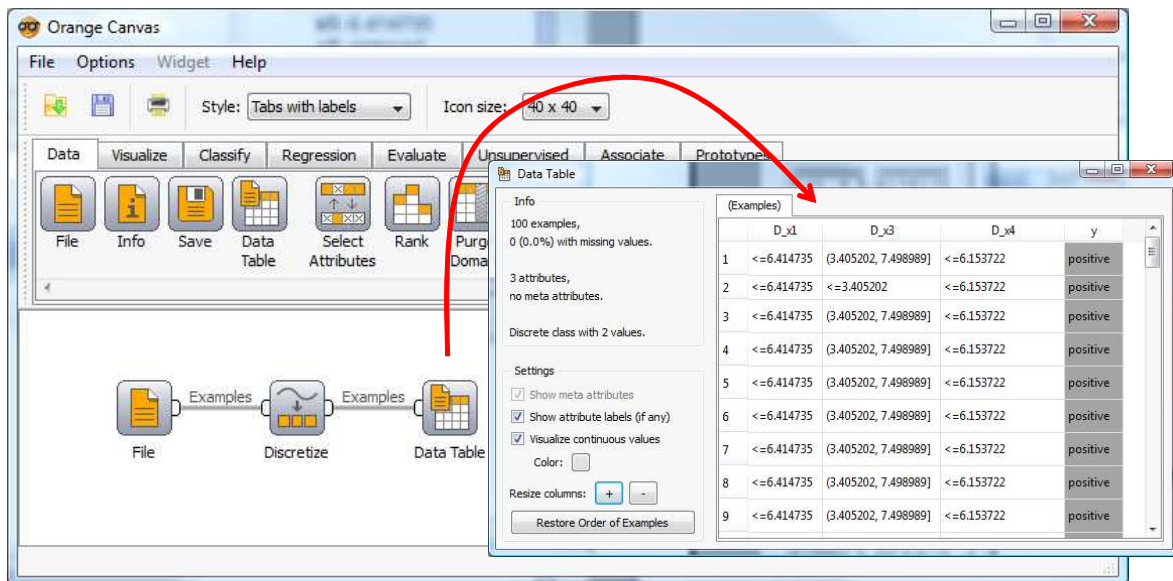


Nous obtenons les bornes lorsque nous actionnons le bouton REPORT.



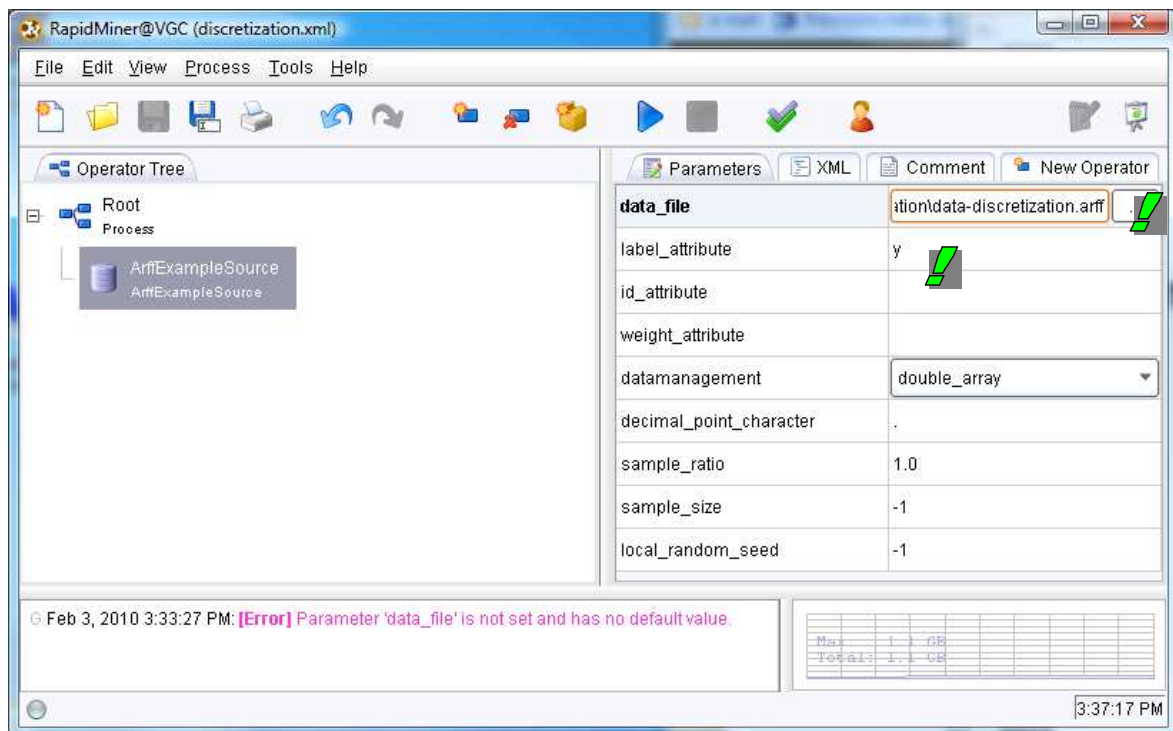
Les intervalles sont les mêmes que ceux de Weka et Tanagra. Les bornes en revanche sont très légèrement différentes. Cela n'a aucune incidence sur la qualité de la modélisation subséquente.

Bien entendu, les variables recodées sont disponibles pour les outils placés à la suite de DISCRETIZE.



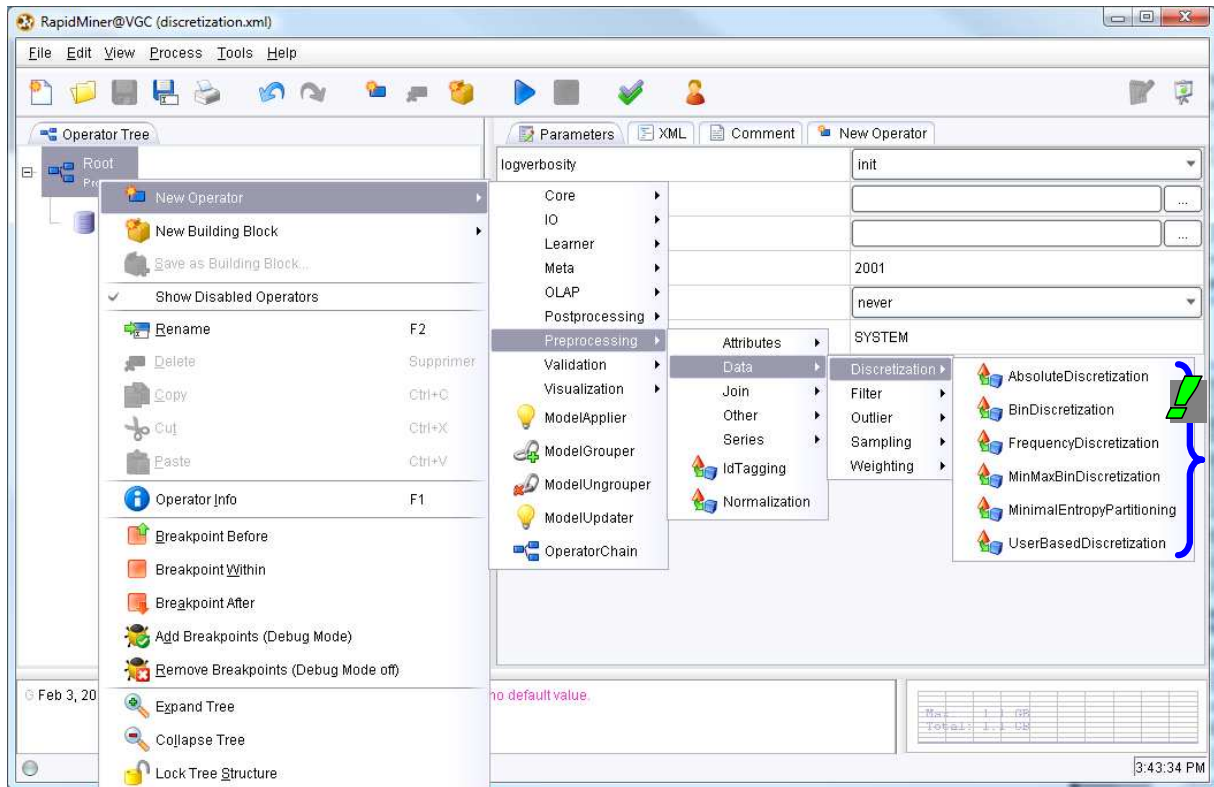
9 Discrétisation avec RapidMiner

Au démarrage de RapidMiner (<http://rapid-i.com/content/view/181/196/>), nous créons un nouveau diagramme (FILE / NEW) et nous insérons le composant d'accès ARFF EXAMPLES SOURCE.

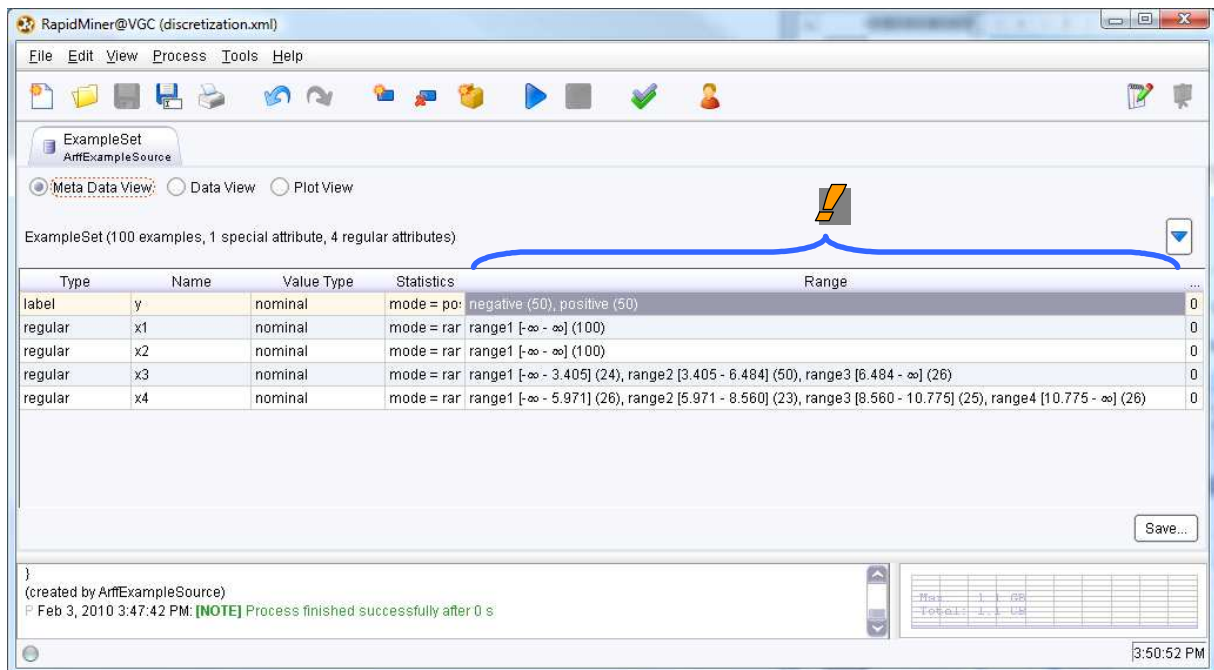


Nous spécifions le nom du fichier **data-discretization.arff** et la variable cible **y**.

Il faut chercher dans les dédales de menus pour trouver les outils de discrétisation.



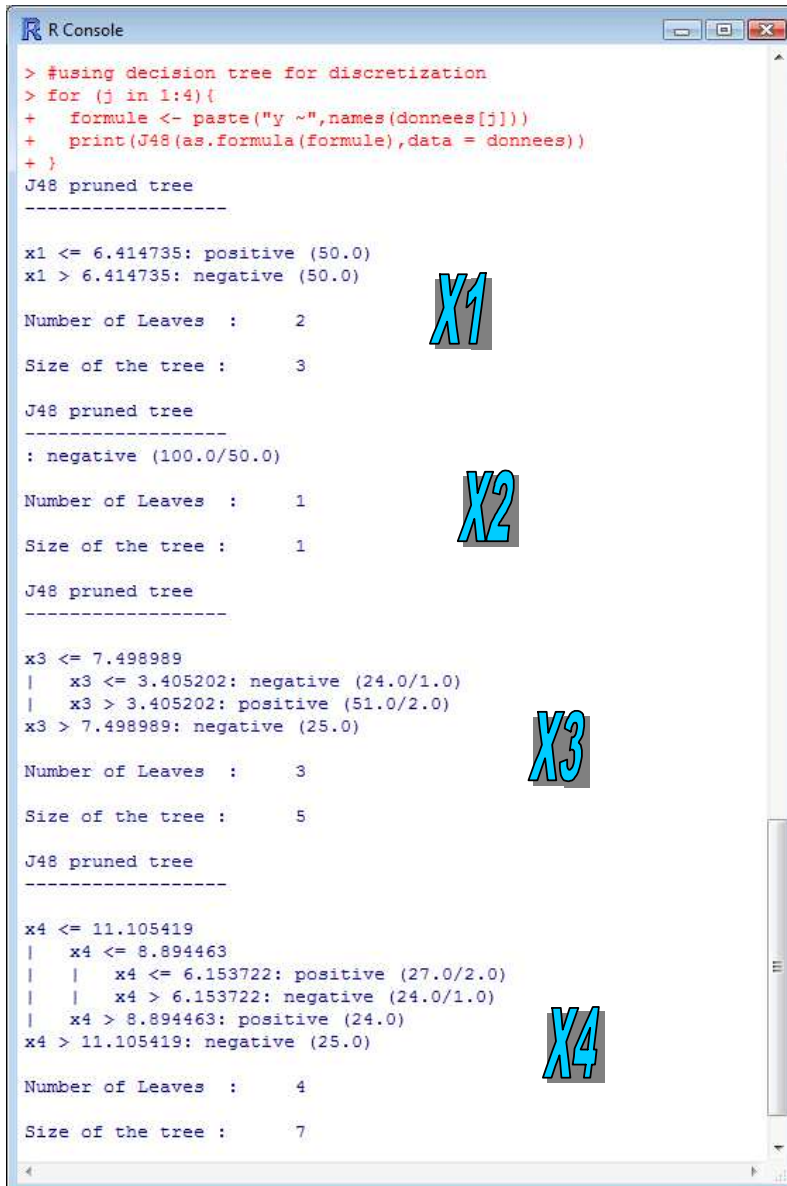
Nous choisissons le composant MINIMAL ENTROPY PARTITIONNING qui cite à la fois l'article de Fayyad et Irani (1993) et de Dougherty et al. (1995). Nous cliquons sur le bouton RUN (bleu) dans la barre d'outils.



RapidMiner fournit plusieurs indications. Nous nous intéressons plus particulièrement à la colonne RANGE où sont décrits les intervalles pour chaque variable. Nous constatons que le logiciel a refusé de découper (à tort) X1 et (à juste titre) X2. Concernant X3 et X4, le nombre d'intervalles correspond à ceux de Weka et Tanagra, certaines bornes sont relativement différentes en revanche.

10 Conclusion

Le découpage en classes d'une variable continue est une opération courante dans l'analyse exploratoire des données. Si les méthodes non supervisées sont assez bien connues des praticiens, il en est autrement des approches supervisées. Pourtant elles existent, et elles fonctionnent plutôt bien dans la grande majorité des cas.



```

R Console
> #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

```

Enfin, il existe une alternative très simple pour découper en classes une variable continue en fonction d'une variable cible qualitative : la construction d'un arbre de décision en ne précisant qu'une seule variable prédictive, celle que l'on souhaite discrétiser. A bien y regarder, on se rend compte d'ailleurs que l'algorithme descendant MDLPC (Fayyad et Irani, 1993) est assimilable à la construction d'un arbre de décision.

Nous avons voulu vérifier cette idée en lançant l'algorithme J48 du package RWeka pour chaque variable de notre fichier de données. Une boucle dans R fait parfaitement l'affaire.

Les résultats correspondent à ceux de MDLPC, tout du moins en ce qui concerne le nombre d'intervalles (nombre de feuilles pour l'arbre).

Les bornes sont très légèrement différentes. Cela s'explique

essentiellement par le fait que critère utilisé lors de la construction de l'arbre est le « gain ratio » de $C_{4.5}$ (Quinlan, 1993) et non pas le simple « gain d'entropie » (MDLPC).

En tous les cas, nous avons obtenu avec l'arbre des solutions qui sont parfaitement viables.