

# 1 Objectif

## Induction de règles floues avec le logiciel KNIME.

Ce didacticiel fait suite à celui consacré à l'induction des règles prédictives (<http://tutoriels-data-mining.blogspot.com/2009/11/induction-de-regles-predictives.html>). Je n'avais pas intégré Knime dans le comparatif car il proposait une technique que je ne connaissais pas bien, l'induction de règles floues, et demandait une préparation particulière de variables qui me paraissait bien étrange. Il fallait notamment que l'attribut cible soit numérique, ce qui paraît assez incongru dans le cadre de l'apprentissage supervisé. Comme il me fallait avancer, j'ai préféré reporter l'étude du logiciel Knime à plus tard (c.-à-d. maintenant) en lui dédiant spécifiquement un didacticiel.

Parmi les logiciels libres (ou accessibles gratuitement) fonctionnant sous forme de diagramme de traitements, Knime est certainement l'un des plus prometteurs, un des rares à pouvoir tailler des croupières aux équivalents commerciaux. Il y a dans ce logiciel une rigueur de conception et un souci du détail qui ne laisse pas indifférent : il est par exemple possible, devant la multitude d'outils disponibles, de créer une section des méthodes favorites ; une autre palette permet également de retrouver les composants les plus fréquemment utilisés ; la documentation est accessible de manière permanente dans la partie droite de la fenêtre principale ; etc. C'est aussi un des seuls logiciels libres à faire des efforts particuliers pour ce qui est de l'accès aux bases de données et la préparation des variables. J'avoue que je prends un réel plaisir à l'utiliser et à l'étudier de manière approfondie.

Concernant l'induction de règles prédictives, Knime (version 2.1.1) implémente l'induction des règles floues. Les articles présentant la méthode sont accessibles en ligne<sup>1,2</sup>. Les lecteurs intéressés par les fondements théoriques de la méthode pourront s'y reporter. Pour ma part, dans ce didacticiel, je m'attacherais avant tout à décrire la mise en œuvre de la méthode en détaillant le pourquoi et comment de la préparation des variables préalable à l'induction, et le mode de lecture du modèle prédictif. Pour avoir un point de repère, nous comparerons les résultats avec ceux fournis par la méthode d'induction de règles proposée par Tanagra.

## 2 Données

Nous utilisons une version à deux dimensions du fichier IRIS<sup>3</sup> dans ce didacticiel. Nous nous appuyons sur les variables PET\_LENGTH et PET\_WIDTH pour prédire le groupe d'appartenance. L'intérêt est de pouvoir représenter graphiquement les caractéristiques des données et celles de la base de règles dans l'espace de représentation.

---

<sup>1</sup> M.R. Berthold, « Mixed fuzzy rule formation », International Journal of Approximate Reasoning, 32, pp. 67-84, 2003.

[http://www.inf.uni-konstanz.de/bi/ml2/publications/Papers2003/Berto3\\_mixedFR\\_ijar.pdf](http://www.inf.uni-konstanz.de/bi/ml2/publications/Papers2003/Berto3_mixedFR_ijar.pdf)

<sup>2</sup> T.R. Gabriel, M.R. Berthold, « Influence of fuzzy norms and other heuristics on mixed fuzzy rule formation », International Journal of Approximate Reasoning, 35, pp.195-202, 2004.

[http://www.inf.uni-konstanz.de/bi/ml2/publications/Papers2004/GaBeo4\\_mixedFRappendix\\_ijar.pdf](http://www.inf.uni-konstanz.de/bi/ml2/publications/Papers2004/GaBeo4_mixedFRappendix_ijar.pdf)

<sup>3</sup> <http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/iris2D.txt>

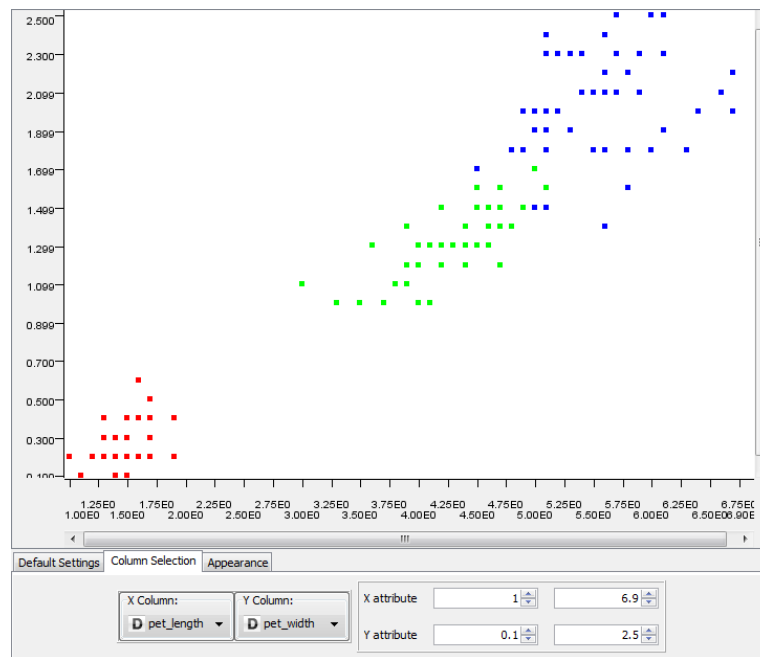


Figure 1 - Fichier Iris - Positionnement des groupes dans le plan (pet.length ; pet.width)

Nous distinguons bien les 3 groupes d'iris, il sera aisé de représenter les frontières induites par le modèle prédictif issu de l'apprentissage supervisé.

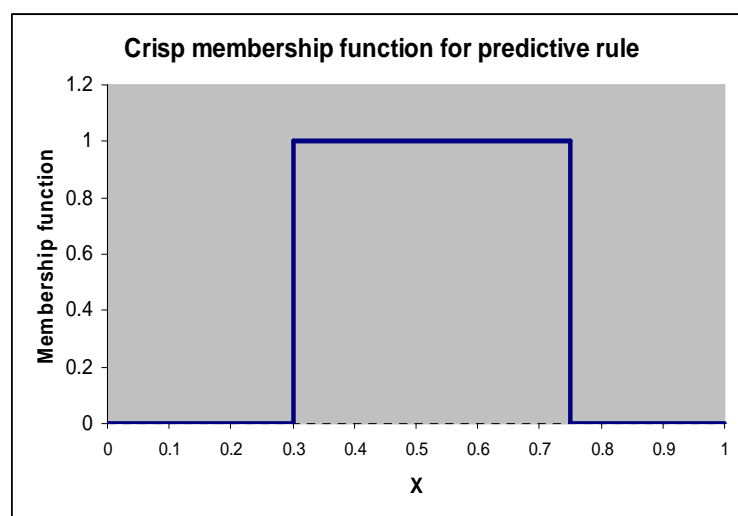
### 3 Induction de règles floues avec Knime

En schématisant, nous dirons que les règles floues introduisent une gradation lors de la définition des régions d'appartenance aux groupes dans l'espace de représentation.

Revenons un instant aux règles usuelles, que l'on appelle règles « crisp » par opposition aux règles floues. Le modèle

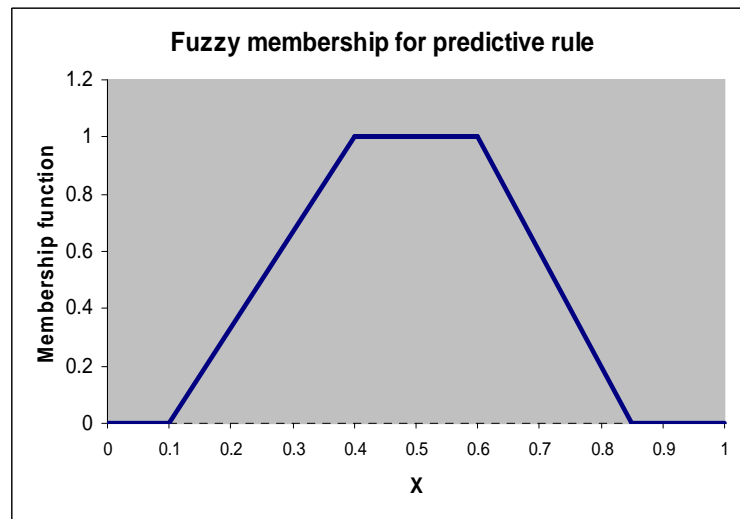
$$\text{Si } (X \geq 0.3) \text{ et } (X < 0.5) \text{ Alors } Y = +$$

Peut être traduite graphiquement de la manière suivante



Pour un individu  $\omega$  donné, si  $X(\omega) \in [0.3 ; 0.5[$ , on lui attribue un degré d'appartenance  $\mu(\omega) = 1$  ; s'il est en dehors de l'intervalle, on lui attribue un degré d'appartenance nul [ $\mu(\omega) = 0$ ].

Sans rentrer dans des discussions compliquées, il est évident que les seuils « 0.3 » et « 0.5 » paraissent un peu arbitraires. Surtout qu'ils ont été estimés à partir d'un échantillon. Ils sont entachés d'une certaine incertitude. Il paraît plus naturel d'introduire une gradation en définissant la région d'appartenance à partir d'un trapèze que l'on peut résumer par une série de 4 points. Par exemple, pour  $\langle 0.1, 0.4, 0.6, 0.85 \rangle$ , nous obtenons le trapèze suivant



Ainsi :

- Pour un individu avec  $X = 0.5$ , il déclenche pleinement à la règle avec  $\mu = 1$ , on conclut qu'il est « + ».
- Pour un autre avec  $X = 0$ , il ne déclenche pas la règle,  $\mu = 0$ .
- Pour un dernier avec  $X = 0.2$ , il déclenche à la règle à hauteur de  $\mu \approx 0.3$ , ce qui réduit d'autant la portée de la conclusion.

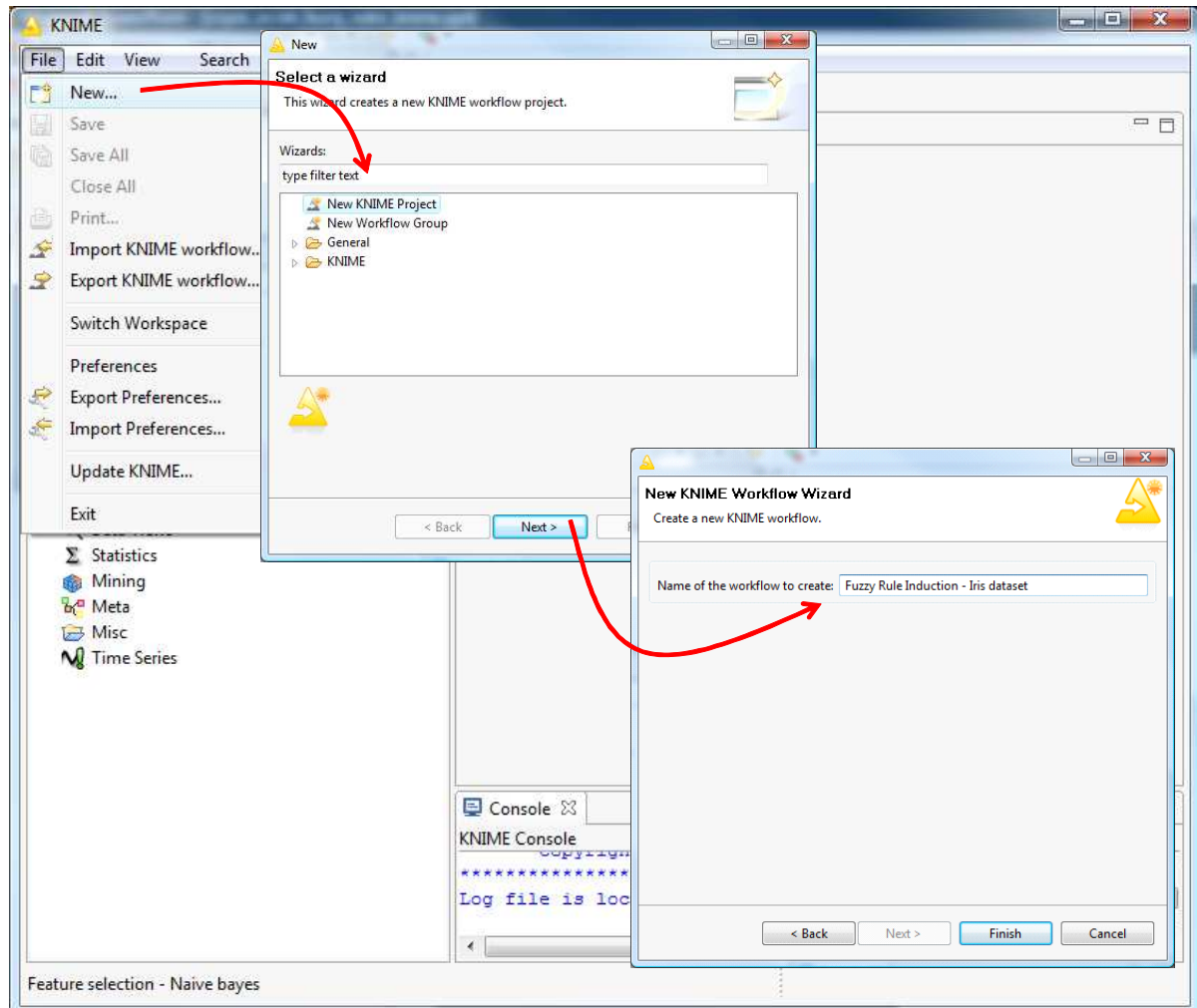
Pour intéressante qu'elle soit, cette nouvelle représentation n'est pas sans poser de problèmes : (1) si l'interprétation de la règle est naturelle, son écriture, notamment son implémentation dans les systèmes d'information, n'est pas aisée ; (2) les régions définies par les règles de la base de connaissances se recouvrent plus ou moins dans l'espace de représentation, le classement d'un nouvel individu déclenchera plusieurs règles, à des degrés divers, il faut mettre en place une stratégie de gestion des conflits.

Enfin, deux remarques pour conclure cette très courte présentation : (1) la notion de degré d'appartenance ne se limite pas aux seules variables prédictives quantitatives, elle peut être étendue aux descripteurs nominaux, notamment pour la gestion des données manquantes ; (2) pour les férus d'apprentissage automatique, en réduisant la variance, les classifieurs flous améliorent souvent les performances en prédiction par rapport aux méthodes « crisp »<sup>4</sup>.

<sup>4</sup> Voir par exemple : C. Olaru, L. Wehenkel, « Bias – variance tradeoff of soft decision tree », <http://www.montefiore.ulg.ac.be/services/stochastic/pubs/2004/OWo4/OWo4.pdf>

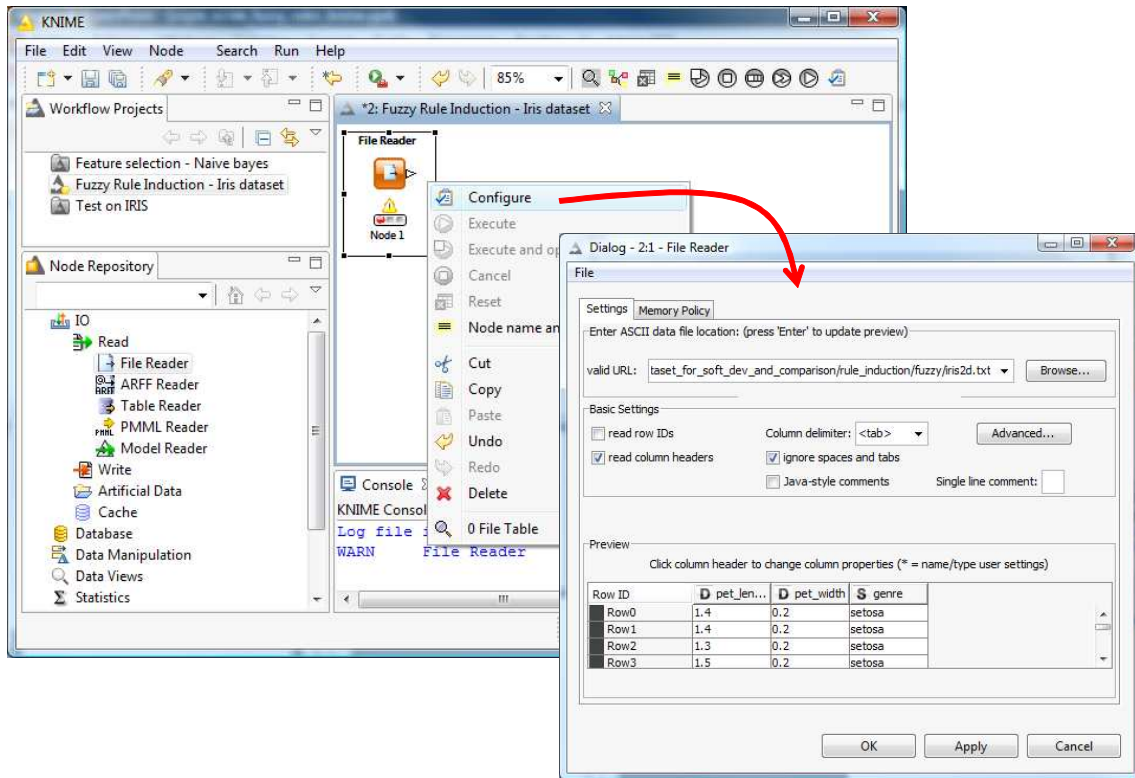
### 3.1 Création d'un diagramme et importation des données

Après avoir démarré le logiciel, nous créons un nouveau projet en actionnant le menu FILE / NEW. Dans la boîte de dialogue qui apparaît, nous demandons un **New KNIME Project**, puis nous cliquons sur le bouton suivant. Nous introduisons le nom du projet « Fuzzy Rule Induction - IRIS dataset ».

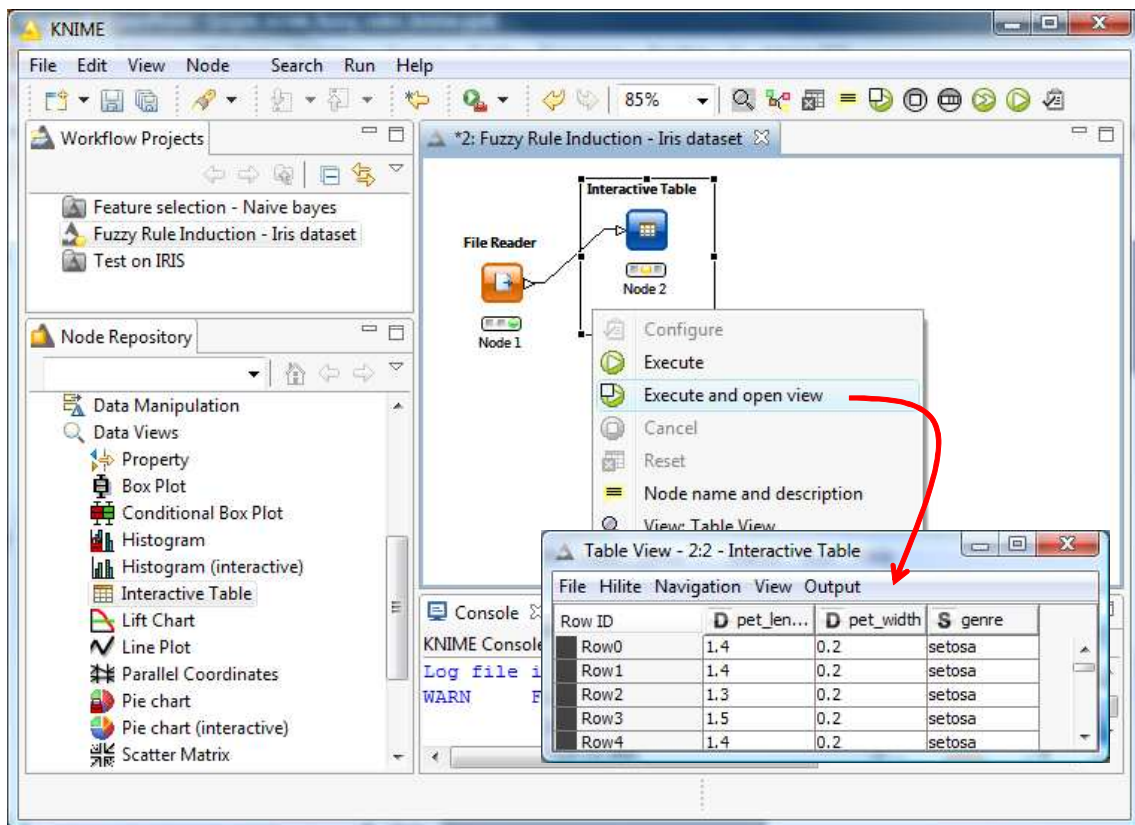


Nous obtenons un nouveau projet dans le **Workflow Projects**.

Pour charger le fichier IRIS2D.TXT, nous insérons le composant **FILE READER** dans l'espace de travail. Nous le paramétrons (menu contextuel CONFIGURE) en sélectionnant le fichier et en spécifiant ses caractéristiques: « tabulation » est le séparateur de colonnes, la première ligne correspond aux noms des variables.



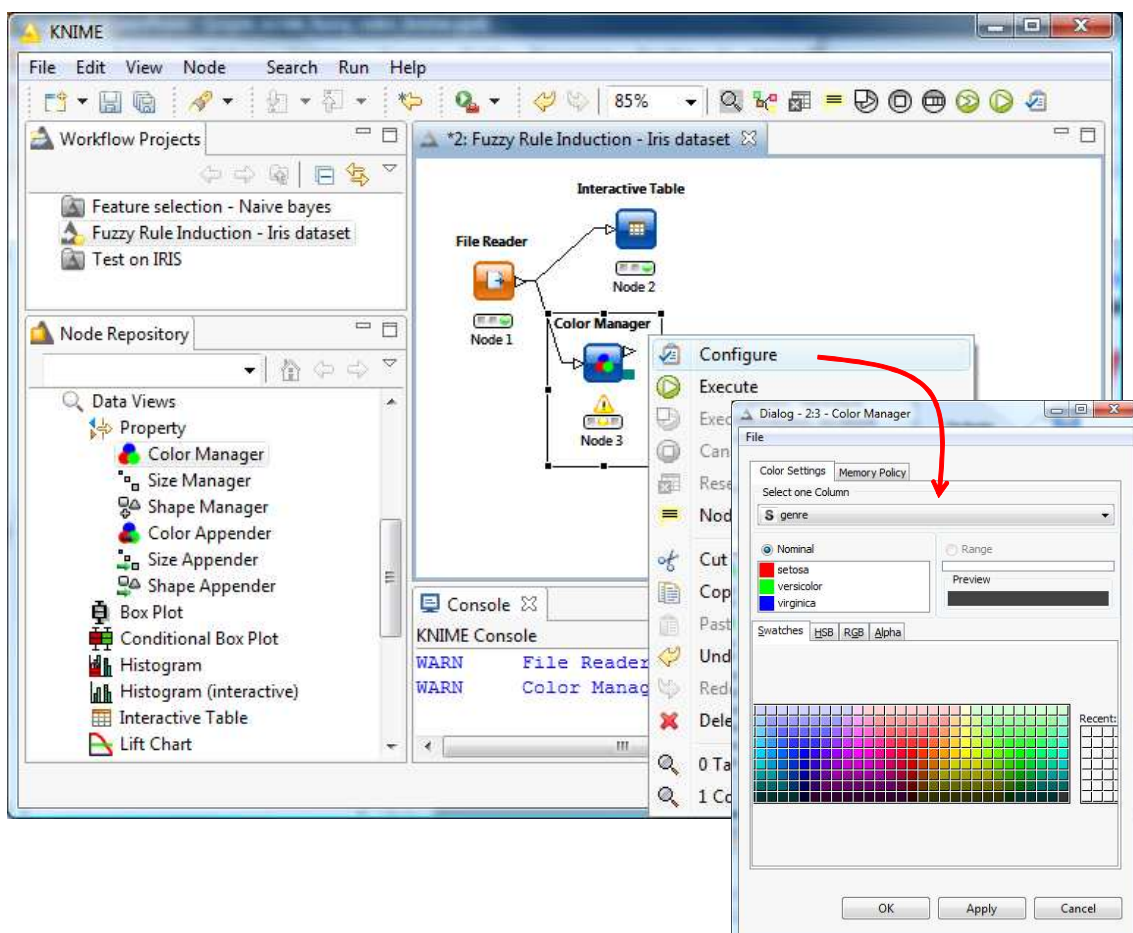
Nous validons ces choix et nous lançons le chargement en cliquant sur le menu contextuel EXECUTE. Pour visualiser le fichier, le plus simple est de lui connecter le composant **INTERACTIVE TABLE**. Avec le menu EXECUTE and OPEN VIEW, nous obtenons une vue des données.



### 3.2 Représentation graphique

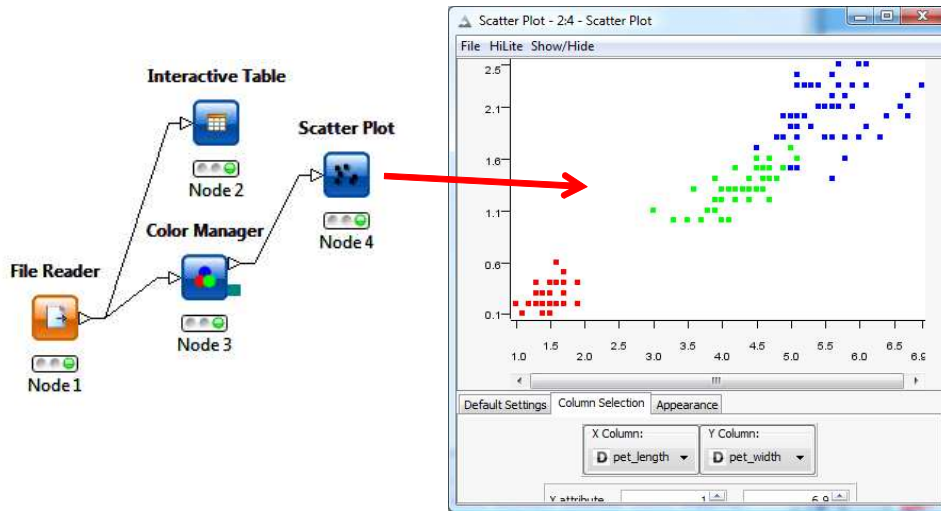
Nous souhaitons reproduire le graphique « nuage de points » présenté en introduction (Figure 1), l'idée est de distinguer les différents groupes de la variable cible dans l'espace de représentation.

Nous introduisons tout d'abord le composant **COLOR MANAGER**, nous lui relierons le composant d'accès aux données, puis nous le paramétrons (menu CONFIGURE) de manière à coloriser les points selon la classe d'appartenance **GENRE**.



Puis nous ajoutons le composant **SCATTER PLOT** (*branche Data Views*) auquel nous connectons **COLOR MANAGER**. Nous actionnons directement le menu **EXECUTE** and **OPEN VIEW** pour visualiser le graphique.

Nous obtenons le graphique présenté dans l'introduction de ce didacticiel. Notons que nous pouvons sélectionner à la volée les variables en abscisse et en ordonnée.

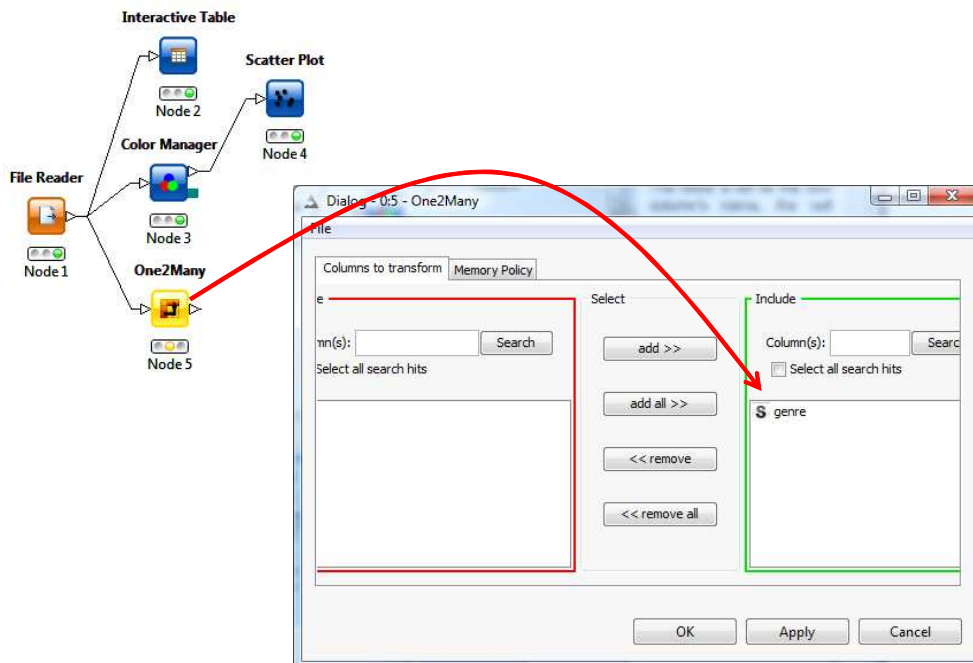


### 3.3 Codage de la variable cible pour l'induction

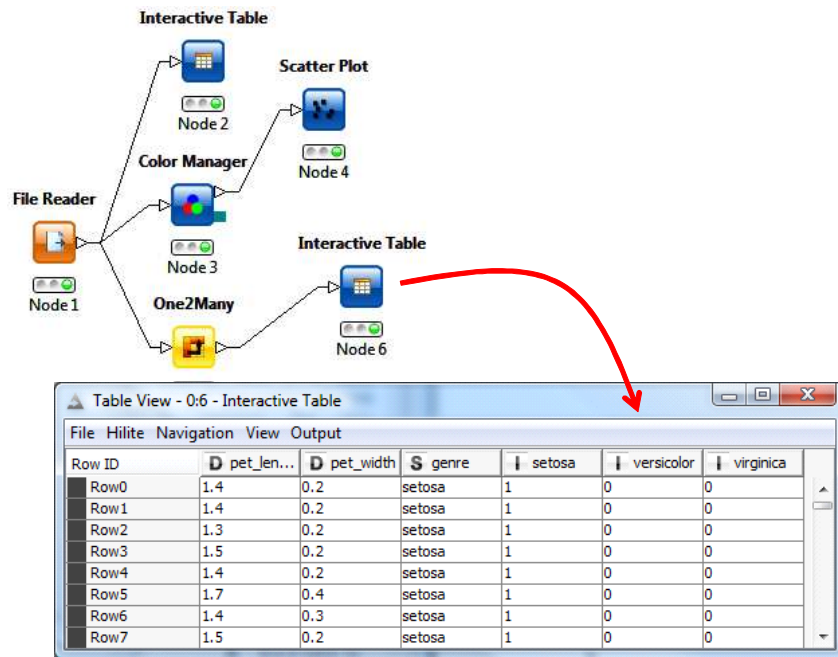
Ici arrive une première particularité de l'approche implémentée dans Knime. En apprentissage supervisé, la variable cible est normalement catégorielle (nominale). Or, Knime attend une ou plusieurs variables numériques, on serait plutôt dans le cadre de la régression ?

Après investigation, je me suis rendu compte que ces variables devaient représenter en réalité le degré d'appartenance à chaque modalité de la variable à prédire. Dans notre cas, puisque les objets appartiennent exclusivement à une des 3 modalités de GENRE (setosa, versicolor et virginica), nous devons produire 3 colonnes de 0/1.

Nous introduisons l'outil de transformation de colonnes **ONE2MANY** (*Data Manipulation / Column / Transform*) dans l'espace de travail. Nous le configurons pour traiter la variable cible GENRE.



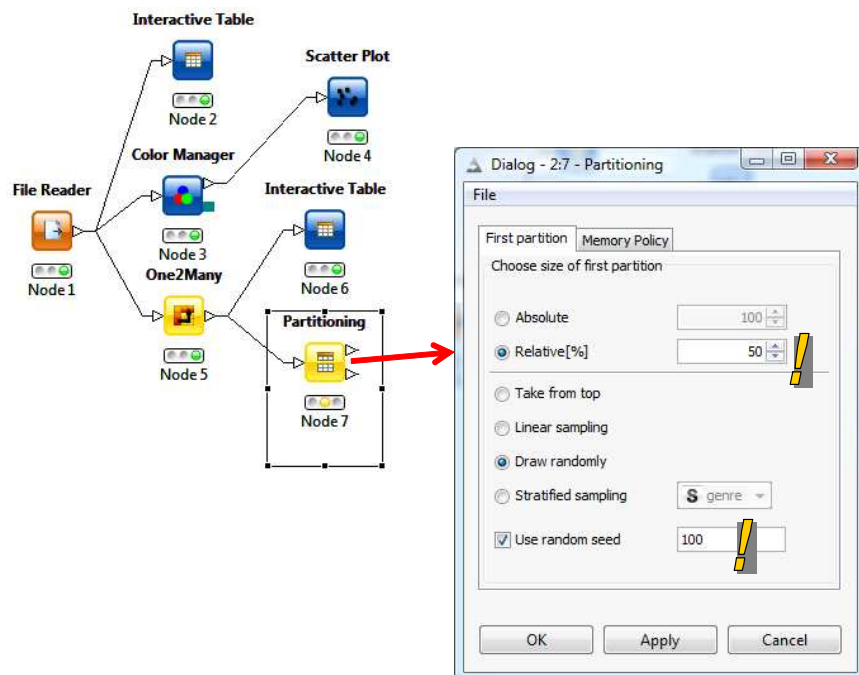
De nouveau, nous connectons un **INTERACTIVE TABLE** (*Data Views*) pour obtenir une vue des données. Les trois nouvelles colonnes sont visibles.



### 3.4 Partitionnement des données en échantillons d'apprentissage et de test

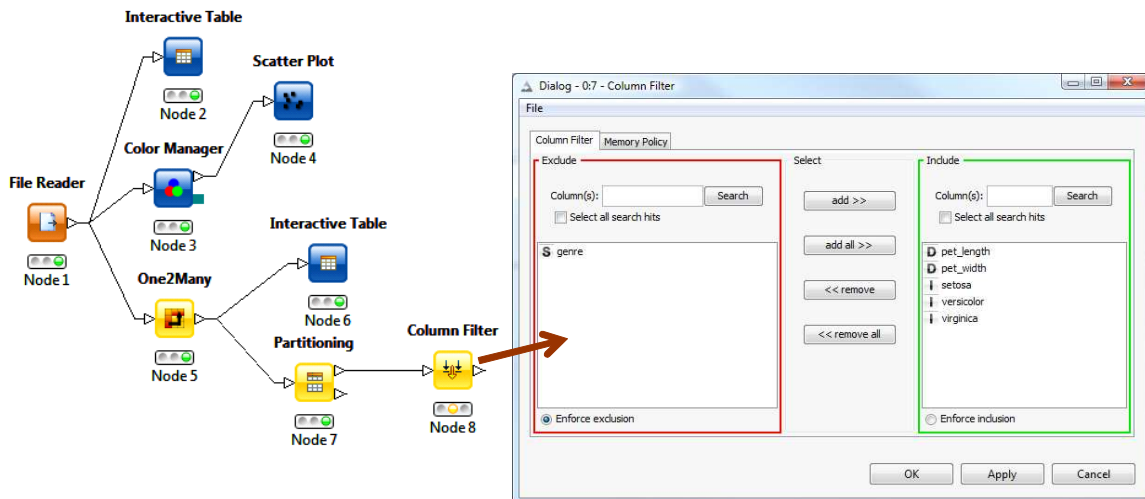
L'usage veut que l'on crée le modèle sur une fraction des données, dite « échantillon d'apprentissage », et que l'on procède à son évaluation sur une autre, dite « échantillon de test ». Les indicateurs de qualité (ex. taux d'erreur, etc.) ainsi calculés ne sont pas biaisés.

Nous utilisons le composant **PARTITIONNING** (branche *DATA MANIPULATION / ROW / TRANSFORM* de la palette des méthodes). Nous lui connectons ONE2MANY et nous le paramétrons en réservant une moitié des observations (RELATIVE = 50%) pour l'apprentissage, l'autre moitié pour le test. Pour que l'expérimentation produise exactement les mêmes résultats sur votre machine, mettez USE RANDOM SEED = 100.



### 3.5 Induction des règles floues

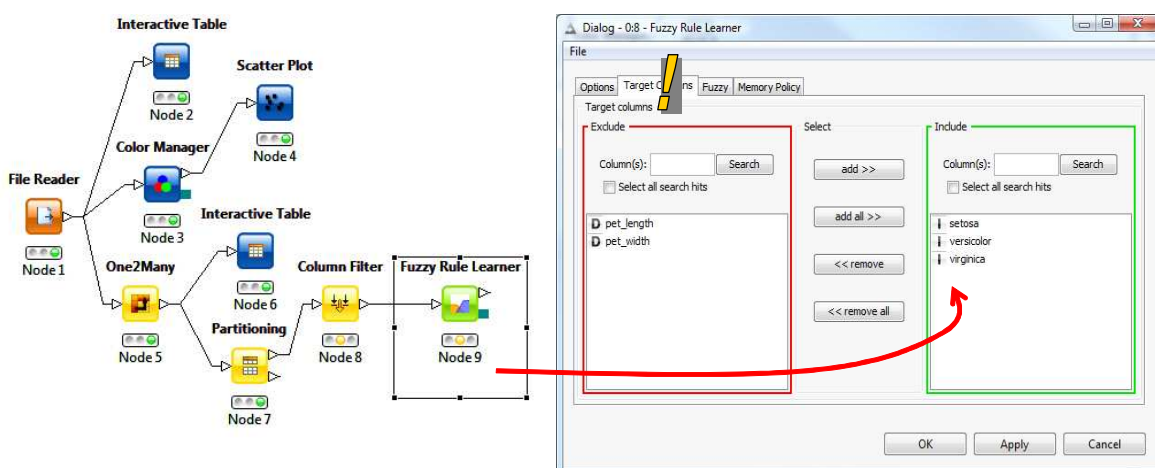
Maintenant, nous devons sélectionner les colonnes qui serviront à l'analyse. Après la création des 3 variables indicatrices ci-dessus, il est clair que la colonne GENRE est redondante, elle ne nous est plus utile. Nous utilisons donc le composant COLUMN FILTER (*Data Manipulation / Column / Filter / Column Filter*) pour spécifier cela.



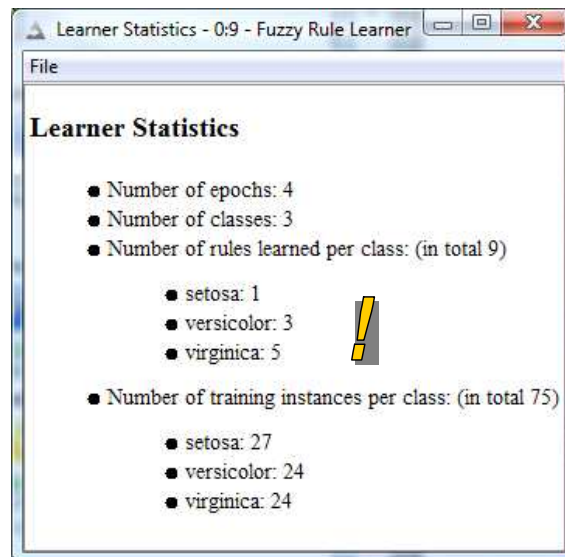
La variable GENRE est exclue du mouvement.

A ce stade, nous sommes prêts à lancer l'apprentissage du modèle prédictif. Nous insérons le composant **FUZZY RULE LEARNER** (*Mining / Rule Induction / Fuzzy Rules*) dans l'espace de travail. Nous lui connectons l'échantillon d'apprentissage issu de PARTITIONNING. Nous actionnons le menu CONFIGURE.

Nous ne nous attardons pas sur les paramètres d'exploration de la méthode (cf. les articles en référence). Le plus important pour nous est d'aller dans l'onglet TARGET COLUMNS. Nous devons y spécifier les colonnes recodées de la variable cible GENRE, à savoir (setosa, versicolor et virginica).

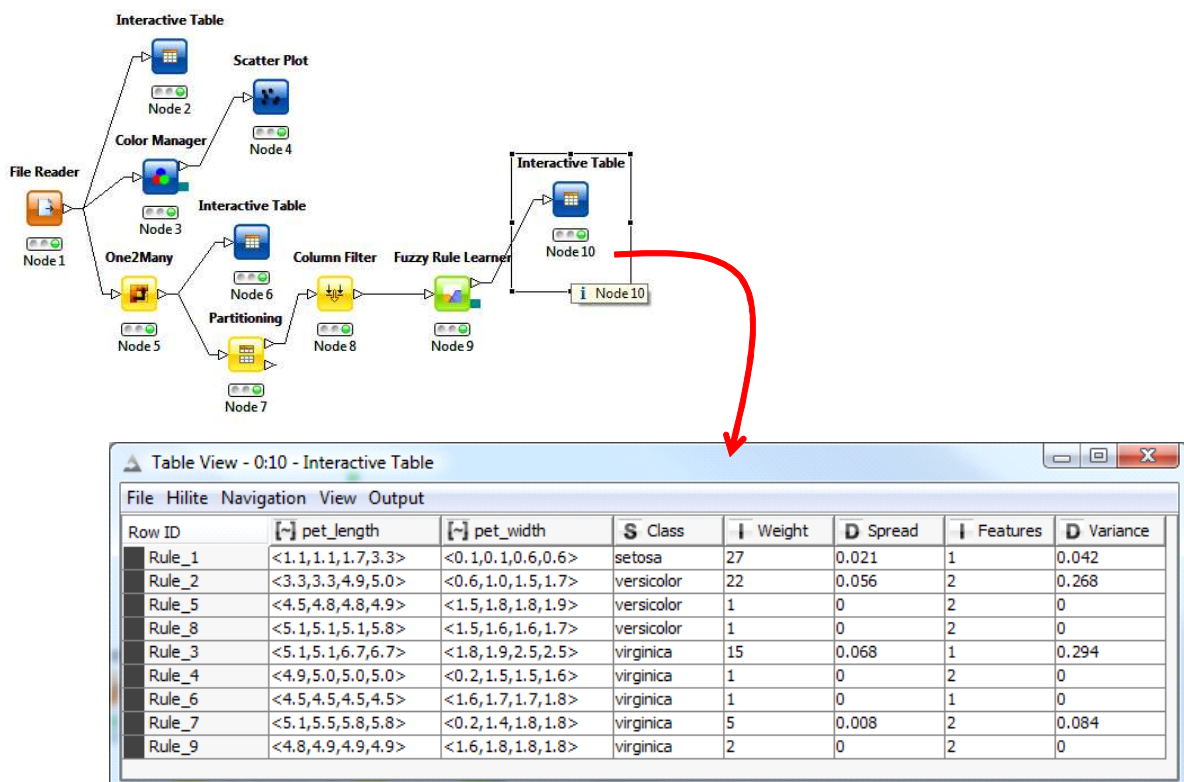


Les variables restantes jouent le rôle de variables explicatives. Nous validons et nous actionnons le menu EXECUTE and OPEN VIEW.



Nous obtenons une série d'informations qu'il faut savoir décrypter : la variable à prédire comporte trois modalités (number of classes) ; la méthode a produit 9 règles, 1 pour la classe « setosa », 3 pour « versicolor » et 5 pour « virginica » ; enfin, nous avons les effectifs par classe dans l'échantillon d'apprentissage.

Pour visualiser les 9 règles, nous ajoutons le composant INTERACTIVE TABLE.



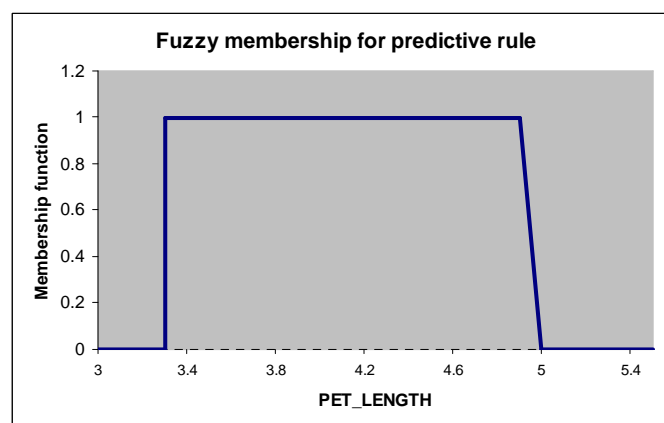
**Remarque 1 :** FUZZY RULE LEARNER n'accepte que les descripteurs quantitatifs. Si nous souhaitons utiliser des variables catégorielles, nous devons préalablement procéder à un codage disjonctif complet (codage 0/1 avec ONE2MANY pour tous les descripteurs discrets).

**Remarque 2 :** Curieusement, la méthode produit 9 règles, dont certains couvrent très peu d'individus. On « sait », le graphique nuage de point est édifiant à ce sujet, que 3 règles suffisent. C'est à ce niveau que jouent les paramètres de l'algorithme d'apprentissage. J'ai un peu regardé. Mais il m'a été difficile de trouver les bons outils pour orienter la technique vers des règles plus ou moins spécialisées. Un article décrit un peu l'influence du paramétrage sur le comportement de l'induction, mais il nous parle surtout des performances en généralisation, pas des caractéristiques des règles générées<sup>5</sup>.

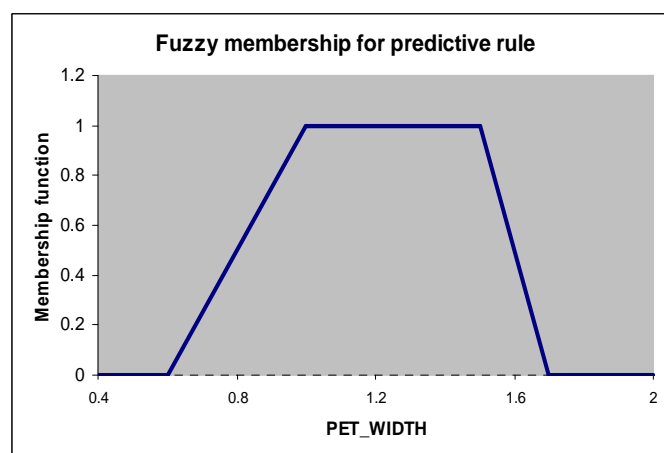
**Remarque 3 :** La lecture d'une règle devient rapidement ingérable dès que le nombre de descripteurs augmente. Il faut lire 4 valeurs pour chacun d'entre eux, et on distingue mal (ou pas du tout ?) ceux qui ne sont pas pertinents.

### 3.6 Interprétation des règles

La lecture du tableau des règles n'est pas aisée. Concentrons sur la n°2, elle désigne la classe « versicolor », son poids est de 22 (elle couvre 22 observations si on simplifie), d'autres indicateurs sont fournis. Pour chaque variable, nous avons les 4 coordonnées sur l'axe des abscisses du trapèze décrivant la fonction d'appartenance. Pour la variable PET\_LENGTH, nous avons **<3.3, 3.3, 4.9, 5.0>**



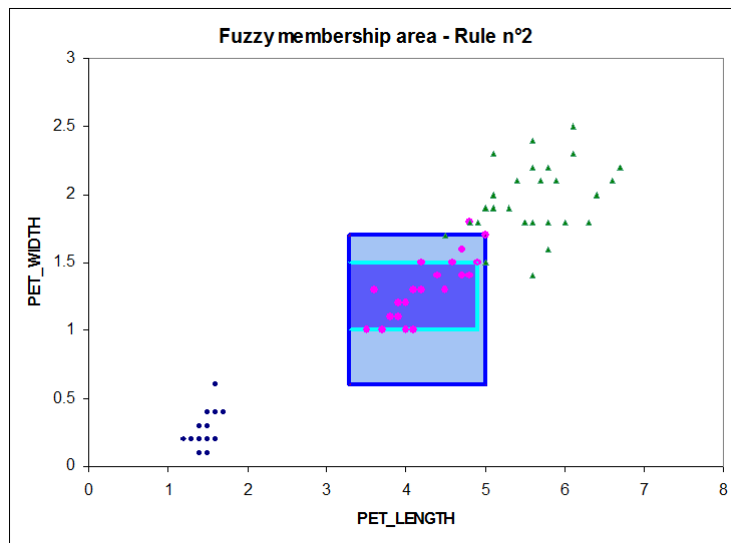
Pour PET\_WIDTH, nous avons **<0.6, 1.0, 1.5, 1.7>**



<sup>5</sup> [http://www.inf.uni-konstanz.de/biomi2/publications/Papers2004/GaBeo4\\_mixedFRappendix\\_ijar.pdf](http://www.inf.uni-konstanz.de/biomi2/publications/Papers2004/GaBeo4_mixedFRappendix_ijar.pdf)

### 3.7 Représentation 2D des règles

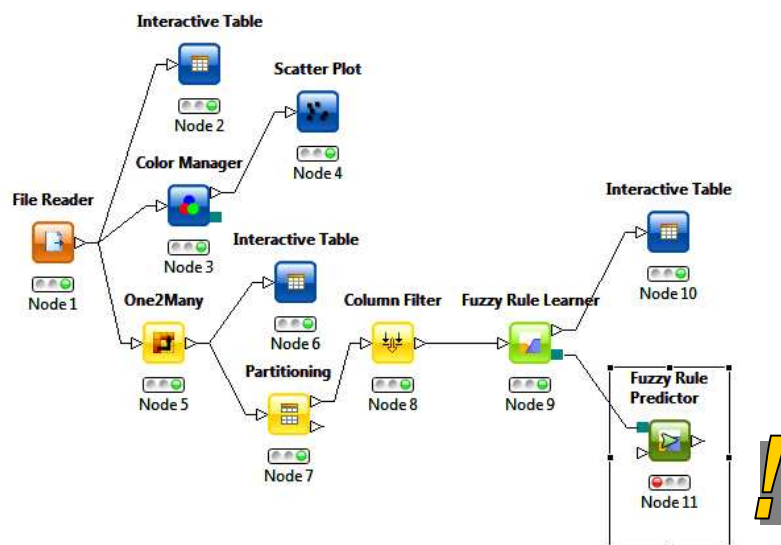
L'ennui est qu'il faut combiner ces deux trapèzes puisque la règle n°2 repose sur les deux variables. Dans le plan, la région d'appartenance peut être représentée de la manière suivante.



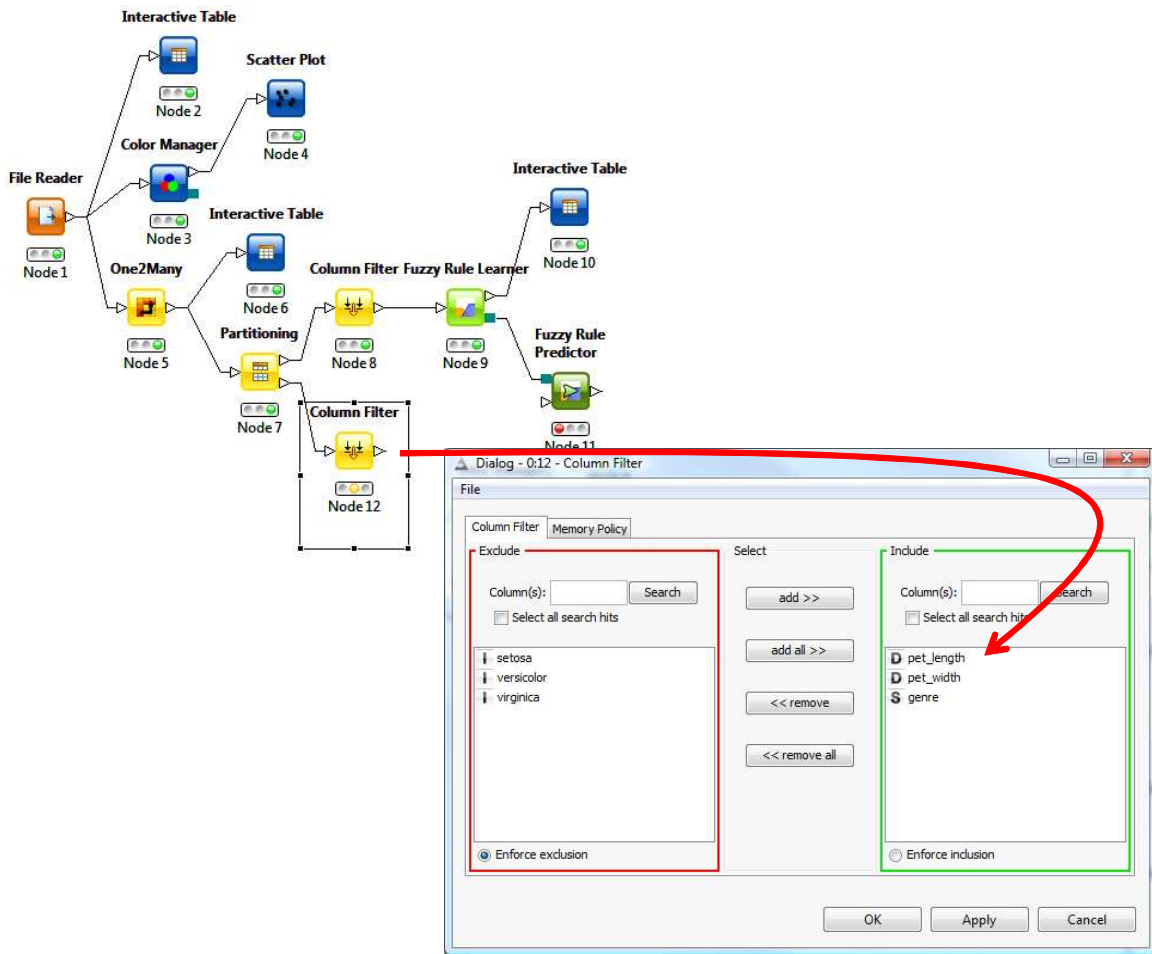
Le rectangle intérieur (bleu foncé) désigne une zone d'appartenance forte ; le rectangle extérieur (bleu clair) désigne la limite de non appartenance (en dehors de cette zone, le degré d'appartenance d'un individu à la règle est nulle) ; du rectangle intérieur vers le rectangle extérieur, il y a une gradation de l'appartenance. On devine aisément que lorsque l'on prend en considération l'ensemble des règles du classifieur, plusieurs rectangles (hyper rectangles si on a plus de deux variables) se recouvrent dans l'espace de représentation. Ce qui rend le classement d'un nouvel individu épineux.

### 3.8 Performances en généralisation – Matrice de confusion

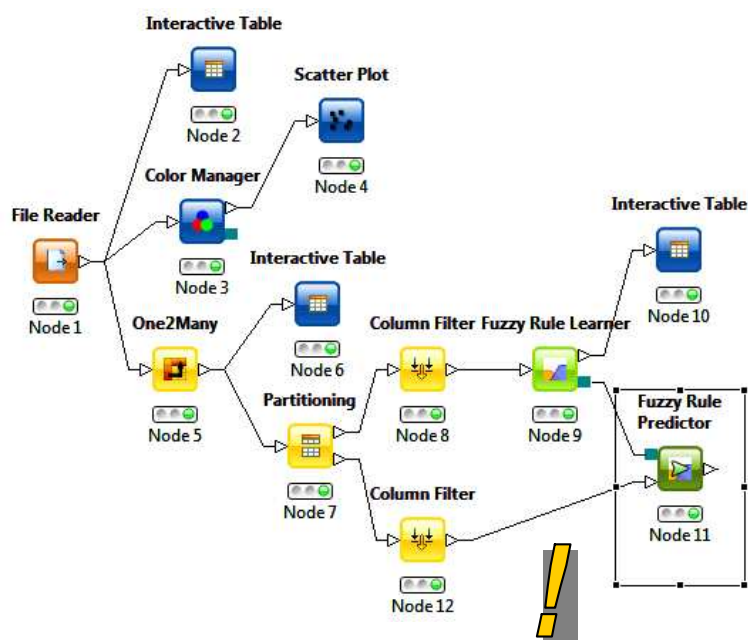
Reste à savoir ce que produit cette base de règles floues en généralisation. Pour cela, nous introduisons le composant **FUZZY RULE PREDICTOR** (*Mining / Rule Induction / Fuzzy Rules*) auquel nous connectons le FUZZY RULE LEARNER.



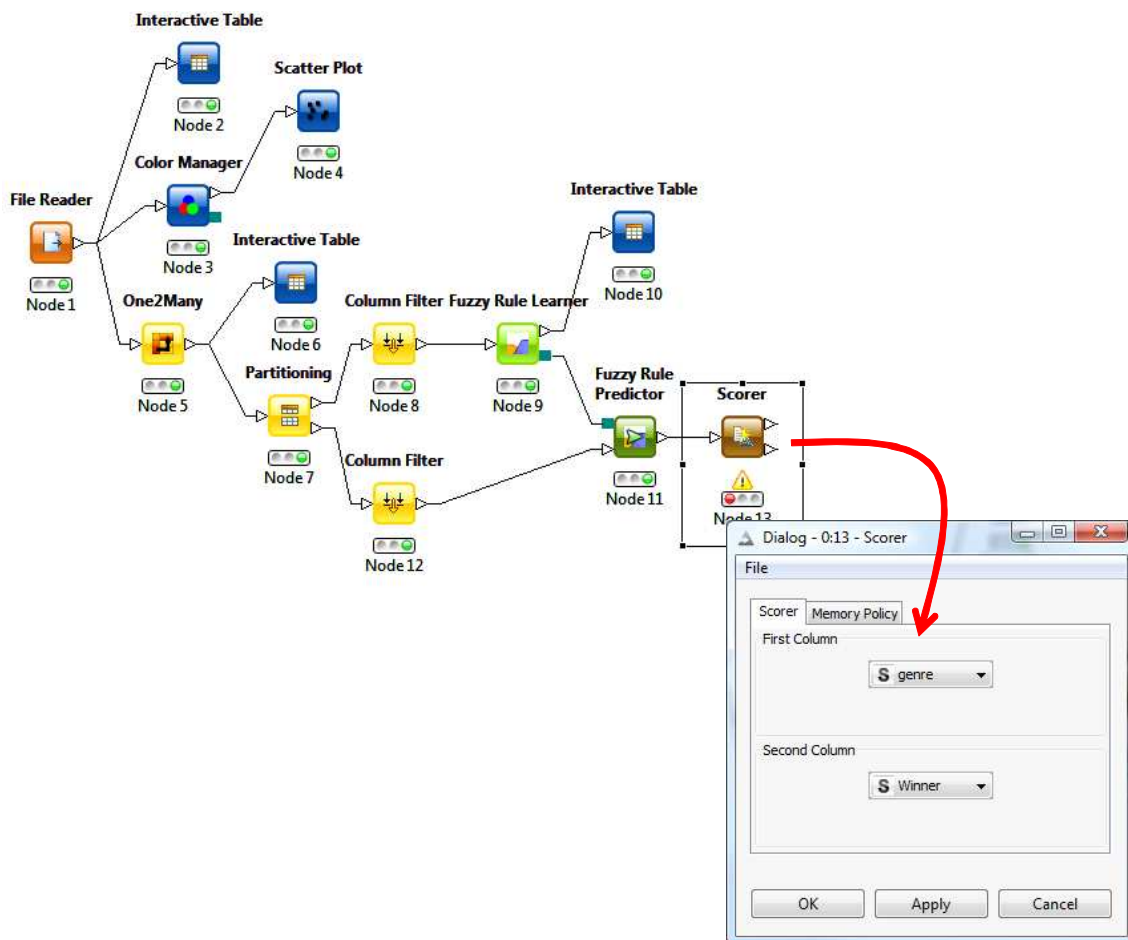
Nous filtrons les colonnes en provenance de PARTITIONNING, nous ne laissons passer que les descripteurs et la variable cible originelle GENRE à l'aide de **COLUMN FILTER**.



Enfin, nous connectons cette dernière à FUZZY RULE PREDICTOR pour réaliser la prédiction sur l'échantillon test.



Il ne reste plus qu'à ajouter le composant SCORER (*Mining / Scoring*) qui se charge de calculer la matrice de confusion. Nous le configurons de manière à opposer GENRE et WINNER (la prédiction du modèle fournie par le composant FUZZY RULE PREDICTOR).



Nous actionnons le menu EXECUTE and OPEN VIEW. Nous obtenons la matrice de confusion suivante avec un taux d'erreur de 4%.

genre \ Wi...	setosa	versicolor	virginica
setosa	23	0	0
versicolor	0	25	1
virginica	0	2	24

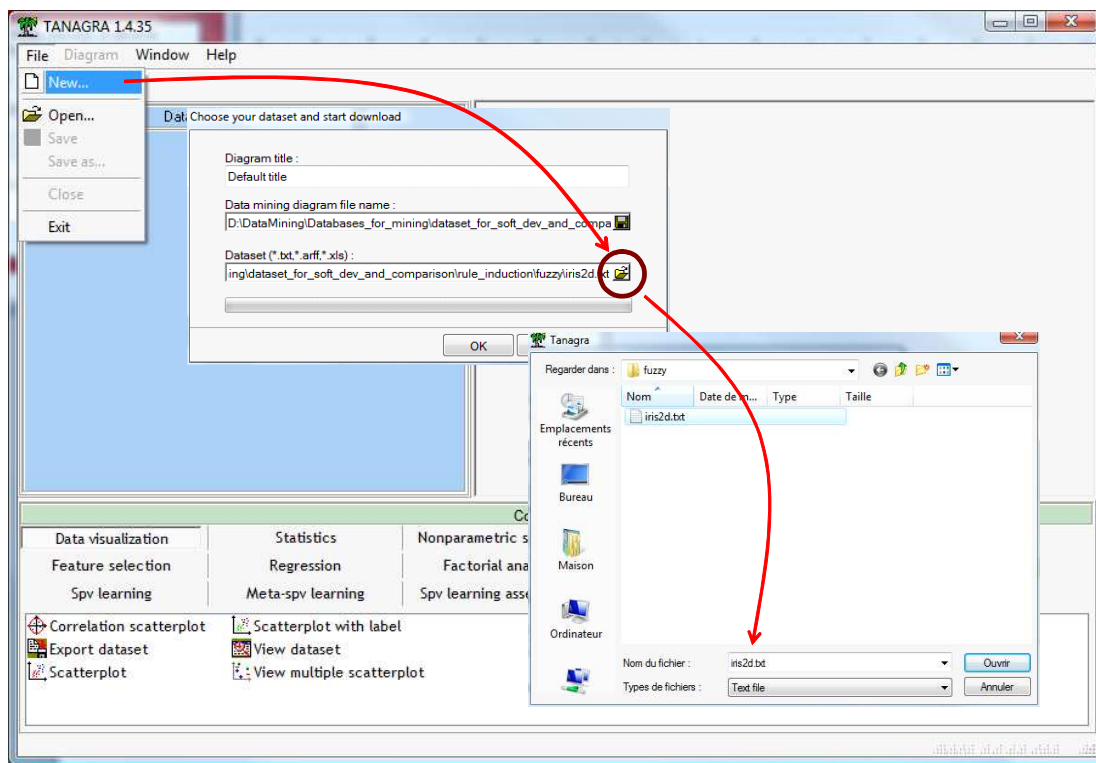
Correct classified: 72      Wrong classified: 3  
Accuracy: 96 %      Error: 4 %

## 4 Induction de règles « crisp » avec Tanagra

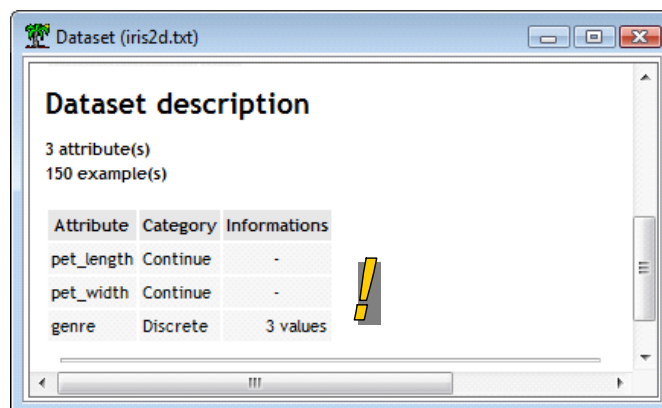
Voyons ce qu'il en est avec les méthodes d'induction de règles « crisp ». Nous utiliserons le composant RULE INDUCTION dans ce didacticiel. Pour les méthodes implémentées dans les autres logiciels (Weka, Orange, R), nous conseillons la lecture du tutoriel « Induction de Règles Prédicatives » (<http://tutoriels-data-mining.blogspot.com/2009/11/induction-de-regles-predictives.html>).

#### 4.1 Création du diagramme et importation des données

Après le démarrage de Tanagra, nous actionnons le menu FILE / NEW. Nous sélectionnons le fichier IRIS2D.TXT dans la boîte de dialogue qui apparaît.



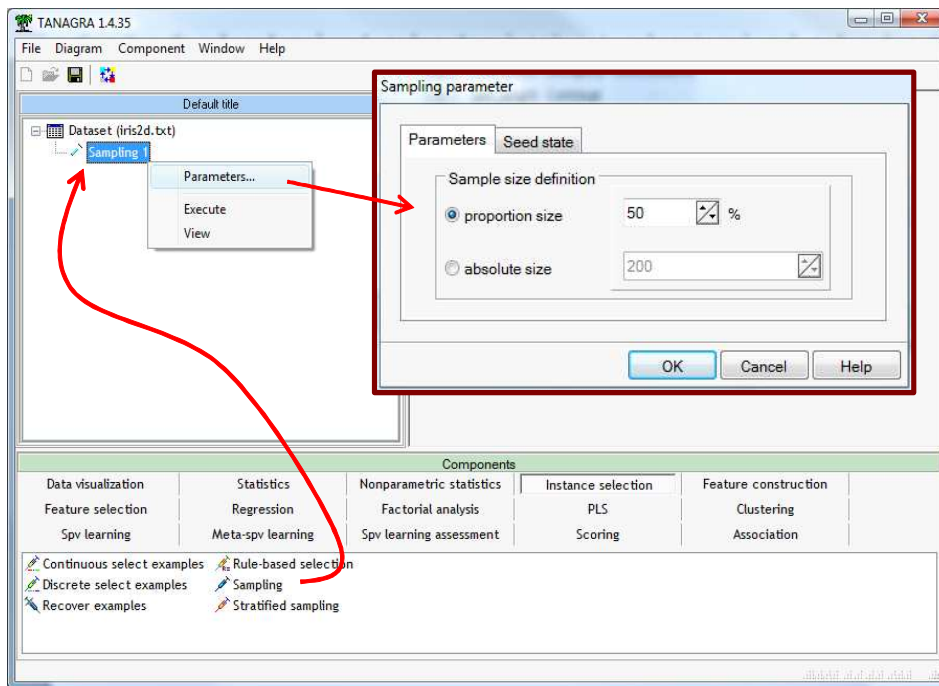
Le fichier est chargé, nous disposons de 150 observations et 3 variables.



**Attention** : Le point décimal est « . » (point) dans le fichier IRIS2D.TXT. Si votre système est configuré pour « , » (virgule), c'est souvent le cas pour Windows en français, il vous faudra ouvrir préalablement le fichier dans un éditeur de texte quelconque (le bloc-notes par exemple), et remplacer les « . » par des « , ». Ensuite seulement, vous le chargerez dans Tanagra.

#### 4.2 Partitionnement en apprentissage et test

Première étape, nous subdivisons le fichier en échantillons d'apprentissage et de test. Le composant SAMPLING (onglet *Instance Selection*) convient pour cela. Nous le paramétrons (menu contextuel PARAMETERS) en demandant 50% des observations pour la partie apprentissage.

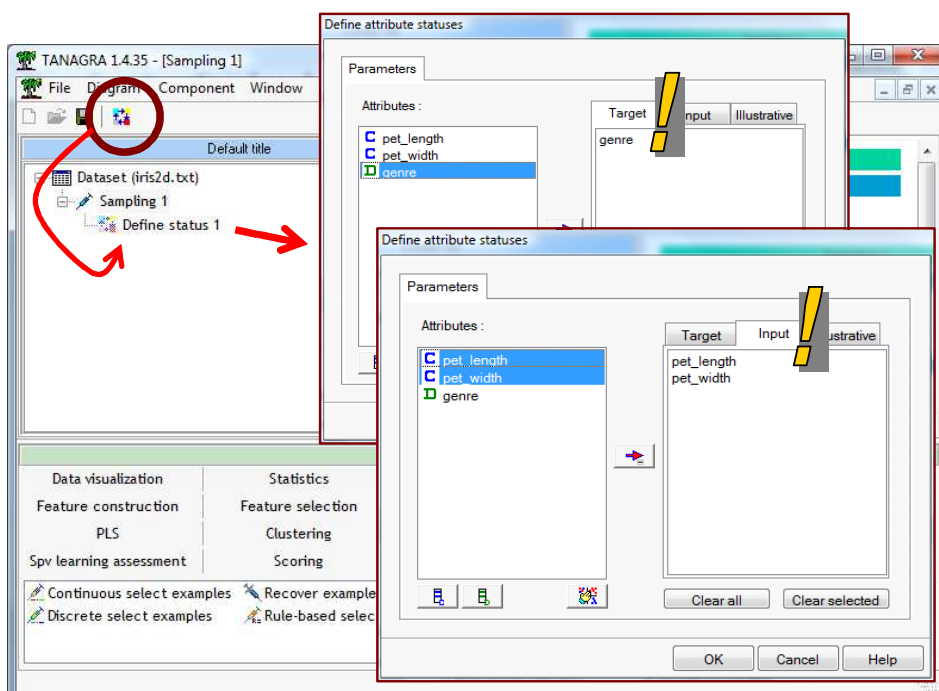


Nous validons et nous cliquons sur le VIEW, 75 observations sont réservées à l'apprentissage.

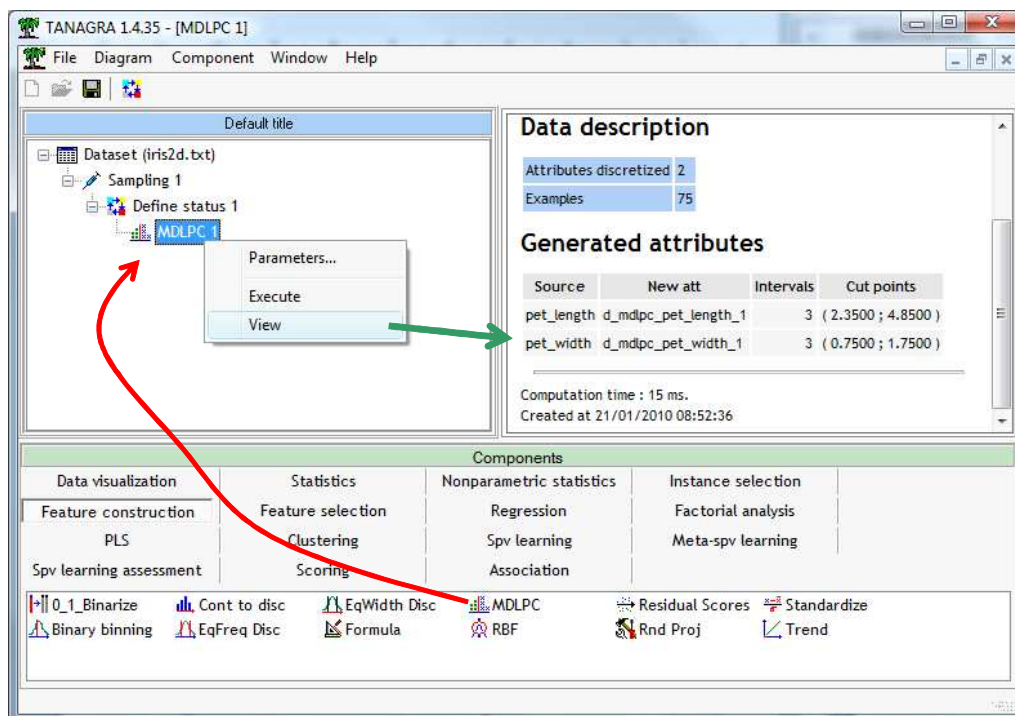
### 4.3 Discrétisation des descripteurs continus

A la différence de celui de Knime, le composant RULE INDUCTION de Tanagra ne gère que les descripteurs discrets. Il nous faut les découper en intervalles, les discrétiser. Il faut que les bornes de découpage soient déterminées de manière à optimiser la prédiction de la variable cible. En d'autres termes, nous devons utiliser une méthode supervisée.

Nous insérons DEFINE STATUS via le raccourci dans la barre d'outils. Nous plaçons en INPUT les attributs à discrétiser (PET\_LENGTH et PET\_WIDTH), en TARGET la variable cible (GENRE).



Puis nous insérons le composant MDLPC (*Feature Construction*). Il n'y a pas de paramètres à manipuler, nous actionnons directement le menu VIEW.

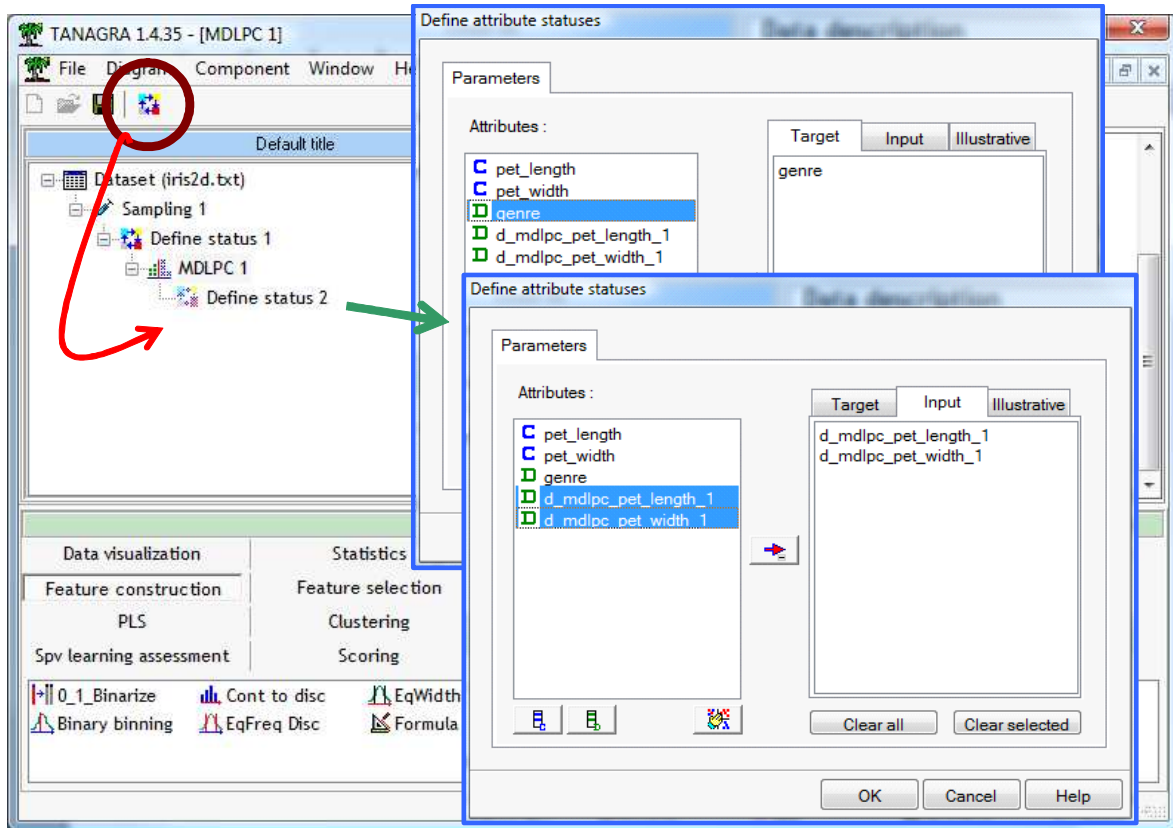


Les deux variables ont été découpées en 3 intervalles. Les seuils sont (2.35 ; 4.85) pour PET\_LENGTH ; (0.75 ; 1.75) pour PET\_WIDTH.

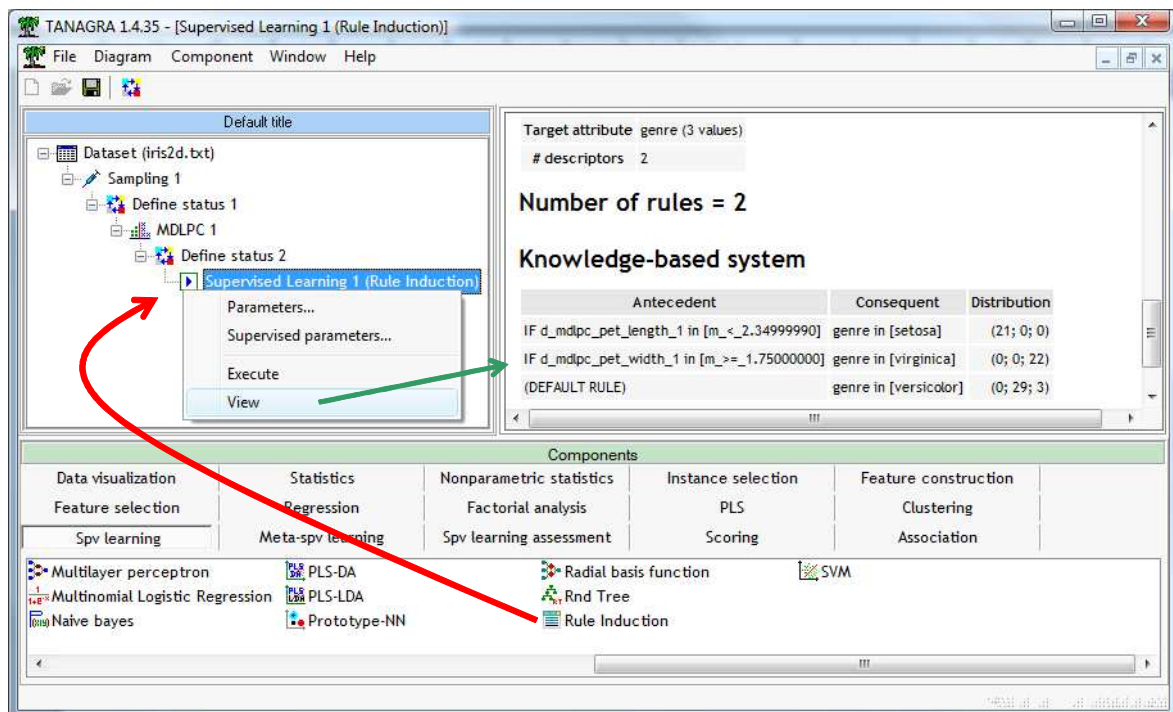
**Remarque :** Il était primordial que la préparation des échantillons d'apprentissage et de test ait été réalisée avant la discrétisation. Ainsi, les bornes ont été obtenues en prenant en compte uniquement la partie apprentissage des données. Dans le cas contraire, si nous avons utilisé toutes les observations disponibles, l'évaluation mise en place par la suite aurait été faussée.

#### 4.4 Induction des règles

Nous pouvons passer à l'induction des règles. Nous insérons de nouveau le composant DEFINE STATUS, nous plaçons en INPUT les variables discrétisées, en TARGET la variable cible GENRE.

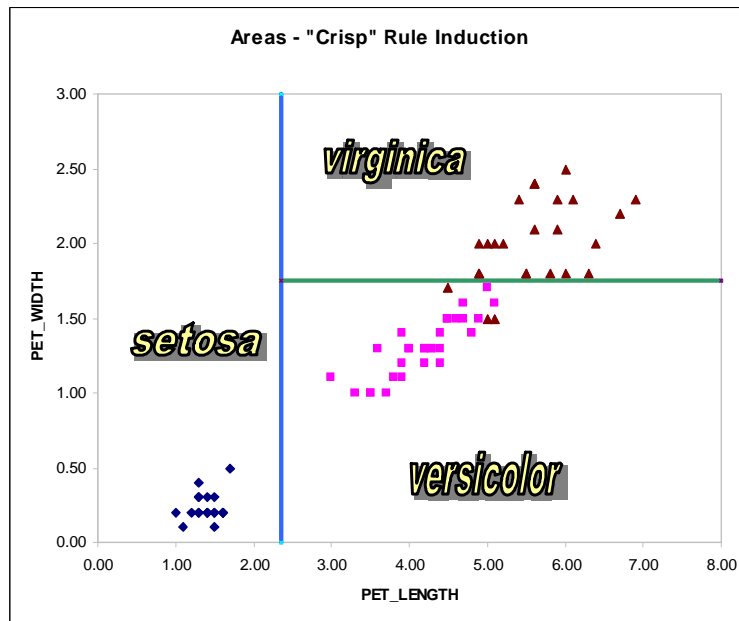


Puis nous introduisons le composant RULE INDUCTION (*Spv Learning*). Nous actionnons directement le menu contextuel VIEW.



Nous obtenons trois règles (la dernière étant la règle par défaut). En se référant à tout ce qui a été dit précédemment, nous devinons que les règles définissent en réalité des zones dans l'espace de représentation.

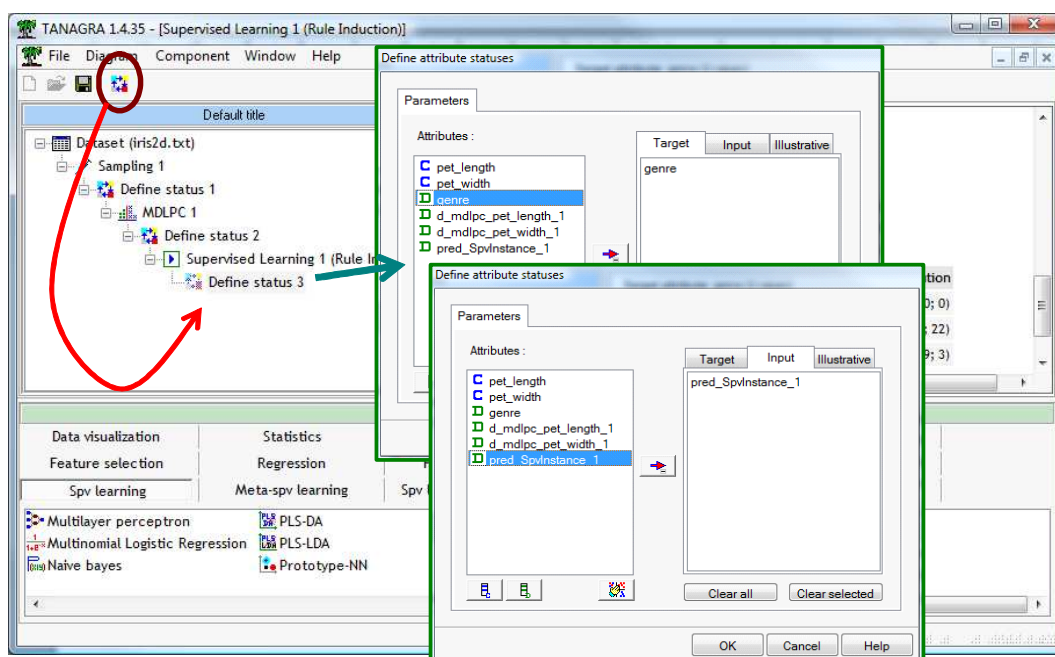
Voyons cela dans un petit graphique qui nous est familier maintenant. Ici les frontières sont « dures » dans le sens où : dès qu'un individu bascule de l'autre côté, on lui attribue pleinement l'étiquette majoritaire de la région.



Bref, les méthodes « crisp » comme les méthodes floues induisent une subdivision de l'espace de représentation en « zones d'influence » de l'une ou l'autre modalité de la variable à prédire. La définition de la frontière en revanche n'est pas la même, les techniques floues introduisent une gradation dans l'appartenance aux zones.

#### 4.5 Evaluation sur l'échantillon test

Pour évaluer les performances, nous insérons DEFINE STATUS. Nous plaçons en TARGET la variable GENRE, en INPUT la prédiction générée par le modèle (PRED\_SPVINSTANCE\_1).



Puis, nous introduisons le composant TEST (*Spv Learning Assessment*). Il calcule automatiquement la matrice de confusion sur l'échantillon test, nous cliquons directement sur VIEW. Nous obtenons la matrice de confusion suivante, avec un taux d'erreur de 4%, le même que pour le classifieur flou.

The screenshot shows the TANAGRA 1.4.35 interface. On the left, a project tree shows a workflow: Dataset (iris2d.txt) -> Sampling 1 -> Define status 1 -> MDLPC 1 -> Define status 2 -> Supervised Learning 1 (Rule Induction) -> Define status 3 -> Test 1. A red arrow points from the 'Test 1' component in the tree to the 'View' button in its context menu. Another red arrow points from the 'View' button to the 'Results' window. The 'Results' window displays the following data:

Results							
pred_Spvinstance_1							
Error rate		0.0400					
Values prediction		Confusion matrix					
Value	Recall	1-Precision		setosa	versicolor	virginica	Sum
setosa	1.0000	0.0000	setosa	29	0	0	29
versicolor	0.9524	0.0909	versicolor	0	20	1	21
virginica	0.9200	0.0417	virginica	0	2	23	25
			Sum	29	22	24	75

The 'Components' palette at the bottom shows various analysis tools, with 'Spv learning assessment' selected. The 'Test' component is also visible in the palette.

A vrai dire, le taux d'erreur est anecdotique dans ce didacticiel. On sait que le fichier IRIS est facile à apprendre. L'enjeu était surtout de montrer qu'une règle trace en fait un hyper rectangle dans l'espace de représentation.

## 5 Conclusion

L'intérêt scientifique de l'induction des règles prédictives floues est indéniable. Pourtant, elle est peu connue en dehors du cercle restreint des chercheurs. Principalement parce qu'elle est peu programmée dans le logiciel, restreignant d'autant la diffusion de la technique. Knime est un des rares outils « généraliste » à la proposer, il me semblait intéressant de la mettre en avant en lui consacrant un didacticiel.