1 Subject

Using cross-validation for the performance evaluation of decision trees with R, KNIME and RAPIDMINER.

This paper takes one of our old study on the implementation of cross-validation for assessing the performance of decision trees. We were compared the procedure to follow for Tanagra, Orange and Weka[1]. In this tutorial, we describe the utilization of other tools: R 2.7.2 (http://www.r-project.org/), Knime 1.3.51 (http://www.knime.org/) and RapidMiner Community Edition (http://rapid-i.com/content/blogcategory/38/69/).

The objectives and the steps are the same. The reader can see our previous tutorial if asked for details on these elements. We use the HEART.TXT dataset (UCI IRVINE, http://archive.ics.uci.edu/ml/datasets/Heart+Disease). We want to predict the occurrence of heart diseases. We have 12 predictors and 270 instances.

2 Decision tree + Cross-validation with R (package rpart)

Loading the rpart library. We want to use the rpart procedure from the rpart package. So we need to install it, then we use the following command.

```
#charger la bibliothèque rpart
library(rpart)
```

Data file importation. We import the dataset[2] in a data frame (**donnees**). We compute some descriptive statistics in order to check the dataset.

```
#charger les données
setwd("D:/DataMining/Databases_for_mining/comparison_TOW/validation_croisee")
donnees <- read.table(file="heart.txt",dec=".",header=TRUE)
summary(donnees)
```

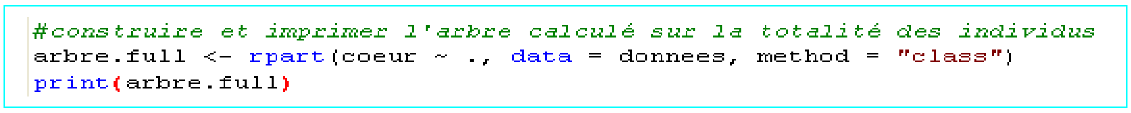
```
      age              sexe      type_douleur    pression        cholester
 Min.   :29.00   feminin : 87   A: 20        Min.   : 94.0   Min.   :126.0
 1st Qu.:48.00   masculin:183   B: 42        1st Qu.:120.0   1st Qu.:213.0
 Median :55.00                  C: 79        Median :130.0   Median :245.0
 Mean   :54.43                  D:129        Mean   :131.3   Mean   :249.7
 3rd Qu.:61.00                               3rd Qu.:140.0   3rd Qu.:280.0
 Max.   :77.00                               Max.   :200.0   Max.   :564.0
 sucre    electro      taux_max      angine      depression         pic
 A:230   A:131    Min.   : 71.0   non:181   Min.   : 0.0   Min.   :1.000
 B: 40   B:  2    1st Qu.:133.0   oui: 89   1st Qu.: 0.0   1st Qu.:1.000
         C:137    Median :153.5             Median : 8.0   Median :2.000
                  Mean   :149.7             Mean   :10.5   Mean   :1.585
                  3rd Qu.:166.0             3rd Qu.:16.0   3rd Qu.:2.000
                  Max.   :202.0             Max.   :62.0   Max.   :3.000
 vaisseau       coeur
 A:160     absence :150
 B: 58     presence:120
 C: 33
 D: 19
```

[1] See http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/en_Tanagra_TOW_Decision_Tree.pdf

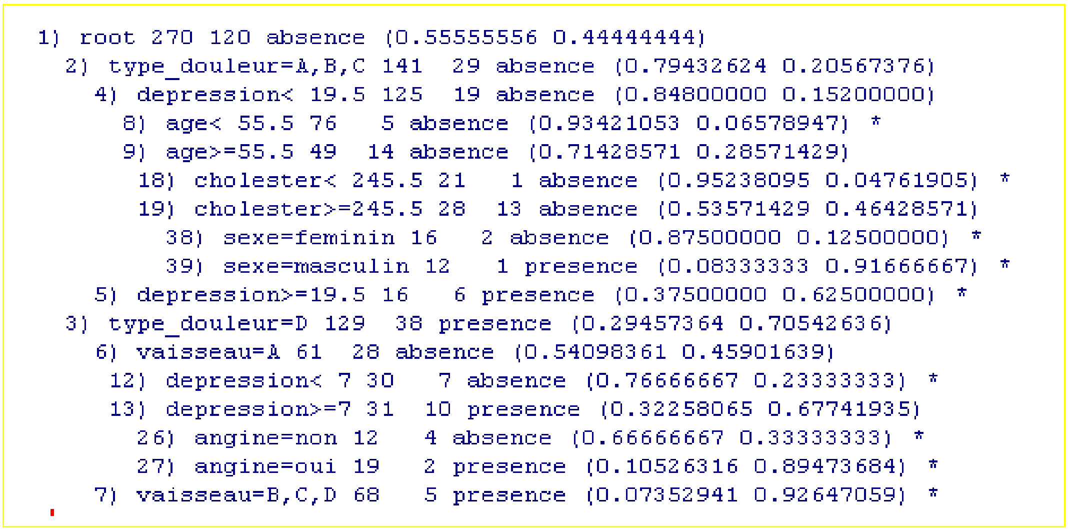
[2] The following URL includes the source code for R, http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/heart.zip

We note, among other things, that the class attribute COEUR has two values (presence and absence), with respectively 150 and 120 examples. There are 270 examples in total. On a small file, partitioning the dataset in learning and test samples is not a good idea. We must use the re-sampling methods in order to evaluate the classifier performances.

Induction of the decision tree on the whole dataset. In a first step, we induce the tree on the whole dataset with the **rpart(.)** command. We use the default parameters.

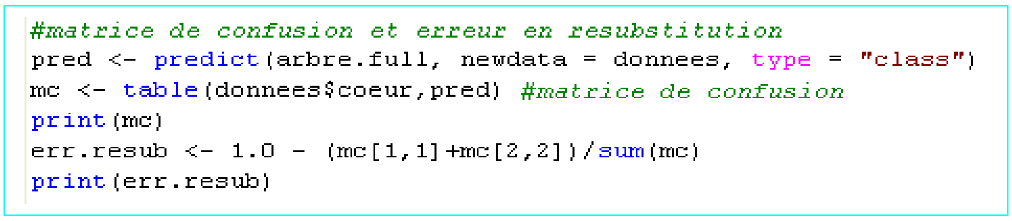


```
#construire et imprimer l'arbre calculé sur la totalité des individus
arbre.full <- rpart(coeur ~ ., data = donnees, method = "class")
print(arbre.full)
```

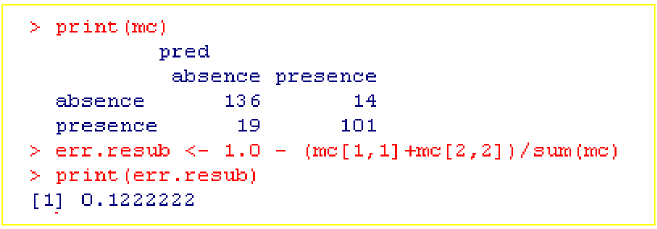


```
1) root 270 120 absence (0.55555556 0.44444444)
  2) type_douleur=A,B,C 141  29 absence (0.79432624 0.20567376)
    4) depression< 19.5 125  19 absence (0.84800000 0.15200000)
      8) age< 55.5 76   5 absence (0.93421053 0.06578947) *
      9) age>=55.5 49  14 absence (0.71428571 0.28571429)
       18) cholester< 245.5 21   1 absence (0.95238095 0.04761905) *
       19) cholester>=245.5 28  13 absence (0.53571429 0.46428571)
         38) sexe=feminin 16   2 absence (0.87500000 0.12500000) *
         39) sexe=masculin 12   1 presence (0.08333333 0.91666667) *
    5) depression>=19.5 16   6 presence (0.37500000 0.62500000) *
  3) type_douleur=D 129  38 presence (0.29457364 0.70542636)
    6) vaisseau=A 61  28 absence (0.54098361 0.45901639)
     12) depression< 7 30   7 absence (0.76666667 0.23333333) *
     13) depression>=7 31  10 presence (0.32258065 0.67741935)
       26) angine=non 12   4 absence (0.66666667 0.33333333) *
       27) angine=oui 19   2 presence (0.10526316 0.89473684) *
    7) vaisseau=B,C,D 68   5 presence (0.07352941 0.92647059) *
```

Resubstitution error rate. In order to obtain the resubstitution error rate i.e. the error rate computed on the learning sample, we calculate the prediction column (**pred**). Then, we get the confusion matrix and the subsequent error rate.



```
#matrice de confusion et erreur en resubstitution
pred <- predict(arbre.full, newdata = donnees, type = "class")
mc <- table(donnees$coeur,pred) #matrice de confusion
print(mc)
err.resub <- 1.0 - (mc[1,1]+mc[2,2])/sum(mc)
print(err.resub)
```



```
> print(mc)
          pred
           absence presence
  absence      136       14
  presence      19      101
> err.resub <- 1.0 - (mc[1,1]+mc[2,2])/sum(mc)
> print(err.resub)
[1] 0.1222222
```

The resubstitution error rate is 12.22%. We know that this indicator is biased; it underestimates the true error rate.

Cross validation. Cross validation is a resampling approach which enables to obtain a more honest error rate estimate of the tree computed on the whole dataset. It is almost available on all the data mining software. With R, we must to program the method, but it is rather simple.

Basically, the cross validation consists to randomly split the data in K folds. We reiterate the following process, by turning the sub-samples: learning the model on (K-1) folds, computing the error rate on the fold number K. The error rate in cross-validation is the mean of these error rates collected. It is a better estimator of the classifier performance than the resubstitution error rate.

From this description, we transcribe the operations in R. We create randomly a column indicating the individuals belonging to the folds.

```
#déterminer le numéro de bloc de chaque individu
n <- nrow(donnees) #nombre d'observations
K <- 10 # pour 10-validation croisée
taille <- n%/%K #déterminer la taille de chaque bloc
set.seed(5) #pour obtenir la même séquence tout le temps
alea <- runif(n) #générer une colonne de valeurs aléatoires
rang <- rank(alea) #associer à chaque individu un rang
bloc <- (rang-1)%/%taille + 1 # associer à chaque individu un numéro de bloc
bloc <- as.factor(bloc) #transformer en factor
print(summary(bloc)) #impression de contrôle
```

We observe that we have the same number of examples in each fold.

```
> print(summary(bloc)) #impression de contrôle
  1  2  3  4  5  6  7  8  9 10
 27 27 27 27 27 27 27 27 27 27
```

We can repeat now the learning process and the test process. We collect each error rate in a vector.

```
#lancer la validation croisée
all.err <- numeric(0)
for (k in 1:K){
  #apprendre le modèle sur tous les individus sauf le bloc k
  arbre <- rpart(coeur ~., data = donnees[bloc!=k,], method = "class")
  #appliquer le modèle sur le bloc numéro k
  pred <- predict(arbre,newdata=donnees[bloc==k,], type = "class")
  #matrice de confusion
  mc <- table(donnees$coeur[bloc==k],pred)
  #taux d'erreur
  err <- 1.0 - (mc[1,1]+mc[2,2])/sum(mc)
  #conserver
  all.err <- rbind(all.err,err)
}

#vecteur des erreurs recueillies
print(all.err)
```

We obtain the following vector.

```
> #vecteur des erreurs recueillies
> print(all.err)
              [,1]
err 0.2962963
err 0.2222222
err 0.1481481
err 0.1481481
err 0.1851852
err 0.1851852
err 0.2962963
err 0.3333333
err 0.1111111
err 0.1851852
```

Figure 1 – Test error rate for each fold

Because we have the same number of examples in each fold, we can compute unweighted mean. This is the cross validation error rate estimation.

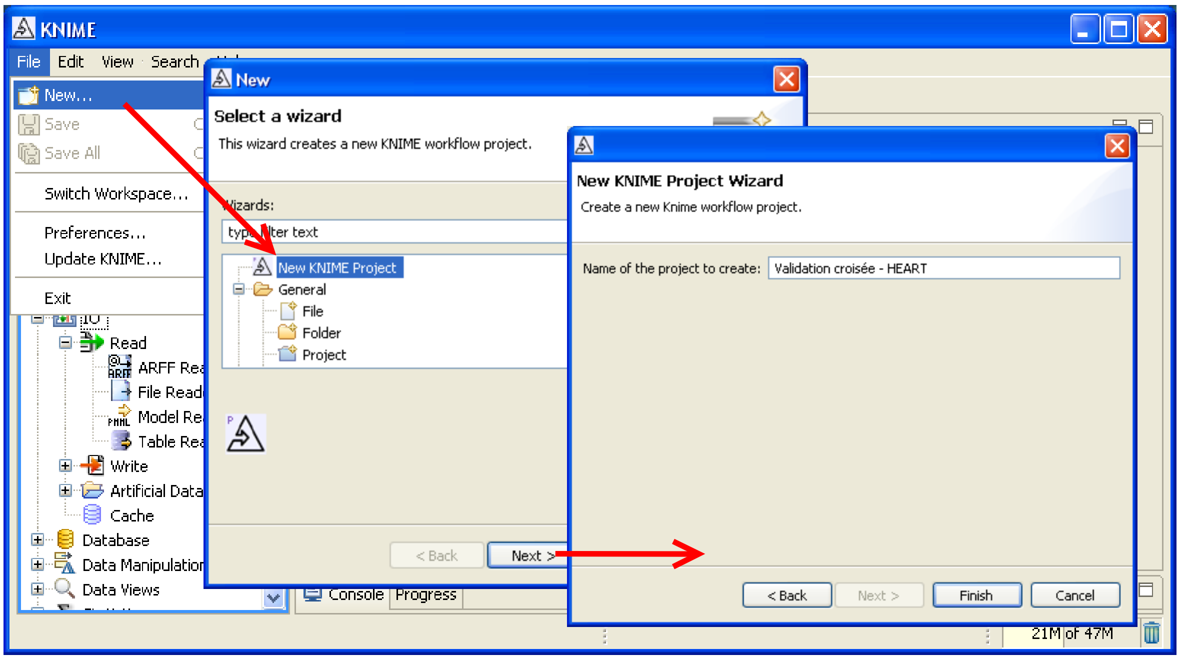
```
#erreur en validation croisée
#on peut se contenter d'une moyenne non pondérée puisque les blocs sont
#de taille identique
err.cv <- mean(all.err)
print(err.cv)
```

```
> print(err.cv)
[1] 0.2111111
```

The cross validation error rate of the rpart(.) method on the HEART dataset is 21.11%.

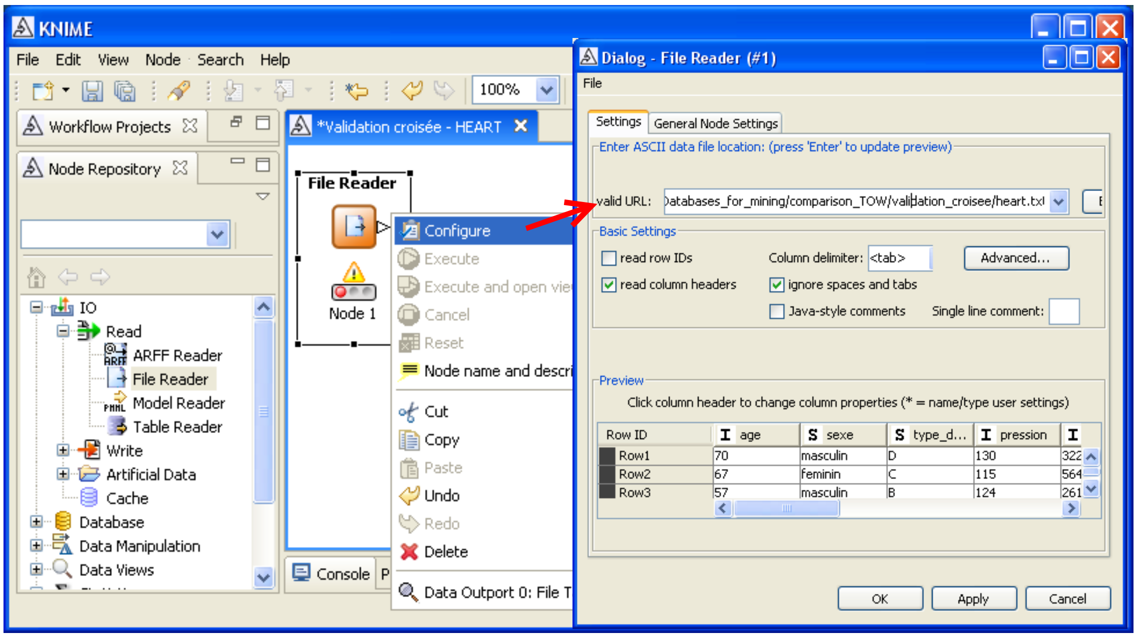
3 Decision tree + Cross validation with KNIME

New workflow and data importation. In order to create a workflow, we click on the FILE / NEW menu. We specify a new project named "*Validation Croisée – HEART*".



Name of the project to create: Validation croisée - HEART

We import the HEART.TXT data file with the FILE READER component. Using the CONFIGURE menu, we select the data file.



Enter ASCII data file location: (press 'Enter' to update preview)

valid URL: Databases_for_mining/comparison_TOW/validation_croisee/heart.txt

Basic Settings

- [ ] read row IDs Column delimiter: <tab> Advanced...
- [✓] read column headers [✓] ignore spaces and tabs
- [ ] Java-style comments Single line comment:

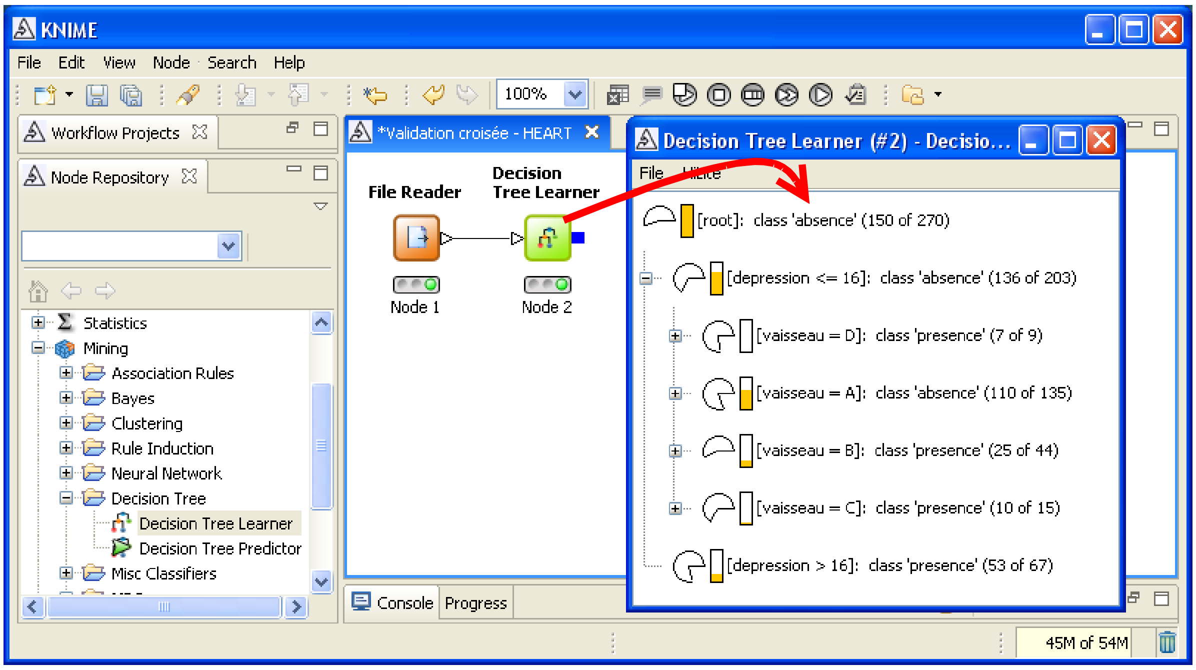
Preview

Click column header to change column properties (* = name/type user settings)

Row ID	I age	S sexe	S type_d...	I pression	I
Row1	70	masculin	D	130	322
Row2	67	feminin	C	115	564
Row3	57	masculin	B	124	261

A part of the dataset appears in a grid. We validate. The EXECUTE menu enables us to finalize the data importation.

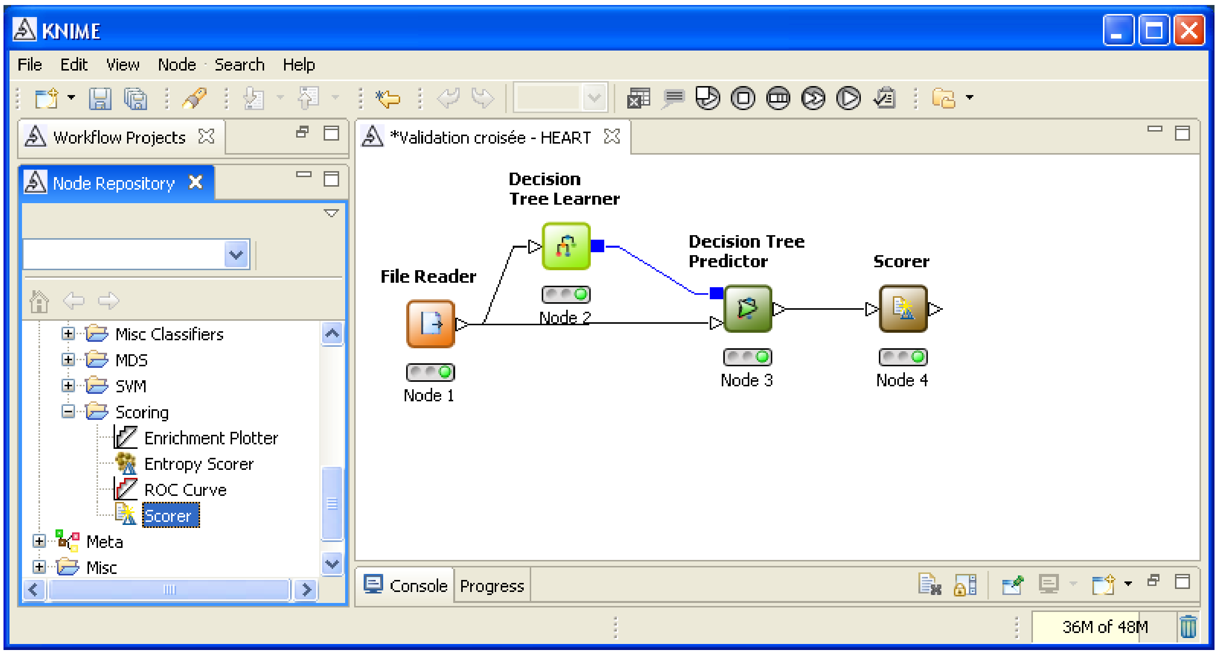
Induction of the tree on the whole dataset. We use the DECISION TREE LEARNER component. We make a connexion with the FILE READER component. We click on the EXECUTE AND OPEN VIEW menu for enabling the learning phase.



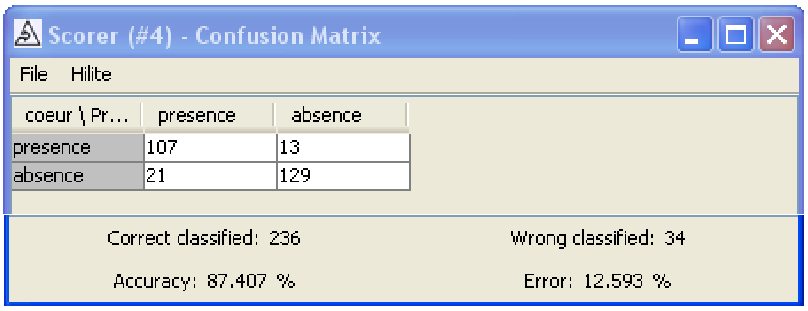
```
[root]: class 'absence' (150 of 270)
  [depression <= 16]: class 'absence' (136 of 203)
    [vaisseau = D]: class 'presence' (7 of 9)
    [vaisseau = A]: class 'absence' (110 of 135)
    [vaisseau = B]: class 'presence' (25 of 44)
    [vaisseau = C]: class 'presence' (10 of 15)
  [depression > 16]: class 'presence' (53 of 67)
```

By default, Knime use the last column as the class attribute. This choice is right for our dataset. We can modify this default selection with the CONFIGURE menu.

Resubstitution error rate. We apply the classifier to the learning set. We use the DECISION TREE PREDICTOR. The SCORER component computes the confusion matrix and the error rate.



The resubstitution error rate is 12.593%.

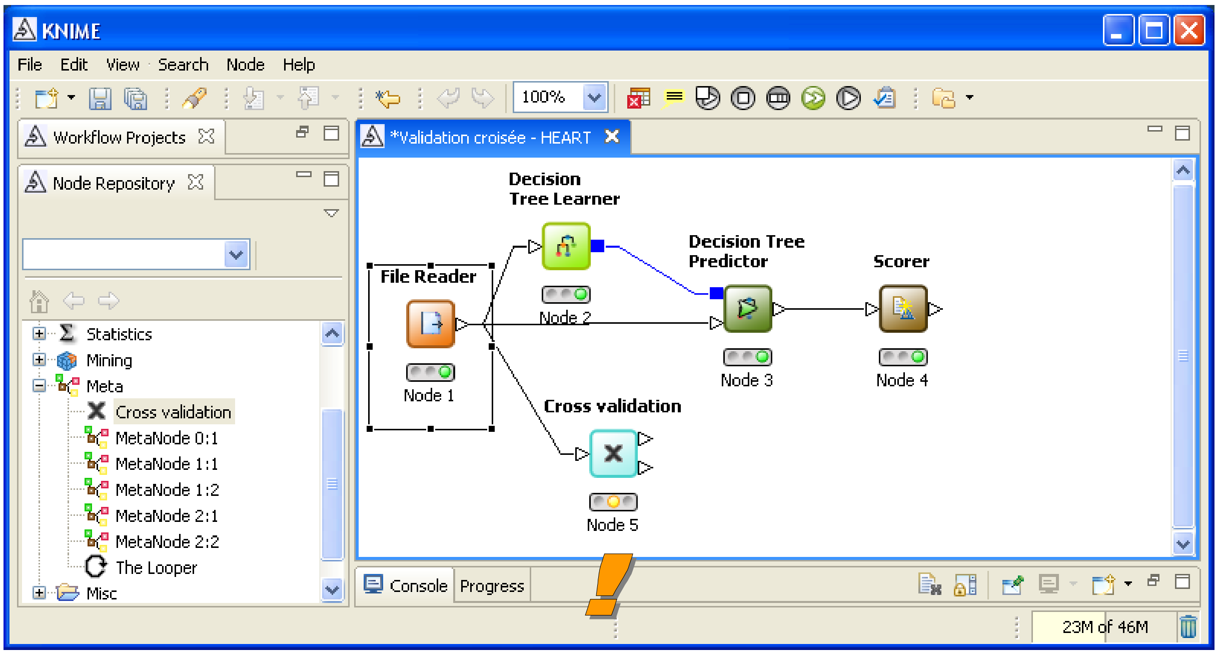


coeur \ Pr...	presence	absence
presence	107	13
absence	21	129

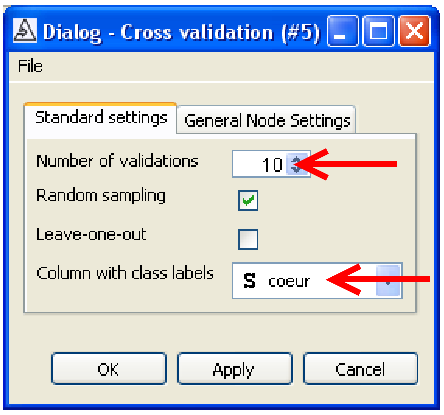
Correct classified: 236 Wrong classified: 34

Accuracy: 87.407 % Error: 12.593 %

Cross validation error rate. In order to implement the cross validation, Knime defines the meta-node concept. It contains a pre-defined sequence of components.

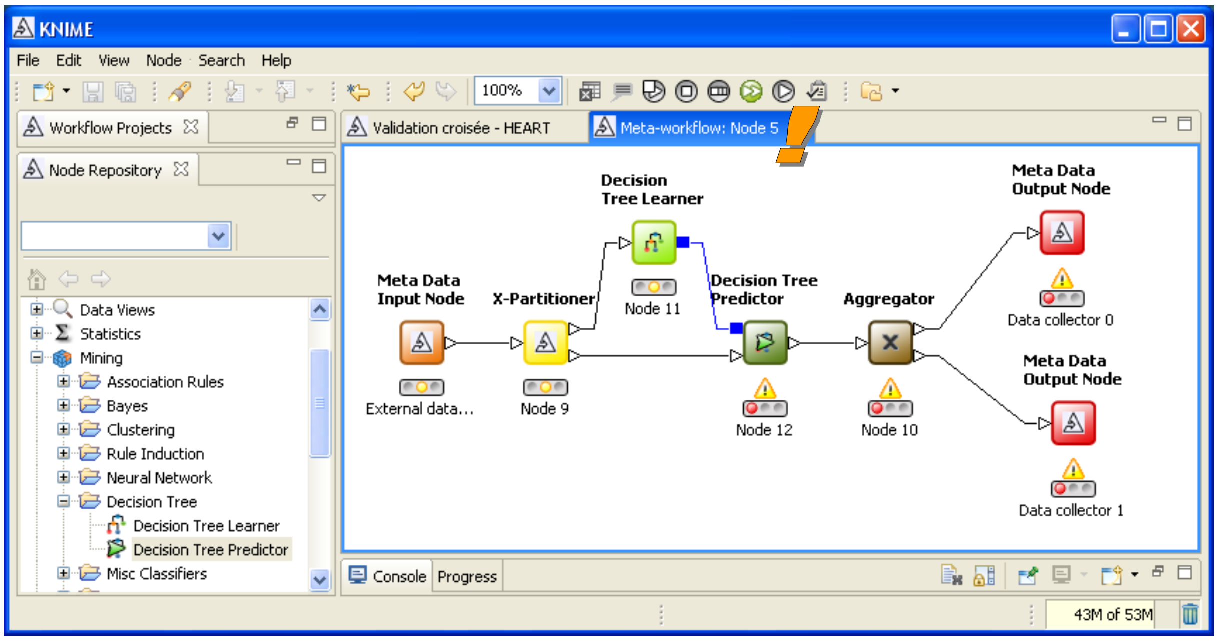


With the CONFIGURE menu, we define: the class attribute (CŒUR) and the number of folds (10).



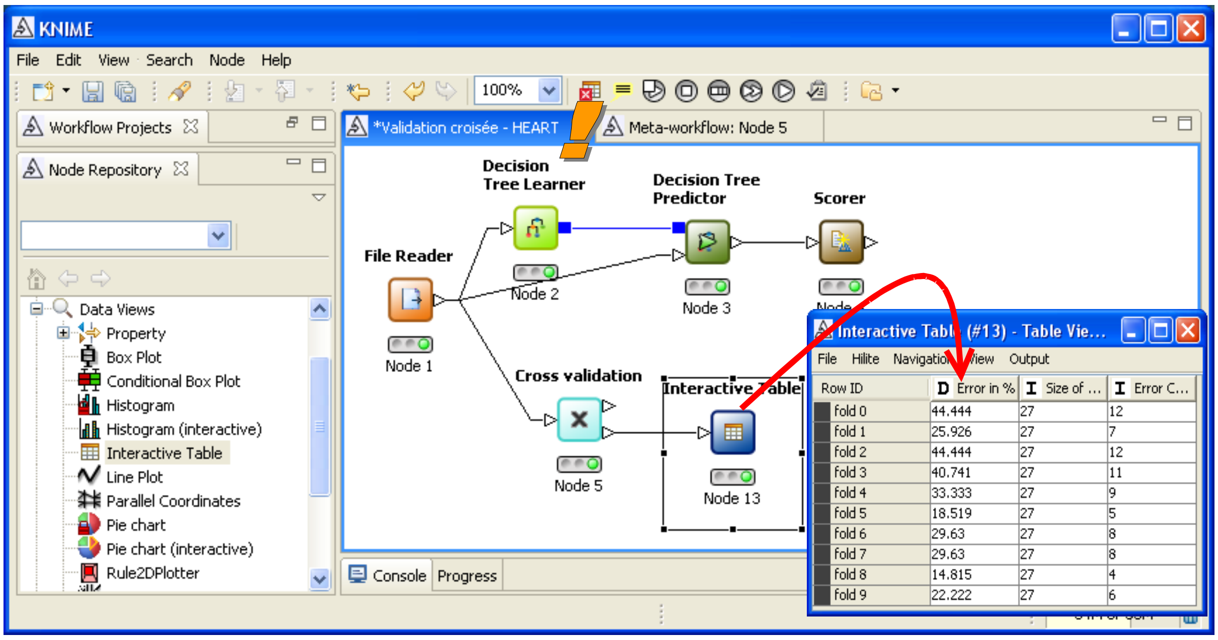
Number of validations: 10
Random sampling: ✓
Leave-one-out: ☐
Column with class labels: coeur

To complete the internal components of the meta node, we click on the menu OPEN META - WORKFLOW EDITOR. A new WORKFLOW EDITOR appears. By analogy with the resubstitution evaluation of classifier, we supplement the meta node with the learning method (DECISION TREE LEARNER) and the predictor (DECISION TREE PREDICTOR).



We return to the original workflow. We can see the results in 2 ways: a table containing the error rate for each fold; a prediction column that we can compare to the values of the class attribute.

In order to obtain the error rate in each fold, we use the interactive component table. We connect this one to the preceding component and we click on the EXECUTE AND OPEN VIEW menu.

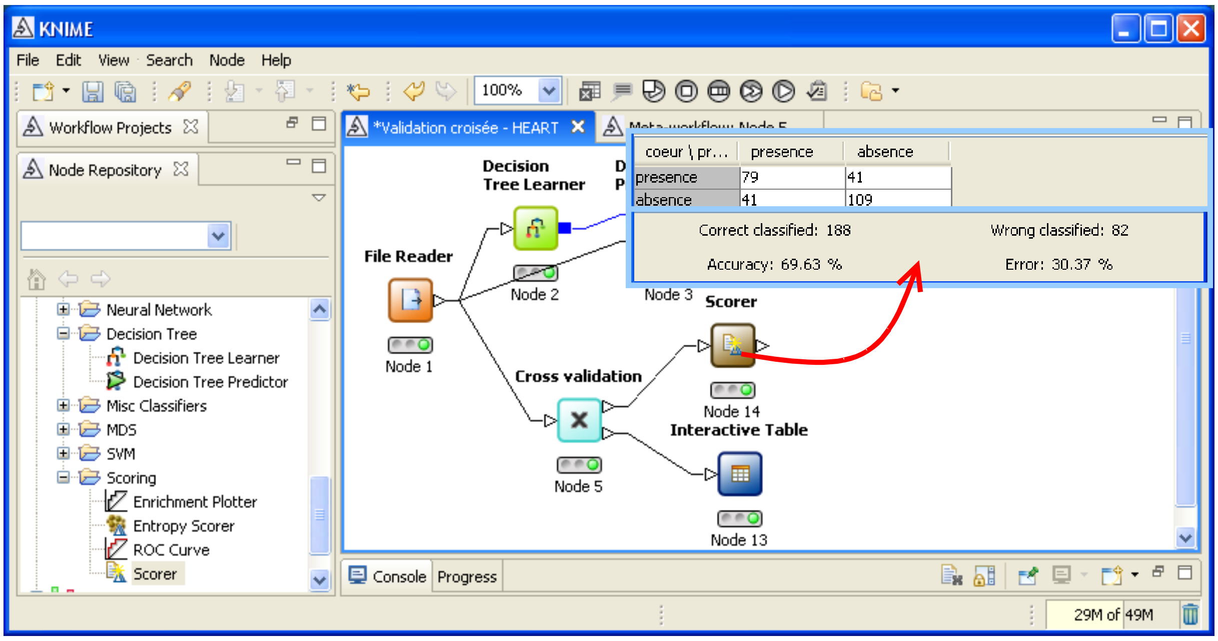


Interactive Table (#13) - Table View

Row ID	D Error in %	I Size of …	I Error C…
fold 0	44.444	27	12
fold 1	25.926	27	7
fold 2	44.444	27	12
fold 3	40.741	27	11
fold 4	33.333	27	9
fold 5	18.519	27	5
fold 6	29.63	27	8
fold 7	29.63	27	8
fold 8	14.815	27	4
fold 9	22.222	27	6

We can compare these values with those of R (Figure 1).

To get the overall cross-validation error rate, we use again the SCORER tool. The confusion matrix is also displayed.



coeur \ pr…	presence	absence
presence	79	41
absence	41	109

Correct classified: 188 Wrong classified: 82

Accuracy: 69.63 % Error: 30.37 %

The cross validation error rate is 30.37%, very different to the resubstitution error rate.

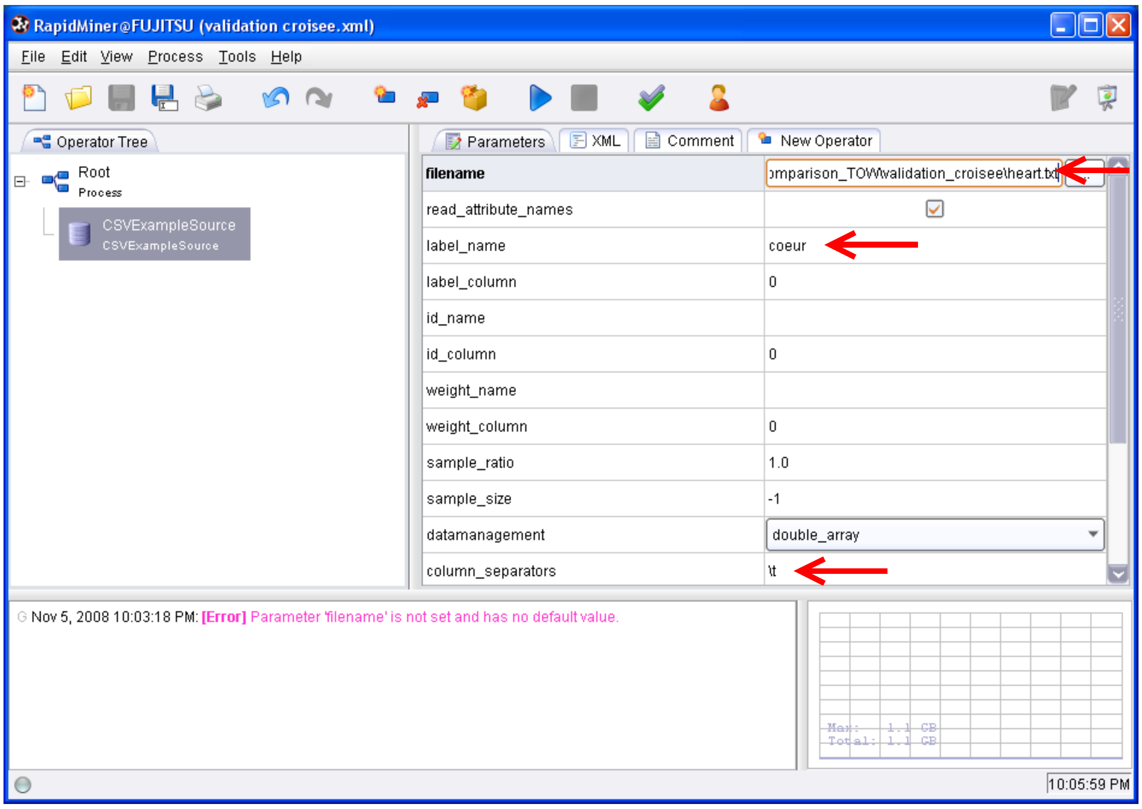
Note: Because they do not use the same induction tree algorithm (not the same parameters also), it is obvious that we cannot obtain the same cross validation error rate with R and Knime. However, such a gap (21.11% vs. 30.37%) deserves that attention be given more about the methods actually programmed. Maybe this will be the subject of an upcoming tutorial.

4 Decision tree + Cross validation with RAPIDMINER

In contrast to other software, we have to define the whole of trafficking before starting the calculations with RAPIDMINER. Indeed, at each computation request, it launches calculations on all components. It is not possible to request a selective execution of the branch of the diagram.

Another notable difference, the cross-validation with RapidMiner generates directly the tree computed on the whole dataset. It is not necessary to make this calculation separately.

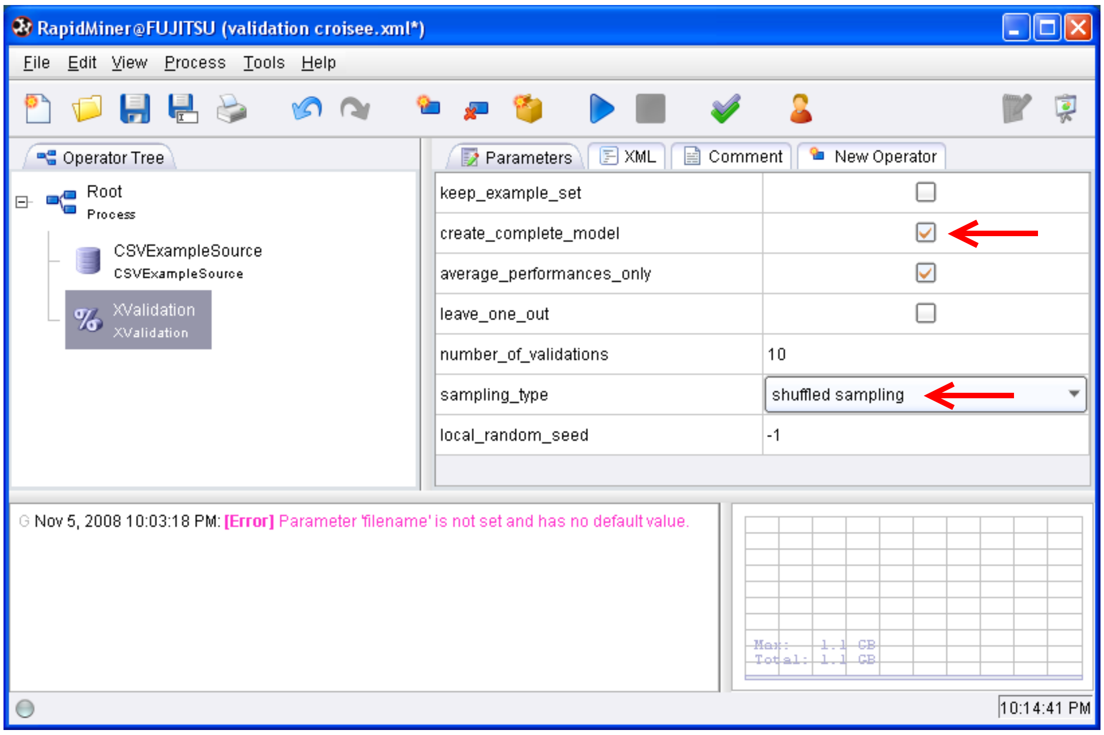
Diagram creation and data importation. We create a new project by clicking on the FILE / NEW menu. We add a CSV EXAMPLE SOURCE at the root of the project. We specify the file name, the class attribute (LABEL_NAME) and the column separator.


RapidMiner@FUJITSU (validation croisee.xml)
File Edit View Process Tools Help
Operator Tree
Root
Process
CSVExampleSource
CSVExampleSource
Parameters XML Comment New Operator

filename	ɔmparison_TOW\validation_croisee\heart.txt
read_attribute_names	✓
label_name	coeur
label_column	0
id_name	
id_column	0
weight_name	
weight_column	0
sample_ratio	1.0
sample_size	-1
datamanagement	double_array
column_separators	\t

```
Nov 5, 2008 10:03:18 PM: [Error] Parameter 'filename' is not set and has no default value.
```

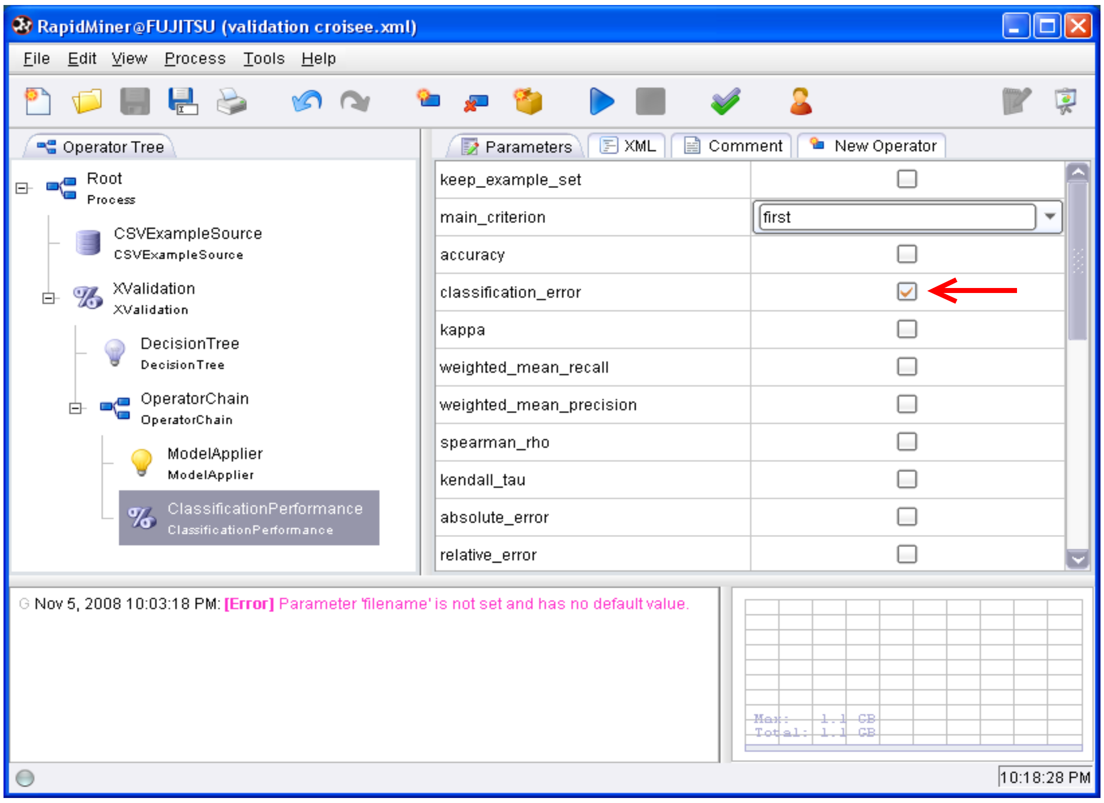
Cross validation. We insert the cross-validation component (XVALIDATION). We set the parameters: we ask the computation of the tree on the whole dataset (CREATE COMPLETE MODEL); we ask a random sampling during the partitioning in folds of the dataset for cross validation (SAMPLING TYPE = SHUFFLED SAMPLING).


RapidMiner@FUJITSU (validation croisee.xml*)
File Edit View Process Tools Help
Operator Tree
Root
Process
CSVExampleSource
CSVExampleSource
XValidation
XValidation
Parameters XML Comment New Operator
10:14:41 PM

keep_example_set	☐
create_complete_model	✓
average_performances_only	✓
leave_one_out	☐
number_of_validations	10
sampling_type	shuffled sampling
local_random_seed	-1

```
Nov 5, 2008 10:03:18 PM: [Error] Parameter 'filename' is not set and has no default value.
```

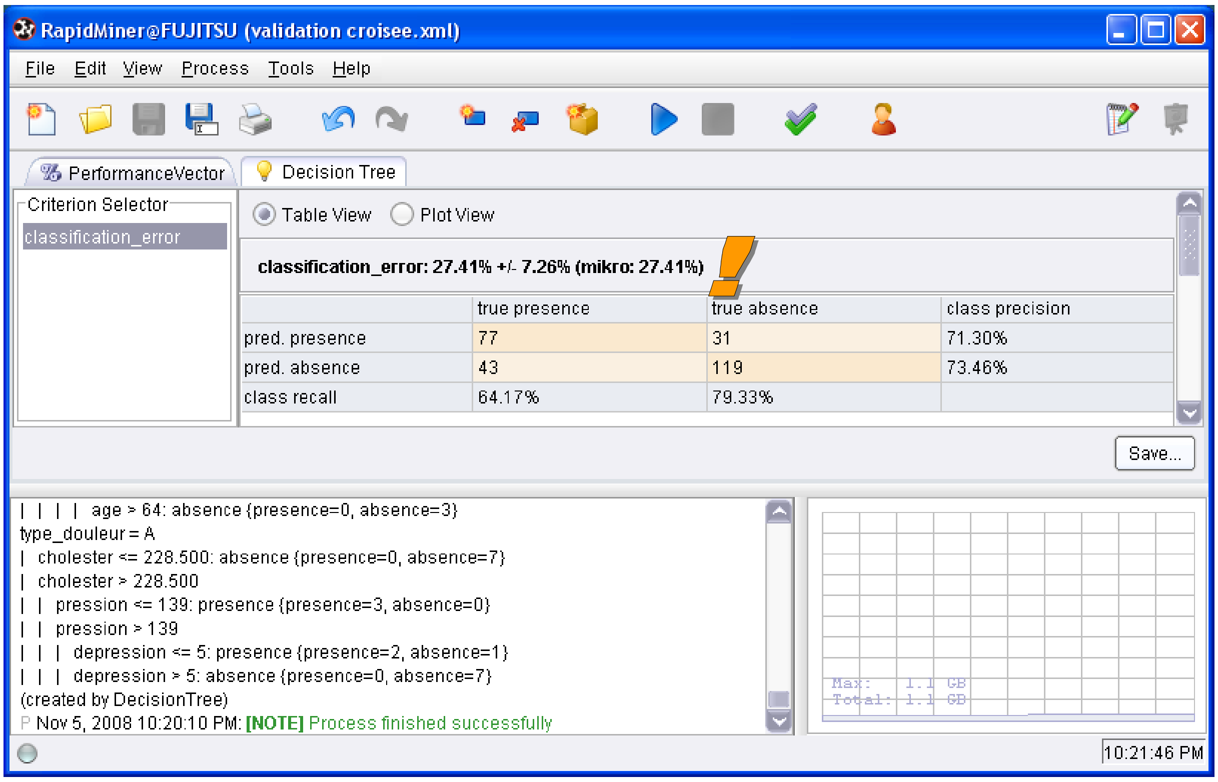
In this branch XVALIDATION, we must insert the learning tool (DECISION TREE) and the sequence "learning and test classifier". The diagram is as follows. We select the error rate as indicator of the classification performance.



keep_example_set	☐
main_criterion	first
accuracy	☐
classification_error	✓
kappa	☐
weighted_mean_recall	☐
weighted_mean_precision	☐
spearman_rho	☐
kendall_tau	☐
absolute_error	☐
relative_error	☐

```
Nov 5, 2008 10:03:18 PM: [Error] Parameter 'filename' is not set and has no default value.
```

Execution of the diagram. We can start the calculations by clicking on the PLAY button in the tool bar. RAPIDMINER gathers the results in a new window with tabs. The first available result is the confusion matrix and the cross validation estimation of the error rate. We have 27.41%.

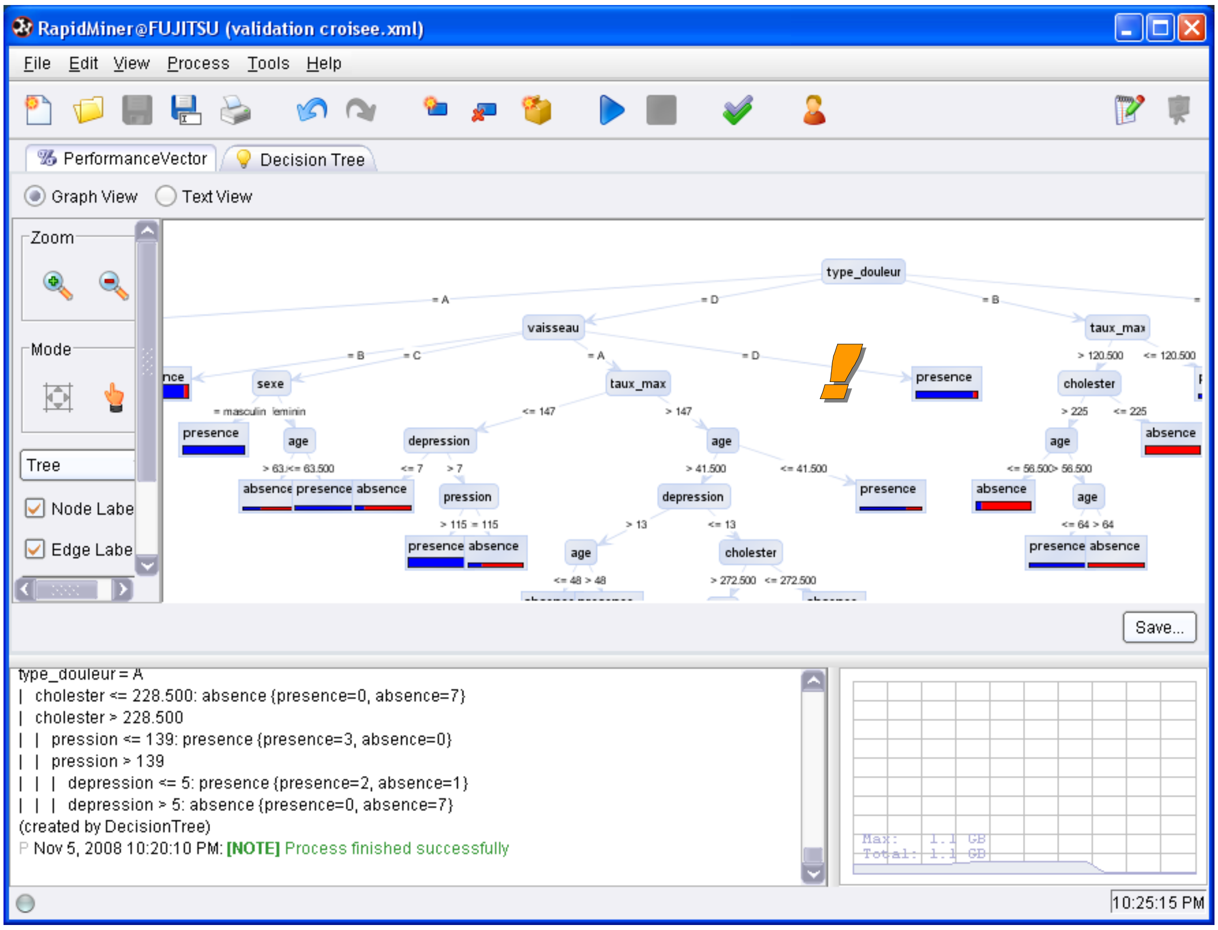


classification_error: 27.41% +/- 7.26% (mikro: 27.41%)

	true presence	true absence	class precision
pred. presence	77	31	71.30%
pred. absence	43	119	73.46%
class recall	64.17%	79.33%	

```
| | | |  age > 64: absence {presence=0, absence=3}
type_douleur = A
|  cholester <= 228.500: absence {presence=0, absence=7}
|  cholester > 228.500
| |  pression <= 139: presence {presence=3, absence=0}
| |  pression > 139
| | |  depression <= 5: presence {presence=2, absence=1}
| | |  depression > 5: absence {presence=0, absence=7}
(created by DecisionTree)
P Nov 5, 2008 10:20:10 PM: [NOTE] Process finished successfully
```

In the DECISION TREE tab, we observe the computed tree on the whole dataset.



```
type_douleur = A
|  cholester <= 228.500: absence {presence=0, absence=7}
|  cholester > 228.500
| |  pression <= 139: presence {presence=3, absence=0}
| |  pression > 139
| | |  depression <= 5: presence {presence=2, absence=1}
| | |  depression > 5: absence {presence=0, absence=7}
(created by DecisionTree)
P Nov 5, 2008 10:20:10 PM: [NOTE] Process finished successfully
```

5 Conclusion

In this tutorial, we showed how to implement the cross-validation evaluation process in three programs: R, KNIME and RAPIDMINER. This document fills up a previous tutorial where we create the same patterns of treatments with three other programs: TANAGRA, ORANGE and WEKA.