1 Topic

Deployment of predictive models using the PMML standard with PDI-CE (Pentaho Data Intergration Community Edition – Kettle).

Model deployment is a crucial task of the data mining process. In the supervised learning, it can be the applying of the predictive model on new unlabeled cases. We have already described this task for various tools (e.g. Tanagra, Sipina, Spad, R). They have as common feature the use of the same tool for the model construction and the model deployment.

In this tutorial, we describe a process where we do not use the same tool for the model construction and the model deployment. This is only possible if **(1)** the model is described in a standard format, **(2)** the tool which used for the deployment can handle both the database with unlabeled instances and the model. Here, we use the PMML standard description for the sharing of the model, and the PDI-CE for the applying of the model on the unseen cases.

The **Predictive Model Markup Language** (PMML) is an XML-based markup language to provide a way for applications to define models related to predictive analytics and data mining and to share those models between PMML-compliant applications. Proprietary issues and incompatibilities are no longer a barrier to the exchange of models between applications[1]. PMML is promoted by the Data Mining Group (DMG) which is an independent vendor led consortium that develops data mining standards[2]. **Pentaho Data Integration**, codenamed Kettle, consists of a core data integration (ETL) engine, and GUI applications that allow the user to define data integration jobs and transformations[3]. We have already described the Community version of this tool (PDI-CE)[4]. We have observed that it is a very convenient tool for the handling of dataset.

In this tutorial, we create a decision tree with various tools such as SIPINA, KNIME or RAPIDMINER; we export the model in the PMML format; then, we use PDI-CE for applying the model on a data file containing unlabeled instances. We see that the use of the PMML standard enhances dramatically the powerful of both the data mining tool and the ETL tool.

In addition, we will describe other solutions for deployment in this tutorial. We will see that Knime has its own PMML reader. It is able to apply a model on unlabeled datasets, whatever the tool used for the construction of the model. The key is that the PMML standard is respected. In this sense, Knime can be substituted to PDI-CE. Another possible solution, Weka, which is included into the Pentaho Community Edition suite, can export the model in a proprietary format that PDI-CE can handle.

2 Dataset

We use the heart[5] dataset in this tutorial. We want to predict the occurrence of the heart disease from the patient characteristics. "heart-train.arff" is to the training set used for the construction of

[1] http://en.wikipedia.org/wiki/Predictive_Model_Markup_Language

[2] http://www.dmg.org/

[3] http://en.wikipedia.org/wiki/Pentaho

[4] http://data-mining-tutorials.blogspot.fr/2012/04/pentaho-data-integration-kettle.html

[5] http://archive.ics.uci.edu/ml/datasets/Heart+Disease

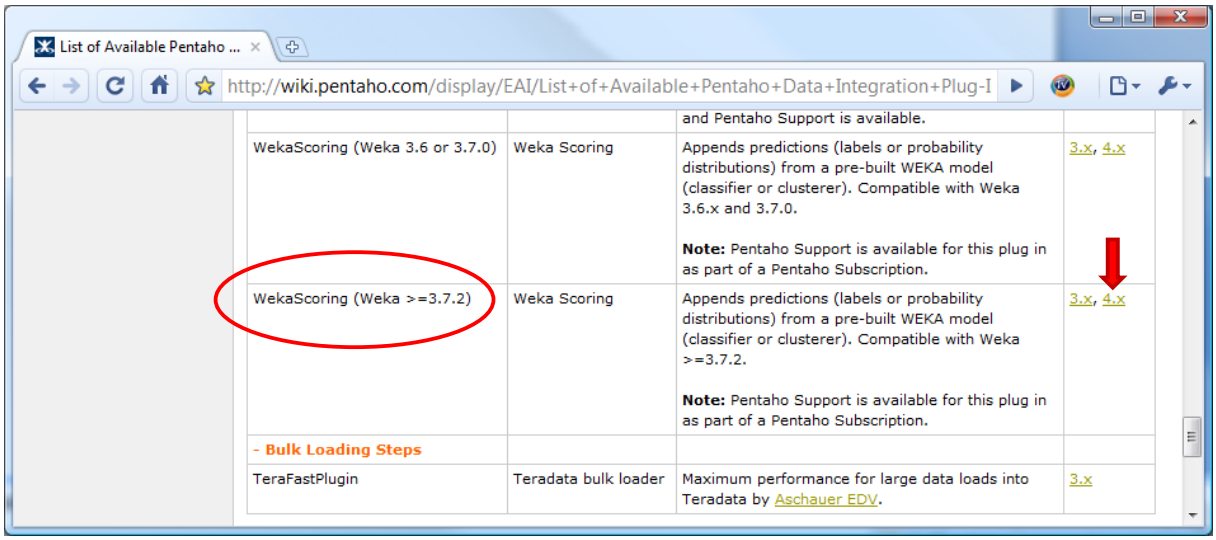
the model. "heart-unlabeled.txt" it is the data file containing the unlabeled instances. The goal of the process is to add the assigned class membership to this dataset using the predictive model.

3 Deployment with PDI-CE

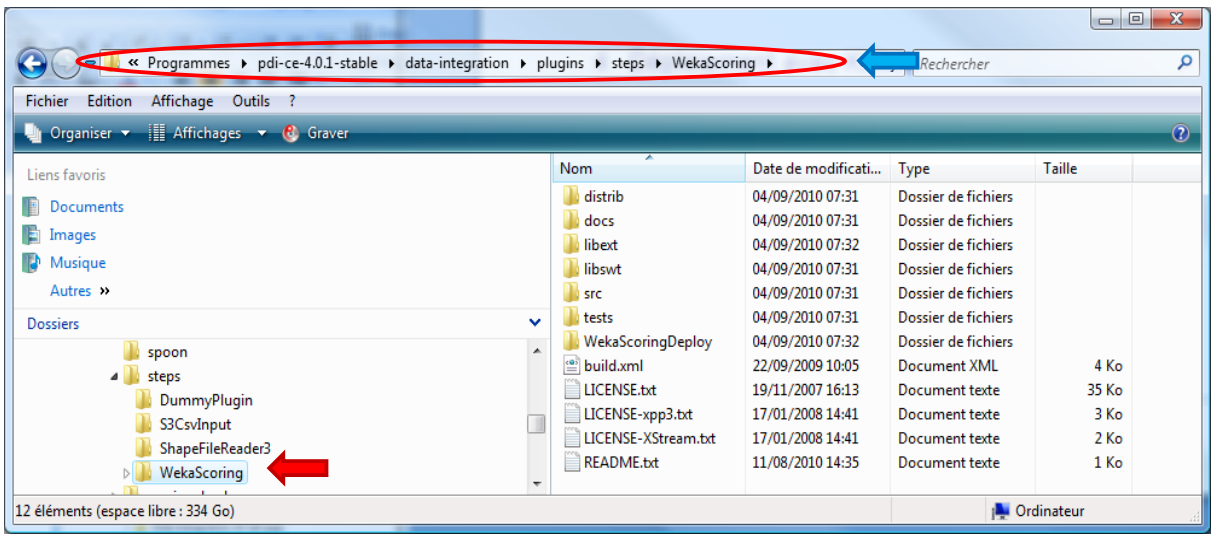
3.1 Installing PDI-CE and the "WekaScoring" plugin

We have described the installation and the use of PDI-CE in a previous tutorial[6]. For the model deployment, we must install the "WekaScoring" plugin. It enables to handle models produced by Weka in a proprietary format, but more generally it can also handle models in the PMML format whatever the tool used for the model creation.

First, we download the plugin from the Pentaho website. We must check carefully the versions.



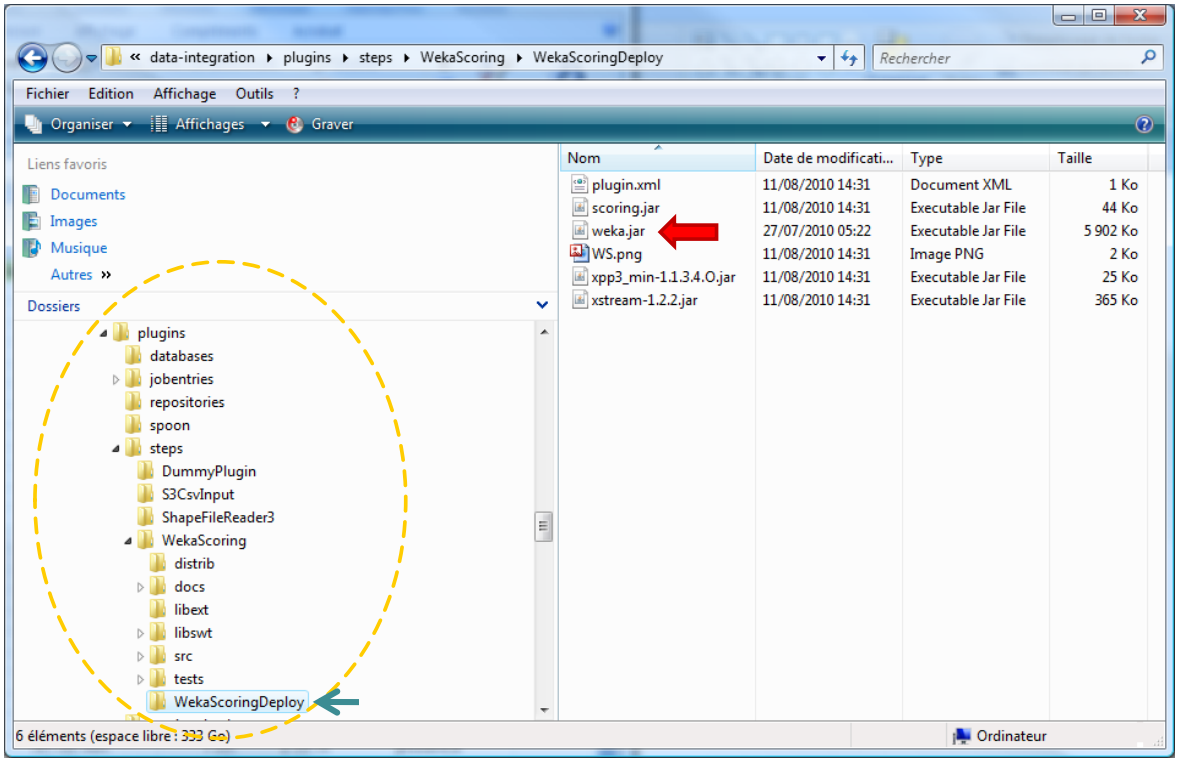
We unpack the file into the PDI-CE directory. We should have a configuration similar to below.



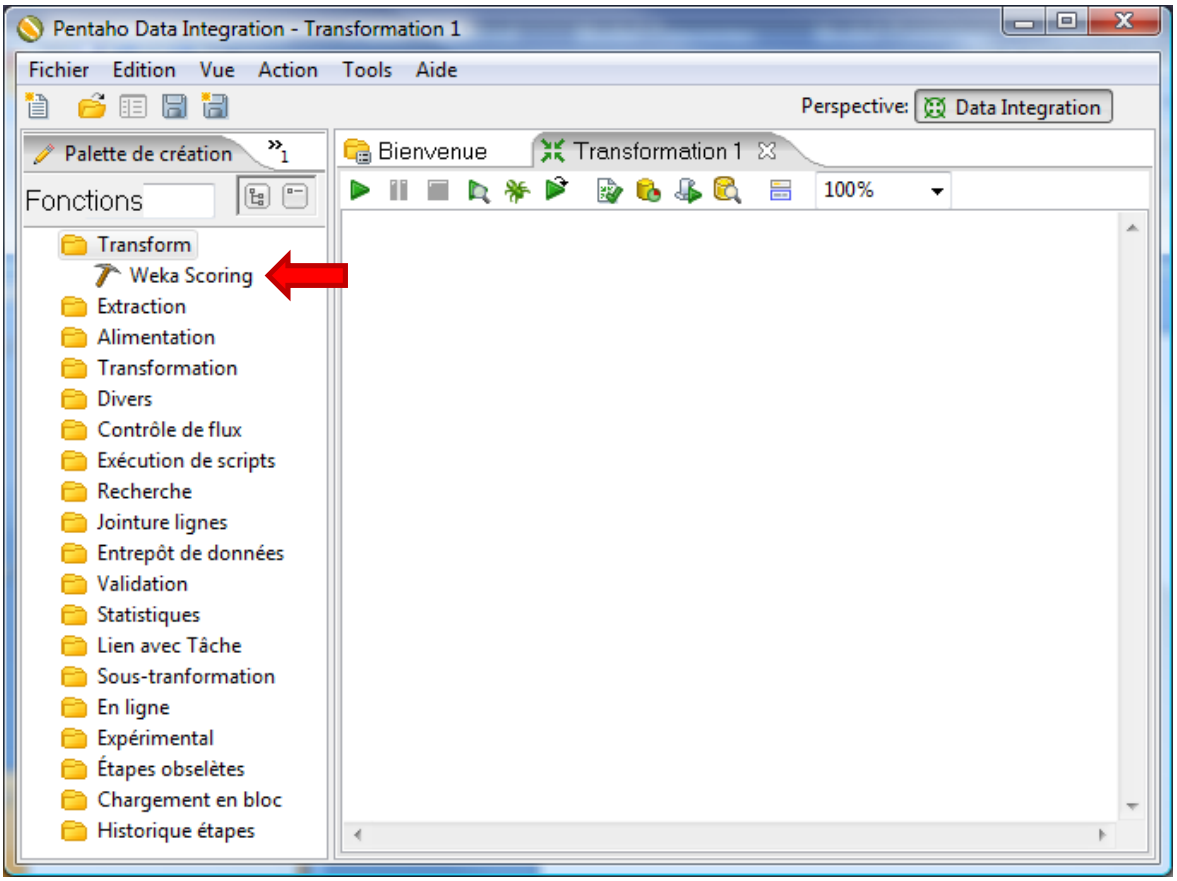
Then, we must copy the "**weka.jar**" file from the Weka distribution[7] (version 3.7.2 for this tutorial) into the sub-directory "**WekaScoringDeploy**".

[6] http://data-mining-tutorials.blogspot.fr/2012/04/pentaho-data-integration-kettle.html

[7] http://www.cs.waikato.ac.nz/ml/weka/



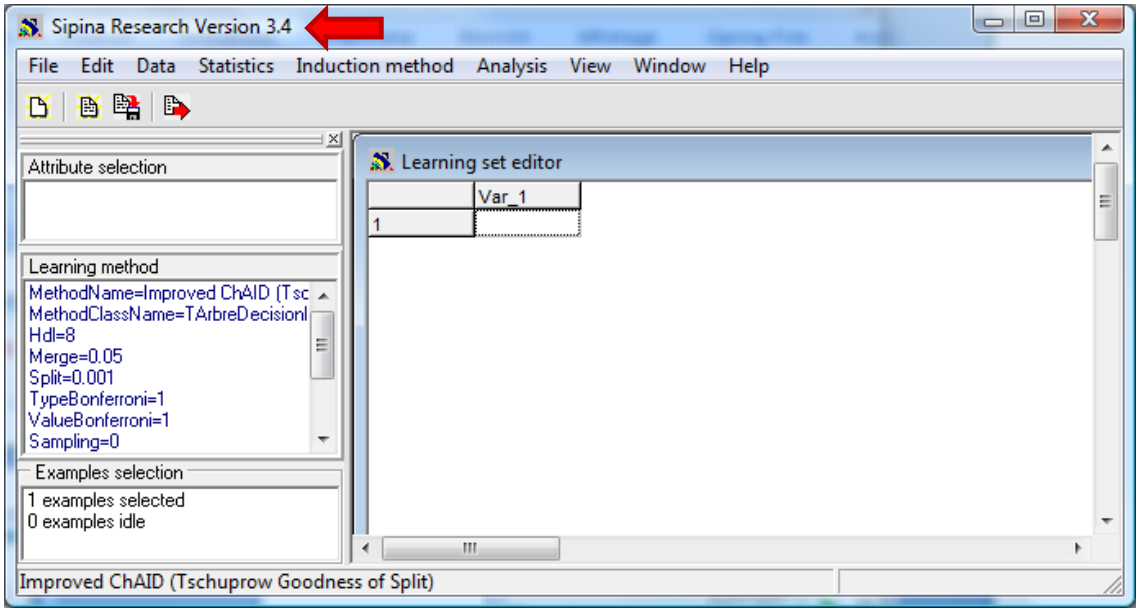
Last, to check the installation of the plugin, we launch PDI-CE. The WekaScoring icon must be visible into the TRANSFORM branch of the creation palette.



To perform the deployment with PDI-CE, we must learn a decision tree and save it in the PMML format. We use the SIPINA tool for that.

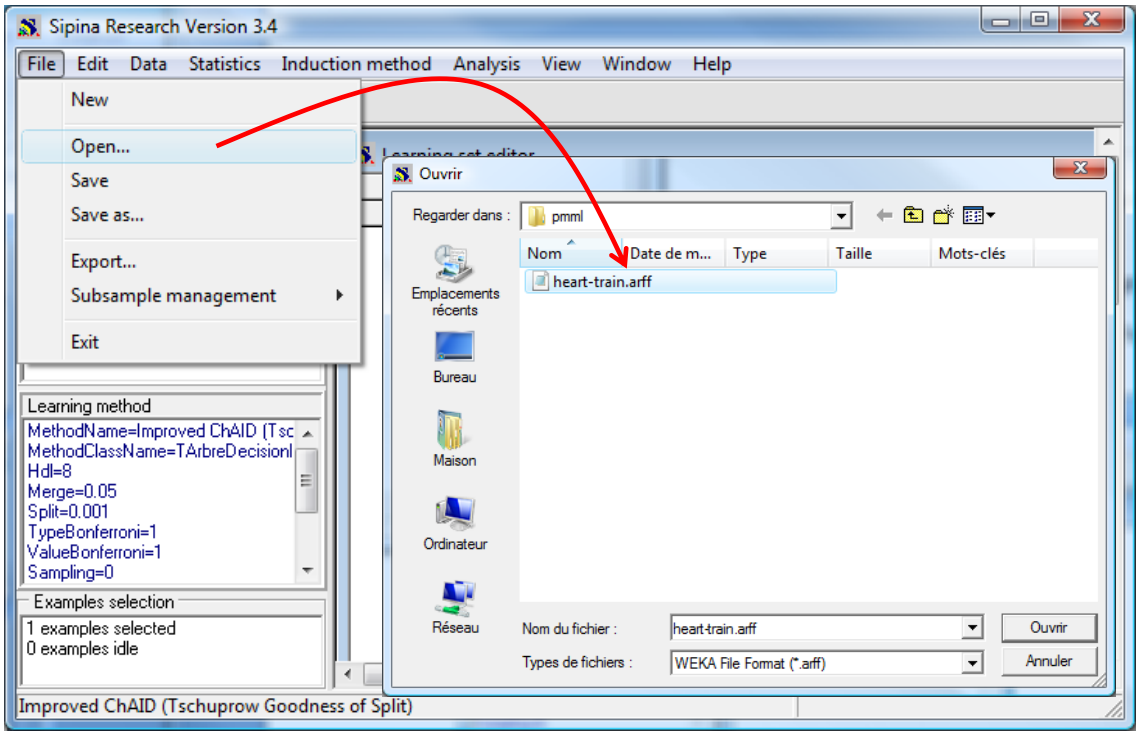
3.2 Learning and storing a model in the PMML format with SIPINA

We use SIPINA 3.4 or later version for this task.

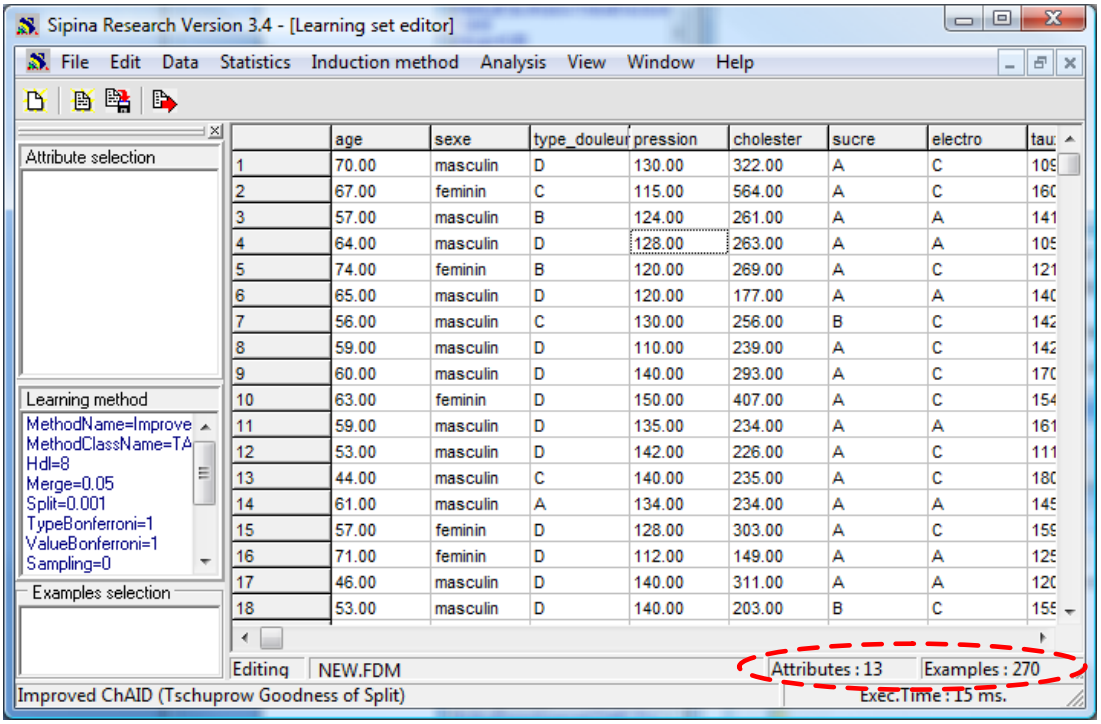


If we have not the right version, we can download the setup file on the Sipina website (http://eric.univ-lyon2.fr/~ricco/sipina.html).

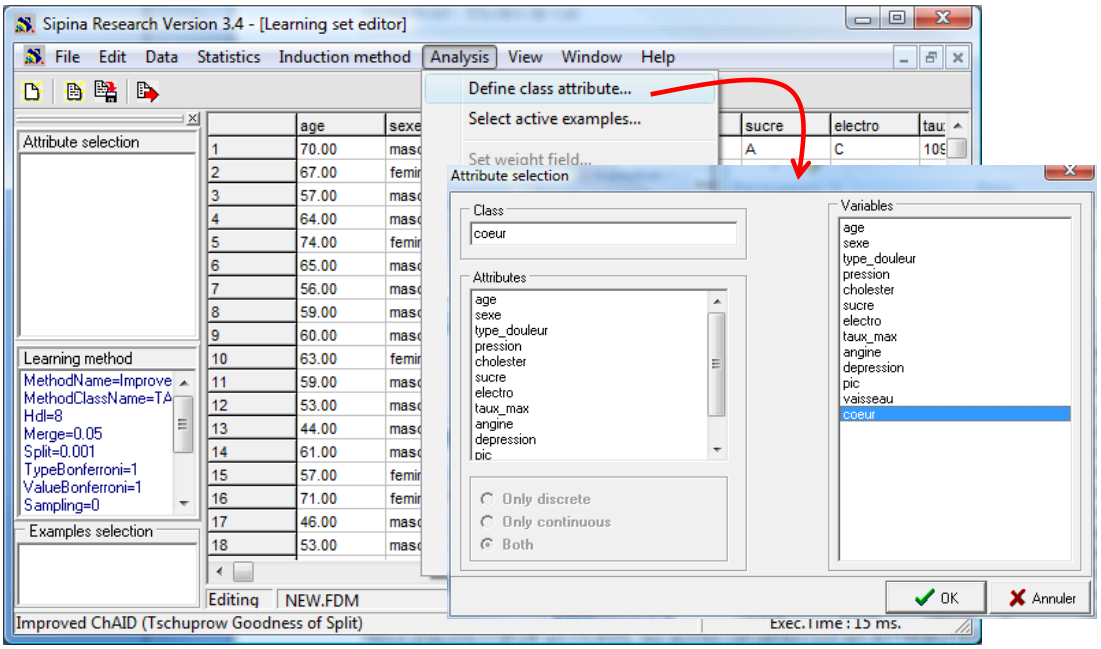
Importing the data file. After we launch SIPINA, we click on the FILE / OPEN to import the dataset. We pick the "heart-train.arff" data file.



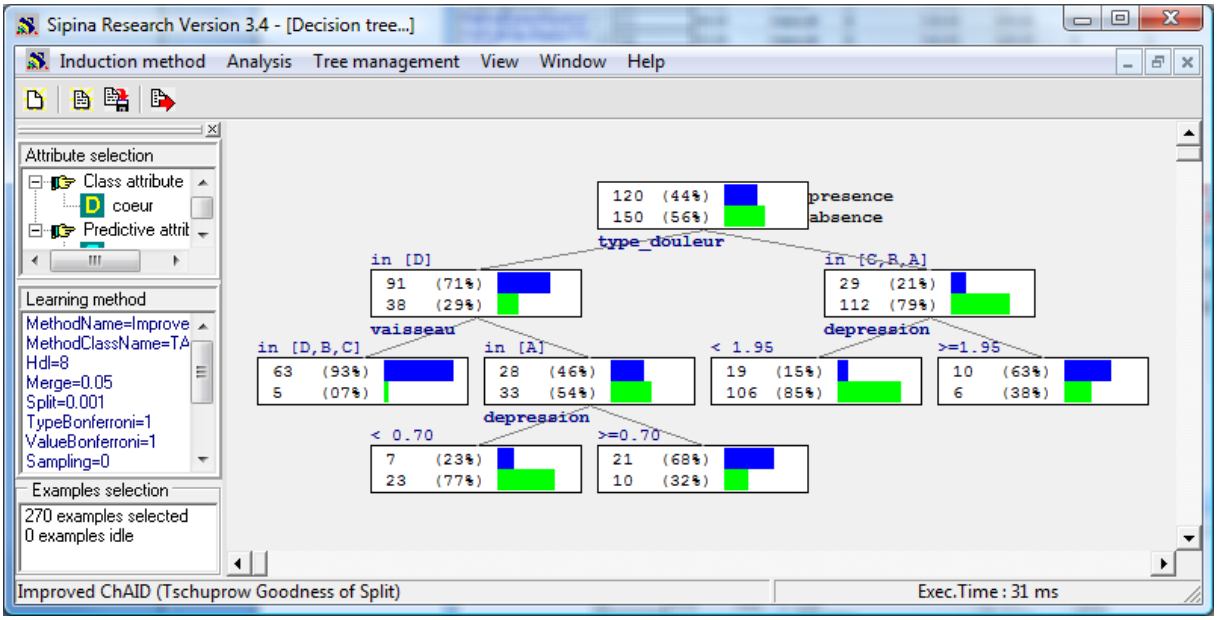
The dataset contains 13 variables and 270 instances.



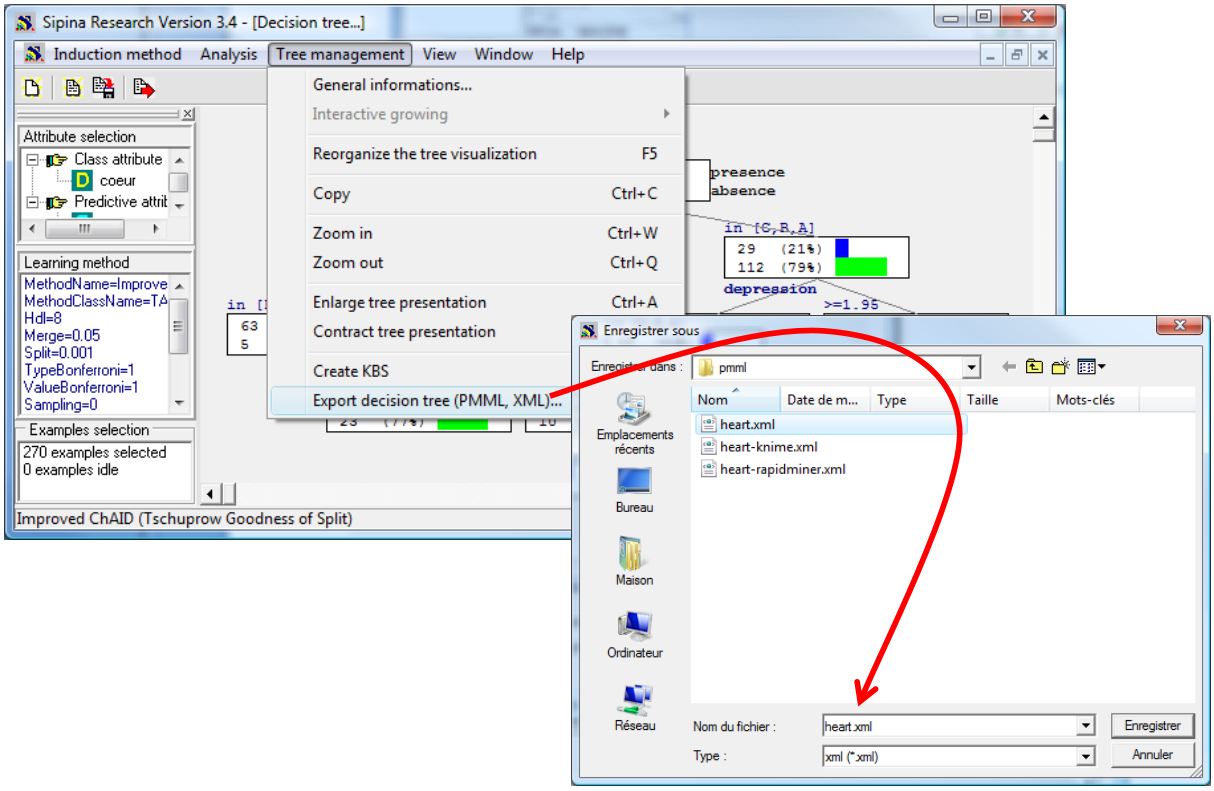
Decision tree learning. We define the role of the variables by clicking on the ANALYSIS / DEFINE CLASS ATTRIBUTE menu. We set COEUR as target variable, the others as input ones.



Then, we click on the ANALYSIS / LEARNING menu to launch the learning process. We obtain a rather simplistic model. Only 3 predictive variables among 12 are selected.



Exporting the tree in the PMML format. We want to export the decision tree in the PMML format. We click on the TREE MANAGEMENT / EXPORT DECISION TREE menu. We set the file name "heart.xml".



Note : Choosing the xml extension seems more appropriate for PDI-CE. In fact, both ".xml" and ".pmml" extensions refer to the PMML file format.

Understanding the PMML format. The "heart.xml" file is subdivided in several parts: (A) the data dictionary; (B) the classification tree, including the "Mining Schema" which describes the role of the variables (class and predictive attributes).

- About the data dictionary (Figure 1), we observe for instance that "sex" is a discrete attribute with two values "masculin" and 'feminin" (in French). On the other hand, we observe that "pression" is a continuous attribute. The minimum (resp. maximum) value "leftMargin" ("rightMargin") is 94 (200). We use the scientific notation.
- Into the mining schema (Figure 2), we observe that "coeur" is the target attribute.
- Last, we have the description of the decision tree (Figure 3). We note the correspondence between the graphical representation of the tree and the PMML description. For each node, we observe the number of instances covered by the node, the rule associated to the node, the number of instances corresponding to each class value.

```
<DataDictionary numberOfFields="13">
<DataField name="age" optype="continuous" dataType="double">
<Interval closure="closedClosed" leftMargin="2.90000000000000E+001" rightMargin="7.70000000000000E+001"/>
</DataField>
<DataField name="sexe" optype="categorical" dataType="string">
<Value value="masculin"/>
<Value value="feminin"/>
</DataField>
<DataField name="type_douleur" optype="categorical" dataType="string">
<Value value="D"/>
<Value value="C"/>
<Value value="B"/>
<Value value="A"/>
</DataField>
<DataField name="pression" optype="continuous" dataType="double">
<Interval closure="closedClosed" leftMargin="9.40000000000000E+001" rightMargin="2.00000000000000E+002"/>
</DataField>
<DataField name="cholester" optype="continuous" dataType="double">
<Interval closure="closedClosed" leftMargin="1.26000000000000E+002" rightMargin="5.64000000000000E+002"/>
</DataField>
<DataField name="sucre" optype="categorical" dataType="string">
<Value value="A"/>
<Value value="B"/>
</DataField>
<DataField name="electro" optype="categorical" dataType="string">
<Value value="C"/>
<Value value="A"/>
<Value value="B"/>
</DataField>
<DataField name="taux_max" optype="continuous" dataType="double">
<Interval closure="closedClosed" leftMargin="7.10000000000000E+001" rightMargin="2.02000000000000E+002"/>
</DataField>
<DataField name="angine" optype="categorical" dataType="string">
<Value value="non"/>
<Value value="oui"/>
</DataField>
<DataField name="depression" optype="continuous" dataType="double">
<Interval closure="closedClosed" leftMargin="0.00000000000000E+000" rightMargin="6.19999980926514E+000"/>
</DataField>
<DataField name="pic" optype="continuous" dataType="double">
<Interval closure="closedClosed" leftMargin="1.00000000000000E+000" rightMargin="3.00000000000000E+000"/>
</DataField>
<DataField name="vaisseau" optype="categorical" dataType="string">
<Value value="D"/>
<Value value="A"/>
<Value value="B"/>
<Value value="C"/>
</DataField>
<DataField name="coeur" optype="categorical" dataType="string">
<Value value="presence"/>
<Value value="absence"/>
</DataField>
</DataDictionary>
```

Figure 1 - PMML – Data dictionary

```
<TreeModel modelName="DecisionTree" functionName="classification" s
<MiningSchema>
<MiningField name="age"/>
<MiningField name="sexe"/>
<MiningField name="type_douleur"/>
<MiningField name="pression"/>
<MiningField name="cholester"/>
<MiningField name="sucre"/>
<MiningField name="electro"/>
<MiningField name="taux_max"/>
<MiningField name="angine"/>
<MiningField name="depression"/>
<MiningField name="pic"/>
<MiningField name="vaisseau"/>
<MiningField name="coeur" usageType="predicted"/>
</MiningSchema>
```

Figure 2 - Mining Schema

```
<Node id="0" score="absence" recordCount="270.0">
<True/>
<ScoreDistribution value="presence" recordCount="120.0"/>
<ScoreDistribution value="absence" recordCount="150.0"/>
<Node id="1" score="presence" recordCount="129.0">
<SimpleSetPredicate field="type_douleur" booleanOperator="isIn">
<Array n="1" type="string">"D" </Array>
</SimpleSetPredicate>
<ScoreDistribution value="presence" recordCount="91.0"/>
<ScoreDistribution value="absence" recordCount="38.0"/>
<Node id="2" score="presence" recordCount="68.0">
<SimpleSetPredicate field="vaisseau" booleanOperator="isIn">
<Array n="3" type="string">"D" "B" "C" </Array>
</SimpleSetPredicate>
<ScoreDistribution value="presence" recordCount="63.0"/>
<ScoreDistribution value="absence" recordCount="5.0"/>
</Node>
<Node id="3" score="absence" recordCount="61.0">
<SimpleSetPredicate field="vaisseau" booleanOperator="isIn">
<Array n="1" type="string">"A" </Array>
</SimpleSetPredicate>
<ScoreDistribution value="presence" recordCount="28.0"/>
<ScoreDistribution value="absence" recordCount="33.0"/>
<Node id="4" score="absence" recordCount="30.0">
<SimplePredicate field="depression" operator="lessThan" value="7.00000047683716E-001"/>
<ScoreDistribution value="presence" recordCount="7.0"/>
<ScoreDistribution value="absence" recordCount="23.0"/>
</Node>
<Node id="5" score="presence" recordCount="31.0">
<SimplePredicate field="depression" operator="greaterOrEqual" value="7.00000047683716E-001"/>
<ScoreDistribution value="presence" recordCount="21.0"/>
<ScoreDistribution value="absence" recordCount="10.0"/>
</Node>
</Node>
</Node>
<Node id="6" score="absence" recordCount="141.0">
<SimpleSetPredicate field="type_douleur" booleanOperator="isIn">
<Array n="3" type="string">"C" "B" "A" </Array>
</SimpleSetPredicate>
<ScoreDistribution value="presence" recordCount="29.0"/>
<ScoreDistribution value="absence" recordCount="112.0"/>
<Node id="7" score="absence" recordCount="125.0">
<SimplePredicate field="depression" operator="lessThan" value="1.95000004768372E+000"/>
<ScoreDistribution value="presence" recordCount="19.0"/>
<ScoreDistribution value="absence" recordCount="106.0"/>
</Node>
<Node id="8" score="presence" recordCount="16.0">
<SimplePredicate field="depression" operator="greaterOrEqual" value="1.95000004768372E+000"/>
<ScoreDistribution value="presence" recordCount="10.0"/>
<ScoreDistribution value="absence" recordCount="6.0"/>
</Node>
</Node>
</Node>
</TreeModel>
```

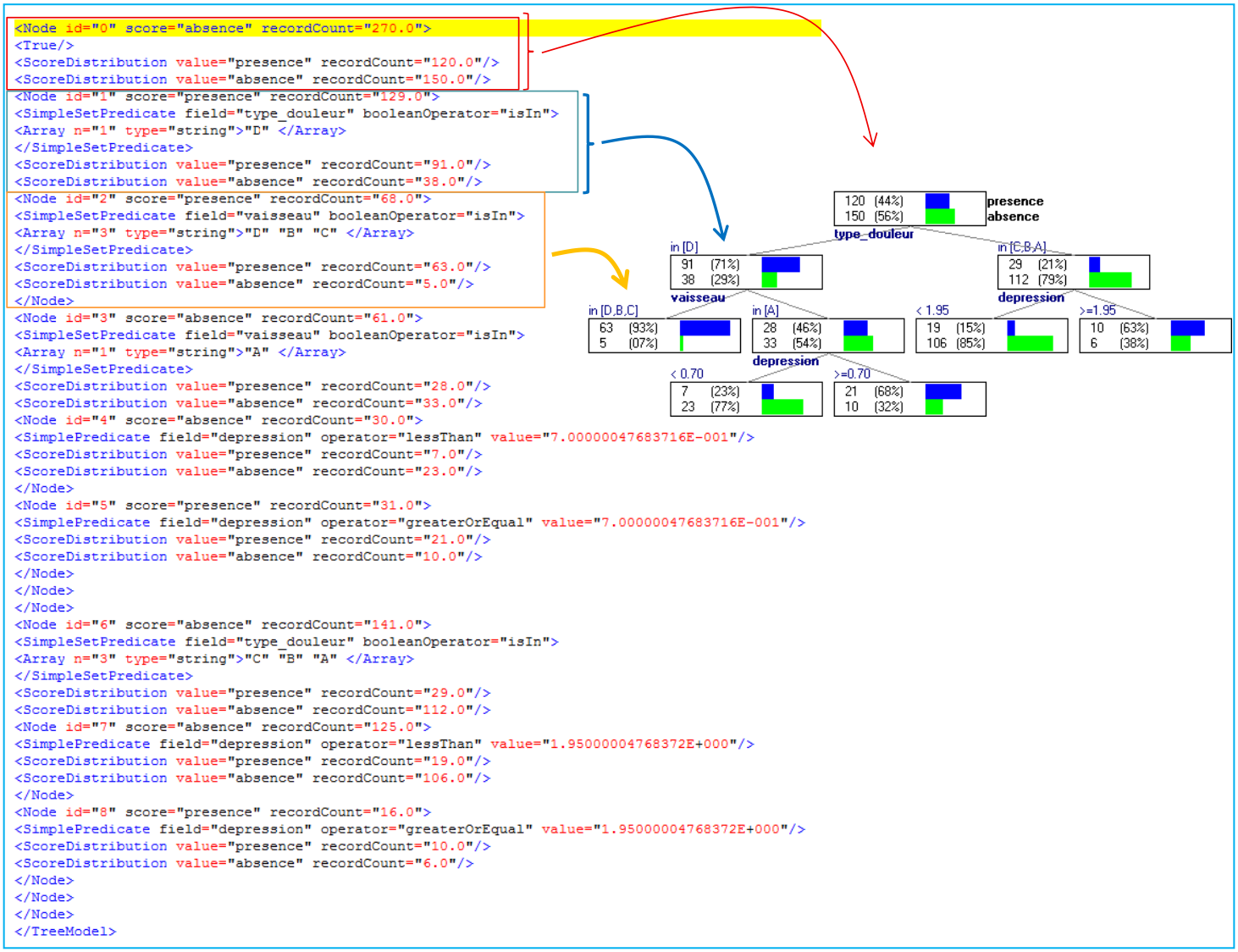
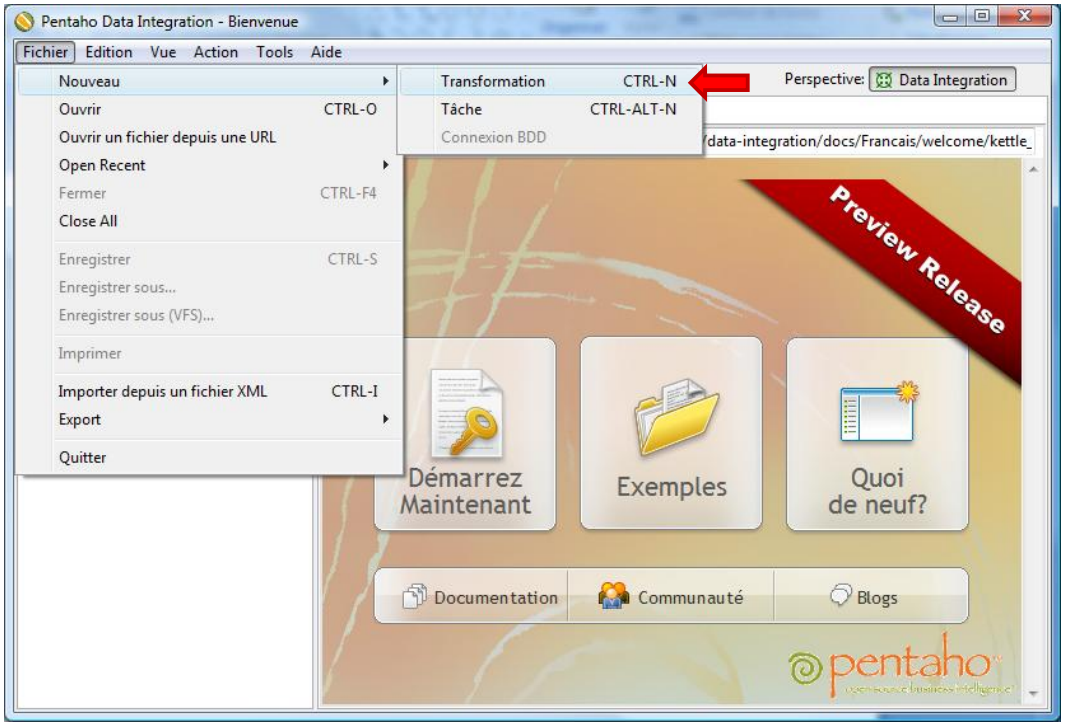


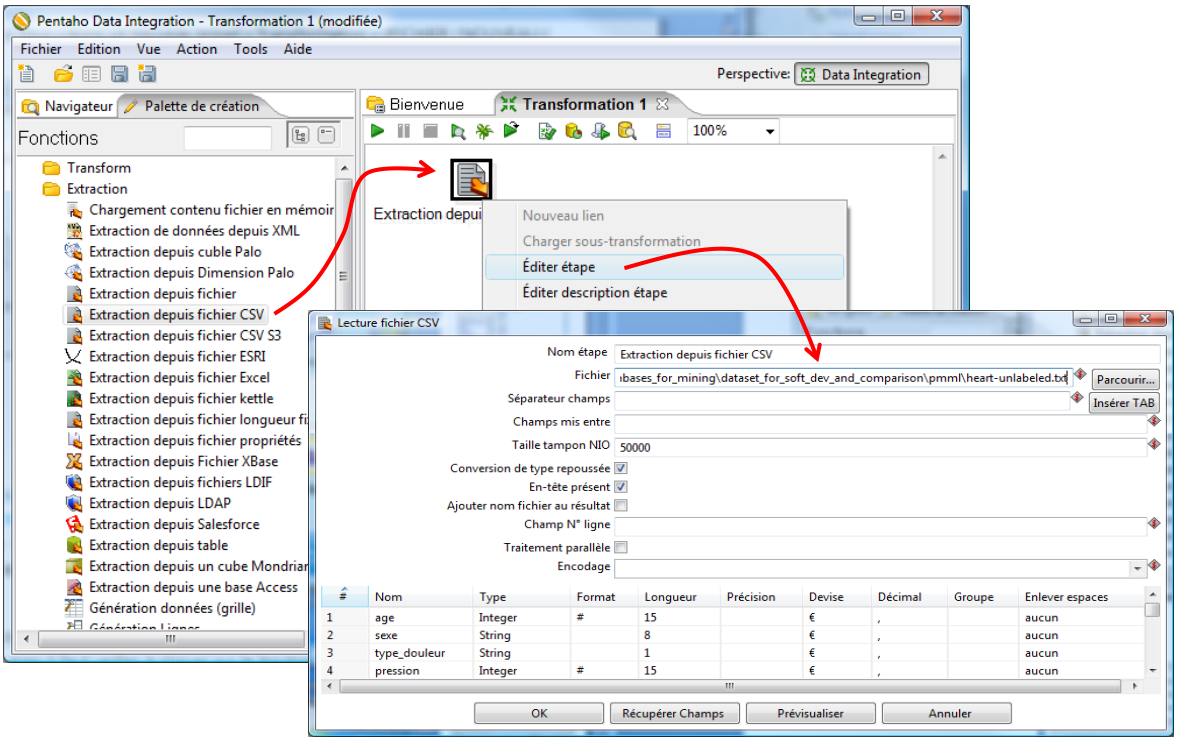
Figure 3- Tree description

3.3 Deployment using PDI-CE and the WekaScoring plugin

We want to use this tree to label the instances from the "**heart-unlabeled.txt**" data file. We start PDI-CE. We create a new transformation project "FICHIER / NOUVEAU / TRANSFORMATION".

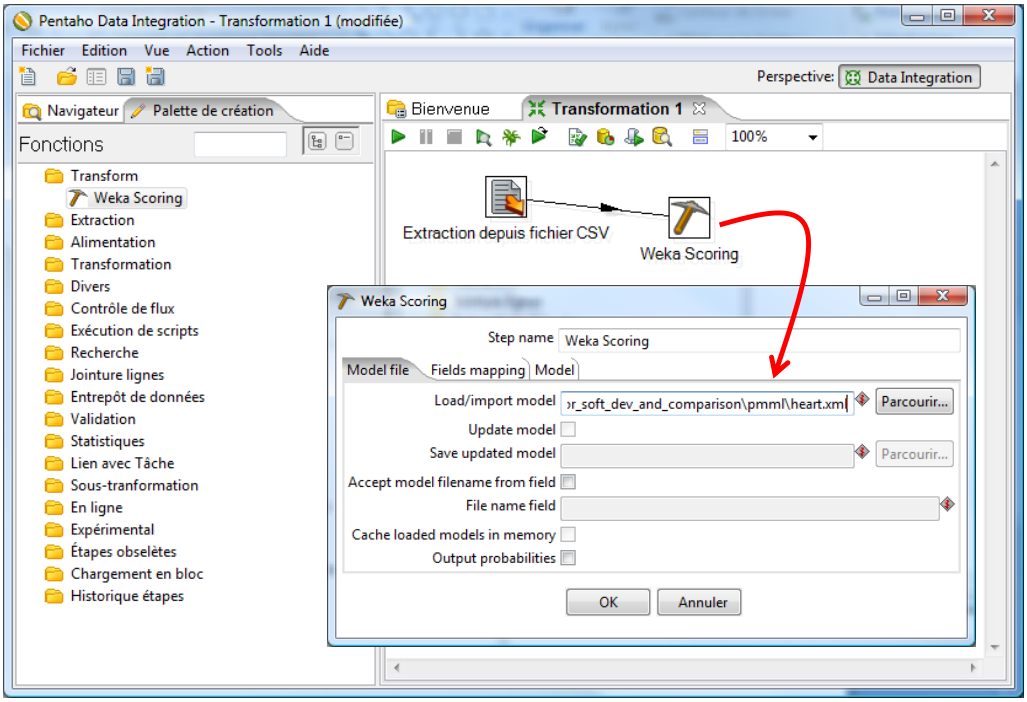


Importing the unlabeled instances. We import the data file with unlabeled instance using the "**Extraction depuis fichier CSV**" component. We set the following parameters:

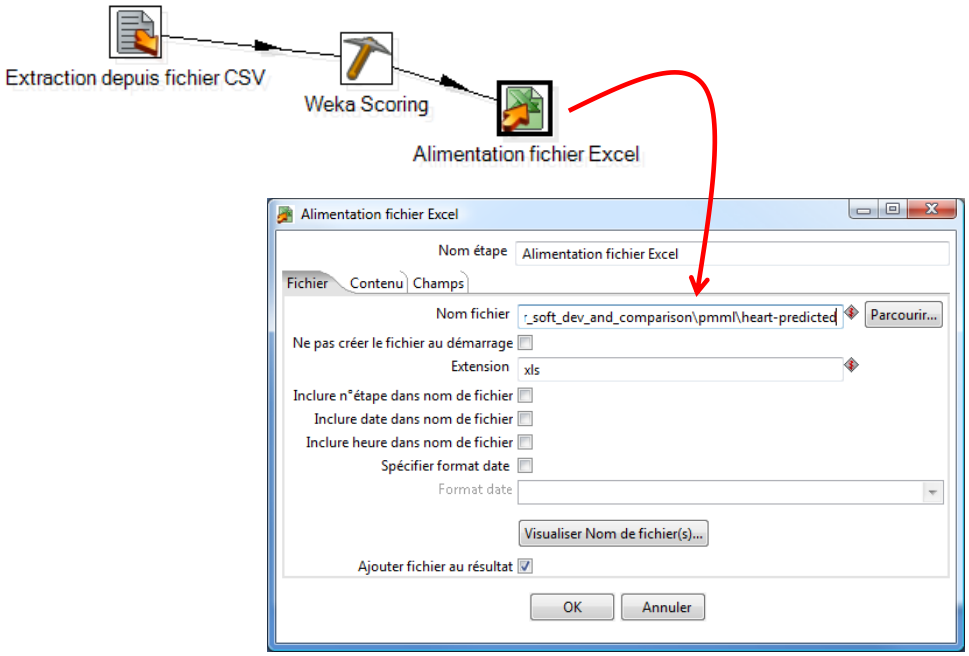


The tabulation character is the column separator. We click on the "Recupérer champs" button to obtain the columns description.

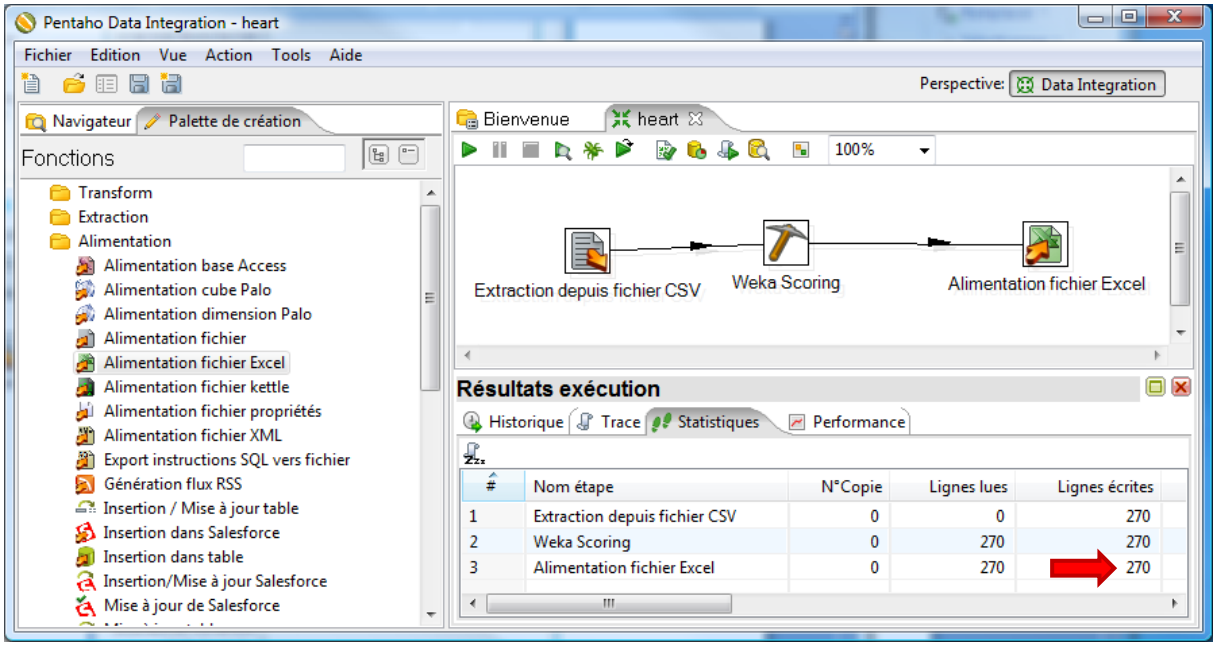
Labelling with the WekaScoring component. We add the **WekaScoring** component into the workflow. We set the model file name "heart.xml" into the Settings dialog.



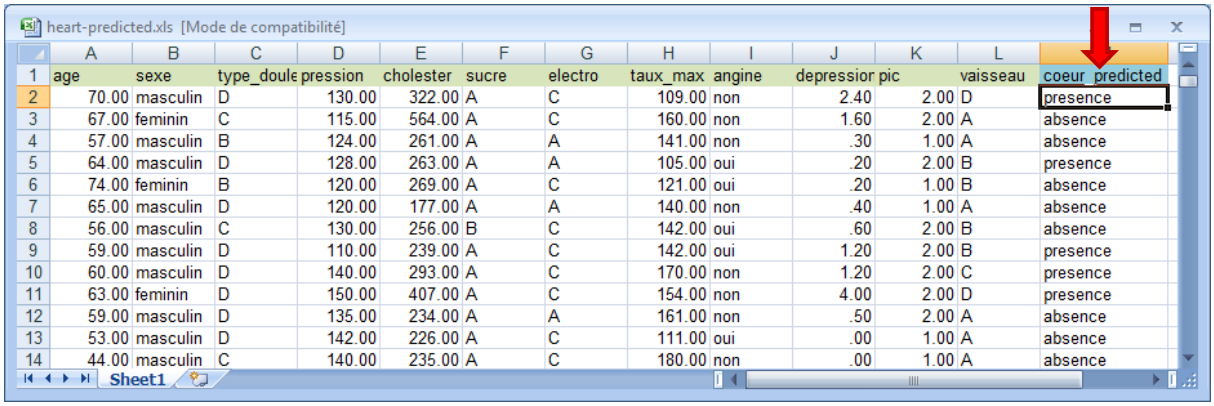
Creating the output file. We want to create the output file containing the values of the predictive variables and the assigned class value in the Excel file format. We add the "**Alimentation Fichier Excel**" into the workflow. We set the output file name "heart-predicted.xls".



Launching the process. We launch the process by clicking on the ACTION / EXECUTER menu (F9).



We obtain the following data file. The predicted values are in the last column.



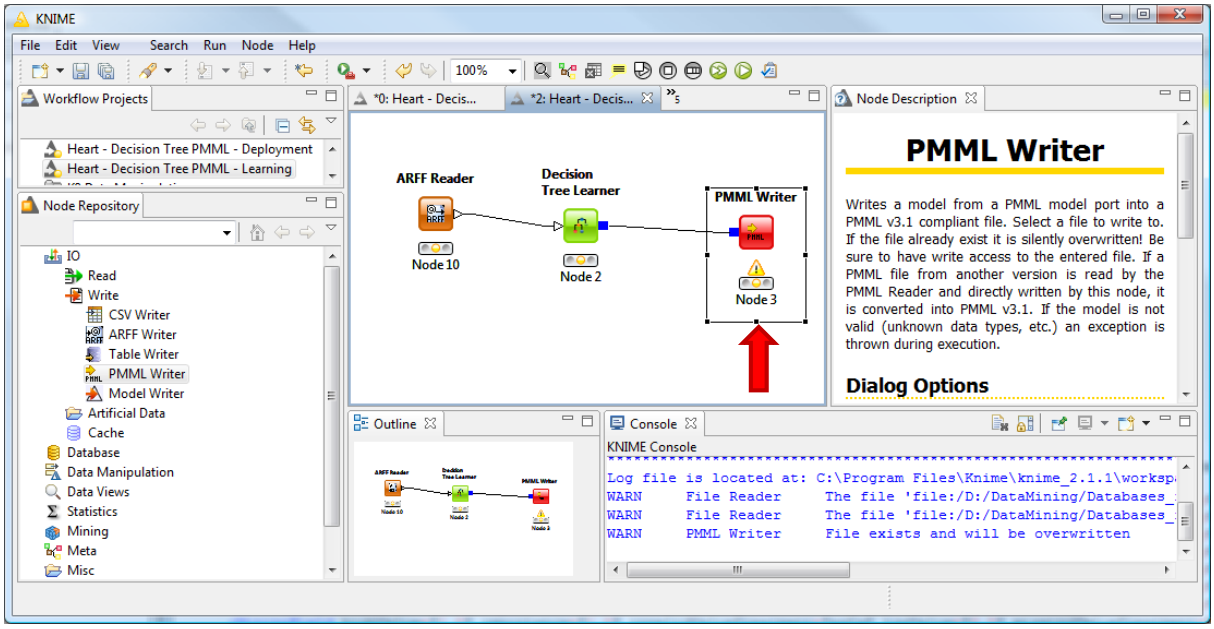
For instance, the value "presence" is assigned to the first individual, "absence" for the second individual, etc.

3.4 Exporting PMML from other tools

We have built our model with SIPINA above. In the two subsections that follow, we show that it is possible to perform the same approach with other software. The key is that they can correctly export models in accordance with the PMML format.

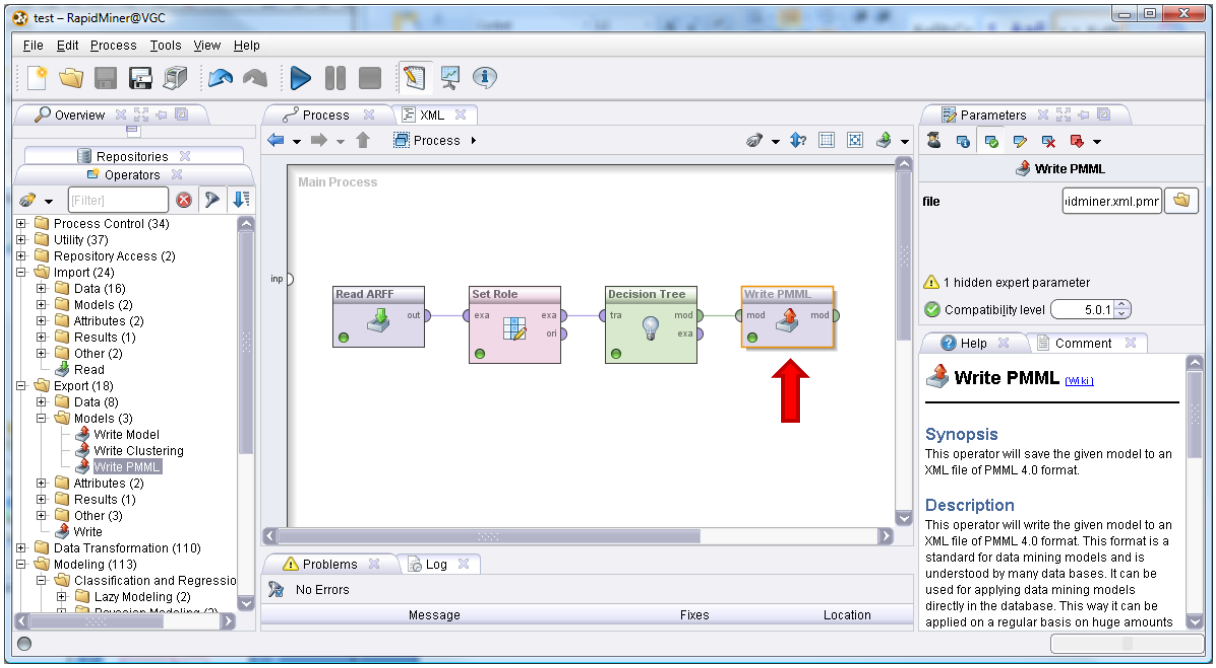
3.4.1 Exporting PMML from Knime

We have defined the following learning process under Knime. The PMML WRITER component is used to export the predictive model.



3.4.2 Exporting PMLL from RapidMiner

In the same way, we define the same process under RapidMiner (5.0.010). The WRITE PMML component enables to export the classification tree in the PMML format.

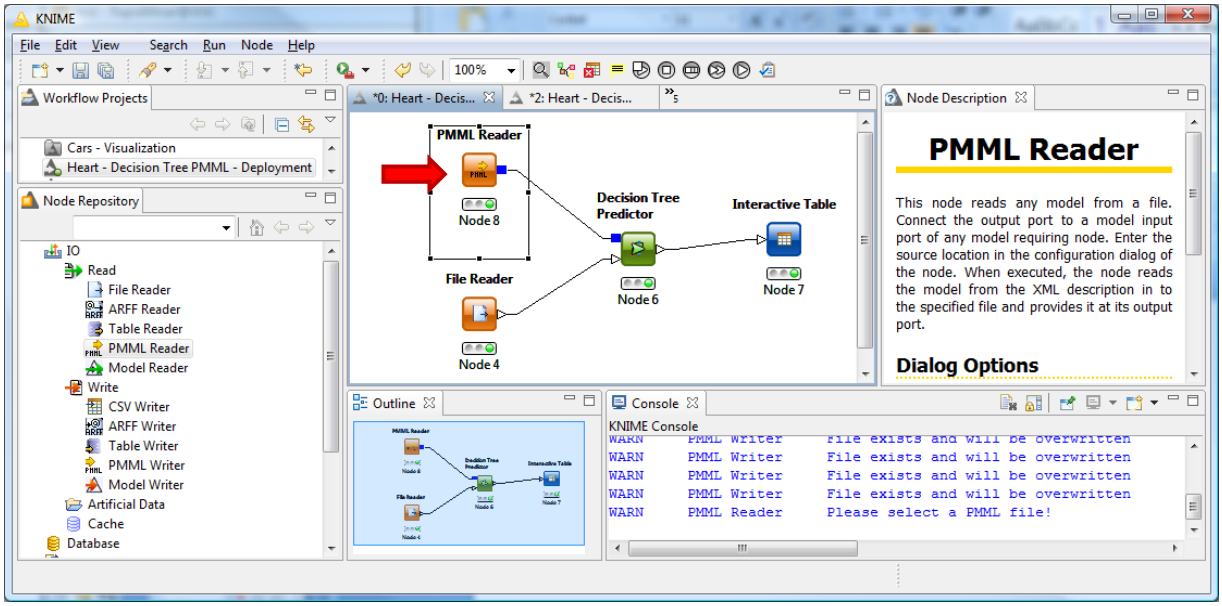


4 Model deployment with Knime

PDI-CE is intended for the data management, its use in the model deployment is therefore self-evident. But other tools can do it also, as long as they are able to manage the data files and to correctly interpret the models. The adoption of the PMML standard is a valuable asset in this process.

We show briefly how to deploy a model with Knime here. The model is the one created with SIPINA previously. We define the following workflow. We notice the two sources of information: FILE READER for the data file containing the unlabeled instances; PMML READER for the model in the

PMML format. We apply the model on the dataset using the DECISION TREE PREDICTOR. We visualize the instances including the assigned labels with the INTERACTIVE TABLE component.

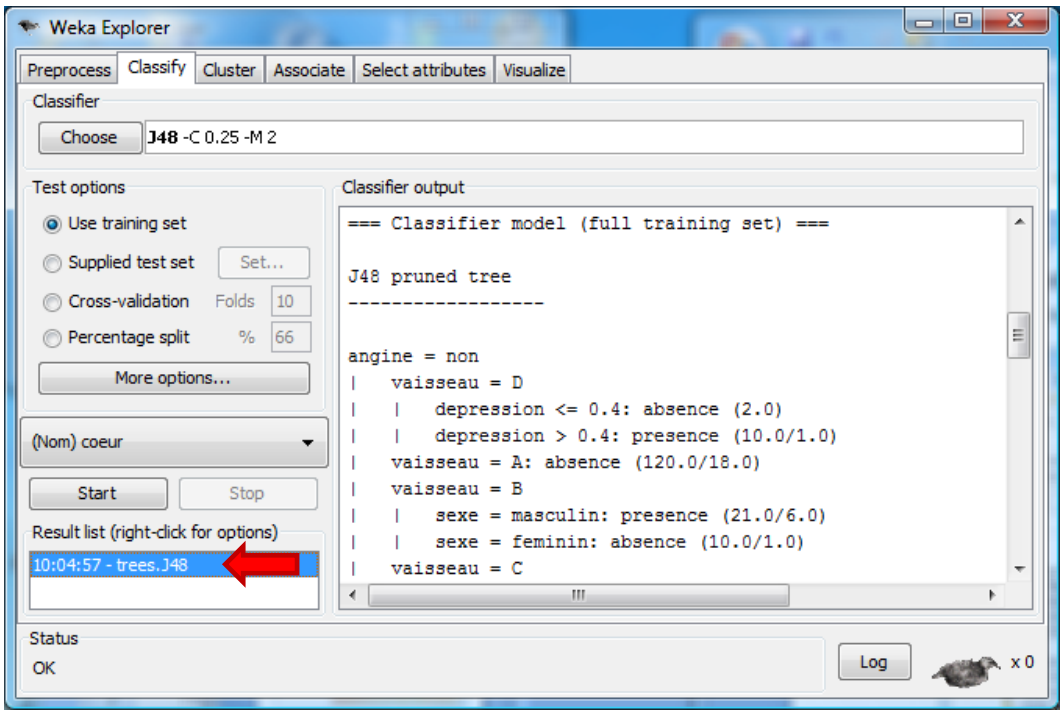


5 Weka model file and PDI-CE

Both Weka and PDI-CE belong to the "Pentaho Community Edition" suite. They can communicate via a proprietary (and binary) file format.

5.1 Creating and storing a model with Weka

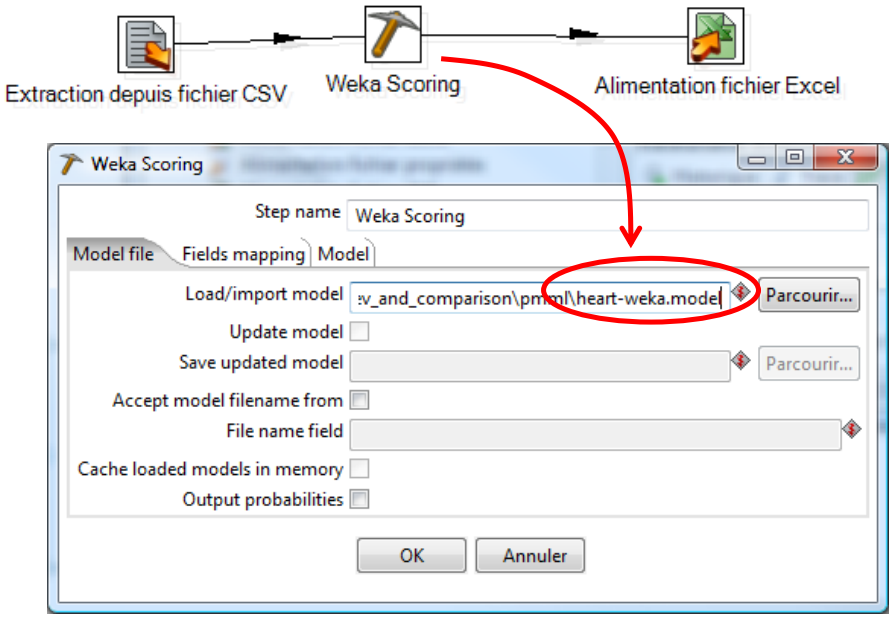
We use the Weka 3.7.2 version in this section. The WekaScoring version must be compatible. After we launch Weka in the "Explorer" mode, we load the "heart-train.arff" data file. We activate the "Classify" tab and we select the J48 induction tree algorithm. We start the learning process by clicking on the START button.



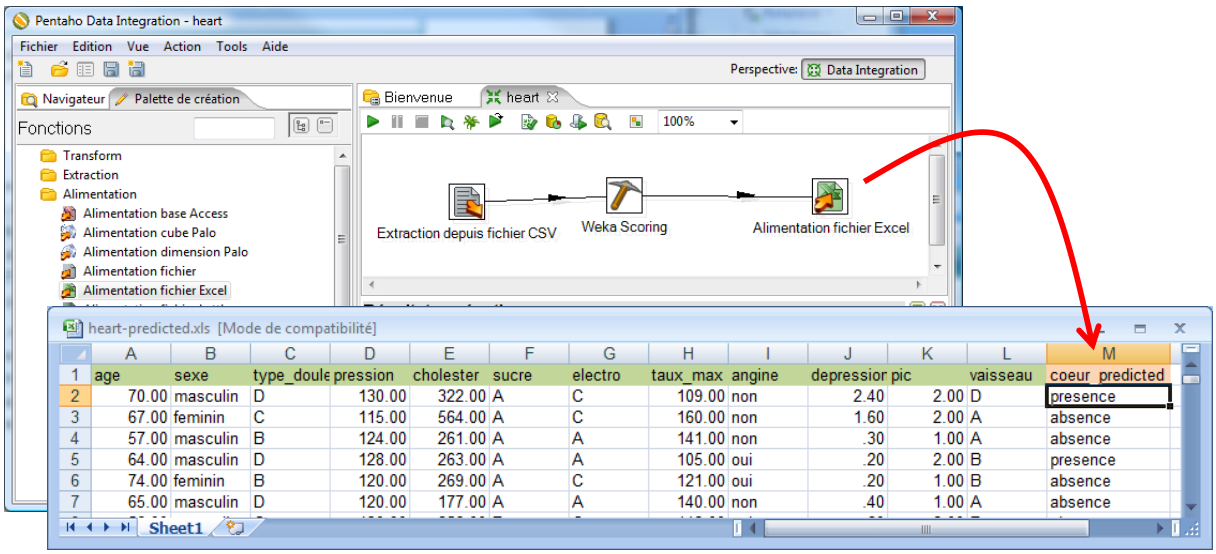
To store the predictive model, we right-click on the model and we select the SAVE MODEL menu item. We set "**heart-weka.model**" as file name.

5.2 Deployment under PDI-CE

We are back in PDI-CE. We use the same diagram as above, except that we change the model. Instead of the PMML file from SIPINA, we select the "heart-weka.model" file created by Weka.



We launch again the process (F9). We obtain a new version of the "heart-predicted.xls" output file.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	age	sexe	type_doule	pression	cholester	sucre	electro	taux_max	angine	depressior	pic	vaisseau	coeur_predicted
2	70.00	masculin	D	130.00	322.00	A	C	109.00	non	2.40	2.00	D	presence
3	67.00	feminin	C	115.00	564.00	A	C	160.00	non	1.60	2.00	A	absence
4	57.00	masculin	B	124.00	261.00	A	A	141.00	non	.30	1.00	A	absence
5	64.00	masculin	D	128.00	263.00	A	A	105.00	oui	.20	2.00	B	presence
6	74.00	feminin	B	120.00	269.00	A	C	121.00	oui	.20	1.00	B	absence
7	65.00	masculin	D	120.00	177.00	A	A	140.00	non	.40	1.00	A	absence

Note: Interestingly, we have compared the predictions of SIPINA and WEKA. We observe that they are inconsistent for 43 cases (18 +25) about 270.

		WEKA		
	Étiquettes de	presence	absence	Total général
SIPINA	presence	90	25	115
	absence	18	137	155
	Total général	**108**	**162**	**270**

6 Conclusion

The deployment of the models is the last step of the data mining process in many situations. In the case of supervised learning, we want to assign labels to unseen instances using the model learning from the labeled dataset. We emphasize the PMML format because it tries to establish a standard for the model description. If this standard is recognized by the most of the tools, it becomes really powerful. In this tutorial, we show that we can create the model using a data mining tool such as SIPINA (or Knime, RapidMiner...) and deploy it using PDI-CE, a tool intended for the data management and the data transformation.

Of course, PMML is not the unique format for the models description. When the tools are included in the same software suite, we can use a proprietary format (e.g. Weka and PDI-CE). We expect that the performances (computation time mainly) are better in this case. It would be interesting to compare the performances (PMML vs. proprietary format) in an upcoming tutorial.