1 Introduction

Applying a predictive model on new unlabeled sample. Estimation of the generalization error rate by resampling approach.

Model deployment. Model deployment is the last step of the Data Mining process. In its simplest form in a supervised learning task, it consists in to apply a predictive model on unlabeled cases. Most of the tools have this kind of capability. The other alternatives are programming a specific tool (in Java for instance, but it can be very costly if we must repeatedly implement models in various contexts), using SQL queries (especially useful when the model can be expressed in a set of rules), or using the PMML framework[1].

Sipina can apply a learned classifier on a data file with unlabeled cases. There are some constraints however: the data file must be in the Sipina binary format (FDM[2]), we must then convert the data file before; we must apply the model (the most often a decision tree but it can be any classifier available into Sipina) into the Sipina environment, we cannot export the model.

Generalization error rate estimation. Applying the model on unseen cases is a very useful functionality. But it would be even more interesting if we could announce its accuracy. Indeed, a misclassification can have dramatic consequences. We must measure the risk we take when we make decisions from a predictive model. An indication about the performance of a classifier is important when we decide or not to deploy it.

The usual approach to assess the accuracy of a classifier is to subdivide the dataset into train and test samples. The first is used for the learning phase; the second is used for the construction of the confusion matrix comparing the observed values of the class attribute and the predicted values of the model. We then obtain an unbiased estimation of the indicators (error rate, recall, precision, etc.). But this approach is not judicious if we have a moderate size dataset. Indeed, on the one hand, we penalize the construction of the classifier by using a reduced size sample during the learning phase. On the other hand, we have often not enough instances into the test sample in order to obtain an accurate estimation of the error rate.

Thus, in the moderate size dataset context, it is more suited to use a resampling approach such as bootstrap to evaluate the generalization error rate. The aim is to obtain an estimation of the error rate of the classifier learned on the whole dataset. During the resampling process, we can learn many models. Studying these individual models has not an intrinsic interest. They contribute only for the error rate evaluation. This is the reason for which they are not displayed in the majority of the Data Mining tools.

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

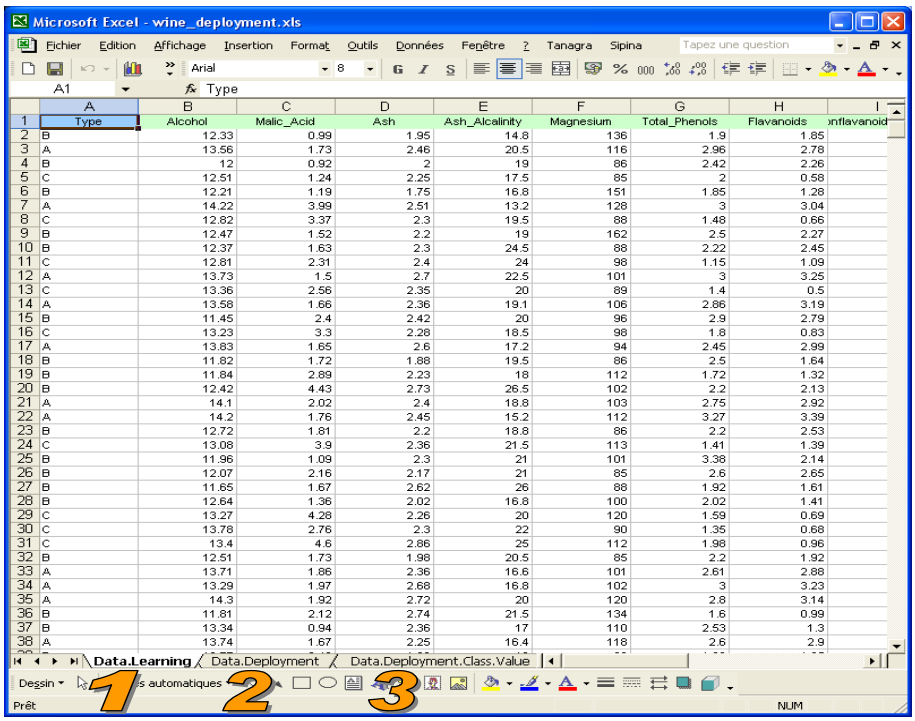
[2] See http://data-mining-tutorials.blogspot.com/2009/11/sipina-supported-file-format.html about the supported file formats in Sipina.

Organization of this tutorial. In this tutorial, we show how to learn a decision tree from a dataset, how to apply the classifier on unlabeled cases, and how to evaluate the generalization error rate using the bootstrap approach. The samples are gathered into Excel workbook. There are several sheets: (1) the dataset used for the learning process and the bootstrap error rate evaluation; (2) the unlabeled instances, only the descriptors are available; (3) the labels of the instances from the second sheet, we can evaluate the true classifier accuracy, we will compare this last one with the value estimated by the bootstrap approach.

2 Dataset

We use the WINE[3] (wine_deployment.xls[4]) dataset. We want to classify alcohols from their chemical properties. The class attribute has 3 values. There are 12 predictive continuous descriptors.

We have the labeled dataset into the first sheet (Data.Learning). We use this sample (100 instances) for the training task and the bootstrap error evaluation. The unlabeled cases (78 instances) are into the second sheet (Data.Deployment). This is the generalization sample. Only the predictive descriptors are available. An identification column ID is added. It allows to control the consistency of this sample with the values into the third sheet. Last, the third sheet (Data.Deployment.Class.Value) contains the true label of the instances into the second sheet. We use this dataset for computing the true accuracy of the classifier.



[3] UCI Machine Learning Repository - http://www.ics.uci.edu/~mlearn/MLSummary.html or ftp://ftp.ics.uci.edu/pub/machine-learning-databases/wine/

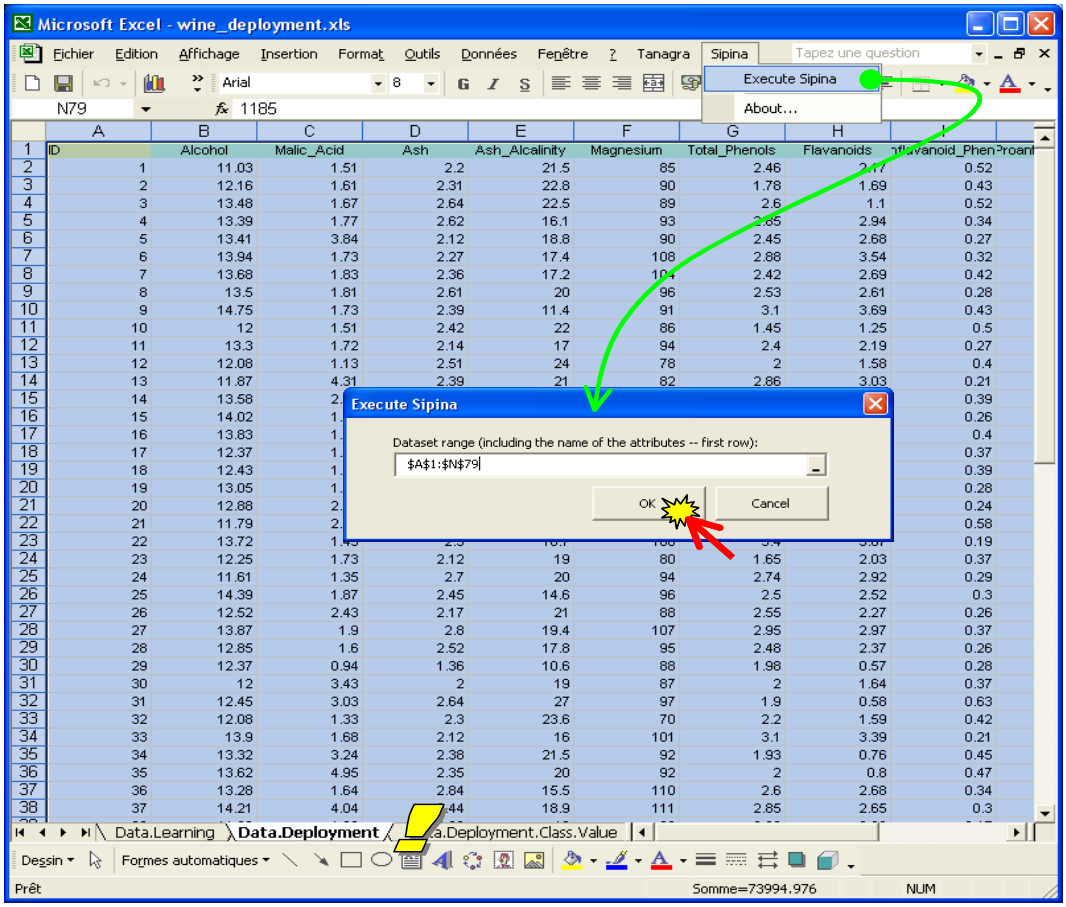
[4] http://eric.univ-lyon2.fr/~ricco/dataset/wine_deployment.xls

3 Learning and deployment

3.1 Preparing the unlabeled sample

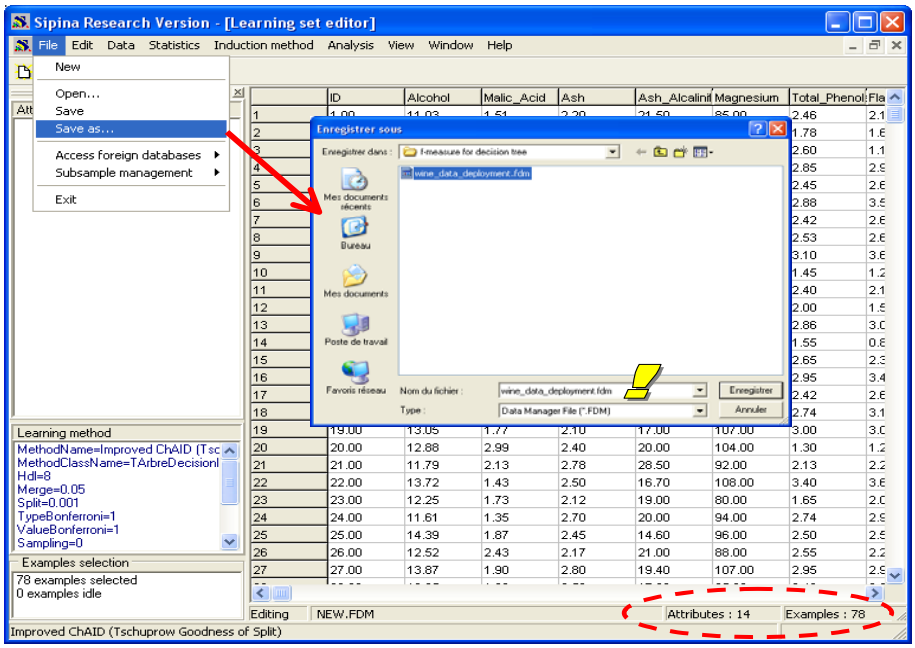
We can launch Sipina from Excel using an add-in[5]. Before to apply a model on a new sample, we must convert it into the native Sipina file format (*.fdm).

First, we select the data range into the Data.Deployment sheet. We click on the SIPINA / EXECUTE SIPINA. We check the selection and we validate.



Sipina is automatically launched. We obtain 14 attributes and 78 instances. In this data preparation phase, we save only this sample in the FDM file format. We click on the FILE / SAVE AS menu. We set "wine_data_deployment.fdm" as file name.

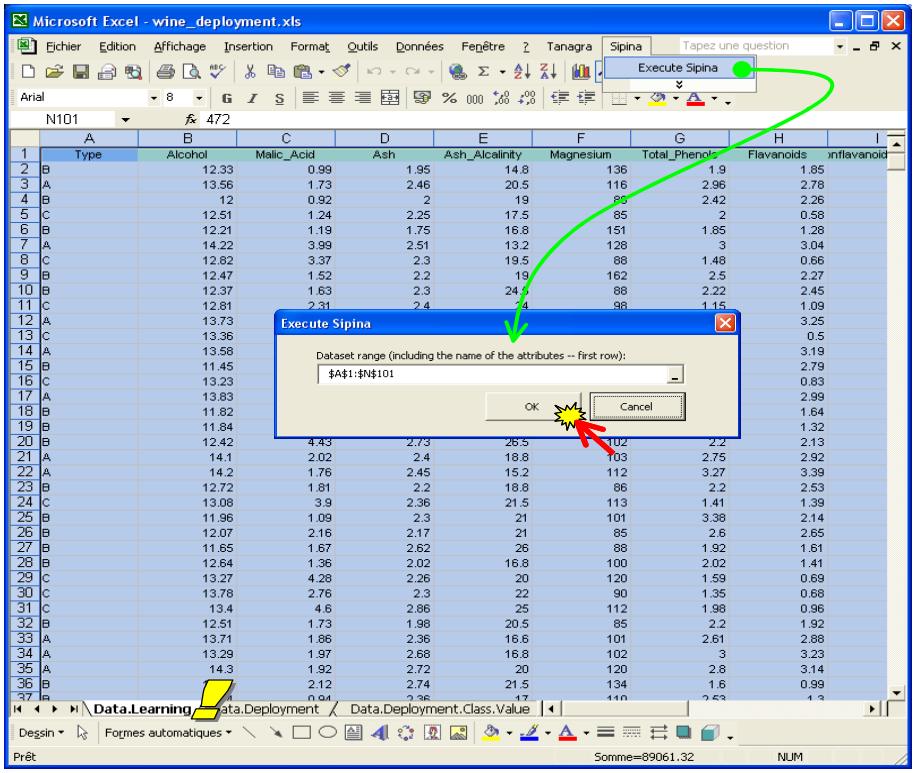
[5] See http://eric.univ-lyon2.fr/~ricco/doc/sipina_xla_installation.htm for the installation of the add-in; http://eric.univ-lyon2.fr/~ricco/doc/sipina_xla_processing.htm, for its utilization.



We can close Sipina now. We are back into the Excel spreadsheet. We select the first sheet.

3.2 Importing the learning sample into SIPINA

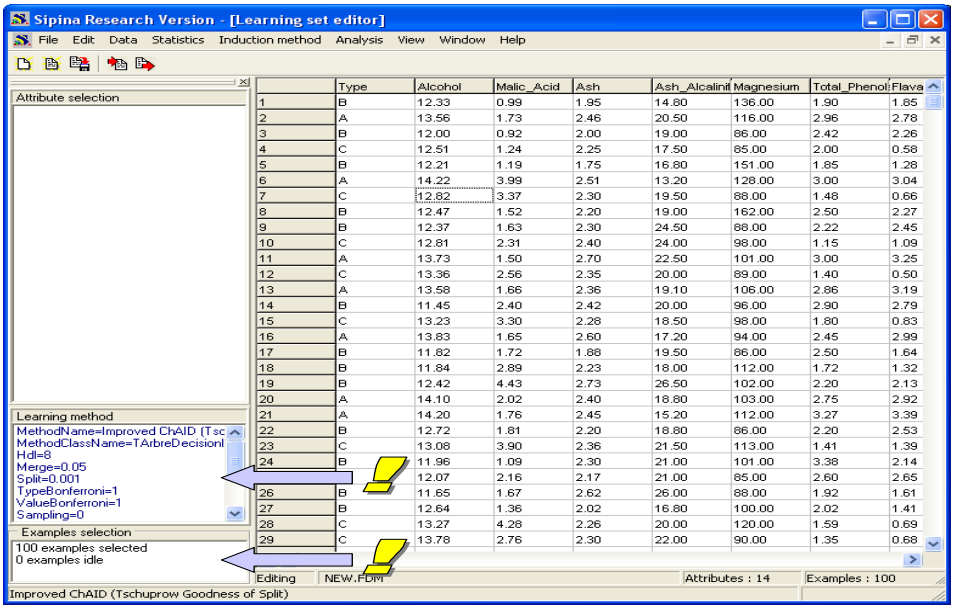
We select the data range into the "Data.Learning" sheet. Again, we click on the SIPINA / EXECUTE SIPINA menu. We check the range selection and we validate.



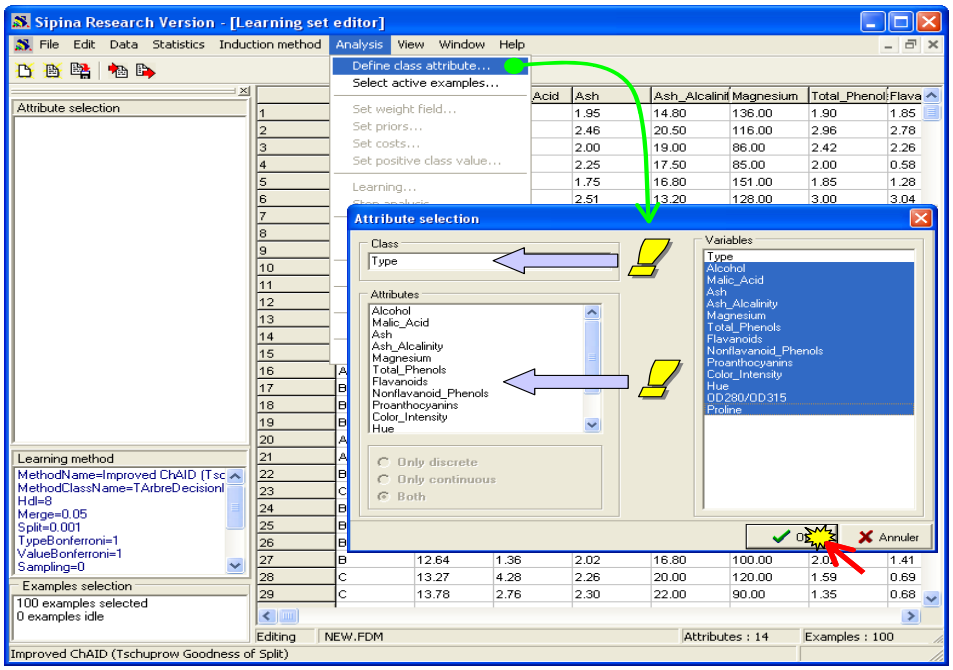
Sipina is automatically launched. We have 100 instances and 15 variables (14 predictive attributes and the class attribute) now.

3.3 Decision tree learning

The default selected method is IMPROVED CHAID. It is a variant of CHAID learning algorithm (Kass, 1980). It is suitable for a first exploration of a dataset. Of course, we can select another decision tree induction method (C4.5, etc.) by clicking on the INDUCTION METHOD / STANDARD ALGORITHM menu. But here, we use the default approach. At the same time, all the instances are automatically selected for the learning process (we can modify this by clicking on the ANALYSIS / SELECT ACTIVE EXAMPLES menu). All these information are described in the left part of the main window.

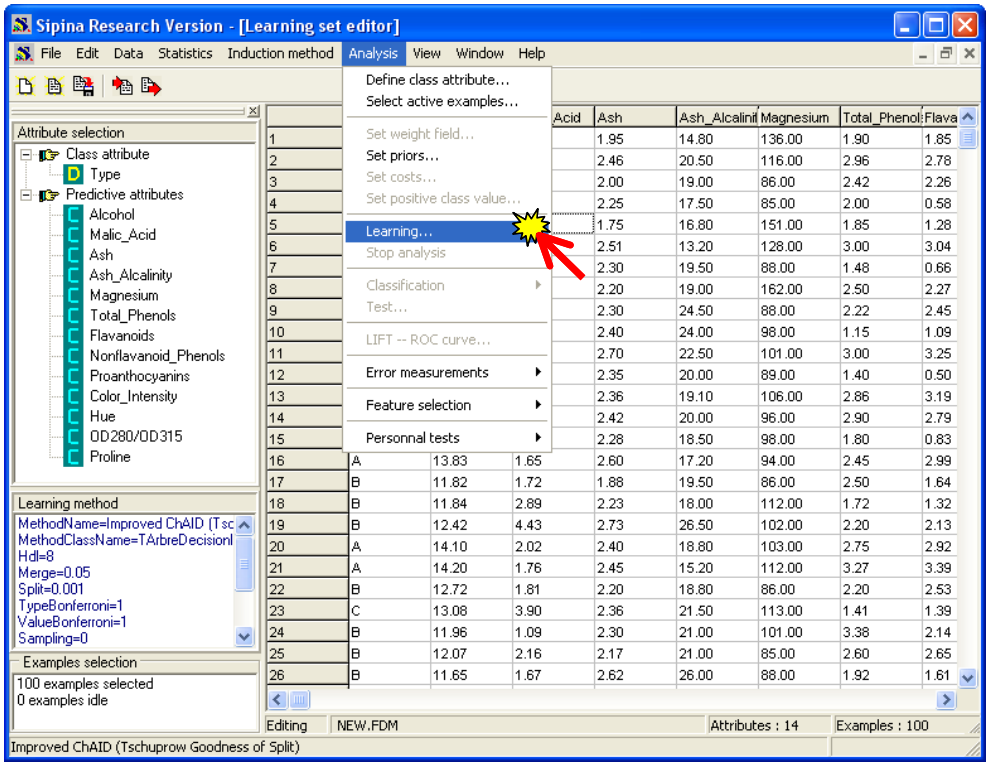


In the next step, we want to specify the target attribute (TYPE) and the input ones (ALCOHOL to PROLINE). We click on the ANALYSIS / DEFINE CLASS ATTRIBUTE menu.

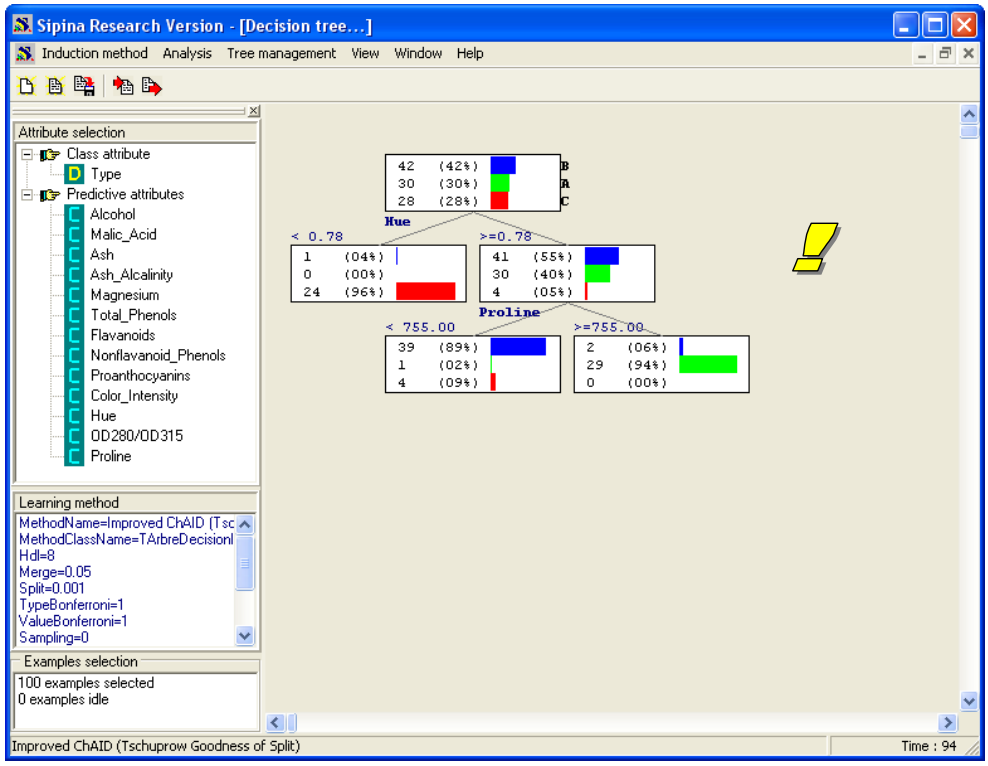


The variable selection is summarized in the left part of the main window.

Then, we can launch the learning process by clicking on the ANALYSIS / LEARNING menu.

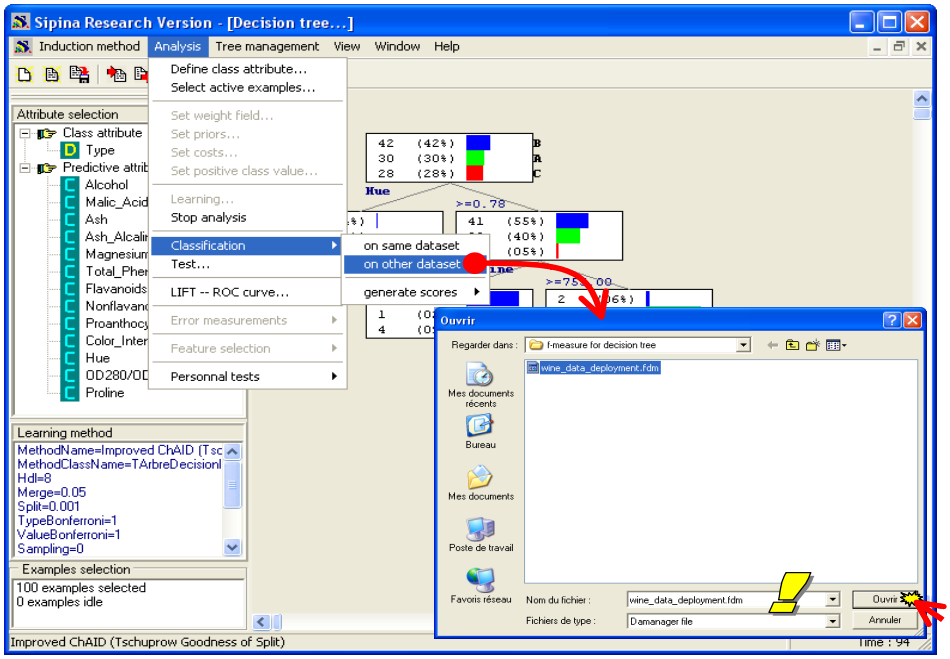


The obtained decision tree is rather simple.

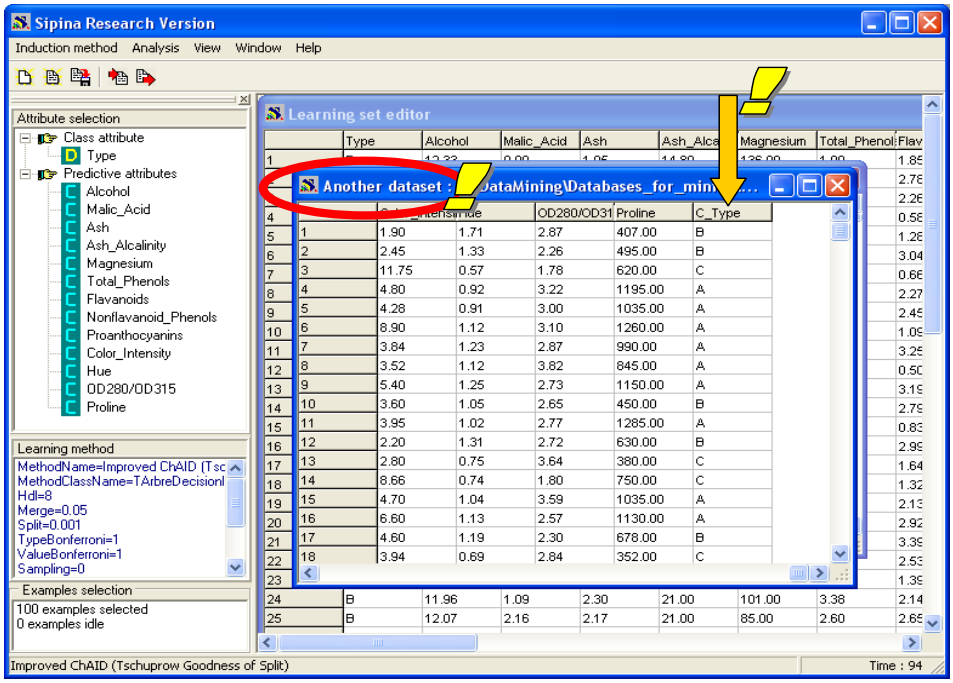


3.4 Applying the decision tree on the unlabeled sample

In order to apply the learned tree on the unlabeled sample, the data file must be in FDM format. Sipina needs only the variables which are selected into the decision tree. It makes the correspondence by using the names of the variables (case sensitive). We click on the ANALYSIS / CLASSIFICATION / **ON OTHER DATASET** menu, we select wine_data_deployment.fdm.



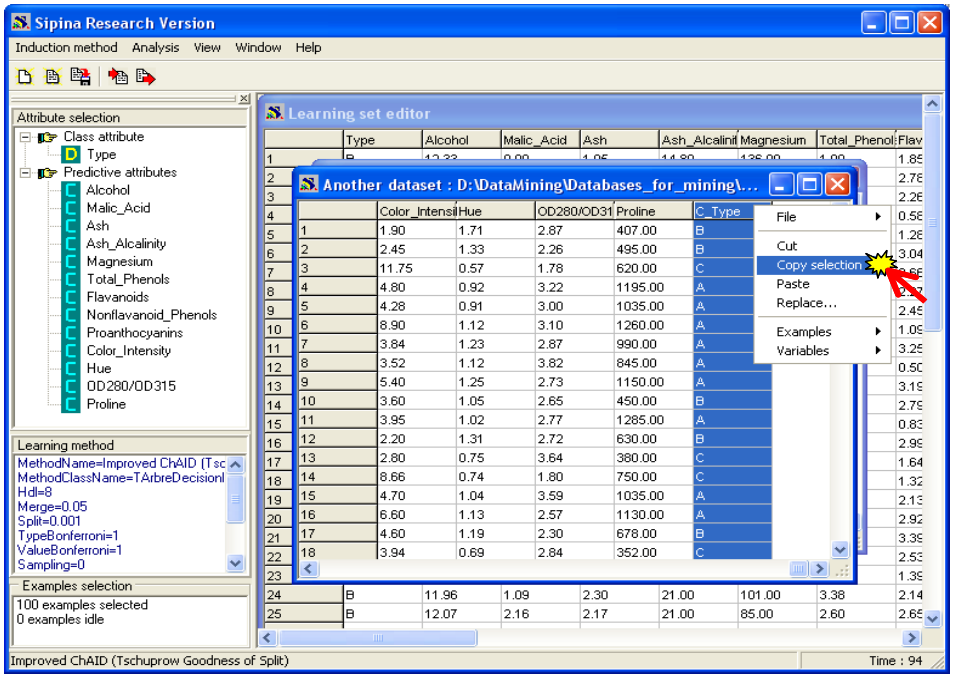
The dataset is loaded into a new visualization grid, and the column with the predicted values is added (C_TYPE in our example).



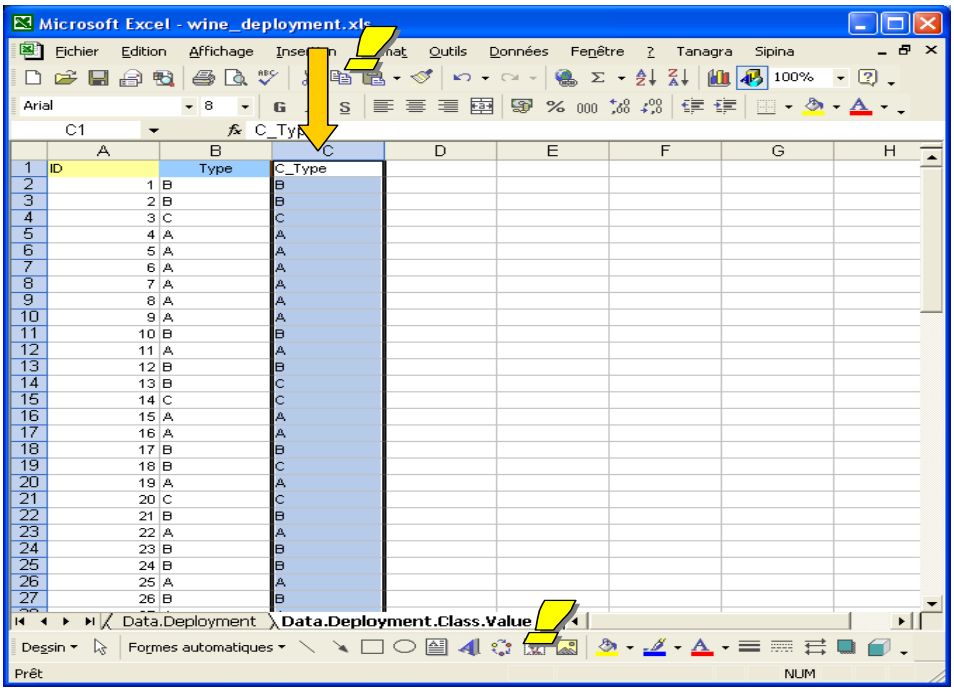
The predicted column is defined as the class attribute, it is a discrete attribute with the same values.

3.5 Retrieving the predicted column

We want to retrieve the predicted column and copying it into the third sheet of the Excel workbook. The aim is to compare latter the observed values and the predicted values of the target attribute. To do this, we activate the contextual menu by (right) clicking on the header of the column. We select the COPY SELECTION menu.



Into Excel, we select the third sheet (Data.Deployment.Class.Value) and we paste the dataset.



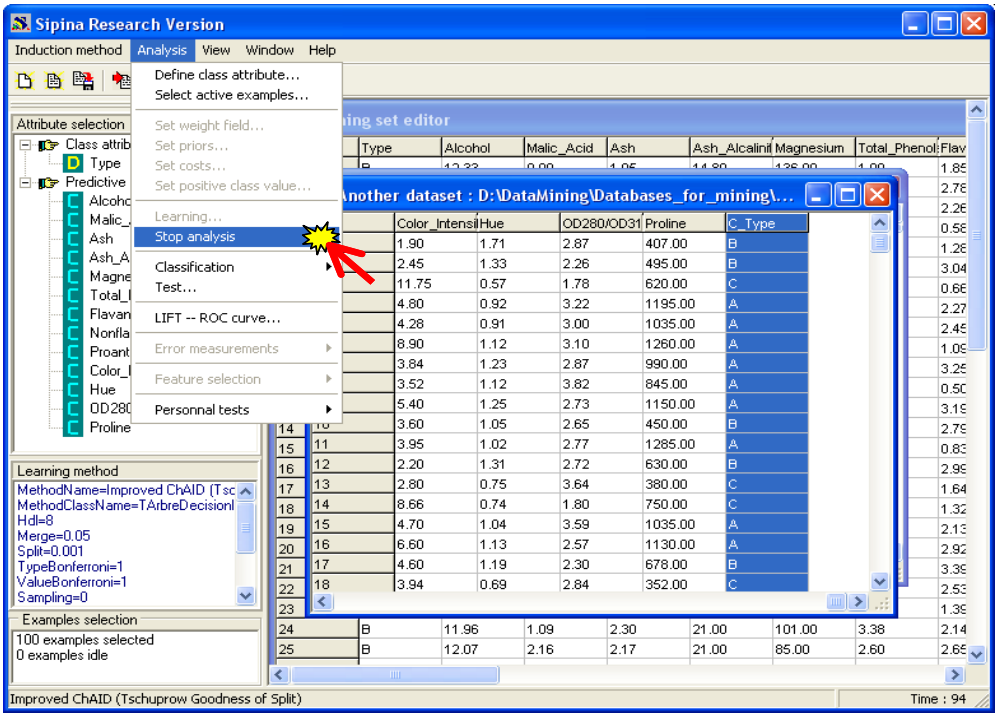
We utilize these values below in order to evaluate the "true" model accuracy.

4 Resampling error rate evaluation

We want to estimate the generalization error rate using the labeled sample only (Data.Learning). Because the size of our dataset is rather small, we cannot reasonably subdivide it into train and test samples for learning and assessing a model. Thus, we use the bootstrap resampling approach.

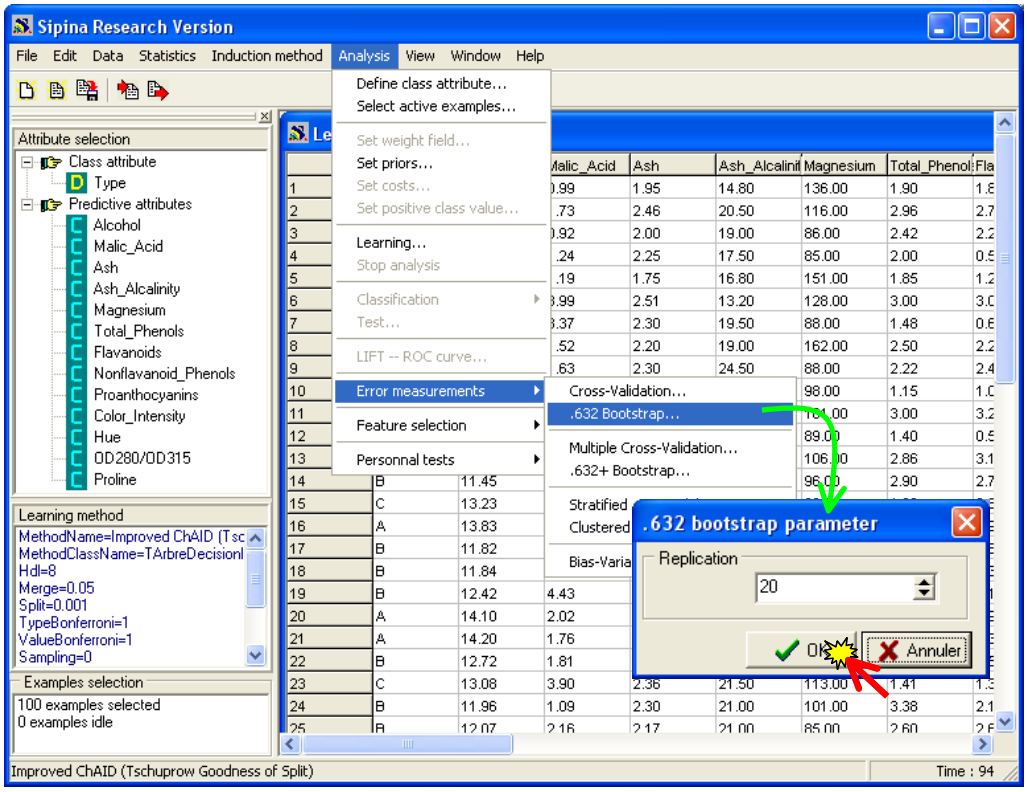
4.1 Bootstrap

To launch the bootstrap process with Sipina, we must stop the current analysis before. We click on the ANALYSIS / STOP ANALYSIS menu.

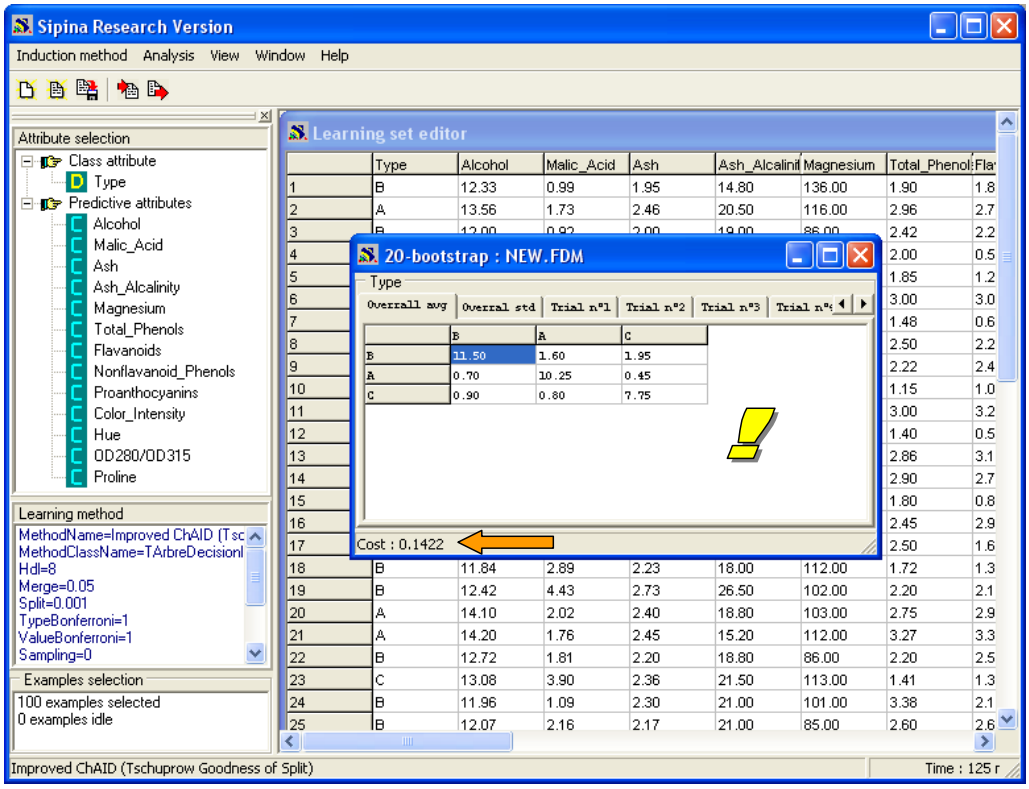


Then, we select the ANALYSIS / ERROR MEASUREMENTS / .632 BOOTSTRAP menu[6]. A dialog box appears. We ask 20 replications. We click on the OK button.

[6] See http://eric.univ-lyon2.fr/~ricco/cours/slides/resampling_evaluation.pdf ; and http://bioinformatics.oxfordjournals.org/cgi/content/full/21/15/3301



The process is rather quick. The decision tree learning algorithm is in general fast and we use a small dataset. We obtain a confusion matrix and the related error rate. We have also the confusion matrix for each trial. But they are not really useful.



The estimated generalization error rate using the bootstrap procedure is 0.1422 i.e. when we apply this classifier on an unlabeled case, the misclassification probability is 14.22%.

4.2 Checking out on the generalization sample

In a normal situation, the labels of the instances in the generalization sample are not available. One of the aim of our experiments is to compare the bootstrap (more generally resampling scheme) error rate estimation with the "true" error rate when we deploy the model.

We come back to our Excel workbook. We use the pivot table[7] in order to cross-tabulate the observed labels and the predicted labels into the third sheet (DATA.DEPLOYMENT.CLASS.VALUE).

Nombre de Type	C_Type			
Type	A	B	C	Total
A	28	1		29
B	2	23	4	29
C		4	16	20
Total	30	28	20	78

The measured error rate is $\varepsilon = \frac{1+2+4+4}{78} = 14.10\%$

The error rate obtained by bootstrap on the learning sample seems to be a good estimation of the "true" error rate measured on the generalization sample. However, the precision of the estimation is not always as well as on our dataset.

5 About the other learning methods

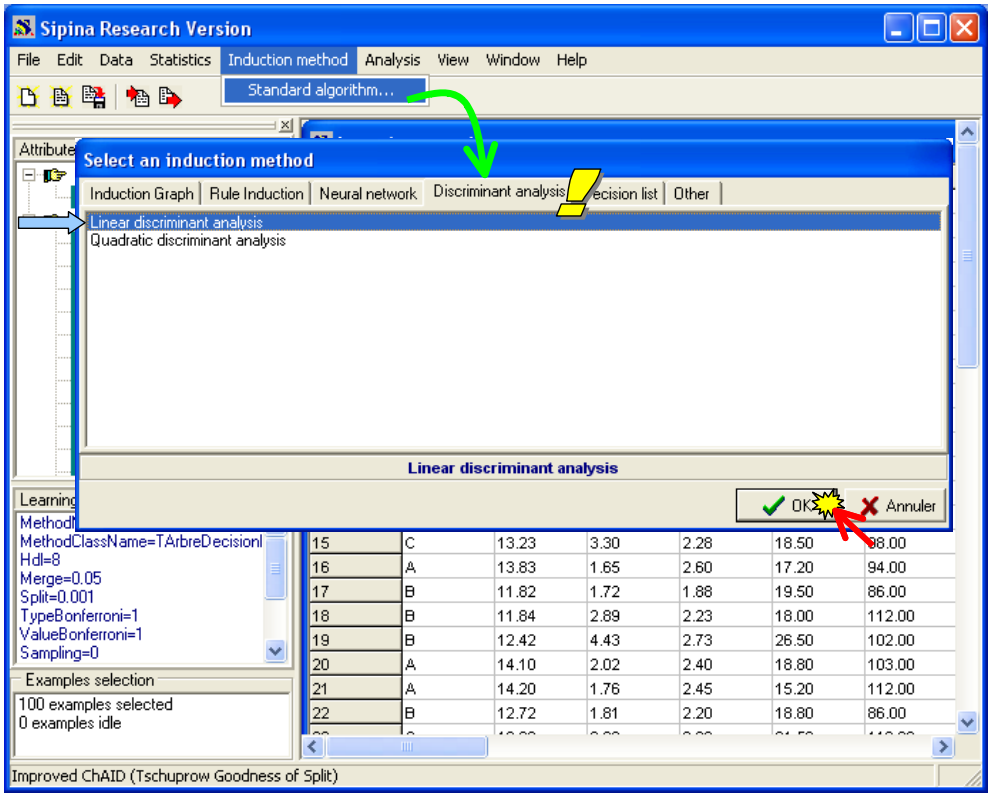
Sipina is especially intended to decision tree. But other supervised learning algorithms are also available. We can use the same process in order to create a classifier and apply it on unlabeled instances.

Let's take a Linear Discriminant Analysis[8] (LDA) for instance.

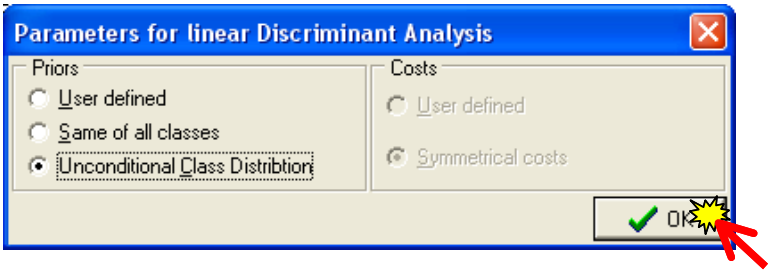
We stop the current analysis by clicking on the WINDOW / CLOSE ALL menu. Then we click on the INDUCTION METHOD / STANDARD ALGORITHM menu. A dialog box appears, we select the DISCRIMINANT ANALYSIS tab. We select the LINEAR DISCRIMINANT ANALYSIS method.

[7] http://en.wikipedia.org/wiki/Pivot_table

[8] http://en.wikipedia.org/wiki/Linear_discriminant_analysis



Another dialog box appears. We validate the default settings.

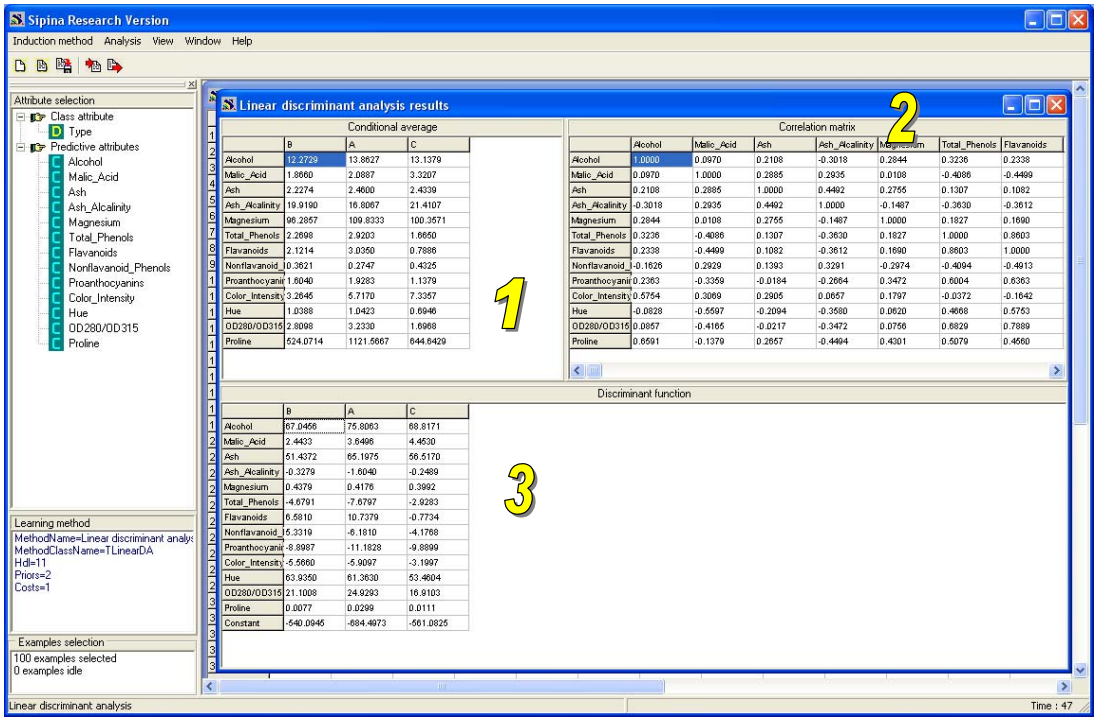


We launch the learning process by clicking again on the ANALYSIS / LEARNING menu.

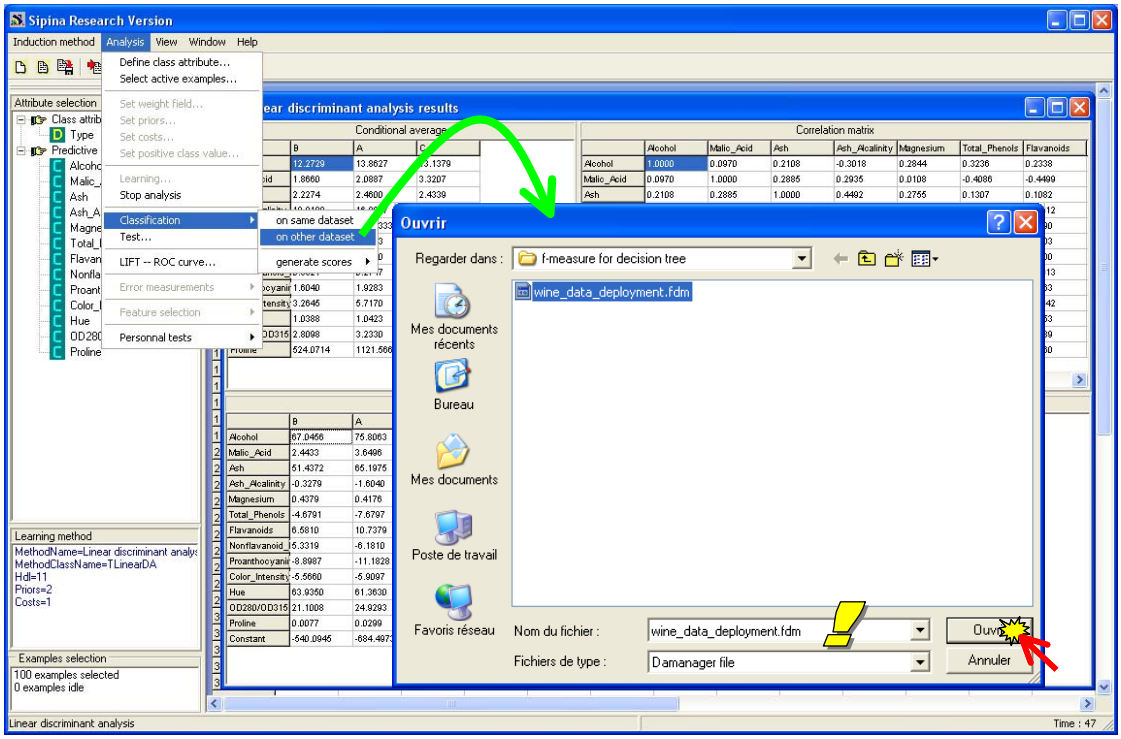
The classifier is described in a new window. There are: (1) the conditional mean of each descriptor according to the values of the target attribute (A, B and C); (2) the correlation matrix; (3) the classification functions which are used to assign the predicted value when we deploy the model.

Certainly, some important indications are missing, such as global significance of the model or individual significance of the descriptors. It is one of the reason for which I advise to use Tanagra (http://eric.univ-lyon2.fr/~ricco/tanagra/) for the other supervised learning methods than the decision tree learning[9].

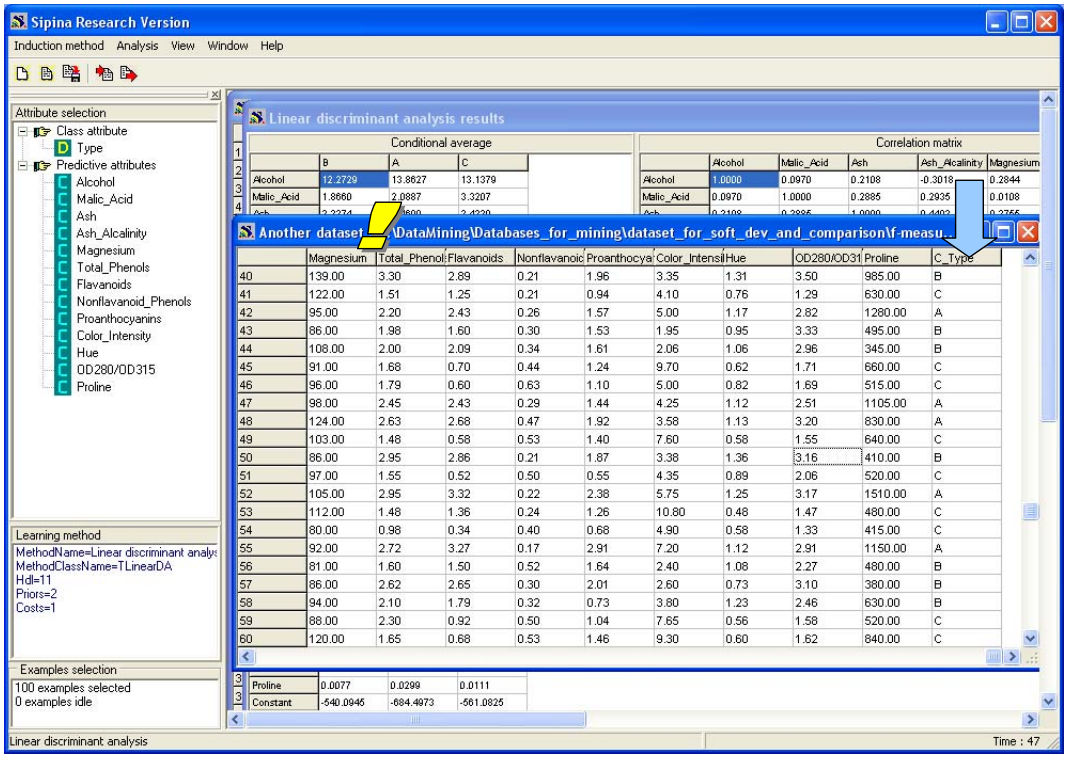
[9] In effect, Tanagra implements also decision tree learning algorithms (CART, C4.5, ID3, etc.). But it does not supply, currently, the same interactive capabilities as Sipina when we explore a tree (choosing manually the split attribute, extracting the examples or computing descriptive statistics on each node, etc.).



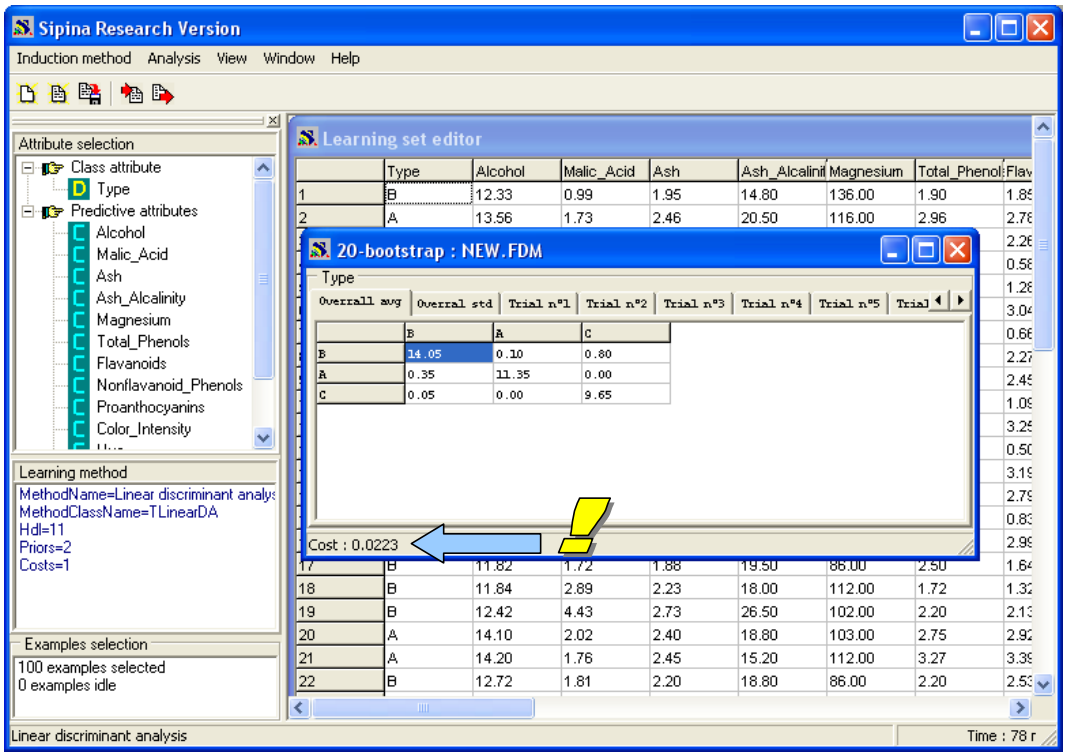
We can apply this model on the unlabeled sample. We click on the ANALYSIS / CLASSIFICATION / ON OTHER DATASET menu. We select « wine_data_deployment.fdm ».



A data visualization grid appears. In addition to the descriptors, we have now a new column with the predicted values of the LDA classifier.



Again, we can estimate the generalization error rate by using the bootstrap method. We stop the current analysis by activating the ANALYSIS / STOP ANALYSIS menu. Afterwards, we click on the ANALYSIS / ERROR MEASUREMENTS / .632 BOOTSTRAP menu.



The estimated generalization error rate is 2.23%. LDA outperforms definitely the decision tree (IMPROVED CHAID) on the Wine dataset.

6 Conclusion

In this tutorial, we show how to apply a classifier on unlabeled sample with Sipina. We show also how to estimate the generalization error rate using a resampling scheme such as bootstrap.

We were able to check out the credibility of this estimation because we have in reality the true labels of the observations of the generalization sample. We note that it is rather accurate.