Subject

Copy paste feature into the diagram.

When we define the data analysis process into Tanagra, it is possible to copy components (or entire branches of components) towards another location into the diagram. This feature is very helpful when we have to repeat sequences of treatments in different parts of the diagram. The settings are also duplicated.

In this tutorial, we show how to copy a component or a branch. We will see that this feature is helpful when, for instance, we deal with the performance comparisons of supervised learning algorithms on the same dataset. In this context, the processing sequence is always the same, only the method that we want to evaluate is different.

We work on the same project here. We cannot copy paste components between two opened projects. But, in another tutorial, we show how to save a part of the diagram in an external file. Thus, the same processing sequence can be applied on multiple datasets[1].

Dataset

We use the SONAR dataset (http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/sonar.xls). We want to compare the accuracy of various supervised learning algorithms: binary logistic regression, linear discriminant analysis, k-nearest neighbor, support vector machine and PLS regression for classification.

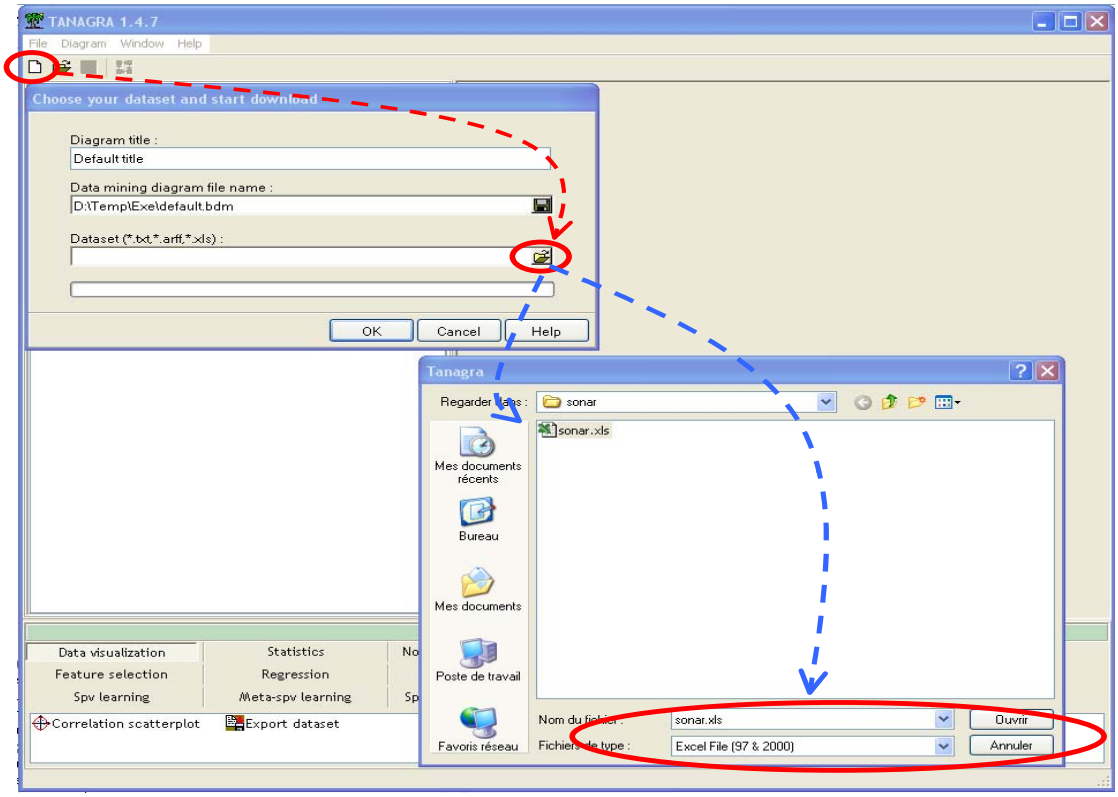
Copy paste feature of the diagram

Importing the dataset

We launch Tanagra. We create a new diagram by clicking on the FILE / NEW menu. We select the SONAR.XLS data file. Tanagra can import the Excel file format, even if the spreadsheet is not installed on the computer[2].

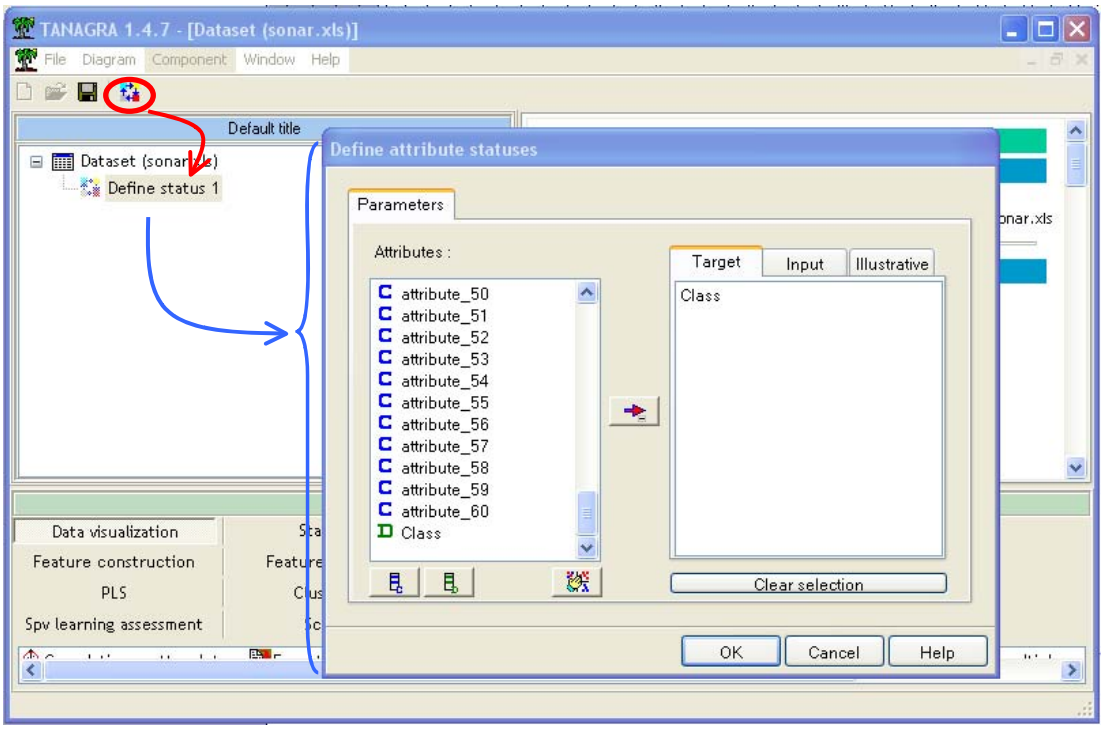
[1] http://data-mining-tutorials.blogspot.com/search/label/Diagram%20management

[2] About the handling of the Excel file format, see http://data-mining-tutorials.blogspot.com/2008/10/excel-file-format-direct-importation.html ; we can also send the data from Excel using an add-in http://data-mining-tutorials.blogspot.com/2008/10/excel-file-handling-using-add-in.html



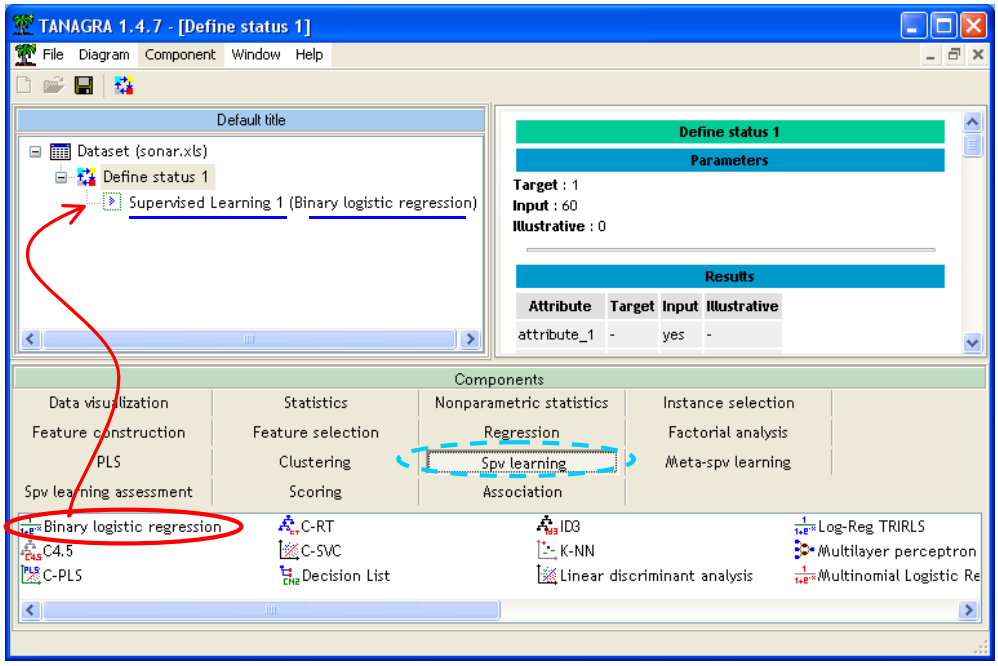
Specifying the type of the variables

We insert the DEFINE STATUS component in order to define the target attribute (CLASS) and the input attributes (ATTRIBUTE_01 … ATTRIBUTE_60).

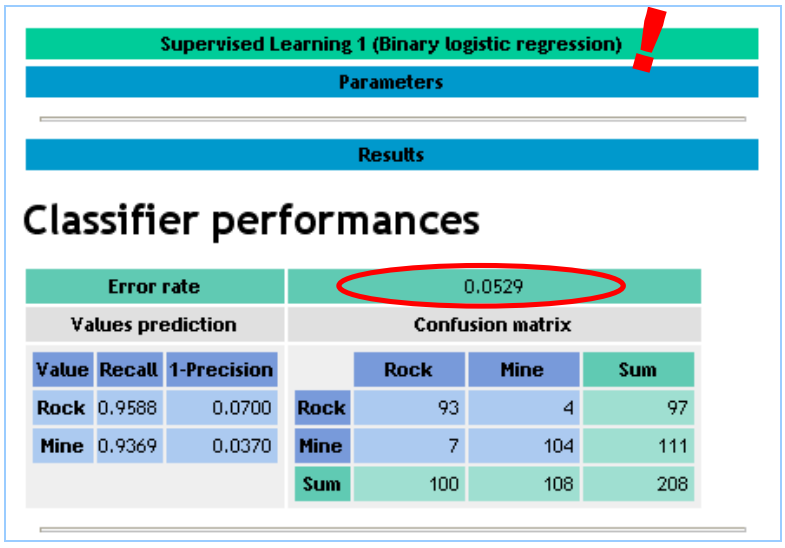


Implementation and assessment of a supervised learning algorithm

We add the BINARY LOGISTIC REGRESSION component (SPV LEARNING tab) into the diagram[3].

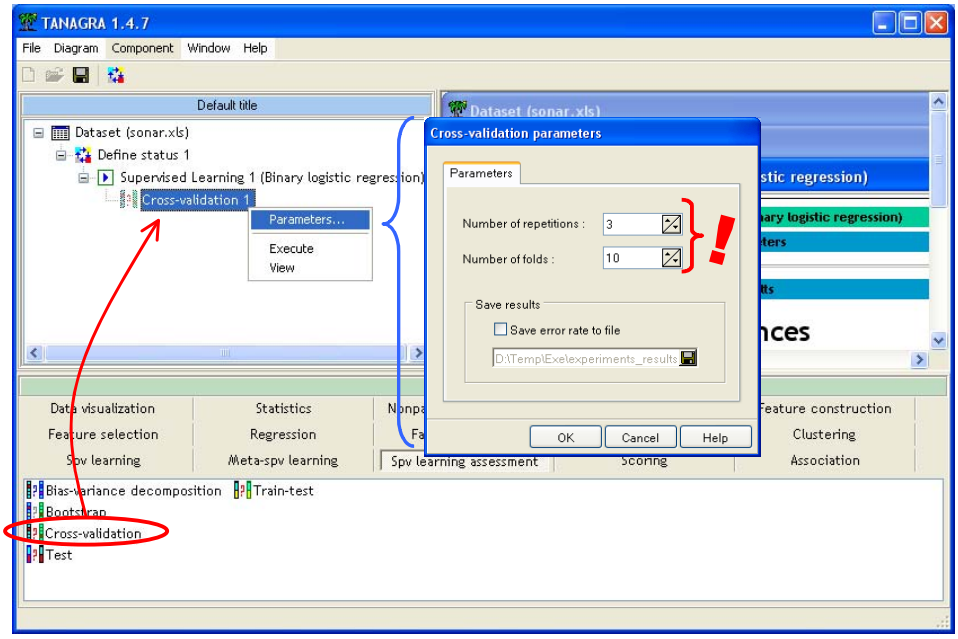


The resubstitution error rate is 5.29%.



[3] Before the 1.4.7 version, this insertion had to be made in two steps: first, inserting the meta supervised learning algorithm (single learning, bagging, boosting, etc.); second, inserting the learning method. Since the 1.4.7 version, we can insert directly the learning algorithm for the standard approach. For the non standard approach (bagging, boosting, cost sensitive learning, etc.), we must still follow the two steps.

We know that the error rate computed on the learning sample is optimistic. In order to obtain a more reliable evaluation, we use a resampling method, here a cross-validation algorithm. The component can be found in the SPV LEARNING ASSESSMENT tab. We click on the PARAMETERS menu. We set the following settings (TRIALS = 3; FOLDS = 10). We must use the same scheme for the other learning algorithms that we want to evaluate.



The cross validation error rate estimate is 30.5%.

Cross-validation 1

Parameters

Cross-validation parameters	
Folds	10
Trials	3

Results

CV error rate

Range	
MIN	0.3000
MAX	0.3100

Trial	Err rate
1	0.3000
2	0.3050
3	0.3100

Overall cross-validation error rate

Error rate	0.3050

Values prediction

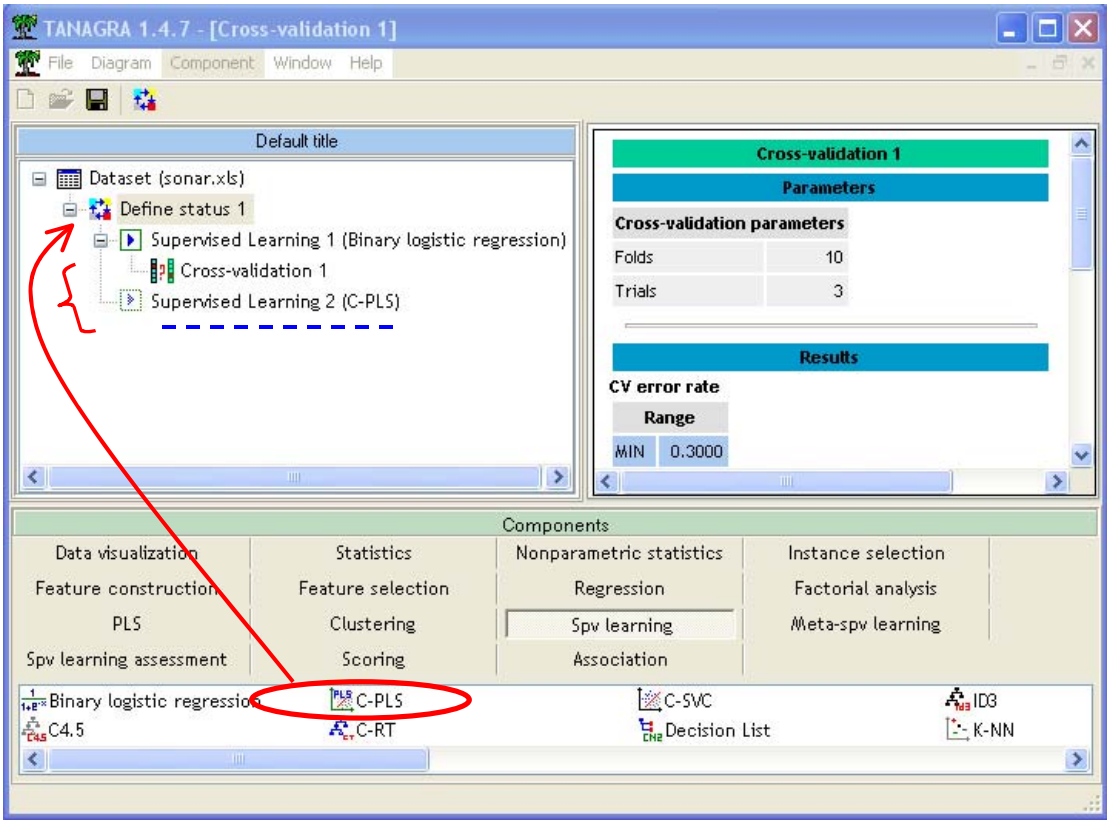
Value	Recall	1-Precision
Rock	0.6403	0.3180
Mine	0.7422	0.2950

Confusion matrix

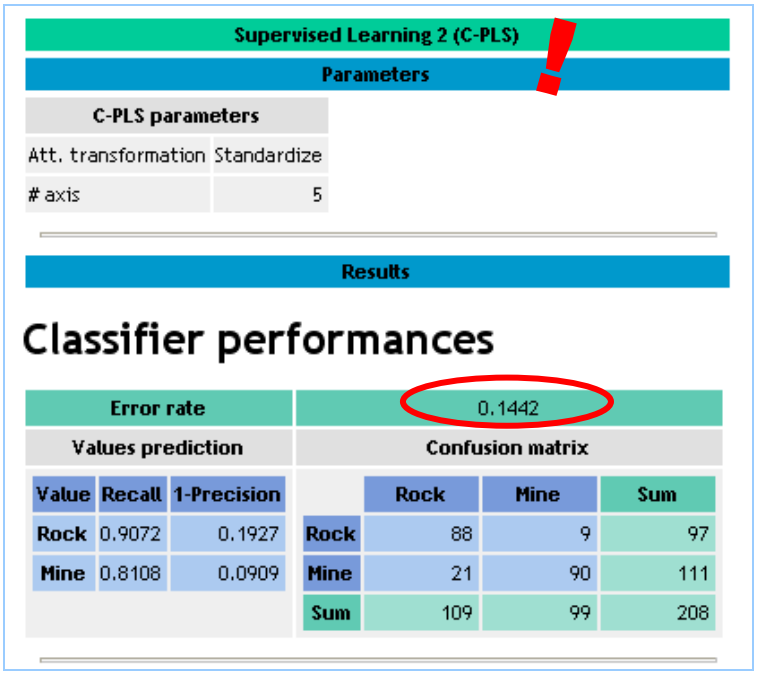
	Rock	Mine	Sum
Rock	178	100	278
Mine	83	239	322
Sum	261	339	600

Implementing and assessing another learning algorithm

We want to apply the same experimentation framework on the PLS Regression (C-PLS from the SPV LEARNING tab). We add the component into the diagram.



The resubstitution error rate estimate is 14.42%.



Supervised Learning 2 (C-PLS)

Parameters

C-PLS parameters	
Att. transformation	Standardize
# axis	5

Results

Classifier performances

Error rate	0.1442

Values prediction

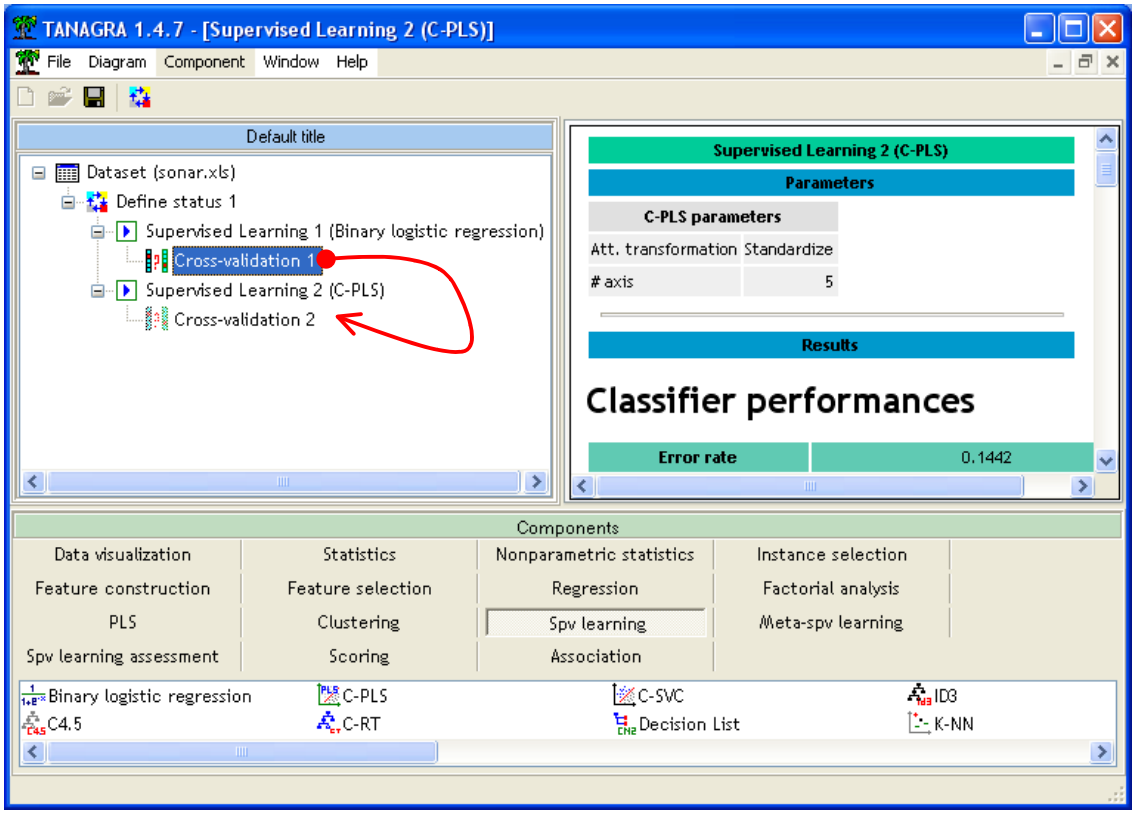
Value	Recall	1-Precision
Rock	0.9072	0.1927
Mine	0.8108	0.0909

Confusion matrix

	Rock	Mine	Sum
Rock	88	9	97
Mine	21	90	111
Sum	109	99	208

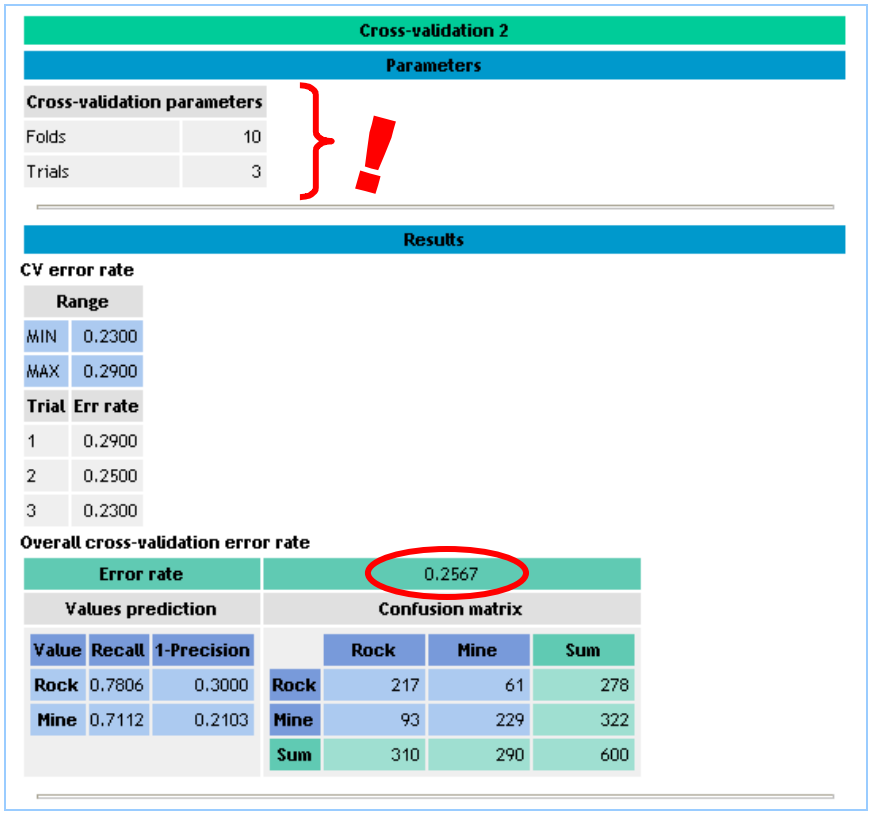
Again, we want to implement the cross-validation with the same settings. If the copy paste feature does not exist, we must insert the component and specify the appropriate settings. It is rather tedious. With the copy paste functionality, we can duplicate the treatment(s) by copying the component from the diagram.

In order to duplicate the existing CROSS-VALIDATION 1 component, we must select it. Then, by using the drag and drop principle, we copy the component on the SPV LEARNING 1 (C-PLS) treatment. The operation must be achieved with the mouse. None shortcut or menu allows to make this.



The new component is automatically numbered (CROSS-VALIDATION 2). By launching the cross-validation, we obtain a reliable estimation of the error rate: 25.67%.

We note that the settings are well transmitted to the duplicated component. We have 3 TRIALS of 10 FOLDS cross validation.



Cross-validation 2

Parameters

Cross-validation parameters	
Folds	10
Trials	3

Results

CV error rate

Range	
MIN	0.2300
MAX	0.2900
Trial	**Err rate**
1	0.2900
2	0.2500
3	0.2300

Overall cross-validation error rate

Error rate	0.2567

Values prediction

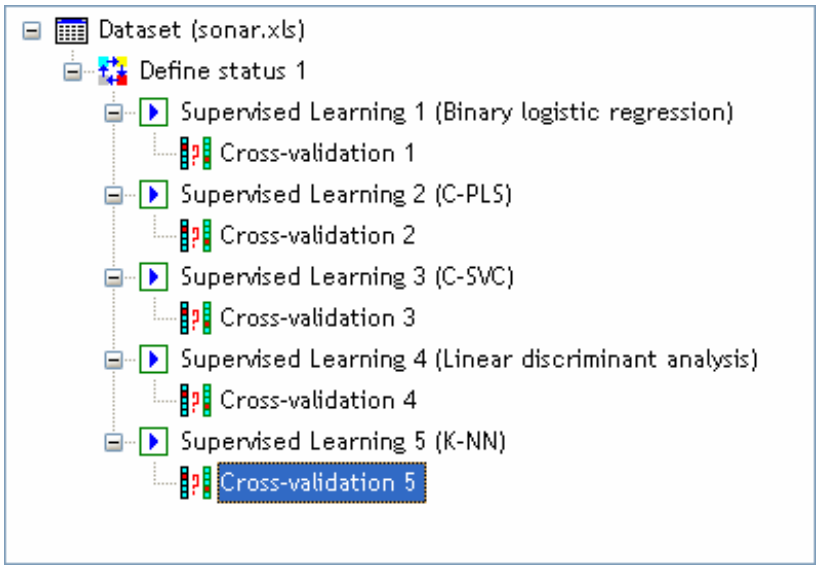
Value	Recall	1-Precision
Rock	0.7806	0.3000
Mine	0.7112	0.2103

Confusion matrix

	Rock	Mine	Sum
Rock	217	61	278
Mine	93	229	322
Sum	310	290	600

Assessing many supervised learning algorithm

By repeating these operations, we fill out the diagram in the following way.



We can then obtain a table which displays the estimated error rate according to the assessed supervised learning method.

Table 1 - Error rate estimate according the learning algorithm

Method	Resubstitution error rate estimate (%)	Cross validation error rate estimate (%)
Logistic regression	5.29	30.5
PLS Regression (C-PLS)	14.42	25.67
Linear SVM (C-SVC)	12.02	25.33
Linear Discriminant Analysis	10.10	23.50
K-Nearest Neighbor	9.62	**14.17**

In this framework, the K-NN algorithm seems the most accurate despite the unfavorable ratio between the number of predictive variables (60) and the number of instances (208).

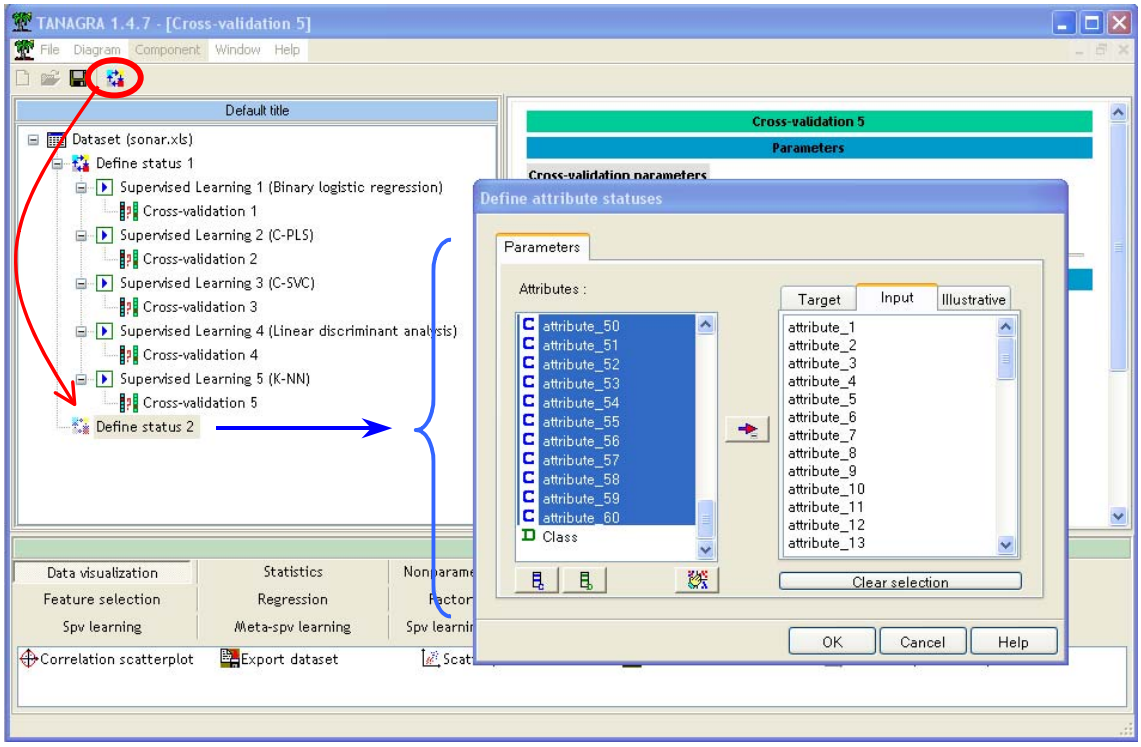
Dimensionality reduction: duplicating a branch of the diagram

Because the number of descriptors is high in relation to the number of observations, a dimensionality reduction seems an appropriate strategy. The goal is to apply a mapping of the instances to a new representation space with fewer dimensions; the loss of information must be as weak as possible.

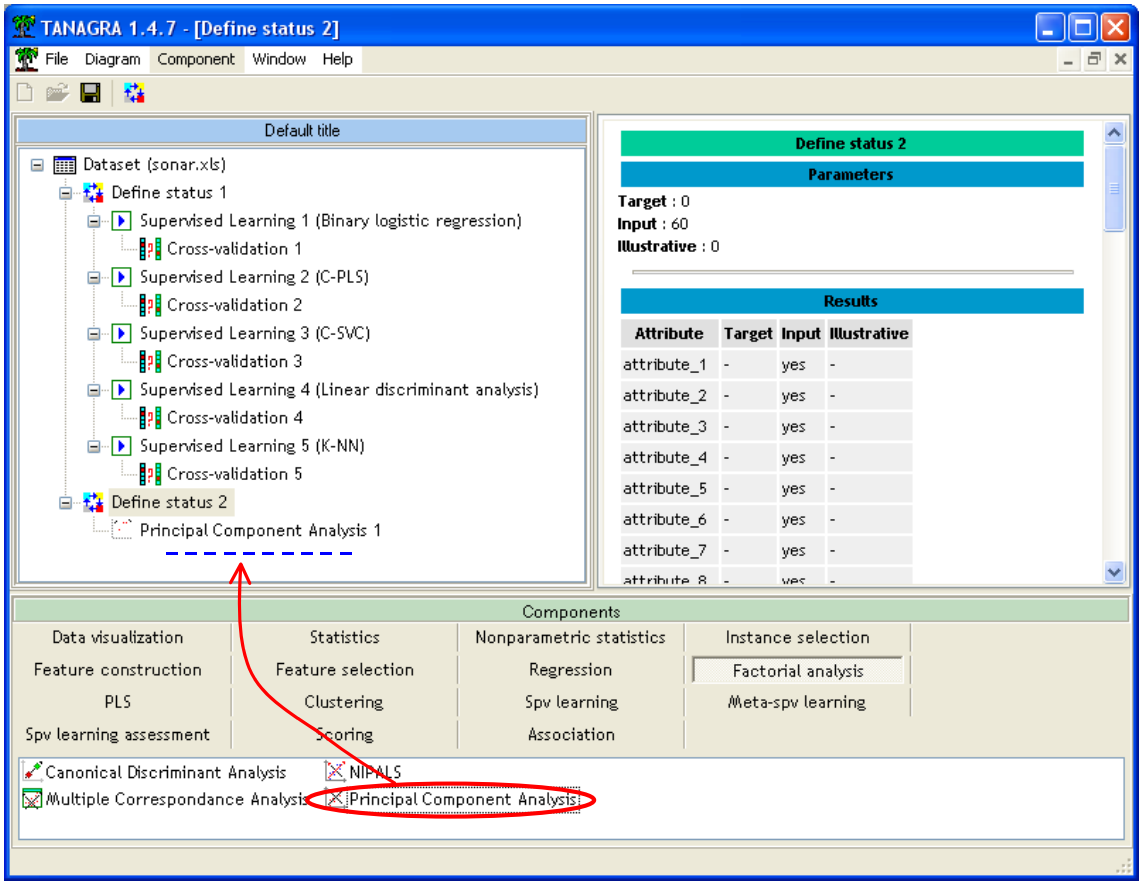
We proceed in the following way: (1) we perform a PCA (Principal Component Analysis); (2) we use the relevant [4] factors as INPUT variables for the K-NN algorithm.

We insert a DEFINE STATUS component into our diagram. We set all the continuous variables (ATTRIBUTE_01 to ATTRIBUTE_60) as INPUT.

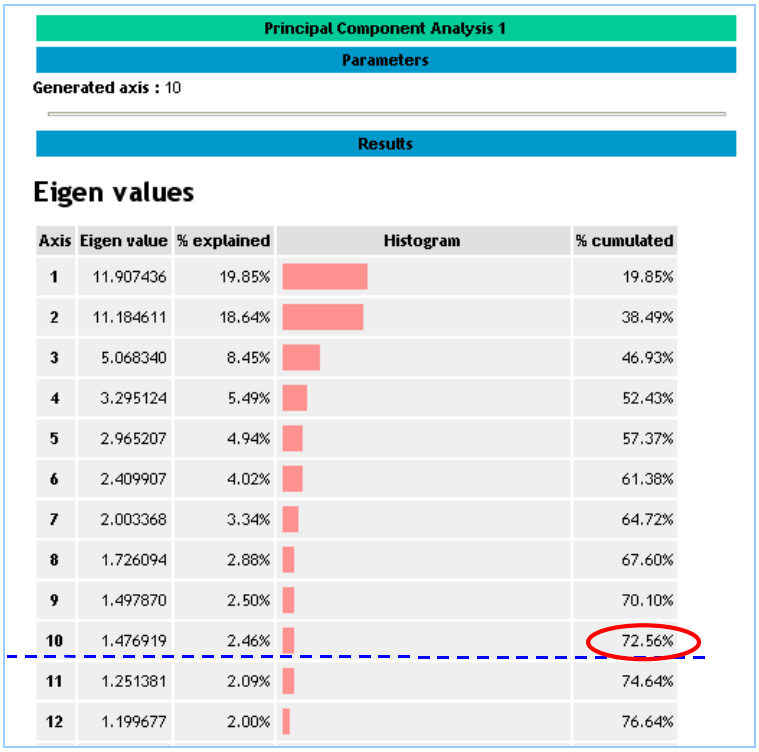
[4] It is an open problem, especially in the context of the dimensionality reduction for subsequent supervised learning algorithm. By varying the number of factors to retain, we can certainly change the accuracy of the classifiers.



We add the PRINCIPAL COMPONENT ANALYSIS component into the diagram. We use the default settings i.e. the component generates the 10 best factors.



We click on the VIEW menu. The report shows that the 10 first factors contain 72% of the available information.



Principal Component Analysis 1

Parameters

Generated axis : 10

Results

Eigen values

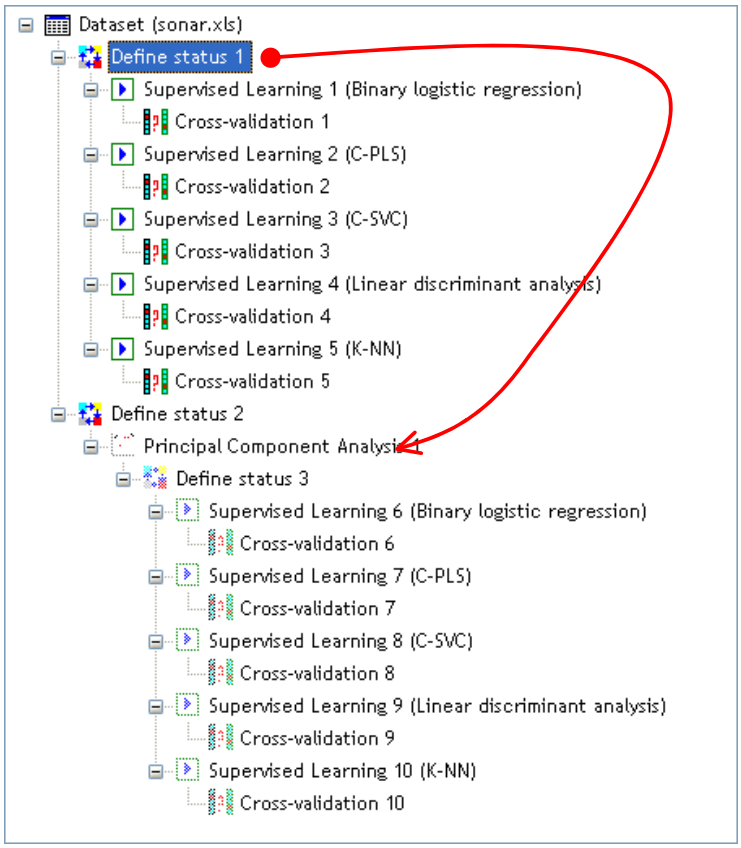
Axis	Eigen value	% explained	Histogram	% cumulated
1	11.907436	19.85%		19.85%
2	11.184611	18.64%		38.49%
3	5.068340	8.45%		46.93%
4	3.295124	5.49%		52.43%
5	2.965207	4.94%		57.37%
6	2.409907	4.02%		61.38%
7	2.003368	3.34%		64.72%
8	1.726094	2.88%		67.60%
9	1.497870	2.50%		70.10%
10	1.476919	2.46%		72.56%
11	1.251381	2.09%		74.64%
12	1.199677	2.00%		76.64%

We want to launch the learning process on the factors generated by the PCA.

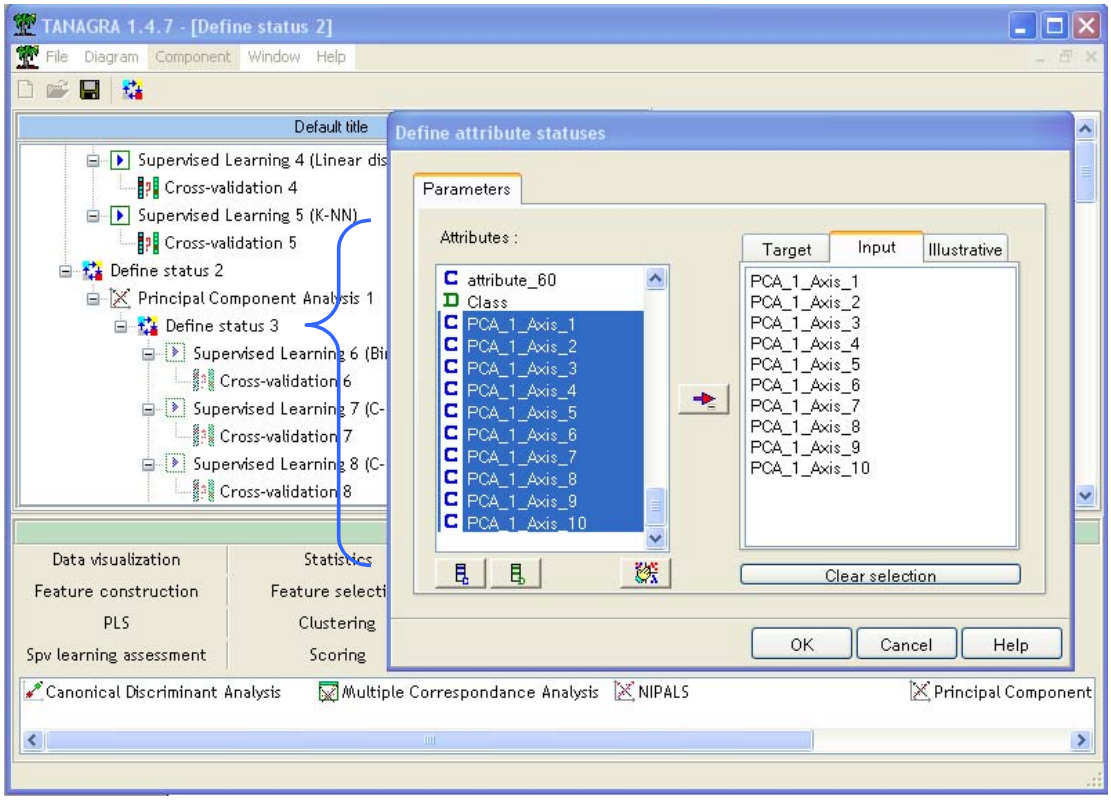
We can then specify the processing sequence above i.e. “learning algorithm” + “cross validation error rate estimate” for each learning method to evaluate. The settings must be the same one. Of course, defining the whole sequence manually is very tedious. Here, by using the mouse drag and drop copy paste principle, we can duplicate all the treatments.

So, we click on the DEFINE STATUS 1 component into the diagram. We drag it with the mouse on the PRINCIPAL COMPONENT ANALYSIS 1. The sub tree below DEFINE STATUS 1 is duplicated. The components are automatically numbered.

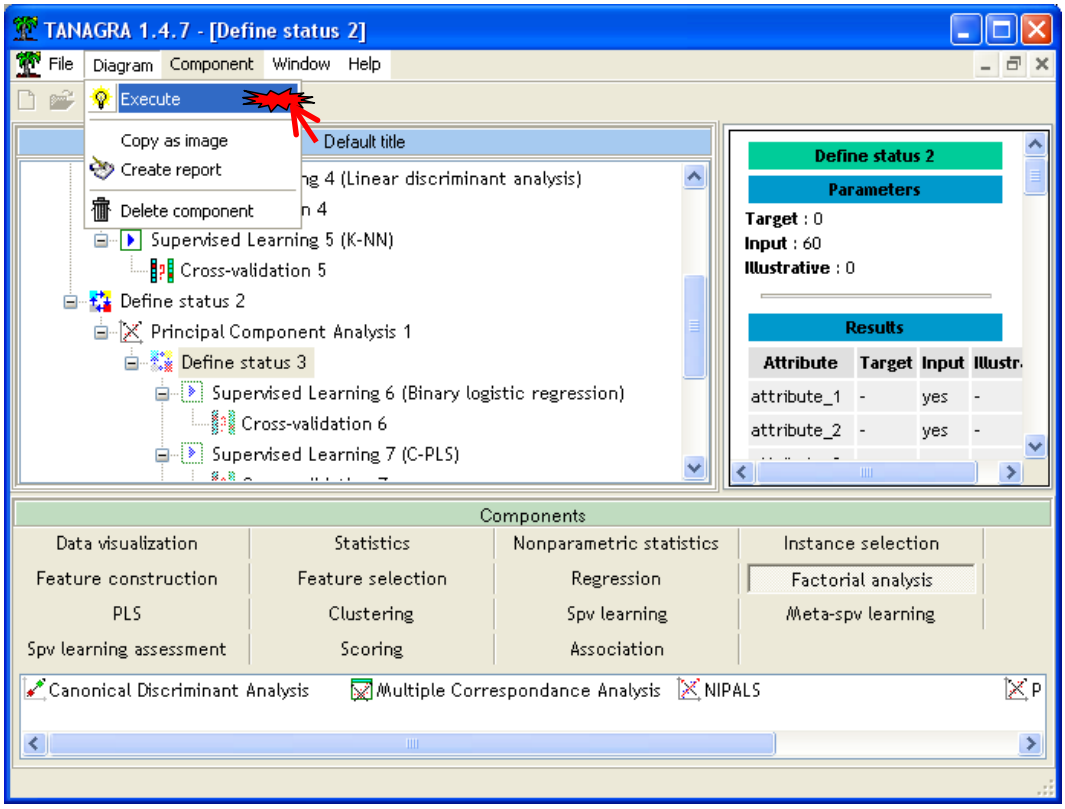
We note that Tanagra performs a "real" cross validation i.e. at each learning step process during the cross validation, all the sequence is launched (PCA + SUPERVISED LEARNING METHOD).



Into the DEFINE STATUS 3 component, the 10 factors as INPUT variables; the TARGET attribute is CLASS.



Now, we must launch the whole diagram. The easiest way is to click on the DIAGRAM / EXECUTE menu.



The Table 2 outlines the performances. We note that all the linear classifiers have benefit from the form of regularization incorporated by the PCA. The result associated to the K-NN seems state that the number of retained factors is important. We can fine tuning this parameter.

Table 2 - Error rate according the learning method

Method	Resubstitution error rate (%)	Cross validation error rate (%)
Logistic Regression	16.35	20.17
PLS Regression(C-PLS)	16.83	21.67
Linear SVM (C-SVC)	18.27	20.83
Discriminant Analysis	15.87	21.50
K-Nearest Neighbor	7.69	**16.33**

We note above all that the copy paste feature is very useful in our context.