

1. Subject

Resampling approaches for prediction error estimation.

The ability to predict correctly is one of the most important criteria to evaluate classifiers in supervised learning. The preferred indicator is the error rate (1 - accuracy rate). It states the probability of misclassification of a classifier. In most cases we do not know the true error rate because we do not have the whole population and we do not know the probability distribution of the data. So we need to compute estimation from the available dataset.

The first estimator, the simplest, is the resubstitution error rate. We calculate the percentage of misclassified on the training set that we were used to learn the classifier. Most of the tools supply this indicator. A contingency table, called confusion matrix, is also displayed. It compares, for all individuals in the sample, the true value of the class attribute and the predicted value of the classifier. We know the resubstitution error rate is highly optimistic. It underestimates the true error rate because we use the same dataset for training and testing the classifier. The optimism will be all the more high that an observation determines the prediction of its value in the final model. A 1-NN for example states an error rate equal to zero because its closest neighbor is itself. In general, classifiers that tend to overfit the dataset provide an optimistic resubstitution error rate.

The hold out error estimation is a method which allows to alleviate this drawback. The principle is that we splitting the data into 2 parts: the first called training (or learning) set (e.g. 2 / 3) is used to create the classifier; the second, called the test set (e.g. 1 / 3), is used to estimate the error rate. It is unbiased. It would be ideal if we were not faced with another problem: when we deal with a small sample size, dedicating a part of the dataset to the test phase penalizes the learning phase, and moreover the error estimation is unreliable because the test sample size is also small.

Thus, in the small sample context, it is preferable to implement the resampling approaches for error rate estimation¹. In this tutorial, we study the behavior of the **cross validation** (cv), **leave one out** (lvo) and **bootstrap** (boot). All of them are based on the repeated train-test process, but in different configurations. We keep in mind that the aim is to evaluate the error rate of the classifier created on the whole sample. Thus, the intermediate classifiers computed on each learning session are not really interesting. This is the reason for which they are rarely provided by the data mining tools.

The main supervised learning method used is the linear discriminant analysis (LDA). We will see at the end of this tutorial that the behavior observed for this learning approach is not the same if we use another approach such as a decision tree learner (C4.5).

2. Dataset

We use a variant² of the Breiman's WAVEFORM dataset³ (Breiman et al., 1984).

¹ <http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section-12.html>

² http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/wave_ab_err_rate.zip

³ <http://mlr.cs.umass.edu/ml/datasets/Waveform+Database+Generator+%28Version+1%29>

The aim is to predict a binary class attribute from 21 continuous predictors. Compared with the original version, we removed one category (the third) of the class attribute.

Because it is an artificial dataset, we can generate as much as instances that we want. Particularly, we can generate an "infinite" size test set in order to obtain an estimation of the "true" error rate as accurate as possible. Thus we use the following experimentation scheme:

- A sample with 500 observations is the available dataset. It corresponds to the learning set for the elaboration of the classifier LDA. We compute the resubstitution error rate on this sample (*e-resub*).
- A sample with 42500 instances is the test set. The size of the sample is sufficiently high in order to obtain a reliable estimation of the true error rate of the classifier (*e-test*).
- In real study, this "infinite" size test sample is not available. We must use the available dataset (500 instances) in order to learn the classifier and assess it. If we use the hold out process, we must subdivide the 500 instances in two sub samples. It is not really efficient. There will be insufficient instances for the learning process (e.g. 350), and the limited size of the test set (e.g. 150) does not allow to obtain a reliable estimation of the error rate. Thus, It is certainly more appropriate to perform resampling approaches to obtain an honest estimate of the error rate of LDA classifier. In this tutorial,
 1. We will compare the estimated error rate by the cross validation, the leave one out and the bootstrap (*e-cv*, *e-lvo*, *e-boot*).
 2. We will see if these estimations are close to the test error rate which represents the true generalization error rate in our context.

The EXCEL workbook (wave_ab_err_rate.xls) contains 2 worksheets:

Class	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9	V_10	V_11	V_12	V_13	V_14	V_15	V_16	V_17	V_18	V_19	V_20	V_21	status
B	-0.60	-1.12	0.38	0.96	0.75	3.59	5.46	4.64	3.03	2.88	6.23	3.13	0.79	2.02	0.83	0.32	0.26	-0.22	0.87	0.12	-0.36	learning
A	0.31	0.04	3.46	3.12	0.48	5.16	4.40	3.56	3.41	4.10	0.74	1.27	1.43	0.26	-0.24	2.61	-0.72	0.52	-1.37	0.37	0.04	learning
A	0.42	-2.04	-0.11	-1.39	1.35	1.04	1.10	2.39	2.92	1.27	0.77	0.77	3.10	0.51	3.71	3.43	2.22	0.69	1.39	0.45	-0.08	learning
A	-0.14	0.41	-0.77	0.25	0.42	2.38	3.86	2.69	2.16	0.81	2.21	2.19	3.32	2.74	2.15	3.92	1.16	1.40	1.67	-0.16	-1.52	learning
A	0.06	0.39	1.51	1.79	4.38	4.52	5.29	5.33	1.46	0.03	-0.07	0.85	0.87	-0.80	-0.95	2.05	0.19	0.30	-1.32	-1.08	-0.75	learning
A	-1.70	1.59	0.84	1.24	-0.98	0.74	0.55	-0.19	-1.39	1.17	1.00	4.52	2.40	4.86	5.68	4.99	2.68	2.98	-0.28	0.70	-0.19	learning
A	0.40	0.65	-1.29	1.01	-0.76	1.67	2.07	0.67	2.51	1.42	3.64	1.89	1.92	2.82	2.79	2.06	4.28	1.80	1.97	0.41	0.22	learning
A	-0.66	0.74	1.51	1.99	2.45	2.47	3.51	3.98	1.59	2.86	2.46	2.45	2.27	3.99	3.49	1.85	2.18	0.14	0.61	1.60	-0.95	learning
A	-0.98	-1.41	3.03	0.69	2.18	3.42	4.14	2.77	3.69	1.81	2.59	0.71	1.15	3.46	1.19	1.84	1.16	0.41	0.08	-0.39	-0.45	learning
A	0.06	1.05	0.22	2.29	1.22	5.49	6.34	3.43	4.40	3.00	1.41	3.26	-0.70	0.00	1.51	1.26	1.29	-1.20	-0.39	0.45	-1.07	learning
B	-0.83	0.30	-0.60	-0.36	0.91	1.58	4.41	3.25	3.79	3.92	4.14	3.34	2.48	0.36	2.25	2.68	0.12	-1.41	-1.00	-0.55	-0.48	learning
B	0.71	1.34	0.40	1.26	2.63	2.96	4.71	4.35	2.41	4.58	2.81	0.49	1.12	1.78	-0.74	0.51	0.36	0.98	0.60	-2.20	-0.19	learning
B	-1.17	0.83	2.42	3.22	4.17	2.63	6.36	3.17	3.60	3.64	2.36	2.80	0.24	0.33	0.78	0.12	1.41	-0.59	0.05	-0.16	-2.57	learning
B	0.90	-0.84	-0.91	2.24	2.29	3.32	4.39	3.19	3.08	3.49	4.14	2.85	2.07	-0.76	0.13	-0.36	1.36	-1.07	-0.05	1.48	-1.88	learning
B	-1.40	1.23	1.34	1.35	3.57	3.36	4.61	4.98	3.28	4.38	4.95	1.32	0.86	0.25	1.80	1.09	-0.37	-0.85	-0.24	1.44	0.79	learning
B	1.09	0.32	2.28	2.48	2.88	3.56	5.23	4.02	5.56	2.11	1.71	1.81	1.92	-0.31	-0.74	1.85	0.50	0.22	0.32	1.05	1.36	learning
A	-0.84	-0.25	0.69	2.66	1.77	1.93	3.62	4.54	0.89	1.77	2.21	2.96	0.79	2.96	2.75	3.10	2.21	1.40	-1.11	-0.73	-1.55	learning
A	-0.16	-0.09	0.82	0.54	3.01	2.72	1.78	0.19	0.08	2.01	1.03	0.24	2.29	4.92	4.99	3.08	1.48	0.14	2.36	1.62	-2.53	learning
A	1.16	0.69	0.09	0.46	0.61	2.96	2.64	3.55	1.92	2.25	1.83	3.44	3.15	3.48	3.71	1.43	3.10	1.28	1.75	0.97	1.40	learning
B	0.24	0.44	0.65	0.85	1.95	2.22	3.77	4.55	4.46	3.96	3.86	1.60	2.75	0.96	0.33	0.03	0.57	1.91	-1.06	-1.15	0.99	learning
B	-1.47	2.25	-0.64	1.61	0.35	1.58	2.90	4.20	3.11	6.01	4.70	3.98	2.97	2.15	1.87	1.53	-1.02	0.43	-0.75	-1.37	-0.54	learning
A	0.08	-0.09	2.00	0.88	2.89	2.52	3.38	3.43	1.40	2.97	1.32	2.90	2.51	2.65	3.78	2.83	2.22	3.03	-0.73	-0.27	0.07	learning
B	-0.03	0.08	1.42	2.26	4.07	3.66	5.64	2.73	4.54	4.48	3.36	0.38	1.52	-0.92	0.40	-0.75	0.11	0.02	-0.19	0.51	0.43	learning
B	0.78	-0.43	0.01	2.97	2.72	6.20	3.95	6.79	4.47	2.64	2.43	2.11	1.92	0.58	-0.44	1.97	-0.39	0.14	-0.43	-2.01	-0.94	learning

- « all dataset » contains 43000 instances, an additional column states the type of the instances (status = learning or status = test);
- « only learning set » contains 500 instances, they corresponds to the “status = learning” of the previous worksheet.

The screenshot shows an Excel spreadsheet with columns labeled A through W and rows numbered 481 to 501. The data consists of numerical values in columns A through S, followed by a 'status' column with values like 'learning'. At the bottom, there are two worksheet tabs: 'all dataset' and 'only learning set'. A blue arrow points to the 'only learning set' tab, which is highlighted with a red dashed box. The status bar at the bottom indicates 'Prêt' and 'NUM'.

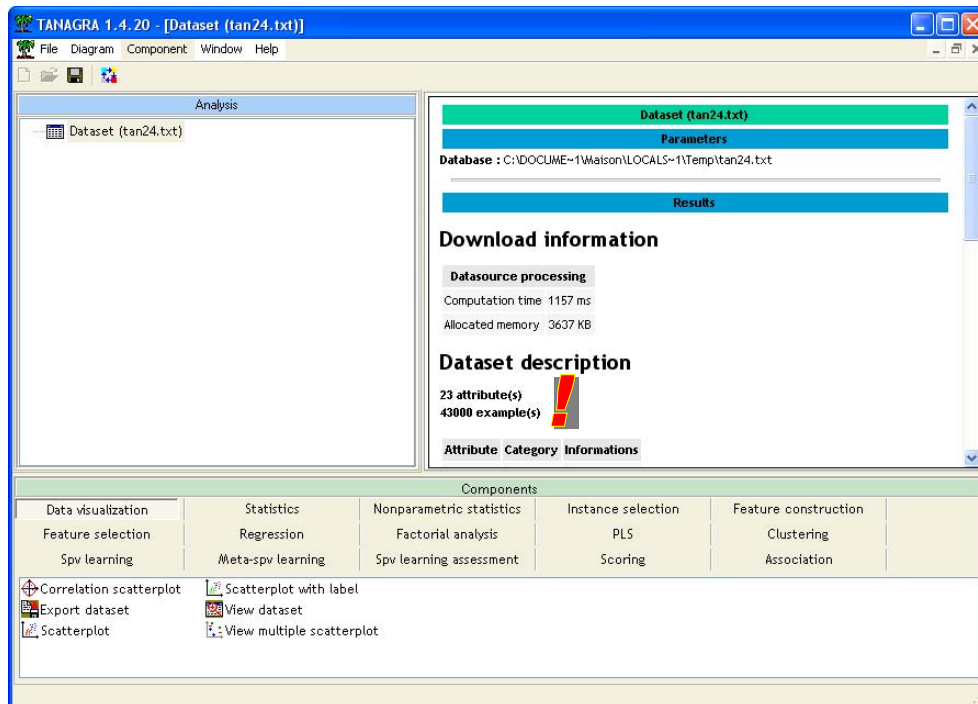
3. Resubstitution error rate and test error rate

First, we work with the “all dataset” worksheet. We select the range of cells and we click on the TANAGRA / EXECUTE TANAGRA menu⁴.

The screenshot shows the 'Execute Tanagra' dialog box open over the 'all dataset' worksheet. The dialog box has a text input field for 'Dataset range (including the name of the attributes -- first row):' with the value '\$A\$1:\$W\$43001' entered. There are 'OK' and 'Cancel' buttons. The background spreadsheet shows columns A through W and rows 1 through 25. The status bar at the bottom indicates 'Somme=1548583.57' and 'NUM'.

⁴ See <http://data-mining-tutorials.blogspot.com/2008/10/excel-file-handling-using-add-in.html> about the installation of the TANAGRA.XLA add-in into EXCEL.

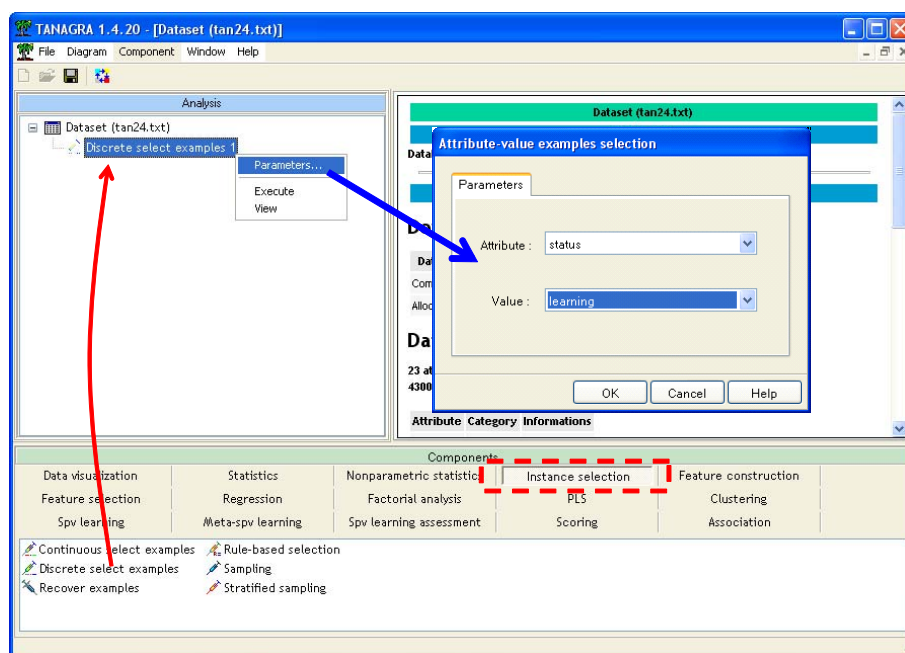
We check the cells range coordinates. Then we click on the OK button. Tanagra is automatically launched and the dataset is imported.



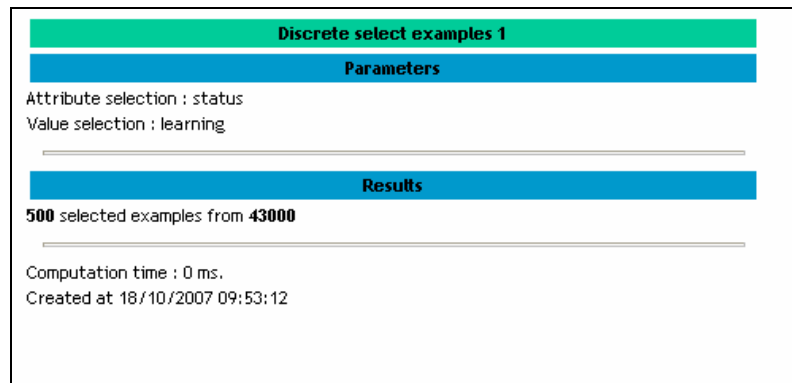
We have 23 columns i.e. 21 continuous descriptors, the class attribute, the column which states the instances membership (learning or test sample). There are 43000 instances.

Subdivision into training and test sets

We want to subdivide the dataset into train and test sets. We use the STATUS column for that. We insert the DISCRETE SELECT EXAMPLES component (INSTANCE SELECTION tab) into the diagram. We activate the PARAMETERS menu. The active examples corresponds to STATUS = LEARNING.

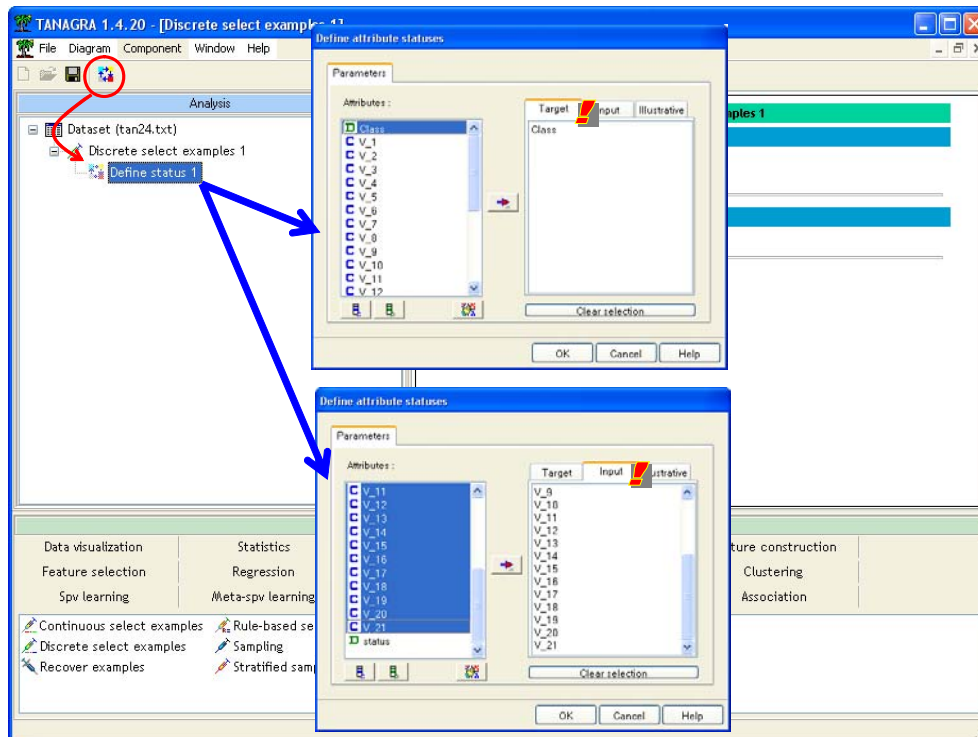


We validate and we click on the VIEW menu. We see that the learning set size is 500.



Class attribute and predictive attributes

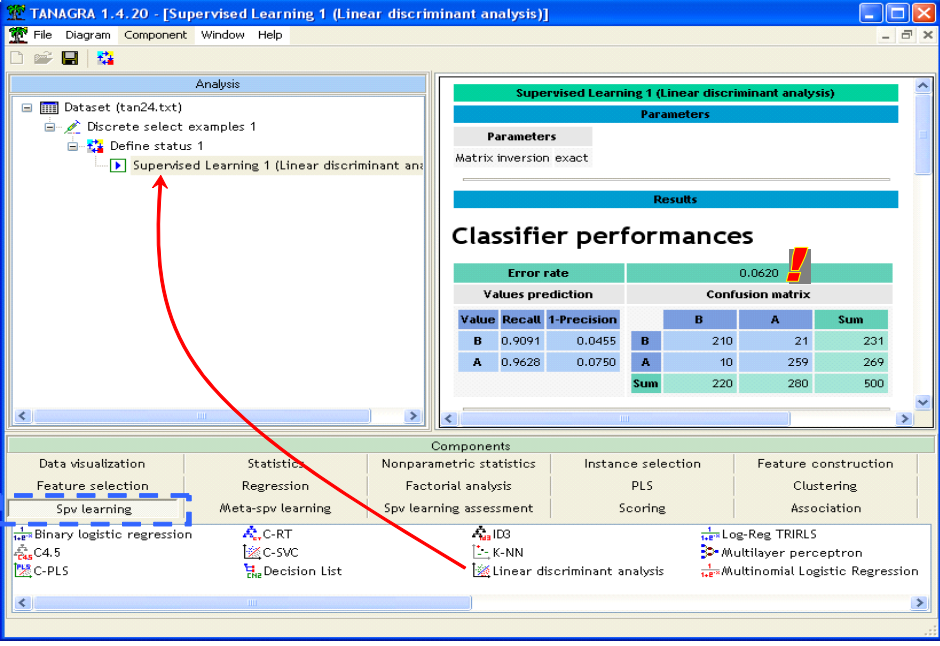
By the shortcut into the tool bar, we add the DEFINE STATUS component into the diagram. We set CLASS as TARGET attribute; the continuous variables (V1 to 21) as INPUT one. We do not use the STATUS column at this step.



Linear discriminant analysis and the resubstitution error rate

We want now to learn the classifier. We add the LINEAR DISCRIMINANT ANALYSIS (SPV LEARNING tab) into the diagram. We activate the VIEW menu.

The coefficients of the model are not really essential in our context. We analyze mainly the accuracy of the classifier. Thus, we inspect the confusion matrix and the resubstitution error rate in the first part of the report.



The screenshot shows the TANAGRA 1.4.20 interface for supervised learning. The main window displays the 'Supervised Learning 1 (Linear discriminant analysis)' component. The 'Classifier performances' section shows the following results:

Error rate		0.0620	
Values prediction			
Confusion matrix			
Value	Recall	1-Precision	
B	0.9091	0.0455	
A	0.9628	0.0750	

The confusion matrix is as follows:

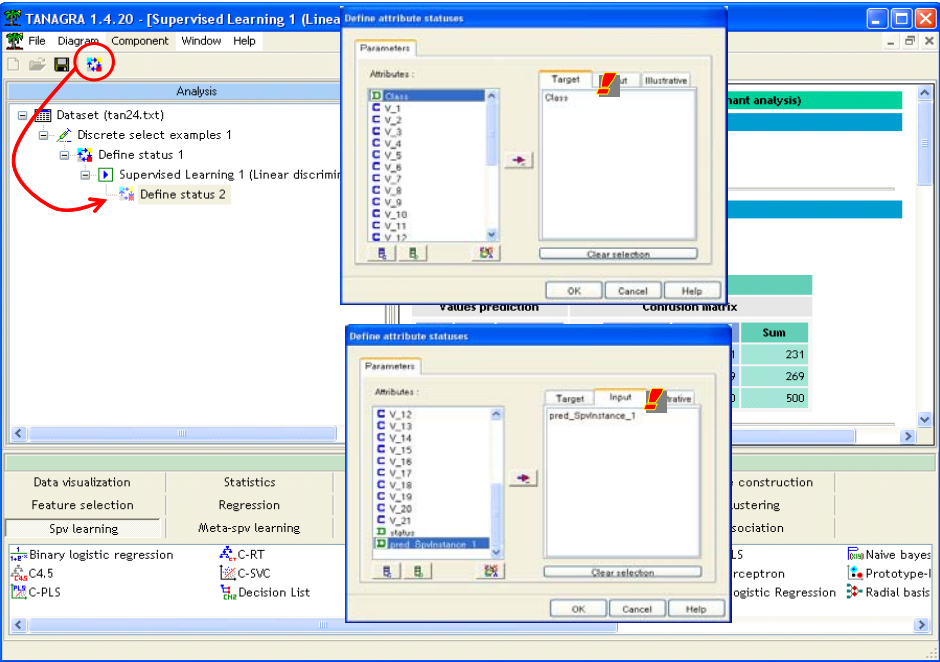
	B	A	Sum
B	210	21	231
A	10	259	269
Sum	220	280	500

The resubstitution error rate is $e_{-resub} = 6.2\%$ i.e. the probability of misclassifying an unseen instance with the classifier is 6.2%. When we compute the test error rate below, we will relativize this result.

Test error rate

In order to compute the test error rate, we must apply the classifier on the unselected examples and compare the observed values with the predicted values of the class attribute. Tanagra generates automatically a new column after a supervised learning component. It contains the predicted values of the classifier, on the learning set, but also on the unselected examples. We go to create the confusion matrix on this second sample which contains 42500 instances.

We add the DEFINE STATUS component. We set CLASS as TARGET and the generated column, PRED_SPV_INSTANCE_1, as INPUT.



The screenshot shows the TANAGRA 1.4.20 interface with the 'Define attribute statuses' dialog box open. The 'Target' is set to 'Class' and the 'Input' is set to 'pred_SpvInstance_1'. The 'Attributes' list includes V_1 through V_12. The 'Define attribute statuses' dialog box is also open, showing the 'Target' set to 'Class' and the 'Input' set to 'pred_SpvInstance_1'.

Then we add the TEST component (SPV LEARNING ASSESSMENT tab). We click on the VIEW menu. The component provides the confusion matrix and computes the error rate.

The screenshot shows the TANAGRA 1.4.20 interface. The 'Analysis' tree on the left shows a project structure with 'Dataset (tan24.txt)', 'Discrete select examples 1', 'Define status 1', 'Supervised Learning 1 (Linear discriminant analysis)', 'Define status 2', and 'Test 1'. A red arrow points from 'Test 1' to the results window. The results window displays the following information:

Test 1
Parameters
Evaluation set : unselected examples

Results
pred_Spvinstance_1

Error rate			Confusion matrix			
0.0839						
Values prediction						
Value	Recall	1-Precision	B	A	Sum	
B	0.9200	0.0880	B	19437	1690	21127
A	0.9123	0.0798	A	1875	19498	21373
Sum						42500

Computation time : 0 ms.
Created at 18/10/2007 10:29:53

The 'Components' panel at the bottom shows various analysis options, with 'Spv learning assessment' highlighted in blue.

Remark 1: We can set many columns as INPUT. It allows for instance to compare the accuracies of various supervised learning algorithms.

Remark 2: By default, the component creates the confusion matrix on the unselected examples i.e. the test set. But we can also compute this one on the active examples i.e. the learning set. In this case, we must find the same confusion matrix as above (the one of the supervised learning component).

The test error rate is 8.39%. In our context, because we had been able to generate an "infinite" size test set, we can consider that this value is near to the true generalization error rate. Again, we note that this kind of test set is not available in real applications. The test set is often a sample extracted from the dataset. Its size is restricted.

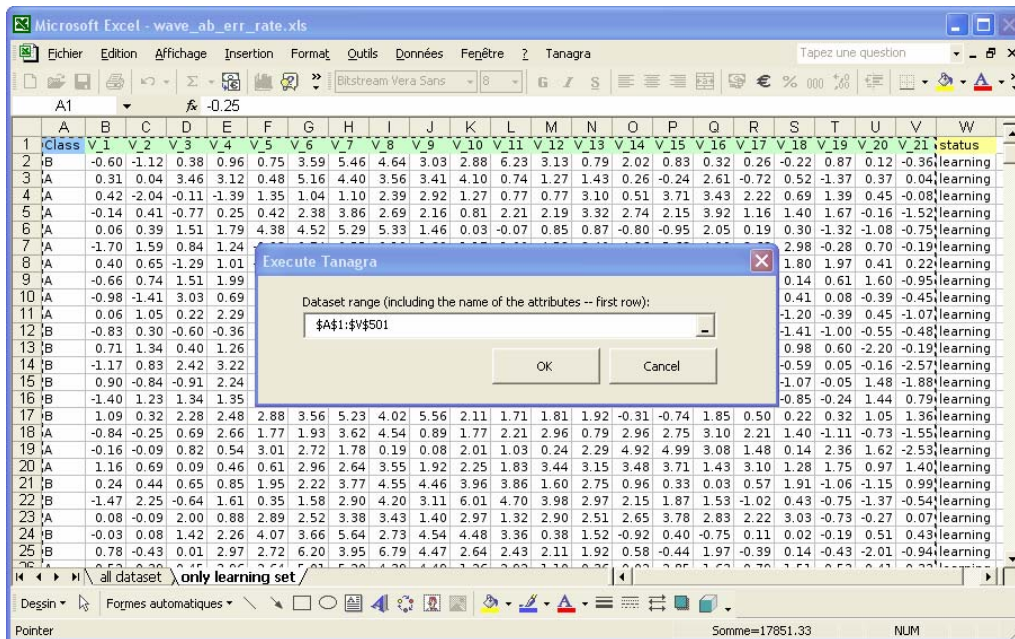
Remark 3: The true error rate is worse than the resubstitution error rate. The model is not as good we thought, even if the deviation from the two measures is not spectacular for LDA. We will see later that the deviation can be higher for some learning methods.

4. Resampling approaches for error rate estimation

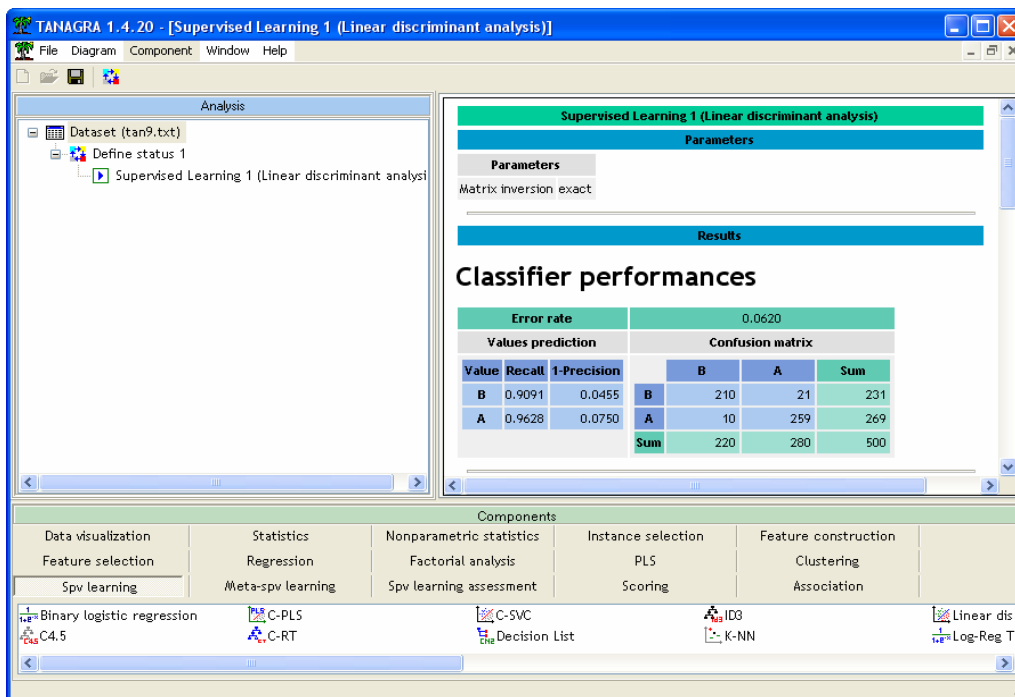
We can close Tanagra at this step of our analysis.

In the EXCEL workbook, we select now the second worksheet "ONLY LEARNING SET". In real applications, the only available dataset is this sample with 500 instances. We must use it both the learning and the evaluation of the classifier. Clearly, the hold out approach is not adapted here. It is more appropriate to implement the resampling methods for the error rate estimation.

As above, we select the range of cells. Here, it is not useful to select the STATUS column; all rows have the same value "LEARNING". The range selection is **\$A\$1:\$V\$501** (L1C1:L501C22)



Tanagra is automatically launched. With the DEFINE STATUS component, we set CLASS as TARGET, the others (V1 to V21) as INPUT. Then we add the LINEAR DISCRIMINANT ANALYSIS component. The resubstitution error rate is 6.2%, exactly the same as above. This is obvious. We use the same dataset for the learning process.



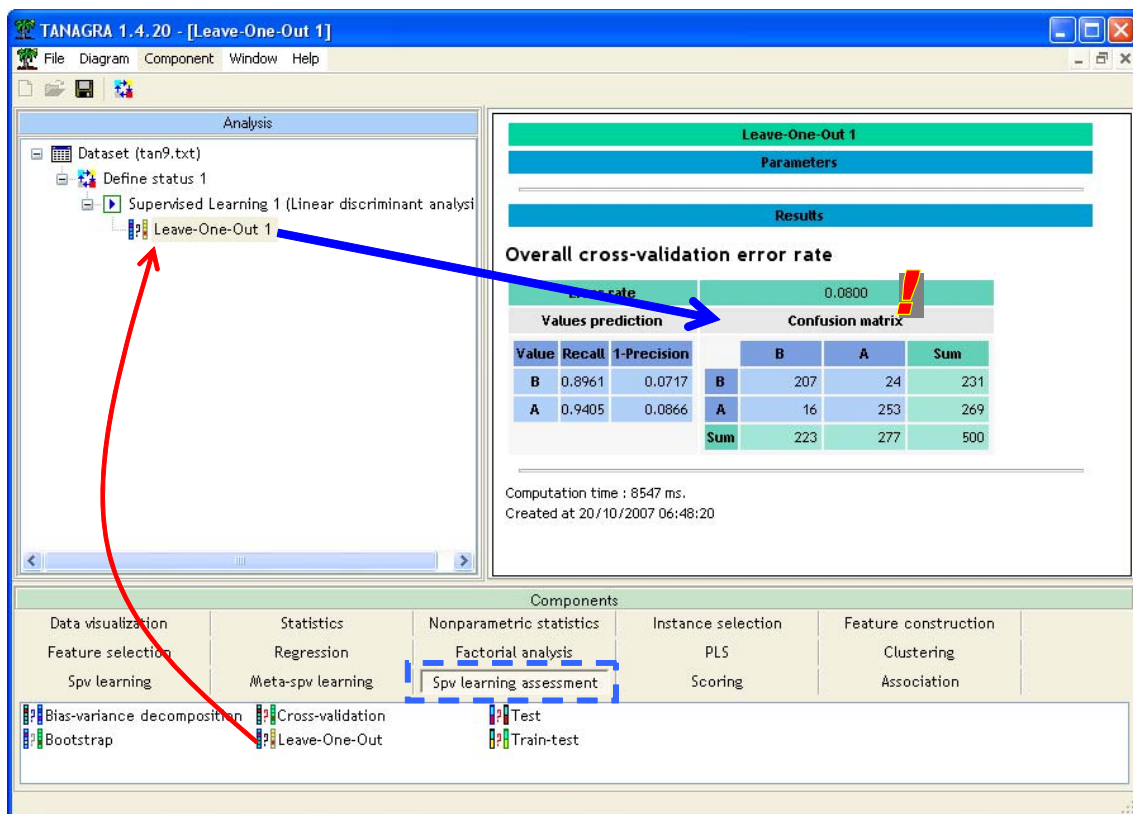
In the following, we use various resampling error rate measures components to estimate the generalization error rate. The better will be the nearest to the true error rate 8.39% computed on the "infinite sample size" test sample above (42500 instances).

Leave one out

The idea underlying the "leave one out" is to remove one instance for the dataset, creating the classifier on the remaining instance, checking the correctness of the prediction on the removed instance. We repeat this process for all the instances. If we set ($d_i = 1$ if the instance is misclassified, $d_i = 0$ otherwise), the leave one out error rate is

$$e - lvo = \frac{1}{n} \sum_i d_i$$

With TANAGRA, we add the LEAVE ONE OUT component (SPV LEARNING ASSESSMENT tab) into the diagram. We click on the VIEW menu. Tanagra repeats 500 times the process "learning on 499 instances and classifying 1 test instance". And yet, the computation time remains reasonable (~8 seconds on my computer).



The leave one out error rate is 8%. Usually, this approach has a low bias but a higher variance than the other approaches. It is not adapted if the classifiers highly overfit the dataset.

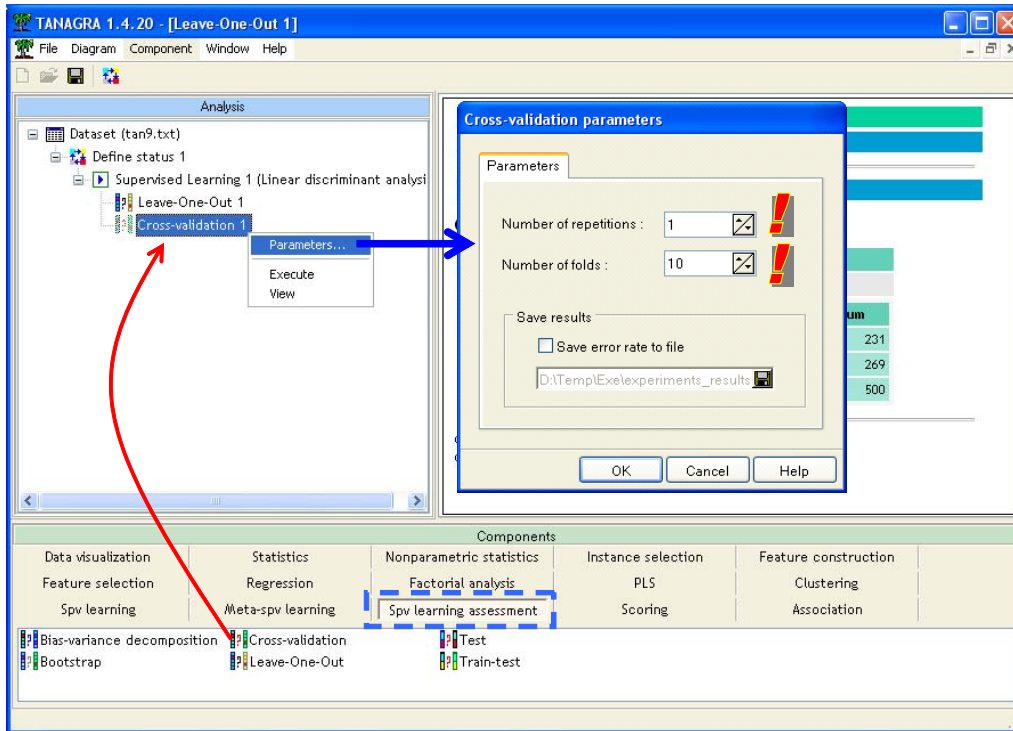
Cross validation

The cross validation is a variant of the leave one out where we subdivide the dataset into K folds. We repeat the following process: learning on the instances of $(K-1)$ fold, testing on the instances of the K^{th} fold. The cross validation error rate is

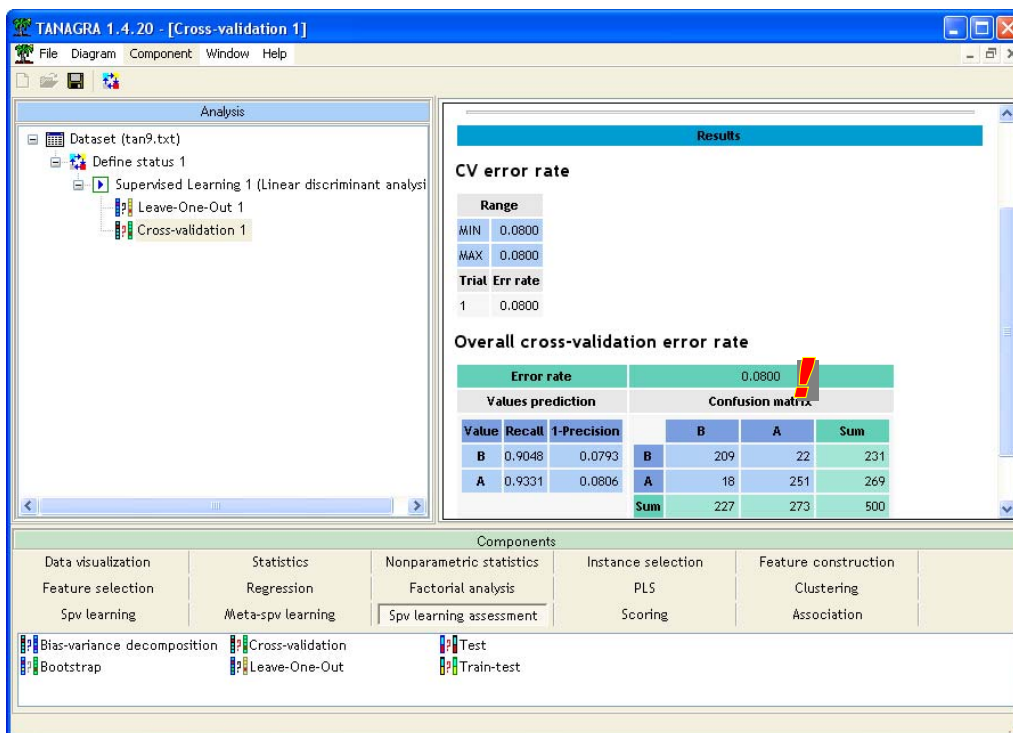
$$e - cv = \frac{1}{K} \sum_k e_k$$

The main advantage is to reduce the amount of calculations by preserving the reliability of the estimation. In general, $K = 10$ seems to be a good compromise.

We add the CROSS VALIDATION component into the diagram. We click on the PARAMETERS menu, we set the following settings.



We set $K=10$ (NUMBER OF FOLDS). It is possible to repeat the process. In our experiment we set NUMBER OF REPETITIONS = 1. We validate and we click on the VIEW menu.

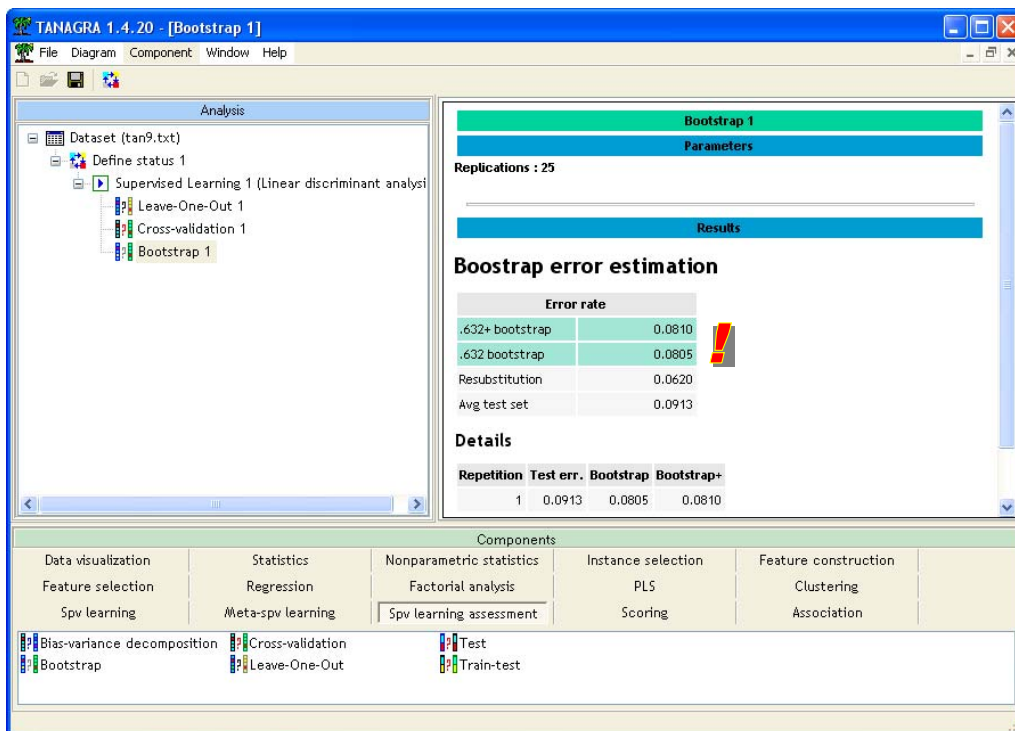


The cross validation error rate is 8 %, the same as the one of the leave one out. It is a coincidence. We note however that in the majority of cases, they provide similar estimations.

Bootstrap

Unlike the other resampling methods, bootstrap approach does not provide a direct estimation of the error rate but rather an estimation of the bias of the resubstitution error rate. There are 2 variants of the bootstrap: the standard 0.632 bootstrap and the 0.632+ bootstrap which intends to take into account the behavior of the classifier that we want to evaluate. The only parameter is the number of replication. In the majority of cases, 25 replications supply a satisfactory result.

We add the BOOTSTRAP component into the diagram. We click on the VIEW menu.



The 0.632+ bootstrap error rate is $e\text{-}boot+ = 8.1\%$.

We summarize the various results in the following table:

Method	LDA	Deviation
Resubstitution	6.2 %	- 2.19
« True » (test)	8.39 %	x
LVO	8 %	- 0.39
CV (10)	8 %	- 0.39
Bootstrap+ (25)	8.1 %	- 0.29

We recall that the true error rate is in fact estimated on a separate test set in this tutorial. Because, the size of this test set is enough large, we can consider that it is a reliable estimation of the true generalization error rate.

About the resampling methods, we observe that they provide similar estimations. The best one seems the bootstrap approach. But we must not conclude hasty conclusion. **The accuracy of the estimation depends on the characteristics of the dataset and of the learning method.**

All the resampling approaches underestimate the true error rate in our experiment. It is a coincidence. Sometimes, they can provide an estimation which is higher than the true error rate.

5. Conclusion

In this tutorial, we use various resampling estimation scheme for estimating the true generalization error rate of a classifier. The reference is the error rate estimated on an "infinite sample size" test set. We outline the main results from an experiment based on the LDA classifier. Here, we have lead the same experiment with another learning method, C4.5 induction tree algorithm, which has different characteristics (especially, it can overfit highly the learning set when the tree size is excessive) in comparison to the LDA. We obtain the following results.

Method	C4.5	Deviation
Resubstitution	2.2%	- 11.34
« True » (test)	13.54%	x
LVO	16%	+ 2.46
CV (10)	16.2%	+ 2.66
Bootstrap+ (25)	12.24%	- 1.3

First, we note that C4.5 is not efficient in our prediction problem, compared with the LDA. The true generalization error rate is much higher.

About the resubstitution error rate, it is very optimistic here. In general, we must not ever consider the resubstitution error rate when we deal with a decision tree.

About the resampling approach, the accuracy of the estimation is lesser. "Leave one out" and "cross validation" overestimate the error rate, "bootstrap" underestimates it.