1. Subject

Three curves for the evaluation of supervised learning methods.

Evaluation of classifiers is an important step of the supervised learning process. We want to measure the performance of the classifier. On one hand we have the confusion matrix and associated indicators, very popular into the academic publications. On the other hand, in the real applications, the users prefers some curves which seem very mysterious for people outside the domain (e.g. **ROC curve** for the epidemiologists, **gain chart or cumulative lift curve** in the marketing domain, **recall-precision curve** in the information retrieval domain, etc.).

In this tutorial, we give first the details of the calculation of these curves by creating them "at the hand" in a spreadsheet. Then, we use Tanagra 1.4.33 and R 2.9.2 for obtaining them. We use these curves for the comparison the performances of the logistic regression and support vector machine (Radial Basis Function kernel) algorithms.

1.1. The confusion matrix and the related indicators

The first way to evaluate a classifier is to compare the observed values of the class attribute Y and the predicted values of the classifier $\hat{Y}$ on a dataset. For this, the confusion matrix is a very popular tool (http://en.wikipedia.org/wiki/Confusion_matrix).

Let's take an example to illustrate the mechanism. We want to predict the occurrence of heart disease (DISEASE: positive or negative) using a logistic regression approach. The predictive attributes are the characteristics of patients (CHOLESTERAL, THALAC and OLDPEAK; see http://archive.ics.uci.edu/ml/datasets/Heart+Disease for the description of the variables). We apply the classifier on a test set with n = 20 instances.

cholesteral	thalac	oldpeak	disease	Prediction
322	109	24	positive	**positive**
564	160	16	negative	*positive*
234	161	5	negative	**negative**
311	120	18	positive	**positive**
233	179	4	negative	**negative**
197	116	11	negative	*positive*
309	147	0	positive	*negative*
237	71	10	positive	**positive**
233	163	6	positive	*negative*
224	173	32	positive	**positive**
244	154	14	positive	*negative*
325	154	0	negative	**negative**
282	174	14	positive	*negative*
234	131	1	negative	**negative**
239	160	12	negative	**negative**
213	165	2	negative	**negative**
204	202	0	negative	**negative**
258	147	4	negative	**negative**
263	97	12	positive	**positive**
219	140	12	positive	*negative*

We see above the table containing the data and the prediction of the logistic regression (good predictions in bold, bad in italics).

The generic form of the confusion matrix for a binary problem is the following.

Obs. vs. Pred.	Positive	Negative	Total
Positive	a	b	a + b
Negative	c	d	c + d
Total	a + c	b + d	n = a + b + c + d

About our dataset, we have

Nombre de disease	Prediction		
disease	positive	negative	Total
positive	5	5	10
negative	2	8	10
Total	7	13	20

We can the compute some performance indicators [http://en.wikipedia.org/wiki/Specificity_(tests)]

- The error rate

$$\varepsilon = \frac{c+b}{n} = 1 - \frac{a+d}{n} = \frac{2+5}{20} = 0.35$$

- We call "target" the instances which are classified as positive i.e.

$$\text{target} = \left\{\omega, \hat{Y}(\omega) = +\right\}$$

  The target size is

$$\#\text{target} = a + c = 7$$

- The sensibility (or recall, or true positive rate - TPR) is defined as $P(\omega \in \text{cible} / Y(\omega) = +)$. From the confusion matrix, we have

$$r = Se = \frac{a}{a+b} = \frac{5}{5+5} = 0.5$$

- The precision is $P(Y(\omega) = + / \omega \in \text{cible})$. The estimation from the confusion matrix is

$$p = \frac{a}{a+c} = \frac{5}{7} = 0.71$$

- The false positive rate (FPR)

$$FPR = \frac{c}{c+d} = \frac{2}{2+8} = 0.2$$

- The specificity

$$Sp = \frac{d}{c+d} = 1 - FPR = \frac{8}{2+8} = 0.8$$

A "good" classifier must have, on the one hand, a high recall and, on the second hand, a high precision and specificity. In practice, we note that when we try to increase the recall, we often decrease both the precision and the specificity, and vice versa.

1.2. The class membership posterior probability – The score value

The prediction of a classifier is often (not always) based on their estimated posterior probability to belong to the positive group, says also "**score**"

$$\hat{\pi}(\omega) = \hat{P}\left[Y(\omega) = +/X(\omega)\right]$$

The classification rule is then

$$\text{Si } \hat{\pi}(\omega) \geq \text{threshold Alors } \hat{Y}(\omega) = + \text{ Sinon } \hat{Y}(\omega) = -$$

The threshold is usually **0.5**. It is consistent to the rule which consists in to assign the label which maximizes the posterior probability.

cholesteral	thalac	oldpeak	disease	score	prediction
322	109	24	positive	**0.9335**	**positive**
564	160	16	negative	**0.8897**	**positive**
234	161	5	negative	0.2417	negative
311	120	18	positive	**0.8537**	**positive**
233	179	4	negative	0.1349	negative
197	116	11	negative	**0.6427**	**positive**
309	147	0	positive	0.3956	negative
237	71	10	positive	**0.9183**	**positive**
233	163	6	positive	0.2397	negative
224	173	32	positive	**0.5433**	**positive**
244	154	14	positive	0.4468	negative
325	154	0	negative	0.3650	negative
282	174	14	positive	0.3446	negative
234	131	1	negative	0.4146	negative
239	160	12	negative	0.3546	negative
213	165	2	negative	0.1620	negative
204	202	0	negative	0.0406	negative
258	147	4	negative	0.3696	negative
263	97	12	positive	**0.8608**	**positive**
219	140	12	positive	0.4910	negative

Figure 1 - Score and prediction based on the discrimination threshold 0.5

We observe the correspondence between the scores supplied by the logistic regression and the predicted values on the DISEASE dataset (Figure 1).

1.3. Modifying the discrimination threshold

The threshold 0.5 is in fact optimal in a well defined context: the dataset is randomly drawn from the population; the misclassification cost matrix is the identity matrix.

Rather than being limited to this threshold, we can assess more broadly the behavior of the classifier in varying the threshold and calculating the confusion matrix for each candidate discrimination threshold. This idea underlies the different graphs that we present in this section.

The individuals which belong to the positive group have higher score [$score(\omega)$][1] than the negative one. So we can sort the dataset into a decreasing order according to the score. The positive examples are in majority in the high part of the table, the negative instances in the low part.

[1] A score is initially defined as a probability. But, we can use any value characterizing the degree of membership of an individual to the positive group.

cholesteral	thalac	oldpeak	disease	score
322	109	24	positive	0.9335
237	71	10	positive	0.9183
564	160	16	negative	0.8897
263	97	12	positive	0.8608
311	120	18	positive	0.8537
197	116	11	negative	0.6427
224	173	32	positive	0.5433
219	140	12	positive	0.4910
244	154	14	positive	0.4468
234	131	1	negative	0.4146
309	147	0	positive	0.3956
258	147	4	negative	0.3696
325	154	0	negative	0.3650
239	160	12	negative	0.3546
282	174	14	positive	0.3446
234	161	5	negative	0.2417
233	163	6	positive	0.2397
213	165	2	negative	0.1620
233	179	4	negative	0.1349
204	202	0	negative	0.0406

Figure 2 - The dataset ranked according to the score (decreasing order)

We note that by varying the threshold, we can obtain various version of the confusion matrix.

<table>
<tr><th>Threshold</th><th>Confusion matrix and related indicators</th></tr>
<tr><td>0.9335</td><td>
<table>
<tr><td>Obs. x Préd.</td><td>positif</td><td>négatif</td><td>total</td></tr>
<tr><td>positif</td><td>1</td><td>9</td><td>10</td></tr>
<tr><td>négatif</td><td>0</td><td>10</td><td>10</td></tr>
<tr><td>total</td><td>1</td><td>19</td><td>20</td></tr>
</table>
<table>
<tr><td>#target</td><td>1</td></tr>
<tr><td>recall</td><td>0.10</td></tr>
<tr><td>TPR</td><td>0.00</td></tr>
<tr><td>specificity</td><td>1.00</td></tr>
</table>
</td></tr>
<tr><td>0.9183</td><td>
<table>
<tr><td>Obs. x Préd.</td><td>positif</td><td>négatif</td><td>total</td></tr>
<tr><td>positif</td><td>2</td><td>8</td><td>10</td></tr>
<tr><td>négatif</td><td>0</td><td>10</td><td>10</td></tr>
<tr><td>total</td><td>2</td><td>18</td><td>20</td></tr>
</table>
<table>
<tr><td>#target</td><td>2</td></tr>
<tr><td>recall</td><td>0.20</td></tr>
<tr><td>TPR</td><td>0.00</td></tr>
<tr><td>specificity</td><td>1.00</td></tr>
</table>
</td></tr>
<tr><td>0.8897</td><td>
<table>
<tr><td>Obs. x Préd.</td><td>positif</td><td>négatif</td><td>total</td></tr>
<tr><td>positif</td><td>2</td><td>8</td><td>10</td></tr>
<tr><td>négatif</td><td>1</td><td>9</td><td>10</td></tr>
<tr><td>total</td><td>3</td><td>17</td><td>20</td></tr>
</table>
<table>
<tr><td>#target</td><td>3</td></tr>
<tr><td>recall</td><td>0.20</td></tr>
<tr><td>TPR</td><td>0.10</td></tr>
<tr><td>specificity</td><td>0.90</td></tr>
</table>
</td></tr>
<tr><td>0.8608</td><td>
<table>
<tr><td>Obs. x Préd.</td><td>positif</td><td>négatif</td><td>total</td></tr>
<tr><td>positif</td><td>3</td><td>7</td><td>10</td></tr>
<tr><td>négatif</td><td>1</td><td>9</td><td>10</td></tr>
<tr><td>total</td><td>4</td><td>16</td><td>20</td></tr>
</table>
<table>
<tr><td>#target</td><td>4</td></tr>
<tr><td>recall</td><td>0.30</td></tr>
<tr><td>TPR</td><td>0.10</td></tr>
<tr><td>specificity</td><td>0.90</td></tr>
</table>
</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>0.0406</td><td>
<table>
<tr><td>Obs. x Préd.</td><td>positif</td><td>négatif</td><td>total</td></tr>
<tr><td>positif</td><td>10</td><td>0</td><td>10</td></tr>
<tr><td>négatif</td><td>10</td><td>0</td><td>10</td></tr>
<tr><td>total</td><td>20</td><td>0</td><td>20</td></tr>
</table>
<table>
<tr><td>#target</td><td>20</td></tr>
<tr><td>recall</td><td>1.00</td></tr>
<tr><td>TPR</td><td>1.00</td></tr>
<tr><td>specificity</td><td>0.00</td></tr>
</table>
</td></tr>
</table>

We will note several interesting properties:

- When we decrease the threshold, the target size (#target) increases. It is quite mechanical. We incorporate more observations in the positive predictions. When we test all possible configurations, and there are not ties on the scores, the increase is one observation of a threshold to another.
- When we decrease the threshold, the recall tends to increase. Indeed, we capture primarily the positive instances that have a higher score than the negatives.
- But in the same time, we often decrease the precision and specificity: we increase the chance to include negative individuals into the target. The false positive rate tends to increase.

The curves which characterize the performance of classifiers are based on these properties. They describe the values of the various indicators according to the successive discrimination thresholds:

- **The ROC curve** (Receiver Operating Characteristic) is a graphical plot of the "sensitivity vs. the (1 – specificity) [i.e. the false positive rate]"[2]. It measures the ability of the classifier to assign a higher score to the positive instances.
- The **gain chart** (or cumulative lift chart) is often used in the marketing domain. It is a graphical plot of the "sensitivity (recall) vs. the target size (in percentage)" [3].
- The **precision recall curve** is mainly computed in the information retrieval domain. It is a graphical plot of the "recall vs. precision"[4].

Of course, because the successive values of the indicators are computed from the same confusion matrices, these curves were strong connections between them.

These curves allow characterization of the classifiers. We can use them also in order to compare their performance. It is thus possible to establish dominations between classifiers. For instance, if the ROC curves of A is always located above that of B, we know it will always be better whatever the combination of misclassification costs used. It also showed that in this case the recall and precision curve of A is always located above that of B.

In our spreadsheet, since the table is already descending by score, we can calculate the quantities (a, b, c and d) of confusion matrices according to successive discrimination threshold values. We can then compute the indicators used to plot the curves (TPR, recall, precision, FPR).

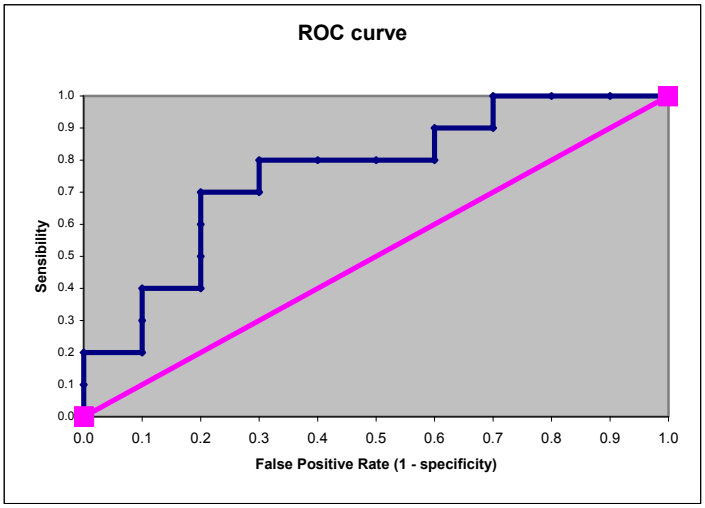
[2] http://en.wikipedia.org/wiki/Receiver_operating_characteristic

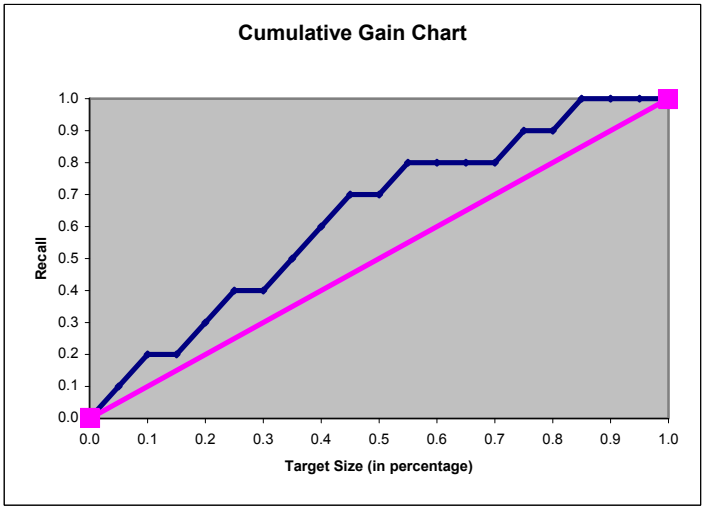
[3] http://www2.cs.uregina.ca/~hamilton/courses/831/notes/lift_chart/lift_chart.html

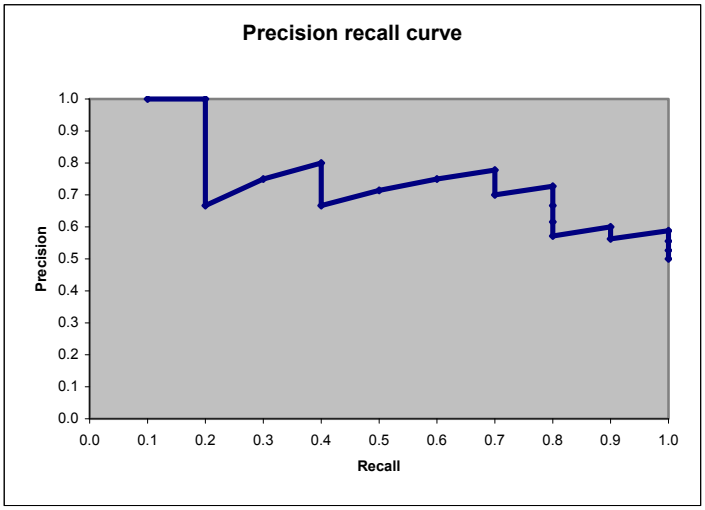
[4] http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-ranked-retrieval-results-1.html

cholesteral	thalac	oldpeak	disease	score	a	b	c	d	TPR	recall	precision	FPR
322	109	24	positive	0.9335	1	9	0	10	0.05	0.10	1.00	0.00
237	71	10	positive	0.9183	2	8	0	10	0.10	0.20	1.00	0.00
564	160	16	negative	0.8897	2	8	1	9	0.15	0.20	0.67	0.10
263	97	12	positive	0.8608	3	7	1	9	0.20	0.30	0.75	0.10
311	120	18	positive	0.8537	4	6	1	9	0.25	0.40	0.80	0.10
197	116	11	negative	0.6427	4	6	2	8	0.30	0.40	0.67	0.20
224	173	32	positive	0.5433	5	5	2	8	0.35	0.50	0.71	0.20
219	140	12	positive	0.4910	6	4	2	8	0.40	0.60	0.75	0.20
244	154	14	positive	0.4468	7	3	2	8	0.45	0.70	0.78	0.20
234	131	1	negative	0.4146	7	3	3	7	0.50	0.70	0.70	0.30
309	147	0	positive	0.3956	8	2	3	7	0.55	0.80	0.73	0.30
258	147	4	negative	0.3696	8	2	4	6	0.60	0.80	0.67	0.40
325	154	0	negative	0.3650	8	2	5	5	0.65	0.80	0.62	0.50
239	160	12	negative	0.3546	8	2	6	4	0.70	0.80	0.57	0.60
282	174	14	positive	0.3446	9	1	6	4	0.75	0.90	0.60	0.60
234	161	5	negative	0.2417	9	1	7	3	0.80	0.90	0.56	0.70
233	163	6	positive	0.2397	10	0	7	3	0.85	1.00	0.59	0.70
213	165	2	negative	0.1620	10	0	8	2	0.90	1.00	0.56	0.80
233	179	4	negative	0.1349	10	0	9	1	0.95	1.00	0.53	0.90
204	202	0	negative	0.0406	10	0	10	0	1.00	1.00	0.50	1.00

We obtain the curves.







In this tutorial, we will see how to obtain these curves using **TANAGRA 1.4.33** (or later) and **R 2.9.2**. Beyond the simple construction of curves, we use them to compare the performances of logistic regression and SVM (support vector machine – Radial Basis Function kernel) on our data.

2. Dataset

We use the HEART_DISEASE_FOR_CURVES.XLS[5] data file. We have 270 observations. Compared to the configuration above where we calculate the coordinates "at the hand" in Excel on a test sample of 20 observations, we modified the test sample size in order to obtain more reliable results: the train sample size is 150; the test sample size is now 200. Thus, we will obtain more smoothed curves.

There is an additional column into the dataset, the SAMPLE column specifies the status of each instance (belonging to train or test sample) (Figure 3).

sample	cholesteral	thalac	oldpeak	disease
train	261	141	3	positive
train	263	105	2	negative
train	269	121	2	negative
train	177	140	4	negative
train	256	142	6	positive
train	239	142	12	positive
train	293	170	12	positive
train	407	154	40	positive
train	226	111	0	negative
train	235	180	0	negative

Figure 3 – The 10 first instances of the DISEASE data file

3. Construction of curves under Tanagra

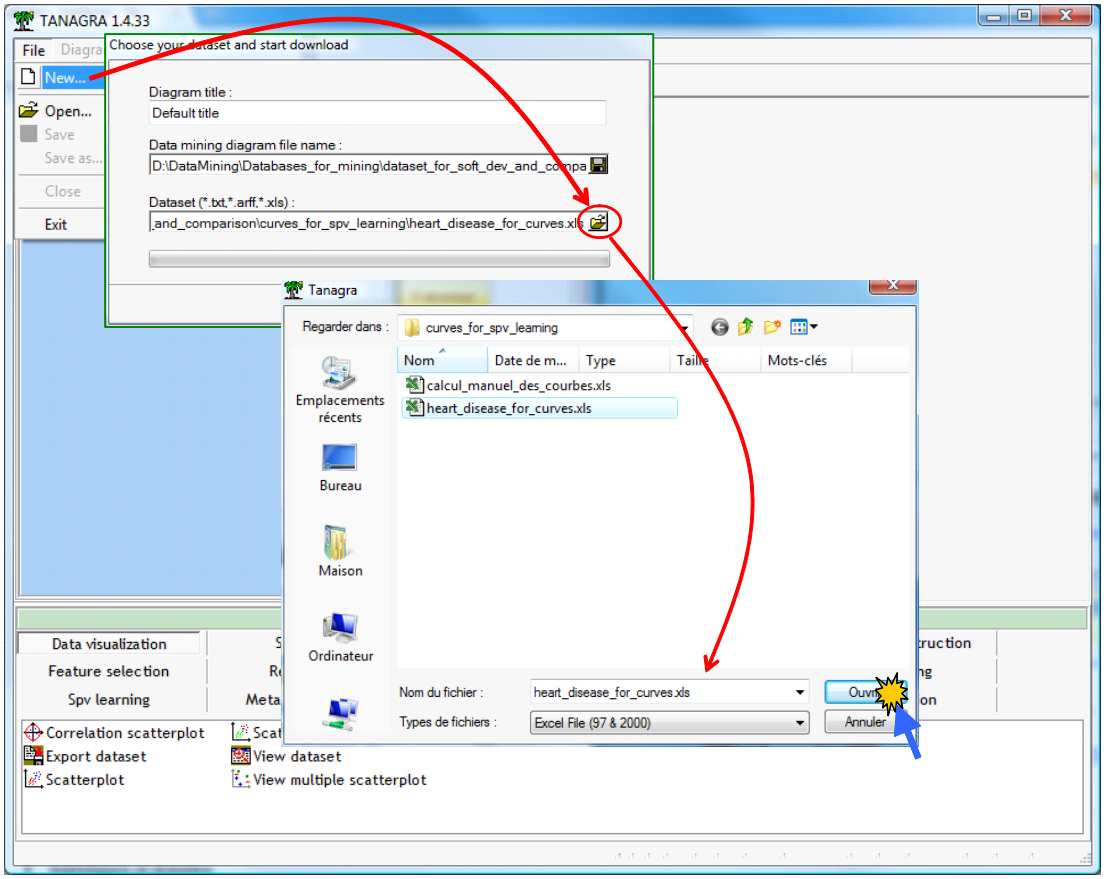
3.1. Importing and preparing the samples

Tanagra can handle directly the XLS file format. The dataset must be into the first sheet[6].

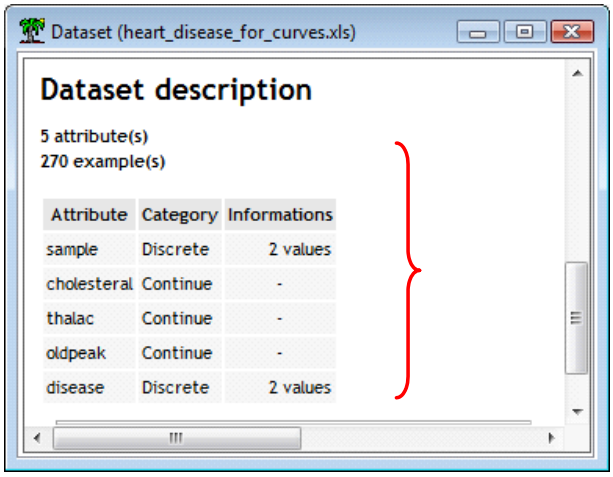
[5] http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/heart_disease_for_curves.zip

[6] See http://data-mining-tutorials.blogspot.com/2008/10/excel-file-format-direct-importation.html

We launch Tanagra, we click on the File / New menu to create a new diagram. We select the HEART_DISEASE_FOR_CURVES.XLS data file.

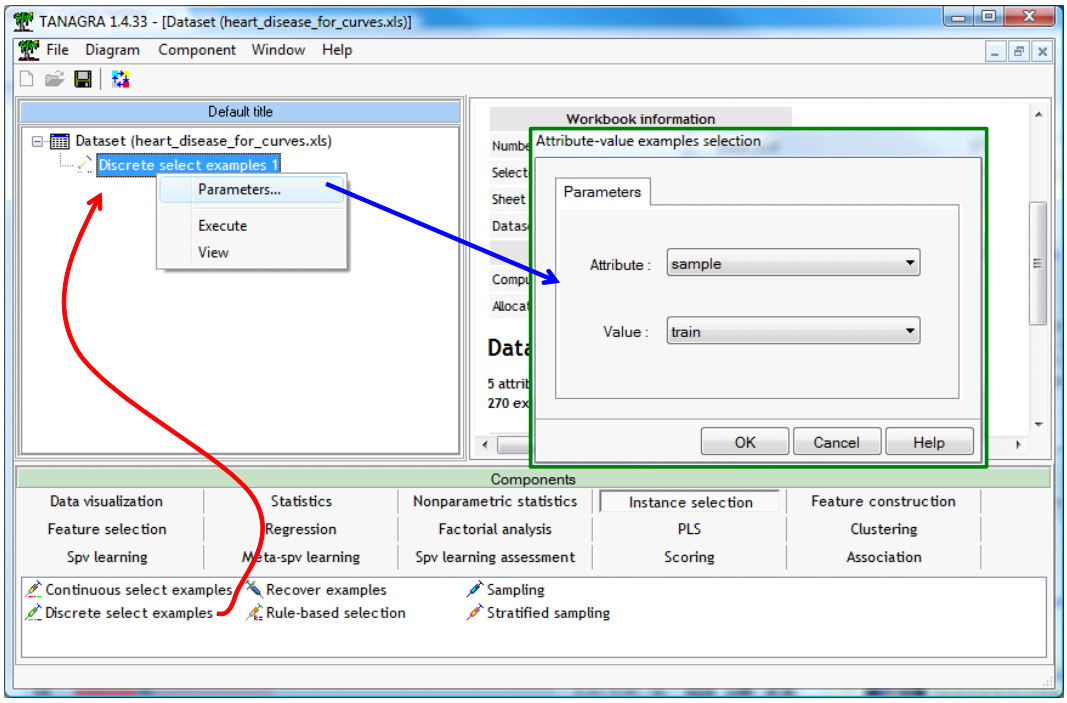


The file is automatically loaded. There are 270 observations and 5 columns in the sheet that has been imported.



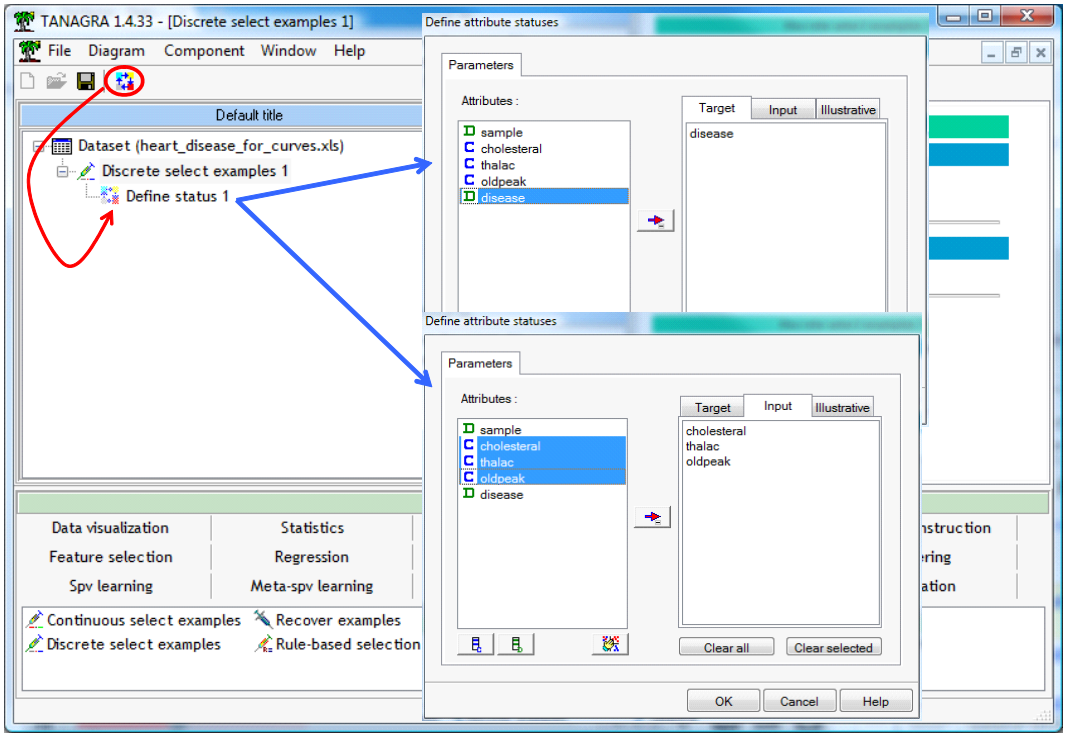
We need to partition the observations into two disjoint subsets using the SAMPLE column: the first serves to the construction of the models, the second will be used to construct the curves.

We add the DISCRETE SELECT EXAMPLES component (INSTANCE SELECTION tab) into the diagram. We click on the PARAMETERS menu to specify the settings. The active individuals, used for learning phase, are SAMPLE = TRAIN.

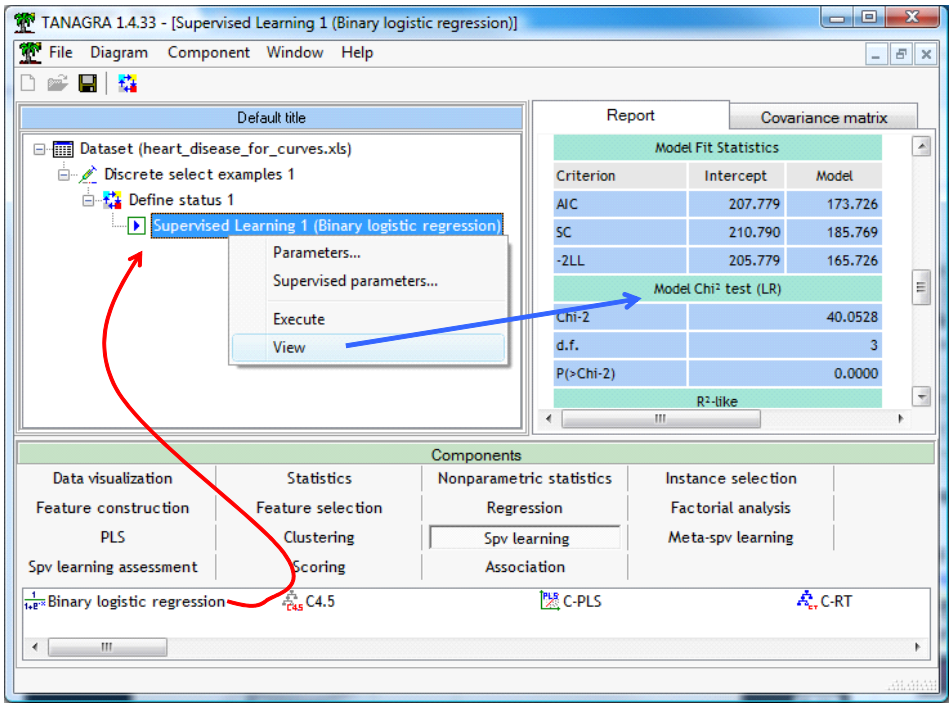


3.2. Model learning and computation of scores

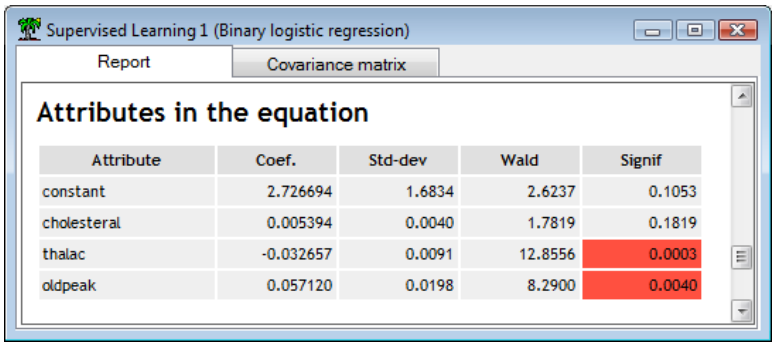
The next step is the construction of the classifier. We must specify the class attribute and the predictive attributes. We add the DEFINE STATUS component into the diagram using the shortcut into the toolbar. We set DISEASE as TARGET; HOLESTERAL, THALAC and OLDPEAK as INPUT. We do not use the SAMPLE column here.



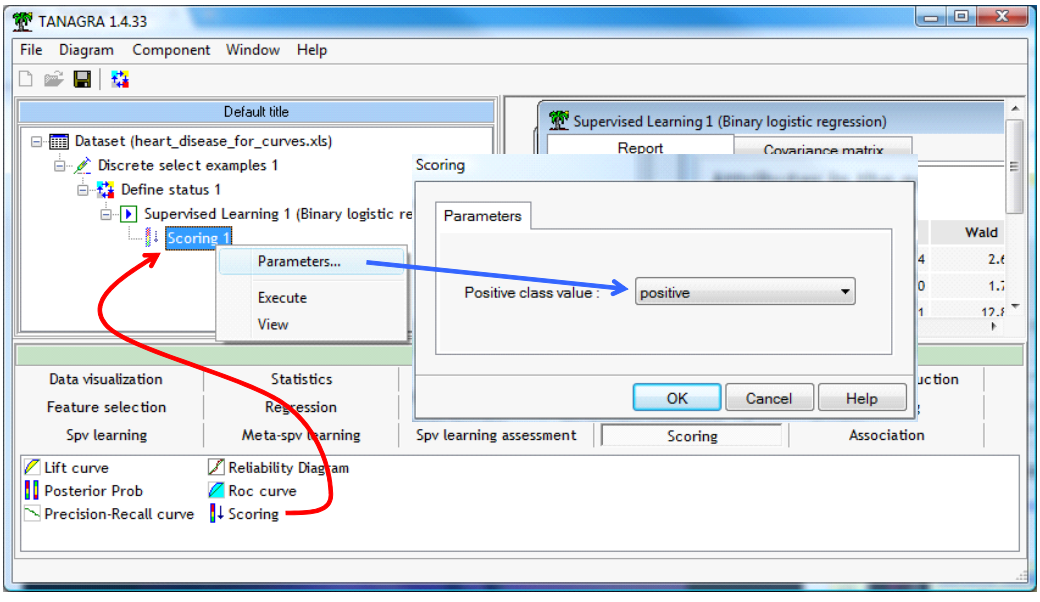
Logistic regression. We insert the BINARY LOGISTIC REGRESSION into the diagram. We click on the VIEW menu to obtain the results.



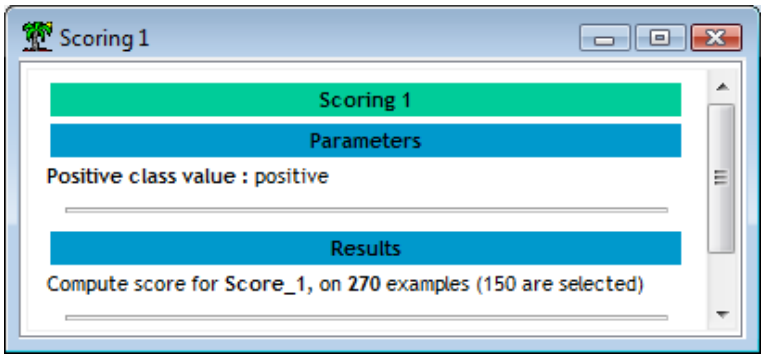
The regression is globally significant (at 5%). The relevant variables are THALAC and OLDPEAK.



We must now compute the score of the individuals using the model. We use the SCORING component, we set the parameter in order to compute the score for DISEASE = POSITIVE.



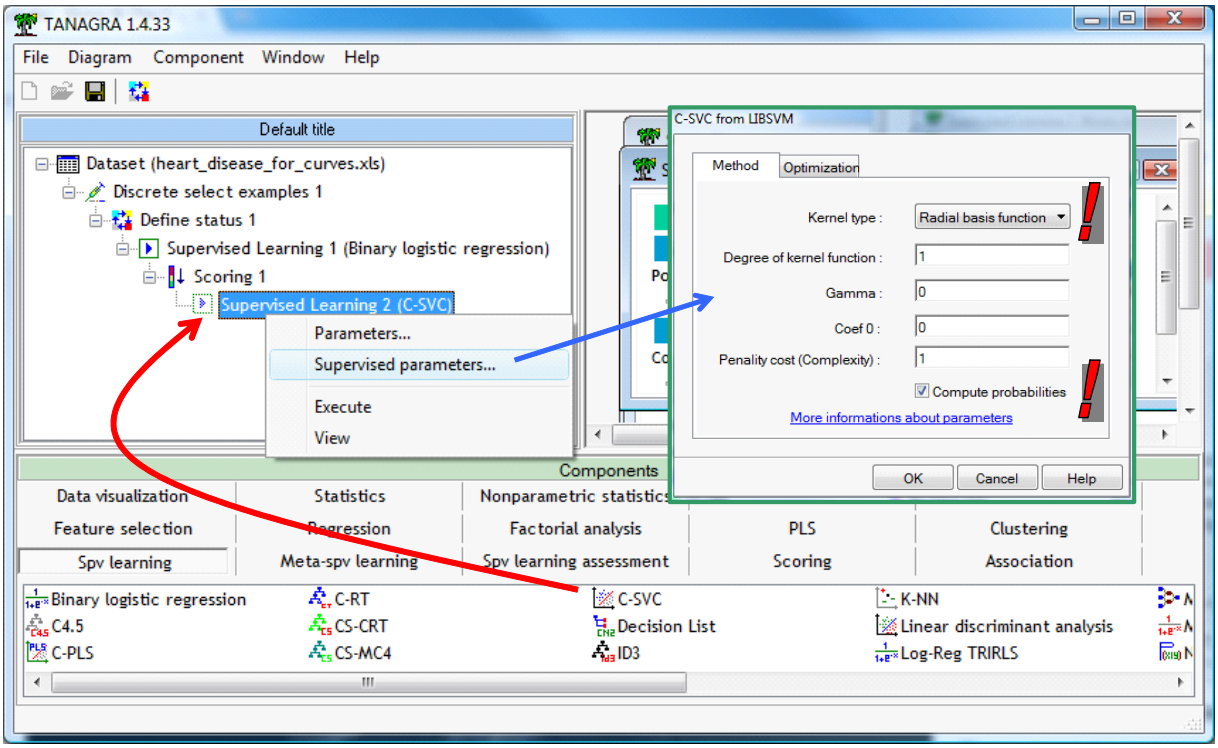
We validate and we click on VIEW. Tanagra calculates the scores for all observations, both for the train and the test samples.



Support vector machine. We repeat the same operations but using a SVM with RBF kernel. We insert the component C-SVC (SPV LEARNING tab) into the diagram.

C-SVC comes from the famous LIBSVM library[7].

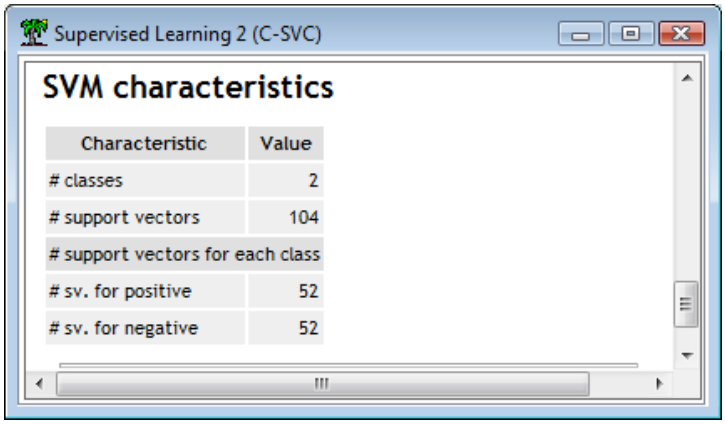
We set the appropriate parameters by clicking on the SUPERVISED PARAMETERS. We ask a RBF kernel (Radial Basis Function). We want to get the tools for calculating scores.



We validate our choices and we click on the VIEW menu.

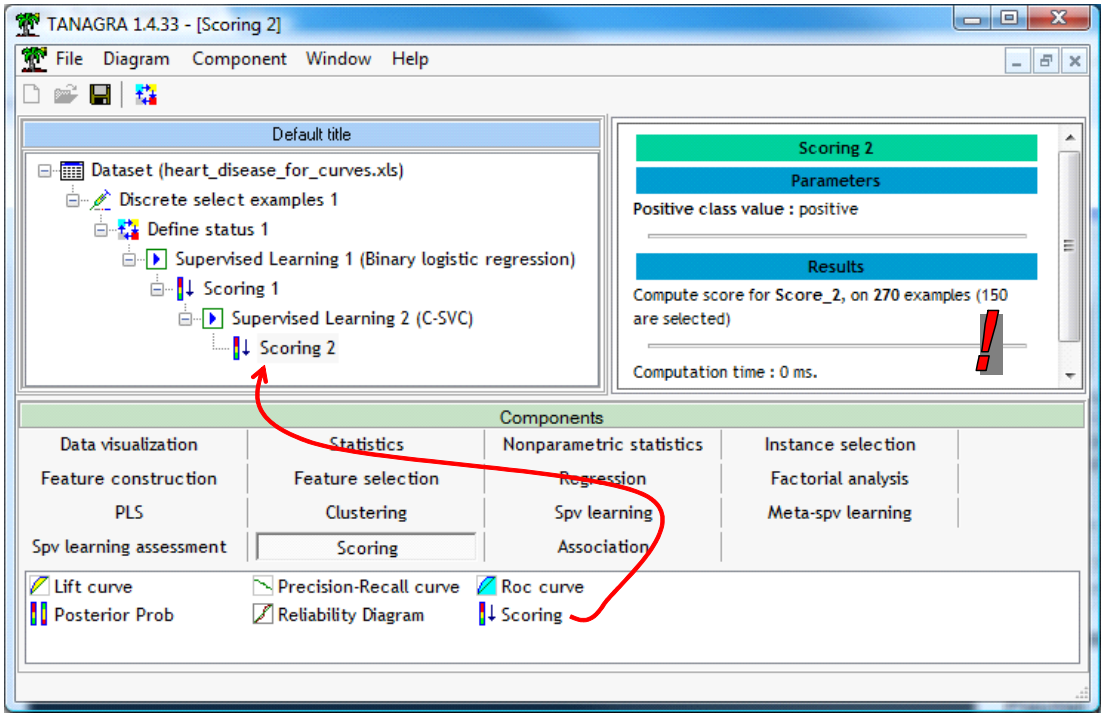
The indications of C-SVC are rather laconic. But it does not matter; the tool enables to assign scores to the individuals. It is the important thing in our context.

[7] http://www.csie.ntu.edu.tw/~cjlin/libsvm/



Let's calculate the scores; we insert the SCORING component into the diagram. We always want to obtain the posterior probabilities of the POSITIVE value of the DISEASE attribute.

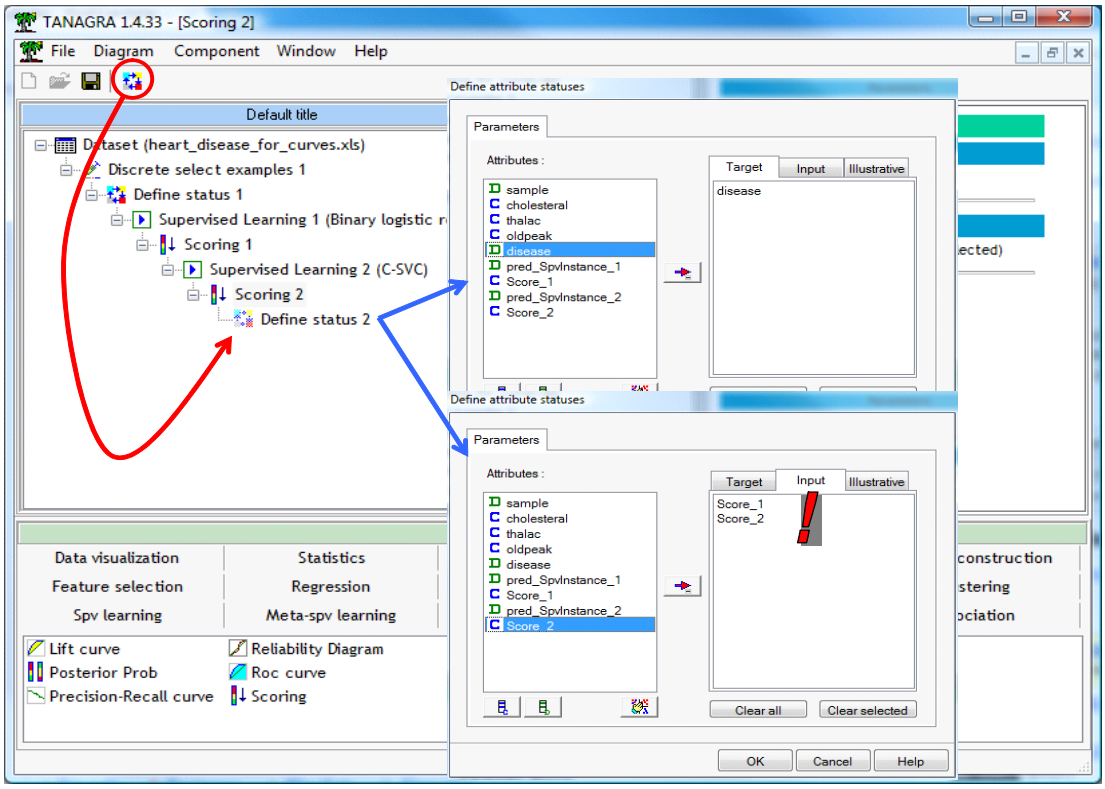
We click on the VIEW menu in order to obtain the results.



3.3. Construction of the ROC curve

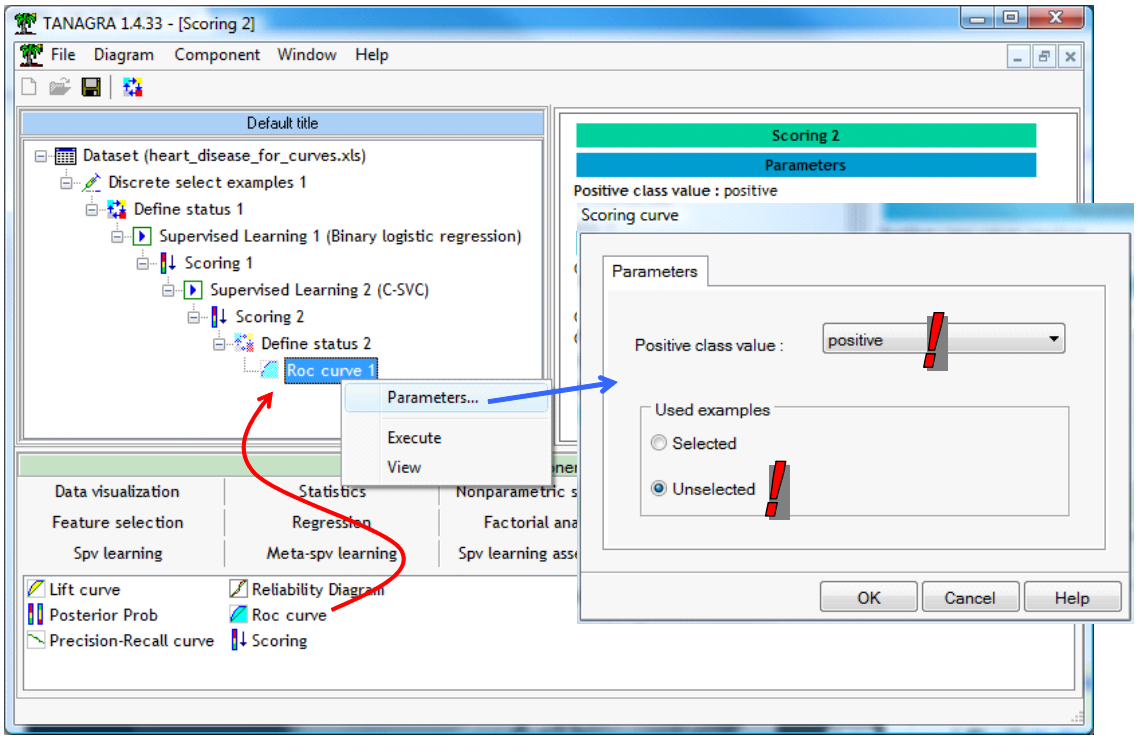
At this step, we can construct the ROC curves on the test sample and compare the performances of the logistic regression and SVM.

Firstly, we must specify the new statuses of the variables. We insert the DEFINE STATUS component. We set DISEASE and TARGET; SCORE_1 (logistic regression scores) and score_2 (SVM scores) as INPUT.

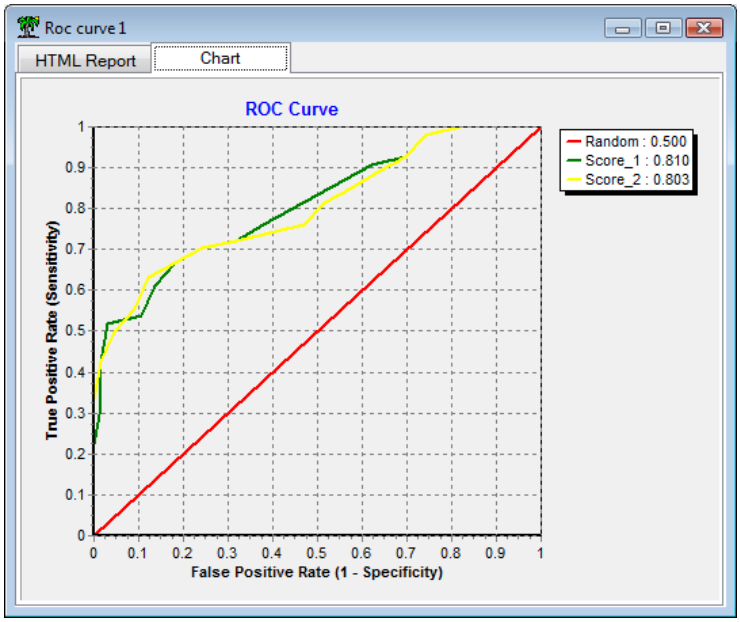


Two important points: (1) We can set as many variables as we want in INPUT. (2) We use ratings provided by supervised learning to construct the curves in this tutorial. But in reality, we can use any quantitative variable which can order the observations according to their degree of belonging to the positive group**.**

We add the ROC CURVE component (SCORING tab) into the diagram. We set the following settings. The curves are computed from the test set.



We click on the VIEW menu, we obtain the ROC curve.

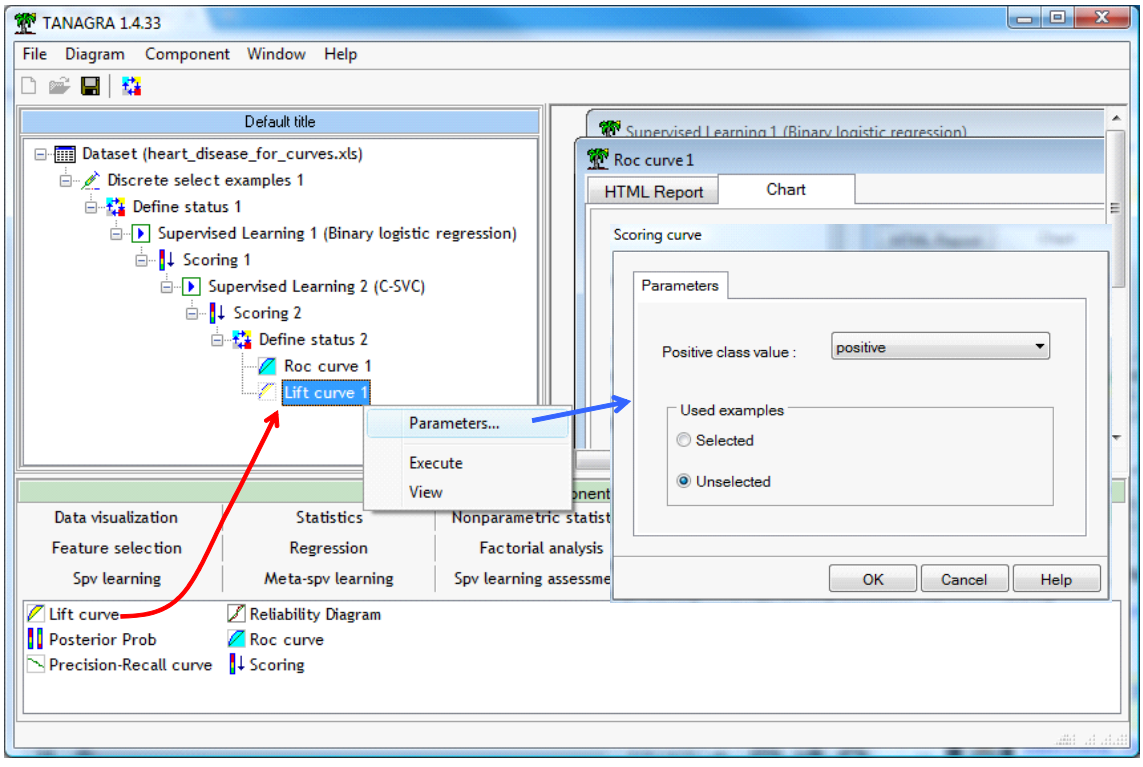


Both methods are equivalent on our dataset. The two curves are very close. Tanagra directly provides the AUC criterion, we have AUC (logistic regression) = 0.810 and AUC (SVM) = 0.803.

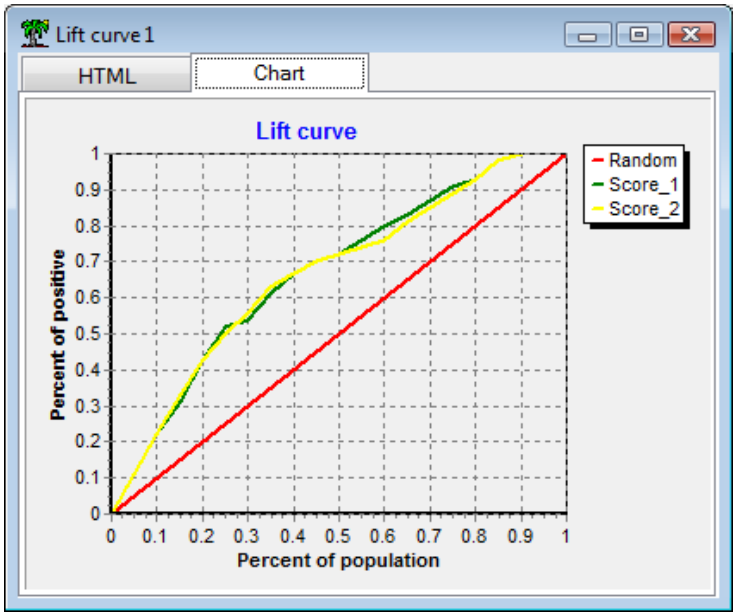
3.4. Construction of the gain chart (cumulative lift curve)

Since all the preparatory operations have been previously made, we can directly insert the LIFT CURVE component (SCORING tab) into the diagram, below the DEFINE STATUS 2 component.

Again, like for the ROC curve, we set the appropriate settings (PARAMETERS menu).



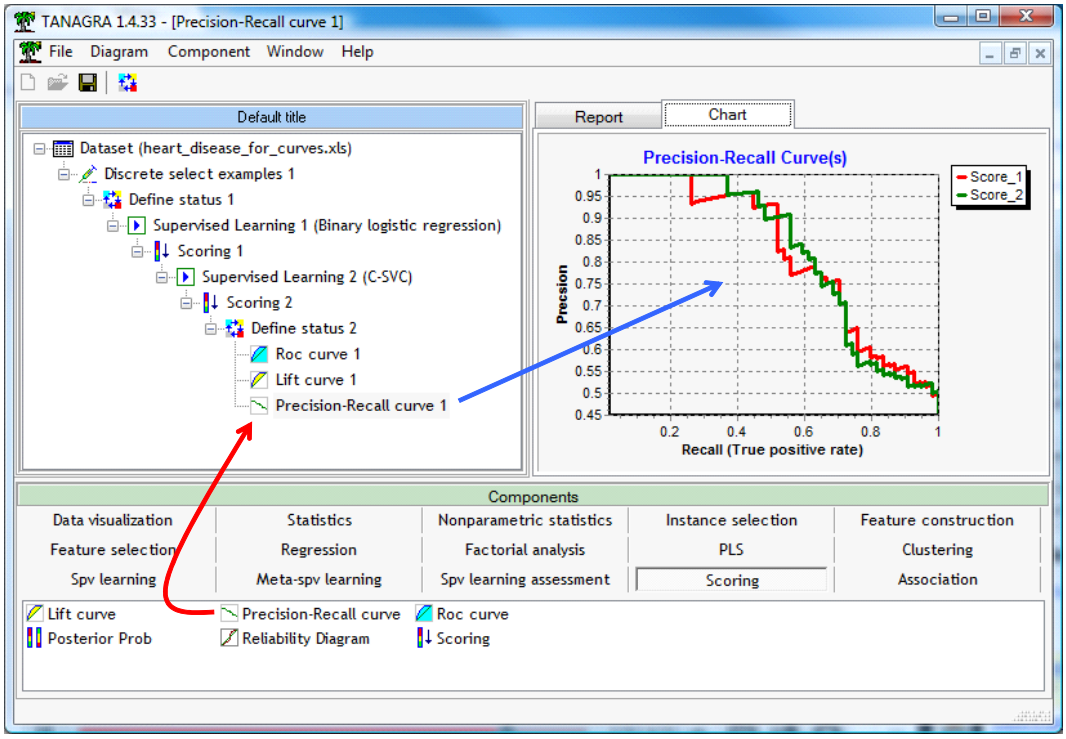
We click on the VIEW menu.



The conclusions are the same as the ROC curve. On our dataset, the two methods are similar in their ability to assign high scores to instances belonging to the positive group.

3.5. Construction of the precision-recall curve

Last, we insert the PRECISION RECALL CURVE (SCORING tab) component into the diagram. We set the same settings (PARAMETERS menu).



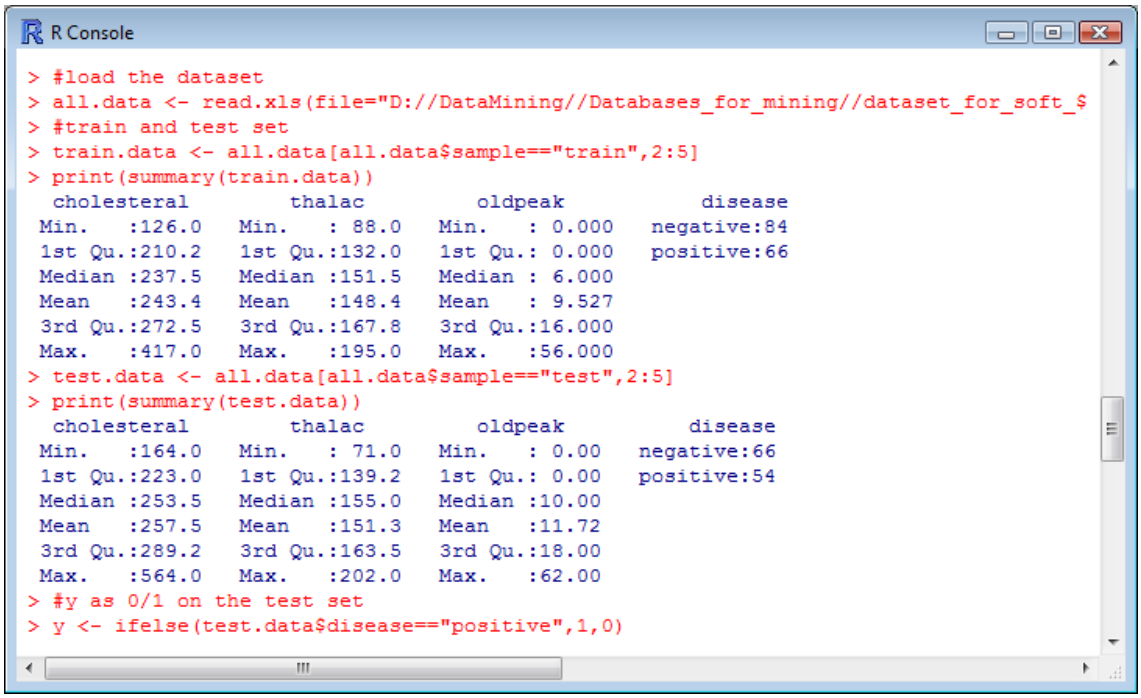
The overall appearance of the curve is very different compared with the previous curves. But, the conclusions remain the same: the performances of the two models are similar on our dataset.

4. Construction of curves under R

We show mainly the source code and the main results in this section.

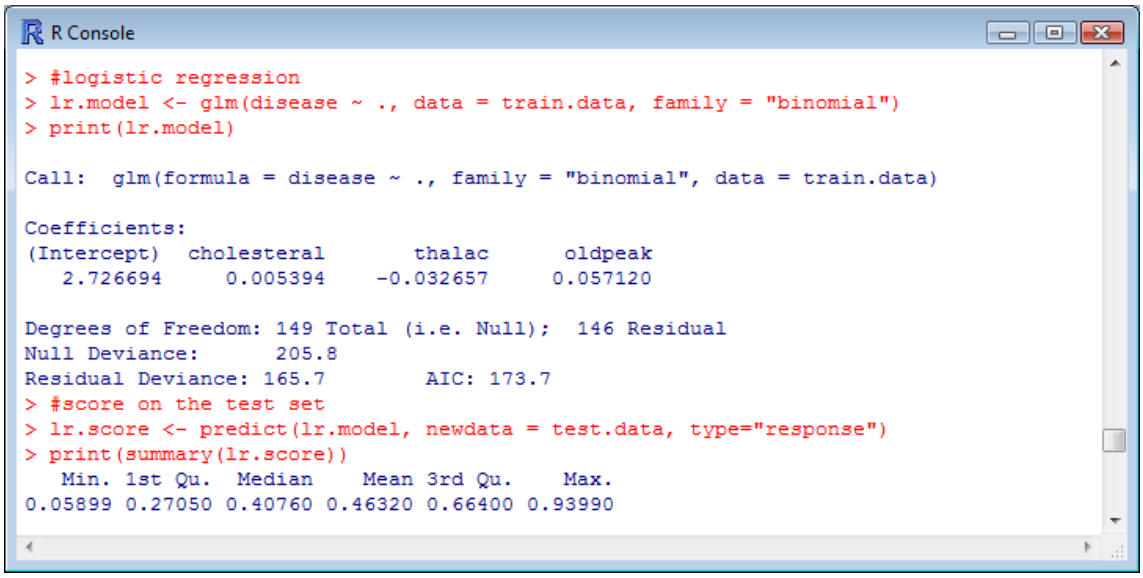
4.1. Importing and preparing the samples

We use the **xlsReadWrite** package for the data file importation. The last command is intended to recode the class attribute (DISEASE) into 1/0 values.

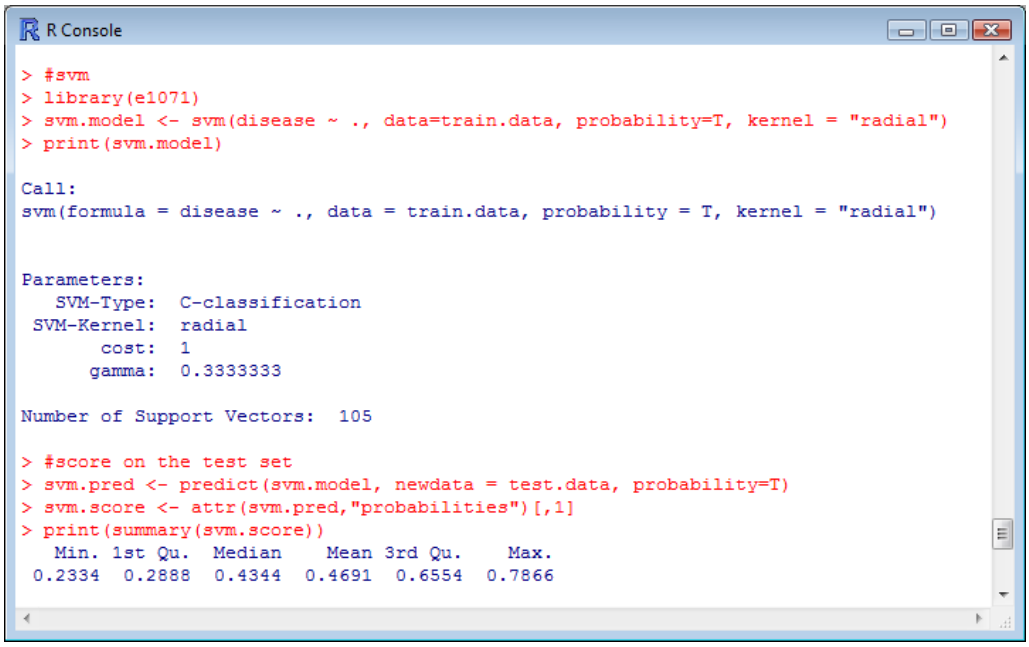


4.2. Learning of the classifiers and assigning scores

Logistic Regression. **glm()** is used for the learning phase; **predict()** enables to obtain the scores on the test sample.



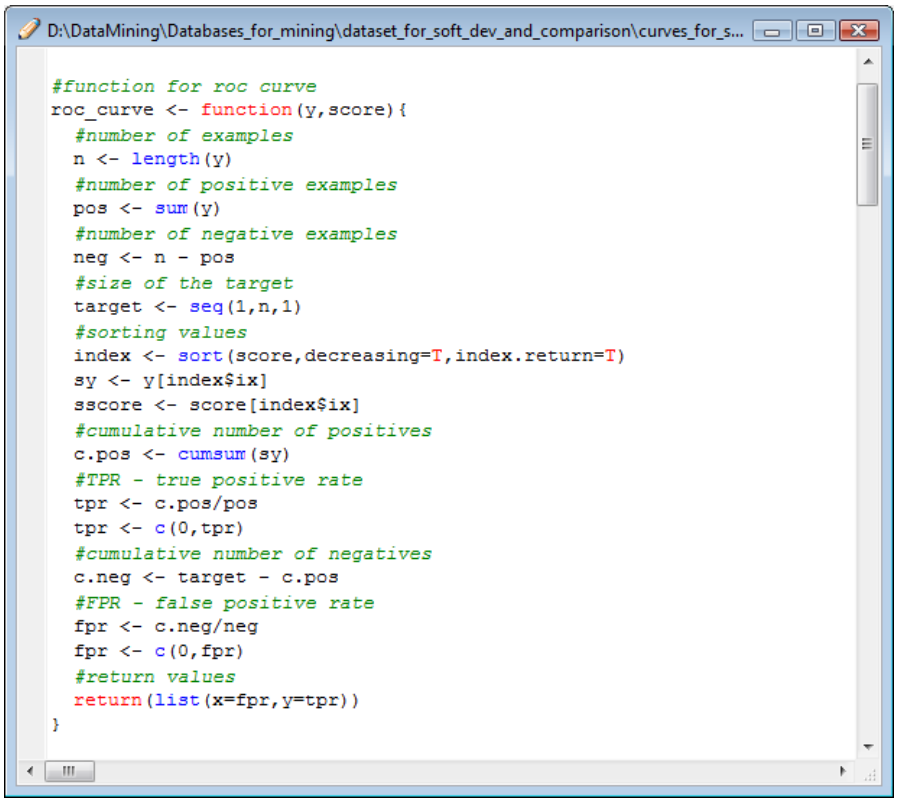
Support vector machine. About the SVM, we use the **svm()** command from the **e1071** package. The number of parameters is impressive; we try to set the same settings than for Tanagra.



4.3. Construction of the ROC curve

The curves are computed in two steps. First we create a function that, starting from the class value (1/0) and the score, computes the coordinates of the curve. Second, we call this function for the two classifiers and we plot the curves into the same chart.

The function[8] which calculates the coordinates is



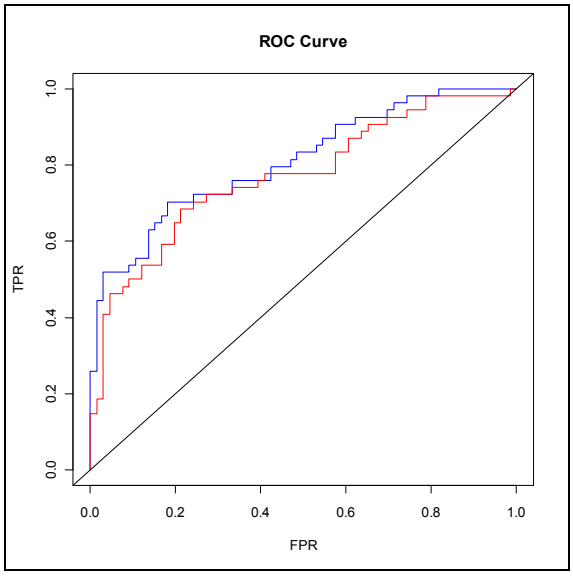
[8] Our source code is very simple and even naive; it is possible to write more efficiently the function.

We create the curves with the following commands.

D:\DataMining\Databases_for_mining\dataset_for_soft_dev_and_comparison\curves_for_spv_learning\heart_di...

```
#ROC curve
lr.roc <- roc_curve(y,lr.score)
svm.roc <- roc_curve(y,svm.score)
plot(lr.roc$x,lr.roc$y,type="l",col="blue",main="ROC Curve",xlab="FPR",ylab="TPR")
lines(svm.roc$x,svm.roc$y,type="l",col="red")
abline(0,1)
```

We obtain the curves, the logistic regression is in blue, and the SVM is in red.



4.4. Construction of the gain chart

We use the same framework. The function is the following:

D:\DataMining\Databases_for_mining\dataset_for_soft_dev_and_comparison\curves_for_spv_learn...

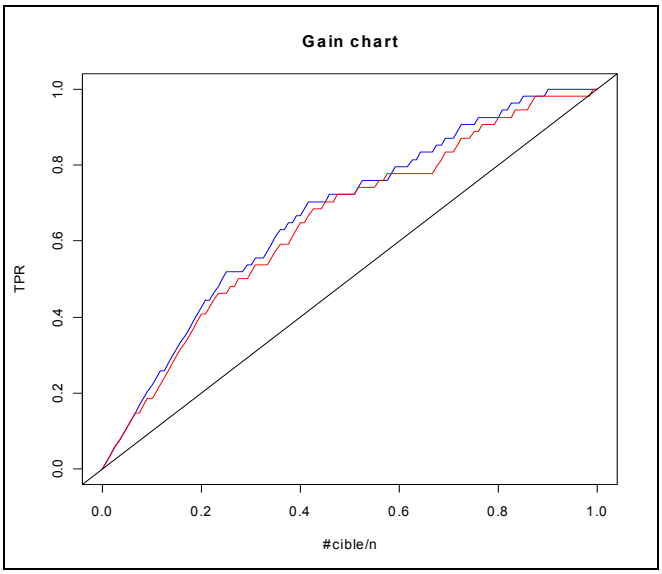
```
#function for cumulative lift curve
cum_lift_curve <- function(y,score){
  #number of examples
  n <- length(y)
  #number of positive examples
  pos <- sum(y)
  #size of the target
  target <- seq(1,n,1)
  #sorting values
  index <- sort(score,decreasing=T,index.return=T)
  sy <- y[index$ix]
  sscore <- score[index$ix]
  #cumulative number of positives
  c.pos <- cumsum(sy)
  #TPR - true positive rate
  tpr <- c.pos/pos
  tpr <- c(0,tpr)
  #relative size of the target
  rel.target <- target/n
  rel.target <- c(0,rel.target)
  #return values
  return(list(x=rel.target,y=tpr))
}
```

In order to obtain the curves, we set

D:\DataMining\Databases_for_mining\dataset_for_soft_dev_and_comparison\curves_for_spv_learning\heart_disease_for_cu...

```
#Cumulative lift curve
lr.lift <- cum_lift_curve(y,lr.score)
svm.lift <- cum_lift_curve(y,svm.score)
plot(lr.lift$x,lr.lift$y,type="l",col="blue",main="Gain chart",xlab="#cible/n",ylab="TPR")
lines(svm.lift$x,svm.lift$y,type="l",col="red")
abline(0,1)
```

We then obtain the chart.



4.5. Construction of the precision recall curve

The function is written as follows.

D:\DataMining\Databases_for_mining\dataset_for_soft_dev_and_comparison\curves_for_sp...

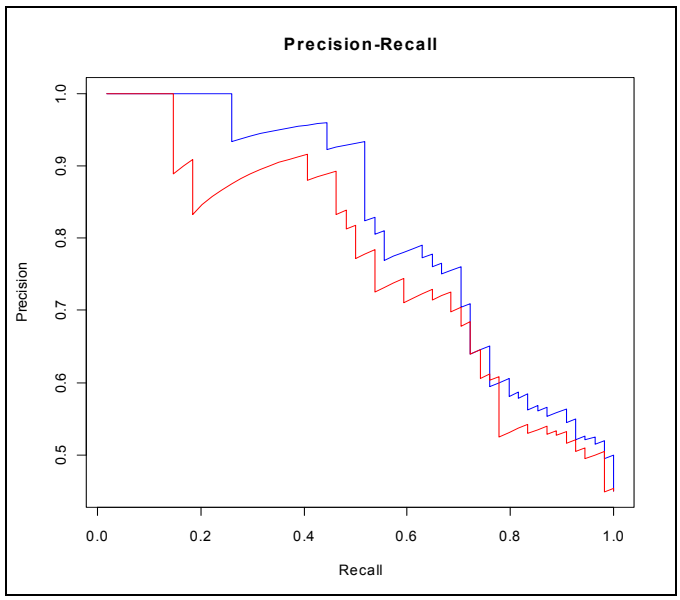
```
#function for recall-precision curve
recall_prec <- function(y,score){
  #number of examples
  n <- length(y)
  #number of positive examples
  pos <- sum(y)
  #size of the target
  target <- seq(1,n,1)
  #sorting values
  index <- sort(score,decreasing=T,index.return=T)
  sy <- y[index$ix]
  sscore <- score[index$ix]
  #cumulative number of positives
  c.pos <- cumsum(sy)
  #TPR - true positive rate
  tpr <- c.pos/pos
  #precision
  prec <- c.pos/target
  #return values
  return(list(x=tpr,y=prec))
}
```

Then...

D:\DataMining\Databases_for_mining\dataset_for_soft_dev_and_comparison\curves_for_spv_learning\heart_disease_for_curves.r*

```
#Recall precision curve
lr.rp <- recall_prec(y,lr.score)
svm.rp <- recall_prec(y,svm.score)
plot(lr.rp$x,lr.rp$y,type="l",col="blue",main="Precision-Recall",xlab="Recall",ylab="Precision")
lines(svm.rp$x,svm.rp$y,type="l",col="red")
```

We obtain



5. Conclusion

In this tutorial, we showed how to calculate a few curves to evaluate the performance of classifiers in a supervised learning framework, "at the hand" first by describing the details of the operations into Excel; then using tools such as Tanagra 1.4.33 and R 2.9.2.

These curves are more sophisticated than the simple (simplistic) error rate associated with a unique version of the confusion matrix. By varying the discrimination threshold, we can define a set of confusion matrices and thus give a broader evaluation of the behavior of the classifiers.