Subject

One of the main advantages of decision trees is the possibility, for the users, to interactively build the prediction model. In this tutorial, we show how, using **SIPINA**[1] and **ORANGE**, we build and manually modify a decision tree; especially we select the split attribute and pruning the tree.

SIPINA is one of my old projects. It was very useful but it had some limitations, which have been rectified in TANAGRA: it was intended only for supervised learning; we cannot define and save the sequences of treatments in a diagram. Nevertheless, I use still this version for my courses, in particular for its functionalities in the interactive construction of decision trees.

SIPINA uses a graphical representation of the tree; ORANGE uses a standard treeview components. We will see that they propose very similar functionalities and provide the same results.

Dataset

We use the IRIS_TREE.TXT dataset (UCI IRVINE). There are 150 examples, 4 descriptors; the class attribute has 3 values.

We use 75 examples for the learning set, 75 examples for the test set. A new column specifies the membership of each example.

Decision Tree exploration with SIPINA

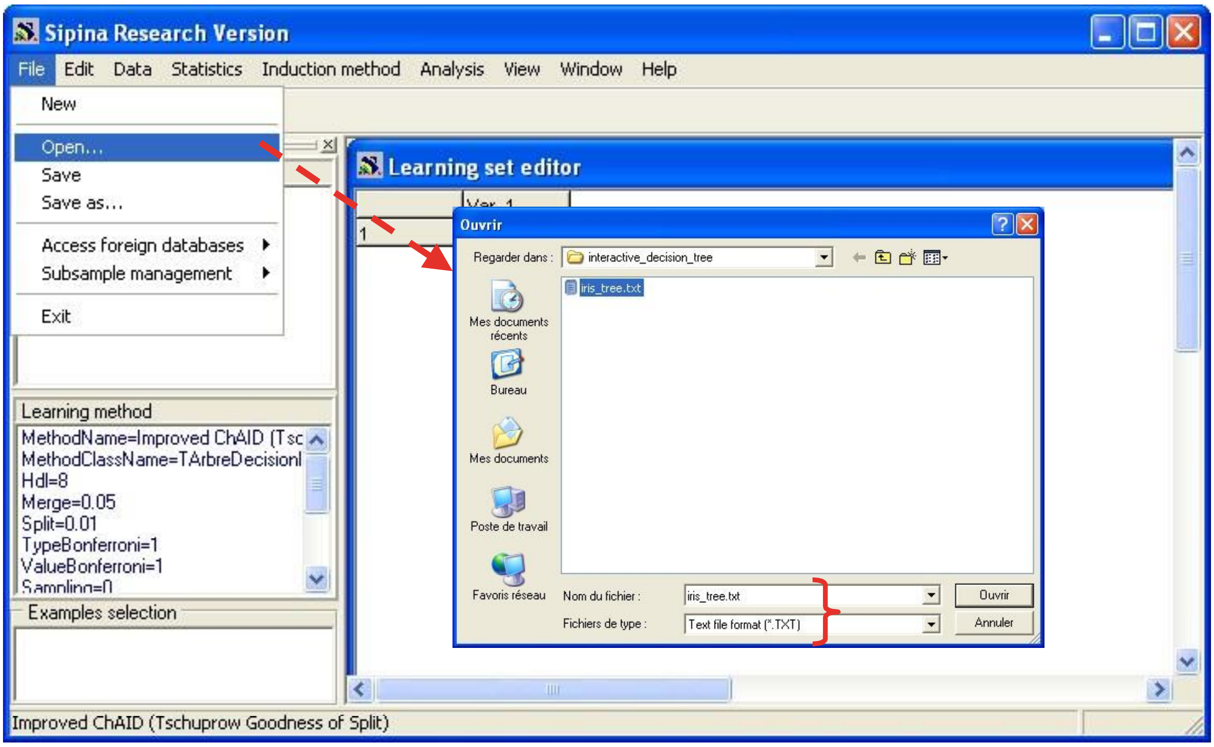
It is not necessary to define a diagram with SIPINA. We must simply click on several menus by respecting an order that we point out here.

SIPINA

When we start SIPINA, we obtain the following main window with several panels: the data grid, the selected attributes, the selected learning method, the learning and test set subdivision.

Data importation

SIPINA can handle various file formats. We use the text file format in this tutorial[2] (TAB separator). We click on the FILE / OPEN menu, a dialog box appears; we choose the right file format (TXT).



Learning method

```
MethodName=Improved ChAID (Tsc
MethodClassName=TArbreDecisionI
Hdl=8
Merge=0.05
Split=0.01
TypeBonferroni=1
ValueBonferroni=1
Sampling=0
```

Regarder dans : interactive_decision_tree

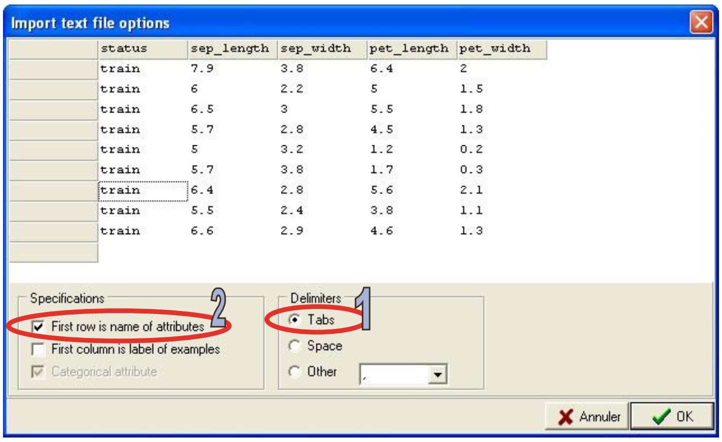
iris_tree.txt

Nom du fichier : iris_tree.txt

Fichiers de type : Text file format (*.TXT)

Improved ChAID (Tschuprow Goodness of Split)

In the next box, we must select the following information: (1) column separator is TAB; (2) the first row of the dataset contains the name of attributes.



status	sep_length	sep_width	pet_length	pet_width
train	7.9	3.8	6.4	2
train	6	2.2	5	1.5
train	6.5	3	5.5	1.8
train	5.7	2.8	4.5	1.3
train	5	3.2	1.2	0.2
train	5.7	3.8	1.7	0.3
train	6.4	2.8	5.6	2.1
train	5.5	2.4	3.8	1.1
train	6.6	2.9	4.6	1.3

Specifications

☑ First row is name of attributes
☐ First column is label of examples
☑ Categorical attribute

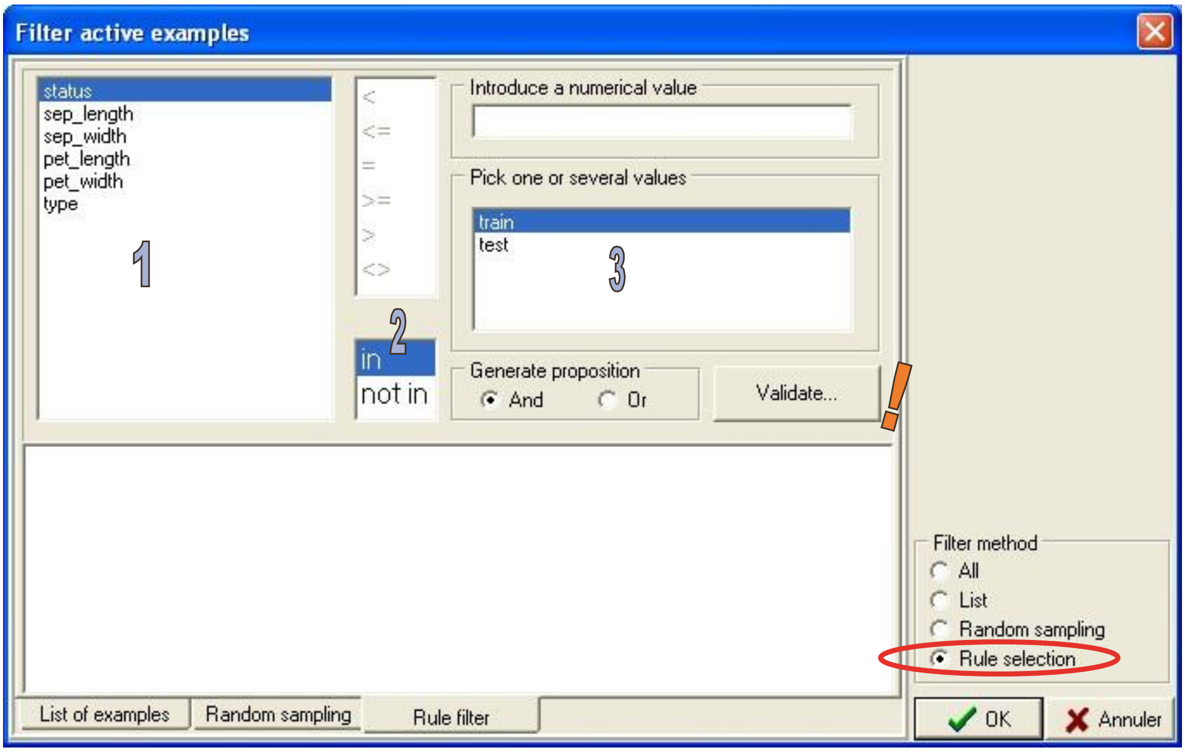
Delimiters

● Tabs
○ Space
○ Other ,

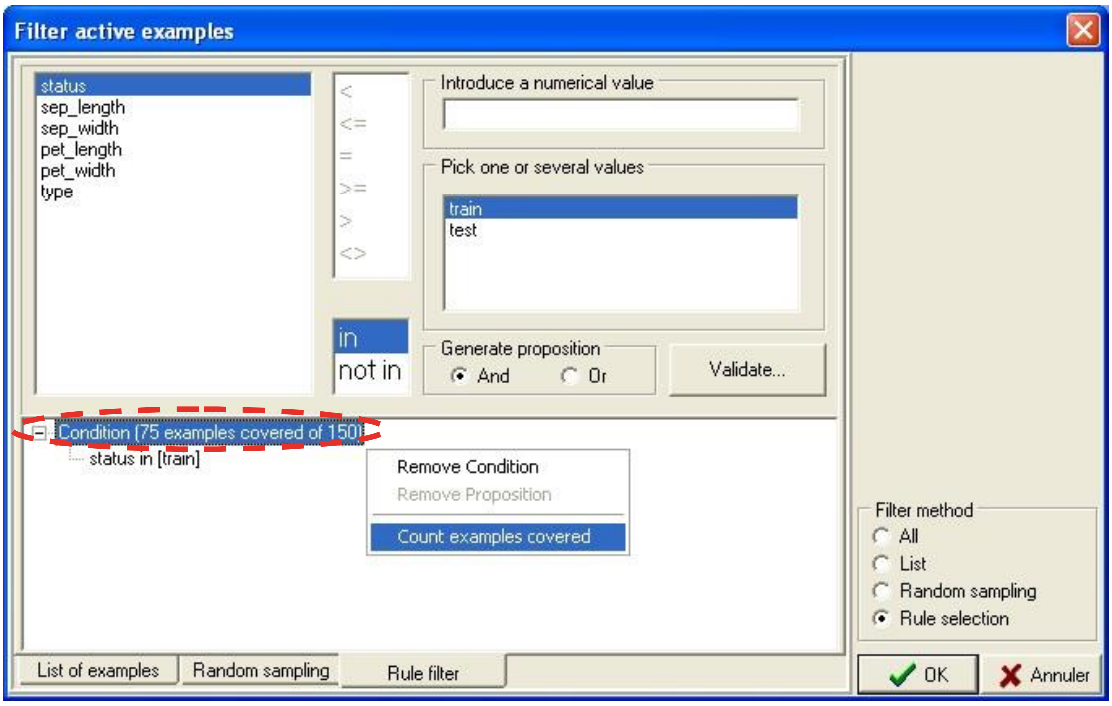
The dataset is downloaded and showed in the main data grid. The type of each attribute is defined by the first value of each column (the second row in the file). If this value is numeric, the type of the attribute will be continuous; if it is alphabetic, the type will be discrete.

Selecting the train and the test set

In order to define the learning set, we click on the ANALYSIS / SELECT ACTIVE EXAMPLES menu, we select the RULE SELECTION option.



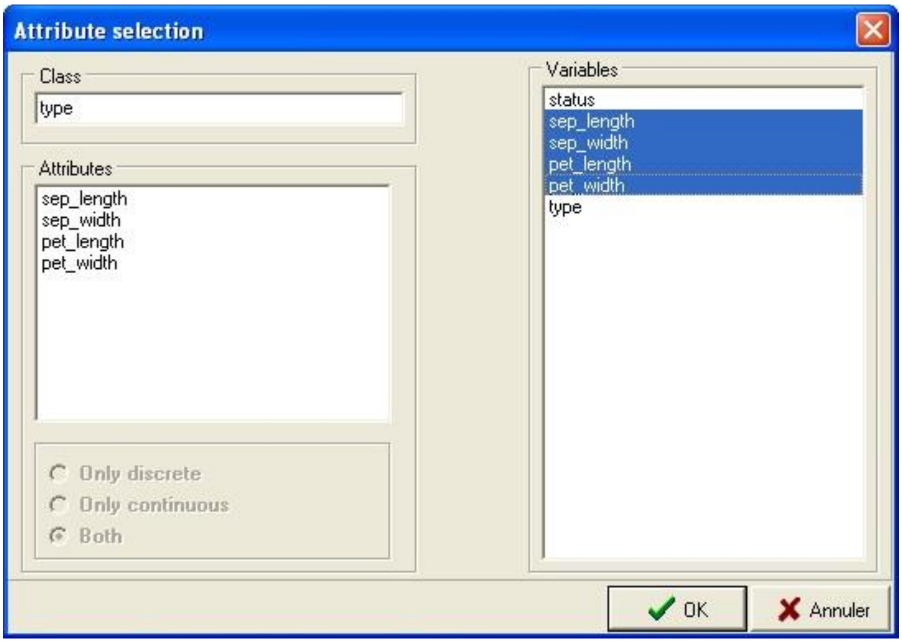
We have to build the rule that allows defining the learning set: (1) we select the STATUS attribute; (2) we select the rule proposition; (3) we select the value of STATUS. We confirm with the VALIDATE button (!). We can count the covered examples (COUNT EXAMPLES COVERED contextual menu), we have select 75 observations, and the remaining belongs to the test set.



Condition (75 examples covered of 150)
status in [train]

Selecting target and input attributes

We click on ANALYSIS / DEFINE CLASS ATTRIBUTE. We use "drag an drop" in order to select the TARGET (class attribute) and the INPUT attributes. Of course, we do not use the STATUS attribute.



Class: type

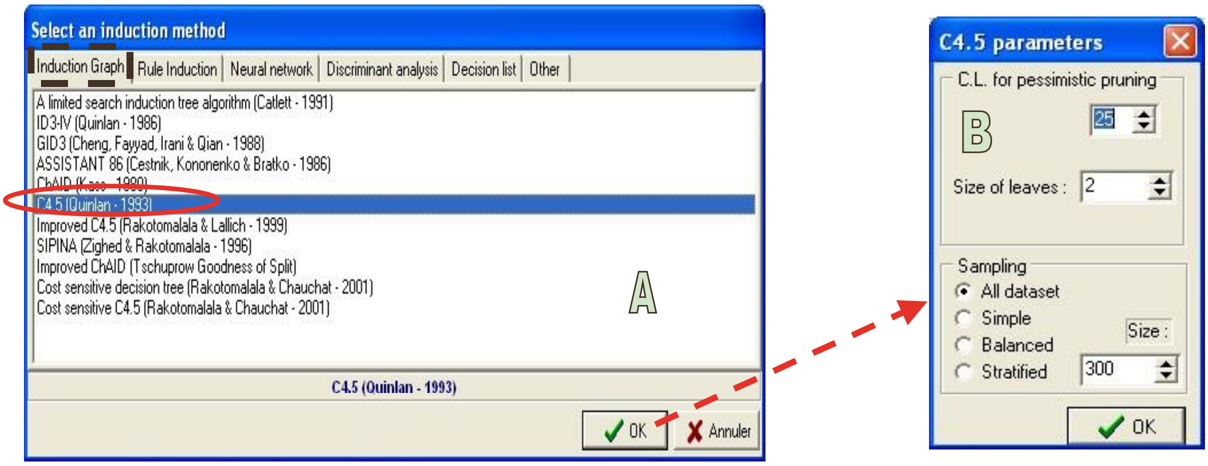
Attributes:
sep_length
sep_width
pet_length
pet_width

○ Only discrete
○ Only continuous
● Both

Variables:
status
sep_length
sep_width
pet_length
pet_width
type

Selecting the learning method

We click on the INDUCTION METHOD / STANDARD ALGORITHM menu. A dialog box appears. We set the C4.5 algorithm [A – Induction Graph tab]; we can modify the default parameters of the method [B].



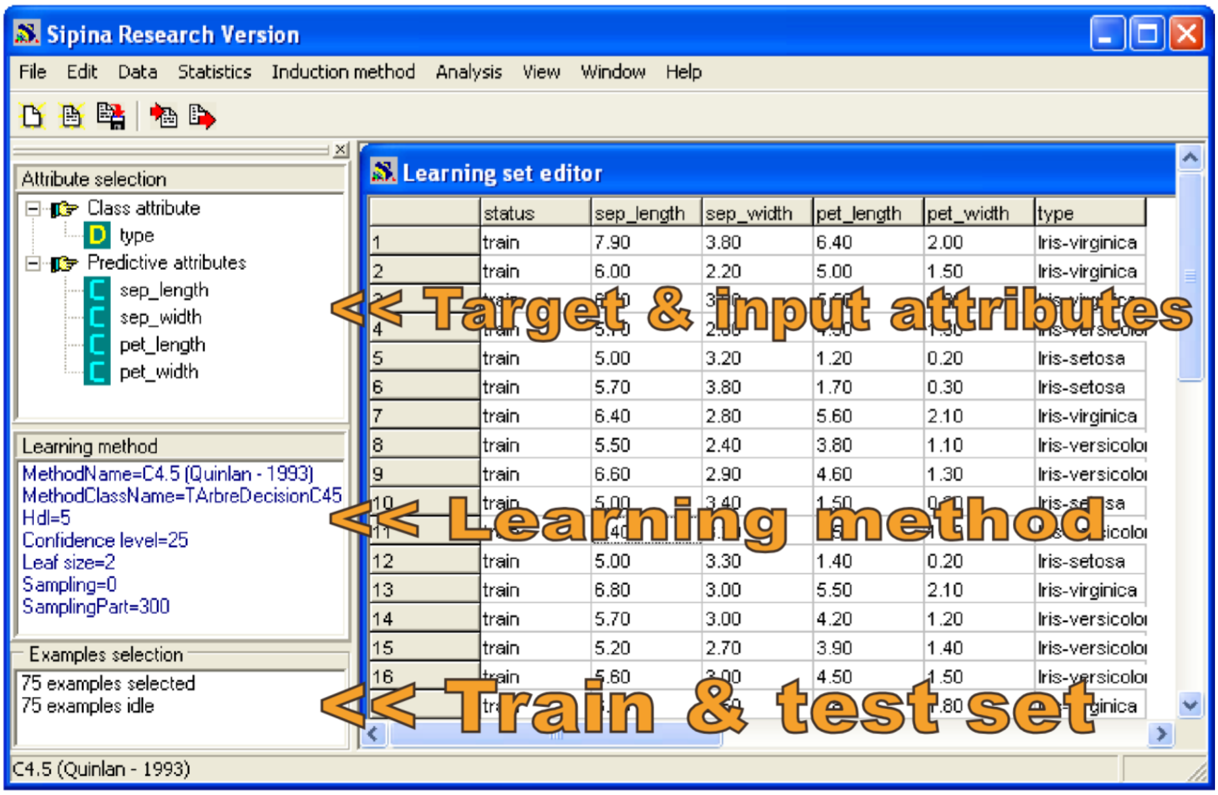
A limited search induction tree algorithm (Catlett - 1991)
ID3-IV (Quinlan - 1986)
GID3 (Cheng, Fayyad, Irani & Qian - 1988)
ASSISTANT 86 (Cestnik, Kononenko & Bratko - 1986)
ChAID (Kass - 1980)
C4.5 (Quinlan - 1993)
Improved C4.5 (Rakotomalala & Lallich - 1999)
SIPINA (Zighed & Rakotomalala - 1996)
Improved ChAID (Tschuprow Goodness of Split)
Cost sensitive decision tree (Rakotomalala & Chauchat - 2001)
Cost sensitive C4.5 (Rakotomalala & Chauchat - 2001)

C4.5 (Quinlan - 1993)

C.L. for pessimistic pruning: 25
Size of leaves: 2

Sampling
● All dataset
○ Simple
○ Balanced
○ Stratified
Size: 300

We obtain on the left of the main window a summary of our parameters.


Sipina Research Version
File Edit Data Statistics Induction method Analysis View Window Help
Learning set editor

Attribute selection
- Class attribute
  - D type
- Predictive attributes
  - C sep_length
  - C sep_width
  - C pet_length
  - C pet_width

Learning method

MethodName=C4.5 (Quinlan - 1993)
MethodClassName=TArbreDecisionC45
Hdl=5
Confidence level=25
Leaf size=2
Sampling=0
SamplingPart=300

Examples selection

75 examples selected
75 examples idle

	status	sep_length	sep_width	pet_length	pet_width	type
1	train	7.90	3.80	6.40	2.00	Iris-virginica
2	train	6.00	2.20	5.00	1.50	Iris-virginica
3	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
4	train	[illegible]	[illegible]	[illegible]	[illegible]	Iris-versicolor
5	train	5.00	3.20	1.20	0.20	Iris-setosa
6	train	5.70	3.80	1.70	0.30	Iris-setosa
7	train	6.40	2.80	5.60	2.10	Iris-virginica
8	train	5.50	2.40	3.80	1.10	Iris-versicolor
9	train	6.60	2.90	4.60	1.30	Iris-versicolor
10	train	5.00	3.40	1.50	[illegible]	Iris-setosa
11	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	Iris-versicolor
12	train	5.00	3.30	1.40	0.20	Iris-setosa
13	train	6.80	3.00	5.50	2.10	Iris-virginica
14	train	5.70	3.00	4.20	1.20	Iris-versicolor
15	train	5.20	2.70	3.90	1.40	Iris-versicolor
16	train	5.60	3.00	4.50	1.50	Iris-versicolor
17	[illegible]	[illegible]	[illegible]	[illegible]	1.80	Iris-virginica

<< Target & input attributes

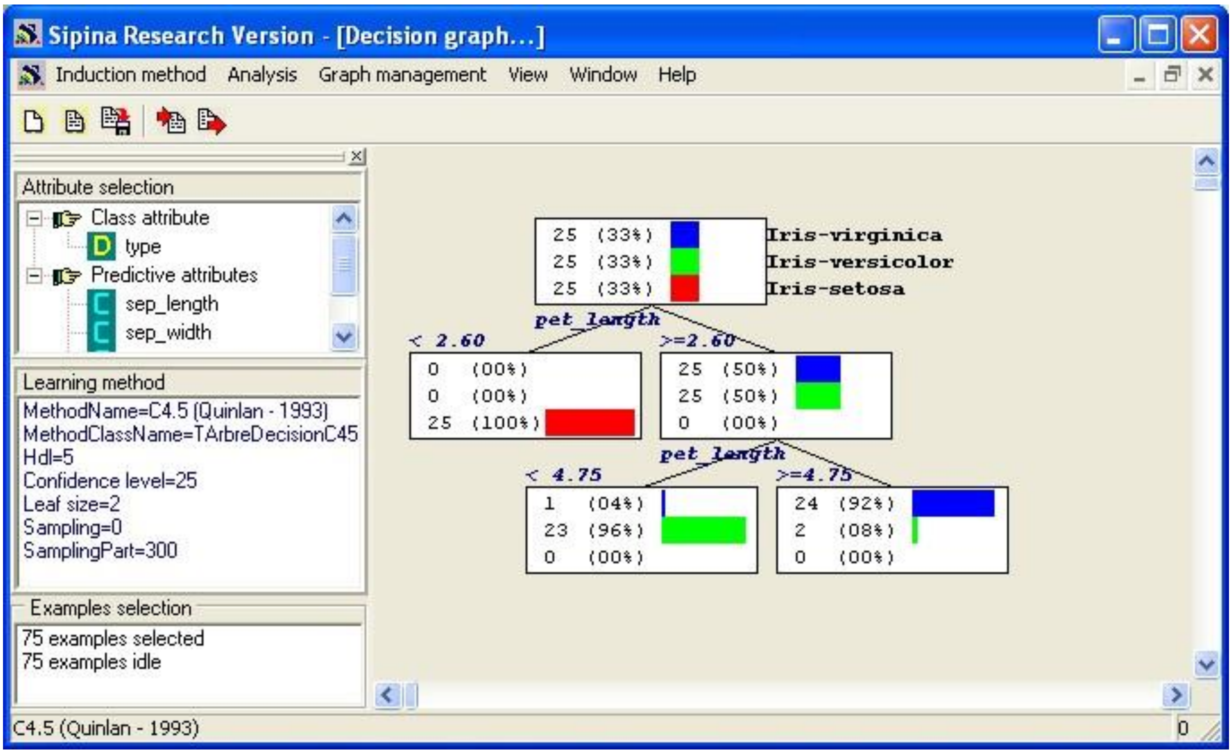
<< Learning method

<< Train & test set

C4.5 (Quinlan - 1993)

Automatic analysis

In order to build the decision tree with the C4.5 algorithm, we click on the ANALYSIS / LEARNING menu.



Attribute selection
- Class attribute
  - D type
- Predictive attributes
  - C sep_length
  - C sep_width

Learning method

MethodName=C4.5 (Quinlan - 1993)
MethodClassName=TArbreDecisionC45
Hdl=5
Confidence level=25
Leaf size=2
Sampling=0
SamplingPart=300

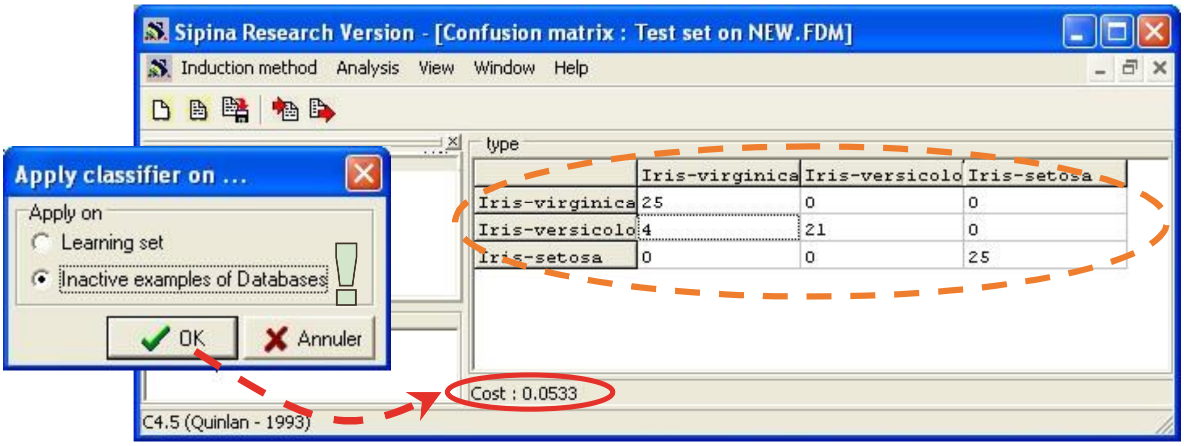
Examples selection

75 examples selected
75 examples idle

C4.5 (Quinlan - 1993)

The tree is very simple, 3 rules enable to determine the type of IRIS from its characteristics, it seems that only variable PET_LENGTH is really relevant.

To evaluate the accuracy of this tree on the test set, we click on the ANALYSIS / TEST menu and select the INACTIVE EXAMPLES OF DATABASE option. The confusion matrix is computed; the test error rate is 5.33%.



Apply on
○ Learning set
● Inactive examples of Databases

type

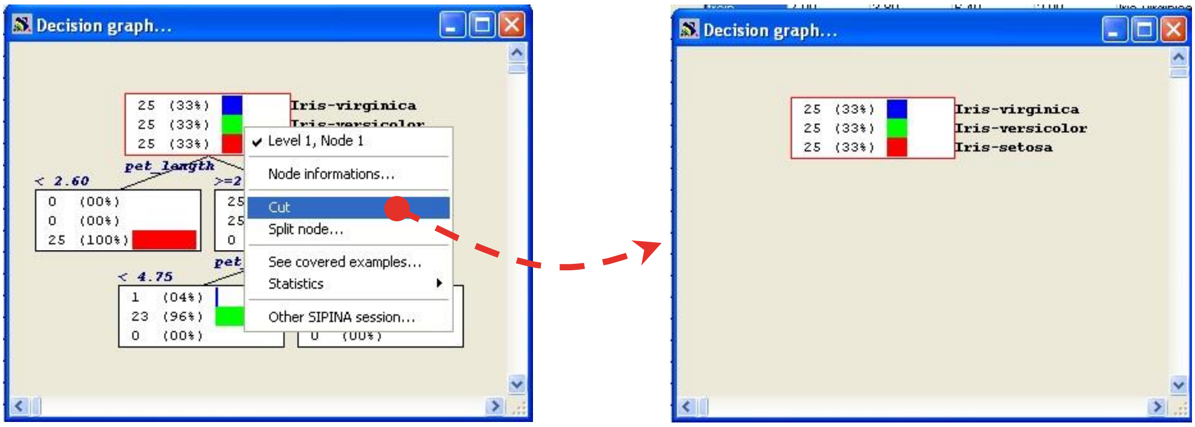
	Iris-virginica	Iris-versicolo	Iris-setosa
Iris-virginica	25	0	0
Iris-versicolo	4	21	0
Iris-setosa	0	0	25

Cost : 0.0533

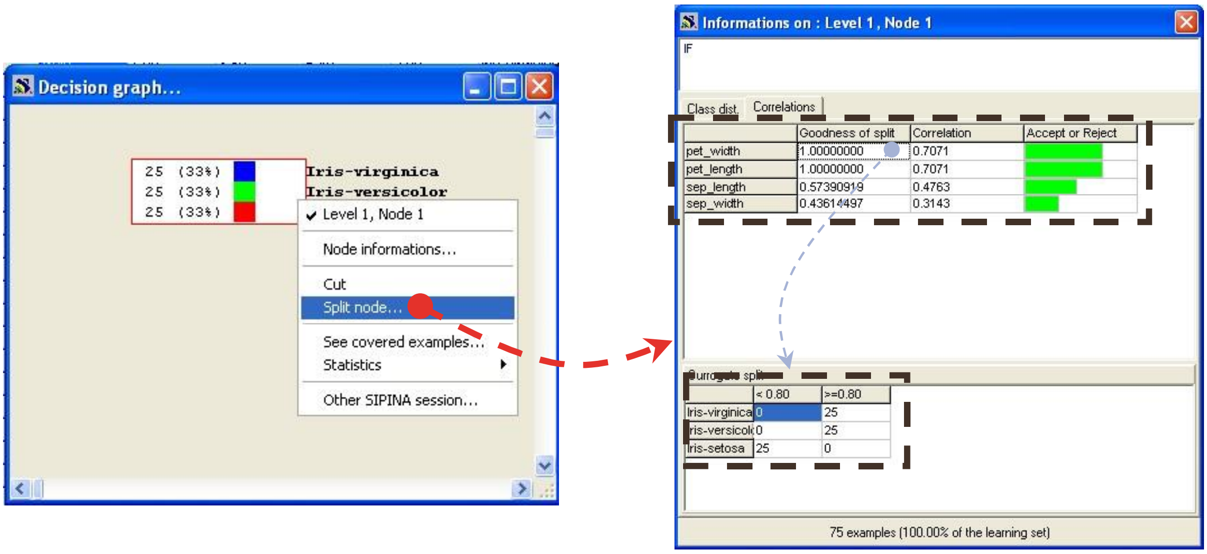
Interactive analysis

Only the PET_LENGTH seems relevant. One can wonder what would be the relevance of the other variables that could take part in the construction of the tree. We want to interactively build the tree and studying the relevance of each candidate attributes when we want to split a node.

We display the tree by clicking the WINDOW / DECISION GRAPH menu. Initially, we must prune the tree starting from its root. The operation is simple: we select the first node; with the contextual menu we click on CUT.



In the next step, we display the segmentations candidates by clicking on SPLIT NODE in the contextual menu. A new window appears with all possible splits on the selected node.



25 (33%)	Iris-virginica
25 (33%)	Iris-versicolor
25 (33%)	

	Goodness of split	Correlation	Accept or Reject
pet_width	1.00000000	0.7071	🟩
pet_length	1.00000000	0.7071	🟩
sep_length	0.57390919	0.4763	🟩
sep_width	0.43614497	0.3143	🟩

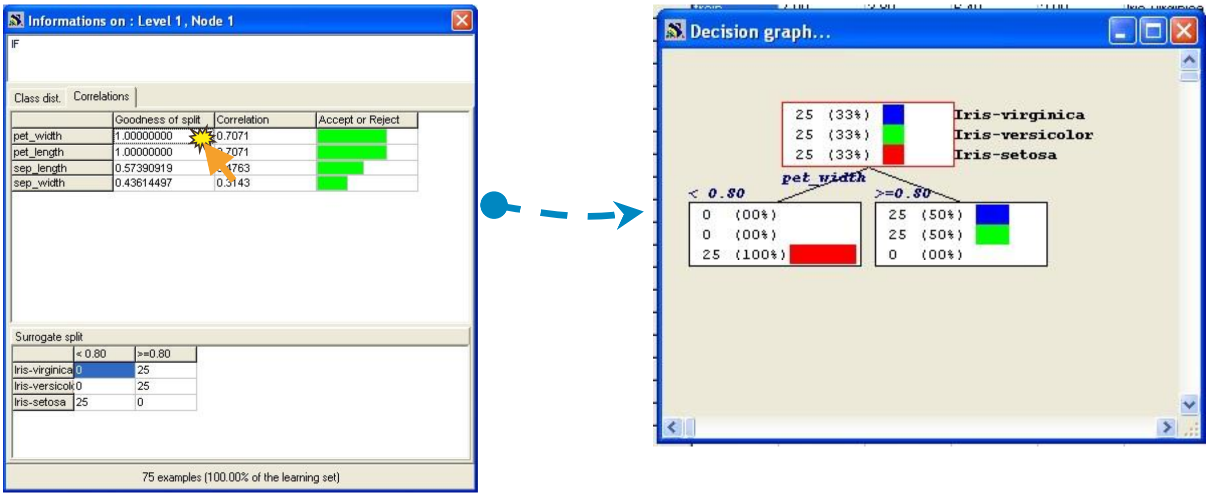
	< 0.80	>=0.80
Iris-virginica	0	25
Iris-versicolor	0	25
Iris-setosa	25	0

We see that PET_WIDTH and PET_LENGTH have the same "goodness of split" (Gain ratio) and the same correlation (Tschuprow's t). In the last column, we observe if the split is accepted (green) or not (red).

So, in this dataset, SIPINA selects the PET_LENGTH attribute to splitting the first node, but the PET_WIDTH has the same relevance.

When we select the PET_WIDTH attribute, we see the cut point used (0.8) if we split the node with this attribute.

In order to split the node with PET_WIDTH, we double-click the GOODNESS OF SPLIT value in the grid.



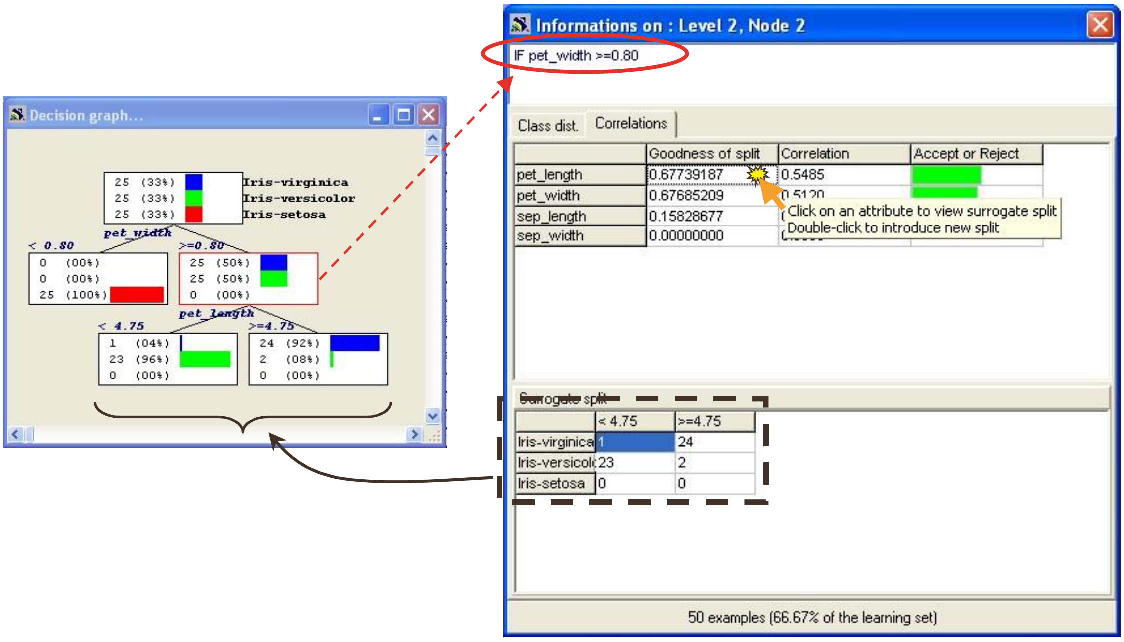
	Goodness of split	Correlation	Accept or Reject
pet_width	1.00000000	0.7071	🟩
pet_length	1.00000000	0.7071	🟩
sep_length	0.57390919	0.4763	🟩
sep_width	0.43614497	0.3143	🟩

	< 0.80	>=0.80
Iris-virginica	0	25
Iris-versicolor	0	25
Iris-setosa	25	0

25 (33%)	Iris-virginica
25 (33%)	Iris-versicolor
25 (33%)	Iris-setosa

< 0.80	>=0.80
0 (00%)	25 (50%)
0 (00%)	25 (50%)
25 (100%)	0 (00%)

If we split the next node with PET_LENGTH, we obtain the following tree.



Informations on : Level 2, Node 2

IF pet_width >=0.80

Class dist. | Correlations

	Goodness of split	Correlation	Accept or Reject
pet_length	0.67739187	0.5485	🟩
pet_width	0.67685209	0.5120	🟩
sep_length	0.15828677		
sep_width	0.00000000		

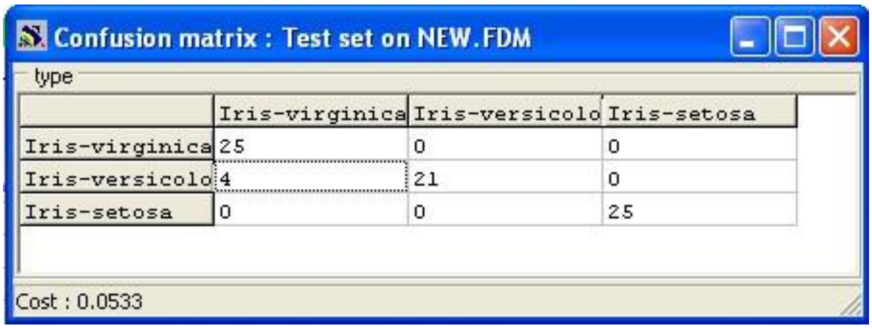
Click on an attribute to view surrogate split
Double-click to introduce new split

Surrogate split

	< 4.75	>=4.75
Iris-virginica	1	24
Iris-versicol	23	2
Iris-setosa	0	0

50 examples (66.67% of the learning set)

We can compute again the accuracy of this tree on the test set (ANALYSIS / TEST).



Confusion matrix : Test set on NEW.FDM

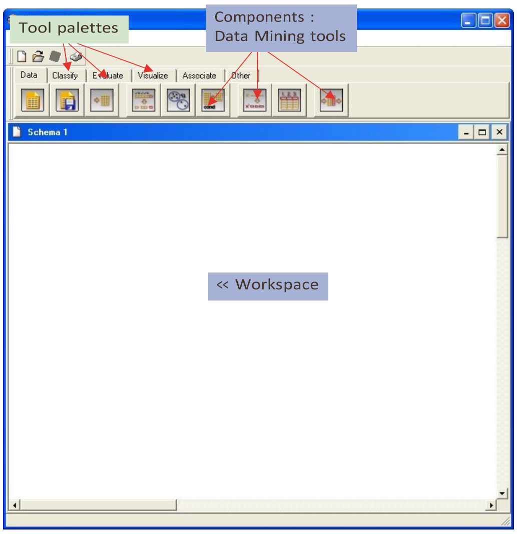
type

	Iris-virginica	Iris-versicolo	Iris-setosa
Iris-virginica	25	0	0
Iris-versicolo	4	21	0
Iris-setosa	0	0	25

Cost : 0.0533

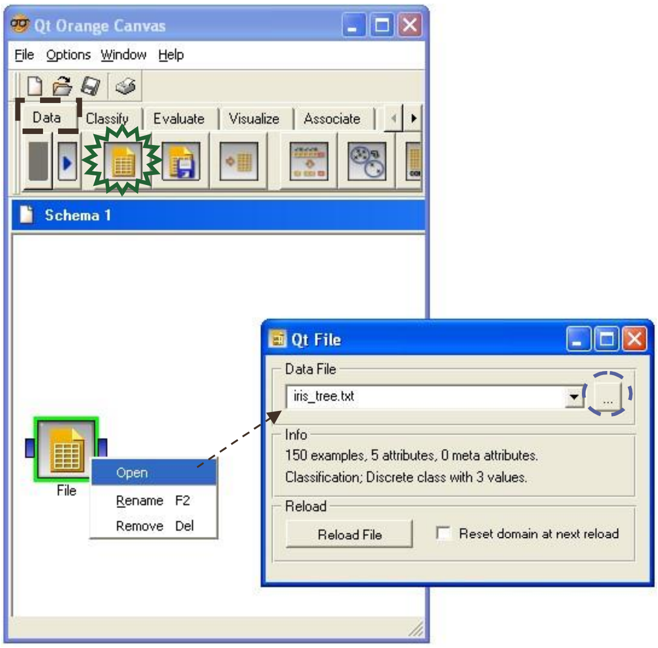
INTERACTIVE TREE BUILDER of ORANGE

When we start ORANGE, we have the following interface.



Data importation

We use the same dataset (IRIS_TREE.TXT). We set the DATA component in the workspace



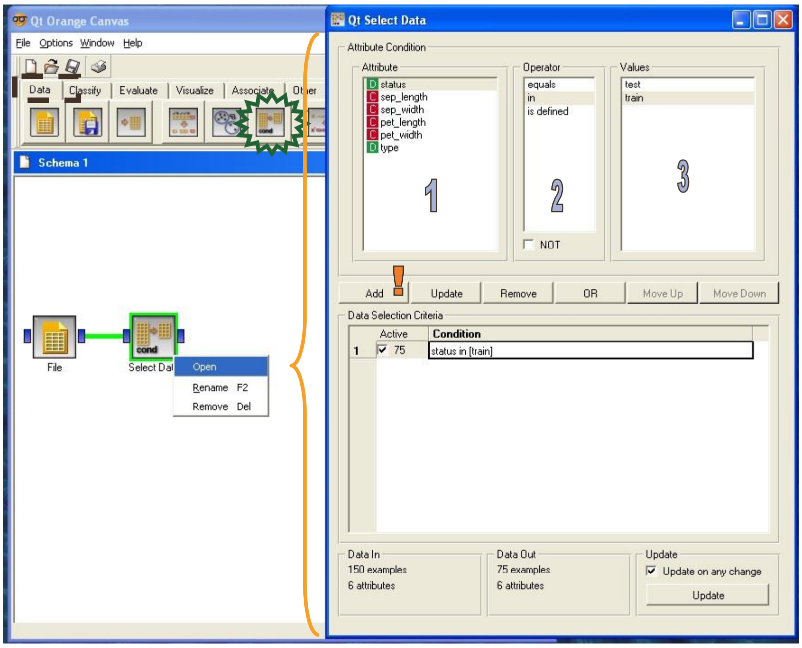
Data File: iris_tree.txt

Info
150 examples, 5 attributes, 0 meta attributes.
Classification; Discrete class with 3 values.

Reload
□ Reset domain at next reload

Selecting the train set

We want to subdivide the dataset into train and test set. We use the SELECT DATA component (DATA tab). The utilization of this component is similar to SIPINA.



Attribute Condition

Attribute: D status, C sep_length, C sep_width, C pet_length, C pet_width, D type

Operator: equals, in, is defined

□ NOT

Values: test, train

Data Selection Criteria

	Active	Condition
1	☑ 75	status in [train]

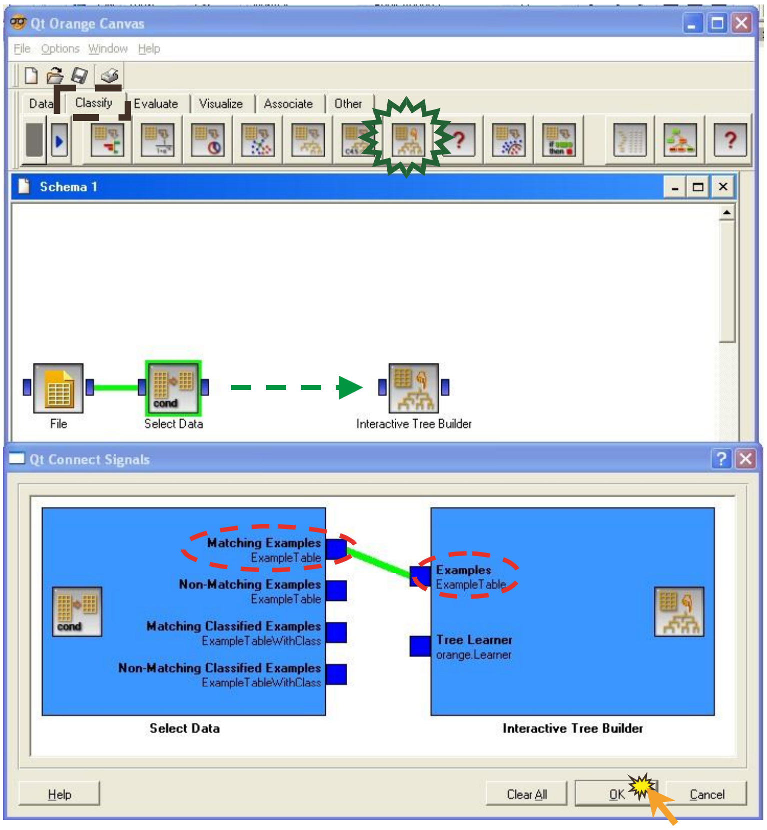
Data In
150 examples
6 attributes

Data Out
75 examples
6 attributes

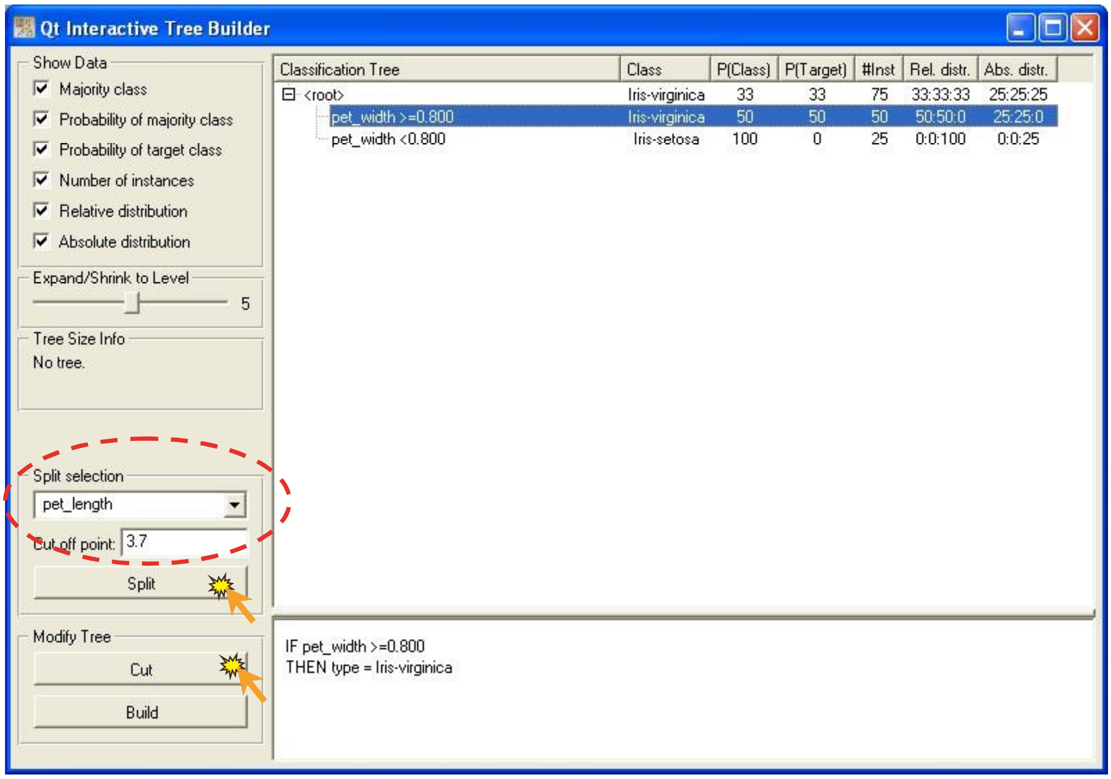
Update
☑ Update on any change

Interactive analysis

The train set is now selected (75 examples). We set the INTERACTIVE TREE BUILDER component (CLASSIFY tab) in the diagram. We must specify the nature of the connection when we connect SELECT DATA to this new component.



We click on the OPEN menu. The interactive tree builder tool appears.



Classification Tree	Class	P(Class)	P(Target)	#Inst	Rel. distr.	Abs. distr.
⊟ <root>	Iris-virginica	33	33	75	33:33:33	25:25:25
pet_width >=0.800	Iris-virginica	50	50	50	50:50:0	25:25:0
pet_width <0.800	Iris-setosa	100	0	25	0:0:100	0:0:25

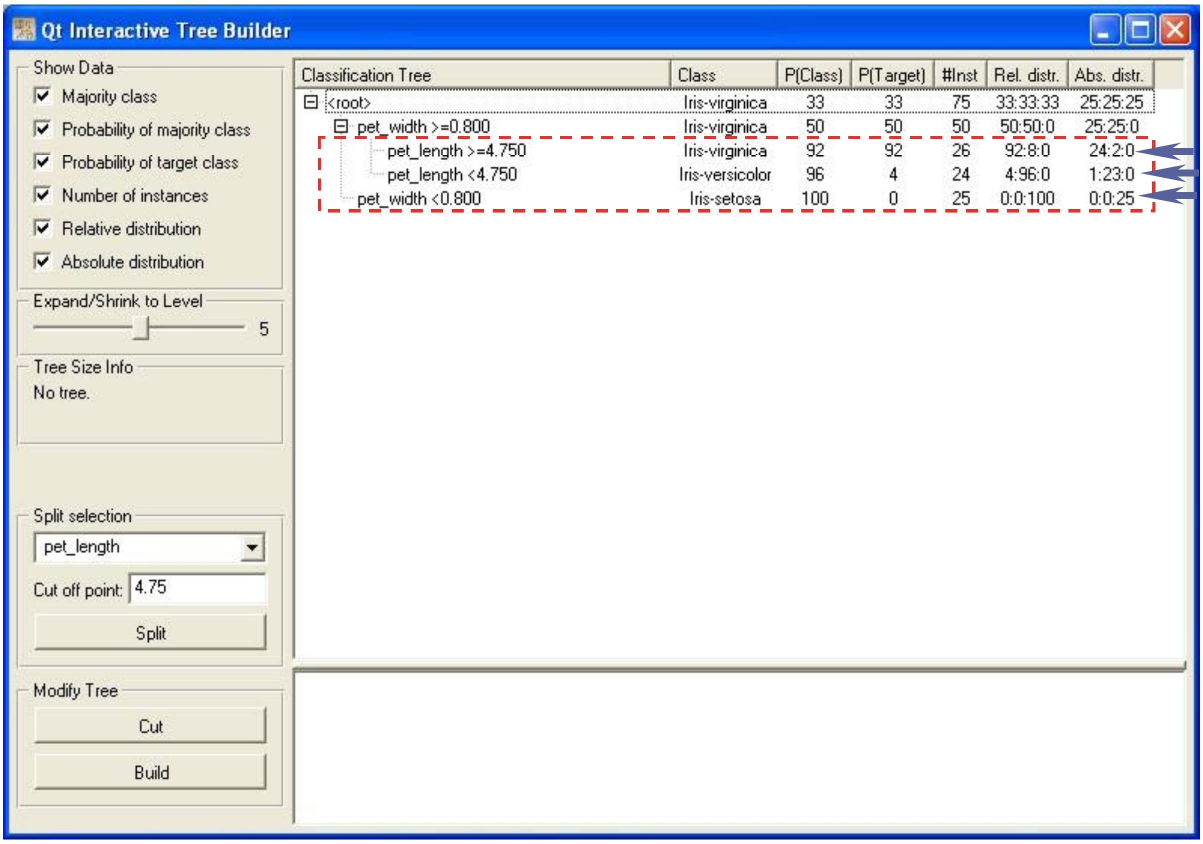
Split selection: pet_length

Cut off point: 3.7

IF pet_width >=0.800
THEN type = Iris-virginica

We can now interactively build the tree. The CUT option enables to prune a node; the SPLIT option enables to split a node. Let us note that we can modify the cut point in ORANGE. By default, it proposes the average of the attribute as the cut point.

In our case, we reproduce the tree that was built with SIPINA; we thus obtain the following result.



Show Data
☑ Majority class
☑ Probability of majority class
☑ Probability of target class
☑ Number of instances
☑ Relative distribution
☑ Absolute distribution

Expand/Shrink to Level 5

Tree Size Info
No tree.

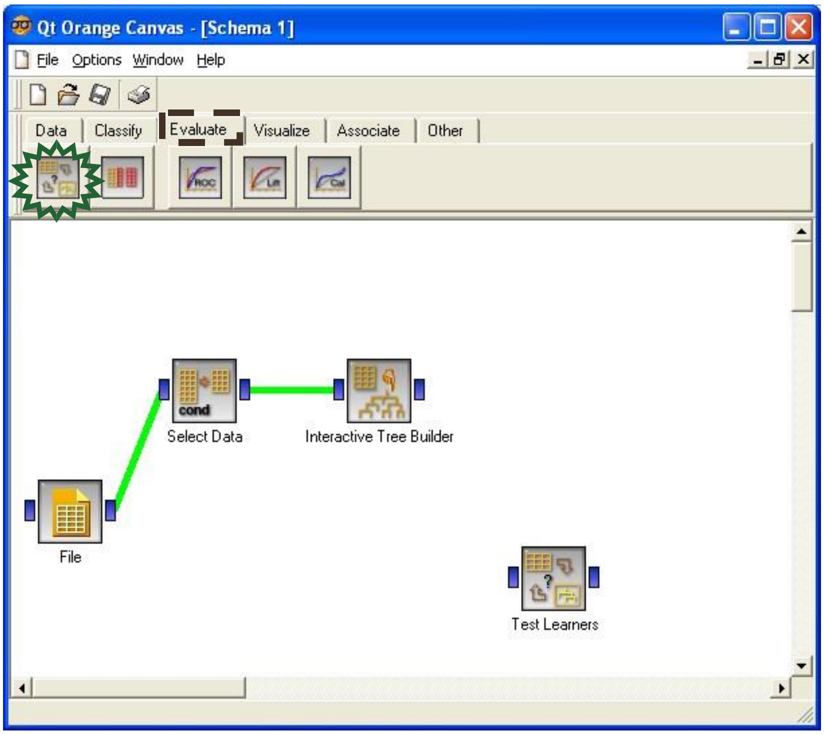
Split selection
pet_length
Cut off point: 4.75

Modify Tree

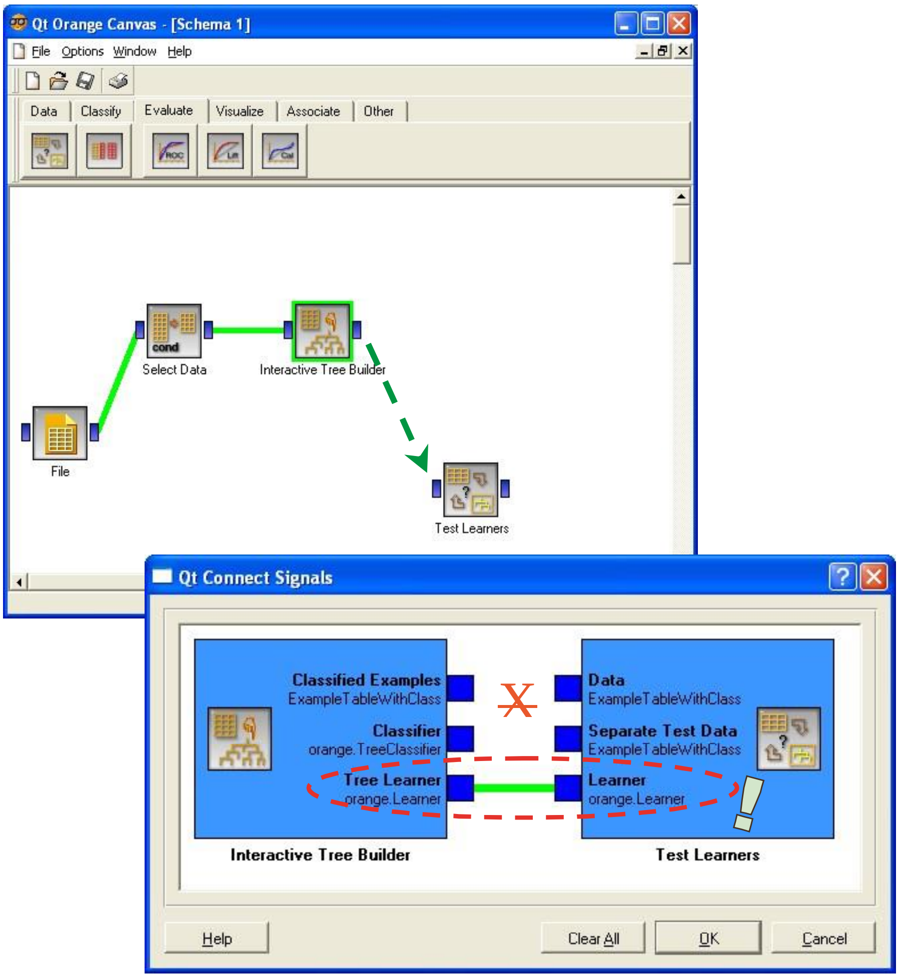
Classification Tree	Class	P(Class)	P(Target)	#Inst	Rel. distr.	Abs. distr.
⊟ <root>	Iris-virginica	33	33	75	33:33:33	25:25:25
⊟ pet_width >=0.800	Iris-virginica	50	50	50	50:50:0	25:25:0
pet_length >=4.750	Iris-virginica	92	92	26	92:8:0	24:2:0
pet_length <4.750	Iris-versicolor	96	4	24	4:96:0	1:23:0
pet_width <0.800	Iris-setosa	100	0	25	0:0:100	0:0:25

Accuracy rate on the test set

We want to compute the accuracy rate of this tree on the test set. We set a TEST LEANERS component on the diagram. We must connect the tree and the test set on this component.

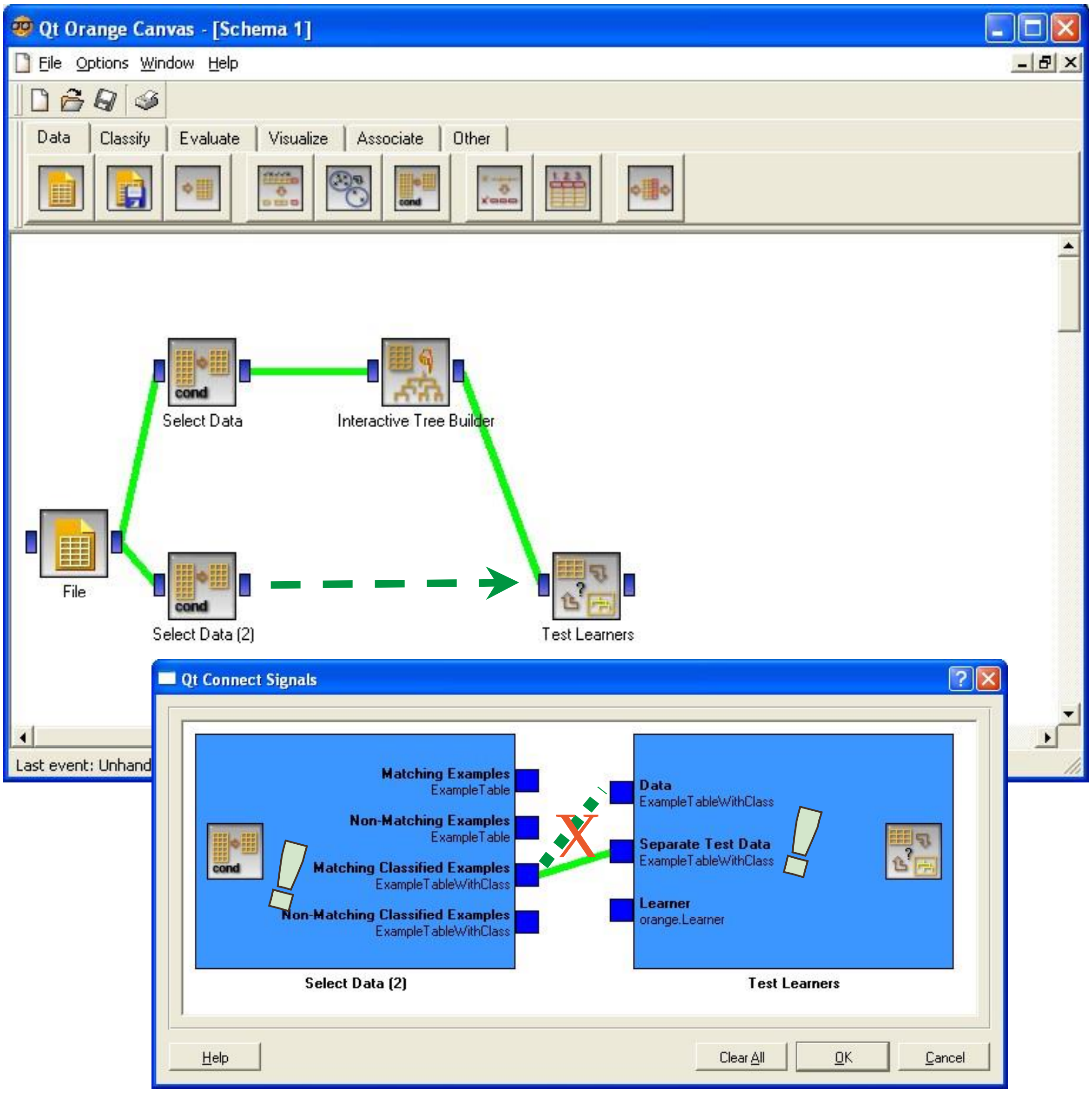

Qt Orange Canvas - [Schema 1]
File Options Window Help
Data Classify Evaluate Visualize Associate Other
Select Data
Interactive Tree Builder
File
Test Learners

In the first step, we want to connect the tree. A dialog box appears, it enables us to specify the right nature of the connection i.e. a LEARNER connection.

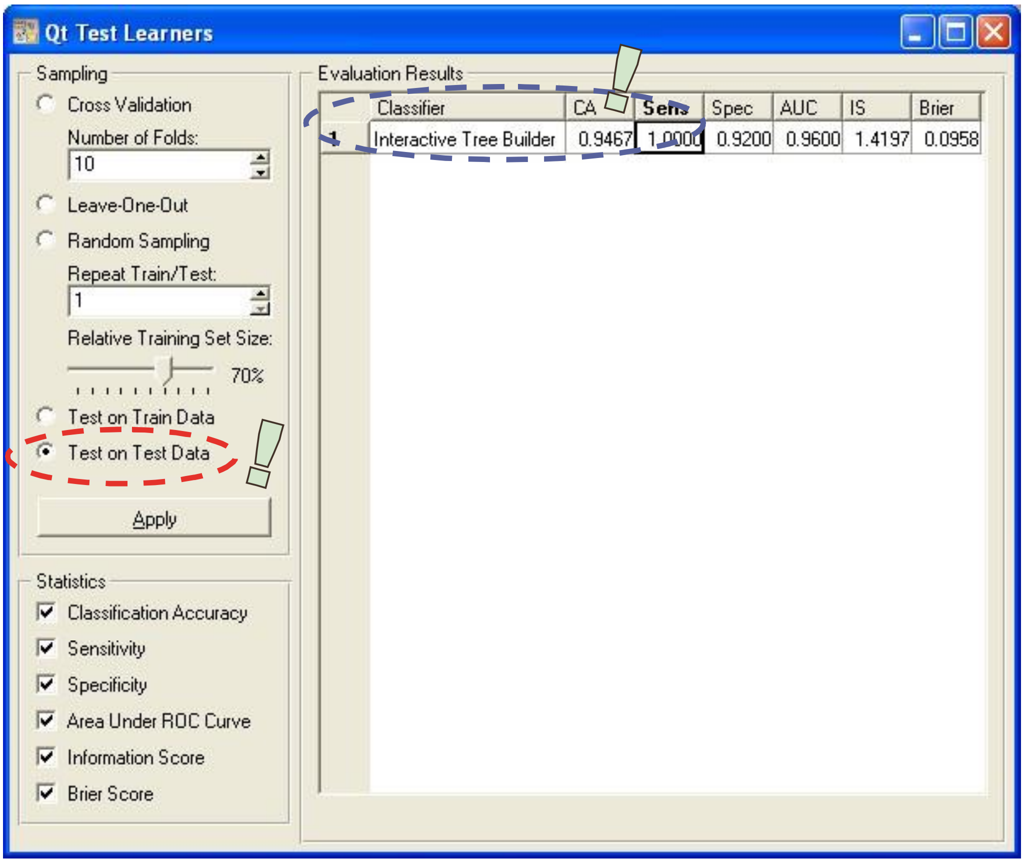


In the second step, we want to connect the test set to the TEST LEANERS component. We add a new SELECT DATA component in the diagram, we use this filtering rule: STATUS = TEST.

Then, we connect SELECT DATA to TEST LEANERS; we must use the SEPARATE TEST SET connection. We obtain the following diagram.



We click on the OPEN menu; we set the TEST option to TEST ON TEST DATA. We find the same results as with SIPINA. This is not surprising, we built identical trees.



Sampling

○ Cross Validation

Number of Folds: 10

○ Leave-One-Out

○ Random Sampling

Repeat Train/Test: 1

Relative Training Set Size: 70%

○ Test on Train Data

● Test on Test Data

Statistics

☑ Classification Accuracy

☑ Sensitivity

☑ Specificity

☑ Area Under ROC Curve

☑ Information Score

☑ Brier Score

Evaluation Results

	Classifier	CA	Sens	Spec	AUC	IS	Brier
1	Interactive Tree Builder	0.9467	1.0000	0.9200	0.9600	1.4197	0.0958

Conclusion

ORANGE and SIPINA are powerful tools for interactive tree construction.

Of course, some functionality miss compared to commercial packages (sorting attributes according the goodness of split for ORANGE, modifying the cut point for SIPINA, merging values for discrete attributes, etc.). But, in the majority of the situations, these tools enable to obtain very satisfactory results.

Another important information, these tools are free.