# 1 Topic

#### "Wrapper" for feature subset selection – Continuation.

This tutorial is the continuation of the preceding one about the wrapper feature selection in the supervised learning context (<u>http://data-mining-tutorials.blogspot.com/2010/03/wrapper-for-feature-selection.html</u>). We analyzed the behavior of **Sipina**<sup>1</sup>, and we have described the source code for the wrapper process (forward search) under R (<u>http://www.r-project.org/</u>). Now, we show the utilization of the same principle under Knime 2.1.1, Weka 3.6.0 and RapidMiner 4.6.

The approach is as follows: (1) we use the training set for the selection of the most relevant variables for classification; (2) we learn the model on selected descriptors; (3) we assess the performance on a test set containing all the descriptors.

This third point is very important. We cannot know the variables that will be finally selected. We do not have to manually prepare the test file by including only those which have been selected by the wrapper procedure. This is essential for the automation of the process. Indeed, otherwise, each change of setting in the wrapper procedure leading to another subset of descriptors would require us to manually edit the test file. This is very tedious.

In the light of this specification, it appeared that only Knime was able to implement the complete process. With the other tools, it is possible to select the relevant variables on the training file. But, I could not (or I did not know) apply the model on a test file containing all the original variables.

The naive bayes classifier is the learning method used in this tutorial<sup>2</sup>.

### 2 Dataset

We have partitioned the dataset into two distinct files<sup>3</sup>. We use the ARFF (Weka) file format. The first data file (**mushroom-train.arff**) is the training set, there are 2000 instances. We use this dataset for the wrapper process in order to select the most relevant descriptors, and for the construction of the final model with the selected descriptors.

The second one corresponds to the test set (6124 instances). We use this dataset for the assessment of the final model. As we say above, it must have the same organization than the training set. We do not know what will be the selected descriptors. We note also that this dataset is never used during the training phase, whether during the search of the relevant variables or during the construction of the final model.

<sup>&</sup>lt;sup>1</sup> <u>http://sipina.over-blog.fr/ext/http://eric.univ-lyon2.fr/~ricco/sipina.html</u>

<sup>&</sup>lt;sup>2</sup> <u>http://en.wikipedia.org/wiki/Naive\_Bayes\_classifier</u>

<sup>&</sup>lt;sup>3</sup> <u>http://eric.univ-lyon2.fr/~ricco/tanagra/fichiers/mushroom.wrapper.arff.zip</u>

## 3 Wrapper process using Knime

### 3.1 Importing the data file

We launch Knime. We create a new project by clicking on the FILE / NEW menu. The name of the new project is « Wrapper – Mushroom dataset ».

ile	Edit View S Sele	ect a wizard 🛛 💙 👘 🔤 😽	
-	New This	wizard creates a new KNIME workflow project.	
	Save	2010 - 20	-
	Save All Wiza	rds:	
	Close All	filter text	
ALTERNO.	Print	2 New KNIME Project	
	Import KNIME wor		
2	Export KNIME wor		∧.
	Switch Workspace	New KNIME Workflow Wizard	5
	Preferences	Create a new KNIME workflow.	_
	Export Preferences		
	Import Preference	Name of the workflow to create: Wrapper - Mushroom dataset	
	Update KNIME	PP	
-			
	😸 Database		
	Data Views		
	Σ Statistics		
100	Mining		
100	terenterenterenterenterenterenterentere		
3	M Time Series		. 1

Then, we add the ARFF READER (IO/READ branch into the "Node Repository") component into the workflow. We set the data filename **mushrom-train.arff**.

ARFF Reader	
	Dialog - 2:1 - ARFF Reader
Node 1	File
	Specify ARFF file Memory Policy
	Enter location of ARFF file:
	Row IDs are build from prefix + row number
	RowID prefix:

We check the dataset by using the INTERACTIVE TABLE component (DATAVIEWS). We can visualize the 2000 instances. "Classe" is the target attribute.

🛓 *2: Wrapper - Mush	room dataset 🛛							
ARFF Reader	Interactiv	e Table						
	>	-						
Node 1	Node	■ 2						
	🛕 Table View - 2	:2 - Interac	tive Table			V		x
	File Hilite Navig	ation View	w Output					
	Row ID	ring-nu	S ring-type	S spore-p	S populat	S habitat	S classe	
	0		e	n	s	g	e	-
	1		р	k	s	g	e	
	2		P	k	n	m	e	
	3			h	v	р	p	
	4		P	W	n	9	e	-
	5		e	W	V	-	p	- 2 1
	7		p D	k	v	d d	e	-
	8		e	w	v	1	D	-
	9		p	w	n	g	e	-
	10		e	w	v	p	p	-
		4						e l

#### 3.2 The « Feature Elimination » meta-node

We can define meta-nodes with Knime. This very enthusiastic functionality allows to define a typical sequence of operations. In our context, we use the predefined FEATURE ELIMINATION meta-node (META). It is based on the backward elimination of the irrelevant variables. It uses explicitly the error rate during the selection process.

Roughly speaking, the approach can be described as follows: we start with all the features; we remove the worst variable i.e. which induces the most favorable evolution of the error rate; we repeat the process until we removed all the variables. The selected subset is the one which is the best according the error rate.

We insert the FEATURE ELIMINATION component into the workflow. We connect twice the ARFF READER to this one. Twice, because we use the training set in order to select the best subset of variables and in order to learn the final model on the selected variables.



We double-click on the FEATURE ELIMINATION component to edit it.



The idea is to program a search loop. In our context, the BACKWARD FEATURE ELIMINATION START component is at the beginning of the loop, BACKWARD FEATURE ELIMINATION END is at the end. Inside of the loop, we partition the dataset (2000 instances) into a learning set (70%), it is used for the construction of the model on the current selection of the variables; and a validation set (30%), it allows to compute an unbiased estimation of the error rate. This subdivision is very important. Indeed, if we use the same dataset during these two phases, we have a biased estimation of the error, and above all, we favor the models with many variables.

**Note**: In fact, we have to use a resampling approach in order to measure the error rate during the wrapper process (e.g. cross validation). But, the implementation of this approach under Knime seems very complicated, even if it should be possible using another meta-node.

We add the PARTITIONING component (DATA MANIPULATION / ROW / TRANSFORM) into the workflow. We connect the first component. We set the parameters (CONFIGURE menu).



We add the NAIVE BAYES LEARNER and the NAIVE BAYES PREDICTOR as follows.



We can launch the wrapper process now. We click on the EXECUTE menu of the BACKWARD FEATURE ELIMINATION END search component.

We can follow the progress of operations in the "Console" window. When the process is finished, all components are green. Then, we configure the BACKWARD ELIMINATION FILTER FEATURE component which allows to choose the "optimal" subset of variables.

In the dialog box, we note that the minimal error rate is 0.002. It corresponds to a subset of 4 variables: odor, stalk-surface-above, ring-type and spore-print-color.



We confirm this subset. They are supplied at the output of the component. We select also the « Include Target Column » option. Then we click on the EXECUTE contextual menu.

### 3.3 Construction of the final model

We return in the original workspace tab. We observe that the two outputs of the FEATURE ELIMINATION meta-node are green.



The second output is very interesting for us. Because, it supplies all the instances of the training set with only the selected variables (and the target column). We add the NAÏVE BAYES LEARNER (CLASSE is the target attribute, CONFIGURE menu). We click on the EXECUTE AND OPEN VIEW menu. We obtain the following results. These are the cross-tabulation of each predictor with the target attribute.



#### 3.4 Error rate evaluation on the test set

We intend to use the test set (6124 instances) in order to evaluate the accuracy of the classifier. We load the data file with the ARFF READER component.

🔺 *2: Wrapper - Mushroom dataset 🛛 🛕 2:3: Feature Elimination	, 🗆
Interactive Table Interactive Table Node 2 ARFF Reader Node 2 ARFF Reader Node 3 Node 3 Node 4 Node 4	E

Then we add the NAIVE BAYES PREDICTOR (*MINING / BAYES*). It takes as input the test sample and the classifier from the "learner".



To compute the confusion matrix and the test error rate, we add the SCORER component (*MINING* / *SCORING*). With the CONFIGURE menu, we indicate that CLASS corresponds to the target attribute and WINNER contains the predicted values of the model.



We click on the EXECUTE AND OPEN VIEW menu. We obtain the following confusion matrix. The test error rate is 0.212% (13 misclassified instances among 6124).

File Hilite			
classe \ Wi	e	p	
e	3180	0	
p	13	2931	

## 4 Wrapper process using Weka

Into the EXPLORER mode, Weka allows to implement efficiently the wrapper selection approach.

Unlike Knime, it is very easy to perform the cross validation error rate evaluation during the process. But, on the other hand, it seems not possible (*I did not found*) to apply the classifier computed on the selected variables on a test set including all the descriptors.

Thus, in this section, we show only how to perform the wrapper feature selection process with Weka. The error rate is computed with the cross validation resampling scheme.

### 4.1 Importing the learning set

We launch Weka. We select the EXPLORER mode.

Weka GUI Chooser		
rogram Visualization Tools Help		
WEKA The University of Waikato	Explorer Experimenter Weka Explorer	
Waikato Environment for Knowledge Analysis	Kno Preprocess Classify Cluster Associate Select attributes Visual	22
(c) 1999 - 2008	Open file Open URL Open DB	Generate Undo Edit Save
The University of Waikato Hamilton, New Zealand	S Filter	
	Choose None	Apply
	Current relation	Selected attribute
	Relation: None Instances: None Attributes: None	Name: None Type: None Missing: None Distinct: None Unique: None
	Attributes	
	All None Invert Patte	ern
		▼ Visualize A
		, b
	Remove	
	Status	
	Welcome to the Weka Explorer	Log 🧼

Into the PREPROCESS tab, we click on the OPEN FILE button. We select mushroom-train.arff.

(	Se Ouvrir				x	
	Rechercher dans	s : 🕕 wrapper		• 🔰 🖬 📰		
<ul> <li>Weka Explorer</li> <li>Preprocess</li> </ul>	Documents	mushroom-test.aiff				
Open file	récents					
Filter Choose None	Bureau	🖤 Weka Explorer	2			×
Current relation Relation: None Instances: None	E.	Preprocess Classify Cluster Associate Sale	t attri	outes Visualize		
Attributes	Documents	Open Open	Ger	ner	Edit Sav	e
	Ordinateur	Choose None		505 II. WIRDAN AR		pply
	<b>N</b>	Current relation Relation: DataTable Instances: 2000 Attributes: 23		Selected attribute Name: cap-shape Missing: 0 (0%) Distinc	Type: No t: 5 Unique: 0	ominal (0%)
	Réseau	Attributes		No. Label	Count	
		All None Inv Pat.	]	1 f 2 x	817 874	_
		No. Name		3 b 4 k	106 195	=
		1 cap-shape 2 cap-surface		5 s 6 c	8	-
	Remove	3 cap-color 4 bruises?	E	Class: classe (Nom)	▼ Visual	ize All
Status Welcome to the Weka E	xplorer	5 odor 6 gill-attachment 7 gill-spacing		817		
		8 gill-size 9 gill-color 10 stalk-shape	-			
		Remove		108	195	
		Status			Log	0 🔿 x 0
		OK			Log	n ×

#### 4.2 Implementing the "wrapper" process using Weka

We activate the SELECT ATTRIBUTES tab. We choose (CHOOSE button) the WRAPPERSUBSETEVAL approach as ATTRIBUTE EVALUATOR.

Preprocess Classify Cluster Asso	ciate Select attributes Visualize	
Attribute Evaluator		
Choose WrapperSubsetEv	al -B weka.classifiers.rules.ZeroR -F 5 -T 0.01 ·	-R 1
Search Method		
Choose BestFirst -D 1 -N 5		
Attribute Selection Mode	Attribute selection output	
Ose full training set		*
Cross-validation Folds	3	
Seed 1		=
(Nom) dasse	•	
Start Stop		
Result list (right-click for options)		
	۲. m	•

We click on the description of the approach to define the settings. We set the learning method and the number of folds for the cross validation.

Preprocess Classify Cluster	Associate Select attributes Visualize			
Attribute Evaluator				
Choose WrapperSub	setEval -B weka.classifiers.rules.ZeroR -F 5 -T 0.01 -R 1 🦳			
Search Method				
Choose BestFirst -D :	weka.gui.GenericObjectEditor			
	weka.attributeSelection.WrapperSubsetEval			
Attribute Selection Mode	About			
Ose full training set	WrapperSubsetEval: More			
O Cross-validation Fol	Capabilities			
Se	Evaluates attribute sets by using a learning scheme.			
(Nom) dasse	dassifier Choose NaiveBayes			
(Nom) classe	dassifier Choose NaiveBayes			
(Nom) dasse	dassifier Choose NaiveBayes			
(Nom) dasse Start S Result list (right-click for optio	dassifier Choose NaiveBayes			
(Nom) dasse Start S Result list (right-dick for optio	dassifier Choose NaiveBayes			
(Nom) classe Start S Result list (right-click for optio	dassifier Choose NaiveBayes folds 5 seed 1 threshold 0.01			
(Nom) classe Start S Result list (right-click for optio	dassifier Choose NaiveBayes folds 5 seed 1 threshold 0.01 Open Save OK Cancel			

About the SEARCH METHOD, we want to use the BACKWARD strategy. We select the GREEDY STEPWISE approach.

Preprocess Classify Cluster Ass	sociate Select attributes Visualize	
Attribute Evaluator		
Choose WrapperSubsetE	val -B weka.classifiers.bayes.NaiveBayes -F 5 -T 0.01 -R 1	
Search Method		
Choose GreedyStenwise	-T -1.7976931348623157E308 -N -1	
circuit circuit stephise		
Attribute Selection Mode	Attribute selection output	
Ose full training set		*
Cross-validation Folds	10	
Seed [	1	E
A		-
(Nom) classe	•	
Start Stop		
Result list (right-click for options)		
		X
Status		
OK	Log	, х

By clicking on the description of the approach, we set SEARCHBACKWARDS = TRUE.

Preprocess (	Classify	Cluster	Associate Select attributes Visualize
Attribute Eva	luator		
Choose	Wrap	perSubs	tEval -B weka.classifiers.bayes.NaiveBayes -F 5 -T 0.01 -R 1
Search Metho	d		
Choose	Greed	lyStepw	e -T -1.7976931348623157E308 -N -1
Use full f     Cross-va	<ul> <li>Use full training set</li> <li>Cross-validation Folds</li> <li>Seed</li> </ul>		weka.attributeSelection.GreedyStepwise     About     GreedyStepwise :     More     Performs a greedy forward or backward search through the
(Nom) classe		Sto	conservativeForwardSelection False
Result list (rig	ht-click f	or option:	generateRanking False
Status OK			numToSelect -1 searchBackwards startSet
			threshold -1.7976931348623157E308

We launch the search process by clicking on the START button. We note that the procedure uses all the available instances (ATTRIBUTE SELECTION MODE = USE FULL TRAINING SET).



Odor, gill-size and stalk-surface-above are the selected variables.

#### 4.3 Weighting of attributes using the cross validation

The approach described above is the same as the one implemented using Sipina, R (<u>http://data-mining-tutorials.blogspot.com/2010/03/wrapper-for-feature-selection.html</u>) and Knime (above).

We obtain an "optimal" subset of variables. But we know that this solution relies heavily on the learning set. If we use another dataset, we could obtain another subset. Sometimes, it can be very different. This is rather embarrassing, especially when we want to interpret the results in relation wih the domain knowledge.

Weka puts forward a very interesting approach to overcome this drawback. Instead of performing a unique search session, it repeats the search using a resampling scheme. We thus obtain various subsets of variables. We profit from these solutions to count the number of occurrence of each variable in these subsets. The goal is to obtain a weighting of the variables according to their apparition during the whole process.

We select the ATTRIBUTE SELECTION MODE = CROSS-VALIDATION (FOLDS = 10) option. We click again on the START button.

eprocess Classify Cluster As	sociate Select attributes Visualize
tribute Evaluator	
Choose WrapperSubsetE	val -B weka.classifiers.bayes.NaiveBayes -F 5 -T 0.01 -R 1
earch Method	
channel Consider Channels	D. T. 1. 202400104040001525000 AL 1
Greedyscepwise	-B -1 -1./9/093134602315/E306 -N -1
tribute Selection Mode	Attribute selection output
🖱 Use full training set	
Cross-validation	10 === Attribute selection 10 fold cross-validation (s
6	number of folds (%) attribute
om) classe	▼ 0( 0 %) 1 cap-snape
	0(0\$) 2 cap-surrace
Start Stop	3(30.8) 4 bruises2
sult list (right-click for options)	10(100 %) 5 odor
	0(0%) 6 gill-attachment
:49:52 - GreedyStepwise + wrap	3(30 %) 7 gill-spacing
24.40 - Greedystepwise + wrap	7(70%) 8 gill-size
	0( 0 %) 9 gill-color
	0( 0 %) 10 stalk-shape
	0( 0 %) 11 stalk-root
	10(100 %) 12 stalk-surface-above-
	0( 0 %) 13 stalk-surface-below-
	0( 0 %) 14 stalk-color-above-ri
	0( 0 %) 15 stalk-color-below-ri
	0( 0 %) 16 veil-type
	0( 0 %) 17 veil-color
	0( 0 %) 18 ring-number
	0( 0 %) 19 ring-type
	0( 0 %) 20 spore-print-color
	0( 0 %) 21 population
	0( 0 %) 22 habitat
	• m •
atus	
1035	

Some attributes are highlighted. For the 10 sessions of the cross validation, 5 of them are included at least one times in an "optimal" subset: bruises (3 times), odor (10), gill-spacing (3), gill-size (7) and stalk-surface-above (10). That relativizes the result with only 3 variables highlighted in the previous section.

### 4.4 « Ranking » of attributes

Weka puts forward another approach to overcome again the drawback of the "crisp" solution supplied by the standard wrapper selection process: this is the "ranking" approach.

About the backward search, the least important variable is the first removed. The most important one is the last removed. Weka assigns a weight to each variable according to this point of view. The result (the ranking) is smoothed by the encapsulation of this process inside a resampling scheme.

We modify the SEARCH METHOD, we set GENERATE RANKING = TRUE.

Preprocess Classify Clus	er Associate Select attributes Visua	alize
Attribute Evaluator		
Choose WrapperS	<b>bsetEval</b> -B weka.classifiers.bayes.Na	iveBayes -F 5 -T 0.01 -R 1
Search Method		
Choose GreedySte	owise -B -T -1.7976931348623157E30	8 -N -1
Attribute Selection Mode	🔷 weka.gui.GenericObjectE	ditor 📃
<ul> <li>Ose full training set</li> <li>Cross-validation</li> </ul>	olds 1 About GreedyStepwise	
(Nom) dasse	Performs a greedy forwar space of attribute subset:	d or backward search through the
(Nom) classe	Performs a greedy forwar space of attribute subset	d or backward search through the
(Nom) classe Start Result list (right-click for op	Performs a greedy forwar space of attribute subset conservativeForwardSelection ons) generateRanking	d or backward search through the s. False
(Nom) classe Start Result list (right-click for op 14:49:52 - GreedyStepwise 15:24:46 - GreedyStepwise	Performs a greedy forwar space of attribute subset conservativeForwardSelection generateRanking + Wrap humToSelect	d or backward search through the False True I True I T
(Nom) classe Start Result list (right-click for op 14:49:52 - GreedyStepwise 15:24:46 - GreedyStepwise	Performs a greedy forwar space of attribute subset conservativeForwardSelection ons) + Wrap + Wrap + Wrap	d or backward search through the False True True
(Nom) classe Start Result list (right-click for op 14:49:52 - GreedyStepwise 15:24:46 - GreedyStepwise	Performs a greedy forwar space of attribute subset stop ons) + Wrap + Wrap + Wrap searchBackwards startSet	d or backward search through the False Induction for the search through through the search through the search through through the search through
(Nom) classe Start Result list (right-click for op 14:49:52 - GreedyStepwise 15:24:46 - GreedyStepwise 4 III	Performs a greedy forwar space of attribute subset conservativeForwardSelection ons) + Wrap + Wrap searchBackwards startSet threshold	d or backward search through the s. False • True • -1 True • -1. True • -1. True •

Then we click again on the START button.



**Average rank**. Because we use a backward search, we must read the result from the bottom to the top. The most important variables are in the bottom part of the table. We have the average of the ranks of the variable. We have also an indication about the reliability of the assigned rank. It seems it is the standard error. We note for instance that "odor" has obtained always the rank 22.

## 5 Wrapper process using RapidMiner

RapidMiner is very similar to Weka about the wrapper process. It can supply a ranking or a weighting of attributes based on a resampling scheme. As Weka also, we cannot deploy the final model with the selected variables on a dataset containing all the original attributes.

### 5.1 Standard wrapper using RapidMiner

We launch RapidMiner. We create a new diagram by clicking on the FILE / NEW menu. We save the diagram "wrapper-mushroom.xml ".



We must first load the dataset. We add the ARFFEXAMPLESOURCE component (*NEW OPERATOR* / *IO* / *EXAMPLES*) into the diagram. We set CLASSE as target (label) variable.

RapidMiner@VGC (wrapper-mushroom.xml*)		
<u>Eile Edit View Process Tools H</u> elp		
🞦 📁 🔒 🖶 🔌 🔗 🔌 鱼	🔎 🐌 📗	🧳 🚨 🕎 🖗
Coperator Tree	Parameters 💽 XML	Comment New }
E Root	data_file	perimushroom-train.arf
- Process	label_attribute	classe 💋
	id_attribute	
	weight_attribute	
	datamanagement	double_array 🔹
	decimal_point_character	
	sample_ratio	1.0
	sample_size	-1
	local_random_seed	-1
G Jan 31, 2010 3:42:04 PM: [Error] Parameter 'data_file' is n default value.	not set and has no	1 - 1 CB
0		3:42:36 PM

We add the FEATURE SELECTION component (*PREPROCESSING / ATTRIBUTES / SELECTION*). We specify the search strategy (SELECTION\_DIRECTION = BACKWARD).

Coperator Tree	Parameters E XML	🖹 Comment 📔 🍋 New Operator	
Root Process	user_result_individual_selection		
ArffExampleSource	show_population_plotter		
ArffExampleSource	plot_generations	10	
FeatureSelection	constraint_draw_range		
reatureselection	draw_dominated_points		
	population_criteria_data_file		
	maximal_fitness	Infinity	
	selection_direction	backward	
	keep_best	4	
	and the second s		

We use the cross validation to evaluate the error rate (5 folds and stratified sampling).

n n 🖇 🗧 📕 🔄 🖉	1 🖉 🖉 🕨 🔳	🖌 🗧 🔣 🦉
Coperator Tree	Parameters 🔄 🖾 XML	. 📄 Comment 🧯 🐿 New Operator
Root	keep_example_set	
- ArffEvormploCourco	create_complete_model	
ArffExampleSource ArffExampleSource	average_performances_only	
	leave_one_out	
Walidation	number_of_validations	5
Walidation	sampling_type	stratified sampling
	local_random_seed	-1
9 Jan 31, 2010 3:42:04 PM: <b>[Error]</b> Parameter 'da Jefault value.	ata_file' is not set and has no.	e: 1.1-CB sl: 1.1 CB

Then we add the train-test sequence inside of the cross-validation (NAIVE BAYES + MODEL APPLIER + CLASSIFICATION PERFORMANCE). The last component allows to compute the error rate on the various folds.

• 🖉 🎽 📗	🖌 🤰 📓 🖌	
Parameters 🕞 XML	📄 Comment ी 🎦 New Operator	
keep_example_set		
main_criterion	(first 🔹	
accuracy		
classification_error		
kappa		
weighted_mean_recall		
weighted_mean_precision		
spearman_rho		
kendall_tau		
absolute_error		
relative_error		
relative_error_lenient		
is not set and has no Heat: Tota	1.1 CB 1.1 CB	
	Parameters XML     keep_example_set     main_criterion     accuracy     classification_error     kappa     weighted_mean_recall     weighted_mean_recision     spearman_rho     kendall_tau     absolute_error     relative_error     relative_error     relative_error     is not set and has no	

We launch the process by clicking on the RUN button. We obtain the selected variables (ATTRIBUTE WEIGHTS) and the confusion matrix of the final model (PERFORMANCE VECTOR).

🚱 RapidMiner@VGC (wrapper-mushroom.xml)			
<u>Eile E</u> dit ⊻iew <u>P</u> rocess <u>T</u> ools <u>H</u> elp			
🖹 📁 📕 🖶 😂 🛛 🛯	🖕 🖉 🚺	🥖 🔒	<b>1</b>
ExampleSet AttributeWeights FeatureSelection	PerformanceVector ClassificationPerformance		
Table View O Plot View			
attribute		weight 🔻	
cap-shape	1		6
cap-surface	1		
cap-color	1		
bruises?	1		
odor	1		
gill-attachment	1		
gill-spacing	1		
gill-size	1		
gill-color	1		
stalk-shape	1		
stalk-root	1		
stalk-surface-above-	1		6
			Save
Penormancevector [ classification_error: 0.35% +/- 0.12% (mikro: 0.3) ConfusionMatrix: True: e p	5%)		
e: 1028 7 p: 0 965 l			
(created by ClassificationPerformance) P Jan 31, 2010 3:55:13 PM: [NOTE] Process finished	d successfully after 2 s	Max: 1.1 GB Total: 1.1 GB	
0			3:56:00 PI

RapidMiner surprisingly selects many variables compared with the other tools.

ile <u>E</u> dit ⊻iew <u>P</u> rocess <u>T</u> ools <u>H</u> elp			
🖞 💭 📕 🛃 😓 🔗 🐼 🖕	🔎 🖉 🌔 🔳 4	V 🚨	<b>F</b>
📲 Operator Tree	Parameters 🕞 XML	📄 Comment 🏾 🤷 New	Operator
Root	number_of_validations	5 🗾	
Process	leave_one_out		
ArffExampleSource	sampling_type	stratified sampling	· •
www.wrapperXValidation	local random seed	-1	
WrapperXValidation	and an association of the second		
😑 🚲 FeatureSelection			
FeatureSelection			
🖃 🌇 XValidation			
XValidation			
NaiveBaves			
NaiveBayes			
NaiveBayes NaiveBayes			
NaiveBayes NaiveBayes ⊖			
Validation NaiveBayes NaiveBayes OperatorChain OperatorChain			
Xualidation     NaiveBayes     OperatorChain     OperatorChain     GyeratorChain     ModelApplier     ModelApplier			
Validation NaiveBayes NaiveBayes OperatorChain OperatorChain ModelApplier ModelApplier ClassificationPerformance ClassificationPerformance			
XValidation     NaiveBayes     NaiveBayes     OperatorChain     OperatorChain     OperatorChain     OperatorChain     ClassificationPerformance     ClassificationPerformance     ClassificationPerformance			
Xoalidation     NaiveBayes     NaiveBayes     NaiveBayes     OperatorChain     OperatorChain     OperatorChain     OperatorChain     ClassificationPerformance     ClassificationPerformance     classificationPerformance     component/leaders			
X0alrdation NaiveBayes NaiveBayes NaiveBayes OperatorChain OperatorChain OperatorChain ClassificationPerformance ClassificationPerformance classificationPerformance reated by FeatureSelection) erformanceVector [classification_error: 0.35% +/- 0.12% (mikro: 0.35%)]			
Avairedation NaiveBayes NaiveBayes NaiveBayes NaiveBayes ModelApplier ModelApplier ClassificationPerformance ClassificationPerformance ClassificationPerformance ClassificationPerformance readed by FeatureSelection) erformanceVector [ classification_error: 0.35% +/- 0.12% (mikro: 0.35%) onfusionMatrix:			
AlveBayes NaiveBayes NaiveBayes NaiveBayes OperatorChain OperatorChain OperatorChain ClassificationPerformance Classificat			
AlveBayes NaiveBayes NaiveBayes NaiveBayes OperatorChain OperatorChain ClassificationPerformance ClassificationPerformance ClassificationPerformance classificationPerformance classificationPerformance classificationPerformance classificationPerformance classificationPerformance classificationPerformance classificationPerformance to the			

### 5.2 Weighting of the attributes according to their relevance

Like Weka, RapidMiner can compute a weighting of attributes using a resampling scheme. We come back to the diagram. We add the WRAPPER **XVALIDATION** (VALIDATION) component. It embeds the selection process into a cross validation. We copy the previous branch of the diagram devoted to the wrapper process inside to this component.

Then we define again the train test sequence using the naive bayes classifier.



We click on the RUN button. Of course, the computation time is higher. We select the ATTRIBUTE WEIGHTS tab.

<u>File Edit View Process Iools Help</u>			
🞦 📁 📗 🖶 🌭 🕜 🔍 🐂 🖈	- 🐌 🕨 🛛	🧭 🔒	2
BerformanceVector ClassificationPerformance (2) & WrapperXValidation			
Table View O Plot View			
attribute		weight 🔻	
cap-surface	1		2
cap-color	1		
odor	1		
gill-attachment	1		
gill-spacing	1		
gill-size	1		
gill-color	1		
stalk-shape	1		
stalk-root	1		
stalk-surface-above-	1		
veil-type	1		
veil-color	1		
spore-print-color	1		
population	1		
stalk-surface-below-	0.800		
stalk-color-above-ri	0.800		
ring-type	0.800		5
			Save
AttributeWeights (containing weights for 22 attributes) (created by WrapperXValidation) P Jan 31 2010 4:07:52 PM: INOTE! Process finished successf	ullv after 16 s	Mas 1 00 Total: 1.1 00	

Here also, the previous result is relativized. But, the number of relevant variables is clearly overestimated. Perhaps it is more appropriate to use the forward strategy with RapidMiner. It seems that it selects the first subset found according to the search direction: the largest subset in the backward search; the smallest subset in the forward search.

## 6 Conclusion

Wrapper strategy explicitly uses an estimation of the error rate to determine the "optimal" subset of descriptors. It does not rely on the specificities of the learning algorithm when it explores the solutions. These are two characteristics that make it a preferred tool in a variable selection process.

However, as we noted in this tutorial, it is demanding on computational resources, especially when we want to highlight the rank or the weight of the attributes using a resampling scheme. Its implementation when dealing with very large databases, both in numbers of variables and observations, becomes very expensive. This characteristic is even more pronounced than we use a supervised learning technique which is also costly in computing time, such as some neural network or support vector machine algorithms.