

WEKA Experimenter Tutorial for Version 3-5-6

David Scuse Peter Reutemann

June 1, 2007

 $\textcircled{O}2002\mathchar`-2007$ David Scuse and University of Waikato

Contents

1	Intr	troduction					
2	Star	ndard I	Experiments	3			
	2.1	Simple	-	3			
		2.1.1	New experiment	3			
		2.1.2	Results destination	3			
		2.1.3	Experiment type	5			
		2.1.4	Datasets	7			
		2.1.5	Iteration control	8			
		2.1.6	Algorithms	8			
		2.1.7	Saving the setup	10			
		2.1.8	Running an Experiment	11			
	2.2	Advan	ced	12			
		2.2.1	Defining an Experiment	12			
		2.2.2	Running an Experiment	15			
		2.2.3	Changing the Experiment Parameters	17			
		2.2.4	Other Result Producers	24			
3	Ren	note E	xperiments	29			
	3.1	Prepar	ration	29			
	3.2	Databa	ase Server Setup	29			
	3.3	Remot	e Engine Setup	30			
	3.4	Config	uring the Experimenter	31			
	3.5	Troubl	leshooting	32			
4	Ana	lysing	Results	33			
	4.1	Setup		33			
	4.2	Saving	the Results	36			
	4.3		ing the Baseline Scheme	36			
	4.4		ical Significance	37			
	4.5		ary Test	37			
	4.6	Rankir	ng Test	38			

1 Introduction

The Weka Experiment Environment enables the user to create, run, modify, and analyse experiments in a more convenient manner than is possible when processing the schemes individually. For example, the user can create an experiment that runs several schemes against a series of datasets and then analyse the results to determine if one of the schemes is (statistically) better than the other schemes.

The Experiment Environment can be run from the command line using the Simple CLI. For example, the following commands could be typed into the CLI to run the OneR scheme on the Iris dataset using a basic train and test process. (Note that the commands would be typed on one line into the CLI.)

```
java weka.experiment.Experiment -r -T data/iris.arff
-D weka.experiment.InstancesResultListener
-P weka.experiment.RandomSplitResultProducer --
-W weka.experiment.ClassifierSplitEvaluator --
-W weka.classifiers.rules.OneR
```

While commands can be typed directly into the CLI, this technique is not particularly convenient and the experiments are not easy to modify.

The Experimenter comes in two flavours, either with a simple interface that provides most of the functionality one needs for experiments, or with an interface with full access to the Experimenter's capabilities. You can choose between those two with the *Experiment Configuration Mode* radio buttons:

- Simple
- Advanced

Both setups allow you to setup *standard* experiments, that are run locally on a single machine, or remote experiments, which are distributed between several hosts. The distribution of experiments cuts down the time the experiments will take until completion, but on the other hand the setup takes more time.

The next section covers the *standard* experiments (both, simple and advanced), followed by the *remote* experiments and finally the *analysing* of the results.

This manual is also available online on the WekaDoc Wiki [7].

2 Standard Experiments

2.1 Simple

2.1.1 New experiment

After clicking New default parameters for an Experiment are defined.

😓 Weka Experiment Environment				_ 🗆 ×
Setup Run Analyse				
xperiment Configuration Mode:		Simple	Advanced	
Open	ş	Save	New	
Results Destination				
ARFF file Filename:				Browse
Experiment Type		Iteration Control		
Cross-validation	-	Number of repetitions:	10	
Number of folds: 10		Data sets first		
Classification	on	Algorithms first		
Datasets		Algorithms		
Add new Edit selecte	Delete select	Add new	Edit selected De	lete selected
Use relative pat				
		1		
Up	Down	Load options	Save options U	p Down
	N	lotes		

2.1.2 Results destination

By default, an ARFF file is the destination for the results output. But you can choose between

- $\bullet~{\rm ARFF}$ file
- $\bullet~\mathrm{CSV}$ file
- JDBC database

ARFF file and JDBC database are discussed in detail in the following sections. CSV is similar to ARFF, but it can be used to be loaded in an external spreadsheet application.

ARFF file

If the file name is left empty a temporary file will be created in the TEMP directory of the system. If one wants to specify an explicit results file, click on *Browse* and choose a filename, e.g., *Experiment1.arff*.

差 Save			×
		_	
Save In: 📑	weka-3-5-6		3 88 8-
			0.0. 0-
📑 changelog	s		
data			
data 📑 data			
auc			
File <u>N</u> ame:	Experiments1.arf		
Films of Tener	ARFF files		-
Files of <u>T</u> ype:	ARFF HIES	 	×
		Save	Cancel

Click on Save and the name will appear in the edit field next to ARFF file.

Weka Experiment Environment					
Setup Run Analyse					
Experiment Configuration Mode:	Simple	Advanced			
Open	Save	New			
Results Destination					
ARFF file Filename: C:(Temp)webs	-3-5-6\Experiments1.arff	Browse			
Experiment Type	Iteration Control				
Cross-validation	 Number of repetitions: 	: 10			
Number of folds: 10	Data sets first				
Classification Cassification	on 🔷 Algorithms first				
Datasets	Algorithms				
Add new Edit selecte	Delete select Add new	Edit selected Delete selected			
Use relative pat					
1					
Up	Down Load options	Save options Up Down			
	Notes				

The advantage of ARFF or CSV files is that they can be created without any additional classes besides the ones from Weka. The drawback is the lack of the ability to resume an experiment that was interrupted, e.g., due to an error or the addition of dataset or algorithms. Especially with time-consuming experiments, this behavior can be annoying.

JDBC database

With JDBC it is easy to store the results in a database. The necessary jar archives have to be in the CLASSPATH to make the JDBC functionality of a particular database available.

After changing ARFF file to JDBC database click on User... to specify JDBC URL and user credentials for accessing the database.

Database Connection Parameters						
Database UR	b jdbc:mysql://localhost:3306/weka_test					
∐sername						
<u>P</u> assword						
Debug						
	QK Cancel					

After supplying the necessary data and clicking on OK, the URL in the main window will be updated.

Note: at this point, the database connection is not tested; this is done when the experiment is started.

🛓 Weka Experiment Environment			
Setup Run Analyse			
experiment Configuration Mode:	Simple	Advanced	
Open	Save	New	
Results Destination			
JDBC database 🔽 URL: jdbc:mysql://localhos	t:3306/weka_test	Usi	er
Experiment Type	Iteration Con	ontrol	
Cross-validation	 Number of r 	repetitions: 10	
Number of folds: 10	Data set		
Classification Classification	n O Algorith	hms first	
Datasets	Algorithms		
Add new Delet	e selected Add new	Edit selected Delete selected	ed
Use relative paths			
		1	
	Lo	oad options Save options	
	Notes		

The advantage of a JDBC database is the possibility to resume an interrupted or extended experiment. Instead of re-running all the other algorithm/dataset combinations again, only the missing ones are computed.

2.1.3 Experiment type

The user can choose between the following three different types

• Cross-validation (default)

performs stratified cross-validation with the given number of folds

• Train/Test Percentage Split (data randomized) splits a dataset according to the given percentage into a train and a test file (one cannot specify explicit training and test files in the Experimenter), after the order of the data has been randomized and stratified

Setup Run Analyse				
speriment Configuration Mode:		Simple	_ <u>A</u> dvar	nced
Open	5	ave	New	
Results Destination				
ARFF file Filename: c:ttemptweka	a-3-5-6\Experiments1.a	rff		Browse
Experiment Type		Iteration Control		
Train/Test Percentage Split (data randomize	ed) 🔻	Number of repetitions:	10	
Train percentage: 66.0		Data sets first		
Classification Classification		Algorithms first		
Datasets		Algorithms		
Add new Edit selecte	Delete select	Add new	Edit selected	Delete selected
Use relative pat				
	-			
Up	Down	Load options	Save options	Up Down

• Train/Test Percentage Split (order preserved)

because it is impossible to specify an explicit train/test files pair, one can *abuse* this type to *un-merge* previously merged train and test file into the two original files (one only needs to find out the correct percentage)

Setup F	Run Analy	se				
periment	Configuration	n Mode:		Simple	⊖ <u>A</u> dva	nced
	Ope	n	5	Save		New
Results De	stination					
ARFF file	▼ Fil	lename: c:\temp\w	/eka-3-5-6\Experiments1.a	rff		Browse
xperimen	it Type			Iteration Control		
frain/Test	t Percentage	Split (order prese	rved) 🔻	Number of repetitions:		
rain perc	entage: 66.0			Data sets first		
Classif	fication	🔾 Regi	ression	Algorithms first		
Datasets			Algorithms			
Add	new	Edit selecte	Delete select	Add new	Edit selected	Delete selected
🗌 Use re	elative pat					
				1		
	Up		Down	Load options	Save options	Up Down

Additionally, one can choose between *Classification* and *Regression*, depending on the datasets and classifiers one uses. For decision trees like J48 (Weka's implementation of Quinlan's C4.5 [3]) and the iris dataset, *Classification* is necessary, for a numeric classifier like M5P, on the other hand, *Regression*. *Classification* is selected by default.

Note: if the percentage splits are used, one has to make sure that the corrected paired T-Tester still produces sensible results with the given ratio [2].

2.1.4 Datasets

One can add dataset files either with an absolute path or with a relative one. The latter makes it often easier to run experiments on different machines, hence one should check *Use relative paths*, before clicking on *Add new....*

🛃 Open		×
Look <u>i</u> n:	🔄 weka-3-5-6	- a 2 3 8 5
📑 chan	gelogs	
📑 data		
📑 doc		
File <u>N</u> ame		
Files of <u>T</u>	ype: Arff data files (*.arff)	
		Open Cancel

In this example, open the *data* directory and choose the *iris.arff* dataset.

Open			×		
Look <u>i</u> n: 📑 d	lata		- A A - 85		
🗋 contact-lei	nses.arff	🗋 søybean.arff			
🗋 cpu.arff		🗋 weather.arff			
Cpu.with.v	endor.arff	🗋 weather.nominal.arff			
🗋 iris.arff					
🗋 labor.arff					
Segment-c	:hallenge.ar	ff			
Segment-t	est.arff				
File Name: iris.arff					
Files of <u>T</u> ype:	Arff data fi	-			
			Open Cancel		

After clicking *Open* the file will be displayed in the datasets list. If one selects a directory and hits *Open*, then all ARFF files will be added recursively. Files can be deleted from the list by selecting them and then clicking on *Delete selected*.

ARFF files are not the only format one can load, but *all* files that can be converted with Weka's *"core converters"*. The following formats are currently supported:

• ARFF (+ compressed)

- C4.5
- $\bullet~\mathrm{CSV}$
- \bullet libsvm
- binary serialized instances
- XRFF (+ compressed)

By default, the class attribute is assumed to be the last attribute. But if a data format contains information about the class attribute, like XRFF or C4.5, this attribute will be used instead.

🍰 Weka Experiment Environment				<u> </u>
Setup Run Analyse				
Experiment Configuration Mode:		Simple	_ <u>A</u> dvan	nced
Open	5	jave		New
Results Destination				
ARFF file Filename: C:\Temp\weka	a-3-5-6\Experiments1.	arff		Browse
Experiment Type		Iteration Control		
Cross-validation	-	Number of repetitions:	10	
Number of folds: 10		Data sets first		
Classification Classification	on	 Algorithms first 		
Datasets		Algorithms		
Add new Edit selecte	Delete select	Add new	Edit selected	Delete selected
✓ Use relative pat				
.\data\iris.arff				
Ир	Down	Load options	Save options	Up Down
	N	otes		

2.1.5 Iteration control

• Number of repetitions

In order to get statistically meaningful results, the default number of iterations is 10. In case of 10-fold cross-validation this means 100 calls of one classifier with training data and tested against test data.

• Data sets first/Algorithms first

As soon as one has more than one dataset and algorithm, it can be useful to switch from datasets being iterated over first to algorithms. This is the case if one stores the results in a database and wants to complete the results for all the datasets for one algorithm as early as possible.

2.1.6 Algorithms

New algorithms can be added via the *Add new...* button. Opening this dialog for the first time, ZeroR is presented, otherwise the one that was selected last.

weka.gui.GenericObjectEditor						
Choose weka.classifiers.rules.ZeroR						
About						
Class for building and using a 0-R classifier. More						
	Capabilities					
debug False	-					
Open Save OK	Cancel					

With the $Choose\ {\rm button}\ {\rm one}\ {\rm can}\ {\rm open}\ {\rm the}\ GenericObjectEditor\ {\rm and}\ {\rm choose}\ {\rm another}\ {\rm classifier}.$

kweka.gui.GenericObjectEditor	_ 🗆 ×
weka classifiers bayes functions ilazy meta mi trees rules ConjunctiveRule DecisionTable JRip M6Rules NNge OneR PART Prism Ridor Zerore Filter Remove filter Close	Capabilities

The *Filter*... button enables one to highlight classifiers that can handle certain attribute and class types. With the *Remove filter* button all the selected capabilities will get cleared and the highlighting removed again.

Additional algorithms can be added again with the $Add\ new...$ button, e.g., the J48 decision tree.

駦 weka.gui.Gener	icObjectEditor		_ 🗆 ×
Choose weka.cl	assifiers.trees.J4	18	
About			
Class for generating	a pruned or unpru	ned C4.	More Capabilities
binarySplits	False		-
confidenceFactor	0.25		
debug	False		•
minNumObj	2		
numFolds	3		
reducedErrorPruning	False		-
saveInstanceData	False		-
seed	1		
subtreeRaising	True		•
unpruned	False		-
useLaplace	False		•
Open	Save	ОК	Cancel

After setting the classifier parameters, one clicks on ${\cal O}K$ to add it to the list of algorithms.

Setup Run Analyse			
xperiment Configuration Mode:		Simple	O <u>A</u> dvanced
Open	S	ave	New
Results Destination ARFF file Filename: C:\Temp\we	ka-3-5-6\Experiments1.a	arfí	Browse
Experiment Type Cross-validation Number of folds: 10		Iteration Control Number of repetitions:	10
Classification Cassification Cassification		Algorithms first Algorithms	
Add new Edit selecte	Delete select	Add new ZeroR J48 -C 0.25 -M 2	Edit selected Delete selected
Up	Down	Load options	Save options Up Down
Up	DOM	Load options	Save uptions Up Down

With the *Load options*... and *Save options*... buttons one can load and save the setup of a selected classifier from and to XML. This is especially useful for highly configured classifiers (e.g., nested meta-classifiers), where the manual setup takes quite some time, and which are used often.

One can also paste classifier settings here by right-clicking (or *Alt-Shift-left-clicking*) and selecting the appropriate menu point from the popup menu, to either add a new classifier or replace the selected one with a new setup. This is rather useful for transferring a classifier setup from the Weka Explorer over to the Experimenter without having to setup the classifier from scratch.

2.1.7 Saving the setup

For future re-use, one can save the current setup of the experiment to a file by clicking on *Save...* at the top of the window.

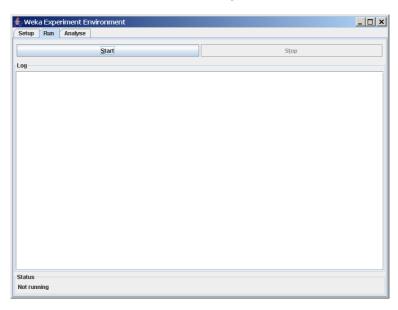
≜ Save		×
Save in: 📑	weka-3-5-6	• 6 6 6 8 5
📑 changelog	ıs	
📑 data		
📑 doc		
File <u>N</u> ame:	<u> </u>	
Files of <u>T</u> ype:	Experiment configuration files (*.exp)	-
		Save Cancel

By default, the format of the experiment files is the binary format that Java serialization offers. The drawback of this format is the possible incompatibility between different versions of Weka. A more robust alternative to the binary format is the XML format.

Previously saved experiments can be loaded again via the Open... button.

2.1.8 Running an Experiment

To run the current experiment, click the Run tab at the top of the Experiment Environment window. The current experiment performs 10 runs of 10-fold stratified cross-validation on the Iris dataset using the ZeroR and J48 scheme.



Click *Start* to run the experiment.

Setup Run Analyse			
<u>S</u>	art	Stop	
.og			
16:17:12: Started 16:17:12: Finished 16:17:12: There were 0 errors			
Status			

If the experiment was defined correctly, the 3 messages shown above will be displayed in the *Log* panel. The results of the experiment are saved to the dataset *Experiment1.arff*.

2.2 Advanced

2.2.1 Defining an Experiment

When the Experimenter is started in Advanced mode, the Setup tab is displayed. Click New to initialize an experiment. This causes default parameters to be defined for the experiment.

Setup Run A	nalyse			
periment Configura	ation Mode:		○ Simple	Advanced
	Open		Save	New
Destination				
Choose Insta	ncesResultListener -	O weka_experiment2561	9.arff	
tesult generator				
Choose Rand	omSplitResultProduc	er -P 66.0 -O splitEvaluto	orOut.zip -W weka.experimer	nt.ClassifierSplitEvaluatorW weka.class
Runs		experiment	Generator properties	
From: 1 To:		Hosts	Disabled	Select property
	By data	set 🔿 By run		
eration control	⊖ Cus	stom generator first		
atasets				
Add new	Edit selecte	Delete select		
Use relative pa	L			Can't edit

To define the dataset to be processed by a scheme, first select $Use\ relative\ paths$ in the Datasets panel of the Setup tab and then click on $Add\ new...$ to open a dialog window.

실 Open				×
Look <u>i</u> n: 📑	weka-3-5-6	-	a û o	D-D: D D-D: D
Changelog	IS			
File <u>N</u> ame: Files of <u>T</u> ype:	data Arff data files (*.arff)			
			Open	Cancel

Double click on the data folder to view the available datasets or navigate to an alternate location. Select *iris.arff* and click *Open* to select the Iris dataset.

	擔 Open			×	
	Look <u>i</u> n:] data	~	A C 288	
	Contact-I	lenses.arff 🛛 🗋 wea	ther.arff		
	🗋 cpu.arff	🗋 weat	ther.nominal.arff		
	iris.arff				
	labor.arfi				
	segment	-challenge.arff test arff			
	soybean.				
	File <u>N</u> ame:	iris.arff			
	Files of Type:	Arff data files (*.arff)	1	•	
				Open Cancel	
				Culter	
Weka Exp	periment Environm	nent			×
Setup Run	Analyse				
xperiment Con	nfiguration Mode:		○ <u>S</u> imple	Advanced	
	Open		Save	New	
N = = 41 = = 1					
vestination					
	InstancesResultListe	ner - O weka_experiment2	5619.arff		
Destination Choose Result generat		ner -O weka_experiment2	5619.arff		
Choose Result generat	tor			ent.ClassifierSplitEvaluatorW wek	a.classifier
Choose Result generat	tor RandomSplitResultPr				a.classifier
Choose Result general Choose	tor RandomSplitResultPr	oducer -P 66.0 -O splitEva	IlutorOut.zip -W weka.experime	s	
Choose Result generat Choose Runs	tor RandomSplitResultPr Distrit To: 10	oducer -P 66.0 -O splitEva oute experiment	ilutorOut.zip -W weka.experim		
Choose Result generat Choose Runs From: 1	tor RandomSplitResultPr To: 10 By	oducer -P 66.0 -O splitEva oute experiment Hosts	IlutorOut.zip -W weka.experime	s	
Choose Result generat Choose Runs From: 1 Iteration contr @ Data sets f	tor RandomSplitResultPr To: 10 By rol	oducer -P 66.0 -O splitEva oute experiment Hosts	IlutorOut.zip -W weka.experime	s	
Choose Result general Choose Runs From: 1 Iteration contr	tor RandomSplitResultPr To: 10 By rol	oducer -P 66.0 - O splitEva pute experiment Hosts data set O By run	IlutorOut.zip -W weka.experime	s	
Choose Result generat Choose Runs From: 1 Iteration contr @ Data sets f	tor RandomSplitResultPr To: 10 0 0 000000000000000000000000000000	aducer - P 66.0 - O splitEva pute experiment Hosts data set O By run C ustom generator first	IlutorOut.zlp -W weka experimine Generator properties	s	
Choose Result generat Choose Runs From: 1 Iteration contr © Data sets f Datasets	tor RandomSplitResultPr To: 10 By rol w Edit sele	aducer - P 66.0 - O splitEva pute experiment Hosts data set O By run C ustom generator first	IlutorOut.zlp -W weka experimine Generator properties	s	
Choose Result generat Choose Runs From: 1 Iteration contr © Data sets f Datasets Add new	tor RandomSplitResultPr To: 10 By rol w Edit sele	aducer - P 66.0 - O splitEva pute experiment Hosts data set O By run C ustom generator first	IlutorOut.zlp -W weka experimine Generator properties	Select property	
Choose Result generat Choose Runs From: 1 Patasion contr © Data sets 1 Datasets Add nev V Use relath Vdata\iris.arff	tor RandomSplitResultPr To: 10 By rol w Edit sele	aducer - P 66.0 - O splitEva pute experiment Hosts data set O By run C ustom generator first	IlutorOut.zlp -W weka experimine Generator properties	Select property	
Choose Result generat Choose Runs From: 1 Choose Runs Uteration contr Data sets 1 Data sets 1 Data sets 1 Use relath vdata\u00edris.arff	tor RandomSplitResultPr To: 10 Distrit @ By rol first C w Edit sele ve pat	aducer -P 66.0 -O splitEva ute experiment Hosts data set O By run) Custom generator first cte Delete select	IlutorOut.zlp -W weka experimine Generator properties	Select property	

The dataset name is now displayed in the *Datasets* panel of the *Setup* tab.

Saving the Results of the Experiment

To identify a dataset to which the results are to be sent, click on the *Instances-ResultListener* entry in the *Destination* panel. The output file parameter is near the bottom of the window, beside the text *outputFile*. Click on this parameter to display a file selection window.

差 weka.gui.GenericObjectEditor	_ 🗆 🗙
weka.experiment.InstancesResultListener About	
Outputs the received results in arff format to a Writer.	More
outputFile weka_experiment25619.artf	
Open Save OK	Cancel

🛓 weka.gui.FileEditor		_ 🗆 ×
Look in: 🔲 weka-3-5-6	▼ A	
📑 changelogs	🗋 documentation.html	🗋 remoteExperim
🗂 data	🗋 Experimenter Tutorial.pdf	🗋 Tutorial.pdf
🗂 doc	🗋 ExplorerGuide.pdf	🗋 weka-src.jar
BayesianNetClassifiers.pdf	🗋 README	🗋 weka.gif
COPYING	README_Experiment_Gui	🗋 weka.ico
documentation.css	README_KnowledgeFlow	🗋 weka.jar
•		Þ
File Name: Experiment1.arf		
Files of Type: All Files		•
	Sel	ect Cancel

Type the name of the output file, click *Select*, and then click close (x). The file name is displayed in the *outputFile* panel. Click on *OK* to close the window.

🚔 weka.gui.GenericObjectEditor	_ 🗆 🗙
weka.experiment.InstancesResultListener About	
Takes results from a result producer and assembles them [into a set of instances.	More
outputFile Experiment1.arff	
Open Save OK	Cancel

The dataset name is displayed in the *Destination* panel of the *Setup* tab.

Setup Run Analyse	1			
periment Configuration Mc	ode:		○ <u>S</u> imple	<u>A</u> dvanced
Open		5	ave	New
estination				
Choose InstancesRe	sultListener - O Experin	nent1.arff		
esult generator				
Choose RandomSplit	ResultProducer -P 66.	0 - 0 splitEvalutorO	ut.zip -W weka.experimer	nt.ClassifierSplitEvaluatorW weka.class
Runs	Distribute experime	ent	Generator properties	
From: 1 To: 10	Ho	ists	Disabled	Select property
	By data set) By run		
teration control				
Data sets first	 Custom gene 	erator first		
Data sets first atasets	O Custom gen	erator first		
atasets		erator first Delete select		
atasets				Can't edit
Add new				Can't edit
Add new				Can't edit
Add new		elete select		Can't edit

Saving the Experiment Definition

The experiment definition can be saved at any time. Select *Save...* at the top of the *Setup* tab. Type the dataset name with the extension *exp* (or select the dataset name if the experiment definition dataset already exists) for binary files or choose *Experiment configuration files* (*.*xml*) from the file types combobox (the XML files are robust with respect to version changes).

🛓 Save		×
Save in: 📑	weka-3-5-6	• 6 6 6 8 5
📑 changelog	s	
📑 data		
doc 📑		
File <u>N</u> ame:	Experiment1.exp	
Files of <u>T</u> ype:	Experiment configuration files (*.exp)	-
		Save Cancel

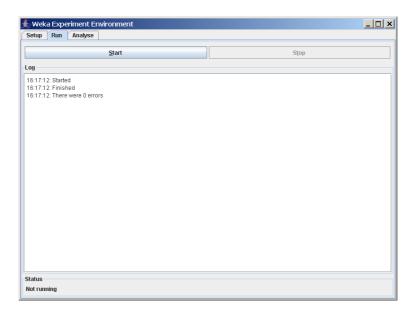
The experiment can be restored by selecting *Open* in the *Setup* tab and then selecting *Experiment1.exp* in the dialog window.

2.2.2 Running an Experiment

To run the current experiment, click the Run tab at the top of the Experiment Environment window. The current experiment performs 10 randomized train and test runs on the Iris dataset, using 66% of the patterns for training and 34% for testing, and using the ZeroR scheme.

🚖 Weka Experiment Environment	
Setup Run Analyse	
Start	Stop
Log	
Status	
Not running	

Click *Start* to run the experiment.



If the experiment was defined correctly, the 3 messages shown above will be displayed in the *Log* panel. The results of the experiment are saved to the dataset *Experiment1.arff*. The first few lines in this dataset are shown below.

```
@relation InstanceResultListener
```

```
@attribute Key_Dataset {iris}
@attribute Key_Run {1,2,3,4,5,6,7,8,9,10}
Cattribute Key_Scheme {weka.classifiers.rules.ZeroR,weka.classifiers.trees.J48}
@attribute Key_Scheme_options {,'-C 0.25 -M 2'}
@attribute Key_Scheme_version_ID {48055541465867954,-217733168393644444}
@attribute Date_time numeric
@attribute Number_of_training_instances numeric
@attribute Number_of_testing_instances numeric
@attribute Number_correct numeric
@attribute Number_incorrect numeric
Qattribute Number_unclassified numeric
@attribute Percent_correct numeric
@attribute Percent_incorrect numeric
@attribute Percent_unclassified numeric
@attribute Kappa_statistic numeric
@attribute Mean_absolute_error numeric
@attribute Root_mean_squared_error numeric
@attribute Relative_absolute_error numeric
@attribute Root_relative_squared_error numeric
@attribute SF_prior_entropy numeric
@attribute SF_scheme_entropy numeric
@attribute SF_entropy_gain numeric
@attribute SF_mean_prior_entropy numeric
@attribute SF_mean_scheme_entropy numeric
@attribute SF_mean_entropy_gain numeric
Qattribute KB_information numeric
```

```
@attribute KB_mean_information numeric
@attribute KB_relative_information numeric
Cattribute True_positive_rate numeric
@attribute Num_true_positives numeric
@attribute False_positive_rate numeric
@attribute Num_false_positives numeric
@attribute True_negative_rate numeric
@attribute Num_true_negatives numeric
Cattribute False_negative_rate numeric
Cattribute Num_false_negatives numeric
Cattribute IR_precision numeric
@attribute IR_recall numeric
Cattribute F_measure numeric
@attribute Area_under_ROC numeric
@attribute Time_training numeric
@attribute Time_testing numeric
Cattribute Summary {'Number of leaves: 3\nSize of the tree: 5\n',
   'Number of leaves: 5\nSize of the tree: 9\n',
   'Number of leaves: 4 \in 1^{n}
Qattribute measureTreeSize numeric
Qattribute measureNumLeaves numeric
Cattribute measureNumRules numeric
```

@data

```
iris,1,weka.classifiers.rules.ZeroR,,48055541465867954,20051221.033,99,51,
17,34,0,33.333333,66.6666667,0,0,0.444444,0.471405,100,100,80.833088,80.833088,
0,1.584963,1.584963,0,0,0,0,1,17,1,34,0,0,0,0,0.333333,1,0.5,0.5,0,0,?,?,?,?
```

2.2.3 Changing the Experiment Parameters

Changing the Classifier

The parameters of an experiment can be changed by clicking on the *Result* generator panel.

≜ weka.gui.G	enericObjectEdi	tor	_ 🗆 🗙			
weka.experiment	RandomSplitResu	tProducer				
About						
Performs a rand evaluator.	lom train and test us	sing a supplied	More			
outputFile	splitEvalutorOut.zip					
randomizeData	True		-			
rawOutput	False					
splitEvaluator	Choose Clas	sifierSplitEvaluator	-W weka.classifi			
trainPercent	66.0					
Open	Save	ОК	Cancel			

The *RandomSplitResultProducer* performs repeated train/test runs. The number of instances (expressed as a percentage) used for training is given in the

trainPercent box. (The number of runs is specified in the Runs panel in the Setup tab.)

A small help file can be displayed by clicking *More* in the *About* panel.

🚔 Information	_ 🗆 ×
NAME weka.experiment.RandomSplitResultProducer	
SYNOPSIS Performs a random train and test using a supplied evaluat	or.
OPTIONS outputFile Set the destination for saving raw output. If the option is selected, then output from the splitEvaluator for in train-test splits is saved. If the destination is a directory, the output is saved to an individual gzip file; if the destination is then each output is saved as an entry in a zip file.	dividual In each
randomizeData Do not randomize dataset and do not per probabilistic rounding if true	rform
rawOutput Save raw output (useful for debugging). If set, output is sent to the destination specified by outputFile	then
splitEvaluator The evaluator to apply to the test data. This classifier, regression scheme etc.	may be a
trainPercent Set the percentage of data to use for training	I.

Click on the *splitEvaluator* entry to display the *SplitEvaluator* properties.

👙 weka.gui.Gener	icObjectEdi	tor	<u>_ 🗆 ×</u>
weka.experiment.Clas	sifierSplitEval	uator	
About			
A SplitEvaluator that scheme on a nomina			on More
attributeID	-1		
classForIRStatistics	0		
classifier	Choose	ZeroR	
predTargetColumn	False		•
Open	Save	ок	Cancel

Click on the classifier entry (ZeroR) to display the scheme properties.

🚔 weka.gui.GenericObjectEditor	
weka.classifiers.rules.ZeroR About	
Class for building and using a 0-R classifier.	More Capabilities
debug False	•
Open Save OK	Cancel

This scheme has no modifiable properties (besides *debug* mode on/off) but most other schemes do have properties that can be modified by the user. The *Capabilities* button opens a small dialog listing all the attribute and class types this classifier can handle. Click on the *Choose* button to select a different scheme. The window below shows the parameters available for the J48 decisiontree scheme. If desired, modify the parameters and then click OK to close the window.

🌲 weka.gui.Generi	cObjectEditor		_ 🗆 ×
weka.classifiers.trees.	J48		
About			
Class for generating a	pruned or unpru	ned C4.	More
			Capabilities
binarySplits	False		-
confidenceFactor	0.25		
debug	False		•
minNumObj	2		
numFolds	3		
reducedErrorPruning	False		•
saveInstanceData	False		-
seed	1		
subtreeRaising	True		•
unpruned	False		-
useLaplace	False		-
Open	Save	ОК	Cancel

The name of the new scheme is displayed in the *Result generator* panel.

🔩 Weka E>	kperiment Er	nvironment			_ 0
Setup Ru	in Analyse]			
xperiment Co	onfiguration Mo	ode:		○ Simple	Advanced
	Open			<u>S</u> ave	New
Destination					
Choose	InstancesRe	sultListener - O Exp	eriment1.arff		
Result gener	rator				
Choose	RandomSplit	ResultProducer - P	66.0 -O splitEvalutor	Out.zip -VV weka.experiment	ClassifierSplitEvaluatorW weka.classifie
Runs		Distribute experi	iment	Generator properties	
From: 1	To: 10		Hosts	Disabled	Select property
		By data set	🔿 By run		
 Iteration con Data sets 		O Custom r	enerator first		
Datasets		0 0000000	,		
Add n	010	Edit selecte	Delete select	1	
✓ Use rela		Luit Sciecte	Delete Select		Can't edit
.\data\iris.arf				-	oun coun
.wata in is.an					
	Up		Down		
				Notes	
				10(03	

Adding Additional Schemes

Additional schemes can be added in the *Generator properties* panel. To begin, change the drop-down list entry from *Disabled* to *Enabled* in the *Generator properties* panel.

Setup Run	Analyse					
kperiment Confi	guration Mod	le:		○ Simple		Advanced
	Open			<u>S</u> ave		New
Destination						
Choose In	stancesRes	ultListener - O Exp	eriment1.arff			
Result generato	r					
Choose R	andomSplitR	tesultProducer - P	66.0 - O splitEvalutor	Out.zip -₩ weka.experime	nt.ClassifierS	SplitEvaluatorW weka.classifie
Runs		Distribute exper	iment	Generator properties		
From: 1 T	f o: 10		Hosts	Disabled	-	Select property
		By data set	⊖ By run	Disabled		
Iteration control Iteration control Iteration control		0.0	generator first	Enabled		
Data sets first Datasets	st	⊖ Custom	generator tirst			
Add new.		dit selecte	Delete select			
🖌 Use relative	pat				Can't	edit
\data\iris.arff						
uata ins.am						
wata wis.arii						
uatauns.ann			Down			

Click *Select property* and expand *splitEvaluator* so that the *classifier* entry is visible in the property list; click *Select*.

≜ Select a property 🛛 🗙
Available properties Control outputFile Control rawOutput Control outputFile Control output Control outpu
Select Cancel

The scheme name is displayed in the ${\it Generator\ properties\ panel}.$

Setup Ru	n Analyse						
operiment C	onfiguration Mo	de:		○ <u>S</u> imple		Advanced	
	Open			<u>S</u> ave		<u>N</u> ew	
Destination							
Choose	InstancesRes	ultListener - O Exp	eriment1.arff				
Result gener	ator						
Choose	RandomSplit	ResultProducer -P	66.0 - OsplitEvalutor	Out.zip -₩ weka.expe	riment.Classifi	erSplitEvaluator\	/V weka.classifie
Runs		Distribute experi	iment	Generator proper	rties		
From: 1	To: 10		Hosts	Enabled	•	Select proj	perty
		By data set	⊖ By run				10
	4			Chooco Zor	roD		ñdd
		Custom g	enerator first	Choose Zer ZeroR	roR		Add
Data sets		🔾 Custom g	jenerator first		roR		Add
Iteration cor Data sets Datasets Add n	s first	Custom g	penerator first		roR		Add
Data sets	ew		-		roR		Add
● Data sets Datasets Add n ⊻ Use rela	ew		-		roR		Add
Data sets Datasets Add n	ew		-		roR		Add
● Data sets Datasets Add n ✔ Use rela	ew		-		roR		Add

To add another scheme, click on the Choose button to display the Generic-ObjectEditor window.

🐇 Weka Exp	periment Er	nvironment						
Setup Run	Analyse							
xperiment Configuration Mode:		C) <u>S</u> imple		Advanced			
	Open		<u>S</u> av	e		New		
Destination								
Choose	InstancesResultListener - 0 Experiment1.arm		t1.arff					
Result genera	tor							
Choose	RandomSplit	ResultProducer -P 66.0 -C) splitEvalutorOut.z	tip -₩ weka.expe	riment.Classifier8	SplitEvaluatorW weka.cla		
Runs		Distribute experiment		Generator prope	rties			
From: 1	To: 10	Hosts By data set	by run	Enabled	-	Select property		
Add new ✓ Use relati vdata¥ris.arff		Edit selecte Dele	te select	- 🗋 BF	8	Down		
				- 0 M ⁶ - 0 NE - 0 R - 0 R R	5P 3Tree andomForest andomTree EPTree mpleCart	tter <u>C</u> lose		

The *Filter...* button enables one to highlight classifiers that can handle certain attribute and class types. With the *Remove filter* button all the selected capabilities will get cleared and the highlighting removed again.

To change to a decision-tree scheme, select J48 (in subgroup *trees*).

🌲 weka.gui.Generi	ObjectEditor		_ 🗆 ×	
weka.classifiers.trees.	J48			
About				
Class for generating a	Class for generating a pruned or unpruned C4. More			
			Capabilities	
binarySplits	False		-	
confidenceFactor	0.25			
debug	False		-	
minNumObj	2			
numFolds	3			
reducedErrorPruning	False		•	
saveInstanceData	False		-	
seed	1			
subtreeRaising	True		-	
unpruned	False		-	
useLaplace	False		-	
Open	Save	ок	Cancel	

The new scheme is added to the $Generator\ properties\ panel.$ Click Add to add the new scheme.

😓 Weka Experiment	: Environment						_ 🗆
Setup Run Analys	se						
kperiment Configuration	Mode:		○ <u>S</u> imple			Advanced	
Ope	n		Save			New	
Destination							
Choose Instances	ResultListener - O Exp	eriment1.arff					
Result generator							
Choose RandomS	plitResultProducer - P	66.0 - OsplitEvaluto	Out.zip -₩ weka.e	xperiment.	ClassifierS	plitEvaluatorV	V weka.classifi
Runs	Distribute exper	iment	Generator pr	operties			
From: 1 To: 10		Hosts	Enabled		.	Select prop	nertv
	By data set	⊖ By run		1			-
teration control			Choose	J48 - C 0.2	25 - M 2		Add
Data sets first	🔾 Custom (jenerator first	ZeroR J48 -C 0.25 -	M 2			
atasets							
Add new	Edit selecte	Delete select					
✓ Use relative pat			_				
data'iris.arff							
Up		Down	Delete		Edit	Up	Down
			Notes				
			NULES				

Now when the experiment is run, results are generated for both schemes. To add additional schemes, repeat this process. To remove a scheme, select the scheme by clicking on it and then click *Delete*.

Adding Additional Datasets

The scheme(s) may be run on any number of datasets at a time. Additional datasets are added by clicking *Add new...* in the *Datasets* panel. Datasets are deleted from the experiment by selecting the dataset and then clicking *Delete Selected*.

Raw Output

The raw output generated by a scheme during an experiment can be saved to a file and then examined at a later time. Open the *ResultProducer* window by clicking on the *Result generator* panel in the *Setup* tab.

🔩 weka.gui.GenericObjectEditor 📃 🗖 🗙					
weka.experimen	t.RandomSplitResultProducer				
About					
Performs a rand evaluator.	Performs a random train and test using a supplied More				
outputFile	splitEvalutorOut.zip				
randomizeData	a True				
rawOutput	True				
splitEvaluator	Choose ClassifierSplitEvaluator -W weka.classifi				
trainPercent	66.0				
Open	Save OK Cancel				

Click on *rawOutput* and select the *True* entry from the drop-down list. By default, the output is sent to the zip file *splitEvaluatorOut.zip*. The output file can be changed by clicking on the *outputFile* panel in the window. Now when the experiment is run, the result of each processing run is archived, as shown below.

x 1		1.1 122 1	[m . c . [B 4
Name	Size	Modified	Ratio	Packed	Path \triangle
rules.ZeroR(version_48055541465867954)	568	21/12/2005 16:	55%	257	1.iris.ClassifierSplitEvaluator:
_trees.J48C_0.25M_2(version217733168393644444)	844	21/12/2005 16:	53%	397	1.iris.ClassifierSplitEvaluator:
rules.ZeroR(version_48055541465867954)	568	21/12/2005 16:	55%	257	10.iris.ClassifierSplitEvaluator:
trees.J48C_0.25M_2(version217733168393644444)	915	21/12/2005 16:	54%	417	10.iris.ClassifierSplitEvaluator:
rules.ZeroR(version_48055541465867954)	568	21/12/2005 16:	55%	257	2.iris.ClassifierSplitEvaluator:
_trees.J48C_0.25M_2(version217733168393644444)	1,001	21/12/2005 16:	58%	425	2.iris.ClassifierSplitEvaluator:
rules.ZeroR(version_48055541465867954)	568	21/12/2005 16:	55%	257	3.iris.ClassifierSplitEvaluator:
_trees.J48C_0.25M_2(version217733168393644444)	844	21/12/2005 16:	53%	395	3.iris.ClassifierSplitEvaluator:
rules.ZeroR(version_48055541465867954)	568	21/12/2005 16:	55%	257	4.iris.ClassifierSplitEvaluator:
_trees.J48C_0.25M_2(version217733168393644444)	997	21/12/2005 16:	57%	433	4.iris.ClassifierSplitEvaluator:
rules.ZeroR(version_48055541465867954)	568	21/12/2005 16:	55%	257	5.iris.ClassifierSplitEvaluator:
_trees.J48C_0.25M_2(version217733168393644444)	919	21/12/2005 16:	55%	414	5.iris.ClassifierSplitEvaluator:
rules.ZeroR(version_48055541465867954)	568	21/12/2005 16:	55%	257	6.iris.ClassifierSplitEvaluator:
_trees.J48C_0.25M_2(version217733168393644444)	1,001	21/12/2005 16:	57%	427	6.iris.ClassifierSplitEvaluator:
rules.ZeroR(version_48055541465867954)	568	21/12/2005 16:	55%	257	7.iris.ClassifierSplitEvaluator:
_trees.J48C_0.25M_2(version217733168393644444)	844	21/12/2005 16:	54%	391	7.iris.ClassifierSplitEvaluator:
rules.ZeroR(version_48055541465867954)	568	21/12/2005 16:	55%	257	8.iris.ClassifierSplitEvaluator:
_trees.J48C_0.25M_2(version217733168393644444)	923	21/12/2005 16:	55%	414	8.iris.ClassifierSplitEvaluator:
rules.ZeroR(version_48055541465867954)	568	21/12/2005 16:	55%	257	9.iris.ClassifierSplitEvaluator:
_trees.J48C_0.25M_2(version217733168393644444)	907	21/12/2005 16:	55%	408	9.iris.ClassifierSplitEvaluator:

The contents of the first run are:

ClassifierSplitEvaluator: weka.classifiers.trees.J48 -C 0.25 -M 2(version -217733168393644444)Classifier model:

```
J48 pruned tree
```

```
petalwidth <= 0.6: Iris-setosa (33.0)
petalwidth > 0.6
| petalwidth <= 1.5: Iris-versicolor (31.0/1.0)
| petalwidth > 1.5: Iris-virginica (35.0/3.0)
Number of Leaves : 3
Size of the tree : 5
```

Correctly Classified Instances	47	92.1569 %
Incorrectly Classified Instances	4	7.8431 %
Kappa statistic	0.8824	
Mean absolute error	0.0723	
Root mean squared error	0.2191	
Relative absolute error	16.2754 %	
Root relative squared error	46.4676 %	
Total Number of Instances	51	
measureTreeSize : 5.0		
measureNumLeaves : 3.0		
measureNumRules : 3.0		

2.2.4 Other Result Producers

Cross-Validation Result Producer

To change from random train and test experiments to cross-validation experiments, click on the *Result generator* entry. At the top of the window, click on the drop-down list and select *CrossValidationResultProducer*. The window now contains parameters specific to cross-validation such as the number of partitions/folds. The experiment performs 10-fold cross-validation instead of train and test in the given example.

📥 weka.gui.(GenericObjectEditor
weka.experime	nt.CrossValidationResultProducer
About	
Performs a cro	ss validation run using a supplied evaluator. More
numFolds	10
outputFile	splitEvalutorOut.zip
rawOutput	False
splitEvaluator	Choose ClassifierSplitEvaluator -W weka.classifier
Open	Save OK Cancel

The *Result generator* panel now indicates that cross-validation will be performed. Click on *More* to generate a brief description of the *CrossValidation-ResultProducer*.

🚖 Information 📃 🗆 🗙
NAME weka.experiment.CrossValidationResultProducer
SYNOPSIS Performs a cross validation run using a supplied evaluator.
OPTIONS numFolds Number of folds to use in cross validation.
outputFile Set the destination for saving raw output. If the rawOutput option is selected, then output from the splitEvaluator for individual folds is saved. If the destination is a directory, then each output is saved to an individual gzip file; if the destination is a file, then each output is saved as an entry in a zip file.
rawOutput Save raw output (useful for debugging). If set, then output is sent to the destination specified by outputFile
splitEvaluator The evaluator to apply to the cross validation folds. This may be a classifier, regression scheme etc.

As with the *RandomSplitResultProducer*, multiple schemes can be run during cross-validation by adding them to the *Generator properties* panel.

	Experimen tun Analy	t Environment					
	configuratio			○ <u>S</u> imple		<u>A</u> dvanced	
	Ope	m		<u>S</u> ave		New	
Destination	ı						
Choose	Instance	sResultListener - O Exp	eriment1.arff				
Result gen	erator						
Choose	CrossVa	lidationResultProducer	-X10-O splitEvalute	orOut.zip -W weka.exp	eriment.Classifie	rSplitEvaluator	-W weka.class
Runs		Distribute exper	iment	Generator prope	erties		
From: 1	To: 1		Hosts	Enabled	•	Select prop	perty
		By data set	O By run				-
teration co				Choose J4 ZeroR	8 -C 0.25 -M 2		Ade
Data se atasets	as first	Ustom g	jenerator first	OneR -B 6			
				J48 -C 0.25 -M 2			
	new	Edit selecte	Delete select				
	lative pat			_			
data'iris.a	rff						
	Up		Down				
		[Delete	Edit	Up	Down
				Notes			

The number of runs is set to 1 in the *Setup* tab in this example, so that only one run of cross-validation for each scheme and dataset is executed.

When this experiment is analysed, the following results are generated. Note that there are 30 (1 run times 10 folds times 3 schemes) result lines processed.

≜ Weka Experimen	t Environment	
Setup Run Analy	se	
Source Got 30 results		File Database Experiment
Configure test		1 Test output
Testing <u>w</u> ith	Paired T-Tester (cor 💌	Tester: weka.experiment.PairedCorrectedTTester Analysing: Percent_correct
Row	Select	Datasets: 1 Resultsets: 3
<u>C</u> olumn	Select	Confidence: 0.05 (two tailed) Sorted by: - Date: 21/12/05 16:47
_	Percent_correct	
Significance Sorting (asc.) by	<pre>cdefault></pre>	Dataset (1) rules.Ze (2) rules (3) trees
Test <u>b</u> ase	Select	(10) 33.33 94.00 v 96.00 v (10) 33.33 94.00 v 96.00 v (10) (10) (10) (10) (10) (10) (10) (10)
Displayed Columns	Columns	
Show std. deviations		Key: (1) rules.ZeroR '' 48055541465867954 (2) rules.OneR '-B 6' -2459427002147861445
<u>O</u> utput Format	Select	(2) rules.unex '-B b' -245942/UU214/sb1445 (3) trees.J48 '-C 0.25 -M 2' -217733168393644444
Perform <u>t</u> est Result list	Save output	
16:47:17 - Percent_corr	rect - rules.ZeroR " 4805 💺	

Averaging Result Producer

An alternative to the CrossValidationResultProducer is the AveragingResultProducer. This result producer takes the average of a set of runs (which are typically cross-validation runs). This result producer is identified by clicking the Result generator panel and then choosing the AveragingResultProducer from the GenericObjectEditor.

🚖 weka.gui.GenericObjec	tEditor
weka.experiment.AveragingRe	sultProducer
Takes the results from a Resu average to the result listener.	ultProducer and submits the More
calculateStdDevs	False 💌
expectedResultsPerAverage	10
keyFieldName	Fold
resultProducer	Choose CrossValidationResultProc
Open Save	OK Cancel

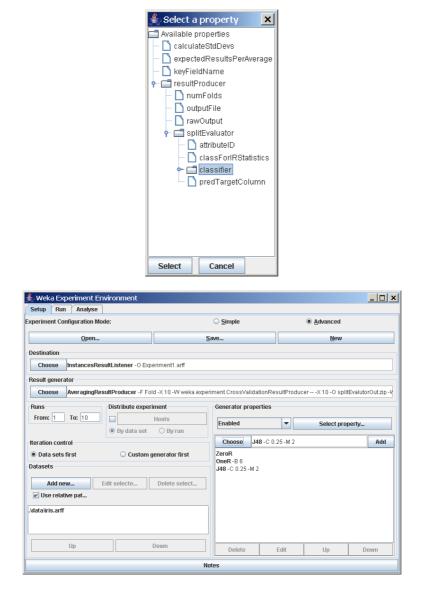
The associated help file is shown below.

🚖 Information
NAME weka.experiment.AveragingResultProducer
SYNOPSIS Takes the results from a ResultProducer and submits the average to the result listener. Normally used with a CrossValidationResultProducer to perform n x m fold cross validation.
OPTIONS calculateStdDevs Record standard deviations for each run.
expectedResultsPerAverage Set the expected number of results to average per run. For example if a CrossValidationResultProducer is being used (with the number of folds set to 10), then the expected number of results per run is 10.
keyFieldName Set the field name that will be unique for a run.
resultProducer Set the resultProducer for which results are to be averaged.

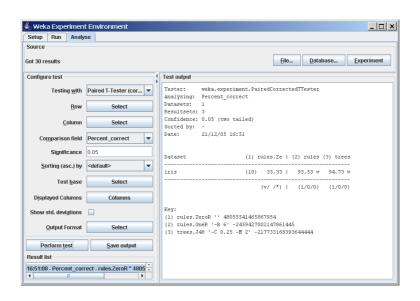
Clicking the *resultProducer* panel brings up the following window.

weka.gui.(GenericObjectEditor
weka.experime About	nt.CrossValidationResultProducer
Performs a cro	oss validation run using a supplied evaluator. More
numFolds	10
outputFile	splitEvalutorOut.zip
rawOutput	False
splitEvaluator	Choose Classifier SplitEvaluator -W weka.classifier
Open	Save OK Cancel

As with the other ResultProducers, additional schemes can be defined. When the *AveragingResultProducer* is used, the classifier property is located deeper in the *Generator properties* hierarchy.



In this experiment, the ZeroR, OneR, and J48 schemes are run 10 times with 10-fold cross-validation. Each set of 10 cross-validation folds is then averaged, producing one result line for each run (instead of one result line for each fold as in the previous example using the *CrossValidationResultProducer*) for a total of 30 result lines. If the raw output is saved, all 300 results are sent to the archive.



3 Remote Experiments

Remote experiments enable you to distribute the computing load across multiple computers. In the following we will discuss the setup and operation for HSQLDB [5] and MySQL [6].

3.1 Preparation

To run a remote experiment you will need:

- A database server.
- A number of computers to run remote engines on.
- To edit the remote engine policy file included in the Weka distribution to allow Java class and dataset loading from your home directory.
- An invocation of the Experimenter on a machine somewhere (any will do).

For the following examples, we assume a user called *johndoe* with this setup:

- Access to a set of computers running a flavour of Unix (pathnames need to be changed for Windows).
- The home directory is located at /home/johndoe.
- Weka is found in /home/johndoe/weka.
- Additional jar archives, i.e., JDBC drivers, are stored in /home/johndoe/jars.
- The directory for the datasets is /home/johndoe/datasets.

Note: The example policy file remote.policy.example is using this setup (available in weka/experiment¹).

3.2 Database Server Setup

- HSQLDB
 - Download the JDBC driver for HSQLDB, extract the hsqldb.jar and place it in the directory /home/johndoe/jars.
 - To set up the database server, choose or create a directory to run the database server from, and start the server with:

```
java -classpath /home/johndoe/jars/hsqldb.jar \
    org.hsqldb.Server \
    -database.0 experiment -dbname.0 experiment
```

Note: This will start up a database with the alias "experiment" (-dbname.0 <alias>) and create a properties and a log file at the current location prefixed with "experiment" (-database.0 <file>).

¹Weka's source code can be found in the weka-src.jar archive or obtained from CVS [4].

• MySQL

We won't go into the details of setting up a MySQL server, but this is rather straightforward and includes the following steps:

- Download a suitable version of MySQL for your server machine.
- Install and start the MySQL server.
- Create a database for our example we will use experiment as database name.
- Download the appropriate JDBC driver, extract the JDBC jar and place it as mysql.jar in /home/johndoe/jars.

3.3 Remote Engine Setup

• First, set up a directory for scripts and policy files:

/home/johndoe/remote_engine

- Unzip the remoteExperimentServer.jar (from the Weka distribution; or build it from the sources² with ant remotejar) into a temporary directory.
- Next, copy remoteEngine.jar and remote.policy.example to the /home/johndoe/remote_engine directory.
- Create a script, called /home/johndoe/remote_engine/startRemoteEngine, with the following content (don't forget to make it executable with chmod a+x startRemoteEngine when you are on Linux/Unix):
 - HSQLDB

```
java -Xmx256m \
  -classpath /home/johndoe/jars/hsqldb.jar:remoteEngine.jar \
  -Djava.security.policy=remote.policy \
  weka.experiment.RemoteEngine &
```

- MySQL

```
java -Xmx256m \
  -classpath /home/johndoe/jars/mysql.jar:remoteEngine.jar \
  -Djava.security.policy=remote.policy \
  weka.experiment.RemoteEngine &
```

- Now we will start the remote engines that run the experiments on the remote computers (note that the same version of Java must be used for the Experimenter and remote engines):
 - Rename the remote.policy.example file to remote.policy.
 - For each machine you want to run a remote engine on:
 - * ssh to the machine.
 - * cd to /home/johndoe/remote_engine.
 - * Run /home/johndoe/startRemoteEngine (to enable the remote engines to use more memory, modify the -Xmx option in the startRemoteEngine script).

²Weka's source code can be found in the weka-src.jar archive or obtained from CVS [4].

3.4 Configuring the Experimenter

Now we will run the Experimenter:

- HSQLDB
 - Copy the DatabaseUtils.props.hsql file from weka/experiment in the weka.jar archive to the /home/johndoe/remote_engine directory and rename it to DatabaseUtils.props.
 - Edit this file and change the "jdbcURL=jdbc:hsqldb:hsql://server_name/database_name" entry to include the name of the machine that is running your database server (e.g., jdbcURL=jdbc:hsqldb:hsql://dodo.company.com/experiment).
 - Now start the Experimenter (inside this directory):

```
java \
```

```
-cp /home/johndoe/jars/hsqldb.jar:remoteEngine.jar:/home/johndoe/weka/weka.jar \
-Djava.rmi.server.codebase=file:/home/johndoe/weka/weka.jar \
weka.gui.experiment.Experimenter
```

- MySQL
 - Copy the DatabaseUtils.props.mysql file from weka/experiment in the weka.jar archive to the /home/johndoe/remote_engine directory and rename it to DatabaseUtils.props.
 - Edit this file and change the "jdbcURL=jdbc:mysql://server_name:3306/database_name" entry to include the name of the machine that is running your database server and the name of the database the result will be stored in (e.g., jdbcURL=jdbc:mysql://dodo.company.com:3306/experiment).
 - Now start the Experimenter (inside this directory):

```
java 🔪
```

```
-cp /home/johndoe/jars/mysql.jar:remoteEngine.jar:/home/johndoe/weka/weka.jar \
-Djava.rmi.server.codebase=file:/home/johndoe/weka/weka.jar \
weka.gui.experiment.Experimenter
```

Note: the database name *experiment* can still be modified in the Experimenter, this is just the default setup.

Now we will configure the experiment:

- First of all select the Advanced mode in the Setup tab
- Now choose the *DatabaseResultListener* in the *Destination* panel. Configure this result producer:
 - HSQLDB

Supply the value **sa** for the username and leave the password empty.

– MySQL

Provide the username and password that you need for connecting to the database.

• From the *Result generator* panel choose either the *CrossValidationResult*-*Producer* or the *RandomSplitResultProducer* (these are the most commonly used ones) and then configure the remaining experiment details (e.g., datasets and classifiers).

- Now enable the *Distribute Experiment* panel by checking the tick box.
- Click on the *Hosts* button and enter the names of the machines that you started remote engines on (<Enter> adds the host to the list).
- You can choose to distribute by run or dataset.
- Save your experiment configuration.
- Now start your experiment as you would do normally.
- Check your results in the *Analyse* tab by clicking either the *Database* or *Experiment* buttons.

3.5 Troubleshooting

• If you get an error at the start of an experiment that looks a bit like this:

01:13:19: RemoteExperiment (//blabla.company.com/RemoteEngine) (sub)experiment (datataset vineyard.arff) failed : java.sql.SQLException: Table already exists: EXPERIMENT_INDEX in statement [CREATE TABLE Experiment_index (Experiment_type LONGVARCHAR, Experiment_setup LONGVARCHAR, Result_table INT)]

01:13:19: dataset :vineyard.arff RemoteExperiment (//blabla.company.com/RemoteEngine) (sub)experiment (datataset vineyard.arff) failed : java.sql.SQLException: Table already exists: EXPERIMENT_INDEX in statement [CREATE TABLE Experiment_index (Experiment_type LONGVARCHAR, Experiment_setup LONGVARCHAR, Result_table INT)]. Scheduling for execution on another host.

then do not panic - this happens because multiple remote machines are trying to create the same table and are temporarily locked out - this will resolve itself so just leave your experiment running - in fact, it is a sign that the experiment is working!

- If you serialized an experiment and then modify your *DatabaseUtils.props* file due to an error (e.g., a missing type-mapping), the Experimenter will use the *DatabaseUtils.props* you had at the time you serialized the experiment. Keep in mind that the serialization process also serializes the *DatabaseUtils* class and therefore stored your props-file! This is another reason for storing your experiments as XML and not in the properietary binary format the Java serialization produces.
- Using a corrupt or incomplete *DatabaseUtils.props* file can cause peculiar interface errors, for example disabling the use of the "User" button along-side the database URL. If in doubt copy a clean *DatabaseUtils.props* from CVS [4].
- If you get NullPointerException at java.util.Hashtable.get() in the Remote Engine do not be alarmed. This will have no effect on the results of your experiment.

4 Analysing Results

4.1 Setup

Weka includes an experiment analyser that can be used to analyse the results of experiments (in this example, the results were sent to an *InstancesResultListener*). The experiment shown below uses 3 schemes, ZeroR, OneR, and J48, to classify the Iris data in an experiment using 10 train and test runs, with 66% of the data used for training and 34% used for testing.

👙 Weka Experiment En	vironment					_ 0
Setup Run Analyse						
experiment Configuration Mo	de:		Simple		Advanced	
Open			<u>S</u> ave		New	
Destination						
Choose InstancesRes	ultListener - O Exp	eriment1.arff				
Result generator						
Choose RandomSplitF	ResultProducer - P	66.0 - OsplitEvalutor	Out.zip -₩ weka.ex	periment.Classifier	SplitEvaluator\	∿weka.classifie
Runs	Distribute experi	ment	Generator pro	perties		
From: 1 To: 10		Hosts	Enabled	•	Select pro	perty
	By data set	O By run				-
Iteration control			Choose	J48 -C 0.25 -M 2		Add
Data sets first Datasets	 Custom g 	enerator first	OneR -B 6			
Datasets			J48 -C 0.25 -N	12		
	Edit selecte	Delete select				
Use relative pat						
.\data\iris.arff						
		0				
Up		Down	Delete	Edit	Up	Down
		1	lotes			

After the experiment setup is complete, run the experiment. Then, to analyse the results, select the *Analyse* tab at the top of the Experiment Environment window.

Click on *Experiment* to analyse the results of the current experiment.

≜ Weka Experimen	t Environment					_ 🗆 ×
Setup Run Analy	rse					
Source						
				File	Database	Experiment
Got 30 results				<u>F</u> ile	Datanase	Experiment
Configure test		1	Test output			
Testing with	Paired T-Tester (cor 🔻	0.000	Available resultsets			
Tooking <u>m</u> kin			(1) rules.ZeroR '' 4805554	146586795	4	
Row	Select		(2) rules.OneR '-B 6' -245			
			(3) trees.J48 '-C 0.25 -M 3	2' -21773	3168393644444	
<u>C</u> olumn	Select					
Comparison field	Percent correct					
Comparison neid	Percent_correct					
Significance	0.05					
Sorting (asc.) by	<default></default>					
Solund (asc.) by	subiduit.					
Test <u>b</u> ase	Select					
Displayed Columns	Columns					
Show std. deviations						
5101 3td. 009 <u>0</u> 0013						
<u>O</u> utput Format	Select					
Perform test	Save output					
Result list						
16:36:04 - Available res	ultsets					

The number of result lines available (Got 30 results) is shown in the Source panel. This experiment consisted of 10 runs, for 3 schemes, for 1 dataset, for a total of 30 result lines. Results can also be loaded from an earlier experiment file by clicking File and loading the appropriate .arff results file. Similarly, results sent to a database (using the DatabaseResultListener) can be loaded from the database.

Select the *Percent_correct* attribute from the *Comparison field* and click *Perform* test to generate a comparison of the 3 schemes.

駦 Weka Experimen	t Environment	
Setup Run Analy	rse	
Source		
Got 30 results		Eile Database Experiment
Configure test		Test output
Testing <u>w</u> ith	Paired T-Tester (cor 💌	Tester: weka.experiment.PairedCorrectedTTester Analysing: Percent_correct
Row	Select	Datasets: 1 Resultsets: 3
<u>C</u> olumn	Select	Confidence: 0.05 (two tailed) Sorted by: -
Comparison field	Percent_correct	Date: 21/12/05 16:37
Significance	0.05	Dataset (1) rules.Ze (2) rules (3) trees
Sorting (asc.) by	<default></default>	iris (10) 33.33 94.31 v 94.90 v
Test <u>b</u> ase	Select	(\(\nu\) /*) (1/0/0) (1/0/0)
Displayed Columns	Columns	
Show std. deviations		Key: (1) rules.ZeroR '' 48055541465867954
<u>O</u> utput Format	Select	(2) rules.OneR '-B 6' -2459427002147861445 (3) trees.J48 '-C 0.25 -M 2' -217733168393644444
Perform test	Save output	
Result list		
16:37:11 - Percent_corr	rect - rules.ZeroR " 4805 🕽	

The schemes used in the experiment are shown in the columns and the datasets used are shown in the rows.

The percentage correct for each of the 3 schemes is shown in each dataset row: 33.33% for ZeroR, 94.31% for OneR, and 94.90% for J48. The annotation v or * indicates that a specific result is statistically better (v) or worse (*) than the baseline scheme (in this case, ZeroR) at the significance level specified (currently 0.05). The results of both OneR and J48 are statistically better than the baseline established by ZeroR. At the bottom of each column after the first column is a count (xx/ yy/ zz) of the number of times that the scheme was better than (xx), the same as (yy), or worse than (zz), the baseline scheme on the datasets used in the experiment. In this example, there was only one dataset and OneR was better than ZeroR once and never equivalent to or worse than ZeroR (1/0/0); J48 was also better than ZeroR on the dataset.

The standard deviation of the attribute being evaluated can be generated by selecting the *Show std. deviations* check box and hitting *Perform test* again. The value (10) at the beginning of the *iris* row represents the number of estimates that are used to calculate the standard deviation (the number of runs in this case).

👙 Weka Experimen	t Environment		
Setup Run Analy	rse		
Source Got 30 results			Elle Database Experiment
Configure test			Test output
Testing <u>w</u> ith	Paired T-Tester (cor	•	Tester: weka.experiment.PairedCorrectedTTester Analysing: Percent_correct
Row	Select		Datasets: 1 Resultsets: 3
<u>C</u> olumn	Select		Confidence: 0.05 (two tailed) Sorted by: -
Comparison field	Percent_correct	-	Date: 21/12/05 16:37
Significance	0.05		Dataset (1) rules.ZeroR '' (2) rules.OneR (3) trees.J48 '
Sorting (asc.) by	<default></default>	•	iris (10) 33.33(0.00) 94.31(2.52) v 94.90(2.95) v
Test <u>b</u> ase	Select		(v/ /*) (1/0/0) (1/0/0)
Displayed Columns	Columns		
Show std. deviations	V		Key: (1) rules.ZeroR '' 48055541465867954
<u>O</u> utput Format	Select		(2) rules.OneR '-B 6' -2459427002147861445 (3) trees.J48 '-C 0.25 -M 2' -21773316839364444
Perform test	Save output		
Result list			
16:37:40 - Percent_con	rect - rules.ZeroR " 4805		

Selecting Number_correct as the comparison field and clicking Perform test generates the average number correct (out of 50 test patterns - 33% of 150 patterns in the Iris dataset).

🌲 Weka Experiment	t Environment	
Setup Run Analy	se	
Source		
Got 30 results		File Database Experiment
Configure test		Test output
Testing <u>w</u> ith	Paired T-Tester (cor 🔻	Tester: weka.experiment.PairedCorrectedTTester Analysing: Number correct
Row	Select	Datasets: 1 Resultsets: 3
<u>C</u> olumn	Select	Confidence: 0.05 (two tailed) Sorted by: -
Comparison field	Number_correct 💌	Date: 21/12/05 16:38
Significance	0.05	Dataset (1) rules.Ze (2) rules (3) trees
Sorting (asc.) by	<default></default>	iris (10) 17.00 48.10 v 48.40 v
Test <u>b</u> ase	Select	 (\v/ /*) (1/0/0) (1/0/0)
Displayed Columns	Columns	
Show std. deviations		Key: (1) rules.ZeroR '' 48055541465867954
<u>O</u> utput Format	Select	(2) rules.OneR '-B 6' -2459427002147861445 (3) trees.J48 '-C 0.25 -M 2' -217733168393644444
Perform <u>t</u> est	Save output	
Result list		
16:38:12 - Number_corr	ect - rules.ZeroR " 4805 🗘	

Clicking on the button for the *Output format* leads to a dialog that lets you choose the precision for the *mean* and the *std. deviations*, as well as the format of the output. Checking the *Show Average* checkbox adds an additional line to the output listing the average of each column. With the *Remove filter classnames* checkbox one can remove the filter name and options from processed datasets (filter names in Weka can be quite lengthy).

The following formats are supported:

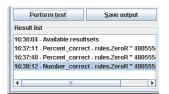
- $\bullet~\mathrm{CSV}$
- GNUPlot
- HTML

- \bullet LaTeX
- Plain text (default)
- Significance only

🚖 Output Format	×
Mean Precision	2
StdDev. Precision	2
Output <u>F</u> ormat	Plain Text 💌
Show <u>A</u> verage	
Remove filter classnames	: 🗌
	<u>O</u> K <u>C</u> ancel

4.2 Saving the Results

The information displayed in the *Test output* panel is controlled by the currentlyselected entry in the *Result list* panel. Clicking on an entry causes the results corresponding to that entry to be displayed.



The results shown in the *Test output* panel can be saved to a file by clicking *Save output*. Only one set of results can be saved at a time but Weka permits the user to save all results to the same file by saving them one at a time and using the *Append* option instead of the *Overwrite* option for the second and subsequent saves.

File query				×
? File exists				
Append	Overwrite	Choose new name	Cancel	

4.3 Changing the Baseline Scheme

The baseline scheme can be changed by clicking *Select base...* and then selecting the desired scheme. Selecting the **OneR** scheme causes the other schemes to be compared individually with the **OneR** scheme.

🚖 Select items	×					
rules.ZeroR " 48055541465867954						
rules.OneR '-B 6' -2459427002147861445						
trees.J48 '-C 0.25 -M 2' -217733168393644	444					
Summary						
Ranking						
Select Pattern Cancel						

If the test is performed on the *Percent_correct* field with OneR as the base scheme, the system indicates that there is no statistical difference between the results for OneR and J48. There is however a statistically significant difference between OneR and ZeroR.

😓 Weka	a Experimer	nt Environment		
Setup	Run Anat	yse		
Source				
Got 30 res	sults		Eile Database Experimen	nt
Configur	e test		Test output	
	Testing <u>w</u> ith	Paired T-Tester (cor 💌	Tester: weka.experiment.PairedCorrectedTTester	
		01.1	Analysing: Percent_correct Datasets: 1	
	Row	Select	Resultsets: 3	
	Column	Select	Confidence: 0.05 (two tailed)	
			Sorted by: - Date: 21/12/05 16:41	
Con	nparison field	Percent_correct	Date: 21/12/05 10:41	
	Significance	0.05	Dataset (2) rules.On (1) rules (3) trees	
<u>S</u> 01	rting (asc.) by	<default></default>	iris (10) 94.31 33.33 * 94.90	
	Test <u>b</u> ase	Select	(v/ /*) (0/0/1) (0/1/0)	
Displa	nyed Columns	Columns		
Show s	td. devi <u>a</u> tions		Key: (1) rules.ZeroR '' 48055541465867954	
C	Output Format	Select	(2) rules.OneR '-B 6' -2459427002147861445 (3) trees.J48 '-C 0.25 -H 2' -217733168393644444	
Per	rform <u>t</u> est	Save output		
Result lis	st			
	- Percent_co	rrect - rules.OneR '-B 6' -2		

4.4 Statistical Significance

The term statistical significance used in the previous section refers to the result of a pair-wise comparison of schemes using either a standard T-Test or the corrected resampled T-Test [2]. The latter test is the default, because the standard T-Test can generate too many significant differences due to dependencies in the estimates (in particular when anything other than one run of an x-fold cross-validation is used). For more information on the T-Test, consult the Weka book [1] or an introductory statistics text. As the significance level is decreased, the confidence in the conclusion increases.

In the current experiment, there is not a statistically significant difference between the $\tt OneR$ and J48 schemes.

4.5 Summary Test

Selecting *Summary* from *Test base* and performing a test causes the following information to be generated.

🚔 Weka Experimer	nt Environment						
Setup Run Anat	yse						
Source Got 30 results		<u>File</u> <u>D</u> atabase <u>Experiment</u>					
Configure test	1	Test output					
Testing <u>w</u> ith	Paired T-Tester (cor 💌	Tester: weka.experiment.PairedCorrectedTTester Analysing: Percent_correct					
Row	Select	Datastis Datastis Confidence: 0.05 (two tailed) Sorted by: - Date: 21/12/05 16:42					
<u>C</u> olumn	Select						
Comparison field	Percent_correct						
Significance	0.05	a b c (No. of datasets where [col] >> [row])					
Sorting (asc.) by	<default></default>	- 1 (1) 1 (1) a = (1) rules.ZeroR '' 48055541465867954 0 (0) - 1 (0) b = (2) rules.OneR '-B 6' -2459427002147861445					
Test <u>b</u> ase	Select	0 (0) 0 (0) - c = (3) trees.J48 '-C 0.25 -H 2' -217733168393644444					
Displayed Columns	Columns						
Show std. deviations							
<u>O</u> utput Format	Select						
Perform <u>t</u> est	Save output						
Result list							
16:42:24 - Percent_co	rrect - Summary						

In this experiment, the first row $(-1\ 1)$ indicates that column b (OneR) is better than row a (ZeroR) and that column c (J48) is also better than row a. The number in brackets represents the number of significant wins for the column with regard to the row. A 0 means that the scheme in the corresponding column did not score a single (significant) win with regard to the scheme in the row.

4.6 Ranking Test

Selecting $Ranking \mbox{ from } Test \mbox{ base } \mbox{ causes the following information to be generated.}$

🚖 Weka Experimen	t Environment					<u></u>
Setup Run Analy	se					
Source						
Got 30 results						Eile Database Experiment
Configure test		Te	st oi	utput		
Testing <u>w</u> ith	Paired T-Tester (cor 🔻		este	r: sing		weka.experiment.PairedCorrectedTTester Percent correct
Row	Select	De		1		
<u>C</u> olumn	Select	Co	onfi		e:	0.05 (two tailed)
Comparison field	Percent_correct	Da	ate:			21/12/05 16:42
Significance	0.05	>-	-<	>	<	Resultset
Sorting (asc.) by	<default></default>	2	1	1 1		trees.J48 '-C 0.25 -M 2' -217733168393644444 rules.OneR '-B 6' -2459427002147861445
Test <u>b</u> ase	Select	-	-2	0	2	rules.ZeroR '' 48055541465867954
Displayed Columns	Columns					
Show std. deviations						
<u>O</u> utput Format	Select					
Perform test	Save output					
Result list						
16:42:48 - Percent_corr	rect - Ranking					

The ranking test ranks the schemes according to the total number of significant wins (>) and losses (<) against the other schemes. The first column (> - <) is the difference between the number of wins and the number of losses. This difference is used to generate the ranking.

References

- [1] Witten, I.H. and Frank, E. (2005) Data Mining: Practical machine learning tools and techniques. 2nd edition Morgan Kaufmann, San Francisco.
- [2] Bengio, Y. and Nadeau, C. (1999) Inference for the Generalization Error.
- [3] Ross Quinlan (1993). C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo, CA.
- [4] CVS http://weka.sourceforge.net/wiki/index.php/CVS
- [5] HSQLDB http://hsqldb.sourceforge.net/
- [6] MySQL http://www.mysql.com/
- [7] WekaDoc http://weka.sourceforge.net/wekadoc/