WEKA: The Waikato Environment for Knowledge Analysis

Stephen R. Garner

Department of Computer Science, University of Waikato, Hamilton.

ABSTRACT

WEKA is a workbench designed to aid in the application of machine learning technology to real world data sets, in particular, data sets from New Zealand's agricultural sector. In order to do this a range of machine learning techniques are presented to the user in such a way as to hide the idiosyncrasies of input and output formats, as well as allow an exploratory approach in applying the technology. The system presented is a component based one that also has application in machine learning research and education.

1. Introduction

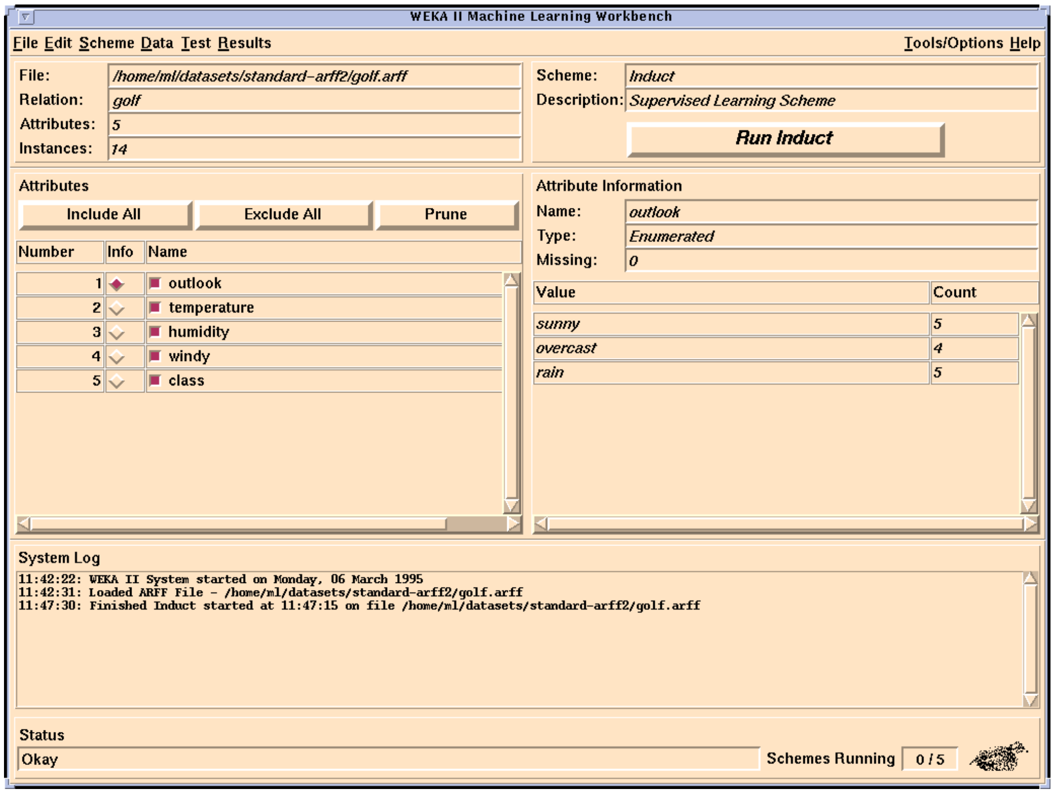
The WEKA machine learning workbench has grown out of the need to be able to apply machine learning to real world data sets in a way that promotes a "what if?…" or exploratory approach. Each machine learning algorithm implementation requires the data to be present in its own format, and has its own way of specifying parameters and output. The WEKA system was designed to bring a range of machine learning techniques or schemes under a common interface so that they may be easily applied to this data in a consistent method. This interface should be flexible enough to encourage the addition of new schemes, and simple enough that users need only concern themselves with the selection of features in the data for analysis and what the output means, rather than how to use a machine learning scheme.

The WEKA system has been used over the past year to work with a variety of agricultural data sets in order to try to answer a range of questions. For example: "Can you find a set of rules that model the factors in the decision to cull a dairy cow from a herd?" The data sets that occur in the agricultural sector tend to be bigger and of a lower quality than those in machine learning research. There are normally more features in the data, lots of missing values, mixtures of integer values and codes in an attribute and subjective measurements of attributes. Application of WEKA to this sort of data has led to insight into how to apply machine learning algorithms to this data, what extra work is required in working with the data, and what tools are needed to the support machine learning in this environment.

2. WEKA Design and Implementation

The WEKA system is not so much a single program as a collection of inter-dependent programs bound together by a common user interface. Typically these modules fall into three categories: data set processing, machine learning schemes, and output processing. The processing of data sets involves extracting information about a data set for the user, splitting data sets into test and training sets, filtering out features in the data not required by the user, and translating the data set into a form suitable for a machine learning scheme to work with. Machine learning schemes are implementations of machine learning algorithms and typically take a converted data set and produce some output, normally a rule set. Output processing modules are concerned with taking the output from a machine learning scheme performing some task with it, such as evaluating a rule set against a test file or displaying the output in a window for the user.

Interaction by the user with WEKA will result in the modules being combined in such a way as to produce the desired output. For example, a typical task might involve selecting a data set to train on, selecting a data set to test with, excluding features not required from the data sets, choosing a machine learning scheme, running the scheme on the training data and then looking at the rules produced and how well they did on the test data.


WEKA II Machine Learning Workbench
File Edit Scheme Data Test Results
Tools/Options Help
Run Induct
Include All
Exclude All
Prune

File: /home/ml/datasets/standard-arff2/golf.arff
Relation: golf
Attributes: 5
Instances: 14

Scheme: Induct
Description: Supervised Learning Scheme

Attributes

Number	Info	Name
1	◆	outlook
2	◇	temperature
3	◇	humidity
4	◇	windy
5	◇	class

Attribute Information

Name: outlook
Type: Enumerated
Missing: 0

Value	Count
sunny	5
overcast	4
rain	5

System Log

```
11:42:22: WEKA II System started on Monday, 06 March 1995
11:42:31: Loaded ARFF File - /home/ml/datasets/standard-arff2/golf.arff
11:47:30: Finished Induct started at 11:47:15 on file /home/ml/datasets/standard-arff2/golf.arff
```

Status: Okay
Schemes Running: 0 / 5

Figure 1. WEKA Workbench Main Screen

The WEKA workbench itself (Figure 1) is written using the TCL/TK [1] scripting language and X window tool kit combination and runs on Sun Microsystems UNIX systems using Solaris 2. The individual modules are typically written in C or using standard UNIX utilities, such as lex, yacc and awk, and can be used outside of the WEKA environment if required, for example in a shell script for multiple data set feature filtering. The machine learning schemes are implemented in a variety of programming languages, depending on the choice of the programmer, and currently WEKA incorporates schemes written in C, C++ and LISP. A simple rule evaluator has been written in C, while a more complete one, PREval, has also been developed in PROLOG.

In the following sections the WEKA common data set format, the machine learning schemes that have been incorporated, and the common WEKA rule format and evaluator will be discussed.

Data Set Format

The WEKA system uses a common file format to store its data sets and thus presents the user with a consistent view of the data regardless of what machine learning scheme may be used. This file format, the Attribute-Relation File Format (ARFF), defines a data set in terms of a relation or table made up of attributes or columns of data. Information about the names of the relation, and the data types of the attributes are stored in the ARFF header, with the examples or instances of data being represented as rows of data in the body of the ARFF file. Attributes are currently allowed to take on three different data types: integers, real or floating point numbers and enumerations. With the numeric attributes an optional range may be specified for range checking and Boolean attributes are treated as an enumeration with two values. An example ARFF file is shown in Figure 2 below.

```
@relation golf

@attribute outlook { sunny, overcast, rain}
@attribute temperature real [0.0,100]
@attribute humidity real
@attribute windy { true, false}
@attribute class { Play, 'Dont Play' }

@data
% 14 instances follow
sunny, 85, 85, false, 'Dont Play'
sunny, 80, 90, true, 'Dont Play'
overcast, 83, 78, false, Play
rain, 70, 96, false, Play
rain, 68, 80, false, Play
rain, 65, 70, true, 'Dont Play'
overcast, 64, 65, true, Play
sunny, 72, 95, false, 'Dont Play'
sunny, 69, 70, false, Play
rain, 75, 80, false, Play
sunny, 75, 70, true, Play
overcast, 72, 90, true, Play
overcast, 81, 75, false, Play
rain, 71, 80, true, 'Dont Play'
```

Figure 2. Sample ARFF Data Set

A variety of tools exist for processing ARFF files, most of which are based around an library of functions that allow ARFF files to be accessed from C programs. Some of these tools include facilities for splitting data sets up into test and training sets, filtering out attributes, providing summary information such as missing value frequencies, and a range of tools for converting an ARFF file to the input format required by a machine learning scheme.

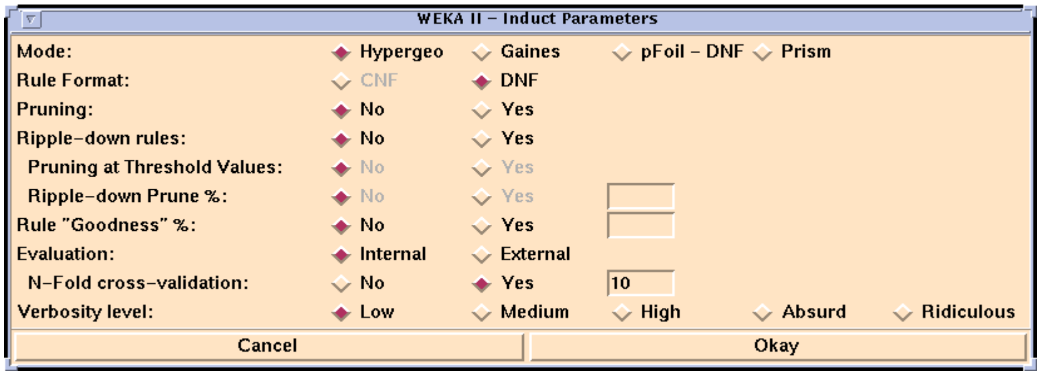
As well, work is continuing on converting the data sets to be found in the Machine Learning Database Repository at the University of California, Irvine [2] into ARFF format so that a set of standard data sets are available for users.

Machine Learning Schemes

WEKA currently incorporates a variety of machine learning schemes from the areas of unsupervised and unsupervised learning.

The unsupervised learning schemes include Classweb, a variant of the COBWEB conceptual clustering system [3], and AutoClass [4], a Bayesian classifier. These schemes are useful for determining classes in data sets where no classes have been identified in advance. In the case of Classweb, the output is a concept hierarchy with the most general concept or class at the top and the most specific concept at the bottom. AutoClass induces a number of classes and instances are assigned probabilities of membership in those classes.

C4.5 [5] , FOIL [6] , Induct [7] , IBL [8], Kstar and 1R [9] make up the set of supervised learning schemes available to the user. All these schemes work with data sets where a classification has already been defined for each instance and this classification may have been either assigned by a human expert or from a scheme such as AutoClass. In the case of C4.5 the output will be a decision tree, with the root and sub-ordinate nodes representing tests on an attribute and leaf node representing the classification. Induct, FOIL and 1R all produce rules that describe a classification based on combinations of attribute tests. Kstar and IBL schemes are instance-based learning schemes that use similarity measures (Kolmogorov complexity in Kstar, Euclidean distance in IBL) to match a new instance to the most similar instances already seen to get a classification.


WEKA II – Induct Parameters

Mode:	[✓ Hypergeo]	Gaines	pFoil – DNF	Prism	
Rule Format:	CNF	[✓ DNF]			
Pruning:	[✓ No]	Yes			
Ripple–down rules:	[✓ No]	Yes			
Pruning at Threshold Values:	[✓ No]	Yes			
Ripple–down Prune %:	[✓ No]	Yes			
Rule "Goodness" %:	[✓ No]	Yes			
Evaluation:	[✓ Internal]	External			
N–Fold cross–validation:	No	[✓ Yes]	10		
Verbosity level:	[✓ Low]	Medium	High	Absurd	Ridiculous

Figure 3. Parameter Selection Dialog for Induct

Adding a new scheme to WEKA involves two tasks: writing a ARFF to input format translator and writing a TCL/TK module to allow the users to configure the scheme's parameters. The translator is typically written using the C library routines for manipulating ARFF files, though some schemes such as Induct and Kstar expect input in ARFF format. The parameter module expects a set of known command line values from WEKA, including data file names and attributes to use, prompts the user to configure the scheme (Figure 3), and then runs the appropriate WEKA modules for filtering and translating the data sets before actually running the scheme itself. The output from the scheme is then either passed back to WEKA for display, or processed further, say by the rule evaluator, before being passed back. When the scheme parameter module has been created an entry about the module is made in a WEKA library file that is consulted upon start up.

Output Processing

In the current version of WEKA any output that is created by a machine learning scheme is passed back to WEKA in text form that is able to displayed in a scrollable text viewer. This view allows text to be copied to other X applications, printed or save to a file. If the user selected external evaluation in one of the schemes that allows it then the output will be passed back as per normal but it will also be converted into the WEKA rule format and evaluated using the WEKA PROLOG rule evaluator, PREval. This output translation varies for the schemes. In FOIL it is simply a matter of converting FOIL rules to PREval rules, and Induct provides an option to produce its output directly in the appropriate format. In the case of C4.5 both decision trees and rules have to be converted into PREval rules. Figure 4 has an example decision tree converted to PREval format.

C4.5 Decision Tree

```
'outlook' = 'overcast': 'Play' (4.0)
'outlook' = 'sunny':
|   'humidity' <= 75 : 'Play' (2.0)
|   'humidity' > 75 : 'Dont Play' (3.0)
'outlook' = 'rain':
|   'windy' = 'true': 'Dont Play' (2.0)
|   'windy' = 'false': 'Play' (3.0)
```

WEKA Rules

```
% Rule 1 - Length 1
'class'('Play') :- ( 'outlook'('overcast') ).
% Rule 2 - Length 2
'class'('Play') :- ( 'humidity'(X_2), X_2 =< 75 ),
       ( 'outlook'('sunny') ).
% Rule 3 - Length 2
'class'('Dont Play') :- ( 'humidity'(X_4), X_4 > 75 ),
       ( 'outlook'('sunny') ).
% Rule 4 - Length 2
'class'('Dont Play') :- ( 'outlook'('rain') ),
       ( 'windy'('true') ).
% Rule 5 - Length 2
'class'('Play') :- ( 'outlook'('rain') ),
       ( 'windy'('false') ).
```

Figure 4. C4.5 Decision Tree and its Mapping in PREval Form

PREval itself takes an rule file and an ARFF file and evaluates how well the rules cover the classifications in the data file. It provides figures for classification accuracy, including percentages correctly classified, incorrectly classified, classified by multiple rules and not classified at all, as well as confusion matrices to show the distribution of miss-classifications and statistics on how each rule does. Recent improvements to PREval include the addition of entropy measures for calculating the complexity of rule sets and data sets with respect to a rule set.

3. Applications

The WEKA system has been applied successfully in a variety of areas including the areas of agriculture, machine learning research and education.

Agricultural

The most significant project so far carried out using the WEKA workbench has been the analysis of dairy herd data for the purposes of isolating rules that describe factors that farmers might use for culling decisions [10]. This involved working with a large data set of 19 103 records containing 705 attributes spread across 10 herds and 6 years. About 40 new attributes were derived, including attributes like age and production index relative to herd, and these were added to the original data set which was then processed by various machine learning schemes. The high level of missing values in the data adversely affected the results, though experts consulted concurred that the rules produced appeared plausible. A decision tree produced by C4.5 using 30% of the data as training data and which classified with an accuracy of 95% is shown in Figure 5.

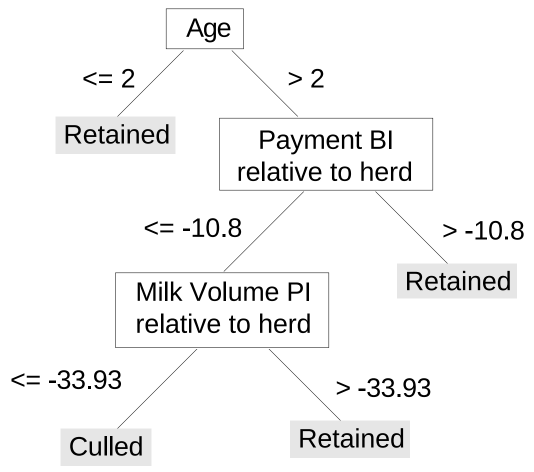


Figure 5. Cow Culling Decision Tree.

Other project in progress currently is the analysis of data pertaining to apple bruising and the prediction of when a cow will be in heat. In the case of the apples a variety of factors affect the size of an apple's bruise during the storage and packing stages of processing. Using these factors, which include apple temperature and the height it may fall, a set of rules that describe what factors results in larger bruises is being sought. A set of rules is also being sought with respect to predicting when a cow will be in heat based upon information contained in a herd milking database. This involves using information such as the volume of milk produced, its conductivity and the order the cow came into the shed, as well as recognizing temporal patterns in the data as the event in question is cyclical.

Research

The WEKA system has also proved to be a valuable in machine learning research.

Firstly it is useful in the area of supporting the development of new machine learning algorithms from both the stand point of implementation and evaluation. The presence of defined data set file formats and tools to access and manipulate the contents of data sets reduces the effort required in getting data into a new scheme. The presence of a common output format which can be evaluated using the PREval tool also takes the effort of evaluation away from the development process. Provided the new scheme supports the ARFF and PREval formats then is reasonably simple to then include the scheme in the WEKA system and evaluate it in the same context as the other machine learning schemes present. Machine learning schemes that have recently been implemented this way include Kstar and IB1 instance based learners, a new version of Induct that copes with both symbolic and numeric attributes and produces ripple-down rules, and a version of the simple 1-R rule inducer 1R.

Secondly WEKA allows the reproduction of previously published experiments for the purpose of evaluation and verification. An example of this is the recent use of WEKA to reproduce the results of simple 1-rule performance against C4.5. This experiment which involved multiple runs across 16 data sets was able to be performed relatively easily using the WEKA tools and was extended to include the schemes FOIL, Induct, IB1 and Kstar.

Education

WEKA has also been used in a limited role to introduce students of an advanced undergraduate course on machine learning to the subject and to the capabilities of the different sorts of schemes.

4. Extensions

Current extensions on WEKA involve two main projects, an Attribute Editor and an Experiment Editor, as well as continuing work on PREval.

The attribute editor is a tool designed to allow direct manipulation of ARFF files. Much of the work done with data sets before applying machine learning schemes to them is based on pre-processing the data in such a way as to create a data set with the appropriate features in it. This often includes creating new attributes that have been derived from other existing attributes (often either as concatenations of attributes, attributes based on conditions, and attributes derived from formulae). A prototype attribute editor has been developed that is accessible from WEKA and allows simple operations on ARFF files to be performed.

The prototype experiment editor allows a user to set up an experiment based on a selection of parameters including schemes, data sets, number of runs and training percentages. Once these parameters have been set a script is built that combines the WEKA tools appropriately and then the experiment is scheduled to run in the background. When the experiment is complete the user is notified by email, and results are recorded in the output file the user specified. Results records in this output file are also processed to present summary information, and may be merged with results records from previous experiments if required. Experiment settings may also be saved or printed out by the user as required.

5. Summary

WEKA has proved itself to be a useful and even essential tool in the analysis of real world data sets. It reduces the level of complexity involved in getting real world data into a variety of machine learning schemes and evaluating the output of those schemes. It has also provided a flexible aid for machine learning research and a tool for introducing people to machine learning in an educational environment.

Acknowledgments

This work is supported by the New Zealand Foundation for Research, Science and Technology. I would like to thank Professor Ian Witten and the other staff and students in the Waikato machine learning group for their contributions to WEKA and its associated research projects.

References

[1] Ousterhout, J. K. TCL and TK toolkit, Addison-Wesley, (1994).

[2] Murphy, P.M. and Aha, D.W. UCI repository of machine learning databases. For information contact ml-repository@ics.uci.edu, (1994).

[3] Fisher, D. Knowledge Acquisition Via Incremental Conceptual Clustering. Machine Learning, 2: 139-172, (1987).

[4] Cheeseman, P., Kelly, J., Self, M., Stutz, J., Taylor, W., & Freeman, D. AUTOCLASS: A Bayesian classification system. Proceedings of the Fifth International Conference on Machine Learning, Ann Arbor, MI: Morgan Kaufmann, 54-64, (1988).

[5] Quinlan, J.R. C4.5: Programs for Machine Learning, Morgan Kaufmann, (1992).

[6] Quinlan, J.R. Learning Logical Definitions from Relations. Machine Learning, 5: 239-266, (1990).

[7] Gaines, B.R. The tradeoff between knowledge and data in knowledge acquisition in knowledge discovery in databases, AAAI Press, 491-505, (1991).

[8] Aha, D.W. Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms. Int. J. Man-Machine Studies, 36: 267-287, (1992).

[9] Holte, R.C. Very Simple Classification Rules Perform Well on Most Commonly Used Datasets. Machine Learning, 11: 63-91, (1993).

[10] McQueen, R.J., Garner, S.R., Nevill-Manning, C.G. and Witten, I.H. Applying Machine Learning to Agricultural Data. To be published in Computers and Electronics, Elsevier Science Publishers, Amsterdam.