The WEKA data mining software: An update

Article *in* ACM SIGKDD Explorations Newsletter · November 2008

CITATIONS
6,067

READS
16,008

6 authors, including:

Mark Hall
The University of Waikato
58 PUBLICATIONS **16,275** CITATIONS
SEE PROFILE

Geoffrey Holmes
The University of Waikato
116 PUBLICATIONS **18,341** CITATIONS
SEE PROFILE

Bernhard Pfahringer
The University of Waikato
179 PUBLICATIONS **16,998** CITATIONS
SEE PROFILE

Ian Witten
The University of Waikato
537 PUBLICATIONS **63,848** CITATIONS
SEE PROFILE

The WEKA Data Mining Software: An Update

Mark Hall

Pentaho Corporation
Suite 340, 5950 Hazeltine National Dr.
Orlando, FL 32822, USA
mhall@pentaho.com

Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer
Peter Reutemann, Ian H. Witten
Department of Computer Science
University of Waikato
Hamilton, New Zealand
{eibe,geoff,bernhard,fracpete,ihw}@cs.waikato.ac.nz

ABSTRACT

More than twelve years have elapsed since the first public release of WEKA. In that time, the software has been rewritten entirely from scratch, evolved substantially and now accompanies a text on data mining [35]. These days, WEKA enjoys widespread acceptance in both academia and business, has an active community, and has been downloaded more than 1.4 million times since being placed on SourceForge in April 2000. This paper provides an introduction to the WEKA workbench, reviews the history of the project, and, in light of the recent 3.6 stable release, briefly discusses what has been added since the last stable version (Weka 3.4) released in 2003.

1. INTRODUCTION

The Waikato Environment for Knowledge Analysis (WEKA) came about through the perceived need for a unified workbench that would allow researchers easy access to state-of-the-art techniques in machine learning. At the time of the project's inception in 1992, learning algorithms were available in various languages, for use on different platforms, and operated on a variety of data formats. The task of collecting together learning schemes for a comparative study on a collection of data sets was daunting at best. It was envisioned that WEKA would not only provide a toolbox of learning algorithms, but also a framework inside which researchers could implement new algorithms without having to be concerned with supporting infrastructure for data manipulation and scheme evaluation.

Nowadays, WEKA is recognized as a landmark system in data mining and machine learning [22]. It has achieved widespread acceptance within academia and business circles, and has become a widely used tool for data mining research. The book that accompanies it [35] is a popular textbook for data mining and is frequently cited in machine learning publications. Little, if any, of this success would have been possible if the system had not been released as open source software. Giving users free access to the source code has enabled a thriving community to develop and facilitated the creation of many projects that incorporate or extend WEKA.

In this paper we briefly review the WEKA workbench and the history of project, discuss new features in the recent 3.6 stable release, and highlight some of the many projects based on WEKA.

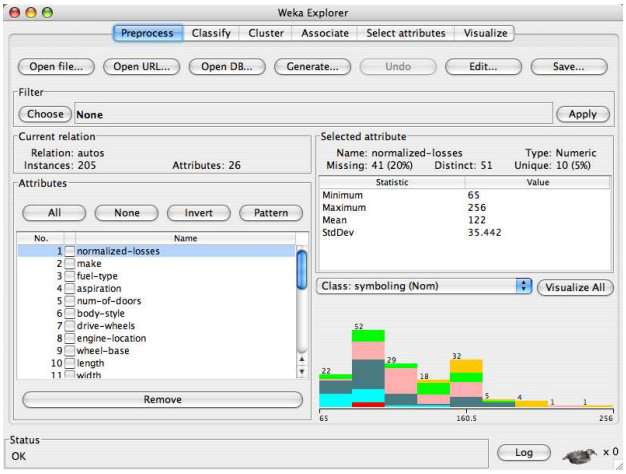

Figure 1: The WEKA Explorer user interface.

2. THE WEKA WORKBENCH

The WEKA project aims to provide a comprehensive collection of machine learning algorithms and data preprocessing tools to researchers and practitioners alike. It allows users to quickly try out and compare different machine learning methods on new data sets. Its modular, extensible architecture allows sophisticated data mining processes to be built up from the wide collection of base learning algorithms and tools provided. Extending the toolkit is easy thanks to a simple API, plugin mechanisms and facilities that automate the integration of new learning algorithms with WEKA's graphical user interfaces.

The workbench includes algorithms for regression, classification, clustering, association rule mining and attribute selection. Preliminary exploration of data is well catered for by data visualization facilities and many preprocessing tools. These, when combined with statistical evaluation of learning schemes and visualization of the results of learning, supports process models of data mining such as CRISP-DM [27].

2.1 User Interfaces

WEKA has several graphical user interfaces that enable easy access to the underlying functionality. The main graphical user interface is the "Explorer". It has a panel-based interface, where different panels correspond to different data mining tasks. In the first panel, called "Preprocess" panel, data can be loaded and transformed using WEKA's data preprocessing tools, called "filters". This panel is shown in

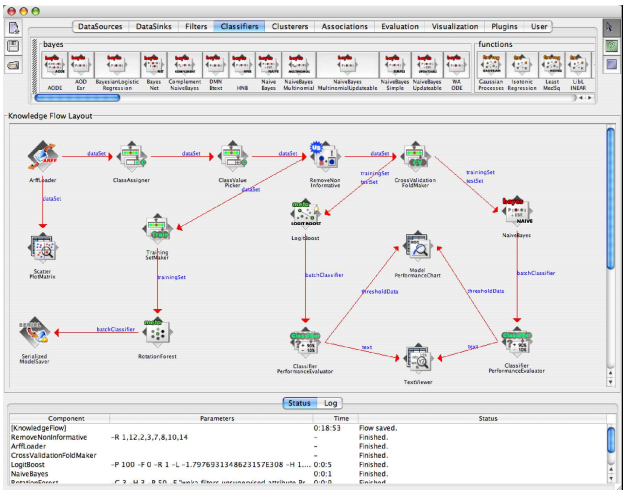

Figure 2: The WEKA Knowledge Flow user interface.

Figure 1. Data can be loaded from various sources, including files, URLs and databases. Supported file formats include WEKA's own ARFF format, CSV, LibSVM's format, and C4.5's format. It is also possible to generate data using an artificial data source and edit data manually using a dataset editor.

The second panel in the Explorer gives access to WEKA's classification and regression algorithms. The corresponding panel is called "Classify" because regression techniques are viewed as predictors of "continuous classes". By default, the panel runs a cross-validation for a selected learning algorithm on the dataset that has been been prepared in the Preprocess panel to estimate predictive performance. It also shows a textual representation of the model built from the full dataset. However, other modes of evaluation, e.g. based on a separate test set, are also supported. If applicable, the panel also provides access to graphical representations of models, e.g. decision trees. Moreover, it can visualize prediction errors in scatter plots, and also allows evaluation via ROC curves and other "threshold curves". Models can also be saved and loaded in this panel.

Along with supervised algorithms, WEKA also supports application of unsupervised algorithms, namely clustering algorithms and methods for association rule mining. These are accessible in the Explorer via the third and fourth panel respectively. The "Cluster" panel enables users to run a clustering algorithm on the data loaded in the Preprocess panel. It provides simple statistics for evaluation of clustering performance: likelihood-based performance for statistical clustering algorithms and comparison to "true" cluster membership if this is specified in one of the attributes in the data. If applicable, visualization of the clustering structure is also possible, and models can be stored persistently if necessary.

WEKA's support for clustering tasks is not as extensive as its support for classification and regression, but it has more techniques for clustering than for association rule mining, which has up to this point been somewhat neglected. Nevertheless, it does contain an implementation of the most well-known algorithm in this area, as well as a few other ones. These methods can be accessed via the "Associate" panel in the Explorer.

Perhaps one of the most important task in practical data mining is the task of identifying which attributes in the data are the most predictive ones. To this end, WEKA's Explorer has a dedicated panel for attribute selection, "Select attributes", which gives access to a wide variety of algorithms and evaluation criteria for identifying the most important attributes in a dataset. Due to the fact that it is possible to combine different search methods with different evaluation criteria, it is possible to configure a wide range of possible candidate techniques. Robustness of the selected attribute set can be validated via a cross-validation-based approach.

Note that the attribute selection panel is primarily designed for exploratory data analysis. WEKA's "FilteredClassifier" (accessible via the Classify panel) should be used to apply attribute selection techniques in conjunction with an underlying classification or regression algorithm to avoid introducing optimistic bias in the performance estimates obtained. This caveat also applies to some of the preprocessing tools—more specifically, the supervised ones—that are available from the Preprocess panel.

In many practical applications, data visualization provides important insights. These may even make it possible to avoid further analysis using machine learning and data mining algorithms. But even if this is not the case, they may inform the process of selecting an appropriate algorithm for the problem at hand. The last panel in the Explorer, called "Visualize", provides a color-coded scatter plot matrix, along with the option of drilling down by selecting individual plots in this matrix and selecting portions of the data to visualize. It is also possible to obtain information regarding individual datapoints, and to randomly perturb data by a chosen amount to uncover obscured data.

The Explorer is designed for batch-based data processing: training data is loaded into memory in its entirety and then processed. This may not be suitable for problems involving large datasets. However, WEKA does have implementations of some algorithms that allow incremental model building, which can be applied in incremental mode from a command-line interface. The incremental nature of these algorithms is ignored in the Explorer, but can be exploited using a more recent addition to WEKA's set of graphical user interfaces, namely the so-called "Knowledge Flow", shown in Figure 2.

Most tasks that can be tackled with the Explorer can also be handled by the Knowledge Flow. However, in addition to batch-based training, its data flow model enables incremental updates with processing nodes that can load and preprocess individual instances before feeding them into appropriate incremental learning algorithms. It also provides nodes for visualization and evaluation. Once a set-up of interconnected processing nodes has been configured, it can be saved for later re-use.

The third main graphical user interface in WEKA is the "Experimenter" (see Figure 3). This interface is designed to facilitate experimental comparison of the predictive performance of algorithms based on the many different evaluation criteria that are available in WEKA. Experiments can involve multiple algorithms that are run across multiple datasets; for example, using repeated cross-validation. Experiments can also be distributed across different compute nodes in a network to reduce the computational load for individual nodes. Once an experiment has been set up, it can be saved in either XML or binary form, so that it can be

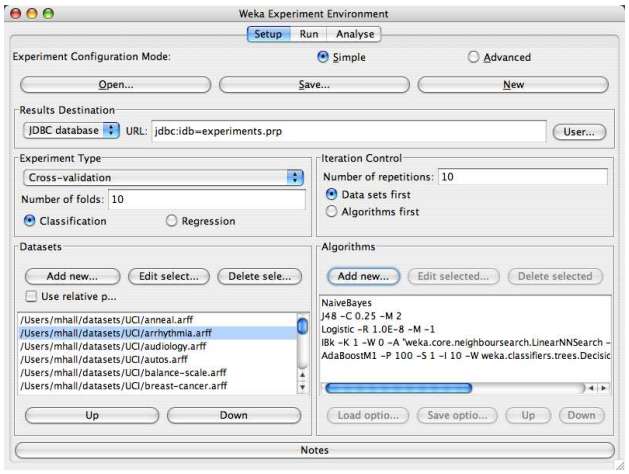

Figure 3: The WEKA Experimenter user interface.

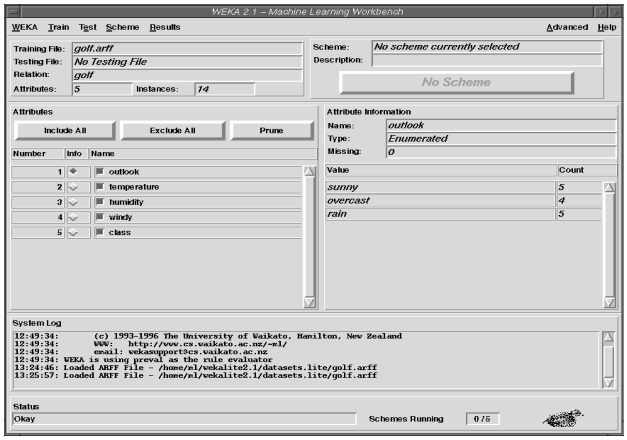

Figure 4: Back then: the WEKA 2.1 workbench user interface.

re-visited if necessary. Configured and saved experiments can also be run from the command-line.

Compared to WEKA's other user interfaces, the Experimenter is perhaps used less frequently by data mining practitioners. However, once preliminary experimentation has been performed in the Explorer, it is often much easier to identify a suitable algorithm for a particular dataset, or collection of datasets, using this alternative interface.

We would like to conclude this brief exposition of WEKA's main graphical user interfaces by pointing out that, regardless of which user interface is desired, it is important to provide the Java virtual machine that is used to run WEKA with a sufficient amount of heap space. The need to prespecify the amount of memory required, which should be set lower than the amount of physical memory of the machine that is used, to avoid swapping, is perhaps the biggest stumbling block to the successful application of WEKA in practice. On the other hand, considering running time, there is no longer a significant disadvantage compared to programs written in C, a commonly-heard argument against Java for data-intensive processing tasks, due to the sophistication of just-in-time compilers in modern Java virtual machines.

3. HISTORY OF THE WEKA PROJECT

The WEKA project was funded by the New Zealand government from 1993 up until recently. The original funding application was lodged in late 1992 and stated the project's goals as:

"The programme aims to build a state-of-the-art facility for developing techniques of machine learning and investigating their application in key areas of the New Zealand economy. Specifically we will create a workbench for machine learning, determine the factors that contribute towards its successful application in the agricultural industries, and develop new methods of machine learning and ways of assessing their effectiveness."

The first few years of the project focused on the development of the interface and infrastructure of the workbench. Most of the implementation was done in C, with some evaluation routines written in Prolog, and the user interface produced using TCL/TK. During this time the WEKA[1] acronym was coined and the Attribute Relation File Format (ARFF) used by the system was created.

The first release of WEKA was internal and occurred in 1994. The software was very much at beta stage. The first public release (at version 2.1) was made in October 1996. Figure 4 shows the main user interface for WEKA 2.1. In July 1997, WEKA 2.2 was released. It included eight learning algorithms (implementations of which were provided by their original authors) that were integrated into WEKA using wrappers based on shell scripts and data pre-processing tools written in C. WEKA 2.2 also sported a facility, based on Unix Makefiles, for configuring and running large-scale experiments based on these algorithms.

By now it was becoming increasingly difficult to maintain the software. Factors such as changes to supporting libraries, management of dependencies and complexity of configuration made the job difficult for the developers and the installation experience frustrating for users. At about this time it was decided to rewrite the system entirely in Java, including implementations of the learning algorithms. This was a somewhat radical decision given that Java was less than two years old at the time. Furthermore, the runtime performance of Java made it a questionable choice for implementing computationally intensive machine learning algorithms. Nevertheless, it was decided that advantages such as "Write Once, Run Anywhere" and simple packaging and distribution outweighed these shortcomings and would facilitate wider acceptance of the software.

May 1998 saw the final release of the TCL/TK-based system (WEKA 2.3) and, at the middle of 1999, the 100% Java WEKA 3.0 was released. This non-graphical version of WEKA accompanied the first edition of the data mining book by Witten and Frank [34]. In November 2003, a stable version of WEKA (3.4) was released in anticipation of the publication of the second edition of the book [35]. In the time between 3.0 and 3.4, the three main graphical user interfaces were developed.

In 2005, the WEKA development team received the SIGKDD Data Mining and Discovery Service Award [22]. The award

[1]The Weka is also an indigenous bird of New Zealand. Like the well-known Kiwi, it is flightless.

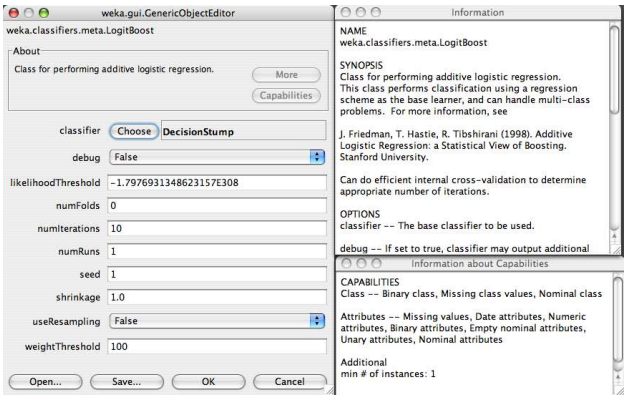

Figure 5: Capabilities and technical information meta-data.

recognized the longevity and widespread adoption of WEKA. In 2006, Pentaho Corporation became a major sponsor of the software and adopted it to form the data mining and predictive analytics component of their business intelligence suite. Pentaho is now an active contributer to the code base, and the first author is currently the maintainer-in-chief of the software. As of this writing, WEKA 3.6 (released in December 2008) is the latest version of WEKA, which, given the even-odd version numbering scheme, is considered to be a feature-stable version.

4. NEW FEATURES SINCE WEKA 3.4

Many new features have been added to WEKA since version 3.4—not only in the form of new learning algorithms, but also pre-processing filters, usability improvements and support for standards. As of writing, the 3.4 code line comprises 690 Java class files with a total of 271,447 lines of code[2]; the 3.6 code line comprises 1,081 class files with a total of 509,903 lines of code. In this section, we discuss some of the most salient new features in WEKA 3.6.

4.1 Core

The largest change to WEKA's core classes is the addition of relation-valued attributes in order to directly support multi-instance learning problems [6]. A relation-valued attribute allows each of its values to reference another set of instances (typically defining a "bag" in the multi-instance setting). Other additions to WEKA's data format include an XML format for ARFF files and support for specifying instance weights in standard ARFF files.

Another addition to the core of WEKA is the "Capabilities" meta-data facility. This framework allows individual learning algorithms and filters to declare what data characteristics they are able to handle. This, in turn, enables WEKA's user interfaces to present this information and provide feedback to the user about the applicability of a scheme for the data at hand. In a similar vein, the "TechnicalInformation" classes allow schemes to supply citation details for the algorithm that they implement. Again, this information is formatted and exposed automatically by the user interface. Figure 5 shows technical information and capabilities for the LogitBoost classifier.

Logging has also been improved in WEKA 3.6 with the addition of a central log file. This file captures all information written to any graphical logging panel in WEKA, along with any output to standard out and standard error.

[2]As computed by the Unix command: wc -l.

4.2 Learning Schemes

Many new learning algorithms have been added since WEKA 3.4 and some existing ones have been improved. An example of the latter category is instance-based learning, where there is now support for pluggable distance functions and new data structures—such as ball trees and KD trees—to speed up the search for nearest neighbors.

Some of the new classification algorithms in WEKA 3.6 include

- *Bayesian logistic regression* [13]: the BLR method for text categorization, with both Gaussian and Laplace priors.
- *Best-first decision tree* [28]: builds a decision tree using a best-first search strategy.
- *Decision table naive Bayes hybrid* [15]: a hybrid learner that combines decision tables and naive Bayes.
- *Discriminative multinomial naive Bayes* [30]: a simple Bayesian classifier with discriminative parameter learning for text categorization.
- *Functional trees* [12]: decision trees with oblique splits and linear functions at the leaves.
- *Gaussian processes* [26]: implements the well-known Gaussian process method for regression.
- *Simple CART* [3]: a decision tree learner that implements minimal cost-complexity pruning.
- *Variants of AODE* [39; 17]: Averaged One-Dependence Estimators with subsumption resolution (AODEsr) and weighted AODE (WAODE).
- *Wrapper classifiers*: allow the well known algorithms provided by the LibSVM [5] and LibLINEAR [9] third-party libraries to be used in WEKA.

In addition to these algorithms, an entire package of multi-instance algorithms has been added to WEKA since version 3.4, most of which were first distributed in the separate MILK package for multi-instance learning [37].

WEKA 3.6 also has new "meta" algorithms that can be wrapped around base learning algorithms to widen applicability or enhance performance:

- *Nested dichotomies* [10; 8]: an approach for handling multi-class classification problems with a hierarchy of two-class classifiers.
- *Dagging* [32]: a meta classifier similar to Bagging, which supplies disjoint subsets of the training data to the chosen base learning algorithm.
- *Rotation forest* [24]: generates an ensemble classifier by training a base learner on a randomly selected subspace of the input data that has been rotated using principal component analysis.

The set of clustering algorithms has also been expanded with the following members:

- *CLOPE clusterer* [38]: a fast clustering scheme for transactional data.
- *Sequential information bottleneck clusterer* [29]: a clusterer that was implemented primarily for document clustering.

4.3 Preprocessing Filters

Just as the list of learning schemes in WEKA has grown, so has the number of preprocessing tools. Some of the new filters in WEKA 3.6 include:

- *Add classification*: adds the predictions of a classifier to a data set.
- *Add ID*: adds an ID attribute to a data set—useful for keeping track of instances.
- *Add values*: adds the labels from a given list to an attribute if they are missing.
- *Attribute reorder*: changes the order of the attributes in a data set.
- *Interquartile range*: tags instances as containing outliers and extreme values based on interquartile ranges.
- *Kernel filter* [2]: converts a given set of predictor variables into a kernel matrix.
- *Numeric cleaner*: "cleanses" numeric values that exceed a threshold or are too close to a certain value by replacing them with user-supplied defaults.
- *Numeric to nominal*: converts a numeric attribute to nominal by simply adding all the observed numeric values to the list of nominal values.
- *Partitioned multi-filter*: applies a supplied list of filters to a corresponding set of attribute ranges and combines the result into a new data set.
- *Propositional to multi-instance and vice versa*: converts to and from multi-instance format.
- *Random subset*: selects a random subset of attributes.
- *RELAGGS* [19]: converts relational data into propositional data using aggregation.
- *Reservoir sample* [33]: processes instances incrementally and performs reservoir sampling, for down-sampling data sets that do not fit in main memory.
- *Subset by expression*: filter instances according to a user-specified expression.
- *Wavelet* [25]: performs a wavelet transformation on the data.

Figure 6: The GUI Chooser.

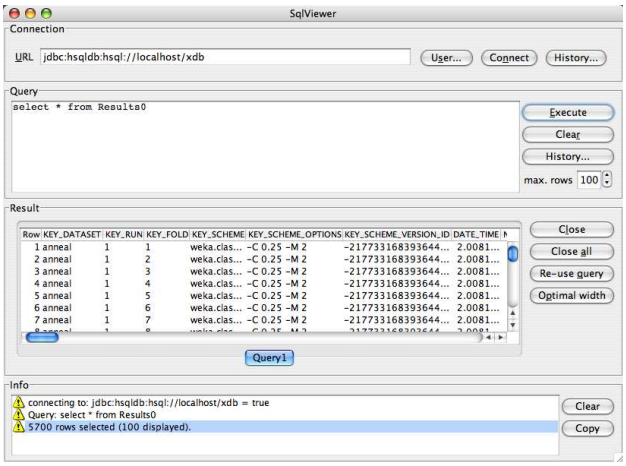

Figure 7: The SQL viewer.

4.4 User Interfaces

Aside from the afore-mentioned exposure of capabilities and technical information meta data, there has been further refinement and improvement to the GUIs in WEKA since version 3.4. The GUI Chooser—WEKA's graphical start point—has undergone a redesign and now provides access to various supporting user interfaces, system information and logging information, as well as the main applications in WEKA. Figure 6 shows the revamped GUI Chooser.

Scatter plots, ROC curves, trees and graphs can all be accessed from entries under the "Visualization" menu. The "Tools" menu provides two new supporting GUIs:

- *SQL viewer*: allows user-entered SQL to be run against a database and the results previewed. This user interface is also used in the Explorer to extract data from a database when the "Open DB" button is pressed.
- *Bayes network editor*: provides a graphical environment for constructing, editing and visualizing Bayesian network classifiers.

Figures 7 and 8 show the SQL viewer and Bayes network editor respectively.

Often it is useful to evaluate an algorithm on synthetic data. As mentioned earlier in this paper, the Explorer user interface now has a facility for generating artificial data sets

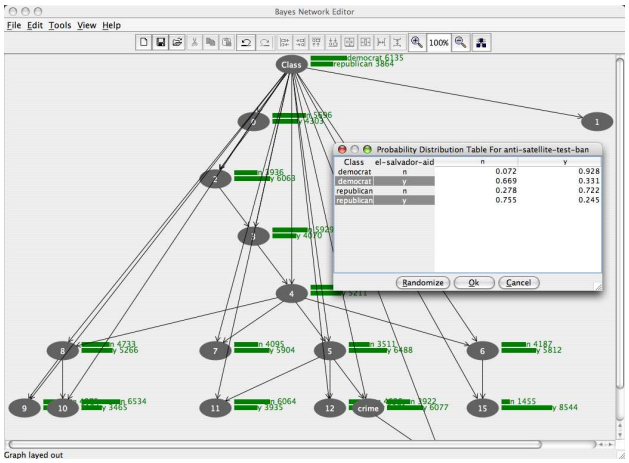

Figure 8: The Bayesian network editor.

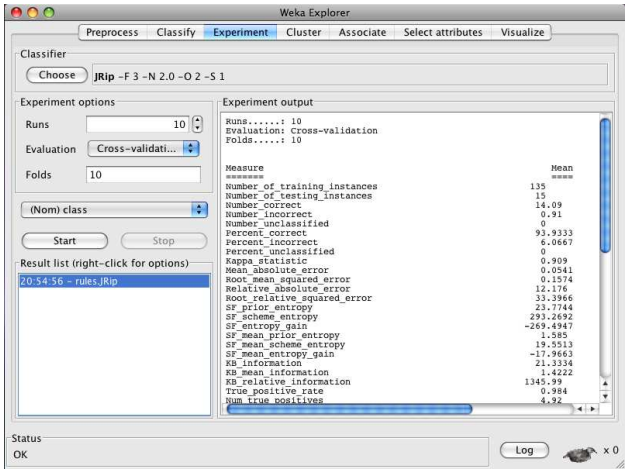

Figure 9: The Explorer with an "Experiment" tab added from a plugin.

using WEKA's data generator tools. Artificial data suitable for classification can be generated from decision lists, radial-basis function networks and Bayesian networks as well as the classic LED24 domain. Artificial regression data can be generated according to mathematical expressions. There are also several generators for producing artificial data for clustering purposes.

The Knowledge Flow interface has also been improved: it now includes a new status area that can provide feedback on the operation of multiple components in a data mining process simultaneously. Other improvements to the Knowledge Flow include support for association rule mining, improved support for visualizing multiple ROC curves and a plugin mechanism.

4.5 Extensibility

A number of plugin mechanisms have been added to WEKA since version 3.4. These allow WEKA to be extended in various ways without having to modify the classes that make up the WEKA distribution.

New tabs in the Explorer are easily added by writing a class that extends `javax.swing.JPanel` and implements the interface `weka.gui.explorer.Explorer.ExplorerPanel`. Figure 9 shows the Explorer with a new tab, provided by a plugin, for running simple experiments. Similar mechanisms allow new visualizations for classifier errors, predictions, trees and graphs to be added to the pop-up menu available in the history list of the Explorer's "Classify" panel. The Knowledge Flow has a plugin mechanism that allows new components to be incorporated by simply adding their jar file (and any necessary supporting jar files) to the `.knowledgeFlow/plugins` directory in the user's home directory. These jar files are loaded automatically when the Knowledge Flow is started and the plugins are made available from a "Plugins" tab.

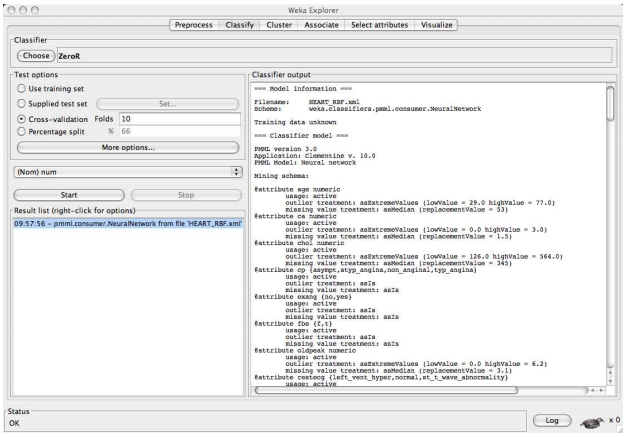

Figure 10: A PMML radial basis function network loaded into the Explorer.

4.6 Standards and Interoperability

WEKA 3.6 includes support for importing PMML models (Predictive Modeling Markup Language). PMML is a vendor-agnostic, XML-based standard for expressing statistical and data mining models that has gained wide-spread support from both proprietary and open-source data mining vendors. WEKA 3.6 supports import of PMML regression, general regression and neural network model types. Import of further model types, along with support for exporting PMML, will be added in future releases of WEKA. Figure 10 shows a PMML radial basis function network, created by the Clementine system, loaded into the Explorer.

Another new feature in WEKA 3.6 is the ability to read and write data in the format used by the well known LibSVM and SVM-Light support vector machine implementations [5]. This complements the new LibSVM and LibLINEAR wrapper classifiers.

5. PROJECTS BASED ON WEKA

There are many projects that extend or wrap WEKA in some fashion. At the time of this writing, there are 46 such projects listed on the Related Projects web page of the WEKA site[3]. Some of these include:

- *Systems for natural language processing.* There are a number of tools that use WEKA for natural language processing: GATE is a NLP workbench [11];

[3] `http://www.cs.waikato.ac.nz/ml/weka/index_related.html`

Balie performs language identification, tokenization, sentence boundary detection and named-entity recognition [21]; Senseval-2 is a system for word sense disambiguation; Kea is a system for automatic keyphrase extraction [36].

- *Knowledge discovery in biology.* Several tools using or based on WEKA have been developed to aid data analysis in biological applications: BioWEKA is an extension to WEKA for tasks in biology, bioinformatics, and biochemistry [14]; the Epitopes Toolkit (EpiT) is a platform based on WEKA for developing epitope prediction tools; maxdView and Mayday [7] provide visualization and analysis of microarray data.

- *Distributed and parallel data mining.* There are a number of projects that have extended WEKA for distributed data mining; Weka-Parallel provides a distributed cross-validation facility [4]; GridWeka provides distributed scoring and testing as well as cross-validation [18]; FAEHIM and Weka4WS [31] make WEKA available as a web service.

- *Open-source data mining systems.* Several well known open-source data mining systems provide plugins to allow access to WEKA's algorithms. The Konstanz Information Miner (KNIME) and RapidMiner [20] are two such systems. The R [23] statistical computing environment also provides an interface to WEKA through the RWeka [16] package.

- *Scientific workflow environment.* The Kepler Weka project integrates all the functionality of WEKA into the Kepler [1] open-source scientific workflow platform.

6. INTEGRATION WITH THE PENTAHO BI SUITE

Pentaho corporation is a provider of commercial open source business intelligence software. The Pentaho BI suite consists of reporting, interactive analysis, dashboards, ETL/data integration and data mining. Each of these is a separate open source project, which are tied together by an enterprise-class open source BI platform. In late 2006, WEKA was adopted as the data mining component of the suite and since then has been integrated into the platform.

The main point of integration between WEKA and the Pentaho platform is with Pentaho Data Integration (PDI), also known as the Kettle project[4]. PDI is a streaming, engine-driven ETL tool. Its rich set of extract and transform operations, combined with support for a large variety of databases, are a natural complement to WEKA's data filters. PDI can easily export data sets in WEKA's native ARFF format to be used immediately for model creation.

Several WEKA-specific transformation steps have been created so that PDI can access WEKA algorithms and be used as both a scoring platform and a tool to automate model creation. The first of these, shown in Figure 11, is called "Weka Scoring." It enables the user to import a serialized WEKA model (classification, regression or clustering) or a supported PMML model and use it to score data as part of an ETL transformation. In an operational scenario the predictive performance of a model may decrease over time.

[4]`http://kettle.pentaho.org/`

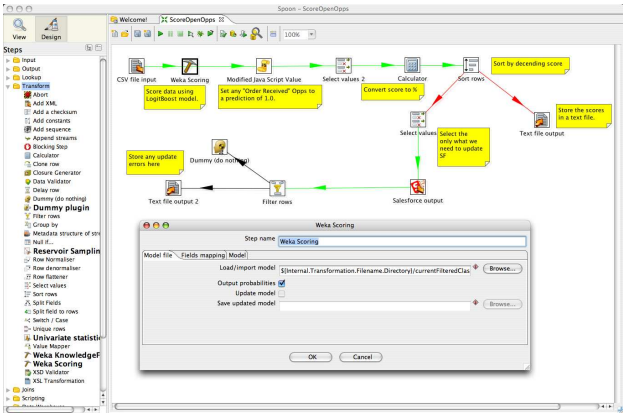

Figure 11: Scoring open sales opportunities as part of an ETL transformation.

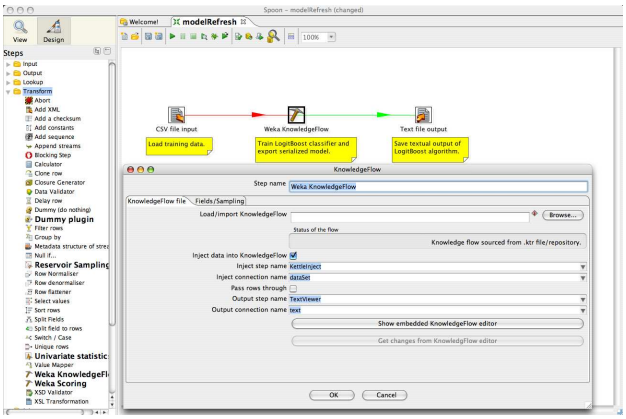

Figure 12: Refreshing a predictive model using the Knowledge Flow PDI component.

This can be caused by changes in the underlying distribution of the data and is sometimes referred to as "concept drift." The second WEKA-specific step for PDI, shown in Figure 12, allows the user to execute an entire Knowledge Flow process as part of an transformation. This enables automated periodic recreation or refreshing of a model.

Since PDI transformations can be executed and used as a source of data by the Pentaho BI server, the results of data mining can be incorporated into an overall BI process and used in reports, dashboards and analysis views.

7. CONCLUSIONS

The WEKA project has come a long way in the 16 years that have elapsed since it inception in 1992. The success it has enjoyed is testament to the passion of its community and many contributors. Releasing WEKA as open source software and implementing it in Java has played no small part in its success. These two factors ensure that it remains maintainable and modifiable irrespective of the commitment or health of any particular institution or company.

8. ACKNOWLEDGMENTS

Many thanks to past and present members of the Waikato machine learning group and the external contributers for all the work they have put into WEKA.

9. REFERENCES

[1] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludscher, and S. Mock. Kepler: An extensible system for design and execution of scientific workflows. In *In SSDBM*, pages 21–23, 2004.

[2] K. Bennett and M. Embrechts. An optimization perspective on kernel partial least squares regression. In J. S. et al., editor, *Advances in Learning Theory: Methods, Models and Applications*, volume 190 of *NATO Science Series, Series III: Computer and System Sciences*, pages 227–249. IOS Press, Amsterdam, The Netherlands, 2003.

[3] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth International Group, Belmont, California, 1984.

[4] S. Celis and D. R. Musicant. Weka-parallel: machine learning in parallel. Technical report, Carleton College, CS TR, 2002.

[5] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[6] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artif. Intell.*, 89(1-2):31–71, 1997.

[7] J. Dietzsch, N. Gehlenborg, and K. Nieselt. Mayday-a microarray data analysis workbench. *Bioinformatics*, 22(8):1010–1012, 2006.

[8] L. Dong, E. Frank, and S. Kramer. Ensembles of balanced nested dichotomies for multi-class problems. In *Proc 9th European Conference on Principles and Practice of Knowledge Discovery in Databases*, Porto, Portugal, pages 84–95. Springer, 2005.

[9] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning. Research*, 9:1871–1874, 2008.

[10] E. Frank and S. Kramer. Ensembles of nested dichotomies for multi-class problems. In *Proc 21st International Conference on Machine Learning*, Banff, Canada, pages 305–312. ACM Press, 2004.

[11] R. Gaizauskas, H. Cunningham, Y. Wilks, P. Rodgers, and K. Humphreys. GATE: an environment to support research and development in natural language engineering. In *In Proceedings of the 8th IEEE International Conference on Tools with Artificial Intelligence*, pages 58–66, 1996.

[12] J. Gama. Functional trees. *Machine Learning*, 55(3):219–250, 2004.

[13] A. Genkin, D. D. Lewis, and D. Madigan. Large-scale bayesian logistic regression for text categorization. Technical report, DIMACS, 2004.

[14] J. E. Gewehr, M. Szugat, and R. Zimmer. BioWeka—extending the weka framework for bioinformatics. *Bioinformatics*, 23(5):651–653, 2007.

[15] M. Hall and E. Frank. Combining naive Bayes and decision tables. In *Proc 21st Florida Artificial Intelligence Research Society Conference*, Miami, Florida. AAAI Press, 2008.

[16] K. Hornik, A. Zeileis, T. Hothorn, and C. Buchta. *RWeka: An R Interface to Weka*, 2009. R package version 0.3-16.

[17] L. Jiang and H. Zhang. Weightily averaged one-dependence estimators. In *Proceedings of the 9th Biennial Pacific Rim International Conference on Artificial Intelligence, PRICAI 2006*, volume 4099 of *LNAI*, pages 970–974, 2006.

[18] R. Khoussainov, X. Zuo, and N. Kushmerick. Grid-enabled Weka: A toolkit for machine learning on the grid. *ERCIM News*, 59, 2004.

[19] M.-A. Krogel and S. Wrobel. Facets of aggregation approaches to propositionalization. In T. Horvath and A. Yamamoto, editors, *Work-in-Progress Track at the Thirteenth International Conference on Inductive Logic Programming (ILP)*, 2003.

[20] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. Yale: Rapid prototyping for complex data mining tasks. In L. Ungar, M. Craven, D. Gunopulos, and T. Eliassi-Rad, editors, *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 935–940, New York, NY, USA, August 2006. ACM.

[21] D. Nadeau. Balie—baseline information extraction : Multilingual information extraction from text with machine learning and natural language techniques. Technical report, University of Ottawa, 2005.

[22] G. Piatetsky-Shapiro. KDnuggets news on SIGKDD service award. `http://www.kdnuggets.com/news/2005/n13/2i.html`, 2005.

[23] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2006. ISBN 3-900051-07-0.

[24] J. J. Rodriguez, L. I. Kuncheva, and C. J. Alonso. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1619–1630, 2006.

[25] K. Sandberg. The haar wavelet transform. `http://amath.colorado.edu/courses/5720/2000Spr/Labs/Haar/haar.html`, 2000.

[26] M. Seeger. Gaussian processes for machine learning. *International Journal of Neural Systems*, 14:2004, 2004.

[27] C. Shearer. The CRISP-DM model: The new blueprint for data mining. *Journal of Data Warehousing*, 5(4), 2000.

[28] H. Shi. Best-first decision tree learning. Master's thesis, University of Waikato, Hamilton, NZ, 2007. COMP594.

[29] N. Slonim, N. Friedman, and N. Tishby. Unsupervised document classification using sequential information maximization. In *Proceedings of the 25th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 129–136, 2002.

[30] J. Su, H. Zhang, C. X. Ling, and S. Matwin. Discriminative parameter learning for bayesian networks. In *ICML 2008*, 2008.

[31] D. Talia, P. Trunfio, and O. Verta. Weka4ws: a wsrf-enabled weka toolkit for distributed data mining on grids. In *Proc. of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2005*, pages 309–320. Springer-Verlag, 2005.

[32] K. M. Ting and I. H. Witten. Stacking bagged and dagged models. In D. H. Fisher, editor, *Fourteenth international Conference on Machine Learning*, pages 367–375, San Francisco, CA, 1997. Morgan Kaufmann Publishers.

[33] J. S. Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, 11(1):37–57, 1985.

[34] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco, 2000.

[35] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, 2 edition, 2005.

[36] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning. Kea: Practical automatic keyphrase extraction. In Y.-L. Theng and S. Foo, editors, *Design and Usability of Digital Libraries: Case Studies in the Asia Pacific*, pages 129–152. Information Science Publishing, London, 2005.

[37] X. Xu. Statistical learning in multiple instance problems. Master's thesis, Department of Computer Science, University of Waikato, 2003.

[38] Y. Yang, X. Guan, and J. You. CLOPE: a fast and effective clustering algorithm for transactional data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 682–687. ACM New York, NY, USA, 2002.

[39] F. Zheng and G. I. Webb. Efficient lazy elimination for averaged-one dependence estimators. In *Proceedings of the Twenty-third International Conference on Machine Learning (ICML 2006)*, pages 1113–1120. ACM Press, 2006.