

Data Mining in Bioinformatics using Weka

Eibe Frank^{1,*}, Mark Hall¹, Len Trigg², Geoffrey Holmes¹, and Ian H. Witten¹

¹Department of Computer Science, University of Waikato, Private Bag 3105, Hamilton, New Zealand

²Reel Two, P O Box 1538, Hamilton, New Zealand

ABSTRACT

Summary: The Weka machine learning workbench provides a general-purpose environment for automatic classification, regression, clustering, and feature selection—common data mining problems in bioinformatics research. It contains an extensive collection of machine learning algorithms and data pre-processing methods complemented by graphical user interfaces for data exploration and the experimental comparison of different machine learning techniques on the same problem. Weka can process data given in the form of a single relational table. Its main objectives are to (a) assist users in extracting useful information from data and (b) enable them to easily identify a suitable algorithm for generating an accurate predictive model from it.

Availability: <http://www.cs.waikato.ac.nz/ml/weka>

Contact: eibe@cs.waikato.ac.nz

Introduction

Bioinformatics research entails many problems that can be cast as machine learning tasks. In classification or regression the task is to predict the outcome associated with a particular individual given a feature vector describing that individual; in clustering, individuals are grouped together because they share certain properties; and in feature selection the task is to select those features that are important in predicting the outcome for an individual.

The Weka data mining suite provides algorithms for all three of these problem types. In the bioinformatics arena it has been used for automated protein annotation (Kretschmann *et al.*, 2001; Bazzan *et al.*, 2002), probe selection for gene-expression arrays (Tobler *et al.*, 2002), experiments with automatic cancer diagnosis (Li *et al.*, 2003a), developing a computational model for frame-shifting sites (Bekaert *et al.*, 2003), plant genotype discrimination (Taylor *et al.*, 2002), classifying gene expression profiles (Li and Wong, 2002), and extracting rules from them (Li *et al.*, 2003b). Many of the algorithms in Weka are described in (Witten and Frank, 2000).

Real datasets vary: no single algorithm is superior on all data mining problems. The algorithm needs to match the structure of the problem to obtain useful information or an accurate model. The aim in developing Weka was to permit a maximum of flexibility when trying machine learning meth-

ods on new datasets. This includes algorithms for learning different types of models (e.g. decision trees, rule sets, linear discriminants), feature selection schemes (fast filtering as well as wrapper approaches) and pre-processing methods (e.g. discretization, arbitrary mathematical transformations and combinations of attributes). By providing a diverse set of methods that are available through a common interface, Weka makes it easy to compare different solution strategies based on the same evaluation method and identify the one that is most appropriate for the problem at hand. It is implemented in Java and runs on almost any computing platform.

The Weka Explorer

The main interface in Weka is the Explorer, shown in Figure 1. It has a set of panels, each of which can be used to perform a certain task. The Preprocess panel, selected in Figure 1, retrieves data from a file, SQL database or URL. (A limitation is that all the data is kept in main memory, so subsampling may be needed for very large datasets.) Then the data can be pre-processed using one of Weka's filtering tools. For example, one can delete all instances (i.e. rows) in the data for which a certain attribute (i.e. column) has a particular value. An undo facility is provided to revert to an earlier state of the data if needed. The Preprocess panel also shows a histogram of the attribute that is currently selected and some statistics about it—histograms for all attributes can be shown simultaneously in a separate window.

Once a dataset has been loaded (and perhaps processed by one or more filters), one of the other panels in the Explorer can be used to perform further analysis. If the data entails a classification or regression problem, it can be processed in the Classify panel. This provides an interface to learning algorithms for classification and regression models (both are called “classifiers” in Weka), and evaluation tools for analyzing the outcome of the learning process. Weka has implementations of all major learning techniques for classification and regression: decision trees, rule sets, Bayesian classifiers, support vector machines, logistic and linear regression, multi-layer perceptrons, and nearest-neighbor methods. It also contains “meta-learners” like bagging, boosting, stacking, and schemes that perform automatic parameter tuning using cross-validation, cost-sensitive classification, etc. Learning algorithms can be evaluated using cross-validation or a hold-out set, and Weka provides standard numeric performance measures (e.g. accu-

*To whom correspondence should be addressed.

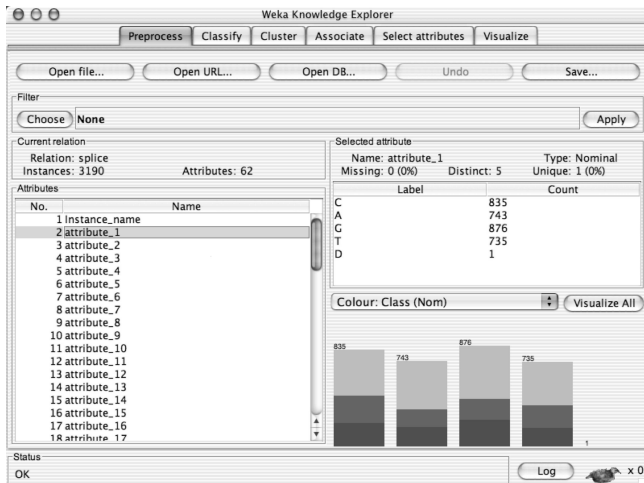


Figure 1: The Weka Explorer.

racy, root mean squared error), as well as graphical means for visualizing classifier performance (e.g. ROC curves and precision-recall curves). It is possible to visualize the predictions of a classification or regression model, enabling the identification of outliers, and to load and save models that have been generated.

The third panel in the Explorer, Cluster, gives access to Weka's clustering algorithms. These include k -means, mixtures of normal distributions with diagonal co-variance matrices estimated using EM, and a heuristic incremental hierarchical clustering scheme. Cluster assignments can be visualized and compared to actual clusters defined by one of the attributes in the data.

Weka also contains algorithms for generating association rules that can be used to identify relationships between groups of attributes in the data. These are available from the Explorer's Associate panel. However, more interesting in the context of bioinformatics is the fifth panel, which offers methods for identifying those subsets of attributes that are predictive of another (target) attribute in the data. Weka contains several methods for searching through the space of attribute subsets, as well as evaluation measures for attributes and attribute subsets. Search methods include best-first search, forward selection, genetic algorithms, and a simple ranking of attributes. Evaluation measures include correlation- and entropy-based criteria as well as the performance of a selected learning scheme (e.g. a decision tree learner) for a particular subset of attributes. Different search and evaluation methods can be combined, making the system very flexible.

The last panel in the Explorer, Visualization, shows a matrix of scatter plots for all pairs of attributes in the data. Any matrix element can be selected and enlarged in a separate window, where one can zoom in on subsets of the data and retrieve information about individual data points. A "jitter" option for exposing obscured data points is also provided.

Other interfaces to Weka

All the learning techniques in Weka can be accessed from the command line, as part of shell scripts, or from within other Java programs using the Weka API. Weka also contains an alternative graphical user interface, called "Knowledge Flow," that can be used instead of the Explorer. It caters for a more process-oriented view of data mining, where individual learning components (represented by Java beans) can be connected graphically to create a "flow" of information. Finally, there is a third graphical user interface—the "Experimenter"—which is designed for experiments that compare the performance of (multiple) learning schemes on (multiple) datasets. Experiments can be distributed across multiple computers running remote experiment servers.

Acknowledgements

Many people have contributed to the Weka project, in particular Richard Kirkby, Ashraf Kibriya, and Bernhard Pfahringer, and we thank them all for their invaluable efforts. We would also like to thank Yu Wang for suggesting that we write this note, and the New Zealand Foundation for Research, Science & Technology for funding the project.

REFERENCES

- Bazzan, A. L., *et al.* (2002) Automated annotation of keywords for proteins related to mycoplastataceae using machine learning techniques. *Bioinformatics*, 35S–43S.
- Bekaert, M., *et al.* (2003) Towards a computational model for -1 eukaryotic frameshifting sites. *Bioinformatics*, 327–335.
- Kretschmann, E., Fleischmann, W., and Apweiler, R. (2001) Automatic rule generation for protein annotation with the C4.5 data mining algorithm applied on SWISS-PROT. *Bioinformatics*, 920–926.
- Li, J., *et al.* (2003a) Discovery of significant rules for classifying cancer diagnosis data. *Bioinformatics*, 93ii–102ii.
- Li, J. and Wong, L. (2002) Identifying good diagnostic gene groups from gene expression profiles using the concept of emerging patterns. *Bioinformatics*, 725–734.
- Li, J., *et al.* (2003b) Simple rules underlying gene expression profiles of more than six subtypes of acute lymphoblastic leukemia (ALL) patients. *Bioinformatics*, 71–78.
- Taylor, J., *et al.* (2002) Application of metabolomics to plant genotype discrimination using statistics and machine learning. *Bioinformatics*, 241S–248S.
- Tobler, J. B., *et al.* (2002) Evaluating machine learning approaches for aiding probe selection for gene-expression arrays. *Bioinformatics*, 164S–171S.
- Witten, I. H. and Frank, E. (2000) *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.