Tanagra: An Evaluation

Jessica Enright Jonathan Klippenstein

November 5th, 2004

1 Introduction to Tanagra

Tanagra was written as an aid to education and research on data mining by Ricco Rakotomalala [1]. On the main page of the Tanagra site, Rakotomalala outlines his intentions for the software. He intended Tanagra to be a free, open-source, user-friendly piece of software for students and researchers to mine their data. He intended for Tanagra to provide inspiration for further pieces of data mining software. He also, as one might expect with open-source software, intended to allow researchers to extend Tanagra for their particular purposes, allowing them to more easily develop tools without building all of the required data mining infrastructure *de novo*.

The entire user operation of Tanagra is based on the stream diagram paradigm. According to Rakotomalala, this paradigm was created in the 1990's by the makers of SPAD - a piece of data analysis software. He asserts that it was then widely used then, and he selected it based on a desire to conform to a familiar interface. Other recent data mining applications also use this approach.

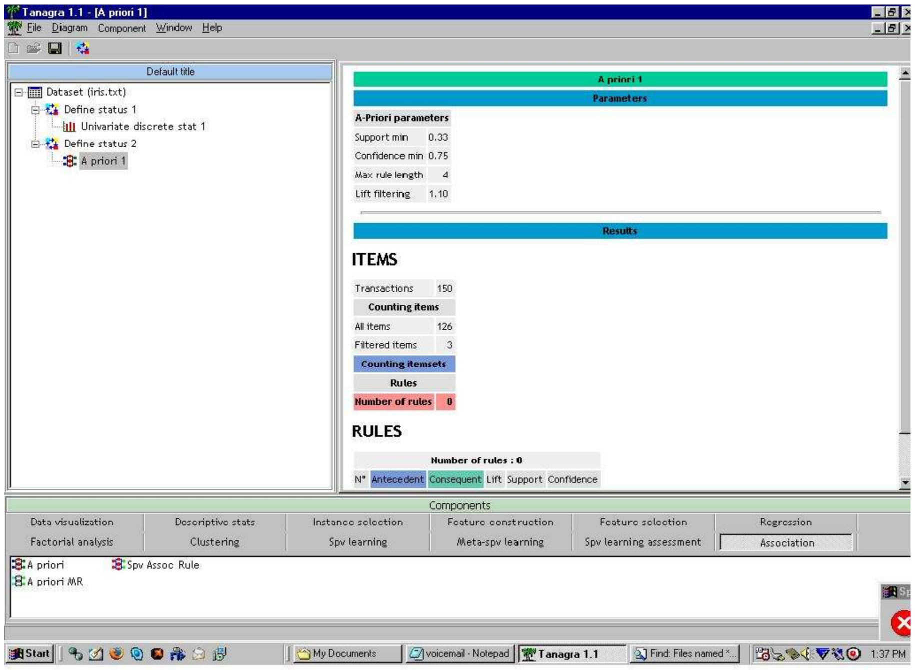
Under the stream diagram paradigm, a user builds a graph specifying the data sources, and operations on the data. Paths through the graph can describe the flow of data through manipulations and analyses. Tanagra simplifies this paradigm by restricting the graph to be a tree. This means that there can only be one parent to each node, and therefore only one data source for each operation. This is a limitation in Tanagra that will be further discussed later in this report. Tanagra's stream diagram interface can seen in the pane on the left hand side of Figure 1.

This report attempts to outline a number of the functionalities of Tanagra, as well as their shortcomings, and conclude with a final recommendation and general evaluation of the suitability of Tanagra for the usage of our fictional company.

Two other data mining programs were briefly examined for comparison: Weka [2], an open-source Java application from the University of Waikato, and PolyAnalyst [3], a commercial offering from MegaPuter Intelligence, Inc. These two programs were chosen as they have a similar sort of graphical interface as Tanagra, unlike some other packages which mainly consist of software libraries and may require custom development.

Weka has multiple interfaces, including an interactive command line interface, a graphical Explorer environment, and a graphical KnowedgeFlow environment. In the Explorer, data is loaded in one tab, filtered in the next, processed in the next, and finally displayed or saved in the last tab. The KnowedgeFlow is similar to Tanagra's stream diagram. Data source and sink objects are created as nodes that can be connected through a series of filter and analysis operation objects.

Due to issues with the PolyAnalyst installer, we were unable to install and run the demo, so the MegaPuter website is used as a reference for PolyAnalyst's capabilities. Although the material is of a marketing nature and must be taken with a grain of salt, it is still useful to get an idea of



A priori 1

Parameters

A-Priori parameters	
Support min	0.33
Confidence min	0.75
Max rule length	4
Lift filtering	1.10

Results

ITEMS

Transactions	150
Counting items	
All items	126
Filtered items	3
Counting itemsets	
Rules	
Number of rules	**0**

RULES

Number of rules : 0

N°	Antecedent	Consequent	Lift	Support	Confidence

Figure 1: Tanagra's stream diagram interface performing association rule mining on an irises dataset.

the general capabilities of PolyAnalyst. PolyAnalyst also seems to use a stream diagram paradigm, but appears to be more similar to Tanagra than Weka.

2 Data Import/Export

2.1 Importing Data

2.1.1 Import Data Formats

Tanagra can import text files with whitespace delimited fields. Each record or transaction appears on its own line. The information within each record or transaction is separated by tabs. Field labels appear tab-delimited on the first line. A decimal point in a number may be indicated either by a '.' or a ',' depending on your Windows setup. It appears that categorical labels must not have whitespace within them.

2.1.2 Compatibility with other formats

While Tanagra cannot directly read or write other, more complex formats, there is a conversion tool available on the Tanagra website to convert from ARFF files, the format used by Weka. Unfortunately, Tanagra cannot directly read from other formats or integrate databases or data warehouses.

This is a deficiency in Tanagra which is not shared by a number of other applications. Weka, for example, is able to load data directly from databases supported by the Java JDBC package, as well as flat comma-separated value (CSV) files and files in it's own ARFF format. PolyAnalyst claims to be able to load data from popular RDBMSs through OLE and ODBC, as well as Microsoft Excel, IBM Visual Warehouse, and Oracle Express formats.

2.1.3 Method for Importing Data

Importing data in Tanagra is a relatively simple operation and is performed at the very beginning of a project due to the use of the stream diagram model. An import data dialog allows a user to select a text file on the local hard drive, and import it. As discussed in the description of the stream diagram paradigm, a tree is then rooted at the imported data, and operations built and performed on that tree.

2.2 Managinging Data Inside a Project

2.2.1 Using Multiple Data Sources

Unfortunately, Tanagra cannot import multiple data sources within one project. This might be desirable in the case of our hypothetical company if we were to have, for example, different data sets from different regions containing the same type of record. One might wish to import data sets from multiple regions and use them together in a data mining analysis. Tanagra would not allow this, because to do so would violate the tree nature of the stream diagram. To perform operations on an amalgam of all the regional data sets, one would first have to merge those data sets outside of Tanagra, and then import them. This could be significantly inconvenient. However, as Tanagra is open source, it seems possible to add additional data set manipulation tools to Tanagra that might merge multiple data sets, producing one data set that could then be imported as the root of the stream diagram tree. This still could not be done further down the tree, so if one wanted to deal in different data sets within one project one would have to merge them all, import them, and then filter out the parts desired at the stream diagram tree nodes.

Other data mining solutions are more flexible than Tanagra with respect to multiple data sources. Weka for example, while not allowing for the combination of data sets, allows multiple data sets and information flows to be created and run concurrently in it's KnowledgeFlow environment.

2.2.2 Transforming Data

Some data transformation can be performed in Tanagra. Tanagra allows a user to transform continuous variables to discrete variables using equal-frequency, equal width, or supervised univariate discretization. This then allows some kinds of analysis only possible for discrete variables. The data can be split into training and testing sets for use in the supervised learning functions. A user can input a standardization function to standardize the data, potentially producing more meaningful analyses. Lastly, new attributes can be created from continuous numerical attributes using an algebraic function input by the user.

Weka and PolyAnalyst have similar functionality, and it would take a specific task to determine if a particular package has the filters and transform operations necessary. However, each has a wide array of operations, so any basic task is probably possible in a particular application. PolyAnalyst even touts a Visual Rule Assistant that allows the user to claims to significantly help the user construct custom data transform functions to transform.

2.2.3 Sampling

Tanagra supplies functions to sample the data before an analysis in several ways. One may randomly sample, or stratified sample, a number of examples in a data set. To allow complementary analysis, there is also a function to recover sampled examples that switches inactive and active samples to the opposite activity status. This sampling has obvious uses in data mining large data sets. Tanagra does not appear provide an utility for under or oversampling, which might be useful to compensate for severely unbalanced classes in a classification attempt. Weka and PolyAnalyst also include random sampling functionality.

2.2.4 Data Cleaning

Tanagra has no data cleaning capability found in either testing the program, or reading the documentation.

Weka and PolyAnalyst both support a number of filtering operations, which may or may not assist in data cleaning. PolyAnalyst allows a number of data cleansing operations, including exceptional record detection and removal. It also claims to contain a drill-through feature that allows interesting data to be easily selected visually, which may help in processing only useful data.

2.3 Exporting Data

The data exporting capability in Tanagra can be found, somewhat confusingly, under the section of data visualization functions. Exporting allows one to save the current data to be used later, or used by another program. This might be particularly useful if one discretized variables and wanted to save them in their new form, or if one wanted to save only certain samples or variables. The export function does not, however, provide much utility for exporting to formats used by other data mining tools.

Most packages can save to the same file formats that they can load, and this is reflected in the fact that Weka can save to flat CSV and ARFF files, as well as directly to a database. PolyAnalyst similarly is able to output to any file format it can read.

3 Analyses Available

3.1 Assocation Rule Mining

Tanagra provides three possibilities for association rule mining - a supervised learning rule miner and two versions of apriori. While association rules are produced for some data sets they cannot be produced for others. This is because the implementation of apriori does not allow for input above a certain size. Rakotomalala provides a warning to this effect in the online documentation for Tanagra. For very large data sets, the association rule mining in Tanagra is ineffective. Sampling could be used to to cut down the size of an input dataset, however his could be an issue for our fictional company depending on the size of our data sets and the parameters set for apriori. Weka also contains an apriori implementation that doesn't appear to have limits, and PolyAnalyst contains an association rule miner, although the algorithm used is not stated (possibly proprietary), and the process is simply referred to as market basket analysis.

3.2 Clustering

Tanagra includes a few basic clustering algorithms. These include k-means, Kohonen's Self Organization Map (SOM), Kohonen's Learning Vector Quantizers (LVQ), and a hybrid Hierarchical Agglomerative Clustering algorithm (HAC). Weka also supports a few basic algorithms, including k-means, expectation maximization (EM), and Cobweb. PolyAnalyst's support seems more limited with respect to the variety of clustering approaches, however information about it's capabilities on the website is fairly sparse.

3.3 Classification

The basic classification, or supervised learning, algorithms implemented in Tanagra include binary logistic regression, k-nearest neighbor, a neural network trained with back-propagation, Quinlan's ID3, linear discriminant analysis, and a naïve Bayesian classifier. Different algorithms are applicable to different types (continuous, discrete, or both) of input data. In some cases the details of how the classification is performed is displayed. For example, the decision tree created by ID3 is shown, which may be useful if our company needs to know the exactly how data is being classified. Additionally, the tree could be extracted and implemented in a real-time classifier for new data if the company required.

Meta-supervised learning is also supported. As well as being able to run any of the classification algorithms once individually, Tanagra includes support for arcing, boosting, and bagging classifiers. These algorithms in general operate on a classification algorithm and run it multiple times manipulating algorithm parameters or input data weight to increase the accuracy of the classifier.

Two learning performance evaluators are included with Tanagra. The first simply splits a dataset into training and test data, while the second performs cross-validation using folds. Evaluation is usually described by accuracy, error, precision and recall rates. A standard confusion matrix is also displayed, allowing for quick inspection of how well a classifier works. Slightly counter-intuitively, these evaluation objects are placed below the classification object in a sub-node. It might make more sense to place the evaluator before the classifier as it manipulates the input data set, not the output. However, it makes sense for evaluation to come afterwards, as a process has to take place before evaluation can occur, so perhaps it is more a minor flaw of the stream diagram paradigm itself.

Weka has very similar support for classification, meta-supervised learning, and evaluation. However, it supports two or three times as many different algorithms in each category. This would be much more useful to someone experimenting with classification as a larger number of algorithms could be tested. PolyAnalyst also supports classification, mostly by using it's own algorithms. There is a lack of specific detail on the PolyAnalyst website, and a more thorough investigation would be required to fully understand it's classification capabilities.

4 Visualization

4.1 Visualization of Data

Tanagra allows a number of different options for visualizing data. A user may view the data directly in a table. Various scatter-plots and graphs can also be produced, but not with particular ease. It

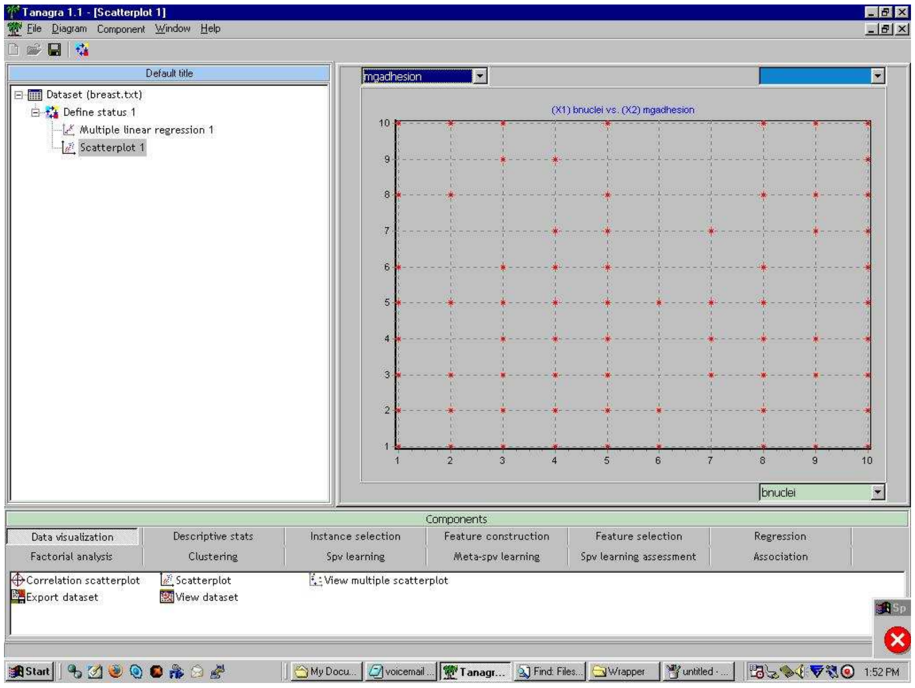


Figure 2: Tanagra scatterplot of a breast cancer dataset.

is sometimes difficult to tell which visualizations can be applied to a given data set. Most require few or no parameters, but must be applied as a sub-node not to the data set itself, but instead to a defined "Define Status" component that specifies the desired variables to view or analyse. This can be seen in Figure 2, where a scatterplot has been drawn on the "Define Status" of a data set on breast cancer data. While the visualizations might be useful to decision-makers in our fictional company, there are potentially useful visualizations missing. There is a distinct lack of histograms, or visualizations in more than two dimensions. Visualizations of results might be more useful.

Weka has more support for histograms and easy visualization of data, but is similarly limited to two dimensional data plots. PolyAnalyst appears to have support for more complicated and specialized graphs that extend into the third dimension.

4.2 Visualization of Analysis Results

Results of analyses in Tanagra are primarily visualized in a two dimensional graph. Colour can be used to differentiate between a class attribute, or clustering or classification result. For data with high dimensionality, two dimensions can be selected for display. It is difficult to intuitively tell if clusters or classification results are meaningful. This may not be useful to a decision-maker with limited statistical knowledge, or one who wants a quick "feel" for the data.

Additionally, results and statistics are often displayed textually. Interpretation of these results requires knowledge of the parameters, abbreviations, and conventions of each analysis algorithm. Again, this may not be useful to the higher-level decision-maker.

As with visualization of data, Weka seems to be about on par with Tanagra, and PolyAnalyst appears to have more detailed visualizations.

5 General Usability

The stream diagram paradigm of data manipulation is shared by other data mining applications such as Weka and PolyAnalyst. Except for the single data-source limitation, it seems reasonably intuitive, and certainly made it easy to understand how data was flowing once the desired tree was set up.

Setting up the desired tree, however, was slightly more difficult. Not all operators are applicable at all nodes of the tree. There is nothing in Tanagra itself to conveniently indicate this. If one drags an operator to an inappropriate location and tries to execute it, it fails, but without a really useful message. It would be helpful if Tanagra did not allow you to drag operators onto inapplicable stages of the tree, or had local documentation describing which operators should be applied where.

6 Support, Documentation, and Error Messages

As Tanagra is not commercial software, support is likely to be minimal, and entirely at the convenience of the author. This might be a problem for our fictional company. As the source code is available, all questions could theoretically be answered by an in-house expert devoted to understanding and working with Tanagra. It is doubtful as to whether we would wish to maintain such a person. There is little help available - the Help option opens a browser to the Tanagra website. There are a number of tutorials available, which were found to be helpful in starting up in Tanagra.

There is a reasonable amount of documentation online, with appropriate citations to algorithms used for particular functions. The only major difficulty in the documentation is in its layout. As it is on many different pages organized by topic, it can be difficult to find the appropriate section if one does not know exactly where to look. A searchable documentation would be better. If the documentation were all combined into one file one could word search on that file.

Error messages are not very useful in Tanagra. If one attempts to execute an operation placed at in incorrect node in the stream diagram tree, it reports that execution has failed, and suggests looking at the attributes. For a novice user, it could be difficult to quickly determine the cause of the error. Popup error messages are predominantly of the type useful only to the programmer - for example informing the user that there has been a negative array index - and are mostly in French.

Weka is similarly an open-source application, and has corresponding levels of support and documentation. Incomplete documentation is available for most of the interface. PolyAnalyst, on the other hand, is a commercial application. It has tutorials that are oriented towards business applications (such as fraud detection and market basket analysis)

7 Conclusion

Whether or not this tool would be useful to our fictional company depends primarily on what exactly it will be used for. Tanagra is open source software designed primarily for research use. Tanagra is free, which is definitely a positive for most companies. If our company has the ability to program any extensions required, then Tanagra may be adequate, as it is open source, and we can add any extra functionality we want. Weka is also open-source and research oriented. However, it supports a much larger set of operations for filtering, clustering, classification, and other analysis techniques. It also has support for more file formats and database access for import/export. Additionally, it is written in Java, and is therefore useable across multiple platforms, which may be helpful. If our company's use is research, or small data set exploration, then one of these free solutions. may be an acceptable choice. If this is the case, we would recommend Weka over Tanagra for it's larger support for file formats, operations, and multi-platform compatibility.

However, if our company will make important decisions based on our analysis, will be using large data sets, or is less experienced with data mining then these research-oriented solutions are likely the wrong choice. They have little support, and are lacking in the ability to deal with, data warehouses, large data sets, or data cleaning. It may be more useful to the fictional company to pay for a more comprehensive and supported commercial package, such as PolyAnalyst, especially if the people who will be using the software are less technical and more interested in making business decisions.

The features of a data mining application that it "good" are somewhat hard to define. In general, it is desirable for applications to support a variety of import and export file formats and databases, so that time is not required to prepare data in a particular format. Additionally, support for a wide range of different filters, transformations, and analysis algorithms is helpful so that comparisons can be made to find the optimal analysis techniques. To the extent made possible by the underlying algorithms, the program should be as transparent so that the user can understand how decisions are made instead of trusting a black box. Finally, output and visualization should be understandable to the user. As described above, these guidelines will change for different uses, and the requirements of a "good" data mining application will be different for the higher level business executive decision-maker, and the graduate student performing research into data mining algorithms.

References

[1] Tanagra - A free data mining software for teaching and research. `http://eric.univ-lyon2.fr/~ricco/tanagra/en/tanagra/`.

[2] Weka 3: Data Mining Software in Java. `http://www.cs.waikato.ac.nz/ml/weka/`.

[3] PolyAnalyst 4.6. `http://www.megaputer.com/products/pa/index.php3`.