Working Paper Series
ISSN 1170-487X

UNDERSTANDING WHAT MACHINE LEARNING PRODUCES
Part II: Knowledge Visualization Techniques

by Matt Humphrey, Sally Jo Cunningham and Ian H. Witten

Working Paper 96/22
October 1996

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

UNDERSTANDING WHAT MACHINE LEARNING PRODUCES

Part II: Knowledge visualization techniques

Matt Humphrey, Sally Jo Cunningham and Ian H. Witten
Department of Computer Science, University of Waikato,
Hamilton, New Zealand.
{matth, sallyjo, ihw}@cs.waikato.ac.nz

Abstract—*Researchers in machine learning use decision trees, production rules, and decision graphs for visualizing classification data. Part I of this paper surveyed these representations, paying particular attention to their comprehensibility for non-specialist users. Part II turns attention to knowledge visualization—the graphic form in which a structure is portrayed and its strong influence on comprehensibility. We analyze the questions that, in our experience, end users of machine learning tend to ask of the structures inferred from their empirical data. By mapping these questions onto visualization tasks, we have created new graphical representations that show the flow of examples through a decision structure. These knowledge visualization techniques are particularly appropriate in helping to answer the questions that users typically ask, and we describe their use in discovering new properties of a data set. In the case of decision trees, an automated software tool has been developed to construct the visualizations.*

Decision trees, production rules and decision graphs are widely used for representing the results of machine learning. As the first part of this paper showed, many schemes have been developed to assist comprehension by reducing the amount of gratuitous information in such representations. What we do now is consider how methods of visualization can further help people to assimilate the information that is generated by applying machine learning methods.

We begin by reviewing two ways in which visualization has been applied to machine learning. The first is the use of decision tables, a kind of visualization for rules that presents a group of rules side by side in a way that allows them to be compared. The second is a tree visualizer with enticing visual effects: users "fly over" a 3D representation of the tree, circle in on a particular node and soar off along a branch.

Are these schemes useful? The purpose of a visualization is to provide answers in situations where the user does not know, or cannot decide, the questions they want to ask (Bertin, 1983). Visualizations are useful when they support the user's tasks, when they help the user answer questions about the data, and when they prompt him or her to pose pertinent queries. To design appropriate visualizations, it is necessary to determine the sorts of questions to which researchers seek answers.

Knowledge workers use decision trees, rules, and graphs to grasp the overall structure of a classification scheme. The standard representations of these constructs show how particular *decision criteria*—selections of data attributes with specific values—lead to one of a predefined set of *classes*—subsets of the example space. But in our experience, users of machine learning also ask a number of critical questions about the significance of decision criteria, the flow of training examples, and the extent to which classes are related, which cannot be answered from the standard visual forms of classification structures.

These questions can be mapped into equivalent *visualization tasks*, or actions that a user can perform on a visualization in order to answer the question. Examples are looking up a value, or comparing two values. Given a diagram of an information structure, a visualization task is *possible* when the information that it calls for is visible in the diagram. The task is *efficient*

when the arrangement of the information allows the eye to perform the associated action quickly, easily, and accurately.

New kinds of visualization can be defined in order to help users interpret the information presented. By discovering questions that users need to answer, and mapping these onto visualization tasks, we have created new styles of knowledge representation for decision trees and for decision graphs. These provide the user with more support than does the traditional tree or graph representation.

1 Existing machine learning visualizations: two examples

Visualization techniques have not yet been widely applied to the knowledge structures produced by machine learning. This section reviews two applications that have been described in the literature.

Decision tables

Decision tables have been used in a variety of contexts in computing. For example, they have been suggested as an alternative programming language for scientific programming (Lew, 1983), as a guide for testing and debugging programs (Goodenough and Gerhart, 1975), as an interim representation for simplifying correctness proofs (Lew, 1984), as a language or design aid for rule-based expert systems (Maes and van Dijk, 1988), and as a general decision making aid (Gregory, 1988). It is surprising that such a common decision/knowledge visualization structure has only recently been explored as an output representation technique for machine learning (Kohavi, 1995).

An example of the conventional decision table representation is shown in Table 1. The first column is labeled first with all rule condition identifiers, and second with all possible rule conclusions, or classifications. Each remaining column corresponds to an individual rule, with column entries indicating its condition/classification values. To use a decision table for performing a classification, it is searched for a set of condition values that exactly match those of the data item. In the classic decision table, at most one column will match; if there is none a default classification is assigned.

A major advantage of the decision table over the rule or tree format is its extreme compactness on a physical page. It is not difficult to squeeze fifty or more rules containing a dozen attributes onto a single sheet of paper (Catlett, 1995). The columnar representation also facilitates comparison of attributes and conditions. However, the relationship between rules is not represented; neither is the relationship with the original data from which the rules were derived. Usability tests comparing tree, table, and rule representations provide weak evidence that people prefer reading and composing decision tables to rules, even though the

Conditions								
Interest		down		up	up			down
Dollar	up		down			down	up	
Profit	up			down	up	down		
Auto Sales		down	up				down	up
Budget deficit	down		up		up	up		
Trade deficit		up		down			down	up
Actions								
Fund 1	sell	buy	buy	sell	buy	buy	sell	buy
Fund 2	sell	buy	buy	buy	sell	buy	buy	sell
Fund 3	buy	sell	sell	buy	buy	sell	buy	sell
Fund 4	buy	sell	sell	buy	buy	sell	sell	buy

Table 1 A decision table for fund investments (from Subramanian *et al.*, 1992)

subjects could themselves produce rules that were significantly more correct than the corresponding tables they composed (Halverson, 1993); and that people may be able to more quickly and accurately perform classifications using decision trees rather than tables (Subramanian *et al.*, 1992). More comprehensive empirical tests are required in order to make firm conclusions about the relative usability of these visualizations.

Kohavi (1995) induces a decision table format that may include multiple exact condition matches for a data item. In this case, the classification assigned is the class of the majority of those matches. If no matches are found for a data item, then the majority class is selected as a default. However, this table format has additional drawbacks with respect to comprehensibility. The statistical classification method in cases of multiple rule matches can be difficult for humans to interpret semantically, and this problem grows worse as the number of rules participating in a vote increase (Corruble *et al.*, 1995). Moreover, because of this statistical component, individual rules cannot be meaningfully examined apart from their context in the table.

Three-dimensional visualization of decision trees

A completely different approach to visualizing knowledge structures is to present a three-dimensional "virtual reality" view which the user can "fly" through, zooming in on interesting portions or pulling back for a bird's-eye view of the structure as a whole.

Figure 1 shows the MineSet tree visualizer, provided by Silicon Graphics for the MLC++ machine learning toolbox. The visualizer is based on an existing system for viewing the structure of computer files and directories. In this example a decision tree derived from the breast cancer database of the UCI machine learning repository is being viewed. The amount of training data that is matched by each node in the tree is represented by the height of the

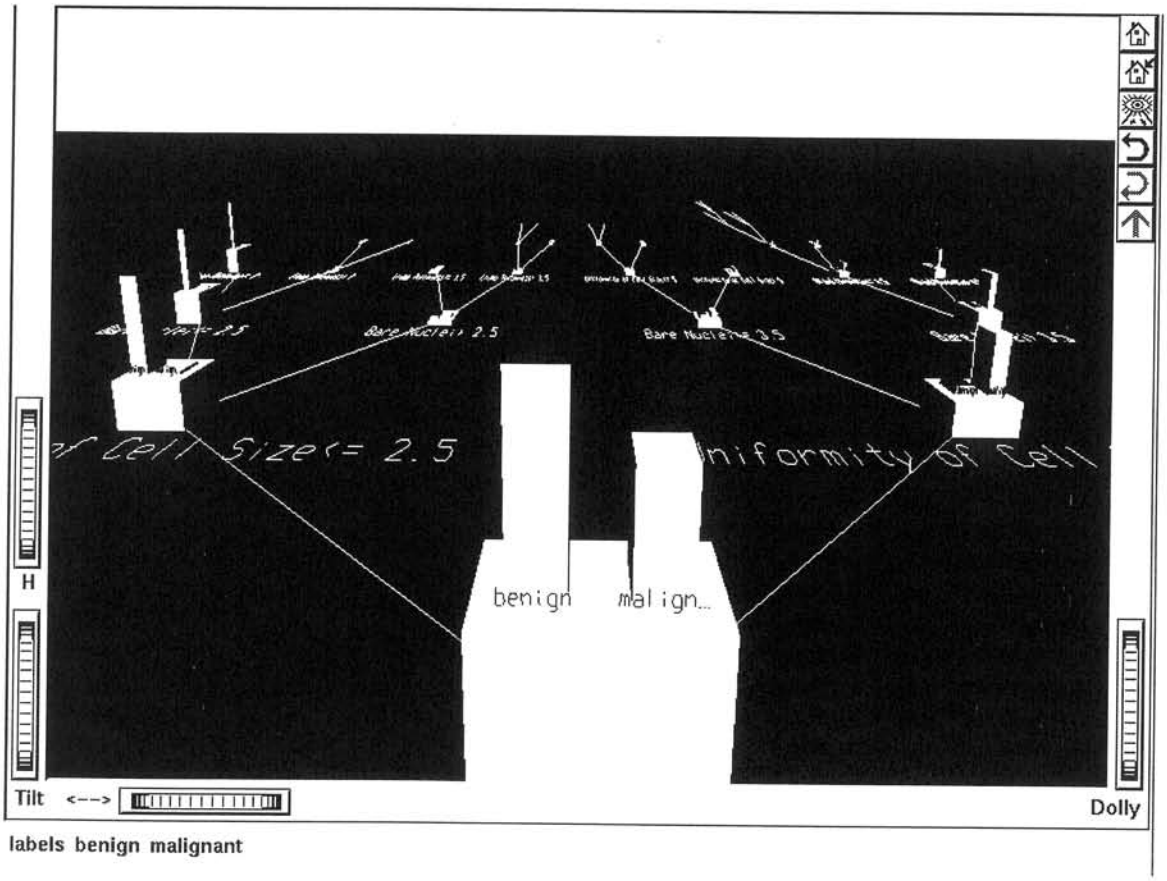

Figure 1 Tree visualizer (from `http://www.sgi.com/Technology/mlc/images/bc.gif`)

"towers" representing the nodes. The color and label of a tower identifies the classification assigned to its data items.

Although the interaction technique is eye-catching, it is not clear that the visualization scheme is more effective in conveying information than more conventional two-dimensional representation schemes.

2 Visualization tasks

To ensure that the visualizations we design are useful and productive, it is necessary to examine how people might use them. There has been extensive research on determining efficient and effective means for extracting information from graphic images, although by the very nature of the problem the outcomes tend to be couched as alternative approaches that are hard to rank in any definitive way.

The actions, both mental and physical, that people perform with visualizations in order to extract their meaning are termed *visualization tasks*. These have been construed variously as high-level, task-specific actions, as a means for providing visual functionality, and as generalizations of user goals. We examine them with an eye towards determining a useful framework for characterizing machine learning tasks.

2.1 High-level tasks for spreadsheets

High-level visualization tasks are ones that directly support user goals. Sparrow (1989) defines six high-level tasks for determining particular kinds of information from spreadsheets: specifics, limits, conjunctions, accumulations, trends, and proportions. *Specifics* are accurate representations of data values in the set. *Limits* are the extreme data values of a particular variable. A *conjunction* is the pairwise correspondence of two values, such as time versus money. *Accumulations* are sums, averages, or other composite values. *Trends* relate how one variable corresponds to another over its range of values, such as the trend of profits in a bar chart. Trends often imply a time component, but may apply to non-temporal variables, as in trends of supply versus demand. *Proportions* relate the relative size of variables, as in the archetypal pie chart.

Construing visualization tasks at a high level permits a direct correspondence between the user's goals and the graphic representation of the data. It becomes easy to determine if the graphic supports the goals. However, Sparrow's tasks are not general—they depend heavily on the spreadsheet-like nature of the underlying data.

2.2 A functional approach

Visualization tasks can also be interpreted as a means of providing functionality. Roth and Mattis (1990) consider six tasks: accurate lookup, comparison of values within (but not among) relations, pairwise (or *n*-ary) comparison, distribution of quantity, functional correlation, and indexing. These tasks are at a lower level than Sparrow's, and are not so strongly tied to a data model, although they have been defined with a relational model in mind.

The *accurate lookup* task is similar to Sparrow's "specifics". *Within-relation comparison* refers to the determination of the limits and ordering of a data variable. *Pair-wise (or n-ary) comparison* among relations relates values from distinct variables. Although this may seem contradictory (how can money be compared to time?), it actually refers to how values correspond within a tuple of a relation, in a similar manner to Sparrow's "conjunction". Determining the *distribution of quantity* involves considering how a value has been partitioned among several visual elements, or how several distinct elements correspond to a whole unit, an operation that takes in both accumulation and proportion. A *functional correlation* is a mathematical relationship between two or more variables. The high-level goal of many users is to grasp correlations in the data—to what degree do some variables influence others? Finally, *indexing* refers to the use of a visual correspondence to map a value

of one variable to a value of another, for example looking up a profit value for a particular year.

Even though their tasks have much in common, Roth and Mattis's framework is simultaneously more general and lower-level than Sparrow's. Their tasks cover more cases and can be applied to many kinds of visualizations, in addition to those produced by spreadsheets. Nevertheless, it is more difficult to describe how a user performs their tasks on a particular visualization—the users goals must be translated into the terms of these tasks.

2.3 Low-level tasks

Visualization tasks can be defined increasingly more generally at an even lower level. In his work on automatically generating visualizations to support particular user tasks, Casner (1991) postulates two basic, low-level tasks: computation and search. These are actually general classes of task and must be "refined" or instantiated for a particular visualization. *Computation* refers to any mental transformation that the user may perform: comparing two or more values, converting a visual effect to a value, constructing conceptual models, and so forth. The power of such a generalization lies in the fact that it does not prescribe the sorts of computations that people may be capable of; it leaves them open. *Search*, on the other hand, is the visual scanning and interpretation of the image. A user searches a visualization for the largest bar or pie slice, and then internally computes the equivalent proportion.

An alternative set of low-level tasks are locating, interpreting and relating (Leung and Apperley, 1993). *Locating* information involves identifying the spatial/visual characteristics of particular values. The user visually scans the image, looking for a value. This task encompasses Sparrow's "specifics" and "limits", Roth and Mattis's "accurate lookup", and Casner's "search", along with some "computation". *Interpreting* information involves translating the appearance of a mark into data values, such as converting a (spatial) point on an axis into the equivalent data value. This is the logical inverse of the "locating" task, because the visual effect is given and the datum is to be derived. It is not apparent how the other techniques describe this task, as it is bound up in the notions of "search" and "lookup". The task of *relating* information involves comparing, computing or otherwise considering one or more data items. To relate a set of values is to determine causal or dependency relationships among them, or to determine which of them compose or contribute to other values, and how. Relationships are often expressed qualitatively rather than quantitatively. The user may be translating one value into another, computing a sum or average, or determining the correspondence between two values. These tasks are similar to Sparrow's "conjunction", "accumulation" and "proportion", and Roth and Mattis's "comparison", "distribution of quantity" and "indexing".

2.4 Discussion

In order to design new, effective visualizations, it is necessary to describe both the high- and low-level tasks. Because we ultimately wish to support users' high-level tasks, we should begin by determining what those are. Understanding the specific tasks focuses and directs the design of supporting visualizations. High-level machine learning tasks are expressed as questions that users might like the visualization to answer. These questions must be construed in terms of low-level tasks so that they can be supported by actual visualizations. In the next section we analyze the sort of tasks that users want to perform and map them on to specific operations that a graphic representation can support.

3 Machine learning questions

The goal of machine learning, insofar as this paper is concerned, is to produce and verify techniques that automatically classify data. Visualizations are not the end-product of such systems, they are an intermediary that helps users assess the structure of the information that is generated.

Existing representations of the results of machine learning, including the visualizations described in Section 1, address questions about the structure of decisions that leads to a given classification, and the circumstances under which this structure produces a unique prediction. They give an overall picture of how decisions might be made. However, they do not relate decision information to the training examples from which it was inferred, and we find that users are keenly interested in relating the decision structure to the training information.

In our experience, users are likely to focus on questions such as those below. These questions frequently appear in various guises in articles that discuss applications of data mining. We couch the operations involved in answering such questions in terms of decision trees, although they pertain equally well to decision graphs too. Some of the issues that we identify relate to the difference between the classes defined in the original learning problem, which we call "distinct" classes, and the classes implicitly defined by the leaves of the tree, which we call "terminal" classes. Several terminal classes may correspond to the same distinct class.

Which attributes have the greatest impact?

One goal of machine learning is to identify predefined distinct classes. The significant attributes are those that have the greatest influence in discriminating between these classes. Such attributes may be near the root of the tree, or may occur at nodes through which many training examples pass, or may identify the largest of the terminal classes. Specific operations that address this question are

- lookup the branch or class size for particular decision criteria;
- search for the largest overall branch or class;
- search for the largest branch that gives rise to a particular class.

Which tests are spurious ?

Not all computer-generated attribute-value tests are relevant. In some cases, tests are created that identify members of the training set perfectly, but do not generalize to the population at large. As discussed in Part I of this paper, many induction methods strive to identify and eliminate such tests; however, no such procedure is foolproof. Small terminal classes, or regions of the tree with many small branches, may involve spurious tests. Specific operations related to this question are:

- lookup the branch size for particular decision criteria;
- search for the smallest overall branch and class;
- search for the smallest branch that gives rise to a particular class.

Which decision criteria are interesting or unexpected?

"Interestingness" and "unexpectedness" tend to be the holy grail of machine learning. Interesting or unexpected decision criteria may be ones involving few examples, or having a distinctive structure. Interesting classes may be large ones that arrive from small branches, or involve a large branch that devolves into several equal partitions on unusual boundaries. Operations that support the detection of such situations include:

- search for large branches that lead to small classes;
- search for small branches that contribute to large classes;
- lookup errors for particular decision criteria;
- relate terminal classes to distinct classes.

What is the structure of the classes?

The class and criteria structure gives a broad understanding of the basic concepts embodied in a decision tree. To make the information apparent, the tree should be laid out neatly or logically, according to some well-formed rules:

- order decision conjunctions hierarchically by level;
- order decision disjunctions sequentially by subclass size;
- relate terminal classes to distinct classes.

How are the training examples distributed?

A decision tree ultimately partitions the training examples into classes and it is useful to know how many of these examples end up in each class. It is also useful to know the number of errors classified by each terminal class. This information can be determined by the following operations:

- lookup the size of particular decision criteria;
- lookup the errors in particular decision criteria;
- order decision criteria by size compared to all others;
- order decision criteria by size compared to their siblings.

Many of these high-level tasks share the same low-level operational components. These low-level tasks guide the design of new visualizations. A visualization that supports them also supports the high-level ones. We have coalesced these tasks into Table 3. The first column shows the general task, taken from Section 2. The second column contains the low-level tasks described above. The next step is to design a visualization that incorporates these primitive tasks.

4 A new decision tree

We have designed a new kind of decision tree to support the visualization tasks that have been identified. Figure 2 shows the representation of a classification decision for different kinds of grasses. This comes from a study whose aim is to determine the mechanisms which influence the persistence of white clover populations in dry hill land, and for this decision tree the predominant type of grass that is already growing on the plot—*huia*, *prop*, or *tahora*—is used as the predicted attribute.

From the top downwards, Figure 2 contains three horizontal regions: an ordered decision tree, a terminal class comparison graph, and a distinct class comparison graph. The upper region is an augmented version of the ordinary decision tree. Each node represents a test on a particular attribute, and the branches correspond to values (or ranges of values) that the attribute could assume. In this case, the nodes are types of grass and branches represent the amount of grass of that type. Nodes without children are terminal classes, but are not necessarily distinct, because the same class may be produced by a variety of decision paths.

Interior nodes also represent classes—subsets of training examples identified by the decision path up to that point. It is here that the tree departs from tradition. Each node is centered in a horizontal bar that is proportional to the number of training examples that it classifies. The root node occupies the full horizontal stretch, because it represents all the examples.

Lookup	branch/class size for particular decision criteria branch/class relative size for particular decision criteria errors for particular decision criteria subclasses of classes
Search	for largest/smallest overall class for largest/smallest relative class
Relate	terminal classes to distinct classes
Order	decision conjunctions hierarchically by level decision disjunctions sequentially by subclass size

Table 3 Specific visualization tasks

Every other node occupies a fraction of its parent's allocation. This makes it possible to see immediately the number of examples that any class contains, relative to the total population, or to the population of the parent, or to any ancestor. For example, *paddock* grass contains about two-fifths of the training examples, and represents about two-thirds of the *OtherGrasses92* class.

Not only is the horizontal space proportional to the number of examples, but the branches themselves are widened. The thicker the branch, the more examples it classifies. Although the information is redundant, it provides a convenient view of paths through the tree. The thickest paths stand out, giving a structural overview of the data.

The branches (and hence children) of each node are ordered in size from left to right. The leftmost branch always represents the largest subclass and the rightmost the smallest. For the root node, it is quite clear that the largest is to the left. However, the children of the *paddock* grasses (third level, leftmost) are more difficult to distinguish.

The middle region, a terminal class comparison chart, situates a column directly below every terminal class. The height of the column is proportional to the number of examples in the class. Placing the terminal classes on the same level allows them to be more easily compared. The horizontal ordering matches the order of the leaf nodes of the decision tree. In this way, it is easy to pick the largest and smallest terminal classes, and to relate them to the structure of the tree.

In this example, the largest terminal class is of type *huia*, arrived at via the path *root* → *Cocksfoot92* → *WhiteClover93*. It is apparent that this is only a secondary branch of the root node: it represents the majority of the root minority. The other sizable node takes the path *root* → *OtherGrasses92* → *Cocksfoot92* and is of type *prop*. Although this path takes the root majority, it is still a minority path.

Each column displays further information. The total number of examples acquired by the class is written at the top of the bar. The number of examples that are incorrectly classified is shown by the darkened portion at the bottom of the bar—this can, of course, never exceed the size of the column. Finally, the left vertical axis indicates the scale of the number of examples, while the right vertical axis indicates the percentage of the total number of examples. Both are labeled with the maximum held by any subclass.

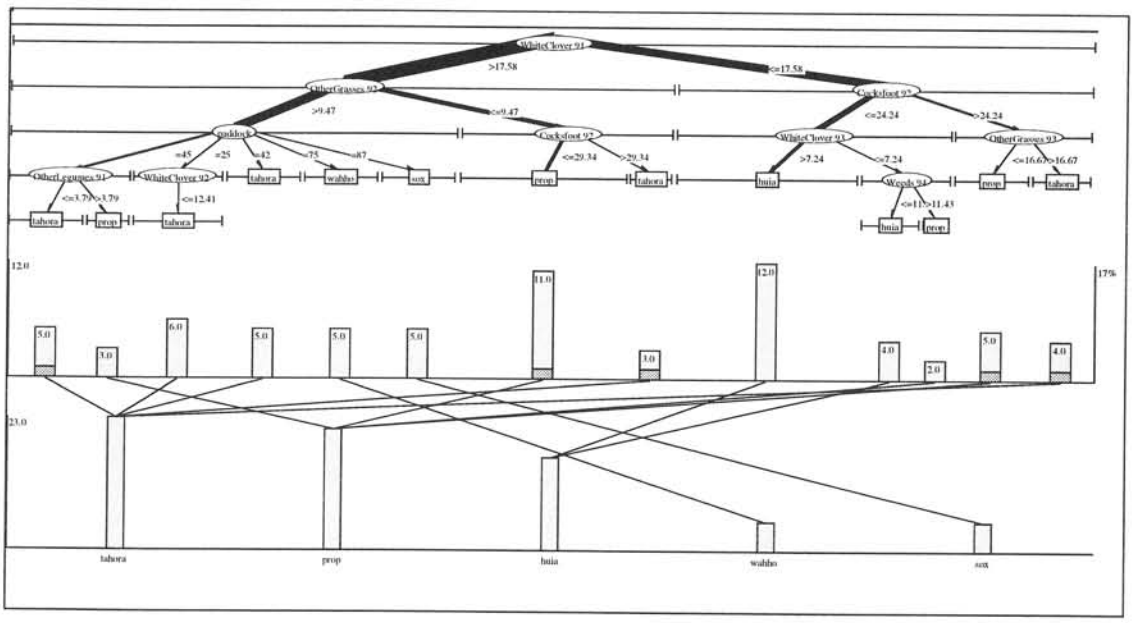

Figure 2 Visualization of classification tree for grasses data

The lower region, a distinct class comparison graph, shows each class as a single column. Columns appear in decreasing order and the vertical axis shows the scale of the number of elements. Each class has edges that connect it back to the terminal classes that comprise it. In this case it shows that *tahora*, the largest class, comprises many terminal classes, while the smallest classes only occur once each. We can also see that the largest terminal class does not become the largest overall class, but rather the third-ranked.

Figure 2 was produced by the Relational Visualization Toolkit, a software package that enables users to design complex visualizations of relational data without traditional programming (Humphrey, 1996). Users draw graphical components of their visualization, termed "graphic relations," and connect them together with relational operators such as select, project and join. The resulting visualizations can be displayed on screen or converted to PostScript.

5 A New Decision Graph

A decision graph is an alternative structure to a decision tree that allows common subparts to be coalesced. For example, Figure 3a shows a graph for a decision structure which requires 39

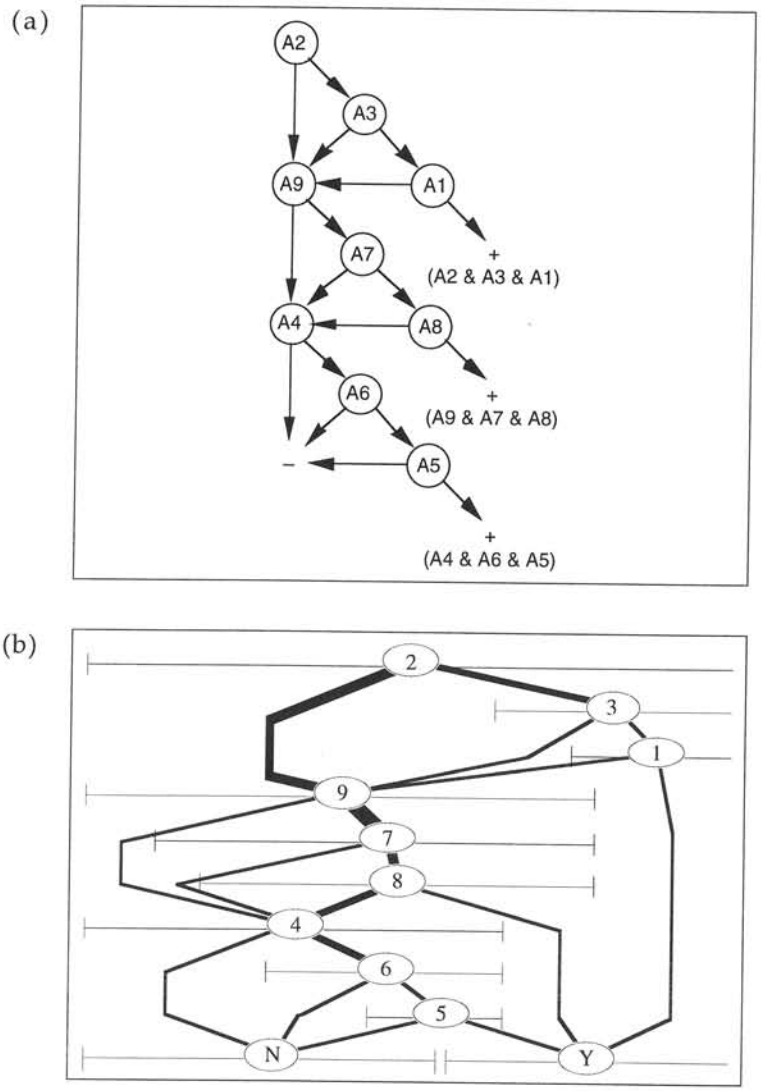

Figure 3 Decision graphs for a problem of Quinlan's
(a) Original graph (after Oliver, 1993)
(b) Restructured for easy visualization

internal nodes and 40 leaves when represented as a tree (Oliver, 1993).

Based on the visual structures of the new decision tree described in Section 4, we can construct a decision graph that is also labeled with counts of the number of training examples. Figure 3b shows this representation for the example of Figure 3a. (Unlike the decision tree above, this figure has not been produced automatically; it was drawn by hand.) This graph contains three new components. As in the tree, nodes are drawn on a horizontal line whose length is proportional to the number of training examples they represent. Subclasses are aligned on a baseline that matches other classes to which they may be compared.

Horizontal node sizes and branch widths are the same as for the decision tree above. Graphs differ from trees, however, because each node has several incoming branches, and the size of a class is the sum of their sizes. Classes are aligned so that each subclass may be easily compared to its parent to determine its relative size. Were the baseline of the subclass not aligned with an edge of its parent, users would not be able to determine the relative size as effectively, for people determine positional proportion most correctly when the objects are aligned.

6 Experience

So far, we have anecdotal evidence supporting the use of the new decision tree format for comprehending the outcome of machine learning techniques. For example, using Figure 2, several characteristics of the structure have been discovered that were not evident from the original decision tree.

- The largest distinct class (*tahora*) is produced from many small terminal classes, and does not incorporate any of the largest terminal classes.
- The largest and second-largest terminal classes are produced by strong paths with few diversions. Nevertheless, these two large classes together account for only one-third of the training examples. The remaining two-thirds are in classes that represent only 4% to 8% of the data.
- Although most of the training examples are directly funneled off to the left, they quickly disperse into several small classes and do not really contribute to the major ones. Nevertheless, half of these classes reconverge to create the largest distinct class, *tahora*.
- The two smallest distinct terminal subclasses are produced uniquely and are not composed of several terminal classes.

Users commented that they felt they could "see" the data more clearly, and with greater structure. They felt more comfortable and confident about the quality of the classification scheme.

While looking at the decision tree on paper, some users wanted to browse the training examples. They wanted to examine particular examples in the context of the decision tree's criteria. Users were very interested in seeing the training examples that were classified by particular decision criteria. For any interior class, terminal class or distinct terminal class users wanted to be able to see the examples that the classification algorithm chose. In addition, they wanted to see the error cases in order to determine exactly which properties were not correctly classified.

One user spoke about "tagging" the original examples in order to see the dispersion, as can be done with statistical data (Becker, 1987). Some item or subset of the data could be tagged and counts of the tagged items could be displayed along the nodes and branches. For example, given a data set that classifies students with respect to their grades in different courses, it might be useful to see how males or females are distributed.

7 Conclusion

The aim of many machine learning users is to comprehend the structures that are abstracted from an empirical dataset, and such users may be far more interested in understanding the structure of their data than in predicting the outcome of new examples. Our experience is that such users are particularly interested in relating decision information to the training data from which it was inferred. Not only is this information concealed by classical representations of decision structures, but the very idea of relating decision structures to training data is anathema to machine learning researchers because of the statistical invalidity of using such information for quantitative evaluation.

By analyzing the questions that users tend to ask of the decision structures inferred from their data, this paper has defined specific low-level operations that a visualization of a decision structure should support. These operations have been used to design a new decision tree visualization technique that makes apparent the relationships between the decision and the data it relates to. Although we have used training data in this paper, because in our experience that is what users want to do in practice, any test data set could be used to provide the information necessary to build the visualization. The production of such visualizations is supported by a software package, the Relational Visualization Toolkit (Humphrey, 1996).

The technique has been extended from decision trees to decision graphs (the extension is conceptual only: it is not yet supported by our visualization software). However, decision graphs tend to be perspicuous only when just two classes are being induced. Complex classifications of more than two terminal classes, or with nodes of outdegree greater than two, will inevitably have lines that cross. The new decision graph structure does not deal with the multi-class case, and new ways to visualize such graphs have yet to be discovered.

Users who are exposed to these visualizations immediately want to interact with them. For example, they want to point to a leaf node and see which examples it represents. Frameworks for interactively browsing visualizations and their underlying data are likely to be a fruitful topic for future research.

References

Bertin, J. (1983) *Semiology of Graphics*, translated by W.J. Berg, University of Wisconsin Press, Milwaukee, Wisconsin, U.S.A.

Becker, R.A., Cleveland, W.S. (1987) "Brushing Scatterplots," *Technometrics*, 29(2); May.

Casner, S. (1991) "A Task–Analytic Approach to the Automated Design of Graphic Presentations," *ACM Transactions of Graphics*, 10(2), 111–151; April.

Catlett, J. (1995) "Rule induction as exploratory data analysis." *IJCAI'95 Workshop on Machine Learning and Comprehensibility*. Available at <URL: `http://www.lri.fr/~cn/web-ijcai/jason.ps`>.

Corruble, V., Thire, F., and Ganascia, J-G (1995) "Comprehensible exploratory induction with decision graphs." *IJCAI'95 Workshop on Machine Learning and Comprehensibility*. Available at <URL: `http://www.lri.fr/~cn/web-ijcai/vincent.ps`>.

Goodenough, J., and Gerhart, S. (1975) "Toward a theory of test data selection," *IEEE Transactions on Software Engineering* SE-1(2), 156–173.

Gregory, G. (1988) *Decision Analysis*. Pitman Publishing, London.

Halverson, R. (1993) "An empirical investigation comparing IF-THEN rules and decision tables for programming rule-based expert systems." *Proceedings of the 26th Hawaii International Conference on System Sciences*, vol. III, 316–323.

Humphrey, M.C. (1996) "A Graphical Notation for the Design of Information Visualisations," D.Phil Thesis, University of Waikato, Hamilton, New Zealand.

Kohavi, R. (1995) "The power of decision tables," *Proceedings of the European Conference on Machine Learning (ECML)*. Available at <URL: ftp://starry.stanford.edu/pub/ronnyk/tables.ps>.

Leung, Y.K., Apperley, M.D. (1993) "E^3: Towards the Metrication of Graphical Presentation Techniques for Large Data Sets," In *Lecture Notes in Computer Science #753*, edited by Goos & Hartmanis, pp. 125–140.

Lew, A. (1983) "Decision tables for general-purpose scientific programming," *Software Practice and Experience* 13, 181–188.

Lew, A. (1984) "Proof of correctness of decision table programs," *Computer Journal* 27(3), 230–232.

Maes, R., and van Kijk, J. (1988) "On the role of ambiguity and incompleteness in the design of decision tables and rule-based systems," *Computer Journal* 31(6), 481–489.

Roth, S., Mattis, J. (1990) "Data Characterization for Intelligent Graphics Presentation," *Proceedings of the SIGCHI '90 conference*, CHI '90, April, pp. 193–200.

Sparrow, J.A. (1989) "Graphical displays in Information Systems: some data properties influencing the effectiveness of alternative forms," *Behaviour and Information Technology*, 8(1), 43–56.

Stone, H. S. (1973) *Discrete Mathematical Structures and Their Applications*, SRA Computer Science Series.

Subramanian, G.H., Nosek, J., Raghunathan, S.P., and Kanitkar, S.S. (1992) "A comparison of the decision table and tree." *Communications of the ACM* 35(1), 89–94.

Tufte, E.R. (1983) *The Visual Display of Quantitative Information*, Graphics Press.

Tufte, E.R. (1991) *Envisioning Information*, Graphics Press.