# A Visualization Tool for Interactive Learning of Large Decision Trees

Trong Dung Nguyen, Tu Bao Ho and Hiroshi Shimodaira
Japan Advanced Institute of Science and Technology
Tatsunokuchi, Ishikawa, 923-1292 JAPAN
{nguyen, bao, sim}@jaist.ac.jp

## Abstract

*Decision tree induction is certainly among the most applicable learning techniques due to its power and simplicity. However, learning decision trees from large datasets, particularly in data mining, is quite different from learning from small or moderately sized datasets. When learning from large datasets, decision tree induction programs often produce very large trees. How to visualize efficiently trees in the learning process, particularly large trees, is still questionable and currently requires efficient tools. This paper presents an visualization tool for interactive learning of large decision trees, that includes a new visualization technique called T2.5D (stands for Trees 2.5 Dimensions). After a brief discussion on requirements for tree visualizers and related work, the paper focuses on presenting developing techniques for the issues (1) how to visualize efficiently large decision trees; and (2) how to visualize decision trees in the learning process.*

## 1 Introduction

Learning decision trees from large datasets is quite different from small or moderately sized datasets. We have to face with new problems that we may never see previously such as converting trees into rules or understanding, accessing, and manipulating large trees, etc. For example, on a workstation Alpha 21264 (OS: Digital UNIX V4.0E; Clock frequency: 500MHz; RAM: 2GB), in our experiments the well-known system C4.5 [13] – from the U.S. census bureau dataset consisting of 199,523 instances with 32 symbolic attributes and 8 numeric attributes, and size of 103 MB [10] – produces a pruned tree after ten minutes but cannot convert the tree into a set of rules after two days.

Though decision trees are a simple notion, we can understand their content and hierarchical structure easily if they are small but cannot understand or understand difficultly if they are large. Research on visualization of decision trees has recently received a great attention from the KDD (knowledge discovery and data mining) community because of its practical importance. Many works have been done, e.g., the 3D Tree Visualizer in system MineSet [3], CAT scan (classification aggregation tablet) for inducing bagged decision trees [14], the interactive visualization in decision tree construction [1], the tree visualizer with a tree map in system CART [2] of Salford Systems, etc. However, it is still difficult to view and navigate large trees with these systems. On the other hand, new approaches in information visualization field for representing large hierarchical structures, e.g., cone trees [15], hyperbolic trees [9], have not been well considered in AI, machine learning and data mining.

This paper concerns with interactive visualization in learning decision trees, in particular large decision trees. The work was motivated by the need of an efficient tree visualizer when we apply our system CABRO [11] to large datasets. Section 2 of this paper addresses the requirements for tree visualizers and related works. Section 3 presents the tree visualizer of our system CABRO. Section 4 describes how this tree visualizer is used in the learning process. The concluding remarks summary features of several tree visualizer including our T2.5D, the conclusions and future work.

## 2 Visualization requirements and related work

The decision tree structure contains two kind of information: structural information associated with the tree, and content information associated with each node. Based on our DTI task analyses, we specify the primary requirements for tree visualizers, which share many common points with [7], as follows:

- *Embedded in the DTI process*: Structural and content information should be available not only after inducing decision trees but also during the induction process. The tree visualizer must provide the ability to



1082-3409/00 $10.00 © 2000 IEEE

```
leaves = norm:
|   seed-discolor = absent:
|   |   temp = lt-norm: rhizoctonia-root-rot (19.0/1.3)
|   |   temp = norm: anthracnose (24.0/1.3)
|   |   temp = gt-norm: anthracnose (0.0)
|   seed-discolor = present:
|   |   canker-lesion = dna: diaporthe-pod-&-stem-blight (5.5/1.2)
|   |   canker-lesion = brown: purple-seed-stain (0.0)
|   |   canker-lesion = dk-brown-blk: purple-seed-stain (0.0)
|   |   canker-lesion = tan: purple-seed-stain (9.0/1.3)
...
```

**Table 1. Part of decision tree displayed by C4.5**

quickly access and manipulate a large tree according to operations in decision tree learning.

- *Comprehension*: The tree visualizer must facilitate the understanding of tree structure and its relation to nodes in focus.
- *Efficiency*: Efficient use of space and visual tools is essential for the tree visualizer to deal with large trees.
- *Interactivity*: Interactive control over the structure and the ability of users to customize the layout to meet their current needs and interests are essential.
- *Esthetics*: Drawing and feedback must be esthetically pleasing.

Most decision tree learning systems employ visualization methods that belong to one of two categories: outlines or node-link diagrams.

The *outline* methods to represent trees are similar to that of PC Shell under DOS or Microsoft Windows. Basically, the number of display lines required in the outline method is equal to the number of nodes in the hierarchy, but with the zooming functions the tree can be reduced dynamically to a smaller one with size is linearly proportional to the number of nodes. However, this method is somehow poor to associate with graphic operators in order to deal with large trees. System C4.5 and its successor C5.0 are in this category. Table 1 shows a portion of a decision tree displayed by C4.5 from the small soybean dataset [10]. From the large U.S. census bureau dataset, C4.5 produces a decision tree of nearly 18,500 nodes with 2624 leaf nodes (about 1,850 times larger than this portion). It is extremely difficult, even imposible, to understand such a big tree in this outline representation.

The *node-link diagram* has a main advantage of being easy to understand as it is a natural mapping of structured relationships. However, the original node-link diagram does not use efficiently the available display space and therefore is not adequate for large trees. Different attempts have been done to overcome this limitation which can be roughly classified into two groups of 2D and 3D visualizers.

Many tree learning systems employ 2D tree visualizers. CART (see [2] – the widely used system of Salford Systems – has a 2D tree visualizer associated with a *tree map* that provides an overview of the tree. Another 2D tree browser having good features of multi-level dynamic queries and pruning is developed in [8].

Two special 2D tree visualizers are that of hyperbolic trees and treemap. The *hyperbolic tree* method at Xerox is excellent for visualizing and manipulating large trees. It lays out the hierarchy in a uniform way on a hyperbolic plane and map this plane onto a display region. The hyperbolic browser can display more than 1000 nodes of which about 50 nearest the focus can show from three to dozens of characters of text [9]. However, it also meets the problem of presenting partially trees for large tree of thousands nodes, and it is somehow not a natural way in learning decision trees. The *treemap* visualization method is a special 2D visualizer that makes 100% use of the available display space, mapping the full hierarchy onto a rectangular region in a space-filling manner [7]. This efficient use of space allows very large hierarchical data to be displayed in their entirety. However it tends to obscure the hierarchical structure of the values and provides no way of focusing on one part of a hierarchy without losing the context.

Among typical 3D tree visualizers are that of MineSet (SGI) [3] and the *cone tree* method at Xerox [4]. Mineset 3D Tree Visualizer can display the tree hierarchy and map attributes to histogram at each node. The *cone tree* method of Xerox PARC embeds trees in a three dimensional space with the rotating function that allows bringing different parts of the tree into focus [4]. However, this technique requires expensive 3D animation support and manipulates difficultly trees with more than approximately 1000 nodes.

One common problem to all tree visualizers is if nodes on the tree, even only leaf nodes, are to be given adequate spacing, the nodes near the root must be placed very far

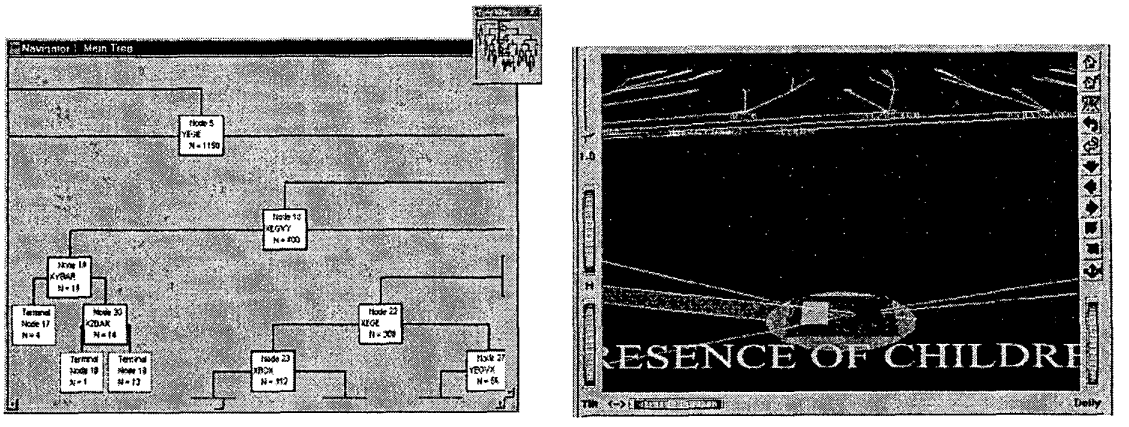

**Figure 1. Difficult to navigate large trees in CART and Mineset**

apart, leaving no nice way to display the context of the entire tree [9]. In fact, the user does not go from node to node randomly. He/she usually wants to move from a node to one of its relatives, and often get lost (unable to find the way) in a huge hierarchy. Figure 1 shows that it is hard to navigate large trees in CART (left) and Mineset (right) where relatives of the node in focus cannot be seen from these screens.

Our work was motivated by the lack of adequate tools for the visualization of large trees. It contributes a new alternative to tree visualization with the node-link diagram. In particular, it can be more natural to human perception and cognition than the hyperbolic trees in learning decision trees, and it use more efficiently the display space with multi-level dynamic browsers and the fish-eye view [6].

## 3 The Tree Visualizer in CABRO

In CABRO, a mining process concerns with *model selection* in which the user try different settings of decision tree induction to attain most appropriate decision trees. To help the user to understand the effects of settings on result trees, the tree visualizer is capable of handling multiple views of different trees at any stage in the induction.

CABRO with its tree visualizer has been implemented in UNIX workstations under the X Window, and recently in MS Windows. Our attempt aims at improving the node-link diagram of CABRO tree visualizer to deal with the above problems for large decision trees. The main features of the CABRO Tree Visualizer are:

1. It provides several modes of view: zoomed, tiny, tightly-coupled and fish-eyed, in each mode the user can interactively change the layout of the structure to fit the current interests.
2. It provides a new and efficient technique, called T2.5D, to make navigating large tree easier.

### 3.1 Different modes of view

CABRO tree visualizer is a graphical interface that displays the same tree in two tightly-coupled views, one a *global view* and the other a *detailed view*. Each of these views can be expanded or collapsed and is with one of the following modes (Figure 3):

- *Tightly-coupled views*: The global view (on the left) shows the tree structure with nodes in same small size without labels and therefore it can display a tree fully or a large part of it, depending on the tree size. The detailed view (on the right) shows the tree structure and nodes with their labels associated with operations to display node information. The global view is associated with a *field-of-view* or *panner* (a wire-frame box) that corresponds to the detailed view [12]. These two views are tightly-coupled as the field-of-view can be moved around in the global view in order to pan the detailed view. Also, when the detailed view is scrolled the position of the field-of-view will be updated accordingly. The windows for these two views can be resized by the user, and the field-of-view shape and size will be automatically changed. Figure 2 shows a screen of CABRO tree visualizer with tightly-coupled views for the well-known soybean dataset.
- *Customizing views*: Initially, according to the user's choice, the tree is either displayed fully or with only

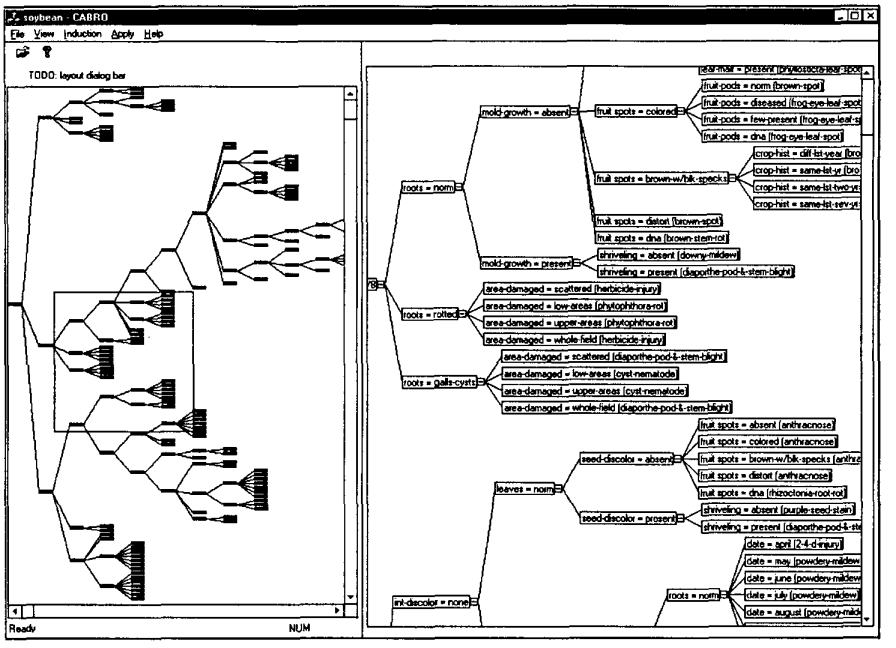


**Figure 2. CABRO Tree Visualizer: tightly-coupled views**

the root node and its direct sub-nodes. The tree then can be *collapsed* or *expanded* partially or fully from the root or from any intermediate node. Any subtree with the root at an uncollapsed node (node with $+$) can be collapsed into one node (at its root with $-$). Thus, the user is able to interactively customize views of the tree to meet his/her need and interests. Also, the user is provided the focus view on one class and its relation to other classes in the whole hierarchical structure with different colors.

- *View of decision/leaf nodes*: The user can click a node to see its information: branching attribute and branched attribute-value, number of covered cases, the major class, the percentage of major class, the leading path from the root, etc. (illustrated in Figure 4). When focusing on one class, only its leaf nodes are highlighted and proportions of cases bearing this class label are indicated approximately.

- *Tiny mode with fish-eye view*

  Note that no current visualization technique allows us to display efficiently the entire tree when it has, says, ten thousands nodes. If we note that, "the foundational period of information visualization is now ending", and "in the next period, information visualization will pass out of the realm of an exotic research specialty and into the mainstream of the user interface and an application design" [4], we might have to think about efficient ways to manipulate large trees without expecting to see it entirely.

  The tightly-coupled views are extended with three viewing modes according to the user's choice: normal size, small size and tiny size. The tiny mode uses much more efficiently the space to visualize the tree structure, on which the user can determine quickly the field-of-view and pan to the region of interest. It allows the user to be able to see the tree structure while focusing on any particular part so that the relationship of parts to the whole can be seen and the focus can be moved to other parts in a smooth and continuous way.

  Fish-eye is an interesting variant of the classic overview-detail browser, proposed in [6]. This view distorts the magnified image so that the center of interest is displayed at high magnification, and the rest of the image is progressively compressed. In CABRO tree visualizer, we define three fish-eye components as follows:

  1. focal point $f$: some node of current interest in the tree;
  2. distance from focal point $f$ to a node $x$: $D(f,x) = d(f,x)$ where $d(x,y)$ between two points $x$ and $y$ on the tree is the number of links intervening on the path connecting them in the tree;

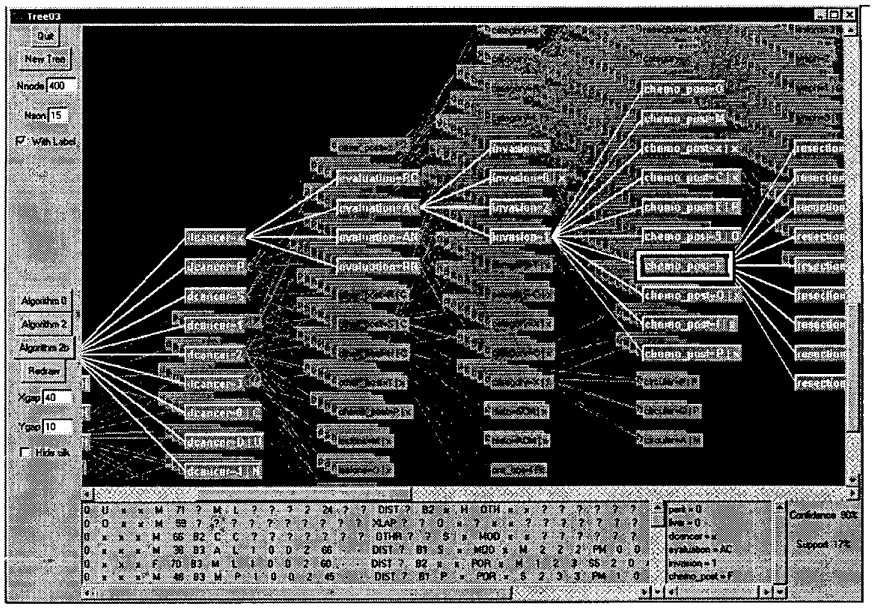

**Figure 3. T2.5D visualization**

3. detail level, importance, resolution: $LOD(x) = -d(r, x)$ where $r$ is the root of the tree.

CABRO tree visualizer permits the user to change dynamically the point of interest on the tree, and a dynamic threshold $k$ for the distance from focus. By default, $k = 2$ for the distance from the left and from the right of focal point. A variant of visualization is to use the global browser in tiny mode with fish-view for the field of interest.

### 3.2 Visualization with T2.5D technique

With very large tree, the user might still find it is so difficult to navigate, even with tiny mode and fish-eye view. To address the problem, we have been developing a new technique called T2.5D (stands for 2.5 Dimensions Tree). The starting points of T2.5D's design are the following:

- The screen space is limited, therefore only a number of nodes of a tree can be displayed simultaneously. Other nodes may be out of screen, pruned or displayed in the background.
- How to display together a large subset of related nodes in a tree within a limited space is essential. For a decision tree, as paths (we will define them latter) correspond to rules, tracing nodes belonging or relating to a path is an common task. T2.5D aims to support doing this task as conveniently as possible.
- Maintaining the structural information of a tree when the user navigates the tree is indispensable. It helps the user easily to identity where she/he is and where she/he wants to go.
- The more nodes are displayed in the screen, the less scrolling operations the user need to do. Therefore, arranging nodes such that a large number of them can be displayed in a compact screen space is important.

To describe the technique we need to clarify some terms that will be used. We define a *wide path* to a node in the tree as the set of all nodes in the path from the root to this node and their siblings. Therefore, the wide path to a node contains all of its direct relatives. When visualizing a tree, at each moment, the wide path to the *node in focus* is called the *active wide path*. We summary below the main ideas and characteristics of T2.5D:

- The active wide path is displayed in the front of the screen with highlighted colors. Other nodes of the tree are displayed in the background with dim colors. They are drawn in a 3D form to save space while the active wide path is drawn in a 2D form in order to give a clear view.

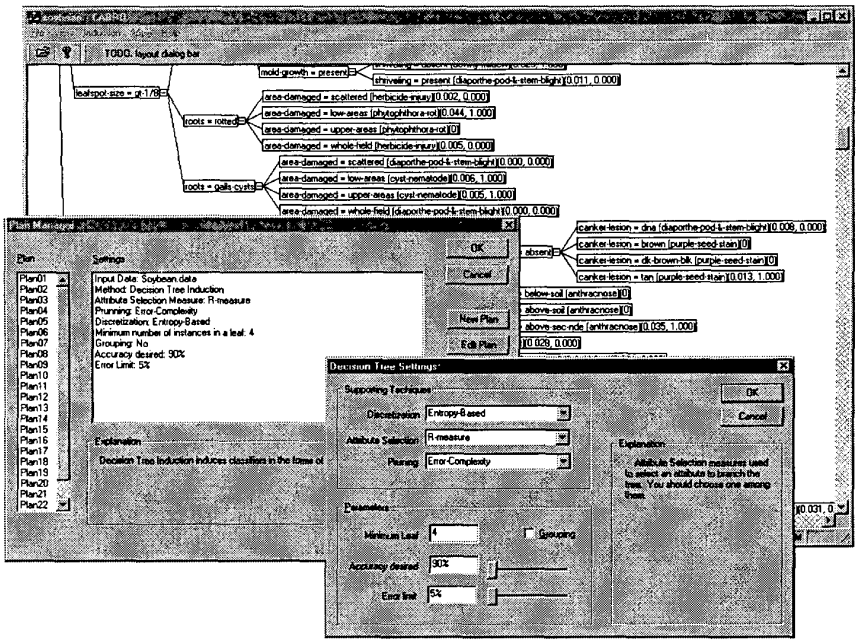

**Figure 4. CABRO Tree Visualizer: support for model selection**

- The user can change the node in focus by clicking to another node of the tree, and its wide path then becomes the active wide path and is highlighted. The user does not have to make a lot of navigation in order to learn about the relatives of the node in focus.

- The labels of some nodes in the background may be overlapped by other nodes. For that case, a floating balloon is provided. The balloon will display clearly the label of the node which is currently pointed by the mouse pointer.

- The mix between 2D and 3D drawing has two main merits. Firstly, the node in focus and its relatives have a clear view, at the same time, many other nodes also can be seen to maintain the tree structural information. Secondly, many nodes can be displayed in a compact screen space.

We have been tried T2.5D with several datasets both real and artificial. In our experiment, the new technique have several advantages:

- T2.5D easily handles decision trees with more than 20000 nodes, and more than 1000 nodes can be displayed together on the screen.

- It is useful to have a very clear view of a wide path to a node in a tree and an image of the overall structure of the tree at the same time.

- Navigation even on huge decision trees is easy and fast. We do not have to perform many operations (scrolling, expanding, collapsing, or using the map) to navigate trees.

Figure 3 shows a screenshot of the CABRO tree visualizer with T2.5D technique. The tree generated from a dataset of stomach cancer data and has more than 2000 nodes. Most of nodes in the tree appear in the window.

# 4 Interactive learning of large decision trees

## 4.1 Support for model selection

There exist various methods for solving three main DTI problems of attribute selection, pruning and discretization, and it is well known that none of them is universally superior than others. It raises in practice the problem of *model selection*, that is how to choose the most appropriate DTI methods/models for a given application task. This problem requires meta-knowledge and/or empirical comparative evaluations of methods/models.

| | 3D in Mineset | 2D in CART | Hyperbolic Tree | T2.5D |
|---|---|---|---|---|
| Embedded in DTI | no | no | n/a | yes |
| Comprehension | average | average | high | high |
| Efficiency | average | average | high | high |
| Interactivity | average | high | high | average |
| Esthetics | high | average | very high | high |

**Table 2. Comparison of several tree visualizers**

Three main criteria for selecting DTI models are size, accuracy and understandability of trees. The tree size and accuracy can be quantitatively evaluated, among them the accuracy is widely considered to be of great importance. The understandability of trees is difficult to be quantified or measured, and the idea here is to use tree visualizer to support the understanding of users.

Though the starting point of CABRO is R-measure, a new proposed measure for attribute selection [11], CABRO associates its tree visualizer with other modules to provide a number of well-known DTI available techniques and to combine them to form DTI models and evaluate them (accuracy, size and understandability).

In the current version of CABRO, the user can generate new models each is composed by an attribute selection measure chosen from the gain-ratio [13], the gini-index [2], $\chi^2$ and R-measure; a pruning technique from error-complexity, reduced-error and pessimistic error [2], [13]; and a discretization technique from the entropy-based and error-based techniques [5].

Information of each trial on a model chosen by the user is registered in a form called *plan* of that model. The realization of that plan yields a model that may or may not be accepted by the user, and the user usually has to try a number of different plans. This process is repeated automatically for different model candidates by a $k$-fold stratified cross validation (by default, $k = 10$).

For each model candidate, CABRO tree visualizer displays graphically the corresponding *pruned tree, its size, its prediction error rate*. It offers the user a multiple view of these trials and facilitates the user to compare results of trials in order to make his/her final selection of techniques/models of interest. Figure 4 shows a screen of model selection in CABRO.

### 4.2 Support for matching of unknown objects

CABRO tree visualizer is used not only in inducing decision trees but also in classifying unknown objects. It plays the role of the interface for visual explanation of the matching process, in a way similar to the explanation in knowledge-based systems. CABRO tree visualizer supports three modes of matching an unknown object according to the way that the unknown object is declared.

- The whole record of the unknown object is read from a database: CABRO directly show the leaf node that matches the object. The path from the tree root until that leaf node will be highlighted. Information accumulated along the path can be viewed at any node.

- Values of attributes are given by the user when answering the system questions: Questions about attributes will be asked according to the hierarchical structure in a top-down manner from the root. From menu the user will choose one value in the list of discrete values of the attribute or enter a numerical value in case of continuous attribute. Questions are asked dynamically according to the stepwise refinement of the matching process.

- The user declares values of attributes he/she knows: The user is able to select attributes that he/she wishes to query on. These attributes can be selected from the attribute list with corresponding values. Once the attribute-values pairs are entered, the tree visualizer will limit the regions on the tree that partially satisfy the data. The system will then ask additional questions to fulfill the match.

Table 2 summarizes our evaluation on the properties of some tree visualizers according to the requirements introduced in section 2.

## 5 Conclusion

In this paper we addressed problems of visualizing large decision trees, related works on tree visualizers and information visualization field. We presented the interactive tree visualizer of system CABRO that employs recent visualization techniques in mining decision trees. We described our attempt to deal with large tree by introducing multiple views with a new tiny mode, a variant of fish-eye view, and the new technique T2.5D for tree visualization. Though there

are still a lot of work for improving this interactive tree visualizer, we believe that it contributes an efficient solution to the state-of-the-art of visualization in decision tree learning, especially T2.5D is very promising as a new display and navigation technique for large trees.

## References


[1] Ankerst, M., Elsen, C., Ester, M., Kriegel, H. P.: Visual Classification: An Interactive Approach to Decision Tree Construction. *Proc. of Fifth Inter. Conf. on Knowledge Discovery and Data Mining* (1999), 392–397.

[2] Breiman, L., Friedman, J., Olshen, R. and Stone, C.: *Classification and Regression Trees*. Belmont, CA: Wadsworth (1984).

[3] Brunk, C., Kelly, J. and Kohavi, R.: MineSet: An Integrated System for Data Mining. *Proc. of Third Inter. Conf. on Knowledge Discovery and Data Mining* (1997) 135–138.

[4] Card, S. K., Mackinlay, J. D., Shneiderman, B.: *Readings in Information Visualization*. Morgan Kaufmann (1999).

[5] Dougherty, J., Kohavi, R., Sahami, M.: Supervised and Unsupervised Discretization of Continuous Features. *Proc. of the 12th Inter. Conf. on Machine Learning* (1995) 194–202.

[6] Furnas, G. W.: The FISHEYE View: A New Look at Structured Files. *Bell Laboratories Technical Memorandum* #81-11221-9 (1981).

[7] Johnson, B., Shneiderman, B.: Treemaps: A Space-Filling Approach to the Visualization of Hierarchical Information Structures. *Proc. of IEEE Information Visualization* (1991) 275–282.

[8] Kumar, H. P., Plaisant, C., Shneiderman, B.: Browsing Hierarchical Data with Multi-level Dynamic Queries and Pruning. *Inter. Journal of Human-Computer Studies*, 46(1) (1997) 103–124.

[9] Lamping, J., Rao, R.: The Hyperbolic Browser: A Focus + Context Techniques for Visualizing Large Hierarchies. *Journal of Visual Languages and Computing*, 7(1) (1997) 33–55.

[10] Murphy, P. M., and Aha, D. W. (1994). *UCI Repository of machine learning databases* [http://www.ics.uci.edu/ mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

[11] Nguyen, D. T., Ho, T. B.: An Interactive-Graphic System for Decision Tree Induction. *Journal of Japanese Society for Artificial Intelligence*, Vol. 14 (1999) 131–138.

[12] Plaisant, C., Carr, D., Shneiderman, B: Image Browser Taxonomy and Guidelines for Designers. *IEEE Software*, 12 (1995) 21–32.

[13] Quinlan, J. R.: *C4.5: Programs for Machine Learning*. Morgan Kaufmann (1993).

[14] Rao, J. S., Potts, W. J. E.: Visualizing Bagged Decision Trees. *Proc. of Third Inter. Conf. on Knowledge Discovery and Data Mining*, AAAI Press (1997) 243–246.

[15] Robertson, G. G., Mackinlay, J. D., Card, S. K.: Cone Trees: Animated 3D Visualization of Hierarchical Information. *Proc. of the ACM Conf. on Human Factors in Computing Systems* (1991) 189–194.