Visualizing Decision Table Classifiers

Barry G. Becker
Silicon Graphics Inc.
2011 N. Shoreline Blvd, MS-500
Mountain View, CA 94043-1389
(becker@engr.sgi.com)

Abstract

Decision tables[1], like decision trees[2] or neural nets[3], are classification models used for prediction. They are induced by machine learning algorithms. A decision table consists of a hierarchical table in which each entry in a higher level table gets broken down by the values of a pair of additional attributes to form another table. The structure is similar to dimensional stacking [4]. Presented here is a visualization method that allows a model based on many attributes to be understood even by those unfamiliar with machine learning. Various forms of interaction are used to make this visualization more useful than other static designs.

keywords: classifier, decision table, dimensional stacking, trellis displays, relational data, data mining.

1. Introduction

In supervised classification learning, an induction algorithm produces a model from a labeled set of training data. The resulting classification model is capable of predicting labels for a new set of unlabeled records that have the same attributes as the training data. Each distinct label value is called a class. The word attribute is used here to refer to a column in a relational dataset.

Many classifier models are applied as a black box. It is not necessary to understand the model in order to use it effectively for prediction. Recently, there has been a realization that great insight can be gained by visualizing the structure of a data mining model. Decision trees were one of the earliest classification models to be represented graphically because of their easy to understand structure [2]. Simple Bayesian classifiers have also been represented graphically [5]. Neural networks have an arcane structure which is difficult to visualize. Fortunately, decision tables have a simple structure suitable for creating an understandable display.

The ability to describe the structure of a classifier in a way that people can easily understand, transforms classifiers from incomprehensible black boxes to tools for knowledge discovery. Classification without an explanation reduces the trust a user has in the system. Spiegelhalter and Knill-Jones [6] found that physicians would reject a system that gave insufficient explanation even if the model had good accuracy. A human may decide not to use a classifier if he or she realizes that it is based on irrelevant attributes, bad data, or if important factors are being ignored.

There have been some initial attempts to display decision tables in the form of a General Logic Diagram (GLD)[7]. In this format each cell in the decision table has a single color to indicate the predicted class.

Several methods of visualizing multi-dimensional datasets that use several variables to form a hierarchy have been proposed. Leblanc, Ward, and Wittels proposed dimensional stacking [4] which shows a single colored cell at the lowest level of the hierarchy. Trellis displays [8], have expanded this concept by generalizing the representation of the cell at the lowest level to be a plot of any type, such as a scatterplot, surface plot, or line graph.

A deficiency of these methods is that a static scene is limited to two or three levels of detail before the display gets too large or the cells become too small to be useful.

Interactive drill-down, drill-up, drill-through, filtering, and animation allow the user to explore in great detail specific regions of interest, while comparing them to other regions that may be higher in the hierarchy.

2. Design Requirements and Solutions

Decision tables tend to have a very large number of cells at the lowest level. For example, 6 attributes with 8 values per attribute, have potentially 6^8 = 1,679,616 cells. Fortunately, the fact that the data is seldom uniformly distributed over the space reduces the cell count considerably in practice. Even so, the size is usually far too great to deal with interactively. Even building such a scene can take prohibitively long. The solution is to provide levels of detail to the users which they can incrementally drill down on. The graphics are not computed for lower levels until a drill-down request occurs. In

general, the design attempts to follow Ben Shneiderman's visual information-seeking mantra: "overview first, zoom and filter, then details-on-demand" [9].

Since the data structure is inherently hierarchical it is easy to start with an overview by showing just the top levels of the hierarchy. Three dimensional manipulation allows flexible navigation for studying regions of interest. Drilling down locally gets the user details on demand, and drill-through can return the actual raw data from which the classifier was induced. Filtering on specific values or ranges of values has also been implemented.

Each of the cells needs to show a distribution of the classes for the records matching that cell. This distribution is shown using a cake chart, which is similar to a pie chart, but rectangular in shape and has rectangular slices instead of wedge shaped slices.

3. Treatment of Attribute Values

Attributes must have some set of discrete values for the classifier to be effective. If an attribute is continuous, the inducer uses entropy based discretization so that the distribution of classes in adjacent bins are as different as possible. This discretization is done globally [10].

The ordering of the bins for the continuous attributes is explicit, but for categorical attributes the values can be ordered in three meaningful ways: alphabetically, numerically by record weights, or numerically by correlation with one of the classes to be predicted. In most cases, it was found that the last method provides the most benefit. If a particular class is selected for sorting by label probability, then the selected class is used in determining the ordering. This type of ordering was also found useful in trellis displays [8] (where it is referred to as main-effects ordering) and in visualizing the structure of a Simple Bayesian Classifier [5]. By looking at how dramatically the distributions change along an axis, it is possible to see how well that categorical attribute correlates with the label, and also how it interacts with the other attribute mapped to that level of the hierarchy. Figure 1 shows the interaction between the *odor* and *spore-print color* attributes in a mushroom dataset. Note that the values for the more edible mushrooms appear in the lower left because the ordering of nominals has been specified to be by correlation with edibility.

It is common for relational data to contain NULL (unknown) values. These NULL values are treated quite differently from 0 or some other value. If an attribute has NULL values, then the unknown values are denoted by a question mark (?) in the visualization. The NULL value, if present, always appears as the first value, and does not get sorted with the rest. Viewing of NULLs can be toggled on or off so that they do not interfere with observing trends in the non-null data.

Nominal values which contain less than a certain percentage of the data may be filtered out in order to simplify the visualization. This is useful because in many datasets there are some nominal attributes with many unique values. The visualization does not perform well if one axis has many more values than its pair at a given level. Usually one is in interested in the values that are most prevalent. A slider is provided to filter values that have less than some small percentage of the total weight.

4. Implementation Issues

Given a training set of labeled records, an induction algorithm creates a classifier. Automatic selection of attributes is at the core of the induction algorithm [11]. Alternatively, a user may apply their knowledge of the domain to specify some or all of the attributes to use at each level of the hierarchy. Using the tool this way may not build a model with maximal accuracy, but it may be more useful for gaining understanding of one's data. Once the model has been built (usually on a server), the structure is passed to the visualization tool for display. As a result, the visualization application never loads the whole dataset into memory. However, drill-through techniques may be used to retrieve some subset of the original data.

The structure of the decision table model is stored simply as a relational table, where each row represents an aggregate of all the records for each combination of values of the attributes used. Once loaded into memory, a hierarchy of tables is constructed, where each new table one level higher up in the hierarchy has two fewer attributes. Finally, the top-most level, a single row represents all the data. Besides a column for each attribute, there is a column for the record count (or more generally, sum of record weights), and a column containing a vector of probabilities (each probability gives the proportion of records in each class). For example, given a dataset with four attributes plus a label which has 3 values (or classes), Table 1 shows the base table, Table 2 shows the next higher table, and finally Table 3, which has only one row, shows the table at the very top.There is a row for every combination of attribute values that have representative records. As a result, no row can have 0 weight.

Table 1: Level 2

Attrib1	*Attrib2*	*Attrib3*	*Attrib4*	*Weight*	*Probability[]*
a	10-20	4-9	yes	10	.3, .6, .1
a	10-20	9-10	yes	34	.61, .37, .02
a	20-30	9-10	yes	123	.1, .6, .3
a	20-30	9-10	no	1	1., 0., 0.
a	20-30	10-18	yes	5	.8, .2, 0
a	20-30	10-18	no	23	.2 , .3, .5
:	:	:	:	:	:
d	40-50	4-9	yes	2	.5, .5, 0.
d	40-50	9-10	no	7	0., 1., 0.
d	40-50	10-18	no	9	.33, .33, .33

Table 2: Level 1

Attrib1	*Attrib2*	*Weight*	*Probability[]*
a	10-20	44	.61, .34, .05
a	20-30	850	.23, .41, .36
a	30-40	230	.35, .2, .45
a	40-50	56	.12, .28, .60
:	:	:	:
d	40-50	120	.17, .31 , .52

Table 3: Level 0

Weight	*Probability[]*
19,345	.32, .33, .35

Each row corresponds to a cake chart in the visualization. The height of the cake chart corresponds to the *Weights* column, and the slices of the cake are determined by the proportion of those records in each class (i.e. the *Probability*[] column). The geometry is represented as an Inventor [12] scene graph.

There were many complicated issues dealing with layout during implementation. The main ones were: aspect ratios, overlapping text, and reducing the amount of geometry to draw.

The aspect ration is the ratio of the length to the width. An aspect ratio of one is optimal [8], but it is easy to have huge extremes if one axis has many more values than the other, or even if one axis has moderately more values for several levels in a row. A solution is to interchange attributes at levels until the aspect ratio becomes as close to one as possible at every level.

Ideally one would like to see text labels annotate every cake chart in the scene. Unfortunately, if that were done the text would obscure other objects or be too small to read. A compromise is to show the value labels only if they are at the edge (right and top), or if their is no data in the region to the left or top of the current matrix of cake charts. Occasionally there is still overlap, but in most cases the text is still readable because of the difference in scale and color. The text in the scene has four levels of detail that are controlled automatically based on the user's distance.

In an early prototype, drill-down was controlled automatically as the user navigated. If the user's viewpoint got close to a cake chart, it was automatically expanded to the next level of detail. Although a nice idea in theory, too much detail was generated in most cases, and the interaction became sluggish. The solution was the custom drill down described in the next section.

5. Interactive Features

Interaction is crucial when you have a large model to be explored and understood. You simply cannot display everything in a static scene and have it be comprehensible.

When you drill down into a cake chart, the data represented by that cake is re-displayed in a new matrix of cake charts where the attributes assigned to that level of detail are used as the axes. The cake charts sit on top of a gray base. If the base is completely covered with cakes, then every combination of values at this level is represented by the data. Often much of the base is uncovered. This shows regions where no data exists. In a very sparse dataset large regions are empty. Selecting a gray base has the same effect as selecting the cake that was one level up in level of detail. Drilling down on a base causes every cake sitting on top of it to expand to the next level of detail.

If a cake chart is selected on the left, then the pie chart on the right will show a distribution which matches that shown by the cake. When more than one cake on the left is selected, then the pie on the right will show the probability distribution (with respect to the label) for the set of records defined by the union of selected cakes (see Figure 4). In this way it is not only possible to extract predicted values based on the model, but it is also possible to define a query for drilling through to the underlying data.

If the classes are nominal they are listed under the pie chart on the right, and are in order of slice size. The class with the largest probability is at the top. As values on the left are selected, this order may change based on the new probabilities. The class that would be predicted given current selections is shown at the top of the list. If the label is a binned attribute the order of the classes is not based on slice size, but on numerical ordering.

If you place the cursor over a base or cake chart without selecting, text is displayed showing the values of the two attributes at that level of detail. Also shown is the weight of records represented; which is the height of the cake chart (see Figure 1).

At each level of detail the attribute names are shown to the left and bottom of the array of cake charts, their values are shown to the right and top, respectively (if there is room). If there are an odd total number of attributes, the lowest level shows only one attribute.

6. Results

Described here are some examples where the visualization has been applied. In the first example, our goal is to understand what factors effect mushroom edibility. There are over a dozen attributes in the original data, but it is not necessary to use all of them in order to build an accurate classification model. The decision table classifier whose structure is shown in Figure 1 uses only four of the attributes. The window on the left shows the top level of detail. *Odor* and *spore-print-color* were chosen by the inducer to be at the top level because doing so improves the accuracy of the underlying classifier. Note the dependency between these two attributes. There is only one top

level cake (*odor*="none" and *spore-print-color*="white") with more than one class is present. When you drill-down to the next level by clicking on this cake chart the attributes *habitat* and *population* are used to resolve the ambiguity (see Figure 2).

Figure 3 shows the structure of a decision table classifier induced from census data containing fifteen attributes, and about 50,000 records. Here the income attribute (which has been binned into three ranges) has been selected as the class label. The attributes selected at level one are: *relationship* and *sex*; level two has *education* and *occupation*; and level three has *hrswk* (hours worked per week) and *age*.

At the top level, *relationship* and *sex* have a strong correlation. Surprisingly, there are 3 male wives and one female husband (indicating a possible data quality problem). We can see distinct weight and salary distributions for each combination of sex and relationship. There is a significant difference between distributions for unmarried males versus females (See Figure 3). There is a cluster of red cake charts in the lower left of the male matrix that does not exist in the female matrix. The female matrix has obvious spikes at *occupation* = "admin-clerical" and "other service". No such spikes are visible in the corresponding male matrix.

Figure 4 shows the result of drilling down on the base for male husbands. One can compare the salary distributions for HS-grad husbands who are in sales with those who are executive managers (these two regions are have been selected in Figure 4). Although the distribution of age and hours worked is similar, the probability of having an income greater than 60,000 class is 34% for this group of managers, compared with 27% for the salesmen. The records in Figure 5 show the result of drilling through on these two selected regions.

7. Conclusions and Future Work

Machine learning can build accurate models to describe data. The knowledge contained in these models is hidden, however, without an effective way to convey it to users. The method of visualizing decision table classifiers presented here takes full advantage of interactive techniques to maximize user control of the model exploration process. With understanding of the underlying model comes a trust the results it yields. The combination of machine learning and data visualization gives a far more powerful tool for understanding than either does independently.

There are several areas for future research: allow the label to be continuous rather than discrete; automatically choose attributes at each level so that understanding rather than accuracy is maximized; allow the user to interactively rearrange the attributes mapped to each level would help exploration; and use the same idea of trellis displays [8], to allow different plots at each cell rather than strictly using a cake chart.

Acknowledgments

The author would like to thank Ronny Kohavi, Roger Crawfis, and Dan Sommerfield for their helpful suggestions. Dan Sommerfield implemented the decision table induction algorithm. The census and mushroom data are available from http://www.ics.uci.edu/~mlearn/MLRepository.html.

References

[1] R. Kohavi, "The Power of Decision Tables", *Proceedings of the European Conference on Machine Learning*, Lecture Notes in Artificial Intelligence 914, Springer Verlag, Berlin, Heidelberg, NY, pages 174-189.

[2] J. Quinlan, *C4.5:Programs For Machine Learning*, Morgan Kaufmann Publishers, Inc., 1993.

[3] J. Hertz, A. Krogh, and R. Palmer, *Introduction to the Theory of Neural Computation*, Addison Wesley, 1991.

[4] J. LeBlanc, M. Ward, and N. Wittels, "Exploring N-Dimensional Databases", *Proceedings of First IEEE Conference on Visualization* (Visualization '90), pages 230-237, 1990.

[5] B. Becker, R. Kohavi, and D. Sommerfield "Visualizing the Simple Bayesian Classifier", to appear in *Issues in the Integration of Data Mining and Visualization*, Springer-Verlag, 1998.

[6] D. Spiegelhalter and R Knill-Jones, "Statistical and Knowledge-based Approaches to Clinical Decision Support Systems, with an Application in Gastroenterology", *Journal of the Royal Statistical Society* A 147, pages 35-37, 1984.

[7] J. Wnek and R. Michalski, "Hypothesis-driven constructive induction in AQ17-HCI: A method and experiments", *Machine Learning* 14(2), pages 139-168, 1994.

[8] R. Becker, W. Cleveland, and M. Shyu, "The Visual Design of Trellis Display", *Journal of Computational and Statistical Graphics*, 1995, vol 5, pages 123-155, 1996.

[9] B. Shneiderman, Designing the User Interface: Strategies for Effective Human-Computer Interaction, Third Edition, Addison-Wesley Publ. Co., Reading, MA, p523, 1998.

[10] R. Kohavi and M. Sahami, "Error-Based and Entropy-Based Discretization of Continuous Features", Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pages 114-119, 1996.

[11] R. Kohavi and D. Sommerfield, "Targeting Business Users with Decision Table Classifiers", to appear in the *Proceedings of Knowledge Discovery and Data Mining*, 1998.

[12] J. Wernecke, *The Inventor Mentor*, Addison-Wesley, 1994