Interactive Interpretation of Hierarchical Clustering

Eric Boudaillier[1] and Georges Hébrail[2]

[1] Postgraduate Student at University of Orsay, France
[2] Electricité de France, Research Center, 1, Av. du Général de Gaulle, F-92141 Clamart Cedex, France
Georges.Hebrail@der.edfgdf.fr

Abstract. Automatic clustering methods are part of data mining methods. They aim at building clusters of items so that similar items fall into the same cluster while dissimilar ones fall into separate clusters. A particular class of clustering methods are hierarchical ones where recursive clusters are formed to grow a tree representing an approximation of similarities between items. We propose a new interactive interface to help the user to interpret the result of such a clustering process, according to the item characteristics. The prototype has been applied successfully to a special case of items providing nice graphical representations (electric load curves) but can also be used with other types of curves or with more standard items.

1 Introduction

Hierarchical clustering methods have been studied and used widely for a long time. They have been developed both in the statistical data analysis field (see for instance [3], [4], and [5]), and in the machine learning field as conceptual clustering methods (see [7]). In this paper, we focus on hierarchical methods developed in statistical data analysis, but much of the work presented here can be transposed to help to interpret results of conceptual clustering methods.

Clustering methods help the user to build partitions of items so that items may be replaced by groups of items in further data analysis processes. These methods are thus very useful for mining large databases: this is a way to reduce the number of items to be processed. Moreover, if a good interpretation is given to each group of items, this helps to provide more readable data mining results. Interpretation of clustering results is consequently a critical step and the aim of our work is to improve this process. The way we propose to achieve this goal is to provide an interface through which the end-user and the computer cooperate to build cluster interpretations.

Hierarchical clustering methods grow a tree from a similarity measure defined within a set of items. The result is a tree where items form the leaves of the tree and where each node of the tree represents a cluster of similar items. The further the node is from the tree root, the more similar are the items under the node.

In standard statistical data analysis software (see for instance [1], [9]), the end-user proceeds as follows:

- execution of hierarchical clustering,
- plotting of the tree on the screen or on a sheet of paper,
- horizontal cutting of the tree to form a partition of items,
- interpretation of each class of the partition using attributes describing items,
- possible reconsideration of the cutting and re-interpretation.

This approach is very long and inefficient in the following ways:

- there is only very basic computer human interaction when the tree is plotted on the screen,
- the cutting of the tree is horizontal while there is a need for more precise cutting as we will see later in the paper,
- interpretation tools are not tightly coupled with the cutting tool so that it is not easy to interpret different cuttings at different levels of the tree.

In this paper, we present a new prototype software whose goal is precisely to overcome these problems and to help the user to interpret the result of a hierarchical clustering. This prototype makes a large use of graphical and window-based interaction possibilities available nowadays on every computer. It has been developed on an UNIX workstation using Tcl and Tk toolkits (see [10]). All statistical steps, such as the clustering and the computation of statistical indicators for interpretation, are performed using the SPLUS language (see [1]).

This prototype has been applied to different datasets, but we focus here on a special dataset describing electric power load curves. These items are of special interest because they can be represented graphically: the clustering tree becomes much more readable. Electric load curves reflect the way EDF customers consume electric power during the day. The prototype software can be used without any further development on other types of curves, such as financial time series, customer expenses evolutions, and so on. It can also be used on standard data, i.e. data without any curve associated with items.

In Section 2, we briefly recall the basic pieces of information returned by a clustering process. Section 3 is devoted to the description of the user's task in cluster interpretation. Section 3 also introduces the dataset of electric load curves and discuss specific needs for cluster interpretation in this case. Section 4 presents how the tree is displayed to the user and the kind of actions available on the tree. We show that this interface enables the user to work in two different ways: a top-down or a bottom-up approach. In Section 5, we focus on interpretation of clusters, using interactive histograms linked together and to the clustering tree. In Section 6, we present the way the end-user achieves its final goal, i.e. the construction of a partition of the items.

2 Hierarchical clustering

Hierarchical clustering methods take in input a set of items. These methods build a tree which approximates similarities between items. These similarities may be given in two ways:

- directly as a similarity or distance matrix between items,

- indirectly: items are described by some characteristics (i.e. attributes) and the similarity between two items is defined according to the similarities between the item characteristics.

The output of these methods is a binary[3] tree where each leaf is an item and where intermediate nodes represent groups of items. The standard agglomerative algorithm is:

- find the two closest items in the dataset and build a node aggregating these two items,
- compute a similarity between this node and the other items,
- recursively aggregate items and/or nodes of aggregated items until the aggregated node covers all the items.

In addition to the tree, these methods associate with each node of the tree a *height* value which is the aggregation level of the node, i.e. the similarity between the two subnodes of the node. Sometimes, the method also provides an order of the items which is compatible with the tree: this order (which is not unique) is such that drawing the items in this order enables the drawing of the whole tree without any branch crossing in the tree display.

The interested reader can refer to [3], [4] and [5] to have more information about these methods, and to [1] and [9] for software to execute corresponding algorithms.

3 Description of the task

The main task of the end-user when performing a clustering of items is to build a partition of the set of items into some disjoint classes. Once items have been partitioned, it is possible to compute aggregates over the different classes. For instance, if the set of items is a set of customers, classes may represent types of customers and aggregates may be the average age or income for each class of customers.

Some clustering methods exist that directly build a partition of items (for instance *k-means* methods). But the user has to choose in advance the number of clusters and different trials are necessary to find the right number of clusters. On the contrary, hierarchical methods provide different numbers of clusters depending on the level of abstraction the user is interested in.

So, the result of a hierarchical clustering interpretation task can be defined as follows:

- a partition of the items into classes: each item is described by a new characteristic which is the class it belongs to (each item belongs to a single class),
- each class is summarized by a label given by the user after a careful interpretation of the characteristics of the items belonging to the class.

There are many ways to define a partition from the tree built by a hierarchical clustering algorithm. The most standard way is to cut the tree horizontally at

[3] Some methods produce n-ary trees: the work presented here can be easily extended to these methods.

a specified level. This determines a partition of items with a number of classes which depends on the level of the horizontal cutting. Standard software performs such cuttings if the user defines as a parameter: either the number of desired classes, or the height of the node at which the cutting has to be done. More sophisticated software helps the user to find the cutting height by choosing a height value just before a large jump in the height values list.
Two criticisms can be made of this approach:

- the cutting is not done graphically but is defined as an ASCII parameter of a program,
- only horizontal cuttings are possible.

This last point needs more discussion. In (simplified) theory, only horizontal cuttings are legal, since non-horizontal cuttings violate the optimality property that two items belonging to the same class are closer from each other than two items from different classes. In practice, there is a need for building classes corresponding to more opportunist cuttings: the user may want to have more details in some classes than in some others, or may want to group into the same class items which appear to be dissimilar according to the clustering criteria. We will see later on that this interface enables the user to build a partition which may not correspond to any horizontal cutting of the tree, but is inspired by the clustering result. In particular, it will be possible to group into the same class several untypical items, even if they appear at very different positions in the clustering tree.

Interpretation of each class is made by observing characteristics of the items assigned to the class. Here it is necessary to distinguish among item characteristics:

- *active* characteristics are used in the clustering process to define the similarity between items,
- *external* characteristics also describe the items but have not been used in the clustering process.

Both active and external characteristics are useful to build interpretations. The clustering process is also a way for the user to study correspondences between two sets of characteristics.

3.1 The case of electric load curves

In our study, we have considered several datasets, but the most interesting one is a dataset describing 2,665 electric load curves of some customers of EDF (Electricité de France)[4]. Each item is a curve of 24 hourly points describing the way a customer consumes electric power during a day. The 24 consumption points are the active characteristics describing items. Much work has been done on clustering of curves (see [2] for some references) and we do not detail here the way of defining a similarity between two curves. The point is that clustering is performed with a similarity which is based on the shape of the curve.

[4] This dataset has been provided by the Service Etudes de Réseaux - Département Consommations, Clientèle et Télécommunications of our research center.

As for external characteristics describing curve items, we consider the observation day and season of the curve, the activity sector of the customer, as well as the characteristics of its contract with EDF. The goal of the study is to find classes of load curves according to their shape and to see the relationships between these classes and external characteristics.

The special feature with curves is that they have a nice graphical representation:

- each curve can be represented by a line plot,
- a set of curves can be represented by the mean curve or by juxtaposition of the curves.

Since each node of a clustering tree represents a set of items, it is possible to associate with each node a graphical representation of the set of curves belonging to it. Curves provide graphical representations which are not available with standard items. We will see in the next sections that the interface can be still very useful when considering standard items.

4 Browsing the clustering tree

4.1 Top-down approach

The basic feature provided by the interface is that the user can manipulate the clustering tree interactively by opening/closing nodes. At the very beginning of the browsing, only one node is presented to the user: the root of the tree which represents the whole set of items. Moreover, the model of the interface is that each node represents a set of items which are the items of the leaves of the node subtree. By selecting a node (by clicking on the mouse right-button), the user instructs the interface to display information about the set of items of the node.

Figure 1 is a screendump of the interface after several node openings performed by the user. Four windows are presented:

- the detailed tree window (upper-left) which displays open nodes of the tree: this is the window where the user can open/close nodes of the clustering tree,
- the attribute window (lower-left) which displays barcharts of the characteristics describing items (this window can be split into several windows - one window per attribute - on user request, as in Figure 1),
- the scaled tree window (upper-right) where the tree is displayed in a summarized way, but where heights of nodes are scaled exactly,
- the detailed selected node window (lower-right) where information is displayed about the selected node and its two subnodes.

In the top-down approach, the user begins with a single displayed node: the root of the tree. In the case of curves, this node box displays the mean curve (and +/- 1.5 standard error curves) of the whole dataset. These curves are only readable as an iconic representation but are very useful for the user. A readable representation is presented in the lower-right window: mean curves of the selected node and its two children are represented in this window. If the

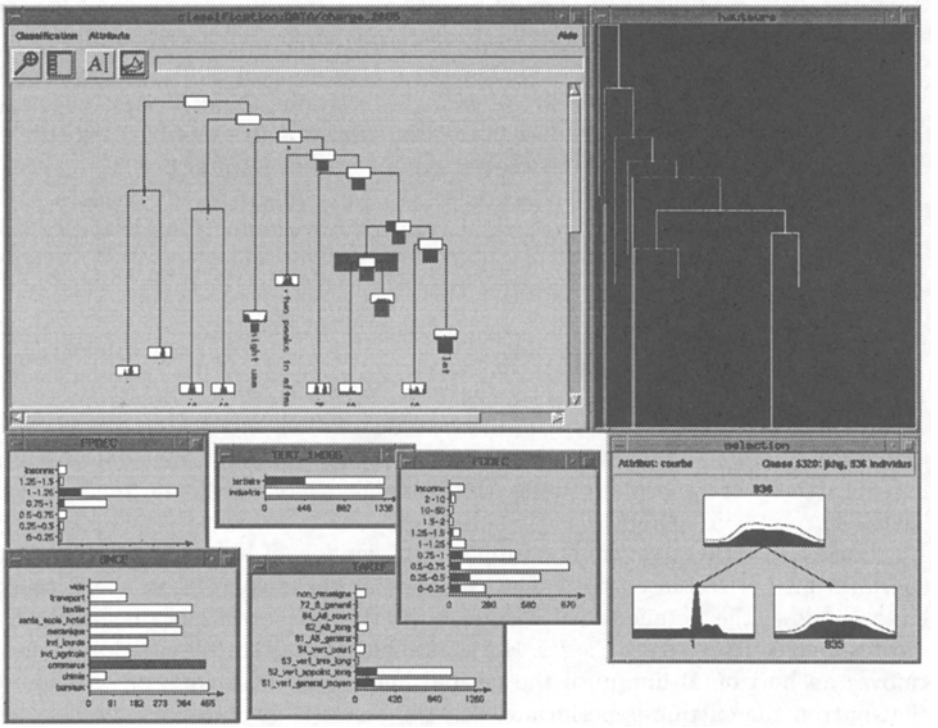

Fig. 1. Screendump of the interface

user wants even more detail about the selected node, it is possible to display a window figuring the juxtaposition of all curves under the node. All curves in that window are mouse sensitive so that the curve under the mouse pointer is instantly highlighted in red: it is thus possible to see the shape of some untypical curves among all the mixed curves.

The user can then click on the mouse middle-button to *open* the root node. The result of *opening* a node is that the two children of the node are displayed on the tree window. Figure 1 shows the state of the interface after opening several nodes. The presentation of the tree then is done according to the following rules:

- leaves of the *displayed tree*[5] contain the iconic mean curve of the corresponding node,
- intermediate nodes contain a box which is proportional to the number of items under the node: this box is split into two pieces which represent the respective sizes of the children nodes (in black for the left child and in white for the right child).

[5] The *displayed tree* is only a part of the clustering tree since some nodes of the clustering tree may be closed. So, *leaves* of the displayed tree may correspond either to items or to nodes of the clustering tree.

This representation enables the user to have an idea of what is happening when splitting the node into its two children: either the two children are well-balanced (as in the right part of the tree of Figure 1), or badly-balanced like the splits just under the root. The situation of badly-balanced splits near the root is very common in agglomerative hierarchical clustering since they correspond to very untypical items which cannot be aggregated with other items until the end of the clustering process. This possibility of seeing directly the shape of very untypical load curves of a set of curves has been particularly appreciated by experts of electric load curves.

So, in the top-down approach, end-users open nodes until they consider that the sets of curves of the left and right children are similar. The leaves of the displayed tree then correspond to a partition of items which is not necessarily an horizontal cutting of the clustering tree.

4.2 Bottom-up approach

Some users prefer a bottom-up approach to define their interesting classes. It is usually impossible to follow a bottom-up approach from the items themselves because there are too many items (for instance there are 2,665 items in our dataset).

Using the scaled tree window (upper-right window of Figure 1), the user can perform an horizontal cutting. The cut tree is then displayed on the detailed tree window: the detailed tree is opened so that leaves of the displayed tree correspond to the cutting classes. Horizontal cutting is very easy to do: the user moves an horizontal line until the desired height; when clicking on the mouse left-button the cutting is performed and the result is displayed on the detailed tree window. It is important to note that this horizontal cutting is done through the scaled tree window: in this window, the tree is represented with height of each node scaled correctly. We recall here that the height of each node is the key information for the end-user to decide where to cut the tree.

Once the user has performed an horizontal cutting into, say, 20 clusters, it is still possible to open/close displayed nodes. Closing nodes where there is too much detail (bottom-up approach) or opening nodes which are still not detailed enough (again the top-down approach).

To help the tree browsing, a zooming is always available in the detailed tree window. It is possible to easily zoom in and out the tree, and to resize the tree so that it fits completely into the window. Resizing the tree to fit into the window is done automatically after an horizontal cutting, but not after a node opening. Automatic resizing after a node opening appeared to be confusing for the user: such resizing is only done on request.

5 Interpretation with item characteristics

The tree browsing described in the previous section could be sufficient to determine a partition of a set of curves. This is due to the graphical representation

of curves. Because standard items do not have this property and because it is of great interest to study external characteristics of curves, an additional window has been designed to display information on the characteristics of the items under the currently selected node. This is the attribute window, which is on the lower-left part of the screen. This window can be one large window containing information about all item attributes or the user can ask for one small window per attribute (as in Figure 1). For each attribute describing items, a barchart is drawn. In the case of numerical attributes, histograms replace barcharts.

5.1 Linked Barcharts

Attribute barcharts are linked to the detailed tree browsing. When the user selects the root node, barcharts represent the distribution of attribute values for the whole set of items. Likewise, when the user selects a node on the detailed tree window, barcharts display the distribution of the items of the node (in black in Figure 1) compared to the distribution of the whole set of items (in white). In Figure 1, the selected node is highlighted in gray.

These linked barcharts have been very appreciated both with standard items where no curve can be displayed for a node, and with curves to visualize easily characteristics of the customers corresponding to the curves.

5.2 Test-values

Moreover, the user can click on bars of attribute barcharts. When the user clicks on a bar, it is equivalent to select all items having the corresponding attribute value in the dataset. Then, the interface displays the following information:

- all barcharts are instantly updated to show the distribution of selected items on the other attributes like in standard software (see for instance SAS-INSIGHT in [9]),
- information is displayed to show nodes of the tree for which the selected attribute value is characteristic (with a gray square) or anti-characteristic (with a black square).[6]

For instance, in Figure 1, the user has clicked on the bar labeled *"commerce"*. The effect is that all curve items corresponding to commercial customers are selected. Then, for every node of the displayed tree, a statistical indicator is computed to indicate if the set of items defined by the node have a high proportion of items labeled *"commerce"* or not. This indicator, called a *test-value* (see [8] for detail), is represented on the detailed tree window by a colored square with a size related to its absolute value. Large gray (resp. black) squares mean that the attribute value is very characteristic (resp. anti-characteristic) of the node, while small squares mean that there is nothing special.

[6] The screendump presented in Figure 1 is a grayscale representation of the interface. The interface actually displays red and blue squares instead of gray and black ones.

5.3 Interpretation dynamics

Standard use of linked histograms and test-values, which are themselves linked to the tree, are the following:

- when the user selects a node, linked histograms show characteristics of items under the selected node,
- if one characteristic is very frequent in the selected node, the user clicks on the corresponding bar to check if this characteristic appears only in the selected node, or may be present in some other nodes: this is represented by the test-value square associated with each node and displayed on the tree.

Combined with the possibility of opening/closing nodes of the tree, the user has thus an efficient tool to interpret some nodes of the tree. The key point for the user is to decide if a node has to be split into its two subnodes or not. Additional information than linked histograms and test-values are available to do so in the detailed selected node window (the lower-right window in Figure 1). The purpose of this window is to show if curves of the two subnodes are significatively different from curves of the selected node. In the case of standard data (where no curve is available), this window is replaced by a window displaying the barchart of the attribute which explains the most the splitting of the node. For this attribute, the node and its two subnodes barcharts are displayed in this window (see Figure 2). Barcharts of other attributes can be displayed on user request. Once a node has been interpreted, the user can associate with it a label which helps him (her) to remind already interpreted nodes. This facility is described in the next section.

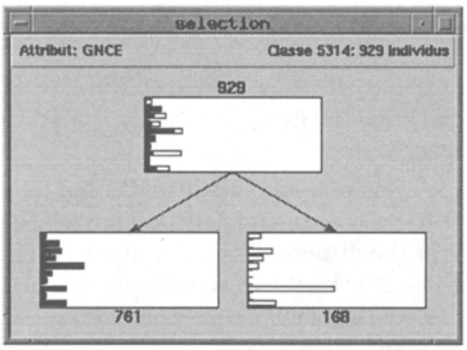

Fig. 2. Detailed selected node window on standard data

6 Building the final partition

As mentioned in Section 3, the final result of the user's interaction is the definition of a partition of items. This partitioning is performed in two steps:

- labeling some nodes of the tree,
- distributing labeled nodes into some disjoint classes we call *superclasses*: each superclass is defined as a collection of selected nodes of the tree.

6.1 Labeling

The labeling of one node can be done at any time by selecting a node and clicking on the labeling button of the detailed tree window. The user is prompted to enter a character string which is then displayed near the node in the detailed tree window when the node is closed. If the node is open, it is not displayed but still kept in memory. Leaves of the tree have default labels which are the item labels and which cannot be changed.

6.2 Collecting nodes

If the user selects the collecting tool, this tool window opens (see Figure 3). This window displays one superclass at a time (here superclass labeled *"Curves with high variations"*) but it is possible to switch easily to other superclasses. When creating a superclass, the user has to give a name. To fill a superclass the user selects nodes and uses a drag and drop facility to move them from the tree to the superclass. The node icon is then displayed in the superclass window as in Figure 3. When a node has been collected into a superclass, it is marked with a cross on the tree: unless removing it from the superclass, it becomes impossible to open it or to drag it to another superclass.

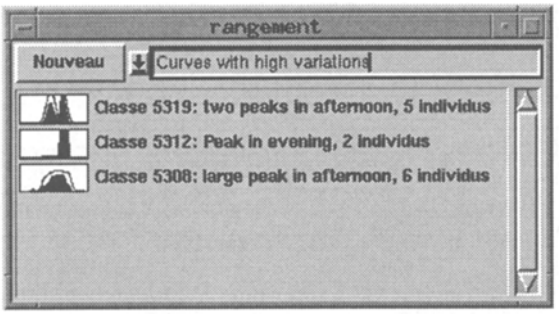

Fig. 3. Collecting tool window

Once the user has defined and filled several superclasses with labeled nodes extracted from the clustering tree, it is possible to export the result in the form of an ASCII file containing, for each item in the dataset: the item ID, its superclass label, and the possible node label if the superclass is composed of several nodes.

This file can then be exported to the user's favorite spreadsheet software to compute desired aggregates.

7 Conclusion and further work

Very little work has been done using recent advances in computer human interaction in the field of clustering interpretation (see [2], [6] for two alternative approaches). Our work is an attempt to introduce more sophisticated interaction tools to interpret the result of a hierarchical clustering.

The design of the interface has been guided by the needs of people who are experts in electric load curve clustering. They have judged that this kind of interface improves drastically their productivity in electric load curve analysis.

This prototype interface has been developed using Tcl and Tk public domain software. It has been used on a SPARC20 workstation on which all user's actions get immediate responses. This response time remains very good even with the dataset of 2,665 curves described by 24 hour consumption values and six external attributes featuring 80 different values. Good performance is obtained by pre-computing all statistical indicators before running the interface:

- all barcharts are pre-computed for each node,
- all test-values are pre-computed for each node and each attribute value,
- all mean load curves are pre-computed for each node.

Nevertheless, much work remains to be done in this area. Among the many directions worth investigating, we are currently working in two directions:

- building a similar tool but dedicated to interpretation of textual data clustering,
- building a similar tool to interpret the result of Kohonen map curve clustering (see [2]).

References

1. Becker R.A., Chambers J.M., Wilks A.R.: The New S Language. *Wadsworth, Pacific Grove*, California, 1988.
2. Chantelou D., Hébrail G., Muller C.: Visualizing 2,665 electric power load curves on a single A4 sheet of paper. *Proceedings of the ISAP'96 Conference*, Orlando, Florida, Jan.1996.
3. Celeux G., Diday E., Govaert G., Lechevallier Y., Ralambondrainy H., Classification automatique des données. *Dunod Informatique Bordas*, Paris, 1989.
4. Everitt B.S.: Cluster Analysis. *Edward Arnold*, London, 3rd Edition, 1993.
5. Hartigan J.A.: Clustering Algorithms. *Wiley New-York*, 1975.
6. Hébrail G. and Tanzy J.E.: Barcharts and Class Characterization with Taxonomic Qualitative Variables. *Proceedings of the COMPSTAT'96 Conference in Barcelona*, Physica-Verlag, 1996.
7. Michalski R.S., Carbonell J.G., Mitchell T.M., Kodratoff Y.: Apprentissage symbolique, une approche de l'intelligence artificielle. Volumes I and II, *Cepadues Editions*, 1993.
8. Morineau A.: Note sur la Caractérisation Statistique d'une Classe et les Valeurs-test. *Bulletin du CESIA*, Vol.2, N. 1-2, Paris, 1984.
9. SAS Version 6 User's Guides, *SAS Institute*.
10. Welch B.B.: Practical Programming in Tcl and Tk. *Prentice Hall PTR*, 1995.