# Chapter 118
## The TREE Procedure

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: TREE Procedure</td>
<td>9696</td>
</tr>
<tr>
<td>Getting Started: TREE Procedure</td>
<td>9696</td>
</tr>
<tr>
<td>Syntax: TREE Procedure</td>
<td>9702</td>
</tr>
<tr>
<td>PROC TREE Statement</td>
<td>9702</td>
</tr>
<tr>
<td>BY Statement</td>
<td>9708</td>
</tr>
<tr>
<td>COPY Statement</td>
<td>9708</td>
</tr>
<tr>
<td>FREQ Statement</td>
<td>9708</td>
</tr>
<tr>
<td>HEIGHT Statement</td>
<td>9709</td>
</tr>
<tr>
<td>ID Statement</td>
<td>9709</td>
</tr>
<tr>
<td>NAME Statement</td>
<td>9709</td>
</tr>
<tr>
<td>PARENT Statement</td>
<td>9709</td>
</tr>
<tr>
<td>Details: TREE Procedure</td>
<td>9710</td>
</tr>
<tr>
<td>Missing Values</td>
<td>9710</td>
</tr>
<tr>
<td>Output Data Set</td>
<td>9710</td>
</tr>
<tr>
<td>Displayed Output</td>
<td>9710</td>
</tr>
<tr>
<td>ODS Table Names</td>
<td>9711</td>
</tr>
<tr>
<td>Examples: TREE Procedure</td>
<td>9712</td>
</tr>
<tr>
<td>Example 118.1: Mammals’ Teeth</td>
<td>9712</td>
</tr>
<tr>
<td>Example 118.2: Iris Data</td>
<td>9719</td>
</tr>
<tr>
<td>References</td>
<td>9720</td>
</tr>
</tbody>
</table>
Overview: TREE Procedure

The TREE procedure reads a data set created by the CLUSTER or VARCLUS procedure and produces a tree diagram (also known as a dendrogram or phenogram), which displays the results of a hierarchical clustering analysis as a tree structure. The TREE procedure uses the data set to produce a diagram of the tree structure in the style of Johnson (1967), with the root at the top. Alternatively, the diagram can be oriented horizontally, with the root at the left. Any numeric variable in the output data set can be used to specify the heights of the clusters. PROC TREE can also create an output data set that contains a variable to indicate the disjoint clusters at a specified level in the tree.

Tree diagrams are discussed in the context of cluster analysis by Duran and Odell (1974); Hartigan (1975); Everitt (1980). Knuth (1973) provides a general treatment of tree diagrams in computer programming.

The literature on tree diagrams contains a mixture of botanical and genealogical terminology. The objects that are clustered are leaves. The cluster that contains all objects is the root. A cluster that contains at least two objects but not all of them is a branch. The general term for leaves, branches, and roots is node. If a cluster A is the union of clusters B and C, then A is the parent of B and C, and B and C are children of A. A leaf is thus a node with no children, and a root is a node with no parent. If every cluster has at most two children, the tree diagram is a binary tree. The CLUSTER procedure always produces binary trees. The VARCLUS procedure can produce tree diagrams with clusters that have many children.

Getting Started: TREE Procedure

The TREE procedure creates tree diagrams from a SAS data set that contains the tree structure. You can create this type of data set with the CLUSTER or VARCLUS procedure. See Chapter 33, “The CLUSTER Procedure,” and Chapter 120, “The VARCLUS Procedure,” for more information.

In the following example, the VARCLUS procedure is used to divide a set of variables into hierarchical clusters and to create the SAS data set that contains the tree structure. The TREE procedure then generates the tree diagrams.

The following data, from Hand et al. (1994), represent the amount of protein consumed from nine food groups for each of 25 European countries. The nine food groups are red meat (RedMeat), white meat (WhiteMeat), eggs (Eggs), milk (Milk), fish (Fish), cereal (Cereal), starch (Starch), nuts (Nuts), and fruits and vegetables (FruVeg).
The following SAS statements create the data set Protein:

```sas
data Protein;
  input Country $15. RedMeat WhiteMeat Eggs Milk Fish Cereal Starch Nuts FruVeg;
  datalines;
  Albania 10.1 1.4 0.5 8.9 0.2 42.3 0.6 5.5 1.7
  Austria 8.9 14.0 4.3 19.9 2.1 28.0 3.6 1.3 4.3
  Belgium 13.5 9.3 4.1 17.5 4.5 26.6 5.7 2.1 4.0
  Bulgaria 7.8 6.0 1.6 8.3 1.2 56.7 1.1 3.7 4.2
  Czechoslovakia 9.7 11.4 2.8 12.5 2.0 34.3 5.0 1.1 4.0
  Denmark 10.6 10.8 3.7 25.0 9.9 21.9 4.8 0.7 2.4
  E Germany 8.4 11.6 3.7 11.1 5.4 24.6 6.5 0.8 3.6
  Finland 9.5 4.9 2.7 33.7 5.8 26.3 5.1 1.0 1.4
  France 18.0 9.9 3.3 19.5 5.7 28.1 4.8 2.4 6.5
  Greece 10.2 3.0 2.8 17.6 5.9 41.7 2.2 7.8 6.5
  Hungary 5.3 12.4 2.9 9.7 0.3 40.1 4.0 5.4 4.2
  Ireland 13.9 10.0 4.7 25.8 2.2 24.0 6.2 1.6 2.9
  Italy 9.0 5.1 2.9 13.7 3.4 36.8 2.1 4.3 6.7
  Netherlands 9.5 13.6 3.6 23.4 2.5 22.4 4.2 1.8 3.7
  Norway 9.4 4.7 2.7 23.3 9.7 23.0 4.6 1.6 2.7
  Poland 6.9 10.2 2.7 19.3 3.0 36.1 5.9 2.0 6.6
  Portugal 6.2 3.7 1.1 4.9 14.2 27.0 5.9 4.7 7.9
  Romania 6.2 6.3 1.5 11.1 1.0 49.6 3.1 5.3 2.8
  Spain 7.1 3.4 3.1 8.6 7.0 29.2 5.7 5.9 7.2
  Sweden 9.9 7.8 3.5 4.7 7.5 19.5 3.7 1.4 2.0
  Switzerland 13.1 10.1 3.1 23.8 2.3 25.6 2.8 2.4 4.9
  UK 17.4 5.7 4.7 20.6 4.3 24.3 4.7 3.4 3.3
  USSR 9.3 4.6 2.1 16.6 3.0 43.6 6.4 3.4 2.9
  W Germany 11.4 12.5 4.1 18.8 3.4 18.6 5.2 1.5 3.8
  Yugoslavia 4.4 5.0 1.2 9.5 0.6 55.9 3.0 5.7 3.2
;```

The data set Protein contains the character variable Country and the nine numeric variables that represent the food groups. The $15.$ in the INPUT statement specifies that the variable Country is a character variable with a length of 15.

The following statements cluster the variables in the data set Protein:

```sas
ods graphics on;

proc varclus data=Protein outtree=Tree centroid maxclusters=4
  plots=dendrogram(vertical height=ncl);
  var RedMeat--FruVeg;
run;
```

The OUTFREE= option creates an output data set named Tree to contain the tree structure. The CENTROID option specifies the centroid clustering method, and the MAXCLUSTERS= option specifies that the largest number of clusters desired is four. The VAR statement specifies that all numeric variables (RedMeat—FruVeg) are used by the procedure. Since ODS Graphics is enabled, PROC VARCLUS creates a dendrogram, which is displayed in Figure 118.1. The option plots=dendrogram(vertical height=ncl) specifies a vertical dendrogram with the number of clusters on the vertical axis. The default is a horizontal dendrogram with, for this cluster analysis, the proportion of variance explained on the horizontal axis.
The output data set `Tree`, created by the OUTTREE= option in the previous statements, contains the following variables:

- `_NAME_` the name of the cluster
- `_PARENT_` the parent of the cluster
- `_NCL_` the number of clusters
- `_VAREXP_` the amount of variance explained by the cluster
- `_PROPOR_` the proportion of variance explained by the clusters at the current level of the tree diagram
- `_MINPRO_` the minimum proportion of variance explained by a cluster
- `_MAXEIGEN_` the maximum second eigenvalue of a cluster
The following statements use PROC TREE to produce a tree diagram of the clusters created by PROC VARCLUS:

```plaintext
proc tree data=tree vaxis=axis1;
  axis1 label=(angle=90);
run;
```

The AXIS1 statement rotates the Y-axis label so that it is displayed in the same way as in the dendrogram from PROC VARCLUS. Figure 118.2 displays the tree diagram.

In each diagram, the name of the cluster is displayed on the horizontal axis and the number of clusters is displayed on the vertical (height) axis.
As you look up from the bottom of either diagram, clusters are progressively joined until a single, all-encompassing cluster is formed at the top (or root) of the tree. Clusters exist at each level of the diagram. For example, at the level where the diagram indicates three clusters, the clusters are as follows:

- Cluster 1: RedMeat WhiteMeat Eggs Milk
- Cluster 2: Fish Starch
- Cluster 3: Cereal Nuts FruVeg

As you proceed up the diagram one level, the number of clusters is two:

- Cluster 1: RedMeat WhiteMeat Eggs Milk Fish Starch
- Cluster 2: Cereal Nuts FruVeg

The following statements illustrate how you can specify the numeric variable that defines the height of each node (cluster) in the tree:

```sas
proc tree data=Tree horizontal haxis=axis1;
  axis1 order=(0 to 1 by 0.2);
  height _PROPOR_;
  label _name_ = 'Variable';
run;
```

The HORIZONTAL option in the PROC TREE statement orients the tree diagram horizontally. The HAXIS= option specifies that the AXIS1 statement be used to customize the appearance of the horizontal axis. The ORDER= option in the AXIS1 statement (see SAS/GRAPH: Reference) specifies the data values in the order in which they are to appear on the axis. The HEIGHT statement specifies the variable _PROPOR_ (the proportion of variance explained) as the height variable. The LABEL statement supplies a short Y-axis label. The resulting tree diagram is shown in Figure 118.3.
As you look from left to right in the diagram, objects and clusters are progressively joined until a single, all-encompassing cluster is formed at the right (or root) of the tree.

Clusters exist at each level of the diagram, represented by horizontal line segments. Each vertical line segment represents a point where leaves and branches are connected into progressively larger clusters.

For example, three clusters are formed at the leftmost point along the axis where three horizontal line segments exist. At that point, where a vertical line segment connects the Cereal-Nuts and FruVeg clusters, the proportion of variance explained is about 0.6 (_PROPOR_ = 0.6). At the next clustering level the variables Fish and Starch are clustered with variables RedMeat through Milk, resulting in a total of two clusters. The proportion of variance explained is about 0.45 at that point.
Syntax: TREE Procedure

The following statements are available in the TREE procedure:

```
PROC TREE <options> ;
   NAME variables ;
   HEIGHT variable ;
   PARENT variables ;
   BY variables ;
   COPY variables ;
   FREQ variable ;
   ID variable ;
```

If the input data set has been created by CLUSTER or VARCLUS, the only statement required is the PROC TREE statement. The BY, COPY, FREQ, HEIGHT, ID, NAME, and PARENT statements are described after the PROC TREE statement.

PROC TREE Statement

```
PROC TREE <options> ;
```

The PROC TREE statement invokes the TREE procedure.

Table 118.1 summarizes the options available in the PROC TREE statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Sets</strong></td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>specifies input data set</td>
</tr>
<tr>
<td>DOCK=</td>
<td>specifies that small clusters not be counted in OUT= data set</td>
</tr>
<tr>
<td>LEVEL=</td>
<td>defines disjoint cluster in OUT= data set</td>
</tr>
<tr>
<td>NCLUSTERS=</td>
<td>specifies number of clusters in OUT= data set</td>
</tr>
<tr>
<td>OUT=</td>
<td>specifies output data set</td>
</tr>
<tr>
<td>ROOT=</td>
<td>displays root of a subtree</td>
</tr>
<tr>
<td><strong>Cluster Heights</strong></td>
<td></td>
</tr>
<tr>
<td>HEIGHT=</td>
<td>specifies variable for the height axis</td>
</tr>
<tr>
<td>DISSIMILAR</td>
<td>specifies that height values indicate dissimilarity</td>
</tr>
<tr>
<td>SIMILAR</td>
<td>specifies that height values indicate similarity</td>
</tr>
<tr>
<td><strong>Horizontal Trees</strong></td>
<td></td>
</tr>
<tr>
<td>HORIZONTAL</td>
<td>specifies that height axis be horizontal</td>
</tr>
<tr>
<td><strong>Sort Order</strong></td>
<td></td>
</tr>
<tr>
<td>DESCENDING</td>
<td>reverses sort order</td>
</tr>
<tr>
<td>SORT</td>
<td>sorts children by HEIGHT variable</td>
</tr>
<tr>
<td><strong>Displayed Output</strong></td>
<td></td>
</tr>
<tr>
<td>INC=</td>
<td>specifies increment between tick values</td>
</tr>
</tbody>
</table>
### Table 118.1  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINEPRINTER</td>
<td>displays tree by using line printer graphics</td>
</tr>
<tr>
<td>LIST</td>
<td>displays all nodes in tree</td>
</tr>
<tr>
<td>MAXHEIGHT=</td>
<td>specifies maximum value on axis</td>
</tr>
<tr>
<td>MINHEIGHT=</td>
<td>specifies minimum value on axis</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>suppresses display of tree</td>
</tr>
<tr>
<td>NTICK=</td>
<td>specifies number of tick intervals</td>
</tr>
<tr>
<td><strong>Graphics</strong></td>
<td></td>
</tr>
<tr>
<td>CFRAME=</td>
<td>specifies color of the frame</td>
</tr>
<tr>
<td>DESCRIPTION=</td>
<td>specifies catalog description</td>
</tr>
<tr>
<td>GOUT=</td>
<td>specifies catalog name</td>
</tr>
<tr>
<td>HAXIS=</td>
<td>customizes horizontal axis</td>
</tr>
<tr>
<td>HORDISPLAY=</td>
<td>displays horizontal tree with leaves on right</td>
</tr>
<tr>
<td>HPAGES=</td>
<td>specifies number of pages to expand tree horizontally</td>
</tr>
<tr>
<td>LINES=</td>
<td>specifies line color and thickness, dots at nodes</td>
</tr>
<tr>
<td>NAME=</td>
<td>specifies name of graph in catalog</td>
</tr>
<tr>
<td>VAXIS=</td>
<td>customizes vertical axis</td>
</tr>
<tr>
<td>VPAGES=</td>
<td>specifies number of pages to expand tree vertically</td>
</tr>
<tr>
<td><strong>Line Printer Graphics</strong></td>
<td></td>
</tr>
<tr>
<td>PAGES=</td>
<td>specifies number of pages</td>
</tr>
<tr>
<td>POS=</td>
<td>specifies number of column positions</td>
</tr>
<tr>
<td>SPACES=</td>
<td>specifies number of spaces between objects</td>
</tr>
<tr>
<td>TICKPOS=</td>
<td>specifies number of column positions between ticks</td>
</tr>
<tr>
<td>FILLCHAR=</td>
<td>specifies fill character between unjoined leaves</td>
</tr>
<tr>
<td>JOINCHAR=</td>
<td>specifies character displayed between joined leaves</td>
</tr>
<tr>
<td>LEAFCHAR=</td>
<td>specifies character representing clusters with no children</td>
</tr>
<tr>
<td>TREECHAR=</td>
<td>specifies character representing clusters with children</td>
</tr>
</tbody>
</table>

**CFRAME=** *color*

specifies a color for the frame, which is the rectangle bounded by the axes.

**DATA=** *SAS-data-set*

specifies the input data set that defines the tree. If you omit the DATA= option, the most recently created SAS data set is used.

**DESCENDING**

**DES**

reverses the sort order for the SORT option.

**DESCRIPTION=** *entry-description*

specifies a description for the graph in the GOUT= catalog. The default is “Proc Tree Graph Output.”
DISSIMILAR
DIS
specifies that the values of the HEIGHT variable are dissimilarities; that is, a large height value means that the clusters are very dissimilar or far apart.

If neither the SIMILAR nor the DISSIMILAR option is specified, PROC TREE attempts to infer from the data whether the height values are similarities or dissimilarities. If PROC TREE cannot tell this from the data, it issues an error message and does not display a tree diagram.

DOCK=n
causes observations in the OUT= data set that have a frequency of n or less to be given missing values for the output variables CLUSTER and CLUSNAME. If the NCLUSTERS= option is also specified, DOCK= also prevents clusters with a frequency of n or less from being counted toward the number of clusters requested by the NCLUSTERS= option. By default, DOCK=0.

FILLCHAR=’c’
FC=’c’
specifies the character displayed between leaves that are not joined into a cluster. The character should be enclosed in single quotes. The default is a blank. The LINEPRINTER option must also be specified.

GOUT=<libref.>member-name
specifies the catalog in which the generated graph is stored. The default is Work.Gseg.

HAXIS=AXISn
specifies that the AXISn statement be used to customize the appearance of the horizontal axis.

HEIGHT=name
H=name
specifies certain conventional variables to be used for the height axis of the tree diagram. For many situations, the only option you need is the HEIGHT= option. Valid values for name and their meanings are as follows:

<table>
<thead>
<tr>
<th>HEIGHT</th>
<th>H</th>
<th>specifies the <em>HEIGHT</em> variable.</th>
</tr>
</thead>
<tbody>
<tr>
<td>LENGTH</td>
<td>L</td>
<td>defines the height of each node as its path length from the root. This can also be interpreted as the number of ancestors of the node.</td>
</tr>
<tr>
<td>MODE</td>
<td>M</td>
<td>specifies the <em>MODE</em> variable.</td>
</tr>
<tr>
<td>NCL</td>
<td>N</td>
<td>specifies the <em>NCL</em> (number of clusters) variable.</td>
</tr>
<tr>
<td>RSQ</td>
<td>R</td>
<td>specifies the <em>RSQ</em> variable.</td>
</tr>
</tbody>
</table>

See also the section “HEIGHT Statement” on page 9709. The HEIGHT statement can specify any variable in the input data set to be used for the height axis. In rare cases, you might need to specify either the DISSIMILAR option or the SIMILAR option.

HORDISPLAY=RIGHT
specifies that the graph be oriented horizontally with the leaf nodes on the right side, when the HORIZONTAL option is also specified. By default, the leaf nodes are on the left side.
PROC TREE Statement ➤ 9705

HORIZONTAL
HOR displays the tree diagram with the height axis oriented horizontally. The leaf nodes are on the side specified in the HORDISPLAY= option. If you do not specify the HORIZONTAL option, the height axis is vertical, with the root at the top. When the tree takes up more than one page, horizontal orientation can make the tree diagram considerably easier to read.

HPAGES=n1 specifies that the original graph be enlarged to cover n1 pages. If you also specify the VPAGES=n2 option, the original graph is enlarged to cover n1 × n2 graphs. For example, if HPAGES=2 and VPAGES=3, then the original graph is generated, followed by 2 × 3 = 6 more graphs. In these six graphs, the original is enlarged by a factor of 2 in the horizontal direction and by a factor of 3 in the vertical direction. The graphs are generated in left-to-right and top-to-bottom order.

INC=n specifies the increment between tick values on the height axis. If the HEIGHT variable is _NCL_, the default is usually 1, although a different value can be specified for consistency with other options. For any other HEIGHT variable, the default is some power of 10 times 1, 2, 2.5, or 5.

JOINCHAR=’c’  JC=’c’ specifies the character displayed between leaves that are joined into a cluster. The character should be enclosed in single quotes. The default is ‘X’. The LINEPRINTER option must also be specified.

LEAFCHAR=’c’  LC=’c’ specifies the character used to represent clusters that have no children. The character should be enclosed in single quotes. The default is a period. The LINEPRINTER option must also be specified.

LEVEL=n specifies the level of the tree that defines disjoint clusters for the OUT= data set. The LEVEL= option also causes only clusters between the root and a height of n to be displayed. The clusters in the output data set are those that exist at a height of n on the tree diagram. For example, if the HEIGHT variable is _NCL_ (number of clusters) and LEVEL=5 is specified, then the OUT= data set contains five disjoint clusters. If the HEIGHT variable is _RSQ_ (R square) and LEVEL=0.9 is specified, then the OUT= data set contains the smallest number of clusters that yields an R square of at least 0.9.

LINEPRINTER specifies that the tree diagram be displayed using line printer graphics.

LINES=( < COLOR=color > < WIDTH=n > < DOTS >) specifies the color and the thickness of the lines of the tree, and whether a dot is drawn at each leaf node. If the frame and the lines are specified to be the same color, PROC TREE selects a different color for the lines.

LIST lists all the nodes in the tree, displaying the height, parent, and children of each node.
**MAXHEIGHT** = n
**MAXH** = n
specifies the maximum value displayed on the height axis.

**MINHEIGHT** = n
**MINH** = n
specifies the minimum value displayed on the height axis.

**NAME** = name
specifies the entry name for the generated graph in the GOUT= catalog. Each time another graph is generated with the same name, the name is modified by appending a number to make it unique.

**NCLUSTERS** = n
**NCL** = n
**N** = n
specifies the number of clusters desired in the OUT= data set. The number of clusters obtained might not equal the number specified if (1) there are fewer than n leaves in the tree, (2) there are more than n unconnected trees in the data set, (3) a multiway tree does not contain a level with the specified number of clusters, or (4) the DOCK= option eliminates too many clusters.

The NCLUSTERS= option uses the _NCL_ variable to determine the order in which the clusters are formed. If there is no _NCL_ variable, the height variable (as determined by the HEIGHT statement or HEIGHT= option) is used instead.

**NTICK** = n
specifies the number of tick intervals on the height axis. The default depends on the values of other options.

**NOPRINT**
suppresses the display of the tree. Specify the NOPRINT option if you want only to create an OUT= data set.

**OUT** = SAS-data-set
creates an output data set that contains one observation for each object in the tree or subtree being processed and variables called CLUSTER and CLUSNAME that show cluster membership at any specified level in the tree. If you specify the OUT= option, you must also specify either the NCLUSTERS= or LEVEL= option in order to define the output partition level. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Language Reference: Concepts*.

**PAGES** = n
specifies the number of pages over which the tree diagram (from root to leaves) is to extend. The default is 1. The LINEPRINTER option must also be specified.

**POS** = n
specifies the number of column positions on the height axis. The default depends on the value of the PAGES= option, the orientation of the tree diagram, and the values specified by the PAGESIZE= and LINESIZE= options. The LINEPRINTER option must also be specified.
**ROOT= ’name’**

specifies the value of the NAME statement variable for the root of a subtree to be displayed if you do not want to display the entire tree. If you also specify the OUT= option, the output data set contains only objects that belong to the subtree specified by the ROOT= option.

**SIMILAR**

**SIM**

specifies that the values of the HEIGHT variable represent similarities; that is, a large height value means that the clusters are very similar or close together.

If neither the SIMILAR nor the DISSIMILAR option is specified, PROC TREE attempts to infer from the data whether the height values are similarities or dissimilarities. If PROC TREE cannot tell this from the data, it issues an error message and does not display a tree diagram.

**SORT**

sorts the children of each node by the HEIGHT variable, in the order of cluster formation. See the DESCENDING option for details.

**SPACES=s**

**S=s**

specifies the number of spaces between objects in the output. The default depends on the number of objects, the orientation of the tree diagram, and the values specified by the PAGESIZE= and LINESIZE= options. The LINEPRINTER option must also be specified.

**TICKPOS=n**

specifies the number of column positions per tick interval on the height axis. The default value is usually between 5 and 10, although a different value can be specified for consistency with other options.

**TREECHAR= ’c’**

**TC= ’c’**

specifies the character used to represent clusters with children. The character should be enclosed in single quotes. The default is ‘X’. The LINEPRINTER option must also be specified.

**VAXIS=AXISn**

specifies that the AXISn statement be used to customize the appearance of the vertical axis.

**VPAGES=n2**

specifies that the original graph be enlarged to cover n2 pages. If you also specify the HPAGES=n1 option, the original graph is enlarged to cover n1 × n2 pages. For example, if HPAGES=2 and VPAGES=3, then the original graph is generated, followed by 2 × 3 = 6 more graphs. In these six graphs, the original is enlarged by a factor of 2 in the horizontal direction and by a factor of 3 in the vertical direction. The graphs are generated in left-to-right and top-to-bottom order.
BY Statement

```
BY variables;
```

You can specify a BY statement with PROC TREE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement for the TREE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the discussion in SAS Language Reference: Concepts. For more information about the DATASETS procedure, see the discussion in the Base SAS Procedures Guide.

COPY Statement

```
COPY variables;
```

The COPY statement specifies one or more character or numeric variables to be copied to the OUT= data set.

FREQ Statement

```
FREQ variable;
```

The FREQ statement specifies one numeric variable that tells how many clustering observations belong to the cluster. If the FREQ statement is omitted, PROC TREE uses the variable _FREQ_ to specify the number of observations per cluster. If neither the FREQ statement nor the _FREQ_ variable is present, each leaf is assumed to represent one clustering observation, and the frequency for each internal node is found by summing the frequencies of its children.
HEIGHT Statement

**HEIGHT** variable ;

The **HEIGHT** statement specifies the name of a numeric variable to define the height of each node (cluster) in the tree. The height variable can also be specified by the **HEIGHT=** option in the **PROC TREE** statement. If both the **HEIGHT** statement and the **HEIGHT=** option are omitted, **PROC TREE** uses the variable **_HEIGHT_.** If the data set does not contain **_HEIGHT_**, **PROC TREE** uses the variable **_NCL_.** If **_NCL_** is not present either, the height of each node is defined to be its path length from the root.

ID Statement

**ID** variable ;

The **ID** variable is used to identify the objects (leaves) in the tree on the output. The **ID** variable can be a character or numeric variable of any length. If the **ID** statement is omitted, the variable in the **NAME** statement is used instead. If both the **ID** and **NAME** statements are omitted, **PROC TREE** uses the variable **_NAME_.** If the **_NAME_** variable is not found in the data set, **PROC TREE** issues an error message and stops. The **ID** variable is copied to the **OUT=** data set.

NAME Statement

**NAME** variable ;

The **NAME** statement specifies a character or numeric variable that identifies the node represented by each observation. The **NAME** statement variable and the **PARENT** statement variable jointly define the tree structure. If the **NAME** statement is omitted, **PROC TREE** uses the variable **_NAME_.** If the **_NAME_** variable is not found in the data set, **PROC TREE** issues an error message and stops.

PARENT Statement

**PARENT** variable ;

The **PARENT** statement specifies a character or numeric variable that identifies the node in the tree that is the parent of each observation. The **PARENT** statement variable must have the same formatted length as the **NAME** statement variable. If the **PARENT** statement is omitted, **PROC TREE** uses the variable **_PARENT_.** If the **_PARENT_** variable is not found in the data set, **PROC TREE** issues an error message and stops.
Details: TREE Procedure

Missing Values

An observation with a missing value for the NAME statement variable is omitted from processing. If the PARENT statement variable has a missing value but the NAME statement variable is present, the observation is treated as the root of a tree. A data set can contain several roots and, hence, several trees.

Missing values of the HEIGHT variable are set to upper or lower bounds determined from the nonmissing values under the assumption that the heights are monotonic with respect to the tree structure.

Missing values of the FREQ variable are inferred from nonmissing values where possible; otherwise, they are treated as zero.

Output Data Set

The OUT= data set contains one observation for each leaf in the tree or subtree being processed. The variables are as follows:

- the BY variables, if any
- the ID variable, or the NAME statement variable if the ID statement is not used
- the COPY variables
- a numeric variable CLUSTER that takes values from 1 to \( c \), where \( c \) is the number of disjoint clusters. The cluster to which the first observation belongs is given the number 1, the cluster to which the next observation belongs that does not belong to cluster 1 is given the number 2, and so on.
- a character variable CLUSNAME that gives the value of the NAME statement variable of the cluster to which the observation belongs

The CLUSTER and CLUSNAME variables are missing if the corresponding leaf has a nonpositive frequency.

Displayed Output

The displayed output from the TREE procedure includes the following:

- the names of the objects in the tree
- the height axis
- the tree diagram.
The leaves of the tree diagram are displayed at the bottom of the graph. Horizontal lines connect the leaves into branches, while the topmost horizontal line indicates the root.

If the LINEPRINTER option is specified, the root (the cluster that contains all the objects) is indicated by a solid line of the character specified by the TREECHAR= option (the default character is ‘X’). At each level of the tree, clusters are shown by unbroken lines of the TREECHAR= symbol with the FILLCHAR= symbol (the default is a blank) separating the clusters. The LEAFCHAR= symbol (the default character is a period) represents single-member clusters.

By default, the tree diagram is oriented with the height axis vertical and the object names at the top of the diagram. If the HORIZONTAL option is specified, then the height axis is horizontal and the object names are on the left.

### ODS Table Names

PROC TREE assigns a name to each table it creates. You can use table names to refer to tables when using the Output Delivery System (ODS) to select tables and create output data sets. The name of PROC TREE’s only table is listed in Table 118.2. For more information about ODS, see Chapter 20, “Using the Output Delivery System.”

<table>
<thead>
<tr>
<th>ODS Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>TreeListing</td>
<td>Listing of all nodes in the tree</td>
<td>PROC</td>
<td>LIST</td>
</tr>
</tbody>
</table>
Examples: TREE Procedure

Example 118.1: Mammals' Teeth

The following statements produce a data set that contains the numbers of different kinds of teeth for a variety of mammals:

```sas
data teeth;
  title 'Mammals' Teeth';
  input mammal $16. v1-v8 @@;
  label v1='Right Top Incisors'
    v2='Right Bottom Incisors'
    v3='Right Top Canines'
    v4='Right Bottom Canines'
    v5='Right Top Premolars'
    v6='Right Bottom Premolars'
    v7='Right Top Molars'
    v8='Right Bottom Molars';
datalines;
Brown Bat   2 3 1 1 3 3 3 3
Silver Hair Bat 2 3 1 1 2 3 3 3
House Bat    2 3 1 1 2 3 3 3
Pika         2 1 0 0 2 3 3 3
Beaver       1 1 0 0 2 1 3 3
Gray Squirrel1 1 0 0 1 1 3 3
Porcupine   1 1 0 0 1 1 3 3
Bear         3 3 1 1 4 4 2 3
Marten       3 3 1 1 4 4 1 2
Wolverine   3 3 1 1 4 4 1 2
River Otter 3 3 1 1 4 3 1 2
Jaguar       3 3 1 1 3 2 1 1
Fur Seal     3 2 1 1 4 4 1 1
Grey Seal   3 2 1 1 3 3 2 2
Reindeer    0 4 1 0 3 3 3 3
Deer         0 4 0 0 3 3 3 3
Mole         3 2 1 0 3 3 3 3
Pigmy Bat    2 3 1 1 2 3 3 3
Red Bat      1 3 1 1 2 3 3 3
Rabbit       2 1 0 0 3 2 3 3
Groundhog    1 1 0 0 2 1 3 3
House Mouse  1 1 0 0 0 0 3 3
Wolf         3 3 1 1 4 4 2 3
Raccoon      3 3 1 1 4 4 3 2
Weasel       3 3 1 1 3 3 1 2
Badger       3 3 1 1 3 3 1 2
Sea Otter    3 2 1 1 3 3 1 2
Cougar       3 3 1 1 3 2 1 1
Sea Lion     3 2 1 1 4 4 1 1
Elephant Seal2 1 1 4 4 1 1
Elk          0 4 1 0 3 3 3 3
Moose        0 4 0 0 3 3 3 3
```

The following statements use the CLUSTER procedure to cluster the mammals by average linkage and use ODS Graphics and the TREE procedure to produce a horizontal tree diagram that uses the average-linkage distance as its height axis:

```sas
ods graphics on;
proc cluster method=average std pseudo noeigen outtree=tree;
  id mammal;
  var v1-v8;
run;
proc tree horizontal;
  label _name_ = 'Animal';
run;
```
Output 118.1.1 displays the information about how the clusters are joined. For example, the cluster history shows that the mammals ‘Wolf’ and ‘Bear’ form cluster 29, which is merged with ‘Raccoon’ to form cluster 11.

### Output 118.1.1 Output from PROC CLUSTER

#### Mammals’ Teeth

**The CLUSTER Procedure**

**Average Linkage Cluster Analysis**

The data have been standardized to mean 0 and variance 1

| Root-Mean-Square Total-Sample Standard Deviation | 1 |
| Root-Mean-Square Distance Between Observations | 4 |

<table>
<thead>
<tr>
<th>Number of Clusters</th>
<th>Clusters Joined</th>
<th>Freq</th>
<th>Pseudo F Statistic</th>
<th>Pseudo F Squared</th>
<th>Norm RMS Distance</th>
<th>Tie</th>
</tr>
</thead>
<tbody>
<tr>
<td>31 Beaver</td>
<td>Groundhog</td>
<td>2</td>
<td>.</td>
<td>.</td>
<td>0</td>
<td>T</td>
</tr>
<tr>
<td>30 Gray Squirrel</td>
<td>Porcupine</td>
<td>2</td>
<td>.</td>
<td>.</td>
<td>0</td>
<td>T</td>
</tr>
<tr>
<td>29 Wolf</td>
<td>Bear</td>
<td>2</td>
<td>.</td>
<td>.</td>
<td>0</td>
<td>T</td>
</tr>
<tr>
<td>28 Marten</td>
<td>Wolverine</td>
<td>2</td>
<td>.</td>
<td>.</td>
<td>0</td>
<td>T</td>
</tr>
<tr>
<td>27 Weasel</td>
<td>Badger</td>
<td>2</td>
<td>.</td>
<td>.</td>
<td>0</td>
<td>T</td>
</tr>
<tr>
<td>26 Jaguar</td>
<td>Cougar</td>
<td>2</td>
<td>.</td>
<td>.</td>
<td>0</td>
<td>T</td>
</tr>
<tr>
<td>25 Fur Seal</td>
<td>Sea Lion</td>
<td>2</td>
<td>.</td>
<td>.</td>
<td>0</td>
<td>T</td>
</tr>
<tr>
<td>24 Reindeer</td>
<td>Elk</td>
<td>2</td>
<td>.</td>
<td>.</td>
<td>0</td>
<td>T</td>
</tr>
<tr>
<td>23 Deer</td>
<td>Moose</td>
<td>2</td>
<td>.</td>
<td>.</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>22 Pigmy Bat</td>
<td>Red Bat</td>
<td>2</td>
<td>281</td>
<td>.</td>
<td>0.2289</td>
<td></td>
</tr>
<tr>
<td>21 CL28</td>
<td>River Otter</td>
<td>3</td>
<td>139</td>
<td>.</td>
<td>0.2292</td>
<td></td>
</tr>
<tr>
<td>20 CL31</td>
<td>CL30</td>
<td>4</td>
<td>83.2</td>
<td>.</td>
<td>0.2357 T</td>
<td></td>
</tr>
<tr>
<td>19 Brown Bat</td>
<td>Silver Hair Bat</td>
<td>2</td>
<td>76.7</td>
<td>.</td>
<td>0.2357 T</td>
<td></td>
</tr>
<tr>
<td>18 Pika</td>
<td>Rabbit</td>
<td>2</td>
<td>73.2</td>
<td>.</td>
<td>0.2357</td>
<td></td>
</tr>
<tr>
<td>17 CL27</td>
<td>Sea Otter</td>
<td>3</td>
<td>67.4</td>
<td>.</td>
<td>0.2462</td>
<td></td>
</tr>
<tr>
<td>16 CL22</td>
<td>House Bat</td>
<td>3</td>
<td>62.9</td>
<td>1.7</td>
<td>0.2859</td>
<td></td>
</tr>
<tr>
<td>15 CL21</td>
<td>CL17</td>
<td>6</td>
<td>47.4</td>
<td>6.8</td>
<td>0.3328</td>
<td></td>
</tr>
<tr>
<td>14 CL25</td>
<td>Elephant Seal</td>
<td>3</td>
<td>45.0</td>
<td>.</td>
<td>0.3362</td>
<td></td>
</tr>
<tr>
<td>13 CL19</td>
<td>CL16</td>
<td>5</td>
<td>40.8</td>
<td>3.5</td>
<td>0.3672</td>
<td></td>
</tr>
<tr>
<td>12 CL15</td>
<td>Grey Seal</td>
<td>7</td>
<td>38.9</td>
<td>2.8</td>
<td>0.4078</td>
<td></td>
</tr>
<tr>
<td>11 CL29</td>
<td>Raccoon</td>
<td>3</td>
<td>38.0</td>
<td>.</td>
<td>0.423</td>
<td></td>
</tr>
<tr>
<td>10 CL18</td>
<td>CL20</td>
<td>6</td>
<td>34.5</td>
<td>10.3</td>
<td>0.4339</td>
<td></td>
</tr>
<tr>
<td>9 CL12</td>
<td>CL26</td>
<td>9</td>
<td>30.0</td>
<td>7.3</td>
<td>0.5071</td>
<td></td>
</tr>
<tr>
<td>8 CL24</td>
<td>CL23</td>
<td>4</td>
<td>28.7</td>
<td>.</td>
<td>0.5473</td>
<td></td>
</tr>
<tr>
<td>7 CL9</td>
<td>CL14</td>
<td>12</td>
<td>25.7</td>
<td>7.0</td>
<td>0.5668</td>
<td></td>
</tr>
<tr>
<td>6 CL10</td>
<td>House Mouse</td>
<td>7</td>
<td>28.3</td>
<td>4.1</td>
<td>0.5792</td>
<td></td>
</tr>
<tr>
<td>5 CL11</td>
<td>CL7</td>
<td>15</td>
<td>26.8</td>
<td>6.9</td>
<td>0.6621</td>
<td></td>
</tr>
<tr>
<td>4 CL13</td>
<td>Mole</td>
<td>6</td>
<td>31.9</td>
<td>7.2</td>
<td>0.7156</td>
<td></td>
</tr>
<tr>
<td>3 CL4</td>
<td>CL8</td>
<td>10</td>
<td>31.0</td>
<td>12.7</td>
<td>0.8799</td>
<td></td>
</tr>
<tr>
<td>2 CL3</td>
<td>CL6</td>
<td>17</td>
<td>27.8</td>
<td>16.1</td>
<td>1.0316</td>
<td></td>
</tr>
<tr>
<td>1 CL2</td>
<td>CL5</td>
<td>32</td>
<td>.</td>
<td>27.8</td>
<td>1.1938</td>
<td></td>
</tr>
</tbody>
</table>
Output 118.1.2 shows the tree diagram produced by PROC CLUSTER.
Output 118.1.3 shows the corresponding tree diagram produced by PROC TREE.

Output 118.1.3  Tree Diagram of Mammal Teeth Clusters

As you view the diagram in Output 118.1.3 from left to right, objects and clusters are progressively joined until a single, all-encompassing cluster is formed at the right (or root) of the tree. Clusters exist at each level of the diagram, and every vertical line connects leaves and branches into progressively larger clusters. For example, the five bats form a cluster at the 0.6 level, while the next cluster consists only of the mole. The mammals ‘Reindeer’, ‘Elk’, ‘Deer’, and ‘Moose’ form the next cluster at the 0.6 level, the mammals ‘Pika’ through ‘House Mouse’ are in the fourth cluster, the mammals ‘Wolf’, ‘Bear’, and ‘Raccoon’ form the fifth cluster, and the last cluster contains the mammals ‘Marten’ through ‘Elephant Seal’.

The following statements create the same tree with line printer graphics in a vertical orientation:

```plaintext
options ps=40;
proc tree lineprinter;
run;
```

The line printer plot is not displayed.
The next statements sort the clusters at each branch in order of formation and use the number of clusters as the height axis:

```
proc tree sort height=n horizontal;
    label _name_ = 'Animal';
run;
```

The resulting tree is displayed in Output 118.1.4.

**Output 118.1.4** PROC TREE with SORT and HEIGHT= Options

![Mammals' Teeth](image)

Because the CLUSTER procedure always produces binary trees, the number of internal (root and branch) nodes in the tree is one less than the number of leaves. Therefore 31 clusters are formed from the 32 mammals in the input data set. These are represented by the 31 vertical line segments in the tree diagram, each at a different value along the horizontal axis.

As you examine the tree from left to right, the first vertical line segment is where ‘Beaver’ and ‘Groundhog’ are clustered and the number of clusters is 31. The next cluster is formed from ‘Gray Squirrel’ and ‘Porcupine’. The third contains ‘Wolf’ and ‘Bear’. Note how the tree graphically displays the clustering order information that was presented in tabular form by the CLUSTER procedure in Output 118.1.1.

The same clusters as in Output 118.1.3 can be seen at the six-cluster level of the tree diagram in Output 118.1.4, although the SORT and HEIGHT= options make them appear in a different order.
The following statements create these six clusters and save the result in the output data set part:

```sas
proc tree noprint out=part nclusters=6;
  id mammal;
  copy v1-v8;
run;

proc sort;
  by cluster;
run;
```

PROC TREE with the NOPRINT option displays no output but creates an output data set that indicates the cluster to which each observation belongs at the six-cluster level in the tree. The following statements print the data set part, with the results shown in Output 118.1.5:

```sas
proc print label uniform;
  id mammal;
  var v1-v8;
  format v1-v8 1.;
  by cluster;
run;
```

**Output 118.1.5**  PROC TREE OUT= Data Set

### Mammals’ Teeth

#### CLUSTER=1

<table>
<thead>
<tr>
<th>mammal</th>
<th>Right Top Incisors</th>
<th>Right Bottom Incisors</th>
<th>Right Top Canines</th>
<th>Right Bottom Canines</th>
<th>Right Top Premolars</th>
<th>Right Bottom Premolars</th>
<th>Right Top Molars</th>
<th>Right Bottom Molars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beaver</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Groundhog</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Gray Squirrel</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Porcupine</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Pika</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Rabbit</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>House Mouse</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

#### CLUSTER=2

<table>
<thead>
<tr>
<th>mammal</th>
<th>Right Top Incisors</th>
<th>Right Bottom Incisors</th>
<th>Right Top Canines</th>
<th>Right Bottom Canines</th>
<th>Right Top Premolars</th>
<th>Right Bottom Premolars</th>
<th>Right Top Molars</th>
<th>Right Bottom Molars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wolf</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Bear</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Raccoon</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>
### Output 118.1.5 continued

#### CLUSTER=3

<table>
<thead>
<tr>
<th>mammal</th>
<th>Right Top Incisors</th>
<th>Right Bottom Incisors</th>
<th>Right Top Canines</th>
<th>Right Bottom Canines</th>
<th>Right Top Premolars</th>
<th>Right Bottom Premolars</th>
<th>Right Top Molars</th>
<th>Right Bottom Molars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marten</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Wolverine</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Weasel</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Badger</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Jaguar</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Cougar</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Fur Seal</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Sea Lion</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>River Otter</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Sea Otter</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Elephant Seal</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Grey Seal</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

#### CLUSTER=4

<table>
<thead>
<tr>
<th>mammal</th>
<th>Right Top Incisors</th>
<th>Right Bottom Incisors</th>
<th>Right Top Canines</th>
<th>Right Bottom Canines</th>
<th>Right Top Premolars</th>
<th>Right Bottom Premolars</th>
<th>Right Top Molars</th>
<th>Right Bottom Molars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reindeer</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Elk</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Deer</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Moose</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

#### CLUSTER=5

<table>
<thead>
<tr>
<th>mammal</th>
<th>Right Top Incisors</th>
<th>Right Bottom Incisors</th>
<th>Right Top Canines</th>
<th>Right Bottom Canines</th>
<th>Right Top Premolars</th>
<th>Right Bottom Premolars</th>
<th>Right Top Molars</th>
<th>Right Bottom Molars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pigmy Bat</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Red Bat</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Brown Bat</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Silver Hair Bat</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>House Bat</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

#### CLUSTER=6

<table>
<thead>
<tr>
<th>mammal</th>
<th>Right Top Incisors</th>
<th>Right Bottom Incisors</th>
<th>Right Top Canines</th>
<th>Right Bottom Canines</th>
<th>Right Top Premolars</th>
<th>Right Bottom Premolars</th>
<th>Right Top Molars</th>
<th>Right Bottom Molars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mole</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
Example 118.2: Iris Data

Fisher (1936)’s iris data give sepal and petal dimensions for three different species of iris. The data, which are available in the Sashelp library, are clustered by kth-nearest-neighbor density linkage by using the CLUSTER procedure with K=8. Observations are identified by species (‘Setosa’, ‘Versicolor’, or ‘Virginica’) in the tree diagram, which is oriented with the height axis horizontal.

The following statements produce the results shown in Output 118.2.1:

```plaintext
title 'Fisher (1936) Iris Data';
ods graphics on;

proc cluster data=sashelp.iris method=twostage print=10
    outtree=tree k=8 noeigen;
    var SepalLength SepalWidth PetalLength PetalWidth;
    copy Species;
run;

proc tree data=tree horizontal lineprinter pages=1 maxh=10;
    id species;
run;
```

The PAGES=1 option specifies that the tree diagram extend over one page from tree to root. Since the HORIZONTAL option is also specified, the horizontal extent of the diagram is one page. The number of vertical pages required for the diagram is dictated by the number of leaves in the tree.

The MAXH=10 limits the values displayed on the height axis to a maximum of 10. This prunes the tree diagram so that only the portion from the leaves to level 10 is produced. The line printer plot is not displayed.

Output 118.2.1  Clustering of Fisher’s Iris Data

Fisher (1936) Iris Data

The CLUSTER Procedure
Two-Stage Density Linkage Clustering

K = 8

Root-Mean-Square Total-Sample Standard Deviation 10.69224
### Output 118.2.1 continued

<table>
<thead>
<tr>
<th>Number of Clusters</th>
<th>Clusters Joined</th>
<th>Freq</th>
<th>Normalized Fusion Density</th>
<th>Lesser</th>
<th>Greater</th>
<th>Tie</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>CL11 OB79</td>
<td>48</td>
<td>0.2879</td>
<td>0.1479</td>
<td>8.3678</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>CL13 OB112</td>
<td>46</td>
<td>0.2802</td>
<td>0.2005</td>
<td>3.5156</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>CL10 OB113</td>
<td>49</td>
<td>0.2699</td>
<td>0.1372</td>
<td>8.3678</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>CL8 OB91</td>
<td>50</td>
<td>0.2586</td>
<td>0.1372</td>
<td>8.3678</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>CL9 OB120</td>
<td>47</td>
<td>0.1412</td>
<td>0.0832</td>
<td>3.5156</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>CL6 OB118</td>
<td>48</td>
<td>0.1070</td>
<td>0.0605</td>
<td>3.5156</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>CL5 OB110</td>
<td>49</td>
<td>0.0969</td>
<td>0.0541</td>
<td>3.5156</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>CL4 OB135</td>
<td>50</td>
<td>0.0715</td>
<td>0.0370</td>
<td>3.5156</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CL7 CL3</td>
<td>100</td>
<td>2.6277</td>
<td>3.5156</td>
<td>8.3678</td>
<td></td>
</tr>
</tbody>
</table>

3 modal clusters have been formed.

---

### References


Subject Index

cluster analysis
tree diagrams, 9696
CLUSTER procedure, see also TREE procedure
dendrogram, 9696

missing values
TREE procedure, 9710

OUT= data sets
TREE procedure, 9710
output data sets
TREE procedure, 9710
output table names
TREE procedure, 9711

phenogram, 9696

tree diagram
binary tree, 9696
branch, 9696
children, 9696
definitions, 9696
leaves, 9696
node, 9696
parent, 9696
root, 9696
tree diagrams
cluster analysis, 9696
TREE procedure, 9696
missing values, 9710
OUT= data sets, 9710
output data sets, 9710
output table names, 9711

VARCLUS procedure, see also TREE procedure
Syntax Index

BY statement
TREE procedure, 9708

CFRAME= option
PROC TREE statement, 9703

COPY statement
TREE procedure, 9708

DATA= option
PROC TREE statement, 9703

DESCENDING option
PROC TREE statement, 9703

DESCRIPTION= option
PROC TREE statement, 9703

DISSIMILAR option
PROC TREE statement, 9704

DOCK= option
PROC TREE statement, 9704

FILLCHAR= option
PROC TREE statement, 9704

FREQ statement
TREE procedure, 9708

GOUT= option
PROC TREE statement, 9704

HAXIS= option
PROC TREE statement, 9704

HEIGHT statement
TREE procedure, 9709

HEIGHT= option
PROC TREE statement, 9704

HORIZONTAL option
PROC TREE statement, 9704

HPAGES= option
PROC TREE statement, 9705

ID statement
TREE procedure, 9709

INC= option
PROC TREE statement, 9705

JOINCHAR= option
PROC TREE statement, 9705

LEAFCHAR= option
PROC TREE statement, 9705

LEVEL= option
PROC TREE statement, 9705

LINEPRINTER option
PROC TREE statement, 9705

LINES= option
PROC TREE statement, 9705

LIST option
PROC TREE statement, 9705

MAXHEIGHT= option
PROC TREE statement, 9706

MINHEIGHT= option
PROC TREE statement, 9706

NAME statement
TREE procedure, 9709

NAME= option
PROC TREE statement, 9706

NCLUSTERS= option
PROC TREE statement, 9706

NOPRINT option
PROC TREE statement, 9706

NTICK= option
PROC TREE statement, 9706

OUT= option
PROC TREE statement, 9706

PAGES= option
PROC TREE statement, 9706

PARENT statement
TREE procedure, 9709

POS= option
PROC TREE statement, 9706

PROC TREE statement, see TREE procedure

ROOT= option
PROC TREE statement, 9707

SIMILAR option
PROC TREE statement, 9707

SORT option
PROC TREE statement, 9707

SPACES= option
PROC TREE statement, 9707

TICKPOS= option
PROC TREE statement, 9707

TREE procedure

syntax, 9702
TREE procedure, BY statement, 9708
TREE procedure, COPY statement, 9708
TREE procedure, FREQ statement, 9708
TREE procedure, HEIGHT statement, 9709
TREE procedure, ID statement, 9709
TREE procedure, NAME statement, 9709
TREE procedure, PARENT statement, 9709
TREE procedure, PROC TREE statement, 9702
  CFRAMES= option, 9703
  DATA= option, 9703
  DESCENDING option, 9703
  DESCRIPTION= option, 9703
  DISSIMILAR option, 9704
  DOCK= option, 9704
  FILLCHAR= option, 9704
  GOUT= option, 9704
  HAXIS= option, 9704
  HEIGHT= option, 9704
  HORDERPLAY= option, 9704
  HORIZONTAL option, 9705
  HPAGES= option, 9705
  INC= option, 9705
  JOINCHAR= option, 9705
  LEAFCHAR= option, 9705
  LEVEL= option, 9705
  LINEPRINTER option, 9705
  LINES= option, 9705
  LIST option, 9705
  MAXHEIGHT= option, 9706
  MINHEIGHT= option, 9706
  NAME= option, 9706
  NCLUSTERS= option, 9706
  NOPRINT option, 9706
  NTICKS= option, 9706
  OUT= option, 9706
  PAGES= option, 9706
  POS= option, 9706
  ROOT= option, 9707
  SIMILAR option, 9707
  SORT option, 9707
  SPACES= option, 9707
  TICKPOS= option, 9707
  TREECHAR= option, 9707
  VAXIS= option, 9707
  VPAGES= option, 9707
  TREECHAR= option
    PROC TREE statement, 9707
  VAXIS= option
    PROC TREE statement, 9707
  VPAGES= option
    PROC TREE statement, 9707