Plot an rpart model. A simplified interface to the prp function.

Description

Plot an rpart model, automatically tailoring the plot for the model's response type.

For an overview, please see the package vignette Plotting rpart trees with the rpart.plot package.

This function is a simplified front-end to prp, with only the most useful arguments of that function, and with different defaults for some of the arguments.

The different defaults for the extra and col.palette arguments mean that this function automatically creates a colored plot suitable for the type of model (whereas prp by default creates a minimal plot). In detail the different defaults are:

	rpart.plot	prp
type	2	0
extra	"auto"	0
fallen.leaves	TRUE	FALSE
varlen	0	-8
faclen	0	3
box.palette	"auto"	0

The function rpart.plot.version1 is compatible with old versions of this function and has the same defaults as prp.

Usage

```
rpart.plot(x=stop("no 'x' arg"),
    type=2, extra="auto",
    under=FALSE, fallen.leaves=TRUE,
    digits=2, varlen=0, faclen=0,
    cex=NULL, tweak=1,
    snip=FALSE,
    box.palette="auto", shadow.col=0,
    ...)
```

Arguments

To start off, look at the arguments x, type and extra. Just those arguments will suffice for many users. If you don't want a colored plot, use box.palette=0.

x	An rpart object. The only required argument.
type	Type of plot. Possible values: **0** Draw a split label at each split and a node label at each leaf. **1** Label all nodes, not just leaves. Similar to text.rpart's all=TRUE. **2** Default. Like 1 but draw the split labels below the node labels. Similar to the plots in the CART book. **3** Draw separate split labels for the left and right directions. **4** Like 3 but label all nodes, not just leaves. Similar to text.rpart's fancy=TRUE. See also clip.right.labs. **5 New in version 2.2.0.** Show the split variable name in the interior nodes.
extra	Display extra information at the nodes. Possible values: **"auto"** (case insensitive) Default. Automatically select a value based on the model type, as follows: extra=106 class model with a binary response extra=104 class model with a response having more than two levels extra=100 other models **0** No extra information. **1** Display the number of observations that fall in the node (per class for class objects; prefixed by the number of events for poisson and exp models). Similar to text.rpart's use.n=TRUE. **2** Class models: display the classification rate at the node, expressed as the number of correct classifications and the number of observations in the node. Poisson and exp models: display the number of events. **3** Class models: misclassification rate at the node, expressed as the number of incorrect classifications and the number of observations in the node. **4** Class models: probability per class of observations in the node (conditioned on the node, sum across a node is 1). **5** Class models: like 4 but don't display the fitted class. **6** Class models: the probability of the second class only. Useful for binary responses. **7** Class models: like 6 but don't display the fitted class. **8** Class models: the probability of the fitted class. **9** Class models: The probability relative to *all* observations – the sum of these probabilities across all leaves is 1. This is in contrast to the options above, which give the probability relative to observations falling *in the node* – the sum of the probabilities across the node is 1. **10 New in version 2.2.0.** Class models: Like 9 but display the probability of the second class only. Useful for binary responses. **11 New in version 2.2.0.** Class models: Like 10 but don't display the fitted class. **+100** Add 100 to any of the above to also display the percentage of observations in the node. For example extra=101 displays the number and percentage of observations in the node. Actually, it's a weighted percentage using the weights passed to rpart. Note: Unlike text.rpart, by default prp uses its own routine for generating node labels (not the function attached to the object). See the node.fun argument of prp.
under	Applies only if extra > 0. Default FALSE, meaning put the extra text *in* the box. Use TRUE to put the text *under* the box.
fallen.leaves	Default TRUE to position the leaf nodes at the bottom of the graph. It can be helpful to use FALSE if the graph is too crowded and the text size is too small.
digits	The number of significant digits in displayed numbers. Default 2. If 0, use getOption("digits"). If negative, use the standard format function (with the absolute value of digits). When digits is positive, the following details apply: Numbers from 0.001 to 9999 are printed without an exponent (and the number of digits is actually only a suggestion, see format for details). Numbers out that range are printed with an "engineering" exponent (a multiple of 3).
varlen	Length of variable names in text at the splits (and, for class responses, the class in the node label). Default 0, meaning display the full variable names. Possible values: **0** use full names (default). **greater than 0** call abbreviate with the given varlen. **less than 0** truncate variable names to the shortest length where they are still unique, but never truncate to shorter than abs(varlen).
faclen	Length of factor level names in splits. Default 0, meaning display the full factor names. Possible values are as varlen above, except that for back-compatibility with text.rpart the special value 1 means represent the factor levels with alphabetic characters (a for the first level, b for the second, etc.).
cex	Default NULL, meaning calculate the text size automatically.
tweak	Adjust the (possibly automatically calculated) cex. Default 1, meaning no adjustment. Use say tweak=1.2 to make the text 20% larger. However, since font sizes are discrete the cex you ask for may not be exactly the cex you get. And a small change to tweak may not actually change the type size, or change it more than you want.
snip	Default FALSE. Set TRUE to interactively trim the tree with the mouse. See the package vignette (or just try it).
box.palette	Palette for coloring the node boxes based on the fitted value. This is a vector of colors, for example **box.palette=c("green", "green2", "green4")**. Small fitted values are displayed with colors at the start of the vector; large values with colors at the end. Quantiles are used to partition the fitted values. The special value **box.palette=0** (default for prp) uses the background color (typically white). The special value **box.palette="auto"** (default for rpart.plot, case insensitive) automatically selects a predefined palette based on the type of model. Otherwise specify a predefined palette e.g. **box.palette="Grays"** for the predefined gray palette (a range of grays). The predefined palettes are (see the show.prp.palettes function): Grays Greys Greens Blues Browns Oranges Reds Purples Gy Gn Bu Bn Or Rd Pu (alternative names for the above palettes) BuGn GnRd BuOr etc. (two-color diverging palettes: any combination of two of the above palettes) RdYlGn GnYlRd BlGnYl YlGnBl (three color palettes) Prefix the palette name with "-" to reverse the order of the colors e.g. box.palette="-auto" or box.palette="-Grays".
shadow.col	Color of the shadow under the boxes. Default 0, no shadow. Try "gray" or "darkgray".
...	Extra arguments passed to prp and the plotting routines. Any of prp's arguments can be used.

Value

The returned value is identical to that of prp.

Author(s)

Stephen Milborrow, borrowing heavily from the rpart package by Terry M. Therneau and Beth Atkinson, and the R port of that package by Brian Ripley.

See Also

The package vignette Plotting rpart trees with the rpart.plot package

prp

rpart.plot.version1

Functions in the rpart package: plot.rpart text.rpart rpart

Examples

```
old.par <- par(mfrow=c(2,2))                    # put 4 figures on one page

data(ptitanic)

#--------------------------------------------------------------------------

binary.model <- rpart(survived ~ ., data=ptitanic, cp=.02)
                                     # cp=.02 for small demo tree

rpart.plot(binary.model,
           main="titanic survived\n(binary response)")

rpart.plot(binary.model,
           box.palette="Grays",         # override default GnBu palette
           main="titanic survived\nbox.palette = \"Grays\"")

#--------------------------------------------------------------------------

anova.model <- rpart(Mileage ~ ., data=cu.summary)

rpart.plot(anova.model,
           shadow.col="gray",           # add shadows just for kicks
           main="miles per gallon\n(continuous response)\n")

#--------------------------------------------------------------------------

multi.class.model <- rpart(Reliability ~ ., data=cu.summary)

rpart.plot(multi.class.model,
           main="vehicle reliability\n(multi class response)")

par(old.par)
```