Package 'rpart.plot'

February 20, 2015

Version 1.5.2

Title Plot rpart Models. An Enhanced Version of plot.rpart

Author Stephen Milborrow

Maintainer Stephen Milborrow <milbo@sonic.net>

Depends rpart (>= 4.1-0)

Description Plot rpart models. Extends plot.rpart and text.rpart in the rpart package.

License GPL-3

URL http://www.milbo.org/rpart-plot

NeedsCompilation no

Repository CRAN

Date/Publication 2015-02-04 15:56:16

R topics documented:

`prp` *Plot an* `rpart` *model. A superset of* `rpart.plot`.

Description

Plot an `rpart` model. The arguments of this function are a superset of those of `rpart.plot`.

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

Usage

```
prp(x=stop("no 'x' arg"),
    type=0, extra=0, under=FALSE, clip.right.labs=TRUE,
    nn=FALSE, ni=FALSE, yesno=TRUE,
    fallen.leaves=FALSE, branch=if(fallen.leaves) 1 else .2,
    uniform=TRUE, left=TRUE, xflip=FALSE, yflip=FALSE,
    Margin=0, space=1, gap=NULL,
    digits=2, varlen=-8, faclen=3,
    cex=NULL, tweak=1,
    compress=TRUE, ycompress=uniform,
    trace=FALSE, snip=FALSE, snip.fun=NULL,

    box.col=0, border.col=col,
    round=NULL, leaf.round=NULL,
    shadow.col=0, prefix="", suffix="", xsep=NULL,

    under.font=font, under.col=1, under.cex=.8,

    split.cex=1, split.font=2, split.family=family, split.col=1,
    split.box.col=0, split.border.col=0,
    split.lty=1, split.lwd=NULL, split.round=0,
    split.shadow.col=0,
    split.prefix="", right.split.prefix=NULL,
    split.suffix="", right.split.suffix=NULL,
    facsep=",", eq=" = ", lt=" < ", ge=" >= ",

    branch.col=if(identical(branch.type, 0)) 1 else "gray",
    branch.lty=1, branch.lwd=NULL,
    branch.type=0, branch.tweak=1,
    min.branch.width=.002, branch.fill=branch.col,

    nn.cex=NULL, nn.font=3, nn.family="", nn.col=1,
    nn.box.col=0, nn.border.col=nn.col,
    nn.lty=1, nn.lwd=NULL, nn.round=.3,

    node.fun=internal.node.labs,
    split.fun=internal.split.labs,
    FUN=text,

    nspace=branch, minbranch=.3, do.par=TRUE,
    add.labs=TRUE, clip.left.labs=FALSE, fam.main="",
    yshift=0, yspace=space, shadow.offset=.4,
    split.adj=NULL, split.yshift=0, split.space=space,
    split.yspace=yspace, split.shadow.offset=shadow.offset,
    nn.adj=.5, nn.yshift=0, nn.space=.8, nn.yspace=.5,
    ygap=gap/2, under.ygap=.5, yesno.yshift=0,
    xcompact=TRUE, ycompact=uniform, xcompact.ratio=.8, min.inter.height=4,
    max.auto.cex=1, min.auto.cex=.15, ycompress.cex=.7, accept.cex=1.1,
```

```
    shift.amounts=c(1.5, 2),
    Fallen.yspace=.1, boxes.include.gap=FALSE,
    ...)
```

Arguments

x	An rpart object. The only required argument.
type	Type of plot. Five possibilities: 0 The default. 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 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.
extra	Display extra information at the nodes. Possible values: 0 No extra information (the default). 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 do not display the fitted class. 6 Class models: the probability of the second class only. Useful for binary responses. 7 Class models: like 6 but do not display the fitted class. 8 Class models: the probability of the fitted class. 9 Class models: the probabilities times the fraction of observations in the node (the probability relative to all observations, sum across all leaves is 1). +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 1: Unlike text.rpart, by default prp uses its own routine for generating node labels (not the function attached to the object). See node.fun. Note 2: The extra argument has special meaning for mvpart objects. See the Appendix to this package's vignette.
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. See also under.cex.

`clip.right.labs`
: Default is `TRUE` meaning "clip" the right split labels, i.e. do not print `variable=`. Applies only if `type=3` or `4`. See also `clip.left.labs`.

`nn`
: Display the node numbers. Default `FALSE`. (In the current implementation some overplotting may occur with `nn=TRUE`.)

`ni`
: Display the node indices, i.e. the row numbers of the nodes in the object's `frame`. Default `FALSE`.

`yesno`
: Default `TRUE`, meaning write `yes` and `no` on the appropriate sides of the top split. Ignored if `type=3` or `4`. (Use `nn.col` and the other `nn` parameters to change the color etc. of the `yesno` text.)

`fallen.leaves`
: Default `FALSE`. If `TRUE`, display the leaves at the bottom of the graph.

`branch`
: Controls the shape of the branch lines. Specify a value between `0` (V shaped branches) and `1` (square shouldered branches).
Default is `if(fallen.leaves) 1 else .2`.

`uniform`
: If `TRUE` (the default), the vertical spacing of the nodes is uniform. If `FALSE`, the nodes are spaced proportionally to the fit (more precisely, to the difference between a node's deviance and the sum of its two children's deviances). Small vertical spaces are automatically artificially expanded to make room for the labels, see `minbranch`. Note: `uniform=FALSE` with `cex=NULL` (the default) can sometimes cause very small text.

`left`
: Default `TRUE`, meaning the left side of a split is the path taken if the split condition is true. With `left=FALSE` the split labels are changed so the right side is true.

`xflip`
: Default `FALSE`. If `TRUE`, flip the tree horizontally.

`yflip`
: Default `FALSE`. If `TRUE`, flip the tree vertically, so the root is at the bottom.

`Margin`
: Extra white space around the tree, as a fraction of the graph width. Default `0`, meaning no extra space. To add say 10% space around the tree use `Margin=0.1`. (This is the `margin` argument of `plot.rpart`. The name was changed to prevent partial matching with `mar`, which can be passed in as a ... argument.)

`gap`
: Minimum horizontal gap between the (possibly invisible) boxes, in character widths. Default `NULL`, meaning automatically choose a suitable value (normally `1`, but if the graph is very crowded will be set to `0`, permitting boxes to touch to allow a bigger `cex`). See also `space`.

`digits`
: The number of significant digits in displayed numbers. Default `2`. If `0`, use `getOption("digits")`.
Details: 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 `-8`, meaning truncate to eight characters. Possible values:
=0 use full names.
>0 call `abbreviate` with the given `varlen`.
<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 3, meaning abbreviate to three characters. Possible values are as varlen above, except that 1 is treated specially, meaning 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. Note that font sizes are discrete, so the cex you ask for may not be the cex you get. And a small tweak may not actually change the type size or change it more than you want.

compress — If TRUE (the default), make more space by shifting nodes horizontally where space is available. This often allows larger text. (This is the same as plot.rpart's argument of the same name, except that here the default is TRUE.)

ycompress — If TRUE (the default unless uniform=FALSE), make more space by shifting labels vertically where space is available. Actually, this only kicks in if the initial automatically calculated cex is less than 0.7. Use ycompress=FALSE if you feel the resulting display is too messy. In the current implementation, the shifting algorithm works a little better (allowing larger text) with type=1, 2, or 3.

trace — Default FALSE. Use TRUE to print the automatically calculated cex, xlim, and ylim. Use integer values greater than 1 for more detailed tracing.

snip — Default FALSE. Set TRUE to interactively trim the tree with the mouse. See the package vignette (or just try it).

snip.fun — Function invoked after each mouse click when snip=TRUE. Default NULL, meaning no function. Otherwise set snip.fun to your own function with the prototype function(tree), where tree is the snipped tree. See the package vignette for an example.

The following control the node labels.

box.col — Color of the boxes around the text. Default 0, meaning use the background color.

border.col — Color of the box border around the text. Default col, the color of the text in the box. Use 0 for no border. (Note: par settings like col can be passed in as ... arguments. If not passed in, par("col") is used.)

round — Controls the rounding of the corners of the node boxes. Default NULL, meaning calculate automatically. Else specify 0 for sharp edges, and values greater than 0 for rounded edges. Bigger is more round. Values too big for the box get silently reduced. (TODO clarify.)

leaf.round — Controls the rounding of the corners of the leaf node boxes. Default NULL, meaning use round. Else specify a value greater than or equal to 0.

shadow.col — Color of the shadow under the boxes. Default 0, no shadow. Try "gray" or "darkgray". (Note: overlapping shadows look better on devices that support alpha channels. If you get the message "Warning: semi-transparency is not supported" please let me know — it means that a fix is needed to the code that determines if the device supports alpha channels.)

prefix — Default "". Prepend this string to the node labels. So could be the name of the fitted response, for instance.

`suffix` Default `""`. Append this string to the node labels. Text after a double newline `"\n\n"` (if any) will be plotted *under* the box. (Actually, double newlines can be used in any of the prefix or suffix arguments for this purpose.)

`xsep` String which separates the individual counts and probabilities in node labels when `extra>0`. Default NULL meaning automatically select: usually `"  "` (two spaces), but `" / "` for rates. Use `xsep="/"` for compatibility with `text.rpart`. See also `facsep`, which separates the factor levels in split labels.

The following control the text under the boxes (apply only if `under=TRUE` or there is a double newline `\n\n` in `prefix` or `suffix`).

`under.font` Font of the text under the box. Default `font` (which can be passed in as a ... argument).

`under.col` Color of the text under the box. Default `1`.

`under.cex` Size of the text under the box relative to the text in the box. Default `.8`, smaller than the text in the box.

The following control the split labels.

`split.cex` Size of the split text relative to `cex` (which by default is calculated automatically). Default `1`.

`split.font` Font for the split labels. Default `2`, bold. (Note: use `font` to change the *node* label text.)

`split.family` Font family for the split labels.
Default `""`, or use something like `split.family="serif"`. (Note: use `family` to change the *node* label text.)

`split.col` Color of the split label text. Default `1`. (Note: use `col` to change the *node* label text.)

`split.box.col` Color of the split boxes. Default `0`, meaning use the background color.

`split.border.col`
Color of the split box borders. Default `0`, invisible.

`split.lty` Line type for the split box borders. The default is `1`, but the border will be invisible unless you change the default `split.border.col`. (Note: use `lty` to change the *node* box borders.)

`split.lwd` Line width of the split box border relative to `cex` (which by default is calculated automatically). The border is by default invisible, see codesplit.border.col.

`split.round` Controls the rounding of the corners of the split boxes. Default `0`, meaning sharp corners. Else specify a value greater than or equal to `0`. The split boxes are by default invisible, see `split.box.col` and `split.border.col`).

`split.shadow.col`
Color of the shadow under the split boxes. Default `0`, no shadow.

`split.prefix` Default `""`. Prepend this string to the split labels.

`right.split.prefix`
Default `split.prefix`. Prepend this string to the right split labels. Applies only when `type=3` or `4`.

`split.suffix` Default `""`. Append this string to the split labels.

`right.split.suffix`
Default `split.suffix`. Append this string to the right split labels. Applies only when `type=3` or `4`.

`facsep` Default `","`. String which separates the factor levels in split labels. See also `xsep`, which separates the individual counts when `extra` is used.

`eq` Default `" = "`. String which separates the factor name from the levels in split labels. The idea is that you can add or remove spaces around the `=`, or use words if that suits you.

`lt` Default `" < "`. String which represents "less than" in split labels.

`ge` Default `" >= "`. String which represents "greater than or equal" in split labels.

The following control the branches.

`branch.col` Color of the branch lines. Default `1`, but set to `"gray"` if `branch.type` is nonzero.

`branch.lty` Branch line type. Default `1`.

`branch.lwd` Line width of the branch lines relative to `cex` (which by default is calculated automatically). (Note: `branch.lwd` does not control the width of the "wide branches" drawn when `branch.type` is nonzero.)

`branch.type` Default `0`. If nonzero draw "wide branches", with branch widths proportional to the parameter selected by `branch.type` as follows:

`0` The default. The branch lines are drawn conventionally.
`1` deviance
`2` sqrt(deviance)
`3` deviance / nobs
`4` sqrt(deviance / nobs) (the standard deviation when `method="anova"`)
`5` weight (`frame$wt`). This is the number of observations at the node, unless `rpart`'s `weight` argument was used.
`6` complexity
`7` abs(predicted value)
`8` predicted value - min(predicted value)
`9` constant (for checking visual perception of the relative width of branches).

Otherwise set `branch.type` to your own function. The function should take a single argument `x` (the `rpart` object) and return a numeric vector of non-negative widths corresponding to rows in `frame`. See `get.branch.widths` in the source code.

Note: with a nonzero `branch.type`, in the current implementation the `branch` argument will be silently changed to `1` (if `branch > .5`) or `0` (if `branch < .5`)

`branch.tweak` Default `1`. Applies only if `branch.type` is nonzero. Use this argument to scale the widths of the branches, for example, `branch.tweak=.5` to halve the width of the branches. (By default, `prp` normalizes the widths so the widest branch is one-fifth the plot width.)

`min.branch.width`
: Default `0.002`. Applies only if `branch.type` is nonzero. The minimum width of a branch, as a fraction of the page width. The width of branches that would be thinner than `min.branch.width` is clamped. Increase `min.branch.width` if the thinnest branches are too skinny on your display device.

`branch.fill`
: Color used to fill the wide branch lines. Applies only if `branch.type` is nonzero. Default `branch.col`.

The following control the node numbers (with `nn=TRUE`).

`nn.cex`
: Default `NULL`, meaning calculate the `cex` of the node numbers automatically. This and the following arguments apply only when `nn=TRUE`.

`nn.font`
: Font for the node numbers. Default `3`, italic.

`nn.family`
: Font family for the node numbers. Default `""`.

`nn.col`
: Color of the node number text. Default `1`.

`nn.box.col`
: Color of the boxes around the node numbers. Default `0`, meaning use the background color.

`nn.border.col`
: Color of the box border around the node numbers. Default `nn.col`.

`nn.lty`
: Line type of the node number box border. Default `1`.

`nn.lwd`
: Line width of the node box border relative to `cex` (which by default is calculated automatically). Default `NULL`, meaning use `lwd` (which can be passed in as a `...` argument).

`nn.round`
: Controls the rounding of the corners of the node number boxes. Default `.3`, meaning small corners. Else specify a value greater than or equal to `0`.

The following are user definable functions.

`node.fun`
: The function that generates the text at the node labels. The default is `internal.node.labs`, which specifies a function internal to `prp`. This is necessary for full support of `extra` as described in the section on `extra` above.
Otherwise set `node.fun` to your own function with the prototype
`function(x, labs, digits, varlen)`
See the package vignette for details.

`split.fun`
: The function that generates the text at the splits. Default `internal.split.labs`, which specifies a function internal to `prp`. Otherwise set `split.fun` to your own function with the prototype `function(x, labs, digits, varlen, faclen)`

`FUN`
: The function that displays the text on the screen. Default `text`.

The following are esoteric parameters, mostly for the graph layout engine.

`nspace`
: Applies only when `compress=TRUE`. Default `nspace=branch`. The size of the space between a split and a leaf, relative to the space between leaves.

`minbranch`
: Applies only when `uniform=FALSE`. Default `.3`. The minimum height between levels is clamped at `minbranch` times the mean interlevel distance. Needed because sometimes a split gives little or no improvement in deviance, and an interlevel distance strictly proportional to the improvement would leave no room for the label.

`do.par` Default `TRUE`, meaning adjust the `mar` parameter so the tree fills the figure region. This also sets `xpd=NA`. These graphic parameters are restored to their original state before `prp` exits. If you explicitly set `mar` or `xpd`, `prp` will use your setting regardless of the setting of `do.par`.

`add.labs` Default `TRUE`, meaning display the labels. If `FALSE`, gives a bare bones display similar to `plot.rpart`.

`clip.left.labs` Like `clip.right.labs` but for the left labels. Default is `FALSE`. Note that `clip.left.labs` and `clip.right.labs` can be vectors, indexed on the split number.

`fam.main` Font family for the main text. Default `""`. The (inconsistent) name was chosen to minimize partial matching with `main` and `family` which can be passed in as in as ... arguments.

`yshift` Vertical position of the labels, in character heights relative to their default position. Default `0`. Negative values move the text down; positive up (the box around the text will follow along).

`space` Horizontal space to the box border on each side of the node label text, in character widths. Default `1`. Use this (and `yspace`) for bigger boxes. Since this affects the size of the (possibly invisible) boxes, it also affects the graph layout and hence also the automatic calculation of `cex`.

`yspace` Vertical space to the box border above and below the node label text, in character heights. Default `space`. See the comments for `space`.

`shadow.offset` Offset of the shadow from the boxes, in character widths. Default `.4` (but the shadow will be invisible unless the default `shadow.col` is changed).

`split.adj` Horizontal position of the split text. In string width units, as is the convention for `adj` arguments. Default `NULL`, meaning use `adj` (which defaults to `0.5` but can be passed in as a ... argument). Use values less/more than `.5` to shift the text left/right (the box around the text will follow along).

`split.yshift` Vertical position of the split labels, in character heights relative to their default positions. Default `0`. Negative values move the text down; positive up (the box around the text will follow along). This adjusts the positions of the split labels relative to the node labels. (Use `yshift` if you want to shift *both* the split and node labels.)

`split.space` Horizontal space between the split label text and the box, in character widths. Default `space`. Affects the size of the box drawn around the text. The split boxes are by default invisible (see `split.box.col` and `split.border.col`), but nevertheless affect the graph layout used in the automatic calculation of `cex`.

`split.yspace` Vertical space between the split label text and the box, in character heights. Default `yspace`.

`split.shadow.offset`
Offset of the shadow from the split boxes, in character widths. Default `shadow.offset`. (but the shadow will be invisible unless the default `shadow.col` is changed).

`nn.adj` Horizontal position of the node label text. Default `.5`.

`nn.yshift` Vertical position of the node numbers, in character heights relative to their default positions. Default `0`.

nn.space — Horizontal space to the box border on each side of the node number text, in character widths. Default .8.

nn.yspace — Vertical space to the box border above and below the node number text, in character heights. Default .5.

under.ygap — Applies if text is plotted under the box (i.e. if under=TRUE or there is a double newline in prefix or suffix). Vertical gap (in char heights) between the lower edge of the box and the top of the text under the box.

yesno.yshift — Vertical position of "yes" and "no" in character heights relative to their default position. Default 0. Applies only when yesno=TRUE.

ygap — Minimum vertical gap between boxes, in character heights. Default gap/2.

xcompact — If TRUE (the default) and there is too much white space, automatically change xlim to compact the entire tree horizontally. This usually only activates for small trees. (The xcompact and ycompact arguments compact the tree as a whole, whereas the compress and ycompress arguments move parts of the tree into available space.)

ycompact — If TRUE (the default) and there is too much vertical space, automatically change ylim to compact the entire tree vertically.

xcompact.ratio — Default .8. Applies only when xcompact=TRUE. The maximum possible without overplotting is 1, but compacting by .8 usually gives more pleasing spacing (it gives more space).

min.inter.height — Default 4. Applies only when ycompact=TRUE. Minimum height (in units of character height) between the lowest label in a layer and the highest label in the layer below it.

max.auto.cex — Clamp the maximum automatically calculated cex at this value Default 1, meaning never expand cex, only contract when necessary.

min.auto.cex — Default .15. Never downscale to less than this when automatically calculating cex, even if overplotted labels result. (The graph layout algorithm is unstable with cex's below 0.15 — meaning that the automatic type size may be smaller than necessary.)

ycompress.cex — Default .7. Applies only when ycompress=TRUE. Apply the ycompress algorithm if the initial automatically calculated cex is less than this. The idea is that we don't want to shift if we get an acceptable cex without shifting. Make Inf to always attempt shifting.

accept.cex — Accept shifting only if it causes at least this much improvement in cex (because we don't want to shift if it gives only a small improvement in cex). Default 1.1 i.e. require at least a 10% improvement. Use 0 to always accept shifts and Inf to never accept (or use ycompress=FALSE).

shift.amounts — Default c(1.5, 2, 3). For ycompress, choose the best cex yielded by shifting nodes by these amounts, in multiples of the box heights (after initial scaling).

Fallen.yspace — Extra space for fallen leaves. Default .1, meaning allow 10% of the vertical space for the fallen leaves. (The name Fallen.yspace uses upper case to avoid partial matching with fallen.leaves.)

`boxes.include.gap`
: Default `FALSE`. Draw the boxes to include `gap` and `ygap`, for debugging. This only affects the way the boxes are drawn, not the graph layout algorithm in any way. With the optimum `cex` at least one pair of boxes displayed in this manner will just touch (but none will overlap).

`...`
: Extra `par` arguments. Only the "important" `par` arguments are supported. Note that arguments like `col` apply only to the *node* labels. To affect the split labels or branch lines, use `split.col` and `branch.col` instead.

Value

A list with the following components. With the default args most of these are automatically calculated.

`obj`
: The `rpart` object. Identical to the `x` argument passed in unless `snip` was used.

`snipped.nodes`
: The snipped nodes, `NULL` unless `snip` was used.

`xlim, ylim`
: The graph limits.

`x, y`
: The node coords.

`branch.x, branch.y`
: The branch line coords.

`labs`
: The node labels.

`cex`
: The node label `cex`.

`boxes`
: The coords of the boxes around the nodes.

`split.labs`
: The split labels.

`split.cex`
: The split label `cex`.

`split.boxes`
: The coords of the boxes around the splits.

See Also

The package vignette "Plotting rpart trees with prp"
`rpart.plot`
`plot.rpart`
`text.rpart`
`rpart`

Examples

```
data(ptitanic)
tree <- rpart(survived ~ ., data=ptitanic, cp=.02)
                        # cp=.02 because want small tree for demo

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

prp(tree, main="default prp\n(type = 0, extra = 0)")

prp(tree, main="type = 4,  extra = 6", type=4, extra=6, faclen=0)
```

```
                        # faclen=0 to print full factor names

cols <- ifelse(tree$frame$yval == 1, "darkred", "green4")
                        # green if survived

prp(tree, main="assorted arguments",
    extra=106,          # display prob of survival and percent of obs
    nn=TRUE,            # display the node numbers
    fallen.leaves=TRUE, # put the leaves on the bottom of the page
    branch=.5,          # change angle of branch lines
    faclen=0,           # do not abbreviate factor levels
    trace=1,            # print the automatically calculated cex
    shadow.col="gray",  # shadows under the leaves
    branch.lty=3,       # draw branches using dotted lines
    split.cex=1.2,      # make the split text larger than the node text
    split.prefix="is ", # put "is " before split text
    split.suffix="?",   # put "?" after split text
    col=cols, border.col=cols,   # green if survived
    split.box.col="lightgray",   # lightgray split boxes (default is white)
    split.border.col="darkgray", # darkgray border on split boxes
    split.round=.5)             # round the split box corners a tad

# the old way for comparison
plot(tree, uniform=TRUE, compress=TRUE, branch=.2)
text(tree, use.n=TRUE, cex=.6, xpd=NA) # cex is a guess, depends on your window size
title("plot.rpart for comparison", cex=.6)

par(old.par)
```

`ptitanic` *Titanic data with passenger names and other details removed.*

Description

Titanic data with passenger names and other details removed.

Format

A data frame with 1046 observations on 6 variables.

`pclass`	passenger class, unordered factor: 1st 2nd 3rd
`survived`	factor: died or survived
`sex`	unordered factor: male female
`age`	age in years, min 0.167 max 80.0
`sibsp`	number of siblings or spouses aboard, integer: 0...8
`parch`	number of parents or children aboard, integer: 0...6

Source

The dataset was compiled by Frank Harrell and Robert Dawson:
http://biostat.mc.vanderbilt.edu/twiki/pub/Main/DataSets/titanic.html.

See also:
http://biostat.mc.vanderbilt.edu/twiki/pub/Main/DataSets/titanic3info.txt.

For this version of the Titanic data, passenger details were deleted, survived was cast as a factor, and the name changed to ptitanic to minimize confusion with other versions.

In this data the crew are conspicuous by their absence.

Contents of ptitanic:

```
      pclass survived    sex    age sibsp parch
1        1st survived female 29.000     0     0
2        1st survived   male  0.917     1     2
3        1st     died female  2.000     1     2
4        1st     died   male 30.000     1     2
5        1st     died female 25.000     1     2
...
1309     3rd     died   male 29.000     0     0
```

How ptitanic was built:

```
load("titanic3.sav") # from Dr. Harrell's web site
# discard name, ticket, fare, cabin, embarked, body, home.dest
ptitanic <- titanic3[,c(1,2,4,5,6,7)]
# change survived from integer to factor
ptitanic$survived <- factor(ptitanic$survived, labels=c("died", "survived"))
save(ptitanic, file="ptitanic.rda")
```

This version of the data differs from etitanic in the earth package in that here survived is a factor (not an integer) and age has some NAs.

Examples

```
data(ptitanic)
summary(ptitanic)
# main indicator of missing data is 3rd class esp. with many children
obs.with.nas <- rowSums(is.na(ptitanic)) > 0
prp(rpart(obs.with.nas~., data=ptitanic, method="class"),
    main="observations with missing data", extra=7)
```

rpart.plot *Plot an* rpart *model.*

Description

Plot an rpart model. This function combines and extends plot.rpart and text.rpart in the rpart package. It automatically scales and adjusts the displayed tree for best fit.

This is a front end to prp, with the most useful arguments of that function.

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

Usage

```
rpart.plot(x=stop("no 'x' arg"),
    type=0, extra=0, under=FALSE, clip.right.labs=TRUE,
    fallen.leaves=FALSE, branch=if(fallen.leaves) 1 else .2,
    uniform=TRUE,
    digits=2, varlen=-8, faclen=3,
    cex=NULL, tweak=1,
    compress=TRUE, ycompress=uniform,
    snip=FALSE,
    ...)
```

Arguments

To start off, look at the arguments x, type and extra. Just those arguments will suffice for many users. For an overview of the other arguments see the package vignette.

An rpart object. The only required argument.

x

type Type of plot. Five possibilities:

0 The default. 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 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.

extra Display extra information at the nodes. Possible values:

0 No extra information (the default).

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 do not display the fitted class.

6 Class models: the probability of the second class only. Useful for binary responses.

7 Class models: like 6 but do not display the fitted class.

8 Class models: the probability of the fitted class.

9 Class models: the probabilities times the fraction of observations in the node (the probability relative to all observations, sum across all leaves is 1).

+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 1: Unlike text.rpart, by default prp uses its own routine for generating node labels (not the function attached to the object). See node.fun.
Note 2: The extra argument has special meaning for mvpart objects. See the Appendix to this package's vignette.

`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.

`clip.right.labs`
Default is TRUE meaning "clip" the right-hand split labels, i.e. do not print variable=. Applies only if type=3 or 4.

`fallen.leaves` Default FALSE. If TRUE, display the leaves at the bottom of the graph.

`branch` Controls the shape of the branch lines. Specify a value between 0 (V shaped branches) and 1 (square shouldered branches).
Default is `if(fallen.leaves) 1 else .2`.

`uniform` If TRUE (the default), the vertical spacing of the nodes is uniform. If FALSE, the nodes are spaced proportionally to the fit (more precisely, to the difference between a node's deviance and the sum of its two children's deviances). Small vertical spaces are automatically artificially expanded to make room for the labels. Note: uniform=FALSE with cex=NULL (the default) can sometimes cause very small text.

`digits` The number of significant digits in displayed numbers. Default 2. If 0, use `getOption("digits")`.
Details: 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 -8, meaning truncate to eight characters. Possible values:
=0 use full names.
>0 call abbreviate with the given varlen.

	<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 `3`, meaning `abbreviate` to three characters. Possible values are as `varlen` above, except that `1` is treated specially, meaning 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. Note that font sizes are discrete, so the `cex` you ask for may not be the `cex` you get. And a small tweak may not actually change the type size or change it more than you want.
`compress`	If `TRUE` (the default), make more space by shifting nodes horizontally where space is available. This often allows larger text. (This is the same as `plot.rpart`'s argument of the same name, except that here the default is `TRUE`.)
`ycompress`	If `TRUE` (the default unless `uniform=FALSE`), make more space by shifting labels vertically where space is available. Actually, this only kicks in if the initial automatically calculated `cex` is less than `0.7`. Use `ycompress=FALSE` if you feel the resulting display is too messy. In the current implementation, the shifting algorithm works a little better (allowing larger text) with `type=1`, `2`, or `3`.
`snip`	Default `FALSE`. Set `TRUE` to interactively trim the tree with the mouse. See the package vignette (or just try it).
`...`	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 prp"
`prp`
`plot.rpart`
`text.rpart`
`rpart`

Examples

```
data(ptitanic)
tree <- rpart(survived ~ ., data=ptitanic, cp=.02)
                         # cp=.02 because want small tree for demo

old.par <- par(mfrow=c(2,2))
```

```
                        # put 4 figures on one page

rpart.plot(tree, main="default rpart.plot\n(type = 0, extra = 0)")

prp(tree, main="type = 4, extra = 6", type=4, extra=6, faclen=0)
                        # faclen=0 to print full factor names

rpart.plot(tree, main="extra = 106,  under = TRUE", extra=106, under=TRUE, faclen=0)

# the old way for comparison
plot(tree, uniform=TRUE, compress=TRUE, branch=.2)
text(tree, use.n=TRUE, cex=.6, xpd=NA) # cex is a guess, depends on your window size
title("rpart.plot for comparison", cex=.6)

par(old.par)
```

Index