**rpart**

rpart(formula, data, method, parms, na.action = na.rpart, control, ...)

* formula: the model formula, as in lm and other R model \_tting functions. The right

hand side may contain both continuous and categorical (factor) terms. If the outcome y has more than two levels, then categorical predictors must be \_t by exhaustive enumeration, which can take a very long time.

* data, weights, subset: as for other R models.
* method: the type of splitting rule to use. Options at this point are **classication, anova, Poisson, and exponential.**
* parms: a list of method speci\_c optional parameters. For classi\_cation, the list can contain any of: the vector of prior probabilities (component prior), the loss matrix (component loss) or the splitting index (component split). The priors must be positive and sum to 1. The loss matrix must have zeros on the diagonal and positive o\_-diagonal elements. **The splitting index can be "gini" or "information"**. Va con una LISTA: ejemplo parms = list(prior = c(.65,.35), split = "information"))
* na.action: the action for missing values. The default action for rpart is na.rpart, this default is not overridden by the options(na.action) global option. The default action removes only those rows for which either the response y or all of the predictors are missing. **This ability to retain partially missing observations is perhaps the single most useful feature of rpart models.**
* control: a list of control parameters, usually the result of the rpart.control function.(EJEMPLO: rpart.control(cp = 0.05))

The list must contain

* minsplit: The minimum number of observations in a node for which the routine will even try to compute a split. The default is 20. **This parameter can save computation time, since smaller nodes are almost always pruned away by crossvalidation.**
* minbucket: The minimum number of observations in a terminal node. **This defaults to minsplit/3.**

**rpart.plot**

rpart.plot(x = stop("no 'x' arg"),

type = 2, extra = "auto",

under = FALSE, fallen.leaves = TRUE,

digits = 2, varlen = 0, faclen = 0, roundint = TRUE,

cex = NULL, tweak = 1,

clip.facs = FALSE, clip.right.labs = TRUE,

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](http://127.0.0.1:37250/help/library/rpart.plot/help/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'sfancy=TRUE. See alsoclip.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 classobjects; 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 exampleextra=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 defaultprp uses its own routine for generating node labels (not the function attached to the object). See the node.funargument of prp. |