Description

The car.test.frame data frame has 60 rows and 8 columns, giving data on makes of cars taken from the April, 1990 issue of Consumer Reports. This is part of a larger dataset, some columns of which are given in cu.summary.

Usage

car.test.frame

Format

This data frame contains the following columns:

Price  a numeric vector giving the list price in US dollars of a standard model
Reliability a numeric vector coded 1 to 5.
Mileage  fuel consumption miles per US gallon, as tested.
Type    a factor with levels Compact Large Medium Small Sporty Van
Weight  kerb weight in pounds.
**Disp.** the engine capacity (displacement) in litres.

**HP** the net horsepower of the vehicle.

### Source

*Consumer Reports*, April, 1990, pp. 235–288 quoted in

### See Also

car90, cu.summary

### Examples

```r
z.auto <- rpart(Mileage ~ Weight, car.test.frame)
summary(z.auto)
```

### Description


### Usage

data(car90)

### Format

The data frame contains the following columns:

- **Country** a factor giving the country in which the car was manufactured
- **Disp** engine displacement in cubic inches
- **Disp2** engine displacement in liters
- **Eng.Rev** engine revolutions per mile, or engine speed at 60 mph
- **Front.Hd** distance between the car’s head-liner and the head of a 5 ft. 9 in. front seat passenger, in inches, as measured by CU
- **Frt.Leg.Room** maximum front leg room, in inches, as measured by CU
- **Frt.Shold** front shoulder room, in inches, as measured by CU
- **Gear.Ratio** the overall gear ratio, high gear, for manual transmission
- **Gear2** the overall gear ratio, high gear, for automatic transmission
- **HP** net horsepower
HP.revs  the red line—the maximum safe engine speed in rpm
Height  height of car, in inches, as supplied by manufacturer
Length  overall length, in inches, as supplied by manufacturer
Luggage  luggage space
Mileage  a numeric vector of gas mileage in miles/gallon as tested by CU; contains NAs.
Model2  alternate name, if the car was sold under two labels
Price  list price with standard equipment, in dollars
Rear.Hd  distance between the car’s head-liner and the head of a 5 ft 9 in. rear seat passenger, in inches, as measured by CU
Rear.Seating  rear fore-and-aft seating room, in inches, as measured by CU
RearShld  rear shoulder room, in inches, as measured by CU
Reliability  an ordered factor with levels ‘Much worse’ < ‘worse’ < ‘average’ < ‘better’ < ‘Much better’; contains NAs.
Rim  factor giving the rim size
Sratio.m  Number of turns of the steering wheel required for a turn of 30 foot radius, manual steering
Sratio.p  Number of turns of the steering wheel required for a turn of 30 foot radius, power steering
Steering  steering type offered: manual, power, or both
Tank  fuel refill capacity in gallons
Tires  factor giving tire size
Trans1  manual transmission, a factor with levels ‘’, ‘man.4’, ‘man.5’ and ‘man.6’
Trans2  automatic transmission, a factor with levels ‘’, ‘auto.3’, ‘auto.4’, and ‘auto.CVT’. No car is missing both the manual and automatic transmission variables, but several had both as options
Turning  the radius of the turning circle in feet
Type  a factor giving the general type of car. The levels are: ‘Small’, ‘Sporty’, ‘Compact’, ‘Medium’, ‘Large’, ‘Van’
Weight  an order statistic giving the relative weights of the cars; 1 is the lightest and 111 is the heaviest
Wheel.base  length of wheelbase, in inches, as supplied by manufacturer
Width  width of car, in inches, as supplied by manufacturer

Source
This is derived (with permission) from the data set car.all in S-PLUS, but with some further clean up of variable names and definitions.

See Also

car.test.frame, cu.summary for extracts from other versions of the dataset.
Examples

data(car90)
plot(car90$Price/1000, car90$Weight, 
    xlab = "Price (thousands)", ylab = "Weight (lbs)"
mlowess <- function(x, y, ...) {
    keep <- !(is.na(x) | is.na(y))
    lowess(x[keep], y[keep], ...)
}
with(car90, lines(mlowess(Price/1000, Weight, f = 0.5)))

Description

The cu.summary data frame has 117 rows and 5 columns, giving data on makes of cars taken from
the April, 1990 issue of Consumer Reports.

Usage

cu.summary

Format

This data frame contains the following columns:

Price  a numeric vector giving the list price in US dollars of a standard model
    ‘Korea’, ‘Mexico’, ‘Sweden’ and ‘USA’
Reliability an ordered factor with levels ‘Much worse’ < ‘worse’ < ‘average’ < ‘better’ <
    ‘Much better’
Mileage fuel consumption miles per US gallon, as tested.
Type  a factor with levels Compact Large Medium Small Sporty Van

Source

Consumer Reports, April, 1990, pp. 235–288 quoted in

See Also

car.test.frame, car90
**Examples**

```r
fit <- rpart(Price ~ Mileage + Type + Country, cu.summary)
par(xpd = TRUE)
plot(fit, compress = TRUE)
text(fit, use.n = TRUE)
```

<table>
<thead>
<tr>
<th>kyphosis</th>
<th>Data on Children who have had Corrective Spinal Surgery</th>
</tr>
</thead>
</table>

**Description**

The `kyphosis` data frame has 81 rows and 4 columns, representing data on children who have had corrective spinal surgery.

**Usage**

```r
kyphosis
```

**Format**

This data frame contains the following columns:

- **Kyphosis**: a factor with levels `absent` `present` indicating if kyphosis (a type of deformation) was present after the operation.
- **Age**: in months
- **Number**: the number of vertebrae involved
- **Start**: the number of the first (topmost) vertebra operated on.

**Source**


**Examples**

```r
fit <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis)
fit2 <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis,
              parms = list(prior = c(0.65, 0.35), split = "information"))
fit3 <- rpart(Kyphosis ~ Age + Number + Start, data=kyphosis,
              control = rpart.control(cp = 0.05))
par(mfrow = c(1,2), xpd = TRUE)
plot(fit)
text(fit, use.n = TRUE)
plot(fit2)
text(fit2, use.n = TRUE)
```
Create Split Labels for an \texttt{rpart} Object

\textbf{Description}

This function provides labels for the branches of an \texttt{rpart} tree.

\textbf{Usage}

\begin{verbatim}
## S3 method for class 'rpart'
labels(object, digits = 4, minlength = 1L, pretty, collapse = TRUE, ...)
\end{verbatim}

\textbf{Arguments}

- \texttt{object}: fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the \texttt{rpart} function.
- \texttt{digits}: the number of digits to be used for numeric values. All of the \texttt{rpart} functions that call labels explicitly set this value, with options("digits") as the default.
- \texttt{minlength}: the minimum length for abbreviation of character or factor variables. If 0 no abbreviation is done; if 1 single English letters are used, first lower case than upper case (with a maximum of 52 levels). If the value is greater than 1, the \texttt{abbreviate} function is used, passed the \texttt{minlength} argument.
- \texttt{pretty}: an argument included for compatibility with the \texttt{tree} package: pretty = 0 implies minlength = 0L, pretty = NULL implies minlength = 1L, and pretty = TRUE implies minlength = 4L.
- \texttt{collapse}: logical. The returned set of labels is always of the same length as the number of nodes in the tree.
  - If collapse = TRUE (default), the returned value is a vector of labels for the branch leading into each node, with "root" as the label for the top node.
  - If FALSE, the returned value is a two column matrix of labels for the left and right branches leading out from each node, with "leaf" as the branch labels for terminal nodes.
  - ... optional arguments to \texttt{abbreviate}.

\textbf{Value}

Vector of split labels (collapse = TRUE) or matrix of left and right splits (collapse = FALSE) for the supplied \texttt{rpart} object. This function is called by printing methods for \texttt{rpart} and is not intended to be called directly by the users.

\textbf{See Also}

\begin{verbatim}
abbreviate
\end{verbatim}
meanvar.rpart

Mean-Variance Plot for an Rpart Object

Description

Creates a plot on the current graphics device of the deviance of the node divided by the number of observations at the node. Also returns the node number.

Usage

meanvar(tree, ...)

## S3 method for class 'rpart'
meanvar(tree, xlab = "ave(y)", ylab = "ave(deviance)", ...)

Arguments

tree fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the rpart function.

xlab x-axis label for the plot.

ylab y-axis label for the plot.

... additional graphical parameters may be supplied as arguments to this function.

Value

an invisible list containing the following vectors is returned.

x fitted value at terminal nodes (yval).

y deviance of node divided by number of observations at node.

label node number.

Side Effects

a plot is put on the current graphics device.

See Also

plot.rpart.

Examples

z.auto <- rpart(Mileage ~ Weight, car.test.frame)
meanvar(z.auto, log = 'xy')
**na.rpart**

*Handles Missing Values in an Rpart Object*

**Description**

Handles missing values in an "rpart" object.

**Usage**

```r
na.rpart(x)
```

**Arguments**

- `x` a model frame.

**Details**

Default function that handles missing values when calling the function `rpart`. It omits cases where part of the response is missing or all the explanatory variables are missing.

---

**path.rpart**

*Follow Paths to Selected Nodes of an Rpart Object*

**Description**

Returns a names list where each element contains the splits on the path from the root to the selected nodes.

**Usage**

```r
path.rpart(tree, nodes, pretty = 0, print.it = TRUE)
```

**Arguments**

- `tree` fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the `rpart` function.
- `nodes` an integer vector containing indices (node numbers) of all nodes for which paths are desired. If missing, user selects nodes as described below.
- `pretty` an integer denoting the extent to which factor levels in split labels will be abbreviated. A value of (0) signifies no abbreviation. A NULL, the default, signifies using elements of letters to represent the different factor levels.
- `print.it` Logical. Denotes whether paths will be printed out as nodes are interactively selected. Irrelevant if `nodes` argument is supplied.
Details

The function has a required argument as an `rpart` object and a list of nodes as optional arguments. Omitting a list of nodes will cause the function to wait for the user to select nodes from the dendrogram. It will return a list, with one component for each node specified or selected. The component contains the sequence of splits leading to that node. In the graphical interaction, the individual paths are printed out as nodes are selected.

Value

A named (by node) list, each element of which contains all the splits on the path from the root to the specified or selected nodes.

Graphical Interaction

A dendrogram of the `rpart` object is expected to be visible on the graphics device, and a graphics input device (e.g. a mouse) is required. Clicking (the selection button) on a node selects that node. This process may be repeated any number of times. Clicking the exit button will stop the selection process and return the list of paths.

References

This function was modified from `path.tree` in S.

See Also

`rpart`

Examples

```r
fit <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis)
print(fit)
path.rpart(fit, node = c(11, 22))
```
Arguments

x a fitted object of class "rpart", containing a classification, regression, or rate tree.

uniform if TRUE, uniform vertical spacing of the nodes is used; this may be less cluttered when fitting a large plot onto a page. The default is to use a non-uniform spacing proportional to the error in the fit.

branch controls the shape of the branches from parent to child node. Any number from 0 to 1 is allowed. A value of 1 gives square shouldered branches, a value of 0 give V shaped branches, with other values being intermediate.

compress if FALSE, the leaf nodes will be at the horizontal plot coordinates of 1:nleaves. If TRUE, the routine attempts a more compact arrangement of the tree. The compaction algorithm assumes uniform=TRUE; surprisingly, the result is usually an improvement even when that is not the case.

nspace the amount of extra space between a node with children and a leaf, as compared to the minimal space between leaves. Applies to compressed trees only. The default is the value of branch.

margin an extra fraction of white space to leave around the borders of the tree. (Long labels sometimes get cut off by the default computation).

minbranch set the minimum length for a branch to minbranch times the average branch length. This parameter is ignored if uniform=TRUE. Sometimes a split will give very little improvement, or even (in the classification case) no improvement at all. A tree with branch lengths strictly proportional to improvement leaves no room to squeeze in node labels.

... arguments to be passed to or from other methods.

Details

This function is a method for the generic function plot, for objects of class rpart. The y-coordinate of the top node of the tree will always be 1.

Value

The coordinates of the nodes are returned as a list, with components x and y.

Side Effects

An unlabeled plot is produced on the current graphics device: one being opened if needed.

In order to build up a plot in the usual S style, e.g., a separate text command for adding labels, some extra information about the plot needs be retained. This is kept in an environment in the package.

See Also

rpart, text.rpart
Examples

```r
fit <- rpart(Price ~ Mileage + Type + Country, cu.summary)
par(xpd = TRUE)
plot(fit, compress = TRUE)
text(fit, use.n = TRUE)
```

Description

Gives a visual representation of the cross-validation results in an `rpart` object.

Usage

```r
plotcp(x, minline = TRUE, lty = 3, col = 1,
       upper = c("size", "splits", "none"), ...)
```

Arguments

- `x` an object of class "rpart"
- `minline` whether a horizontal line is drawn 1SE above the minimum of the curve.
- `lty` line type for this line
- `col` colour for this line
- `upper` what is plotted on the top axis: the size of the tree (the number of leaves), the number of splits or nothing.
- `...` additional plotting parameters

Details

The set of possible cost-complexity prunings of a tree from a nested set. For the geometric means of the intervals of values of `cp` for which a pruning is optimal, a cross-validation has (usually) been done in the initial construction by `rpart`. The `cptable` in the fit contains the mean and standard deviation of the errors in the cross-validated prediction against each of the geometric means, and these are plotted by this function. A good choice of `cp` for pruning is often the leftmost value for which the mean lies below the horizontal line.

Value

None.

Side Effects

A plot is produced on the current graphical device.

See Also

`rpart`, `printcp`, `rpart.object`
PostScript Presentation Plot of an Rpart Object

Description

Generates a PostScript presentation plot of an \texttt{rpart} object.

Usage

\begin{verbatim}
post(tree, ...) 

## S3 method for class 'rpart'
post(tree, title = , 
    filename = paste(deparse(substitute(tree)), ".ps", sep = ""), 
    digits = getOption("digits") - 2, pretty = TRUE, 
    use.n = TRUE, horizontal = TRUE, ...) 
\end{verbatim}

Arguments

- \texttt{tree}: fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the \texttt{rpart} function.
- \texttt{title}: a title which appears at the top of the plot. By default, the name of the \texttt{rpart} endpoint is printed out.
- \texttt{filename}: ASCII file to contain the output. By default, the name of the file is the name of the object given by \texttt{rpart} (with the suffix \texttt{.ps} added). If \texttt{filename = ""}, the plot appears on the current graphical device.
- \texttt{digits}: number of significant digits to include in numerical data.
- \texttt{pretty}: an integer denoting the extent to which factor levels will be abbreviated in the character strings defining the splits; (0) signifies no abbreviation of levels. A \texttt{NULL} signifies using elements of letters to represent the different factor levels. The default (\texttt{TRUE}) indicates the maximum possible abbreviation.
- \texttt{use.n}: Logical. If \texttt{TRUE} (default), adds to label (\texttt{\#events level1/ \#events level2/etc.} for method \texttt{class}, \texttt{n} for method \texttt{anova}, and \texttt{\#events/n} for methods \texttt{poisson} and \texttt{exp}).
- \texttt{horizontal}: Logical. If \texttt{TRUE} (default), plot is horizontal. If \texttt{FALSE}, plot appears as landscape.
- \ldots: other arguments to the \texttt{postscript} function.

Details

The plot created uses the functions \texttt{plot.rpart} and \texttt{text.rpart} (with the \texttt{fancy} option). The settings were chosen because they looked good to us, but other options may be better, depending on the \texttt{rpart} object. Users are encouraged to write their own function containing favorite options.
Side Effects

A plot of rpart is created using the postscript driver, or the current device if filename = "".

See Also

plot.rpart, rpart, text.rpart, abbreviate

Examples

z.auto <- rpart(Mileage ~ Weight, car.test.frame)
post(z.auto, file = "")  # display tree on active device
  # now construct postscript version on file "pretty.ps"
  # with no title
post(z.auto, file = "pretty.ps", title = "")
z.hp <- rpart(Mileage ~ Weight + HP, car.test.frame)
post(z.hp)

predict.rpart  Predictions from a Fitted Rpart Object

Description

Returns a vector of predicted responses from a fitted rpart object.

Usage

## S3 method for class 'rpart'
predict(object, newdata,
    type = c("vector", "prob", "class", "matrix"),
    na.action = na.pass, ...)  

Arguments

object  
fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the rpart function.

newdata  
data frame containing the values at which predictions are required. The predictors referred to in the right side of formula(object) must be present by name in newdata. If missing, the fitted values are returned.

type  
character string denoting the type of predicted value returned. If the rpart object is a classification tree, then the default is to return prob predictions, a matrix whose columns are the probability of the first, second, etc. class. (This agrees with the default behavior of tree). Otherwise, a vector result is returned.

na.action  
a function to determine what should be done with missing values in newdata. The default is to pass them down the tree using surrogates in the way selected when the model was built. Other possibilities are na.omit and na.fail.

...  
further arguments passed to or from other methods.
Details

This function is a method for the generic function predict for class "rpart". It can be invoked by calling predict for an object of the appropriate class, or directly by calling predict.rpart regardless of the class of the object.

Value

A new object is obtained by dropping newdata down the object. For factor predictors, if an observation contains a level not used to grow the tree, it is left at the deepest possible node and frame$yval at the node is the prediction.

If type = "vector":
vector of predicted responses. For regression trees this is the mean response at the node, for Poisson trees it is the estimated response rate, and for classification trees it is the predicted class (as a number).

If type = "prob":
(for a classification tree) a matrix of class probabilities.

If type = "matrix":
a matrix of the full responses (frame$yval2 if this exists, otherwise frame$yval). For regression trees, this is the mean response, for Poisson trees it is the response rate and the number of events at that node in the fitted tree, and for classification trees it is the concatenation of at least the predicted class, the class counts at that node in the fitted tree, and the class probabilities (some versions of rpart may contain further columns).

If type = "class":
(for a classification tree) a factor of classifications based on the responses.

See Also

predict, rpart.object

Examples

z.auto <- rpart(Mileage ~ Weight, car.test.frame)
predict(z.auto)

fit <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis)
predict(fit, type = "prob")  # class probabilities (default)
predict(fit, type = "vector")  # level numbers
predict(fit, type = "class")  # factor
predict(fit, type = "matrix")  # level number, class frequencies, probabilities

sub <- c(sample(1:50, 25), sample(51:100, 25), sample(101:150, 25))
fit <- rpart(Species ~ ., data = iris, subset = sub)
fit
table(predict(fit, iris[-sub,], type = "class"), iris[-sub, "Species"])
Description

This function prints an rpart object. It is a method for the generic function print of class "rpart".

Usage

```r
## S3 method for class 'rpart'
print(x, minlength = 0, spaces = 2, cp, digits = getOption("digits"), ...)
```

Arguments

- `x`: fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the rpart function.
- `minlength`: Controls the abbreviation of labels: see `labels.rpart`.
- `spaces`: the number of spaces to indent nodes of increasing depth.
- `digits`: the number of digits of numbers to print.
- `cp`: prune all nodes with a complexity less than cp from the printout. Ignored if unspecified.
- `...`: arguments to be passed to or from other methods.

Details

This function is a method for the generic function print for class "rpart". It can be invoked by calling print for an object of the appropriate class, or directly by calling `print.rpart` regardless of the class of the object.

Side Effects

A semi-graphical layout of the contents of x$frame is printed. Indentation is used to convey the tree topology. Information for each node includes the node number, split, size, deviance, and fitted value. For the "class" method, the class probabilities are also printed.

See Also

`print.rpart.object`, `summary.rpart`, `printcp`
Examples

```r
z.auto <- rpart(Mileage ~ Weight, car.test.frame)
z.auto
```

```r
## Not run: node, split, n, deviance, yval
* denotes terminal node
1) root 60 1354.58300 24.58333
   2) Weight>=2567.5 45 361.20000 22.46667
   5) Weight<3087.5 45 101.20000 14.90909 *
   6) Weight<3087.5 33 167.20000 22.34375
      7) Weight<2747.5 22 110.20000 12.16667 *
      8) Weight<2747.5 11 38.20000 9.99091 *
      9) Weight<2747.5 11 40.20000 14.18182 *
   10) Weight<2547.5 15 60.40000 23.80000 *
```

```r
## End(Not run)
```

---

### `printcp`

**Displays CP table for Fitted Rpart Object**

**Description**

Displays the cp table for fitted rpart object.

**Usage**

```r
printcp(x, digits = getOption("digits") - 2)
```

**Arguments**

- `x`: fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the `rpart` function.

- `digits`: the number of digits of numbers to print.

**Details**

Prints a table of optimal prunings based on a complexity parameter.

**See Also**

- `summary.rpart`, `rpart.object`
Examples

z.auto <- rpart(Mileage ~ Weight, car.test.frame)
printcp(z.auto)
## Not run:
Regression tree:
rpart(formula = Mileage ~ Weight, data = car.test.frame)

Variables actually used in tree construction:
[1] Weight

Root node error: 1354.6/60 = 22.576

CP nsplit rel error xerror xstd
1 0.505349 0 1.00000 1.03436 0.178526
2 0.134528 1 0.40465 0.60508 0.105217
3 0.012828 2 0.27012 0.45153 0.083330
4 0.010000 3 0.25729 0.44826 0.076998

## End(Not run)

prune.rpart

Cost-complexity Pruning of an Rpart Object

Description

Determines a nested sequence of subtrees of the supplied rpart object by recursively snipping off the least important splits, based on the complexity parameter (cp).

Usage

prune(tree, ...)

## S3 method for class 'rpart'
prune(tree, cp, ...)

Arguments

tree fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the rpart function.

cp Complexity parameter to which the rpart object will be trimmed.

... further arguments passed to or from other methods.

Value

A new rpart object that is trimmed to the value cp.
residuals.rpart

See Also

rpart

Examples

    z.auto <- rpart(Mileage ~ Weight, car.test.frame)
    zm <- prune(z.auto, cp = 0.1)
    plot(zm) #plot smaller rpart object

residuals.rpart  Residuals From a Fitted Rpart Object

Description

Method for residuals for an rpart object.

Usage

    ## S3 method for class 'rpart'
    residuals(object, type = c("usual", "pearson", "deviance"), ...)

Arguments

    object    fitted model object of class "rpart".
    type      Indicates the type of residual desired.
               For regression or anova trees all three residual definitions reduce to y - fitted.
               This is the residual returned for user method trees as well.
               For classification trees the usual residuals are the misclassification losses L(actual, 
               predicted) where L is the loss matrix. With default losses this residual is 0/1 for 
               correct/incorrect classification. The pearson residual is (1-fitted)/sqrt(fitted(1- 
               fitted)) and the deviance residual is sqrt(minus twice logarithm of fitted).
               For poisson and exp (or survival) trees, the usual residual is the observed - 
               expected number of events. The pearson and deviance residuals are as defined 
               in McCullagh and Nelder.
    ...

Value

    Vector of residuals of type type from a fitted rpart object.

References

Examples

```r
fit <- rpart(skips ~ Opening + Solder + Mask + PadType + Panel,
             data = solder, method = "anova")
summary(residuals(fit))
plot(predict(fit), residuals(fit))
```

Description

Fit a rpart model

Usage

```r
rpart(formula, data, weights, subset, na.action = na.rpart, method,
       model = FALSE, x = FALSE, y = TRUE, parms, control, cost, ...)
```

Arguments

- `formula` a formula, with a response but no interaction terms. If this a a data frame, that is taken as the model frame (see `model.frame`).
- `data` an optional data frame in which to interpret the variables named in the formula.
- `weights` optional case weights.
- `subset` optional expression saying that only a subset of the rows of the data should be used in the fit.
- `na.action` the default action deletes all observations for which `y` is missing, but keeps those in which one or more predictors are missing.
- `method` one of "anova", "poisson", "class" or "exp". If method is missing then the routine tries to make an intelligent guess. If `y` is a survival object, then `method = "exp"` is assumed, if `y` has 2 columns then `method = "poisson"` is assumed, if `y` is a factor then `method = "class"` is assumed, otherwise `method = "anova"` is assumed. It is wisest to specify the method directly, especially as more criteria may added to the function in future. Alternatively, method can be a list of functions named `init`, `split` and `eval`. Examples are given in the file ‘tests/usersplits.R’ in the sources, and in the vignettes ‘User Written Split Functions’.
- `model` if logical: keep a copy of the model frame in the result? If the input value for model is a model frame (likely from an earlier call to the rpart function), then this frame is used rather than constructing new data.
- `x` keep a copy of the x matrix in the result.
- `y` keep a copy of the dependent variable in the result. If missing and model is supplied this defaults to FALSE.
**rpart**

`parms` optional parameters for the splitting function.

Anova splitting has no parameters.

Poisson splitting has a single parameter, the coefficient of variation of the prior distribution on the rates. The default value is 1.

Exponential splitting has the same parameter as Poisson.

For classification splitting, 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 off-diagonal elements. The splitting index can be `gini` or `information`. The default priors are proportional to the data counts, the losses default to 1, and the split defaults to `gini`.

`control` a list of options that control details of the `rpart` algorithm. See `rpart.control`.

`cost` a vector of non-negative costs, one for each variable in the model. Defaults to one for all variables. These are scalings to be applied when considering splits, so the improvement on splitting on a variable is divided by its cost in deciding which split to choose.

... arguments to `rpart.control` may also be specified in the call to `rpart`. They are checked against the list of valid arguments.

**Details**

This differs from the `tree` function in S mainly in its handling of surrogate variables. In most details it follows Breiman *et. al* (1984) quite closely. R package **tree** provides a re-implementation of `tree`.

**Value**

An object of class `rpart`. See `rpart.object`.

**References**


**See Also**

`rpart.control`, `rpart.object`, `summary.rpart`, `print.rpart`

**Examples**

```r
fit <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis)
fit2 <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis, 
  parms = list(prior = c(.65,.35), split = "information"))
fit3 <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis, 
  control = rpart.control(cp = 0.05))
par(mfrow = c(1,2), xpd = NA) # otherwise on some devices the text is clipped
plot(fit)
text(fit, use.n = TRUE)
plot(fit2)
text(fit2, use.n = TRUE)
```
Description

Various parameters that control aspects of the rpart fit.

Usage

```r
rpart.control(minsplit = 20, minbucket = round(minsplit/3), cp = 0.01,
               maxcompete = 4, maxsurrogate = 5, usesurrogate = 2, xval = 10,
               surrogatestyle = 0, maxdepth = 30, ...)
```

Arguments

- **minsplit**: the minimum number of observations that must exist in a node in order for a split to be attempted.
- **minbucket**: the minimum number of observations in any terminal <leaf> node. If only one of minbucket or minsplit is specified, the code either sets minsplit to minbucket or minbucket to minsplit/3, as appropriate.
- **cp**: complexity parameter. Any split that does not decrease the overall lack of fit by a factor of cp is not attempted. For instance, with anova splitting, this means that the overall R-squared must increase by cp at each step. The main role of this parameter is to save computing time by pruning off splits that are obviously not worthwhile. Essentially, the user informs the program that any split which does not improve the fit by cp will likely be pruned off by cross-validation, and that hence the program need not pursue it.
- **maxcompete**: the number of competitor splits retained in the output. It is useful to know not just which split was chosen, but which variable came in second, third, etc.
- **maxsurrogate**: the number of surrogate splits retained in the output. If this is set to zero the compute time will be reduced, since approximately half of the computational time (other than setup) is used in the search for surrogate splits.
- **usesurrogate**: how to use surrogates in the splitting process. 0 means display only; an observation with a missing value for the primary split rule is not sent further down the tree. 1 means use surrogates, in order, to split subjects missing the primary variable; if all surrogates are missing the observation is not split. For value 2, if all surrogates are missing, then send the observation in the majority direction. A value of 0 corresponds to the action of tree, and 2 to the recommendations of Breiman et.al (1984).
- **xval**: number of cross-validations.
- **surrogatestyle**: controls the selection of a best surrogate. If set to 0 (default) the program uses the total number of correct classification for a potential surrogate variable, if set to 1 it uses the percent correct, calculated over the non-missing values of the surrogate. The first option more severely penalizes covariates with a large number of missing values.
**maxdepth**  Set the maximum depth of any node of the final tree, with the root node counted as depth 0. Values greater than 30 `rpart` will give nonsense results on 32-bit machines.

... mop up other arguments.

**Value**

A list containing the options.

**See Also**

`rpart`

---

**Description**

This function does the initialization step for `rpart`, when the response is a survival object. It rescales the data so as to have an exponential baseline hazard and then uses Poisson methods. This function would rarely if ever be called directly by a user.

**Usage**

```r
rpart.exp(y, offset, parms, wt)
```

**Arguments**

- `y`: the response, which will be of class `Surv`
- `offset`: optional offset
- `parms`: parameters controlling the fit. This is a list with components `shrink` and `method`. The first is the prior for the coefficient of variation of the predictions. The second is either "deviance" or "sqrt" and is the measure used for cross-validation. If values are missing the defaults are used, which are "deviance" for the method, and a shrinkage of 1.0 for the deviance method and 0 for the square root.
- `wt`: case weights, if present

**Value**

a list with the necessary initialization components

**Author(s)**

Terry Therneau

**See Also**

`rpart`
rpart.object  

Recursive Partitioning and Regression Trees Object

Description

These are objects representing fitted rpart trees.

Value

- **frame**: data frame with one row for each node in the tree. The *row.names* of *frame* contain the (unique) node numbers that follow a binary ordering indexed by node depth. Columns of *frame* include *var*, a factor giving the names of the variables used in the split at each node (leaf nodes are denoted by the level "<leaf>"); *n*, the number of observations reaching the node; *wt*, the sum of case weights for observations reaching the node; *dev*, the deviance of the node; *yval*, the fitted value of the response at the node; and *splits*, a two column matrix of left and right split labels for each node. Also included in the frame are *complexity*, the complexity parameter at which this split will collapse, *ncompete*, the number of competitor splits recorded, and *nsurrogate*, the number of surrogate splits recorded.

  Extra response information which may be present is in *yval2*, which contains the number of events at the node (poisson tree), or a matrix containing the fitted class, the class counts for each node, the class probabilities and the ‘node probability’ (classification trees).

- **where**: an integer vector of the same length as the number of observations in the root node, containing the row number of *frame* corresponding to the leaf node that each observation falls into.

- **call**: an image of the call that produced the object, but with the arguments all named and with the actual formula included as the formula argument. To re-evaluate the call, say `update(tree)`.

- **terms**: an object of class *c("terms", "formula")* (see *terms.object*) summarizing the formula. Used by various methods, but typically not of direct relevance to users.

- **splits**: a numeric matrix describing the splits: only present if there are any. The row label is the name of the split variable, and columns are count, the number of observations (which are not missing and are of positive weight) sent left or right by the split (for competitor splits this is the number that would have been sent left or right had this split been used, for surrogate splits it is the number missing the primary split variable which were decided using this surrogate), *ncat*, the number of categories or levels for the variable (+/-1 for a continuous variable), *improve*, which is the improvement in deviance given by this split, or, for surrogates, the concordance of the surrogate with the primary, and *index*, the numeric split point. The last column *adj* gives the adjusted concordance for surrogate splits. For a factor, the *index* column contains the row number of the csplit matrix. For a continuous variable, the sign of *ncat* determines whether the subset $x < \text{cutpoint}$ or $x \geq \text{cutpoint}$ is sent to the left.
rsq.rpart

rsq.rpart

Plots the Approximate R-Square for the Different Splits

Description

Produces 2 plots. The first plots the r-square (apparent and apparent - from cross-validation) versus the number of splits. The second plots the Relative Error(cross-validation) +/- 1-SE from cross-validation versus the number of splits.

Usage

rsq.rpart(x)
Arguments

x fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the rpart function.

Side Effects

Two plots are produced.

Note

The labels are only appropriate for the "anova" method.

Examples

z.auto <- rpart(Mileage ~ Weight, car.test.frame)
rsq.rpart(z.auto)

snip.rpart(x, toss)

Description

Creates a "snipped" rpart object, containing the nodes that remain after selected subtrees have been snipped off. The user can snip nodes using the toss argument, or interactively by clicking the mouse button on specified nodes within the graphics window.

Usage

snip.rpart(x, toss)

Arguments

x fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the rpart function.

toss an integer vector containing indices (node numbers) of all subtrees to be snipped off. If missing, user selects branches to snip off as described below.

Details

A dendrogram of rpart is expected to be visible on the graphics device, and a graphics input device (e.g., a mouse) is required. Clicking (the selection button) on a node displays the node number, sample size, response y-value, and Error (dev). Clicking a second time on the same node snips that subtree off and visually erases the subtree. This process may be repeated an number of times. Warnings result from selecting the root or leaf nodes. Clicking the exit button will stop the snipping process and return the resulting rpart object.

See the documentation for the specific graphics device for details on graphical input techniques.
**Value**

A *rpart* object containing the nodes that remain after specified or selected subtrees have been snipped off.

**Warning**

Visually erasing the plot is done by over-plotting with the background colour. This will do nothing if the background is transparent (often true for screen devices).

**See Also**

*plot.rpart*

**Examples**

```r
## dataset not in R
## Not run:
z.survey <- rpart(market.survey) # grow the rpart object
plot(z.survey) # plot the tree
z.survey2 <- snip.rpart(z.survey, toss = 2) # trim subtree at node 2
plot(z.survey2) # plot new tree

# can also interactively select the node using the mouse in the
# graphics window

## End(Not run)
```

**solder**

_Soldering of Components on Printed-Circuit Boards_

**Description**

The `solder` data frame has 720 rows and 6 columns, representing a balanced subset of a designed experiment varying 5 factors on the soldering of components on printed-circuit boards.

**Usage**

`solder`

**Format**

This data frame contains the following columns:

- **Opening** a factor with levels 'L', 'M' and 'S' indicating the amount of clearance around the mounting pad.
- **Solder** a factor with levels 'Thick' and 'Thin' giving the thickness of the solder used.
- **Mask** a factor with levels 'A1.5', 'A3', 'B3' and 'B6' indicating the type and thickness of mask used.

Panel 1:3 indicating the panel on a board being tested.

skips a numeric vector giving the number of visible solder skips.

Source


Examples

```r
fit <- rpart(skips ~ Opening + Solder + Mask + PadType + Panel,
             data = solder, method = "anova")
summary(residuals(fit))
plot(predict(fit), residuals(fit))
```

---

**stagec**  
*Stage C Prostate Cancer*

**Description**

A set of 146 patients with stage C prostate cancer, from a study exploring the prognostic value of flow cytometry.

**Usage**

data(stagec)

**Format**

A data frame with 146 observations on the following 8 variables.

- **pgtime**  Time to progression or last follow-up (years)
- **pgstat**  1 = progression observed, 0 = censored
- **age**  age in years
- **eet**  early endocrine therapy, 1 = no, 2 = yes
- **g2**  percent of cells in G2 phase, as found by flow cytometry
- **grade**  grade of the tumor, Farrow system
- **gleason**  grade of the tumor, Gleason system
- **ploidy**  the ploidy status of the tumor, from flow cytometry. Values are ‘diploid’, ‘tetraploid’, and ‘aneuploid’
Details

A tumor is called diploid (normal complement of dividing cells) if the fraction of cells in G2 phase was determined to be 13% or less. Aneuploid cells have a measurable fraction with a chromosome count that is neither 24 nor 48, for these the G2 percent is difficult or impossible to measure.

Examples

```r
require(survival)
rpart(Surv(pgtime, pgstat) ~ ., stagec)
```

summary.rpart

---

**Summary a Fitted Rpart Object**

### Description

Returns a detailed listing of a fitted rpart object.

### Usage

```r
## S3 method for class 'rpart'
summary(object, cp = 0, digits = getOption("digits"), file, ...)
```

### Arguments

- `object`: fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the `rpart` function.
- `digits`: Number of significant digits to be used in the result.
- `cp`: trim nodes with a complexity of less than `cp` from the listing.
- `file`: write the output to a given file name. (Full listings of a tree are often quite long).
- `...`: arguments to be passed to or from other methods.

### Details

This function is a method for the generic function `summary` for class "rpart". It can be invoked by calling `summary` for an object of the appropriate class, or directly by calling `summary.rpart` regardless of the class of the object.

It prints the call, the table shown by `printcp`, the variable importance (summing to 100) and details for each node (the details depending on the type of tree).

### See Also

`summary.rpart.object, printcp`
Examples

```r
## a regression tree
z.auto <- rpart(Mileage ~ Weight, car.test.frame)
summary(z.auto)

## a classification tree with multiple variables and surrogate splits.
summary(rpart(Kyphosis ~ Age + Number + Start, data = kyphosis))
```

---

text.rpart

### Place Text on a Dendrogram Plot

**Description**

Labels the current plot of the tree dendrogram with text.

**Usage**

```r
## S3 method for class 'rpart'
text(x, splits = TRUE, label = NULL, FUN = text, all = FALSE,
      pretty = NULL, digits =getOption("digits") - 3, use.n = FALSE,
      fancy = FALSE, fwidth = 0.8, fheight = 0.8, bg = par("bg"),
      minlength = 1L, ...)
```

**Arguments**

- **x**: fitted model object of class "rpart". This is assumed to be the result of some function that produces an object with the same named components as that returned by the rpart function.
- **splits**: logical flag. If TRUE (default), then the splits in the tree are labeled with the criterion for the split.
- **label**: For compatibility with rpart2, ignored in this version (with a warning).
- **FUN**: the name of a labeling function, e.g. text.
- **all**: Logical. If TRUE, all nodes are labeled, otherwise just terminal nodes.
- **minlength**: the length to use for factor labels. A value of 1 causes them to be printed as ‘a’, ‘b’, ..... Larger values use abbreviations of the label names. See the labels.rpart function for details.
- **pretty**: an alternative to the minlength argument, see labels.rpart.
- **digits**: number of significant digits to include in numerical labels.
- **use.n**: Logical. If TRUE, adds to label (\#events level1/\#events level2/etc. for class, \#events/n for anova, and \#events/n for poisson and exp).
- **fancy**: Logical. If TRUE, nodes are represented by ellipses (interior nodes) and rectangles (leaves) and labeled by yval. The edges connecting the nodes are labeled by left and right splits.
### fwidth
Relates to option `fancy` and the width of the ellipses and rectangles. If `fwidth < 1` then it is a scaling factor (default = 0.8). If `fwidth > 1` then it represents the number of character widths (for current graphical device) to use.

### fheight
Relates to option `fancy` and the height of the ellipses and rectangles. If `fheight < 1` then it is a scaling factor (default = 0.8). If `fheight > 1` then it represents the number of character heights (for current graphical device) to use.

### bg
The color used to paint the background to annotations if `fancy = TRUE`.

### ... Graphical parameters may also be supplied as arguments to this function (see `par`). As labels often extend outside the plot region it can be helpful to specify `xpd = TRUE`.

### Side Effects
the current plot of a tree dendrogram is labeled.

### See Also
text, plot.rpart, rpart.labels.rpart, abbreviate

### Examples
```r
freen.tr <- rpart(y ~ ., freeny)
par(xpd = TRUE)
plot(freen.tr)
text(freen.tr, use.n = TRUE, all = TRUE)
```

---

### xpred.rpart

<table>
<thead>
<tr>
<th>Return Cross-Validated Predictions</th>
</tr>
</thead>
</table>

### Description
Gives the predicted values for an `rpart` fit, under cross validation, for a set of complexity parameter values.

### Usage
```r
xpred.rpart(fit, xval = 10, cp, return.all = FALSE)
```

### Arguments
- **fit**: a object of class "rpart".
- **xval**: number of cross-validation groups. This may also be an explicit list of integers that define the cross-validation groups.
- **cp**: the desired list of complexity values. By default it is taken from the `cptable` component of the fit.
- **return.all**: if FALSE return only the first element of the prediction
Details

Complexity penalties are actually ranges, not values. If the cp values found in the table were .36, .28, and .13, for instance, this means that the first row of the table holds for all complexity penalties in the range [.36, 1], the second row for cp in the range [.28, .36) and the third row for [.13, .28). By default, the geometric mean of each interval is used for cross validation.

Value

A matrix with one row for each observation and one column for each complexity value. If return.all is TRUE and the prediction for each node is a vector, then the result will be an array containing all of the predictions. When the response is categorical, for instance, the result contains the predicted class followed by the class probabilities of the selected terminal node; result[1,] will be the matrix of predicted classes, result[2,] the matrix of class 1 probabilities, etc.

See Also

rpart

Examples

```r
fit <- rpart(Mileage ~ Weight, car.test.frame)
xmat <- xpred.rpart(fit)
xerr <- (xmat - car.test.frame$Mileage)^2
apply(xerr, 2, sum)  # cross-validated error estimate

# approx same result as rel. error from printcp(fit)
apply(xerr, 2, sum)/var(car.test.frame$Mileage)
printcp(fit)
```
Index

*Topic **datasets**
  - car.test.frame, 2
  - car90, 3
  - cu.summary, 5
  - kyphosis, 6
  - solder, 27
  - stagec, 28

*Topic **methods**
  - rpart.object, 24

*Topic **tree**
  - labels.rpart, 7
  - meanvar.rpart, 8
  - na.rpart, 9
  - path.rpart, 9
  - plot.rpart, 10
  - plotcp, 12
  - post.rpart, 13
  - predict.rpart, 14
  - print.rpart, 16
  - printcp, 17
  - prune.rpart, 18
  - residuals.rpart, 19
  - rpart, 20
  - rpart.control, 22
  - rpart.exp, 23
  - rpart.object, 24
  - rsq.rpart, 25
  - snip.rpart, 26
  - summary.rpart, 29
  - text.rpart, 30
  - xpred.rpart, 31

abbreviate, 7, 14, 31
attributes, 25

- car.test.frame, 2, 4, 5
- car90, 3, 3, 5
- cu.summary, 2–4, 5
- formula, 20

kyphosis, 6
- labels.rpart, 7, 16, 30, 31
- meanvar (meanvar.rpart), 8
- meanvar.rpart, 8
- model.frame, 20, 25
- na.fail, 14
- na.omit, 14
- na.rpart, 9
- path.rpart, 9
- plot.rpart, 8, 10, 14, 27, 31
- plotcp, 12
- post (post.rpart), 13
- post.rpart, 13
- predict, 15
- predict.rpart, 14
- print, 16
- print.rpart, 16, 21
- printcp, 12, 16, 17, 29
- prune (prune.rpart), 18
- prune.rpart, 18
- residuals.rpart, 19
- rpart, 10–12, 14, 19, 20, 23, 25, 31, 32
- rpart.control, 21, 22
- rpart.exp, 23
- rpart.object, 12, 15–17, 21, 24, 29
- rsq.rpart, 25
- snip.rpart, 26
- solder, 27
- stagec, 28
- summary, 29
- summary.rpart, 16, 17, 21, 25, 29
- terms.object, 24
- text, 31
- text.rpart, 11, 14, 30
tree, 14
xpred.rpart, 31