Package 'Cubist'

October 25, 2025

Type Package

Title Rule- And Instance-Based Regression Modeling

Version 0.5.1

Maintainer Max Kuhn <mxkuhn@gmail.com>

Description Regression modeling using rules with added instance-based corrections.

License GPL-3

URL https://topepo.github.io/Cubist/, https://github.com/topepo/Cubist

BugReports https://github.com/topepo/Cubist/issues

Depends lattice

Imports reshape2, utils

Suggests covr, dplyr (>= 0.7.4), knitr, mlbench, modeldata, rlang, rmarkdown, rules, testthat (>= 3.0.0)

VignetteBuilder knitr

Biarch true

Config/Needs/website pkgdown, caret, tidymodels, rules

Config/testthat/edition 3

Encoding UTF-8

LazyLoad yes

RoxygenNote 7.3.2

NeedsCompilation yes

Author Max Kuhn [aut, cre],
Steve Weston [ctb],
Chris Keefer [ctb],
Nathan Coulter [ctb],
Ross Quinlan [aut] (Author of imported C code),
Rulequest Research Pty Ltd. [cph] (Copyright holder of imported C code)

Repository CRAN

Date/Publication 2025-10-25 05:10:18 UTC

Contents

`cubist.default` *Fit a Cubist model*

Description

This function fits the rule-based model described in Quinlan (1992) (aka M5) with additional corrections based on nearest neighbors in the training set, as described in Quinlan (1993a).

Usage

```
## Default S3 method:
cubist(x, y, committees = 1, control = cubistControl(), weights = NULL, ...)
```

Arguments

`x`	a matrix or data frame of predictor variables. Missing data are allowed but (at this time) only numeric, character and factor values are allowed. Must have column names.
`y`	a numeric vector of outcome
`committees`	an integer: how many committee models (e.g.. boosting iterations) should be used?
`control`	options that control details of the `cubist` algorithm. See `cubistControl()`
`weights`	an optional vector of case weights (the same length as `y`) for how much each instance should contribute to the model fit. From the RuleQuest website: "The relative weight assigned to each case is its value of this attribute divided by the average value; if the value is undefined, not applicable, or is less than or equal to zero, the case's relative weight is set to 1."
`...`	optional arguments to pass (not currently used)

Details

Cubist is a prediction-oriented regression model that combines the ideas in Quinlan (1992) and Quinlan (1993).

Although it initially creates a tree structure, it collapses each path through the tree into a rule. A regression model is fit for each rule based on the data subset defined by the rules. The set of rules are pruned or possibly combined. and the candidate variables for the linear regression models are the

predictors that were used in the parts of the rule that were pruned away. This part of the algorithm is consistent with the "M5" or Model Tree approach.

Cubist generalizes this model to add boosting (when `committees` > 1) and instance based corrections (see `predict.cubist()`). The number of instances is set at prediction time by the user and is not needed for model building.

This function links R to the GPL version of the C code given on the RuleQuest website.

The RuleQuest code differentiates missing values from values that are not applicable. Currently, this packages does not make such a distinction (all values are treated as missing). This will produce slightly different results.

To tune the cubist model over the number of committees and neighbors, the `caret::train()` function in the `caret` package has bindings to find appropriate settings of these parameters.

Value

an object of class `cubist` with elements:

`data`, `names`, `model`
: character strings that correspond to their counterparts for the command-line program available from RuleQuest

`output`
: basic cubist output captured from the C code, including the rules, their terminal models and variable usage statistics

`control`
: a list of control parameters passed in by the user

`composite`, `neighbors`, `committees`
: mirrors of the values to these arguments that were passed in by the user

`dims`
: the output if `dim(x)`

`splits`
: information about the variables and values used in the rule conditions

`call`
: the function call

`coefs`
: a data frame of regression coefficients for each rule within each committee

`vars`
: a list with elements `all` and `used` listing the predictors passed into the function and used by any rule or model

`fitted.values`
: a numeric vector of predictions on the training set.

`usage`
: a data frame with the percent of models where each variable was used. See `summary.cubist()` for a discussion.

Author(s)

R code by Max Kuhn, original C sources by R Quinlan and modifications be Steve Weston

References

Quinlan. Learning with continuous classes. Proceedings of the 5th Australian Joint Conference On Artificial Intelligence (1992) pp. 343-348

Quinlan. Combining instance-based and model-based learning. Proceedings of the Tenth International Conference on Machine Learning (1993a) pp. 236-243

Quinlan. **C4.5: Programs For Machine Learning** (1993b) Morgan Kaufmann Publishers Inc. San Francisco, CA

Wang and Witten. Inducing model trees for continuous classes. Proceedings of the Ninth European Conference on Machine Learning (1997) pp. 128-137

http://rulequest.com/cubist-info.html

See Also

cubistControl(), predict.cubist(), summary.cubist(), dotplot.cubist(), caret::train()

Examples

```
library(mlbench)
data(BostonHousing)

## 1 committee, so just an M5 fit:
mod1 <- cubist(x = BostonHousing[, -14], y = BostonHousing$medv)
mod1

## Now with 10 committees
mod2 <- cubist(x = BostonHousing[, -14], y = BostonHousing$medv, committees = 10)
mod2
```

---

cubistControl *Various parameters that control aspects of the Cubist fit.*

Description

Most of these values are discussed at length in http://rulequest.com/cubist-unix.html

Usage

```
cubistControl(
  unbiased = FALSE,
  rules = 100,
  extrapolation = 100,
  sample = 0,
  seed = sample.int(4096, size = 1) - 1L,
  label = "outcome"
)
```

Arguments

unbiased	a logical: should unbiased rules be used?
rules	an integer (or NA): define an explicit limit to the number of rules used (NA let's Cubist decide).

extrapolation	a number between 0 and 100: since Cubist uses linear models, predictions can be outside of the outside of the range seen the training set. This parameter controls how much rule predictions are adjusted to be consistent with the training set.
sample	a number between 0 and 99.9: this is the percentage of the data set to be randomly selected for model building (not for out-of-bag type evaluation).
seed	an integer for the random seed (in the C code)
label	a label for the outcome (when printing rules)

Value

A list containing the options.

Author(s)

Max Kuhn

References

Quinlan. Learning with continuous classes. Proceedings of the 5th Australian Joint Conference On Artificial Intelligence (1992) pp. 343-348

Quinlan. Combining instance-based and model-based learning. Proceedings of the Tenth International Conference on Machine Learning (1993) pp. 236-243

Quinlan. **C4.5: Programs For Machine Learning** (1993) Morgan Kaufmann Publishers Inc. San Francisco, CA

http://rulequest.com/cubist-info.html

See Also

cubist(), predict.cubist(), summary.cubist(), predict.cubist(), dotplot.cubist()

Examples

```
cubistControl()
```

---

dotplot.cubist *Visualization of Cubist Rules and Equations*

Description

Lattice dotplots of the rule conditions or the linear model coefficients produced by cubist() objects

Usage

```
## S3 method for class 'cubist'
dotplot(x, data = NULL, what = "splits", committee = NULL, rule = NULL, ...)
```

Arguments

x	a cubist() object
data	not currently used (here for lattice compatibility)
what	either "splits" or "coefs"
committee	which committees to plot
rule	which rules to plot
...	options to pass to lattice::dotplot()

Details

For the splits, a panel is created for each predictor. The x-axis is the range of the predictor scaled to be between zero and one and the y-axis has a line for each rule (within each committee). Areas are colored as based on their region. For example, if one rule has `var1 < 10`, the linear for this rule would be colored. If another rule had the complementary region of `var1 <= 10`, it would be on another line and shaded a different color.

For the coefficient plot, another dotplot is made. The layout is the same except the the x-axis is in the original units and has a dot if the rule used that variable in a linear model.

Value

a lattice::dotplot() object

Author(s)

R code by Max Kuhn, original C sources by R Quinlan and modifications be Steve Weston

References

Quinlan. Learning with continuous classes. Proceedings of the 5th Australian Joint Conference On Artificial Intelligence (1992) pp. 343-348

Quinlan. Combining instance-based and model-based learning. Proceedings of the Tenth International Conference on Machine Learning (1993) pp. 236-243

Quinlan. **C4.5: Programs For Machine Learning** (1993) Morgan Kaufmann Publishers Inc. San Francisco, CA

http://rulequest.com/cubist-info.html

See Also

cubist(), cubistControl(), predict.cubist(), summary.cubist(), predict.cubist(), lattice::dotplot()

Examples

```
library(mlbench)
data(BostonHousing)

## 1 committee and no instance-based correction, so just an M5 fit:
mod1 <- cubist(x = BostonHousing[, -14], y = BostonHousing$medv)
```

```
dotplot(mod1, what = "splits")
dotplot(mod1, what = "coefs")

## Now with 10 committees
mod2 <- cubist(x = BostonHousing[, -14],
               y = BostonHousing$medv,
               committees = 10)
dotplot(mod2, scales = list(y = list(cex = .25)))
dotplot(mod2, what = "coefs",
        between = list(x = 1, y = 1),
        scales = list(x = list(relation = "free"),
                      y = list(cex = .25)))
```

`exportCubistFiles` *Export Cubist Information To the File System*

Description

For a fitted cubist object, text files consistent with the RuleQuest command-line version can be exported.

Usage

```
exportCubistFiles(x, neighbors = 0, path = getwd(), prefix = NULL)
```

Arguments

`x`	a `cubist()` object
`neighbors`	how many, if any, neighbors should be used to correct the model predictions
`path`	the path to put the files
`prefix`	a prefix (or "filestem") for creating files

Details

Using the RuleQuest specifications, `model`, `names` and `data` files are created for use with the command-line version of the program.

Value

No value is returned. Three files are written out.

Author(s)

Max Kuhn

References

Quinlan. Learning with continuous classes. Proceedings of the 5th Australian Joint Conference On Artificial Intelligence (1992) pp. 343-348

Quinlan. Combining instance-based and model-based learning. Proceedings of the Tenth International Conference on Machine Learning (1993) pp. 236-243

Quinlan. **C4.5: Programs For Machine Learning** (1993) Morgan Kaufmann Publishers Inc. San Francisco, CA

http://rulequest.com/cubist-info.html

See Also

cubist(), predict.cubist(), summary.cubist(), predict.cubist()

Examples

```
library(mlbench)
data(BostonHousing)

mod1 <- cubist(x = BostonHousing[, -14], y = BostonHousing$medv)
exportCubistFiles(mod1, neighbors = 8, path = tempdir(), prefix = "BostonHousing")
```

---

predict.cubist *Predict method for cubist fits*

Description

Prediction using the parametric model are calculated using the method of Quinlan (1992). If neighbors is greater than zero, these predictions are adjusted by training set instances nearby using the approach of Quinlan (1993).

Usage

```
## S3 method for class 'cubist'
predict(object, newdata = NULL, neighbors = 0, ...)
```

Arguments

object	an object of class cubist
newdata	a data frame of predictors (in the same order as the original training data). Must have column names.
neighbors	an integer from 0 to 9: how many instances to use to correct the rule-based prediction?
...	other options to pass through the function (not currently used)

Details

Note that the predictions can fail for various reasons. For example, as shown in the examples, if the model uses a qualitative predictor and the prediction data has a new level of that predictor, the function will throw an error.

Value

a numeric vector is returned

Author(s)

R code by Max Kuhn, original C sources by R Quinlan and modifications be Steve Weston

References

Quinlan. Learning with continuous classes. Proceedings of the 5th Australian Joint Conference On Artificial Intelligence (1992) pp. 343-348

Quinlan. Combining instance-based and model-based learning. Proceedings of the Tenth International Conference on Machine Learning (1993) pp. 236-243

Quinlan. **C4.5: Programs For Machine Learning** (1993) Morgan Kaufmann Publishers Inc. San Francisco, CA

http://rulequest.com/cubist-info.html

See Also

cubist(), cubistControl(), summary.cubist(), predict.cubist(), dotplot.cubist()

Examples

```
library(mlbench)
data(BostonHousing)

## 1 committee and no instance-based correction, so just an M5 fit:
mod1 <- cubist(x = BostonHousing[, -14], y = BostonHousing$medv)
predict(mod1, BostonHousing[1:4, -14])

## now add instances
predict(mod1, BostonHousing[1:4, -14], neighbors = 5)

# Example error
iris_test <- iris
iris_test$Species <- as.character(iris_test$Species)

mod <- cubist(x = iris_test[1:99, 2:5],
              y = iris_test$Sepal.Length[1:99])

# predict(mod, iris_test[100:151, 2:5])
# Error:
# *** line 2 of `undefined.cases':
# bad value of 'virginica' for attribute 'Species'
```

summary.cubist *Summarizing Cubist Fits*

Description

This function echoes the output of the RuleQuest C code, including the rules, the resulting linear models as well as the variable usage summaries.

Usage

```
## S3 method for class 'cubist'
summary(object, ...)
```

Arguments

object	a cubist() object
...	other options (not currently used)

Details

The Cubist output contains variable usage statistics. It gives the percentage of times where each variable was used in a condition and/or a linear model. Note that this output will probably be inconsistent with the rules shown above. At each split of the tree, Cubist saves a linear model (after feature selection) that is allowed to have terms for each variable used in the current split or any split above it. Quinlan (1992) discusses a smoothing algorithm where each model prediction is a linear combination of the parent and child model along the tree. As such, the final prediction is a function of all the linear models from the initial node to the terminal node. The percentages shown in the Cubist output reflects all the models involved in prediction (as opposed to the terminal models shown in the output).

Value

an object of class summary.cubist with elements

output	a text string of the output
call	the original call to cubist()

Author(s)

R code by Max Kuhn, original C sources by R Quinlan and modifications be Steve Weston

References

Quinlan. Learning with continuous classes. Proceedings of the 5th Australian Joint Conference On Artificial Intelligence (1992) pp. 343-348

Quinlan. Combining instance-based and model-based learning. Proceedings of the Tenth International Conference on Machine Learning (1993) pp. 236-243

Quinlan. **C4.5: Programs For Machine Learning** (1993) Morgan Kaufmann Publishers Inc. San Francisco, CA

http://rulequest.com/cubist-info.html

See Also

cubist(), cubistControl(), predict.cubist(), dotplot.cubist()

Examples

```
library(mlbench)
data(BostonHousing)

## 1 committee and no instance-based correction, so just an M5 fit:
mod1 <- cubist(x = BostonHousing[, -14], y = BostonHousing$medv)
summary(mod1)

## example output:

## Cubist [Release 2.07 GPL Edition]  Sun Apr 10 17:36:56 2011
## -----------------------------------
##
##     Target attribute `outcome'
##
## Read 506 cases (14 attributes) from undefined.data
##
## Model:
##
##   Rule 1: [101 cases, mean 13.84, range 5 to 27.5, est err 1.98]
##
##     if
##     nox > 0.668
##     then
##     outcome = -1.11 + 2.93 dis + 21.4 nox - 0.33 lstat + 0.008 b
##               - 0.13 ptratio - 0.02 crim - 0.003 age + 0.1 rm
##
##   Rule 2: [203 cases, mean 19.42, range 7 to 31, est err 2.10]
##
##     if
##     nox <= 0.668
##     lstat > 9.59
##     then
##     outcome = 23.57 + 3.1 rm - 0.81 dis - 0.71 ptratio - 0.048 age
##               - 0.15 lstat + 0.01 b - 0.0041 tax - 5.2 nox + 0.05 crim
##               + 0.02 rad
##
##   Rule 3: [43 cases, mean 24.00, range 11.9 to 50, est err 2.56]
##
##     if
##     rm <= 6.226
##     lstat <= 9.59
##     then
```

```
##     outcome = 1.18 + 3.83 crim + 4.3 rm - 0.06 age - 0.11 lstat - 0.003 tax
##               - 0.09 dis - 0.08 ptratio
##
##   Rule 4: [163 cases, mean 31.46, range 16.5 to 50, est err 2.78]
##
##     if
##     rm > 6.226
##     lstat <= 9.59
##     then
##     outcome = -4.71 + 2.22 crim + 9.2 rm - 0.83 lstat - 0.0182 tax
##               - 0.72 ptratio - 0.71 dis - 0.04 age + 0.03 rad - 1.7 nox
##               + 0.008 zn
##
##
## Evaluation on training data (506 cases):
##
##     Average  |error|               2.07
##     Relative |error|               0.31
##     Correlation coefficient        0.94
##
##
##     Attribute usage:
##       Conds  Model
##
##        80%   100%    lstat
##        60%    92%    nox
##        40%   100%    rm
##              100%    crim
##              100%    age
##              100%    dis
##              100%    ptratio
##               80%    tax
##               72%    rad
##               60%    b
##               32%    zn
##
##
## Time: 0.0 secs
```

Index