Package 'cv'

January 27, 2024

Type Package

Title Cross-Validation of Regression Models

Version 1.1.0

Date 2024-01-10

Description Cross-validation methods of regression models that exploit features of various modeling functions to improve speed. Some of the methods implemented in the package are novel, as described in the package vignettes; for general introductions to cross-validation, see, for example, Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani (2021, ISBN 978-1-0716-1417-4, Secs. 5.1, 5.3), ``An Introduction to Statistical Learning with Applications in R, Second Edition'', and Trevor Hastie, Robert Tibshirani, and Jerome Friedman (2009, ISBN 978-0-387-84857-0, Sec. 7.10), ``The Elements of Statistical Learning, Second Edition''.

Depends R (>= 3.5.0), doParallel

Imports car, foreach, insight, lme4, MASS, methods, nlme

Suggests boot, carData, dplyr, effects, glmmTMB, ISLR2, knitr, lattice, latticeExtra, leaps, Metrics, microbenchmark, nnet, rmarkdown, spelling, testthat

LazyData TRUE

VignetteBuilder knitr, rmarkdown

License GPL (>= 2)

Encoding UTF-8

Language en-US

RoxygenNote 7.3.1

URL https://gmonette.github.io/cv/, https://CRAN.R-project.org/package=cv

BugReports https://github.com/gmonette/cv/issues

NeedsCompilation no

Author John Fox [aut] (<https://orcid.org/0000-0002-1196-8012>), Georges Monette [aut, cre]

Maintainer Georges Monette <georges+cv@yorku.ca>

Repository CRAN

Date/Publication 2024-01-27 18:30:02 UTC

R topics documented:

---

cv *Cross-Validate Regression Models*

Description

A parallelized generic k-fold (including n-fold, i.e., leave-one-out) cross-validation function, with a default method, and specific methods for linear and generalized-linear models that can be much more computationally efficient.

Usage

```
cv(model, data, criterion, k, reps = 1, seed, ...)

## Default S3 method:
cv(
  model,
  data = insight::get_data(model),
  criterion = mse,
  k = 10,
  reps = 1,
  seed,
  confint = n >= 400,
  level = 0.95,
  ncores = 1,
  type = "response",
  ...
)

## S3 method for class 'cv'
print(x, digits = getOption("digits"), ...)

## S3 method for class 'cvList'
```

```
print(x, ...)

## S3 method for class 'lm'
cv(
  model,
  data = insight::get_data(model),
  criterion = mse,
  k = 10,
  reps = 1,
  seed,
  confint = n >= 400,
  level = 0.95,
  method = c("auto", "hatvalues", "Woodbury", "naive"),
  ncores = 1,
  ...
)

## S3 method for class 'glm'
cv(
  model,
  data = insight::get_data(model),
  criterion = mse,
  k = 10,
  reps = 1,
  seed,
  confint = n >= 400,
  level = 0.95,
  method = c("exact", "hatvalues", "Woodbury"),
  ncores = 1,
  ...
)

## S3 method for class 'rlm'
cv(model, data, criterion, k, reps = 1, seed, ...)
```

Arguments

model	a regression model object (see Details).
data	data frame to which the model was fit (not usually necessary).
criterion	cross-validation criterion ("cost" or lack-of-fit) function of form f(y, yhat) where y is the observed values of the response and yhat the predicted values; the default is mse (the mean-squared error).
k	perform k-fold cross-validation (default is 10); k may be a number or "loo" or "n" for n-fold (leave-one-out) cross-validation.
reps	number of times to replicate k-fold CV (default is 1)
seed	for R's random number generator; optional, if not supplied a random seed will be selected and saved; not needed for n-fold cross-validation

...	to match generic; passed to predict() for the default method.
confint	if TRUE (the default if the number of cases is 400 or greater), compute a confidence interval for the bias-corrected CV criterion, if the criterion is the average of casewise components.
level	confidence level (default 0.95).
ncores	number of cores to use for parallel computations (default is 1, i.e., computations aren't done in parallel)
type	for the default method, value to be passed to the type argument of predict(); the default is type="response", which is appropriate, e.g., for a "glm" model and may be recognized or ignored by predict() methods for other model classes.
x	a "cv" or "cvList" object to be printed
digits	significant digits for printing, default taken from the "digits" option
method	computational method to apply to a linear (i.e. "lm") model or to a generalized linear (i.e., "glm") model. See Details for an explanation of the available options.

Details

The default cv() method uses update() to refit the model to each fold, and should work if there are appropriate update() and predict() methods, and if the default method for GetResponse() works or if a GetResponse() method is supplied.

The "lm" and "glm" methods can use much faster computational algorithms, as selected by the method argument. The linear-model method accommodates weighted linear models.

For both classes of models, for the leave-one-out (n-fold) case, fitted values for the folds can be computed from the hat-values via method="hatvalues" without refitting the model; for GLMs, this method is approximate, for LMs it is exact.

Again for both classes of models, when more than one case is omitted in each fold, fitted values may be obtained without refitting the model by exploiting the Woodbury matrix identity via method="Woodbury". As for hatvalues, this method is exact for LMs and approximate for GLMs.

The default for linear models is method="auto", which is equivalent to method="hatvalues" for n-fold cross-validation and method="Woodbury" otherwise; method="naive" refits the model via update() and is generally much slower. The default for generalized linear models is method="exact", which employs update().

There is also a method for robust linear models fit by rlm() in the **MASS** package (to avoid inheriting the "lm" method for which the default "auto" computational method would be inappropriate).

For additional details, see the "Cross-validation of regression models" vignette (vignette("cv", package="cv")).

cv() is designed to be extensible to other classes of regression models; see the "Extending the cv package" vignette (vignette("cv-extend", package="cv")).

Value

The cv() methods return an object of class "cv", with the CV criterion ("CV crit"), the bias-adjusted CV criterion ("adj CV crit"), the criterion for the model applied to the full data ("full

crit"), the confidence interval and level for the bias-adjusted CV criterion ("confint"), the number of folds ("k"), and the seed for R's random-number generator ("seed"). Some methods may return a subset of these components and may add additional information. If reps > 1, then an object of class "cvList" is returned, which is literally a list of "cv" objects.

Methods (by class)

- cv(default): default method
- cv(lm): "lm" method
- cv(glm): "glm" method
- cv(rlm): "rlm" method (to avoid inheriting the "lm" method)

Methods (by generic)

- print(cv): print() method

Functions

- print(cvList): print() method

See Also

cvMixed, cvSelect.

Examples

```
data("Auto", package="ISLR2")
m.auto <- lm(mpg ~ horsepower, data=Auto)
cv(m.auto,  k="loo")
cv(m.auto, seed=1234)
cv(m.auto, seed=1234, reps=3)

data("Mroz", package="carData")
m.mroz <- glm(lfp ~ ., data=Mroz, family=binomial)
cv(m.mroz, criterion=BayesRule, seed=123)

data("Duncan", package="carData")
m.lm <- lm(prestige ~ income + education, data=Duncan)
m.rlm <- MASS::rlm(prestige ~ income + education,
                   data=Duncan)
cv(m.lm, k="loo", method="Woodbury")
cv(m.rlm, k="loo")
```

cvMixed *Cross-Validate Mixed-Effects Model*

Description

`cv()` methods for models of class `"merMod"`, fit by the `lmer()` and `glmer()` functions in the **lme4** package; for models of class `"lme"` fit by the `lme()` function in the **nlme** package; and for models of class `"glmmTMB"` fit by the `glmmTMB()` function in the **glmmTMB** package. The `cvMixed()` function is meant to be called by `cv()` methods for mixed-effect models and not directly by the user. It can be used to extend `cv()` to other classes of mixed-effects models.

Usage

```
cvMixed(
  model,
  package,
  data = insight::get_data(model),
  criterion = mse,
  k,
  reps = 1,
  confint,
  level = 0.95,
  seed,
  ncores = 1,
  clusterVariables,
  predict.clusters.args = list(object = model, newdata = data),
  predict.cases.args = list(object = model, newdata = data),
  ...
)

## S3 method for class 'merMod'
cv(
  model,
  data = insight::get_data(model),
  criterion = mse,
  k,
  reps = 1,
  seed,
  ncores = 1,
  clusterVariables,
  ...
)

## S3 method for class 'lme'
cv(
  model,
  data = insight::get_data(model),
```

```
  criterion = mse,
  k,
  reps = 1,
  seed,
  ncores = 1,
  clusterVariables,
  ...
)

## S3 method for class 'glmmTMB'
cv(
  model,
  data = insight::get_data(model),
  criterion = mse,
  k,
  reps = 1,
  seed,
  ncores = 1,
  clusterVariables,
  ...
)
```

Arguments

model	a mixed-effects model object for which a cv() method is available.
package	the name of the package in which mixed-modeling function (or functions) employed resides; used to get the namespace of the package.
data	data frame to which the model was fit (not usually necessary)
criterion	cross-validation ("cost" or lack-of-fit) criterion function of form f(y, yhat) where y is the observed values of the response and yhat the predicted values; the default is mse (the mean-squared error)
k	perform k-fold cross-validation; k may be a number or "loo" or "n" for n-fold (leave-one-out) cross-validation; the default is 10 if cross-validating individual cases and "loo" if cross-validating clusters.
reps	number of times to replicate k-fold CV (default is 1),
confint	if TRUE (the default if the number of cases is 400 or greater), compute a confidence interval for the bias-corrected CV criterion, if the criterion is the average of casewise components.
level	confidence level (default 0.95).
seed	for R's random number generator; optional, if not supplied a random seed will be selected and saved; not needed for n-fold cross-validation
ncores	number of cores to use for parallel computations (default is 1, i.e., computations aren't done in parallel)
clusterVariables	a character vector of names of the variables defining clusters for a mixed model with nested or crossed random effects; if missing, cross-validation is performed for individual cases rather than for clusters

`predict.clusters.args`
: a list of arguments to be used to predict the whole data set from a mixed model when performing CV on clusters; the first two elements should be `model` and `newdata`; see the "Extending the cv package" vignette (`vignette("cv-extend", package="cv")`).

`predict.cases.args`
: a list of arguments to be used to predict the whole data set from a mixed model when performing CV on cases; the first two elements should be `model` and `newdata`; see the "Extending the cv package" vignette (`vignette("cv-extend", package="cv")`).

`...`
: for `cv()` methods, to match generic, and for `cvMixed()`, arguments to be passed to `update()`.

Details

For mixed-effects models, cross-validation can be done by "clusters" or by individual observations. If the former, predictions are based only on fixed effects; if the latter, predictions include the random effects (i.e., are the best linear unbiased predictors or "BLUPS").

Value

`cvMixed()`, and functions based on it, such as the methods `cv.merMod()`, `cv.lme()`, and `cv.glmmTMB()`, return objects of class `"cv"`, or, if `reps` > 1, of class `"cvList"` (see `cv()`).

Functions

- `cvMixed()`: not to be called directly
- `cv(merMod)`: `cv()` method
- `cv(lme)`: `cv()` method
- `cv(glmmTMB)`: `cv()` method

See Also

`cv`, `lmer`, `glmer`, `lme`, `glmmTMB`

Examples

```
library("lme4")
# from ?lmer:
(fm1 <- lmer(Reaction ~ Days + (Days | Subject), sleepstudy))
cv(fm1, clusterVariables="Subject") # LOO CV of clusters
cv(fm1, seed=447) # 10-fold CV of cases
cv(fm1, clusterVariables="Subject", k=5,
   seed=834, reps=3) # 5-fold CV of clusters, repeated 3 times

library(nlme)
# from ?lme
(fm2 <- lme(distance ~ age + Sex, data = Orthodont,
            random = ~ 1))
```

```
cv(fm2) # LOO CV of cases
cv(fm2, clusterVariables="Subject", k=5, seed=321) # 5-fold CV of clusters

library("glmmTMB")
# from ?glmmTMB
(m1 <- glmmTMB(count ~ mined + (1|site),
               zi=~mined,
               family=poisson, data=Salamanders))
cv(m1, seed=97816, k=5, clusterVariables="site") # 5-fold CV of clusters
cv(m1, seed=34506, k=5) # 5-fold CV of cases
```

`cvSelect` *Cross-Validate a Model-Selection Procedure*

Description

`cvSelect()` is a general function to cross-validate a model-selection procedure; `selectStepAIC()` is a procedure that applies the `stepAIC()` model-selection function in the **MASS** package; `selectTrans()` is a procedure for selecting predictor and response transformations in regression, which uses the `powerTransform()` function in the **car** package; and `selectTransAndStepAIC()` combines predictor and response transformation with predictor selection.

Usage

```
cvSelect(
  procedure,
  data,
  criterion = mse,
  model,
  y.expression,
  k = 10,
  confint = n >= 400,
  level = 0.95,
  reps = 1,
  save.coef = k <= 10,
  seed,
  ncores = 1,
  ...
)

selectStepAIC(
  data,
  indices,
  model,
  criterion = mse,
  AIC = TRUE,
  save.coef = TRUE,
  ...
```

```
)

selectTrans(
  data,
  indices,
  save.coef = TRUE,
  model,
  criterion = mse,
  predictors,
  response,
  family = c("bcPower", "bcnPower", "yjPower", "basicPower"),
  family.y = c("bcPower", "bcnPower", "yjPower", "basicPower"),
  rounded = TRUE,
  ...
)

selectTransStepAIC(
  data,
  indices,
  save.coef = TRUE,
  model,
  criterion = mse,
  predictors,
  response,
  family = c("bcPower", "bcnPower", "yjPower", "basicPower"),
  family.y = c("bcPower", "bcnPower", "yjPower", "basicPower"),
  rounded = TRUE,
  AIC = TRUE,
  ...
)

compareFolds(object, digits = 3, ...)

## S3 method for class 'cvSelect'
coef(object, average, NAs = 0, ...)
```

Arguments

procedure	a model-selection procedure function (see Details).
data	full data frame for model selection.
criterion	a CV criterion ("cost" or lack-of-fit) function.
model	a regression model object fit to data.
y.expression	normally the response variable is found from the model argument; but if, for a particular selection procedure, the model argument is absent, or if the response can't be inferred from the model, the response can be specified by an expression, such as expression(log(income)), to be evaluated within the data set provided by the data argument.

k	perform k-fold cross-validation (default is 10); k may be a number or "loo" or "n" for n-fold (leave-one-out) cross-validation.
confint	if TRUE (the default if the number of cases is 400 or greater), compute a confidence interval for the bias-corrected CV criterion, if the criterion is the average of casewise components.
level	confidence level (default 0.95).
reps	number of times to replicate k-fold CV (default is 1)
save.coef	save the coefficients from the selected models? (default is TRUE if k is 10 or smaller, FALSE otherwise)
seed	for R's random number generator; not used for n-fold cross-validation.
ncores	number of cores to use for parallel computations (default is 1, i.e., computations aren't done in parallel)
...	for cvSelect(), arguments to be passed to procedure(); for selectStepAIC(), arguments to be passed to stepAIC().
indices	indices of cases in data defining the current fold.
AIC	if TRUE (the default) use the AIC as the model-selection criterion; if FALSE, use the BIC. The k argument to stepAIC() is set accordingly (note that this is distinct from the number of folds k).
predictors	character vector of names of the predictors in the model to transform; if missing, no predictors will be transformed.
response	name of the response variable; if missing, the response won't be transformed.
family	transformation family for the predictors, one of "bcPower", "bcnPower", "yjPower", "basicPower", with "bcPower" as the default. These are the names of transformation functions in the **car** package; see bcPower()
family.y	transformation family for the response, with "bcPower" as the default.
rounded	if TRUE (the default) use nicely rounded versions of the estimated transformation parameters (see bcPower()).
object	an object of class "cvSelect".
digits	significant digits for printing coefficients (default 3).
average	if supplied, a function, such as mean or median, to use us in averaging estimates across folds; if missing, the estimates for each fold are returned.
NAs	values to substitute for NAs in calculating averaged estimates; the default, 0, is appropriate, e.g., for regression coefficients; the value 1 might be appropriate for power-transformation estimates.

Details

The model-selection function supplied as the procedure argument to cvSelect() should accept the following arguments:

data set to the data argument to cvSelect().

indices the indices of the rows of data defining the current fold; if missing, the model-selection procedure is applied to the full data.

other arguments to be passed via ... from cvSelect().

procedure() should return a list with the following named elements: fit.i, the vector of predicted values for the cases in the current fold computed from the model omitting these cases; crit.all.i, the CV criterion computed for all of the cases using the model omitting the current fold; and (optionally) coefficients, parameter estimates from the model computed omitting the current fold.

When the indices argument is missing, procedure() returns the cross-validation criterion for all of the cases based on the model fit to all of the cases.

For examples of model-selection functions for the procedure argument, see the code for selectStepAIC(), selectTrans(), and selectTransAndStepAIC().

For additional information, see the "Cross-validation of regression models" vignette (vignette("cv", package="cv")) and the "Extending the cv package" vignette (vignette("cv-extend", package="cv")).

Value

An object of class "cvSelect", inheriting from class "cv", with the CV criterion ("CV crit"), the bias-adjusted CV criterion ("adj CV crit"), the criterion for the model applied to the full data ("full crit"), the confidence interval and level for the bias-adjusted CV criterion ("confint"), the number of folds ("k"), the seed for R's random-number generator ("seed"), and (optionally) a list of coefficients (or, in the case of selectTrans(), estimated transformation parameters, and in the case of selectTransAndStepAIC(), both regression coefficients and transformation parameters) for the selected models for each fold ("coefficients"). If reps > 1, then an object of class c("cvSelectList", "cvList") is returned, which is literally a list of c("cvSelect", "cv") objects.

Methods (by generic)

- coef(cvSelect): extract the coefficients from the selected models for the several folds and possibly average them.

Functions

- cvSelect(): apply cross-validation to a model-selection procedure.
- selectStepAIC(): select a model using the stepAIC() function in the **MASS** package.
- selectTrans(): select transformations of the predictors and response.
- selectTransStepAIC(): select transformations of the predictors and response, and then select predictors.
- compareFolds(): print the coefficients from the selected models for the several folds.

See Also

stepAIC, bcPower, powerTransform, cv.

Examples

```
data("Auto", package="ISLR2")
m.auto <- lm(mpg ~ . - name - origin, data=Auto)
cvSelect(selectStepAIC, Auto, seed=123, model=m.auto)
cvSelect(selectStepAIC, Auto, seed=123, model=m.auto,
         AIC=FALSE, k=5, reps=3) # via BIC

data("Prestige", package="carData")
m.pres <- lm(prestige ~ income + education + women,
             data=Prestige)
cvt <- cvSelect(selectTrans, data=Prestige, model=m.pres, seed=123,
                predictors=c("income", "education", "women"),
                response="prestige", family="yjPower")
cvt
compareFolds(cvt)
coef(cvt, average=median, NAs=1) # NAs not really needed here
cv(m.pres, seed=123)

Auto$year <- as.factor(Auto$year)
Auto$origin <- factor(Auto$origin,
                      labels=c("America", "Europe", "Japan"))
rownames(Auto) <- make.names(Auto$name, unique=TRUE)
Auto$name <- NULL
m.auto <- lm(mpg ~ . , data=Auto)
cvs <- cvSelect(selectTransStepAIC, data=Auto, seed=76692, model=m.auto,
                criterion=medAbsErr,
                predictors=c("cylinders", "displacement", "horsepower",
                             "weight", "acceleration"),
                response="mpg", AIC=FALSE)
cvs
compareFolds(cvs)
```

GetResponse *Extract Response Variable*

Description

Generic function to extract the response variable from a fitted model.

Usage

```
GetResponse(model, ...)

## Default S3 method:
GetResponse(model, ...)

## S3 method for class 'merMod'
GetResponse(model, ...)
```

```
## S3 method for class 'lme'
GetResponse(model, ...)

## S3 method for class 'glmmTMB'
GetResponse(model, ...)
```

Arguments

model	a fitted model
...	additional parameters for specific methods

Details

The supplied default method returns the model$y component of the model object, or, if model is an S4 object, the result returned by the get_response() function in the **insight** package. If this result is NULL, the result of model.response(model.frame(model)) is returned, checking in any case whether the result is a numeric vector.

There is also an "lme" method, and "merMod" and "glmmTMB" methods that convert factor responses to numeric 0/1 responses, as would be appropriate for a generalized linear mixed model with a binary response.

Value

a numeric vector containing the values of the response variable.

Methods (by class)

- GetResponse(default): default method
- GetResponse(merMod): merMod method
- GetResponse(lme): merMod method
- GetResponse(glmmTMB): glmmTMB method

Examples

```
fit <- lm(mpg ~ gear, mtcars)
GetResponse(fit)
```

---

models *Cross-Validate Several Models Fit to the Same Data*

Description

A cv() method for an object of class "modlist", created by the models() function. This cv() method simplifies the process of cross-validating several models on the same set of CV folds. models() performs some "sanity" checks, warning if the models are of different classes, and reporting an error if they are fit to apparently different data sets or different response variables.

Usage

```
models(...)

## S3 method for class 'modList'
cv(model, data, criterion = mse, k, reps = 1, seed, quietly = TRUE, ...)

## S3 method for class 'cvModList'
print(x, ...)

## S3 method for class 'cvModList'
plot(
  x,
  y,
  spread = c("range", "sd"),
  confint = TRUE,
  xlab = "",
  ylab,
  main,
  axis.args = list(labels = names(x), las = 3L),
  col = palette()[2L],
  lwd = 2,
  grid = TRUE,
  ...
)
```

Arguments

`...`	for `models()`, two or more competing models fit to the the same data; the several models may be named. For `cv()`, additional arguments to be passed to the `cv()` method applied to each model. For the `print()` method, arguments to be passed to the `print()` method for the individual model cross-validations. For the `plot()`, method, arguments to be passed to the base `plot()` function.
`model`	a list of regression model objects, created by `models()`.
`data`	(required) the data set to which the models were fit.
`criterion`	the CV criterion ("cost" or lack-of-fit) function, defaults to `mse`.
`k`	the number of CV folds; may be omitted, in which case the value will depend on the default for the `cv()` method invoked for the individual models.
`reps`	number of replications of CV for each model.
`seed`	(optional) seed for R's pseudo-random-number generator, to be used to create the same set of CV folds for all of the models; if omitted, a seed will be randomly generated and saved.
`quietly`	if `TRUE` (the default), simple messages (for example about the value to which the random-number generator seed is set), but not warnings or errors, are suppressed.
`x`	an object of class `"cvModList"` to be printed or plotted.

y	the name of the element in each "cv" object to be plotted; defaults to "adj CV crit", if it exists, or to "CV crit".
spread	if "range", the default, show the range of CV criteria for each model along with their average; if "sd", show the average plus or minus 1 standard deviation.
confint	if TRUE (the default) and if confidence intervals are in any of the "cv" objects, then plot the confidence intervals around the CV criteria.
xlab	label for the x-axis (defaults to blank).
ylab	label for the y-axis (if missing, a label is constructed).
main	main title for the graph (if missing, a label is constructed).
axis.args	a list of arguments for the axis() function, used to draw the horizontal axis. In addition to the axis arguments given explicitly, side=1 (the horizontal axis) and at=seq(along=x) (i.e., 1 to the number of models) are used and can't be modified.
col	color for the line and points, defaults to the second element of the color palette; see palette().
lwd	line width for the line (defaults to 2).
grid	if TRUE (the default), include grid lines on the graph.

Value

models() returns a "modList" object, the cv() method for which returns a "cvModList" object.

Functions

- models(): create a list of models
- cv(modList): cv() method for "modList" objects
- print(cvModList): print() method for "cvModList" objects
- plot(cvModList): plot() method for "cvModList" objects

See Also

cv, cvMixed.

Examples

```
data("Duncan", package="carData")
m1 <- lm(prestige ~ income + education, data=Duncan)
m2 <- lm(prestige ~ income + education + type, data=Duncan)
m3 <- lm(prestige ~ (income + education)*type, data=Duncan)
(cv.models <- cv(models(m1=m1, m2=m2, m3=m3),
                 data=Duncan, seed=7949, reps=5))
plot(cv.models)
(cv.models.ci <- cv(models(m1=m1, m2=m2, m3=m3),
                    data=Duncan, seed=5962, confint=TRUE, level=0.50))
                 # nb: n too small for accurate CIs
plot(cv.models.ci)
```

mse *Cost Functions for Fitted Regression Models*

Description

Compute cost functions (cross-validation criteria) for fitted regression models.

Usage

```
mse(y, yhat)

rmse(y, yhat)

medAbsErr(y, yhat)

BayesRule(y, yhat)

BayesRule2(y, yhat)
```

Arguments

y	response
yhat	fitted value

Details

Cost functions (cross-validation criteria) are meant to measure lack-of-fit. Several cost functions are provided:

1. `mse()` returns the mean-squared error of prediction for a numeric response variable `y` and predictions `yhat`; and `rmse()` returns the root-mean-squared error and is just the square-root of `mse()`.
2. `medAbsErr()` returns the median absolute error of prediction for a numeric response `y` and predictions `yhat`.
3. `BayesRule()` and `BayesRule2()` report the proportion of incorrect predictions for a dichotomous response variable `y`, assumed coded (or coercible to) `0` and `1`. The `yhat` values are predicted probabilities and are rounded to 0 or 1. The distinction between `BayesRule()` and `BayesRule2()` is that the former checks that the `y` values are all either `0` or `1` and that the `yhat` values are all between 0 and 1, while the latter doesn't and is therefore faster.

Value

In general, cost functions should return a single numeric value measuring lack-of-fit. `mse()` returns the mean-squared error; `rmse()` returns the root-mean-squared error; `medAbsErr()` returns the median absolute error; and BayesRule()andBayesRule2()‘ return the proportion of misclassified cases.

Functions

- `mse()`: Mean-square error
- `rmse()`: Root-mean-square error
- `medAbsErr()`: Median absolute error
- `BayesRule()`: Bayes Rule for a binary response
- `BayesRule2()`: Bayes rule for a binary response (without bounds checking)

See Also

`cv`, `cvSelect`

Examples

```
data("Duncan", package="carData")
m.lm <- lm(prestige ~ income + education, data=Duncan)
mse(Duncan$prestige, fitted(m.lm))

data("Mroz", package="carData")
m.glm <- glm(lfp ~ ., data=Mroz, family=binomial)
BayesRule(Mroz$lfp == "yes", fitted(m.glm))
```

Pigs *Body Weights of 48 Pigs in 9 Successive Weeks*

Description

This data set appears in Table 3.1 of Diggle, Liang, and Zeger (1994).

Usage

```
data("Pigs", package = "cv")
```

Format

A data frame with 432 rows and 3 columns.

id Pig id number, 1–48.

week Week number, 1–9.

weight Weight in kg.

Source

P. J. Diggle, K.-Y. Liang, and S. L. Zeger, *Analysis of Longitudinal Data* (Oxford, 1994).

Examples

```
library("lme4")
m.p <- lmer(weight ~ week + (1 | id) + (1 | week),
            data=Pigs, REML=FALSE,
            control=lmerControl(optimizer="bobyqa"))
summary(m.p)
cv(m.p, clusterVariables=c("id", "week"), k=10, seed=8469)
```

Index