Package 'projpred'

October 28, 2020

Encoding UTF-8

Title Projection Predictive Feature Selection

Version 2.0.2

Maintainer Alejandro Catalina <alecatfel@gmail.com>

Description Performs projection predictive feature selection for generalized linear models and generalized linear and additive multilevel models (see, Piironen, Paasiniemi and Vehtari, 2020, <https://projecteuclid.org/euclid.ejs/1589335310>, Catalina, Bürkner and Vehtari, 2020, <arXiv:2010.06994>). The package is compatible with the 'rstanarm' and 'brms' packages, but other reference models can also be used. See the package vignette for more information and examples.

Depends R (>= 3.5.0)

Imports methods, dplyr, loo (>= 2.0.0), rstantools (>= 2.0.0), lme4, optimx, ggplot2, Rcpp, utils, rngtools (>= 1.2.4), tidyverse, MASS, magrittr, mgcv, gamm4

LinkingTo Rcpp, RcppArmadillo

License GPL-3

LazyData TRUE

RoxygenNote 7.1.1

Suggests rstanarm, brms, testthat, knitr, rmarkdown, glmnet, bayesplot (>= 1.5.0)

VignetteBuilder knitr

URL https://mc-stan.org/projpred/, https://discourse.mc-stan.org

BugReports https://github.com/stan-dev/projpred/issues/

NeedsCompilation yes

Author Juho Piironen [aut],
Markus Paasiniemi [aut],
Alejandro Catalina [cre, aut],
Aki Vehtari [aut],
Jonah Gabry [ctb],
Marco Colombo [ctb],
Paul-Christian Bürkner [ctb]

Repository CRAN

Date/Publication 2020-10-28 12:50:02 UTC

R topics documented:

`break_up_matrix_term` *Sometimes there can be terms in a formula that refer to a matrix instead of a single predictor. Because we can handle search_terms of predictors, this function breaks the matrix term into individual predictors to handle separately, as that is probably the intention of the user.*

Description

Sometimes there can be terms in a formula that refer to a matrix instead of a single predictor. Because we can handle search_terms of predictors, this function breaks the matrix term into individual predictors to handle separately, as that is probably the intention of the user.

Usage

```
break_up_matrix_term(formula, data)
```

Arguments

formula	A formula for a valid model.
data	The original data frame with a matrix as predictor.

Value

a list containing the expanded formula and the expanded data frame.

---

cv-indices *Create cross-validation indices*

Description

Divide indices from 1 to n into subsets for k-fold cross validation. These functions are potentially useful when creating the cvfits and cvfun arguments for init_refmodel. The returned value is different for these two methods, see below for details.

Usage

```
cvfolds(n, K, seed = NULL)

cv_ids(n, K, out = c("foldwise", "indices"), seed = NULL)
```

Arguments

n	Number of data points.
K	Number of folds. Must be at least 2 and not exceed n.
seed	Random seed so that the same division could be obtained again if needed.
out	Format of the output, either 'foldwise' (default) or 'indices'. See below for details.

Value

cvfolds returns a vector of length n such that each element is an integer between 1 and k denoting which fold the corresponding data point belongs to. The returned value of cv_ids depends on the out-argument. If out='foldwise', the returned value is a list with k elements, each having fields tr and ts which give the training and test indices, respectively, for the corresponding fold. If out='indices', the returned value is a list with fields tr and ts each of which is a list with k elements giving the training and test indices for each fold.

Examples

```
### compute sample means within each fold
n <- 100
y <- rnorm(n)
cv <- cv_ids(n, K=5)
cvmeans <- lapply(cv, function(fold) mean(y[fold$tr]))
```

---

cv_varsel *Cross-validated variable selection (varsel)*

Description

Perform cross-validation for the projective variable selection for a generalized linear model or generalized lienar and additive multilevel models.

Usage

```
cv_varsel(object, ...)

## Default S3 method:
cv_varsel(object, ...)

## S3 method for class 'refmodel'
cv_varsel(
  object,
  method = NULL,
  cv_method = NULL,
  ndraws = NULL,
  nclusters = NULL,
  ndraws_pred = NULL,
  nclusters_pred = NULL,
  cv_search = TRUE,
  nterms_max = NULL,
  intercept = NULL,
  penalty = NULL,
  verbose = TRUE,
  nloo = NULL,
  K = NULL,
  lambda_min_ratio = 1e-05,
  nlambda = 150,
  thresh = 1e-06,
  regul = 1e-04,
  validate_search = TRUE,
  seed = NULL,
```

```
  search_terms = NULL,
  ...
)
```

Arguments

object	Same as in varsel.
...	Additional arguments to be passed to the get_refmodel-function.
method	Same as in varsel.
cv_method	The cross-validation method, either 'LOO' or 'kfold'. Default is 'LOO'.
ndraws	Number of posterior draws used for selection. Ignored if nclusters is provided or if method='L1'.
nclusters	Number of clusters used for selection. Default is 1 and ignored if method='L1' (L1-search uses always one cluster).
ndraws_pred	Number of samples used for prediction (after selection). Ignored if nclusters_pred is given.
nclusters_pred	Number of clusters used for prediction (after selection). Default is 5.
cv_search	Same as in varsel.
nterms_max	Same as in varsel.
intercept	Same as in varsel.
penalty	Same as in varsel.
verbose	Whether to print out some information during the validation, Default is TRUE.
nloo	Number of observations used to compute the LOO validation (anything between 1 and the total number of observations). Smaller values lead to faster computation but higher uncertainty (larger errorbars) in the accuracy estimation. Default is to use all observations, but for faster experimentation, one can set this to a small value such as 100. Only applicable if cv_method = 'LOO'.
K	Number of folds in the K-fold cross validation. Default is 5 for genuine reference models and 10 for datafits (that is, for penalized maximum likelihood estimation).
lambda_min_ratio	Same as in varsel.
nlambda	Same as in varsel.
thresh	Same as in varsel.
regul	Amount of regularization in the projection. Usually there is no need for regularization, but sometimes for some models the projection can be ill-behaved and we need to add some regularization to avoid numerical problems.
validate_search	Whether to cross-validate also the selection process, that is, whether to perform selection separately for each fold. Default is TRUE and we strongly recommend not setting this to FALSE, because this is known to bias the accuracy estimates for the selected submodels. However, setting this to FALSE can sometimes be useful because comparing the results to the case where this parameter is TRUE

	gives idea how strongly the feature selection is (over)fitted to the data (the difference corresponds to the search degrees of freedom or the effective number of parameters introduced by the selectin process).
seed	Random seed used in the subsampling LOO. By default uses a fixed seed.
search_terms	User defined list of terms to consider for selection.

Value

An object of type vsel that contains information about the feature selection. The fields are not meant to be accessed directly by the user but instead via the helper functions (see the vignettes or type ?projpred to see the main functions in the package.)

Examples

```
if (requireNamespace('rstanarm', quietly=TRUE)) {
  ### Usage with stanreg objects
  n <- 30
  d <- 5
  x <- matrix(rnorm(n*d), nrow=n)
  y <- x[,1] + 0.5*rnorm(n)
  data <- data.frame(x,y)
  fit <- rstanarm::stan_glm(y ~ X1 + X2 + X3 + X4 + X5, gaussian(),
     data=data, chains=2, iter=500)
  cvs <- cv_varsel(fit)
  plot(cvs)
}
```

---

df_binom *Binomial toy example.*

Description

Binomial toy example.

Usage

```
df_binom
```

Format

A simulated classification dataset containing 100 observations.

y target, 0 or 1.

x features, 30 in total.

Source

http://web.stanford.edu/~hastie/glmnet/glmnetData/BNExample.RData

---

df_gaussian *Gaussian toy example.*

Description

Gaussian toy example.

Usage

```
df_gaussian
```

Format

A simulated regression dataset containing 100 observations.

y target, real-valued.

x features, 20 in total. Mean and sd approximately 0 and 1.

Source

http://web.stanford.edu/~hastie/glmnet/glmnetData/QSExample.RData

---

extend_family *Add extra fields to the family object.*

Description

Add extra fields to the family object.

Usage

```
extend_family(family)
```

Arguments

family	Family object.

Value

Extended family object.

extra-families *Extra family objects.*

Description

Family objects not in the set of default family-objects.

Usage

```
Student_t(link = "identity", nu = 3)
```

Arguments

link	Specification of the link function, as for the default family-objects.
nu	Degrees of freedom for the Student-t distribution.

Value

A family object analogous to those described in family

get-refmodel *Get reference model structure*

Description

Generic function that can be used to create and fetch the reference model structure for all those objects that have this method. All these implementations are wrappers to the init_refmodel-function so the returned object has the same type.

Usage

```
get_refmodel(object, ...)

## S3 method for class 'refmodel'
get_refmodel(object, ...)

## S3 method for class 'vsel'
get_refmodel(object, ...)

## Default S3 method:
get_refmodel(
  object,
  data,
  y,
  formula,
```

```
  ref_predfun,
  proj_predfun,
  div_minimizer,
  fetch_data,
  family = NULL,
  wobs = NULL,
  folds = NULL,
  cvfits = NULL,
  offset = NULL,
  cvfun = NULL,
  dis = NULL,
  ...
)

## S3 method for class 'stanreg'
get_refmodel(
  object,
  data = NULL,
  ref_predfun = NULL,
  proj_predfun = NULL,
  div_minimizer = NULL,
  folds = NULL,
  ...
)

init_refmodel(
  object,
  data,
  formula,
  family,
  ref_predfun = NULL,
  div_minimizer = NULL,
  proj_predfun = NULL,
  folds = NULL,
  extract_model_data = NULL,
  cvfun = NULL,
  cvfits = NULL,
  dis = NULL,
  ...
)
```

Arguments

object	Object on which the reference model is created. See possible types below.
...	Arguments passed to the methods.
data	Data on which the reference model was fitted.
y	Target response.
formula	Reference model's lme4-like formula.

ref_predfun	Prediction function for the linear predictor of the reference model.
proj_predfun	Prediction function for the linear predictor of the projections.
div_minimizer	Maximum likelihood estimator for the underlying projection.
fetch_data	Wrapper function for fetching the data without directly accessing it. It should have a prototype fetch_data(data, data_points, newdata = NULL), where data_points is a vector of data indices and newdata, if not NULL, is a data frame with new data for testing.
family	A family object that represents the observation model for the reference model.
wobs	A weights vector for the observations in the data. The default is a vector of ones.
folds	Only used for K-fold variable selection. It is a vector of fold indices for each data point in data.
cvfits	Only used for K-fold variable selection. A list of K-fold fitted objects on which reference models are created.
offset	A vector of offsets per observation to add to the linear predictor.
cvfun	Only used for K-fold variable selection. A function that, given a folds vector, fits a reference model per fold and returns the fitted object.
dis	A dispersion vector for each observation.
extract_model_data	A function with prototype extract_model_data(object, newdata, wrhs, orhs), where object is a reference model fit, newdata is either NULL or a data frame with new observations, wrhs is a right hand side formula to recover the weights from the data frame and orhs is a right hand side formula to recover the offset from the data frame.

Value

An object of type `refmodel` (the same type as returned by init_refmodel) that can be passed to all the functions that take the reference fit as the first argument, such as varsel, cv_varsel, project, proj_predict and proj_linpred.

Examples

```
if (requireNamespace('rstanarm', quietly=TRUE)) {
  ### Usage with stanreg objects
  dat <- data.frame(y = rnorm(100), x = rnorm(100))
  fit <- rstanarm::stan_glm(y ~ x, family = gaussian(), data = dat)
  ref <- get_refmodel(fit)
  print(class(ref))

  # variable selection, use the already constructed reference model
  vs <- varsel(ref)
  # this will first construct the reference model and then execute
  # exactly the same way as the previous command (the result is identical)
  vs <- varsel(fit)
}
```

helper_formula *Utilities to handle formulas for the external user*

Description

Utilities to handle formulas for the external user

mesquite *Mesquite data set.*

Description

The mesquite bushes yields data set from Gelman and Hill (2007) (http://www.stat.columbia.edu/~gelman/arm/).

Usage

```
mesquite
```

Format

The outcome variable is the total weight (in grams) of photosynthetic material as derived from actual harvesting of the bush. The predictor variables are:

diam1 diameter of the canopy (the leafy area of the bush) in meters, measured along the longer axis of the bush.

diam2 canopy diameter measured along the shorter axis

canopy height height of the canopy.

total height total height of the bush.

density plant unit density (# of primary stems per plant unit).

group group of measurements (0 for the first group, 1 for the second group)

Source

http://www.stat.columbia.edu/~gelman/arm/examples/

`plot.vsel` *Plot summary statistics related to variable selection*

Description

Plot summary statistics related to variable selection

Usage

```
## S3 method for class 'vsel'
plot(
  x,
  nterms_max = NULL,
  stats = "elpd",
  deltas = FALSE,
  alpha = 0.32,
  baseline = NULL,
  ...
)
```

Arguments

`x`	The object returned by varsel or cv_varsel.
`nterms_max`	Maximum submodel size for which the statistics are calculated. For `plot.vsel` it must be at least 1.
`stats`	One or several strings determining which statistics to calculate. Available statistics are: • elpd: (Expected) sum of log predictive densities • mlpd: Mean log predictive density, that is, elpd divided by the number of datapoints. • mse: Mean squared error (gaussian family only) • rmse: Root mean squared error (gaussian family only) • acc/pctcorr: Classification accuracy (binomial family only) • auc: Area under the ROC curve (binomial family only) Default is `"elpd"`.
`deltas`	If `TRUE`, the submodel statistics are estimated relative to the baseline model (see argument `baseline`) instead of estimating the actual values of the statistics. Defaults to `FALSE`.
`alpha`	A number indicating the desired coverage of the credible intervals. For example `alpha=0.32` corresponds to 68% probability mass within the intervals, that is, one standard error intervals.
`baseline`	Either 'ref' or 'best' indicating whether the baseline is the reference model or the best submodel found. Default is 'ref' when the reference model exists, and 'best' otherwise.
`...`	Currently ignored.

Examples

```
### Usage with stanreg objects
if (requireNamespace('rstanarm', quietly=TRUE)) {
  n <- 30
  d <- 5
  x <- matrix(rnorm(n*d), nrow=n)
  y <- x[,1] + 0.5*rnorm(n)
  data <- data.frame(x,y)

  fit <- rstanarm::stan_glm(y ~ X1 + X2 + X3 + X4 + X5, gaussian(),
    data=data, chains=2, iter=500)
  vs <- cv_varsel(fit)
  plot(vs)
}
```

`predict.refmodel` *Predict method for reference model objects*

Description

Compute the predictions using the reference model, that is, compute the expected value for the next observation, or evaluate the log-predictive density at a given point.

Usage

```
## S3 method for class 'refmodel'
predict(
  object,
  newdata,
  ynew = NULL,
  offsetnew = NULL,
  weightsnew = NULL,
  type = "response",
  ...
)
```

Arguments

`object`	The object of class `refmodel`.
`newdata`	Matrix of predictor values used in the prediction.
`ynew`	New (test) target variables. If given, then the log predictive density for the new observations is computed.
`offsetnew`	Offsets for the new observations. By default a vector of zeros. By default we take the weights from newdata as in the original model. Either NULL or right hand side formulas.

weightsnew	Weights for the new observations. For binomial model, corresponds to the number trials per observation. Has effect only if ynew is specified. By default a vector of ones. By default we take the weights from newdata as in the original model. Either NULL or right hand side formulas.
type	Scale on which the predictions are returned. Either 'link' (the latent function value, from -inf to inf) or 'response' (the scale on which the target y is measured, obtained by taking the inverse-link from the latent value).
...	Currently ignored.

Value

Returns either a vector of predictions, or vector of log predictive densities evaluated at ynew if ynew is not NULL.

---

print-vsel *Print methods for vsel/vsel objects*

Description

The print methods for vsel/vsel objects created by varsel or cv_varsel) rely on summary.vsel to display a summary of the results of the projection predictive variable selection.

Usage

```
## S3 method for class 'vsel'
print(x, digits = 2, ...)
```

Arguments

x	An object of class vsel/vsel.
digits	Number of decimal places to be reported (2 by default).
...	Further arguments passed to summary.vsel.

Value

Returns invisibly the data frame produced by summary.vsel.

proj-pred *Extract draws of the linear predictor and draw from the predictive distribution of the projected submodel*

Description

proj_linpred extracts draws of the linear predictor and proj_predict draws from the predictive distribution of the projected submodel or submodels. If the projection has not been performed, the functions also perform the projection.

Usage

```
proj_linpred(
  object,
  newdata,
  offsetnew = NULL,
  weightsnew = NULL,
  nterms = NULL,
  transform = FALSE,
  integrated = FALSE,
  seed = NULL,
  ...
)

proj_predict(
  object,
  newdata,
  offsetnew = NULL,
  weightsnew = NULL,
  nterms = NULL,
  ndraws = 1000,
  seed = NULL,
  ...
)
```

Arguments

object	Either an object returned by varsel, cv_varsel or init_refmodel, or alternatively any object that can be converted to a reference model.
newdata	The predictor values used in the prediction. If solution_terms is specified, then newdata should either be a dataframe containing column names that correspond to solution_terms or a matrix with the number and order of columns corresponding to solution_terms. If solution_terms is unspecified, then newdata must either be a dataframe containing all the column names as in the original data or a matrix with the same columns at the same positions as in the original data.

offsetnew	Offsets for the new observations. By default a vector of zeros. By default we take the weights from newdata as in the original model. Either NULL or right hand side formula.
weightsnew	Weights for the new observations. For binomial model, corresponds to the number trials per observation. For proj_linpred, this argument matters only if newdata is specified. By default we take the weights from newdata as in the original model. Either NULL or right hand side formula.
nterms	Number of terms in the submodel (the variable combination is taken from the variable selection information). If a vector with several values, then results for all specified model sizes are returned. Ignored if solution_terms is specified. By default use the automatically suggested model size.
transform	Should the linear predictor be transformed using the inverse-link function? Default is FALSE. For proj_linpred only.
integrated	If TRUE, the output is averaged over the parameters. Default is FALSE. For proj_linpred only.
seed	An optional seed to use for drawing from the projection. For proj_predict only.
...	Additional argument passed to project if object is an object returned by varsel or cv_varsel.
ndraws	Number of draws to return from the predictive distribution of the projection. The default is 1000. For proj_predict only.

Value

If the prediction is done for one submodel only (nterms has length one or solution_terms is specified) and newdata is unspecified, a matrix or vector of predictions (depending on the value of integrated). If newdata is specified, returns a list with elements pred (predictions) and lpd (log predictive densities). If the predictions are done for several submodel sizes, returns a list with one element for each submodel.

Examples

```
if (requireNamespace('rstanarm', quietly=TRUE)) {
  ### Usage with stanreg objects
  n <- 30
  d <- 5
  x <- matrix(rnorm(n*d), nrow=n)
  y <- x[,1] + 0.5*rnorm(n)
  data <- data.frame(x,y)

 fit <- rstanarm::stan_glm(y ~ X1 + X2 + X3 + X4 + X5, gaussian(), data=data, chains=2, iter=500)
  vs <- varsel(fit)

  # compute predictions with 4 variables at the training points
  pred <- proj_linpred(vs, newdata = data, nv = 4)
  pred <- proj_predict(vs, newdata = data, nv = 4)
}
```

project *Projection to submodels*

Description

Perform projection onto submodels of selected sizes or a specified feature combination.

Usage

```
project(
  object,
  nterms = NULL,
  solution_terms = NULL,
  cv_search = TRUE,
  ndraws = 400,
  nclusters = NULL,
  intercept = NULL,
  seed = NULL,
  regul = 1e-04,
  ...
)
```

Arguments

object	Either a refmodel-type object created by get_refmodel or init_refmodel, or an object which can be converted to a reference model using get_refmodel.
nterms	Number of terms in the submodel (the variable combination is taken from the varsel information). If a list, then the projection is performed for each model size. Default is the model size suggested by the variable selection (see function suggest_size). Ignored if solution_terms is specified.
solution_terms	Variable indices onto which the projection is done. If specified, nterms is ignored.
cv_search	If TRUE, then the projected coefficients after L1-selection are computed without any penalization (or using only the regularization determined by regul). If FALSE, then the coefficients are the solution from the L1-penalized projection. This option is relevant only if L1-search was used. Default is TRUE for genuine reference models and FALSE if object is datafit (see init_refmodel).
ndraws	Number of posterior draws to be projected. Ignored if nclusters is specified. Default is 400.
nclusters	Number of clusters in the clustered projection.
intercept	Whether to use intercept. Default is TRUE.

seed	A seed used in the clustering (if nclusters!=ndraws). Can be used to ensure same results every time. @param regul Amount of regularization in the projection. Usually there is no need for regularization, but sometimes for some models the projection can be ill-behaved and we need to add some regularization to avoid numerical problems.
regul	Ridgre regularization constant to fit the projections.
...	Currently ignored.

Value

A list of submodels (or a single submodel if projection was performed onto a single variable combination), each of which contains the following elements:

kl The KL divergence from the reference model to the submodel.

weights Weights for each draw of the projected model.

dis Draws from the projected dispersion parameter.

alpha Draws from the projected intercept.

beta Draws from the projected weight vector.

solution_terms The order in which the variables were added to the submodel.

intercept Whether or not the model contains an intercept.

family A modified family-object.

Examples

```
if (requireNamespace("rstanarm", quietly = TRUE)) {
  ### Usage with stanreg objects
  n <- 30
  d <- 5
  x <- matrix(rnorm(n * d), nrow = n)
  y <- x[, 1] + 0.5 * rnorm(n)
  data <- data.frame(x, y)

  fit <- rstanarm::stan_glm(y ~ X1 + X2 + X3 + X4 + X5, gaussian(),
    data = data, chains = 2, iter = 500)
  vs <- varsel(fit)

  # project onto the best model with 4 variables
  proj4 <- project(vs, nterms = 4)

  # project onto an arbitrary variable combination (variable indices 1, 3 and 5)
  proj <- project(fit, solution_terms = c(1, 3, 5))
}
```

`projpred` *Projection predictive feature selection*

Description

Description

projpred is an R package to perform projection predictive variable (feature) selection for generalized linear models, generalized linear multilevel models and generalized additive multilevel models. The package is aimed to be compatible with **rstanarm** but also other reference models can be used (see function `init_refmodel`).

Currently, the supported models (family objects in R) include Gaussian, Binomial and Poisson families, but more will be implemented later. See the quickstart-vignette and quickstart-glmm-vignette for examples.

Functions

varsel, cv_varsel, init_refmodel, suggest_size Perform and cross-validate the variable selection. init_refmodel can be used to initialize a reference model other than **rstanarm**-fit.

project Get the projected posteriors of the reduced models.

proj_predict, proj_linpred Make predictions with reduced number of features.

plot, summary Visualize and get some key statistics about the variable selection.

References

Dupuis, J. A. and Robert, C. P. (2003). Variable selection in qualitative models via an entropic explanatory power. *Journal of Statistical Planning and Inference*, 111(1-2):77–94.

Goutis, C. and Robert, C. P. (1998). Model choice in generalised linear models: a Bayesian approach via Kullback–Leibler projections. *Biometrika*, 85(1):29–37.

Juho Piironen and Aki Vehtari (2017). Comparison of Bayesian predictive methods for model selection. *Statistics and Computing*, 27(3):711-735. doi:10.1007/s11222-016-9649-y. (Online).

`solution_terms` *Recovers solution path from a variable selection object.*

Description

Recovers solution path from a variable selection object.

Usage

```
solution_terms(object)
```

Arguments

object	A vsel object returned by varsel or cv_varsel.

Value

Variable selection solution path

---

suggest_size *Suggest model size*

Description

This function can be used for suggesting an appropriate model size based on a certain default rule. Notice that the decision rules are heuristic and should be interpreted as guidelines. It is recommended that the user studies the results via varsel_plot and/or summary and makes the final decision based on what is most appropriate for the given problem.

Usage

```
suggest_size(object, ...)

## S3 method for class 'vsel'
suggest_size(
  object,
  stat = "elpd",
  alpha = 0.32,
  pct = 0,
  type = "upper",
  baseline = NULL,
  warnings = TRUE,
  ...
)
```

Arguments

object	The object returned by varsel or cv_varsel.
...	Currently ignored.
stat	Statistic used for the decision. Default is 'elpd'. See summary for other possible choices.
alpha	A number indicating the desired coverage of the credible intervals based on which the decision is made. E.g. alpha=0.32 corresponds to 68% probability mass within the intervals (one standard error intervals). See details for more information.
pct	Number indicating the relative proportion between baseline model and null model utilities one is willing to sacrifice. See details for more information.

type	Either 'upper' (default) or 'lower' determining whether the decisions are based on the upper or lower credible bounds. See details for more information.
baseline	Either 'ref' or 'best' indicating whether the baseline is the reference model or the best submodel found. Default is 'ref' when the reference model exists, and 'best' otherwise.
warnings	Whether to give warnings if automatic suggestion fails, mainly for internal use. Default is TRUE, and usually there is no reason to set to FALSE.

Details

The suggested model size is the smallest model for which either the lower or upper (depending on argument type) credible bound of the submodel utility u_k with significance level alpha falls above

$$u_base - pct * (u_base - u_0)$$

Here u_base denotes the utility for the baseline model and u_0 the null model utility. The baseline is either the reference model or the best submodel found (see argument baseline). The lower and upper bounds are defined to contain the submodel utility with probability 1-alpha (each tail has mass alpha/2).

By default ratio=0, alpha=0.32 and type='upper' which means that we select the smallest model for which the upper tail exceeds the baseline model level, that is, which is better than the baseline model with probability 0.16 (and consequently, worse with probability 0.84). In other words, the estimated difference between the baseline model and submodel utilities is at most one standard error away from zero, so the two utilities are considered to be close.

NOTE: Loss statistics like RMSE and MSE are converted to utilities by multiplying them by -1, so call such as suggest_size(object,stat='rmse',type='upper') should be interpreted as finding the smallest model whose upper credible bound of the *negative* RMSE exceeds the cutoff level (or equivalently has the lower credible bound of RMSE below the cutoff level). This is done to make the interpretation of the argument type the same regardless of argument stat.

Examples

```
if (requireNamespace('rstanarm', quietly=TRUE)) {
  ### Usage with stanreg objects
  n <- 30
  d <- 5
  x <- matrix(rnorm(n*d), nrow=n)
  y <- x[,1] + 0.5*rnorm(n)
  data <- data.frame(x,y)
  fit <- rstanarm::stan_glm(y ~ X1 + X2 + X3 + X4 + X5, gaussian(),
           data=data, chains=2, iter=500)
  vs <- cv_varsel(fit)
  suggest_size(vs)
}
```

summary.vsel *Summary statistics related to variable selection*

Description

Summary statistics related to variable selection

Usage

```
## S3 method for class 'vsel'
summary(
  object,
  nterms_max = NULL,
  stats = "elpd",
  type = c("mean", "se"),
  deltas = FALSE,
  alpha = 0.32,
  baseline = NULL,
  ...
)
```

Arguments

object	The object returned by varsel or cv_varsel.
nterms_max	Maximum submodel size for which the statistics are calculated. For plot.vsel it must be at least 1.
stats	One or several strings determining which statistics to calculate. Available statistics are: • elpd: (Expected) sum of log predictive densities • mlpd: Mean log predictive density, that is, elpd divided by the number of datapoints. • mse: Mean squared error (gaussian family only) • rmse: Root mean squared error (gaussian family only) • acc/pctcorr: Classification accuracy (binomial family only) • auc: Area under the ROC curve (binomial family only) Default is "elpd".
type	One or more items from 'mean', 'se', 'lower' and 'upper' indicating which of these to compute (mean, standard error, and lower and upper credible bounds). The credible bounds are determined so that 1-alpha percent of the mass falls between them.
deltas	If TRUE, the submodel statistics are estimated relative to the baseline model (see argument baseline) instead of estimating the actual values of the statistics. Defaults to FALSE.

alpha	A number indicating the desired coverage of the credible intervals. For example alpha=0.32 corresponds to 68% probability mass within the intervals, that is, one standard error intervals.
baseline	Either 'ref' or 'best' indicating whether the baseline is the reference model or the best submodel found. Default is 'ref' when the reference model exists, and 'best' otherwise.
...	Currently ignored.

Examples

```
if (requireNamespace('rstanarm', quietly=TRUE)) {
  ### Usage with stanreg objects
  n <- 30
  d <- 5
  x <- matrix(rnorm(n*d), nrow=n)
  y <- x[,1] + 0.5*rnorm(n)
  data <- data.frame(x,y)

 fit <- rstanarm::stan_glm(y ~ X1 + X2 + X3 + X4 + X5, gaussian(), data=data, chains=2, iter=500)
  vs <- cv_varsel(fit)
  plot(vs)

  # print out some stats
  summary(vs, stats=c('mse'), type = c('mean','se'))
}
```

---

varsel *Variable selection for generalized linear models*

Description

Perform the projection predictive variable selection for generalized linear models, generalized linear and additive multilevel models using generic reference models.

Usage

```
varsel(object, ...)

## Default S3 method:
varsel(object, ...)

## S3 method for class 'refmodel'
varsel(
  object,
  d_test = NULL,
```

```
  method = NULL,
  ndraws = NULL,
  nclusters = NULL,
  ndraws_pred = NULL,
  nclusters_pred = NULL,
  cv_search = TRUE,
  nterms_max = NULL,
  intercept = TRUE,
  verbose = TRUE,
  lambda_min_ratio = 1e-05,
  nlambda = 150,
  thresh = 1e-06,
  regul = 1e-04,
  penalty = NULL,
  search_terms = NULL,
  ...
)
```

Arguments

object	Either a refmodel-type object created by get_refmodel, a init_refmodel, an object which can be converted to a reference model using get_refmodel or a vsel object resulting from varsel or cv_varsel.
...	Additional arguments to be passed to the get_refmodel-function.
d_test	A test dataset, which is used to evaluate model performance. If not provided, training data is used. Currently this argument is for internal use only.
method	The method used in the variable selection. Possible options are 'L1' for L1-search and 'forward' for forward selection. Default is 'forward' if the number of variables in the full data is at most 20,' and 'L1' otherwise.
ndraws	Number of posterior draws used in the variable selection. Cannot be larger than the number of draws in the reference model. Ignored if nclusters is set.
nclusters	Number of clusters to use in the clustered projection. Overrides the ndraws argument. Defaults to 1.
ndraws_pred	Number of projected draws used for prediction (after selection). Ignored if nclusters_pred is given. Note that setting less draws or clusters than posterior draws in the reference model may result in slightly inaccurate projection performance, although increasing this argument linearly affects the computation time.
nclusters_pred	Number of clusters used for prediction (after selection). Default is 5.
cv_search	If TRUE, then the projected coefficients after L1-selection are computed without any penalization (or using only the regularization determined by regul). If FALSE, then the coefficients are the solution from the' L1-penalized projection. This option is relevant only if method='L1'. Default is TRUE for genuine reference models and FALSE if object is datafit (see init_refmodel).
nterms_max	Maximum number of varibles until which the selection is continued. Defaults to min(20, D, floor(0.4*n)) where n is the number of observations and D the number of variables.

intercept	Whether to use intercept in the submodels. Defaults to TRUE.
verbose	If TRUE, may print out some information during the selection. Defaults to FALSE.
lambda_min_ratio	Ratio between the smallest and largest lambda in the L1-penalized search. This parameter essentially determines how long the search is carried out, i.e., how large submodels are explored. No need to change the default value unless the program gives a warning about this.
nlambda	Number of values in the lambda grid for L1-penalized search. No need to change unless the program gives a warning about this.
thresh	Convergence threshold when computing L1-path. Usually no need to change this.
regul	Amount of regularization in the projection. Usually there is no need for regularization, but sometimes for some models the projection can be ill-behaved and we need to add some regularization to avoid numerical problems.
penalty	Vector determining the relative penalties or costs for the variables. Zero means that those variables have no cost and will therefore be selected first, whereas Inf means those variables will never be selected. Currently works only if method == 'L1'. By default 1 for each variable.
search_terms	A custom list of terms to evaluate for variable selection. By default considers all the terms in the reference model's formula.

Value

An object of type vsel that contains information about the feature selection. The fields are not meant to be accessed directly by the user but instead via the helper functions (see the vignettes or type ?projpred to see the main functions in the package.)

Examples

```
if (requireNamespace('rstanarm', quietly=TRUE)) {
  ### Usage with stanreg objects
  n <- 30
  d <- 5
  x <- matrix(rnorm(n*d), nrow=n)
  y <- x[,1] + 0.5*rnorm(n)
  data <- data.frame(x,y)
  fit <- rstanarm::stan_glm(y ~ X1 + X2 + X3 + X4 + X5, gaussian(), data=data,
    chains=2, iter=500)
  vs <- varsel(fit)
  plot(vs)
}
```

Index