Package ‘autoMrP’

July 22, 2025

Type Package

Title Improving MrP with Ensemble Learning

Version 1.0.6

Description A tool that improves the prediction performance of multilevel regression with post-stratification (MrP) by combining a number of machine learning methods. For information on the method, please refer to Broniecki, Wüest, Leemann (2020) "Improving Multilevel Regression with Post-Stratification Through Machine Learning (autoMrP)" in the 'Journal of Politics'. Final pre-print version: <https://lucasleemann.files.wordpress.com/2020/07/automrp-r2pa.pdf>.

URL https://github.com/retowuest/autoMrP

BugReports https://github.com/retowuest/autoMrP/issues

Depends R (>= 3.6)

Imports rlang (>= 0.4.5), dplyr (>= 1.0.2), lme4 (>= 1.1), gbm (>= 2.1.5), e1071 (>= 1.7-3), tibble (>= 3.0.1), glmmLasso (>= 1.5.1), EBMAforecast (>= 1.0.0), foreach (>= 1.5.0), doParallel (>= 1.0.15), doRNG (>= 1.8.2), ggplot2 (>= 3.3.2), knitr (>= 1.29), tidyr (>= 1.1.2), purrr (>= 0.3.4), forcats (>= 0.5.1), vglmer (>= 1.0.3), stringr (>= 1.5.0)

Suggests R.rsp

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

VignetteBuilder R.rsp

NeedsCompilation no

Author Reto Wüest [aut] (ORCID: <https://orcid.org/0000-0002-7502-6489>),
Lucas Leemann [aut] (ORCID: <https://orcid.org/0000-0001-5201-869X>),
Florian Schaffner [aut] (ORCID:
<https://orcid.org/0000-0003-3352-6191>),
Philipp Broniecki [aut, cre] (ORCID:

<https://orcid.org/0000-0001-9214-4404>),
Hadley Wickham [ctb]

Maintainer Philipp Broniecki <philippbroniecki@gmail.com>

Repository CRAN

Date/Publication 2024-01-30 18:10:02 UTC

Contents

`absentee_census` *Quasi census data.*

Description

The census file is generated from the full 2008 Cooperative Congressional Election Studies item cc419_1 by dissaggregating the 64 ideal type combinations of the individual level variables L1x1, L2x2 and L1x3. A row is an ideal type in a given state.

Usage

```
data(absentee_census)
```

Format

A data frame with 2934 rows and 13 variables:

state U.S. state

L2.unit U.S. state id

region U.S. region (four categories: 1 = Northeast; 2 = Midwest; 3 = South; 4 = West)

L1x1 Age group (four categories)

L1x2 Education level (four categories)

L1x3 Gender-race combination (six categories)

proportion State-level proportion of respondents of that ideal type in the population

L2.x1 State-level share of votes for the Republican candidate in the previous presidential election

L2.x2 State-level percentage of Evangelical Protestant or Mormon respondents

L2.x3 State-level percentage of the population living in urban areas

L2.x4 State-level unemployment rate

L2.x5 State-level share of Hispanics

L2.x6 State-level share of Whites

Source

The data set (excluding L2.x3, L2.x4, L2.x5, L2.x6) is taken from the article: Buttice, Matthew K, and Benjamin Highton. 2013. "How does multilevel regression and poststrat-stratification perform with conventional national surveys?" Political Analysis 21(4): 449-467. L2.x3, L2.x3, L2.x4, L2.x5 and L2.x6 are available at https://www.census.gov.

---

absentee_voting *A sample of the absentee voting item from the CCES 2008*

Description

The Cooperative Congressional Election Stuides (CCES) item (cc419_1) asked: "States have tried many new ways to run elections in recent years. Do you support or oppose any of the following ways of voting or conducting elections in your state? Election Reform - Allow absentee voting over the Internet?" The original 2008 CCES item contains 26,934 respondents. This sample mimics a typical national survey. It contains at least 5 respondents from each state but is otherwise a random sample.

Usage

```
data(absentee_voting)
```

Format

A data frame with 1500 rows and 13 variables:

YES 1 if individual supports use of troops; 0 otherwise

L1x1 Age group (four categories: 1 = 18-29; 2 = 30-44; 3 = 45-64; 4 = 65+)

L1x2 Education level (four categories: 1 = < high school; 2 = high school graduate; 3 = some college; 4 = college graduate)

L1x3 Gender-race combination (six categories: 1 = white male; 2 = black male; 3 = hispanic male; 4 = white female; 5 = black female; 6 = hispanic female)

state U.S. state

L2.unit U.S. state id

region U.S. region (four categories: 1 = Northeast; 2 = Midwest; 3 = South; 4 = West)

L2.x1 State-level share of votes for the Republican candidate in the previous presidential election

L2.x2 State-level percentage of Evangelical Protestant or Mormon respondents

L2.x3 State-level percentage of the population living in urban areas

L2.x4 State-level unemployment rate

L2.x5 State-level share of Hispanics

L2.x6 State-level share of Whites

Source

The data set (excluding L2.x3, L2.x4, L2.x5, L2.x6) is taken from the article: Buttice, Matthew K, and Benjamin Highton. 2013. "How does multilevel regression and poststrat-stratification perform with conventional national surveys?" Political Analysis 21(4): 449-467. It is a random sample with at least 5 respondents per state. L2.x3, L2.x3, L2.x4, L2.x5 and L2.x6 are available at https://www.census.gov.

---

auto_MrP *Improve MrP through ensemble learning.*

Description

This package improves the prediction performance of multilevel regression with post-stratification (MrP) by combining a number of machine learning methods through ensemble Bayesian model averaging (EBMA).

Usage

```
auto_MrP(
  y,
  L1.x,
  L2.x,
  L2.unit,
  L2.reg = NULL,
  L2.x.scale = TRUE,
  pcs = NULL,
  folds = NULL,
  bin.proportion = NULL,
  bin.size = NULL,
  survey,
  census,
  ebma.size = 1/3,
  cores = 1,
  k.folds = 5,
  cv.sampling = "L2 units",
  loss.unit = c("individuals", "L2 units"),
  loss.fun = c("msfe", "cross-entropy", "f1", "MSE"),
  best.subset = TRUE,
  lasso = TRUE,
  pca = TRUE,
  gb = TRUE,
  svm = TRUE,
  mrp = FALSE,
  deep.mrp = FALSE,
  oversampling = FALSE,
  best.subset.L2.x = NULL,
```

```
  lasso.L2.x = NULL,
  pca.L2.x = NULL,
  gb.L2.x = NULL,
  svm.L2.x = NULL,
  mrp.L2.x = NULL,
  gb.L2.unit = TRUE,
  gb.L2.reg = FALSE,
  svm.L2.unit = TRUE,
  svm.L2.reg = FALSE,
  deep.L2.x = NULL,
  deep.L2.reg = TRUE,
  deep.splines = TRUE,
  lasso.lambda = NULL,
  lasso.n.iter = 100,
  gb.interaction.depth = c(1, 2, 3),
  gb.shrinkage = c(0.04, 0.01, 0.008, 0.005, 0.001),
  gb.n.trees.init = 50,
  gb.n.trees.increase = 50,
  gb.n.trees.max = 1000,
  gb.n.minobsinnode = 20,
  svm.kernel = c("radial"),
  svm.gamma = NULL,
  svm.cost = NULL,
  ebma.n.draws = 100,
  ebma.tol = c(0.01, 0.005, 0.001, 5e-04, 1e-04, 5e-05, 1e-05),
  verbose = FALSE,
  uncertainty = FALSE,
  boot.iter = NULL
)
```

Arguments

`y`	Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
`L1.x`	Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
`L2.x`	Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`. To exclude context-level variables, set `L2.x = NULL`.
`L2.unit`	Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
`L2.reg`	Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped (`L2.unit` must be nested within `L2.reg`). Default is `NULL`.
`L2.x.scale`	Scale context-level covariates. A logical argument indicating whether the context-level covariates should be normalized. Default is `TRUE`. Note that if set to

FALSE, then the context-level covariates should be normalized prior to calling auto_MrP().

pcs
: Principal components. A character vector containing the column names of the principal components of the context-level variables in survey and census. Default is NULL.

folds
: EBMA and cross-validation folds. A character scalar containing the column name of the variable in survey that specifies the fold to which an observation is allocated. The variable should contain integers running from 1 to $k + 1$, where k is the number of cross-validation folds. Value $k + 1$ refers to the EBMA fold. Default is NULL. *Note:* if folds is NULL, then ebma.size, k.folds, and cv.sampling must be specified.

bin.proportion
: Proportion of ideal types. A character scalar containing the column name of the variable in census that indicates the proportion of individuals by ideal type and geographic unit. Default is NULL. *Note:* if bin.proportion is NULL, then bin.size must be specified.

bin.size
: Bin size of ideal types. A character scalar containing the column name of the variable in census that indicates the bin size of ideal types by geographic unit. Default is NULL. *Note:* ignored if bin.proportion is provided, but must be specified otherwise.

survey
: Survey data. A data.frame whose column names include y, L1.x, L2.x, L2.unit, and, if specified, L2.reg, pcs, and folds.

census
: Census data. A data.frame whose column names include L1.x, L2.x, L2.unit, if specified, L2.reg and pcs, and either bin.proportion or bin.size.

ebma.size
: EBMA fold size. A number in the open unit interval indicating the proportion of respondents to be allocated to the EBMA fold. Default is $1/3$. *Note:* ignored if folds is provided, but must be specified otherwise.

cores
: The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

k.folds
: Number of cross-validation folds. An integer-valued scalar indicating the number of folds to be used in cross-validation. Default is 5. *Note:* ignored if folds is provided, but must be specified otherwise.

cv.sampling
: Cross-validation sampling method. A character-valued scalar indicating whether cross-validation folds should be created by sampling individual respondents (individuals) or geographic units (L2 units). Default is L2 units. *Note:* ignored if folds is provided, but must be specified otherwise.

loss.unit
: Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (individuals), geographic units (L2 units) or at both levels. Default is c("individuals", "L2 units"). With multiple loss units, parameters are ranked for each loss unit and the loss unit with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

loss.fun
: Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (MSE), the mean absolute error (MAE), binary cross-entropy (cross-entropy), mean squared false error (msfe), the f1 score (f1), or a combination thereof. Default is c("MSE","cross-entropy","msfe",

"f1"). With multiple loss functions, parameters are ranked for each loss function and the parameter combination with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

best.subset
: Best subset classifier. A logical argument indicating whether the best subset classifier should be used for predicting outcome y. Default is TRUE.

lasso
: Lasso classifier. A logical argument indicating whether the lasso classifier should be used for predicting outcome y. Default is TRUE.

pca
: PCA classifier. A logical argument indicating whether the PCA classifier should be used for predicting outcome y. Default is TRUE.

gb
: GB classifier. A logical argument indicating whether the GB classifier should be used for predicting outcome y. Default is TRUE.

svm
: SVM classifier. A logical argument indicating whether the SVM classifier should be used for predicting outcome y. Default is TRUE.

mrp
: MRP classifier. A logical argument indicating whether the standard MRP classifier should be used for predicting outcome y. Default is FALSE.

deep.mrp
: Deep MRP classifier. A logical argument indicating whether the deep MRP classifier should be used for predicting outcome y. Default is FALSE.

oversampling
: Over sample to create balance on the dependent variable. A logical argument. Default is FALSE.

best.subset.L2.x
: Best subset context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the best subset classifier. If NULL and best.subset is set to TRUE, then best subset uses the variables specified in L2.x. Default is NULL.

lasso.L2.x
: Lasso context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the lasso classifier. If NULL and lasso is set to TRUE, then lasso uses the variables specified in L2.x. Default is NULL.

pca.L2.x
: PCA context-level covariates. A character vector containing the column names of the context-level variables in survey and census whose principal components are to be used by the PCA classifier. If NULL and pca is set to TRUE, then PCA uses the principal components of the variables specified in L2.x. Default is NULL.

gb.L2.x
: GB context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the GB classifier. If NULL and gb is set to TRUE, then GB uses the variables specified in L2.x. Default is NULL.

svm.L2.x
: SVM context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the SVM classifier. If NULL and svm is set to TRUE, then SVM uses the variables specified in L2.x. Default is NULL.

mrp.L2.x
: MRP context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the MRP classifier. The character vector *empty* if no context-level variables should be used by the MRP classifier. If NULL and mrp is set to TRUE, then MRP uses the

variables specified in `L2.x`. Default is `NULL`. Note: For the empty MrP model, set `L2.x = NULL` and `mrp.L2.x = ""`.

`gb.L2.unit` GB L2.unit. A logical argument indicating whether `L2.unit` should be included in the GB classifier. Default is `FALSE`.

`gb.L2.reg` GB L2.reg. A logical argument indicating whether `L2.reg` should be included in the GB classifier. Default is `FALSE`.

`svm.L2.unit` SVM L2.unit. A logical argument indicating whether `L2.unit` should be included in the SVM classifier. Default is `FALSE`.

`svm.L2.reg` SVM L2.reg. A logical argument indicating whether `L2.reg` should be included in the SVM classifier. Default is `FALSE`.

`deep.L2.x` Deep MRP context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` to be used by the deep MRP classifier. If `NULL` and `deep.mrp` is set to `TRUE`, then deep MRP uses the variables specified in `L2.x`. Default is `NULL`.

`deep.L2.reg` Deep MRP L2.reg. A logical argument indicating whether `L2.reg` should be included in the deep MRP classifier. Default is `TRUE`.

`deep.splines` Deep MRP splines. A logical argument indicating whether splines should be used in the deep MRP classifier. Default is `TRUE`.

`lasso.lambda` Lasso penalty parameter. A numeric `vector` of non-negative values. The penalty parameter controls the shrinkage of the context-level variables in the lasso model. Default is a sequence with minimum 0.1 and maximum 250 that is equally spaced on the log-scale. The number of values is controlled by the `lasso.n.iter` parameter.

`lasso.n.iter` Lasso number of lambda values. An integer-valued scalar specifying the number of lambda values to search over. Default is 100. *Note:* Is ignored if a vector of `lasso.lambda` values is provided.

`gb.interaction.depth`
GB interaction depth. An integer-valued vector whose values specify the interaction depth of GB. The interaction depth defines the maximum depth of each tree grown (i.e., the maximum level of variable interactions). Default is `c(1, 2, 3)`.

`gb.shrinkage` GB learning rate. A numeric vector whose values specify the learning rate or step-size reduction of GB. Values between 0.001 and 0.1 usually work, but a smaller learning rate typically requires more trees. Default is `c(0.04, 0.01, 0.008, 0.005, 0.001)`.

`gb.n.trees.init`
GB initial total number of trees. An integer-valued scalar specifying the initial number of total trees to fit by GB. Default is 50.

`gb.n.trees.increase`
GB increase in total number of trees. An integer-valued scalar specifying by how many trees the total number of trees to fit should be increased (until `gb.n.trees.max` is reached). Default is 50.

`gb.n.trees.max` GB maximum number of trees. An integer-valued scalar specifying the maximum number of trees to fit by GB. Default is 1000.

gb.n.minobsinnode
: GB minimum number of observations in the terminal nodes. An integer-valued scalar specifying the minimum number of observations that each terminal node of the trees must contain. Default is 20.

svm.kernel
: SVM kernel. A character-valued scalar specifying the kernel to be used by SVM. The possible values are `linear`, `polynomial`, `radial`, and `sigmoid`. Default is `radial`.

svm.gamma
: SVM kernel parameter. A numeric vector whose values specify the gamma parameter in the SVM kernel. This parameter is needed for all kernel types except linear. Default is a sequence with minimum = 1e-5, maximum = 1e-1, and length = 20 that is equally spaced on the log-scale.

svm.cost
: SVM cost parameter. A numeric vector whose values specify the cost of constraints violation in SVM. Default is a sequence with minimum = 0.5, maximum = 10, and length = 5 that is equally spaced on the log-scale.

ebma.n.draws
: EBMA number of samples. An integer-valued scalar specifying the number of bootstrapped samples to be drawn from the EBMA fold and used for tuning EBMA. Default is 100.

ebma.tol
: EBMA tolerance. A numeric vector containing the tolerance values for improvements in the log-likelihood before the EM algorithm stops optimization. Values should range at least from 0.01 to 0.001. Default is `c(0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001)`.

verbose
: Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is `FALSE`.

uncertainty
: Uncertainty estimates. A logical argument indicating whether uncertainty estimates should be computed. Default is `FALSE`.

boot.iter
: Number of bootstrap iterations. An integer argument indicating the number of bootstrap iterations to be computed. Will be ignored unless `uncertainty = TRUE`. Default is `200` if `uncertainty = TRUE` and `NULL` if `uncertainty = FALSE`.

Details

Bootstrapping samples the level two units, sometimes referred to as the cluster bootstrap. For the multilevel model, for example, when running MrP only, the bootstrapped median level two predictions will differ from the level two predictions without bootstrapping. We recommend assessing the difference by running autoMrP without bootstrapping alongside autoMrP with bootstrapping and then comparing level two predictions from the model without bootstrapping to the median level two predictions from the model with bootstrapping.

To ensure reproducability of the results, use the `set.seed()` function to specify a seed.

Value

The context-level predictions. A list with two elements. The first element, `EBMA`, contains the post-stratified ensemble bayesian model avaeraging (EBMA) predictions. The second element, `classifiers`, contains the post-stratified predictions from all estimated classifiers.

Examples

```
# An MrP model without machine learning
set.seed(123)
m <- auto_MrP(
  y = "YES",
  L1.x = c("L1x1"),
  L2.x = c("L2.x1", "L2.x2"),
  L2.unit = "state",
  bin.proportion = "proportion",
  survey = taxes_survey,
  census = taxes_census,
  ebma.size = 0,
  cores = 2,
  best.subset = FALSE,
  lasso = FALSE,
  pca = FALSE,
  gb = FALSE,
  svm = FALSE,
  mrp = TRUE
)

# summarize and plot results
summary(m)
plot(m)

# An MrP model without context-level predictors
m <- auto_MrP(
  y = "YES",
  L1.x = "L1x1",
  L2.x = NULL,
  mrp.L2.x = "",
  L2.unit = "state",
  bin.proportion = "proportion",
  survey = taxes_survey,
  census = taxes_census,
  ebma.size = 0,
  cores = 1,
  best.subset = FALSE,
  lasso = FALSE,
  pca = FALSE,
  gb = FALSE,
  svm = FALSE,
  mrp = TRUE
  )


# Predictions with machine learning

# detect number of available cores
max_cores <- parallel::detectCores()

# autoMrP with machine learning
```

```
ml_out <- auto_MrP(
  y = "YES",
  L1.x = c("L1x1", "L1x2", "L1x3"),
  L2.x = c("L2.x1", "L2.x2", "L2.x3", "L2.x4", "L2.x5", "L2.x6"),
  L2.unit = "state",
  L2.reg = "region",
  bin.proportion = "proportion",
  survey = taxes_survey,
  census = taxes_census,
  gb.L2.reg = TRUE,
  svm.L2.reg = TRUE,
  cores = min(2, max_cores)
  )
```

---

best_subset_classifier *Best subset classifier*

Description

`best_subset_classifier` applies best subset classification to a data set.

Usage

```
best_subset_classifier(
  model,
  data.train,
  model.family,
  model.optimizer,
  n.iter,
  y,
  verbose = c(TRUE, FALSE)
)
```

Arguments

`model`	Multilevel model. A model formula describing the multilevel model to be estimated on the basis of the provided training data.
`data.train`	Training data. A data.frame containing the training data used to train the model.
`model.family`	Model family. A variable indicating the model family to be used by glmer. Defaults to binomial(link = "probit").
`model.optimizer`	Optimization method. A character-valued scalar describing the optimization method to be used by glmer. Defaults to "bobyqa".
`n.iter`	Iterations. A integer-valued scalar specifying the maximum number of function evaluations tried by the optimization method.

y	Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in survey.
verbose	Verbose output. A logical vector indicating whether or not verbose output should be printed.

Value

The multilevel model. An glmer object.

---

binary_cross_entropy *Estimates the inverse binary cross-entropy, i.e. 0 is the best score and 1 the worst.*

Description

binary_cross_entropy() estimates the inverse binary cross-entropy on the individual and state-level.

Usage

```
binary_cross_entropy(
  pred,
  data.valid,
  loss.unit = c("individuals", "L2 units"),
  y,
  L2.unit
)
```

Arguments

pred	Predictions of outcome. A numeric vector of outcome predictions.
data.valid	Test data set. A tibble of data that was not used for prediction.
loss.unit	Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (individuals) or geographic units (L2 units). Default is individuals.
y	Outcome variable. A character vector containing the column names of the outcome variable.
L2.unit	Geographic unit. A character scalar containing the column name of the geographic unit in survey and census at which outcomes should be aggregated.

Value

Returns a tibble containing two binary cross-entropy prediction errors. The first is measured at the level of individuals and the second is measured at the context level. The tibble dimensions are 2x3 with variables: measure, value and level.

boot_auto_mrp *Bootstrappinng wrapper for auto_mrp*

Description

boot_auto_mrp estimates uncertainty for auto_mrp via botstrapping.

Usage

```
boot_auto_mrp(
  y,
  L1.x,
  L2.x,
  mrp.L2.x,
  L2.unit,
  L2.reg,
  L2.x.scale,
  pcs,
  folds,
  bin.proportion,
  bin.size,
  survey,
  census,
  ebma.size,
  k.folds,
  cv.sampling,
  loss.unit,
  loss.fun,
  best.subset,
  lasso,
  pca,
  gb,
  svm,
  mrp,
  deep.mrp,
  best.subset.L2.x,
  lasso.L2.x,
  pca.L2.x,
  pc.names,
  gb.L2.x,
  svm.L2.x,
  svm.L2.unit,
  svm.L2.reg,
  gb.L2.unit,
  gb.L2.reg,
  deep.L2.x,
  deep.L2.reg,
```

```
    deep.splines,
    lasso.lambda,
    lasso.n.iter,
    gb.interaction.depth,
    gb.shrinkage,
    gb.n.trees.init,
    gb.n.trees.increase,
    gb.n.trees.max,
    gb.n.minobsinnode,
    svm.kernel,
    svm.gamma,
    svm.cost,
    ebma.tol,
    boot.iter,
    cores
  )
```

Arguments

- `y` — Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
- `L1.x` — Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
- `L2.x` — Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`. To exclude context-level variables, set `L2.x = NULL`.
- `mrp.L2.x` — MRP context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` to be used by the MRP classifier. The character vector *empty* if no context-level variables should be used by the MRP classifier. If `NULL` and `mrp` is set to `TRUE`, then MRP uses the variables specified in `L2.x`. Default is `NULL`. Note: For the empty MrP model, set `L2.x = NULL` and `mrp.L2.x = ""`.
- `L2.unit` — Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
- `L2.reg` — Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped (`L2.unit` must be nested within `L2.reg`). Default is `NULL`.
- `L2.x.scale` — Scale context-level covariates. A logical argument indicating whether the context-level covariates should be normalized. Default is `TRUE`. Note that if set to `FALSE`, then the context-level covariates should be normalized prior to calling `auto_MrP()`.
- `pcs` — Principal components. A character vector containing the column names of the principal components of the context-level variables in `survey` and `census`. Default is `NULL`.

folds
: EBMA and cross-validation folds. A character scalar containing the column name of the variable in survey that specifies the fold to which an observation is allocated. The variable should contain integers running from 1 to $k + 1$, where k is the number of cross-validation folds. Value $k + 1$ refers to the EBMA fold. Default is NULL. *Note:* if folds is NULL, then ebma.size, k.folds, and cv.sampling must be specified.

bin.proportion
: Proportion of ideal types. A character scalar containing the column name of the variable in census that indicates the proportion of individuals by ideal type and geographic unit. Default is NULL. *Note:* if bin.proportion is NULL, then bin.size must be specified.

bin.size
: Bin size of ideal types. A character scalar containing the column name of the variable in census that indicates the bin size of ideal types by geographic unit. Default is NULL. *Note:* ignored if bin.proportion is provided, but must be specified otherwise.

survey
: Survey data. A data.frame whose column names include y, L1.x, L2.x, L2.unit, and, if specified, L2.reg, pcs, and folds.

census
: Census data. A data.frame whose column names include L1.x, L2.x, L2.unit, if specified, L2.reg and pcs, and either bin.proportion or bin.size.

ebma.size
: EBMA fold size. A number in the open unit interval indicating the proportion of respondents to be allocated to the EBMA fold. Default is $1/3$. *Note:* ignored if folds is provided, but must be specified otherwise.

k.folds
: Number of cross-validation folds. An integer-valued scalar indicating the number of folds to be used in cross-validation. Default is 5. *Note:* ignored if folds is provided, but must be specified otherwise.

cv.sampling
: Cross-validation sampling method. A character-valued scalar indicating whether cross-validation folds should be created by sampling individual respondents (individuals) or geographic units (L2 units). Default is L2 units. *Note:* ignored if folds is provided, but must be specified otherwise.

loss.unit
: Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (individuals), geographic units (L2 units) or at both levels. Default is c("individuals", "L2 units"). With multiple loss units, parameters are ranked for each loss unit and the loss unit with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

loss.fun
: Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (MSE), the mean absolute error (MAE), binary cross-entropy (cross-entropy), mean squared false error (msfe), the f1 score (f1), or a combination thereof. Default is c("MSE","cross-entropy","msfe", "f1"). With multiple loss functions, parameters are ranked for each loss function and the parameter combination with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

best.subset
: Best subset classifier. A logical argument indicating whether the best subset classifier should be used for predicting outcome y. Default is TRUE.

lasso
: Lasso classifier. A logical argument indicating whether the lasso classifier should be used for predicting outcome y. Default is TRUE.

pca	PCA classifier. A logical argument indicating whether the PCA classifier should be used for predicting outcome y. Default is TRUE.
gb	GB classifier. A logical argument indicating whether the GB classifier should be used for predicting outcome y. Default is TRUE.
svm	SVM classifier. A logical argument indicating whether the SVM classifier should be used for predicting outcome y. Default is TRUE.
mrp	MRP classifier. A logical argument indicating whether the standard MRP classifier should be used for predicting outcome y. Default is FALSE.
deep.mrp	Deep MRP classifier. A logical argument indicating whether the deep MRP classifier should be used for predicting outcome y. Default is FALSE.
best.subset.L2.x	Best subset context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the best subset classifier. If NULL and best.subset is set to TRUE, then best subset uses the variables specified in L2.x. Default is NULL.
lasso.L2.x	Lasso context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the lasso classifier. If NULL and lasso is set to TRUE, then lasso uses the variables specified in L2.x. Default is NULL.
pca.L2.x	PCA context-level covariates. A character vector containing the column names of the context-level variables in survey and census whose principal components are to be used by the PCA classifier. If NULL and pca is set to TRUE, then PCA uses the principal components of the variables specified in L2.x. Default is NULL.
pc.names	A character vector of the principal component variable names in the data.
gb.L2.x	GB context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the GB classifier. If NULL and gb is set to TRUE, then GB uses the variables specified in L2.x. Default is NULL.
svm.L2.x	SVM context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the SVM classifier. If NULL and svm is set to TRUE, then SVM uses the variables specified in L2.x. Default is NULL.
svm.L2.unit	SVM L2.unit. A logical argument indicating whether L2.unit should be included in the SVM classifier. Default is FALSE.
svm.L2.reg	SVM L2.reg. A logical argument indicating whether L2.reg should be included in the SVM classifier. Default is FALSE.
gb.L2.unit	GB L2.unit. A logical argument indicating whether L2.unit should be included in the GB classifier. Default is FALSE.
gb.L2.reg	GB L2.reg. A logical argument indicating whether L2.reg should be included in the GB classifier. Default is FALSE.
deep.L2.x	Deep MRP context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the deep MRP classifier. If NULL and deep.mrp is set to TRUE, then deep MRP uses the variables specified in L2.x. Default is NULL.

deep.L2.reg
: Deep MRP L2.reg. A logical argument indicating whether `L2.reg` should be included in the deep MRP classifier. Default is `TRUE`.

deep.splines
: Deep MRP splines. A logical argument indicating whether splines should be used in the deep MRP classifier. Default is `TRUE`.

lasso.lambda
: Lasso penalty parameter. A numeric `vector` of non-negative values. The penalty parameter controls the shrinkage of the context-level variables in the lasso model. Default is a sequence with minimum 0.1 and maximum 250 that is equally spaced on the log-scale. The number of values is controlled by the `lasso.n.iter` parameter.

lasso.n.iter
: Lasso number of lambda values. An integer-valued scalar specifying the number of lambda values to search over. Default is 100. *Note:* Is ignored if a vector of `lasso.lambda` values is provided.

gb.interaction.depth
: GB interaction depth. An integer-valued vector whose values specify the interaction depth of GB. The interaction depth defines the maximum depth of each tree grown (i.e., the maximum level of variable interactions). Default is `c(1, 2, 3)`.

gb.shrinkage
: GB learning rate. A numeric vector whose values specify the learning rate or step-size reduction of GB. Values between 0.001 and 0.1 usually work, but a smaller learning rate typically requires more trees. Default is `c(0.04, 0.01, 0.008, 0.005, 0.001)`.

gb.n.trees.init
: GB initial total number of trees. An integer-valued scalar specifying the initial number of total trees to fit by GB. Default is 50.

gb.n.trees.increase
: GB increase in total number of trees. An integer-valued scalar specifying by how many trees the total number of trees to fit should be increased (until `gb.n.trees.max` is reached). Default is 50.

gb.n.trees.max
: GB maximum number of trees. An integer-valued scalar specifying the maximum number of trees to fit by GB. Default is 1000.

gb.n.minobsinnode
: GB minimum number of observations in the terminal nodes. An integer-valued scalar specifying the minimum number of observations that each terminal node of the trees must contain. Default is 20.

svm.kernel
: SVM kernel. A character-valued scalar specifying the kernel to be used by SVM. The possible values are `linear`, `polynomial`, `radial`, and `sigmoid`. Default is `radial`.

svm.gamma
: SVM kernel parameter. A numeric vector whose values specify the gamma parameter in the SVM kernel. This parameter is needed for all kernel types except linear. Default is a sequence with minimum = 1e-5, maximum = 1e-1, and length = 20 that is equally spaced on the log-scale.

svm.cost
: SVM cost parameter. A numeric vector whose values specify the cost of constraints violation in SVM. Default is a sequence with minimum = 0.5, maximum = 10, and length = 5 that is equally spaced on the log-scale.

ebma.tol
: EBMA tolerance. A numeric vector containing the tolerance values for improvements in the log-likelihood before the EM algorithm stops optimization. Values should range at least from 0.01 to 0.001. Default is `c(0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001)`.

boot.iter
: Number of bootstrap iterations. An integer argument indicating the number of bootstrap iterations to be computed. Will be ignored unless `uncertainty = TRUE`. Default is `200` if `uncertainty = TRUE` and `NULL` if `uncertainty = FALSE`.

cores
: The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

---

census *Quasi census data.*

Description

The census file is generated from the full 2008 Cooperative Congressional Election Studies item cc418_1 by dissaggregating the 64 ideal type combinations of the individual level variables L1x1, L2x2 and L1x3. A row is an ideal type in a given state.

Usage

```
census
```

Format

A data frame with 2934 rows and 13 variables:

state U.S. state

L2.unit U.S. state id

region U.S. region (four categories: 1 = Northeast; 2 = Midwest; 3 = South; 4 = West)

L1x1 Age group (four categories)

L1x2 Education level (four categories)

L1x3 Gender-race combination (six categories)

proportion State-level proportion of respondents of that ideal type in the population

L2.x1 State-level share of votes for the Republican candidate in the previous presidential election

L2.x2 State-level percentage of Evangelical Protestant or Mormon respondents

L2.x3 State-level percentage of the population living in urban areas

L2.x4 State-level unemployment rate

L2.x5 State-level share of Hispanics

L2.x6 State-level share of Whites

Source

The data set (excluding L2.x3, L2.x4, L2.x5, L2.x6) is taken from the article: Buttice, Matthew K, and Benjamin Highton. 2013. "How does multilevel regression and poststrat-stratification perform with conventional national surveys?" Political Analysis 21(4): 449-467. L2.x3, L2.x3, L2.x4, L2.x5 and L2.x6 are available at https://www.census.gov.

`cv_folding` *Generates folds for cross-validation*

Description

`cv_folding` creates folds used in classifier training within the survey data.

Usage

```
cv_folding(data, L2.unit, k.folds, cv.sampling = c("individuals", "L2 units"))
```

Arguments

`data`	The survey data; must be a tibble.
`L2.unit`	The column name of the factor variable identifying the context-level unit
`k.folds`	An integer value indicating the number of folds to be generated.
`cv.sampling`	Cross-validation sampling method. A character-valued scalar indicating whether cross-validation folds should be created by sampling individual respondents (`individuals`) or geographic units (`L2 units`). Default is `L2 units`. *Note:* ignored if `folds` is provided, but must be specified otherwise.

Value

Returns a list with length specified by `k.folds` argument. Each element is a tibble with a fold used in k-fold cross-validation.

`deep_mrp_classifier` *Deep MrP classifier*

Description

`deep_mrp_classifier` applies Deep MrP implemented in the **vglmer** package to a data set.

Usage

```
deep_mrp_classifier(y, form, data, verbose)
```

Arguments

`y`	Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
`form`	Model formula. A two-sided linear formula describing the model to be fit, with the outcome on the LHS and the covariates separated by + operators on the RHS.
`data`	Data. A data.frame containing the data used to train the model.
`verbose`	Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is `FALSE`.

Value

A Deep MrP model. A `vglmer` object.

`ebma` *Bayesian Ensemble Model Averaging EBMA*

Description

`ebma` tunes EBMA and generates weights for classifier averaging.

Usage

```
ebma(
  ebma.fold,
  y,
  L1.x,
  L2.x,
  L2.unit,
  L2.reg,
  pc.names,
  post.strat,
  n.draws,
  tol,
  best.subset.opt,
  pca.opt,
  lasso.opt,
  gb.opt,
  svm.opt,
  deep.mrp,
  verbose,
  cores
)
```

Arguments

- `ebma.fold` New data for EBMA tuning. A list containing the the data that must not have been used in classifier training.
- `y` Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
- `L1.x` Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
- `L2.x` Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`. To exclude context-level variables, set `L2.x = NULL`.
- `L2.unit` Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
- `L2.reg` Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped (`L2.unit` must be nested within `L2.reg`). Default is `NULL`.
- `pc.names` Principal Component Variable names. A character vector containing the names of the context-level principal components variables.
- `post.strat` Post-stratification results. A list containing the best models for each of the tuned classifiers, the individual level predictions on the data classifier trainig data and the post-stratified context-level predictions.
- `n.draws` EBMA number of samples. An integer-valued scalar specifying the number of bootstrapped samples to be drawn from the EBMA fold and used for tuning EBMA. Default is 100. Passed on from `ebma.n.draws`.
- `tol` EBMA tolerance. A numeric vector containing the tolerance values for improvements in the log-likelihood before the EM algorithm stops optimization. Values should range at least from 0.01 to 0.001. Default is `c(0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001)`. Passed on from `ebma.tol`.
- `best.subset.opt` Tuned best subset parameters. A list returned from `run_best_subset()`.
- `pca.opt` Tuned best subset with principal components parameters. A list returned from `run_pca()`.
- `lasso.opt` Tuned lasso parameters. A list returned from `run_lasso()`.
- `gb.opt` Tuned gradient tree boosting parameters. A list returned from `run_gb()`.
- `svm.opt` Tuned support vector machine parameters. A list returned from `run_svm()`.
- `deep.mrp` Deep MRP classifier. A logical argument indicating whether the deep MRP classifier should be used for predicting outcome `y`. Default is `FALSE`.
- `verbose` Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is `FALSE`.
- `cores` The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

`ebma_folding` *Generates data fold to be used for EBMA tuning*

Description

#' `ebma_folding()` generates a data fold that will not be used in classifier tuning. It is data that is needed to determine the optimal tolerance for EBMA.

Usage

```
ebma_folding(data, L2.unit, ebma.size)
```

Arguments

`data`	The full survey data. A tibble.
`L2.unit`	Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
`ebma.size`	EBMA fold size. A number in the open unit interval indicating the proportion of respondents to be allocated to the EBMA fold. Default is $1/3$.

Value

Returns a list with two elements which are both tibble. List element one is named `ebma_fold` and contains the tibble used in Ensemble Bayesian Model Averaging Tuning. List element two is named `cv_data` and contains the tibble used for classifier tuning.

`ebma_mc_draws` *EBMA multicore tuning - parallelises over draws.*

Description

`ebma_mc_draws` is called from within `ebma`. It tunes using multiple cores.

Usage

```
ebma_mc_draws(
  train.preds,
  train.y,
  ebma.fold,
  y,
  L1.x,
  L2.x,
  L2.unit,
  L2.reg,
  pc.names,
```

```
  model.bs,
  model.pca,
  model.lasso,
  model.gb,
  model.svm,
  model.mrp,
  model_deep,
  tol,
  n.draws,
  cores
)
```

Arguments

train.preds	Predictions of classifiers on the classifier training data. A tibble.
train.y	Outcome variable of the classifier training data. A numeric vector.
ebma.fold	New data for EBMA tuning. A list containing the the data that must not have been used in classifier training.
y	Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in survey.
L1.x	Individual-level covariates. A character vector containing the column names of the individual-level variables in survey and census used to predict outcome y. Note that geographic unit is specified in argument L2.unit.
L2.x	Context-level covariates. A character vector containing the column names of the context-level variables in survey and census used to predict outcome y. To exclude context-level variables, set L2.x = NULL.
L2.unit	Geographic unit. A character scalar containing the column name of the geographic unit in survey and census at which outcomes should be aggregated.
L2.reg	Geographic region. A character scalar containing the column name of the geographic region in survey and census by which geographic units are grouped (L2.unit must be nested within L2.reg). Default is NULL.
pc.names	Principal Component Variable names. A character vector containing the names of the context-level principal components variables.
model.bs	The tuned model from the multilevel regression with best subset selection classifier. An glmer object.
model.pca	The tuned model from the multilevel regression with principal components as context-level predictors classifier. An glmer object.
model.lasso	The tuned model from the multilevel regression with L1 regularization classifier. A glmmLasso object.
model.gb	The tuned model from the gradient boosting classifier. A gbm object.
model.svm	The tuned model from the support vector machine classifier. An svm object.
model.mrp	The standard MrP model. An glmer object
model_deep	The tuned model from the deep mrp classifier. An vglmer object.

tol	EBMA tolerance. A numeric vector containing the tolerance values for improvements in the log-likelihood before the EM algorithm stops optimization. Values should range at least from 0.01 to 0.001. Default is `c(0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001)`. Passed on from `ebma.tol`.
n.draws	EBMA number of samples. An integer-valued scalar specifying the number of bootstrapped samples to be drawn from the EBMA fold and used for tuning EBMA. Default is 100. Passed on from `ebma.n.draws`.
cores	The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

Value

The classifier weights. A numeric vector.

`ebma_mc_tol` *EBMA multicore tuning - parallelises over tolerance values.*

Description

`ebma_mc_tol` is called from within `ebma`. It tunes using multiple cores.

Usage

```
ebma_mc_tol(
  train.preds,
  train.y,
  ebma.fold,
  y,
  L1.x,
  L2.x,
  L2.unit,
  L2.reg,
  pc.names,
  model.bs,
  model.pca,
  model.lasso,
  model.gb,
  model.svm,
  model.mrp,
  model_deep,
  tol,
  n.draws,
  cores
)
```

Arguments

`train.preds`	Predictions of classifiers on the classifier training data. A tibble.
`train.y`	Outcome variable of the classifier training data. A numeric vector.
`ebma.fold`	The data used for EBMA tuning. A tibble.
`y`	Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
`L1.x`	Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
`L2.x`	Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`. To exclude context-level variables, set `L2.x = NULL`.
`L2.unit`	Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
`L2.reg`	Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped (`L2.unit` must be nested within `L2.reg`). Default is `NULL`.
`pc.names`	Principal Component Variable names. A character vector containing the names of the context-level principal components variables.
`model.bs`	The tuned model from the multilevel regression with best subset selection classifier. An `glmer` object.
`model.pca`	The tuned model from the multilevel regression with principal components as context-level predictors classifier. An `glmer` object.
`model.lasso`	The tuned model from the multilevel regression with L1 regularization classifier. A `glmmLasso` object.
`model.gb`	The tuned model from the gradient boosting classifier. A `gbm` object.
`model.svm`	The tuned model from the support vector machine classifier. An `svm` object.
`model.mrp`	The standard MrP model. An `glmer` object
`model_deep`	The tuned model from the deep mrp classifier. An `vglmer` object.
`tol`	The tolerance values used for EBMA. A numeric vector.
`n.draws`	EBMA number of samples. An integer-valued scalar specifying the number of bootstrapped samples to be drawn from the EBMA fold and used for tuning EBMA. Default is 100. Passed on from `ebma.n.draws`.
`cores`	The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

Value

The classifier weights. A numeric vector.

Examples

```
## Not run:
# not yet

## End(Not run)
```

---

error_checks *Catches user input errors*

Description

`error_checks()` checks for incorrect data entry in `autoMrP()` call.

Usage

```
error_checks(
  y,
  L1.x,
  L2.x,
  L2.unit,
  L2.reg,
  L2.x.scale,
  pcs,
  folds,
  bin.proportion,
  bin.size,
  survey,
  census,
  ebma.size,
  k.folds,
  cv.sampling,
  loss.unit,
  loss.fun,
  best.subset,
  lasso,
  pca,
  gb,
  svm,
  mrp,
  best.subset.L2.x,
  lasso.L2.x,
  gb.L2.x,
  svm.L2.x,
  mrp.L2.x,
  gb.L2.unit,
  gb.L2.reg,
  lasso.lambda,
```

```
  lasso.n.iter,
  uncertainty,
  boot.iter
)
```

Arguments

- `y` — Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
- `L1.x` — Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
- `L2.x` — Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`. To exclude context-level variables, set `L2.x = NULL`.
- `L2.unit` — Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
- `L2.reg` — Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped (`L2.unit` must be nested within `L2.reg`). Default is `NULL`.
- `L2.x.scale` — Scale context-level covariates. A logical argument indicating whether the context-level covariates should be normalized. Default is `TRUE`. Note that if set to `FALSE`, then the context-level covariates should be normalized prior to calling `auto_MrP()`.
- `pcs` — Principal components. A character vector containing the column names of the principal components of the context-level variables in `survey` and `census`. Default is `NULL`.
- `folds` — EBMA and cross-validation folds. A character scalar containing the column name of the variable in `survey` that specifies the fold to which an observation is allocated. The variable should contain integers running from 1 to $k + 1$, where k is the number of cross-validation folds. Value $k + 1$ refers to the EBMA fold. Default is `NULL`. *Note:* if `folds` is `NULL`, then `ebma.size`, `k.folds`, and `cv.sampling` must be specified.
- `bin.proportion` — Proportion of ideal types. A character scalar containing the column name of the variable in `census` that indicates the proportion of individuals by ideal type and geographic unit. Default is `NULL`. *Note:* if `bin.proportion` is `NULL`, then `bin.size` must be specified.
- `bin.size` — Bin size of ideal types. A character scalar containing the column name of the variable in `census` that indicates the bin size of ideal types by geographic unit. Default is `NULL`. *Note:* ignored if `bin.proportion` is provided, but must be specified otherwise.
- `survey` — Survey data. A `data.frame` whose column names include `y`, `L1.x`, `L2.x`, `L2.unit`, and, if specified, `L2.reg`, `pcs`, and `folds`.
- `census` — Census data. A `data.frame` whose column names include `L1.x`, `L2.x`, `L2.unit`, if specified, `L2.reg` and `pcs`, and either `bin.proportion` or `bin.size`.

`ebma.size` EBMA fold size. A number in the open unit interval indicating the proportion of respondents to be allocated to the EBMA fold. Default is $1/3$. *Note:* ignored if `folds` is provided, but must be specified otherwise.

`k.folds` Number of cross-validation folds. An integer-valued scalar indicating the number of folds to be used in cross-validation. Default is 5. *Note:* ignored if `folds` is provided, but must be specified otherwise.

`cv.sampling` Cross-validation sampling method. A character-valued scalar indicating whether cross-validation folds should be created by sampling individual respondents (`individuals`) or geographic units (`L2 units`). Default is `L2 units`. *Note:* ignored if `folds` is provided, but must be specified otherwise.

`loss.unit` Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (`individuals`), geographic units (`L2 units`) or at both levels. Default is `c("individuals", "L2 units")`. With multiple loss units, parameters are ranked for each loss unit and the loss unit with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

`loss.fun` Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (`MSE`), the mean absolute error (`MAE`), binary cross-entropy (`cross-entropy`), mean squared false error (`msfe`), the f1 score (`f1`), or a combination thereof. Default is `c("MSE","cross-entropy","msfe", "f1")`. With multiple loss functions, parameters are ranked for each loss function and the parameter combination with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

`best.subset` Best subset classifier. A logical argument indicating whether the best subset classifier should be used for predicting outcome `y`. Default is `TRUE`.

`lasso` Lasso classifier. A logical argument indicating whether the lasso classifier should be used for predicting outcome `y`. Default is `TRUE`.

`pca` PCA classifier. A logical argument indicating whether the PCA classifier should be used for predicting outcome `y`. Default is `TRUE`.

`gb` GB classifier. A logical argument indicating whether the GB classifier should be used for predicting outcome `y`. Default is `TRUE`.

`svm` SVM classifier. A logical argument indicating whether the SVM classifier should be used for predicting outcome `y`. Default is `TRUE`.

`mrp` MRP classifier. A logical argument indicating whether the standard MRP classifier should be used for predicting outcome `y`. Default is `FALSE`.

`best.subset.L2.x`
Best subset context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` to be used by the best subset classifier. If `NULL` and `best.subset` is set to `TRUE`, then best subset uses the variables specified in `L2.x`. Default is `NULL`.

`lasso.L2.x` Lasso context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` to be used by the lasso classifier. If `NULL` and `lasso` is set to `TRUE`, then lasso uses the variables specified in `L2.x`. Default is `NULL`.

`gb.L2.x`	GB context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` to be used by the GB classifier. If `NULL` and `gb` is set to `TRUE`, then GB uses the variables specified in `L2.x`. Default is `NULL`.
`svm.L2.x`	SVM context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` to be used by the SVM classifier. If `NULL` and `svm` is set to `TRUE`, then SVM uses the variables specified in `L2.x`. Default is `NULL`.
`mrp.L2.x`	MRP context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` to be used by the MRP classifier. The character vector *empty* if no context-level variables should be used by the MRP classifier. If `NULL` and `mrp` is set to `TRUE`, then MRP uses the variables specified in `L2.x`. Default is `NULL`. Note: For the empty MrP model, set `L2.x = NULL` and `mrp.L2.x = ""`.
`gb.L2.unit`	GB L2.unit. A logical argument indicating whether `L2.unit` should be included in the GB classifier. Default is `FALSE`.
`gb.L2.reg`	GB L2.reg. A logical argument indicating whether `L2.reg` should be included in the GB classifier. Default is `FALSE`.
`lasso.lambda`	Lasso penalty parameter. A numeric `vector` of non-negative values. The penalty parameter controls the shrinkage of the context-level variables in the lasso model. Default is a sequence with minimum 0.1 and maximum 250 that is equally spaced on the log-scale. The number of values is controlled by the `lasso.n.iter` parameter.
`lasso.n.iter`	Lasso number of lambda values. An integer-valued scalar specifying the number of lambda values to search over. Default is 100. *Note:* Is ignored if a vector of `lasso.lambda` values is provided.
`uncertainty`	Uncertainty estimates. A logical argument indicating whether uncertainty estimates should be computed. Default is `FALSE`.
`boot.iter`	Number of bootstrap iterations. An integer argument indicating the number of bootstrap iterations to be computed. Will be ignored unless `uncertainty = TRUE`. Default is `200` if `uncertainty = TRUE` and `NULL` if `uncertainty = FALSE`.

Value

No return value, called for detection of errors in autoMrP() call.

`f1_score` *Estimates the inverse f1 score, i.e. 0 is the best score and 1 the worst.*

Description

`f1_score()` estimates the inverse f1 scores on the individual and state levels.

Usage

```
f1_score(pred, data.valid, y, L2.unit)
```

Arguments

pred	Predictions of outcome. A numeric vector of outcome predictions.
data.valid	Test data set. A tibble of data that was not used for prediction.
y	Outcome variable. A character vector containing the column names of the outcome variable.
L2.unit	Geographic unit. A character scalar containing the column name of the geographic unit in survey and census at which outcomes should be aggregated.

Value

Returns a tibble containing two f1 prediction errors. The first is measured at the level of individuals and the second is measured at the context level. The tibble dimensions are 2x3 with variables: measure, value and level.

---

gb_classifier *GB classifier*

Description

gb_classifier applies gradient boosting classification to a data set.

Usage

```
gb_classifier(
  y,
  form,
  distribution,
  data.train,
  n.trees,
  interaction.depth,
  n.minobsinnode,
  shrinkage,
  verbose = c(TRUE, FALSE)
)
```

Arguments

y	Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in survey.
form	Model formula. A two-sided linear formula describing the model to be fit, with the outcome on the LHS and the covariates separated by + operators on the RHS.
distribution	Model distribution. A character string specifying the name of the distribution to be used.
data.train	Training data. A data.frame containing the training data used to train the model.

n.trees	Total number of trees. An integer-valued scalar specifying the total number of trees to be fit.
interaction.depth	Interaction depth. An integer-valued scalar specifying the maximum depth of each tree.
n.minobsinnode	Minimum number of observations in terminal nodes. An integer-valued scalar specifying the minimum number of observations in the terminal nodes of the trees.
shrinkage	Learning rate. A numeric scalar specifying the shrinkage or learning rate applied to each tree in the expansion.
verbose	Verbose output. A logical vector indicating whether or not verbose output should be printed.

Value

A gradient tree boosting model. A `gbm` object.

---

gb_classifier_update *GB classifier update*

Description

`gb_classifier_update()` grows additional trees in gradient tree boosting ensemble.

Usage

```
gb_classifier_update(object, n.new.trees, verbose = c(TRUE, FALSE))
```

Arguments

object	Gradient tree boosting output. A gbm object.
n.new.trees	Number of additional trees to grow. A numeric scalar.
verbose	Verbose output. A logical vector indicating whether or not verbose output should be printed.

Value

An updated gradient tree boosting model. A `gbm.more` object.

`lasso_classifier` *Lasso classifier*

Description

`lasso_classifier` applies lasso classification to a data set.

Usage

```
lasso_classifier(
  L2.fix,
  L1.re,
  data.train,
  lambda,
  model.family,
  y,
  verbose = c(TRUE, FALSE)
)
```

Arguments

`L2.fix`	Fixed effects. A two-sided linear formula describing the fixed effects part of the model, with the outcome on the LHS and the fixed effects separated by + operators on the RHS.
`L1.re`	Random effects. A named list object, with the random effects providing the names of the list elements and ~ 1 being the list elements.
`data.train`	Training data. A data.frame containing the training data used to train the model.
`lambda`	Tuning parameter. Lambda is the penalty parameter that controls the shrinkage of fixed effects.
`model.family`	Model family. A variable indicating the model family to be used by glmmLasso. Defaults to binomial(link = "probit").
`y`	Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
`verbose`	Verbose output. A logical vector indicating whether or not verbose output should be printed.

Value

A multilevel lasso model. An `glmmLasso` object.

`log_spaced` *Sequence that is equally spaced on the log scale*

Description

Sequence that is equally spaced on the log scale

Usage

```
log_spaced(min, max, n)
```

Arguments

`min`	The minimum value of the sequence. A positive numeric scalar (min > 0).
`max`	The maximum value of the sequence. a positive numeric scalar (max > 0).
`n`	The length of the sequence. An integer valued scalar.

Value

Returns a numeric vector with length specified in argument `n`. The vector elements are equally spaced on the log-scale.

`loss_function` *Estimates loss value.*

Description

`loss_function()` estimates the loss based on a loss function.

Usage

```
loss_function(
  pred,
  data.valid,
  loss.unit = c("individuals", "L2 units"),
  loss.fun = c("MSE", "MAE", "cross-entropy"),
  y,
  L2.unit
)
```

Arguments

`pred`	Predictions of outcome. A numeric vector of outcome predictions.
`data.valid`	Test data set. A tibble of data that was not used for prediction.
`loss.unit`	Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (`individuals`) or geographic units (`L2 units`). Default is `individuals`.
`loss.fun`	Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (`MSE`) or the mean absolute error (`MAE`). Default is `MSE`.
`y`	Outcome variable. A character vector containing the column names of the outcome variable.
`L2.unit`	Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.

Value

Returns a tibble with number of rows equal to the number of loss functions tested (defaults to 4 for cross-entropy, f1, MSE, and msfe). The number of columns is 2 where the first is called measure and contains the names of the loss-functions and the second is called value and contains the loss-function scores.

`loss_score_ranking` *Ranks tuning parameters according to loss functions*

Description

`loss_score_ranking()` ranks tuning parameters according to the scores received in multiple loss functions.

Usage

```
loss_score_ranking(score, loss.fun)
```

Arguments

`score`	A data set containing loss function names, the loss function values, and the tuning parameter values.
`loss.fun`	Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (`MSE`) or the mean absolute error (`MAE`). Default is `MSE`.

Value

Returns a tibble containing the parameter grid as well as a rank column that corresponds to the cross-validation rank of a parameter combination across all loss function scores.

mean_absolute_error *Estimates the mean absolute prediction error.*

Description

`mean_absolute_error()` estimates the mean absolute error for the desired loss unit.

Usage

```
mean_absolute_error(pred, data.valid, y, L2.unit)
```

Arguments

`pred`	Predictions of outcome. A numeric vector of outcome predictions.
`data.valid`	Test data set. A tibble of data that was not used for prediction.
`y`	Outcome variable. A character vector containing the column names of the outcome variable.
`L2.unit`	Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.

Value

Returns a tibble containing two mean absolute prediction errors. The first is measured at the level of individuals and the second is measured at the context level. The tibble dimensions are 2x3 with variables: measure, value and level.

mean_squared_error *Estimates the mean squared prediction error.*

Description

`mean_squared_error()` estimates the mean squared error for the desired loss unit.

Usage

```
mean_squared_error(pred, data.valid, y, L2.unit)
```

Arguments

`pred`	Predictions of outcome. A numeric vector of outcome predictions.
`data.valid`	Test data set. A tibble of data that was not used for prediction.
`y`	Outcome variable. A character vector containing the column names of the outcome variable.
`L2.unit`	Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.

Value

Returns a tibble containing two mean squared prediction errors. The first is measured at the level of individuals and the second is measured at the context level. The tibble dimensions are 2x3 with variables: measure, value and level.

---

mean_squared_false_error

Estimates the mean squared false error.

Description

`msfe()` estimates the inverse f1 scores on the individual and state levels.

Usage

```
mean_squared_false_error(pred, data.valid, y, L2.unit)
```

Arguments

`pred`	Predictions of outcome. A numeric vector of outcome predictions.
`data.valid`	Test data set. A tibble of data that was not used for prediction.
`y`	Outcome variable. A character vector containing the column names of the outcome variable.
`L2.unit`	Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.

Value

Returns a tibble containing two mean squared false prediction errors. The first is measured at the level of individuals and the second is measured at the context level. The tibble dimensions are 2x3 with variables: measure, value and level.

---

model_list

A list of models for the best subset selection.

Description

`model_list()` generates an exhaustive list of lme4 model formulas from the individual level and context level variables as well as geographic unit variables to be iterated over in best subset selection.

Usage

```
model_list(y, L1.x, L2.x, L2.unit, L2.reg = NULL)
```

Arguments

`y`	Outcome variable. A character vector containing the column names of the outcome variable.
`L1.x`	Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
`L2.x`	Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`.
`L2.unit`	Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
`L2.reg`	Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped (`L2.unit` must be nested within `L2.reg`). Default is `NULL`.

Value

Returns a list with the number of elements equal to 2^k where k is the number context-level variables. Each element is of class formula.

`model_list_pca` *A list of models for the best subset selection with PCA.*

Description

`model_list_pca()` generates an exhaustive list of lme4 model formulas from the individual level and context level principal components as well as geographic unit variables to be iterated over in best subset selection with principal components.

Usage

```
model_list_pca(y, L1.x, L2.x, L2.unit, L2.reg = NULL)
```

Arguments

`y`	Outcome variable. A character vector containing the column names of the outcome variable.
`L1.x`	Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
`L2.x`	Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`.
`L2.unit`	Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
`L2.reg`	Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped (`L2.unit` must be nested within `L2.reg`). Default is `NULL`.

Value

Returns a list with the number of elements k+1 where k is the number of context-level variables. Each element is of class formula. The first element is a model with context-level variables and the following models iteratively add the principal components as context-level variables.

---

multicore *Register cores for multicore computing*

Description

`multicore()` registers cores for parallel processing.

Usage

```
multicore(cores = 1, type, cl = NULL)
```

Arguments

`cores`	Number of cores to be used. An integer. Default is `1`.
`type`	Whether to start or end parallel processing. A character string. The possible values are `open`, `close`.
`cl`	The registered cluster. Default is `NULL`

Value

No return value, called to register or un-register clusters for parallel processing.

---

output_table *A table for the summary function*

Description

`output_table()` ...

Usage

```
output_table(object, col.names, format, digits)
```

Arguments

`object`	An `autoMrP()` object for which a summary is desired.
`col.names`	The column names of the table. A
`format`	The table format. A character string passed to `kable`. Default is `simple`.
`digits`	The number of digits to be displayed. An integer scalar. Default is `4`.

Value

No return value, prints a table to the console.

`plot.autoMrP` *A plot method for autoMrP objects. Plots unit-level preference estiamtes.*

Description

`plot.autoMrP()` plots unit-level preference estimates and error bars.

Usage

```
## S3 method for class 'autoMrP'
plot(x, algorithm = "ebma", ci.lvl = 0.95, ...)
```

Arguments

`x`	An `autoMrP()` object.
`algorithm`	The algorithm/classifier fo which preference estimates are desired. A character-valued scalar indicating either `ebma` or the classifier to be used. Allowed choices are: "ebma", "best_subset", "lasso", "pca", "gb", "svm", and "mrp". Default is `ebma`.
`ci.lvl`	The level of the confidence intervals. A proportion. Default is `0.95`. Confidence intervals are based on bootstrapped estimates and will not be printed if bootstrapping was not carried out.
`...`	Additional arguments affecting the summary produced.

Value

Returns a `ggplot2` object of the preference estimates for the selected classifier.

`post_stratification` *Apply post-stratification to classifiers.*

Description

Apply post-stratification to classifiers.

Usage

```
post_stratification(
  y,
  L1.x,
  L2.x,
  L2.unit,
  L2.reg,
  best.subset.opt,
  lasso.opt,
  lasso.L2.x,
  pca.opt,
  gb.opt,
  svm.opt,
  svm.L2.reg,
  svm.L2.unit,
  svm.L2.x,
  mrp.include,
  n.minobsinnode,
  L2.unit.include,
  L2.reg.include,
  kernel,
  mrp.L2.x,
  data,
  ebma.fold,
  census,
  verbose,
  deep.mrp,
  deep.L2.x,
  deep.L2.reg,
  deep.splines
)
```

Arguments

`y`	Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
`L1.x`	Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
`L2.x`	Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`. To exclude context-level variables, set `L2.x = NULL`.
`L2.unit`	Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
`L2.reg`	Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped

(L2.unit must be nested within L2.reg). Default is NULL.

best.subset.opt
: Optimal tuning parameters from best subset selection classifier. A list returned by run_best_subset().

lasso.opt
: Optimal tuning parameters from lasso classifier A list returned by run_lasso().

lasso.L2.x
: Lasso context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the lasso classifier. If NULL and lasso is set to TRUE, then lasso uses the variables specified in L2.x. Default is NULL.

pca.opt
: Optimal tuning parameters from best subset selection with principal components classifier A list returned by run_pca().

gb.opt
: Optimal tuning parameters from gradient tree boosting classifier A list returned by run_gb().

svm.opt
: Optimal tuning parameters from support vector machine classifier A list returned by run_svm().

svm.L2.reg
: SVM L2.reg. A logical argument indicating whether L2.reg should be included in the SVM classifier. Default is FALSE.

svm.L2.unit
: SVM L2.unit. A logical argument indicating whether L2.unit should be included in the SVM classifier. Default is FALSE.

svm.L2.x
: SVM context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the SVM classifier. If NULL and svm is set to TRUE, then SVM uses the variables specified in L2.x. Default is NULL.

mrp.include
: Whether to run MRP classifier. A logical argument indicating whether the standard MRP classifier should be used for predicting outcome y. Passed from autoMrP() argument mrp.

n.minobsinnode
: GB minimum number of observations in the terminal nodes. An integer-valued scalar specifying the minimum number of observations that each terminal node of the trees must contain. Passed from autoMrP() argument gb.n.minobsinnode.

L2.unit.include
: GB L2.unit. A logical argument indicating whether L2.unit should be included in the GB classifier. Passed from autoMrP() argument gb.L2.unit.

L2.reg.include
: A logical argument indicating whether L2.reg should be included in the GB classifier. Passed from autoMrP() argument GB L2.reg.

kernel
: SVM kernel. A character-valued scalar specifying the kernel to be used by SVM. The possible values are linear, polynomial, radial, and sigmoid. Passed from autoMrP() argument svm.kernel.

mrp.L2.x
: MRP context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the MRP classifier. The character vector *empty* if no context-level variables should be used by the MRP classifier. If NULL and mrp is set to TRUE, then MRP uses the variables specified in L2.x. Default is NULL. Note: For the empty MrP model, set L2.x = NULL and mrp.L2.x = "".

data
: A data.frame containing the survey data used in classifier training.

ebma.fold	A data.frame containing the data not used in classifier training.
census	Census data. A data.frame whose column names include L1.x, L2.x, L2.unit, if specified, L2.reg and pcs, and either bin.proportion or bin.size.
verbose	Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is FALSE.
deep.mrp	Deep MRP classifier. A logical argument indicating whether the deep MRP classifier should be used for predicting outcome y. Default is FALSE.
deep.L2.x	Deep MRP context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the deep MRP classifier. If NULL and deep.mrp is set to TRUE, then deep MRP uses the variables specified in L2.x. Default is NULL.
deep.L2.reg	Deep MRP L2.reg. A logical argument indicating whether L2.reg should be included in the deep MRP classifier. Default is TRUE.
deep.splines	Deep MRP splines. A logical argument indicating whether splines should be used in the deep MRP classifier. Default is TRUE.

---

predict_glmmLasso *Predicts on newdata from glmmLasso objects*

Description

glmmLasso() predicts on newdata objects from a glmmLasso object.

Usage

```
predict_glmmLasso(census, m, L1.x, lasso.L2.x, L2.unit, L2.reg)
```

Arguments

census	Census data. A data.frame whose column names include L1.x, L2.x, L2.unit, if specified, L2.reg and pcs, and either bin.proportion or bin.size.
m	A glmmLasso() object.
L1.x	Individual-level covariates. A character vector containing the column names of the individual-level variables in survey and census used to predict outcome y. Note that geographic unit is specified in argument L2.unit.
lasso.L2.x	Lasso context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the lasso classifier. If NULL and lasso is set to TRUE, then lasso uses the variables specified in L2.x. Default is NULL.
L2.unit	Geographic unit. A character scalar containing the column name of the geographic unit in survey and census at which outcomes should be aggregated.
L2.reg	Geographic region. A character scalar containing the column name of the geographic region in survey and census by which geographic units are grouped (L2.unit must be nested within L2.reg). Default is NULL.

Value

Returns a numeric vector of predictions from a `glmmLasso()` object.

`quiet` *Suppress cat in external package*

Description

`quiet()` suppresses cat output.

Usage

```
quiet(x)
```

Arguments

`x`	Input. It can be any kind.

`run_best_subset` *Apply best subset classifier to MrP.*

Description

`run_best_subset` is a wrapper function that applies the best subset classifier to a list of models provided by the user, evaluates the models' prediction performance, and chooses the best-performing model.

Usage

```
run_best_subset(
  y,
  L1.x,
  L2.x,
  L2.unit,
  L2.reg,
  loss.unit,
  loss.fun,
  data,
  verbose,
  cores
)
```

Arguments

- `y` — Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
- `L1.x` — Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
- `L2.x` — Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`. To exclude context-level variables, set `L2.x = NULL`.
- `L2.unit` — Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
- `L2.reg` — Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped (`L2.unit` must be nested within `L2.reg`). Default is `NULL`.
- `loss.unit` — Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (`individuals`), geographic units (`L2 units`) or at both levels. Default is `c("individuals", "L2 units")`. With multiple loss units, parameters are ranked for each loss unit and the loss unit with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.
- `loss.fun` — Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (`MSE`), the mean absolute error (`MAE`), binary cross-entropy (`cross-entropy`), mean squared false error (`msfe`), the f1 score (`f1`), or a combination thereof. Default is `c("MSE","cross-entropy","msfe", "f1")`. With multiple loss functions, parameters are ranked for each loss function and the parameter combination with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.
- `data` — Data for cross-validation. A `list` of k `data.frames`, one for each fold to be used in k-fold cross-validation.
- `verbose` — Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is `FALSE`.
- `cores` — The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

Value

A model formula of the winning best subset classifier model.

run_best_subset_mc *Best subset multicore tuning.*

Description

run_best_subset_mc is called from within run_best_subset. It tunes using multiple cores.

Usage

```
run_best_subset_mc(
  y,
  L1.x,
  L2.x,
  L2.unit,
  L2.reg,
  loss.unit,
  loss.fun,
  data,
  cores,
  models,
  verbose
)
```

Arguments

y	Outcome variable. A character scalar containing the column name of the outcome variable in survey.
L1.x	Individual-level covariates. A character vector containing the column names of the individual-level variables in survey and census used to predict outcome y. Note that geographic unit is specified in argument L2.unit.
L2.x	Context-level covariates. A character vector containing the column names of the context-level variables in survey and census used to predict outcome y.
L2.unit	Geographic unit. A character scalar containing the column name of the geographic unit in survey and census at which outcomes should be aggregated.
L2.reg	Geographic region. A character scalar containing the column name of the geographic region in survey and census by which geographic units are grouped (L2.unit must be nested within L2.reg). Default is NULL.
loss.unit	Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (individuals) or geographic units (L2 units). Default is individuals.
loss.fun	Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (MSE) or the mean absolute error (MAE). Default is MSE.
data	Data for cross-validation. A list of k data.frames, one for each fold to be used in k-fold cross-validation.

cores	The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.
models	The models to perform best subset selection on. A list of model formulas.
verbose	Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is TRUE.

Value

The cross-validation errors for all models. A list.

Examples

```
## Not run:
# not yet

## End(Not run)
```

---

run_classifiers *Optimal individual classifiers*

Description

run_classifiers tunes classifiers, post-stratifies and carries out EMBA.

Usage

```
run_classifiers(
  y,
  L1.x,
  L2.x,
  mrp.L2.x,
  L2.unit,
  L2.reg,
  L2.x.scale,
  pcs,
  pc.names,
  folds,
  bin.proportion,
  bin.size,
  cv.folds,
  cv.data,
  ebma.fold,
  census,
  ebma.size,
  ebma.n.draws,
  k.folds,
  cv.sampling,
```

```
  loss.unit,
  loss.fun,
  best.subset,
  lasso,
  pca,
  gb,
  svm,
  mrp,
  deep.mrp,
  best.subset.L2.x,
  lasso.L2.x,
  pca.L2.x,
  gb.L2.x,
  svm.L2.x,
  gb.L2.unit,
  gb.L2.reg,
  svm.L2.unit,
  svm.L2.reg,
  deep.L2.x,
  deep.L2.reg,
  deep.splines,
  lasso.lambda,
  lasso.n.iter,
  gb.interaction.depth,
  gb.shrinkage,
  gb.n.trees.init,
  gb.n.trees.increase,
  gb.n.trees.max,
  gb.n.minobsinnode,
  svm.kernel,
  svm.gamma,
  svm.cost,
  ebma.tol,
  cores,
  verbose
)
```

Arguments

`y`	Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
`L1.x`	Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
`L2.x`	Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`. To exclude context-level variables, set `L2.x = NULL`.

mrp.L2.x
: MRP context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the MRP classifier. The character vector *empty* if no context-level variables should be used by the MRP classifier. If NULL and mrp is set to TRUE, then MRP uses the variables specified in L2.x. Default is NULL. Note: For the empty MrP model, set L2.x = NULL and mrp.L2.x = "".

L2.unit
: Geographic unit. A character scalar containing the column name of the geographic unit in survey and census at which outcomes should be aggregated.

L2.reg
: Geographic region. A character scalar containing the column name of the geographic region in survey and census by which geographic units are grouped (L2.unit must be nested within L2.reg). Default is NULL.

L2.x.scale
: Scale context-level covariates. A logical argument indicating whether the context-level covariates should be normalized. Default is TRUE. Note that if set to FALSE, then the context-level covariates should be normalized prior to calling auto_MrP().

pcs
: Principal components. A character vector containing the column names of the principal components of the context-level variables in survey and census. Default is NULL.

pc.names
: A character vector of the principal component variable names in the data.

folds
: EBMA and cross-validation folds. A character scalar containing the column name of the variable in survey that specifies the fold to which an observation is allocated. The variable should contain integers running from 1 to $k + 1$, where k is the number of cross-validation folds. Value $k + 1$ refers to the EBMA fold. Default is NULL. *Note:* if folds is NULL, then ebma.size, k.folds, and cv.sampling must be specified.

bin.proportion
: Proportion of ideal types. A character scalar containing the column name of the variable in census that indicates the proportion of individuals by ideal type and geographic unit. Default is NULL. *Note:* if bin.proportion is NULL, then bin.size must be specified.

bin.size
: Bin size of ideal types. A character scalar containing the column name of the variable in census that indicates the bin size of ideal types by geographic unit. Default is NULL. *Note:* ignored if bin.proportion is provided, but must be specified otherwise.

cv.folds
: Data for cross-validation. A list of k data.frames, one for each fold to be used in k-fold cross-validation.

cv.data
: A data.frame containing the survey data used in classifier training.

ebma.fold
: A data.frame containing the data not used in classifier training.

census
: Census data. A data.frame whose column names include L1.x, L2.x, L2.unit, if specified, L2.reg and pcs, and either bin.proportion or bin.size.

ebma.size
: EBMA fold size. A number in the open unit interval indicating the proportion of respondents to be allocated to the EBMA fold. Default is $1/3$. *Note:* ignored if folds is provided, but must be specified otherwise.

ebma.n.draws
: EBMA number of samples. An integer-valued scalar specifying the number of bootstrapped samples to be drawn from the EBMA fold and used for tuning EBMA. Default is 100.

k.folds
: Number of cross-validation folds. An integer-valued scalar indicating the number of folds to be used in cross-validation. Default is 5. *Note:* ignored if folds is provided, but must be specified otherwise.

cv.sampling
: Cross-validation sampling method. A character-valued scalar indicating whether cross-validation folds should be created by sampling individual respondents (individuals) or geographic units (L2 units). Default is L2 units. *Note:* ignored if folds is provided, but must be specified otherwise.

loss.unit
: Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (individuals), geographic units (L2 units) or at both levels. Default is c("individuals", "L2 units"). With multiple loss units, parameters are ranked for each loss unit and the loss unit with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

loss.fun
: Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (MSE), the mean absolute error (MAE), binary cross-entropy (cross-entropy), mean squared false error (msfe), the f1 score (f1), or a combination thereof. Default is c("MSE","cross-entropy","msfe", "f1"). With multiple loss functions, parameters are ranked for each loss function and the parameter combination with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

best.subset
: Best subset classifier. A logical argument indicating whether the best subset classifier should be used for predicting outcome y. Default is TRUE.

lasso
: Lasso classifier. A logical argument indicating whether the lasso classifier should be used for predicting outcome y. Default is TRUE.

pca
: PCA classifier. A logical argument indicating whether the PCA classifier should be used for predicting outcome y. Default is TRUE.

gb
: GB classifier. A logical argument indicating whether the GB classifier should be used for predicting outcome y. Default is TRUE.

svm
: SVM classifier. A logical argument indicating whether the SVM classifier should be used for predicting outcome y. Default is TRUE.

mrp
: MRP classifier. A logical argument indicating whether the standard MRP classifier should be used for predicting outcome y. Default is FALSE.

deep.mrp
: Deep MRP classifier. A logical argument indicating whether the deep MRP classifier should be used for predicting outcome y. Default is FALSE.

best.subset.L2.x
: Best subset context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the best subset classifier. If NULL and best.subset is set to TRUE, then best subset uses the variables specified in L2.x. Default is NULL.

lasso.L2.x
: Lasso context-level covariates. A character vector containing the column names of the context-level variables in survey and census to be used by the lasso classifier. If NULL and lasso is set to TRUE, then lasso uses the variables specified in L2.x. Default is NULL.

pca.L2.x
: PCA context-level covariates. A character vector containing the column names of the context-level variables in survey and census whose principal components are to be used by the PCA classifier. If NULL and pca is set to TRUE, then

PCA uses the principal components of the variables specified in `L2.x`. Default is `NULL`.

`gb.L2.x` GB context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` to be used by the GB classifier. If `NULL` and `gb` is set to `TRUE`, then GB uses the variables specified in `L2.x`. Default is `NULL`.

`svm.L2.x` SVM context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` to be used by the SVM classifier. If `NULL` and `svm` is set to `TRUE`, then SVM uses the variables specified in `L2.x`. Default is `NULL`.

`gb.L2.unit` GB L2.unit. A logical argument indicating whether `L2.unit` should be included in the GB classifier. Default is `FALSE`.

`gb.L2.reg` GB L2.reg. A logical argument indicating whether `L2.reg` should be included in the GB classifier. Default is `FALSE`.

`svm.L2.unit` SVM L2.unit. A logical argument indicating whether `L2.unit` should be included in the SVM classifier. Default is `FALSE`.

`svm.L2.reg` SVM L2.reg. A logical argument indicating whether `L2.reg` should be included in the SVM classifier. Default is `FALSE`.

`deep.L2.x` Deep MRP context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` to be used by the deep MRP classifier. If `NULL` and `deep.mrp` is set to `TRUE`, then deep MRP uses the variables specified in `L2.x`. Default is `NULL`.

`deep.L2.reg` Deep MRP L2.reg. A logical argument indicating whether `L2.reg` should be included in the deep MRP classifier. Default is `TRUE`.

`deep.splines` Deep MRP splines. A logical argument indicating whether splines should be used in the deep MRP classifier. Default is `TRUE`.

`lasso.lambda` Lasso penalty parameter. A numeric `vector` of non-negative values. The penalty parameter controls the shrinkage of the context-level variables in the lasso model. Default is a sequence with minimum 0.1 and maximum 250 that is equally spaced on the log-scale. The number of values is controlled by the `lasso.n.iter` parameter.

`lasso.n.iter` Lasso number of lambda values. An integer-valued scalar specifying the number of lambda values to search over. Default is 100. *Note:* Is ignored if a vector of `lasso.lambda` values is provided.

`gb.interaction.depth`
GB interaction depth. An integer-valued vector whose values specify the interaction depth of GB. The interaction depth defines the maximum depth of each tree grown (i.e., the maximum level of variable interactions). Default is `c(1, 2, 3)`.

`gb.shrinkage` GB learning rate. A numeric vector whose values specify the learning rate or step-size reduction of GB. Values between 0.001 and 0.1 usually work, but a smaller learning rate typically requires more trees. Default is `c(0.04, 0.01, 0.008, 0.005, 0.001)`.

`gb.n.trees.init`
GB initial total number of trees. An integer-valued scalar specifying the initial number of total trees to fit by GB. Default is 50.

gb.n.trees.increase
: GB increase in total number of trees. An integer-valued scalar specifying by how many trees the total number of trees to fit should be increased (until `gb.n.trees.max` is reached). Default is 50.

gb.n.trees.max
: GB maximum number of trees. An integer-valued scalar specifying the maximum number of trees to fit by GB. Default is 1000.

gb.n.minobsinnode
: GB minimum number of observations in the terminal nodes. An integer-valued scalar specifying the minimum number of observations that each terminal node of the trees must contain. Default is 20.

svm.kernel
: SVM kernel. A character-valued scalar specifying the kernel to be used by SVM. The possible values are `linear`, `polynomial`, `radial`, and `sigmoid`. Default is `radial`.

svm.gamma
: SVM kernel parameter. A numeric vector whose values specify the gamma parameter in the SVM kernel. This parameter is needed for all kernel types except linear. Default is a sequence with minimum = 1e-5, maximum = 1e-1, and length = 20 that is equally spaced on the log-scale.

svm.cost
: SVM cost parameter. A numeric vector whose values specify the cost of constraints violation in SVM. Default is a sequence with minimum = 0.5, maximum = 10, and length = 5 that is equally spaced on the log-scale.

ebma.tol
: EBMA tolerance. A numeric vector containing the tolerance values for improvements in the log-likelihood before the EM algorithm stops optimization. Values should range at least from 0.01 to 0.001. Default is `c(0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001)`.

cores
: The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

verbose
: Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is `FALSE`.

---

run_gb *Apply gradient boosting classifier to MrP.*

Description

run_gb is a wrapper function that applies the gradient boosting classifier to data provided by the user, evaluates prediction performance, and chooses the best-performing model.

Usage

```
run_gb(
  y,
  L1.x,
  L2.x,
  L2.eval.unit,
  L2.unit,
```

```
  L2.reg,
  loss.unit,
  loss.fun,
  interaction.depth,
  shrinkage,
  n.trees.init,
  n.trees.increase,
  n.trees.max,
  cores = cores,
  n.minobsinnode,
  data,
  verbose
)
```

Arguments

`y` Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.

`L1.x` Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.

`L2.x` Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`. To exclude context-level variables, set `L2.x = NULL`.

`L2.eval.unit` Geographic unit for the loss function. A character scalar containing the column name of the geographic unit in `survey` and `census`.

`L2.unit` Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.

`L2.reg` Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped (`L2.unit` must be nested within `L2.reg`). Default is `NULL`.

`loss.unit` Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (`individuals`) or geographic units (`L2 units`). Default is `individuals`.

`loss.fun` Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (`MSE`) or the mean absolute error (`MAE`). Default is `MSE`.

`interaction.depth`
GB interaction depth. An integer-valued vector whose values specify the interaction depth of GB. The interaction depth defines the maximum depth of each tree grown (i.e., the maximum level of variable interactions). Default is `c(1, 2, 3)`.

`shrinkage` GB learning rate. A numeric vector whose values specify the learning rate or step-size reduction of GB. Values between 0.001 and 0.1 usually work, but a smaller learning rate typically requires more trees. Default is `c(0.04, 0.01, 0.008, 0.005, 0.001)`.

n.trees.init
: GB initial total number of trees. An integer-valued scalar specifying the initial number of total trees to fit by GB. Default is 50.

n.trees.increase
: GB increase in total number of trees. An integer-valued scalar specifying by how many trees the total number of trees to fit should be increased (until `n.trees.max` is reached) or an integer-valued vector of length `length(shrinkage)` with each of its values being associated with a learning rate in `shrinkage`. Default is 50.

n.trees.max
: GB maximum number of trees. An integer-valued scalar specifying the maximum number of trees to fit by GB or an integer-valued vector of length `length(shrinkage)` with each of its values being associated with a learning rate and an increase in the total number of trees. Default is 1000.

cores
: The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

n.minobsinnode
: GB minimum number of observations in the terminal nodes. An integer-valued scalar specifying the minimum number of observations that each terminal node of the trees must contain. Default is 5.

data
: Data for cross-validation. A `list` of k `data.frames`, one for each fold to be used in k-fold cross-validation.

verbose
: Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is `TRUE`.

Value

The tuned gradient boosting parameters. A list with three elements: `interaction_depth` contains the interaction depth parameter, `shrinkage` contains the learning rate, `n_trees` the number of trees to be grown.

---

run_gb_mc *GB multicore tuning.*

Description

`run_gb_mc` is called from within `run_gb`. It tunes using multiple cores.

Usage

```
run_gb_mc(
  y,
  L1.x,
  L2.eval.unit,
  L2.unit,
  L2.reg,
  form,
  gb.grid,
  n.minobsinnode,
```

```
    loss.unit,
    loss.fun,
    data,
    cores
  )
```

Arguments

- `y` — Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
- `L1.x` — Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
- `L2.eval.unit` — Geographic unit for the loss function. A character scalar containing the column name of the geographic unit in `survey` and `census`.
- `L2.unit` — Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
- `L2.reg` — Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped (`L2.unit` must be nested within `L2.reg`). Default is `NULL`.
- `form` — The model formula. A formula object.
- `gb.grid` — The hyper-parameter search grid. A matrix of all hyper-parameter combinations.
- `n.minobsinnode` — GB minimum number of observations in the terminal nodes. An integer-valued scalar specifying the minimum number of observations that each terminal node of the trees must contain. Default is 5.
- `loss.unit` — Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (`individuals`) or geographic units (`L2 units`). Default is `individuals`.
- `loss.fun` — Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (`MSE`) or the mean absolute error (`MAE`). Default is `MSE`.
- `data` — Data for cross-validation. A `list` of k `data.frames`, one for each fold to be used in k-fold cross-validation.
- `cores` — The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

Value

The tuning parameter combinations and there associated loss function scores. A list.

run_lasso *Apply lasso classifier to MrP.*

Description

run_lasso is a wrapper function that applies the lasso classifier to data provided by the user, evaluates prediction performance, and chooses the best-performing model.

Usage

```
run_lasso(
  y,
  L1.x,
  L2.x,
  L2.unit,
  L2.reg,
  n.iter,
  loss.unit,
  loss.fun,
  lambda,
  data,
  verbose,
  cores
)
```

Arguments

y	Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in survey.
L1.x	Individual-level covariates. A character vector containing the column names of the individual-level variables in survey and census used to predict outcome y. Note that geographic unit is specified in argument L2.unit.
L2.x	Context-level covariates. A character vector containing the column names of the context-level variables in survey and census used to predict outcome y. To exclude context-level variables, set L2.x = NULL.
L2.unit	Geographic unit. A character scalar containing the column name of the geographic unit in survey and census at which outcomes should be aggregated.
L2.reg	Geographic region. A character scalar containing the column name of the geographic region in survey and census by which geographic units are grouped (L2.unit must be nested within L2.reg). Default is NULL.
n.iter	Lasso number of lambda values. An integer-valued scalar specifying the number of lambda values to search over. Default is 100. *Note:* Is ignored if a vector of lasso.lambda values is provided.

loss.unit — Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (individuals), geographic units (L2 units) or at both levels. Default is c("individuals", "L2 units"). With multiple loss units, parameters are ranked for each loss unit and the loss unit with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

loss.fun — Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (MSE), the mean absolute error (MAE), binary cross-entropy (cross-entropy), mean squared false error (msfe), the f1 score (f1), or a combination thereof. Default is c("MSE","cross-entropy","msfe", "f1"). With multiple loss functions, parameters are ranked for each loss function and the parameter combination with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

lambda — Lasso penalty parameter. A numeric vector of non-negative values. The penalty parameter controls the shrinkage of the context-level variables in the lasso model. Default is a sequence with minimum 0.1 and maximum 250 that is equally spaced on the log-scale. The number of values is controlled by the lasso.n.iter parameter.

data — Data for cross-validation. A list of k data.frames, one for each fold to be used in k-fold cross-validation.

verbose — Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is FALSE.

cores — The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

Value

The tuned lambda value. A numeric scalar.

---

run_lasso_mc_lambda *Lasso multicore tuning.*

Description

run_lasso_mc_lambda is called from within run_lasso. It tunes using multiple cores.

Usage

```
run_lasso_mc_lambda(
  y,
  L1.x,
  L2.x,
  L2.unit,
  L2.reg,
  loss.unit,
  loss.fun,
```

```
    data,
    cores,
    L2.fe.form,
    L1.re,
    lambda
)
```

Arguments

y
: Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in survey.

L1.x
: Individual-level covariates. A character vector containing the column names of the individual-level variables in survey and census used to predict outcome y. Note that geographic unit is specified in argument L2.unit.

L2.x
: Context-level covariates. A character vector containing the column names of the context-level variables in survey and census used to predict outcome y. To exclude context-level variables, set L2.x = NULL.

L2.unit
: Geographic unit. A character scalar containing the column name of the geographic unit in survey and census at which outcomes should be aggregated.

L2.reg
: Geographic region. A character scalar containing the column name of the geographic region in survey and census by which geographic units are grouped (L2.unit must be nested within L2.reg). Default is NULL.

loss.unit
: Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (individuals), geographic units (L2 units) or at both levels. Default is c("individuals", "L2 units"). With multiple loss units, parameters are ranked for each loss unit and the loss unit with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

loss.fun
: Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (MSE), the mean absolute error (MAE), binary cross-entropy (cross-entropy), mean squared false error (msfe), the f1 score (f1), or a combination thereof. Default is c("MSE","cross-entropy","msfe", "f1"). With multiple loss functions, parameters are ranked for each loss function and the parameter combination with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.

data
: Data for cross-validation. A list of k data.frames, one for each fold to be used in k-fold cross-validation.

cores
: The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

L2.fe.form
: The fixed effects part of the Lasso classifier formula. The formula is inherited from run_lasso.

L1.re
: A list of random effects for the Lasso classifier formula. The formula is inherited from run_lasso.

lambda — Lasso penalty parameter. A numeric vector of non-negative values. The penalty parameter controls the shrinkage of the context-level variables in the lasso model. Default is a sequence with minimum 0.1 and maximum 250 that is equally spaced on the log-scale. The number of values is controlled by the lasso.n.iter parameter.

Value

The cross-validation errors for all models. A list.

---

run_pca — *Apply PCA classifier to MrP.*

Description

run_pca is a wrapper function that applies the PCA classifier to data provided by the user, evaluates prediction performance, and chooses the best-performing model.

Usage

```
run_pca(
  y,
  L1.x,
  L2.x,
  L2.unit,
  L2.reg,
  loss.unit,
  loss.fun,
  data,
  cores,
  verbose
)
```

Arguments

y — Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in survey.

L1.x — Individual-level covariates. A character vector containing the column names of the individual-level variables in survey and census used to predict outcome y. Note that geographic unit is specified in argument L2.unit.

L2.x — Context-level covariates. A character vector containing the column names of the context-level variables in survey and census used to predict outcome y. To exclude context-level variables, set L2.x = NULL.

L2.unit — Geographic unit. A character scalar containing the column name of the geographic unit in survey and census at which outcomes should be aggregated.

L2.reg	Geographic region. A character scalar containing the column name of the geographic region in survey and census by which geographic units are grouped (L2.unit must be nested within L2.reg). Default is NULL.
loss.unit	Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (individuals), geographic units (L2 units) or at both levels. Default is c("individuals", "L2 units"). With multiple loss units, parameters are ranked for each loss unit and the loss unit with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.
loss.fun	Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (MSE), the mean absolute error (MAE), binary cross-entropy (cross-entropy), mean squared false error (msfe), the f1 score (f1), or a combination thereof. Default is c("MSE","cross-entropy","msfe", "f1"). With multiple loss functions, parameters are ranked for each loss function and the parameter combination with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.
data	Data for cross-validation. A list of k data.frames, one for each fold to be used in k-fold cross-validation.
cores	The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.
verbose	Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is FALSE.

Value

A model formula of the winning best subset classifier model.

---

run_svm *Apply support vector machine classifier to MrP.*

Description

run_svm is a wrapper function that applies the support vector machine classifier to data provided by the user, evaluates prediction performance, and chooses the best-performing model.

Usage

```
run_svm(
  y,
  L1.x,
  L2.x,
  L2.eval.unit,
  L2.unit,
  L2.reg,
  kernel = "radial",
  loss.fun,
```

```
  loss.unit,
  gamma,
  cost,
  data,
  verbose,
  cores
)
```

Arguments

- `y` — Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
- `L1.x` — Individual-level covariates. A character vector containing the column names of the individual-level variables in `survey` and `census` used to predict outcome `y`. Note that geographic unit is specified in argument `L2.unit`.
- `L2.x` — Context-level covariates. A character vector containing the column names of the context-level variables in `survey` and `census` used to predict outcome `y`. To exclude context-level variables, set `L2.x = NULL`.
- `L2.eval.unit` — Geographic unit for the loss function. A character scalar containing the column name of the geographic unit in `survey` and `census`.
- `L2.unit` — Geographic unit. A character scalar containing the column name of the geographic unit in `survey` and `census` at which outcomes should be aggregated.
- `L2.reg` — Geographic region. A character scalar containing the column name of the geographic region in `survey` and `census` by which geographic units are grouped (`L2.unit` must be nested within `L2.reg`). Default is `NULL`.
- `kernel` — SVM kernel. A character-valued scalar specifying the kernel to be used by SVM. The possible values are `linear`, `polynomial`, `radial`, and `sigmoid`. Default is `radial`.
- `loss.fun` — Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (`MSE`) or the mean absolute error (`MAE`). Default is `MSE`.
- `loss.unit` — Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (`individuals`), geographic units (`L2 units`) or at both levels. Default is `c("individuals", "L2 units")`. With multiple loss units, parameters are ranked for each loss unit and the loss unit with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.
- `gamma` — SVM kernel parameter. A numeric vector whose values specify the gamma parameter in the SVM kernel. This parameter is needed for all kernel types except linear. Default is a sequence with minimum = 1e-5, maximum = 1e-1, and length = 20 that is equally spaced on the log-scale.
- `cost` — SVM cost parameter. A numeric vector whose values specify the cost of constraints violation in SVM. Default is a sequence with minimum = 0.5, maximum = 10, and length = 5 that is equally spaced on the log-scale.

data — Data for cross-validation. A list of k data.frames, one for each fold to be used in k-fold cross-validation.

verbose — Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is FALSE.

cores — The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.

Value

The support vector machine tuned parameters. A list.

---

run_svm_mc — *SVM multicore tuning.*

Description

run_svm_mc is called from within run_svm. It tunes using multiple cores.

Usage

```
run_svm_mc(
  y,
  L1.x,
  L2.x,
  L2.eval.unit,
  L2.unit,
  L2.reg,
  form,
  loss.unit,
  loss.fun,
  data,
  cores,
  svm.grid,
  verbose
)
```

Arguments

y — Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in survey.

L1.x — Individual-level covariates. A character vector containing the column names of the individual-level variables in survey and census used to predict outcome y. Note that geographic unit is specified in argument L2.unit.

L2.x — Context-level covariates. A character vector containing the column names of the context-level variables in survey and census used to predict outcome y. To exclude context-level variables, set L2.x = NULL.

L2.eval.unit	Geographic unit for the loss function. A character scalar containing the column name of the geographic unit in survey and census.
L2.unit	Geographic unit. A character scalar containing the column name of the geographic unit in survey and census at which outcomes should be aggregated.
L2.reg	Geographic region. A character scalar containing the column name of the geographic region in survey and census by which geographic units are grouped (L2.unit must be nested within L2.reg). Default is NULL.
form	The model formula. A formula object.
loss.unit	Loss function unit. A character-valued scalar indicating whether performance loss should be evaluated at the level of individual respondents (individuals), geographic units (L2 units) or at both levels. Default is c("individuals", "L2 units"). With multiple loss units, parameters are ranked for each loss unit and the loss unit with the lowest rank sum is chosen. Ties are broken according to the order in the search grid.
loss.fun	Loss function. A character-valued scalar indicating whether prediction loss should be measured by the mean squared error (MSE) or the mean absolute error (MAE). Default is MSE.
data	Data for cross-validation. A list of k data.frames, one for each fold to be used in k-fold cross-validation.
cores	The number of cores to be used. An integer indicating the number of processor cores used for parallel computing. Default is 1.
svm.grid	The hyper-parameter search grid. A matrix of all hyper-parameter combinations.
verbose	Verbose output. A logical argument indicating whether or not verbose output should be printed. Default is FALSE.

Value

The cross-validation errors for all models. A list.

---

summary.autoMrP *A summary method for autoMrP objects.*

Description

summary.autoMrP() ...

Usage

```
## S3 method for class 'autoMrP'
summary(
  object,
  ci.lvl = 0.95,
  digits = 4,
  format = "simple",
```

```
    classifiers = NULL,
    n = 10,
    ...
)
```

Arguments

object	An autoMrP() object for which a summary is desired.
ci.lvl	The level of the confidence intervals. A proportion. Default is 0.95. Confidence intervals are based on bootstrapped estimates and will not be printed if bootstrapping was not carried out.
digits	The number of digits to be displayed. An integer scalar. Default is 4.
format	The table format. A character string passed to kable. Default is simple.
classifiers	Summarize a single classifier. A character string. Must be one of best_subset, lasso, pca, gb, svm, or mrp. Default is NULL.
n	Number of rows to be printed. An integer scalar. Default is 10.
...	Additional arguments affecting the summary produced.

Value

No return value, prints a summary of the context level preference estimates to the console.

---

survey_item *A sample of a survey item from the CCES 2008*

Description

The Cooperative Congressional Election Stuides (CCES) item (cc418_1) asked: "Would you approve of the use of U.S. military troops in order to ensure the supply of oil?" The original 2008 CCES item contains 36,832 respondents. This sample mimics a typical national survey. It contains at least 5 respondents from each state but is otherwise a random sample.

Usage

```
survey_item
```

Format

A data frame with 1500 rows and 13 variables:

YES 1 if individual supports use of troops; 0 otherwise

L1x1 Age group (four categories: 1 = 18-29; 2 = 30-44; 3 = 45-64; 4 = 65+)

L1x2 Education level (four categories: 1 = < high school; 2 = high school graduate; 3 = some college; 4 = college graduate)

L1x3 Gender-race combination (six categories: 1 = white male; 2 = black male; 3 = hispanic male; 4 = white female; 5 = black female; 6 = hispanic female)

state U.S. state

L2.unit U.S. state id

region U.S. region (four categories: 1 = Northeast; 2 = Midwest; 3 = South; 4 = West)

L2.x1 Normalized state-level share of votes for the Republican candidate in the previous presidential election

L2.x2 Normalized state-level percentage of Evangelical Protestant or Mormon respondents

L2.x3 Normalized state-level percentage of the population living in urban areas

L2.x4 Normalized state-level unemployment rate

L2.x5 Normalized state-level share of Hispanics

L2.x6 Normalized state-level share of Whites

Source

The data set (excluding L2.x3, L2.x4, L2.x5, L2.x6) is taken from the article: Buttice, Matthew K, and Benjamin Highton. 2013. "How does multilevel regression and poststrat-stratification perform with conventional national surveys?" Political Analysis 21(4): 449-467. It is a random sample with at least 5 respondents per state. L2.x3, L2.x3, L2.x4, L2.x5 and L2.x6 are available at `https://www.census.gov`.

`svm_classifier` *SVM classifier*

Description

`svm_classifier` applies support vector machine classification to a data set.

Usage

```
svm_classifier(
  y,
  form,
  data,
  kernel,
  type,
  probability,
  svm.gamma,
  svm.cost,
  verbose = c(TRUE, FALSE)
)
```

Arguments

`y`	Outcome variable. A character vector containing the column names of the outcome variable. A character scalar containing the column name of the outcome variable in `survey`.
`form`	Model formula. A two-sided linear formula describing the model to be fit, with the outcome on the LHS and the covariates separated by + operators on the RHS.
`data`	Data. A data.frame containing the cross-validation data used to train and evaluate the model.
`kernel`	Kernel for SVM. A character string specifying the kernel to be used for SVM. The possible types are linear, polynomial, radial, and sigmoid. Default is radial.
`type`	svm can be used as a classification machine, as a regression machine, or for novelty detection. Depending of whether y is a factor or not, the default setting for type is C-classification or eps-regression, respectively, but may be overwritten by setting an explicit value. Valid options are: #' 1. C-classification 2. nu-classification 3. one-classification (for novelty detection) 4. eps-regression 5. nu-regression
`probability`	Probability predictions. A logical argument indicating whether the model should allow for probability predictions
`svm.gamma`	Gamma parameter for SVM. This parameter is needed for all kernels except linear.
`svm.cost`	Cost parameter for SVM. This parameter specifies the cost of constraints violation.
`verbose`	Verbose output. A logical vector indicating whether or not verbose output should be printed.

Value

The support vector machine model. An `svm` object.

`taxes_census` *Quasi census data.*

Description

The census file is generated from the full 2008 National Annenberg Election Studies item CBb01 by dissaggregating the 64 ideal type combinations of the individual level variables L1x1, L2x2 and L1x3. A row is an ideal type in a given state.

Usage

```
data(taxes_census)
```

Format

A data frame with 2934 rows and 13 variables:

state U.S. state

L2.unit U.S. state id

region U.S. region (four categories: 1 = Northeast; 2 = Midwest; 3 = South; 4 = West)

L1x1 Age group (four categories)

L1x2 Education level (four categories)

L1x3 Gender-race combination (six categories)

freq State-level frequency of ideal type

proportion State-level proportion of respondents of that ideal type in the population

L2.x1 State-level share of votes for the Republican candidate in the previous presidential election

L2.x2 State-level percentage of Evangelical Protestant or Mormon respondents

L2.x3 State-level percentage of the population living in urban areas

L2.x4 State-level unemployment rate

L2.x5 State-level share of Hispanics

L2.x6 State-level share of Whites

Source

The data set (excluding L2.x3, L2.x4, L2.x5, L2.x6) is taken from the article: Buttice, Matthew K, and Benjamin Highton. 2013. "How does multilevel regression and poststrat-stratification perform with conventional national surveys?" Political Analysis 21(4): 449-467. L2.x3, L2.x3, L2.x4, L2.x5 and L2.x6 are available at `https://www.census.gov`.

`taxes_survey` *Sample on raising taxes from the 2008 National Annenberg Election Studies.*

Description

The 2008 National Annenberg Election Studies (NAES) item (CBb01) asked: "I'm going to read you some options about federal income taxes. Please tell me which one comes closest to your view on what we should be doing about federal income taxes: (1) Cut taxes; (2) Keep taxes as they are; (3) Raise taxes if necessary; (4) None of these; (998) Don't know; (999) No answer. Category (3) was turned into a 'raise taxes response,' categories (1) and (2) were combined into a 'do not raise taxes' response. The original item from the phone and online surveys contains 50,483 respondents. This sample mimics a typical national survey. It contains at least 5 respondents from each state but is otherwise a random sample.

The 2008 National Annenberg Election Studies (NAES) item (CBb01) asked: "I'm going to read you some options about federal income taxes. Please tell me which one comes closest to your view on what we should be doing about federal income taxes: (1) Cut taxes; (2) Keep taxes as they are; (3) Raise taxes if necessary; (4) None of these; (998) Don't know; (999) No answer. Category (3) was turned into a 'raise taxes response,' categories (1) and (2) were combined into a 'do not raise taxes' response. The original item from the phone and online surveys contains 50,483 respondents. This sample mimics a typical national survey. It contains at least 5 respondents from each state but is otherwise a random sample.

Usage

```
data(taxes_survey)
```


```
data(taxes_survey)
```


Format

A data frame with 1500 rows and 13 variables:

YES 1 if individual supports raising taxes; 0 otherwise

L1x1 Age group (four categories: 1 = 18-29; 2 = 30-44; 3 = 45-64; 4 = 65+)

L1x2 Education level (four categories: 1 = < high school; 2 = high school graduate; 3 = some college; 4 = college graduate)

L1x3 Gender-race combination (six categories: 1 = white male; 2 = black male; 3 = hispanic male; 4 = white female; 5 = black female; 6 = hispanic female)

state U.S. state

L2.unit U.S. state id

region U.S. region (four categories: 1 = Northeast; 2 = Midwest; 3 = South; 4 = West)

L2.x1 State-level share of votes for the Republican candidate in the previous presidential election

L2.x2 State-level percentage of Evangelical Protestant or Mormon respondents

L2.x3 State-level percentage of the population living in urban areas

L2.x4 State-level unemployment rate

L2.x5 State-level share of Hispanics

L2.x6 State-level share of Whites

A data frame with 1500 rows and 13 variables:

YES 1 if individual supports raising taxes; 0 otherwise

L1x1 Age group (four categories: 1 = 18-29; 2 = 30-44; 3 = 45-64; 4 = 65+)

L1x2 Education level (four categories: 1 = < high school; 2 = high school graduate; 3 = some college; 4 = college graduate)

L1x3 Gender-race combination (six categories: 1 = white male; 2 = black male; 3 = hispanic male; 4 = white female; 5 = black female; 6 = hispanic female)

state U.S. state

L2.unit U.S. state id

region U.S. region (four categories: 1 = Northeast; 2 = Midwest; 3 = South; 4 = West)

L2.x1 State-level share of votes for the Republican candidate in the previous presidential election

L2.x2 State-level percentage of Evangelical Protestant or Mormon respondents

L2.x3 State-level percentage of the population living in urban areas

L2.x4 State-level unemployment rate

L2.x5 State-level share of Hispanics

L2.x6 State-level share of Whites

Source

The data set (excluding L2.x3, L2.x4, L2.x5, L2.x6) is taken from the article: Buttice, Matthew K, and Benjamin Highton. 2013. "How does multilevel regression and poststrat-stratification perform with conventional national surveys?" Political Analysis 21(4): 449-467. It is a random sample with at least 5 respondents per state. L2.x3, L2.x3, L2.x4, L2.x5 and L2.x6 are available at https://www.census.gov.

The data set (excluding L2.x3, L2.x4, L2.x5, L2.x6) is taken from the article: Buttice, Matthew K, and Benjamin Highton. 2013. "How does multilevel regression and poststrat-stratification perform with conventional national surveys?" Political Analysis 21(4): 449-467. It is a random sample with at least 5 respondents per state. L2.x3, L2.x3, L2.x4, L2.x5 and L2.x6 are available at https://www.census.gov.

Index