

Package: sae.projection (via r-universe)

September 26, 2026

Type Package

Title Small Area Estimation Using Model-Assisted Projection Method

Version 0.1.5

Description Combines information from two independent surveys using a model-assisted projection method. Designed for survey sampling scenarios where a large sample collects only auxiliary information (Survey 1) and a smaller sample provides data on both variables of interest and auxiliary variables (Survey 2). Implements a working model to generate synthetic values of the variable of interest by fitting the model to Survey 2 data and predicting values for Survey 1 based on its auxiliary variables (Kim & Rao, 2012) <[doi:10.1093/biomet/asr063](https://doi.org/10.1093/biomet/asr063)>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL <https://github.com/Alfrzlp/sae.projection>

BugReports <https://github.com/Alfrzlp/sae.projection/issues>

Imports survey, cli, doParallel, dplyr, methods, parsnip, recipes, rlang, rsample, stats, tune, workflows, yardstick, bonsai, ranger, randomForest, themis, lightgbm, caret

Depends R (>= 4.3.0), tidymodels

NeedsCompilation no

VignetteBuilder knitr

Suggests knitr, rmarkdown, quarto, testthat (>= 3.0.0)

Config/testthat/edition 3

Language en-US

Config/roxygen2/version 8.0.0

Config/pak/sysreqs cmake make libicu-dev libuv1-dev

Repository <https://ridsonap.r-universe.dev>

Date/Publication 2026-09-26 02:17:08 UTC

RemoteUrl <https://github.com/ridsonap/sae.projection>
RemoteRef HEAD
RemoteSha 4ef20cd4ad156b8a3da1dd24b1c1649df4e4b4ab

Contents

df_svy_A	2
df_svy_B	3
df_svy22	4
df_svy23	4
ma_projection	5
projection_randomforest	8
Index	12

df_svy_A	df_svy_A
----------	----------

Description

A simulation dataset from a small sample survey, presented only at provincial level (Domain 1).

Usage

df_svy_A

Format

A data frame with 2000 rows and 20 variables with 40 domains.

- province** Province code
- id_ind** Unique identifier for each respondent
- num** Sample number
- weight** Weight from survey
- x1** Predictor variables X1
- x2** Predictor variables X2
- x3** Predictor variables X3
- x4** Predictor variables X4
- x5** Predictor variables X5
- x6** Predictor variables X6
- x7** Predictor variables X7
- x8** Predictor variables X8
- x9** Predictor variables X9

x10 Predictor variables X10
x11 Predictor variables X11
x12 Predictor variables X12
x13 Predictor variables X13
x14 Predictor variables X14
x15 Predictor variables X15
Y Target variable (1: Yes, 0: No)

df_svy_B

df_svy_B

Description

A simulation dataset from a large sample survey, presented at the regency level (Domain 2).

Usage

```
df_svy_B
```

Format

A data frame with 8000 rows and 20 variables with 40 domains.

province Province code
regency Regency code
id_ind Unique identifier for each respondent
num Sample number
weight Weight from survey
x1 Predictor variables X1
x2 Predictor variables X2
x3 Predictor variables X3
x4 Predictor variables X4
x5 Predictor variables X5
x6 Predictor variables X6
x7 Predictor variables X7
x8 Predictor variables X8
x9 Predictor variables X9
x10 Predictor variables X10
x11 Predictor variables X11
x12 Predictor variables X12
x13 Predictor variables X13
x14 Predictor variables X14
x15 Predictor variables X15

df_svy22

df_svy22

Description

A dataset from a survey conducted at the province level in Indonesia in 2022.

Usage

df_svy22

Format

A data frame with 74.070 rows and 11 variables.

PSU Primary Sampling Unit

WEIGHT Weight from survey

PROV province code

REGENCY regency/municipality code

STRATA Strata

income Income

neet Not in education employment or training status

sex sex (1: male, 2: female)

age age

disability disability status (0: False, 1: True)

edu last completed education

df_svy23

df_svy23

Description

A dataset from a survey conducted at the province level in Indonesia in 2023.

Usage

df_svy23

Format

A data frame with 66,245 rows and 11 variables.

PSU Primary Sampling Unit

WEIGHT Weight from survey

PROV province code

REGENCY regency/municipality code

STRATA Strata

income Income

neet Not in education employment or training status

sex sex (1: male, 2: female)

age age

disability disability status (0: False, 1: True)

edu last completed education

ma_projection

Model-Assisted Projection Estimator

Description

The function addresses the problem of combining information from two or more independent surveys, a common challenge in survey sampling. It focuses on cases where:

- **Survey 1:** A large sample collects only auxiliary information.
- **Survey 2:** A much smaller sample collects both the variables of interest and the auxiliary variables.

The function implements a model-assisted projection estimation method based on a working model. The working models that can be used include several machine learning models that can be seen in the details section.

Usage

```
ma_projection(
  formula,
  cluster_ids,
  weight,
  strata = NULL,
  domain,
  summary_function = "mean",
  working_model,
  data_model,
  data_proj,
```

```

  model_metric = NULL,
  cv_folds = 3,
  tuning_grid = 10,
  parallel_over = "resamples",
  cores = NULL,
  seed = 1,
  bias_correction = FALSE,
  return_yhat = FALSE,
  ...
)

```

Arguments

formula	A model formula. All variables used must exist in both <code>data_model</code> and <code>data_proj</code> .
cluster_ids	Column name (character) or formula specifying cluster identifiers from highest to lowest level. Use <code>~0</code> or <code>~1</code> if there are no clusters.
weight	Column name in <code>data_proj</code> (and <code>data_model</code>) representing the survey weights.
strata	Column name for stratification; use <code>NULL</code> if no strata are used.
domain	Character vector or formula specifying domain variable names. Must be present in <code>data_proj</code> .
summary_function	A function to compute domain-level estimates (default: <code>"mean"</code> , <code>"total"</code> , <code>"variance"</code>).
working_model	A <code>parsnip</code> model object specifying the working model (see <code>@details</code>).
data_model	Data frame (small sample, Survey 2) containing both target and auxiliary variables.
data_proj	Data frame (large sample, Survey 1) containing auxiliary variables and domain variables.
model_metric	A <code>yardstick::metric_set()</code> function, or <code>NULL</code> to use default metrics.
cv_folds	Number of folds for k-fold cross-validation.
tuning_grid	Either a data frame with tuning parameters or a positive integer specifying the number of grid search candidates.
parallel_over	Specifies parallelization mode: <code>"resamples"</code> , <code>"everything"</code> , or <code>NULL</code> .
cores	Integer specifying number of cores for parallel execution. If <code>NULL</code> , defaults to at most 2 cores.
seed	Integer seed for reproducibility.
bias_correction	Logical; if <code>TRUE</code> , applies domain-level bias correction from Survey 2 where available. If <code>FALSE</code> (default), uses the pure model-assisted projection estimator as formulated in Kim & Rao (2012).
return_yhat	Logical; if <code>TRUE</code> , returns survey estimates of the target variable from <code>data_model</code> .
...	Additional arguments passed to svydesign .

Details

The following working models are supported via the **parsnip** interface:

- `linear_reg()` – Linear regression
- `logistic_reg()` – Logistic regression
- `linear_reg(engine = "stan")` – Bayesian linear regression
- `logistic_reg(engine = "stan")` – Bayesian logistic regression
- `poisson_reg()` – Poisson regression
- `decision_tree()` – Decision tree
- `nearest_neighbor()` – k-Nearest Neighbors (k-NN)
- `naive_bayes()` – Naive Bayes classifier
- `mlp()` – Multi-layer perceptron (neural network)
- `svm_linear()` – Support vector machine with linear kernel
- `svm_poly()` – Support vector machine with polynomial kernel
- `svm_rbf()` – Support vector machine with radial basis function (RBF) kernel
- `bag_tree()` – Bagged decision tree
- `bart()` – Bayesian Additive Regression Trees (BART)
- `rand_forest(engine = "ranger")` – Random forest (via ranger)
- `rand_forest(engine = "aorsf")` – Accelerated oblique random forest (AORF; Jaeger et al. 2022, 2024)
- `boost_tree(engine = "lightgbm")` – Gradient boosting (LightGBM)
- `boost_tree(engine = "xgboost")` – Gradient boosting (XGBoost)

For a complete list of supported models and engines, see [Tidy Modeling With R](#).

Value

A list containing:

<code>working_model</code>	The extracted fit engine from the final workflow.
<code>model</code>	Alias for <code>working_model</code> .
<code>prediction</code>	A vector of predicted values or probabilities on <code>data_proj</code> .
<code>projection</code>	A data frame with domain-level estimates. • <code>domain</code> – Domain identifier(s). • <code>ypr</code> – Projection estimator results for each domain. • <code>var_ypr</code> – Estimated variance of the projection estimator. • <code>rse_ypr</code> – Relative standard error (in \
<code>df_result</code>	Alias for <code>projection</code> .
<code>return_yhat</code>	(Optional) Survey estimates of the target variable from <code>data_model</code> , present when <code>return_yhat = TRUE</code> .

References

1. Kim, J. K., & Rao, J. N. (2012). Combining data from two independent surveys: a model-assisted approach. *Biometrika*, 99(1), 85-100.

Examples

```
## Not run:
library(sae.projection)
library(dplyr)
library(bonsai)

df_svy22_income <- df_svy22 %>% filter(!is.na(income))
df_svy23_income <- df_svy23 %>% filter(!is.na(income))

# Linear regression
lm_proj <- ma_projection(
  income ~ age + sex + edu + disability,
  cluster_ids = "PSU", weight = "WEIGHT", strata = "STRATA",
  domain = c("PROV", "REGENCY"),
  working_model = linear_reg(),
  data_model = df_svy22_income,
  data_proj = df_svy23_income,
  nest = TRUE
)

df_svy22_neet <- df_svy22 %>%
  filter(between(age, 15, 24))
df_svy23_neet <- df_svy23 %>%
  filter(between(age, 15, 24))

# Logistic regression
lr_proj <- ma_projection(
  formula = neet ~ sex + edu + disability,
  cluster_ids = "PSU",
  weight = "WEIGHT",
  strata = "STRATA",
  domain = c("PROV", "REGENCY"),
  working_model = logistic_reg(),
  data_model = df_svy22_neet,
  data_proj = df_svy23_neet,
  nest = TRUE
)

## End(Not run)
```

Description

Kim and Rao (2012), the synthetic data obtained through the model-assisted projection method can provide a useful tool for efficient domain estimation when the size of the sample in survey B is much larger than the size of sample in survey A.

The function projects estimated values from a small survey (survey A) onto an independent large survey (survey B) using the random forest classification algorithm. The two surveys are statistically independent, but the projection relies on shared auxiliary variables. The process includes data preprocessing, feature selection, model training, and domain-specific estimation based on survey design principles "two stages one phase". The function automatically selects standard estimation or bias-corrected estimation based on the parameter `bias_correction`.

`bias_correction = TRUE` can only be used if there is `psu`, `ssu`, `strata` on the `data_model`. If it doesn't, then it will automatically be `bias_correction = FALSE`

Usage

```
projection_randomforest(
  data_model,
  target_column,
  predictor_cols,
  data_proj,
  domain1,
  domain2,
  psu,
  ssu = NULL,
  strata = NULL,
  weights,
  split_ratio = 0.8,
  feature_selection = TRUE,
  bias_correction = FALSE
)
```

Arguments

<code>data_model</code>	The training dataset, consisting of auxiliary variables and the target variable.
<code>target_column</code>	The name of the target column in the <code>data_model</code> .
<code>predictor_cols</code>	A vector of predictor column names.
<code>data_proj</code>	The data for projection (prediction), which needs to be projected using the trained model. It must contain the same auxiliary variables as the <code>data_model</code>
<code>domain1</code>	Domain variables for survey estimation (e.g., "province")
<code>domain2</code>	Domain variables for survey estimation (e.g., "regency")
<code>psu</code>	Primary sampling units, representing the structure of the sampling frame.
<code>ssu</code>	Secondary sampling units, representing the structure of the sampling frame (default is <code>NULL</code>).
<code>strata</code>	Stratification variable, ensuring that specific subgroups are represented (default is <code>NULL</code>).

<code>weights</code>	Weights used for the direct estimation from <code>data_model</code> and indirect estimation from <code>data_proj</code> .
<code>split_ratio</code>	Proportion of data used for training (default is 0.8, meaning 80 percent for training and 20 percent for validation).
<code>feature_selection</code>	Selection of predictor variables (default is TRUE)
<code>bias_correction</code>	Logical; if TRUE, then bias correction is applied, if FALSE, then bias correction is not applied. Default is FALSE.

Value

A list containing the following elements:

- `model` The trained Random Forest model.
- `importance` Feature importance showing which features contributed most to the model's predictions.
- `train_accuracy` Accuracy of the model on the training set.
- `validation_accuracy` Accuracy of the model on the validation set.
- `validation_performance` Confusion matrix for the validation set, showing performance metrics like accuracy, precision, recall, etc.
- `data_proj` The projection data with predicted values.

if `bias_correction = FALSE`:

- `Domain1` Estimations for Domain 1, including estimated values, variance, and relative standard error (RSE).
- `Domain2` Estimations for Domain 2, including estimated values, variance, and relative standard error (RSE).

if `bias_correction = TRUE`:

- `Direct` Direct estimations for Domain 1, including estimated values, variance, and relative standard error (RSE).
- `Domain1_corrected_bias` Bias-corrected estimations for Domain 1, including estimated values, variance, and relative standard error (RSE).
- `Domain2_corrected_bias` Bias-corrected estimations for Domain 2, including estimated values, variance, and relative standard error (RSE).

References

1. Kim, J. K., & Rao, J. N. (2012). Combining data from two independent surveys: a model-assisted approach. *Biometrika*, 99(1), 85-100.

Examples

```
library(survey)
library(caret)
library(dplyr)

data_A <- df_svy_A
data_B <- df_svy_B

# Get predictor variables from data_model
x_predictors <- data_A %>% select(5:19) %>% names()

# Run projection_randomforest with bias correction
rf_proj_corrected <- projection_randomforest(
  data_model = data_A,
  target_column = "Y",
  predictor_cols = x_predictors,
  data_proj = data_B,
  domain1 = "province",
  domain2 = "regency",
  psu = "num",
  ssu = NULL,
  strata = NULL,
  weights = "weight",
  feature_selection = TRUE,
  bias_correction = TRUE)

rf_proj_corrected$Direct
rf_proj_corrected$Domain1_corrected_bias
rf_proj_corrected$Domain2_corrected_bias
```

Index

* datasets

df_svy22, [4](#)

df_svy23, [4](#)

df_svy_A, [2](#)

df_svy_B, [3](#)

df_svy22, [4](#)

df_svy23, [4](#)

df_svy_A, [2](#)

df_svy_B, [3](#)

ma_projection, [5](#)

projection_randomforest, [8](#)

svydesign, [6](#)