Package 'sae.projection'

April 5, 2025

Type Package

Title Small Area Estimation Using Model-Assisted Projection Method

Version 0.1.3

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>.

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

RoxygenNote 7.3.2

Depends R (>= 4.3.0), tidymodels

NeedsCompilation no

Author Ridson Al Farizal P [aut, cre, cph] (<https://orcid.org/0000-0003-0617-0214>), Azka Ubaidillah [aut] (<https://orcid.org/0000-0002-3597-0459>), Silvi Ajeng Larasati [aut], Amelia Rahayu [aut]

Maintainer Ridson Al Farizal P <ridsonalfarizal15@gmail.com>

Repository CRAN

Date/Publication 2025-04-05 10:40:02 UTC

Contents

df_survey_A . . . 2
df_survey_B . . . 3
df_svy22 . . . 4
df_svy23 . . . 5
df_svy_A . . . 6
df_svy_B . . . 7
projection . . . 8
projection_randomforest . . . 12
projection_xgboost . . . 15

Index **18**

`df_survey_A` *df_survey_A*

Description

The dataset comes from a large-scale survey conducted at a broader geographic level, corresponding to Domain 1.

Usage

```
df_survey_A
```

Format

A data frame with 5313 rows and 29 variables.

weight Survey weight

psu Primary Sampling Unit

ssu Secondary Sampling Unit

strata Strata used for sampling

ID ID number

no_sample Sample number

no_household Household number

province Province code

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

X16 Predictor variables X16

X17 Predictor variables X17

X18 Predictor variables X18

X19 Predictor variables X19

X20 Predictor variables X20

Y Target variable(1: Yes, 0: No)

`df_survey_B` *df_survey_B*

Description

The dataset comes from a survey conducted at a more specific geographic level, corresponding to Domain 2.

Usage

```
df_survey_B
```

Format

A data frame with 105242 rows and 29 variables.

weight Survey weight

psu Primary Sampling Unit

ssu Secondary Sampling Unit

strata Strata used for sampling

ID ID number

no_sample Sample number

no_household Household number

province Province code

regency Regency or municipality code

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

X16 Predictor variables X16

X17 Predictor variables X17

X18 Predictor variables X18

X19 Predictor variables X19

X20 Predictor variables X20

`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

Source

https://www.bps.go.id

`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

Source

https://www.bps.go.id

df_svy_A *df_svy_A*

Description

A dataset from a survey that was conducted in a certain province within Indonesia, presented only at provincial level (Domain 1).

Usage

```
df_svy_A
```

Format

A data frame with 3655 rows and 33 variables with 6 domains.

ID Unique identifier for each respondent

no_sample Sample number

no_household Household number

no_member Household member number

weight Weight from survey

province Province code

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

X16 Predictor variables X16

X17 Predictor variables X17

X18 Predictor variables X18

X19 Predictor variables X19

X20 Predictor variables X20

X21 Predictor variables X21

X22 Predictor variables X22

X23 Predictor variables X23

X24 Predictor variables X24

X25 Predictor variables X25

X26 Predictor variables X26

Y Target variable (1: Yes, 0: No)

`df_svy_B` *df_svy_B*

Description

A dataset from a survey that was conducted in a certain province within Indonesia, presented at the regency level (Domain 2).

Usage

```
df_svy_B
```

Format

A data frame with 18842 rows and 37 variables with 6 domains.

year Year the survey was conducted

psu Primary Sampling Unit (PSU)

ssu Secondary Sampling Unit (SSU)

strata Strata used for sampling

ID Unique identifier for each respondent

no_sample Sample number

no_household Household number

no_member Household member number

weight Weight from survey

province Province code

regency Regency or municipality code

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

X16 Predictor variables X16

X17 Predictor variables X17

X18 Predictor variables X18

X19 Predictor variables X19

X20 Predictor variables X20

X21 Predictor variables X21

X22 Predictor variables X22

X23 Predictor variables X23

X24 Predictor variables X24

X25 Predictor variables X25

X26 Predictor variables X26

`projection` *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

```
projection(
  formula,
  id,
  weight,
  strata = NULL,
  domain,
  fun = "mean",
  model,
  data_model,
  data_proj,
  model_metric,
  kfold = 3,
  grid = 10,
  parallel_over = "resamples",
  seed = 1,
  est_y = FALSE,
  ...
)
```

Arguments

formula	An object of class formula that contains a description of the model to be fitted. The variables included in the formula must be contained in the data_model dan data_proj.
id	Column name specifying cluster ids from the largest level to the smallest level, where ~0 or ~1 represents a formula indicating the absence of clusters.
weight	Column name in data_proj representing the survey weight.
strata	Column name specifying strata, use NULL for no strata
domain	Column names in data_model and data_proj representing specific domains for which disaggregated data needs to be produced.
fun	A function taking a formula and survey design object as its first two arguments (default = "mean", "total", "varians").
model	The working model to be used in the projection estimator. Refer to the details for the available working models.
data_model	A data frame or a data frame extension (e.g., a tibble) representing the second survey, characterized by a much smaller sample, provides information on both the variable of interest and the auxiliary variables.
data_proj	A data frame or a data frame extension (e.g., a tibble) representing the first survey, characterized by a large sample that collects only auxiliary information or general-purpose variables.
model_metric	A yardstick::metric_set(), or NULL to compute a standard set of metrics (rmse for regression and f1-score for classification).
kfold	The number of partitions of the data set (k-fold cross validation).

grid	A data frame of tuning combinations or a positive integer. The data frame should have columns for each parameter being tuned and rows for tuning parameter candidates. An integer denotes the number of candidate parameter sets to be created automatically.
parallel_over	A single string containing either "resamples" or "everything" describing how to use parallel processing. Alternatively, NULL is allowed, which chooses between "resamples" and "everything" automatically. If "resamples", then tuning will be performed in parallel over resamples alone. Within each resample, the preprocessor (i.e. recipe or formula) is processed once, and is then reused across all models that need to be fit. If "everything", then tuning will be performed in parallel at two levels. An outer parallel loop will iterate over resamples. Additionally, an inner parallel loop will iterate over all unique combinations of preprocessor and model tuning parameters for that specific resample. This will result in the preprocessor being re-processed multiple times, but can be faster if that processing is extremely fast.
seed	A single value, interpreted as an integer
est_y	A logical value indicating whether to return the estimation of y in data_model. If TRUE, the estimation is returned; otherwise, it is not.
...	Further argument to the svydesign.

Details

The available working models include:

- Linear Regression `linear_reg()`
- Logistic Regression `logistic_reg()`
- Poisson Regression `poisson_reg()`
- Decision Tree `decision_tree()`
- KNN `nearest_neighbor()`
- Naive Bayes `naive_bayes()`
- Multi Layer Perceptron `mlp()`
- Accelerated Oblique Random Forests (Jaeger et al. 2022, Jaeger et al. 2024) `rand_forest(engine = 'aorsf')`
- LightGBM `boost_tree(engine = 'lightgbm')`

A complete list of models can be seen at the following link Tidy Modeling With R

Value

The function returns a list with the following objects (`model`, `prediction` and `df_result`): `model` The working model used in the projection. `prediction` A vector containing the prediction results from the working model. `df_result` A data frame with the following columns:

- `domain` The name of the domain.
- `ypr` The estimation results of the projection for each domain.
- `var_ypr` The sample variance of the projection estimator for each domain.
- `rse_ypr` The Relative Standard Error (RSE) in percentage (%).

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 <- projection(
  income ~ age + sex + edu + disability,
  id = "PSU", weight = "WEIGHT", strata = "STRATA",
  domain = c("PROV", "REGENCY"),
  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 <- projection(
  formula = neet ~ sex + edu + disability,
  id = ~ PSU,
  weight = ~ WEIGHT,
  strata = ~ STRATA,
  domain = ~ PROV + REGENCY,
  model = logistic_reg(),
  data_model = df_svy22_neet,
  data_proj = df_svy23_neet,
  nest = TRUE
)

# LightGBM regression with hyperparameter tunning
show_engines("boost_tree")
lgbm_model <- boost_tree(
  mtry = tune(), trees = tune(), min_n = tune(),
  tree_depth = tune(), learn_rate = tune(),
  engine = "lightgbm"
)

lgbm_proj <- projection(
  formula = neet ~ sex + edu + disability,
```

```
  id = "PSU",
  weight = "WEIGHT",
  strata = "STRATA",
  domain = c("PROV", "REGENCY"),
  model = lgbm_model,
  data_model = df_svy22_neet,
  data_proj = df_svy23_neet,
  kfold = 3,
  grid = 10,
  nest = TRUE
)

## End(Not run)
```

`projection_randomforest` *Projection Estimator with Random Forest Algorithm*

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,
  strata,
  weights,
  split_ratio = 0.8,
  metric = "Accuracy",
  bias_correction = FALSE
)
```

Arguments

data_model	The training dataset, consisting of auxiliary variables and the target variable.
target_column	The name of the target column in the data_model.
predictor_cols	A vector of predictor column names.
data_proj	The data for projection (prediction), which needs to be projected using the trained model. It must contain the same auxiliary variables as the data_model
domain1	Domain variables for survey estimation (e.g., "province")
domain2	Domain variables for survey estimation (e.g., "regency")
psu	Primary sampling units, representing the structure of the sampling frame.
ssu	Secondary sampling units, representing the structure of the sampling frame.
strata	Stratification variable, ensuring that specific subgroups are represented.
weights	Weights used for the direct estimation from data_model and indirect estimation from data_proj.
split_ratio	Proportion of data used for training (default is 0.8, meaning 80 percent for training and 20 percent for validation).
metric	The metric used for model evaluation (default is Accuracy, other options include "AUC", "F1", etc.).
bias_correction	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
x_predictors <- data_A %>% select(7:32) %>% names()

# The alternative of calculating "psu, ssu, strata" if not present in the data_model is:
data_A <- data_A %>%
left_join(data_B %>% select(psu, ssu, strata, no_sample, no_household),
          by = c('no_sample', 'no_household'),
          multiple = 'any'
)

# Run projection_randomforest without bias correction
result_standard <- projection_randomforest(
                           data_model = data_A,
                           target_column = "Y",
                           predictor_cols = x_predictors,
                           data_proj = data_B,
                           domain1 = "province",
                           domain2 = "regency",
                           psu = "psu",
                           ssu = "ssu",
                           strata = "strata",
                           weights = "weight",
                           metric = "Accuracy",
                           bias_correction = FALSE)
print(result_standard)

# Run projection_randomforest with bias correction
result_bias_corrected <- projection_randomforest(
                           data_model = data_A,
                           target_column = "Y",
                           predictor_cols = x_predictors,
                           data_proj = data_B,
                           domain1 = "province",
                           domain2 = "regency",
                           psu = "psu",
```

```
                          ssu = "ssu",
                          strata = "strata",
                          weights = "weight",
                          metric = "Accuracy",
                          bias_correction = TRUE)
print(result_bias_corrected)
```

`projection_xgboost` *Projection Estimator*

Description

Kim and Rao (2012), proposed a model-assisted projection estimation method for two independent surveys, where the first survey (**A1**) has a large sample that only collects auxiliary variables, while the second survey (**A1**) has a smaller sample but contains information on both the focal variable and auxiliary variables. This method uses a **Working Model (WM)** to relate the focal variable to the auxiliary variable based on data from **A2**, and then predicts the value of the focal variable for **A1**. A projection estimator is then obtained from the (**A2**) sample using the resulting synthetic values. This approach produces estimators that are asymptotically unbiased and can improve the efficiency of domain estimation, especially when the sample size in survey 1 is much larger compared to survey 2.

This function applies the XGBoost algorithm to project estimated values from a small survey onto an independent larger survey. While the two surveys are statistically independent, the projection is based on common auxiliary variables. The process in this function involves data preprocessing, feature selection, getting the best model with hyperparameter tuning, and performing domain-specific estimation following survey design principles.

Usage

```
projection_xgboost(
  target_col,
  data_model,
  data_proj,
  id,
  STRATA = NULL,
  domain1,
  domain2,
  weight,
  task_type,
  test_size = 0.2,
  nfold = 5,
  corrected_bias = FALSE
)
```

Arguments

target_col	The name of the column that contains the target variable in the data_model.
data_model	A data frame or a data frame extension (e.g., a tibble) representing the training dataset, which consists of auxiliary variables and the target variable. This dataset is characterized by a smaller sample size and provides information on both the variable of interest and the auxiliary variables.
data_proj	A data frame or a data frame extension (e.g., a tibble) representing the projection dataset, which is characterized by a larger sample size that collects only auxiliary information or general-purpose variables. This dataset must contain the same auxiliary variables as the data_model and is used for making predictions based on the trained model.
id	Column name specifying cluster ids from the largest level to the smallest level, where ~0 or ~1 represents a formula indicating the absence of clusters.
STRATA	The name of the column that specifies the strata; set to NULL if no stratification is required.#' @param test_size Proportion of data used for training (default is 0.8, meaning 80% for training and 20% for validation).
domain1	Domain variables for higher-level survey estimation. (e.g., "province")
domain2	Domain variables for more granular survey estimation at a lower administrative level. (e.g., "regency")
weight	The name of the column in data_proj that represents the survey weight, usually used for the purpose of indirect estimation .
task_type	A string that specifies the modeling objective, indicating whether the task is for classification or regression. Use "classification" for tasks where the goal is to categorize data into discrete classes, such as predicting whether an email is spam or not. Use "regression" for tasks where the goal is to predict a continuous outcome, such as forecasting sales revenue or predicting house prices.
test_size	The proportion of data used for testing, with the remaining data used for training.
nfold	The number of data partitions used for cross-validation (n-fold validation).
corrected_bias	A logical value indicating whether to apply bias correction to the estimation results from the modeling process. When set to TRUE, this parameter ensures that the estimates are adjusted to account for any systematic biases, leading to more accurate and reliable predictions.

Value

The function returns a list containing: params_used The hyperparameters and settings of the trained model used for projection. nfeatures The number of features selected and used in the trained model for projection. domain1_estimation Projected estimation results for each domain 1, example province, including :

- Estimation Estimated values.
- RSE Relative Standard Error.
- var Variance of the estimation.

domain2_estimation Projected estimation results for each domain2, example regency (kabupaten/kota), including :

- `Estimation` Estimated values.
- `RSE` Relative Standard Error.
- `var` Variance of the estimation.

If `task_type` is `"classification"`: `mean_train_accuracy` Mean training accuracy. `final_accuracy` Final model accuracy on the test data. `confusion_matrix` Confusion matrix of the classification model.

If `task_type` is `"regression"`: `mean_train_rmse` Mean training RMSE. `final_rmse` Final RMSE on the test data.

If `corrected_bias` is `TRUE`: `direct_estimation` Direct estimation before bias correction. `koreksi_bias_domain1` Bias-corrected estimation for provinces. `koreksi_bias_domain2` Bias-corrected estimation for regencies.

References

1. Kim, J. K., & Rao, J. N. (2012). Combining data from two independent surveys: a model-assisted approach. Biometrika, 99(1), 85-100.
2. 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 1 is much larger than the size of sample in survey 2.

Examples

```
library(xgboost)
library(caret)
library(FSelector)
library(glmnet)
library(recipes)

Data_A <- df_survey_A
Data_B <- df_survey_B

hasil <- projection_xgboost(
                            target_col = "Y",
                            data_model = Data_A,
                            data_proj = Data_B,
                            id = "psu+ssu",
                            STRATA = "strata",
                            domain1 = "province",
                            domain2 = "regency",
                            weight = "weight",
                            task_type = "classification")
```

Index

∗ **datasets**
df_survey_A, 2
df_survey_B, 3
df_svy22, 4
df_svy23, 5
df_svy_A, 6
df_svy_B, 7

df_survey_A, 2
df_survey_B, 3
df_svy22, 4
df_svy23, 5
df_svy_A, 6
df_svy_B, 7

projection, 8
projection_randomforest, 12
projection_xgboost, 15

svydesign, *10*