Package ‘sae.projection’

February 18, 2025

Type Package

Title Small Area Estimation Using Model-Assisted Projection Method

Version 0.1.2

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

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]

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

Repository CRAN

Date/Publication 2025-02-18 08:20:02 UTC

Contents

df_susenas_mar2020 2
df_susenas_sep2020 3
df_svy22 5
df_svy23 6
projection 7
proj_random_forest 11

Index **14**

`df_susenas_mar2020` *df_susenas_mar2020: Maret 2020 National Socio-Economic Survey (Susenas) Dataset for DKI Jakarta, Indonesia*

Description

A dataset from the March 2020 National Socio-Economic Survey (Susenas) KOR Module, conducted in DKI Jakarta, Indonesia, which is held annually, presented at the regency level.

Usage

```
df_susenas_mar2020
```

Format

A data frame with 18842 rows and 38 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

urban_rural Urban or rural classification (1: Urban, 2: Rural)

marital_status Marital status (1: Married, 0: Other)

sex Sex (1: Male, 2: Female)

age Age of the respondent

attending_school Currently attending school (0: No, 1: Yes)

highest_edu Highest education completed (0: Did not complete elementary school, 1: Elementary school, 2: Junior high school, 3: Senior high school, 4: University/College)

job_status Employment status (1: Employed, 0: Not employed)

sector_type Type of employment sector (1: Agriculture, 0: Non-agriculture)

job_position Job position or role

building_ownership Ownership status of residence (1: Owned, 0: Other)

floor_area Floor area of residence (in square meters)

pension_ins Has pension insurance (0: No, 1: Yes)

old_age_ins Has old-age insurance (0: No, 1: Yes)

work_ins Has work insurance (0: No, 1: Yes)

life_ins Has life insurance (0: No, 1: Yes)

severance_pay Receives severance pay (0: No, 1: Yes)

kks_card Has a KKS (Kartu Keluarga Sejahtera) card (0: No, 1: Yes)

pkh_recipient Is the respondent a recipient of PKH (Program Keluarga Harapan) assistance? (0: No, 1: Yes)

pkh_disbursement Location where PKH funds are disbursed

pkh_food PKH funds used for food assistance (0: No, 1: Yes)

pkh_housing PKH funds used for housing assistance (0: No, 1: Yes)

pkh_healthcare PKH funds used for healthcare assistance (0: No, 1: Yes)

pkh_maternity PKH funds used for maternity assistance (0: No, 1: Yes)

pkh_school PKH funds used for school assistance (0: No, 1: Yes)

pkh_other PKH funds used for other types of assistance (0: No, 1: Yes)

bpnt_program Receives BPNT (Bantuan Pangan Non-Tunai) program assistance (0: No, 1: Yes)

Source

https://www.bps.go.id

`df_susenas_sep2020` *df_susenas_sep2020: September 2020 National Socio-Economic Survey (Susenas) Dataset for DKI Jakarta, Indonesia*

Description

A dataset from the September 2020 National Socio-Economic Survey (Susenas) Social Resilience Module, conducted in DKI Jakarta, Indonesia, which is held every three years, presented at the provincial level.

Usage

```
df_susenas_sep2020
```

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

urban_rural Urban or rural classification (1: Urban, 2: Rural)

marital_status Marital status (1: Married, 0: Other)

sex Sex (1: Male, 2: Female)

age Age of the respondent

attending_school Currently attending school (0: No, 1: Yes)

highest_edu Highest education completed (0: Did not complete elementary school, 1: Elementary school, 2: Junior high school, 3: Senior high school, 4: University/College)

job_status Employment status (1: Employed, 0: Not employed)

sector_type Type of employment sector (1: Agriculture, 0: Non-agriculture)

job_position Job position or role

building_ownership Ownership status of residence (1: Owned, 0: Other)

floor_area Floor area of residence (in square meters)

pension_ins Has pension insurance (0: No, 1: Yes)

old_age_ins Has old-age insurance (0: No, 1: Yes)

work_ins Has work insurance (0: No, 1: Yes)

life_ins Has life insurance (0: No, 1: Yes)

severance_pay Receives severance pay (0: No, 1: Yes)

kks_card Has a KKS (Kartu Keluarga Sejahtera) card (0: No, 1: Yes)

pkh_recipient Is the respondent a recipient of PKH (Program Keluarga Harapan) assistance? (0: No, 1: Yes)

pkh_disbursement Location where PKH funds are disbursed

pkh_food PKH funds used for food assistance (0: No, 1: Yes)

pkh_housing PKH funds used for housing assistance (0: No, 1: Yes)

pkh_healthcare PKH funds used for healthcare assistance (0: No, 1: Yes)

pkh_maternity PKH funds used for maternity assistance (0: No, 1: Yes)

pkh_school PKH funds used for school assistance (0: No, 1: Yes)

pkh_other PKH funds used for other types of assistance (0: No, 1: Yes)

bpnt_program Receives BPNT (Bantuan Pangan Non-Tunai) program assistance (0: No, 1: Yes)

uses_public_transport Using public transportation (0: No, 1: Yes), which includes motorized vehicles with specific routes

Source

https://www.bps.go.id

df_svy22 *df_svy22: August 2022 National Labor Force Survey Dataset for East Java, Indonesia.*

Description

A dataset from the August 2022 National Labor Force Survey (Sakernas) conducted in East Java, Indonesia.

Usage

```
df_svy22
```

Format

A data frame with 74.070 rows and 11 variables with 38 domains.

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: August 2023 National Labor Force Survey Dataset for East Java, Indonesia.*

Description

A dataset from the August 2023 National Labor Force Survey (Sakernas) conducted in East Java, Indonesia.

Usage

```
df_svy23
```

Format

A data frame with 66.245 rows and 11 variables with 38 domains.

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

`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()`
- Random Forest `rand_forest()`
- Accelerated Oblique Random Forests (Jaeger et al. 2022, Jaeger et al. 2024) `rand_forest(engine = 'aorsf')`
- XGBoost `boost_tree(engine = 'xgboost')`
- 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
)

# Random forest regression with hyperparameter tunning
```

```
rf_proj <- projection(
  income ~ age + sex + edu + disability,
  id = "PSU", weight = "WEIGHT", strata = "STRATA",
  domain = c("PROV", "REGENCY"),
  model = rand_forest(mtry = tune(), trees = tune(), min_n = tune()),
  data_model = df_svy22_income,
  data_proj = df_svy23_income,
  kfold = 3,
  grid = 10,
  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)
```

`proj_random_forest` *proj_random_forest*

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 2 is much larger than the size of sample in survey 1.

This function projects estimated values from a small survey onto an independent large survey using the random forest 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. The function automatically selects standard estimation or bias-corrected estimation based on the parameter `bias_correction`.

Usage

```
proj_random_forest(
  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 in the for 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`, applies bias correction using `projection_rf_CorrectedBias`. Default is `FALSE`.

Value

The output of either `projection_rf` or `projection_rf_CorrectedBias`, depending on `bias_correction`.

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)

df_susenas_sep2020 <- df_susenas_sep2020 %>%
left_join(df_susenas_mar2020 %>% select(psu, ssu, strata, no_sample, no_household),
          by = c('no_sample', 'no_household'),
          multiple = 'any'
)

df_sep20 <- df_susenas_sep2020
df_mar20 <- df_susenas_mar2020
x_predictors <- df_sep20 %>% select(7:32) %>% names()

# Run projection_random_forest without bias correction
result_standard <- proj_random_forest(
                          data_model = df_sep20,
                          target_column = "uses_public_transport",
                          predictor_cols = x_predictors,
                          data_proj = df_mar20,
                          domain1 = "province",
                          domain2 = "regency",
                          psu = "psu",
                          ssu = "ssu",
                          strata = "strata",
                          weights = "weight",
                          metric = "Accuracy",
                          bias_correction = FALSE)
```

```
print(result_standard)

# Run projection_random_forest with bias correction
result_bias_corrected <- proj_random_forest(
                          data_model = df_sep20,
                          target_column = "uses_public_transport",
                          predictor_cols = x_predictors,
                          data_proj = df_mar20,
                          domain1 = "province",
                          domain2 = "regency",
                          psu = "psu",
                          ssu = "ssu",
                          strata = "strata",
                          weights = "weight",
                          metric = "Accuracy",
                          bias_correction = TRUE)
print(result_bias_corrected)
```

Index

∗ **datasets**
    df_susenas_mar2020, 2
    df_susenas_sep2020, 3
    df_svy22, 5
    df_svy23, 6

df_susenas_mar2020, 2
df_susenas_sep2020, 3
df_svy22, 5
df_svy23, 6

proj_random_forest, 11
projection, 7

svydesign, *8*