Package: dsPUcopula (via r-universe)

May 21, 2026

Type Package

Title DataSHIELD Server Functions for Partition-of-Unity Copula Based Synthesis

Version 0.1.0

Description Implements the server-side components required to generate privacy-protecting synthetic data in the DataSHIELD infrastructure. The package orchestrates the preprocessing, copula fitting, synthetic data generation and privacy scoring workflows by combining R and Python tooling.

License MIT + file LICENSE

URL https://github.com/amaendle/dsPUcopula

BugReports https://github.com/amaendle/dsPUcopula/issues

Depends R (>= 4.1.0)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports caret, dplyr, dsBase, logspline, magrittr, PUcopula, RANN, reticulate, tidyselect

Remotes amaendle/PUcopula

Additional_repositories https://amaendle.r-universe.dev

Config/reticulate list(packages = list(list(package = ``anonymeter''), list(package = ``syndat'')))

Config/pak/sysreqs
cmake make libicu-dev libpng-dev libuv1-dev libssl-dev python3 libx11-dev zlib1g-dev

Repository https://amaendle.r-universe.dev

Date/Publication 2026-04-13 08:08:58 UTC

RemoteUrl https://github.com/amaendle/dsPUcopula

RemoteRef HEAD

RemoteSha ce5cc23fb917ab0eaf7144d1ec8ade67d37ec045

Contents

`estimateMarginalsDS` *Estimate marginal distributions for the fitted copula*

Description

Given the preprocessed data this function fits marginal models for each column. Continuous variables are modelled using `logspline` densities and discrete variables through empirical probability tables.

Usage

```
estimateMarginalsDS(data_str, method = "spline", k = 3)
```

Arguments

`data_str`	Character name of the processed data object on the server.
`method`	Character vector controlling the estimation method for numeric and ordered categorical variables. Currently only "spline" is supported for numeric variables.
`k`	Numeric or list specifying the smoothing neighbourhood used before fitting the marginals.

Value

A named list of marginal models compatible with `generateSyntheticDS()`.

`fitPUcopulaDS` *Fit the PU copula model on the server side*

Description

`fitPUcopulaDS()` is the main server-side entry point used by DataSHIELD to build the Partition-of-Unity (PU) copula-based synthesiser. It evaluates the symbol provided by the client, optionally applies jittering and binning to satisfy disclosure control constraints, and fits a `PUcopula::PUCopula` model describing the dependence structure.

Usage

```
fitPUcopulaDS(
  data_str,
  driver_strength_factor = 0.5,
  bin_size = 3,
  jitter = FALSE,
  family = "binom"
)
```

Arguments

`data_str`	Character name of the processed server-side training data object.
`driver_strength_factor`	Numeric scalar or vector controlling the driver strength passed to `PUcopula::PUCopula()`. Values between 0 and 1 are interpreted as proportions of available rows.
`bin_size`	Numeric or list specifying the smoothing bin size applied to the ranks of each variable. When a single value is provided it is recycled across variables.
`jitter`	Logical, numeric or named list controlling the amount of numeric jittering applied to the input columns.
`family`	Character string selecting the driver distribution passed to `PUcopula::PUCopula()`.

Details

This function represents the first step of the synthetic data generation workflow. The fitted copula can later be combined with separately estimated marginal distributions (see `estimateMarginalsDS()`) to simulate synthetic data.

The function operates on server-side data within a DataSHIELD environment. Prior to fitting, optional smoothing (via `bin_size`) and perturbation (via `jitter`) can be applied to reduce disclosure risks.

Only the copula model is estimated here. Marginal distributions must be fitted separately using `estimateMarginalsDS()`. Synthetic data can then be generated using `generateSyntheticDS()`, optionally including privacy and utility scores.

Value

A fitted PUcopula::PUCopula object representing the dependence structure of the data.

See Also

estimateMarginalsDS(), simulateCopulaDS(), generateSyntheticDS(), PUcopula::PUCopula()

---

generateSyntheticDS *Generate synthetic data and privacy scores*

Description

Combines the fitted copula and marginal models to draw synthetic data. The helper also orchestrates the privacy evaluation workflow by calling the Python-based scoring helpers when requested.

Usage

```
generateSyntheticDS(
  n = "n_rSynthetic",
  copula_str = "PU_copula_model",
  marginals_str = "marginal_models",
  training_data = "D_ori",
  control_data = "D_control",
  singling_out_check = TRUE,
  inference_check = TRUE,
  inference_check_ignore_na = FALSE,
  syndat_scores = TRUE,
  return_scores = FALSE
)
```

Arguments

n	Number of records to generate or a character name pointing to an object containing that number.
copula_str	Character name of the fitted copula object.
marginals_str	Character name of the list of fitted marginal models.
training_data	Character name of the training data set used to fit the synthesiser.
control_data	Character name of the hold-out control data set.
singling_out_check	Logical; when TRUE the Anonymeter singling out risk is evaluated.
inference_check	Logical; when TRUE inference attacks are assessed.
inference_check_ignore_na	Logical; passed to py_anonymeter_Inference() to control the removal of missing values.

syndat_scores	Logical; evaluate utility scores using py_syndat_scores().
return_scores	Logical; when TRUE the function returns both the synthetic data and the computed score objects.

Value

A synthetic data.frame. When return_scores = TRUE a list with the synthetic data and the associated privacy/utility scores.

---

postprocessDataDS *Restore the original factor structure after synthesis*

Description

Undo the dummy encoding produced by preprocessDataDS() by projecting the synthetic dummy variables back to the closest original factor levels.

Usage

```
postprocessDataDS(data_str, cat_dummy_levels_str)
```

Arguments

data_str	Character name of the data object that contains the dummy encoded variables.
cat_dummy_levels_str	Character name of the metadata object storing the dummy variable structure returned by preprocessDataDS().

Value

A data.frame whose factor variables are restored to their original levels and ordering.

---

preprocess_helpers *Helpers for preserving and restoring server-side metadata*

Description

Functions that capture the original structure of the training data, prepare it for copula fitting and later restore the factor levels of the synthetic outputs.

Arguments

data_str	Character name of the data object stored on the DataSHIELD server.
cat_dummy_levels_str	Character name of the metadata object returned by preprocessDataDS().

Value

`save_original_varnamesDS()` returns a character vector of column names.

`save_original_classesDS()` returns the associated classes.

`preprocessDataDS()` returns a list containing the processed data set and metadata describing factor levels.

`postprocessDataDS()` returns a `data.frame` with factor variables reconstructed from dummy encodings.

`preprocessDataDS` *Prepare server-side data prior to copula fitting*

Description

This helper applies a set of preprocessing steps. It converts categorical variables into dummy variables, stores the original levels and renames factor columns so that post-processing can restore the input structure.

Usage

```
preprocessDataDS(data_str)
```

Arguments

`data_str`	Character name of the data object on the server.

Value

A list with the processed data (`data`) and the `original_levels` metadata required by `postprocessDataDS()`.

`py_anonymeter_Inference`
Run the Anonymeter inference attack evaluation

Description

The inference attack estimates the risk of inferring secret attributes from synthetic records using auxiliary quasi-identifiers. This function acts as a thin wrapper around the Anonymeter Python API.

Usage

```
py_anonymeter_Inference(
  ori,
  syn,
  aux_cols,
  secret,
  inference_check_ignore_na = FALSE,
  control = NULL,
  return_evaluator = FALSE
)
```

Arguments

ori	The training data set used to fit the synthesis model.
syn	The synthetic data set to be assessed.
aux_cols	Character vector naming the auxiliary columns used by the attacker.
secret	Character scalar giving the sensitive column for which the disclosure risk should be evaluated.
inference_check_ignore_na	Logical; when TRUE observations with missing secrets are dropped from both the original and the control data before running the attack.
control	Optional control data set used to benchmark the attack.
return_evaluator	Logical; when TRUE the evaluator object provided by Anonymeter is returned.

Value

Either the list of risk metrics returned by Anonymeter or the evaluator object when `return_evaluator = TRUE`.

---

py_anonymeter_SinglingOut

Run the Anonymeter singling out attack evaluation

Description

This function calls the Anonymeter Python package to quantify the singling out risk of synthetic data. It is primarily used on the DataSHIELD server to evaluate privacy risks before releasing data to the client.

Usage

```
py_anonymeter_SinglingOut(ori, syn, control = NULL, return_evaluator = FALSE)
```

Arguments

`ori`	The training data set used to fit the synthesis model.
`syn`	The synthetic data set to be assessed.
`control`	An optional control data set. When supplied the control population is used as a baseline for the attack simulation.
`return_evaluator`	Logical; when `TRUE` the underlying Python evaluator object is returned instead of its summary risk values.

Value

Either a list with risk metrics (the default) or the evaluator object returned by Anonymeter when `return_evaluator = TRUE`.

`py_syndat_scores` *Compute utility scores using the* `syndat` *Python module*

Description

The function bridges to the `syndat` Python package to evaluate different quality metrics between an original and a synthetic data set.

Usage

```
py_syndat_scores(ori, syn, control = NULL)
```

Arguments

`ori`	A `data.frame` or matrix with the original data used for training.
`syn`	A `data.frame` or matrix with the generated synthetic data.
`control`	A `data.frame` providing hold-out control data against which the synthetic data are compared.

Value

A named list with distribution, discrimination and correlation scores for both the original and the control data.

save_original_classesDS

Capture the original server-side classes

Description

Similar to save_original_varnamesDS(), this helper stores the original column classes so that the synthetic output can be coerced back to the same types once post-processing is completed.

Usage

```
save_original_classesDS(data_str)
```

Arguments

data_str	Character name of the data object on the server.

Value

A character vector describing the classes of the input columns.

save_original_varnamesDS

Capture the original server-side variable names

Description

Helper used in the DataSHIELD workflow to store the original variable names before preprocessing steps modify them.

Usage

```
save_original_varnamesDS(data_str)
```

Arguments

data_str	Character name of the data object on the server.

Value

A character vector with the column names of the original data.

setup_python *Configure the Python environment used by dsPUcopula*

Description

This helper creates or updates the Python environment declared in the package's `Config/reticulate` field so that the Python packages required for the disclosure control checks are available.

Usage

```
setup_python()
```

Value

Invisibly returns `TRUE` when the environment has been configured.

See Also

`reticulate::configure_environment()`

simulateCopulaDS *Draw samples from the fitted PUcopula model*

Description

Generates random variates from the server-side copula model previously stored in `PU_copula_model`.

Usage

```
simulateCopulaDS(n)
```

Arguments

`n`	Number of samples to generate.

Value

A matrix of simulated copula draws.

Index