Package ‘CausalGPS’

September 30, 2023

Type Package

Title Matching on Generalized Propensity Scores with Continuous Exposures

Version 0.4.1

Maintainer Naeem Khoshnevis <nkhoshnevis@g.harvard.edu>


License GPL-3

Language en-US

URL https://github.com/NSAPH-Software/CausalGPS

BugReports https://github.com/NSAPH-Software/CausalGPS/issues

Copyright Harvard University

Imports parallel, data.table, SuperLearner, xgboost, gam, MASS, polycor, wCorr, stats, ggplot2, rlang, logger, Rcpp, gnm, locpol, Ecume, KernSmooth, cowplot

Encoding UTF-8

RoxygenNote 7.2.3

Suggests covr, knitr, rmarkdown, ranger, earth, testthat, gridExtra

VignetteBuilder knitr

Depends R (>= 3.5.0)

LinkingTo Rcpp

NeedsCompilation yes

Author Naeem Khoshnevis [aut, cre] (<https://orcid.org/0000-0003-4315-1426>, HUIT), Xiao Wu [aut] (<https://orcid.org/0000-0002-4884-657X>, HSPH), Danielle Braun [aut] (<https://orcid.org/0000-0002-5177-8598>, HSPH)
CausalGPS-package

Repository CRAN
Date/Publication 2023-09-29 23:52:39 UTC

R topics documented:

CausalGPS-package .................................................. 2
absolute_corr_fun ................................................. 3
absolute_weighted_corr_fun ................................. 4
check_covar_balance ............................................ 5
compile_pseudo_pop ............................................. 6
estimate_gps ......................................................... 8
estimate_npmetric_erf ......................................... 9
estimate_pmetric_erf .......................................... 11
estimate_semipmetric_erf .................................... 12
generate_pseudo_pop ........................................... 13
generate_syn_data ............................................... 16
get_logger .......................................................... 17
plot.gpsm_erf ..................................................... 18
plot.gpsm_pspop ................................................ 18
print.gpsm_erf .................................................... 19
print.gpsm_pspop ............................................... 20
set_logger ......................................................... 20
summary.gpsm_erf ............................................... 21
summary.gpsm_pspop .......................................... 21
synthetic_us_2010 ................................................. 22

Index 25

CausalGPS-package The 'CausalGPS' package.

Description
An R package for implementing matching and weighting on generalized propensity scores with continuous exposures.

Details
We developed an innovative approach for estimating causal effects using observational data in settings with continuous exposures, and introduce a new framework for GPS caliper matching.

Author(s)
Naeem Khoshnevis
Xiao Wu
Danielle Braun
absolute_corr_fun

**References**


---

**absolute_corr_fun**  
*Check covariate balance using absolute approach*

**Description**

Checks covariate balance based on absolute correlations for given data sets.

**Usage**

```r
absolute_corr_fun(w, c)
```

**Arguments**

- `w` A vector of observed continuous exposure variable.
- `c` A data.frame of observed covariates variable.

**Value**

The function returns a list including:

- `absolute_corr`: the absolute correlations for each pre-exposure covariates;
- `mean_absolute_corr`: the average absolute correlations for all pre-exposure covariates.

**Examples**

```r
set.seed(291)
n <- 100
mydata <- generate_syn_data(sample_size=100)
year <- sample(x=c("2001","2002","2003","2004","2005"),size = n,
    replace = TRUE)
region <- sample(x=c("North","South", "East", "West"),size = n,
    replace = TRUE)
mydata$year <- as.factor(year)
mydata$region <- as.factor(region)
mydata$cf5 <- as.factor(mydata$cf5)
cor_val <- absolute_corr_fun(mydata[,2], mydata[, 3:length(mydata)])
print(cor_val$mean_absolute_corr)
```
absolute_weighted_corr_fun

Check Weighted Covariate Balance Using Absolute Approach

Description

Checks covariate balance based on absolute weighted correlations for given data sets.

Usage

absolute_weighted_corr_fun(w, vw, c)

Arguments

w A vector of observed continuous exposure variable.
vw A vector of weights.
c A data.table of observed covariates variable.

Value

The function returns a list saved the measure related to covariate balance absolute_corr: the absolute correlations for each pre-exposure covairates; mean_absolute_corr: the average absolute correlations for all pre-exposure covairates.

Examples

set.seed(639)
n <- 100
mydata <- generate_syn_data(sample_size=100)
region <- sample(x=c("North", "South", "East", "West"), size = n, replace = TRUE)
mydata$year <- as.factor(year)
mydata$region <- as.factor(region)
mydata$cf5 <- as.factor(mydata$cf5)
cor_val <- absolute_weighted_corr_fun(mydata[,2], runif(n),
                                                mydata[, 3:length(mydata)])
print(cor_val$mean_absolute_corr)
**check_covar_balance**  
*Check covariate balance*

**Description**

Checks the covariate balance of original population or pseudo population.

**Usage**

```r
check_covar_balance(w, c, ci_appr, counter_weight = NULL, nthread = 1, ...)
```

**Arguments**

- `w`: A vector of observed continuous exposure variable.
- `c`: A `data.frame` of observed covariates variable.
- `ci_appr`: The causal inference approach.
- `counter_weight`: A weight vector in different situations. If the matching approach is selected, it is an integer `data.table` of counters. In the case of the weighting approach, it is `weight data.table`.
- `nthread`: The number of available threads.
- `...`: Additional arguments passed to different models.

**Details**

**Additional parameters:**

- For `ci_appr == matching`:
  - `covar_bl_method`
  - `covar_bl_trs`

**Value**

output object:

- `corr_results`
  - `absolute_corr`
  - `mean_absolute_corr`
- `pass` (TRUE,FALSE)
Examples

```r
set.seed(422)
n <- 100
mydata <- generate_syn_data(sample_size=100)
year <- sample(x=c("2001", "2002", "2003", "2004", "2005"), size = n,
        replace = TRUE)
region <- sample(x=c("North", "South", "East", "West"), size = n,
        replace = TRUE)
mydata$year <- as.factor(year)
mydata$region <- as.factor(region)
mydata.cf5 <- as.factor(mydata.cf5)

pseudo_pop <- generate_pseudo_pop(mydata[, c("id", "w")],
        mydata[, c("id", "cf1", "cf2", "cf3",
        "cf4", "cf5", "cf6", "year",
        "region")],
        ci_appr = "matching",
        gps_density = "kernel",
        exposure_trim_qtls = c(0.01, 0.99),
        sl_lib = c("m_xgboost"),
        covar_bl_method = "absolute",
        covar_bl_trs = 0.1,
        covar_bl_trs_type = "mean",
        max_attempt = 1,
        dist_measure = "l1",
        delta_n = 1,
        scale = 0.5,
        nthread = 1)

adjusted_corr_obj <- check_covar_balance(w = pseudo_pop$pseudo_pop[, c("w")],
        c = pseudo_pop$pseudo_pop[, pseudo_pop$covariate_cols_name],
        counter = pseudo_pop$pseudo_pop[, c("counter_weight")],
        ci_appr = "matching",
        nthread = 1,
        covar_bl_method = "absolute",
        covar_bl_trs = 0.1,
        covar_bl_trs_type = "mean")
```

**compile_pseudo_pop**

**Compile pseudo population**

**Description**

Compiles pseudo population based on the original population and estimated GPS value.
**compile_pseudo_pop**

**Usage**

```r
compile_pseudo_pop(
  data_obj,
  ci_appr,
  gps_density,
  bin_seq,
  exposure_col_name,
  nthread,
  ...
)
```

**Arguments**

- **data_obj**: A S3 object including the following:
  - Original data set + GPS values
  - `e_gps_pred`
  - `e_gps_std_pred`
  - `w_resid`
  - `gps_mx` (min and max of gps)
  - `w_mx` (min and max of w).
- **ci_appr**: Causal inference approach.
- **gps_density**: Model type which is used for estimating GPS value, including `normal` and `kernel`.
- **bin_seq**: Sequence of w (treatment) to generate pseudo population. If NULL is passed the default value will be used, which is `seq(min(w)+delta_n/2,max(w), by=delta_n)`.
- **exposure_col_name**: Exposure data column name.
- **nthread**: An integer value that represents the number of threads to be used by internal packages.
- **...**: Additional parameters.

**Value**

`compile_pseudo_pop` returns the pseudo population data that is compiled based on the selected causal inference approach.

**Examples**

```r
set.seed(112)
md <- generate_syn_data(sample_size = 100)
data_with_gps <- estimate_gps(md[, c("id", "w")],
  md[, c("id", "cf1","cf2","cf3","cf4","cf5","cf6")],
gps_density = "normal",
params = list(xgb_max_depth = c(3,4,5),
  xgb_nrounds=c(10,20,30,40,50,60)),
```
estimate_gps

Estimate generalized propensity score (GPS) values

Description

Estimates GPS value for each observation using normal or kernel approaches.

Usage

```r
estimate_gps(
  w,
  c,
  gps_density = "normal",
  params = list(),
  sl_lib = c("m_xgboost"),
  nthread = 1,
  ...
)
```

Arguments

- `w`: A data frame of observed continuous exposure variable. Including id and w columns.
- `c`: A data frame of observed covariates variable. Also includes id column.
- `gps_density`: Model type which is used for estimating GPS value, including normal (default) and kernel.
- `params`: Includes list of parameters that are used internally. Unrelated parameters will be ignored.

```r
pd <- compile_pseudo_pop(data_obj = data_with_gps,
                          ci_appr = "matching",
                          gps_density = "normal",
                          bin_seq = NULL,
                          exposure_col_name = c("w"),
                          nthread = 1,
                          dist_measure = "l1",
                          covar_bl_method = 'absolute',
                          covar_bl_trs = 0.1,
                          covar_bl_trs_type= "mean",
                          delta_n = 0.5,
                          scale = 1)
```
**sl_lib**
A vector of prediction algorithms.

**nthread**
An integer value that represents the number threads to be used in a shared memory system.

... Additional arguments passed to the model.

**Value**

The function returns a S3 object. Including the following:

- **dataset**: `id, w, gps`
- **e_gps_pred**
- **e_gps_std_pred**
- **w_resid**
- **gps_mx (min and max of gps)**
- **w_mx (min and max of w)**
- **used_params**

**Note**

If `internal.use` is set to be FALSE, only original data set + GPS will be returned.

The outcome variable is not used in estimating the GPS value. However, it is used in compiling the data set with GPS values.

**Examples**

```r
m_d <- generate_syn_data(sample_size = 100)
data_with_gps <- estimate_gps(m_d[, c("id", "w")],
    m_d[, c("id", "cf1", "cf2", "cf3",
    "cf4", "cf5", "cf6")],
    gps_density = "normal",
    params = list(xgb_max_depth = c(3,4,5),
    xgb_nrounds=c(10,20,30,40,50,60)),
    nthread = 1,
    sl_lib = c("m_xgboost")
)
```

**estimate_npmetric_erf**

Estimate smoothed exposure-response function (ERF) for pseudo population

**Description**

Estimate smoothed exposure-response function (ERF) for matched and weighted data set using non-parametric models.
estimate_npmetric_erf

Usage

estimate_npmetric_erf(
  m_Y,
  m_w,
  counter_weight,
  bw_seq,
  w_vals,
  nthread,
  kernel_appr = "locpol"
)

Arguments

- **m_Y**: A vector of outcome variable in the matched set.
- **m_w**: A vector of continuous exposure variable in the matched set.
- **counter_weight**: A vector of counter or weight variable in the matched set.
- **bw_seq**: A vector of bandwidth values.
- **w_vals**: A vector of values that you want to calculate the values of the ERF at.
- **nthread**: The number of available cores.
- **kernel_appr**: Internal kernel approach. Available options are `locpol` and `kernsmooth`.

Details

Estimate Functions Using Local Polynomial kernel regression.

Value

The function returns a gpsm_erf object. The object includes the following attributes:

- **params**
- **m_Y**
- **m_w**
- **bw_seq**
- **w_vals**
- **erf**
- **fcall**

Examples

```r
set.seed(697)
m_d <- generate_syn_data(sample_size = 200)
pseudo_pop <- generate_pseudo_pop(m_d[, c("id", "w")],
                                  m_d[, c("id", "cf1","cf2","cf3",
                                       "cf4","cf5","cf6")],
                                  ci_appr = "matching",
                                  ci_seq = "logit")
```
pred_model = "sl",
sl_lib = c("m_xgboost"),
params = list(xgb_nrounds=c(10,20,30),
            xgb_eta=c(0.1,0.2,0.3)),
nthread = 1,
covar_bl_method = "absolute",
covar_bl_trs = 0.1,
covar_bl_trs_type="mean",
max_attempt = 1,
dist_measure = "l1",
delta_n = 1,
scale = 0.5)

data <- merge(m_d[, c("id", "Y")], pseudo_pop$pseudo_pop, by = "id")
erf_obj <- estimate_npmetric_erf(data$Y,
data$w,
data$counter_weight,
bw_seq=seq(0.2,2,0.2),
w_vals = seq(2,20,0.5),
nthread = 1)

---

estimate_pmetric_erf  Estimate Parametric Exposure Response Function

Description
Estimate a constant effect size for matched and weighted data set using parametric models

Usage
estimate_pmetric_erf(formula, family, data, ...)

Arguments
formula a vector of outcome variable in matched set.
family a description of the error distribution (see ?gnm)
data dataset that formula is build upon (Note that there should be a counter_weight column in this data.)
... Additional parameters for further fine tuning the gnm model.

Details
This method uses generalized nonlinear model (gnm) from gnm package.

Value
returns an object of class gnm
**estimate_semipmetric_erf**

*Estimate semi-exposure-response function (semi-ERF).*

**Description**

Estimates the smoothed exposure-response function using a generalized additive model with splines.

**Usage**

```r
estimate_semipmetric_erf(formula, family, data, ...)
```

**Arguments**

- `formula` a vector of outcome variable in matched set.
- `family` a description of the error distribution (see ?gam).
- `data` dataset that formula is build upon Note that there should be a counter_weight column in this data.
- `...` Additional parameters for further fine tuning the gam model.

**Details**

This approach uses Generalized Additive Model (gam) using mgcv package.

---

**Examples**

```r
m_d <- generate_syn_data(sample_size = 100)
pseudo_pop <- generate_pseudo_pop(m_d[, c("id", "w")],
    m_d[, c("id", "cf1","cf2","cf3",
        "cf4","cf5","cf6")],
    ci_appr = "matching",
    sl_lib = c("m_xgboost"),
    params = list(xgb_nrounds=c(10,20,30),
        xgb_eta=c(0.1,0.2,0.3)),
    nthread = 1,
    covar_bl_method = "absolute",
    covar_bl_trs = 0.1,
    covar_bl_trs_type= "mean",
    max_attempt = 1,
    dist_measure = "l1",
    delta_n = 1,
    scale = 0.5)
data <- merge(m_d[, c("id", "Y")], pseudo_pop$pseudo_pop, by = "id")outcome_m <- estimate_semipmetric_erf(formula = Y ~ w,
    family = gaussian,
    data = data)
```
**generate_pseudo_pop**

Generate pseudo population

**Description**

Generates pseudo population data set based on user-defined causal inference approach. The function uses an adaptive approach to satisfies covariate balance requirements. The function terminates either by satisfying covariate balance or completing the requested number of iteration, whichever comes first.

**Usage**

```r
generate_pseudo_pop(
  w,
  c,
  ci_appr,
  gps_density = "normal",
  use_cov_transform = FALSE,
  transformers = list("pow2", "pow3"),
  bin_seq = NULL,
```

**Value**

returns an object of class gam

**Examples**

```r
m_d <- generate_syn_data(sample_size = 100)
pseudo_pop <- generate_pseudo_pop(m_d[, c("id", "w")],
                                  m_d[, c("id", "cf1", "cf2", "cf3",
                                  "cf4", "cf5", "cf6")],
                                  ci_appr = "matching",
                                  sl_lib = c("m_xgboost"),
                                  params = list(xgb_nrounds=c(10,20,30),
                                                xgb_eta=c(0.1,0.2,0.3)),
                                  nthread = 1,
                                  covar_bl_method = "absolute",
                                  covar_bl_trs = 0.1,
                                  covar_bl_trs_type = "mean",
                                  max_attempt = 1,
                                  dist_measure = "l1",
                                  delta_n = 1,
                                  scale = 0.5)
data <- merge(m_d[, c("id", "Y")], pseudo_pop$pseudo_pop, by = "id")
outcome_m <- estimate_semipmetric_erf (formula = Y ~ w,
                                        family = gaussian,
                                        data = data)
```
exposure_trim_qtls = c(0.01, 0.99),
gps_trim_qtls = c(0, 1),
params = list(),
sl_lib = c("m_xgboost"),
nthread = 1,
include_original_data = FALSE,
gps_obj = NULL,
)

Arguments

w A data.frame comprised of two columns: one contains the observed exposure variable, and the other is labeled as 'id'. The column for the outcome variable can be assigned any name as per your requirements.
c A data.frame of includes observed covariate variables. It should also consist of a column named 'id'.
ci_appr The causal inference approach. Possible values are:
  • "matching": Matching by GPS
  • "weighting": Weighting by GPS
gps_density Model type which is used for estimating GPS value, including normal (default) and kernel.
use_cov_transform If TRUE, the function uses transformer to meet the covariate balance.
transformers A list of transformers. Each transformer should be a unary function. You can pass name of customized function in the quotes. Available transformers:
  • pow2: to the power of 2
  • pow3: to the power of 3
bin_seq Sequence of w (treatment) to generate pseudo population. If NULL is passed the default value will be used, which is seq(min(w)+delta_n/2,max(w), by=delta_n).
exposure_trim_qtls A numerical vector of two. Represents the trim quantile level for exposure values. Both numbers should be in the range of [0,1] and in increasing order (default: c(0.01, 0.99)).
gps_trim_qtls A numerical vector of two. Represents the trim quantile level for the gps values. Both numbers should be in the range of [0,1] and in increasing order (default: c(0.0, 1.0)).
params Includes list of params that is used internally. Unrelated parameters will be ignored.
sl_lib A vector of prediction algorithms.
nthread An integer value that represents the number of threads to be used by internal packages.
include_original_data If TRUE, includes the original data in the outcome.
generate_pseudo_pop

A gps object that is generated with estimate_gps function. If it is provided, the number of iteration will forced to 1 (Default: NULL).

Additional arguments passed to different models.

Details

Additional parameters:

-Causal Inference Approach (ci_appr):
  - if ci_appr = 'matching':
    - dist_measure: Matching function. Available options:
      * 1L: Manhattan distance matching
    - delta_n: caliper parameter.
    - scale: a specified scale parameter to control the relative weight that is attributed to the distance measures of the exposure versus the GPS.
    - covar_bl_method: covariate balance method. Available options:
      * 'absolute'
    - covar_bl_trs: covariate balance threshold
    - covar_bl_type: covariate balance type (mean, median, maximal)
    - max_attempt: maximum number of attempt to satisfy covariate balance.
    - See create_matching() for more details about the parameters and default values.
  - if ci_appr = 'weighting':
    - covar_bl_method: Covariate balance method.
    - covar_bl_trs: Covariate balance threshold
    - max_attempt: Maximum number of attempt to satisfy covariate balance.

Value

Returns a pseudo population (gpsm_pspop) object that is generated or augmented based on the selected causal inference approach (ci_appr). The object includes the following objects:

  - params
    - ci_appr
    - params
  - pseudo_pop
  - adjusted_corr_results
  - original_corr_results
  - best_gps_used_params
  - effect size of generated pseudo population

Examples

m_d <- generate_syn_data(sample_size = 100)
pseudo_pop <- generate_pseudo_pop(m_d[, c("id", "w")],
  m_d[, c("id", "cf1","cf2","cf3","cf4","cf5","cf6")],
generate_syn_data

Generate synthetic data for the CausalGPS package

Description

Generates synthetic data set based on different GPS models and covariates.

Usage

generate_syn_data(
  sample_size = 1000,
  outcome_sd = 10,
  gps_spec = 1,
  cova_spec = 1,
  vectorized_y = FALSE
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sample_size</td>
<td>A positive integer number that represents a number of data samples.</td>
</tr>
<tr>
<td>outcome_sd</td>
<td>A positive double number that represents standard deviation used to generate the outcome in the synthetic data set.</td>
</tr>
<tr>
<td>gps_spec</td>
<td>A numerical integer values ranging from 1 to 7. The complexity and form of the relationship between covariates and treatment variables are determined by the <code>gps_spec</code>. Below, you will find a concise definition for each of these values:</td>
</tr>
<tr>
<td></td>
<td>• <code>gps_spec: 1</code>: The treatment is generated using a normal distribution <code>stats::rnorm</code> and a linear function of covariates (cf1 to cf6).</td>
</tr>
</tbody>
</table>
• `gps_spec`: 2: The treatment is generated using a Student’s t-distribution \( \text{stats::rt} \) and a linear function of covariates, but is also truncated to be within a specific range (-5 to 25).

• `gps_spec`: 3: The treatment includes a quadratic term for the third covariate.

• `gps_spec`: 4: The treatment is calculated using an exponential function within a fraction, creating logistic-like model.

• `gps_spec`: 5: The treatment also uses logistic-like model but with different parameters.

• `gps_spec`: 6: The treatment is calculated using the natural logarithm of the absolute value of a linear combination of the covariates.

• `gps_spec`: 7: The treatment is generated similarly to `gps_spec = 2`, but without truncation.

- **cova_spec**: A numerical value (1 or 2) to modify the covariates. It determines how the covariates in the synthetic data set are transformed. If `cova_spec` equals 2, the function applies non-linear transformation to the covariates, which can add complexity to the relationships between covariates and outcomes in the synthetic data. See the code for more details.

- **vectorized_y**: A Boolean value indicates how Y internally is generated. (Default = FALSE). This parameter is introduced for backward compatibility. `vectorized_y = TRUE` performs better.

**Value**

- `synthetic_data`: The function returns a data.frame saved the constructed synthetic data.

**Examples**

```r
set.seed(298)
s_data <- generate_syn_data(sample_size = 100,
                           outcome_sd = 10,
                           gps_spec = 1,
                           cova_spec = 1)
```

---

**Description**

Returns current logger settings.

**Usage**

```r
get_logger()
```
Value

Returns a list that includes `logger_file_path` and `logger_level`.

Examples

```r
set_logger("mylogger.log", "INFO")
log_meta <- get_logger()
```

---

**plot.gpsm_erf**  
*Extend generic plot functions for gpsm_erf class*

**Description**

A wrapper function to extend generic plot functions for gpsm_erf class.

**Usage**

```r
## S3 method for class 'gpsm_erf'
plot(x, ...)
```

**Arguments**

- `x`  
  A gpsm_erf object.

- `...`  
  Additional arguments passed to customize the plot.

**Value**

Returns a ggplot2 object, invisibly. This function is called for side effects.

---

**plot.gpsm_pspop**  
*Extend generic plot functions for gpsm_erf class*

**Description**

A wrapper function to extend generic plot functions for gpsm_erf class.

**Usage**

```r
## S3 method for class 'gpsm_pspop'
plot(x, ...)
```
print.gpsm_erf

Arguments

x        A gpsm_erf object.

...      Additional arguments passed to customize the plot.

Details

Additional parameters:

* include_details: If set to TRUE, the plot will include run details (Default = FALSE).

Value

Returns a ggplot2 object, invisibly. This function is called for side effects.

---

print.gpsm_erf  Extend print function for gpsm_erf object

Description

Extend print function for gpsm_erf object

Usage

```r
## S3 method for class 'gpsm_erf'
print(x, ...)
```

Arguments

x        A gpsm_erf object.

...      Additional arguments passed to customize the results.

Value

No return value. This function is called for side effects.
print.gpsm_pspop  
Extend print function for gpsm_pspop object

Description

Extend print function for gpsm_pspop object

Usage

## S3 method for class 'gpsm_pspop'
print(x, ...)

Arguments

x  A gpsm_pspop object.
...
   Additional arguments passed to customize the results.

Value

No return value. This function is called for side effects.

set_logger  
Set Logger Settings

Description

Updates logger settings, including log level and location of the file.

Usage

set_logger(logger_file_path = "CausalGPS.log", logger_level = "INFO")

Arguments

logger_file_path  A path (including file name) to log the messages. (Default: CausalGPS.log)
logger_level  The log level. Available levels include:
   • TRACE
   • DEBUG
   • INFO (Default)
   • SUCCESS
   • WARN
   • ERROR
   • FATAL
Value

No return value. This function is called for side effects.

Examples

```r
set_logger("Debug")
```

---

**summary.gpsm_erf**  
*print summary of gpsm_erf object*

Description

print summary of gpsm_erf object

Usage

```r
## S3 method for class 'gpsm_erf'
summary(object, ...)
```

Arguments

- `object`: A `gpsm_erf` object.
- `...`: Additional arguments passed to customize the results.

Value

Returns summary of data

---

**summary.gpsm_pspop**  
*print summary of gpsm_pspop object*

Description

print summary of gpsm_pspop object

Usage

```r
## S3 method for class 'gpsm_pspop'
summary(object, ...)
```

Arguments

- `object`: A `gpsm_pspop` object.
- `...`: Additional arguments passed to customize the results.
**Value**

Returns summary of data

### Description

A dataset containing exposure, confounders, and outcome for causal inference studies. The dataset is hosted on Harvard dataverse doi:10.7910/DVN/L7YF2G. This dataset was produced from five different resources. Please see https://github.com/NSAPH-Projects/synthetic_data/ for the data processing pipelines. In the following

#### Exposure Data

The exposure parameter is PM2.5. Di et al. (2019) provided daily, and annual PM2.5 estimates at 1 km×1 km grid cells in the entire United States. The data can be downloaded from Di et al. (2021). Features in this category starts with `qd_` prefix.

#### Census Data

The main reference for getting the census data is the United States Census Bureau. There are numerous studies and surveys for different geographical resolutions. We use 2010 county level American County Survey at the county level (`acs5`). Features in this category starts with `cs_` prefix.

#### CDC Data

The Centers for Disease Control and Prevention (CDC), provides the Behavioral Risk Factor Surveillance System (Centers for Disease Control and Prevention (2021)), which is the nation’s premier system of health-related telephone surveys that collect state data about U.S. residents regarding their health-related risk behaviors.

#### GridMET Data

Climatology Lab at the University of California, Merced, provides the GridMET data (Abatzoglou (2013)). The data set is daily surface meteorological data covering the contiguous United States.

#### CMS Data

The Centers for Medicare and Medicaid Services (CMS) provides synthetic data at the county level for 2008-2010 (Centers for Medicare & Medicaid Services (2021)).

The definition of each variables are provided below. All data are collected for 2010 and aggregated into the county level and in the contiguous United States.

### Usage

```r
data(synthetic_us_2010)
```
Format

A data frame with 3109 rows and 46 variables:

- **qd_mean_pm25**: Mean PM2.5 (microgram/m³)
- **cs_poverty**: The proportion of below poverty level population among 65+ years old.
- **cs_hispanic**: The proportion of Hispanic or Latino population among 65+ years old.
- **cs_black**: The proportion of Black or African American population among 65+ years old.
- **cs_white**: The proportion of White population among 65 years and over.
- **cs_native**: The proportion of American Indian or Alaska native population among 65 years and over.
- **cs_asian**: The proportion of Asian population among 65 years and over.
- **cs_other**: The proportion of other races population among 65 years and over.
- **cs_ed_below_highschool**: The proportion of the population with below high school level education among 65 years and over.
- **cs_household_income**: Median Household income in the past 12 months (in 2010 inflation-adjusted dollars) where householder is 65 years and over.
- **cs_median_house_value**: Median house value (USD)
- **cs_total_population**: Total Population
- **cs_area**: Area of each county (square miles)
- **cs_population_density**: The number of the population in one square mile.
- **cdc_mean_bmi**: Body Mass Index.
- **cdc_pct_cusmoker**: The proportion of current smokers.
- **cdc_pct_sdsmoker**: The proportion of some days smokers.
- **cdc_pct_fmsmoker**: The proportion of former smokers.
- **cdc_pct_nvsmoker**: The proportion of never smokers.
- **cdc_pct_nmsmoker**: The proportion of not known smokers.
- **gmet_mean_tmmn**: Annual mean of daily minimum temperature (K)
- **gmet_mean_summer_tmmn**: The mean of daily minimum temperature during summer (K)
- **gmet_mean_winter_tmmn**: The mean of daily minimum temperature during winter (K)
- **gmet_mean_tmmx**: Annual mean of daily maximum temperature (K)
- **gmet_mean_summer_tmmx**: The mean of daily maximum temperature during summer (K)
- **gmet_mean_winter_tmmx**: The mean of daily maximum temperature during winter (K)
- **gmet_mean_rmn**: Annual mean of daily minimum relative humidity (%) 
- **gmet_mean_summer_rmn**: The mean of daily minimum relative humidity during summer (%) 
- **gmet_mean_winter_rmn**: The mean of daily minimum relative humidity during winter (%) 
- **gmet_mean_rmx**: Annual mean of daily maximum relative humidity (%) 
- **gmet_mean_summer_rmx**: The mean of daily maximum relative humidity during summer (%) 
- **gmet_mean_winter_rmx**: The mean of daily maximum relative humidity during winter (%)
gmet_mean_sph  Annual mean of daily mean specific humidity (kg/kg)
gmet_mean_summer_sph  The mean of daily mean specific humidity during summer(kg/kg)
gmet_mean_winter_sph  The mean of daily mean specific humidity during winter(kg/kg)
cms_mortality_pct  The proportion of deceased patients.
cms_white_pct  The proportion of White patients.
cms_black_pct  The proportion of Black patients.
cms_hispanic_pct  The proportion of Hispanic patients.
cms_others_pct  The proportion of Other patients.
cms_female_pct  The proportion of Female patients.
region  The region that the county is located in.
   NORTHEAST=("NY","MA","PA","RI","NH","ME","VT","CT","NJ")
   SOUTH=("DC","VA","NC","WV","KY","SC","GA","FL","AL","TN","MS","AR","MD","DE","OK","TX","LA")
   MIDWEST=c("OH","IN","MI","IA","MO","WI","MN","SD","ND","IL","KS","NE")
   WEST=c("MT","CO","WY","ID","UT","NV","CA","OR","WA","AZ","NM")
FIPS  Federal Information Processing Standards, a unique ID for each county.
NAME  County, State name.
STATE  State abbreviation.
STATE_CODE  State numerical code.

References

Index

* datasets
  synthetic_us_2010, 22

absolute_corr_fun, 3
absolute_weighted_corr_fun, 4

CausalGPS (CausalGPS-package), 2
CausalGPS-package, 2
check_covar_balance, 5
compile_pseudo_pop, 6
create_matching(), 15

estimate_gps, 8
estimate_npmetric_erf, 9
estimate_pmetric_erf, 11
estimate_seminipmetric_erf, 12

generate_pseudo_pop, 13
generate_syn_data, 16
get_logger, 17

plot.gpsm_erf, 18
plot.gpsm_pspop, 18
print.gpsm_erf, 19
print.gpsm_pspop, 20

set_logger, 20
summary.gpsm_erf, 21
summary.gpsm_pspop, 21
synthetic_us_2010, 22