Package 'Indicator'

January 20, 2025

Type Package

Title Composite 'Indicator' Construction and Imputation Data

Version 0.1.3

Maintainer Gianmarco Borrata <gianmarco.borrata@unina.it>

Description Different functions includes constructing composite indicators, imputing missing data, and evaluating imputation techniques. Additionally, different tools for data normalization. Detailed methodologies of 'Indicator' package are: OECD/European Union/EC-JRC (2008), ``Handbook on Constructing Composite Indicators: Methodology and User Guide'', OECD Publishing, Paris, <DOI:10.1787/533411815016>, Matteo Mazziotta & Adriano Pareto, (2018) ``Measuring Well-Being Over Time: The Adjusted Mazziotta–Pareto Index Versus Other Non-compensatory Indices'' <DOI:10.1007/s11205-017-1577-5> and De Muro P., Mazziotta M., Pareto A. (2011), ``Composite Indices of Development and Poverty: An Application to MDGs'' <DOI:10.1007/s11205-010-9727-z>.

License Unlimited

Depends R (>= 4.0)

Imports FactoMineR, missMethods, stats, norm

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Suggests spelling

Language en-US

URL https://github.com/GianmarcoBorrata/Indicator

BugReports https://github.com/GianmarcoBorrata/Indicator/issues

NeedsCompilation no

Author Gianmarco Borrata [aut, cre],
Pasquale Pipiciello [aut]

Repository CRAN

Date/Publication 2024-11-27 22:40:01 UTC

Contents

`Indicator-package` *Indicator: A package for constructing composite indicators, imputing and evaluating missing data imputation*

Description

Provides three main categories of functions: construction of composite indicators, imputation and evaluation of missing data, and data normalization.

Details

Key features include:

- Construction of composite indicators, such as Mazziotta-Pareto Index, Adjusted Mazziotta-Pareto Index, Geometric aggregation, Linear aggregation, and other functions;
- Imputation of missing data through techniques such as Linear Regression Imputation, Hot Deck Imputation, etc;
- Evaluation of missing data imputation using metrics such as R^2, RMSE, and MAE;
- Several functions to standardize and normalize data, such as Standardization by Adjusted Mazziotta-Pareto method, Normalization by Adjusted Mazziotta-Pareto method, and other functions.

Author(s)

Gianmarco Borrata <gianmarco.borrata@unina.it> and Pasquale Pipiciello <pasqualepipiciello24@gmail.com>

References

- OECD/European Union/EC-JRC (2008), "Handbook on Constructing Composite Indicators: Methodology and User Guide", OECD Publishing, Paris, <DOI:10.1787/533411815016>
- Matteo Mazziotta & Adriano Pareto (2018), "Measuring Well-Being Over Time: The Adjusted Mazziotta–Pareto Index Versus Other Non-compensatory Indices",Social Indicators Research, Springer, vol. 136(3), pages 967-976, April <DOI:10.1007/s11205-017-1577-5>
- De Muro P., Mazziotta M., Pareto A. (2011), "Composite Indices of Development and Poverty: An Application to MDGs",Social Indicators Research, Volume 104, Number 1, pp. 1-18 <DOI:10.1007/s11205-010-9727-z>

See Also

Useful links:

- https://github.com/GianmarcoBorrata/Indicator
- Report bugs at https://github.com/GianmarcoBorrata/Indicator/issues

---

`columns_with_nan` *Function to get the names of the columns with NAN values*

Description

This function identifies and returns the names of the columns in a DataFrame that contain missing values (NaN). It is particularly useful for missing data imputation and preliminary analysis, allowing for quick identification of columns that need to be handled due to the presence of missing values

Usage

```
columns_with_nan(data)
```

Arguments

`data`	The DataFrame's rows represent observations and the columns represent variables

Value

It returns a vector of columns with NAN

Examples

```
data("airquality")
columns_with_nan(airquality)
```

compute_CI *Calculation of Condition Indices*

Description

Diagnosis of collinearity in matrix `X`

Usage

```
compute_CI(matrix)
```

Arguments

`matrix`	a matrix of data where rows = observations and columns = variables

Details

Collinearities can inflate the variance of the estimated regression coefficients and numerical stability. The condition indices are calculated by the eigenvalues of the crossproduct matrix of the scaled but uncentered explanatory variables. Indices > 30 may indicate collinearity

Value

It returns the condition index of the matrix

References

Belsley, D. , Kuh, E. and Welsch, R. E. (1979), Regression Diagnostics: Identifying Influential Data and Sources of Collinearity, John Wiley (New York)

Examples

```
data("Education")
compute_CI(Education)
```

Education *Education*

Description

This dataset contains educational and cultural data related to Italian regions. Each row represents a region, and there are 20 Italian regions. The eight variables included in the dataset provide information on various aspects of the educational system and cultural activities in the regions in 2020.

Usage

```
Education
```

Format

A data frame with 20 observations on the following 8 variables:

- Participation in the school system of 4-5 year olds
- People with at least a high school diploma (25-64 years old)
- Graduates and other tertiary degrees (30-34 years)
- Participation in continuing education
- People earning a STEM tertiary degree in the year Cultural
- Participation outside the home
- Reading of books and newspapers
- Library use

Source

https://www.istat.it/en/well-being-and-sustainability/the-measurement-of-well-being/indicators

---

geometric_aggregation *Geometric Aggregation*

Description

The purpose of the Geometric Aggregation function is to calculate a synthetic index based on the geometric mean for a set of variables

Usage

```
geometric_aggregation(
  data,
  weights = rep(1/ncol(data), ncol(data)),
  geo_wo = 0
)
```

Arguments

`data`	dataframe with rows = observations and columns = quantitative
`weights`	vector of weights (default all weights = 1/ ncol(dataframe))
`geo_wo`	(geometric mean workaround to deal with negative values), positive number to make all variables positive

Details

This is simply the product of each indicator to the power of its weight, all raised the the power of the inverse of the sum of the weights

The geometric mean is less compensatory than the arithmetic mean – low values in one indicator only partially substitute high values in others. For this reason, the geometric mean may sometimes be preferred when indicators represent "essentials". An example might be quality of life: a longer life expectancy perhaps should not compensate severe restrictions on personal freedoms

Value

It returns a dataframe with rows = observations and column = composite indicator

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("Education")
Indicator=geometric_aggregation(Education)
print(Indicator)

#-----When there are negative values
set.seed(123)
Data=matrix(rnorm(100),nrow = 10,ncol = 10)
Indicator=geometric_aggregation(Data,geo_wo = 100)
print(Indicator)
```

---

get_all_performance *Function to evaluate different nan imputation methods*

Description

The get_all_performance_boot function is designed to evaluate different methods of imputing missing values into a dataset

Usage

```
get_all_performance(data, to_impute, regressors)
```

Arguments

data	dataframe with rows = observations and columns = quantitative variables
to_impute	string , name of the variables where there are NANs to impute
regressors	vector of string with names of the variables to use to apply 1st,4th imputation method

Details

The function calculates performance metrics, such as:

- $R^2 = [1/N * [(\sum_{i=1}^{N}(P_i - (\bar{P})(O_i - (\bar{O})]/\sigma_P * \sigma_O]^2$,

- $RMSE = (1/N * (\sum_{i=1}^{N}(P_i - O_i)^2)^{1/2}$

and

- $MAE = 1/N * \sum_{i=1}^{N}|P_i - O_i|$

for each imputation method

Supported Imputation Methods:

1. Linear Regression Imputation (lm_imputation): it uses a linear regression model to predict and impute missing values

2. Median Imputation (median_imputation): it replaces missing values with the median of observed values

3. Mean Imputation (mean_imputation): it replaces missing values with the mean of observed values

4. Hot Deck Imputation (hot_deck_imputation): it replaces missing values with similar observed values

5. Expectation-Maximization Imputation (EM_imputation): it uses the Expectation-Maximization algorithm to estimate and impute missing values

It evaluate different methods of imputing missing values and calculate performance metrics for each method

Value

It returns a performance measures dataframe with rows = methods and columns = methods' performances

Note

For the methods Median Imputation and Mean Imputation, it is not possible to calculate the R^2 value. This is because the standard deviation is zero based on the following R^2 formula:

$R^2 = [1/N * [(\sum_{i=1}^{N}(P_i - (\bar{P})(O_i - (\bar{O})]/\sigma_P * \sigma_O]^2$

where:

- N is the number of imputations,

- O_i are the observed data point,

- P_i are the imputed data point,

- $\bar{O}$ are the average of the observed data,

- $\bar{P}$ are the average of the imputed data,

- σ_P are the standard deviation of the imputed data,

- σ_O are the standard deviation of the observed data

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("airquality")
regressors<-colnames(airquality[,c(3,4)])
suppressWarnings(get_all_performance(data =airquality,"Ozone",regressors = regressors))
```

`get_all_performance_boot` *Function to evaluate different nan imputation methods with bootstrap*

Description

The get_all_performance_boot function is designed to evaluate different methods of imputing missing values into a dataset. The evaluation is performed using bootstrapping to ensure robustness of the results

Usage

```
get_all_performance_boot(data, to_impute, regressors, nb = 1)
```

Arguments

`data`	dataframe with rows = observations and columns = quantitative variables
`to_impute`	string , name of the variables where there are NANs to impute
`regressors`	vector of string with names of the variables to use to apply 1st, 4th imputation method
`nb`	number of bootstrap samples

Details

The function calculates performance metrics, such as:

- $R^2 = [1/N * [(\sum_{i=1}^{N}(P_i - (\bar{P})(O_i - (\bar{O})]/\sigma_P * \sigma_O]^2$,

- $RMSE = (1/N * (\sum_{i=1}^{N}(P_i - O_i)^2)^{1/2}$

and

- $MAE = 1/N * \sum_{i=1}^{N} |P_i - O_i|$

for each imputation method

Supported Imputation Methods:

1. Linear Regression Imputation (lm_imputation): it uses a linear regression model to predict and impute missing values

2. Median Imputation (median_imputation): it replaces missing values with the median of observed values

3. Mean Imputation (mean_imputation): it replaces missing values with the mean of observed values

4. Hot Deck Imputation (hot_deck_imputation): it replaces missing values with similar observed values

5. Expectation-Maximization Imputation (EM_imputation): it uses the Expectation-Maximization algorithm to estimate and impute missing values

Evaluate different methods of imputing missing values using bootstrapping and calculate performance metrics for each method

Value

It returns a performance measures dataframe with rows = methods and columns = methods' performances averaged over bootstraps.

Note

For the methods Median Imputation and Mean Imputation, it is not possible to calculate the R^2 value. This is because the standard deviation is zero based on the following R^2 formula:

$R^2 = [1/N * [(\sum_{i=1}^{N}(P_i - (\bar{P})(O_i - (\bar{O})]/\sigma_P * \sigma_O]^2$

where:

- N is the number of imputations,

- O_i are the observed data point,

- P_i are the imputed data point,

- $\bar{O}$ are the average of the observed data,

- $\bar{P}$ are the average of the imputed data,

- σ_P are the standard deviation of the imputed data,

- σ_O are the standard deviation of the observed data.

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("airquality")
regressors<-colnames(airquality[,c(3,4)])
suppressWarnings(get_all_performance_boot(data =airquality,"Ozone",regressors = regressors,nb=100))
```

Jevons_aggregation *Jevons static aggregation*

Description

The Jevons_aggregation function computes an index using the Jevons method of static aggregation. This method calculates the geometric mean, multiplied for 100.

Usage

```
Jevons_aggregation(data)
```

Arguments

data	dataframe with rows = observations and columns = quantitative variables

Value

It returns a dataframe with rows = observations and column = composite indicator

References

Massoli, P., Mazziotta, M., Pareto, A., Rinaldelli, C. (2013). Synthesis Methodologies and Spatial Analysis. Composite indices for BES,DAYS OF RESEARCH IN ISTAT, NOVEMBER 10-11, 2014

Examples

```
data("Education")
Indicator=Jevons_aggregation(Education)
print(Indicator)
```

linear_aggregation *Linear Aggregation*

Description

This is a function to apply linear aggregation index

Usage

```
linear_aggregation(data, weights = rep(1/ncol(data), ncol(data)))
```

Arguments

data	dataframe with rows = observations and columns = quantitative variables
weights	vector of weights (default all weights = 1/ ncol(dataframe))

Value

It returns a dataframe with rows = observations and column = composite indicator

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("Education")
Indicator=linear_aggregation(Education)
print(Indicator)
```

`linear_aggregation_AMPI`

Adjusted Mazziotta-Pareto index

Description

The Adjusted Mazziotta-Pareto Index (AMPI) is a composite index for summarizing a set of indicators that are assumed to be non-substitutable, i.e., all components must be balanced. It is based on a non-linear function which, starting from the arithmetic mean, introduces a penalty for the units with unbalanced values of the indicators

Usage

```
linear_aggregation_AMPI(data, pol = "pos")
```

Arguments

`data`	dataframe with rows = observations and columns = quantitative variables
`pol`	pol if not selected is "positive"", otherwise write "neg" (see details)

Details

The 'polarity' of an indicator is the sign of the relation between the indicator and the phenomenon to be measured (+ if the indicator represents a dimension considered positive and - otherwise)

Value

It returns a dataframe with rows = observations and column = composite indicator

References

Matteo Mazziotta & Adriano Pareto, 2018. "Measuring Well-Being Over Time: The Adjusted Mazziotta–Pareto Index Versus Other Non-compensatory Indices," Social Indicators Research: An International and Interdisciplinary Journal for Quality-of-Life Measurement, Springer, vol. 136(3), pages 967-976, April

Examples

```
data("Education")
Indicator=linear_aggregation_AMPI(Education)
print(Indicator)

#----With negative polarity
Indicator_neg=linear_aggregation_AMPI(Education,"neg")
print(Indicator_neg)
```

---

linear_aggregation_MPI

Mazziotta-Pareto index

Description

The Mazziotta–Pareto index (MPI) is a composite index for summarizing a set of individual indicators that are assumed to be not fully substitutable. It is based on a non-linear function which, starting from the arithmetic mean of the normalized indicators, introduces a penalty for the units with unbalanced values of the indicators

Usage

```
linear_aggregation_MPI(data, pol = "pos")
```

Arguments

data	dataframe with rows = observations and columns = quantitative
pol	polarity if not selected is positive, otherwise write neg (see details)

Details

The 'polarity' of an indicator is the sign of the relation between the indicator and the phenomenon to be measured (+ if the indicator represents a dimension considered positive and - otherwise)

Value

It returns a dataframe with rows = observations and column = composite indicator

References

De Muro P., Mazziotta M., Pareto A. (2011), "Composite Indices of Development and Poverty: An Application to MDGs", Social Indicators Research, Volume 104, Number 1, pp. 1-18

Examples

```
data("Education")
Indicator=linear_aggregation_MPI(Education)
print(Indicator)

#----With negative polarity
Indicator_neg=linear_aggregation_MPI(Education,"neg")
print(Indicator_neg)
```

---

lm_imputation *Function to apply nan inputation with linear regression*

Description

The lm_imputation function aims to replace missing values (NA) in a dataset with values estimated using a linear regression model. This technique allows the existing relationships between variables in the dataset to be used to accurately estimate missing values

Usage

```
lm_imputation(data, to_impute, regressors)
```

Arguments

data	dataframe with rows = observations and columns = quantitative variables
to_impute	string , name of the variables whre there are NANs to impute
regressors	vector of string with names of the variables to use to apply linear regression imputation

Value

It returns a dataframe with imputed values

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("airquality")
regressors<-colnames(airquality[,c(3,4)])
lm_imputation(data =airquality,"Ozone",regressors = regressors)
```

MAD *Mean absolute difference of rank*

Description

Function to calculate the mean absolute difference of rank for different methods

Usage

```
MAD(matrix_data)
```

Arguments

matrix_data	data matrix of indicator

Details

Function to calculate the mean absolute difference of rank for different methods. Create the matrix of ranking for different columns, the rank is the high value is the first. Calculate the different in absolute values for different columns and calculate the mean for different methods

Value

It returns a data frame of mean absolute difference of rank for different methods

References

Matteo Mazziotta & Adriano Pareto, 2018. "Measuring Well-Being Over Time: The Adjusted Mazziotta–Pareto Index Versus Other Non-compensatory Indices," Social Indicators Research: An International and Interdisciplinary Journal for Quality-of-Life Measurement, Springer, vol. 136(3), pages 967-976, April

Examples

```
data("Education")
Indicator_MPI=linear_aggregation_MPI(Education)
Indicator_AMPI=linear_aggregation_AMPI(Education)
Indicator_GA=geometric_aggregation(Education)
All_Indicator=cbind(Indicator_MPI,Indicator_AMPI,Indicator_GA)
MAD=MAD(All_Indicator)
print(MAD)
```

min_max *Min-max normalization*

Description

Min-max normalization transforms each value by subtracting its minimum and dividing by its range (maximum-minimum). The result is a new variable with a minimum of zero and a maximum of one

Usage

```
min_max(data)
```

Arguments

data	dataframe with rows = observations and columns = quantiative variables

Details

Change the value of variable to negative if it has negative polarity

Value

It returns a datafame of normalized data

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("Education")
Normalization=min_max(Education)
print(Normalization)
```

min_max_GM *Normalization for the Geometric Mean*

Description

This is a data normalization function for the geometric mean, where we multiplied the normalized data by 198 and add 1, with positive or negative polarity

Usage

```
min_max_GM(data, pol = "pos")
```

Arguments

data	dataframe with rows = observations and columns = quantiative
pol	polarity if not selected is "positive"", otherwise write "neg"

Details

The 'polarity' of an indicator is the sign of the relation between the indicator and the phenomenon to be measured (+ if the indicator represents a dimension considered positive and - otherwise)

Value

It returns a datafame of normalized data

References

Massoli, P., Mazziotta, M., Pareto, A., Rinaldelli, C. (2013). Synthesis Methodologies and Spatial Analysis. Composite indices for BES,DAYS OF RESEARCH IN ISTAT, NOVEMBER 10-11, 2014

Examples

```
data("Education")
Normalization=min_max_GM(Education)
print(Normalization)

#----With negative polarity
Normalization_neg=linear_aggregation_AMPI(Education,"neg")
print(Normalization_neg)
```

---

normalization_abov_below_mean

Normalization above or below the mean

Description

This transformation considers the indicators which are above and below an arbitrarily defined threshold, p, around the mean. The threshold p builds a neutral region around the mean, where the transformed indicator is zero. This reduces the sharp discontinuity, from -1 to +1, which exists across the mean value to two minor discontinuities, from -1 to 0 and from 0 to +1, across the thresholds

Usage

```
normalization_abov_below_mean(data, p = 0.01)
```

Arguments

data	dataframe with rows = observations and columns = quantiative variables
p	threshold for the window

Details

This function to normalize in -1, 0, and 1

Value

It returns a datafame of normalized data

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("Education")
Indicator=normalization_abov_below_mean(Education)
print(Indicator)

#---With different threshold
Indicator=normalization_abov_below_mean(Education,p=0.1)
print(Indicator)
```

`pca_weighting` *Function that weight the quantitative variable by PCA method*

Description

The pca_weighting function is designed to perform a principal component analysis (PCA) on the input data to calculate weights that correct for overlapping information between related indicators. This process makes it possible to create a composite indicator that captures as much information as possible from individual indicators while reducing the dimensionality of the data

Usage

```
pca_weighting(data)
```

Arguments

`data`	dataframe with rows = observations and columns = quantitative variables

Value

It returns a dataframe with rows = observations and column = composite indicator

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("Education")
Indicator_pca=pca_weighting(Education)
print(Indicator_pca)
```

`performance_nan_imputation` *Function to evaluate nan imputation method's performance*

Description

This function evaluates the performance of various missing value imputation methods in a quantitative dataframe. It is designed to examine and compare five different imputation methods using standard performance measures

Usage

```
performance_nan_imputation(data, to_impute, regressors, method = 1)
```

Arguments

`data`	A dataframe containing the observations (rows) and quantitative variables (columns) to be analyzed. This dataframe includes variables with missing values to be imputed
`to_impute`	A string specifying the name of the variable in the dataframe that contains the missing values to be imputed
`regressors`	A vector of strings indicating the names of the variables to be used as regressors for imputation in the case of methods 1 (lm_imputation) and 4 (hot deck imputation)
`method`	An integer between 1 and 5 that specifies the imputation method to be used. The supported methods are: 1: lm_imputation (Imputation by linear model) 2: median imputation (imputation by median) 3: mean imputation (imputation by mean) 4: hot deck imputation (imputation via hot deck) 5: EM imputation (imputation via Expectation-Maximization)

Details

This function is useful for comparing the effectiveness of different methods of imputing missing values, allowing the most appropriate method to be chosen based on measured performance

Value

The function returns a dataframe that contains a row for each imputation method and columns with performance measures. The performance measures included are:

R^2: Coefficient of Determination, which measures how well the imputed values fit the observed values

RMSE: Root Mean Squared Error, which provides a measure of the mean square deviation between imputed and observed values

MAE: Mean Absolute Error, which represents the mean absolute deviation between the imputed and observed values

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("airquality")
regressors<-colnames(airquality[,c(3,4)])

#---Methods 1 = Imputation by linear model
performance_nan_imputation(data =airquality,"Ozone",regressors = regressors,method = 1)

#---Methods 2 = Imputation by Median
suppressWarnings(performance_nan_imputation(data =airquality,"Ozone",method = 2))

#---Methods 3 = Imputation by Mean
suppressWarnings(performance_nan_imputation(data =airquality,"Ozone",method = 3))

#---Methods 4 = Hot Deck imputation
performance_nan_imputation(data =airquality,"Ozone",regressors = regressors,method = 4)

#---Methods 5 = Expectation-Maximization imputation
performance_nan_imputation(data =airquality,"Ozone",regressors = regressors,method = 5)
```

---

rank_aggregation *Ranking Aggregation*

Description

It's a function to apply ranking aggregation. The highest value is the first value of rank

Usage

```
rank_aggregation(data)
```

Arguments

data	dataframe with rows = observations and columns = quantiative variables

Value

It returns a dataframe with rows = observations and column = composite indicator

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("Education")
Indicator_rank=rank_aggregation(Education)
print(Indicator_rank)
```

---

rank_normalisation *Rank normalization*

Description

It's a function that normalize by ranking method. Create the matrix of ranking for different columns, the rank is the high value is the first

Usage

```
rank_normalisation(data)
```

Arguments

data	dataframe with rows = observations and columns = quantiative variables

Value

It returns a datafame of normalized data

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("Education")
Normalized_rank=rank_normalisation(Education)
print(Normalized_rank)
```

standardization *Standardization*

Description

It performs a standardization of data, i.e., centering and scaling, so that the data is expressed in terms of standard deviation (i.e., mean = 0, SD = 1). When applied to a statistical model, this function extracts the dataset, standardizes it, and refits the model with this standardized version of the dataset

Usage

```
standardization(data)
```

Arguments

data	dataframe with rows = observations and columns = quantitative variables

Value

It returns a dataframe of scaled data

References

OECD/European Union/EC-JRC (2008), Handbook on Constructing Composite Indicators: Methodology and User Guide, OECD Publishing, Paris, <https://doi.org/10.1787/9789264043466-en>

Examples

```
data("Education")
Standardization=standardization(Education)
print(Standardization)
```

Standardization_AMPI *Standardization of data with Adjusted Maziotta-Pareto index*

Description

This is a function that standardized the data with Adjusted Maziotta-Pareto index with positive or negative polarity

Usage

```
Standardization_AMPI(data, pol = "pos")
```

Arguments

data	dataframe with rows = observations and columns = quantitative variables
pol	polarity if not selected is "positive"", otherwise write "neg" (see details)

Details

The 'polarity' of an indicator is the sign of the relation between the indicator and the phenomenon to be measured (+ if the indicator represents a dimension considered positive and - otherwise)

Value

It returns a dataframe of normalized data

References

Matteo Mazziotta & Adriano Pareto, 2018. "Measuring Well-Being Over Time: The Adjusted Mazziotta–Pareto Index Versus Other Non-compensatory Indices," Social Indicators Research: An International and Interdisciplinary Journal for Quality-of-Life Measurement, Springer, vol. 136(3), pages 967-976, April

Examples

```
data("Education")
Standardization_AMPI=Standardization_AMPI(Education)
print(Standardization_AMPI)

#----With negative polarity
Standardization_AMPI_neg=Standardization_AMPI(Education,"neg")
print(Standardization_AMPI_neg)
```

---

standardization_MPI *Standardization of data with Maziotta-Pareto index*

Description

This is a function that standardized the data with Maziotta-Pareto index

Usage

```
standardization_MPI(data)
```

Arguments

data	dataframe with rows = observations and columns = quantitative variables

Value

It returns a dataframe of normalized data

References

De Muro P., Mazziotta M., Pareto A. (2011), "Composite Indices of Development and Poverty: An Application to MDGs", Social Indicators Research, Volume 104, Number 1, pp. 1-18

Examples

```
data("Education")
Standardization_MPI=standardization_MPI(Education)
print(Standardization_MPI)
```

Index