Package ‘IBclust’

August 14, 2025

Type Package

Title Information Bottleneck Methods for Clustering Mixed-Type Data

Version 1.2.1

Description Implements multiple variants of the Information Bottleneck ('IB') method for clustering datasets containing mixed-type variables (nominal, ordinal, and continuous). The package provides deterministic, agglomerative, generalized, and standard 'IB' clustering algorithms that preserve relevant information while forming interpretable clusters. The Deterministic Information Bottleneck is described in Costa et al. (2024) <doi:10.48550/arXiv.2407.03389>. The standard 'IB' method originates from Tishby et al. (2000) <doi:10.48550/arXiv.physics/0004057>, the agglomerative variant from Slonim and Tishby (1999) <`https://papers.nips.cc/paper/1651-agglomerative-information-bottleneck`>, and the generalized 'IB' from Strouse and Schwab (2017) <doi:10.1162/NECO_a_00961>.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 3.5.0)

Imports Rcpp, stats, utils, np, rje, Rdpack, RcppEigen

Suggests mclust

LinkingTo Rcpp, RcppArmadillo, RcppEigen

RoxygenNote 7.3.2

RdMacros Rdpack

NeedsCompilation yes

Author Efthymios Costa [aut],
Ioanna Papatsouma [aut],
Angelos Markos [aut, cre]

Maintainer Angelos Markos <`amarkos@gmail.com`>

Archs x64

Contents

AIBcat — *Cluster Categorical Data Using the Agglomerative Information Bottleneck Algorithm*

Description

The AIBcat function implements the Agglomerative Information Bottleneck (AIB) algorithm for hierarchical clustering of datasets containing categorical variables. This method merges clusters so that information retention is maximised at each step to create meaningful clusters, leveraging bandwidth parameters to handle different categorical data types (nominal and ordinal) effectively (Slonim and Tishby 1999).

Usage

```
AIBcat(X, lambda = -1, nomkernel = "aitchisonaitken", ordkernel = "liracine")
```

Arguments

X	A data frame containing the categorical data to be clustered. All variables should be categorical, either `factor` (for nominal variables) or `ordered` (for ordinal variables).
lambda	A numeric value or vector specifying the bandwidth parameter for categorical variables. The default value is -1, which enables automatic determination of the optimal bandwidth. For nominal variables and `nomkernel = 'aitchisonaitken'`, the maximum allowable value of `lambda` is $(l-1)/l$, where l represents the number of categories, whereas for `nomkernel = 'liracine'` the maximum allowable value is 1. For ordinal variables, the maximum allowable value of `lambda` is 1, regardless of what `ordkernel` is being used.
nomkernel	Kernel used for nominal (unordered categorical) variables. Can be one of `aitchisonaitken` (default) or `liracine`.
ordkernel	Kernel used for ordinal (ordered categorical) variables. Can be one of `liracine` (default) or `wangvanryzin`.

Details

The AIBcat function applies the Agglomerative Information Bottleneck algorithm to do hierarchical clustering of datasets containing only categorical variables, both nominal and ordinal. The algorithm uses an information-theoretic criterion to merge clusters so that information retention is maximised at each step to create meaningful clusters with maximal information about the original distribution.

To estimate the distributions of categorical features, the function utilizes specialized kernel functions.

For nominal (unordered categorical) variables, the kernel functions implemented are:

- *Aitchison & Aitken kernel (Aitchison and Aitken 1976):*

$$K_u(x = x'; \lambda) = \begin{cases} 1 - \lambda, & \text{if } x = x' \\ \frac{\lambda}{\ell - 1}, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq \frac{\ell - 1}{\ell}.$$

- *Li & Racine kernel (Ouyang et al. 2006):*

$$K_u(x = x'; \lambda) = \begin{cases} 1, & \text{if } x = x' \\ \lambda, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq 1.$$

For ordinal (ordered categorical) variables, the kernel functions implemented are:

- *Li & Racine kernel (Li and Racine 2003):*

$$K_o(x = x'; \nu) = \begin{cases} 1, & \text{if } x = x' \\ \nu^{|x - x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

- *Wang & van Ryzin kernel (Wang and Van Ryzin 1981):*

$$K_o(x = x'; \nu) = \begin{cases} 1 - \nu, & \text{if } x = x' \\ \frac{1-\nu}{2} \nu^{|x - x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

Here, λ, and ν are bandwidth or smoothing parameters, while ℓ is the number of levels of the categorical variable. The lambda parameter is automatically determined by the algorithm if not provided by the user. For ordinal variables, the lambda parameter of the function is used to define ν.

Value

A list containing the following elements:

`merges` A data frame with 2 columns and n rows, showing which observations are merged at each step.

`merge_costs` A numeric vector tracking the cost incurred by each merge $I(Z_m; Y) - I(Z_{m-1}; Y)$.

`partitions` A list containing n sub-lists. Each sub-list includes the cluster partition at each step.

`I_Z_Y` A numeric vector including the mutual information $I(Z_m; Y)$ as the number of clusters m increases.

`I_X_Y` A numeric value of the mutual information $I(X; Y)$ between observation indices and location.

`info_ret` A numeric vector of length n including the fraction of the original information retained after each merge.

`dendrogram` A dendrogram visualising the cluster hierarchy. The height is determined by the cost of cluster merges.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Slonim N, Tishby N (1999). “Agglomerative Information Bottleneck.” *Advances in Neural Information Processing Systems*, **12**.

Aitchison J, Aitken CG (1976). “Multivariate binary discrimination by the kernel method.” *Biometrika*, **63**(3), 413–420.

Li Q, Racine J (2003). “Nonparametric estimation of distributions with categorical and continuous data.” *Journal of Multivariate Analysis*, **86**(2), 266–292.

Ouyang D, Li Q, Racine J (2006). “Cross-validation and the estimation of probability distributions with categorical data.” *Journal of Nonparametric Statistics*, **18**(1), 69–100.

Wang M, Van Ryzin J (1981). “A class of smooth estimators for discrete distributions.” *Biometrika*, **68**(1), 301–309.

See Also

AIBmix, AIBcont

Examples

```
# Simulated categorical data
set.seed(123)
X <- data.frame(
  Var1 = as.factor(sample(letters[1:3], 200, replace = TRUE)),  # Nominal variable
  Var2 = as.factor(sample(letters[4:6], 200, replace = TRUE)),  # Nominal variable
  Var3 = factor(sample(c("low", "medium", "high"), 200, replace = TRUE),
                levels = c("low", "medium", "high"), ordered = TRUE)  # Ordinal variable
)

# Run AIBcat with automatic lambda selection
result <- AIBcat(X = X, lambda = -1)

# Print clustering results
plot(result$dendrogram, xlab = "", sub = "")  # Plot dendrogram
```

AIBcont *Cluster Continuous Data Using the Agglomerative Information Bottleneck Algorithm*

Description

The AIBcont function implements the Agglomerative Information Bottleneck (AIB) algorithm for hierarchical clustering of datasets containing categorical variables. This method merges clusters so that information retention is maximised at each step to create meaningful clusters, leveraging bandwidth parameters to handle different categorical data types (nominal and ordinal) effectively (Slonim and Tishby 1999).

Usage

```
AIBcont(X, s = -1, scale = TRUE, contkernel = "gaussian")
```

Arguments

`X` A data frame containing the categorical data to be clustered. All variables should be categorical, either `factor` (for nominal variables) or `ordered` (for ordinal variables).

`s` A numeric value or vector specifying the bandwidth parameter(s) for continuous variables. The values must be greater than 0. The default value is -1, which enables the automatic selection of optimal bandwidth(s).

`scale` A logical value indicating whether the continuous variables should be scaled to have unit variance before clustering. Defaults to `TRUE`.

`contkernel` Kernel used for continuous variables. Can be one of `gaussian` (default) or `epanechnikov`.

Details

The `AIBcont` function applies the Agglomerative Information Bottleneck algorithm to do hierarchical clustering of datasets containing only continuous variables, both nominal and ordinal. The algorithm uses an information-theoretic criterion to merge clusters so that information retention is maximised at each step to create meaningful clusters with maximal information about the original distribution.

To estimate the distributions of continuous features, the function utilizes specialized kernel functions:

- *Gaussian (RBF) kernel (Silverman 1998):*

$$K_c\left(\frac{x-x'}{s}\right) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{(x-x')^2}{2s^2}\right\}, \quad s > 0.$$

- *Epanechnikov kernel (Epanechnikov 1969):*

$$K_c(x-x';s) = \begin{cases} \frac{3}{4\sqrt{5}}\left(1-\frac{(x-x')^2}{5s^2}\right), & \text{if } \frac{(x-x')^2}{s^2} < 5 \\ 0, & \text{otherwise} \end{cases}, \quad s > 0.$$

The bandwidth parameter s, which controls the smoothness of the density estimate, is automatically determined by the algorithm if not provided by the user.

Value

A list containing the following elements:

`merges` A data frame with 2 columns and n rows, showing which observations are merged at each step.

`merge_costs` A numeric vector tracking the cost incurred by each merge $I(Z_m;Y)-I(Z_{m-1};Y)$.

`partitions` A list containing n sub-lists. Each sub-list includes the cluster partition at each step.

`I_Z_Y` A numeric vector including the mutual information $I(Z_m;Y)$ as the number of clusters m increases.

`I_X_Y` A numeric value of the mutual information $I(X;Y)$ between observation indices and location.

`info_ret` A numeric vector of length n including the fraction of the original information retained after each merge.

`dendrogram` A dendrogram visualising the cluster hierarchy. The height is determined by the cost of cluster merges.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Slonim N, Tishby N (1999). "Agglomerative Information Bottleneck." *Advances in Neural Information Processing Systems*, **12**.

Silverman BW (1998). *Density Estimation for Statistics and Data Analysis (1st Ed.)*. Routledge.

Epanechnikov VA (1969). "Non-parametric estimation of a multivariate probability density." *Theory of Probability & Its Applications*, **14**(1), 153–158.

See Also

AIBmix, AIBcat

Examples

```
# Generate simulated continuous data
set.seed(123)
X <- matrix(rnorm(1000), ncol = 5)  # 200 observations, 5 features

# Run AIBcont with automatic bandwidth selection
result <- AIBcont(X = X, s = -1, scale = TRUE)

# Print clustering results
plot(result$dendrogram, xlab = "", sub = "")  # Plot dendrogram
```

AIBmix — *Agglomerative Information Bottleneck Clustering for Mixed-Type Data*

Description

The AIBmix function implements the Agglomerative Information Bottleneck (AIB) algorithm for hierarchical clustering of datasets containing mixed-type variables, including categorical (nominal and ordinal) and continuous variables. This method merges clusters so that information retention is maximised at each step to create meaningful clusters, leveraging bandwidth parameters to handle different categorical data types (nominal and ordinal) effectively (Slonim and Tishby 1999).

Usage

```
AIBmix(X, catcols, contcols, lambda = -1, s = -1,
       scale = TRUE, contkernel = "gaussian",
       nomkernel = "aitchisonaitken", ordkernel = "liracine",
       cat_first = FALSE)
```

Arguments

`X` A data frame containing the categorical data to be clustered. All variables should be categorical, either `factor` (for nominal variables) or `ordered` (for ordinal variables).

`catcols` A vector indicating the indices of the categorical variables in `X`.

`contcols` A vector indicating the indices of the continuous variables in `X`.

`lambda` A numeric value or vector specifying the bandwidth parameter for categorical variables. The default value is -1, which enables automatic determination of the optimal bandwidth. For nominal variables and `nomkernel = 'aitchisonaitken'`, the maximum allowable value of `lambda` is $(l-1)/l$, where l represents the number of categories, whereas for `nomkernel = 'liracine'` the maximum allowable value is 1. For ordinal variables, the maximum allowable value of `lambda` is 1, regardless of what `ordkernel` is being used.

`s` A numeric value or vector specifying the bandwidth parameter(s) for continuous variables. The values must be greater than 0. The default value is -1, which enables the automatic selection of optimal bandwidth(s).

`scale` A logical value indicating whether the continuous variables should be scaled to have unit variance before clustering. Defaults to `TRUE`.

`contkernel` Kernel used for continuous variables. Can be one of `gaussian` (default) or `epanechnikov`.

`nomkernel` Kernel used for nominal (unordered categorical) variables. Can be one of `aitchisonaitken` (default) or `liracine`.

`ordkernel` Kernel used for ordinal (ordered categorical) variables. Can be one of `liracine` (default) or `wangvanryzin`.

`cat_first` A logical value indicating whether bandwidth selection is prioritised for the categorical variables, instead of the continuous. Defaults to `FALSE`. Set to `TRUE` if you suspect that the continuous variables are not informative of the cluster structure. Can only be `TRUE` when all bandwidths are selected automatically (i.e. `s = -1`, `lambda = -1`).

Details

The `AIBmix` function produces a hierarchical agglomerative clustering of the data while retaining maximal information about the original variable distributions. The Agglomerative Information Bottleneck algorithm uses an information-theoretic criterion to merge clusters so that information retention is maximised at each step, hence creating meaningful clusters with maximal information about the original distribution. Bandwidth parameters for categorical (nominal, ordinal) and continuous variables are adaptively determined if not provided. This process identifies stable and interpretable cluster assignments by maximizing mutual information while controlling complexity. The method is well-suited for datasets with mixed-type variables and integrates information from all variable types effectively.

The following kernel functions can be used to estimate densities for the clustering procedure. For continuous variables:

- *Gaussian (RBF) kernel (Silverman 1998):*

$$K_c\left(\frac{x-x'}{s}\right) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{(x-x')^2}{2s^2}\right\}, \quad s > 0.$$

- *Epanechnikov kernel (Epanechnikov 1969):*

$$K_c(x-x';s) = \begin{cases} \frac{3}{4\sqrt{5}}\left(1-\frac{(x-x')^2}{5s^2}\right), & \text{if } \frac{(x-x')^2}{s^2} < 5 \\ 0, & \text{otherwise} \end{cases}, \quad s > 0.$$

For nominal (unordered categorical variables):

- *Aitchison & Aitken kernel (Aitchison and Aitken 1976):*

$$K_u(x=x';\lambda) = \begin{cases} 1-\lambda, & \text{if } x = x' \\ \frac{\lambda}{\ell-1}, & \text{otherwise} \end{cases}, \quad 0 \le \lambda \le \frac{\ell-1}{\ell}.$$

- *Li & Racine kernel (Ouyang et al. 2006):*

$$K_u(x=x';\lambda) = \begin{cases} 1, & \text{if } x = x' \\ \lambda, & \text{otherwise} \end{cases}, \quad 0 \le \lambda \le 1.$$

For ordinal (ordered categorical) variables:

- *Li & Racine kernel (Li and Racine 2003):*

$$K_o(x=x';\nu) = \begin{cases} 1, & \text{if } x = x' \\ \nu^{|x-x'|}, & \text{otherwise} \end{cases}, \quad 0 \le \nu \le 1.$$

- *Wang & van Ryzin kernel (Wang and Van Ryzin 1981):*

$$K_o(x=x';\nu) = \begin{cases} 1-\nu, & \text{if } x = x' \\ \frac{1-\nu}{2}\nu^{|x-x'|}, & \text{otherwise} \end{cases}, \quad 0 \le \nu \le 1.$$

The bandwidth parameters s, λ, and ν control the smoothness of the density estimate and are automatically determined by the algorithm if not provided by the user. ℓ is the number of levels of the categorical variable. For ordinal variables, the lambda parameter of the function is used to define ν.

Value

A list containing the following elements:

`merges` A data frame with 2 columns and n rows, showing which observations are merged at each step.

`merge_costs` A numeric vector tracking the cost incurred by each merge $I(Z_m;Y) - I(Z_{m-1};Y)$.

`partitions` A list containing n sub-lists. Each sub-list includes the cluster partition at each step.

`I_Z_Y` A numeric vector including the mutual information $I(Z_m;Y)$ as the number of clusters m increases.

`I_X_Y` A numeric value of the mutual information $I(X;Y)$ between observation indices and location.

`info_ret` A numeric vector of length n including the fraction of the original information retained after each merge.

`dendrogram` A dendrogram visualising the cluster hierarchy. The height is determined by the cost of cluster merges.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Slonim N, Tishby N (1999). "Agglomerative Information Bottleneck." *Advances in Neural Information Processing Systems*, **12**.

Aitchison J, Aitken CG (1976). "Multivariate binary discrimination by the kernel method." *Biometrika*, **63**(3), 413–420.

Li Q, Racine J (2003). "Nonparametric estimation of distributions with categorical and continuous data." *Journal of Multivariate Analysis*, **86**(2), 266–292.

Silverman BW (1998). *Density Estimation for Statistics and Data Analysis (1st Ed.)*. Routledge.

Ouyang D, Li Q, Racine J (2006). "Cross-validation and the estimation of probability distributions with categorical data." *Journal of Nonparametric Statistics*, **18**(1), 69–100.

Wang M, Van Ryzin J (1981). "A class of smooth estimators for discrete distributions." *Biometrika*, **68**(1), 301–309.

Epanechnikov VA (1969). "Non-parametric estimation of a multivariate probability density." *Theory of Probability & Its Applications*, **14**(1), 153–158.

See Also

AIBcat, AIBcont

Examples

```
# Example dataset with categorical, ordinal, and continuous variables
set.seed(123)
data <- data.frame(
  cat_var = factor(sample(letters[1:3], 100, replace = TRUE)),    # Nominal categorical variable
  ord_var = factor(sample(c("low", "medium", "high"), 100, replace = TRUE),
                   levels = c("low", "medium", "high"),
                   ordered = TRUE),                               # Ordinal variable
  cont_var1 = rnorm(100),                                       # Continuous variable 1
  cont_var2 = runif(100)                                        # Continuous variable 2
)

# Perform Mixed-Type Hierarchical Clustering with Agglomerative IB
result <- AIBmix(X = data, catcols = 1:2, contcols = 3:4, lambda = -1, s = -1, scale = TRUE)

# Print clustering results
plot(result$dendrogram, xlab = "", sub = "")  # Plot dendrogram
```

DIBcat — *Cluster Categorical Data Using the Deterministic Information Bottleneck Algorithm*

Description

The DIBcat function implements the Deterministic Information Bottleneck (DIB) algorithm for clustering datasets containing categorical variables. This method balances information retention and data compression to create meaningful clusters, leveraging bandwidth parameters to handle different categorical data types (nominal and ordinal) effectively (Costa et al. 2025).

Usage

```
DIBcat(X, ncl, randinit = NULL, lambda = -1,
       maxiter = 100, nstart = 100,
       nomkernel = "aitchisonaitken", ordkernel = "liracine",
       verbose = FALSE)
```

Arguments

`X`	A data frame containing the categorical data to be clustered. All variables should be categorical, either `factor` (for nominal variables) or `ordered` (for ordinal variables).
`ncl`	An integer specifying the number of clusters to form.
`randinit`	Optional. A vector specifying initial cluster assignments. If `NULL`, cluster assignments are initialized randomly.
`lambda`	A numeric value or vector specifying the bandwidth parameter for categorical variables. The default value is -1, which enables automatic determination of the optimal bandwidth. For nominal variables and `nomkernel = 'aitchisonaitken'`, the maximum allowable value of `lambda` is $(l-1)/l$, where l represents the number of categories, whereas for `nomkernel = 'liracine'` the maximum allowable value is 1. For ordinal variables, the maximum allowable value of `lambda` is 1, regardless of what `ordkernel` is being used.
`maxiter`	The maximum number of iterations for the clustering algorithm. Defaults to 100.
`nstart`	The number of random initializations to run. The best clustering result (based on the information-theoretic criterion) is returned. Defaults to 100.
`nomkernel`	Kernel used for nominal (unordered categorical) variables. Can be one of `aitchisonaitken` (default) or `liracine`.
`ordkernel`	Kernel used for ordinal (ordered categorical) variables. Can be one of `liracine` (default) or `wangvanryzin`.
`verbose`	Logical. Defaults to `FALSE` to suppress progress messages. Change to `TRUE` to print.

Details

The `DIBcat` function applies the Deterministic Information Bottleneck algorithm to cluster datasets containing only categorical variables, both nominal and ordinal. The algorithm optimizes an information-theoretic objective to balance the trade-off between data compression and the retention of information about the original distribution.

To estimate the distributions of categorical features, the function utilizes specialized kernel functions.

For nominal (unordered categorical) variables, the kernel functions implemented are:

- *Aitchison & Aitken kernel (Aitchison and Aitken 1976):*

$$K_u(x = x'; \lambda) = \begin{cases} 1-\lambda, & \text{if } x = x' \\ \frac{\lambda}{\ell-1}, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq \frac{\ell-1}{\ell}.$$

- *Li & Racine kernel (Ouyang et al. 2006):*

$$K_u(x = x'; \lambda) = \begin{cases} 1, & \text{if } x = x' \\ \lambda, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq 1.$$

For ordinal (ordered categorical) variables, the kernel functions implemented are:

- *Li & Racine kernel (Li and Racine 2003):*

$$K_o(x = x'; \nu) = \begin{cases} 1, & \text{if } x = x' \\ \nu^{|x-x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

- *Wang & van Ryzin kernel (Wang and Van Ryzin 1981):*

$$K_o(x = x'; \nu) = \begin{cases} 1 - \nu, & \text{if } x = x' \\ \frac{1-\nu}{2}\nu^{|x-x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

Here, λ, and ν are bandwidth or smoothing parameters, while ℓ is the number of levels of the categorical variable. The lambda parameter is automatically determined by the algorithm if not provided by the user. For ordinal variables, the lambda parameter of the function is used to define ν.

Value

A list containing the following elements:

`Cluster` An integer vector indicating the cluster assignment for each data point at convergence.

`Entropy` A numeric value representing the entropy of the cluster assignments at the end of the iterative procedure.

`CondEntropy` A numeric value representing the conditional entropy of cluster assignment, given the observation weights $H(T \mid X)$.

`MutualInfo` A numeric value representing the mutual information, $I(Y;T)$, between the data distribution and the cluster assignments.

`InfoXT` A numeric value representing the mutual information, $I(X;T)$, between the original observations weights (X) and the cluster assignments (T).

`lambda` A numeric vector of bandwidth parameters for categorical variables, controlling how categories are weighted in the clustering.

`beta` A numeric vector of the final beta values used during the iterative optimization.

`ents` A numeric vector tracking the entropy values across iterations, providing insights into the convergence pattern.

`mis` A numeric vector tracking the mutual information values across iterations.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Costa E, Papatsouma I, Markos A (2025). “A Deterministic Information Bottleneck Method for Clustering Mixed-Type Data.” doi:10.48550/arXiv.2407.03389, arXiv:2407.03389, `https://arxiv.org/abs/2407.03389`.

Aitchison J, Aitken CG (1976). “Multivariate binary discrimination by the kernel method.” *Biometrika*, **63**(3), 413–420.

Li Q, Racine J (2003). “Nonparametric estimation of distributions with categorical and continuous data.” *Journal of Multivariate Analysis*, **86**(2), 266–292.

Ouyang D, Li Q, Racine J (2006). “Cross-validation and the estimation of probability distributions with categorical data.” *Journal of Nonparametric Statistics*, **18**(1), 69–100.

Wang M, Van Ryzin J (1981). “A class of smooth estimators for discrete distributions.” *Biometrika*, **68**(1), 301–309.

See Also

DIBmix, DIBcont

Examples

```
# Simulated categorical data
set.seed(123)
X <- data.frame(
  Var1 = as.factor(sample(letters[1:3], 200, replace = TRUE)),  # Nominal variable
  Var2 = as.factor(sample(letters[4:6], 200, replace = TRUE)),  # Nominal variable
  Var3 = factor(sample(c("low", "medium", "high"), 200, replace = TRUE),
                levels = c("low", "medium", "high"), ordered = TRUE)  # Ordinal variable
)

# Run DIBcat with automatic lambda selection and multiple initializations
result <- DIBcat(X = X, ncl = 3, lambda = -1, nstart = 10)

# Print clustering results
print(result$Cluster)       # Cluster assignments
print(result$Entropy)       # Final entropy
print(result$MutualInfo)    # Mutual information
```

DIBcont *Cluster Continuous Data Using the Deterministic Information Bottleneck Algorithm*

Description

The DIBcont function implements the Deterministic Information Bottleneck (DIB) algorithm for clustering continuous data. This method optimizes an information-theoretic objective to preserve relevant information while forming concise and interpretable cluster representations (Costa et al. 2025).

Usage

```
DIBcont(X, ncl, randinit = NULL, s = -1, scale = TRUE,
        maxiter = 100, nstart = 100,
        contkernel = "gaussian", verbose = FALSE)
```

Arguments

X	A numeric matrix or data frame containing the continuous data to be clustered. All variables should be of type numeric.
ncl	An integer specifying the number of clusters to form.
randinit	Optional. A vector specifying initial cluster assignments. If NULL, cluster assignments are initialized randomly.

`s` A numeric value or vector specifying the bandwidth parameter(s) for continuous variables. The values must be greater than 0. The default value is -1, which enables the automatic selection of optimal bandwidth(s).

`scale` A logical value indicating whether the continuous variables should be scaled to have unit variance before clustering. Defaults to `TRUE`.

`maxiter` The maximum number of iterations allowed for the clustering algorithm. Defaults to 100.

`nstart` The number of random initializations to run. The best clustering result (based on the information-theoretic criterion) is returned. Defaults to `100`.

`contkernel` Kernel used for continuous variables. Can be one of `gaussian` (default) or `epanechnikov`.

`verbose` Logical. Default to `FALSE` to suppress progress messages. Change to `TRUE` to print.

Details

The `DIBcont` function applies the Deterministic Information Bottleneck algorithm to cluster datasets comprising only continuous variables. This method leverages an information-theoretic objective to optimize the trade-off between data compression and the preservation of relevant information about the underlying data distribution.

To estimate the distributions of continuous features, the function utilizes specialized kernel functions:

- *Gaussian (RBF) kernel (Silverman 1998):*

$$K_c\left(\frac{x-x'}{s}\right) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{(x-x')^2}{2s^2}\right\}, \quad s > 0.$$

- *Epanechnikov kernel (Epanechnikov 1969):*

$$K_c(x-x';s) = \begin{cases} \frac{3}{4\sqrt{5}}\left(1-\frac{(x-x')^2}{5s^2}\right), & \text{if } \frac{(x-x')^2}{s^2} < 5 \\ 0, & \text{otherwise} \end{cases}, \quad s > 0.$$

The bandwidth parameter s, which controls the smoothness of the density estimate, is automatically determined by the algorithm if not provided by the user.

Value

A list containing the following elements:

`Cluster` An integer vector indicating the cluster assignment for each observation.

`Entropy` A numeric value representing the entropy of the cluster assignments at convergence.

`CondEntropy` A numeric value representing the conditional entropy of cluster assignment, given the observation weights $H(T \mid X)$.

`MutualInfo` A numeric value representing the mutual information, $I(Y;T)$, between the underlying data distribution and the cluster assignments.

`InfoXT` A numeric value representing the mutual information, $I(X;T)$, between the original observations weights (X) and the cluster assignments (T).

`beta` A numeric vector of the final beta values used during the iterative optimization.

s	A numeric value or vector of bandwidth parameters used for the continuous variables. Typically, this will be a single value if all continuous variables share the same bandwidth.
ents	A numeric vector tracking the entropy values over the iterations, providing insight into the convergence process.
mis	A numeric vector tracking the mutual information values over the iterations.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Costa E, Papatsouma I, Markos A (2025). "A Deterministic Information Bottleneck Method for Clustering Mixed-Type Data." doi:10.48550/arXiv.2407.03389, arXiv:2407.03389, https://arxiv.org/abs/2407.03389.

Silverman BW (1998). *Density Estimation for Statistics and Data Analysis (1st Ed.)*. Routledge.

Epanechnikov VA (1969). "Non-parametric estimation of a multivariate probability density." *Theory of Probability & Its Applications*, **14**(1), 153–158.

See Also

DIBmix, DIBcat

Examples

```
# Generate simulated continuous data
set.seed(123)
X <- matrix(rnorm(200), ncol = 5)  # 200 observations, 5 features

# Run DIBcont with automatic bandwidth selection and multiple initializations
result <- DIBcont(X = X, ncl = 3, s = -1, nstart = 10)

# Print clustering results
print(result$Cluster)       # Cluster assignments
print(result$Entropy)       # Final entropy
print(result$MutualInfo)    # Mutual information
```

DIBmix *Deterministic Information Bottleneck Clustering for Mixed-Type Data*

Description

The DIBmix function implements the Deterministic Information Bottleneck (DIB) algorithm for clustering datasets containing mixed-type variables, including categorical (nominal and ordinal) and continuous variables. This method optimizes an information-theoretic objective to preserve relevant information in the cluster assignments while achieving effective data compression (Costa et al. 2025).

Usage

```
DIBmix(X, ncl, catcols, contcols, randinit = NULL,
       lambda = -1, s = -1, scale = TRUE,
       maxiter = 100, nstart = 100,
       contkernel = "gaussian",
       nomkernel = "aitchisonaitken", ordkernel = "liracine",
       cat_first = FALSE, verbose = FALSE)
```

Arguments

`X` A data frame containing the input data to be clustered. It should include categorical variables (`factor` for nominal and `Ord.factor` for ordinal) and continuous variables (`numeric`).

`ncl` An integer specifying the number of clusters.

`catcols` A vector indicating the indices of the categorical variables in `X`.

`contcols` A vector indicating the indices of the continuous variables in `X`.

`randinit` An optional vector specifying the initial cluster assignments. If `NULL`, cluster assignments are initialized randomly.

`lambda` A numeric value or vector specifying the bandwidth parameter for categorical variables. The default value is -1, which enables automatic determination of the optimal bandwidth. For nominal variables and `nomkernel = 'aitchisonaitken'`, the maximum allowable value of `lambda` is $(l-1)/l$, where l represents the number of categories, whereas for `nomkernel = 'liracine'` the maximum allowable value is 1. For ordinal variables, the maximum allowable value of `lambda` is 1, regardless of what `ordkernel` is being used.

`s` A numeric value or vector specifying the bandwidth parameter(s) for continuous variables. The values must be greater than 0. The default value is -1, which enables the automatic selection of optimal bandwidth(s).

`scale` A logical value indicating whether the continuous variables should be scaled to have unit variance before clustering. Defaults to `TRUE`.

`maxiter` The maximum number of iterations allowed for the clustering algorithm. Defaults to 100.

`nstart` The number of random initializations to run. The best clustering solution is returned. Defaults to 100.

`contkernel` Kernel used for continuous variables. Can be one of `gaussian` (default) or `epanechnikov`.

`nomkernel` Kernel used for nominal (unordered categorical) variables. Can be one of `aitchisonaitken` (default) or `liracine`.

`ordkernel` Kernel used for ordinal (ordered categorical) variables. Can be one of `liracine` (default) or `wangvanryzin`.

`cat_first` A logical value indicating whether bandwidth selection is prioritised for the categorical variables, instead of the continuous. Defaults to `FALSE`. Set to `TRUE` if you suspect that the continuous variables are not informative of the cluster structure. Can only be `TRUE` when all bandwidths are selected automatically (i.e. `s = -1`, `lambda = -1`).

`verbose` Logical. Defaults to `FALSE` to suppress progress messages. Change to `TRUE` to print.

Details

The `DIBmix` function clusters data while retaining maximal information about the original variable distributions. The Deterministic Information Bottleneck algorithm optimizes an information-theoretic objective that balances information preservation and compression. Bandwidth parameters for categorical (nominal, ordinal) and continuous variables are adaptively determined if not provided. This iterative process identifies stable and interpretable cluster assignments by maximizing mutual information while controlling complexity. The method is well-suited for datasets with mixed-type variables and integrates information from all variable types effectively.

The following kernel functions can be used to estimate densities for the clustering procedure. For continuous variables:

- *Gaussian (RBF) kernel (Silverman 1998):*

$$K_c\left(\frac{x-x'}{s}\right) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{(x-x')^2}{2s^2}\right\}, \quad s > 0.$$

- *Epanechnikov kernel (Epanechnikov 1969):*

$$K_c(x-x';s) = \begin{cases} \frac{3}{4\sqrt{5}}\left(1-\frac{(x-x')^2}{5s^2}\right), & \text{if } \frac{(x-x')^2}{s^2} < 5 \\ 0, & \text{otherwise} \end{cases}, \quad s > 0.$$

For nominal (unordered categorical variables):

- *Aitchison & Aitken kernel (Aitchison and Aitken 1976):*

$$K_u(x=x';\lambda) = \begin{cases} 1-\lambda, & \text{if } x = x' \\ \frac{\lambda}{\ell-1}, & \text{otherwise} \end{cases}, \quad 0 \le \lambda \le \frac{\ell-1}{\ell}.$$

- *Li & Racine kernel (Ouyang et al. 2006):*

$$K_u(x=x';\lambda) = \begin{cases} 1, & \text{if } x = x' \\ \lambda, & \text{otherwise} \end{cases}, \quad 0 \le \lambda \le 1.$$

For ordinal (ordered categorical) variables:

- *Li & Racine kernel (Li and Racine 2003):*

$$K_o(x=x';\nu) = \begin{cases} 1, & \text{if } x = x' \\ \nu^{|x-x'|}, & \text{otherwise} \end{cases}, \quad 0 \le \nu \le 1.$$

- *Wang & van Ryzin kernel (Wang and Van Ryzin 1981):*

$$K_o(x=x';\nu) = \begin{cases} 1-\nu, & \text{if } x = x' \\ \frac{1-\nu}{2}\nu^{|x-x'|}, & \text{otherwise} \end{cases}, \quad 0 \le \nu \le 1.$$

The bandwidth parameters s, λ, and ν control the smoothness of the density estimate and are automatically determined by the algorithm if not provided by the user. ℓ is the number of levels of the categorical variable. For ordinal variables, the lambda parameter of the function is used to define ν.

Value

A list containing the following elements:

`Cluster` An integer vector giving the cluster assignments for each data point.

`Entropy` A numeric value representing the entropy of the cluster assignments at convergence.

`CondEntropy` A numeric value representing the conditional entropy of cluster assignment, given the observation weights $H(T \mid X)$.

`MutualInfo` A numeric value representing the mutual information, $I(Y;T)$, between the original labels (Y) and the cluster assignments (T).

`InfoXT` A numeric value representing the mutual information, $I(X;T)$, between the original observations weights (X) and the cluster assignments (T).

`beta` A numeric vector of the final beta values used in the iterative procedure.

`s` A numeric vector of bandwidth parameters used for the continuous variables.

`lambda` A numeric vector of bandwidth parameters used for the categorical variables.

`ents` A numeric vector tracking the entropy values across iterations.

`mis` A numeric vector tracking the mutual information values across iterations.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Costa E, Papatsouma I, Markos A (2025). “A Deterministic Information Bottleneck Method for Clustering Mixed-Type Data.” doi:10.48550/arXiv.2407.03389, arXiv:2407.03389, `https://arxiv.org/abs/2407.03389`.

Aitchison J, Aitken CG (1976). “Multivariate binary discrimination by the kernel method.” *Biometrika*, **63**(3), 413–420.

Li Q, Racine J (2003). “Nonparametric estimation of distributions with categorical and continuous data.” *Journal of Multivariate Analysis*, **86**(2), 266–292.

Silverman BW (1998). *Density Estimation for Statistics and Data Analysis (1st Ed.)*. Routledge.

Ouyang D, Li Q, Racine J (2006). “Cross-validation and the estimation of probability distributions with categorical data.” *Journal of Nonparametric Statistics*, **18**(1), 69–100.

Wang M, Van Ryzin J (1981). “A class of smooth estimators for discrete distributions.” *Biometrika*, **68**(1), 301–309.

Epanechnikov VA (1969). “Non-parametric estimation of a multivariate probability density.” *Theory of Probability & Its Applications*, **14**(1), 153–158.

See Also

`DIBcont`, `DIBcat`

Examples

```
# Example 1: Basic Mixed-Type Clustering
set.seed(123)

# Create a more realistic dataset with mixed variable types
data <- data.frame(
  # Categorical variables
  education = factor(sample(c("High School", "Bachelor", "Master", "PhD"), 150,
                           replace = TRUE, prob = c(0.4, 0.3, 0.2, 0.1))),
  employment = factor(sample(c("Full-time", "Part-time", "Unemployed"), 150,
                            replace = TRUE, prob = c(0.6, 0.25, 0.15))),

  # Ordinal variable
  satisfaction = factor(sample(c("Low", "Medium", "High"), 150, replace = TRUE),
                       levels = c("Low", "Medium", "High"), ordered = TRUE),

  # Continuous variables
  income = rlnorm(150, meanlog = 10, sdlog = 0.5),  # Log-normal income
  age = rnorm(150, mean = 35, sd = 10),             # Normally distributed age
  experience = rpois(150, lambda = 8)               # Years of experience
)

# Perform Mixed-Type Clustering
result <- DIBmix(X = data, ncl = 3,
                catcols = c(1, 2, 3),    # education, employment, satisfaction
                contcols = c(4, 5, 6),   # income, age, experience
                nstart = 50)

# View results
print(paste("Number of clusters found:", length(unique(result$Cluster))))
print(paste("Mutual Information:", round(result$MutualInfo, 3)))
table(result$Cluster)

# Example 2: Comparing cat_first parameter
# When categorical variables are more informative
result_cat_first <- DIBmix(X = data, ncl = 3,
                          catcols = c(1, 2, 3),
                          contcols = c(4, 5, 6),
                          cat_first = TRUE,  # Prioritize categorical variables
                          nstart = 50)

# When continuous variables are more informative (default)
result_cont_first <- DIBmix(X = data, ncl = 3,
                           catcols = c(1, 2, 3),
                           contcols = c(4, 5, 6),
                           cat_first = FALSE,
                           nstart = 50)

# Compare clustering performance
if (requireNamespace("mclust", quietly = TRUE)){  # For adjustedRandIndex
  print(paste("Agreement between approaches:",
              round(mclust::adjustedRandIndex(result_cat_first$Cluster,
                    result_cont_first$Cluster), 3)))
  }
```

GIBcat — *Cluster Categorical Data Using the Generalised Information Bottleneck Algorithm*

Description

The GIBcat function implements the Generalised Information Bottleneck (GIB) algorithm for fuzzy clustering of datasets containing categorical variables. This method balances information retention and data compression to create meaningful clusters, leveraging bandwidth parameters to handle different categorical data types (nominal and ordinal) effectively (Strouse and Schwab 2019).

Usage

```
GIBcat(X, ncl, beta, alpha, randinit = NULL, lambda = -1,
       maxiter = 100, nstart = 100,
       nomkernel = "aitchisonaitken", ordkernel = "liracine",
       verbose = FALSE)
```

Arguments

X	A data frame containing the categorical data to be clustered. All variables should be categorical, either factor (for nominal variables) or ordered (for ordinal variables).
beta	Regularisation strength.
alpha	Strength of conditional entropy term. Must be in the range $[0, 1]$. Setting alpha = 0 calls the DIBcat function and ignores the value of beta, while alpha = 1 calls IBcat instead.
ncl	An integer specifying the number of clusters to form.
randinit	Optional. A vector specifying initial cluster assignments. If NULL, cluster assignments are initialized randomly.
lambda	A numeric value or vector specifying the bandwidth parameter for categorical variables. The default value is -1, which enables automatic determination of the optimal bandwidth. For nominal variables and nomkernel = 'aitchisonaitken', the maximum allowable value of lambda is $(l-1)/l$, where l represents the number of categories, whereas for nomkernel = 'liracine' the maximum allowable value is 1. For ordinal variables, the maximum allowable value of lambda is 1, regardless of what ordkernel is being used.
maxiter	The maximum number of iterations for the clustering algorithm. Defaults to 100.
nstart	The number of random initializations to run. The best clustering result (based on the information-theoretic criterion) is returned. Defaults to 100.
nomkernel	Kernel used for nominal (unordered categorical) variables. Can be one of aitchisonaitken (default) or liracine.
ordkernel	Kernel used for ordinal (ordered categorical) variables. Can be one of liracine (default) or wangvanryzin.
verbose	Logical. Defaults to FALSE to suppress progress messages. Change to TRUE to print.

Details

The `GIBcat` function applies the Generalised Information Bottleneck algorithm to do fuzzy clustering of datasets containing only categorical variables, both nominal and ordinal. The algorithm optimizes an information-theoretic objective to balance the trade-off between data compression and the retention of information about the original distribution. Set $\alpha = 1$ and $\alpha = 0$ to recover the Information Bottleneck and its Deterministic variant, respectively. If $\alpha = 0$, the algorithm ignores the value of the regularisation parameter β.

To estimate the distributions of categorical features, the function utilizes specialized kernel functions.

For nominal (unordered categorical) variables, the kernel functions implemented are:

- *Aitchison & Aitken kernel (Aitchison and Aitken 1976):*

$$K_u(x = x'; \lambda) = \begin{cases} 1 - \lambda, & \text{if } x = x' \\ \frac{\lambda}{\ell - 1}, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq \frac{\ell - 1}{\ell}.$$

- *Li & Racine kernel (Ouyang et al. 2006):*

$$K_u(x = x'; \lambda) = \begin{cases} 1, & \text{if } x = x' \\ \lambda, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq 1.$$

For ordinal (ordered categorical) variables, the kernel functions implemented are:

- *Li & Racine kernel (Li and Racine 2003):*

$$K_o(x = x'; \nu) = \begin{cases} 1, & \text{if } x = x' \\ \nu^{|x - x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

- *Wang & van Ryzin kernel (Wang and Van Ryzin 1981):*

$$K_o(x = x'; \nu) = \begin{cases} 1 - \nu, & \text{if } x = x' \\ \frac{1 - \nu}{2} \nu^{|x - x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

Here, λ, and ν are bandwidth or smoothing parameters, while ℓ is the number of levels of the categorical variable. The lambda parameter is automatically determined by the algorithm if not provided by the user. For ordinal variables, the lambda parameter of the function is used to define ν.

Value

A list containing the following elements:

`Cluster` A cluster membership matrix.

`Entropy` A numeric value representing the entropy of the cluster assignments at the end of the iterative procedure.

`CondEntropy` A numeric value representing the conditional entropy of cluster assignment, given the observation weights $H(T \mid X)$.

`MutualInfo` A numeric value representing the mutual information, $I(Y; T)$, between the data distribution and the cluster assignments.

`InfoXT` A numeric value representing the mutual information, $I(X; T)$, between the original observations weights (X) and the cluster assignments (T).

`beta` A numeric value of the regularisation strength beta used.

`alpha` A numeric value of the strength of conditional entropy used.

`lambda` A numeric vector of bandwidth parameters for categorical variables, controlling how categories are weighted in the clustering.

`ht` A numeric vector tracking the entropy value of the cluster assignments across iterations.

`hy_t` A numeric vector tracking the conditional entropy values between the cluster assignments and observations weights across iterations.

`iyt` A numeric vector tracking the mutual information values between original labels and cluster assignments across iterations.

`losses` A numeric vector tracking the final loss values across iterations.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Strouse DJ, Schwab DJ (2017). "The Deterministic Information Bottleneck." *Neural Computation*, **29**(6), 1611–1630.

Aitchison J, Aitken CG (1976). "Multivariate binary discrimination by the kernel method." *Biometrika*, **63**(3), 413–420.

Li Q, Racine J (2003). "Nonparametric estimation of distributions with categorical and continuous data." *Journal of Multivariate Analysis*, **86**(2), 266–292.

Ouyang D, Li Q, Racine J (2006). "Cross-validation and the estimation of probability distributions with categorical data." *Journal of Nonparametric Statistics*, **18**(1), 69–100.

Wang M, Van Ryzin J (1981). "A class of smooth estimators for discrete distributions." *Biometrika*, **68**(1), 301–309.

See Also

`GIBmix`, `GIBcont`

Examples

```
# Simulated categorical data
set.seed(123)
X <- data.frame(
  Var1 = as.factor(sample(letters[1:3], 200, replace = TRUE)),  # Nominal variable
  Var2 = as.factor(sample(letters[4:6], 200, replace = TRUE)),  # Nominal variable
  Var3 = factor(sample(c("low", "medium", "high"), 200, replace = TRUE),
                levels = c("low", "medium", "high"), ordered = TRUE)  # Ordinal variable
)

# Run GIBcat with automatic lambda selection and multiple initializations
result <- GIBcat(X = X, ncl = 2, beta = 25, alpha = 0.75, lambda = -1, nstart = 10)

# Print clustering results
print(result$Cluster)       # Cluster membership matrix
print(result$Entropy)       # Entropy of final clustering
print(result$CondEntropy)   # Conditional entropy of final clustering
print(result$MutualInfo)    # Mutual information between Y and T
```

GIBcont *Cluster Continuous Data Using the Generalised Information Bottleneck Algorithm*

Description

The GIBcont function implements the Generalised Information Bottleneck (GIB) algorithm for fuzzy clustering of continuous data. This method optimizes an information-theoretic objective to preserve relevant information while forming concise and interpretable cluster representations (Strouse and Schwab 2019).

Usage

```
GIBcont(X, ncl, beta, alpha, randinit = NULL, s = -1, scale = TRUE,
        maxiter = 100, nstart = 100, contkernel = "gaussian",
        verbose = FALSE)
```

Arguments

X A numeric matrix or data frame containing the continuous data to be clustered. All variables should be of type numeric.

ncl An integer specifying the number of clusters to form.

beta Regularisation strength.

alpha Strength of conditional entropy term. Must be in the range $[0, 1]$. Setting alpha = 0 calls the DIBcont function and ignores the value of beta, while alpha = 1 calls IBcont instead.

randinit Optional. A vector specifying initial cluster assignments. If NULL, cluster assignments are initialized randomly.

s A numeric value or vector specifying the bandwidth parameter(s) for continuous variables. The values must be greater than 0. The default value is -1, which enables the automatic selection of optimal bandwidth(s).

scale A logical value indicating whether the continuous variables should be scaled to have unit variance before clustering. Defaults to TRUE.

maxiter The maximum number of iterations allowed for the clustering algorithm. Defaults to 100.

nstart The number of random initializations to run. The best clustering result (based on the information-theoretic criterion) is returned. Defaults to 100.

contkernel Kernel used for continuous variables. Can be one of gaussian (default) or epanechnikov.

verbose Logical. Default to FALSE to suppress progress messages. Change to TRUE to print.

Details

The GIBcont function applies the Generalised Information Bottleneck algorithm to do fuzzy clustering of datasets comprising only continuous variables. This method leverages an information-theoretic objective to optimize the trade-off between data compression and the preservation of relevant information about the underlying data distribution. Set $\alpha = 1$ and $\alpha = 0$ to recover the

Information Bottleneck and its Deterministic variant, respectively. If $\alpha = 0$, the algorithm ignores the value of the regularisation parameter β.

To estimate the distributions of continuous features, the function utilizes specialized kernel functions:

- *Gaussian (RBF) kernel (Silverman 1998):*

$$K_c\left(\frac{x - x'}{s}\right) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{(x - x')^2}{2s^2}\right\}, \quad s > 0.$$

- *Epanechnikov kernel (Epanechnikov 1969):*

$$K_c(x - x'; s) = \begin{cases} \frac{3}{4\sqrt{5}}\left(1 - \frac{(x-x')^2}{5s^2}\right), & \text{if } \frac{(x-x')^2}{s^2} < 5 \\ 0, & \text{otherwise} \end{cases}, \quad s > 0.$$

The bandwidth parameter s, which controls the smoothness of the density estimate, is automatically determined by the algorithm if not provided by the user.

Value

A list containing the following elements:

`Cluster` A cluster membership matrix.

`Entropy` A numeric value representing the entropy of the cluster assignments at the end of the iterative procedure.

`CondEntropy` A numeric value representing the conditional entropy of cluster assignment, given the observation weights $H(T \mid X)$.

`MutualInfo` A numeric value representing the mutual information, $I(Y;T)$, between the data distribution and the cluster assignments.

`InfoXT` A numeric value representing the mutual information, $I(X;T)$, between the original observations weights (X) and the cluster assignments (T).

`beta` A numeric value of the regularisation strength beta used.

`alpha` A numeric value of the strength of conditional entropy used.

`s` A numeric vector of bandwidth parameters used for the continuous variables.

`ht` A numeric vector tracking the entropy value of the cluster assignments across iterations.

`hy_t` A numeric vector tracking the conditional entropy values between the cluster assignments and observations weights across iterations.

`iyt` A numeric vector tracking the mutual information values between original labels and cluster assignments across iterations.

`losses` A numeric vector tracking the final loss values across iterations.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Strouse DJ, Schwab DJ (2017). “The Deterministic Information Bottleneck.” *Neural Computation*, **29**(6), 1611–1630.

Silverman BW (1998). *Density Estimation for Statistics and Data Analysis (1st Ed.)*. Routledge.

Epanechnikov VA (1969). “Non-parametric estimation of a multivariate probability density.” *Theory of Probability & Its Applications*, **14**(1), 153–158.

See Also

GIBmix, GIBcat

Examples

```
# Generate simulated continuous data
set.seed(123)
X <- matrix(rnorm(200), ncol = 5)  # 200 observations, 5 features

# Run GIBcont with automatic bandwidth selection and multiple initializations
result <- GIBcont(X = X, ncl = 2, beta = 50, alpha = 0.75, s = -1, nstart = 10)

# Print clustering results
print(result$Cluster)       # Cluster membership matrix
print(result$Entropy)       # Entropy of final clustering
print(result$CondEntropy)   # Conditional entropy of final clustering
print(result$MutualInfo)    # Mutual information between Y and T
```

GIBmix *Generalised Information Bottleneck Clustering for Mixed-Type Data*

Description

The `GIBmix` function implements the Generalised Information Bottleneck (GIB) algorithm for clustering datasets containing mixed-type variables, including categorical (nominal and ordinal) and continuous variables. This method optimizes an information-theoretic objective to preserve relevant information in the cluster assignments while achieving effective data compression (Strouse and Schwab 2017).

Usage

```
GIBmix(X, ncl, beta, alpha, catcols, contcols, randinit = NULL,
       lambda = -1, s = -1, scale = TRUE,
       maxiter = 100, nstart = 100,
       contkernel = "gaussian",
       nomkernel = "aitchisonaitken", ordkernel = "liracine",
       cat_first = FALSE, verbose = FALSE)
```

Arguments

`X`	A data frame containing the input data to be clustered. It should include categorical variables (`factor` for nominal and `Ord.factor` for ordinal) and continuous variables (`numeric`).

`ncl` An integer specifying the number of clusters.

`beta` Regularisation strength.

`alpha` Strength of conditional entropy term. Must be in the range $[0, 1]$. Setting `alpha = 0` calls the `DIBmix` function and ignores the value of `beta`, while `alpha = 1` calls `IBmix` instead.

`catcols` A vector indicating the indices of the categorical variables in `X`.

`contcols` A vector indicating the indices of the continuous variables in `X`.

`randinit` An optional vector specifying the initial cluster assignments. If `NULL`, cluster assignments are initialized randomly.

`lambda` A numeric value or vector specifying the bandwidth parameter for categorical variables. The default value is -1, which enables automatic determination of the optimal bandwidth. For nominal variables and `nomkernel = 'aitchisonaitken'`, the maximum allowable value of `lambda` is $(l-1)/l$, where l represents the number of categories, whereas for `nomkernel = 'liracine'` the maximum allowable value is 1. For ordinal variables, the maximum allowable value of `lambda` is 1, regardless of what `ordkernel` is being used.

`s` A numeric value or vector specifying the bandwidth parameter(s) for continuous variables. The values must be greater than 0. The default value is -1, which enables the automatic selection of optimal bandwidth(s).

`scale` A logical value indicating whether the continuous variables should be scaled to have unit variance before clustering. Defaults to `TRUE`.

`maxiter` The maximum number of iterations allowed for the clustering algorithm. Defaults to 100.

`nstart` The number of random initializations to run. The best clustering solution is returned. Defaults to 100.

`contkernel` Kernel used for continuous variables. Can be one of `gaussian` (default) or `epanechnikov`.

`nomkernel` Kernel used for nominal (unordered categorical) variables. Can be one of `aitchisonaitken` (default) or `liracine`.

`ordkernel` Kernel used for ordinal (ordered categorical) variables. Can be one of `liracine` (default) or `wangvanryzin`.

`cat_first` A logical value indicating whether bandwidth selection is prioritised for the categorical variables, instead of the continuous. Defaults to `FALSE`. Set to `TRUE` if you suspect that the continuous variables are not informative of the cluster structure. Can only be `TRUE` when all bandwidths are selected automatically (i.e. `s = -1`, `lambda = -1`).

`verbose` Logical. Defaults to `FALSE` to suppress progress messages. Change to `TRUE` to print.

Details

The `GIBmix` function produces a fuzzy clustering of the data while retaining maximal information about the original variable distributions. The Generalised Information Bottleneck algorithm optimizes an information-theoretic objective that balances information preservation and compression. Bandwidth parameters for categorical (nominal, ordinal) and continuous variables are adaptively determined if not provided. This iterative process identifies stable and interpretable cluster assignments by maximizing mutual information while controlling complexity. The method is well-suited for datasets with mixed-type variables and integrates information from all variable types effectively.

Set $\alpha = 1$ and $\alpha = 0$ to recover the Information Bottleneck and its Deterministic variant, respectively. If $\alpha = 0$, the algorithm ignores the value of the regularisation parameter β.

The following kernel functions can be used to estimate densities for the clustering procedure. For continuous variables:

- *Gaussian (RBF) kernel (Silverman 1998):*

$$K_c\left(\frac{x - x'}{s}\right) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{(x - x')^2}{2s^2}\right\}, \quad s > 0.$$

- *Epanechnikov kernel (Epanechnikov 1969):*

$$K_c(x - x'; s) = \begin{cases} \frac{3}{4\sqrt{5}}\left(1 - \frac{(x-x')^2}{5s^2}\right), & \text{if } \frac{(x-x')^2}{s^2} < 5 \\ 0, & \text{otherwise} \end{cases}, \quad s > 0.$$

For nominal (unordered categorical variables):

- *Aitchison & Aitken kernel (Aitchison and Aitken 1976):*

$$K_u(x = x'; \lambda) = \begin{cases} 1 - \lambda, & \text{if } x = x' \\ \frac{\lambda}{\ell - 1}, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq \frac{\ell - 1}{\ell}.$$

- *Li & Racine kernel (Ouyang et al. 2006):*

$$K_u(x = x'; \lambda) = \begin{cases} 1, & \text{if } x = x' \\ \lambda, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq 1.$$

For ordinal (ordered categorical) variables:

- *Li & Racine kernel (Li and Racine 2003):*

$$K_o(x = x'; \nu) = \begin{cases} 1, & \text{if } x = x' \\ \nu^{|x - x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

- *Wang & van Ryzin kernel (Wang and Van Ryzin 1981):*

$$K_o(x = x'; \nu) = \begin{cases} 1 - \nu, & \text{if } x = x' \\ \frac{1-\nu}{2}\nu^{|x - x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

The bandwidth parameters s, λ, and ν control the smoothness of the density estimate and are automatically determined by the algorithm if not provided by the user. ℓ is the number of levels of the categorical variable. For ordinal variables, the lambda parameter of the function is used to define ν.

Value

A list containing the following elements:

`Cluster` A cluster membership matrix.

`Entropy` A numeric value representing the entropy of the cluster assignments at the end of the iterative procedure.

`CondEntropy` A numeric value representing the conditional entropy of cluster assignment, given the observation weights $H(T \mid X)$.

`MutualInfo` A numeric value representing the mutual information, $I(Y;T)$, between the data distribution and the cluster assignments.

`InfoXT` A numeric value representing the mutual information, $I(X;T)$, between the original observations weights (X) and the cluster assignments (T).

`beta` A numeric value of the regularisation strength beta used.

`alpha` A numeric value of the strength of conditional entropy used.

`s` A numeric vector of bandwidth parameters used for the continuous variables.

`lambda` A numeric vector of bandwidth parameters used for the categorical variables.

`ht` A numeric vector tracking the entropy value of the cluster assignments across iterations.

`hy_t` A numeric vector tracking the conditional entropy values between the cluster assignments and observations weights across iterations.

`iyt` A numeric vector tracking the mutual information values between original labels and cluster assignments across iterations.

`losses` A numeric vector tracking the final loss values across iterations.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Strouse DJ, Schwab DJ (2017). "The Deterministic Information Bottleneck." *Neural Computation*, **29**(6), 1611–1630.

Aitchison J, Aitken CG (1976). "Multivariate binary discrimination by the kernel method." *Biometrika*, **63**(3), 413–420.

Li Q, Racine J (2003). "Nonparametric estimation of distributions with categorical and continuous data." *Journal of Multivariate Analysis*, **86**(2), 266–292.

Silverman BW (1998). *Density Estimation for Statistics and Data Analysis (1st Ed.)*. Routledge.

Ouyang D, Li Q, Racine J (2006). "Cross-validation and the estimation of probability distributions with categorical data." *Journal of Nonparametric Statistics*, **18**(1), 69–100.

Wang M, Van Ryzin J (1981). "A class of smooth estimators for discrete distributions." *Biometrika*, **68**(1), 301–309.

Epanechnikov VA (1969). "Non-parametric estimation of a multivariate probability density." *Theory of Probability & Its Applications*, **14**(1), 153–158.

See Also

`GIBcat`, `GIBcont`

Examples

```
# Example dataset with categorical, ordinal, and continuous variables
set.seed(123)
data <- data.frame(
 cat_var = factor(sample(letters[1:3], 100, replace = TRUE)),     # Nominal categorical variable
  ord_var = factor(sample(c("low", "medium", "high"), 100, replace = TRUE),
                   levels = c("low", "medium", "high"),
                   ordered = TRUE),                          # Ordinal variable
  cont_var1 = rnorm(100),                                  # Continuous variable 1
  cont_var2 = runif(100)                                   # Continuous variable 2
)

# Perform Mixed-Type Fuzzy Clustering with Generalised IB
result <- GIBmix(X = data, ncl = 3, beta = 2, alpha = 0.5,
catcols = 1:2, contcols = 3:4, nstart = 20)

# Print clustering results
print(result$Cluster)       # Cluster membership matrix
print(result$Entropy)       # Entropy of final clustering
print(result$CondEntropy)   # Conditional entropy of final clustering
print(result$MutualInfo)    # Mutual information between Y and T
```

IBcat *Cluster Categorical Data Using the Information Bottleneck Algorithm*

Description

The IBcat function implements the Information Bottleneck (IB) algorithm for fuzzy clustering of datasets containing categorical variables. This method balances information retention and data compression to create meaningful clusters, leveraging bandwidth parameters to handle different categorical data types (nominal and ordinal) effectively (Strouse and Schwab 2019).

Usage

```
IBcat(X, ncl, beta, randinit = NULL, lambda = -1,
      maxiter = 100, nstart = 100,
      nomkernel = "aitchisonaitken", ordkernel = "liracine",
      verbose = FALSE)
```

Arguments

X	A data frame containing the categorical data to be clustered. All variables should be categorical, either factor (for nominal variables) or ordered (for ordinal variables).
beta	Regularisation strength.
ncl	An integer specifying the number of clusters to form.
randinit	Optional. A vector specifying initial cluster assignments. If NULL, cluster assignments are initialized randomly.
lambda	A numeric value or vector specifying the bandwidth parameter for categorical variables. The default value is -1, which enables automatic determination of the optimal bandwidth. For nominal variables and nomkernel = 'aitchisonaitken', the maximum allowable value of lambda is $(l-1)/l$, where l represents the number of categories, whereas for nomkernel = 'liracine' the maximum allowable value is 1. For ordinal variables, the maximum allowable value of lambda is 1, regardless of what ordkernel is being used.
maxiter	The maximum number of iterations for the clustering algorithm. Defaults to 100.
nstart	The number of random initializations to run. The best clustering result (based on the information-theoretic criterion) is returned. Defaults to 100.

nomkernel Kernel used for nominal (unordered categorical) variables. Can be one of aitchisonaitken (default) or liracine.

ordkernel Kernel used for ordinal (ordered categorical) variables. Can be one of liracine (default) or wangvanryzin.

verbose Logical. Defaults to FALSE to suppress progress messages. Change to TRUE to print.

Details

The IBcat function applies the Information Bottleneck algorithm to do fuzzy clustering of datasets containing only categorical variables, both nominal and ordinal. The algorithm optimizes an information-theoretic objective to balance the trade-off between data compression and the retention of information about the original distribution.

To estimate the distributions of categorical features, the function utilizes specialized kernel functions.

For nominal (unordered categorical) variables, the kernel functions implemented are:

- *Aitchison & Aitken kernel (Aitchison and Aitken 1976):*

$$K_u(x = x'; \lambda) = \begin{cases} 1 - \lambda, & \text{if } x = x' \\ \frac{\lambda}{\ell - 1}, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq \frac{\ell - 1}{\ell}.$$

- *Li & Racine kernel (Ouyang et al. 2006):*

$$K_u(x = x'; \lambda) = \begin{cases} 1, & \text{if } x = x' \\ \lambda, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq 1.$$

For ordinal (ordered categorical) variables, the kernel functions implemented are:

- *Li & Racine kernel (Li and Racine 2003):*

$$K_o(x = x'; \nu) = \begin{cases} 1, & \text{if } x = x' \\ \nu^{|x - x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

- *Wang & van Ryzin kernel (Wang and Van Ryzin 1981):*

$$K_o(x = x'; \nu) = \begin{cases} 1 - \nu, & \text{if } x = x' \\ \frac{1-\nu}{2} \nu^{|x - x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

Here, λ, and ν are bandwidth or smoothing parameters, while ℓ is the number of levels of the categorical variable. The lambda parameter is automatically determined by the algorithm if not provided by the user. For ordinal variables, the lambda parameter of the function is used to define ν.

Value

A list containing the following elements:

Cluster A cluster membership matrix.

Entropy A numeric value representing the entropy of the cluster assignments at the end of the iterative procedure.

`CondEntropy` A numeric value representing the conditional entropy of cluster assignment, given the observation weights $H(T \mid X)$.

`MutualInfo` A numeric value representing the mutual information, $I(Y;T)$, between the data distribution and the cluster assignments.

`InfoXT` A numeric value representing the mutual information, $I(X;T)$, between the original observations weights (X) and the cluster assignments (T).

`beta` A numeric value of the regularisation strength beta used.

`lambda` A numeric vector of bandwidth parameters for categorical variables, controlling how categories are weighted in the clustering.

`ixt` A numeric vector tracking the mutual information values between original observation weights and cluster assignments across iterations.

`iyt` A numeric vector tracking the mutual information values between original labels and cluster assignments across iterations.

`losses` A numeric vector tracking the final loss values across iterations.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Strouse DJ, Schwab DJ (2019). "The information bottleneck and geometric clustering." *Neural Computation*, **31**(3), 596–612.

Aitchison J, Aitken CG (1976). "Multivariate binary discrimination by the kernel method." *Biometrika*, **63**(3), 413–420.

Li Q, Racine J (2003). "Nonparametric estimation of distributions with categorical and continuous data." *Journal of Multivariate Analysis*, **86**(2), 266–292.

Ouyang D, Li Q, Racine J (2006). "Cross-validation and the estimation of probability distributions with categorical data." *Journal of Nonparametric Statistics*, **18**(1), 69–100.

Wang M, Van Ryzin J (1981). "A class of smooth estimators for discrete distributions." *Biometrika*, **68**(1), 301–309.

See Also

IBmix, IBcont

Examples

```
# Simulated categorical data
set.seed(123)
X <- data.frame(
  Var1 = as.factor(sample(letters[1:3], 200, replace = TRUE)),  # Nominal variable
  Var2 = as.factor(sample(letters[4:6], 200, replace = TRUE)),  # Nominal variable
  Var3 = factor(sample(c("low", "medium", "high"), 200, replace = TRUE),
                levels = c("low", "medium", "high"), ordered = TRUE)  # Ordinal variable
)

# Run IBcat with automatic lambda selection and multiple initializations
result <- IBcat(X = X, ncl = 3, beta = 15, lambda = -1, nstart = 10)

# Print clustering results
```

```
print(result$Cluster)       # Cluster membership matrix
print(result$InfoXT)       # Mutual information between X and T
print(result$MutualInfo)    # Mutual information between Y and T
```

`IBcont` *Cluster Continuous Data Using the Information Bottleneck Algorithm*

Description

The `IBcont` function implements the Information Bottleneck (IB) algorithm for fuzzy clustering of continuous data. This method optimizes an information-theoretic objective to preserve relevant information while forming concise and interpretable cluster representations (Strouse and Schwab 2019).

Usage

```
IBcont(X, ncl, beta, randinit = NULL, s = -1, scale = TRUE,
       maxiter = 100, nstart = 100, contkernel = "gaussian",
       verbose = FALSE)
```

Arguments

`X` A numeric matrix or data frame containing the continuous data to be clustered. All variables should be of type `numeric`.

`ncl` An integer specifying the number of clusters to form.

`beta` Regularisation strength.

`randinit` Optional. A vector specifying initial cluster assignments. If `NULL`, cluster assignments are initialized randomly.

`s` A numeric value or vector specifying the bandwidth parameter(s) for continuous variables. The values must be greater than 0. The default value is -1, which enables the automatic selection of optimal bandwidth(s).

`scale` A logical value indicating whether the continuous variables should be scaled to have unit variance before clustering. Defaults to `TRUE`.

`maxiter` The maximum number of iterations allowed for the clustering algorithm. Defaults to 100.

`nstart` The number of random initializations to run. The best clustering result (based on the information-theoretic criterion) is returned. Defaults to `100`.

`contkernel` Kernel used for continuous variables. Can be one of `gaussian` (default) or `epanechnikov`.

`verbose` Logical. Default to `FALSE` to suppress progress messages. Change to `TRUE` to print.

Details

The `IBcont` function applies the Information Bottleneck algorithm to do fuzzy clustering of datasets comprising only continuous variables. This method leverages an information-theoretic objective to optimize the trade-off between data compression and the preservation of relevant information about the underlying data distribution.

To estimate the distributions of continuous features, the function utilizes specialized kernel functions:

- *Gaussian (RBF) kernel (Silverman 1998):*

$$K_c\left(\frac{x-x'}{s}\right) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{(x-x')^2}{2s^2}\right\}, \quad s > 0.$$

- *Epanechnikov kernel (Epanechnikov 1969):*

$$K_c(x-x';s) = \begin{cases} \frac{3}{4\sqrt{5}}\left(1-\frac{(x-x')^2}{5s^2}\right), & \text{if } \frac{(x-x')^2}{s^2} < 5 \\ 0, & \text{otherwise} \end{cases}, \quad s > 0.$$

Value

A list containing the following elements:

`Cluster` A cluster membership matrix.

`Entropy` A numeric value representing the entropy of the cluster assignments at the end of the iterative procedure.

`CondEntropy` A numeric value representing the conditional entropy of cluster assignment, given the observation weights $H(T \mid X)$.

`MutualInfo` A numeric value representing the mutual information, $I(Y;T)$, between the data distribution and the cluster assignments.

`InfoXT` A numeric value representing the mutual information, $I(X;T)$, between the original observations weights (X) and the cluster assignments (T).

`beta` A numeric value of the regularisation strength beta used.

`s` A numeric vector of bandwidth parameters used for the continuous variables.

`ixt` A numeric vector tracking the mutual information values between original observation weights and cluster assignments across iterations.

`iyt` A numeric vector tracking the mutual information values between original labels and cluster assignments across iterations.

`losses` A numeric vector tracking the final loss values across iterations.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Strouse DJ, Schwab DJ (2019). “The information bottleneck and geometric clustering.” *Neural Computation*, **31**(3), 596–612.

Silverman BW (1998). *Density Estimation for Statistics and Data Analysis (1st Ed.)*. Routledge.

Epanechnikov VA (1969). “Non-parametric estimation of a multivariate probability density.” *Theory of Probability & Its Applications*, **14**(1), 153–158.

See Also

`IBmix`, `IBcat`

Examples

```
# Generate simulated continuous data
set.seed(123)
X <- matrix(rnorm(200), ncol = 5)  # 200 observations, 5 features

# Run IBcont with automatic bandwidth selection and multiple initializations
result <- IBcont(X = X, ncl = 3, beta = 50, s = -1, nstart = 10)

# Print clustering results
print(result$Cluster)       # Cluster membership matrix
print(result$InfoXT)       # Mutual information between X and T
print(result$MutualInfo)    # Mutual information between Y and T
```

IBmix *Information Bottleneck Clustering for Mixed-Type Data*

Description

The `IBmix` function implements the Information Bottleneck (IB) algorithm for clustering datasets containing mixed-type variables, including categorical (nominal and ordinal) and continuous variables. This method optimizes an information-theoretic objective to preserve relevant information in the cluster assignments while achieving effective data compression (Strouse and Schwab 2019).

Usage

```
IBmix(X, ncl, beta, catcols, contcols, randinit = NULL,
       lambda = -1, s = -1, scale = TRUE,
       maxiter = 100, nstart = 100,
       contkernel = "gaussian",
       nomkernel = "aitchisonaitken", ordkernel = "liracine",
       cat_first = FALSE, verbose = FALSE)
```

Arguments

`X`	A data frame containing the input data to be clustered. It should include categorical variables (`factor` for nominal and `Ord.factor` for ordinal) and continuous variables (`numeric`).
`ncl`	An integer specifying the number of clusters.
`beta`	Regularisation strength.
`catcols`	A vector indicating the indices of the categorical variables in `X`.
`contcols`	A vector indicating the indices of the continuous variables in `X`.
`randinit`	An optional vector specifying the initial cluster assignments. If `NULL`, cluster assignments are initialized randomly.
`lambda`	A numeric value or vector specifying the bandwidth parameter for categorical variables. The default value is -1, which enables automatic determination of the optimal bandwidth. For nominal variables and `nomkernel = 'aitchisonaitken'`, the maximum allowable value of `lambda` is $(l-1)/l$, where l represents the number of categories, whereas for `nomkernel = 'liracine'` the maximum allowable value is 1. For ordinal variables, the maximum allowable value of `lambda` is 1, regardless of what `ordkernel` is being used.

`s` A numeric value or vector specifying the bandwidth parameter(s) for continuous variables. The values must be greater than 0. The default value is -1, which enables the automatic selection of optimal bandwidth(s).

`scale` A logical value indicating whether the continuous variables should be scaled to have unit variance before clustering. Defaults to `TRUE`.

`maxiter` The maximum number of iterations allowed for the clustering algorithm. Defaults to 100.

`nstart` The number of random initializations to run. The best clustering solution is returned. Defaults to 100.

`contkernel` Kernel used for continuous variables. Can be one of `gaussian` (default) or `epanechnikov`.

`nomkernel` Kernel used for nominal (unordered categorical) variables. Can be one of `aitchisonaitken` (default) or `liracine`.

`ordkernel` Kernel used for ordinal (ordered categorical) variables. Can be one of `liracine` (default) or `wangvanryzin`.

`cat_first` A logical value indicating whether bandwidth selection is prioritised for the categorical variables, instead of the continuous. Defaults to `FALSE`. Set to `TRUE` if you suspect that the continuous variables are not informative of the cluster structure. Can only be `TRUE` when all bandwidths are selected automatically (i.e. `s = -1`, `lambda = -1`).

`verbose` Logical. Defaults to `FALSE` to suppress progress messages. Change to `TRUE` to print.

Details

The `IBmix` function produces a fuzzy clustering of the data while retaining maximal information about the original variable distributions. The Information Bottleneck algorithm optimizes an information-theoretic objective that balances information preservation and compression. Bandwidth parameters for categorical (nominal, ordinal) and continuous variables are adaptively determined if not provided. This iterative process identifies stable and interpretable cluster assignments by maximizing mutual information while controlling complexity. The method is well-suited for datasets with mixed-type variables and integrates information from all variable types effectively.

The following kernel functions can be used to estimate densities for the clustering procedure. For continuous variables:

- *Gaussian (RBF) kernel (Silverman 1998):*

$$K_c\left(\frac{x-x'}{s}\right) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{(x-x')^2}{2s^2}\right\}, \quad s > 0.$$

- *Epanechnikov kernel (Epanechnikov 1969):*

$$K_c(x-x';s) = \begin{cases} \frac{3}{4\sqrt{5}}\left(1-\frac{(x-x')^2}{5s^2}\right), & \text{if } \frac{(x-x')^2}{s^2} < 5 \\ 0, & \text{otherwise} \end{cases}, \quad s > 0.$$

For nominal (unordered categorical variables):

- *Aitchison & Aitken kernel (Aitchison and Aitken 1976):*

$$K_u(x = x';\lambda) = \begin{cases} 1-\lambda, & \text{if } x = x' \\ \frac{\lambda}{\ell-1}, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq \frac{\ell-1}{\ell}.$$

- *Li & Racine kernel (Ouyang et al. 2006):*

$$K_u(x = x'; \lambda) = \begin{cases} 1, & \text{if } x = x' \\ \lambda, & \text{otherwise} \end{cases}, \quad 0 \leq \lambda \leq 1.$$

For ordinal (ordered categorical) variables:

- *Li & Racine kernel (Li and Racine 2003):*

$$K_o(x = x'; \nu) = \begin{cases} 1, & \text{if } x = x' \\ \nu^{|x-x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

- *Wang & van Ryzin kernel (Wang and Van Ryzin 1981):*

$$K_o(x = x'; \nu) = \begin{cases} 1 - \nu, & \text{if } x = x' \\ \frac{1-\nu}{2}\nu^{|x-x'|}, & \text{otherwise} \end{cases}, \quad 0 \leq \nu \leq 1.$$

The bandwidth parameters s, λ, and ν control the smoothness of the density estimate and are automatically determined by the algorithm if not provided by the user. ℓ is the number of levels of the categorical variable. For ordinal variables, the lambda parameter of the function is used to define ν.

Value

A list containing the following elements:

`Cluster` A cluster membership matrix.

`Entropy` A numeric value representing the entropy of the cluster assignments at the end of the iterative procedure.

`CondEntropy` A numeric value representing the conditional entropy of cluster assignment, given the observation weights $H(T \mid X)$.

`MutualInfo` A numeric value representing the mutual information, $I(Y;T)$, between the data distribution and the cluster assignments.

`InfoXT` A numeric value representing the mutual information, $I(X;T)$, between the original observations weights (X) and the cluster assignments (T).

`beta` A numeric value of the regularisation strength beta used.

`s` A numeric vector of bandwidth parameters used for the continuous variables.

`lambda` A numeric vector of bandwidth parameters used for the categorical variables.

`ixt` A numeric vector tracking the mutual information values between original observation weights and cluster assignments across iterations.

`iyt` A numeric vector tracking the mutual information values between original labels and cluster assignments across iterations.

`losses` A numeric vector tracking the final loss values across iterations.

Author(s)

Efthymios Costa, Ioanna Papatsouma, Angelos Markos

References

Strouse DJ, Schwab DJ (2019). "The information bottleneck and geometric clustering." *Neural Computation*, **31**(3), 596–612.

Aitchison J, Aitken CG (1976). "Multivariate binary discrimination by the kernel method." *Biometrika*, **63**(3), 413–420.

Li Q, Racine J (2003). "Nonparametric estimation of distributions with categorical and continuous data." *Journal of Multivariate Analysis*, **86**(2), 266–292.

Silverman BW (1998). *Density Estimation for Statistics and Data Analysis (1st Ed.)*. Routledge.

Ouyang D, Li Q, Racine J (2006). "Cross-validation and the estimation of probability distributions with categorical data." *Journal of Nonparametric Statistics*, **18**(1), 69–100.

Wang M, Van Ryzin J (1981). "A class of smooth estimators for discrete distributions." *Biometrika*, **68**(1), 301–309.

Epanechnikov VA (1969). "Non-parametric estimation of a multivariate probability density." *Theory of Probability & Its Applications*, **14**(1), 153–158.

See Also

DIBcont, DIBcat

Examples

```
# Example dataset with categorical, ordinal, and continuous variables
set.seed(123)
data <- data.frame(
  cat_var = factor(sample(letters[1:3], 100, replace = TRUE)),    # Nominal categorical variable
  ord_var = factor(sample(c("low", "medium", "high"), 100, replace = TRUE),
                   levels = c("low", "medium", "high"),
                   ordered = TRUE),                                # Ordinal variable
  cont_var1 = rnorm(100),                                         # Continuous variable 1
  cont_var2 = runif(100)                                          # Continuous variable 2
)

# Perform Mixed-Type Fuzzy Clustering
result <- IBmix(X = data, ncl = 3, beta = 2, catcols = 1:2, contcols = 3:4, nstart = 10)

# Print clustering results
print(result$Cluster)       # Cluster membership matrix
print(result$InfoXT)       # Mutual information between X and T
print(result$MutualInfo)    # Mutual information between Y and T
```

Index