Package 'nTARP'

May 9, 2026

Type Package

Title Cluster Analysis Using Thresholding After Random Projections (n-TARP)

Version 0.1.0

Description Implements the high-dimensional clustering technique Thresholding After Random Projections (n-TARP). Provides functionality to iteratively decompose larger datasets using contextual variables or within-cluster sum of squares. See Tarun & Boutin (2018) <doi:10.48550/arXiv.1806.05297> and Tarun & Boutin (2018) <doi:10.4 inal method and applications.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports stats

Suggests knitr, rmarkdown, HDclassif

VignetteBuilder knitr

NeedsCompilation no

Author David Reeping [aut, cre], Yunmeng Han [aut], Nahal Rashedi [rev]

Maintainer David Reeping <reepindp@ucmail.uc.edu>

Repository CRAN

Date/Publication 2026-03-20 12:40:02 UTC

Contents

`build_solution_from_labeled_clusters` *Combine nTARP BestCluster solutions into one table and assign stable final cluster IDs*

Description

Consolidates a set of nTARP "best cluster" solutions (branching runs) that contain ‘LabeledClusters‘ into a single ID-level data frame. For each run, the function adds a per-run cluster assignment column, constructs a concatenated ‘ClusterPath‘, and assigns a stable numeric ‘FinalClusterID‘ based on unique ‘ClusterPath‘ values.

Usage

```
build_solution_from_labeled_clusters(
  nTARP_best_clusters,
  ids = NULL,
  contextual_variables_df = NULL
)
```

Arguments

`nTARP_best_clusters`
: A list of run objects. Each run object must either:

  - contain ‘LabeledClusters‘ at the top level, or
  - contain a top-level list element that contains ‘LabeledClusters‘.

  ‘LabeledClusters‘ must be a named list with ‘"Cluster 1"‘ and ‘"Cluster 2"‘ entries containing IDs. The list should be named (recommended). Names are used as run identifiers (column suffixes). If unnamed, runs are labeled sequentially (‘run1‘, ‘run2‘, ...).

`ids`
: A vector of IDs (character or coercible) that defines the universe of rows in the output.

`contextual_variables_df`
: Optional ‘data.frame‘ of contextual variables to merge in. If supplied, it must contain a column named ‘id_name‘ (default ‘"mcid"‘). The merge is a left join on the provided ‘ids‘ vector, so every ID in ‘ids‘ appears in the output.

Value

A ‘data.frame‘ with one row per ID in ‘ids‘, optionally merged with contextual variables, plus per-run cluster columns, ‘ClusterPath‘, and ‘FinalClusterID‘.

Examples

```
data <- data.frame(X1 = c(0.5, -0.2, 0.1, 0.3, -0.1, 0.2, 5.2, 4.8, 5.1, 5.0,
                          -4.5, -5.2, -4.8, -5.1, -4.9, -5.3, 0.0, 0.2, 5.3, -5.0),
  X2 = c(0.3, -0.1, 0.2, 0.1, 0.0, 0.2, 5.0, 4.9, 5.3, 5.1,
         5.0, 5.2, 4.7, 4.9, 5.1, 4.8, -0.2, 0.0, 5.2, -4.9),
  X3 = c(0.4, 0.0, 0.1, -0.1, 0.2, 0.0, 5.1, 4.7, 5.2, 5.0,
         -5.0, -4.8, -5.3, -5.1, -4.9, -5.2, 0.1, 0.3, 5.0, -5.1)
)
nTARP_result <- nTARP_bisecting(data = data,number_of_projections = 100,withinss_threshold = 0.36)
result <- build_solution_from_labeled_clusters(nTARP_best_clusters = nTARP_result$BestClusters,
ids = 1:10, contextual_variables_df = data)
str(result)
```

---

consolidate_clusters *Merge clusters as post-processing*

Description

This function combines user-specified clusters after ‘nTARP‘ has run. The input to the function is the output of the ‘build_solution_from_labeled_clusters‘ function, along with the two cluster labels the user would like to merge. The output is returned in the same format as ‘build_solution_from_labeled_clusters‘, with the final solution labels renamed to reflect the merging.

Usage

```
consolidate_clusters(
  cluster_path_matrix,
  first_cluster_to_combine,
  second_cluster_to_combine
)
```

Arguments

`cluster_path_matrix`
: Data frame — output of ‘build_solution_from_labeled_clusters‘ showing which branch each observation belongs to from the ‘nTARP‘ clustering

`first_cluster_to_combine`
: Numeric — label of the first cluster to merge

`second_cluster_to_combine`
: Numeric — label of the second cluster to merge

Value

A data frame in the same format as the first argument, with the final column showing the cluster IDs relabeled based on the chosen merge.

Examples

```
data <- data.frame(X1 = c(0.5, -0.2, 0.1, 0.3, -0.1, 0.2, 5.2, 4.8, 5.1, 5.0,
                          -4.5, -5.2, -4.8, -5.1, -4.9, -5.3, 0.0, 0.2, 5.3, -5.0),
  X2 = c(0.3, -0.1, 0.2, 0.1, 0.0, 0.2, 5.0, 4.9, 5.3, 5.1,
         5.0, 5.2, 4.7, 4.9, 5.1, 4.8, -0.2, 0.0, 5.2, -4.9),
  X3 = c(0.4, 0.0, 0.1, -0.1, 0.2, 0.0, 5.1, 4.7, 5.2, 5.0,
         -5.0, -4.8, -5.3, -5.1, -4.9, -5.2, 0.1, 0.3, 5.0, -5.1)
)
nTARP_result <- nTARP_bisecting(data = data,number_of_projections = 100,
withinss_threshold = 0.36, minimum_cluster_size_percent = 30)
result <- build_solution_from_labeled_clusters(nTARP_best_clusters = nTARP_result$BestClusters,
ids = 1:20, contextual_variables_df = data)
str(result)
result <- consolidate_clusters(result,first_cluster_to_combine = 1,second_cluster_to_combine = 2)
str(result)
```

---

nTARP *Thresholding After Random Projections (n-TARP) Clustering*

Description

Implements the n-TARP clustering technique by projecting the data into a one-dimensional space and performing k-means. The data can be either unlabeled or labeled. The only required parameters are the number of projections and the within-cluster sum of squares threshold. Suggested starting values: ‘number_of_projections = 1000‘ and ‘withinss_threshold = 0.36‘.

Usage

```
nTARP(data, number_of_projections, withinss_threshold, ids = NULL)
```

Arguments

`data`	Numeric matrix — dataset to be clustered using ‘nTARP‘
`number_of_projections`	Numeric — number of random projections for ‘nTARP‘ to try for each run
`withinss_threshold`	Numeric — maximum value defining what a "quality cluster" is, based on the solution’s normalized within-cluster sum of squares (typically 0.36)
`ids`	Numeric or character vector — identifying labels for individuals in the clusters

Value

A list containing results and supporting data from the k-means clustering analysis: (1) ‘Optimal-Solution‘: the optimal clustering solution, including cluster assignments and centroids, (2) ‘OptimalProjection‘: the projection vector associated with the optimal solution, (3) ‘Threshold‘: the threshold used for determining cluster membership or filtering, (4) ‘Direction‘: indicates where a

new data point should be placed if using the result as a classifier, (5) 'OptimalWithinss': the within-cluster sum of squares for the optimal solution, (6) 'AllWithinss': the within-cluster sum of squares for all candidate solutions, (7) 'Clusterings': all clustering solutions generated during analysis, (8) 'OriginalData': the original dataset used for clustering, (9) 'OriginalIDs': the identifiers of the original observations.

References

Tarun, Y.; Boutin, M. (2018). n-TARP Binary Clustering Code. Purdue University Research Repository. doi:10.4231/R74B2ZJV

Examples

```
data <- data.frame(X1 = c(0.5, -0.2, 0.1, 5.2, 4.8, 5.1, -4.5, -5.2, -4.8, -5.1),
X2 = c(0.3, -0.1, 0.2, 5.0, 4.9, 5.3, 5.0, 5.2, 4.7, 4.9),
X3 = c(0.4, 0.0, 0.1, 5.1, 4.7, 5.2, -5.0, -4.8, -5.3, -5.1))
result <- nTARP(data = data,number_of_projections = 100,withinss_threshold = 0.36)
str(result)
```

---

nTARP_bisecting *Run nTARP repeatedly in a bisecting fashion*

Description

Repeatedly applies 'nTARP' to iteratively bisect a dataset until a minimum cluster size threshold is reached.

Usage

```
nTARP_bisecting(
  data,
  number_of_projections,
  withinss_threshold,
  ids = NULL,
  minimum_cluster_size_percent = 20,
  contextual_variable = NULL
)
```

Arguments

`data`	Numeric matrix — dataset to be clustered using 'nTARP'
`number_of_projections`	Numeric — number of random projections for 'nTARP' to try for each run
`withinss_threshold`	Numeric — maximum value defining what a "quality cluster" is, based on the solution's normalized within-cluster sum of squares (typically 0.36)
`ids`	Numeric or character vector — identifying labels for individuals in the clusters

`minimum_cluster_size_percent`
: Numeric — minimum size allowable for a cluster (expressed as a percentage)

`contextual_variable`
: Vector of integers or characters — variable to use as the basis for comparing clusters. This is 'NULL' by default, which analytically corresponds to option (1).

Details

This function supports two strategies for selecting the optimal split at each step:

(1) Within-Cluster Compactness Criterion: The optimal solution is selected based on the normalized within-cluster sum of squares (WSS). The split that minimizes normalized WSS is retained.

(2) Contextual Purity Criterion: The optimal solution is selected using a contextual variable. Inspired by decision tree learning, the algorithm evaluates candidate splits based on improvements in class purity (i.e., Gini reduction) with respect to the contextual variable. The split that maximizes purity gain is retained.

The process continues recursively (bisecting the largest eligible cluster) until no resulting cluster meets the user-defined minimum size threshold.

Value

A list containing: (1) Complete solutions (i.e., outputs from the 'nTARP' function), (2) Clusters with the best gains identified using the 'pull_best_solution_and_gain' function, (3) Within-cluster sum of squares for each solution, (4) Gains for each solution (if a contextual variable is used).

Examples

```
# 20-point example dataset
data <- data.frame(
  X1 = c(0.5, -0.2, 0.1, 0.3, -0.1, 0.2, 5.2, 4.8, 5.1, 5.0,
         -4.5, -5.2, -4.8, -5.1, -4.9, -5.3, 0.0, 0.2, 5.3, -5.0),
  X2 = c(0.3, -0.1, 0.2, 0.1, 0.0, 0.2, 5.0, 4.9, 5.3, 5.1,
         5.0, 5.2, 4.7, 4.9, 5.1, 4.8, -0.2, 0.0, 5.2, -4.9),
  X3 = c(0.4, 0.0, 0.1, -0.1, 0.2, 0.0, 5.1, 4.7, 5.2, 5.0,
         -5.0, -4.8, -5.3, -5.1, -4.9, -5.2, 0.1, 0.3, 5.0, -5.1)
)

# Run nTARP without contextual variable
result1 <- nTARP_bisecting(
  data = data,
  number_of_projections = 10,
  withinss_threshold = 0.36
)
str(result1)

# Add a latent group as contextual variable
latent_group <- c(1, 1, 1, 1, 1, 1, 2, 2, 2, 2,
                  3, 3, 3, 3, 3, 3, 1, 1, 2, 3)

# Run nTARP with contextual variable
```

```
result2 <- nTARP_bisecting(
  data = data,
  number_of_projections = 10,
  withinss_threshold = 0.36,
  contextual_variable = latent_group
)
str(result2)
```

`nTARP_complete_solution_no_contextual_variable` *Run nTARP repeatedly in a bisecting fashion (using normalized within sum of squares)*

Description

#' @keywords internal

Usage

```
nTARP_complete_solution_no_contextual_variable(
  data,
  number_of_projections,
  withinss_threshold,
  ids,
  minimum_cluster_size_percent
)
```

Arguments

`data`	Numeric matrix — dataset to be clustered using ‘nTARP‘
`number_of_projections`	Numeric — number of random projections for ‘nTARP‘ to try for each run (usually 1000 to start)
`withinss_threshold`	Numeric — maximum value defining what a "quality cluster" is, based on the solution’s normalized within-cluster sum of squares (typically 0.36)
`ids`	Numeric or character vector — identifying labels for individuals in the clusters
`minimum_cluster_size_percent`	Numeric — minimum size allowable for a cluster to be further bisected (as a percentage)

Details

Repeatedly applies ‘nTARP‘ to iteratively bisect a dataset until a minimum cluster size threshold is reached, using within-cluster compactness to select optimal splits.

At each step, the algorithm evaluates candidate splits based on the normalized within-cluster sum of squares (WSS). The split that minimizes normalized WSS is retained.

The process continues recursively, bisecting the largest eligible cluster, until no resulting cluster meets the user-defined minimum size threshold.

Value

A list containing: (1) Complete solutions (i.e., outputs from the ‘nTARP‘ function), (2) Clusters with the best gains identified using the ‘pull_best_solution_and_gain‘ function, (3) Within-cluster sum of squares for each solution

---

`nTARP_complete_solution_with_contextual_variable` *Run nTARP repeatedly in a bisecting fashion (using contextual variable)*

Description

#’ @keywords internal

Usage

```
nTARP_complete_solution_with_contextual_variable(
  data,
  number_of_projections,
  withinss_threshold,
  ids,
  contextual_variable,
  minimum_cluster_size_percent
)
```

Arguments

`data`	Numeric matrix — dataset to be clustered using ‘nTARP‘
`number_of_projections`	Numeric — number of random projections for ‘nTARP‘ to try for each run (usually 1000 to start)
`withinss_threshold`	Numeric — maximum value defining what a "quality cluster" is, based on the solution’s normalized within-cluster sum of squares (typically 0.36)
`ids`	Numeric or character vector — identifying labels for individuals in the clusters

`contextual_variable`
: Vector of integers or characters — variable to use as the basis for comparing clusters

`minimum_cluster_size_percent`
: Numeric — minimum size allowable for a cluster to be further bisected (as a percentage)

Details

Repeatedly applies ‘nTARP‘ to iteratively bisect a dataset until a minimum cluster size threshold is reached, using a contextual variable to select optimal splits.

At each step, the algorithm evaluates candidate splits based on improvements in class purity of the contextual variable (e.g., Gini reduction). The split that maximizes purity gain is retained.

The process continues recursively, bisecting the largest eligible cluster, until no resulting cluster meets the user-defined minimum size threshold.

Value

A list containing: (1) Complete solutions (i.e., outputs from the ‘nTARP‘ function), (2) Clusters with the best gains identified using the ‘pull_best_solution_and_gain‘ function, (3) Within-cluster sum of squares for each solution, (4) Gains for each solution.

Index