

Package ‘discoCVI’

May 5, 2026

Type Package

Title Implementation of the DISCO Metric for Internal Clustering Evaluation

Version 0.1.1

Description Implementation of the DISCO (Density-based Internal Score for Clusterings with nOise) metric, a cluster validity index for evaluating density-based clustering results without ground truth labels. DISCO is the first index to explicitly assess the quality of noise point assignments in addition to cluster quality. It uses density-connectivity distance derived from a minimum spanning tree of the mutual-reachability graph, providing interpretable, bounded scores in $[-1, 1]$. Higher scores indicate better clustering. Based on Beer et al. (2025) [doi:10.48550/arXiv.2503.00127](https://doi.org/10.48550/arXiv.2503.00127).

License MIT + file LICENSE

Encoding UTF-8

Language en-GB

Depends R ($\geq 3.6.0$)

Imports FNN, stats

Suggests dbscan, testthat ($\geq 3.0.0$)

RoxygenNote 7.3.3

URL <https://github.com/aminentezari/discoCVI>

BugReports <https://github.com/aminentezari/discoCVI/issues>

NeedsCompilation no

Author Amin Entezari [aut, cre] (ORCID: <https://orcid.org/0009-0005-4733-6032>),
Davide Chicco [ctb] (ORCID: <https://orcid.org/0000-0001-9655-7142>),
Supervision and guidance),
Anna Beer [aut] (Original Python implementation (arXiv:2503.00127)),
Lena Krieger [aut] (Original Python implementation (arXiv:2503.00127)),
Pascal Weber [aut] (Original Python implementation (arXiv:2503.00127))

Maintainer Amin Entezari <amin_entezari@outlook.com>

Repository CRAN

Date/Publication 2026-05-05 18:20:13 UTC

Contents

compute_dc_distances	2
disco_samples	3
disco_score	4
p_cluster	4
p_noise	5

Index 7

compute_dc_distances *Compute density-connectivity (DC) distances*

Description

The main entry point for constructing the DC-distance matrix used throughout the DISCO metric. The procedure is:

1. Compute mutual-reachability distances ([calculate_reachability_distance](#)).
2. Build the minimum spanning tree of those distances ([get_mst_edges](#)).
3. Extract pairwise DC distances as minimax-path weights in the MST ([extract_dc_distances_from_mst](#)).

Usage

```
compute_dc_distances(X, min_points = 5)
```

Arguments

`X` A numeric matrix of shape $n \times p$, or a `data.frame` that will be coerced to a matrix.

`min_points` A positive integer for the core-distance neighbourhood size. Default is 5.

Value

A symmetric $n \times n$ numeric matrix of DC distances.

Examples

```
set.seed(1)
X <- matrix(rnorm(40), ncol = 2)
D <- compute_dc_distances(X, min_points = 5)
dim(D) # 20 x 20
```

disco_samples	<i>Per-sample DISCO scores</i>
---------------	--------------------------------

Description

Computes a pointwise DISCO score for every observation in X . The score lies in $[-1, 1]$:

- Values close to 1 indicate a well-placed point (either tightly within its cluster or a noise point far from all clusters).
- Values close to 0 indicate borderline placement.
- Values close to -1 indicate a misplaced point (or a noise point that should belong to a cluster).

Four cases are handled:

1. **All noise** — every point labelled -1 : all scores are -1 .
2. **Single cluster with no noise** — all scores are 0 .
3. **One real cluster + noise** — cluster points are scored via [p_cluster](#); noise points are scored via [p_noise](#).
4. **Two or more clusters (optional noise)** — non-noise points use [p_cluster](#) on the non-noise sub-matrix; noise points use [p_noise](#).

Usage

```
disco_samples(X, labels, min_points = 5)
```

Arguments

<code>X</code>	A numeric matrix ($n \times p$) or data frame.
<code>labels</code>	An integer vector of length n containing cluster labels. Use -1 to denote noise/outlier points.
<code>min_points</code>	A positive integer for the core-distance neighbourhood size. Default is 5.

Value

A numeric vector of length n with per-point DISCO scores.

See Also

[disco_score](#), [p_cluster](#), [p_noise](#)

Examples

```
set.seed(42)
X <- matrix(rnorm(100), ncol = 2)
labels <- rep(c(0L, 1L), each = 25)
s <- disco_samples(X, labels)
hist(s, main = "Per-sample DISCO scores")
```

disco_score	<i>Overall DISCO score</i>
-------------	----------------------------

Description

Returns the mean of the per-sample DISCO scores produced by `disco_samples`. This is the single-number summary of clustering quality used for model selection and comparison.

Usage

```
disco_score(X, labels, min_points = 5)
```

Arguments

X	A numeric matrix ($n \times p$) or data.frame.
labels	An integer vector of cluster labels (-1 for noise).
min_points	Positive integer; core-distance neighbourhood size. Default is 5.

Value

A single numeric value in $[-1, 1]$.

See Also

[disco_samples](#)

Examples

```
set.seed(42)
X <- matrix(rnorm(100), ncol = 2)
labels <- rep(c(0L, 1L), each = 25)
disco_score(X, labels)
```

p_cluster	<i>Silhouette-like cluster scores using DC distances</i>
-----------	--

Description

Computes a silhouette coefficient for each non-noise point using the precomputed (or internally derived) DC-distance matrix rather than raw Euclidean distances. This matches the formula used by `sklearn.metrics.silhouette_samples` with `metric="precomputed"` (Python reference: `disco.py`, line 280):

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

where $a(i)$ is the mean DC distance from i to all other points in the same cluster and $b(i)$ is the minimum over other clusters of the mean DC distance from i to that cluster.

Usage

```
p_cluster(X, labels, min_points = 5, precomputed_dc_dists = FALSE)
```

Arguments

X Either (a) a numeric matrix of raw data ($n \times p$) when `precomputed_dc_dists = FALSE`, or (b) a precomputed $n \times n$ DC-distance matrix when `precomputed_dc_dists = TRUE`.

labels An integer vector of cluster labels of length n . Noise labels (-1) should not be present; pass only the non-noise subset.

min_points Positive integer; used only when `precomputed_dc_dists = FALSE`. Default is 5.

precomputed_dc_dists Logical. If TRUE, X is treated as an already-computed DC-distance matrix; no further distances are computed. Default is FALSE.

Value

A numeric vector of length n with per-point silhouette-style scores in $[-1, 1]$.

See Also

[disco_samples](#), [compute_dc_distances](#)

Examples

```
set.seed(7)
X <- matrix(rnorm(60), ncol = 2)
labels <- rep(c(0L, 1L, 2L), each = 10)
p_cluster(X, labels)
```

p_noise

Noise-point scores

Description

Assigns a quality score to each point labelled as noise (-1). Two complementary sub-scores are computed and the element-wise minimum is taken as the final score:

p_{sparse} Measures whether the noise point is sparser (more peripheral in density) than the densest part of every real cluster. For cluster c , let M_c be the maximum core distance over all points in c . Then

$$p_{\text{sparse}}(i) = \min_c \frac{\text{core}_k(i) - M_c}{\max(\text{core}_k(i), M_c)}.$$

p_{far} Measures whether the noise point is far from every cluster in DC-distance space. For cluster c , let $d_{\min,c}(i)$ be the minimum DC distance from i to any point in c . Then

$$p_{\text{far}}(i) = \min_c \frac{d_{\min,c}(i) - M_c}{\max(d_{\min,c}(i), M_c)}.$$

Both sub-scores lie in $[-1, 1]$: positive means the noise point is legitimately sparse/far; negative means it is denser/closer than the cluster boundary and might be a misclassified inlier.

Usage

```
p_noise(X, labels, min_points = 5, dc_dists = NULL)
```

Arguments

<code>X</code>	A numeric matrix ($n \times p$) or data.frame of all points (including non-noise).
<code>labels</code>	An integer vector of length n ; -1 denotes noise.
<code>min_points</code>	Positive integer ≥ 2 ; core-distance neighbourhood size. Default is 5.
<code>dc_dists</code>	Optional precomputed $n \times n$ DC-distance matrix. If NULL (default), it is computed internally.

Value

A named list with two numeric vectors, each of length equal to the number of noise points:

p_sparse Sparsity-based noise scores.

p_far DC-distance-based noise scores.

See Also

[disco_samples](#)

Examples

```
set.seed(3)
X      <- matrix(rnorm(60), ncol = 2) # 30 rows x 2 cols
labels <- c(rep(0L, 10), rep(1L, 10), rep(-1L, 10)) # 30 labels
nr     <- p_noise(X, labels, min_points = 3)
str(nr)
```

Index

`calculate_reachability_distance`, 2

`compute_dc_distances`, 2, 5

`disco_samples`, 3, 4–6

`disco_score`, 3, 4

`extract_dc_distances_from_mst`, 2

`get_mst_edges`, 2

`p_cluster`, 3, 4

`p_noise`, 3, 5