

Package: manydist (via r-universe)

October 19, 2026

Type Package

Title Distance-Based Learning for Mixed-Type Data

Version 0.5.2

Description Provides tools for constructing, computing, and using distance measures for numerical, categorical, and mixed-type data. The package implements a flexible framework in which continuous and categorical components can be combined under additive, commensurable, and association-aware specifications. Supported methods include classical distances such as Gower, Euclidean, Manhattan, and Mahalanobis-type distances; categorical dissimilarities such as simple matching, occurrence-frequency, and association-based measures; and mixed-type presets designed to reduce biases due to variable type, scale, distribution, redundancy, and number of categories. The package also provides scaling options, supervised and unsupervised distance constructions, leave-one-variable-out tools for distance-based variable importance, and integration with distance-based learning workflows such as nearest-neighbour prediction, partitioning around medoids, and spectral clustering. Methods are motivated by van de Velden, Iodice D'Enza, Markos, and Cavicchia (2026) <[doi:10.1080/10618600.2026.2680181](https://doi.org/10.1080/10618600.2026.2680181)> and related work on categorical and mixed-type dissimilarities.

URL https://alfonsoiodicede.github.io/manydist_package/,
https://github.com/alfonsoIodiceDE/manydist_package/tree/main/manydist

BugReports https://github.com/alfonsoIodiceDE/manydist_package/issues

Imports aricode, cluster, clusterGeneration, data.table, dials,
distances, dplyr, entropy, fastDummies, forcats, fpc, generics,
ggplot2, kdml, magrittr, Matrix, parsnip, philentropy, purrr,
readr, recipes, rlang, rsample, stats, tibble, tidyr,
tidyselect, tune

Depends R (>= 4.5.0)

Suggests arules, clustMixType, FD, klaR, mclust, palmerpenguins,
parallelDist, StatMatch, testthat (>= 3.0.0), workflows,
workflowsets

License GPL-3

Encoding UTF-8

LazyData true

NeedsCompilation no

Config/roxygen2/version 8.0.0

Config/testthat/edition 3

Config/pak/sysreqs cmake make libicu-dev libuv1-dev libx11-dev

Repository <https://alfonsoiodicede.r-universe.dev>

Date/Publication 2026-10-19 11:32:35 UTC

RemoteUrl https://github.com/alfonsoiodicede/manydist_package

RemoteRef HEAD

RemoteSha 53711e475d47628de5f2423f458b2cd21b505825

RemoteSubdir manydist

Contents

all_dist_method_specs	3
autoplot.MDistBenchmark	4
benchmark_mdists	5
compare_lovo_mdists	7
congruence_coeff	9
dist_methods_tbl	10
dist_methods_tbl_cat	11
fifa_nl	11
gen_mixed	13
generate_dataset	14
knn_dist	15
lovo_mdists	16
lovo_method_spec	19
make_mdists_recipe	20
mdists	21
mdists_summary_impl	24
pam_dist	24
response_aware_methods	25
spectral_dist	26
spectral_from_dist	27
step_mdists	27
summary.MDistBenchmark	30
wdi_2022	31

Index

all_dist_method_specs *Create a grid of 'mdist()' method specifications*

Description

Construct a tibble of valid method specifications that can be passed to [benchmark_mdists()] or iterated over manually to benchmark [mdist()] across multiple configurations.

Usage

```
all_dist_method_specs(
  method_cat = NULL,
  method_num = NULL,
  preset = NULL,
  commensurable = NULL
)
```

Arguments

method_cat	Optional character vector restricting the categorical methods used in custom specifications.
method_num	Optional character vector restricting the numerical preprocessing methods used in custom specifications.
preset	'NULL' (the default) returns custom combinations only. A character vector selects named presets; "all" selects all named presets. Include "custom" explicitly to also return custom combinations, for example 'c("all", "custom)". Component filters apply only to custom rows.
commensurable	Optional logical vector restricting the commensurable values used in custom specifications.

Details

Select named presets or construct custom specifications from the categorical methods and numerical preprocessing options listed in [dist_methods_tbl()].

Rows with 'preset = "custom"' use the explicit categorical, numerical, and commensurability options. Named presets determine those settings internally.

With 'commensurable = TRUE', custom grids retain only "std" and "pc_scores" numerical preprocessing. Variable-wise affine scaling cancels under commensurability, so "robust" is omitted as redundant with "std". The existing exclusions of "none" and "range" are retained. Range scaling clips new observations outside the training range, so its equivalence on training data does not extend to all new observations. These restrictions affect grid construction, not the options accepted by [mdist()].

Named preset rows retain 'NA' in their component columns: their settings are determined by the preset. Supplying component filters without requesting custom rows gives an error rather than silently ignoring those filters.

Value

A tibble where each row represents one valid `'mdist()'` specification. The tibble contains `'preset'`, `'method_cat'`, `'method_num'`, and `'commensurable'`.

Examples

```
all_dist_method_specs()
all_dist_method_specs(preset = c("gower", "u_indep", "u_dep"))
all_dist_method_specs(preset = "all")
all_dist_method_specs(preset = c("all", "custom"))
all_dist_method_specs(method_cat = c("tvd", "le_and_ho"))
```

```
autoplot.MDistBenchmark
```

Plot pairwise distance benchmark comparisons

Description

Draws an annotated triangular heatmap of the pairwise diagnostics from `[benchmark_mdists()]`.

Usage

```
## S3 method for class 'MDistBenchmark'
autoplot(
  object,
  metric = c("relative_distance", "mad", "alienation", "mds_congruence", "ari"),
  cluster_method = NULL,
  digits = 2,
  ...
)
```

Arguments

<code>object</code>	An object returned by <code>[benchmark_mdists()]</code> .
<code>metric</code>	Character string selecting <code>"mad"</code> , <code>"relative_distance"</code> , <code>"mds_congruence"</code> , <code>"alienation"</code> , or <code>"ari"</code> . A specific ARI column, such as <code>"ari_pam"</code> , can also be supplied.
<code>cluster_method</code>	Optional clustering method used when <code>'metric = "ari"'</code> . If <code>'NULL'</code> , all available ARI metrics are shown, with one facet per clustering method.
<code>digits</code>	Number of decimal places used for cell labels.
<code>...</code>	Currently unused.

Details

The default fill gradient runs from coral (`#E76F51`) for lower values to blue (`#008CFF`) for higher values, with regular-weight white cell labels. To use another palette, add a fill scale to the returned plot. Self-comparisons on the main diagonal are left blank. ARI plots use fixed colour limits of 0 and 1, so the same ARI has the same colour across plots. Negative ARIs retain their numerical labels but use the low-end colour.

Value

A 'ggplot' object.

benchmark_mdists	<i>Benchmark and compare multiple 'mdists' specifications</i>
------------------	---

Description

Applies `[mdist()]` repeatedly over a tibble of distance-method specifications, typically generated with `[all_dist_method_specs()]`. It then compares every pair of successful distance specifications using distance, configuration, and optional clustering diagnostics.

Usage

```
benchmark_mdists(
  x,
  response = NULL,
  specs = all_dist_method_specs(),
  dims = 2,
  cluster_k = NULL,
  cluster_methods = c("pam", "hclust", "spectral"),
  hclust_method = "average",
  spectral_sigma = NULL,
  spectral_nstart = 50
)
```

Arguments

<code>x</code>	A data frame or tibble of predictors, optionally including the response column.
<code>response</code>	Optional response column inside 'x', supplied either unquoted or as a character string.
<code>specs</code>	A tibble of method specifications. By default, this is generated with <code>[all_dist_method_specs()]</code> . It must contain the columns 'preset', 'method_cat', 'method_num', and 'com-mensurable'. An optional 'label' column supplies display labels for comparisons and plots.
<code>dims</code>	Integer. Number of dimensions used by classical multidimensional scaling when computing congruence and alienation coefficients.

cluster_k	Optional integer. Number of clusters used for pairwise adjusted Rand indices. If 'NULL', no clustering is performed.
cluster_methods	Character vector specifying the clustering methods used when 'cluster_k' is supplied. Possible values are "pam", "hclust", and "spectral".
hclust_method	Character string specifying the linkage method passed to [stats::hclust()] when "hclust" is requested.
spectral_sigma	Optional numeric value for the Gaussian affinity bandwidth used by spectral clustering. If 'NULL', the default used by [spectral_dist()] is applied.
spectral_nstart	Integer. Number of random starts used by the k-means step in spectral clustering.

Details

Each row of 'specs' is interpreted as one valid 'mdist()' configuration. Preset-based and custom specifications are both supported. Failed specifications are caught and returned in the output rather than stopping the full benchmark.

Preset specifications use the 'preset' column and ignore 'method_cat', 'method_num', and 'commensurable'. Component specifications are evaluated as 'preset = "custom"' and use 'method_cat', 'method_num', and 'commensurable'.

Pairwise mean absolute difference ('mad') is computed from the lower triangle of each dissimilarity matrix. The symmetric relative distance is defined as

$$\frac{2 \text{mean}(|d_a - d_b|)}{\text{mean}(|d_a|) + \text{mean}(|d_b|)}.$$

Classical multidimensional scaling configurations are compared using [congruence_coeff()]. The corresponding alienation coefficient is $\sqrt{1 - c^2}$.

When 'cluster_k' is supplied, each requested clustering method is applied once to every successful distance specification. Partitions produced by the same clustering method are then compared pairwise using the adjusted Rand index. When 'cluster_k = NULL', no clustering is performed and no ARI columns are included in the comparisons.

Value

An object of class "MDistBenchmark", which is also a tibble. It contains the supplied specifications together with:

result The corresponding output of [mdist()], or an error object if the specification failed.

ok Logical indicator; 'TRUE' if the run completed successfully, 'FALSE' otherwise.

error Error message for failed runs, 'NA' otherwise.

Use 'summary()' to display upper-triangular tables of pairwise diagnostics and invisibly obtain their complete tibble, and [ggplot2::autoplot()] to draw an annotated triangular heatmap. The default print method displays a compact run overview; use 'tibble::as_tibble()' to inspect the complete underlying run table.

Examples

```
if (requireNamespace("palmerpenguins", quietly = TRUE)) {
  data("penguins", package = "palmerpenguins")

  penguins_small <- palmerpenguins::penguins |>
    dplyr::select(
      species, bill_length_mm, bill_depth_mm, flipper_length_mm,
      body_mass_g, island, sex
    ) |>
    tidyr::drop_na()

  specs <- all_dist_method_specs(
    preset = c("gower", "u_indep", "u_dep")
  )

  res <- benchmark_mdists(
    penguins_small,
    response = species,
    specs = specs
  )

  res
  pairwise_results <- summary(res)
  ggplot2::autoplot(res, metric = "relative_distance")
}
```

compare_lovo_mdists	<i>Compare LOVO diagnostics across multiple distance specifications</i>
---------------------	---

Description

This function compares **leave-one-variable-out (LOVO)** diagnostics across multiple distance definitions supported by `manydist`.

Usage

```
compare_lovo_mdists(
  x,
  methods,
  mds = FALSE,
  dims = 2,
  keep_dist = FALSE,
  .progress = FALSE,
  ...
)
```

Arguments

<code>x</code>	A data frame or tibble containing the predictors.
<code>methods</code>	A **named list** describing the distance specifications to compare. Each element must be a list of arguments passed to <code>lovo_mdist</code>. For example: <pre> methods = list(gower = list(preset = "gower"), u_dep = list(preset = "unbiased_dependent"), custom = list(preset = "custom", method_cat = "matching", method_num = "std", commensurable = TRUE)) </pre>
<code>mds</code>	Logical. If 'TRUE', compute MDS-based congruence and alienation diagnostics for every distance specification. The default is 'FALSE'.
<code>dims</code>	Number of dimensions used for the MDS configuration when 'mds = TRUE'. Ignored when 'mds = FALSE'.
<code>keep_dist</code>	Logical; if TRUE, distance matrices from the LOVO computations are retained. This increases memory usage.
<code>.progress</code>	Logical; if TRUE, progress messages are printed while computing LOVO diagnostics for each method.
<code>...</code>	Additional arguments passed to <code>lovo_mdist</code> unless overridden in the method-specific argument list. These may include optional clustering diagnostics, for example <code>cluster_k</code>, <code>cluster_methods</code>, and <code>hclust_method</code>.

Details

For each distance specification, the function:

1. Computes the full mixed-type distance using `mdist()`.
2. Recomputes the distance repeatedly leaving out one variable at a time.
3. Measures the impact of each variable using mean absolute deviation (MAD) and, when requested, MDS- or clustering-based diagnostics.

The results are combined across methods and returned as an `MDistLOVOCompare` object, which supports `print()`, `summary()`, and `ggplot2::autoplot()`.

In `autoplot()`, `reorder = TRUE` ranks variables by the selected metric averaged across methods, separately within each variable type. Categorical variables precede numerical variables, preserving the blue/red background bands. `top_n` still selects variables globally before ordering. If type information is absent, ordering is global.

Value

An object of class `MDistLOVCompare` containing:

results A tibble with one row per method-variable combination.

methods The list of distance specifications used.

mds Whether MDS diagnostics were computed.

dims Number of MDS dimensions used, or 'NULL' when 'mds = FALSE'.

n_obs Number of observations in the dataset.

See Also

[lovo_mdists](#), [mdists](#)

Examples

```
## Not run:
library(manydist)
library(palmerpenguins)

data <- penguins |>
  dplyr::select(-species) |>
  tidyr::drop_na()

cmp <- compare_lovo_mdists(
  x = data,
  methods = list(
    gower = list(preset = "gower"),
    u_dep = list(preset = "unbiased_dependent")
  ),
  cluster_k = 3
)

summary(cmp)
autoplot(cmp, metric = "mad_importance")
autoplot(cmp, metric = "pam_importance")

## End(Not run)
```

congruence_coeff

Congruence coefficient between two configurations

Description

Computes the congruence coefficient between two data configurations using the Frobenius inner product of their pairwise distance matrices.

Usage

```
congruence_coeff(L1, L2)
```

Arguments

L1 A numeric matrix or data frame (rows = observations)
L2 A numeric matrix or data frame with the same number of rows as L1

Details

The congruence coefficient is defined as

$$\frac{\langle D_1, D_2 \rangle_F}{\sqrt{\langle D_1, D_1 \rangle_F \langle D_2, D_2 \rangle_F}}$$

where D_1 and D_2 are the pairwise distance matrices derived from L1 and L2.

Value

A scalar in $[-1, 1]$ measuring similarity between the two configurations.

dist_methods_tbl	List available 'manydist' methods
------------------	-----------------------------------

Description

Returns a tibble describing the categorical methods, numerical preprocessing options, and preset specifications currently available in 'manydist'.

Usage

```
dist_methods_tbl()
```

Details

The table is used by helpers such as [all_dist_method_specs()] and [benchmark_mdists()] to construct valid 'mdists()' specifications.

Value

A tibble with one row per available method. The columns are:

method Name of the method or preset.

argument The 'mdists()' argument to which the method belongs: "method_cat", "method_num", or "preset".

data_type Type of variables targeted by the method.

distance_basis Broad methodological family.

response_aware Logical; whether the method can use a response variable.

engine Implementation source.

Examples

```
dist_methods_tbl()

dist_methods_tbl() |>
  dplyr::count(argument)
```

dist_methods_tbl_cat	<i>List available categorical dissimilarities</i>
----------------------	---

Description

Convenience wrapper around [dist_methods_tbl()] returning only methods that can be used as categorical dissimilarities through ‘method_cat’.

Usage

```
dist_methods_tbl_cat()
```

Value

A tibble containing the rows of [dist_methods_tbl()] with ‘argument == "method_cat"’.

Examples

```
dist_methods_tbl_cat()
```

fifa_nl	<i>FIFA 21 Player Data - Dutch League</i>
---------	---

Description

The fifa_nl dataset contains information on players in the Dutch League from the FIFA 21 video game. This dataset includes various attributes of players, such as demographics, club details, skill ratings, and physical characteristics.

Usage

```
data("fifa_nl")
```

Format

A data frame with observations on various attributes describing the players.

`player_positions` Primary playing positions of the player.

`nationality` The country the player represents.

`team_position` Player's assigned position within their club.

`club_name` Name of the club the player is part of.

`work_rate` The player's work rate, describing defensive and attacking intensity.

`weak_foot` Skill rating for the player's non-dominant foot, ranging from 1 to 5.

`skill_moves` Skill moves rating, indicating technical skill and ability to perform complex moves, on a scale of 1 to 5.

`international_reputation` Player's reputation on an international scale, from 1=local to 3=global star.

`body_type` Body type of the player (Lean, Normal, Stocky.

`preferred_foot` Dominant foot of the player, either Left or Right.

`age` Age of the player in years.

`height_cm` Height of the player in centimeters.

`weight_kg` Weight of the player in kilograms.

`overall` Overall skill rating of the player out of 100.

`potential` Potential skill rating the player may achieve in the future.

`value_eur` Estimated market value of the player in Euros.

`wage_eur` Player's weekly wage in Euros.

`release_clause_eur` Release clause value in Euros, which other clubs must pay to buy out the player's contract.

`pace` Speed rating of the player out of 100.

`shooting` Shooting skill rating out of 100.

`passing` Passing skill rating out of 100.

`dribbling` Dribbling skill rating out of 100.

`defending` Defending skill rating out of 100.

`physic` Physicality rating out of 100.

Details

This dataset provides a snapshot of player attributes and performance indicators as represented in FIFA 21 for players in the Dutch League. It can be used to analyze player characteristics, compare skills across players, and explore potential relationships among variables such as age, position, and various skill ratings.

References

Stefano Leone. (2021). *FIFA 21 Complete Player Dataset*. Retrieved from <https://www.kaggle.com/datasets/stefanoleone992/fifa-21-complete-player-dataset>.

Examples

```
data(fifa_n1)
summary(fifa_n1)
```

gen_mixed

*Generate mixed-type clustering data with signal and noise***Description**

Generates synthetic mixed datasets with controllable numerical and categorical signal/noise structure and balanced cluster sizes.

Usage

```
gen_mixed(
  k_true,
  clustSizeEq = 50,
  numsignal = 2,
  numnoise = 2,
  catsignal = 2,
  catnoise = 2,
  q = 5,
  q_err = 9,
  numsep = 0.1,
  catsep = 0.5,
  seed = NULL,
  error_type = c("normal", "chisq"),
  error_df = 2,
  error_scale = 1
)
```

Arguments

k_true	Number of clusters
clustSizeEq	Observations per cluster
numsignal	Number of numerical signal variables
numnoise	Number of numerical noise variables
catsignal	Number of categorical signal variables
catnoise	Number of categorical noise variables
q	Number of categories for signal categorical variables
q_err	Number of categories for categorical noise
numsep	Separation for numerical signal
catsep	Separation for categorical signal
seed	Optional seed

error_type	Error distribution ("normal" or "chisq")
error_df	Degrees of freedom for chi-square noise
error_scale	Scale of noise

Value

A list with elements df, X_num, X_cat, y, num_cols, cat_cols

generate_dataset	<i>generate toy mixed datasets for the supplementary material: figures 1 to 3</i>
------------------	---

Description

generate toy mixed datasets for the supplementary material: figures 1 to 3

Usage

```
generate_dataset(  
  n,  
  porig,  
  pn,  
  pnnoise,  
  pcnoise,  
  sigma,  
  qoptions,  
  seed = NULL,  
  mode = "per_variable"  
)
```

Arguments

n	number of observations
porig	number of original informative continuous variables
pn	number of continuous variables (total, before adding noise variables)
pnnoise	number of extra numeric noise variables
pcnoise	number of extra categorical noise variables
sigma	sd for noise added to informative variables
qoptions	number of bins for categorization (vector if per_variable, scalar if shared)
seed	optional seed
mode	either "per_variable" or "shared"

knn_dist	<i>Direct nearest-neighbour prediction</i>
----------	--

Description

Predict outcomes from precomputed test-to-training distances, or compute those distances from tabular training and test predictors before predicting.

Usage

```
knn_dist(
  x,
  y = NULL,
  new_data = NULL,
  k = 5,
  dist_args = list(),
  type = c("class", "prob", "numeric"),
  response = NULL
)
```

Arguments

x	Tabular training predictors when new_data is supplied. Otherwise, a numeric test-to-training distance matrix, data frame, or MDist object.
y	Training outcomes, in the same order as the distance columns or training rows. Use a factor for classification and numeric outcomes for regression. Missing outcomes are not supported. May be omitted when response selects an outcome column from tabular training data.
new_data	Optional tabular test predictors. When supplied, <code>mdist()</code> computes their distances to the training data in x. Leave NULL when x already contains test-to-training distances.
k	Integer number of neighbours, between one and the number of training observations.
dist_args	Named list of additional arguments passed to <code>mdist()</code> , such as <code>list(preset = "gower")</code> . Used only with tabular input.
type	Prediction type: "class" for class labels, "prob" for class probabilities, or "numeric" for regression predictions.
response	Optional outcome column name (a string or unquoted column name) in tabular x. Supply either y or response, not both. The selected column is excluded from the training and test predictors before distance computation. This selects the prediction outcome, not the response argument of <code>mdist()</code> .

Details

Each distance row corresponds to a test observation and each column to a training observation. Distances must be finite and nonnegative. Classification uses an unweighted majority vote; regression uses the mean of the neighbours' outcomes. Distance ties are resolved in training order and class-vote ties in factor-level order. Class probabilities are neighbour class proportions.

No training-to-training matrix needs to be supplied. For tabular input, `mdist()` uses `x` as training data and `new_data` as test data. The function computes distances on each call; it does not return a fitted model. For fitted tidymodels workflows, use `nearest_neighbor_dist()` with `step_md` instead.

Value

A factor for `type = "class"`, a data frame with one column per outcome level for `type = "prob"`, or a numeric vector for `type = "numeric"`. Each result has one row or element per test observation.

See Also

`mdist()`, `nearest_neighbor_dist()`, `step_md`

Examples

```
train <- data.frame(value = c(0, 2, 8, 10))
test <- data.frame(value = c(1, 9))
labels <- factor(c("low", "low", "high", "high"))
knn_dist(train, labels, new_data = test, k = 1,
  dist_args = list(preset = "gower"))
train$label <- labels
knn_dist(train, response = "label", new_data = test, k = 1,
  dist_args = list(preset = "gower"))
d <- mdist(train["value"], new_data = test, preset = "gower")
knn_dist(d, labels, k = 1)
knn_dist(d, labels, k = 2, type = "prob")
knn_dist(d, c(0, 2, 8, 10), k = 2, type = "numeric")
```

lovo_md

Leave-one-variable-out diagnostics for distance-based variable importance

Description

Computes leave-one-variable-out (LOVO) diagnostics for a distance specification. The function first computes the full dissimilarity matrix using `[mdist()]`. It then removes one predictor at a time, recomputes the dissimilarity matrix, and compares each leave-one-variable-out matrix with the full one.

Usage

```
lovo_mdists(
  x,
  response = NULL,
  ...,
  mds = FALSE,
  dims = 2,
  keep_dist = FALSE,
  cluster_k = NULL,
  cluster_methods = c("pam", "hclust", "spectral"),
  hclust_method = "average",
  spectral_sigma = NULL,
  spectral_nstart = 50,
  response_used = TRUE
)
```

Arguments

<code>x</code>	A data frame or object coercible to a tibble. Rows are observations and columns are variables used to compute the dissimilarity.
<code>response</code>	Optional response variable. It can be supplied as an unquoted column name or as a character string. When supplied and <code>'response_used = TRUE'</code> , it is passed to <code>[mdist()]</code> for response-aware distance construction. The response column is not treated as a predictor in the leave-one-variable-out loop.
<code>...</code>	Additional arguments passed to <code>[mdist()]</code> , such as <code>'preset'</code> , <code>'method_cat'</code> , <code>'method_num'</code> , <code>'commensurable'</code> , or <code>'interaction'</code> .
<code>mds</code>	Logical. If <code>'TRUE'</code> , compute classical multidimensional scaling diagnostics for the full and leave-one-variable-out dissimilarities. The default is <code>'FALSE'</code> .
<code>dims</code>	Integer. Number of dimensions used by classical multidimensional scaling when <code>'mds = TRUE'</code> . Ignored when <code>'mds = FALSE'</code> .
<code>keep_dist</code>	Logical. If <code>'TRUE'</code> , store the full dissimilarity matrix and all leave-one-variable-out dissimilarity matrices in the returned object.
<code>cluster_k</code>	Optional integer. Number of clusters used when computing clustering-based LOVO diagnostics. If <code>'NULL'</code> , clustering diagnostics are not computed.
<code>cluster_methods</code>	Character vector specifying the clustering methods used for clustering-based diagnostics. Possible values are <code>"pam"</code> , <code>"hclust"</code> , and <code>"spectral"</code> .
<code>hclust_method</code>	Character string specifying the linkage method passed to <code>[stats::hclust()]</code> when <code>"hclust"</code> is included in <code>'cluster_methods'</code> .
<code>spectral_sigma</code>	Optional numeric value for the Gaussian affinity bandwidth used by spectral clustering. If <code>'NULL'</code> , the default used by <code>[spectral_dist()]</code> is applied.
<code>spectral_nstart</code>	Integer. Number of random starts used by the k-means step in spectral clustering.

response_used Logical. If 'TRUE', the response variable, when supplied, is used in the distance construction. If 'FALSE', the response column is removed before computing distances.

Details

'lovo_md()' is useful for assessing how strongly each predictor contributes to a distance-based representation. A predictor is considered influential when removing it produces a large change in the dissimilarity matrix or an optional downstream representation, such as a multidimensional scaling configuration or clustering partition.

In 'autoplot()', 'reorder = TRUE' ranks variables by the selected metric within each variable type, with categorical variables before numerical variables. This preserves the blue/red background bands. 'top_n' selection remains global; without type information, ordering is global.

The returned object contains several LOVO diagnostics. The main distance contribution is measured by the mean absolute difference between the full dissimilarity matrix and each leave-one-variable-out matrix ('mad_importance'). The normalized version is stored as 'relative_distance'.

If 'mds = TRUE', the function also compares classical multidimensional scaling configurations computed from the full and leave-one-variable-out dissimilarities. These diagnostics are stored as 'mds_congruence' / 'cc_importance' and 'ac_importance', the latter corresponding to an alienation coefficient. They are omitted when 'mds = FALSE'.

If 'cluster_k' is supplied, the function additionally computes clustering partitions from the full and leave-one-variable-out dissimilarities and compares them using the adjusted Rand index. The corresponding importance measures are defined as '1 - ARI' and are stored as 'pam_importance', 'hclust_importance', or 'spectral_importance', depending on the selected clustering methods.

Clustering-based diagnostics require the suggested package 'mclust'.

Value

An object of class '"MDistLOVO"'. The main results are stored in the '\$results' field as a tibble with one row per left-out variable. The object also has print, summary, and autoplot methods.

See Also

[mdist()], [compare_lovo_md()], [spectral_dist()]

Examples

```
if (requireNamespace("palmerpenguins", quietly = TRUE)) {
  data("penguins", package = "palmerpenguins")

  penguins_small <- palmerpenguins::penguins |>
    dplyr::select(
      species, bill_length_mm, bill_depth_mm, flipper_length_mm,
      body_mass_g, island, sex
    ) |>
    tidyr::drop_na()

  # Distance-based LOVO diagnostics for a Gower distance
  res <- lovo_md(
```

```

    penguins_small,
    preset = "gower",
    response = species,
    response_used = FALSE
  )

  res
  summary(res)

  # Plot the relative distance contribution of each predictor
  p <- res$autoplot(metric = "relative_distance")
  p
}

```

lovo_method_spec

*Create a LOVO method specification for 'compare_lovo_mdists()'***Description**

Helper to build one method specification for [compare_lovo_mdists()]. This allows tidy-style specification of 'response', e.g. 'response = Name', while storing the method definition as a regular list.

Usage

```
lovo_method_spec(response = NULL, ...)
```

Arguments

response	Optional response column, supplied either unquoted (e.g. 'Name') or quoted (e.g. "Name").
...	Additional arguments passed on to [lovo_mdists()] through [compare_lovo_mdists()].

Value

A named list of arguments suitable for one element of the 'methods' argument in [compare_lovo_mdists()].

Examples

```

## Not run:
methods <- list(
  tvd_sup = lovo_method_spec(
    response = species,
    preset = "custom",
    method_cat = "tvd",
    method_num = "std",
    commensurable = TRUE,
    response_used = TRUE
  )
)

```

```

    ),
    tvd_unsup = lovo_method_spec(
      response = species,
      preset = "custom",
      method_cat = "tvd",
      method_num = "std",
      commensurable = TRUE,
      response_used = FALSE
    )
  )
)

## End(Not run)

```

make_mdists_recipe

Build a recipe that computes mdist-based distance features

Description

This helper creates a tidymodels recipe using `step_mdists()`. It supports both mdist presets and custom param sets.

Usage

```
make_mdists_recipe(df, mdist_type, mdist_preset, param_set, outcome)
```

Arguments

<code>df</code>	A data frame.
<code>mdist_type</code>	"preset" or "custom".
<code>mdist_preset</code>	Name of the preset (if <code>mdist_type == "preset"</code>).
<code>param_set</code>	A list of custom mdist arguments (if <code>mdist_type == "custom"</code>).
<code>outcome</code>	Name of the outcome variable.

Value

A `'recipes::recipe()'` object.

mdist

*Mixed-type dissimilarities for distance-based learning***Description**

Computes a dissimilarity object for numerical, categorical, or mixed-type data. The function combines continuous and categorical components according to either a predefined ‘preset’ or a user-defined custom specification.

Usage

```
mdist(
  x,
  new_data = NULL,
  response = NULL,
  method_cat = "tvd",
  method_num = "std",
  commensurable = TRUE,
  ncomp = NULL,
  threshold = NULL,
  preset = "custom",
  interaction = FALSE,
  prop_nn = 0.1,
  score = "ba",
  decision = "prior_corrected",
  gower_average = TRUE
)
```

Arguments

x	A data frame or matrix containing the training observations. Columns can be numeric, factors, or a mixture of both.
new_data	Optional data frame or matrix containing new observations. If supplied, distances are computed from rows of ‘new_data’ to rows of ‘x’, producing a rectangular test-to-training dissimilarity matrix.
response	Optional response variable used for response-aware categorical dissimilarities. It can be supplied as an unquoted column name or as a character string. The response column is removed from the predictors before computing distances.
method_cat	Character string specifying the categorical-variable dissimilarity used when ‘preset = "custom"’. Common values include “matching” and “tvd”. Use <code>[all_dist_method_specs()]</code> to inspect available methods.
method_num	Character string specifying numerical-variable preprocessing. Supported values are “none” for no preprocessing, “std” for standard-deviation scaling, “range” for range scaling, “robust” for inter-quartile-range-based scaling, and “pc_scores” for principal-component score scaling. This argument is used directly when ‘preset = "custom"’. When ‘preset = "euclidean”’ and all predictors

	are numeric, it can also override the default <code>"std"</code> preprocessing; in particular, <code>"none"</code> gives ordinary Euclidean distance on the original variables.
commensurable	Logical. If <code>TRUE</code> , dissimilarities are scaled so that the average contribution of each variable to the overall distance is equal to 1 on the training data. Variable-specific means are computed from training pairs, before aggregation, and never from <code>'new_data'</code> . The current variable-wise convention includes the zero diagonal of the training square.
ncomp	Integer or <code>NULL</code> . Number of principal components to retain when <code>'method_num = "pc_scores"</code> or <code>'preset = "u_dep_bw"</code> . If <code>NULL</code> , all available components are used unless <code>'threshold'</code> is supplied and supported by the underlying method.
threshold	Numeric or <code>NULL</code> . Optional cumulative variance threshold used when <code>'method_num = "pc_scores"</code> or <code>'preset = "u_dep_bw"</code> .
preset	Character string specifying a predefined distance specification. Available values include <code>"custom"</code> , <code>"gower"</code> , <code>"unbiased_dependent"</code> , <code>"u_dep"</code> , <code>"u_dep_bw"</code> , <code>"u_indep"</code> , <code>"u_mix"</code> , <code>"hl"</code> , <code>"gudmm"</code> , <code>"dkss"</code> , <code>"mod_gower"</code> , and <code>"euclidean"</code> . When <code>'preset'</code> is not <code>"custom"</code> , arguments such as <code>'method_cat'</code> , <code>'method_num'</code> , <code>'commensurable'</code> , and <code>'interaction'</code> are normally handled by the preset and user-supplied values are ignored. The exception is <code>'method_num'</code> for <code>'preset = "euclidean"</code> when all predictors are numeric.
interaction	Logical. If <code>TRUE</code> , adds an interaction-aware continuous-categorical component based on local predictive separability.
prop_nn	Numeric. Proportion of nearest neighbours used when <code>'interaction = TRUE'</code> .
score	Character string specifying the score used when <code>'interaction = TRUE'</code> . Available values include <code>"ba"</code> for balanced accuracy and <code>"logloss"</code> .
decision	Character string specifying the decision rule used when <code>'score = "ba"</code> . The default is <code>"prior_corrected"</code> .
gower_average	Logical; only used when <code>'preset = "gower"</code> . If <code>TRUE</code> , returns the standard Gower dissimilarity averaged over variables, matching the scale of <code>[cluster::daisy()]</code> with <code>'metric = "gower"</code> . If <code>FALSE</code> , returns the sum of per-variable Gower contributions, equivalent to multiplying the averaged Gower dissimilarity by the number of active variables.

Details

`'mdist()'` is the main distance-construction function in `'manydist'`. It can return ordinary train-train dissimilarities or rectangular test-to-training dissimilarities when `'new_data'` is supplied. The resulting object stores both the dissimilarity matrix and metadata about the distance specification that was used.

With `'preset = "custom"`, users manually choose the numerical preprocessing, categorical dissimilarity, commensurability, and optional interaction term.

The `"gower"` preset follows the usual Gower construction based on range scaling for continuous variables and matching dissimilarities for categorical variables. The `'gower_average'` argument controls whether the result is averaged over variables or returned as a sum of variable-wise contributions.

The `"u_dep"`, `"unbiased_dependent"`, `"u_indep"`, and `"u_mix"` presets are convenience specifications for unbiased or commensurable mixed-variable dissimilarities. The `"u_dep_bw"` preset

uses the same categorical construction as `"u_dep"`, but whitens the retained numerical principal components and scales their complete Manhattan distance by the number of original numerical variables divided by its mean over distinct training pairs. The `"euclidean"` preset computes Euclidean distance after standardizing numerical variables and one-hot encoding and standardizing categorical variables. For numerical-only inputs, `"std"` remains the default, but `'method_num'` can be overridden. Setting `'method_num = "none"` produces the same numerical distances as `[stats::dist()]` applied directly to the original variables. The `"gudmm"`, `"dkss"`, and `"mod_gower"` presets provide additional mixed-type distance constructions. Some presets currently support only train-train distances and will stop if `'new_data'` is supplied.

Use `[all_dist_method_specs()]` to inspect the available distance components and method specifications.

Value

An object of class `"MDist"`. The object contains the computed dissimilarity in its `'$distance'` field, the selected `'preset'`, the training data, and a list of parameters describing the fitted distance specification. Square train-train dissimilarities are stored as `"dissimilarity"/"dist"` objects; rectangular test-to-training dissimilarities are stored as `"dissimilarity"/"matrix"` objects.

See Also

`[step_mdists()]`, `[all_dist_method_specs()]`

Examples

```
if (requireNamespace("palmerpenguins", quietly = TRUE)) {
  data("penguins", package = "palmerpenguins")

  penguins_small <- palmerpenguins::penguins |>
    dplyr::select(
      bill_length_mm, bill_depth_mm, flipper_length_mm,
      body_mass_g, species, island, sex
    ) |>
    tidyr::drop_na()

  # Gower distance on mixed-type data
  d_gower <- mdist(penguins_small, preset = "gower")
  d_gower

  # Custom mixed-type specification
  d_custom <- mdist(
    penguins_small,
    preset = "custom",
    method_cat = "matching",
    method_num = "std",
    commensurable = TRUE
  )

  d_custom

  # Train-to-new-data distances
```

```

penguin_split <- rsample::initial_split(penguins_small, prop = 0.75)
penguin_train <- rsample::training(penguin_split)
penguin_test  <- rsample::testing(penguin_split)

d_new <- mdist(
  penguin_train,
  new_data = penguin_test,
  preset = "gower"
)

d_new
}

```

mdist_summary_impl	<i>Internal: summary method implementation for MDist R6 objects</i>
--------------------	---

Description

Internal: summary method implementation for MDist R6 objects

Usage

```
mdist_summary_impl(object, ...)
```

Arguments

object	An 'MDist' object to summarize.
...	Additional arguments currently not used.

Value

Invisibly returns 'object'.

pam_dist	<i>PAM clustering specification based on manydist dissimilarities</i>
----------	---

Description

PAM clustering specification based on manydist dissimilarities

Usage

```
pam_dist(num_clusters = NULL)
```

Arguments

num_clusters Number of clusters.

Value

A 'pam_dist_spec' object.

response_aware_methods

List response-aware methods

Description

Returns the methods in [dist_methods_tbl()] that can use a response variable. The output can optionally be restricted by data type or by 'mdist()' argument.

Usage

```
response_aware_methods(data_type = NULL, argument = NULL)
```

Arguments

data_type Optional character vector used to restrict the returned methods by data type, for example "categorical" or "mixed".

argument Optional character vector used to restrict the returned methods by argument, for example "method_cat" or "preset".

Value

A character vector of response-aware method names.

Examples

```
response_aware_methods()  
response_aware_methods(argument = "method_cat")  
response_aware_methods(argument = "preset")
```

spectral_dist

*Spectral clustering specification based on manydist dissimilarities***Description**

Spectral clustering specification based on manydist dissimilarities

Usage

```
spectral_dist(num_clusters = NULL, sigma = NULL, nstart = 50)
```

Arguments

num_clusters	Number of clusters.
sigma	Optional bandwidth for the Gaussian affinity. If 'NULL', the median pairwise distance is used.
nstart	Number of random starts for k-means.

Value

A 'spectral_dist_spec' object.

Examples

```
## Not run:
library(manydist)
library(palmerpenguins)
library(recipes)
library(generics)

data <- penguins |>
  dplyr::select(-species) |>
  tidyr::drop_na()

rec <- recipes::recipe(~ ., data = data) |>
  step_mdist(all_predictors(), preset = "gower", output = "pairwise")

spec <- spectral_dist(num_clusters = 3)

fit_obj <- generics::fit(spec, recipe = rec, data = data)

print(fit_obj)
predict(fit_obj)
predict(fit_obj, type = "embed")

## End(Not run)
```

spectral_from_dist	<i>Spectral clustering from a distance matrix</i>
--------------------	---

Description

Spectral clustering from a distance matrix

Usage

```
spectral_from_dist(D, k, affinity_method = "selftune")
```

Arguments

D	A distance matrix
k	Number of clusters
affinity_method	Method to build affinity

Value

A vector of cluster labels

step_mdists	<i>Add many distance dissimilarities to a recipe</i>
-------------	--

Description

'step_mdists()' is a [recipes::recipe()] step that replaces selected predictors by a distance-based representation computed with [mdists()]. It is designed for distance-based learning workflows, especially nearest-neighbour prediction and clustering models that operate on dissimilarity matrices.

Usage

```
step_mdists(
  recipe,
  ...,
  role = "predictor",
  trained = FALSE,
  output = "distance_to_training",
  response_used = TRUE,
  preset = "custom",
  method_cat = "tot_var_dist",
  method_num = NULL,
  commensurable = FALSE,
  ncomp = NULL,
```

```

threshold = NULL,
columns = NULL,
response_col = NULL,
train_predictors = NULL,
preprocessor = NULL,
skip = FALSE,
id = recipes::rand_id("mdist")
)

```

Arguments

recipe	A recipe object.
...	Selector(s) for the predictor columns to be used in [mdist()]. These are passed to [recipes::recipes_eval_select()] during preparation.
role	Role for the new distance columns. The default is "predictor".
trained	Logical for recipes internals. Do not set manually.
output	Character string specifying the type of distance output. "distance_to_training" returns distances from the baked data to the training observations and is the usual choice for prediction workflows. "pairwise" returns the within-training pairwise dissimilarity matrix and is intended for training-only distance-based clustering workflows.
response_used	Logical. If 'TRUE' (the default) and the recipe has exactly one outcome, response-aware distance specifications use that outcome when the step is prepared. Set to 'FALSE' to construct the distance from predictors only. Specifications that are not response-aware never use the outcome.
preset	Character string specifying the distance preset passed to [mdist()]. Available values include "custom", "gower", "unbiased_dependent", "u_dep", "u_dep_bw", "u_indep", "u_mix", "hl", "gudmm", "dkss", "mod_gower", and "euclidean". Preset parameters are normally fixed by the selected preset. The exception is 'method_num' for 'preset = "euclidean"' when all predictors selected by the step are numeric.
method_cat	Character string specifying the categorical-variable dissimilarity passed to [mdist()] when 'preset = "custom"'. Common values include "matching" and "tvd". Use [all_dist_method_specs()] to inspect available methods.
method_num	Character string or 'NULL' specifying numerical-variable preprocessing. Supported values are "none" for no preprocessing, "std" for standard-deviation scaling, "range" for range scaling, "robust" for inter-quartile-range-based scaling, and "pc_scores" for principal-component score scaling. With 'preset = "custom"', 'NULL' defaults to "none". With 'preset = "euclidean"', the default is "std"; when all predictors selected by the step are numeric, an explicitly supplied value can override that default.
commensurable	Logical. If 'TRUE', dissimilarities are scaled so that the average contribution of each variable to the overall distance is equal to 1, when supported by the selected distance specification.
ncomp	Integer or 'NULL'. Number of principal components to retain when 'method_num = "pc_scores"' or 'preset = "u_dep_bw"'. If 'NULL', all available components are used unless 'threshold' is supplied and supported by the underlying method.

threshold	Numeric or 'NULL'. Optional cumulative variance threshold used when 'method_num = "pc_scores"' or 'preset = "u_dep_bw"'.
columns	Names of columns selected at prep time. Used internally by recipes.
response_col	Name of the outcome selected at prep time when it is used by the distance specification. Used internally by recipes.
train_predictors	Training predictors stored at prep time. Used internally by recipes to compute distances from new observations to the training observations.
preprocessor	Internal fitted manydist preprocessor.
skip	Logical. Standard recipes argument indicating whether the step should be skipped when baking new data.
id	Character string. Unique step identifier.

Details

The step can produce either distances from new observations to the training observations, or the within-training pairwise dissimilarity matrix. The former is the usual choice for supervised prediction workflows; the latter is useful for distance-based clustering workflows fitted on the training data.

During [recipes::prep()], 'step_mdists()' stores the selected training predictors and fits the internal manydist preprocessor. During [recipes::bake()], the selected predictors are removed and replaced by distance columns named 'dist_1', 'dist_2', and so on.

When 'response_used = TRUE', the step discovers the outcome from the recipe variable roles. For response-aware presets and custom categorical methods, the outcome from the current analysis set is supplied to [mdists()] during preparation. The fitted response-aware profiles are then reused when new data are baked; assessment and test outcomes are never required or used.

With 'output = "distance_to_training"', baking the training data returns the training pairwise distances, while baking new data returns distances from each new observation to each training observation. This rectangular representation is suitable for nearest-neighbour prediction models.

With 'output = "pairwise"', the step returns the within-training pairwise dissimilarity matrix. Baking genuinely new data is not supported in this mode, because the output is intended for training-only clustering workflows such as [pam_dist()] or [spectral_dist()].

Value

An updated recipe with a manydist step.

See Also

[mdists()], [nearest_neighbor_dist()], [pam_dist()], [spectral_dist()], [all_dist_method_specs()]

Examples

```
if (requireNamespace("palmerpenguins", quietly = TRUE)) {
  data("penguins", package = "palmerpenguins")

  penguins_small <- palmerpenguins::penguins |>
```

```

dplyr::select(
  species, bill_length_mm, bill_depth_mm, flipper_length_mm,
  body_mass_g, island, sex
) |>
tidyr::drop_na()

# Distance-to-training representation for prediction workflows
rec <- recipes::recipe(species ~ ., data = penguins_small) |>
  step_mdists(
    recipes::all_predictors(),
    preset = "gower",
    output = "distance_to_training"
  )

rec_prep <- recipes::prep(rec, training = penguins_small)
baked <- recipes::bake(rec_prep, new_data = penguins_small)

baked |> dplyr::slice_head(n=5)

# Pairwise representation for clustering workflows
rec_pairwise <- recipes::recipe(~ ., data = penguins_small) |>
  step_mdists(
    recipes::all_predictors(),
    preset = "gower",
    output = "pairwise"
  )

rec_pairwise_prep <- recipes::prep(rec_pairwise, training = penguins_small)
pairwise_dist <- recipes::bake(rec_pairwise_prep, new_data = penguins_small)

pairwise_dist
}

```

summary.MDistBenchmark

Summarize a distance benchmark

Description

Displays the pairwise diagnostics already computed by `benchmark_mdists()`, without refitting distances or recomputing diagnostics.

Usage

```

## S3 method for class 'MDistBenchmark'
summary(object, metric = NULL, digits = 3L, ..., n = Inf, width = NULL)

```

Arguments

<code>object</code>	An object returned by <code>benchmark_mdists()</code> .
<code>metric</code>	Character vector selecting diagnostics to display. NULL (the default) displays all available diagnostics. Choices include "mad", "relative_distance", "mds_congruence", "alienation", and available ARI columns such as "ari_pam". "ari" selects all available ARI columns.
<code>digits</code>	Number of decimal places in the displayed tables.
<code>...</code>	Additional arguments passed to the matrix print method.
<code>n</code>	Maximum number of methods to display in each table. The default, Inf, displays every successful method. The returned tibble is not limited.
<code>width</code>	Width of the printed output; NULL uses the default width.

Value

Invisibly returns the complete tibble with one row per unique pair of successful specifications, regardless of `metric`, `digits`, or `n`. Columns include the method labels, MAD, symmetric relative distance, MDS congruence, alienation, and any requested clustering ARIs. If fewer than two specifications succeed, the tibble has zero rows.

Examples

```
## Not run:
pairwise_results <- summary(benchmark_result)
summary(benchmark_result, metric = "relative_distance")
summary(benchmark_result, metric = c("alienation", "ari"))

## End(Not run)
```

wdi_2022

*Selected World Development Indicators for 2022***Description**

A fixed snapshot of selected economic, demographic, and labour-market indicators for 181 countries, together with World Bank regional, income, and lending classifications. It provides a compact mixed-type dataset for illustrating distance construction and distance-based learning.

Format

A data frame with 181 rows and 9 variables:

country Country or economy name.

region World Bank region, as a factor with 7 levels.

income World Bank income group, as a factor with 5 levels.

lending World Bank lending category, as a factor with 4 levels.

gdp_pc_kusd GDP per capita in thousands of current US dollars.

life_exp Life expectancy at birth, in years.

unemployment Unemployment as a percentage of the total labour force, based on the modelled ILO estimate.

urban_pop Urban population as a percentage of the total population.

pop_growth Annual population growth, in percent.

Details

The reference year is 2022 and the snapshot was retrieved from the World Bank Indicators API on 2026-05-05. Aggregate regions and observations with missing values in any selected field were removed. GDP per capita was divided by 1,000 and rounded to one decimal place; life expectancy, unemployment, and urban population were rounded to one decimal place; and population growth was rounded to two decimal places.

The numerical indicators and their World Bank codes are:

- GDP per capita: NY.GDP.PCAP.CD;
- life expectancy: SP.DYN.LE00.IN;
- unemployment: SL.UEM.TOTL.ZS;
- urban population: SP.URB.TOTL.IN.ZS;
- population growth: SP.POP.GROW.

License

The original World Development Indicators data are distributed by the World Bank under the Creative Commons Attribution 4.0 International license, subject to the World Bank Dataset Terms. Attribution: The World Bank: World Development Indicators.

Source

The World Bank, *World Development Indicators*, <https://datacatalog.worldbank.org/search/dataset/0037712/world-development-indicators>.

References

World Bank Dataset Terms, <https://www.worldbank.org/ext/en/legal/terms-conditions/datasets>.

Examples

```
data("wdi_2022", package = "manydist")
dplyr::glimpse(wdi_2022)
```

Index

* datasets

fifa_nl, [11](#)
wdi_2022, [31](#)

all_dist_method_specs, [3](#)
autoplot.MDistBenchmark, [4](#)

benchmark_mdists, [5](#)
benchmark_mdists(), [30](#), [31](#)

compare_lovo_mdists, [7](#)
congruence_coeff, [9](#)

dist_methods_tbl, [10](#)
dist_methods_tbl_cat, [11](#)

fifa_nl, [11](#)

gen_mixed, [13](#)
generate_dataset, [14](#)

knn_dist, [15](#)

lovo_mdists, [8](#), [9](#), [16](#)
lovo_method_spec, [19](#)

make_mdists_recipe, [20](#)
mdists, [9](#), [21](#)
mdists(), [15](#), [16](#)
mdists_summary_impl, [24](#)

nearest_neighbor_dist(), [16](#)

pam_dist, [24](#)

response_aware_methods, [25](#)

spectral_dist, [26](#)
spectral_from_dist, [27](#)
step_mdists, [27](#)
step_mdists(), [16](#)
summary.MDistBenchmark, [30](#)

wdi_2022, [31](#)