Package 'tidydann'

August 29, 2026

Title Add the 'dann' Model and the 'sub_dann' Model to the 'tidymodels' Ecosystem

Version 1.0.2

Description Provides model specifications and tuning parameters that make the models in the 'dann' package available to the 'tidymodels' ecosystem. Models based on Hastie (1996) <https://web.stanford.edu/~hastie/Papers/dann_IEEE.pdf>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Suggests testthat (>= 3.0.0), recipes, mlbench, modeldata, workflows, rsample, dplyr, magrittr, tune, scales, yardstick

Config/testthat/edition 3

Imports dials, dann (>= 1.3.0), generics, tibble, parsnip, rlang

URL https://github.com/gmcmacran/tidydann

BugReports https://github.com/gmcmacran/tidydann/issues

NeedsCompilation no

Author Greg McMahan [aut, cre]

Maintainer Greg McMahan <gmcmacran@gmail.com>

Repository CRAN

Date/Publication 2026-08-29 18:00:02 UTC

Contents

matrix_diagonal 2
nearest_neighbor_adaptive 3
neighborhood 4
sphere 5
tidydann_set_threads 6

tunable.nearest_neighbor_adaptive 7
update.nearest_neighbor_adaptive 7
weighted 9

Index **10**

`matrix_diagonal` *Softening*

Description

Scales the identity matrix added to the between class covariance, which keeps the neighborhood from collapsing onto the class boundary.

Usage

```
matrix_diagonal(range = c(0, 2), trans = NULL)
```

Arguments

`range`	A two-element vector holding the defaults for the smallest and largest possible values, respectively. If a transformation is specified, these values should be in the transformed units.
`trans`	A trans object from the scales package, such as scales::log10_trans() or scales::reciprocal_trans(). If not provided, the default is used, which matches the units used in range. If no transformation, NULL.

Details

A value of 1 matches the publication. Of the tuning parameters, this one usually has the smallest effect on performance.

Value

An S3 class of type quant_param from the dials package.

Examples

```
library(tidydann)

matrix_diagonal()
```

nearest_neighbor_adaptive *Discriminant Adaptive Nearest Neighbor Classification*

Description

Defines a nearest neighbor model where the shape of the neighborhood is learned from the data. This is a parsnip specification for the models in the dann package.

Usage

```
nearest_neighbor_adaptive(
  mode = "classification",
  neighbors = NULL,
  neighborhood = NULL,
  matrix_diagonal = NULL,
  weighted = NULL,
  sphere = NULL,
  num_comp = NULL
)
```

Arguments

mode	A single character string for the type of model. The only possible value for this model is "classification".
neighbors	The number of nearest neighbors used to classify a point. Identical to k in standard k nearest neighbors.
neighborhood	The number of nearest neighbors used to estimate the between and within class covariance matrices that shape the neighborhood.
matrix_diagonal	Softening parameter. Scales the identity matrix added to the between class covariance, which keeps the neighborhood from collapsing onto the class boundary. 1 matches the publication.
weighted	Should the between class covariance matrices be weighted? FALSE matches the publication. Passed to `fpc::ncoord()`. Only used by the sub_dann engine.
sphere	Type of covariance matrix used to sphere the data. One of "mcd", "mve", "classical", or "none". Passed to `fpc::ncoord()`. Only used by the sub_dann engine.
num_comp	Number of dimensions in the subspace dann is fit on. Only used by the sub_dann engine.

Details

Discriminant Adaptive Nearest Neighbor (dann) is a variation of k nearest neighbors where the shape of the neighborhood is data driven. The neighborhood is elongated along class boundaries and shrunk in the orthogonal direction.

Two engines are available. The dann engine fits `dann::dann()` and uses neighbors, neighborhood, and matrix_diagonal. The sub_dann engine fits `dann::sub_dann()`, which first projects the predictors onto a lower dimensional subspace and then fits dann there. It uses every argument. Setting weighted, sphere, or num_comp with the dann engine is an error.

Prediction is parallelized. See `tidydann_set_threads()` to control how many threads are used.

Value

An S3 class of type nearest_neighbor_adaptive.

Examples

```
library(parsnip)
library(tidydann)

data("two_class_dat", package = "modeldata")

previous <- tidydann_set_threads(2)

model <- nearest_neighbor_adaptive(neighbors = 2) |>
  set_engine("dann") |>
  fit(formula = Class ~ A + B, data = two_class_dat)

model |>
  predict(new_data = two_class_dat)

tidydann_set_threads(previous)
```

---

neighborhood *Neighborhood size*

Description

The number of nearest neighbors used to estimate the between and within class covariance matrices that shape the neighborhood.

Usage

```
neighborhood(range = c(2L, dials::unknown()), trans = NULL)
```

Arguments

range	A two-element vector holding the defaults for the smallest and largest possible values, respectively. If a transformation is specified, these values should be in the transformed units.
trans	A trans object from the scales package, such as scales::log10_trans() or scales::reciprocal_trans(). If not provided, the default is used, which matches the units used in range. If no transformation, NULL.

Details

The upper end of the range depends on the size of the training data, so it is unknown until the data are seen. Use get_n() or finalize() from dials to fill it in.

When tuning with cross validation, each model only sees part of the training data. Use get_n_frac() with frac set to 1/V. See the README for a worked example.

Value

An S3 class of type quant_param from the dials package.

Examples

```
library(dials)
library(tidydann)

data("taxi", package = "modeldata")
neighborhood() |> finalize(taxi)

neighborhood() |> get_n(taxi)
```

---

sphere *Sphering method*

Description

Type of covariance matrix used to sphere the data. Only used by the sub_dann engine.

Usage

```
sphere(values = c("mcd", "mve", "classical", "none"))
```

Arguments

values	A vector of candidate values. Any combination of "mcd", "mve", "classical", and "none".

Details

Passed to the sphere argument of `fpc::ncoord()`.

Value

An S3 class of type qual_param from the dials package.

Examples

```
library(tidydann)

sphere()
```

`tidydann_set_threads` *Control the number of threads tidydann uses*

Description

Get and set the number of threads the dann and sub_dann engines use when predicting. These are thin wrappers around `dann::dann_set_threads()`, `dann::dann_get_threads()`, and `dann::dann_has_openmp()`.

Usage

```
tidydann_set_threads(n = NULL)

tidydann_get_threads()

tidydann_has_openmp()
```

Arguments

`n`	The number of threads to use. A positive whole number, or NULL to restore the default.

Details

On Linux and Windows, the prediction loop inside the dann package is parallelized with OpenMP. By default it uses every core the OpenMP runtime makes available. These functions change that count for the dann package alone, so no other package that uses OpenMP is affected. The setting lasts for the R session and is not stored on model objects.

This allows the user to move threading from a single model's predict to tidy model's tune or other logic. Thread usage can be changed without refitting.

`n` is clamped to the number of threads the OpenMP runtime makes available, with a message. Without OpenMP support, prediction runs on a single thread, `tidydann_get_threads` returns 1 no matter what was set, and `tidydann_has_openmp` returns FALSE.

Value

`tidydann_set_threads` returns the previous setting invisibly: a positive whole number, or NULL if dann was using the default. `tidydann_get_threads` returns the number of threads the next prediction will use. `tidydann_has_openmp` returns TRUE if dann was compiled with OpenMP.

Examples

```
library(tidydann)

# Limit dann to two threads.
previous <- tidydann_set_threads(2)
tidydann_get_threads()
```

```
# Put it back.
tidydann_set_threads(previous)

tidydann_has_openmp()
```

`tunable.nearest_neighbor_adaptive` *Declare tunable parameters*

Description

Returns information on potential hyper-parameters that can be optimized.

Usage

```
## S3 method for class 'nearest_neighbor_adaptive'
tunable(x, ...)
```

Arguments

`x`	A model specification of type nearest_neighbor_adaptive.
`...`	Other arguments passed to methods.

Details

The result depends on the engine. The dann engine does not use weighted, sphere, or num_comp, so those are omitted for it. When no engine has been set, every parameter is returned.

Value

A tibble with a column for the parameter name, information on the default method for generating a corresponding parameter object, the source of the parameter (e.g. "recipe", etc.), and the component within the source.

`update.nearest_neighbor_adaptive` *Update a model specification*

Description

If parameters of a model specification need to be modified, update() can be used in lieu of recreating the object from scratch.

Usage

```
## S3 method for class 'nearest_neighbor_adaptive'
update(
  object,
  parameters = NULL,
  neighbors = NULL,
  neighborhood = NULL,
  matrix_diagonal = NULL,
  weighted = NULL,
  sphere = NULL,
  num_comp = NULL,
  fresh = FALSE,
  ...
)
```

Arguments

`object`	A model specification.
`parameters`	A 1-row tibble or named list with main parameters to update. Use either parameters or the main arguments directly when updating. If the main arguments are used, these will supersede the values in parameters. Also, using engine arguments in this object will result in an error.
`neighbors`	The number of nearest neighbors used to classify a point. Identical to k in standard k nearest neighbors.
`neighborhood`	The number of nearest neighbors used to estimate the between and within class covariance matrices that shape the neighborhood.
`matrix_diagonal`	Softening parameter. Scales the identity matrix added to the between class covariance, which keeps the neighborhood from collapsing onto the class boundary. 1 matches the publication.
`weighted`	Should the between class covariance matrices be weighted? FALSE matches the publication. Passed to `fpc::ncoord()`. Only used by the sub_dann engine.
`sphere`	Type of covariance matrix used to sphere the data. One of "mcd", "mve", "classical", or "none". Passed to `fpc::ncoord()`. Only used by the sub_dann engine.
`num_comp`	Number of dimensions in the subspace dann is fit on. Only used by the sub_dann engine.
`fresh`	A logical for whether the arguments should be modified in-place or replaced wholesale.
`...`	Not used for update().

Value

An updated S3 class of type nearest_neighbor_adaptive.

Examples

```
library(parsnip)
library(tidydann)

model <- nearest_neighbor_adaptive(neighbors = 2)

model |> update(neighbors = 5)
```

`weighted` *Weighted between class covariance*

Description

Should the between class covariance matrices be weighted? FALSE matches the publication. Only used by the sub_dann engine.

Usage

```
weighted(values = c(FALSE, TRUE))
```

Arguments

`values`	A vector of candidate values. Any combination of FALSE and TRUE.

Details

Passed to the weighted argument of `fpc::ncoord()`.

Value

An S3 class of type qual_param from the dials package.

Examples

```
library(tidydann)

weighted()
```

Index

dann::dann(), *4*
dann::dann_get_threads(), *6*
dann::dann_has_openmp(), *6*
dann::dann_set_threads(), *6*
dann::sub_dann(), *4*

fpc::ncoord(), *3*, *5*, *8, 9*

matrix_diagonal, 2

nearest_neighbor_adaptive, 3
neighborhood, 4

sphere, 5

tidydann_get_threads
        *(*tidydann_set_threads*)*, 6
tidydann_has_openmp
        *(*tidydann_set_threads*)*, 6
tidydann_set_threads, 6
tidydann_set_threads(), *4*
tunable.nearest_neighbor_adaptive, 7

update.nearest_neighbor_adaptive, 7

weighted, 9