Package 'validann'

July 22, 2025

Type Package

Title Validation Tools for Artificial Neural Networks

Version 1.2.1

Date 2016-11-08

Description Methods and tools for analysing and validating the outputs and modelled functions of artificial neural networks (ANNs) in terms of predictive, replicative and structural validity. Also provides a method for fitting feed-forward ANNs with a single hidden layer.

Depends R (>= 3.1.0)

Imports moments

Suggests nnet, knitr, rmarkdown

License GPL (>= 2)

URL http://github.com/gbhumphrey1/validann

BugReports http://github.com/gbhumphrey1/validann/issues

RoxygenNote 5.0.1

NeedsCompilation no

Author Greer B. Humphrey [aut, cre]

Maintainer Greer B. Humphrey <greer.humphrey@student.adelaide.edu.au>

Repository CRAN

Date/Publication 2017-04-20 08:35:10 UTC

Contents

ann *Fit Artificial Neural Networks.*

Description

Fits a single hidden layer ANN model to input data `x` and output data `y`.

Usage

```
ann(x, y, size, act_hid = c("tanh", "sigmoid", "linear", "exp"),
  act_out = c("linear", "sigmoid", "tanh", "exp"), Wts = NULL, rang = 0.5,
  objfn = NULL, method = "BFGS", maxit = 1000, abstol = 1e-04,
  reltol = 1e-08, trace = TRUE, ...)
```

Arguments

`x`	matrix, data frame or vector of numeric input values, with `ncol(x)` equal to the number of inputs/predictors and `nrow(x)` equal to the number of examples. A vector is considered to comprise examples of a single input or predictor variable.
`y`	matrix, data frame or vector of target values for examples.
`size`	number of hidden layer nodes. Can be zero.
`act_hid`	activation function to be used at the hidden layer. See 'Details'.
`act_out`	activation function to be used at the output layer. See 'Details'.
`Wts`	initial weight vector. If `NULL` chosen at random.
`rang`	initial random weights on [-rang,rang]. Default value is 0.5.
`objfn`	objective function to be minimised when fitting weights. This function may be user-defined with the first two arguments corresponding to `y` (the observed target data) and `y_hat` (the ANN output). If this function has additional parameters which require optimizing, these must be defined in argument `par_of` (see AR(1) case in 'Examples'). Default is `sse` (internal function to compute sum squared error, with error given by `y - y_hat`) when `objfn = NULL`.
`method`	the method to be used by `optim` for minimising the objective function. May be "Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN" or "Brent". Can be abbreviated. Default is "BFGS".
`maxit`	maximum number of iterations used by `optim`. Default value is 1000.
`abstol`	absolute convergence tolerance (stopping criterion) used by `optim`. Default is `1e-4`.
`reltol`	relative convergence tolerance (stopping criterion) used by `optim`. Optimization stops if the value returned by `objfn` cannot be reduced by a factor of `reltol * (abs(val) + reltol)` at a step. Default is `1e-8`.
`trace`	logical. Should optimization be traced? Default = TRUE.
`...`	arguments to be passed to user-defined `objfn`. Initial values of any parameters (in addition to the ANN weights) requiring optimization must be supplied in argument `par_of` (see AR(1) case in 'Examples').

Details

The "linear" activation, or transfer, function is the identity function where the output of a node is equal to its input $f(x) = x$.

The "sigmoid" function is the standard logistic sigmoid function given by $f(x) = \frac{1}{1+e^{-x}}$.

The "tanh" function is the hyperbolic tangent function given by $f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

The "exp" function is the exponential function given by $f(x) = e^x$

The default configuration of activation functions is `act_hid = "tanh"` and `act_out = "linear"`.

Optimization (minimization) of the objective function (`objfn`) is performed by `optim` using the method specified.

Derivatives returned are first-order partial derivatives of the hidden and output nodes with respect to their inputs. These may be useful for sensitivity analyses.

Value

object of class 'ann' with components describing the ANN structure and the following output components:

`wts`	best set of weights found.
`par_of`	best values of additional `objfn` parameters. This component will only be returned if a user-defined `objfn` is supplied and argument `par_of` is included in the function call (see AR(1) case in 'Examples').
`value`	value of objective function.
`fitted.values`	fitted values for the training data.
`residuals`	residuals for the training data.
`convergence`	integer code returned by `optim`. 0 indicates successful completion, see `optim` for possible error codes.
`derivs`	matrix of derivatives of hidden (columns `1:size`) and output (final column) nodes.

See Also

`predict.ann`, `validann`

Examples

```
## fit 1-hidden node ann model with tanh activation at the hidden layer and
## linear activation at the output layer.
## Use 200 random samples from ar9 dataset.
## ---
data("ar9")
samp <- sample(1:1000, 200)
y <- ar9[samp, ncol(ar9)]
x <- ar9[samp, -ncol(ar9)]
x <- x[, c(1,4,9)]
fit <- ann(x, y, size = 1, act_hid = "tanh", act_out = "linear", rang = 0.1)
```

```
## fit 3-hidden node ann model to ar9 data with user-defined AR(1) objective
## function
## ---
ar1_sse <- function(y, y_hat, par_of) {
  err <- y - y_hat
  err[-1] <- err[-1] - par_of * err[-length(y)]
  sum(err ^ 2)
}
fit <- ann(x, y, size = 3, act_hid = "tanh", act_out = "linear", rang = 0.1,
           objfn = ar1_sse, par_of = 0.7)
```

---

ar9 *Data generated by autoregressive AR9 model.*

Description

Synthetically generated dataset containing values of dependent variable `x_t` given values of `x_t-1`, `x_t-2`, `...`, `x_t-15`.

Usage

```
ar9
```

Format

A data frame with 1000 rows and 16 variables:

x_t-1,x_t-2,x_t-3,x_t-4,x_t-5,x_t-6,x_t-7,x_t-8,x_t-9,x_t-10,x_t-11,x_t-12,

x_t-13,x_t-14,x_t-15 lagged values of x_t in columns 1:15

x_t dependent variable in column 16

Details

This dataset was generated using the AR9 model first described in Sharma (2000) and given by:

$x_t = 0.3x_{t-1} - 0.6x_{t-4} - 0.5x_{t-9} + \epsilon_t$

where ϵ_t

References

Sharma, A. (2000), Seasonal to interannual rainfall probabilistic forecasts for improved water supply management: Part 1 - a strategy for system predictor identification, Journal of Hydrology, 239(1-4), 232-239, http://dx.doi.org/10.1016/S0022-1694(00)00346-2.

observed *Return observed target values.*

Description

Return observed target values used for fitting 'ann' or 'nnet' ANN models.

Usage

```
observed(object)
```

Arguments

object	an object of class 'ann' as returned by ann or of class 'nnet' as returned by nnet.

Details

This function can be invoked by calling observed(x) for an object x of class 'ann' or 'nnet'.

Value

a 1-column matrix of observed target values.

Examples

```
# Get observed values of y used to train ann object `fit'.
# ---
data("ar9")
samp <- sample(1:1000, 200)
y <- ar9[samp, ncol(ar9)]
x <- ar9[samp, -ncol(ar9)]
x <- x[, c(1,4,9)]
fit <- ann(x, y, size = 1, act_hid = "tanh", act_out = "linear", rang = 0.1)
y_obs <- observed(fit)
```

plot.validann *Plot ANN validation results.*

Description

Plot method for objects of class 'validann'. Produces a series of plots used for validating and assessing ANN models based on results returned by validann.

Usage

```
## S3 method for class 'validann'
plot(x, obs, sim, gof = TRUE, resid = TRUE, sa = TRUE,
  display = c("multi", "single"), profile = c("all", "median"), ...)
```

Arguments

x	object of class 'validann' as returned by validann. This is a list comprising metrics and statistics that can be used for validating ANN models.
obs, sim	vectors comprising observed (obs) and simulated (sim) examples of a single response variable used for computing x object.
gof	logical; should goodness-of-fit plots be produced? Default = TRUE.
resid	logical; should residual analysis plots be produced? Default = TRUE.
sa	logical; should input sensitivity analysis plots be produced? Default = TRUE.
display	character string defining how plots should be displayed. The default is "multi" where multiple plots are displayed together according to whether they are goodness-of-fit, residual analysis or sensitivity analysis plots. For "single", each plot is displayed on its own. If the session is interactive, the user will be asked to confirm a new page whether display is "single" or "multi".
profile	character string defining which structural validity Profile method outputs should be plotted. The default is "all" where outputs corresponding to 5 summary statistics are plotted together with the median predicted response for each input value. For "median", only the median response is plotted.
...	Arguments to be passed to plot (not currently used).

Details

This function can be invoked by calling plot(x, obs, sim) for an object x of class 'validann'.

To produce plots for all types of validation metrics and statistics, gof, resid and sa must be TRUE and corresponding results must have been successfully computed by validann and returned in object x.

If gof is TRUE, a scatter plot, Q-Q plot and time/sample plot of observed (obs) versus predicted (sim) data are produced.

If resid is TRUE and x$residuals is not NULL, plots of the model residuals are produced including histogram, Q-Q plot (standardized residuals compared to standard normal), autocorrelation (acf), partial autocorrelation (pacf), standardized residual versus predicted output (i.e. sim) and standardized residual versus time/order of the data.

If sa is TRUE and x$y_hat is not NULL, model response values resulting from the Profile sensitivity analysis are plotted against percentiles of each input. If x$rs is not NULL, the relative sensitivities of each input, as computed by the partial derivative (PaD) sensitivity analysis, are plotted against predicted output.

Setting gof, resid and/or sa to FALSE will 'turn off' the respective validation plots.

See Also

validann

Examples

```
## Build ANN model and compute replicative and structural validation results
data("ar9")
samp <- sample(1:1000, 200)
y <- ar9[samp, ncol(ar9)]
x <- ar9[samp, -ncol(ar9)]
x <- x[, c(1,4,9)]
fit <- ann(x, y, size = 1, act_hid = "tanh", act_out = "linear", rang = 0.1)
results <- validann(fit, x = x)
obs <- observed(fit)
sim <- fitted(fit)

## Plot replicative and structural validation results to the current device
## - a single page for each type of validation
plot(results, obs, sim)

## Plot results to the current device - a single page for each plot
plot(results, obs, sim, display = "single")

## Plot replicative and structural validation results to single file
pdf("RepStructValidationPlots.pdf")
plot(results, obs, sim)
dev.off()

## Get predictive validation results for above model based on a new sample
## of ar9 data.
samp <- sample(1:1000, 200)
y <- ar9[samp, ncol(ar9)]
x <- ar9[samp, -ncol(ar9)]
x <- x[, c(1,4,9)]
obs <- y
sim <- predict(fit, newdata = x)
results <- validann(fit, obs = obs, sim = sim, x = x)

## Plot predictive results only to file
pdf("PredValidationPlots.pdf")
plot(results, obs, sim, resid = FALSE, sa = FALSE)
dev.off()
```

`predict.ann` *Predict new examples using a trained neural network.*

Description

Predict new examples using a trained neural network.

Usage

```
## S3 method for class 'ann'
predict(object, newdata = NULL, derivs = FALSE, ...)
```

Arguments

object	an object of class 'ann' as returned by function ann.
newdata	matrix, data frame or vector of input data. A vector is considered to comprise examples of a single input or predictor variable. If x is NULL, fitted outputs derived from object will be returned.
derivs	logical; should derivatives of hidden and output nodes be returned? Default is FALSE.
...	additional arguments affecting the predictions produced (not currently used).

Details

This function is a method for the generic function predict() for class 'ann'. It can be invoked by calling predict(x) for an object x of class 'ann'.

predict.ann produces predicted values, obtained by evaluating the 'ann' model given newdata, which contains the inputs to be used for prediction. If newdata is omitted, the predictions are based on the data used for the fit.

Derivatives may be returned for sensitivity analyses, for example.

Value

if derivs = FALSE, a vector of predictions is returned.

Otherwise, a list with the following components is returned:

values	matrix of values returned by the trained ANN.
derivs	matrix of derivatives of hidden (columns 1:object$size) and output (final column) nodes.

See Also

ann

Examples

```
## fit 1-hidden node `ann' model to ar9 data
data("ar9")
samp <- sample(1:1000, 200)
y <- ar9[samp, ncol(ar9)]
x <- ar9[samp, -ncol(ar9)]
x <- x[, c(1,4,9)]

fit <- ann(x, y, size = 1, act_hid = "tanh", act_out = "linear", rang = 0.1)

## get model predictions based on a new sample of ar9 data.
samp <- sample(1:1000, 200)
y <- ar9[samp, ncol(ar9)]
x <- ar9[samp, -ncol(ar9)]
x <- x[, c(1,4,9)]
```

```
sim <- predict(fit, newdata = x)

## if derivatives are required...
tmp <- predict(fit, newdata = x, derivs = TRUE)
sim <- tmp$values
derivs <- tmp$derivs
```

---

validann *Validate Artificial Neural Networks.*

Description

Compute metrics and statistics for predictive, replicative and/or structural validation of artificial neural networks (ANNs).

Usage

```
validann(...)

## S3 method for class 'ann'
validann(net, obs = NULL, sim = NULL, x = NULL,
  na.rm = TRUE, ...)

## S3 method for class 'nnet'
validann(net, obs = NULL, sim = NULL, x = NULL,
  na.rm = TRUE, ...)

## Default S3 method:
validann(obs, sim, wts = NULL, nodes = NULL,
  na.rm = TRUE, ...)
```

Arguments

net	an object of class 'ann' (as returned by function ann) or 'nnet' (as returned using nnet). This is a list object comprising information about the fitted ANN model, including values of weights, fitted target values, number of layers and numbers of nodes in each layer, for example.
obs, sim	vectors comprising observed (obs) and simulated (sim) examples of a single response variable. These vectors are used to compute model fit statistics. Optional if net is supplied (see 'Details').
x	matrix, data frame or vector of input data used for fitting net object. A vector is considered to comprise examples of a single input or predictor variable. While x is optional, sensitivity analyses useful for structural validation cannot be performed if it is not supplied.
na.rm	logical; should missing values (including NaN) be removed from calculations? Default = TRUE.

wts	vector of ANN weights used to compute input 'relative importance' measures if net object is not supplied. Must be supplied together with nodes in order to compute such metrics. See 'Details' for ordering of wts vector.
nodes	vector indicating the number of nodes in each layer of the ANN model. This vector should have 3 elements: nodes in input layer, nodes in hidden layer (can be 0), and nodes in output layer. If net object is not supplied, nodes must be supplied together with wts if any structural validation metrics are to be computed.
...	arguments to be passed to different validann methods, see specific formulations for details.

Details

To compute all possible validation metrics and statistics, net must be supplied and must be of class 'ann' (as returned by ann) or 'nnet' (as returned by nnet). However, a partial derivative (PaD) sensitivity analysis (useful for structural validation) will only be carried out if net is of class 'ann'.

If obs and sim data are supplied in addition to net, validation metrics are computed based on these. Otherwise, metrics and statistics are computed based on obs and sim datasets derived from the net object (i.e. the data used to fit net and the fitted values). As such, both obs and sim must be supplied if validation is to be based either on data not used for training or on unprocessed training data (if training data were preprocessed). If either obs or sim is specified but the other isn't, both obs and sim will be derived from net if supplied (and a warning will be given). Similarly, this will occur if obs and sim are of different lengths.

If net is not supplied, both obs and sim are required. This may be necessary if validating an ANN model not built using either the nnet or ann functions. In this case, both wts and nodes are also required if any structural validation metrics are to be returned. If an ANN model has K input nodes, J hidden nodes and a single output O, with a bias node for both the hidden and output layers, the wts vector must be ordered as follows:

```
c(Wi1h1,Wi1h2,...Wi1hJ,Wi2h1,...Wi2hJ,...,WiKh1,...,WiKhJ,Wi0h1,...,Wi0hJ,
Wh1O,...,WhJO,Wh0O)
```

where Wikhj is the weight between the kth input and the jth hidden node and WhjO is the weight between the jth hidden node and the output. The bias weight on the jth hidden layer node is labelled Wi0hj while the bias weight on the output is labelled Wh0O. The wts vector assumes the network is fully connected; however, missing connections may be substituted by zero weights. Skip-layer connections are not allowed.

Value

list object of class 'validann' with components dependent on arguments passed to validann function:

metrics	a data frame consisting of metrics: AME, PDIFF, MAE, ME, RMSE, R4MS4E, AIC, BIC, NSC, RAE, PEP, MARE, MdAPE, MRE, MSRE, RVE, RSqr, IoAd, CE, PI, MSLE, MSDE, IRMSE, VE, KGE, SSE and R. See Dawson et al. (2007) for definitions.

`obs_stats` a data frame consisting of summary statistics about the `obs` dataset including mean, minimum, maximum, variance, standard deviation, skewness and kurtosis.

`sim_stats` a data frame consisting of summary statistics about the `sim` dataset including mean, minimum, maximum, variance, standard deviation, skewness and kurtosis.

`residuals` a 1-column matrix of model residuals (`sim - obs`).

`resid_stats` a data frame consisting of summary statistics about the model `residuals` including mean, minimum, maximum, variance, standard deviation, skewness and kurtosis.

`ri` a data frame consisting of 'relative importance' values for each input. Only returned if `net` or `wts` and `nodes` are supplied.

If `net` is supplied, relative importance values computed using the following 4 methods are returned:

Garson's (Garson); connection weight (CW); Profile sensitivity analysis (Profile); and partial derivative sensitivity analysis (PaD).

In addition, if `net` is of class 'ann' (as returned by function ann) and the activation function used at the hidden layer (`act_hid`) is "tanh", relative importance values computed using the modified CW (MCW) are also returned. This method requires that the hidden layer activation function be symmetric about the origin.

If `wts` and `nodes` are supplied, only relative importance values computed using the Garson and CW methods are returned.

See Gevrey et al. (2003), Olden et al. (2004) and Kingston et al. (2006) for details of the relative importance methods.

`y_hat` a matrix of dimension `c(101, ncol(x) * 6)` of model response values indicating the local sensitivity of the model to each input in `x`. Only returned if `net` and `x` are supplied.

The response values returned in `y_hat` are calculated using the 'Profile' sensitivity analysis method described in Gevrey et al. (2003). Using this method, the local sensitivity of each input in `x` is considered successively. For each input `x[,i]`, 5 synthetic data sets are generated where inputs `x[,-i]` are successively fixed at their minimum, 1st quartile, median, 3rd quartile and maximum values (as calculated from `x`), while input `x[,i]` is varied between its minimum and maximum value, increasing in increments of 1% (giving 101 synthetic values of `x[,i]` for each of the 5 sets of fixed `x[,-i]`). These data are input into `net` and model response values corresponding to the 5 summary statistics are computed. These 5 sets of response values, together with a set of computed median responses, are returned as y_hat[,(i - 1) * 6 + 1:6]. This process is repeated for each input variable in `x`. See Gevrey et al. (2003) for further details.

`as` a matrix of dimension `dim(x)` of 'absolute sensitivity' values for each input in `x` given the model output values (i.e. `sim`). Only returned if `net` and `x` are supplied and `net` is of class 'ann'.

The values in `as` are calculated according to the partial derivative (PaD) sensitivity analysis method described in Gevrey et al. (2003), which involves computing the first-order partial derivatives of the ANN output with respect to each input. `net` must be of class 'ann' in order to access partial derivatives of the hidden layer nodes as returned by ann.

`rs` a matrix of dimension `dim(x)` of 'relative sensitivity' values for each input in `x` given the model output values (i.e. `sim`). Only returned if `net` and `x` are supplied and `net` is of class 'ann'.

To compute the values in `rs`, the `as` values are normalised by multiplying by `x[,i]/sim` as described in Mount et al. (2013). As for `as`, `net` must be of class 'ann' in order to access partial derivatives of the hidden layer nodes as returned by `ann`.

Methods (by class)

- `ann`: Compute validation metrics when `net` is of class 'ann'.
- `nnet`: Compute validation metrics when `net` is of class 'nnet'.
- `default`: Useful for predictive validation only or when ANN model has not been developed using either `ann` or `nnet`. Limited structural validation metrics may be computed and only if `wts` and `nodes` are supplied.

References

Dawson, C.W., Abrahart, R.J., See, L.M., 2007. HydroTest: A web-based toolbox of evaluation metrics for the standardised assessment of hydrological forecasts. Environmental Modelling & Software, 22(7), 1034-1052. http://dx.doi.org/10.1016/j.envsoft.2006.06.008.

Olden, J.D., Joy, M.K., Death, R.G., 2004. An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data. Ecological Modelling 178, 389-397. http://dx.doi.org/10.1016/j.ecolmodel.2004.03.013.

Gevrey, M., Dimopoulos, I., Lek, S., 2003. Review and comparison of methods to study the contribution of variables in artificial neural network models. Ecological Modelling 160, 249-264. http://dx.doi.org/10.1016/S0304-3800(02)00257-0.

Kingston, G.B., Maier, H.R., Lambert, M.F., 2006. Forecasting cyanobacteria with Bayesian and deterministic artificial neural networks, in: IJCNN '06. International Joint Conference on Neural Networks, 2006., IEEE. pp. 4870-4877. http://dx.doi.org/10.1109/ijcnn.2006.247166.

Mount, N.J., Dawson, C.W., Abrahart, R.J., 2013. Legitimising data-driven models: exemplification of a new data-driven mechanistic modelling framework. Hydrology and Earth System Sciences 17, 2827-2843. http://dx.doi.org/10.5194/hess-17-2827-2013.

See Also

`ann`, `plot.validann`, `predict.ann`

Examples

```
# get validation results for 1-hidden node `ann' model fitted to ar9 data
# based on training data.
# ---
data("ar9")
samp <- sample(1:1000, 200)
y <- ar9[samp, ncol(ar9)]
x <- ar9[samp, -ncol(ar9)]
x <- x[, c(1,4,9)]
```

```
fit <- ann(x, y, size = 1, act_hid = "tanh", act_out = "linear", rang = 0.1)
results <- validann(fit, x = x)

# get validation results for above model based on a new sample of ar9 data.
# ---
samp <- sample(1:1000, 200)
y <- ar9[samp, ncol(ar9)]
x <- ar9[samp, -ncol(ar9)]
x <- x[, c(1,4,9)]

obs <- y
sim <- predict(fit, newdata = x)
results <- validann(fit, obs = obs, sim = sim, x = x)

# get validation results for `obs' and `sim' data without ANN model.
# In this example `sim' is generated using a linear model. No structural
# validation of the model is possible, but `wts' are provided to compute the
# number of model parameters needed for the calculation of certain
# goodness-of-fit metrics.
# ---
samp <- sample(1:1000, 200)
y <- ar9[samp, ncol(ar9)]
x <- ar9[samp, -ncol(ar9)]
x <- as.matrix(x[, c(1,4,9)])
lmfit <- lm.fit(x, y)
sim <- lmfit$fitted.values
obs <- y
results <- validann(obs = obs, sim = sim, wts = lmfit$coefficients)

# validann would be called in the same way if the ANN model used to generate
# `sim' was not available or was not of class `ann' or `nnet'. Ideally in
# this case, however, both `wts' and `nodes' should be supplied such that
# some structural validation metrics may be computed.
# ---
obs <- c(0.257, -0.891, -1.710, -0.575, -1.668, 0.851, -0.350, -1.313,
         -2.469, 0.486)
sim <- c(-1.463, 0.027, -2.053, -1.091, -1.602, 2.018, 0.723, -0.776,
         -2.351, 1.054)
wts <- c(-0.05217, 0.08363, 0.07840, -0.00753, -7.35675, -0.00066)
nodes <- c(3, 1, 1)
results <- validann(obs = obs, sim = sim, wts = wts, nodes = nodes)
```

Index