Package 'nnfor'

January 16, 2019

Type Package

Title Time Series Forecasting with Neural Networks

Version 0.9.6

Description Automatic time series modelling with neural networks.
Allows fully automatic, semi-manual or fully manual specification of networks. For details of the specification methodol-
ogy see: (i) Crone and Kourentzes (2010) <doi:10.1016/j.neucom.2010.01.017>;
and (ii) Kourentzes et al. (2014) <doi:10.1016/j.eswa.2013.12.011>.

License GPL-3

Encoding UTF-8

LazyData true

Depends forecast

Imports glmnet, neuralnet, plotrix, MASS, tsutils, uroot

Suggests thief

URL http://kourentzes.com/forecasting/2019/01/16/tutorial-for-the-nnfor-r-package/

BugReports https://github.com/trnnick/nnfor/issues

RoxygenNote 6.1.1

NeedsCompilation no

Author Nikolaos Kourentzes [aut, cre]

Maintainer Nikolaos Kourentzes <nikolaos@kourentzes.com>

Repository CRAN

Date/Publication 2019-01-16 14:20:03 UTC

R topics documented:

`elm` *Extreme learning machines for time series forecasting*

Description

This function fits ELM neural networks for time series forecasting.

Usage

```
elm(y, m = frequency(y), hd = NULL, type = c("lasso", "ridge",
  "step", "lm"), reps = 20, comb = c("median", "mean", "mode"),
  lags = NULL, keep = NULL, difforder = NULL, outplot = c(FALSE,
  TRUE), sel.lag = c(TRUE, FALSE), direct = c(FALSE, TRUE),
  allow.det.season = c(TRUE, FALSE), det.type = c("auto", "bin",
  "trg"), xreg = NULL, xreg.lags = NULL, xreg.keep = NULL,
  barebone = c(FALSE, TRUE), model = NULL, retrain = c(FALSE, TRUE))
```

Arguments

`y`	Input time series. Can be ts or msts object.
`m`	Frequency of the time series. By default it is picked up from y.
`hd`	Number of hidden nodes. This can be a vector, where each number represents the number of hidden nodes of a different hidden layer. Use NULL to automatically specify.
`type`	Estimation type for output layer weights. Can be "lasso" (lasso with CV), "ridge" (ridge regression with CV), "step" (stepwise regression with AIC) or "lm" (linear regression).
`reps`	Number of networks to train, the result is the ensemble forecast.
`comb`	Combination operator for forecasts when reps > 1. Can be "median", "mode" (based on KDE estimation) and "mean".
`lags`	Lags of y to use as inputs. If none provided then 1:frequency(y) is used. Use 0 for no univariate lags.
`keep`	Logical vector to force lags to stay in the model if sel.lag == TRUE. If NULL then it keep = rep(FALSE,length(lags)).

`difforder`	Vector including the differencing lags. For example c(1,12) will apply first and seasonal (12) differences. For no differencing use 0. For automatic selection use NULL.
`outplot`	Provide plot of model fit. Can be TRUE or FALSE.
`sel.lag`	Automatically select lags. Can be TRUE or FALSE.
`direct`	Use direct input-output connections to model strictly linear effects. Can be TRUE or FALSE.
`allow.det.season`	Permit modelling seasonality with deterministic dummies.
`det.type`	Type of deterministic seasonality dummies to use. This can be "bin" for binary or "trg" for a sine-cosine pair. With "auto" if ony a single seasonality is used and periodicity is up to 12 then "bin" is used, otherwise "trg".
`xreg`	Exogenous regressors. Each column is a different regressor and the sample size must be at least as long as the target in-sample set, but can be longer.
`xreg.lags`	This is a list containing the lags for each exogenous variable. Each list is a numeric vector containing lags. If xreg has 3 columns then the xreg.lags list must contain three elements. If NULL then it is automatically specified.
`xreg.keep`	List of logical vectors to force lags of xreg to stay in the model if sel.lag == TRUE. If NULL then all exogenous lags can be removed.
`barebone`	Use an alternative elm implementation (written in R) that is faster when the number of inputs is very high. Typically not needed.
`model`	A previously trained mlp object. If this is provided then the same model is fitted to y, without re-estimating any model parameters.
`retrain`	If a previous model is provided, retrain the network or not. If the network is retrained the size of the hidden layer is reset.

Value

Return object of class `elm`. If barebone == TRUE then the object inherits a second class "`elm.fast`". The function `plot` produces a plot the network architecture. `elm` contains:

- `net` - ELM networks. If it is of class "`elm.fast`" then this is NULL.
- `hd` - Number of hidden nodes. If it is of class "`elm.fast`" this is a vector with a different number for each repetition.
- `W.in` - NULL unless it is of class "`elm.fast`". Contains the input weights.
- `W` - Output layer weights for each repetition.
- `b` - Contains the output node bias for each training repetition.
- `W.dct` - Contains the direct connection weights if argument direct == TRUE. Otherwise is NULL.
- `lags` - Input lags used.
- `xreg.lags` - `xreg` lags used.
- `difforder` - Differencing used.
- `sdummy` - Use of deterministic seasonality.

- `ff` - Seasonal frequencies detected in data (taken from ts or msts object).
- `ff.det` - Seasonal frequencies coded using deterministic dummies.
- `det.type` - Type of determistic seasonality.
- `y` - Input time series.
- `minmax` - Scaling structure.
- `xreg.minmax` - Scaling structure for xreg variables.
- `comb` - Combination operator used.
- `type` - Estimation used for output layer weights.
- `direct` - Presence of direct input-output connections.
- `fitted` - Fitted values.
- `MSE` - In-sample Mean Squared Error.

Note

To use elm with Temporal Hierarchies (thief package) see `elm.thief`. The elm function by default calls the `neuralnet` function. If barebone == TRUE then it uses an alternative implementation (`TStools:::elm.fast`) which is more appropriate when the number of inputs is several hundreds.

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

References

- For an introduction to neural networks see: Ord K., Fildes R., Kourentzes N. (2017) Principles of Business Forecasting 2e. *Wessex Press Publishing Co.*, Chapter 10.
- For combination operators see: Kourentzes N., Barrow B.K., Crone S.F. (2014) Neural network ensemble operators for time series forecasting. *Expert Systems with Applications*, **41**(**9**), 4235-4244.
- For variable selection see: Crone S.F., Kourentzes N. (2010) Feature selection for time series prediction – A combined filter and wrapper approach for neural networks. *Neurocomputing*, **73**(**10**), 1923-1936.
- For ELMs see: Huang G.B., Zhou H., Ding X. (2006) Extreme learning machine: theory and applications. *Neurocomputing*, **70**(**1**), 489-501.

See Also

`forecast.elm`, `elm.thief`, `mlp`.

Examples

```
## Not run:
 fit <- elm(AirPassengers)
 print(fit)
 plot(fit)
```

```
frc <- forecast(fit,h=36)
plot(frc)

## End(Not run)
```

---

elm.fast *ELM (fast) neural network.*

Description

Fit ELM (fast) neural network. This is an ELM implementation that does not rely on neuralnets package.

Usage

```
elm.fast(y, x, hd = NULL, type = c("lasso", "ridge", "step", "ls"),
  reps = 20, comb = c("median", "mean", "mode"), direct = c(FALSE,
  TRUE), linscale = c(TRUE, FALSE), output = c("linear", "logistic"),
  core = c("FALSE", "TRUE"), ortho = c(FALSE, TRUE))
```

Arguments

y	Target variable.
x	Explanatory variables. Each column is a variable.
hd	Starting number of hidden nodes (scalar). Use NULL to automatically specify.
type	Estimation type for output layer weights. Can be "lasso" (lasso with CV), "ridge" (ridge regression with CV), "step" (stepwise regression with AIC) or "lm" (linear regression).
reps	Number of networks to train.
comb	Combination operator for forecasts when reps > 1. Can be "median", "mode" (based on KDE estimation) and "mean".
direct	Use direct input-output connections to model strictly linear effects. Can be TRUE or FALSE.
linscale	Scale inputs linearly between -0.8 to 0.8. If output == "logistic" then scaling is between 0 and 1.
output	Type of output layer. It can be "linear" or "logistic". If "logistic" then type must be set to "lasso".
core	If TRUE skips calculation of final fitted values and MSE. Called internally by "elm" function.
ortho	If TRUE then the initial weights between the input and hidden layers are orthogonal (only when number of input variable <= sample size).

Value

An object of class "elm.fast". The function plot produces a plot the network fit. An object of class "elm.fast" is a list containing the following elements:

- hd - Number of hidden nodes. This is a vector with a different number for each training repetition.
- W.in - Input weights for each training repetition.
- W - Output layer weights for each repetition.
- b - Output node bias for each training repetition.
- W.dct - Direct connection weights argument if direct == TRUE for each training repetition. Otherwuse NULL.
- fitted.all - Fitted values for each training repetition.
- fitted - Ensemble fitted values.
- y - Target variable.
- type - Estimation used for output layer weights.
- comb - Combination operator used.
- direct - Presence of direct input-output connections.
- minmax - If scaling is used this contains the scaling information for the target variable.
- minmax.x - If scaling is used this contains the scaling information for the input variables.
- MSE - In-sample Mean Squared Error.

Note

This implementation of ELM is more appropriate when the number of inputs is several hundreds. For time series modelling use elm instead.

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

References

- For combination operators see: Kourentzes N., Barrow B.K., Crone S.F. (2014) Neural network ensemble operators for time series forecasting. *Expert Systems with Applications*, **41**(**9**), 4235-4244.
- For ELMs see: Huang G.B., Zhou H., Ding X. (2006) Extreme learning machine: theory and applications. *Neurocomputing*, **70**(**1**), 489-501.

See Also

elm.

Examples

```
## Not run:
 p <- 2000
 n <- 150
 X <- matrix(rnorm(p*n),nrow=n)
 b <- cbind(rnorm(p))
 Y <- X %*% b
 fit <- elm.fast(Y,X)
 print(fit)

## End(Not run)
```

`elm.thief` *ELM network for THieF.*

Description

Function for ELM forecasting with Temporal Hierarchies.

Usage

```
elm.thief(y, h = NULL, ...)
```

Arguments

`y`	Input time series. Can be ts or msts object.
`h`	Forecast horizon. If NULL then h is set to match frequency of time series.
`...`	Additional arguments passed to `elm`.

Value

An object of classes "`forecast.net`" and "`forecast`". The function `plot` produces a plot of the forecasts. An object of class "`forecast.net`" is a list containing the following elements:

- `method` - The name of the forecasting method as a character string
- `mean` - Point forecasts as a time series
- `all.mean` - An array h x reps of all ensemble members forecasts, where reps are the number of ensemble members.
- `x` - The original time series (either `fit` used to create the network.
- `fitted` - Fitted values. Any values not fitted for the initial period of the time series are imputted with NA.
- `residuals` - Residuals from the fitted network.

Note

This function is created to work with Temporal Hierarchied (thief package). For conventional ELM networks use `elm`.

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

References

- For forecasting with temporal hierarchies see: Athanasopoulos G., Hyndman R.J., Kourentzes N., Petropoulos F. (2017) Forecasting with Temporal Hierarchies. *European Journal of Operational research*, **262**(**1**), 60-74.
- For combination operators see: Kourentzes N., Barrow B.K., Crone S.F. (2014) Neural network ensemble operators for time series forecasting. *Expert Systems with Applications*, **41**(**9**), 4235-4244.

See Also

`elm`, `mlp.thief`.

Examples

```
## Not run:
  library(thief)
  frc <- thief(AirPassengers,forecastfunction=elm.thief)
  plot(frc)

## End(Not run)
```

`forecast.elm` *Forecast using ELM neural network.*

Description

Create forecasts using ELM neural networks.

Usage

```
## S3 method for class 'elm'
forecast(object, h = NULL, y = NULL, xreg = NULL, ...)
```

Arguments

object	ELM network object, produced using elm.
h	Forecast horizon. If NULL then h is set to match frequency of time series.
y	Optionally forecast using different data than what the network was trained on. Expected to create havoc and do really bad things!
xreg	Exogenous regressors. Each column is a different regressor and the sample size must be at least as long as the target in-sample set plus the forecast horizon, but can be longer. Set it to NULL if no xreg inputs are used.
...	Unused argument.

Value

An object of classes "forecast.net" and "forecast". The function plot produces a plot of the forecasts. An object of class "forecast.net" is a list containing the following elements:

- method - The name of the forecasting method as a character string
- mean - Point forecasts as a time series
- all.mean - An array h x reps of all ensemble members forecasts, where reps are the number of ensemble members.
- x - The original time series used to create the network.
- fitted - Fitted values.
- residuals - Residuals from the fitted network.

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

See Also

elm, elm.thief, mlp.

Examples

```
## Not run:
 fit <- elm(AirPassengers)
 plot(fit)
 frc <- forecast(fit,h=36)
 plot(frc)

## End(Not run)
```

`forecast.mlp` *Forecast using MLP neural network.*

Description

Create forecasts using MLP neural networks.

Usage

```
## S3 method for class 'mlp'
forecast(object, h = NULL, y = NULL, xreg = NULL, ...)
```

Arguments

`object`	MLP network object, produced using `mlp`.
`h`	Forecast horizon. If NULL then h is set to match frequency of time series.
`y`	Optionally forecast using different data than what the network was trained on. Expected to create havoc and do really bad things!
`xreg`	Exogenous regressors. Each column is a different regressor and the sample size must be at least as long as the target in-sample set plus the forecast horizon, but can be longer. Set it to NULL if no xreg inputs are used.
`...`	Unused argument.

Value

An object of classes "`forecast.net`" and "`forecast`". The function `plot` produces a plot of the forecasts. An object of class "`forecast.net`" is a list containing the following elements:

- `method` - The name of the forecasting method as a character string
- `mean` - Point forecasts as a time series
- `all.mean` - An array h x reps of all ensemble members forecasts, where reps are the number of ensemble members.
- `x` - The original time series used to create the network.
- `fitted` - Fitted values.
- `residuals` - Residuals from the fitted network.

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

See Also

`mlp`, `mlp.thief`, `elm`.

Examples

```
## Not run:
 fit <- mlp(AirPassengers)
 plot(fit)
 frc <- forecast(fit,h=36)
 plot(frc)

## End(Not run)
```

`linscale` *Apply minmax linear scaling to a vector.*

Description

Apply minmax linear scaling to a vector.

Usage

```
linscale(x, minmax = NULL, rev = c(FALSE, TRUE))
```

Arguments

`x`	Input vector.
`minmax`	minmax must be a list with elements "mn", "mx", "mn.orig" and "mx.orig", where "mn" and "mx" refer to the target min and max, and the remaining two refer to the current vector min and max. By default mn=-1 and mx=1. mn.orig and mx.orig can be missing, unless the scaling is reversed.
`rev`	Reverse scaling back to original: TRUE or FALSE.

Value

Outputs a list with elements:

- `x` - Scaled vector.
- `minmax` - List with resulting mn, mx, mn.orig and mx.orig. Can be used as input to reverse scaling.

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

Examples

```
y <- rnorm(20)*100
sc <- linscale(y)
x <- sc$x
print(c(min(y),max(y)))
print(c(min(x),max(x)))
sc.rev <- linscale(x,minmax=sc$minmax,rev=TRUE)
print(c(min(sc.rev$x),max(sc.rev$x)))
```

---

mlp *Multilayer Perceptron for time series forecasting*

Description

This function fits MLP neural networks for time series forecasting.

Usage

```
mlp(y, m = frequency(y), hd = NULL, reps = 20, comb = c("median",
  "mean", "mode"), lags = NULL, keep = NULL, difforder = NULL,
  outplot = c(FALSE, TRUE), sel.lag = c(TRUE, FALSE),
  allow.det.season = c(TRUE, FALSE), det.type = c("auto", "bin",
  "trg"), xreg = NULL, xreg.lags = NULL, xreg.keep = NULL,
  hd.auto.type = c("set", "valid", "cv", "elm"), hd.max = NULL,
  model = NULL, retrain = c(FALSE, TRUE), ...)
```

Arguments

y	Input time series. Can be ts or msts object.
m	Frequency of the time series. By default it is picked up from y.
hd	Number of hidden nodes. This can be a vector, where each number represents the number of hidden nodes of a different hidden layer.
reps	Number of networks to train, the result is the ensemble forecast.
comb	Combination operator for forecasts when reps > 1. Can be "median", "mode" (based on KDE estimation) and "mean".
lags	Lags of y to use as inputs. If none provided then 1:frequency(y) is used. Use 0 for no univariate lags.
keep	Logical vector to force lags to stay in the model if sel.lag == TRUE. If NULL then it keep = rep(FALSE,length(lags)).
difforder	Vector including the differencing lags. For example c(1,12) will apply first and seasonal (12) differences. For no differencing use 0. For automatic selection use NULL.
outplot	Provide plot of model fit. Can be TRUE or FALSE.

sel.lag	Automatically select lags. Can be TRUE or FALSE.
allow.det.season	Permit modelling seasonality with deterministic dummies.
det.type	Type of deterministic seasonality dummies to use. This can be "bin" for binary or "trg" for a sine-cosine pair. With "auto" if ony a single seasonality is used and periodicity is up to 12 then "bin" is used, otherwise "trg".
xreg	Exogenous regressors. Each column is a different regressor and the sample size must be at least as long as the target in-sample set, but can be longer.
xreg.lags	This is a list containing the lags for each exogenous variable. Each list is a numeric vector containing lags. If xreg has 3 columns then the xreg.lags list must contain three elements. If NULL then it is automatically specified.
xreg.keep	List of logical vectors to force lags of xreg to stay in the model if sel.lag == TRUE. If NULL then all exogenous lags can be removed. The syntax for multiple xreg is the same as for xreg.lags.
hd.auto.type	Used only if hd==NULL. "set" fixes hd=5. "valid" uses a 20% validation set (randomly) sampled to find the best number of hidden nodes. "cv" uses 5-fold cross-validation. "elm" uses ELM to estimate the number of hidden nodes (experimental).
hd.max	When hd.auto.type is set to either "valid" or "cv" then this argument can be used to set the maximum number of hidden nodes to evaluate, otherwise the maximum is set automatically.
model	A previously trained mlp object. If this is provided then the same model is fitted to y, without re-estimating any model parameters.
retrain	If a previous model is provided, retrain the network or not.
...	Additional inputs for neuralnet function.

Value

Return object of class mlp. The function plot produces a plot the network architecture. mlp contains:

- net - MLP networks.
- hd - Number of hidden nodes.
- lags - Input lags used.
- xreg.lags - xreg lags used.
- difforder - Differencing used.
- sdummy - Use of deterministic seasonality.
- ff - Seasonal frequencies detected in data (taken from ts or msts object).
- ff.det - Seasonal frequencies coded using deterministic dummies.
- det.type - Type of determistic seasonality.
- y - Input time series.
- minmax - Scaling structure.
- xreg.minmax - Scaling structure for xreg variables.

- `comb` - Combination operator used.
- `fitted` - Fitted values.
- `MSE` - In-sample Mean Squared Error.
- `MSEH` - If `hd.auto.type` is set to either "valid" or "cv" an array of the MSE error for each network size is provided. Otherwise this is NULL.

Note

To use mlp with Temporal Hierarchies (thief package) see `mlp.thief`.

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

References

- For an introduction to neural networks see: Ord K., Fildes R., Kourentzes N. (2017) Principles of Business Forecasting 2e. *Wessex Press Publishing Co.*, Chapter 10.
- For combination operators see: Kourentzes N., Barrow B.K., Crone S.F. (2014) Neural network ensemble operators for time series forecasting. *Expert Systems with Applications*, **41**(**9**), 4235-4244.
- For variable selection see: Crone S.F., Kourentzes N. (2010) Feature selection for time series prediction – A combined filter and wrapper approach for neural networks. *Neurocomputing*, **73**(**10**), 1923-1936.

See Also

`forecast.mlp`, `mlp.thief`, `elm`.

Examples

```
## Not run:
 fit <- mlp(AirPassengers)
 print(fit)
 plot(fit)
 frc <- forecast(fit,h=36)
 plot(frc)

## End(Not run)
```

`mlp.thief` *MLP network for THieF.*

Description

Function for MLP forecasting with Temporal Hierarchies.

Usage

```
mlp.thief(y, h = NULL, ...)
```

Arguments

`y`	Input time series. Can be ts or msts object.
`h`	Forecast horizon. If NULL then h is set to match frequency of time series.
`...`	Additional arguments passed to `mlp`.

Value

An object of classes "`forecast.net`" and "`forecast`". The function `plot` produces a plot of the forecasts. An object of class "`forecast.net`" is a list containing the following elements:

- `method` - The name of the forecasting method as a character string
- `mean` - Point forecasts as a time series
- `all.mean` - An array h x reps of all ensemble members forecasts, where reps are the number of ensemble members.
- `x` - The original time series (either `fit` used to create the network.
- `fitted` - Fitted values. Any values not fitted for the initial period of the time series are imputted with NA.
- `residuals` - Residuals from the fitted network.

Note

This function is created to work with Temporal Hierarchied (thief package). For conventional MLP networks use `mlp`.

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

References

- For forecasting with temporal hierarchies see: Athanasopoulos G., Hyndman R.J., Kourentzes N., Petropoulos F. (2017) Forecasting with Temporal Hierarchies. *European Journal of Operational research*, **262**(**1**), 60-74.
- For combination operators see: Kourentzes N., Barrow B.K., Crone S.F. (2014) Neural network ensemble operators for time series forecasting. *Expert Systems with Applications*, **41**(**9**), 4235-4244.

See Also

mlp, elm.thief.

Examples

```
## Not run:
  library(thief)
  frc <- thief(AirPassengers,forecastfunction=mlp.thief)
  plot(frc)

## End(Not run)
```

---

nnfor *nnfor:Time Series Forecasting with Neural Networks*

Description

The **nnfor** package provides automatic time series modelling with neural networks. It facilitates fully automatic, semi-manual or fully manual specification of networks, using multilayer perceptrons (mlp) and extreme learning machines (elm).

Note

You can find a tutorial how to use the package here.

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

References

- For an introduction to neural networks see: Ord K., Fildes R., Kourentzes N. (2017) Principles of Business Forecasting 2e. *Wessex Press Publishing Co.*, Chapter 10.
- For ensemble combination operators see: Kourentzes N., Barrow B.K., Crone S.F. (2014) Neural network ensemble operators for time series forecasting. *Expert Systems with Applications*, **41**(**9**), 4235-4244.
- For variable selection see: Crone S.F., Kourentzes N. (2010) Feature selection for time series prediction – A combined filter and wrapper approach for neural networks. *Neurocomputing*, **73**(**10**), 1923-1936.

plot.elm *Plot ELM network.*

Description

Produces a plot of the ELM network architecture.

Usage

```
## S3 method for class 'elm'
plot(x, r = 1, ...)
```

Arguments

x	ELM network object, produced using elm.
r	Ensemple member to plot.
...	Unused argument.

Value

None. Function produces a plot.

Note

Neurons are coloured with "lightgrey". Seasonal dummies are coloured with "lightpink" and xreg with "lightblue".

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

See Also

elm, mlp.

Examples

```
## Not run:
 fit <- elm(AirPassengers)
 print(fit)
 plot(fit)
 frc <- forecast(fit,h=36)
 plot(frc)

## End(Not run)
```

plot.mlp *Plot MLP network.*

Description

Produces a plot of the MLP network architecture.

Usage

```
## S3 method for class 'mlp'
plot(x, r = 1, ...)
```

Arguments

x	MLP network object, produced using mlp.
r	Ensemple member to plot.
...	Unused argument.

Value

None. Function produces a plot.

Note

Neurons are coloured with "lightgrey". Seasonal dummies are coloured with "lightpink" and xreg with "lightblue".

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

See Also

elm, mlp.

Examples

```
## Not run:
 fit <- mlp(AirPassengers)
 print(fit)
 plot(fit)
 frc <- forecast(fit,h=36)
 plot(frc)

## End(Not run)
```

predict.elm.fast *Predictions for ELM (fast) network.*

Description

Calculate predictions for ELM (fast) network.

Usage

```
## S3 method for class 'elm.fast'
predict(object, newx, na.rm = c(FALSE, TRUE), ...)
```

Arguments

object	ELM network object, produced using elm.fast.
newx	Explanatory variables. Each column is a variable.
na.rm	If TRUE remove columns and object produces an ensemble forecast, then remove any members that give NA in their forecasts.
...	Unused argument.

Value

Returns a list with:

- Y.hat - Ensemble prediction.
- Y.all - Predictions of each training repetition.

Author(s)

Nikolaos Kourentzes, <nikolaos@kourentzes.com>

See Also

elm.fast.

Examples

```
## Not run:
 p <- 2000
 n <- 150
 X <- matrix(rnorm(p*n),nrow=n)
 b <- cbind(rnorm(p))
 Y <- X %*% b
 fit <- elm.fast(Y,X)
 predict(fit,X)

## End(Not run)
```

Index