Package 'ahnr'

June 18, 2018

Type Package

Title An Implementation of the Artificial Hydrocarbon Networks

Version 0.3.1

Description Implementation of the Artificial Hydrocarbon Networks for data modeling.

Depends R (>= 3.3.0)

License GPL-3 | file LICENSE

Encoding UTF-8

LazyData true

Suggests knitr, rmarkdown

URL https://github.com/jroberayalas/ahnr

BugReports https://github.com/jroberayalas/ahnr/issues

VignetteBuilder knitr

Imports matrixcalc, pracma, purrr, pdist, ggplot2, visNetwork, magrittr

RoxygenNote 6.0.1

NeedsCompilation no

Author Jose Roberto Ayala Solares [aut, cre]

Maintainer Jose Roberto Ayala Solares <ichbinjras@gmail.com>

Repository CRAN

Date/Publication 2018-06-18 21:57:16 UTC

R topics documented:

`fit` *fit*

Description

Function to train an Artificial Hydrocarbon Network (AHN).

Usage

```
fit(Sigma, n, eta, maxIter = 2000)
```

Arguments

`Sigma`	a list with two data frames. One for the inputs X, and one for the outputs Y.
`n`	number of particles to use.
`eta`	learning rate of the algorithm. Default is `0.01`.
`maxIter`	maximum number of iterations.

Value

an object of class "`ahn`" with the following components:

- network: structure of the AHN trained.
- Yo: original output variable.
- Ym: predicted output variable.
- eta: learning rate.
- minOverallError: minimum error achieved.
- variableNames: names of the input variables.

Examples

```
# Create data
x <- 2 * runif(1000) - 1;
x <- sort(x)

y <- (x < 0.1) * (0.05 * runif(100) + atan(pi*x)) +
    (x >= 0.1 & x < 0.6) * (0.05 * runif(1000) + sin(pi*x)) +
    (x >= 0.6) * (0.05 * runif(1000) + cos(pi*x))

# Create Sigma list
Sigma <- list(X = data.frame(x = x), Y = data.frame(y = y))

# Train AHN
ahn <- fit(Sigma, 5, 0.01, 500)
```

is.ahn *Checks if argument is a* ahn *object*

Description

Checks if argument is a ahn object

Usage

```
is.ahn(x)
```

Arguments

x	An R object

predict.ahn *predict*

Description

Function to simulate a trained Artificial Hydrocarbon Network.

Usage

```
## S3 method for class 'ahn'
predict(object, ...)
```

Arguments

object	an object of class "ahn" produced from the fit function.
...	further arguments passed to or from other methods.

Value

predicted output values for inputs in newdata.

Examples

```
## Not run:
# Create data
x <- 2 * runif(1000) - 1;
x <- sort(x)

y <- (x < 0.1) * (0.05 * runif(100) + atan(pi*x)) +
    (x >= 0.1 & x < 0.6) * (0.05 * runif(1000) + sin(pi*x)) +
    (x >= 0.6) * (0.05 * runif(1000) + cos(pi*x))
```

```
# Create Sigma list
Sigma <- list(X = data.frame(x = x), Y = data.frame(y = y))

# Train AHN
ahn <- fit(Sigma, 5, 0.01, 500)

# Test AHN
X <- data.frame(x = x)
ysim <- predict(ahn, X)

## End(Not run)
```

---

summary.ahn *Summary Artificial Hydrocarbon Network*

Description

Summary method for objects of class ahn.

Usage

```
## S3 method for class 'ahn'
summary(object, ...)
```

Arguments

object	an object of class "ahn" produced from the fit function.
...	further arguments passed to or from other methods.

Value

summary description of the AHN.

Examples

```
## Not run:
# Create data
x <- 2 * runif(1000) - 1;
x <- sort(x)

y <- (x < 0.1) * (0.05 * runif(100) + atan(pi*x)) +
    (x >= 0.1 & x < 0.6) * (0.05 * runif(1000) + sin(pi*x)) +
    (x >= 0.6) * (0.05 * runif(1000) + cos(pi*x))

# Create Sigma list
Sigma <- list(X = data.frame(x = x), Y = data.frame(y = y))
```

```
# Train AHN
ahn <- fit(Sigma, 5, 0.01, 500)

# Summary AHN
summary(ahn)

## End(Not run)
```

`visualize` *Visualize Artificial Hydrocarbon Network*

Description

Visualize method for objects of class ahn.

Usage

```
visualize(x, ...)
```

Arguments

x	an object of class "ahn" produced from the fit function.
...	further arguments passed to visNetwork functions.

Value

dynamic visualization of the AHN.

Examples

```
## Not run:
# Create data
x <- 2 * runif(1000) - 1;
x <- sort(x)

y <- (x < 0.1) * (0.05 * runif(100) + atan(pi*x)) +
    (x >= 0.1 & x < 0.6) * (0.05 * runif(1000) + sin(pi*x)) +
    (x >= 0.6) * (0.05 * runif(1000) + cos(pi*x))

# Create Sigma list
Sigma <- list(X = data.frame(x = x), Y = data.frame(y = y))

# Train AHN
ahn <- fit(Sigma, 5, 0.01, 500)

# Visualize AHN
visualize(ahn)

## End(Not run)
```

Index