Package 'yap'

December 17, 2019

Title Yet Another Probabilistic oNeural Network

Version 0.1.0

Author WenSui Liu

Maintainer WenSui Liu <liuwensui@gmail.com>

Description Another implementation of Probabilistic Neural Network in R based on Specht (1990) <DOI:10.1016/0893-6080(90)90049-Q>. It is applicable to the pattern recognition with a N-level response, where N > 2.

URL https://github.com/statcompute/yap

Depends R (>= 3.6.0)

Imports stats, randtoolbox, lhs, MLmetrics, parallel

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

R topics documented:

dummies *Convert a N-category vector to a N-dimension matrix*

Description

The function dummies converts a N-category vector to a N-dimension matrix

Usage

```
dummies(x)
```

Arguments

x	A N-category vector

Value

A N-dimension matrix with 0/1 values

Examples

```
dummies(x)
```

folds *Generate a list of index for the n-fold cross-validation*

Description

The function folds generates a list of index for the n-fold cross-validation

Usage

```
folds(idx, n, seed = 1)
```

Arguments

idx	A vector of index list
n	The number of n folds
seed	The seed value to generate random n-fold index

Value

A list of n-fold index

Examples

```
folds(seq(10), 3, 2020)
```

`gen_latin` *Generate random numbers of latin hypercube sampling*

Description

The function `gen_latin` generates a vector of random numbers by latin hypercube sampling

Usage

```
gen_latin(min = 0, max = 1, n, seed = 1)
```

Arguments

`min`	The minimum value of random numbers
`max`	The maxinum value of random numbers
`n`	The number of random numbers to gernate
`seed`	The seed value of random number generation

Value

A vector of random numbers bounded by the min and max

Examples

```
gen_latin(0, 1, 10, 2020)
```

`gen_sobol` *Generate sobol sequence*

Description

The function `gen_sobol` generates a vector of scrambled sobol sequence

Usage

```
gen_sobol(min = 0, max = 1, n, seed = 1)
```

Arguments

`min`	The minimum value of random numbers
`max`	The maxinum value of random numbers
`n`	The number of random numbers to gernate
`seed`	The seed value of random number generation

Value

A vector of sobol sequence bounded by the min and max

Examples

```
gen_sobol(0, 1, 10, 2020)
```

`gen_unifm` *Generate Uniform random numbers*

Description

The function `gen_unifm` generates a vector of uniform random numbers

Usage

```
gen_unifm(min = 0, max = 1, n, seed = 1)
```

Arguments

`min`	The minimum value of random numbers
`max`	The maxinum value of random numbers
`n`	The number of random numbers to gernate
`seed`	The seed value of random number generation

Value

A vector of uniform random numbers bounded by the min and max

Examples

```
gen_unifm(0, 1, 10, 2020)
```

pnn.fit *Create a probabilistic neural network*

Description

The function pnn.fit creates a Probabilistic Neural Network (PNN)

Usage

```
pnn.fit(x, y, sigma = 1)
```

Arguments

x	A matrix of predictors
y	A vector of N-category factors
sigma	A scalar with the positive value

Value

A PNN object

References

Donald Specht. (1990). Probabilistic Neural Networks.

Examples

```
pnn.fit(x, y)
```

pnn.optmiz_logl *Optimize the optimal value of PNN smoothing parameter based on Log Loss*

Description

The function pnn.optmiz_logl optimize the optimal value of PNN smoothing parameter by cross-validation.

Usage

```
pnn.optmiz_logl(net, lower = 0, upper, nfolds = 4, seed = 1, method = 1)
```

Arguments

net	A PNN object generated by pnn.fit()
lower	A scalar for the lower bound of the smoothing parameter
upper	A scalar for the upper bound of the smoothing parameter
nfolds	A scalar for the number of n-fold, 4 by default
seed	The seed value for the n-fold cross-validation, 1 by default
method	A scalar referring to the optimization method, 1 for Golden section searc and 2 for Brent's method

Value

The best outcome

See Also

pnn.search_logl

Examples

```
pnn.optmiz_logl(net = pnn, upper = 1)
```

---

pnn.parpred *Calculate predicted probabilities of PNN by using parallelism*

Description

The function pnn.parpred calculates a matrix of PNN predicted probabilities based on an input matrix

Usage

```
pnn.parpred(net, x)
```

Arguments

net	The PNN object generated by pnn.fit()
x	The matrix of input predictors

Value

A matrix of predicted probabilities

See Also

pnn.predict

Examples

```
pnn.parpred(pnn, x)
```

`pnn.predict` *Calculate a matrix of predicted probabilities*

Description

The function `pnn.predict` calculates a matrix of predicted probabilities based on a matrix of predictors

Usage

```
pnn.predict(net, x)
```

Arguments

`net`	The PNN object generated by pnn.fit()
`x`	The matrix of input predictors

Value

A matrix of predicted probabilities for all categories

See Also

`pnn.predone`

Examples

```
pnn.predict(pnn, x)
```

pnn.predone *Calculate the predicted probability for each category of PNN*

Description

The function pnn.predone calculates the predicted probability for each category of PNN

Usage

```
pnn.predone(net, x)
```

Arguments

net	A PNN object created by pnn.fit()
x	A vector of input predictors

Value

A one-row matrix of predicted probabilities

See Also

pnn.fit

Examples

```
pnn.predone(net, x)
```

pnn.search_logl *Search for the optimal value of PNN smoothing parameter based on the log loss*

Description

The function pnn.search_logl searches for the optimal value of PNN smoothing parameter by cross-validation.

Usage

```
pnn.search_logl(net, sigmas, nfolds = 4, seed = 1)
```

Arguments

net	A PNN object generated by pnn.fit()
sigmas	A numeric vector to search for the best smoothing parameter
nfolds	A scalar for the number of n-fold, 4 by default
seed	The seed value for the n-fold cross-validation, 1 by default

Value

The list of all searching outcomes and the best outcome

Examples

```
pnn.search_logl(pnn, gen_sobol(1, 3, 10))
```

Index