Package ‘spnn’

October 14, 2022

Type Package

Title Scale Invariant Probabilistic Neural Networks

Version 1.2.1

Date 2020-01-07

Author Romin Ebrahimi

Maintainer Romin Ebrahimi <romin.ebrahimi@utexas.edu>

Description

Scale invariant version of the original PNN proposed by Specht (1990) <doi:10.1016/0893-6080(90)90049-q> with the added functionality of allowing for smoothing along multiple dimensions while accounting for covariances within the data set. It is written in the R statistical programming language. Given a data set with categorical variables, we use this algorithm to estimate the probabilities of a new observation vector belonging to a specific category. This type of neural network provides the benefits of fast training time relative to backpropagation and statistical generalization with only a small set of known observations.

License GPL (>= 2)

Imports MASS (>= 3.1-20), Rcpp (>= 1.0.0)

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.1.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-01-08 20:30:02 UTC

R topics documented:

spnn-package *Scale Invariant Probabilistic Neural Networks*

Description

Scale invariant version of the original PNN proposed by Specht (1990) <doi:10.1016/0893-6080(90)90049-q> with the added functionality of allowing for smoothing along multiple dimensions while accounting for covariances within the data set. It is written in the R statistical programming language. Given a data set with categorical variables, we use this algorithm to estimate the probabilities of a new observation vector belonging to a specific category. This type of neural network provides the benefits of fast training time relative to backpropagation and statistical generalization with only a small set of known observations.

Details

The package exports 4 main functions:

- spnn.learn Create or update a Scale Invariant Probabilistic Neural Network.
- spnn.predict Estimates the category probabilities of new observations using a fitted SPNN.
- cspnn.learn Create or update a Condensed Scale Invariant Probabilistic Neural Network.
- cspnn.predict Estimates the category probabilities of new observations using a fitted CSPNN.

Author(s)

Romin Ebrahimi <romin.ebrahimi@utexas.edu>

References

[1] Specht, Donald F. "Probabilistic neural networks." Neural networks 3.1 (1990): 109-118.

[2] Specht, Donald F. "Enhancements to probabilistic neural networks." Neural Networks, 1992.IJCNN., International Joint Conference on. Vol. 1. IEEE, 1992.

[3] Ebrahimi, Romin "Scale Invariant Probabilistic Neural Networks." The University of Texas, 2018 https://repositories.lib.utexas.edu/handle/2152/65166

See Also

`spnn.learn`, `spnn.predict`, `cspnn.learn`, `cspnn.predict`

Examples

```
library(spnn)
library(datasets)

data(iris)

# shuffle the iris data set
indexRandom <- sample(1:nrow(iris), size = nrow(iris), replace = FALSE)
```

```
# use 100 observations for training set
trainData <- iris[indexRandom[1:100],]

# use remaining observations for testing
testData <- iris[indexRandom[101:length(indexRandom)],]

# fit spnn
spnn <- spnn.learn(set = trainData, category.column = 5)

# estimate probabilities
predictions <- spnn.predict(nn = spnn, newData = testData[,1:4])

# reference matrix must be supplied
# this is not the optimal reference matrix
# this matrix is provided as a simple example
xr <- matrix(c(c(5.00, 3.41, 1.44, 0.24),
               c(5.88, 2.75, 4.23, 1.30),
               c(6.61, 2.97, 5.59, 2.01)),
             nrow = length(unique(trainData$Species)),
             ncol = ncol(trainData) - 1,
             byrow = TRUE)

# fit cspnn
cspnn <- cspnn.learn(set = trainData, xr = xr, category.column = 5)

# estimate probabilities
predictions <- cspnn.predict(nn = cspnn, newData = testData[,1:4])
```

---

cspnn.learn *cspnn.learn*

Description

Create or update a Condensed Scale Invariant Probabilistic Neural Network.

Usage

```
cspnn.learn(set, nn, xr, sigma, category.column = 1)
```

Arguments

set	data.frame or matrix representing the training set. The first column (default category.column = 1) is used to define the category or class of each observation.
nn	(optional) A Condensed Scale Invariant Probabilistic Neural Network object. If provided, the training data set input is concatenated to the current training data set of the neural network. If not provided, a new CSPNN object is created.

`xr` The m by n reference matrix containing optimal parameters for probability estimation. Where m is the number of unique categories and n is the number of input factors used. This matrix must be provided.

`sigma` An n by n square matrix of smoothing parameters where n is the number of input factors. Defaults to using the covariance matrix of the training data set excluding the category.column.

`category.column`
The column number of category data. Default is 1.

Details

The function cspnn.learn creates a new Condensed Scale Invariant Probabilistic Neural Network with a given training data set or updates the training data of an existing CSPNN. It sets the parameters: model, set, xr, category.column, categories, sigma, sigmaInverse, k, and n for the CSPNN.

Value

A trained Condensed Scale Invariant Probabilistic Neural Network (CSPNN)

See Also

`spnn-package`, `cspnn.predict`, `iris`

Examples

```
library(spnn)
library(datasets)

data(iris)

# shuffle the iris data set
indexRandom <- sample(1:nrow(iris), size = nrow(iris), replace = FALSE)

# use 100 observations for training set
trainData <- iris[indexRandom[1:100],]

# use remaining observations for testing
testData <- iris[indexRandom[101:length(indexRandom)],]

# reference matrix must be supplied
# this is not the optimal reference matrix
# this matrix is provided as a simple example
xr <- matrix(c(c(5.00, 3.41, 1.44, 0.24),
               c(5.88, 2.75, 4.23, 1.30),
               c(6.61, 2.97, 5.59, 2.01)),
             nrow = length(unique(trainData$Species)),
             ncol = ncol(trainData) - 1,
             byrow = TRUE)

# fit cspnn
cspnn <- cspnn.learn(set = trainData, xr = xr, category.column = 5)
```

```
# estimate probabilities
predictions <- cspnn.predict(nn = cspnn, newData = testData[,1:4])
```

`cspnn.predict` *cspnn.predict*

Description

Estimates the category probabilities of new observations using a fitted CSPNN.

Usage

```
cspnn.predict(nn, newData)
```

Arguments

`nn` A trained Condensed Scaled Invariant Probabilistic Neural Network.

`newData` A matrix of new observations where each row represents a single observation vector.

Details

Given a trained Condensed Scale Invariant Probabilistic Neural Network and new data, the function cspnn.predict returns the category with the highest probability and the probability estimates for each category.

Value

A list of the guessed categories and the probability estimates of each category.

See Also

`spnn-package`, `cspnn.learn`, `iris`

Examples

```
library(spnn)
library(datasets)

data(iris)

# shuffle the iris data set
indexRandom <- sample(1:nrow(iris), size = nrow(iris), replace = FALSE)

# use 100 observations for training set
trainData <- iris[indexRandom[1:100],]
```

```
# use remaining observations for testing
testData <- iris[indexRandom[101:length(indexRandom)],]

# reference matrix must be supplied
# this is not the optimal reference matrix
# this matrix is provided as a simple example
xr <- matrix(c(c(5.00, 3.41, 1.44, 0.24),
               c(5.88, 2.75, 4.23, 1.30),
               c(6.61, 2.97, 5.59, 2.01)),
             nrow = length(unique(trainData$Species)),
             ncol = ncol(trainData) - 1,
             byrow = TRUE)

# fit cspnn
cspnn <- cspnn.learn(set = trainData, xr = xr, category.column = 5)

# estimate probabilities
predictions <- cspnn.predict(nn = cspnn, newData = testData[,1:4])
```

spnn.learn *spnn.learn*

Description

Create or update a Scale Invariant Probabilistic Neural Network.

Usage

```
spnn.learn(set, nn, sigma, category.column = 1)
```

Arguments

`set` data.frame or matrix representing the training set. The first column (default category.column = 1) is used to define the category or class of each observation.

`nn` (optional) A Scale Invariant Probabilistic Neural Network object. If provided, the training data set input is concatenated to the current training data set of the neural network. If not provided, a new SPNN object is created.

`sigma` An n by n square matrix of smoothing parameters where n is the number of input factors. Defaults to using the covariance matrix of the training data set excluding the category.column.

`category.column` The column number of category data. Default is 1.

Details

The function spnn.learn creates a new Scale Invariant Probabilistic Neural Network with a given training data set or updates the training data of an existing SPNN. It sets the parameters: model, set, category.column, categories, sigma, sigmaInverse, k, and n for the SPNN.

Value

A trained Scale Invariant Probabilistic Neural Network (SPNN)

See Also

spnn-package, spnn.predict, iris

Examples

```
library(spnn)
library(datasets)

data(iris)

# shuffle the iris data set
indexRandom <- sample(1:nrow(iris), size = nrow(iris), replace = FALSE)

# use 100 observations for training set
trainData <- iris[indexRandom[1:100],]

# use remaining observations for testing
testData <- iris[indexRandom[101:length(indexRandom)],]

# fit spnn
spnn <- spnn.learn(set = trainData, category.column = 5)

# estimate probabilities
predictions <- spnn.predict(nn = spnn, newData = testData[,1:4])
```

---

spnn.predict *spnn.predict*

Description

Estimates the category probabilities of new observations using a fitted SPNN.

Usage

```
spnn.predict(nn, newData)
```

Arguments

nn	A trained Scaled Invariant Probabilistic Neural Network.
newData	A matrix of new observations where each row represents a single observation vector.

Details

Given a trained Scale Invariant Probabilistic Neural Network and new data, the function spnn.predict returns the category with the highest probability and the probability estimates for each category.

Value

A list of the guessed categories and the probability estimates of each category.

See Also

spnn-package, spnn.learn, iris

Examples

```
library(spnn)
library(datasets)

data(iris)

# shuffle the iris data set
indexRandom <- sample(1:nrow(iris), size = nrow(iris), replace = FALSE)

# use 100 observations for training set
trainData <- iris[indexRandom[1:100],]

# use remaining observations for testing
testData <- iris[indexRandom[101:length(indexRandom)],]

# fit spnn
spnn <- spnn.learn(set = trainData, category.column = 5)

# estimate probabilities
predictions <- spnn.predict(nn = spnn, newData = testData[,1:4])
```

Index