Package: simpleNeural (via r-universe)

October 16, 2024

Version 0.1.3

Date 2020-02-18

Title An Easy to Use Multilayer Perceptron

Description Trains neural networks (multilayer perceptrons with one hidden layer) for bi- or multi-class classification.

Depends R (>= 3.6)

Suggests verification

License MIT + file LICENSE

LazyData true

Contact mailto:me@daviddernoncourt.com

NeedsCompilation no

Author David Dernoncourt [aut, cre] (LUNAM Universite, Universite Angers, Laboratoire d'ergonomie et d'epidemiologie en sante au travail (LEEST), w/ support from the French National Research Program for Environmental and Occupational Health of Anses (2012/18))

Maintainer David Dernoncourt <me@daviddernoncourt.com>

Date/Publication 2020-02-18 15:50:02 UTC

Repository https://ac4bb21b.r-universe.dev

RemoteUrl https://github.com/cran/simpleNeural

RemoteRef HEAD

RemoteSha f01270dc5998495464bff0311b74687e77909748

Contents

`sN.MLPpredict` *Runs a multilayer perceptron*

Description

Runs a multilayer perceptron

Usage

```
sN.MLPpredict(nnModel, X, raw = FALSE)
```

Arguments

`nnModel`	A list containing the coefficients for the MLP (as produced with sN.MLPtrain())
`X`	Matrix of predictors
`raw`	If true, returns score of each output option. If false, returns the output option with highest value.

Value

The predicted values obtained by the MLP

Examples

```
data(UCI.transfusion);
X=as.matrix(sN.normalizeDF(as.data.frame(UCI.transfusion[,1:4])));
y=as.matrix(UCI.transfusion[,5]);
myMLP=sN.MLPtrain(X=X,y=y,hidden_layer_size=4,it=50,lambda=0.5,alpha=0.5);
myPrediction=sN.MLPpredict(nnModel=myMLP,X=X,raw=TRUE);
#library('verification');
#roc.area(y,myPrediction[,2]);
```

`sN.MLPtrain` *Trains a multilayer perceptron with 1 hidden layer*

Description

Trains a multilayer perceptron with 1 hidden layer and a sigmoid activation function, using back-propagation and gradient descent. Don’t forget to normalize the data first - sN.normalizeDF(), provided in the package, can be used to do so.

Usage

```
sN.MLPtrain(X, y, hidden_layer_size = 5, it = 50, lambda = 0.5,
  alpha = 0.5)
```

Arguments

- `X` Matrix of predictors
- `y` Vector of output (the ANN learns y=ANN(X)). Classes should be assigned an integer number, starting at 0 for the first class.
- `hidden_layer_size` Number of units in the hidden layer
- `it` Number of iterations for the gradient descent. The default value of 50 may be a little low in some cases. 100 to 1000 are generally sensible values.
- `lambda` Penalization for model coefficients (regularization parameter)
- `alpha` Speed multiplier (learning rate) for gradient descent

Value

The coefficients of the MLP, in a list (Theta1 between input and hidden layers, Theta2 between hidden and output layers)

References

M.W Gardner, S.R Dorling, Artificial neural networks (the multilayer perceptron)- a review of applications in the atmospheric sciences, Atmospheric Environment, Volume 32, Issues 14-15, 1 August 1998, Pages 2627-2636, ISSN 1352-2310, doi: 10.1016/S1352-2310(97)00447-0 [http://www.sciencedirect.com/science/article/pii/S1352231097004470]

Jain, A.K.; Jianchang Mao; Mohiuddin, K.M., "Artificial neural networks: a tutorial," Computer , vol.29, no.3, pp.31,44, Mar 1996. doi: 10.1109/2.485891 [http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=485891&isnumber=10412]

Rumelhart, David E., Geoffrey E. Hinton, and R. J. Williams. "Learning Internal Representations by Error Propagation". David E. Rumelhart, James L. McClelland, and the PDP research group (editors). Parallel distributed processing: Explorations in the microstructure of cognition, Volume 1: Foundations. MIT Press, 1986.

Examples

```
# NB: the provided examples are just here to help use the package's functions.
# In real use cases you should perform a proper validation (cross-validation,
# external validation data...)
data(UCI.BCD.Wisconsin);
X=as.matrix(sN.normalizeDF(as.data.frame(UCI.BCD.Wisconsin[,3:32])));
y=as.matrix(UCI.BCD.Wisconsin[,2]);
myMLP=sN.MLPtrain(X=X,y=y,hidden_layer_size=20,it=50,lambda=0.5,alpha=0.5);
myPrediction=sN.MLPpredict(nnModel=myMLP,X=X,raw=TRUE);
#library('verification');
#roc.area(y,myPrediction[,2]);
```

`sN.normalizeDF` *Normalize data*

Description

Normalize all columns of a dataframe so that all values are in [0;1] and for each column the maximum value is 1 and the minimum 0.

newx=(x-min(X))/(max(X)-min(X))

Usage

```
sN.normalizeDF(dframe)
```

Arguments

`dframe`	The dataframe to be normalized

Value

The normalized dataframe

`UCI.BCD.Wisconsin` *Breast Cancer Wisconsin (Diagnostic) Data Set*

Description

Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

Usage

```
data(UCI.BCD.Wisconsin)
```

Format

A data frame with 569 rows and 32 variables

Details

Separating plane described above was obtained using Multisurface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree Construction Via Linear Programming." Proceedings of the 4th Midwest Artificial Intelligence and Cognitive Science Society, pp. 97-101, 1992], a classification method which uses linear programming to construct a decision tree. Relevant features were selected using an exhaustive search in the space of 1-4 features and 1-3 separating planes.

The actual linear program used to obtain the separating plane in the 3-dimensional space is that described in: [K. P. Bennett and O. L. Mangasarian: "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets", Optimization Methods and Software 1, 1992, 23-34].

The variables are as follows:

- ID number
- Diagnosis (1 = malignant, 0 = benign)
- Ten real-valued features are computed for each cell nucleus

Source

Dataset downloaded from the UCI Machine Learning Repository. http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

Creators:

1. Dr. William H. Wolberg, General Surgery Dept. University of Wisconsin, Clinical Sciences Center Madison, WI 53792 wolberg 'at' eagle.surgery.wisc.edu

2. W. Nick Street, Computer Sciences Dept. University of Wisconsin, 1210 West Dayton St., Madison, WI 53706 street 'at' cs.wisc.edu 608-262-6619

3. Olvi L. Mangasarian, Computer Sciences Dept. University of Wisconsin, 1210 West Dayton St., Madison, WI 53706 olvi 'at' cs.wisc.edu

Donor: Nick Street

References

W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction for breast tumor diagnosis. IS&T/SPIE 1993 International Symposium on Electronic Imaging: Science and Technology, volume 1905, pages 861-870, San Jose, CA, 1993.

Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

`UCI.ISOLET.ABC` *ISOLET Data Set (ABC)*

Description

This data set was generated as follows. 150 subjects spoke the name of each letter of the alphabet twice. Hence, we have 52 training examples from each speaker.

Usage

```
data(UCI.ISOLET.ABC)
```

Format

A data frame with 900 rows and 618 variables

Details

To reduce package size, only the 3 first letters are included here. The full dataset can be obtained from http://archive.ics.uci.edu/ml/datasets/ISOLET.

The features are described in the paper by Cole and Fanty cited below. The features include spectral coefficients; contour features, sonorant features, pre-sonorant features, and post-sonorant features. Exact order of appearance of the features is not known.

Source

Dataset downloaded from the UCI Machine Learning Repository. http://archive.ics.uci.edu/ml/datasets/ISOLET

Creators:

Ron Cole and Mark Fanty Department of Computer Science and Engineering, Oregon Graduate Institute, Beaverton, OR 97006. cole 'at' cse.ogi.edu, fanty 'at' cse.ogi.edu

Donor:

Tom Dietterich Department of Computer Science Oregon State University, Corvallis, OR 97331 tgd 'at' cs.orst.edu

References

Fanty, M., Cole, R. (1991). Spoken letter recognition. In Lippman, R. P., Moody, J., and Touretzky, D. S. (Eds). Advances in Neural Information Processing Systems 3. San Mateo, CA: Morgan Kaufmann. [http://rexa.info/paper/bee6de062d0d168c5c5b5290b11cd6b12ca8472e]

Examples

```
# NB: 50 iterations isn't enough in this case,
# it was chosen so that the example runs fast enough on CRAN check farm
data(UCI.ISOLET.ABC);
X=as.matrix(sN.normalizeDF(as.data.frame(UCI.ISOLET.ABC[,1:617])));
y=as.matrix(UCI.ISOLET.ABC[,618]-1);
myMLP=sN.MLPtrain(X=X,y=y,hidden_layer_size=20,it=50,lambda=0.5,alpha=0.5);
myPrediction=sN.MLPpredict(nnModel=myMLP,X=X,raw=FALSE);
table(y,myPrediction);
```

UCI.transfusion *Blood Transfusion Service Center Data Set*

Description

Data taken from the Blood Transfusion Service Center in Hsin-Chu City in Taiwan. To demonstrate the RFMTC marketing model (a modified version of RFM), this study adopted the donor database of Blood Transfusion Service Center in Hsin-Chu City in Taiwan. The center passes their blood transfusion service bus to one university in Hsin-Chu City to gather blood donated about every three months. To build a FRMTC model, we selected 748 donors at random from the donor database. These 748 donor data, each one included R (Recency - months since last donation), F (Frequency - total number of donation), M (Monetary - total blood donated in c.c.), T (Time - months since first donation), and a binary variable representing whether he/she donated blood in March 2007 (1 stand for donating blood; 0 stands for not donating blood).

Usage

```
data(UCI.transfusion)
```

Format

A data frame with 748 rows and 5 variables

Details

The variables are as follows:

- R. Recency - months since last donation
- F. Frequency - total number of donations
- M. Monetary - total blood donated in c.c. (mL)
- T. Time - months since first donation
- y. a binary variable representing whether he/she donated blood in March 2007 (1=yes; 0 =no)

Source

Dataset downloaded from the UCI Machine Learning Repository. http://archive.ics.uci.edu/ml/datasets/Blood+Transfusion+Service+Center

Original Owner and Donor: Prof. I-Cheng Yeh Department of Information Management Chung-Hua University Hsin Chu, Taiwan 30067, R.O.C. e-mail: icyeh 'at' chu.edu.tw

References

Yeh, I-Cheng, Yang, King-Jang, and Ting, Tao-Ming, "Knowledge discovery on RFM model using Bernoulli sequence", Expert Systems with Applications, 2008. DOI: 10.1016/j.eswa.2008.07.018

Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Index