Package ‘simpleNeural’

April 28, 2015

Version 0.1.1
Date 2015-04-27
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 (&gt;= 3.1)
Suggests verification
License MIT + file LICENSE
LazyData true
URL https://bitbucket.org/patheticcockroach/simpleneural
BugReports 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>
Repository CRAN
Date/Publication 2015-04-28 20:24:29

R topics documented:

  sN.MLPpredict .......................................................... 2
  sN.MLPtrain ............................................................ 2
  sN.normalizeDF ....................................................... 4
  UCI.BCD.Wisconsin .................................................. 4
  UCI.ISOLET.ABC ..................................................... 5
  UCI.transfusion ...................................................... 7

Index 8
sN.MLPpredict  

*Runs a multilayer perceptron*

**Description**

Runs a multilayer perceptron

**Usage**

```r
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**

```r
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**

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

\begin{itemize}
  \item \textbf{x} \quad \text{Matrix of predictors}
  \item \textbf{y} \quad \text{Vector of output (the ANN learns } y=\text{ANN}(x))\text{. Classes should be assigned an integer number, starting at 0 for the first class.}
  \item \textbf{hidden\_layer\_size} \quad \text{Number of units in the hidden layer}
  \item \textbf{it} \quad \text{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.}
  \item \textbf{lambda} \quad \text{Penalization for model coefficients (regularization parameter)}
  \item \textbf{alpha} \quad \text{Speed multiplier (learning rate) for gradient descent}
\end{itemize}

Value

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

References


Examples

\begin{verbatim}
# 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...)
library(sN.MLPtrain)
data(UCI.BoD.Wisconsin);
X = as.matrix(sN.normalizeDF(as.data.frame(UCI.BoD.Wisconsin[,3:32])));
y = as.matrix(UCI.BoD.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]);
\end{verbatim}
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.

\[ \text{newx} = \frac{x - \text{min}(X)}{\text{max}(X) - \text{min}(X)} \]

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


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


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


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);
**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**


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**


Index

*Topic datasets
  UCI.BCD.Wisconsin, 4
  UCI.ISOLET.ABC, 5
  UCI.transfusion, 7

sN.MLPPredict, 2
sN.MLPtrain, 2
sN.normalizeDF, 4

UCI.BCD.Wisconsin, 4
UCI.ISOLET.ABC, 5
UCI.transfusion, 7