Package ‘evclass’

March 20, 2017

Type Package
Title Evidential Distance-Based Classification
Version 1.1.1
Date 2017-03-19
Author Thierry Denoeux
Maintainer Thierry Denoeux <tdenoeux@utc.fr>
Description Different evidential distance-based classifiers, which provide outputs in the form of Dempster-Shafer mass functions. The methods are: the evidential K-nearest neighbor rule and the evidential neural network.
License GPL-3
Depends R (>= 3.1.0)
Imports FNN
LazyData TRUE
RoxygenNote 6.0.1
VignetteBuilder knitr
Suggests knitr,rmardown,datasets
NeedsCompilation no
Repository CRAN
Date/Publication 2017-03-20 05:22:18 UTC

R topics documented:

decision ................................. 2
EkNNfit ................................ 3
EkNNinit ................................ 5
EkNNval ................................ 6
evclass ................................ 7
glass .................................... 8
ionosphere ............................. 9
proDSfit ............................... 10
decision

Description
decision returns decisions from a loss matrix and mass functions computed by an evidential classifier.

Usage
decision(m, L = 1 - diag(ncol(m) - 1), rule = c("upper", "lower", "pignistic", "hurwicz"), rho = 0.5)

Arguments
m Matrix of masses for n test cases. Each row is a mass function. The first M columns correspond to the mass assigned to each of the M classes. The last column corresponds to the mass assigned to the whole set of classes.

L The loss matrix of dimension (na,M) or (na,M+1), where na is the set of actions. L[k,j] is the loss incurred of action j is chosen and the true class if ω_k. If L has M+1 rows, the last row corresponds to the unknown class.

rule Decision rule to be used. Must be one of these: 'upper' (upper expectation), 'lower' (lower expectations), 'pignistic' (pignistic expectation), 'hurwicz' (weighted sum of the lower and upper expectations).

rho Parameter between 0 and 1. Used only if rule='rho'.

Details
This function implements the decision rules described in Denoeux (1997), with an arbitrary loss function. The decision rules are the minimization of the lower, upper or pignistic expectation, and Jaffray’s decision rule based on minimizing a convex combination of the lower and upper expectations. The function also handles the case where there is an "unknown" class, in addition to the classes represented in the training set.

Value
A n-vector with the decisions (integers between 1 and na).

Author(s)
Thierry Denoeux.
References

Available from https://www.hds.utc.fr/~tdenoeux.

See Also

EkNNval, proDSval

Examples

```r
## Example with M=2 classes
m <- matrix(c(0.9, 0.1, 0.4, 0.6, 0, 0.1, 0.8), 3, 3, byrow = TRUE)
## Loss matrix with na=4 acts: assignment to class 1, assignment to class 2,
## rejection, and assignment to the unknown class.
L <- matrix(c(0.1, 1, 0.1, 0.2, 0.2, 0.2, 0.25, 0.25, 0), 3, 4)
d <- decision(m, L, 'upper') ## instances 2 and 3 are rejected
d <- decision(m, L, 'lower') ## instance 2 is rejected, instance 3 is
## assigned to the unknown class
```

---

EkNNfit

Training of the EkNN classifier

Description

EkNNfit optimizes the parameters of the EkNN classifier.

Usage

```r
EkNNfit(x, y, K, param = NULL, alpha = 0.95,
        lambda = 1/max(as.numeric(y)), optimize = TRUE, options = list(maxiter =
        300, eta = 0.1, gain_min = 1e-06, disp = TRUE))
```

Arguments

- `x` Input matrix of size n x d, where n is the number of objects and d the number of attributes.
- `y` Vector of class labels (of length n). May be a factor, or a vector of integers from 1 to M (number of classes).
- `K` Number of neighbors.
- `param` Initial parameters (default: NULL).
- `alpha` Parameter α (default: 0.95)
- `lambda` Parameter of the cost function. If lambda=1, the cost function measures the error between the plausibilities and the 0-1 target values. If lambda=1/M, where M is the number of classes (default), the pignistic probabilities are considered in the cost function. If lambda=0, the beliefs are used.
EkNNfit

optimize Boolean. If TRUE (default), the parameters are optimized.

options A list of parameters for the optimization algorithm: maxiter (maximum number of iterations), eta (initial step of gradient variation), gain_min (minimum gain in the optimisation loop), disp (Boolean; if TRUE, intermediate results are displayed during the optimization).

Details

If the argument param is not supplied, the function EkNNinit is called.

Value

A list with five elements:

param The optimized parameters.
cost Final value of the cost function.
err Leave-one-out error rate.
ypred Leave-one-out predicted class labels (coded as integers from 1 to M).
m Leave-one-out predicted mass functions. The first M columns correspond to the mass assigned to each class. The last column corresponds to the mass assigned to the whole set of classes.

Author(s)

Thierry Denoeux.

References


Available from https://www.hds.utc.fr/~tdenoeux.

See Also

EkNNinit, EkNNval

Examples

```r
## Iris dataset
data(iris)
x <- iris[, 1:4]
y <- iris[, 5]
fit <- EkNNfit(x, y, K = 5)
```
EkNNinit

**Description**

EkNNinit returns initial parameter values for the EkNN classifier.

**Usage**

EkNNinit(x, y, alpha = 0.95)

**Arguments**

- **x**: Input matrix of size n x d, where n is the number of objects and d the number of attributes.
- **y**: Vector of class labels (of length n). May be a factor, or a vector of integers from 1 to M (number of classes).
- **alpha**: Parameter $\alpha$.

**Details**

Each parameter $\gamma_k$ is set to the inverse of the square root of the mean Euclidean distances within class k. Note that $\gamma_k$ here is the square root of the $\gamma_k$ as defined in (Zouhal and Denoeux, 1998). By default, parameter alpha is set to 0.95. This value normally does not have to be changed.

**Value**

A list with two elements:

- **gamma**: Vector of parameters $\gamma_k$, of length c, the number of classes.
- **alpha**: Parameter $\alpha$, set to 0.95.

**Author(s)**

Thierry Denoeux.

**References**


Available from [https://www.hds.utc.fr/~tdenoeux](https://www.hds.utc.fr/~tdenoeux).

**See Also**

EkNNfit, EkNNval
EkNNval

**Classification of a test set by the EkNN classifier**

**Examples**

```r
## Iris dataset
data(iris)
x<-iris[,1:4]
y<-iris[,5]
param<-EkNNinit(x,y)
param
```

**Description**

EkNNval classifies instances in a test set using the EkNN classifier.

**Usage**

```r
EkNNval(xtrain, ytrain, xtst, K, ytst = NULL, param = NULL)
```

**Arguments**

- `xtrain`: Matrix of size ntrain x d, containing the values of the d attributes for the training data.
- `ytrain`: Vector of class labels for the training data (of length ntrain). May be a factor, or a vector of integers from 1 to M (number of classes).
- `xtst`: Matrix of size ntst x d, containing the values of the d attributes for the test data.
- `K`: Number of neighbors.
- `ytst`: Vector of class labels for the test data (optional). May be a factor, or a vector of integers from 1 to M (number of classes).
- `param`: Parameters, as returned by `EkNNfit`.

**Details**

If class labels for the test set are provided, the test error rate is also returned. If parameters are not supplied, they are given default values by `EkNNinit`.

**Value**

A list with three elements:

- `m`: Predicted mass functions for the test data. The first M columns correspond to the mass assigned to each class. The last column corresponds to the mass assigned to the whole set of classes.
- `ypred`: Predicted class labels for the test data (coded as integers from 1 to M).
- `err`: Test error rate.
Author(s)

Thierry Denoeux.

References


Available from https://www.hds.utc.fr/~tdenoeux.

See Also

EkNNinit, EkNNfit

Examples

```r
# Iris dataset
data(iris)
train <- sample(150, 100)
xtrain <- iris[train, 1:4]
ytrain <- iris[train, 5]
xtst <- iris[-train, 1:4]
ystst <- iris[-train, 5]
K <- 5
fit <- EkNNfit(xtrain, ytrain, K)
test <- EkNNval(xtrain, ytrain, xtst, K, ytst, fit$param)
```

Description

The evclass package currently contains functions for two evidential classifiers: the evidential K-nearest neighbor (EK-NN) rule (Denoeux, 1995; Zouhal and Denoeux, 1998) and the evidential neural network (Denoeux, 2000). In contrast with classical statistical classifiers, evidential classifier quantify the uncertainty of the classification using Dempster-Shafer mass functions.

Details

The main functions are: EkNNinit, EkNNfit and EkNNval for the initialization, training and evaluation of the EK-NN classifier, proDSinit, proDSfit and proDSval for the evidential neural network classifier, and decision for decision-making.
References


Available from https://www.hds.utc.fr/~tdenoeux.

See Also

EkNNinit, EkNNfit, EkNNval, proDSinit, proDSfit, proDSval.

---

glass

Glass dataset

Description

This data set contains the description of 214 fragments of glass originally collected for a study in the context of criminal investigation. Each fragment has a measured reflectivity index and chemical composition (weight percent of Na, Mg, Al, Si, K, Ca, Ba and Fe). As suggested by Ripley (1994), 29 instances were discarded, and the remaining 185 were re-grouped in four classes: window float glass (70), window non-float glass (76), vehicle window glass (17) and other (22). The data set was split randomly in a training set of size 89 and a test set of size 96.

Usage

data(glass)

Format

A list with two elements:

x The 185 x 9 object-attribute matrix.

y A 185-vector containing the class labels.

References

P. M. Murphy and D. W. Aha. UCI Repository of machine learning databases. [Machine readable data repository]. University of California, Department of Information and Computer Science, Irvine, CA.


Examples

data(glass)
table(glass$y)

---

Ionosphere dataset

Description

This dataset was collected by a radar system and consists of phased array of 16 high-frequency antennas with a total transmitted power of the order of 6.4 kilowatts. The targets were free electrons in the ionosphere. “Good” radar returns are those showing evidence of some type of structure in the ionosphere. “Bad” returns are those that do not. There are 351 instances and 34 numeric attributes. The first 175 instances are training data, the rest are test data. This version of dataset was used by Zouhal and Denoeux (1998).

Usage

data(ionosphere)

Format

A list with two elements:

- \textit{x} The 351 x 18 object-attribute matrix.
- \textit{y} A 351-vector containing the class labels.

References

P. M. Murphy and D. W. Aha. UCI Reposition of machine learning databases. [Machine readable data repository]. University of California, Departement of Information and Computer Science, Irvine, CA.


Examples

data(ionosphere)
table(vehicles$y)
Description

proDSfit performs parameter optimization for the evidential neural network classifier.

Usage

proDSfit(x, y, param, lambda = 1/max(as.numeric(y)), mu = 0,
    optimProto = TRUE, options = list(maxiter = 500, eta = 0.1, gain_min =
    1e-04, disp = 10))

Arguments

x      Input matrix of size n x d, where n is the number of objects and d the number of
       attributes.

y      Vector of class labels (of length n). May be a factor, or a vector of integers from
       1 to M (number of classes).

param  Initial parameters (see link{proDSinit}).

lambda Parameter of the cost function. If lambda=1, the cost function measures the error
       between the plausibilities and the 0-1 target values. If lambda=1/M, where M is
       the number of classes (default), the piginistic probabilities are considered in the
       cost function. If lambda=0, the beliefs are used.

mu     Regularization hyperparameter (default=0).

optimProto Boolean. If TRUE, the prototypes are optimized (default). Otherwise, they are
       fixed.

options A list of parameters for the optimization algorithm: maxiter (maximum number
       of iterations), eta (initial step of gradient variation), gain_min (minimum gain
       in the optimisation loop), disp (integer; if >0, intermediate results are displayed
       every disp iterations).

Details

The prototypes are initialized by the k-means algorithms. The initial membership values $u_{ik}$ of each
prototype $p_i$ to class $\omega_k$ are normally defined as the proportion of training samples from class $\omega_k$ in
the neighborhood of prototype $p_i$. If arguments crisp and nprotoPerClass are set to TRUE, the
prototypes are assigned to one and only one class.

Value

A list with three elements:

param Optimized network parameters.

cost Final value of the cost function.

err Training error rate.
**Description**

`proDSinit` returns initial parameter values for the evidential neural network classifier.

**Usage**

```
proDSinit(x, y, nproto, nprotoPerClass = FALSE, crisp = FALSE)
```

**Arguments**

- **x**  
  Input matrix of size n x d, where n is the number of objects and d the number of attributes.
- **y**  
  Vector of class labels (of length n). May be a factor, or a vector of integers from 1 to M (number of classes).
- **nproto**  
  Number of prototypes.
- **nprotoPerClass**  
  Boolean. If TRUE, there are nproto prototypes per class. If FALSE (default), the total number of prototypes is equal to nproto.
- **crisp**  
  Boolean. If TRUE, the prototypes have full membership to only one class. (Available only if nprotoPerClass=TRUE).
Details

The prototypes are initialized by the k-means algorithms. The initial membership values $u_{ik}$ of each prototype $p_i$ to class $\omega_k$ are normally defined as the proportion of training samples from class $\omega_k$ in the neighborhood of prototype $p_i$. If arguments `crisp` and `nprotoPerClass` are set to TRUE, the prototypes are assigned to one and only one class.

Value

A list with four elements containing the initialized network parameters

- **alpha** Vector of length $r$, where $r$ is the number of prototypes.
- **gamma** Vector of length $r$
- **beta** Matrix of size $(r,M)$, where $M$ is the number of classes.
- **W** Matrix of size $(r,d)$, containing the prototype coordinates.

Author(s)

Thierry Denoeux.

References


Available from [https://www.hds.utc.fr/~tdenoeux](https://www.hds.utc.fr/~tdenoeux).

See Also

`proDSfit`, `proDSval`

Examples

```r
## Glass dataset
data(glass)
xapp <- glass$x[1:89,]
yapp <- glass$y[1:89]
paramP <- proDSinit(xapp, yapp, nproto=7)
paramP

proDSval(x, param, y = NULL)
```

---

**proDSval**

*Classification of a test set by the evidential neural network classifier*

Description

`proDSval` classifies instances in a test set using the evidential neural network classifier.

Usage

```r
proDSval(x, param, y = NULL)
```
Arguments

- **x**: Matrix of size n x d, containing the values of the d attributes for the test data.
- **param**: Neural network parameters, as provided by `proDSfit`.
- **y**: Optional vector of class labels for the test data. May be a factor, or a vector of integers from 1 to M (number of classes).

Details

If class labels for the test set are provided, the test error rate is also returned.

Value

A list with three elements:

- **m**: Predicted mass functions for the test data. The first M columns correspond to the mass assigned to each class. The last column corresponds to the mass assigned to the whole set of classes.
- **ypred**: Predicted class labels for the test data.
- **err**: Test error rate (if the class label of test data has been provided).

Author(s)

Thierry Denoeux.

References


Available from [https://www.hds.utc.fr/~tdenoeux](https://www.hds.utc.fr/~tdenoeux).

See Also

`proDSinit`, `proDSfit`

Examples

```r
## Glass dataset
data(glass)
xapp<-glass$x[1:89,]
yapp<-glass$y[1:89]
x tst<-glass$x[90:185,]
ytst<-glass$y[90:185]
## Initialization
param0<-proDSinit(xapp,yapp,nproto=7)
## Training
fit<-proDSfit(xapp,yapp,param0)
## Test
val<-proDSval(xtst,fit$param,ytst)
## Confusion matrix
table(ytst,val$ypred)
```
vehicles

Vehicles dataset

Description
This dataset was collected from silhouettes by the HIPS (Hierarchical Image Processing System) extension BINATTS. Four model vehicles were used for the experiment: bus, Chevrolet van, Saab 9000, and Opel Manta. The data were used to distinguish 3D objects within a 2-D silhouette of the objects. There are 846 instances and 18 numeric attributes. The first 564 objects are training data, the rest are test data. This version of dataset was used by Zouhal and Denoeux (1998).

Usage

data(vehicles)

Format
A list with two elements:

x The 846 x 18 object-attribute matrix.
y A 846-vector containing the class labels.

References
P. M. Murphy and D. W. Aha. UCI Reposition of machine learning databases. [Machine readable data repository]. University of California, Department of Information and Computer Science, Irvine, CA.


Examples

data(vehicles)
table(vehicles$y)
Index

*Topic datasets
  glass, 8
  ionosphere, 9
  vehicles, 14

decision, 2, 7

EkNNfit, 3, 5–8
EkNNinit, 4, 5, 6–8
EkNNval, 3–5, 6, 7, 8
evclass, 7
evclass-package (evclass), 7
glass, 8

ionosphere, 9

proDSfit, 7, 8, 10, 12, 13
proDSinit, 7, 8, 11, 11, 13
proDSval, 3, 7, 8, 11, 12, 12

vehicles, 14