Package ‘isoboost’

January 27, 2026

Title Isotonic Boosting Classification Rules

Version 1.0.2

Date 2026-01-27

Description In classification problems a monotone relation between some predictors and the classes may be assumed. In this package 'isoboost' we propose new boosting algorithms, based on LogitBoost, that incorporate this isotonicity information, yielding more accurate and easily interpretable rules.

Imports Iso, isotone, rpart, methods

License GPL-2 | GPL-3

NeedsCompilation no

Author David Conde [aut, cre],
Miguel A. Fernandez [aut],
Cristina Rueda [aut],
Bonifacio Salvador [aut]

Maintainer David Conde <daconrio@gmail.com>

Repository CRAN

Date/Publication 2026-01-27 14:00:02 UTC

Contents

isoboost-package *Isotonic Boosting Classification Rules*

Description

In this package we present new boosting classification rules based on LogitBoost when it can be assumed that higher (or lower) values of some predictors are related to higher levels of the response.

Details

Package: isoboost

Type: Package

Version: 1.0.2

Date: 2026-01-27

License: GPL-2 | GPL-3

For a complete list of functions with individual help pages, use `library(help = "isoboost")`.

Author(s)

David Conde, Miguel A. Fernandez, Cristina Rueda, Bonifacio Salvador

Maintainer: David Conde <daconrio@gmail.com>

amilb *(Adjacent-categories) Multiple Isotonic LogitBoost*

Description

Train and predict logitboost-based classification algorithm using multivariate isotonic regression (linear regression for no monotone features) as weak learners, based on the adjacent-categories logistic model (see Agresti (2010)). For full details on this algorithm, see Conde et al. (2020).

Usage

```
amilb(xlearn, ...)

## S3 method for class 'formula'
amilb(formula, data, ...)

## Default S3 method:
amilb(xlearn, ylearn, xtest = xlearn, mfinal = 100,
monotone_constraints = rep(0, dim(xlearn)[2]), prior = NULL, ...)
```

Arguments

`formula` A formula of the form `groups ~ x1 + x2 + ...`. That is, the response is the class variable and the right hand side specifies the explanatory variables.

`data` Data frame from which variables specified in `formula` are to be taken.

`xlearn` (Required if no formula is given as the principal argument.) A data frame or matrix containing the explanatory variables.

`ylearn` (Required if no formula is given as the principal argument.) A numeric vector or factor with numeric levels specifying the class for each observation.

`xtest` A data frame or matrix of cases to be classified, containing the features used in `formula` or `xlearn`.

`mfinal` Maximum number of iterations of the algorithm.

`monotone_constraints`
Numerical vector consisting of 1, 0 and -1, its length equals the number of features in `xlearn`. 1 is increasing, -1 is decreasing and 0 is no constraint.

`prior` The prior probabilities of class membership. If unspecified, equal prior probabilities are used. If present, the probabilities must be specified in the order of the factor levels.

`...` Arguments passed to or from other methods.

Value

A list containing the following components:

`call` The (matched) function call.

`trainset` Matrix with the training set used (first columns) and the class for each observation (last column).

`prior` Prior probabilities of class membership used.

`apparent` Apparent error rate.

`mfinal` Number of iterations of the algorithm.

`loglikelihood` Log-likelihood.

`posterior` Posterior probabilities of class membership for `xtest` set.

`class` Labels of the class with maximal probability for `xtest` set.

Note

This function may be called using either a formula and data frame, or a data frame and grouping variable, or a matrix and grouping variable as the first two arguments. All other arguments are optional.

Classes must be identified, either in a column of `data` or in the `ylearn` vector, by natural numbers varying from 1 to the number of classes. The number of classes must be greater than 1.

If there are missing values in either `data`, `xlearn` or `ylearn`, corresponding observations will be deleted.

Author(s)

David Conde

References

Agresti, A. (2010). Analysis of Ordinal Categorical Data, 2nd edition. John Wiley and Sons. New Jersey.

Conde, D., Fernandez, M. A., Rueda, C., and Salvador, B. (2020). Isotonic boosting classification rules. *Advances in Data Analysis and Classification*, 1-25.

See Also

asilb, csilb, cmilb

Examples

```
data(motors)
table(motors$condition)
##  1  2  3  4
## 83 67 70 60

## Let us consider the first three variables as predictors
data <- motors[, 1:3]
grouping = motors$condition
##
## Lower values of the amplitudes are expected to be
## related to higher levels of damage severity, so
## we can consider the following monotone constraints
monotone_constraints = rep(-1, 3)

set.seed(7964)
values <- runif(dim(data)[1])
trainsubset <- values < 0.2
obj <- amilb(data[trainsubset, ], grouping[trainsubset],
               data[-trainsubset, ], 100, monotone_constraints)

## Apparent error
obj$apparent
## 4.761905

## Error rate
100*mean(obj$class != grouping[-trainsubset])
## 15.41219
```

asilb *(Adjacent-categories) Simple Isotonic LogitBoost*

Description

Train and predict logitboost-based classification algorithm using isotonic regression (decision stumps for no monotone features) as weak learners, based on the adjacent-categories logistic model (see Agresti (2010)). For full details on this algorithm, see Conde et al. (2020).

Usage

```
asilb(xlearn, ...)

## S3 method for class 'formula'
asilb(formula, data, ...)

## Default S3 method:
asilb(xlearn, ylearn, xtest = xlearn, mfinal = 100,
monotone_constraints = rep(0, dim(xlearn)[2]), prior = NULL, ...)
```

Arguments

`formula` A formula of the form `groups ~ x1 + x2 + ....` That is, the response is the class variable and the right hand side specifies the explanatory variables.

`data` Data frame from which variables specified in `formula` are to be taken.

`xlearn` (Required if no formula is given as the principal argument.) A data frame or matrix containing the explanatory variables.

`ylearn` (Required if no formula is given as the principal argument.) A numeric vector or factor with numeric levels specifying the class for each observation.

`xtest` A data frame or matrix of cases to be classified, containing the features used in `formula` or `xlearn`.

`mfinal` Number of iterations of the algorithm.

`monotone_constraints`
Numerical vector consisting of 1, 0 and -1, its length equals the number of features in `xlearn`. 1 is increasing, -1 is decreasing and 0 is no constraint.

`prior` The prior probabilities of class membership. If unspecified, equal prior probabilities are used. If present, the probabilities must be specified in the order of the factor levels.

`...` Arguments passed to or from other methods.

Value

A list containing the following components:

`call` The (matched) function call.

`trainset` Matrix with the training set used (first columns) and the class for each observation (last column).

`prior` Prior probabilities of class membership used.

`apparent` Apparent error rate.

`mfinal` Number of iterations of the algorithm.

`loglikelihood` Log-likelihood.

`posterior` Posterior probabilities of class membership for `xtest` set.

`class` Labels of the class with maximal probability for `xtest` set.

Note

This function may be called using either a formula and data frame, or a data frame and grouping variable, or a matrix and grouping variable as the first two arguments. All other arguments are optional.

Classes must be identified, either in a column of `data` or in the `ylearn` vector, by natural numbers varying from 1 to the number of classes. The number of classes must be greater than 1.

If there are missing values in either `data`, `xlearn` or `ylearn`, corresponding observations will be deleted.

Author(s)

David Conde

References

Agresti, A. (2010). Analysis of Ordinal Categorical Data, 2nd edition. John Wiley and Sons. New Jersey.

Conde, D., Fernandez, M. A., Rueda, C., and Salvador, B. (2020). Isotonic boosting classification rules. *Advances in Data Analysis and Classification*, 1-25.

See Also

`amilb`, `csilb`, `cmilb`

Examples

```
data(motors)
table(motors$condition)
##  1  2  3  4
## 83 67 70 60

## Let us consider the first three variables as predictors
data <- motors[, 1:3]
grouping = motors$condition
##
## Lower values of the amplitudes are expected to be
## related to higher levels of damage severity, so
## we can consider the following monotone constraints
```

```
monotone_constraints = rep(-1, 3)

set.seed(7964)
values <- runif(dim(data)[1])
trainsubset <- values < 0.2
obj <- asilb(data[trainsubset, ], grouping[trainsubset],
               data[-trainsubset, ], 50, monotone_constraints)

## Apparent error
obj$apparent
## 4.761905

## Error rate
100*mean(obj$class != grouping[-trainsubset])
## 14.69534
```

`cmilb` *Cumulative probabilities Multiple Isotonic LogitBoost*

Description

Train and predict logitboost-based classification algorithm using multivariate isotonic regression (linear regression for no monotone features) as weak learners, based on the cumulative probabilities logistic model (see Agresti (2010)). For full details on this algorithm, see Conde et al. (2020).

Usage

```
cmilb(xlearn, ...)

## S3 method for class 'formula'
cmilb(formula, data, ...)

## Default S3 method:
cmilb(xlearn, ylearn, xtest = xlearn, mfinal = 100,
monotone_constraints = rep(0, dim(xlearn)[2]), prior = NULL, ...)
```

Arguments

`formula`	A formula of the form `groups ~ x1 + x2 + ...`. That is, the response is the class variable and the right hand side specifies the explanatory variables.
`data`	Data frame from which variables specified in `formula` are to be taken.
`xlearn`	(Required if no formula is given as the principal argument.) A data frame or matrix containing the explanatory variables.
`ylearn`	(Required if no formula is given as the principal argument.) A numeric vector or factor with numeric levels specifying the class for each observation.
`xtest`	A data frame or matrix of cases to be classified, containing the features used in `formula` or `xlearn`.

mfinal — Maximum number of iterations of the algorithm.

monotone_constraints — Numerical vector consisting of 1, 0 and -1, its length equals the number of features in xlearn. 1 is increasing, -1 is decreasing and 0 is no constraint.

prior — The prior probabilities of class membership. If unspecified, equal prior probabilities are used. If present, the probabilities must be specified in the order of the factor levels.

... — Arguments passed to or from other methods.

Value

A list containing the following components:

call	The (matched) function call.
trainset	Matrix with the training set used (first columns) and the class for each observation (last column).
prior	Prior probabilities of class membership used.
apparent	Apparent error rate.
mfinal	Number of iterations of the algorithm.
loglikelihood	Log-likelihood.
posterior	Posterior probabilities of class membership for xtest set.
class	Labels of the class with maximal probability for xtest set.

Note

This function may be called using either a formula and data frame, or a data frame and grouping variable, or a matrix and grouping variable as the first two arguments. All other arguments are optional.

Classes must be identified, either in a column of data or in the ylearn vector, by natural numbers varying from 1 to the number of classes. The number of classes must be greater than 1.

If there are missing values in either data, xlearn or ylearn, corresponding observations will be deleted.

Author(s)

David Conde

References

Agresti, A. (2010). Analysis of Ordinal Categorical Data, 2nd edition. John Wiley and Sons. New Jersey.

Conde, D., Fernandez, M. A., Rueda, C., and Salvador, B. (2020). Isotonic boosting classification rules. *Advances in Data Analysis and Classification*, 1-25.

See Also

asilb, amilb, csilb

Examples

```
data(motors)
table(motors$condition)
##  1  2  3  4
## 83 67 70 60

## Let us consider the first three variables as predictors
data <- motors[, 1:3]
grouping = motors$condition
##
## Lower values of the amplitudes are expected to be
## related to higher levels of damage severity, so
## we can consider the following monotone constraints
monotone_constraints = rep(-1, 3)

set.seed(7964)
values <- runif(dim(data)[1])
trainsubset <- values < 0.2
obj <- cmilb(data[trainsubset, ], grouping[trainsubset],
               data[-trainsubset, ], 20, monotone_constraints)

## Apparent error
obj$apparent
## 4.761905

## Error rate
100*mean(obj$class != grouping[-trainsubset])
## 15.77061
```

csilb *Cumulative probabilities Simple Isotonic LogitBoost*

Description

Train and predict logitboost-based classification algorithm using isotonic regression (decision stumps for no monotone features) as weak learners, based on the cumulative probabilities logistic model (see Agresti (2010)). For full details on this algorithm, see Conde et al. (2020).

Usage

```
csilb(xlearn, ...)

## S3 method for class 'formula'
csilb(formula, data, ...)

## Default S3 method:
csilb(xlearn, ylearn, xtest = xlearn, mfinal = 100,
monotone_constraints = rep(0, dim(xlearn)[2]), prior = NULL, ...)
```

Arguments

`formula` A formula of the form `groups ~ x1 + x2 + ...`. That is, the response is the class variable and the right hand side specifies the explanatory variables.

`data` Data frame from which variables specified in `formula` are to be taken.

`xlearn` (Required if no formula is given as the principal argument.) A data frame or matrix containing the explanatory variables.

`ylearn` (Required if no formula is given as the principal argument.) A numeric vector or factor with numeric levels specifying the class for each observation.

`xtest` A data frame or matrix of cases to be classified, containing the features used in `formula` or `xlearn`.

`mfinal` Number of iterations of the algorithm.

`monotone_constraints`
Numerical vector consisting of 1, 0 and -1, its length equals the number of features in `xlearn`. 1 is increasing, -1 is decreasing and 0 is no constraint.

`prior` The prior probabilities of class membership. If unspecified, equal prior probabilities are used. If present, the probabilities must be specified in the order of the factor levels.

`...` Arguments passed to or from other methods.

Value

A list containing the following components:

`call` The (matched) function call.

`trainset` Matrix with the training set used (first columns) and the class for each observation (last column).

`prior` Prior probabilities of class membership used.

`apparent` Apparent error rate.

`mfinal` Number of iterations of the algorithm.

`loglikelihood` Log-likelihood.

`posterior` Posterior probabilities of class membership for `xtest` set.

`class` Labels of the class with maximal probability for `xtest` set.

Note

This function may be called using either a formula and data frame, or a data frame and grouping variable, or a matrix and grouping variable as the first two arguments. All other arguments are optional.

Classes must be identified, either in a column of `data` or in the `ylearn` vector, by natural numbers varying from 1 to the number of classes. The number of classes must be greater than 1.

If there are missing values in either `data`, `xlearn` or `ylearn`, corresponding observations will be deleted.

Author(s)

David Conde

References

Agresti, A. (2010). Analysis of Ordinal Categorical Data, 2nd edition. John Wiley and Sons. New Jersey.

Conde, D., Fernandez, M. A., Rueda, C., and Salvador, B. (2020). Isotonic boosting classification rules. *Advances in Data Analysis and Classification*, 1-25.

See Also

asilb, amilb, cmilb

Examples

```
data(motors)
table(motors$condition)
##  1  2  3  4
## 83 67 70 60

## Let us consider the first three variables as predictors
data <- motors[, 1:3]
grouping = motors$condition
##
## Lower values of the amplitudes are expected to be
## related to higher levels of damage severity, so
## we can consider the following monotone constraints
monotone_constraints = rep(-1, 3)

set.seed(7964)
values <- runif(dim(data)[1])
trainsubset <- values < 0.2
obj <- csilb(data[trainsubset, ], grouping[trainsubset],
               data[-trainsubset, ], 100, monotone_constraints)

## Apparent error
obj$apparent
## 4.761905

## Error rate
100*mean(obj$class != grouping[-trainsubset])
## 17.92115
```

motors *Diagnostic of electrical induction motors*

Description

Electrical induction motors are widely used in industry. In the industrual context, the early detection of possible damage in the motor is very important since failures can result in financial losses. Motor Current Signature Analysis is the most widespread technique to diagnose a faulty motor, see Choudhary et al. (2019). This technique is based on the spectral analysis of the stator current: motor faults cause an asymmetry that reflects as additional harmonics in the current spectrum, so side bands around the main frequency are considered and amplitudes of these side bands around odd harmonics are measured.

The data were generated by Oscar Duque and Daniel Morinigo at the Electrical Engineering laboratory of the Universidad de Valladolid.

Four condition states of damage severity are considered: 1 - undamaged, 2 - incipient fault, 3 - moderate damage, 4 - severe damage.

Usage

```
data(motors)
```

Format

A data frame with 280 observations on 7 variables, six are numerical and one nominal defining the condition state of the motors.

[,1]	amplitude_l.1	Amplitude of the first lower side band around harmonic 1
[,2]	amplitude_u.1	Amplitude of the first upper side band around harmonic 1
[,3]	amplitude_l.5	Amplitude of the first lower side band around harmonic 5
[,4]	amplitude_u.5	Amplitude of the first upper side band around harmonic 5
[,5]	amplitude_l.7	Amplitude of the first lower side band around harmonic 7
[,6]	amplitude_u.7	Amplitude of the first upper side band around harmonic 7
[,7]	condition	Condition state

Source

- Creator: Oscar Duque and Daniel Morinigo, Electrical Engineering Department laboratory, Universidad de Valladolid, Valladolid, Spain.

References

Choudhary, A. & Goyal, D. & Shimi, S. L. & Akula, A. (2019). Condition monitoring and fault diagnosis of induction motors: A review. Archives of Computational Methods in Engineering. In press. doi:10.1007/s11831-018-9286-z.

Garcia-Escudero, L. A., Duque-Perez, O., Fernandez-Temprano, M., Morinigo-Sotelo, D. (2016). Robust Detection of Incipient Faults in VSI-Fed Induction Motors Using Quality Control Charts. IEEE Transactions on Industry Applications, 53(3), 3076-3085.

Examples

```
data(motors)
summary(motors)
```

Index