Package ‘fastAdaboost’

August 29, 2016

Type Package
Title a Fast Implementation of Adaboost
Description Implements Adaboost based on C++ backend code. This is blazingly fast and especially useful for large, in memory data sets. The package uses decision trees as weak classifiers. Once the classifiers have been trained, they can be used to predict new data. Currently, we support only binary classification tasks. The package implements the Adaboost.M1 algorithm and the real Adaboost(SAMME.R) algorithm.

Version 1.0.0
Date 2016-02-23
Author Sourav Chatterjee [aut, cre]
Maintainer Sourav Chatterjee <souravc83@gmail.com>
License MIT + file LICENSE

URL https://github.com/souravc83/fastAdaboost

BugReports https://github.com/souravc83/fastAdaboost/issues

Depends R (>= 3.1.2)
Imports Rcpp, rpart
Suggests testthat, knitr, MASS
LazyData yes

LinkingTo Rcpp (>= 0.12.0)

RoxygenNote 5.0.1

NeedsCompilation yes
Repository CRAN

Date/Publication 2016-02-28 09:59:32
R topics documented:

- adaboost ................................................................. 2
- fastAdaboost .......................................................... 3
- get_tree ................................................................. 4
- predict.adaboost ...................................................... 4
- predict.real_adaboost ............................................. 5
- print.adaboost ........................................................ 6
- print.real_adaboost ............................................... 7
- real_adaboost ......................................................... 8

Index 10

<table>
<thead>
<tr>
<th>adaboost</th>
<th>Adaboost.M1 algorithm</th>
</tr>
</thead>
</table>

Description

Implements Freund and Schapire’s Adaboost.M1 algorithm

Usage

adaboost(formula, data, nIter, ...)

Arguments

- formula: Formula for models
- data: Input dataframe
- nIter: no. of classifiers
- ...: other optional arguments, not implemented now

Details

This implements the Adaboost.M1 algorithm for a binary classification task. The target variable must be a factor with exactly two levels. The final classifier is a linear combination of weak decision tree classifiers.

Value

object of class adaboost

References

See Also

real_adaboost, predict.adaboost

Examples

```r
fakedata <- data.frame( X = c(rnorm(100,0,1), rnorm(100,1)), Y = c(rep(0,100), rep(1,100)) )
fakedata$Y <- factor(fakedata$Y)
test_adaboost <- adaboost(Y~X, data=fakedata, 10)
```
get_tree

Fetches a decision tree

Description
returns a single weak decision tree classifier which is part of the strong classifier

Usage
get_tree(object, tree_num)

Arguments
object object of class adaboost
tree_num integer describing the tree to get

Details
returns an individual tree from the adaboost object. This can provide the user with some clarity on
the individual building blocks of the strong classifier

Value
object of class rpart

See Also
adaboost

Examples
fakedata <- data.frame( X=c(rnorm(100,0,1),rnorm(100,1,1)), Y=c(rep(0,100),rep(1,100)) )
fakedata$Y <- factor(fakedata$Y)
test_adaboost <- adaboost(Y~X, fakedata, 10)
tree <- get_tree(test_adaboost,5)

predict.adaboost

predict method for adaboost objects

Description
predictions for model corresponding to adaboost.m1 algorithm

Usage
## S3 method for class 'adaboost'
predict(object, newdata, ...)

**predict.real_adaboost**

Arguments

- **object**: an object of class adaboost
- **newdata**: dataframe on which we are looking to predict
- ... arguments passed to predict.default

Details

makes predictions for an adaboost object on a new dataset. The target variable is not required for the prediction to work. However, the user must ensure that the test data has the same columns which were used as inputs to fit the original model. The error component of the prediction object(as in `pred$error`) can be used to get the error of the test set if the test data is labeled.

Value

predicted object, which is a list with the following components

- **formula**: the formula used.
- **votes**: total weighted votes achieved by each class
- **class**: the class predicted by the classifier
- **prob**: a matrix with predicted probability of each class for each observation
- **error**: The error on the test data if labeled, otherwise NA

See Also

- adaboost

Examples

```r
fakedata <- data.frame( X= c(rnorm(100,0,1), rnorm(100,1,1)), Y= c(rep(0,100), rep(1,100)) )
fakedata$Y <- factor(fakedata$Y)
test_adaboost <- adaboost(Y~X, fakedata, 10)
pred <- predict( test_adaboost, newdata=fakedata)
print(pred$error)
print( table(pred$class, fakedata$Y) )
```

**predict.real_adaboost**  
*predict method for real_adaboost objects*

Description

predictions for model corresponding to real_adaboost algorithm

Usage

```r
## S3 method for class 'real_adaboost'
predict(object, newdata, ...)
```
Arguments

object an object of class real_adaboost
newdata dataframe on which we are looking to predict
... arguments passed to predict.default

Details

makes predictions for an adaboost object on a new dataset using the real_adaboost algorithm. The target variable is not required for the prediction to work. However, the user must ensure that the test data has the same columns which were used as inputs to fit the original model. The error component of the prediction object (as in pred$error) can be used to get the error of the test set if the test data is labeled.

Value

predicted object, which is a list with the following components

formula the formula used.
votes total weighted votes achieved by each class
class the class predicted by the classifier
prob a matrix with predicted probability of each class for each observation
error The error on the test data if labeled, otherwise NA

See Also

real_adaboost

Examples

fakedata <- data.frame( X=c(rnorm(100,0,1),rnorm(100,1,1)), Y=c(rep(0,100),rep(1,100) ) )
fakedata$Y <- factor(fakedata$Y)
test_real_adaboost <- real_adaboost(Y~X, fakedata, 10)
pred <- predict(test_real_adaboost, newdata=fakedata)
print(pred$error)
print( table(pred$class, fakedata$Y) )

Description

S3 method to print an adaboost object
print.real_adaboost

Usage

## S3 method for class 'adaboost'
print(x, ...)  

Arguments

x           object of class adaboost
...          arguments passed to print.default

Details

Displays basic information on the model, such as function call, dependent variable, the number of
trees, and weights assigned to each tree

Value

None

See Also

print.real_adaboost

Examples

fakedata <- data.frame( X=c(rnorm(100,0,1), rnorm(100,1,1)), Y=c(rep(0,100), rep(1,100)) )
fakedata$Y <- factor(fakedata$Y)
test_adaboost <- adaboost(Y~X, fakedata, 10)
print(test_adaboost)

---

print.real_adaboost    Print real adaboost model summary

Description

S3 method to print a real_adaboost object

Usage

## S3 method for class 'real_adaboost'
print(x, ...)  

Arguments

x           object of class real_adaboost
...          arguments passed to print.default
Details

Displays basic information on the model, such as function call, dependent variable and the number of trees

Value

None

See Also

print.adaboost

Examples

```r
fakedata <- data.frame(X=c(rnorm(100,0,1),rnorm(100,1,1)), Y=c(rep(0,100),rep(1,100)))
fakedata$Y <- factor(fakedata$Y)
test_real_adaboost <- real_adaboost(Y~X, fakedata, 10)
print(test_real_adaboost)
```

---

real_adaboost Real Adaboost algorithm

Description

Implements Zhu et al’s real adaboost or SAMME.R algorithm

Usage

```r
real_adaboost(formula, data, nIter, ...)
```

Arguments

- `formula` : Formula for models
- `data` : Input dataframe
- `nIter` : no. of classifiers
- `...` : other optional arguments, not implemented now

Details

This implements Zhu et al’s real adaboost or SAMME.R algorithm. The target variable must be a factor with exactly two levels. The final classifier is a linear combination of weak decision tree classifiers. Real adaboost uses the class probabilities of the weak classifiers to iteratively update example weights. It has been found to have lower generalization errors than adaboost.m1 for the same number of iterations.

Value

Object of class `real_adaboost`
**References**


**See Also**

`adaboost`, `predict.real.adaboost`

**Examples**

```r
fakedata <- data.frame(X=c(rnorm(100,0,1),rnorm(100,1,1)), Y=c(rep(0,100),rep(1,100)) )
fakedata$Y <- factor(fakedata$Y)
test_adaboost <- real_adaboost(Y~X, data=fakedata,10)
```
Index

adaboost, 2, 4, 5, 9
fastAdaboost, 3
fastAdaboost-package (fastAdaboost), 3
get_tree, 4
predict.adaboost, 3, 4
predict.real_adaboost, 5, 9
print.adaboost, 6, 8
print.real_adaboost, 7, 7
real_adaboost, 3, 6, 8