Package 'adabag'

January 2, 2012

Type Package

Title Applies AdaBoost.M1, AdaBoost-SAMME and Bagging

Version 3.0

Date 2011-12-22

Author Alfaro-Cortes, Esteban; Gamez-Martinez, Matias and Garcia-Rubio, Noelia

Maintainer Esteban Alfaro-Cortes <Esteban.Alfaro@uclm.es>

Depends rpart, mlbench

Description This package implements Freund and Schapire's Adaboost.M1 algorithm and Breiman's Bagging algorithm
using classification trees as individual classifiers. Once these classifiers have been trained,they can be used to predict on new data. Also, cross validation predictions can be done.
Since version 2.0 a new function ''margins'' is available to calculate the margins for these classifiers.
Also a higher flexibility is achieved giving access to the ''rpart.control'' argument of ''rpart''.
Four important new features have been introduced on version 3.0, AdaBoost-SAMME (Zhu et al., 2009) is implemented
and a new function ''errorevol'' shows the error of the ensembles as a function of the number of iterations.
In addition, the ensembles can be pruned using the option ''newmfinal'' in the predict.bagging and predict.boosting
functions and the posterior probability of each class for observations can be obtained.

License GPL (>= 2)

LazyLoad yes

Repository CRAN

Date/Publication 2011-12-22 13:37:30

R topics documented:

---

adabag-package *Applies AdaBoost.M1, AdaBoost-SAMME and Bagging*

Description

This package implements Freund and Schapire's Adaboost.M1 algorithm and Breiman's Bagging algorithm using classification trees as individual classifiers. Once these classifiers have been trained, they can be used to predict on new data. Also, cross validation predictions can be done. Since version 2.0 a new function "margins" is available to calculate the margins for these classifiers. Also a higher flexibility is achieved giving access to the "rpart.control" argument of "rpart". Four important new features have been introduced on version 3.0, AdaBoost-SAMME (Zhu et al., 2009) is implemented and a new function "errorevol" shows the error of the ensembles as a function of the number of iterations. In addition, the ensembles can be pruned using the option "newmfinal" in the predict.bagging and predict.boosting functions and the posterior probability of each class for observations can be obtained.

Details

Package:	adabag
Type:	Package
Version:	3.0
Date:	2011-12-14
License:	GPL(>= 2)
LazyLoad:	yes

Author(s)

Author: Esteban Alfaro-Cortes, Matias Gamez-Martinez and Noelia Garcia-Rubio
Maintainer: Esteban Alfaro-Cortes <Esteban.Alfaro@uclm.es>

References

Alfaro, E., Gamez, M. and Garcia, N. (2007): "Multiclass corporate failure prediction by Adaboost.M1". International Advances in Economic Research, Vol 13, 3, pp. 301–312.

Alfaro, E., Garcia, N., Gamez, M. and Elizondo, D. (2008): "Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks". Decision Support Systems, 45, pp. 110–122.

Breiman, L. (1998): "Arcing classifiers". The Annals of Statistics, Vol 26, 3, pp. 801–849.

Freund, Y. and Schapire, R.E. (1996): "Experiments with a new boosting algorithm". In Proceedings of the Thirteenth International Conference on Machine Learning, pp. 148–156, Morgan Kaufmann.

Zhu, J., Zou, H., Rosset, S. and Hastie, T. (2009): "Multi-class AdaBoost". Statistics and Its Interface, 2, pp. 349–360.

See Also

boosting, predict.boosting, boosting.cv, bagging, predict.bagging, bagging.cv, margins errorevol

Examples

```
## rpart library should be loaded
library(rpart)
data(iris)
names(iris)<-c("LS","AS","LP","AP","Especies")
iris.adaboost <- boosting(Especies~LS +AS +LP+ AP, data=iris, boos=TRUE,
mfinal=10)

library(rpart)
data(iris)
names(iris)<-c("LS","AS","LP","AP","Especies")
sub <- c(sample(1:50, 25), sample(51:100, 25), sample(101:150, 25))
iris.bagging <- bagging(Especies ~ ., data=iris[sub,], mfinal=10)
iris.predbagging<- predict.bagging(iris.bagging, newdata=iris[-sub,])
```

---

bagging *Applies the Bagging algorithm to a data set*

Description

Fits the Bagging algorithm proposed by Breiman in 1996 using classification trees as single classifiers.

Usage

```
bagging(formula, data, mfinal = 100, control)
```

Arguments

formula	a formula, as in the lm function.
data	a data frame in which to interpret the variables named in the formula
mfinal	an integer, the number of iterations for which boosting is run or the number of trees to use. Defaults to mfinal=100 iterations.
control	options that control details of the rpart algorithm. See rpart.control for more details.

Details

Unlike boosting, individual classifiers are independent among them in bagging

Value

An object of class bagging, which is a list with the following components:

formula	the formula used.
trees	the trees grown along the iterations.
votes	a matrix describing, for each observation, the number of trees that assigned it to each class.
prob	a matrix describing, for each observation, the posterior probability of each class. These probabilities are calculated using the proportion of votes in the final ensemble.
class	the class predicted by the ensemble classifier.
samples	the bootstrap samples used along the iterations.
importance	returns the relative importance of each variable in the classification task. This measure is the number of times each variable is selected to split.

Author(s)

Esteban Alfaro-Cortes <Esteban.Alfaro@uclm.es>, Matias Gamez-Martinez <Matias.Gamez@uclm.es> and Noelia Garcia-Rubio <Noelia.Garcia@uclm.es>

References

Alfaro, E., Gamez, M. and Garcia, N. (2007): "Multiclass corporate failure prediction by Adaboost.M1". International Advances in Economic Research, Vol 13, 3, pp. 301–312.

Alfaro, E., Garcia, N., Gamez, M. and Elizondo, D. (2008): "Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks". Decision Support Systems, 45, pp. 110–122.

Breiman, L. (1996): "Bagging predictors". Machine Learning, Vol 24, 2, pp.123–140.

Breiman, L. (1998). "Arcing classifiers". The Annals of Statistics, Vol 26, 3, pp. 801–849.

See Also

predict.bagging, bagging.cv

Examples

```
## rpart library should be loaded
library(rpart)
data(iris)
names(iris)<-c("LS","AS","LP","AP","Especies")
lirios.bagging <- bagging(Especies~LS +AS +LP+ AP, data=iris, mfinal=10)

## rpart and mlbench libraries should be loaded
library(rpart)
library(mlbench)
data(BreastCancer)
l <- length(BreastCancer[,1])
sub <- sample(1:l,2*l/3)
BC.bagging <- bagging(Class ~.,data=BreastCancer[,-1],mfinal=25, control=rpart.control(maxdepth=3))
BC.bagging.pred <- predict.bagging(BC.bagging,newdata=BreastCancer[-sub,-1])
BC.bagging.pred$confusion
BC.bagging.pred$error

# Data Vehicle (four classes)
library(rpart)
library(mlbench)
data(Vehicle)
l <- length(Vehicle[,1])
sub <- sample(1:l,2*l/3)
Vehicle.bagging <- bagging(Class ~.,data=Vehicle[sub, ],mfinal=50, control=rpart.control(maxdepth=5))
Vehicle.bagging.pred <- predict.bagging(Vehicle.bagging,newdata=Vehicle[-sub, ])
Vehicle.bagging.pred$confusion
Vehicle.bagging.pred$error
```

`bagging.cv` *Runs v-fold cross validation with Bagging*

Description

The data are divided into v non-overlapping subsets of roughly equal size. Then, `bagging` is applied on `(v-1)` of the subsets. Finally, predictions are made for the left out subsets, and the process is repeated for each of the v subsets.

Usage

```
bagging.cv(formula, data, v = 10, mfinal = 100, control)
```

Arguments

`formula`	a formula, as in the `lm` function.
`data`	a data frame in which to interpret the variables named in `formula`

v	An integer, specifying the type of v-fold cross validation. Defaults to 10. If v is set as the number of observations, leave-one-out cross validation is carried out. Besides this, every value between two and the number of observations is valid and means that roughly every v-th observation is left out.
mfinal	an integer, the number of iterations for which boosting is run or the number of trees to use. Defaults to mfinal=100 iterations.
control	options that control details of the rpart algorithm. See rpart.control for more details.

Value

An object of class bagging.cv, which is a list with the following components:

class	the class predicted by the ensemble classifier.
confusion	the confusion matrix which compares the real class with the predicted one.
error	returns the average error.

Author(s)

Esteban Alfaro-Cortes <Esteban.Alfaro@uclm.es>, Matias Gamez-Martinez <Matias.Gamez@uclm.es> and Noelia Garcia-Rubio <Noelia.Garcia@uclm.es>

References

Alfaro, E., Gamez, M. and Garcia, N. (2007): "Multiclass corporate failure prediction by Adaboost.M1". International Advances in Economic Research, Vol 13, 3, pp. 301–312.

Alfaro, E., Garcia, N., Gamez, M. and Elizondo, D. (2008): "Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks". Decision Support Systems, 45, pp. 110–122.

Breiman, L. (1996): "Bagging predictors". Machine Learning, Vol 24, 2, pp. 123–140.

Breiman, L. (1998). "Arcing classifiers". The Annals of Statistics, Vol 26, 3, pp. 801–849.

See Also

bagging, predict.bagging

Examples

```
## rpart library should be loaded
library(rpart)
data(iris)
names(iris)<-c("LS","AS","LP","AP","Especies")
iris.baggingcv <- bagging.cv(Especies ~ ., v=10, data=iris, mfinal=10,control=rpart.control(maxdepth=3))

data(kyphosis)
kyphosis.baggingcv <- bagging.cv(Kyphosis ~ Age + Number + Start, v=5,
data=kyphosis, mfinal=15)

## rpart and mlbench libraries should be loaded
## Data Vehicle (four classes)
```

```
library(rpart)
library(mlbench)
data(Vehicle)
Vehicle.bagging.cv <- bagging.cv(Class ~.,data=Vehicle,v=5,mfinal=20, control=rpart.control(maxdepth=5))
Vehicle.bagging.cv[-1]
```

`boosting` *Applies the AdaBoost.M1 and SAMME algorithms to a data set*

Description

Fits the AdaBoost.M1 (Freund and Schapire, 1996) and SAMME (Zhu et al., 2009) algorithms using classification trees as single classifiers.

Usage

```
  boosting(formula, data, boos = TRUE, mfinal = 100, coeflearn = 'Breiman',
control)
```

Arguments

`formula`	a formula, as in the `lm` function.
`data`	a data frame in which to interpret the variables named in `formula`.
`boos`	if `TRUE` (by default), a bootstrap sample of the training set is drawn using the weights for each observation on that iteration. If `FALSE`, every observation is used with its weights.
`mfinal`	an integer, the number of iterations for which boosting is run or the number of trees to use. Defaults to `mfinal=100` iterations.
`coeflearn`	if 'Breiman'(by default), `alpha=1/2ln((1-err)/err)` is used. If 'Freund' `alpha=ln((1-err)/err)` is used. In both cases the AdaBoost.M1 algorithm is used and `alpha` is the weight updating coefficient. On the other hand, if coeflearn is 'Zhu' the SAMME algorithm is implemented with `alpha=ln((1-err)/err)+ln(nclasses-1)`
`control`	options that control details of the rpart algorithm. See rpart.control for more details.

Details

AdaBoost.M1 and SAMME are simple generalizations of AdaBoost for more than two classes. In AdaBoost-SAMME the individual trees are required to have an error lower than 1-1/nclasses instead of 1/2 of the AdaBoost.M1

Value

An object of class boosting, which is a list with the following components:

formula	the formula used.
trees	the trees grown along the iterations.
weights	a vector with the weighting of the trees of all iterations.
votes	a matrix describing, for each observation, the number of trees that assigned it to each class, weighting each tree by its alpha coefficient.
prob	a matrix describing, for each observation, the posterior probability of each class. These probabilities are calculated using the proportion of votes in the final ensemble.
class	the class predicted by the ensemble classifier.
importance	returns the relative importance of each variable in the classification task. This measure is the number of times each variable is selected to split.

Author(s)

Esteban Alfaro-Cortes <Esteban.Alfaro@uclm.es>, Matias Gamez-Martinez <Matias.Gamez@uclm.es> and Noelia Garcia-Rubio <Noelia.Garcia@uclm.es>

References

Alfaro, E., Gamez, M. and Garcia, N. (2007): "Multiclass corporate failure prediction by Adaboost.M1". International Advances in Economic Research, Vol 13, 3, pp. 301–312.

Alfaro, E., Garcia, N., Gamez, M. and Elizondo, D. (2008): "Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks". Decision Support Systems, 45, pp. 110–122.

Breiman, L. (1998): "Arcing classifiers". The Annals of Statistics, Vol 26, 3, pp. 801–849.

Freund, Y. and Schapire, R.E. (1996): "Experiments with a new boosting algorithm". In Proceedings of the Thirteenth International Conference on Machine Learning, pp. 148–156, Morgan Kaufmann.

Zhu, J., Zou, H., Rosset, S. and Hastie, T. (2009): "Multi-class AdaBoost". Statistics and Its Interface, 2, pp. 349–360.

See Also

predict.boosting, boosting.cv

Examples

```
## rpart library should be loaded
library(rpart)
data(iris)
names(iris)<-c("LS","AS","LP","AP","Especies")
iris.adaboost <- boosting(Especies~LS +AS +LP+ AP, data=iris, boos=TRUE,
mfinal=10)
```

```
## rpart and mlbench libraries should be loaded
## Comparing the test error of rpart and adaboost.M1
library(rpart)
library(mlbench)
data(BreastCancer)
l <- length(BreastCancer[,1])
sub <- sample(1:l,2*l/3)

BC.rpart <- rpart(Class~.,data=BreastCancer[sub,-1], maxdepth=3)
BC.rpart.pred <- predict(BC.rpart,newdata=BreastCancer[-sub,-1],type="class")
tb <-table(BC.rpart.pred,BreastCancer$Class[-sub])
error.rpart <- 1-(sum(diag(tb))/sum(tb))
tb
error.rpart

BC.adaboost <- boosting(Class ~.,data=BreastCancer[,-1],mfinal=25, coeflearn="Freund", boos=FALSE , control=rpart
BC.adaboost.pred <- predict.boosting(BC.adaboost,newdata=BreastCancer[-sub,-1])
BC.adaboost.pred$confusion
BC.adaboost.pred$error


## Data Vehicle (four classes)
library(rpart)
library(mlbench)
data(Vehicle)
l <- length(Vehicle[,1])
sub <- sample(1:l,2*l/3)
mfinal <- 25
maxdepth <- 5

Vehicle.rpart <- rpart(Class~.,data=Vehicle[sub,],maxdepth=maxdepth)
Vehicle.rpart.pred <- predict(Vehicle.rpart,newdata=Vehicle[-sub, ],type="class")
tb <- table(Vehicle.rpart.pred,Vehicle$Class[-sub])
error.rpart <- 1-(sum(diag(tb))/sum(tb))
tb
error.rpart

Vehicle.adaboost <- boosting(Class ~.,data=Vehicle[sub, ],mfinal=mfinal, coeflearn="Zhu",
control=rpart.control(maxdepth=maxdepth))
Vehicle.adaboost.pred <- predict.boosting(Vehicle.adaboost,newdata=Vehicle[-sub, ])
Vehicle.adaboost.pred$confusion
Vehicle.adaboost.pred$error
```

---

boosting.cv *Runs v-fold cross validation with AdaBoost.M1 or SAMME*

Description

The data are divided into v non-overlapping subsets of roughly equal size. Then, boosting is applied on (v-1) of the subsets. Finally, predictions are made for the left out subsets, and the process is repeated for each of the v subsets.

Usage

```
boosting.cv(formula, data, v = 10, boos = TRUE, mfinal = 100,
 coeflearn = "Breiman", control)
```

Arguments

formula	a formula, as in the lm function.
data	a data frame in which to interpret the variables named in formula
boos	if TRUE (by default), a bootstrap sample of the training set is drawn using the weights for each observation on that iteration. If FALSE, every observation is used with its weights.
v	An integer, specifying the type of v-fold cross validation. Defaults to 10. If v is set as the number of observations, leave-one-out cross validation is carried out. Besides this, every value between two and the number of observations is valid and means that roughly every v-th observation is left out.
mfinal	an integer, the number of iterations for which boosting is run or the number of trees to use. Defaults to mfinal=100 iterations.
coeflearn	if 'Breiman'(by default), alpha=1/2ln((1-err)/err) is used. If 'Freund' alpha=ln((1-err)/err) is used. In both cases the AdaBoost.M1 algorithm is used and alpha is the weight updating coefficient. On the other hand, if coeflearn is 'Zhu' the SAMME algorithm is implemented with alpha=ln((1-err)/err)+ln(nclasses-1)
control	options that control details of the rpart algorithm. See rpart.control for more details.

Value

An object of class boosting.cv, which is a list with the following components:

class	the class predicted by the ensemble classifier.
confusion	the confusion matrix which compares the real class with the predicted one.
error	returns the average error.

Author(s)

Esteban Alfaro-Cortes <Esteban.Alfaro@uclm.es>, Matias Gamez-Martinez <Matias.Gamez@uclm.es> and Noelia Garcia-Rubio <Noelia.Garcia@uclm.es>

References

Alfaro, E., Gamez, M. and Garcia, N. (2007): "Multiclass corporate failure prediction by Adaboost.M1". International Advances in Economic Research, Vol 13, 3, pp. 301–312.

Alfaro, E., Garcia, N., Gamez, M. and Elizondo, D. (2008): "Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks". Decision Support Systems, 45, pp. 110–122.

Breiman, L. (1998): "Arcing classifiers". The Annals of Statistics, Vol 26, 3, pp. 801–849.

Freund, Y. and Schapire, R.E. (1996): "Experiments with a new boosting algorithm". In Proceedings of the Thirteenth International Conference on Machine Learning, pp. 148–156, Morgan Kaufmann.

Zhu, J., Zou, H., Rosset, S. and Hastie, T. (2009): "Multi-class AdaBoost". Statistics and Its Interface, 2, pp. 349–360.

See Also

boosting, predict.boosting

Examples

```
## rpart library should be loaded
library(rpart)
data(iris)
names(iris)<-c("LS","AS","LP","AP","Especies")
iris.boostcv <- boosting.cv(Especies ~ ., v=10, data=iris, mfinal=10, control=rpart.control(cp=0.01))
iris.boostcv[-1]


data(kyphosis)
kyphosis.boostcv <- boosting.cv(Kyphosis ~ Age + Number + Start, data=kyphosis, v=5,
mfinal=15)
kyphosis.boostcv[-1]

## rpart and mlbench libraries should be loaded
## Data Vehicle (four classes)
library(rpart)
library(mlbench)
data(Vehicle)
Vehicle.boost.cv <- boosting.cv(Class ~.,data=Vehicle,v=5, mfinal=20, coeflearn="Zhu", control=rpart.control(max
Vehicle.boost.cv[-1]
```

`errorevol` *Shows the error evolution of the ensemble*

Description

Calculates the error evolution of an AdaBoost.M1, AdaBoost-SAMME or Bagging classifier for a data frame as the ensemble size grows

Usage

```
errorevol(object, newdata)
```

Arguments

object	This object must be the output of one of the functions bagging or boosting. This is assumed to be the result of some function that produces an object with two components named formula and trees, as those returned for instance by the bagging function.
newdata	Could be the same data frame used in object or a new one

Details

This can be useful to see how fast Bagging, boosting reduce the error of the ensemble. in addition, it can detect the presence of overfitting and, therefore, the convenience of pruning the ensemble using predict.bagging or predict.boosting.

Value

An object of class errorevol, which is a list with only one component:

error	a vector with the error evolution.

Author(s)

Esteban Alfaro-Cortes <Esteban.Alfaro@uclm.es>, Matias Gamez-Martinez <Matias.Gamez@uclm.es> and Noelia Garcia-Rubio <Noelia.Garcia@uclm.es>

References

Alfaro, E., Gamez, M. and Garcia, N. (2007): "Multiclass corporate failure prediction by Adaboost.M1". International Advances in Economic Research, Vol 13, 3, pp. 301–312.

Alfaro, E., Garcia, N., Gamez, M. and Elizondo, D. (2008): "Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks". Decision Support Systems, 45, pp. 110–122.

Breiman, L. (1996): "Bagging predictors". Machine Learning, Vol 24, 2, pp.123–140.

Freund, Y. and Schapire, R.E. (1996): "Experiments with a new boosting algorithm". In Proceedings of the Thirteenth International Conference on Machine Learning, pp. 148–156, Morgan Kaufmann.

Zhu, J., Zou, H., Rosset, S. and Hastie, T. (2009): "Multi-class AdaBoost". Statistics and Its Interface, 2, pp. 349–360.

See Also

boosting, predict.boosting, bagging, predict.bagging,

Examples

```
data(iris)
train <- c(sample(1:50, 25), sample(51:100, 25), sample(101:150, 25))

cntrl<-rpart.control(maxdepth=1)
iris.adaboost <- boosting(Species ~ ., data=iris[train,], mfinal=50, control=cntrl)

#Error evolution along the iterations in training set
errorevol(iris.adaboost,iris[train,])->evol.train
plot(evol.train$error, type="l", main="Adaboost error Vs number of trees",  col = "blue")

#comparing error evolution in training and test set
errorevol(iris.adaboost,iris[-train,])->evol.test
plot(evol.test$error, type="l", ylim=c(0,1),  main="Adaboost error Vs number of trees",
xlab="Iterations", ylab="Error", col = "red")
lines(evol.train$error, cex = .5 ,col="blue", lty=2)
legend("topright", c("test","train"), col = c("red", "blue"), lty=1:2)


data(BreastCancer)
l <- length(BreastCancer[,1])
sub <- sample(1:l,2*l/3)
cntrl <- rpart.control(maxdepth = 3, minsplit = 0,  cp = -1)

BC.bagging <- bagging(Class ~.,data=BreastCancer[sub,-1],mfinal=75, control=cntrl)

errorevol(BC.bagging,newdata=BreastCancer[sub,-1])->evol.train
errorevol(BC.bagging,newdata=BreastCancer[-sub,-1])->evol.test

par(mfrow = c(1, 2))
plot(evol.train$error, type="l", main="Bagging training error Vs number of trees",
xlab="Iterations", ylab="Error", ylim=c(0,0.5), col = "blue")
plot(evol.test$error, type="l", main="Bagging test error Vs number of trees",
xlab="Iterations", ylab="Error", ylim=c(0,0.5), col = "red")


data(Vehicle)
l <- length(Vehicle[,1])
sub <- sample(1:l,2*l/3)

Vehicle.adaboost <- boosting(Class ~.,data=Vehicle[sub, ],mfinal=100, coeflearn="Zhu",
control=rpart.control(maxdepth=5))

errorevol(Vehicle.adaboost,newdata=Vehicle[sub, ])->evol.train
errorevol(Vehicle.adaboost,newdata=Vehicle[-sub, ])->evol.test

#comparing error evolution in training and test set
plot(evol.test$error, type="l",   main="Adaboost error Vs number of trees",
xlab="Iterations", ylab="Error", col = "red")
lines(evol.train$error, cex = .5 ,col="blue", lty=2)
```

```
legend("topright", c("test","train"), col = c("red", "blue"), lty=1:2)
```

`margins` *Calculates the margins*

Description

Calculates the margins of an AdaBoost.M1, AdaBoost-SAMME or Bagging classifier for a data frame

Usage

```
margins(object, newdata)
```

Arguments

`object`	This object must be the output of one of the functions `bagging`, `boosting`, `predict.bagging` or `predict.boosting`. This is assumed to be the result of some function that produces an object with two components named `formula` and `class`, as those returned for instance by the `bagging` function.
`newdata`	The same data frame used for building the `object`

Details

Intuitively, the margin for an observation is related to the certainty of its classification. It is calculated as the difference between the support of the correct class and the maximum support of an incorrect class

Value

An object of class `margins`, which is a list with only one component:

`margins`	a vector with the margins.

Author(s)

Esteban Alfaro-Cortes <Esteban.Alfaro@uclm.es>, Matias Gamez-Martinez <Matias.Gamez@uclm.es> and Noelia Garcia-Rubio <Noelia.Garcia@uclm.es>

References

Alfaro, E., Gamez, M. and Garcia, N. (2007): "A Boosting approach for corporate failure prediction". Applied Intelligence. 27, pp. 29–37.

Alfaro, E., Garcia, N., Gamez, M. and Elizondo, D. (2008): "Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks". Decision Support Systems, 45, pp. 110–122.

Schapire, R.E., Freund, Y., Bartlett, P. and Lee, W.S. (1998): "Boosting the margin: A new explanation for the effectiveness of voting methods". The Annals of Statistics, vol 26, 5, pp. 1651–1686.

See Also

boosting, predict.boosting, bagging, predict.bagging,

Examples

```
#Iris example
library(rpart)
data(iris)
names(iris)<-c("LS","AS","LP","AP","Especies")
sub <- c(sample(1:50, 25), sample(51:100, 25), sample(101:150, 25))
iris.adaboost <- boosting(Especies ~ ., data=iris[sub,], mfinal=10)
margins(iris.adaboost,iris[sub,])->iris.margins # training set

# test set
iris.predboosting<- predict.boosting(iris.adaboost, newdata=iris[-sub,])
margins(iris.predboosting,iris[-sub,])->iris.predmargins


#Examples with bagging
iris.bagging <- bagging(Especies ~ ., data=iris[sub,], mfinal=10)
margins(iris.bagging,iris[sub,])->iris.baggging.margins # training set


iris.predbagging<- predict.bagging(iris.bagging, newdata=iris[-sub,])
margins(iris.predbagging,iris[-sub,])->iris.baggging.predmargins # test set

# We change the name to the margins object to save space
# and use the same code for the graph
#iris.margins<-iris.predmargins
#iris.margins<-iris.baggging.margins
#iris.margins<-iris.baggging.predmargins

# Example of a graph using the margins
par(bg="lightyellow")

# 1 Rotulos en castellano, con disculpas por la falta de acentos por el codigo
plot(sort(iris.margins[[1]]), (1:length(iris.margins[[1]]))/length(iris.margins[[1]]),
type="l", xlim=c(-1,1),main="Grafico de la distribucion de los margenes", xlab="m",
ylab="% conjunto entrenamiento", col="blue3")
abline(v=0, col="red",lty=2)

# 2 English headings
plot(sort(iris.margins[[1]]), (1:length(iris.margins[[1]]))/length(iris.margins[[1]]),
type="l", xlim=c(-1,1),main="Margin cumulative distribution graph", xlab="m",
ylab="% training set", col="blue3")
abline(v=0, col="red",lty=2)


## rpart and mlbench libraries should be loaded
library(rpart)
library(mlbench)
```

```
data(BreastCancer)
l <- length(BreastCancer[,1])
sub <- sample(1:l,2*l/3)

BC.adaboost <- boosting(Class ~.,data=BreastCancer[sub,-1],mfinal=25, control=rpart.control(maxdepth=3))
BC.adaboost.pred <- predict.boosting(BC.adaboost,newdata=BreastCancer[-sub,-1])

BC.margins<-margins(BC.adaboost,BreastCancer[sub,-1]) # training set
BC.predmargins<-margins(BC.adaboost.pred,BreastCancer[-sub,-1]) # test set
```

`predict.bagging` *Predicts from a fitted Bagging object*

Description

Classifies a dataframe using a fitted bagging object.

Usage

```
## S3 method for class 'bagging'
predict(object, newdata, newmfinal=length(object$trees), ...)
```

Arguments

`object`	fitted model object of class `bagging`. This is assumed to be the result of some function that produces an object with the same named components as that returned by the `bagging` function.
`newdata`	data frame containing the values at which predictions are required. The predictors referred to in the right side of `formula(object)` must be present by name in `newdata`.
`newmfinal`	The number of trees of the bagging object to be used in the prediction. This argument allows the user to prune the ensemble. By default all the trees in the bagging object are used newmfinal=length(object$trees)
`...`	further arguments passed to or from other methods.

Value

An object of class `predict.bagging`, which is a list with the following components:

`formula`	the formula used.
`votes`	a matrix describing, for each observation, the number of trees that assigned it to each class.

prob	a matrix describing, for each observation, the posterior probability of each class. These probabilities are calculated using the proportion of votes in the final ensemble.
class	the class predicted by the ensemble classifier.
confusion	the confusion matrix which compares the real class with the predicted one.
error	returns the average error.

Author(s)

Esteban Alfaro-Cortes <Esteban.Alfaro@uclm.es>, Matias Gamez-Martinez <Matias.Gamez@uclm.es> and Noelia Garcia-Rubio <Noelia.Garcia@uclm.es>

References

Alfaro, E., Gamez, M. and Garcia, N. (2007): "Multiclass corporate failure prediction by Adaboost.M1". International Advances in Economic Research, Vol 13, 3, pp. 301–312.

Alfaro, E., Garcia, N., Gamez, M. and Elizondo, D. (2008): "Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks". Decision Support Systems, 45, pp. 110–122.

Breiman, L. (1996): "Bagging predictors". Machine Learning, Vol 24, 2, pp. 123–140.

Breiman, L. (1998). "Arcing classifiers". The Annals of Statistics, Vol 26, 3, pp. 801–849.

See Also

bagging, bagging.cv

Examples

```
library(rpart)
data(iris)
names(iris)<-c("LS","AS","LP","AP","Especies")
sub <- c(sample(1:50, 25), sample(51:100, 25), sample(101:150, 25))
iris.bagging <- bagging(Especies ~ ., data=iris[sub,], mfinal=10)
iris.predbagging<- predict.bagging(iris.bagging, newdata=iris[-sub,])
iris.predbagging

## rpart and mlbench libraries should be loaded
library(rpart)
library(mlbench)
data(BreastCancer)
l <- length(BreastCancer[,1])
sub <- sample(1:l,2*l/3)
BC.bagging <- bagging(Class ~.,data=BreastCancer[,-1],mfinal=25, control=rpart.control(maxdepth=3))
BC.bagging.pred <- predict.bagging(BC.bagging,newdata=BreastCancer[-sub,-1])
BC.bagging.pred$prob
BC.bagging.pred$confusion
BC.bagging.pred$error

# Data Vehicle (four classes)
library(rpart)
```

```
library(mlbench)
data(Vehicle)
l <- length(Vehicle[,1])
sub <- sample(1:l,2*l/3)
Vehicle.bagging <- bagging(Class ~.,data=Vehicle[sub, ],mfinal=50, control=rpart.control(minsplit=15))
#Using the pruning option
Vehicle.bagging.pred <- predict.bagging(Vehicle.bagging,newdata=Vehicle[-sub, ], newmfinal=30)
Vehicle.bagging.pred$confusion
Vehicle.bagging.pred$error
```

`predict.boosting` *Predicts from a fitted boosting object*

Description

Classifies a dataframe using a fitted boosting object.

Usage

```
## S3 method for class 'boosting'
predict(object, newdata, newmfinal=length(object$trees), ...)
```

Arguments

`object`	fitted model object of class `boosting`. This is assumed to be the result of some function that produces an object with the same named components as that returned by the `boosting` function.
`newdata`	data frame containing the values at which predictions are required. The predictors referred to in the right side of `formula(object)` must be present by name in newdata.
`newmfinal`	The number of trees of the boosting object to be used in the prediction. This argument allows the user to prune the ensemble. By default all the trees in `object` are used
`...`	further arguments passed to or from other methods.

Value

An object of class predict.boosting, which is a list with the following components:

`formula`	the formula used.
`votes`	a matrix describing, for each observation, the number of trees that assigned it to each class, weighting each tree by its `alpha` coefficient.
`prob`	a matrix describing, for each observation, the posterior probability of each class. These probabilities are calculated using the proportion of votes in the final ensemble.

class	the class predicted by the ensemble classifier.
confusion	the confusion matrix which compares the real class with the predicted one.
error	returns the average error.

Author(s)

Esteban Alfaro-Cortes <Esteban.Alfaro@uclm.es>, Matias Gamez-Martinez <Matias.Gamez@uclm.es> and Noelia Garcia-Rubio <Noelia.Garcia@uclm.es>

References

Alfaro, E., Gamez, M. and Garcia, N. (2007): "Multiclass corporate failure prediction by Adaboost.M1". International Advances in Economic Research, Vol 13, 3, pp. 301–312.

Alfaro, E., Garcia, N., Gamez, M. and Elizondo, D. (2008): "Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks". Decision Support Systems, 45, pp. 110–122.

Breiman, L. (1998): "Arcing classifiers". The Annals of Statistics, Vol 26, 3, pp. 801–849.

Freund, Y. and Schapire, R.E. (1996): "Experiments with a new boosting algorithm". En Proceedings of the Thirteenth International Conference on Machine Learning, pp. 148–156, Morgan Kaufmann.

Zhu, J., Zou, H., Rosset, S. and Hastie, T. (2009): "Multi-class AdaBoost". Statistics and Its Interface, 2, pp. 349–360.

See Also

boosting, boosting.cv

Examples

```
## rpart library should be loaded
library(rpart)
data(iris)
names(iris)<-c("LS","AS","LP","AP","Especies")
sub <- c(sample(1:50, 25), sample(51:100, 25), sample(101:150, 25))
iris.adaboost <- boosting(Especies ~ ., data=iris[sub,], mfinal=10)
iris.predboosting<- predict.boosting(iris.adaboost, newdata=iris[-sub,])
iris.predboosting$prob

## rpart and mlbench libraries should be loaded
## Comparing the test error of rpart and adaboost.M1
library(rpart)
library(mlbench)
data(BreastCancer)
l <- length(BreastCancer[,1])
sub <- sample(1:l,2*l/3)

BC.rpart <- rpart(Class~.,data=BreastCancer[sub,-1], maxdepth=3)
BC.rpart.pred <- predict(BC.rpart,newdata=BreastCancer[-sub,-1],type="class")
tb <-table(BC.rpart.pred,BreastCancer$Class[-sub])
error.rpart <- 1-(sum(diag(tb))/sum(tb))
```

```
tb
error.rpart

BC.adaboost <- boosting(Class ~.,data=BreastCancer[sub,-1],mfinal=25, control=rpart.control(maxdepth=3))

#Using the pruning option
BC.adaboost.pred <- predict.boosting(BC.adaboost,newdata=BreastCancer[-sub,-1], newmfinal=15)
BC.adaboost.pred$confusion
BC.adaboost.pred$error


## Data Vehicle (four classes)
library(rpart)
library(mlbench)
data(Vehicle)
l <- length(Vehicle[,1])
sub <- sample(1:l,2*l/3)
mfinal <- 25
maxdepth <- 5

Vehicle.rpart <- rpart(Class~.,data=Vehicle[sub,],maxdepth=maxdepth)
Vehicle.rpart.pred <- predict(Vehicle.rpart,newdata=Vehicle[-sub, ],type="class")
tb <- table(Vehicle.rpart.pred,Vehicle$Class[-sub])
error.rpart <- 1-(sum(diag(tb))/sum(tb))
tb
error.rpart

Vehicle.adaboost <- boosting(Class ~.,data=Vehicle[sub, ],mfinal=mfinal, coeflearn="Zhu",
control=rpart.control(maxdepth=maxdepth))
Vehicle.adaboost.pred <- predict.boosting(Vehicle.adaboost,newdata=Vehicle[-sub, ])
Vehicle.adaboost.pred$confusion
Vehicle.adaboost.pred$error
```

Index