Package 'maboost'

February 20, 2015

Version 1.0-0

Date 2014-11-01

Title Binary and Multiclass Boosting Algorithms

Author Tofigh Naghibi

Depends R(>= 2.10),rpart,C50

Description

Performs binary and multiclass boosting in maximum-margin, sparse, smooth and normal settings as described in ``A Boosting Framework on Grounds of Online Learning'' by T. Naghibi and B. Pfister, (2014).
For further information regarding the algorithms, please refer to http://arxiv.org/abs/1409.7202

Maintainer Tofigh Naghibi <tofigh@gmail.com>

License GPL (>= 2)

Repository CRAN

Date/Publication 2014-11-25 17:03:41

Collate 'maboost-package.r' 'maboost.R' 'maboost.default.R' 'maboost.formula.R' 'maboost.machine.bin.R' 'maboost.machine.multi.R' 'predict.maboost.R' 'print.maboost.R' 'projsplx.R' 'projsplx_k.R' 'summary.maboost.R' 'update.maboost.R' 'varplot.maboost.R'

NeedsCompilation no

R topics documented:

maboost *Binary and Multiclass Boosting Algorithms*

Description

'maboost' is used to fit a variety of stochastic boosting models for binary and multiclass responses as described in *A Boosting Framework on Grounds of Online Learning* by T. Naghibi and B. Pfister, (2014).

Usage

```
maboost(x,...)
## Default S3 method:
maboost(x, y,test.x=NULL,test.y=NULL,breg=c("entrop","l2")
,type=c("normal","maxmargin","smooth","sparse"),C50tree=FALSE,iter=100, nu=1
,bag.frac=0.5,random.feature=TRUE,random.cost=TRUE,smoothfactor=1
,sparsefactor=FALSE,verbose=FALSE,...,na.action=na.rpart)

## S3 method for class 'formula'
maboost(formula, data, ..., subset, na.action=na.rpart)
```

Arguments

x	matrix of descriptors.
y	vector of responses (class labels).
formula	a symbolic description of the model to be fit.
data	dataframe containing variables and a column corresponding to class labels.
test.x	testing matrix of discriptors (optional)
test.y	vector of testing responses (optional)
breg	breg="l2" (default) selects quadratic Bregman divergence and breg="entrop" uses KL-divergence which results in a adaboost-like algorithm (with a different choice of eta).
type	determine the type of the algorithm to be used. Default is running the algorithm in the normal mode. type="maxmargin": it guarantees that the margin of the final hypothesis converges to max-margin (at each round t, it divides eta by t^.5). type="sparse": It uses SparseBoost and only works with breg="l2". It generates sparse weight vectors by projecting the weight vectors onto R+. It can be used for multiclass but it is kind of meaningless since the multiclass setting uses a weight matrix instead of weight vector and increasing the sparsity of this matrix does not result in the sparsity of the weight vector (which is the sum over col. of the weight matrix). type="smooth": flag to start smooth boosting. Only works for breg="l2" and for binary classification. Note that for type="smooth", smoothfactor parameter should also be set, accordingly

C50tree	flag to use C5.0 as the weak classifier. It is only recommended for multiclass setting where rpart maybe too weak to satisfy boostability condition. If it is used, don't forget to set the CF and minCases parameters in C50Control properly
iter	number of boosting iterations to perform. Default = 100.
nu	shrinkage parameter for boosting, default taken as 1. It is multiplied in eta and controls its largeness. Note that in the case of using sparseboost, nu can also be increased to enhance sparsity, at the expense of increasing the risk of divergence
bag.frac	sampling fraction for samples taken out-of-bag. This allows one to use random permutation which improves performance.
random.feature	flag to grow a random forest type trees. If TRUE, at each round a small set of features (num_feat^.5) are selected to grow a tree. It generally speeds up the convergence specially for large data sets and improves the performance.
random.cost	flag to assign random costs to selected features. By assigning random costs (look at cost in rpart.control) to the selected features (if random.forest=TRUE) it tries to decorrelates the trees and usually in combination with random.feature it improves the generalization error.
smoothfactor	an integer between 1 to N (number of examples in data) and have to be set if smooth is TRUE. If smoothfactor=K then examples weights are <=1/K and the final error is <K/N
sparsefactor	When it is true, an explicit l1 norm regularization term is used in the projection step of the algorithm (see [1]) to enhance the sparsity. Default is FALSE to guarantee the convergence of the Sparseboost algorithm. Note that, this parameter can also be set to a numeric value which is directly multiplied in the l1-norm regularization factor (see def. of alpha in [1]).
verbose	flag to output more details about internal parameters, error, num_zero, sum of weights, eta (classifeir coefficient) and max of weights, at each round of boosting. Default is FALSE
subset	an optional vector specifying a subset of observations to be used in the fitting process.
na.action	a function that indicates how to process 'NA' values. Default=na.rpart for rpart and na.pass for C5.0.
...	arguments passed to rpart.control and C50Control. For stumps, use maxdepth=1,cp=-1,minsplit=0 maxdepth controls the depth of trees, and cp controls the complexity of trees. For C5.0 use CF,minCases control the complexity and size of the tree. The smaller the CF is, the less complex the tree and the larger the minCases, the smaller the size of the C5.0 tree

Details

This function directly follows the algorithms listed in *"Boosting on Grounds of Online Learning"*.

When using usage 'maboost(y~.)': data must be in a data frame. Response can have factor or numeric values (preferably factor form). missing values can be present in the descriptor data, whenever na.action is set to any option other than na.pass.

After the model is fit, 'maboost' prints a summary of the function call, the method used for boosting, the number of iterations, the final confusion matrix (observed classification vs predicted classification; labels for classes are same as in response), the error for the training set, and testing, training , and kappa estimates of the appropriate number of iterations.

A summary of this information can also be obtained with the command 'print(x)'.

Corresponding functions (Use help with summary.maboost, predict.maboost, ... varplot.maboost for additional information on these commands):

summary : function to print a summary of the original function call, method used for boosting, number of iterations, final confusion matrix, accuracy, and kappa statistic (a measure of agreement between the observed classification and predicted classification). 'summary' can be used for training, testing, or validation data.

predict : function to predict the response for any data set (train, test, or validation)

varplot.maboost : plot of variables ordered by the variable importance measure (based on improvement).

update : add more trees to the `maboost` object.

Value

`model`	The following items are the different components created by the algorithms: trees: ensemble of rpart or C5.0 trees used to fit the model alpha: the weights of the trees used in the final aggregate model F : F[[1]] corresponds to the training sum, F[[2]]], ... corresponds to testing sums. errs : matrix of errs, training, kappa, testing 1, kappa 1, ... lw : last weights calculated, used by update routine num_zero: a vector of length iter containing the number of zeros in the weight vector at each round.
`fit`	The predicted classification for each observation in the original level of the response.
`call`	The function call.
`nu`	shrinkage parameter
`breg`	The type of maboost performed: '"l2"', '"entrop"'.
`confusion`	The confusion matrix (True value vs. Predicted value) for the training data.
`iter`	The number of boosting iterations that were performed.
`actual`	The original response vector.

Warnings

(a) Choose type="normal" or "maxmargin" for multiclass classification. SmoothBoost do not work in multiclass setting and SparseBoost does not make sense to be used for multiclass classification (where we have to deal with a weight matrix rather than a weight vector).

(b) cost variable in rpart.control is the only variable in rpart.control that CANNOT be set through maboost. It is reserved for random.cost.

Author(s)

Tofigh Naghibi, ETH Zurich

Special thanks to Dr. Mark Culp and his colleagues who developed the 'ada' package. A big part of this package has been built upon their code. In particular, summary, print and varplot.maboost functions are imported from 'ada' package with almost no changes. For further info about 'ada' which implements different variations of Anyboost, look at [2]

References

[1] Naghibi, T., Pfister, B. (2014). *A Boosting Framework on Grounds of Online Learning.* NIPS.

[2] Culp, M., Johnson, K., Michailidis, G. (2006). *maboost: an R Package for Stochastic Boosting* Journal of Statistical Software, 16.

See Also

print.maboost,summary.maboost,predict.maboost ,update.maboost,varplot.maboost

Examples

```
## fit maboost model
data(iris)
##drop setosa
iris[iris$Species!="setosa",]->iris
##set up testing and training data (60% for training)
n<-dim(iris)[1]
trind<-sample(1:n,floor(.6*n),FALSE)
teind<-setdiff(1:n,trind)
iris[,5]<- as.factor((levels(iris[,5])[2:3])[as.numeric(iris[,5])-1])
##fit a tree with maxdepth=6 (a variable pass to rpart.control).
gdis<-maboost(Species~.,data=iris[trind,],iter=50,nu=2
                  ,breg="l2", type="sparse",bag.frac=1,random.feature=FALSE
                  ,random.cost=FALSE, C50tree=FALSE, maxdepth=6,verbose=TRUE)
##to see the average zeros in the weighting vectors over the 40 rounds of boosting
print(mean(gdis$model$num_zero))
##prediction
pred.gdis= predict(gdis,iris,type="class");
##variable selection
varplot.maboost(gdis)
```

`predict.maboost` *Predict a data set using maboost*

Description

`predict` classifies a new set of observations from a previously built classifier. This function will provide either a vector of new classes, class probability estimates, or both.

Usage

```
## S3 method for class 'maboost'
predict(object, newdata = NULL, type = c("class", "prob", "both","F"),n.iter=NULL,...)
```

Arguments

object	object generated by maboost.
newdata	new data set to predict. This data set must be of type 'data.frame'. Default = NULL. When default = NULL, predict produces predictions for the original training set.
type	choice for preditions. type="class" returns the default class labels. type="prob" returns the probability class estimates. type="both" returns both the default class labels and probability class estimates. type="F" returns the ensamble average. This is mainly usefull for the multiclass case.
n.iter	number of iterations to consider for the prediction. By default this is iter from the maboost call (n.iter< iter)
...	other arguments not used by this function.

Details

This function was modeled after predict.rpart and predict.rpart and predict.C5.0.

Value

fit	a vector of fitted responses. Fit will be returned if type="class".
prob	a matrix of class probability estimates. The first column corresponds to the first label in the 'levels' of the response. The second column corresponds to the second label in the 'levels' of the response. Probs are returned whenever type="prob".
both	returns both the vector of fitted responses and class probability estimates. The first element returns the fitted responses and will be labeled as 'class'. The second element returns the class probability estimates and will be labeled as 'prob'.
F	this can be used in the multiclass case.

Note

This function is invoked by the summary S3 generics invoked with an maboost object. If an error occurs in one of the above commands then try using this command directly to track possible errors. Also, the newdata data set must be of type 'data.frame' when invoking summary.

See Also

summary.maboost,print.maboost, update.maboost

print.maboost *Model Information for Ada*

Description

`print` lists the model information and final confusion matrix for submitted data.

Usage

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

Arguments

`x`	object generated by the function `maboost`.
`...`	other arguments not used by this function.

Details

`print` produces a summary of the original function call, method used for boosting, number of iterations, final confusion matrix, error from data used to build the model, and estimates of M.

Note: any object of class `maboost` invokes `print`, when printed to the screen.

Value

No value returned.

See Also

`summary.maboost`,`predict.maboost`,

summary.maboost *Summary of model fit for arbitrary data (test, validation, or training)*

Description

`summary` lists the model information for fitted model and final confusion matrix.

Usage

```
## S3 method for class 'maboost'
summary(object, n.iter=NULL, ...)
```

Arguments

object	object generated by 'maboost'.
n.iter	specific iteration to obtain the training and testing information at.
...	other arguments not used by this function.

Details

summary produces a summary of the original function call, method used for boosting for a specific iteration, accuracy, and kappa statistic (a measure of agreement between the observed classification and predicted classification) for the training data.

In addition, if any other data set (i.e. test or validation) has been incorporated to the maboost object, summary produces analogous information.

See Also

maboost,predict.maboost

---

update.maboost *Add more trees to an maboost object*

Description

maboost.update updates the maboost object to have additional trees given a new number of iterations.

Usage

```
## S3 method for class 'maboost'
update(object, x, y, test.x, test.y = NULL, n.iter, ...)
```

Arguments

object	object generated by the function maboost.
x	x training data
y	training response
test.x	x testing data (optional)
test.y	the true labeling for this testing data (optional)
n.iter	new number of iterations, must be provided and n.iter>iter
...	other arguments not used by this function.

Value

updated maboost object.

See Also

maboost,summary.maboost,predict.maboost,

varplot.maboost *Variable selection with maboost*

Description

varplot.maboost ranks the variable according to their importance.

Usage

```
varplot.maboost(x,plot.it=TRUE, type=c("none","scores"),max.var.show=30,...)
```

Arguments

x	object generated by the function maboost.
plot.it	flag to plot a figure
type	default="none", if scores is selected the scores of variables are returned.
max.var.show	default="30", set it to larger value if you have more than 30 variables.
...	other arguments not used by this function.

Details

varplot.maboost This command provides a sense of variable importance. The more frequently a variable is selected for boosting, the more likely the variable contains useful information for classification. for rpart, the standard criteria explained in (Hastie et al,2001, pg332) is used while for C5.0. the function C5imp is used to calculate the variables importance for each C5.0 model and these scores are averaged over all trees.

Value

scores are returned.

See Also

summary.maboost,predict.maboost,

Index