Package 'BigTSP'

December 20, 2018

Type Package

Title Top Scoring Pair based methods for classification

Version 1.0

Date 2012-08-20

Author Xiaolin Yang,Han Liu

Maintainer Xiaolin Yang <xyang@stat.cmu.edu>

Description This package is trying to implement Top Scoring Pair based methods for classification including LDCA, TSP-tree, TSP-random forest and TSP gradient boosting algorithm.

Depends glmnet,tree,randomForest,gbm

License GPL (>= 2)

Repository CRAN

Date/Publication 2012-08-21 06:34:06

R topics documented:

BigTSP-package *Top Scoring Pair based methods for classification.*

Description

This package is trying to implement Top Scoring Pair based methods for classification including LDCA, TSP-tree, TSP-random forest and TSP gradient boosting algorithm.

Details

Package:	BigTSP
Type:	Package
Version:	1.0
Date:	2012-08-20
License:	GPL(>= 2)

LDCA, tsp.tree, tsp.randomForest, tsp.gbm

Author(s)

Xiaolin Yang, Han Liu

Maintainer: Who to complain to <xyang@stat.cmu.edu> Xiaolin Yang

cv.LDCA *Cross validation for LDCA*

Description

Cross validation for LDCA

Usage

```
cv.LDCA(X, y, lambda = NULL, nfolds)
```

Arguments

X	input matrix, of dimension nobs x nvars; each row is an observation vector.
y	response variable.
lambda	user specified lambda sequence
nfolds	number of folds - default is 10.

Value

an object of class "cv.LDCA" is returned, which is a list with the ingredients of the cross-validation fit.

lambda	the values of lambda used in the fits.
cvm	The mean cross-validated error - a vector of length length(lambda).
cvsd	estimate of standard error of cvm.
cvup	upper curve = cvm+cvsd.
cvlo	lower curve = cvm-cvsd.
nzero	number of non-zero coefficients at each lambda.
name	a text string indicating type of measure (for plotting purposes).
glmnet.fit	a fitted glmnet object for the full data.
lambda.min	value of lambda that gives minimum cvm.
lambda.1se	largest value of lambda such that error is within 1 standard error of the minimum.

Author(s)

Xiaolin Yang, Han Liu

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) *Regularization Paths for Generalized Linear Models via Coordinate Descent*, http://www.stanford.edu/~hastie/Papers/glmnet.pdf
Journal of Statistical Software, Vol. 33(1), 1-22 Feb 2010
http://www.jstatsoft.org/v33/i01/
Simon, N., Friedman, J., Hastie, T., Tibshirani, R. (2011) *Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent, Journal of Statistical Software, Vol. 39(5) 1-13*
http://www.jstatsoft.org/v39/i05/

See Also

print.cv.LDCA,predict.cv.LDCA,

Examples

```
library(glmnet)
x=matrix(rnorm(50*20),50,20)
y=rbinom(50,1,0.5)
cvfit=cv.LDCA(x,y,nfolds=5)
predict(cvfit,x[1:10,],s="lambda.min")
```

LDCA *Linear Discriminant Analysis based on Top Scoring Pair*

Description

Linear Discriminant Analysis based on Top Scoring Pair

Usage

```
LDCA(X,y,nlambda=100,lambda=NULL,threshold=1e-07)
```

Arguments

X	input matrix, of dimension nobs x nvars; each row is an observation vector.
y	response variable.
nlambda	The number of lambda values - default is 100.
lambda	user specified lambda sequence
threshold	Convergence threshold for coordinate descent. A parameter from "glmnet" package. Defaults value is 1E-7.

Value

An object with S3 class "LDCA","glmnet"

call	the call that produced this object
a0	Intercept sequence of length length(lambda)
beta	For "elnet" and "lognet" models, a nvars x length(lambda) matrix of coefficients, stored in sparse column format ("CsparseMatrix"). For "multnet", a list of nc such matrices, one for each class.
lambda	The actual sequence of lambda values used
dev.ratio	The fraction of (null) deviance explained (for "elnet", this is the R-square). The deviance calculations incorporate weights if present in the model. The deviance is defined to be 2*(loglike_sat - loglike), where loglike_sat is the log-likelihood for the saturated model (a model with a free parameter per observation). Hence dev.ratio=1-dev/nulldev.
nulldev	Null deviance (per observation). This is defined to be 2*(loglike_sat -loglike(Null)); The NULL model refers to the intercept model, except for the Cox, where it is the 0 model.
df	The number of nonzero coefficients for each value of lambda. For "multnet", this is the number of variables with a nonzero coefficient for *any* class.
dim	dimension of coefficient matrix (ices)
nobs	number of observations
npasses	total passes over the data summed over all lambda values
offset	a logical variable indicating whether an offset was included in the model
jerr	error flag, for warnings and errors (largely for internal debugging).

Author(s)

Xiaolin Yang, Han Liu

References

Geman, D., dAvignon, C.: Classifying gene expression profiles from pairwise mRNA comparisons. Statistical Applications in Genetics and Molecular Biology, 3(1):19 (2007)

See Also

summary.LDCA,print.LDCA,predict.LDCA,plot.LDCA

Examples

```
library(glmnet)
x=matrix(rnorm(100*20),100,20)
y=rbinom(100,1,0.5)
fit=LDCA(x,y)
print(fit)
predict(fit,newx=x[1:10,]) # make predictions
```

---

predict.cv.LDCA *prediction function for cv.LDCA*

Description

prediction function for cv.LDCA

Usage

```
## S3 method for class 'cv.LDCA'
predict(object, newx, s = c("lambda.lse", "lambda.min"), ...)
```

Arguments

object	a cv.LDCA object
newx	new data matrix
s	lambda value at which the prediction is returned.
...	other arguments

Author(s)

Xiaolin Yang, Han Liu

Examples

```
library(glmnet)
x=matrix(rnorm(50*20),50,20)
y=rbinom(50,1,0.5)
cvfit=cv.LDCA(x,y,nfolds=5)
predict(cvfit,x[1:10,],s="lambda.min")
```

predict.LDCA *predict function for LDCA*

Description

predict function for LDCA

Usage

```
## S3 method for class 'LDCA'
predict(object, newx, s = NULL, type = c("link", "response", "coefficients", "nonzero", "class"), ex
```

Arguments

object	an LDCA object
newx	new data matrix
s	lambda value at which the prediction is returned.
type	Type of prediction required. Type "link" gives the linear predictors for "binomial", "multinomial", "poisson" or "cox" models; for "gaussian" models it gives the fitted values. Type "response" gives the fitted probabilities for "binomial" or "multinomial", fitted mean for "poisson" and the fitted relative-risk for "cox"; for "gaussian" type "response" is equivalent to type "link". Type "coefficients" computes the coefficients at the requested values for s. Note that for "binomial" models, results are returned only for the class corresponding to the second level of the factor response. Type "class" applies only to "binomial" or "multinomial" models, and produces the class label corresponding to the maximum probability. Type "nonzero" returns a list of the indices of the nonzero coefficients for each value of s.
exact	By default (exact=FALSE) the predict function uses linear interpolation to make predictions for values of s that do not coincide with those used in the fitting algorithm. Currently exact=TRUE is not implemented, but prints an error message telling the user how to achieve the exact predictions. This is done my rerunning the algorithm with the desired values interspersed (in order) with the values used in the original fit. This is easily achieved via the R command lamba=sort(c(object$lambda, new.lambda))
offset	If an offset is used in the fit, then one must be supplied for making predictions (except for type="coefficients" or type="nonzero")
...	other arguments.

Author(s)

Xiaolin Yang, Han Liu

Examples

```
library(glmnet)
x=matrix(rnorm(50*20),50,20)
y=rbinom(50,1,0.5)
cvfit=cv.LDCA(x,y,nfolds=5)
predict(cvfit,x[1:10,],s="lambda.min")
```

predict.tsp.gbm *prediction function for tsp.gbm*

Description

prediction function for tsp.gbm

Usage

```
## S3 method for class 'tsp.gbm'
predict(object, newdata, n.trees, type = "link", single.tree = FALSE, ...)
```

Arguments

object	a tsp.gbm object
newdata	new data matrix
n.trees	Number of trees used in the prediction. n.trees may be a vector in which case predictions are returned for each iteration specified
type	The scale on which gbm makes the predictions
single.tree	If single.tree=TRUE then predict.tsp.gbm returns only the predictions from tree(s) n.trees
...	not used.

Author(s)

Xiaolin Yang, Han Liu

References

gbm package

Examples

```
library(gbm)
x=matrix(rnorm(100*20),100,20)
y=rbinom(100,1,0.5)
fit=tsp.gbm(x,y)
predict(fit,x[1:10,],n.trees=5)
```

predict.tsp.randomForest

prediction function for tsp.randomForest

Description

prediction function for tsp.randomForest

Usage

```
## S3 method for class 'tsp.randomForest'
predict(object, newdata, type = "response", norm.votes = TRUE, predict.all = FALSE, proximity = FALS
```

Arguments

object	a tsp.randomForest object
newdata	new data matrix
type	one of response, prob. or votes, indicating the type of output: predicted values, matrix of class probabilities, or matrix of vote counts. class is allowed, but automatically converted to "response", for backward compatibility.
norm.votes	Should the vote counts be normalized (i.e., expressed as fractions)?
predict.all	Should the predictions of all trees be kept?
proximity	Should proximity measures be computed?
nodes	Should the terminal node indicators (an n by ntree matrix) be return? If so, it is in the "nodes" attribute of the returned object.
cutoff	A vector of length equal to number of classes. The 'winning' class for an observation is the one with the maximum ratio of proportion of votes to cutoff.
...	not used.

Author(s)

Xiaolin Yang, Han Liu

References

randomForest package.

Examples

```
library(randomForest)
x=matrix(rnorm(100*20),100,20)
y=rbinom(100,1,0.5)
y=as.factor(y)
fit=tsp.randomForest(x,y)
predict(fit,x[1:10,])
```

predict.tsp.tree *prediction function for tsp.tree*

Description

prediction function for tsp.tree

Usage

```
## S3 method for class 'tsp.tree'
predict(object, newdata, type = c("vector", "tree", "class", "where"), split = FALSE, nwts, eps = 0.0
```

Arguments

object	a tsp.tree object
newdata	new data matrix
type	character string denoting whether the predictions are returned as a vector (default) or as a tsp.tree object.
split	governs the handling of missing values. If false, cases with missing values are dropped down the tree until a leaf is reached or a node for which the attribute is missing, and that node is used for prediction. If split = TRUE cases with missing attributes are split into fractional cases and dropped down each side of the split. The predicted values are averaged over the fractions to give the prediction.
nwts	weights for the newdata cases, used when predicting a tsp.tree.
eps	a lower bound for the probabilities, used if events of predicted probability zero occur in newdata when predicting a tree.
...	other arguments.

Author(s)

Xiaolin Yang, Han Liu

Examples

```
library(tree)
x=matrix(rnorm(100*20),100,20)
y=rbinom(100,1,0.5)
y=as.factor(y)
data=data.frame(y,x)
tr=tsp.tree(x,y)
predict(tr,data[1:10,])
```

print.cv.LDCA *print function for cv.LDCA*

Description

print function for cv.LDCA

Usage

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

Arguments

x	cv.LDCA object
...	other arguments

Author(s)

Xiaolin Yang

Examples

```
library(glmnet)
x=matrix(rnorm(50*20),50,20)
y=rbinom(50,1,0.5)
cvfit=cv.LDCA(x,y,nfolds=5)
print(cvfit)
```

print.LDCA *print the LDCA object*

Description

print the LDCA object

Usage

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

Arguments

x	the LDCA object
...	other arguments.

Author(s)

Xiaolin Yang, Han Liu

Examples

```
library(glmnet)
x=matrix(rnorm(100*20),100,20)
y=rbinom(100,1,0.5)
fit=LDCA(x,y)
print(fit)
```

`tsp.gbm` *Fits generalized boosted logistic regression models based on Top Scoring Pairs.*

Description

Fits generalized boosted logistic regression models based on Top Scoring Pairs.

Usage

```
tsp.gbm(x, y, offset = NULL, misc = NULL, distribution = "bernoulli", w = NULL, var.monotone = NULL, n
```

Arguments

`x`	input matrix, of dimension nobs x nvars; each row is an observation vector.
`y`	response variable.
`offset`	a vector of values for the offset
`misc`	is an R object that is simply passed on to the gbm engine. (refer to "gbm.fit" function in the "gbm" package)
`distribution`	A character string specifying the name of the distribution to use or a list with a component. The default value is "bernoulli" for logistic regression.
`w`	w is a vector of weights of the same length as the y.
`var.monotone`	an optional vector, the same length as the number of predictors, indicating which variables have a monotone increasing (+1), decreasing (-1), or arbitrary (0) relationship with the outcome.
`n.trees`	the total number of trees to fit. This is equivalent to the number of iterations and the number of basis functions in the additive expansion.
`interaction.depth`	The maximum depth of variable interactions. 1 implies an additive model, 2 implies a model with up to 2-way interactions, etc.
`n.minobsinnode`	minimum number of observations in the trees terminal nodes. Note that this is the actual number of observations not the total weight.
`shrinkage`	a shrinkage parameter applied to each tree in the expansion. Also known as the learning rate or step-size reduction.
`bag.fraction`	the fraction of the training set observations randomly selected to propose the next tree in the expansion.
`train.fraction`	The first train.fraction * nrows(data) observations are used to fit the gbm and the remainder are used for computing out-of-sample estimates of the loss function.
`keep.data`	a logical variable indicating whether to keep the data and an index of the data stored with the object.
`verbose`	If TRUE, tsp.gbm will print out progress and performance indicators.

Value

See "gbm" package for returned values

Author(s)

Xiaolin Yang, Han Liu

References

See references for the "gbm" package.

See Also

predict.tsp.gbm

Examples

```
library(gbm)
x=matrix(rnorm(100*20),100,20)
y=rbinom(100,1,0.5)
fit=tsp.gbm(x,y)
predict(fit,x[1:10,],n.trees=5)
```

---

tsp.randomForest *Classification with Random Forest based on Top Scoring Pairs*

Description

Classification with Random Forest based on Top Scoring Pairs

Usage

```
tsp.randomForest(x, y = NULL, xtest = NULL, ytest = NULL, ntree = 500, type = "classification", mtry =
```

Arguments

x	a data frame or a matrix of predictors, or a formula describing the model to be fitted
y	A response vector. If omitted, tsp.randomForest will run in unsupervised mode.
xtest	a data frame or matrix (like x) containing predictors for the test set.
ytest	response for the test set.
ntree	Number of trees to grow.
type	turn on the "classification" mode in "randomForest".
mtry	Number of top scoring pairs randomly sampled as candidates at each split.
replace	Should sampling of cases be done with or without replacement?
classwt	Priors of the classes. Need not add up to one. Ignored for regression.
cutoff	(Classification only) A vector of length equal to number of classes. The 'winning' class for an observation is the one with the maximum ratio of proportion of votes to cutoff. Default is 1/k where k is the number of classes (i.e., majority vote wins).

strata	A (factor) variable that is used for stratified sampling.
sampsize	Size(s) of sample to draw. For classification, if sampsize is a vector of the length the number of strata, then sampling is stratified by strata, and the elements of sampsize indicate the numbers to be drawn from the strata.
nodesize	Minimum size of terminal nodes. Setting this number larger causes smaller trees to be grown (and thus take less time).
maxnodes	Maximum number of terminal nodes trees in the forest can have.
importance	Should importance of top scoring pairs be assessed?
localImp	Should casewise importance measure be computed?
nPerm	Number of times the OOB data are permuted per tree for assessing top scoring pair importance.
proximity	Should proximity measure among the rows be calculated?
oob.prox	Should proximity be calculated only on "out-of-bag" data?
norm.votes	If TRUE (default), the final result of votes are expressed as fractions. If FALSE, raw vote counts are returned (useful for combining results from different runs). Ignored for regression.
do.trace	If set to TRUE, give a more verbose output as randomForest is run. If set to some integer, then running output is printed for every do.trace trees.
keep.forest	If set to FALSE, the forest will not be retained in the output object. If xtest is given, defaults to FALSE.
keep.inbag	Should an n by ntree matrix be returned that keeps track of which samples are "in-bag" in which trees (but not how many times, if sampling with replacement)
...	Additional arguments.

Value

call	the original call to randomForest
type	one of regression, classification, or unsupervised.
predicted	the predicted values of the input data based on out-of-bag samples.
importance	a matrix with nclass + 2 (for classification) or two (for regression) columns. For classification, the first nclass columns are the class-specific measures computed as mean descrease in accuracy. The nclass + 1st column is the mean descrease in accuracy over all classes. The last column is the mean decrease in Gini index. For Regression, the first column is the mean decrease in accuracy and the second the mean decrease in MSE. If importance=FALSE, the last measure is still returned as a vector.
importanceSD	The "standard errors" of the permutation-based importance measure. For classification, a p by nclass + 1 matrix corresponding to the first nclass + 1 columns of the importance matrix. For regression, a length p vector.
localImp	a p by n matrix containing the casewise importance measures, the [i,j] element of which is the importance of i-th variable on the j-th case. NULL if localImp=FALSE.
ntree	number of trees grown.
mtry	number of predictors sampled for spliting at each node.
forest	(a list that contains the entire forest; NULL if randomForest is run in unsupervised mode or if keep.forest=FALSE.

`err.rate`	(classification only) vector error rates of the prediction on the input data, the i-th element being the (OOB) error rate for all trees up to the i-th.
`confusion`	(classification only) the confusion matrix of the prediction (based on OOB data).
`votes`	(classification only) a matrix with one row for each input data point and one column for each class, giving the fraction or number of (OOB) 'votes' from the random forest.
`oob.times`	number of times cases are 'out-of-bag' (and thus used in computing OOB error estimate)
`proximity`	if `proximity=TRUE` when `randomForest` is called, a matrix of proximity measures among the input (based on the frequency that pairs of data points are in the same terminal nodes).
`mse`	(regression only) vector of mean square errors: sum of squared residuals divided by `n`.
`rsq`	(regression only) "pseudo R-squared": 1 - `mse` / Var(y).
`test`	if test set is given (through the `xtest` or additionally `ytest` arguments), this component is a list which contains the corresponding `predicted`, `err.rate`, `confusion`, `votes` (for classification) or `predicted`, `mse` and `rsq` (for regression) for the test set. If `proximity=TRUE`, there is also a component, `proximity`, which contains the proximity among the test set as well as proximity between test and training data.

Author(s)

Xiaolin Yang, Han Liu

References

Breiman, L. (2001), *Random Forests, Machine Learning* Breiman, L. (2002), *"Manual On Setting Up, Using, And Understanding Random Forests V3.1"*, http://oz.berkeley.edu/users/breiman/Using_random_forests_V3.1.pdf.

See Also

`predict.tsp.randomForest`

Examples

```
library(randomForest)
x=matrix(rnorm(100*20),100,20)
y=rbinom(100,1,0.5)
y=as.factor(y)
fit=tsp.randomForest(x,y)
predict(fit,x[1:10,])
plot(fit)
```

tsp.tree *Fit a Classification Tree based on Top Scoring Pairs.*

Description

Fit a Classification Tree based on Top Scoring Pairs.

Usage

```
tsp.tree(X, response, control = tree.control(dim(X)[1], ...), method = "recursive.partition", split
```

Arguments

X	input matrix, of dimension nobs x nvars, each row is an observation vector.
response	response variable.
control	A list as returned by `tree.control`.
method	character string giving the method to use. The only other useful value is `"model.frame"`.
split	Splitting criterion to use.
x	logical. If true, the matrix of variables for each case is returned.
y	logical. If true, the response variable is returned.
wts	logical. If true, the weights are returned.
...	Additional arguments

Value

frame	A data frame with a row for each node, and row.names giving the node numbers. The columns include var, the variable used at the split (or `"<leaf>"` for a terminal node), n, the `(weighted)` number of cases reaching that node, dev the deviance of the node, yval, the fitted value at the node (the mean for regression trees, a majority class for classification trees) and split, a two-column matrix of the labels for the left and right splits at the node. Classification trees also have yprob, a matrix of fitted probabilities for each response level.
where	An integer vector giving the row number of the frame detailing the node to which each case is assigned.
terms	The terms of the formula.
call	The matched call to Tree.
model	If `model = TRUE`, the model frame.
x	If `x = TRUE`, the model matrix.
y	If `y = TRUE`, the response.
wts	If `wts = TRUE`, the weights.

Author(s)

Xiaolin Yang, Han Liu

References

Czajkowski,M., Kretowski, M. (2011) *Top scoring pair decision tree for gene expression data analysis.* Advances in experimental medicine and biology

Breiman L., Friedman J. H., Olshen R. A., and Stone, C. J. (1984) *Classification and Regression Trees.* Wadsworth.

Ripley, B. D. (1996) *Pattern Recognition and Neural Networks.* Cambridge University Press, Cambridge. Chapter 7.

See Also

predict.tsp.tree

Examples

```
library(tree)
x=matrix(rnorm(100*20),100,20)
y=rbinom(100,1,0.5)
y=as.factor(y)
data=data.frame(y,x)
tr=tsp.tree(x,y)
predict(tr,data[1:10,])
plot(tr)
text(tr)
```

Index