Package 'mvpa'

May 11, 2015

Type Package

Title What the package does (short line)

Version 1.0

Date 2015-03-29

Author Ben Smith

Maintainer Who to complain to <benjsmith@gmail.com>

Depends fmri, oro.nifti, doMC, caret, abind

Description More about what it does (maybe more than one line)

License GPL-3

R topics documented:

mvpa-package *Multi-voxel pattern analysis*

Description

Provides shortcuts to streamline MVPA using the R caret package

Details

Package:	mvpa
Type:	Package
Version:	1.0
Date:	2015-03-29
License:	GPL 3.0

This package streamlines testing MVPA solutions in R using caret. caret.train.model.list takes a list of different caret functions and trains the data on each of them, passing the result back in a single list so results can be easily compared.

Author(s)

This package was authored by Ben Smith.

Maintainer: Ben Smith <benjsmith@gmail.com>

align.3D.img.in.2D *Display a 3d image on screen in 3D by tiling it.*

Description

Display a 3d image on screen in 3D by tiling it.

Usage

```
align.3D.img.in.2D(img = NULL, dimension = 1)
```

Arguments

img	the image to put into 3D
dimension	the dimension to reduce across

`caret.train.model.list`

Intended for use with caret; takes a list of items generated by companion function get.caret.model.spec, each of which has one value "method" and one value "tuning" These specify changeable parameters for models It provides a changeable framework so that multiple mthods can all be tested at once There is a list of models here: http://topepo.github.io/caret/modelList.html

---

Description

Intended for use with caret; takes a list of items generated by companion function get.caret.model.spec, each of which has one value "method" and one value "tuning" These specify changeable parameters for models It provides a changeable framework so that multiple mthods can all be tested at once There is a list of models here: http://topepo.github.io/caret/modelList.html

Usage

```
caret.train.model.list(..., trControl, training.list)
```

Arguments

`trControl`	passed directly to train caret; a list of values that define how this function acts. Default value if each item doesn't have its own trControl. See trainControl and http://topepo.github.io/caret/training.html#custom. (NOTE: If given, this argument must be named.)
`x`	passed directly to train caret; an object where samples are in rows and features are in columns. This could be a simple matrix, data frame or other type (e.g. sparse matrix)
`y`	passed directly to train caret; a numeric or factor vector containing the outcome for each sample.
`training.list;`	a list describing a list of train caret values to run. Should be a list of objects generated by get.caret.model.spec. Each should contain exactly two values, to be passed to train caret: method and tuning. If tuning is an integer, it will be passed to tuneLength. If tuning is a data frame, it will be passed to tuneGrid. If it is null, train's default values for tuneLength will apply. Otherwise an error is generated.

`caret.train.model.list.default` *Can take x and y values as the defaults.*

Description

Can take x and y values as the defaults.

Usage

```
## Default S3 method:
caret.train.model.list(x, y, trControl, training.list, ...)
```

Arguments

`x`	passed directly to train caret; an object where samples are in rows and features are in columns. This could be a simple matrix, data frame or other type (e.g. sparse matrix)
`y`	passed directly to train caret; a numeric or factor vector containing the outcome for each sample.
`trControl`	passed directly to train caret; a list of values that define how this function acts. Default value if each item doesn't have its own trControl. See trainControl and http://topepo.github.io/caret/training.html#custom. (NOTE: If given, this argument must be named.)
`training.list;`	a list describing a list of train caret values to run. Should be a list of objects generated by get.caret.model.spec. Each should contain exactly two values, to be passed to train caret: method and tuning. If tuning is an integer, it will be passed to tuneLength. If tuning is a data frame, it will be passed to tuneGrid. If it is null, train's default values for tuneLength will apply. Otherwise an error is generated.

Examples

```
obs <- 500
x.vars <-100
y.vals <- sample(c(1,2),obs,replace = TRUE)
x.vals <- as.data.frame(matrix(rnorm(obs*x.vars,0,1),nrow=obs,ncol=x.vars))
x.vals <- apply(x.vals, 2, function(x.col){return(x.col+y.vals)})
trControl <- trainControl(method="repeatedcv", number=10, repeats=3)
  caret.train.model.list(x.vals
  ,y.vals
  ,trControl
  ,list(
   get.caret.model.spec("svmLinear")
   ,get.caret.model.spec("knn")
 )
 )
```

`caret.train.model.list.formula` *Can take x and y values as the defaults.*

Description

Can take x and y values as the defaults.

Usage

```
## S3 method for class 'formula'
caret.train.model.list(formula, data, trControl,
  training.list, ...)
```

Arguments

`formula`	
`data`	
`trControl`	passed directly to train caret; a list of values that define how this function acts. Default value if each item doesn't have its own trControl. See trainControl and http://topepo.github.io/caret/training.html#custom. (NOTE: If given, this argument must be named.)
`training.list;`	a list describing a list of train caret values to run. Should be a list of objects generated by get.caret.model.spec. Each should contain exactly two values, to be passed to train caret: method and tuning. If tuning is an integer, it will be passed to tuneLength. If tuning is a data frame, it will be passed to tuneGrid. If it is null, train's default values for tuneLength will apply. Otherwise an error is generated.

`createFoldsByGroup` *Intended for use with caret; creates folds based on the group allocations Use in place of createFolds*

Description

Intended for use with caret; creates folds based on the group allocations Use in place of createFolds

Usage

```
createFoldsByGroup(group.allocation)
```

Arguments

`group.allocation`	a vector describing which group/fold each item belongs to.

Examples

```
createFoldsByGroup(c(1,1,1,1,2,2,2,2,3,3,3))
#create 3 folds; two with 4 members each and the third with 3 members.
```

---

display.2D.img *Display a 2d image on screen.*

Description

Display a 2d image on screen.

Usage

```
display.2D.img(ds)
```

Examples

```
display.2D.img(array(rnorm(100^2),dim=c(100, 100)))
```

---

display.3D.img.in.2D *Display a 3d image on screen in 3D by tiling it.*

Description

Display a 3d image on screen in 3D by tiling it.

Usage

```
display.3D.img.in.2D(img, dimension = 3)
```

Examples

```
data <- array(sample(1:100,10^3,replace=TRUE),c(10,10,10))
display.3D.img.in.2D(data,1)
```

`extract.roi` *Extract an ROI from an image timeseries using a mask. Expects an image in the format of NIFTI file. If it's not, we get trouble.*

Description

Extract an ROI from an image timeseries using a mask. Expects an image in the format of NIFTI file. If it's not, we get trouble.

Usage

```
extract.roi(img.ts, roi.mask)
```

Arguments

`img.ts`	a 4D image timeseries from which to extract the mask
`roi.mask`	the roi mask to use to extract the file.

Examples

```
image.filename <- system.file("extdata", "haxby2001subj1bold.nii.gz", package = "mvpa")
mask.filename <- system.file("extdata", "haxby2001subj1mask.nii.gz", package = "mvpa")
fmri.image <- read.image(image.filename)
mask.image <- read.image(mask.filename)
roi.data <- extract.roi(fmri.image,mask.image)
summary(roi.data)
```

`get.caret.model.spec` *Intended for use with caret; specifies a caret model specification for use by companion function caret.train.model.list There is a list of models here: http://topepo.github.io/caret/modelList.html*

Description

Intended for use with caret; specifies a caret model specification for use by companion function caret.train.model.list There is a list of models here: http://topepo.github.io/caret/modelList.html

Usage

```
get.caret.model.spec(method, tuning = NULL, preProcess = NULL, ...)
```

Arguments

method	method to be passed to train caret
tuning	tuning method to be passed to train caret etiher a tuneGrid data frame or an integer to be passed to tuneLength
...	Values that will be passed directly to the function; in caret, these are values that aren't supported as tuning parameters.

Examples

```
get.caret.model.spec("knn",tuning = 5, preProcess = "pca")
#get a knn model spec, with 5 a tuneLength of 5, and use PCA pre-processing.
```

---

load.preprocess.subject

load and pre-process a subject

Description

load and pre-process a subject

Usage

```
load.preprocess.subject(img.path, mask.path, run.path = NULL,
  run.factor.list = NULL)
```

Arguments

run.path	list of the run objects as a file to look up
run.factor.list	should be a data frame, the first column of which is called "labels" and describes the labels applied to each image; the second column called "chunks" and describes any applicable chunks (e.g., runs)
x	fmri timeseries

---

prepropress.fmri.run.set

Apply preprocessing stuff.

Description

Apply preprocessing stuff.

Usage

```
prepropress.fmri.run.set(x, run)
```

Arguments

x	fmri timeseries
run	list of the run objects

`prepropress.haxby.run.set` *Apply preprocessing stuff.*

Description

Apply preprocessing stuff.

Usage

```
prepropress.haxby.run.set(x, run)
```

Arguments

file	
file.format	file format the mri file is stored in. Currently only NIFTI

`read.image` *Read a brain image http://hilaryparker.com/2014/04/29/writing-an-r-package-from-scratch/ This function allows you to read a brain image very nicely*

Description

Read a brain image http://hilaryparker.com/2014/04/29/writing-an-r-package-from-scratch/ This function allows you to read a brain image very nicely

Usage

```
read.image(file, dim, file.format = "NIFTI")
```

Arguments

file	
file.format	file format the mri file is stored in. Currently only NIFTI

`register.machine.cores` *Detects and registers the number of machine cores. This should be run before using cross-validation with caret Will allow R to use the doMC package to utilize the machine's multiple processors. You will need to call this function for it to be applied.*

Description

Detects and registers the number of machine cores. This should be run before using cross-validation with caret Will allow R to use the doMC package to utilize the machine's multiple processors. You will need to call this function for it to be applied.

Usage

```
register.machine.cores()
```

Examples

```
register.machine.cores()
```