graphclass

February 21, 2017

`construct_D` *Constructor for D penalty matrix, use it for efficiency when running the classifier multiple times.*

Description

Constructor for D penalty matrix, use it for efficiency when running the classifier multiple times.

Usage

```
construct_D(nodes = 264)
```

Arguments

`nodes`	Number of nodes in the network, by default is 264 (Power parcellation).

Value

A sparse D matrix

Examples

```
D = construct_D(100)
```

`gc_predict` *Predict function for graph classifier*

Description

Predict function for graph classifier

Usage

```
gc_predict(gc, X, type = "class", Ytest)
```

Arguments

gc	A trained graph classifier object
X	A matrix containing rows with vectorized upper triangular adjacency matrices (column-major order)
type	Indicates the type of response. class: predicted classes. prob: predicted probabilities. error: misclassification error
Ytest	If type = "error", true classes to compare.

Value

A vector containing the predicted classes.

Examples

```
X = matrix(rnorm(100*34453), nrow = 100)
Y = 2*(runif(100) > 0.5) - 1
gc = graphclass(X, Y = Y)
Xtest = matrix(rnorm(100*34453), nrow = 100)
```

---

get_matrix *Returns a matrix from a vectorized network*

Description

Returns a matrix from a vectorized network

Usage

```
get_matrix(beta, type = "intersection")
```

Arguments

beta	Vectorized adjacency matrix.
type	Either intersection for undirected networks, union for directed.

Value

Adjacency matrix for a vectorized network

graphclass *Train a graph classifier using regularized logistic regression.*

Description

Train a graph classifier using regularized logistic regression.

Usage

```
graphclass(X = NULL, Adj_list = NULL, Y = NULL, Xtest = NULL,
  Ytest = NULL, type = "intersection", lambda1 = NULL, lambda2 = NULL,
  lambda = 0, rho = 0, gamma = 1e-05, params = NULL, id = "",
  verbose = F, D = NULL)
```

Arguments

X	A matrix with the training sample, in wich each row represents a vectorized (by column order) upper triangular part of a network.
Adj_list	A list of of symmetric matrices with 0 diagonal for training the classifier
Y	A vector containing the class labels of the training sample (for now only 2 classes are supported).
Xtest	A optional test matrix.
Ytest	Labels of test set.
type	should be either "intersection", "union" or "fusion", only "intersection" is currently supported.
lambda	penalty parameter $lambda$, by default is set to 0.
rho	penalty parameter rho controlling sparsity, by default is set to 0.
gamma	ridge parameter (for numerical purposes).
params	A list containing threshold parameters for the algorithm (see details)
verbose	whether output is printed
D	matrix D of the penalty; precomputing it can save time.

Value

An object containing the trained graph classifier.

Examples

```
X = matrix(rnorm(100*34453), nrow = 100)
Y = 2*(runif(100) > 0.5) - 1
gc = graphclass(X, Y = Y)
gc$train_error
```

plot_adjmatrix *Plot a vectorized adjacency matrix.*

Description

Plot a vectorized adjacency matrix.

Usage

```
plot_adjmatrix(beta, type = "intersection")
```

Arguments

beta	Vectorized adjacency matrix. For undirected networks use only upper triangle in column-major order, for directed use both
type	Either intersection for undirected networks, union for directed.

Examples

```
B = runif(34453)
plot_adjmatrix(B)
```

plot_square_adj_mat *Plot a vectorized adjacency matrix with cells divisions*

Description

Plot a vectorized adjacency matrix with cells divisions

Usage

```
plot_square_adj_mat(edge_values, communities = NULL, type = "real",
  community_labels = c(1:13, -1), main = "", cut_at, sel_cells)
```

Arguments

edge_values	Vectorized adjacency matrix. Only undirected networks are supported for now.
communities	Community of each node
type	Either "real" for valued networks, "prob" for [0,1] valued networks or "prob_cells" for equal value on each cell
community_labels	Name of each community that will appear on the plot.
main	Title of the plot

Index