Package 'SODC'

July 20, 2018

Type Package

Title Optimal Discriminant Clustering(ODC) and Sparse Optimal Discriminant Clustering(SODC)

Version 1.0

Date 2013-05-15

Author Yanhong Wang

Maintainer Yanhong Wang <wangyanhongws@gmail.com>

Description To implement two clustering methods, ODC and SODC, for clustering datasets using optimal scoring, can also be used as an dimension reduction tool.

License GPL-2

Depends magic,ppls,psych,MASS

NeedsCompilation no

Repository CRAN

Date/Publication 2013-05-23 15:53:52

R topics documented:

SODC-package *Sparse Optimal Discriminant Clustering*

Description

To perform Sparse Optimal Discriminant Clustering and Optimal Discriminant Clustering

Details

Package:	SODC
Type:	Package
Version:	1.0
Date:	2013-05-13
License:	GPL-2

Author(s)

Yanhong Wang

Maintainer: Yanhong Wang <wangyanhongws@gmail.com>

References

Yanhong Wang, Yixin Fang, Junhui Wang, Sparse Optimal Discriminant Clustering

Examples

```
# generate a dataset having 150 observations and 10 varialbles,
# the first 2 are informative varialbes,
# the other 8 are noise varialbes.
# This dataset has 3 clusters.
# the returned data has two items, x is the dataset, y is the true clustering assignment.

data.all = my.normdata.gen(2.4,10,150,2)
data = data.all$x

# get kappa coefficient value clus1 and clus2
# clus1 and clus2 are varialbe selection results for input data matrix.
clus1 = c(1,2,3)
clus2 = c(1,2,6)
p = ncol(data)
clust.kappa(clus1, clus2, p)

# perform hierarchical clustering on input data matrix, the desired clusters numbers is 3.
hclust.wrap(data, centers=3)
```

```
# choose optimal lambda2 for a given dataset.
rlt = odc.optimallambda2(data, centers=3, cv.num = 5, lambda2.idx = seq(-3, 3, by = 6/20))
rlt$opt.lambda2

# get ODC component.
rlt.odc = odc.cv(data, k=3, lambda2=0.01)
rlt.odc$Z

#perform ODC clustering, use defualt setting.
rlt.odc.res = odc.clust(data, centers=3)
rlt.odc.res

# get SODC component.
rlt.SODC = my.lasso.classify(data, c=3, lambda1=0.1, lambda2=0.01, tol = 10^(-10), iter.max = 50)
rlt.SODC$Z

# choose optimal lambda1 for a given dataset.
rlt1 = sodc.optimallambda1.boot.all(data, center=3, boot.num = 5, l1.idx = seq(-3, 3, by = 6/10))

# perform SODC clustering given l1 and l2
rlt.sodc.res = sodc.clust(data, centers=3, l1=1, l2=15.8)
rlt.sodc.res
```

`clust.kappa` *Agreement of Two Subset*

Description

To calculate Cohen's kappa coefficients between two subsets.

Usage

```
clust.kappa(clus1, clus2, p)
```

Arguments

`clus1`	The first subset
`clus2`	The second subset
`p`	The total number of varialbes of a dataset

Details

In dimension reduction problems, kappa's coefficients can be used as an criteria to select tuning parameters. In detail, use subset to indicate the selected variables index after applying dimension redcution technoloty on one dataset, we can calculate Cohen's kappa coefficients between two selected variable index subsets corresponding two dataset samples.

Value

ka The Kappa coefficient of the two subsets

hclust.wrap *hierarchical Clustering*

Description

To perform hierarchical clustering

Usage

```
hclust.wrap(x, centers)
```

Arguments

x	Numeric matrix of data,or an object that can be coerced to such a matrix (such as a numeric vector or a data frame with all numeric columns).
centers	An integer scalar or vector with the desired number of groups

Value

hclust.wrap returns a vector with group memberships if centers are scalar, otherwise a matrix with group memberships is returned where each column corresponds to the elements of k, respectively (which are also used as column names).

my.lasso.classify *Sparse Optimal Discriminant Clustering*

Description

To obtain SODC components and other relevant predictions in SODC method.

Usage

```
my.lasso.classify(data, c, lambda1, lambda2, tol = 10^(-10), iter.max = 50)
```

Arguments

data	A numberic dataset matrix.
c	An integer scalar with the desired number of groups.
lambda1	L1 penalty parameter, if lambda1=-1, then odc.clust choose the optimal lambda2 automatically given lambda2.idx. Otherwise perfoms SODC clustering using given lambda1.
lambda2	L2 penalty parameter, if lambda2=-1, then odc.clust choose the optimal lambda2 automatically given lambda2.idx. Otherwise perfoms ODC clustering using given lambda2.
tol	A tolerance value indicating the degree of prediction error of W.
iter.max	the maximum number of iterations allowed.

Value

Z	The SODC component.It is an n by k-1 matrix where n is the number of observations.
varset	An array indicating the selected varialbes index numbers.
what	Predicted W using SODC method.
nvarselected	The number of selected varialbes by SODC.The smaller the value, the sparser.

---

my.normdata.gen *Multipal normal data set with three clusters generation*

Description

To generate multipal normal data set with three clusters

Usage

```
my.normdata.gen(u, p, nobs, nvar)
```

Arguments

u	The parameters to contral the overlap degree among clusters. The cluster centers are (-u,u),(u,u),(u,-u) respectively. The larger u, the clusters more separated
p	Total number of variables
nobs	Number of observations
nvar	Number of informative variables, nvar takes even value;

Details

To generate a multiple normal dataset with three clusters, the first nvar varialbes are informative, and the remaining p-nvar are non-informative variables. The three clusters are separated when u is large, and overlapped when u is small.

Value

x	Multiple normal dataset generated
y	The true cluster assignment for the generated x dataset

`odc.clust` *Optimal Discriminant Clustering*

Description

To perform Optimal Discriminant Clustering

Usage

```
odc.clust(x, centers, cv.num = 5, l2 = -1, clus = kmeans, l2.idx = seq(-3, 3, by = 6/20))
```

Arguments

`x`	A numberic dataset matrix.
`centers`	An integer scalar with the desired number of groups.
`cv.num`	The numbers of cross validation iteration for selecting tuning parameter lambda2.
`l2`	L2 penalty parameter, if l2=-1, then odc.clust choose the optimal lambda2 automatically given lambda2.idx. Otherwise perfoms ODC clustering using given lambda2.
`clus`	The clustering method applied on ODC component.
`l2.idx`	A sequence of index numbers from which one can get a sequence of lambda2 values by calculating 10^l2.idx.

Value

`res`	An object of class "kmeans" or other class depending on the value of argument "clus".
`opt.lambda2`	The optimal lambda2 selected when setting argument l2=-1. This value Will not be returned if setting l2!= -1.

References

Zhang, Z. and Dai, G. (2009). Optimal scoring for unsupervised learning, Advances in Neural Information Processing Systems 23 12: 2241-2249.

odc.cv *Optimal Discriminant Clustering*

Description

To obtain ODC components and other relevant predictions in ODC method.

Usage

```
odc.cv(data, k, lambda2)
```

Arguments

data	A numberic dataset matrix.
k	An integer scalar with the desired number of groups.
lambda2	L2 penalty parameter, if lambda2=-1, then odc.clust choose the optimal lambda2 automatically given lambda2.idx. Otherwise perfoms ODC clustering using given lambda2.

Value

Z	The ODC component.It is an n by k-1 matrix where n is the number of observations.
yhat	Predicted scoring matrix Y using ODC method.
what	Predicted W using ODC method.
s	Predicted S using ODC method.
hnx	The product of an n by n centering matrix and the input data matrix.

odc.optimallambda2 *Prediction and Cross-Validation Selection of Tuning parameter lambda2 in ODC*

Description

To perform cross-validation selection method, for selecting tuning parameter lambda2 in ODC

Usage

```
odc.optimallambda2(data, centers, cv.num = 5, lambda2.idx = seq(-3, 3, by = 6/20))
```

Arguments

data	A numberic dataset matrix.
centers	An integer scalar with the desired number of groups.
cv.num	The numbers of cross validation iteration for selecting tuning parameter lambda2.
lambda2.idx	A sequence of index numbers from which one can get a sequence of lambda2 values by calculating 10^lambda2.idx.

Value

cv.r	An array of numbers indicating the average prediction error corresponding lambda2 array.
sigma	A array of numbers lambda2 from which to choose the optimal lambda2.
opt.lambda2	The chosen optimal lambda2.

---

sodc.clust *Sparse Optimal Discriminant Clustering*

Description

To perform Sparse Optimal Discriminant Clustering

Usage

```
sodc.clust(x, centers, l1 = -1, l2 = -1, cv.num = 5, clus = kmeans, boot.num = 20,
l2.idx = seq(-3, 3, by = 6/20), l1.idx = seq(-3, 3, by = 6/20))
```

Arguments

x	A numberic dataset matrix.
centers	An integer scalar with the desired number of groups.
l1	L1 penalty parameter. The larger lambda1, the sparser result. if l1==-1, will automatically select optimal lambda1.
l2	L2 penalty parameter. if l2==-1, will automatically select optimal lambda2.
cv.num	The numbers of cross validation iteration for selecting tuning parameter lambda2.
clus	The clustering method applied on SODC component.
boot.num	The numbers of bootstrap iteration for selecting tuning parameter lambda1.
l2.idx	A sequence of index numbers from which one can get a sequence of lambda2 values by calculating 10^l2.idx.
l1.idx	A sequence of index numbers from which one can get a sequence of lambda1 values by calculating 10^l1.idx.

Value

cl	An object of class "kmeans" or other class depending on the value of argument "clus" using SODC component.
clvar	An object of class "kmeans" or other class depending on the value of argument "clus" using SODC selected varialbes.
opt.lambda1	optimal lambda1 selected. If l1!=-1, will return opt.lambda1=l1.
opt.lambda2	optimal lambda2 selected. If l2!=-1, will return opt.lambda2=l2.

---

sodc.optimallambda1.boot.all

Prediction and Kappa Selection of Tuning parameter lambda1 in SODC

Description

To perform bootstrap Kappa selection method , for selecting tuning parameter lambda1 in SODC.

Usage

```
sodc.optimallambda1.boot.all(data, centers, boot.num = 20, l1.idx = seq(-3, 3, by = 6/20))
```

Arguments

data	A numberic dataset matrix.
centers	An integer scalar with the desired number of groups.
boot.num	The numbers of bootstrap iteration for selecting tuning parameter lambda1.
l1.idx	A sequence of index numbers from which one can get a sequence of lambda1 values by calculating 10^l1.idx.

Value

boot.r	An array of numbers indicating the average kappa coefficient values corresponding lambda1 array.
boot.s	A matrix indicating the kappa coefficient values, each boot.num-dimension column vector indicates the kappa coefficient value for the coresonding lambda1 value in lambda array.
lambda1	A array of numbers lambda1 from which to choose the optimal lambda1.
opt.lambda1	The chosen optimal lambda1.

Index