Package ‘DA’

October 12, 2022

Version 1.2.0
Date 2021-07-11
Title Discriminant Analysis for Evolutionary Inference
Description Discriminant Analysis (DA) for evolutionary inference (Qin, X. et al, 2020, <doi:10.22541/au.159256808.83862168>), especially for population genetic structure and community structure inference. This package incorporates the commonly used linear and non-linear, local and global supervised learning approaches (discriminant analysis), including Linear Discriminant Analysis of Kernel Principal Components (LDAKPC), Local (Fisher) Linear Discriminant Analysis (LFDA), Local (Fisher) Discriminant Analysis of Kernel Principal Components (LFDAKPC) and Kernel Local (Fisher) Discriminant Analysis (KLFDA). These discriminant analyses can be used to do ecological and evolutionary inference, including demography inference, species identification, and population/community structure inference.

biocViews BiomedicalInformatics, ChIPSeq, Clustering, Coverage, DNAmethylation, DifferentialExpression, DifferentialMethylation, Software, DifferentialSplicing, Epigenetics, FunctionalGenomics, GeneExpression, GeneSetEnrichment, Genetics, ImmunoOncology, MultipleComparison, Normalization, Pathways, QualityControl, RNASeq, Regression, SAGE, Sequencing, Software, SystemsBiology, TimeCourse, Transcription, Transcriptomics

Depends R (>= 3.5)
License GPL-3
SystemRequirements GNU make
BugReports https://github.com/xinghuq/DA/issues
Imports adegenet.lfda, MASS, kernlab, klaR, plotly, rARPACK, grDevices, stats, utils
VignetteBuilder knitr
NeedsCompilation no
RoxygenNote 6.1.1
Suggests knitr, testthat, rmarkdown
**KLFDA**

**Description**

Kernel Local Fisher Discriminant Analysis (KLFDA). This function implements the Kernel Local Fisher Discriminant Analysis with an unified Kernel function. Different from KLFDA function, which adopts the Multinominal Kernel as an example, this function employs the kernel function that allows you to choose various types of kernels. See the kernel function from "kernelMatrix" (kernlab).

**Usage**

```r
KLFDA(x, y, kernel = kernlab::polydot(degree = 1, scale = 1, offset = 1),
       r = 20, tol, prior, CV = FALSE, usekernel = TRUE,
       fL = 0.5, metric = c("weighted", "orthonormalized", "plain"),
       knn = 6, reg = 0.001, ...)
```

**Arguments**

- `x`: The input training data
- `y`: The training labels
- `kernel`: The kernel function used to calculate kernel matrix. Choose the corresponding kernel you want, see details.
- `r`: The number of reduced features you want to keep.
The tolerance used to reject the uni-variance. This is important when the variance between classes is small, and setting the large tolerance will avoid the data distortion.

The weight of each class, or the proportion of each class.

Whether to do cross validation.

whether to use kernel classifier, if TRUE, pass to Naive Bayes classifier.

If usekernel is TRUE, pass to the kernel function.

type of metric in the embedding space (default: 'weighted') 'weighted' - weighted eigenvectors 'orthonormalized' - orthonormalized 'plain' - raw eigenvectors

The number of nearest neighbours

The regularization parameter

additional arguments for the classifier

This function employs three different classifiers, the basic linear classifier, the Naive Bayes (Bayes rule and the Mahalanobis distance), and Naive Bayes classifier. The argument "kernel" in the klfd function is the kernel function used to calculate the kernel matrix. If usekernel is TRUE, the corresponding kernel parameters will pass the the Naive Bayes kernel classifier. The kernel parameter can be set to any function, of class kernel, which computes the inner product in feature space between two vector arguments. kernlab provides the most popular kernel functions which can be initialized by using the following functions:

- rbf: Radial Basis kernel function
- poly: Polynomial kernel function
- vanill: Linear kernel function
- tanh: Hyperbolic tangent kernel function
- laplac: Laplacian kernel function
- bessel: Bessel kernel function
- anova: ANOVA RBF kernel function
- spline: the Spline kernel
  
(see example.)

kernelFast is mainly used in situations where columns of the kernel matrix are computed per invocation. In these cases, evaluating the norm of each row-entry over and over again would cause significant computational overhead.

The results give the classified classes and the posterior possibility of each class using different classifier.

The class labels from linear classifier

The posterior possibility of each class from linear classifier
KLFDA

bayes.judgement
Discrimintion results using the Mabayes classifier

bayes_assignment
Discrimintion results using the Naive bayes classifier

Z
The reduced features

Author(s)
qinxinghu@gmail.com

References


Original Matlab Implementation: http://www.ms.k.u-tokyo.ac.jp/software.html#LFDA


See Also
predict.KLFDA, KLFDA

Examples

```r
require(kernlab)
btest = KLFDA(as.matrix(iris[,1:4]), as.matrix(as.data.frame(iris[,5])),
kernel = kernlab::rbfdot(sigma = 0.1),
r = 3, prior = NULL, tol = 1e-90,
reg = 0.01, metric = 'plain')
pred = predict.KLFDA(btest, testData = as.matrix(iris[1:10,1:4]), prior = NULL)
```
**Description**

This function performs Kernel Local Fisher Discriminant Analysis. The function provided here allows users to carry out the KLFDA using a pairwise matrix. We used the gaussian matrix as example. Users can compute different kernel matrix or distance matrix as the input for this function.

**Usage**

```r
KLFDAM(kdata, y, r,
metric = c("weighted", "orthonormalized", "plain"),
tol=1e-5,knn = 6, reg = 0.001)
```

**Arguments**

- **kdata**: The input dataset (kernel matrix). The input data can be a genotype matrix, dataframe, species occurrence matrix, or principal components. The dataset have to convert to a kernel matrix before feed into this function.
- **y**: The group labels
- **r**: Number of reduced features
- **metric**: Type of metric in the embedding space (default: 'weighted') 'weighted' - weighted eigenvectors 'orthonormalized' - orthonormalized 'plain' - raw eigenvectors
- **knn**: The number of nearest neighbours
- **tol**: Tolerance to avoid singular values
- **reg**: The regularization parameter

**Details**

Kernel Local Fisher Discriminant Analysis for any kernel matrix. It was proposed in Sugiyama, M (2006, 2007) as a non-linear improvement for discriminant analysis. This function is adopted from Tang et al. 2019.

**Value**

- **Z**: The reduced features
- **Tr**: The transformation matrix

**References**


See Also

KLFDA

Examples

kmat <- kmatrixGauss(iris[, -5], sigma=1)
zklfda=KLFDAM(kmat, iris[, 5], r=3, metric = "plain", tol=1e-5)
print(zklfda$Z)

KLFDA_mk

Kernel Local Fisher Discriminant Analysis (KLFDA) with Multinomial kernel

Description

Kernel Local Fisher Discriminant Analysis (KLFDA). This function implements the Kernel Local Fisher Discriminant Analysis with a Multinomial kernel.

Usage

KLFDA_mk(X, Y, r, order, regParam, usekernel = TRUE, fL = 0.5, priors, tol, reg, metric, plotFigures = FALSE, verbose, ...)

Arguments

X The input training data
Y The training labels
r The number of reduced features
order The order passing to Multinomial Kernel
regParam The regularization parameter for kernel matrix
usekernel Whether to used kernel classifier
fL pass to kernel classifier if usekernel is TRUE
priors The weight of each class
tol The tolerance for rejecting uni-variance
reg The regularization parameter
metric Type of metric in the embedding space (default: 'weighted') 'weighted' - weighted eigenvectors 'orthonormalized' - orthonormalized 'plain' - raw eigenvectors
plotFigures whether to plot the reduced features, 3D plot
verbose silence the processing
... additional arguments for the classifier
Details

This function uses Multinomial Kernel, users can replace the Multinomial Kernel based on your own purpose. The final discrimination employs three classifiers, the basic linear classifier, the Mabayes (Bayes rule and the Mahalanobis distance), and Niave Bayes classifier.

Value

class
posterior
bayes_judgement
bayes_assignment
Z

Author(s)
qinxinghu@gmail.com

References


Original Matlab Implementation: http://www.ms.k.u-tokyo.ac.jp/software.html#LFDA


See Also

predict.KLFDA_mk, klfda_1

Examples

btest=KLFDA_mk(X=as.matrix(iris[,1:4]),
Y=as.matrix(as.data.frame(iris[,5])),r=3,order=2,regParam=0.25,
usekernel=TRUE,fL=0.5,
priors=NULL,tol=1e-90,reg=0.01,metric = 'plain',plotFigures=FALSE,
```
verbatim
#pred=predict.KLFDA_mk(btest,as.matrix(iris[1:10,1:4]))
```

**kmatrixGauss**  
*Estimating Gaussian Kernel matrix*

**Description**

This function estimates Gaussian kernel computation for klfda, which maps the original data space to non-linear and higher dimensions. See the details of kmatrixGauss from lfda.

**Usage**

```
kmatrixGauss(x, sigma = 1)
```

**Arguments**

- `x`  
  Input data matrix or dataframe
- `sigma`  
  The Gaussian kernel parameter

**Details**

Return a n*n matrix

**Value**

Return a n*n matrix

**References**


---

**LDAKPC**  
*Linear Fisher discriminant analysis of kernel principal components (DAKPC)*

**Description**

Linear Fisher discriminant analysis of kernel principal components (DAKPC). This function employs the LDA and kpca. This function is called Kernel Fisher Discriminant Analysis (KFDA) in other package (kfda). "KFDA" is the misleading name and "KFDA" has crucial error in package kfda. This function rectifies the current existing error for kfda.
Usage

LDKPC(x, y, n.pc, usekernel = FALSE,
fl = 0, kernel.name = "rbfdot",
kpar = list(0.001), kernel = "gaussian",
threshold = 1e-05, ...)

Arguments

x       Input training data
y       Input labels
n.pc    number of pcs that will be kept in analysis
usekernel Whether to use kernel function, if TRUE, it will pass to the kernel.names
fl      if using kernel, pass to kernel function
kernel.name if usekernel is TRUE, this will take the kernel name and use the parameters set as you defined
kpar    the list of hyper-parameters (kernel parameters). This is a list which contains the parameters to be used with the kernel function. Valid parameters for existing kernels are:
sigma inverse kernel width for the Radial Basis kernel function "rbfdot" and the Laplacian kernel "laplacedot".
degree, scale, offset for the Polynomial kernel "polydot"
scale, offset for the Hyperbolic tangent kernel function "tanhdot"
sigma, order, degree for the Bessel kernel "besseldot".
sigma, degree for the ANOVA kernel "anovadot".
Hyper-parameters for user defined kernels can be passed through the kpar parameter as well.
kernel kernel name if all the above are not used
threshold the threshold for kpc: value of the eigenvalue under which principal components are ignored (only valid when features = 0). (default : 0.0001)
...    additional arguments for the classifier

Value

k pca Results of kernel principal component analysis. Kernel Principal Components Analysis is a nonlinear form of principal component analysis
kpc Kernel principal components. The scores of the components
LDKPC Linear discriminant analysis of kernel principal components
LDs The discriminant function. The scores of the components
label The corresponding class of the data
n.pc Number of Pcs kept in analysis

Author(s)

qinxinghu@gmail.com
References


Examples

```r
data(iris)
train=LDAKPC(iris[,1:4],y=iris[,5],n.pc=3,kernel.name = "rbfdot")
pred=predict.LDAKPC(train,testData = iris[1:10,1:4])
```

LFDA

**Local Fisher Discriminant Analysis (LFDA)**

Description

This function implements local Fisher discriminant analysis. It gives the discriminant function with the posterior possibility of each class.

Usage

```r
LFDA(x, y, r, prior = proportions,
CV = FALSE, usekernel = TRUE, fL = 0,
tol, kernel = "gaussian",
metric = c("orthonormalized", "plain", "weighted"),
knn = 5, ...)
```

Arguments

- **x**: Input training data
- **y**: Training labels
- **r**: Number of reduced features that will be kept
- **prior**: Prior possibility of each class
- **CV**: Whether to do cross validation
- **usekernel**: Whether to use the kernel discrimination in native bayes classifier
- **fL**: Feed to native bayes classifier. Factor for Laplace correction, default factor is 0, i.e. no correction.
- **tol**: The tolerance used in Mabayes discrimination, see Mabayes
- **kernel**: If usekernel is TRUE, specifying the kernel names, see NaiveBaye.
- **metric**: The type of metric in the embedding space (no default), e.g., 'weighted', weighted eigenvectors; 'orthonormalized', orthonormalized; 'plain', raw eigenvectors.
- **knn**: Number of nearest neighbors
- **...**: additional arguments for the classifier
Details

The results give the classified classes and the posterior possibility of each class using different classifier.

Value

class  The class labels
posterior  The posterior possibility of each class
bayes_judgement  Discrimination results using the Mabayes classifier
bayes_assigment  Discrimination results using the Naive bayes classifier
Z  The reduced features

Author(s)

qinxinghu@gmail.com

References


Examples

LFDAtest=LFDA(iris[,1:4],y=iris[,5],r=3,
CV=FALSE,usekernel = TRUE, fL = 0,
kernel="gaussian",metric = "plain",knn = 6,tol = 1)
LFDApred=predict.LFDA(LFDAtest,iris[1:10,1:4],prior=NULL)
Local Fisher Discriminant Analysis of Kernel principle components

Usage

LFDAKPC(x, y, n.pc,
usekernel = FALSE, fL = 0,
kernel.name = "rbfdot",
kpar = list(0.001), kernel = "gaussian",
threshold = 1e-05, ...)

Arguments

x
Input traing data

y
Input labels

n.pc
number of pcs that will be kept in analysis

usekernel
Whether to use kernel function, if TRUE, it will pass to the kernel.names

fL
if using kernel, pass to kernel function

kernel.name
if usekernel is TURE, this will take the kernel name and use the parameters set as you defined

kpar
the list of hyper-parameters (kernel parameters). This is a list which contains the parameters to be used with the kernel function. Valid parameters for existing kernels are:
sigma inverse kernel width for the Radial Basis kernel function "rbfdot" and the Laplacian kernel "laplacedot".
degree, scale, offset for the Polynomial kernel "polydot"
scale, offset for the Hyperbolic tangent kernel function "tanhdot"
sigma, order, degree for the Bessel kernel "besseldot".
sigma, degree for the ANOVA kernel "anovadot".
Hyper-parameters for user defined kernels can be passed through the kpar parameter as well.

kernel
kernel name if all the above are not used

threshold
the threshold for kpc: value of the eigenvalue under which principal components are ignored (only valid when features = 0). (default : 0.0001)

... additional arguments for the classifier
Value

k pca  Results of kernel principal component analysis. Kernel Principal Components Analysis is a nonlinear form of principal component analysis
kpc  Kernel principal components. The scores of the components
LFDAKPC  LOcal linear discriminant analysis of kernel principal components
LDs  The discriminant function. The scores of the components
label  The corresponding class of the data
n.pc  Number of Pcs kept in analysis

Author(s)

qinxinghu@gmail.com

References


Examples

train=LFDAKPC(iris[,1:4],y=iris[,5],tol=1,n.pc=3,kernel.name = "rbfdot")
pred=predict.LFDAKPC(train,prior=NULL,testData = iris[1:10,1:4])

Mabayes  Membership assignment by weighted Mahalanobis distance and bayes rule

Description

The function gives the discrimination of the potential classes based on Bayes rule and the Mahalanobis distance. This function adopts the function from Bingpei Wu, 2012, WMDB 1.0 with some corrections of the judgement rule.

Usage

Mabayes(TrnX, TrnG, p = rep(1, length(levels(TrnG))), TstX = NULL, var.equal = FALSE, tol)
Arguments

TrnX  Training data
TrnG  Training label
p    prior or proportion of each class
TstX  Test data
var.equal  whether the variance or the weight is equal between classes
tol  The threshold or tolerance value for the covariance and distance

Value

posterior and class    The posterior possibility and class labels

Author(s)

qinxinghu@gmail.com

References

Bingpei Wu, 2012, WMDB 1.0: Discriminant Analysis Methods by Weight Mahalanobis Distance and bayes.


Examples

```r
data(iris)
train=Mabayes(iris[,1:4],iris[,5],TstX=iris[1:10,1:4],tol = 1)
predict
```

Description

Predict method for DA.
predict

Usage

## S3 method for class 'KLFDA_mk'
predict(object, prior, testData, ...)
## S3 method for class 'KLFDA'
predict(object, prior, testData, ...)
## S3 method for class 'LDAKPC'
predict(object, prior, testData, ...)
## S3 method for class 'LFDA'
predict(object, prior, testData, ...)
## S3 method for class 'LFDAKPC'
predict(object, prior, testData, ...)

Arguments

- **object**: One of the trained object from discriminant analysis
- **prior**: The weights of the groups.
- **testData**: The test data or new data
- **...**: Arguments passed to the classifiers

Value

The predict function will output the predicted points and their predicted possibility
Index

* KLFDA_mk
  KLFDA_mk, 6
* KLFDA
  KLFDA, 2
* LDAKPC
  LDAKPC, 8
* LFDAKPC
  LFDAKPC, 12
* LFDA
  LFDA, 10
* predict
  predict, 14

KLFDA, 2
KLFDA_mk, 6
KLFDAM, 5
kmatrixGauss, 8

LDAKPC, 8
LFDA, 10
LFDAKPC, 12

Mabayes, 13

predict, 14