

Package ‘transDA’

June 9, 2025

Type Package
Title Transformation Discriminant Analysis
Date 2025-05-18
Version 1.0.1
Maintainer Jing Li <jli178@crimson.ua.edu>
Author Jing Li [aut, cre],
Yana Melnykov [aut]
Description Performs transformation discrimination analysis and non-transformation discrimination analysis. It also includes functions for Linear Discriminant Analysis, Quadratic Discriminant Analysis, and Mixture Discriminant Analysis. In the context of mixture discriminant analysis, it offers options for both common covariance matrix (common sigma) and individual covariance matrices (uncommon sigma) for the mixture components.
License GPL (>= 2)
Encoding UTF-8
Repository CRAN
ByteCompile true
Imports mvtnorm
Config/testthat/edition 3
RoxygenNote 7.3.1
NeedsCompilation no
Date/Publication 2025-06-09 18:10:02 UTC

Contents

transDA-package	2
predict.tda	3
summary.tda	4
tda	4
Index	7

transDA-package

Transformation Discriminant Analysis

Description

Transformation and mixture discriminant analysis.

Details

Package:	transDA
Type:	Package
Version:	1.0.0
Date:	2024-6-10
License:	GPL (>= 2)
LazyLoad:	no

Function 'tda' is the main function of the package that allows running traditional linear and quadratic discriminant procedures as well as more flexible options such as mixture discriminant analysis, transformation discriminant analysis, or their combination.

Function 'predict.tda' makes classification predictions for a provided testing data set.

Author(s)

Jing Li and Yana Melnykov

Maintainer: Jing Li <jli178@crimson.ua.edu>

References

Hastie, T., & Tibshirani, R. (1996). *Discriminant analysis by Gaussian mixtures*. Journal of the Royal Statistical Society Series B: Statistical Methodology, 58(1), 155-176.

Yana Melnykov & Jing Li. (2024) *Transformation Discriminant Analysis*. (under review)

See Also

[tda](#), [predict.tda](#)

Examples

```
set.seed(123)
data(iris)
MDA <- tda(x = iris[,1:4], max_k = 2, ID = iris$Species, trans = FALSE)
summary(MDA)
```

predict.tda	<i>Classification of new observations based on the model obtained by function 'tda'</i>
-------------	---

Description

Predicts class memberships for new observations based on the model returned by function 'tda'.

Usage

```
## S3 method for class 'tda'  
predict(object, newdata, ...)
```

Arguments

object	An object of class 'tda' produced by function tda
newdata	A data frame or matrix with data to be classified
...	Additional arguments that are passed to or from other methods

Value

classification	A factor containing predicted class labels for newdata
Z	A matrix of posterior probabilities for observations from newdata

See Also

[tda](#)

Examples

```
set.seed(123)  
split_indices <- sample(nrow(iris), size = round(0.8 * nrow(iris)), replace = FALSE)  
split_indices <- sort(split_indices)  
traindata <- iris[split_indices,]  
testdata <- iris[-split_indices,]  
MDA <- tda(x = traindata[,1:4], max_k = 2, ID = traindata$Species, trans = FALSE)  
pred <- predict(MDA, testdata[, -5]); pred
```

summary.tda	<i>Summarizing discriminant analysis results from function 'tda'</i>
-------------	--

Description

Summary for the results of discriminant analysis obtained by function 'tda'

Usage

```
## S3 method for class 'tda'
summary(object, ...)
## S3 method for class 'summary.tda'
print(x, ...)
```

Arguments

object	An object of class 'tda' produced by function 'tda'
x	An object of class 'summary.tda' obtained by function 'summary.tda'
...	Additional arguments that are passed to or from other methods

Value

Function 'summary.tda' calculates and returns a list of summary statistics for the model estimated by function 'tda'

See Also

[tda](#)

Examples

```
set.seed(123)
model <- tda(x = iris[,1:4], max_k = 2, ID = iris$Species, trans = FALSE)
summary(model)
```

tda	<i>Transformation Discriminant Analysis</i>
-----	---

Description

Implements discriminant analysis methods including traditional linear (LDA), quadratic (QDA), transformation (TDA), mixture (MDA) discriminant analysis, and their combinations such as TQDA or TLMDA. The user chooses a specific method by specifying options for common or varying transformation parameters as well as covariance matrices.

Usage

```
tda(x, max_k, ID, trans = TRUE, common_lambda = FALSE,
    common_sigma = FALSE, iter = 50, subgroup = NULL,
    tol= 0.001, lambda0 = 0.015)
```

Arguments

x	A frame or matrix containing a training data set
max_k	The maximum number of mixture components within each class to be fitted
ID	A variable containing class memberships for all observations
trans	A transformation indicator: 'trans = TRUE' if transformation is needed, 'trans = FALSE' if transformation is not needed
common_lambda	A parameter that regulates transformations. If 'common_lambda = TRUE', each mixture component or class has the same transformation parameter vector. If 'common_lambda = FALSE', each component or class has a different transformation vector
common_sigma	A homoscedasticity parameter: if 'common_sigma = TRUE', all subgroups across all classes have a common covariance matrix, if 'common_sigma = FALSE', groups have varying covariance matrices
iter	A maximum number of iterations of the EM algorithm; the default value is 50
subgroup	A vector containing the number of mixture components per each class to be fitted
tol	Tolerance level for a stopping critetion based on the relative difference in two consecutive log-likelihood values
lambda0	Starting value for transformation parameters

Value

BIC	Values of the Bayesian Information Criterion calculated for each evaluated model
subprior	Estimated component priors for each class
mu	Estimated component means for each class
sigma	Estimated component covariance matrices for each group
lambda	Estimated transformation parameters
loglik	The log-likelihood value for the model with the lowest BIC
pred_ID	Estimated classification of observations in the training data set
prior	Estimated class priors
misclassification_rate	Misclassification rate for the training data set
ARI	Adjusted Rand index value
Z	Matrix of posterior probabilities for the training data set

See Also

[summary.tda](#), [predict.tda](#)

Examples

```
set.seed(123)
# Example 1:
MDA <- tda(x = iris[,1:4], max_k = 2, ID = iris$Species, trans = FALSE)
print(MDA)
summary(MDA)

# Example 2:
LDA <- tda(x = iris[,1:4], max_k = 1, ID = iris$Species, trans = FALSE,
           common_sigma = TRUE)
print(LDA)
summary(LDA)

# Example 3:
QDA <- tda(x = iris[,1:4], subgroup = c(1, 1, 1), ID = iris$Species,
           trans = FALSE, common_sigma = FALSE)
print(QDA)
summary(QDA)

# Example 4:
TQDA <- tda(x = iris[,1:4], subgroup = c(1, 1, 1), ID = iris$Species,
            trans = TRUE, common_sigma = FALSE, common_lambda = TRUE)
print(TQDA)
summary(TQDA)
```

Index

- * **Transformation**

- transDA-package, [2](#)

- * **discriminant analysis**

- tda, [4](#)

- * **discriminant**

- transDA-package, [2](#)

- * **mixture model**

- tda, [4](#)

- * **mixture**

- transDA-package, [2](#)

- * **transformation**

- tda, [4](#)

predict.tda, [2](#), [3](#), [5](#)

print.summary.tda(summary.tda), [4](#)

print.tda(tda), [4](#)

summary.tda, [4](#), [5](#)

tda, [2-4](#), [4](#)

transDA(transDA-package), [2](#)

transDA-package, [2](#)