Package ‘biClassify’

July 29, 2021

Type Package

Title Binary Classification Using Extensions of Discriminant Analysis

Version 1.2

Date 2021-7-27

Maintainer Alexander F. Lapanowski <aflapan@gmail.com>


License GPL-3

Imports Rcpp (>= 1.0.1), Matrix, stats, fields, MASS, datasets, mvtnorm, expm, DAAG

Depends R (>= 2.10)

LinkingTo Rcpp, RcppArmadillo

Suggests testthat, knitr, rmarkdown

RoxygenNote 6.1.1

LazyData true

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation yes

Author Alexander F. Lapanowski [aut, cre], Irina Gaynanova [aut]

Repository CRAN

Date/Publication 2021-07-29 08:30:16 UTC

R topics documented:

KOS .......................................................... 2
KOS_Data .................................................. 4
KOS

Function which generates feature weights, discriminant vector, and class predictions.

Description

Returns a (m x 1) vector of predicted group membership (either 1 or 2) for each data point in X. Uses Data and Cat to train the classifier.

Usage

KOS(TestData = NULL, TrainData, TrainCat, Method = "Full",
Mode = "Automatic", m1 = NULL, m2 = NULL, s = NULL,
Sigma = NULL, Gamma = NULL, Lambda = NULL, Epsilon = 1e-05)

Arguments

TestData (m x p) Matrix of unlabelled data with numeric features to be classified. Cannot have missing values.
TrainData (n x p) Matrix of training data with numeric features. Cannot have missing values.
TrainCat (n x 1) Vector of class membership corresponding to Data. Values must be either 1 or 2.
Method A string of characters which determines which version of KOS to use. Must be either "Full" or "Subsampled". Default is "Full".
Mode A string of characters which determines how the reduced sample parameters will be inputted for each method. Must be either "Research", "Interactive", or "Automatic". Default is "Automatic".
m1 The number of class 1 compressed samples to be generated. Must be a positive integer.
m2 The number of class 2 compressed samples to be generated. Must be a positive integer.
s The sparsity level used in compression. Must satisfy 0 < s < 1.
Sigma Scalar Gaussian kernel parameter. Default set to NULL and is automatically generated if user-specified value not provided. Must be > 0. User-specified parameters must satisfy hierarchical ordering.
Gamma
Scalar ridge parameter used in kernel optimal scoring. Default set to NULL and is automatically generated if user-specified value not provided. Must be > 0. User-specified parameters must satisfy hierarchical ordering.

Lambda
Scalar sparsity parameter on weight vector. Default set to NULL and is automatically generated by the function if user-specified value not provided. Must be >= 0. When Lambda = 0, SparseKOS defaults to kernel optimal scoring of [Lapanowski and Gaynanova, preprint] without sparse feature selection. User-specified parameters must satisfy hierarchical ordering.

Epsilon
Numerical stability constant with default value 1e-05. Must be > 0 and is typically chosen to be small.

Details
Function which handles classification. Generates feature weight vector and discriminant coefficients vector in sparse kernel optimal scoring. If a matrix X is provided, the function classifies each data point using the generated feature weight vector and discriminant vector. Will use user-supplied parameters Sigma, Gamma, and Lambda if any are given. If any are missing, the function will run SelectParams to generate the other parameters. User-specified values must satisfy hierarchical ordering.

Value
A list of

- Predictions (m x 1) Vector of predicted class labels for the data points in TestData. Only included in non-null value of X is provided.
- Weights (p x 1) Vector of feature weights.
- Dvec (n x 1) Discriminant coefficients vector.

References

Examples

```r
Sigma <- 1.325386 #Set parameter values equal to result of SelectParam.
Gamma <- 0.07531579 #Speeds up example.
Lambda <- 0.002855275

TrainData <- KOS_Data$TrainData
TrainCat <- KOS_Data$TrainCat
TestData <- KOS_Data$TestData
TestCat <- KOS_Data$TestCat

KOS(TestData = TestData,
    TrainData = TrainData,
    TrainCat = TrainCat ,
    Sigma = Sigma ,
```
A list consisting of Training and Test data along with corresponding class labels.

**Description**

A list consisting of Training and Test data along with corresponding class labels.

**Usage**

KOS_Data

**Format**

A list consisting of:

**TrainData** (179 x 4) Matrix of training data features. The first two features satisfy \( \sqrt{x_{i1}^2 + x_{i2}^2} > 2/3 \) if the \( i \)th sample is in class 1. Otherwise, they satisfy \( \sqrt{x_{i1}^2 + x_{i2}^2} < 2/3 - 1/10 \) if the \( i \)th sample is in class 2. The third and fourth features are generated as independent \( N(0, 1/2) \) noise.

**TestData** (94 x 4) Matrix of test data features. The first two features satisfy \( \sqrt{x_{i1}^2 + x_{i2}^2} > 2/3 \) if the \( i \)th sample is in class 1. Otherwise, they satisfy \( \sqrt{x_{i1}^2 + x_{i2}^2} < 2/3 - 1/10 \) if the \( i \)th sample is in class 2. The third and fourth features are generated as independent \( N(0, 1/2) \) noise.

**CatTrain** (179 x 1) Vector of class labels for the training data.

**CatTest** (94 x 1) Vector of class labels for the test data. ...

**Source**

Simulation model 1 from [Lapanowski and Gaynanova, preprint].

**References**

**Description**

A wrapper function for the various LDA implementations available in this package.

Generates class predictions for `TestData`.

**Usage**

```r
LDA(TrainData, TrainCat, TestData, Method = "Full", Mode = "Automatic",
    m1 = NULL, m2 = NULL, m = NULL, s = NULL, gamma = 1e-05,
    type = "Rademacher")
```

**Arguments**

- **TrainData** A (n x p) numeric matrix without missing values consisting of n training samples each with p features.
- **TrainCat** A vector of length n consisting of group labels of the n training samples in `TrainData`. Must consist of 1s and 2s.
- **TestData** A (m x p) numeric matrix without missing values consisting of m training samples each with p features. The number of features must equal the number of features in `TrainData`.
- **Method** A string of characters which determines which version of LDA to use. Must be either "Full", "Compressed", "Subsampled", "Projected", or "fastRandomFisher". Default is "Full".
- **Mode** A string of characters which determines how the reduced sample parameters will be inputted for each method. Must be either "Research", "Interactive", or "Automatic". Default is "Automatic".
- **m1** The number of class 1 compressed samples to be generated. Must be a positive integer.
- **m2** The number of class 2 compressed samples to be generated. Must be a positive integer.
- **m** The number of total compressed samples to be generated. Must be a positive integer.
- **s** The sparsity level used in compression. Must satisfy 0 < s < 1.
- **gamma** A numeric value for the stabilization amount gamma * I added to the covariance matrix used in the LDA decision rule. Default amount is 1E-5. Cannot be negative.
- **type** A string of characters determining the type of compression matrix used. The accepted values are Rademacher, Gaussian, and Count.

**Details**

Function which handles all implementations of LDA.
Value

A list containing

- **Predictions**: (m x 1) Vector of predicted class labels for the data points in `TestData`.
- **Dvec**: (p x 1) Discriminant vector used to predict the class labels.

References


Examples

```r
TrainData <- LDA_Data$TrainData
TrainCat <- LDA_Data$TrainCat
TestData <- LDA_Data$TestData
plot(TrainData[,2]~TrainData[,1], col = c("blue","orange")[as.factor(TrainCat)])

#----- Full LDA -------
LDA(TrainData = TrainData, 
   TrainCat = TrainCat, 
   TestData = TestData, 
   Method = "Full", 
   gamma = 1E-5)

#----- Compressed LDA -------
m1 <- 700
m2 <- 300
s <- 0.01
LDA(TrainData = TrainData, 
    TrainCat = TrainCat, 
    TestData = TestData, 
    Method = "Compressed", 
    Mode = "Research", 
    m1 = m1, 
    m2 = m2, 
    s = s, 
    gamma = 1E-5)

LDA(TrainData = TrainData, 
    TrainCat = TrainCat, 
    TestData = TestData, 
    Method = "Compressed", 
    Mode = "Automatic", 
    gamma = 1E-5)

#----- Sub-sampled LDA ------
m1 <- 700
m2 <- 300
```
LDA(TrainData = TrainData,  
    TrainCat = TrainCat,  
    TestData = TestData,  
    Method = "Subsampled",  
    Mode = "Research",  
    m1 = m1,  
    m2 = m2,  
    gamma = 1E-5)

LDA(TrainData = TrainData,  
    TrainCat = TrainCat,  
    TestData = TestData,  
    Method = "Subsampled",  
    Mode = "Automatic",  
    gamma = 1E-5)

#----- Projected LDA ------
m1 <- 700  
m2 <- 300  
s <- 0.01  
LDA(TrainData = TrainData,  
    TrainCat = TrainCat,  
    TestData = TestData,  
    Method = "Projected",  
    Mode = "Research",  
    m1 = m1,  
    m2 = m2,  
    s = s,  
    gamma = 1E-5)

LDA(TrainData = TrainData,  
    TrainCat = TrainCat,  
    TestData = TestData,  
    Method = "Projected",  
    Mode = "Automatic",  
    gamma = 1E-5)

#----- Fast Random Fisher ------
m <- 1000  
s <- 0.01  
LDA(TrainData = TrainData,  
    TrainCat = TrainCat,  
    TestData = TestData,  
    Method = "fastRandomFisher",  
    Mode = "Research",  
    m = m,  
    s = s,  
    gamma = 1E-5)

LDA(TrainData = TrainData,  
    TrainCat = TrainCat,  
    TestData = TestData,  
    Method = "fastRandomFisher",  
    Mode = "Research",  
    gamma = 1E-5)
```
Mode = "Automatic",
gamma = 1E-5)
```

---

**LDA_Data**

A list consisting of **Training and Test data along with corresponding class labels.**

---

**Description**

A list consisting of Training and Test data along with corresponding class labels.

**Usage**

```r
LDA_Data
```

**Format**

A list consisting of:

- **TrainData** (10000 x 10) Matrix of independent normally-distributed training samples conditioned on class membership. There are 7000 samples belonging to class 1, and 3000 samples belonging to class 2. The class 1 mean vector is the vector of length 10 consisting only of -2. Likewise, the class 2 mean vector is the vector of length 10 consisting only of 2. The shared covariance matrix has (i,j) entry (0.5)^|i-j|.
- **TestData** (1000 x 10) Matrix of independent test data features with the same distributions and class proportions as **TrainData**.
- **Train** (10000 x 1) Vector of class labels for the samples in **TrainData**.
- **TestCat** (1000 x 1) Vector of class labels for the samples in **TestData**.

**References**


---

**QDA**

*Quadratic Discriminant Analysis (QDA)*

---

**Description**

A wrapper function for the various QDA implementations available in this package.

Generates class predictions for **TestData**.

**Usage**

```r
QDA(TrainData, TrainCat, TestData, Method = "Full", Mode = "Automatic",
m1 = NULL, m2 = NULL, m = NULL, s = NULL, gamma = 1e-05)
```
Arguments

TrainData  A (n x p) numeric matrix without missing values consisting of n training samples each with p features.

TrainCat  A vector of length n consisting of group labels of the n training samples in TrainData. Must consist of 1s and 2s.

TestData  A (m x p) numeric matrix without missing values consisting of m training samples each with p features. The number of features must equal the number of features in TrainData.

Method  A string of characters which determines which version of QDA to use. Must be either "Full", "Compressed", or "Subsampled".

Mode  A string of characters which determines how the reduced sample parameters will be inputted for each method. Must be either "Research", "Interactive", or "Automatic". Default is "Automatic".

m1  The number of class 1 compressed samples to be generated. Must be a positive integer.

m2  The number of class 2 compressed samples to be generated. Must be a positive integer.

m  The number of total compressed samples to be generated. Must be a positive integer.

s  The sparsity level used in compression. Must satisfy 0 < s < 1.

gamma  A numeric value for the stabilization amount gamma * I added to the covariance matrix used in the LDA decision rule. Default amount is 1E-5. Cannot be negative.

Details

Function which handles all implementations of LDA.

Value

Predictions  (m x 1) Vector of predicted class labels for the data points in TestData.

References


Examples

TrainData <- QDA_Data$TrainData
TrainCat <- QDA_Data$TrainCat
TestData <- QDA_Data$TestData
plot(TrainData[,2]-TrainData[,1], col = c("blue","orange") [as.factor(TrainCat)])

#----- Full QDA -------
QDA(TrainData = TrainData,
   TrainCat = TrainCat,
QDA_Data

A list consisting of Training and Test data along with corresponding class labels.

Description

A list consisting of Training and Test data along with corresponding class labels.
SelectParams

Usage

QDA_Data

Format

A list consisting of:

TrainData (10000 x 10) Matrix of independent normally-distributed training samples conditioned on class membership. There are 7000 samples belonging to class 1, and 3000 samples belonging to class 2. The class 1 mean vector is the vector of length 10 consisting only of -2. Likewise, the class 2 mean vector is the vector of length 10 consisting only of 2. The class 1 covariance matrix has (i,j) entry (0.5)^|i-j|. The class 2 covariance matrix has (i,j) entry (-0.5)^|i-j|.

TestData (1000 x 10) Matrix of independent test data features with the same distributions and class proportions as TrainData.

Train (10000 x 1) Vector of class labels for the samples in TrainData.

TestCat (1000 x 1) Vector of class labels for the samples in TestData.

References


SelectParams

Generates parameters.

Description

Generates parameters to be used in sparse kernel optimal scoring.

Usage

SelectParams(TrainData, TrainCat, Sigma = NULL, Gamma = NULL, Epsilon = 1e-05)

Arguments

TrainData  (n x p) Matrix of training data with numeric features. Cannot have missing values.

TrainCat  (n x 1) Vector of class membership. Values must be either 1 or 2.

Sigma  Scalar Gaussian kernel parameter. Default set to NULL and is automatically generated if user-specified value not provided. Must be > 0. User-specified parameters must satisfy hierarchical ordering.

Gamma  Scalar ridge parameter used in kernel optimal scoring. Default set to NULL and is automatically generated if user-specified value not provided. Must be > 0. User-specified parameters must satisfy hierarchical ordering.

Epsilon  Numerical stability constant with default value 1e-05. Must be > 0 and is typically chosen to be small.
Details
Generates the gaussian kernel, ridge, and sparsity parameters for use in sparse kernel optimal scoring using the methods presented in [Lapanowski and Gaynanova, preprint]. The Gaussian kernel parameter is generated using five-fold cross-validation of the misclassification error rate across the .05, .1, .2, .3, .5 quantiles of squared-distances between groups. The ridge parameter is generated using a stabilization technique developed in Lapanowski and Gaynanova (2019). The sparsity parameter is generated by five-fold cross-validation over a logarithmic grid of 20 values in an automatically-generated interval.

Value
A list of
- Sigma: Gaussian kernel parameter.
- Gamma: Ridge Parameter.
- Lambda: Sparsity parameter.

References

Examples
```r
Sigma <- 1.325386 #Set parameter values equal to result of SelectParam.
Gamma <- 0.07531579 #Speeds up example

TrainData <- KOS_Data$TrainData
TrainCat <- KOS_Data$TrainCat

SelectParams(TrainData = TrainData,
             TrainCat = TrainCat,
             Sigma = Sigma,
             Gamma = Gamma)
```
Index

* datasets
  KOS_Data, 4
  LDA_Data, 8
  QDA_Data, 10

KOS, 2
KOS_Data, 4

LDA, 5
LDA_Data, 8

QDA, 8
QDA_Data, 10

SelectParams, 11