Package ‘Immigrate’

May 23, 2020

Title Iterative Max-Min Entropy Margin-Maximization with Interaction Terms for Feature Selection

Version 0.2.1

Author Ruzhang Zhao, Pengyu Hong, Jun S. Liu

Description Based on large margin principle, this package performs feature selection methods: "IM4E" (Iterative Margin-Maximization under Max-Min Entropy Algorithm); "Immigrate" (Iterative Max-Min Entropy Margin-Maximization with Interaction Terms Algorithm); "BIM" (Boosted version of IMMIGRATE algorithm); "Simba" (Iterative Search Margin Based Algorithm); "LFE" (Local Feature Extraction Algorithm). This package also performs prediction for the above feature selection methods.

Maintainer Ruzhang Zhao <ruzhangzhao@gmail.com>

Depends R (>= 3.0.0)

License GPL (>= 2)

URL https://cran.r-project.org/package=Immigrate,
     https://arxiv.org/abs/1810.02658,
     https://github.com/RuzhangZhao/Immigrate/

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0.9000

LinkingTo Rcpp

Imports Rcpp, pROC, stats

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-05-22 22:42:22 UTC

R topics documented:

BIM ........................................ 2
IM4E ........................................ 3
Immigrate .................................. 4
Description

This function performs BIM algorithm (Boosted version of IMMIGRATE).

Usage

```r
BIM(
  xx,
  yy,
  nBoost = 3,
  max_iter = 5,
  removesmall = FALSE,
  sigstart = 0.02,
  sigend = 4
)
```

Arguments

- `xx`: model matrix of explanatory variables
- `yy`: label vector
- `nBoost`: number of classifiers in BIM, default to be 3
- `max_iter`: maximum number of iteration for IMMIGRATE classifier, default to be 5
- `removesmall`: whether remove features with small weights, default to be FALSE
- `sigstart`: start of sigma used in algorithm, default to be 0.02
- `sigend`: end of sigma used in algorithm, default to be 4

Value

- `matrix`: list of weight matrices
- `weights`: coefficient vectors for classifiers
- `sample_wt`: sample weights, refer to cost function in link below for more details
References


See Also


Examples

data(park)
xx<-park$xx
yy<-park$yy
re<-BIM(xx,yy)

Description

This function performs IM4E(Iterative Margin-Maximization under Max-Min Entropy) algorithm.

Usage

IM4E(
  xx,
  yy,
  epsilon = 0.01,
  sig = 1,
  lambda = 1,
  max_iter = 10,
  removesmall = FALSE
)

Arguments

xx               model matrix of explanatory variables
yy               label vector
epsilon         criterion for stopping iteration, default to be 0.01
sig              sigma used in algorithm, default to be 1
lambda           lambda used in algorithm, default to be 1
max_iter         maximum number of iteration
removesmall      whether remove features with small weights, default to be FALSE
Immigrate

Value

- **w**: weight vector obtained by IM4E algorithm
- **iter_num**: number of iteration for convergence
- **final_c**: final cost value. Refer to the cost function in reference below for more details

References


Examples

```r
data(park)
xx<-park$xx
yy<-park$yy
re<-IM4E(xx,yy)
print(re)
```

---

Description

This function performs IMMIGRATE (Iterative Max-Min Entropy Margin-Maximization with Interaction Terms) algorithm. IMMIGRATE is a hypothesis-margin based feature selection method with interaction terms. Its weight matrix reflects the relative importance of features and their interactions, which can be used for feature selection.

Usage

```r
Immigrate(
  xx,
  yy,
  w0,
  epsilon = 0.01,
  sig = 1,
  max_iter = 10,
  removesmall = FALSE,
  randomw0 = FALSE
)
```

Arguments

- **xx**: model matrix of explanatory variables
- **yy**: label vector
- **w0**: initial weight matrix, default to be diagonal matrix when missing
LFE

epsilon criterion for stopping iteration
sig sigma used in algorithm, default to be 1. Refer to the cost function in the link below for more details
max_iter maximum number of iteration
removesmall whether to remove features with small weights, default to be FALSE
randomw0 whether to use randomly initial weights, default to be FALSE

Value

w weight matrix obtained by IMMIGRATE algorithm
iter_num number of iteration for convergence
final_c final cost value. Refer to the cost function in link below for more details

References


See Also

Please refer to https://github.com/RuzhangZhao/Immigrate/ for implementation demo.

Examples

data(park)
xx<-park$xx
yy<-park$yy
re<-Immigrate(xx,yy)
print(re)

Data Description

This function performs LFE(Local Feature Extraction) algorithm.

Usage

LFE(xx, yy, T = 5)

Arguments

xx model matrix of explanatory variables
yy label vector
T number of instance used to update weights, default to be 5
one.IM4E

Value

\[ w \quad \text{new weight matrix after LFE algorithm} \]

References


Examples

```r
data(park)
xx<-park$xx
yy<-park$yy
re<-LFE(xx,yy)
print(re)
```

one.IM4E

Description

This function performs (IM4E)Iterative Margin-Maximization under Max-Min Entropy algorithm for one loop.

Usage

```r
one.IM4E(train_xx, train_yy, w, sig = 1, lambda = 1)
```

Arguments

- `train_xx`: model matrix of explanatory variables
- `train_yy`: label vector
- `w`: initial weight
- `sig`: sigma used in algorithm, default to be 1
- `lambda`: lambda used in algorithm, default to be 1

Value

- `w`: new weight vector after one loop
- `C`: cost after one loop
Description

This function performs Immigrate (Iterative Max-Min Entropy Margin-Maximization with Interaction Terms) algorithm for one loop.

Usage

one.Immigrate(train_xx, train_yy, W, sig = 1)

Arguments

- **train_xx**: model matrix of explanatory variables
- **train_yy**: label vector
- **W**: initial weight matrix
- **sig**: sigma used in algorithm, default to be 1

Value

- **W**: new weight matrix after one loop
- **C**: cost after one loop

See Also

Please refer to [https://github.com/RuzhangZhao/Immigrate/](https://github.com/RuzhangZhao/Immigrate/) for implementation demo.

Examples

data(park)
xx<-park$xx
yy<-park$yy
W0 <- diag(rep(1,ncol(xx)),ncol(xx))/sqrt(ncol(xx))
re<-one.Immigrate(xx,yy,W0)
print(re$w)
park

*Parkinsons Dataset*

**Description**

Parkinsons Dataset

**Usage**

data(park)

**Format**

An object of class

**Source**

parkinsons

**References**


**Examples**

```r
data(park)
xx <- park$xx
yy <- park$yy
```

---

`pred.values` **pred.values**

**Description**

This function performs some statistical value prediction

**Usage**

`pred.values(y_train, y_test, pred_train, pred_test)`

**Arguments**

- `y_train` label vector for training data
- `y_test` label vector for test data
- `pred_train` predicted probabilities for training data
- `pred_test` predicted probabilities for test data
**predict.BIM**  

**Value**

- `AUC_train`  
  AUC for training data
- `AUC_test`  
  AUC for test data
- `accuracy_test`  
  accuracy for test data
- `precision_test`  
  precision for test data
- `recall_test`  
  recall for test data
- `F1_test`  
  F1 score for test data
- `thre`  
  threshold to separate two labels, obtained from training data

**Examples**

```r
y_train<-c(0,1,0,1,0,1)
y_test<-c(0,1,0,1)
pred_train<-c(0.77,0.89,0.32,0.96,0.10,0.67)
pred_test<-c(0.68,0.75,0.50,0.81)
re<-pred.values(y_train,y_test,pred_train,pred_test)
print(re)
```

**Description**

This function performs the predition for BIM algorithm (Boosted version of IMMIGRATE).

**Usage**

```r
## S3 method for class 'BIM'
predict(object, xx, yy, newx, type = "both", ...)
```

**Arguments**

- `object`  
  result of BIM algorithm
- `xx`  
  model matrix of explanatory variables
- `yy`  
  label vector
- `newx`  
  new model matrix to be predicted
- `type`  
  the form of final output
- `...`  
  further arguments passed to or from other methods

**Value**

- `response`  
  predicted probabilities for for new data (newx)
- `class`  
  predicted class for for new data (newx)
**References**


**See Also**


**Examples**

```r
data(park)
xx<-park$xx
yy<-park$yy
index<-c(1:floor(nrow(xx)*0.3))
train_xx<-xx[-index,]
test_xx<-xx[index,]
train_yy<-yy[-index]
test_yy<-yy[index]
re<-BIM(train_xx,train_yy)
res<-predict(re,train_xx,train_yy,test_xx,type="class")
print(res)
```

**Description**

This function performs the prediction for IM4E (Iterative Margin-Maximization under Max-Min Entropy) algorithm.

**Usage**

```r
## S3 method for class 'IM4E'
predict(object, xx, yy, newx, sig = 1, type = "both", ...)
```

**Arguments**

- `object`: weight or result of IM4E algorithm
- `xx`: model matrix of explanatory variables
- `yy`: label vector
- `newx`: new model matrix to be predicted
- `sig`: sigma used in algorithm, default to be 1
- `type`: the form of final output, default to be "both". One can also choose "response" (predicted probabilities) or "class" (predicted labels).
- `...`: further arguments passed to or from other methods
predict.Immigrate

Value

response  predicted probabilities for new data (newx)
class     predicted class labels for new data (newx)

References


Examples

data(park)
xx<-park$xx
yy<-park$yy
index<-c(1:floor(nrow(xx)*0.3))
train_xx<-xx[-index,]
test_xx<-xx[index,]
train_yy<-yy[-index]
test_yy<-yy[index]
re<-IM4E(train_xx,train_yy)
res<-predict(re,train_xx,train_yy,test_xx,type="class")
print(res)

predict.Immigrate  predict.Immigrate

Description

This function performs the predition for Immigrate(Iterative Max-Min Entropy Margin-Maximization with Interaction Terms) algorithm.

Usage

## S3 method for class 'Immigrate'
predict(object, xx, yy, newx, sig = 1, type = "both", ...)

Arguments

object  result of Immigrate algorithm
xx      model matrix of explanatory variables
yy      label vector
newx    new model matrix to be predicted
sig     sigma used in prediction function, default to be 1. Refer to the prediction function in the link below for more details
type    the form of final output, default to be "both". One can also choose "response"(predicted probabilities) or "class"(predicted labels).
...     further arguments passed to or from other methods
Value

- response: predicted probabilities for new data (newx)
- class: predicted class labels for new data (newx)

References


See Also


Please refer to https://github.com/RuzhangZhao/Immigrate/ for implementation demo.

Examples

```r
data(park)
xx<-park$xx
yy<-park$yy
index<-c(1:floor(nrow(xx)*0.3))
train_xx<-xx[-index,]
test_xx<-xx[index,]
train_yy<-yy[-index]
test_yy<-yy[index]
re<-Immigrate(train_xx,train_yy)
res<-predict(re,train_xx,train_yy,test_xx,type="class")
print(res)
```

Description

This function performs prediction for LFE(Local Feature Extraction) algorithm.

Usage

```r
## S3 method for class 'LFE'
predict(object, xx, yy, newx, ...)
```

Arguments

- **object**: weights obtained from LFE
- **xx**: model matrix of explanatory variables
- **yy**: label vector
- **newx**: new model matrix to be predicted
- **...**: further arguments passed to or from other methods
Simba

Value

predicted labels for new data (newx)

References


Examples

data(park)
xx<-park$xx
yy<-park$yy
w<-LFE(xx,yy)
pred<-predict(w,xx,yy,xx)
pred
Examples

data(park)
xx<-park$xx
yy<-park$yy
re<-Simba(xx,yy)
print(re)
Index

*Topic **BIM**
  BIM, 2
  predict.BIM, 9
*Topic **IM4E**
  IM4E, 3
  one.IM4E, 6
  predict.IM4E, 10
*Topic **Immigrate**
  Immigrate, 4
  one.Immigrate, 7
  predict.Immigrate, 11
*Topic **LFE**
  LFE, 5
  predict.LFE, 12
*Topic **Simba**
  Simba, 13
*Topic **based**
  predict.BIM, 9
  predict.IM4E, 10
  predict.Immigrate, 11
  predict.LFE, 12
*Topic **datasets**
  park, 8
*Topic **data**
  predict.BIM, 9
  predict.IM4E, 10
  predict.Immigrate, 11
  predict.LFE, 12
*Topic **label**
  predict.BIM, 9
  predict.IM4E, 10
  predict.Immigrate, 11
  predict.LFE, 12
*Topic **new**
  predict.BIM, 9
  predict.IM4E, 10
  predict.Immigrate, 11
  predict.LFE, 12
*Topic **of**
  predict.BIM, 9
  predict.IM4E, 10
  predict.Immigrate, 11
  predict.LFE, 12

BIM, 2
IM4E, 3
Immigrate, 4
LFE, 5
one.IM4E, 6
one.Immigrate, 7
park, 8
pred.values, 8
predict.BIM, 9
predict.IM4E, 10
predict.Immigrate, 11
predict.LFE, 12
Simba, 13