

Package ‘LibOPF’

April 7, 2021

Type Package

Title Design of Optimum-Path Forest Classifiers

Version 2.6.1

Maintainer Rafael Junqueira Martarelli <rfe1.jm@gmail.com>

Description

The 'LibOPF' is a framework to develop pattern recognition techniques based on optimum-path forests (OPF), João P. Papa and Alexandre X. Falcão (2008) <doi:10.1007/978-3-540-89639-5_89>, with methods for supervised learning and data clustering.

Imports methods

License BSD_2_clause + file LICENSE

Encoding UTF-8

URL <https://github.com/RafaelJM/LibOPF-in-R/>,
<https://github.com/jppbsi/LibOPF/wiki/>,
<https://www.ic.unicamp.br/~afalcao/libopf/>

BugReports <https://github.com/RafaelJM/LibOPF-in-R/issues/>

RoxygenNote 7.0.2

NeedsCompilation yes

Author Rafael Junqueira Martarelli [aut, cre],
João Paulo Papa [aut],
Alexandre Xavier Falcão [cph]

Repository CRAN

Date/Publication 2021-04-07 10:20:07 UTC

R topics documented:

opf2svm	2
opf2txt	3
opf_accuracy	4
opf_accuracy4label	4
opf_check	5

opf_classify	6
opf_cluster	7
opf_create_subGraph	7
opf_distance	8
opf_fold	9
opf_info	10
opf_knn_classify	10
opf_knn_train	11
opf_learn	12
opf_merge	13
opf_normalize	13
opf_pruning	14
opf_read_classification	15
opf_read_distances	15
opf_read_modelfile	16
opf_read_subGraph	16
opf_run_example	17
opf_semi	18
opf_split	19
opf_train	19
opf_write_classification	20
opf_write_distances	21
opf_write_modelfile	22
opf_write_subGraph	22
SNode-class	23
subGraph-class	23
svm2opf	23
txt2opf	24
Index	25

opf2svm	<i>Converts an OPF subGraph object to a LIBSVM file</i>
---------	---

Description

Converts an OPF subGraph object to a LIBSVM file

Usage

```
opf2svm(data, outputFile)
```

Arguments

data	The subGraph object
outputFile	LIBSVM output file name

Value

‘NULL’

Examples

```
dataset <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
File <- file.path(tempdir(), "boat.svm")
opf2svm(dataset,File)
opf_check(File)
```

opf2txt

Converts an OPF subGraph object to a text file

Description

Converts an OPF subGraph object to a text file

Usage

```
opf2txt(data, outputFile)
```

Arguments

data	OPF subGraph object
outputFile	Text output file name

Value

‘NULL’

Examples

```
dataset <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
File <- file.path(tempdir(), "boat.txt")
opf2txt(dataset,File)
```

opf_accuracy *Computes the OPF accuracy*

Description

Computes the OPF accuracy

Usage

```
opf_accuracy(dataSet, classification)
```

Arguments

dataSet Data object used in the opf_classify function (subGraph object), normally is the testing object

classification The output list produced by opf_classify function

Value

Returns the classification accuracy

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat", package = "LibOPF"))
X <- opf_split(dat, 0.5, 0, 0.5, 0)
T <- X$training
T2 <- X$testing
Y <- opf_train(T)
class <- opf_classify(T2, Y$classifier)
acc <- opf_accuracy(T2, class)
```

opf_accuracy4label *Computes the OPF accuracy for each class of a given set*

Description

Computes the OPF accuracy for each class of a given set

Usage

```
opf_accuracy4label(dataSet, classification)
```

Arguments

- `dataSet` Data object used in in the `opf_classify` function (subGraph object), normaly is the testing object
- `classification` The output list produced by `opf_classify` function

Value

Returns the classification accuracy for each class

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
X <- opf_split(dat,0.5,0,0.5,0)
T <- X$training
T2 <- X$testing
Y <- opf_train(T)
class <- opf_classify(T2, Y$classifier)
acc <- opf_accuracy4label(T2, class)
```

opf_check

Checks the OPF file for proper formatting purposes

Description

Checks the OPF file for proper formatting purposes

Usage

```
opf_check(file)
```

Arguments

- `file` The text OPF file name

Details

usage `opf_check` <input ASCII file in the LibOPF format>: Note that the input file for `opf_check` must be a text file. Use `opf2txt` to convert your OPF binary file into a text file.

Value

‘NULL’

Examples

```
dataset <- opf_read_subGraph(system.file("extdata/boat.dat", package = "LibOPF"))
File <- file.path(tempdir(), "boat.txt")
opf2txt(dataset, File)
opf_check(File)
```

opf_classify
Executes the test phase of the OPF classifier

Description

Executes the test phase of the OPF classifier

Usage

```
opf_classify(dataSet, classifier, precomputedDistance = NA)
```

Arguments

dataSet	The testing data object produced by the <code>opf_split</code> function (subGraph object)
classifier	The classifier object produced by one of the classification functions (model object)
precomputedDistance	The precomputed distance matrix produced by the <code>opf_distance</code> (leave it in blank if you are not using this resource)

Value

Returns the given subGraph classification list (predicted labels)

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat", package = "LibOPF"))
X <- opf_split(dat, 0.5, 0, 0.5, 0)
T <- X$training
T2 <- X$testing
Y <- opf_train(T)
class <- opf_classify(T2, Y$classifier)
acc <- opf_accuracy(T2, class)
```

opf_cluster *Computes clusters by unsupervised OPF*

Description

Computes clusters by unsupervised OPF

Usage

```
opf_cluster(dataSet, kmax, calculateOp, value, precomputedDistance = NA)
```

Arguments

dataSet	The training object produced by the opf_split function (subGraph object)
kmax	The kmax (maximum degree for the knn graph)
calculateOp	Clusters by: 0 for height, 1 for area and 2 for volume
value	Value of parameter "calculateOp" in [0-1]
precomputedDistance	The precomputed distance matrix produced by the opf_distance (leave it in blank if you are not using this resource)

Value

Returns a list which contains the classifier object and the classification list object (i.e., clusters' id)

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat", package = "LibOPF"))
X <- opf_split(dat, 0.8, 0, 0.2, 0)
T <- X$training
T2 <- X$testing
Y <- opf_cluster(T, 100, 1, 0.2)
class <- opf_knn_classify(T2, Y$classifier)
acc <- opf_accuracy(T2, class)
```

opf_create_subGraph *Creates an empty subGraph structure*

Description

Creates an empty subGraph structure

Usage

```
opf_create_subGraph(nnodes)
```

Arguments

nnodes Number of nodes

Value

Returns an empty subGraph

Examples

```
EmptySubgraph <- opf_create_subGraph(10)
```

opf_distance	<i>Generates the precomputed distance file for the OPF classifier</i>
--------------	---

Description

Generates the precomputed distance file for the OPF classifier

Usage

```
opf_distance(dataSet, distanceOp, normalize = 0)
```

Arguments

dataSet The subGraph object, normally is the whole data
distanceOp Distance calculation option
normalize Distance normalization? 1- yes 0 - no

Details

Options for distance calculation:

- 1 - Euclidean
- 2 - Chi-Square
- 3 - Manhattan (L1)
- 4 - Canberra
- 5 - Squared Chord
- 6 - Squared Chi-Squared
- 7 - BrayCurtis

Value

Returns the distance matrix

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
dist <- opf_distance(dat,3,0)
X <- opf_split(dat,0.5,0,0.5,0)
T <- X$training
T2 <- X$testing
Y <- opf_train(T,dist)
class <- opf_classify(T2, Y$classifier,dist)
acc <- opf_accuracy(T2, class)
```

opf_fold

Generates k folds (objects) for the OPF classifier

Description

Generates k folds (objects) for the OPF classifier

Usage

```
opf_fold(dataSet, k, normalize = 0)
```

Arguments

dataSet	The subGraph object
k	Number of folds
normalize	Distance normalization? 1- yes 0 - no

Value

Returns a list of subGraph objects

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
folds <- opf_fold(dat, 4)
```

opf_info *Gives information about the OPF file*

Description

Gives information about the OPF file

Usage

```
opf_info(dataSet)
```

Arguments

dataSet The OPF file

Value

'NULL'

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat", package = "LibOPF"))
opf_info(dat)
```

opf_knn_classify *Executes the test phase of the OPF classifier with knn adjacency*

Description

Executes the test phase of the OPF classifier with knn adjacency

Usage

```
opf_knn_classify(dataSet, classifier, precomputedDistance = NA)
```

Arguments

dataSet The testing object produced by the opf_split (subGraph object)

classifier The classifier object produced by one of the classification functions (model object)

precomputedDistance The precomputed distance matrix produced by the opf_distance (leave it in blank if you are not using this resource)

Value

Returns the given subGraph classification list

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
X <- opf_split(dat,0.8,0,0.2,0)
T <- X$training
T2 <- X$testing
Y <- opf_cluster(T,100,1,0.2)
class <- opf_knn_classify(T2, Y$classifier)
acc <- opf_accuracy(T2, class)
```

opf_knn_train	<i>Executes the training phase of the OPF classifier with knn adjacency</i>
---------------	---

Description

Executes the training phase of the OPF classifier with knn adjacency

Usage

```
opf_knn_train(trainFile, evaluatFile, kmax, precomputedDistance = NA)
```

Arguments

trainFile	The training object produced by the opf_split (subGraph object)
evaluatFile	The evaluation object produced by the opf_split (subGraph object)
kmax	The kmax (maximum degree for the knn graph)
precomputedDistance	The precomputed distance matrix produced by the opf_distance (leave it in blank if you are not using this resource)

Details

Returns a list which contains the classifier object and the classification list object

Value

Returns a list which contains the classifier object and the classification list object (i.e., clusters' id)

Examples

```

dat <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
X <- opf_split(dat,0.3,0.2,0.5,0)
T <- X$training
T2 <- X$testing
E <- X$evaluating
Y <- opf_knn_train(T,E,100)
class <- opf_knn_classify(T2, Y$classifier)
acc <- opf_accuracy(T2, class)

```

opf_learn

Executes the learning phase of the OPF classifier

Description

Executes the learning phase of the OPF classifier

Usage

```
opf_learn(trainFile, evaluatFile, precomputedDistance = NA)
```

Arguments

trainFile The training object produced by the opf_split (subGraph object)
evaluatFile The evaluation produced object by the opf_split (subGraph object)
precomputedDistance
 The precomputed distance matrix produced by the opf_distance (leave it in blank
 if you are not using this resource)

Details

Executes the training phase

Value

Returns a list which contains the classifier model object

Examples

```

dat <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
X <- opf_split(dat,0.3,0.2,0.5,0)
T <- X$training
T2 <- X$testing
E <- X$evaluating
Y <- opf_learn(T,E)
class <- opf_classify(T2, Y$classifier)
acc <- opf_accuracy(T2, class)

```

opf_merge	<i>Merge subGraphs</i>
-----------	------------------------

Description

Merge subGraphs

Usage

```
opf_merge(dataSets)
```

Arguments

dataSets An array of subGraph objects

Value

Returns the merged subGraph object

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
dat2 <- opf_read_subGraph(system.file("extdata/Z1LINE.dat",package = "LibOPF"))
dataSet <- opf_merge(c(dat,dat2))
```

opf_normalize	<i>Normalizes data for the OPF classifier</i>
---------------	---

Description

Normalizes data for the OPF classifier

Usage

```
opf_normalize(dataSet)
```

Arguments

dataSet The subGraph object

Value

Returns the normalized subGraph

Examples

```
dataset <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
dat <- opf_normalize(dataset)
```

opf_pruning	<i>Executes the pruning algorithm</i>
-------------	---------------------------------------

Description

Executes the pruning algorithm

Usage

```
opf_pruning(
  dataTraining,
  dataEvaluating,
  percentageAccuracy,
  precomputedDistance = NA
)
```

Arguments

`dataTraining` The training object produced by the `opf_split` (subGraph object)

`dataEvaluating` The evaluating object produced by the `opf_split` (subGraph object)

`percentageAccuracy`
Max percentage of lost accuracy [0,1]

`precomputedDistance`
The precomputed distance matrix produced by the `opf_distance` (leave it in blank if you are not using this resource)

Value

Returns a list which contains the classifier model object

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
X <- opf_split(dat,0.3,0.2,0.5,0)
T <- X$training
T2 <- X$testing
E <- X$evaluating
Y <- opf_pruning(T,E,0.8)
class <- opf_classify(T2, Y$classifier)
acc <- opf_accuracy(T2, class)
```

`opf_read_classification`*Reads a file which contains the nodes' predicted labels*

Description

Reads a file which contains the nodes' predicted labels

Usage

```
opf_read_classification(file)
```

Arguments

`file` The file which contains the nodes' predicted labels

Value

Returns the predicted labels list

Examples

```
File <- system.file("extdata/classification.txt", package = "LibOPF")  
classification <- opf_read_classification(File)
```

`opf_read_distances`*Reads a file which contains the precalculated distances*

Description

Reads a file which contains the precalculated distances

Usage

```
opf_read_distances(file)
```

Arguments

`file` The file which contains the distances matrix

Value

Returns the precalculated distances matrix

Examples

```
distances <- opf_read_distances(system.file("extdata/distances.dat", package = "LibOPF"))
```

opf_read_modelfile *Reads a file which contains the learned model*

Description

Reads a file which contains the learned model

Usage

```
opf_read_modelfile(file)
```

Arguments

file The file which contains the learned model

Value

Returns the learned model object

Examples

```
classifier <- opf_read_modelfile(system.file("extdata/classifier.opf", package = "LibOPF"))
```

opf_read_subGraph *Reads a file which contains the subGraph*

Description

Reads a file which contains the subGraph

Usage

```
opf_read_subGraph(file)
```

Arguments

file The file name

Value

Returns the subGraph object

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat", package = "LibOPF"))
X <- opf_split(dat, 0.5, 0, 0.5, 0)
T <- X$training
T2 <- X$testing
Y <- opf_train(T)
class <- opf_classify(T2, Y$classifier)
acc <- opf_accuracy(T2, class)
```

opf_run_example	<i>Runs an usage example</i>
-----------------	------------------------------

Description

This function will run this example:

```
dat <- opf_read_subGraph(dataset) (dataset is the subgraph file)
X <- opf_split(dat, 0.5, 0, 0.5, 0)
T <- X$training
T2 <- X$testing
Y <- opf_train(T)
class <- opf_classify(T2, Y$classifier)
acc <- opf_accuracy(T2, class)
```

Usage

```
opf_run_example(dataset)
```

Arguments

dataset	A dataset folder for the test
---------	-------------------------------

Value

Returns the accuracy

`opf_semi`*Executes the semi supervised training phase*

Description

Executes the semi supervised training phase

Usage

```
opf_semi(  
  labeledTrainSubGraph,  
  unLabeledTrainSubGraph,  
  evaluatFile = NA,  
  precomputedDistance = NA  
)
```

Arguments

`labeledTrainSubGraph`
The labeled training object (subGraph object)

`unLabeledTrainSubGraph`
The unlabeled training object (subGraph object)

`evaluatFile` The evaluation object produced by the `opf_split` (subGraph object)

`precomputedDistance`
The precomputed distance matrix produced by the `opf_distance` (leave it in blank if you are not using this resource)

Details

Returns the learned model object

Value

Returns a list which contains the classifier object and the classification list object

Examples

```
Training <- opf_read_subGraph(system.file("extdata/Z1LINE.dat", package = "LibOPF"))  
TUnlabeled <- opf_read_subGraph(system.file("extdata/Z1DOUBLELINE.dat", package = "LibOPF"))  
Testing <- opf_read_subGraph(system.file("extdata/Z3.dat", package = "LibOPF"))  
Y <- opf_semi(Training, TUnlabeled)  
class <- opf_classify(Testing, Y$classifier)  
acc <- opf_accuracy(Testing, class)
```

opf_split	<i>Generates training, evaluation and test sets for the OPF classifier</i>
-----------	--

Description

Generates training, evaluation and test sets for the OPF classifier

Usage

```
opf_split(dataSet, training_p, evaluating_p, testing_p, normalize = 0)
```

Arguments

dataSet	The data (subGraph object)
training_p	Percentage for the training set size [0,1]
evaluating_p	Percentage for the evaluation set size [0,1] (leave 0 in the case of no learning)
testing_p	Percentage for the test set size [0,1]
normalize	Distance normalization? 1- yes 0 - no

Value

Returns the training, evaluating and the testing objects

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat", package = "LibOPF"))
X <- opf_split(dat, 0.5, 0, 0.5, 0)
T <- X$training
T2 <- X$testing
Y <- opf_train(T)
class <- opf_classify(T2, Y$classifier)
acc <- opf_accuracy(T2, class)
```

opf_train	<i>Executes the training phase of the OPF classifier</i>
-----------	--

Description

Executes the training phase of the OPF classifier

Usage

```
opf_train(dataSet, precomputedDistance = NA)
```

Arguments

dataSet The training object produced by the opf_split (subGraph object)
precomputedDistance The precomputed distance matrix produced by the opf_distance (leave it in blank if you are not using this resource)

Value

Returns a list which contains the classifier object and the classification list object

Examples

```

dat <- opf_read_subGraph(system.file("extdata/boat.dat",package = "LibOPF"))
X <- opf_split(dat,0.5,0,0.5,0)
T <- X$training
T2 <- X$testing
Y <- opf_train(T)
class <- opf_classify(T2, Y$classifier)
acc <- opf_accuracy(T2, class)
  
```

opf_write_classification

Writes into a file the predicted labels produced by the opf classifier

Description

Writes into a file the predicted labels produced by the opf classifier

Usage

```
opf_write_classification(classes, file)
```

Arguments

classes The classification list (i.e.,predicted labels) produced by the classifier
file Where you want to save the classification vector

Value

‘NULL’

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat", package = "LibOPF"))
X <- opf_split(dat, 0.5, 0, 0.5, 0)
T <- X$training
T2 <- X$testing
Y <- opf_train(T)
opf_write_classification(Y$classification, file.path(tempdir(), "classification.txt"))
```

opf_write_distances *Writes into a file the precalculated distances computed by opf_distances function*

Description

Writes into a file the precalculated distances computed by opf_distances function

Usage

```
opf_write_distances(distances, file)
```

Arguments

distances The matrix produced by the opf_distances function
file The file name where you want to save the distances

Value

‘NULL’

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat", package = "LibOPF"))
dist <- opf_distance(dat, 3, 0)
opf_write_distances(dist, file.path(tempdir(), "distances.dat"))
```

opf_write_modelfile *Writes into a file the trained OPF classifier*

Description

Writes into a file the trained OPF classifier

Usage

```
opf_write_modelfile(g, file)
```

Arguments

g	The classifier object
file	The file name to save the classifier

Value

‘NULL’

Examples

```
dat <- opf_read_subGraph(system.file("extdata/boat.dat", package = "LibOPF"))
X <- opf_split(dat, 0.5, 0, 0.5, 0)
T <- X$training
T2 <- X$testing
Y <- opf_train(T)
opf_write_modelfile(Y$classifier, file.path(tempdir(), "classifier.opf"))
```

opf_write_subGraph *Writes into a file a subGraph*

Description

Writes into a file a subGraph

Usage

```
opf_write_subGraph(g, file)
```

Arguments

g	The subGraph object
file	The file name to save the subGraph

Value

‘NULL’

Examples

```
dataset <- opf_read_subGraph(system.file("extdata/boat.dat", package = "LibOPF"))
opf_write_subGraph(dataset, file.path(tempdir(), "boat.dat"))
```

SNode-class	<i>Subgraphs' node class</i>
-------------	------------------------------

Description

Subgraphs' node class

subGraph-class	<i>Subgraph class</i>
----------------	-----------------------

Description

Subgraph class

svm2opf	<i>Converts a LIBSVM file to an OPF subGraph object</i>
---------	---

Description

Converts a LIBSVM file to an OPF subGraph object

Usage

```
svm2opf(inputFile)
```

Arguments

inputFile	LIBSVM input file
-----------	-------------------

Value

Returns the OPF object

Examples

```
dataset <- svm2opf(system.file("extdata/boat.svm", package = "LibOPF"))
```

txt2opf	<i>Converts a text file to an OPF subGraph object</i>
---------	---

Description

Converts a text file to an OPF subGraph object

Usage

```
txt2opf(inputFile)
```

Arguments

inputFile Text input file

Value

Returns the OPF object

Examples

```
dataset <- txt2opf(system.file("extdata/boat.txt", package = "LibOPF"))
```


Index

opf2svm, [2](#)
opf2txt, [3](#)
opf_accuracy, [4](#)
opf_accuracy4label, [4](#)
opf_check, [5](#)
opf_classify, [6](#)
opf_cluster, [7](#)
opf_create_subGraph, [7](#)
opf_distance, [8](#)
opf_fold, [9](#)
opf_info, [10](#)
opf_knn_classify, [10](#)
opf_knn_train, [11](#)
opf_learn, [12](#)
opf_merge, [13](#)
opf_normalize, [13](#)
opf_pruning, [14](#)
opf_read_classification, [15](#)
opf_read_distances, [15](#)
opf_read_modelfile, [16](#)
opf_read_subGraph, [16](#)
opf_run_example, [17](#)
opf_semi, [18](#)
opf_split, [19](#)
opf_train, [19](#)
opf_write_classification, [20](#)
opf_write_distances, [21](#)
opf_write_modelfile, [22](#)
opf_write_subGraph, [22](#)

SNode (SNode-class), [23](#)
SNode-class, [23](#)
subGraph (subGraph-class), [23](#)
subGraph-class, [23](#)
svm2opf, [23](#)

txt2opf, [24](#)