Package 'kernelTDA'

July 3, 2019

Type Package

Title Statistical Learning with Kernel for Persistence Diagrams

License GPL-3

Version 0.1.1

Date 2019-07-01

Maintainer Tullia Padellini <tullia.padellini@uniroma1.it>

Description

Provides tools for exploiting topological information into standard statistical learning algorithms. To this aim, this package contains the most popular kernels defined on the space of persistence diagrams, and persistence images.
Moreover, it provides a solver for kernel Support Vector Machines problems, whose kernels are not necessarily positive semidefinite,
based on the C++ library 'LIBSVM' <https://www.csie.ntu.edu.tw/~cjlin/libsvm/>, and on its R implementation 'e1071'.
Additionally, it allows to compute Wasserstein distance between persistence diagrams with an arbitrary ground metric,
building an R interface for the C++ library 'HERA' <https://bitbucket.org/grey_narn/hera/src/master/>.

Imports Rcpp (>= 1.0.1), mvtnorm, Rdpack, methods, stats

Suggests TDA, knitr, rmarkdown, SparseM, Matrix, kernlab, viridis

LinkingTo Rcpp, RcppEigen, BH

RoxygenNote 6.1.1

VignetteBuilder knitr

RdMacros Rdpack

NeedsCompilation yes

Author Tullia Padellini [aut, cre],
Francesco Palini [aut],
Pierpaolo Brutti [ctb],
David Meyer [ctb, cph] (libsvm to R code (e1071 package)),
Chih-Chung Chang [ctb, cph] (LIBSVM C++ code),
Chih-Chen Lin [ctb, cph] (LIBSVM C++ code),
Michael Kerber [ctb, cph] (HERA C++ code),

Dmitriy Morozov [ctb, cph] (HERA C++ code),
Arnur Nigmetov [ctb, cph] (HERA C++ code)

Repository CRAN

Date/Publication 2019-07-03 16:20:02 UTC

R topics documented:

`gaus.kernel` *Geodesic Gaussian Kernel (GGK)*

Description

Computes the Geodesic Gaussian Kernel (GGK) between persistence diagrams.

Usage

```
gaus.kernel(d1, d2 = NULL, h, dimension)
```

Arguments

`d1`	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time) or a list of diagrams.
`d2`	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time).
`h`	bandwidth of the kernel.
`dimension`	The dimension of the topological feature (0 for connected components, 1 for cycles etc).

Value

If `d1` is a list of Persistence Diagrams, this function returns a matrix whose (i,j) entry is the GGK computed in (`d1[[i]]`, `d2[[j]]`), otherwise it returns the value for the GGK computed in (`d1`, `d2`).

Author(s)

Tullia Padellini

References

Padellini T, Brutti P (2017). "Supervised Learning with Indefinite Topological Kernels." *arXiv preprint arXiv:1709.07100*.

Examples

```
diag1 <- matrix(c(1,1,1,0,2,3,2,2.5,4), ncol = 3, byrow = FALSE)
diag2 <- matrix(c(1,1,0,1,1,2), ncol = 3, byrow = FALSE)
gaus.kernel(d1 = diag1, d2 = diag2, h = 1, dimension = 1)
```

---

kernelTDA *kernelTDA: Kernels for Persistence Diagrams*

Description

The kerenelTDA package provides implementation for the most common kernels defined in the space of Persistence Diagrams. These kernel can then be used in most kernel based method.

---

krein.svm *Krein Support Vector Machine*

Description

Solves a kernelized Support Vector Machine in the case where the kernel used may not be positive semidefinite.

Usage

```
krein.svm(kernelmat, ...)
```

Arguments

kernelmat	the kernel matrix computed for all observations
...	additional parameters, see krein.svm.default for more details on how to use this function

Details

This function implements the Krein Support Vector Machine solver as defined by Loosli et al. (2015). The implementation of the solver is a modified version of the popular C++ library 'LIBSVM', while the connection to 'R' heavily relies on the 'R'-package **e1701**.

Value

An object of class `krein.svm` containing the fitted model, including:

`SV` a matrix containing the Support Vectors

`index` index of the resulting support vectors in the data matrix

`coefs` a matrix containing corresponding coefficients times the training labels

`rho` value of the (negative) intercept

Author(s)

Tullia Padellini, Francesco Palini, David Meyer. The included C++ library LIBSVM is authored by Chih-Chung Chang and Chih-Jen Lin)

References

Loosli G, Canu S, Ong CS (2015). "Learning SVM in Krein spaces." *IEEE transactions on pattern analysis and machine intelligence*, **38**(6), 1204–1216.

Chang C, Lin C (2011). "LIBSVM: A library for support vector machines." *ACM transactions on intelligent systems and technology (TIST)*, **2**(3), 27.

Dimitriadou E, Hornik K, Leisch F, Meyer D, Weingessel A (2008). "Misc functions of the Department of Statistics (e1071), TU Wien." *R package*, **1**, 5–24.

Examples

```
library(TDA)
set.seed(123)
foo.data = list()
for(i in 1:20){
foo = circleUnif(100)
foo.data[[i]] = ripsDiag(foo, 1,1)$diagram}
for(i in 21:40){
foo = cbind(runif(100), runif(100))
foo.data[[i]] = ripsDiag(foo, 1,1)$diagram
}
GSWkernel = gaus.kernel(foo.data, h =1, dimension = 1,  q = 2)
GGKclass = krein.svm(kernelmat = GSWkernel, y = rep(c(1,2), c(20,20)))
```

---

krein.svm.default *Krein Support Vector Machine*

Description

Solves a kernelized Support Vector Machine in the case where the kernel used may not be positive semidefinite.

Usage

```
## Default S3 method:
krein.svm(kernelmat = NULL, y = NULL, cost = 1,
  class.weights = NULL, cross = 0, probability = FALSE,
  fitted = TRUE, subset, ...)
```

Arguments

kernelmat	the kernel matrix computed for all observations
y	a vector of labels
cost	cost of violating the constraint
class.weights	a named vector of weights for the different classes, used for asymmetric class sizes. Not all factor levels have to be supplied (default weight: 1). All components have to be named. Specifying "inverse" will choose the weights inversely proportional to the class distribution.
cross	number of fold in a k-fold cross validation
probability	logical indicating whether the model should allow for probability predictions (default: FALSE).
fitted	logical indicating whether the fitted values should be computed and included in the model or not (default: TRUE)
subset	an index vector specifying the cases to be used in the training sample. (NOTE: If given, this argument must be named.)
...	additional parameters

Details

This function implements the Krein Support Vector Machine solver as defined by Loosli et al. (2015). The implementation of the solver is a modified version of the popular C++ library 'LIBSVM', while the connection to 'R' heavily relies on the 'R'-package **e1701**.

Value

An object of class krein.svm containing the fitted model, including:

SV a matrix containing the Support Vectors

index index of the resulting support vectors in the data matrix

coefs a matrix containing corresponding coefficients times the training labels

rho value of the (negative) intercept

Author(s)

Tullia Padellini, Francesco Palini, David Meyer. The included C++ library LIBSVM is authored by Chih-Chung Chang and Chih-Jen Lin)

References

Loosli G, Canu S, Ong CS (2015). "Learning SVM in Krein spaces." *IEEE transactions on pattern analysis and machine intelligence*, **38**(6), 1204–1216.

Chang C, Lin C (2011). "LIBSVM: A library for support vector machines." *ACM transactions on intelligent systems and technology (TIST)*, **2**(3), 27.

Dimitriadou E, Hornik K, Leisch F, Meyer D, Weingessel A (2008). "Misc functions of the Department of Statistics (e1071), TU Wien." *R package*, **1**, 5–24.

Examples

```
# library(TDA)
# set.seed(123)
# foo.data = list()
# for(i in 1:20){
#    foo = circleUnif(100)
#    foo.data[[i]] = ripsDiag(foo, 1,1)$diagram}
#    for(i in 21:40){
#     foo = cbind(runif(100), runif(100))
#     foo.data[[i]] = ripsDiag(foo, 1,1)$diagram
#     }
# GSWkernel = gaus.kernel(foo.data, h =1, dimension = 1,  q = 2)
# GGKclass = krein.svm(kernelmat = GSWkernel, y = rep(c(1,2), c(20,20)))
```

`lapl.kernel` *Geodesic Laplacian Kernel (GLK)*

Description

Computes the Geodesic Laplacian Kernel (GLK) between persistence diagrams.

Usage

```
lapl.kernel(d1, d2 = NULL, h, dimension)
```

Arguments

`d1`	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time) or a list of diagrams
`d2`	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time).
`h`	bandwidth of the kernel.
`dimension`	The dimension of the topological feature (0 for connected components, 1 for cycles etc)

Value

If d1 is a list of Persistence Diagrams, this function returns a matrix whose (i,j) entry is the GLK computed in (d1[[i]], d2[[j]]), otherwise it returns the value for the GLK computed in (d1, d2).

Author(s)

Tullia Padellini

References

Padellini T, Brutti P (2017). "Supervised Learning with Indefinite Topological Kernels." *arXiv preprint arXiv:1709.07100*.

Examples

```
diag1 <- matrix(c(1,1,1,0,2,3,2,2.5,4), ncol = 3, byrow = FALSE)
diag2 <- matrix(c(1,1,0,1,1,2), ncol = 3, byrow = FALSE)
lapl.kernel(d1 = diag1, d2 = diag2, h = 1, dimension = 1)
```

---

pers.image *Persistence Image*

Description

Compute the Persistence Image for a given diagram, using piecewise linear weight functions and Gaussian baseline distribution.

Usage

```
pers.image(d1, nbins, dimension, h)
```

Arguments

d1	A persistence diagram, in the form of a matrix with 3 columns (first one is the dimension, second is the birth-time, last one is the death-time).
nbins	Number of bins for the discretization of the Persistence Surface into the Persistence Image.
dimension	Dimension of the topological features of interest (0 for connected components, 1 for cycles etc).
h	Standard deviation of the Gaussian baseline used to compute the Persistence Surface.

Value

a nbins x nbins matrix containing the Persistence Image.

Author(s)

Tullia Padellini

References

Adams H, Emerson T, Kirby M, Neville R, Peterson C, Shipman P, Chepushtanova S, Hanson E, Motta F, Ziegelmeier L (2017). "Persistence images: A stable vector representation of persistent homology." *The Journal of Machine Learning Research*, **18**(1), 218–252.

Examples

```
diag1 <- matrix(c(1,1,1,0,2,3,2,2.5,4), ncol = 3, byrow = FALSE)
colnames(diag1) <- c("dimension", "birth", "death")
pi1 <- pers.image(d1 = diag1, nbins = 20, dimension = 1, h = 1)
image(pi1)
```

---

pf.kernel *Persistence Fisher Kernel (PFK)*

Description

Computes the Persistence Fisher Kernel (PFK) between persistence diagrams.

Usage

```
pf.kernel(d1, d2 = NULL, h, dimension, sigma)
```

Arguments

d1	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time) or a list of diagrams.
d2	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time).
h	bandwidth of the PFK.
dimension	The dimension of the topological feature (0 for connected components, 1 for cycles etc)
sigma	standard deviation of the base Gaussian Kernel.

Value

If d1 is a list of Persistence Diagrams, this function returns a matrix whose (i,j) entry is the PFK computed in (d1[[i]], d2[[j]]), otherwise it returns the value for the PFK computed in (d1, d2).

Author(s)

Tullia Padellini

References

Le T, Yamada M (2018). "Persistence fisher kernel: A riemannian manifold kernel for persistence diagrams." In *Advances in Neural Information Processing Systems*, 10007–10018.

Examples

```
diag1 <- matrix(c(1,1,1,0,2,3,2,2.5,4), ncol = 3, byrow = FALSE)
diag2 <- matrix(c(1,1,0,1,1,2), ncol = 3, byrow = FALSE)
pf.kernel(d1 = diag1, d2 = diag2, h = 1, dimension = 1, sigma = 1)
```

---

pss.kernel *Persistence Scale Space Kernel (PSSK)*

Description

Computes the Persistence Scale Space Kernel (PSSK) between persistence diagrams

Usage

```
pss.kernel(d1, d2 = NULL, h, dimension)
```

Arguments

d1	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time) or a list of diagrams
d2	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time).
h	bandwidth of the kernel.
dimension	The dimension of the topological feature (0 for connected components, 1 for cycles etc).

Value

If d1 is a list of Persistence Diagrams, this function returns a matrix whose (i,j) entry is the PSSK computed in (d1[[i]], d2[[j]]), otherwise it returns the value for the PSSK computed in (d1, d2).

Author(s)

Tullia Padellini

References

Reininghaus J, Huber S, Bauer U, Kwitt R (2015). "A stable multi-scale kernel for topological machine learning." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4741–4748.

Examples

```
diag1 <- matrix(c(1,1,1,0,2,3,2,2.5,4), ncol = 3, byrow = FALSE)
diag2 <- matrix(c(1,1,0,1,1,2), ncol = 3, byrow = FALSE)
pss.kernel(d1 = diag1, d2 = diag2, h = 1, dimension = 1)
```

---

sw.kernel *Persistence Sliced Wasserstein Kernel (SWK)*

Description

Computes the Persistence Sliced Wasserstein Kernel (SWK) between persistence diagrams.

Usage

```
sw.kernel(d1, d2 = NULL, h, dimension, M = 10)
```

Arguments

d1	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time) or a list of diagrams
d2	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time).
h	bandwidth of the kernel
dimension	The dimension of the topological feature (0 for connected components, 1 for cycles etc)
M	number of directions on which to approximate the Sliced Wasserstein Distance

Value

If d1 is a list of Persistence Diagrams, this function returns a matrix whose (i,j) entry is the SWK computed in (d1[[i]], d2[[j]]), otherwise it returns the value for the SWK computed in (d1, d2)

Author(s)

Tullia Padellini

References

Carriere M, Cuturi M, Oudot S (2017). "Sliced wasserstein kernel for persistence diagrams." In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 664–673. JMLR. org.

Examples

```
diag1 <- matrix(c(1,1,1,0,2,3,2,2.5,4), ncol = 3, byrow = FALSE)
diag2 <- matrix(c(1,1,0,1,1,2), ncol = 3, byrow = FALSE)
sw.kernel(d1 = diag1, d2 = diag2, h = 1, dimension = 1)
```

wass.kernel *L_infty q-Wasserstein Kernel (WK)*

Description

Computes the L_infty q-Wasserstein Kernel (WK) between persistence diagrams.

Usage

```
wass.kernel(d1, d2 = NULL, h, dimension, q)
```

Arguments

d1	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time) or a list of diagrams
d2	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time).
h	bandwidth of the kernel.
dimension	The dimension of the topological feature (0 for connected components, 1 for cycles etc).
q	order of the q-Wasserstein distance.

Value

If d1 is a list of Persistence Diagrams, this function returns a matrix whose (i,j) entry is the WK computed in (d1[[i]], d2[[j]]), otherwise it returns the value for the WK computed in (d1, d2).

Author(s)

Tullia Padellini

Examples

```
diag1 <- matrix(c(1,1,1,0,2,3,2,2.5,4), ncol = 3, byrow = FALSE)
diag2 <- matrix(c(1,1,0,1,1,2), ncol = 3, byrow = FALSE)
wass.kernel(d1 = diag1, d2 = diag2, h = 1, dimension = 1, q = 2)
```

`wasserstein.distance` *L_p q-Wasserstein Distance*

Description

Compute the q-Wasserstein distance between persistence diagrams using an arbitrary L_p norm as ground metric.

Usage

```
wasserstein.distance(d1, d2, dimension, q, p = 2)
```

Arguments

`d1`	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time).
`d2`	A persistence diagram (matrix with 3 col where the first one is the dimension, the second is the birth-time and the third is the death-time).
`dimension`	Dimension of the topological features of interest (0 for connected components, 1 for cycles etc).
`q`	Order of the q-Wasserstein distance.
`p`	Order of the L_p norm to be used as a ground metric in the computation of the Wasserstein distance.

Details

This function provides an R interface for the efficient C++ library 'HERA' by Michael Kerber, Dmitriy Morozov, and Arnur Nigmetov (https://bitbucket.org/grey_narn/hera/src/master/).

Value

The value for the L_p q-Wassesterstein between `d1` and `d2`.

Author(s)

Tullia Padellini, Francesco Palini. The included C++ library is authored by Michael Kerber, Dmitriy Morozov, and Arnur Nigmetov

References

Kerber M, Morozov D, Nigmetov A (2017). "Geometry helps to compare persistence diagrams." *Journal of Experimental Algorithmics (JEA)*, **22**, 1–4.

Examples

```
diag1 <- matrix(c(1,1,1,0,2,3,2,2.5,4), ncol = 3, byrow = FALSE)
diag2 <- matrix(c(1,1,0,1,1,2), ncol = 3, byrow = FALSE)
wasserstein.distance(diag1, diag2, dimension = 1, q = 1, p = 2)
```

Index