Package ‘shattering’

October 10, 2020

Title Estimate the Shattering Coefficient for a Particular Dataset

Version 1.0.2

Description The Statistical Learning Theory (SLT) provides the theoretical background to ensure that a supervised algorithm generalizes the mapping f:X -> Y given f is selected from its search space bias F. This formal result depends on the Shattering coefficient function N(F,2n) to upper bound the empirical risk minimization principle, from which one can estimate the necessary training sample size to ensure the probabilistic learning convergence and, most importantly, the characterization of the capacity of F, including its under and overfitting abilities while addressing specific target problems. In this context, we propose a new approach to estimate the maximal number of hyperplanes required to shatter a given sample, i.e., to separate every pair of points from one another, based on the recent contributions by Har-Peled and Jones in the dataset partitioning scenario, and use such foundation to analytically compute the Shattering coefficient function for both binary and multiclass problems. As main contributions, one can use our approach to study the complexity of the search space bias F, estimate training sample sizes, and parametrize the number of hyperplanes a learning algorithm needs to address some supervised task, what is specially appealing to deep neural networks. Reference: de Mello, R.F. (2019) ``On the Shattering Coefficient of Supervised Learning Algorithms" <arXiv:1911.05461>; de Mello, R.F., Ponti, M.A. (2018, ISBN: 978-3319949888) ``Machine Learning: A Practical Approach on the Statistical Learning Theory".

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1.9000

Imports FNN, pdist, slam, grDevices, base, Ryacas, rmarkdown, pracma, graphics

Suggests testthat

NeedsCompilation no

Author Rodrigo F. de Mello [aut, cre] (<https://orcid.org/0000-0002-0435-3992>)

Maintainer Rodrigo F. de Mello <mello@icmc.usp.br>

Repository CRAN

Date/Publication 2020-10-10 10:00:02 UTC

R topics documented:

`complexity_analysis` *Produce a PDF report analyzing the lower and upper shattering coefficient functions*

Description

Full analysis on the lower and upper shattering coefficient functions for a given supervised dataset

Usage

```
complexity_analysis(
  X = NULL,
  Y = NULL,
  my.delta = 0.05,
  my.epsilon = 0.05,
  directory = tempdir(),
  file = "myreport",
  length = 5,
  quantile.percentage = 0.5,
  epsilon = 1e-07
)
```

Arguments

`X`	matrix defining the input space of your dataset
`Y`	numerical vector defining the output space (labels/classes) of your dataset
`my.delta`	upper bound for the probability of the empirical risk minimization principle (in range (0,1))
`my.epsilon`	acceptable divergence between the empirical and (expected) risks (in range (0,1))
`directory`	directory used to generate the report for your dataset
`file`	name of the PDF file to be generated (without extension)
`length`	number of points to divide the sample while computing the shattering coefficient
`quantile.percentage`	real number to define the quantile of distances to be considered (e.g. 0.1 means 10%)
`epsilon`	a real threshold to be removed from distances in order to measure the open balls in the underlying topology

Value

A list including the number of hyperplanes and the shattering coefficient function. A report is generated in the user-defined directory.

References

de Mello, R.F. (2019) "On the Shattering Coefficient of Supervised Learning Algorithms" arXiv:https://arxiv.org/abs/1911.05461

de Mello, R.F., Ponti, M.A. (2018, ISBN: 978-3319949888) "Machine Learning: A Practical Approach on the Statistical Learning Theory"

Examples

```
# Analyzing the complexity of the shattering coefficients functions
#  (lower and upper bounds) for the Iris dataset
# require(datasets)
# complexity_analysis(X=as.matrix(iris[,1:4]), Y=as.numeric(iris[,5]))
```

---

compress_space   *Function to compress the space based on the equivalence relations.*

Description

This function compresses the input space according to the equivalence relations, i.e., it compresses whenever an example has other elements inside its open ball but having the same class label as the ball-centered instance.

Usage

```
compress_space(M, Y)
```

Arguments

M	sparse matrix representing all equivalence relations
Y	numerical vector indentifying the output space of variables

Value

A list containing sparse vectors (from package slam) identifying the equivalence relations

equivalence_relation *Function to compute equivalence relations among input space points.*

Description

This function computes the greatest as possible open ball connecting a given input example to every other under the same class label, thus homogeneizing space regions.

Usage

```
equivalence_relation(
  X,
  Y,
  quantile.percentage = 1,
  epsilon = 1e-07,
  chunk = 250
)
```

Arguments

`X`	matrix indentifying the input space of variables
`Y`	numerical vector indentifying the output space of variables
`quantile.percentage`	real number to define the quantile of distances to be considered (e.g. 0.1 means 10%)
`epsilon`	a real threshold to be removed from distances in order to measure the open balls in the underlying topology
`chunk`	number of elements to compute the Euclidean distances at once (if you set a large number, you might have memory limitations to perform the operations)

Value

A list with the equivalence relations in form of a list

estimate_number_hyperplanes *Function to estimate the number of hyperplanes required to classify such a data sample.*

Description

This function estimates the number of hyperplanes

Usage

```
estimate_number_hyperplanes(
  X,
  Y,
  length = 20,
  quantile.percentage = 0.05,
  epsilon = 1e-07
)
```

Arguments

X	matrix indentifying the input space of variables
Y	numerical vector indentifying the output space of variables
length	number of data points used to estimate the shattering coefficient
quantile.percentage	real number to define the quantile of distances to be considered (e.g. 0.1 means 10%)
epsilon	a real threshold to be removed from distances in order to measure the open balls in the underlying topology

Value

A data frame whose columns are: (1) the original sample size; (2) the reduced sample size after connecting homogeneous space regions; (3) the lower bound for the number of hyperplanes required to shatter the input space; and (4) the upper bound for the number of hyperplanes required to shatter the input space

Examples

```
# Generating some random dataset with 2 classes:
# 50 examples in class 1 and 50 in class 2 (last column)
data = cbind(rnorm(mean=1, sd=1, n=50), rnorm(mean=1, sd=1, n=50), rep(1, 50))
data = rbind(data, cbind(rnorm(mean=-1, sd=1, n=50), rnorm(mean=-1, sd=1, n=50), rep(2, 50)))

# Building up the input and output sets
X = data[,1:2]
Y = data[,3]

# Plotting our dataset using classes as colors
plot(X, col=Y, main="Original dataset", xlab="Attribute 1", ylab="Attribute 2")

# Here we estimate the number of hyperplanes required to shatter (divide) the given sample
# in all possible ways according to the organization of points in the input space
Hyperplanes = estimate_number_hyperplanes(X, Y, length=10, quantile.percentage=0.1, epsilon=1e-7)
```

`shattering` *shattering: A package to estimate the shattering coefficient for labeled data samples.*

Description

Description: The Statistical Learning Theory (SLT) provides the theoretical background to ensure that a supervised algorithm generalizes the mapping f:X -> Y given f is selected from its search space bias F. This formal result depends on the Shattering coefficient function N(F,2n) to upper bound the empirical risk minimization principle, from which one can estimate the necessary training sample size to ensure the probabilistic learning convergence and, most importantly, the characterization of the capacity of F, including its under and overfitting abilities while addressing specific target problems. In this context, we propose a new approach to estimate the maximal number of hyperplanes required to shatter a given sample, i.e., to separate every pair of points from one another, based on the recent contributions by Har-Peled and Jones in the dataset partitioning scenario, and use such foundation to analytically compute the Shattering coefficient function for both binary and multi-class problems. As main contributions, one can use our approach to study the complexity of the search space bias F, estimate training sample sizes, and parametrize the number of hyperplanes a learning algorithm needs to address some supervised task, what is specially appealing to deep neural networks. Reference: https://arxiv.org/abs/1911.05461

References

de Mello, R.F. (2019) "On the Shattering Coefficient of Supervised Learning Algorithms" arXiv:`https://arxiv.org/abs/1911.05461`

de Mello, R.F., Ponti, M.A. (2018, ISBN: 978-3319949888) "Machine Learning: A Practical Approach on the Statistical Learning Theory"

Shattering functions

This packages comes with functions to estimate the shattering coefficient.

Index