

Package ‘LVQTools’

February 14, 2012

Type Package

Title Learning Vector Quantization Tools

Version 1.0

Date 2010-08-14

Author Sander Kelders <sakelders@gmail.com>

Maintainer Sander Kelders <sakelders@gmail.com>

Description This package provides several LVQ algorithms along with different usage methods and output visualization.

Depends methods

License GPL-2

Repository CRAN

Date/Publication 2010-08-30 12:42:44

R topics documented:

LVQTools-package	2
getCostcurve	2
getPrototypeProgress	3
getPrototypes	4
getRelevanceProgress	5
getRelevances	6
getTestError	7
getTestErrorProgress	7
getTrainError	8
getTrainErrorProgress	9
ifoldoutput-class	10
show	11
showAll	12
trainoutput-class	13
traintestoutput-class	14
validate	15

Index**19**

LVQTools-package

*Learning Vector Quantization Tools***Description**

This package provides several LVQ algorithms along with different usage methods and output visualization.

Details

Package: LVQTools
 Type: Package
 Version: 1.0
 Date: 2010-08-14
 License: GPL-2
 LazyLoad: yes

See package documentation: LVQTools Bachelor project: implementing LVQ in R, LVQTools Usage: implementing LVQ in R, LVQTools Output Usage: implementing LVQ in R, LVQTools Documentation: implementing LVQ in R.

Author(s)

Sander Kelders

Maintainer: Who to complain to <sakelders@gmail.com>

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate](#) [show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

getCostcurve

*getCostcurve***Description**

This function returns the progress of the costfunction.

Usage

```
getCostcurve(LVQout, fold = NA)
```

Arguments

LVQout	The output-class containing the output to be returned.
fold	Determines from which subset the output is to be returned.

Value

Either a vector containing the values of the costfunction at after each epoch or a list of such vectors in the case of nfoldcross validation.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

`getPrototypeProgress` *getPrototypeProgress*

Description

This function returns the progress of the prototypes.

Usage

```
getPrototypeProgress(LVQout, fold = NA)
```

Arguments

LVQout	The output-class containing the output to be returned.
fold	Determines from which subset the output is to be returned.

Value

A list of matrices containing the values of the prototypes with classlabels attached after each epoch, or a list of such lists in case of nfoldcross validation.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

getPrototypes

getPrototypes

Description

This function returns the end-configuration of the prototypes.

Usage

```
getPrototypes(LVQout, fold = NA)
```

Arguments

LVQout	The output-class containing the output to be returned.
fold	Determines from which subset the output is to be returned.

Value

A matrix of prototype endvalues with the classlabels attached or a list of such matrices in case of nfoldcross validation.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

getRelevanceProgress *getRelevanceProgress*

Description

This function returns the progress of the relevances.

Usage

```
getRelevanceProgress(LVQout, fold = NA)
```

Arguments

LVQout	The output-class containing the output to be returned.
fold	Determines from which subset the output is to be returned.

Value

A list of relevances (which is a vector or a matrix depending on the variant of relevances used) containing the relevance values after each epoch. Or a list of such lists in case of `nfoldcross` validation or the use of local or classwise relevances. Or a list of lists of such lists in case of both `nfoldcross` validation and the use of local or classwise relevances.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate](#), [show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

getRelevances	<i>getRelevances</i>
---------------	----------------------

Description

This function returns the end-configuration of the relevances.

Usage

```
getRelevances(LVQout, fold = NA)
```

Arguments

LVQout	The output-class containing the output to be returned.
fold	Determines from which subset the output is to be returned.

Value

A vector or matrix (depending on the relevance version used) containing the end relevance values. Or a list of such vectors or matrices in case of nfoldcross validation or the use of local or classwise relevances. Or a list of lists of such vectors or matrices in case of both nfoldcross validation and the use of local or classwise relevances.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate](#), [show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

getTestError *getTestError*

Description

This function returns the number of missclassifications that were encountered when classifying the testset with the prototype end-configuration.

Usage

```
getTestError(LVQout, fold = NA)
```

Arguments

LVQout	The output-class containing the output to be returned.
fold	Determines from which subset the output is to be returned.

Value

A numeric which is the testerror after the last epoch.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate](#), [show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

getTestErrorProgress *getTestErrorProgress*

Description

This function returns the number of missclassifications that were encountered when classifying the testset with all the configurations of the prototypes.

Usage

```
getTestErrorProgress(LVQout, fold = NA)
```

Arguments

LVQout	The output-class containing the output to be returned.
fold	Determines from which subset the output is to be returned.

Value

A vector containing the testerror after each epoch.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

getTrainError

getTrainError

Description

This function returns the number of missclassifications that were encountered when classifying the trainingset with the prototype end-configuration.

Usage

```
getTrainError(LVQout, fold = NA)
```

Arguments

LVQout	The output-class containing the output to be returned.
fold	Determines from which subset the output is to be returned.

Value

A numeric which is the trainerror after the last epoch.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

`getTrainErrorProgress` *getTrainErrorProgress*

Description

This function returns the number of missclassifications that were encountered when classifying the trainingset with all the configurations of the prototypes.

Usage

```
getTrainErrorProgress(LVQout, fold = NA)
```

Arguments

LVQout	The output-class containing the output to be returned.
fold	Determines from which subset the output is to be returned.

Value

A vector containing the trainerror after each epoch.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

nfoldoutput-class *Class "nfoldoutput"*

Description

This class contains all the output of a `validate -run` according to `nfoldercrossvalidation`.

Slots

prototypes: Object of class "list" ~~
relevances: Object of class "list" ~~
costcurve: Object of class "list" ~~
protoprogress: Object of class "list" ~~
relevanceprogress: Object of class "list" ~~
trainerror: Object of class "vector" ~~
testerror: Object of class "vector" ~~
trainerrorprogress: Object of class "list" ~~
testerrorprogress: Object of class "list" ~~
nfold: Object of class "numeric" ~~
nrofrelevances: Object of class "numeric" ~~

Methods

No methods defined with class "nfoldoutput" in the signature.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate](#), [show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

show	<i>show</i>
------	-------------

Description

This function shows all the selected output. The selection is made by providing the appropriate parameters with TRUE.

Usage

```
show(LVQoutput, prototypes = FALSE, relevances = FALSE, costcurve = FALSE, prototypeprogress = FALSE, re
```

Arguments

LVQoutput	The output-class containing all the output.
prototypes	This determines if the end-configuration of the prototypes should be among the output.
relevances	This determines if the end-configuration of the relevances should be among the output.
costcurve	This determines if the progress of the costfunction should be among the output.
prototypeprogress	This determines if all the configurations of the prototypes should be among the output.
relevanceprogress	This determines if all the configurations of the relevances should be among the output.
trainerror	This determines if the number of missclassifications when testing the end-configuration of the prototypes with the testset should be among the output.
testerror	This determines if the number of missclassifications when testing the end-configuration of the prototypes with the testset should be among the output.
trainerrorprogress	This determines if the progress of the number of missclassifications when testing the end-configuration of the prototypes with the trainingset should be among the output.
testerrorprogress	This determines if the progress of the number of missclassifications when testing the end-configuration of the prototypes with the testset should be among the output.
protofold	Selects which prototype-end-configuration should be shown. -1 selects all.
relfold	Selects which relevance-end-configuration should be shown. -1 selects all.
costfold	Selects which costfunction-progress should be shown. -1 selects all.
protoproifold	Selects which prototype-progress should be shown. -1 selects all.
relproifold	Selects which prototype-progress should be shown. -1 selects all.

`trainerrorprogfold`
 Selects which trainerror-progress should be shown. -1 selects all.
`testerrorprogfold`
 Selects which testerror-progress should be shown. -1 selects all.
`relevancenumber`
 Selects which relevances should be shown in the case of classwise or local relevances. -1 selects all.
`relevanceprognumber`
 Selects of which relevances the progress should be shown in the case of classwise or local relevances. -1 selects all.

Value

Graphics and/or values printed to screen depending on the output class and the chosen options.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

`showAll`

showAll

Description

This function shows all the available output, by calling `show` with all the appropriate parameters to `TRUE`.

Usage

```
showAll(LVQoutput)
```

Arguments

`LVQoutput` The output-class containing all the output.

Value

Graphics and/or values printed to screen depending on the output object.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate](#), [show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

trainoutput-class *Class "trainoutput"*

Description

This class contains all the output of a validate-run according to the train-scheme.

Slots

prototypes: Object of class "matrix" ~~
relevances: Object of class "vector" ~~
costcurve: Object of class "vector" ~~
protoprogress: Object of class "list" ~~
relevanceprogress: Object of class "list" ~~
trainerror: Object of class "integer" ~~
trainerrorprogress: Object of class "vector" ~~
nrofrelevances: Object of class "numeric" ~~

Methods

No methods defined with class "trainoutput" in the signature.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate](#), [show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

traintestoutput-class *Class "traintestoutput"*

Description

This class contains all the output of a validate-run according to the traintest-scheme.

Slots

prototypes: Object of class "matrix" ~~
relevances: Object of class "vector" ~~
costcurve: Object of class "vector" ~~
protoprogress: Object of class "list" ~~
relevanceprogress: Object of class "list" ~~
trainerror: Object of class "integer" ~~
testerror: Object of class "integer" ~~
trainerrorprogress: Object of class "vector" ~~
testerrorprogress: Object of class "vector" ~~
nrofrelevances: Object of class "numeric" ~~

Methods

No methods defined with class "traintestoutput" in the signature.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate](#), [show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

validate *Learning vector Quantization validation*

Description

This is the entripoint of the LVQTools. This function performs LVQ and validation according to the given parameters.

Usage

```
validate(validatescheme = "train", testdatapath = NA, nfold = 8, LVQscheme = "LVQ1", optimisationscheme
```

Arguments

- validatescheme** Determines how training and testing is to be executed. **train** only trains the prototypes, while **traintest** also tests the prototypes after training with a different set of testdata. The **nfold**-scheme will apply **nfoldcross-validation** with the **nfold**-parameter determining in how many sets the data is to be divided. The sets are divided randomly without consideration to class. Available values: **train**, **traintest**, **nfold**. Default value: **train**
- testdatapath** he location of the file containing data used for testing. If this value is **NA** **testinp** will be used, otherwise the specified file will be used. Default value: **NA**.
- nfold** Determines the number of sets the data is divided in when using **nfoldcross-validation**. Available values: any whole positive number in the range of $[2 \dots \text{number of datapoints}]$. Default value: **8**.
- LVQscheme** Determines which version of LVQ is used. Together with **distscheme**, **relevancemode**, **relevancescheme** and **optimisationscheme** this determines the complete **distancemeasure**. Available values: **LVQ1**, **cauchyschwarz**, **renyi**. Default value: **LVQ1**.
- optimisationscheme** Determines which type of **costfunction** is used and thus how the prototypes are updated. The **normal** **optimisationscheme** uses the winner takes all principle and only updates the closest prototype. The **general** **optimisationscheme** is used for generalized LVQ. It uses stochastic gradient descent to determine the prototype updates. The following function is used for this purpose: $\Sigma_i \Phi(\mu)$ with $\mu = \frac{d_J^\Lambda - d_K^\Lambda}{d_J^\Lambda + d_K^\Lambda}$. And with d_J^Λ as the distance to the nearest prototype of the appropriate class and d_K^Λ as the distance to the nearest prototype of another class. Available uses: **normal**, **general**. Default value: **normal**.
- inp** The data used for training. If **datapath** is **NA** **inp** will be used, otherwise **datapath** will be used. Default value: **NA**.
- testinp** The data used for training. If **datapath** is **NA** **inp** will be used, otherwise **datapath** will be used. Default value: **NA**.
- prototypeoutput** When **TRUE** records the **endconfiguration** of the prototypes of a training and returns it (possibly among other things) after terminating. Possible values: **TRUE**, **FALSE**. Default value: **TRUE**.

relevanceoutput	When TRUE records the endconfiguration of the relevance-vector or matrix of a training and returns it (possibly among other things) after terminating. Possible values: TRUE, FALSE . Default value: FALSE .
costcurve	When TRUE the cost is calculated after each epoch and the value stored. When the program ends this cost (possibly among other things) is returned. Available values: TRUE, FALSE . Default value: FALSE .
progress	When TRUE records the value of all prototypes before the first and after each epoch and returns it (possibly among other things) after terminating. Possible values: TRUE, FALSE . Default value: FALSE .
relevanceprogress	When TRUE records the value of the relevance-vector or matrix before the first and after each epoch and returns it (possibly among other things) after terminating. Possible values: TRUE, FALSE . Default value: FALSE .
trainerror	After training tests with the trainingset and stores the number of missclassifications and returns (possibly among other things) it after terminating. Possible values: TRUE, FALSE . Default value: FALSE .
testerror	When testing with a different set than the trainingset stores the number of missclassifications after training and returns (possibly among other things) it after terminating. Possible values: TRUE, FALSE . Default value: FALSE .
trainerrorprogress	When TRUE the trainerror is calculated after every epoch and stored to be returned (possibly) among other output.
testerrorprogress	When using testdata to test the outcome of a training and testerrorprogress is TRUE calculates the testerror after very epoch and stores it to return (possibly) among other output.
datapath	The location of the file containing data used for training or nfoldcrossvalidation. If this value is NA inp will be used, otherwise the specified file will be used. Default value: NA .
normalizescheme	Determines how the data is normalized. ztransform subtracts the mean and divides by variance. iqr subtracts the median and divides by the InterQuantile Range. sumone makes each datapoint sum up to one by dividing all values by the datapoint's sum. Available schemes: ztransform, iqr, sumone, none . Default value: none
normalclasswise	Determines the class on which the normalisation is based for classwise normalisation. Default value: none .
replaceNA	Determines whether or not the NA values in the input will be replaced. If TRUE they will be replaced by the overall median, unless classwise replacement is used. Available values: TRUE, FALSE . Default value: FALSE .
replaceclasswise	Determines whether or not the replacement of NA values will be classwise. If TRUE NA values will be replaced by the median of the class to which the datapoint belongs to, otherwise the overall median is used.

prototypes	Determines the number of prototypes for each class. This vector must have entries accessible by strings representing the classlabels. Each entry lists the number of prototypes for the class whose label was used for accessing it. A usable default value is not present. This parameter has to be specified manually.
learningrate	Determines the rate at which the prototypes are adjusted. This can be a single value to be used throughout the whole training process or a vector of length epochs , which will use each value once in order. Default value: 0.01 .
epochs	The number of epochs used in training. Default value: 10 .
initscheme	Determines the way the prototypes are initialized. mean initializes all prototypes at the mean of all the datapoints. randomsample initializes all prototypes by selecting a different random sample for each prototype and using its values for initialisation. randomwindow initializes all prototypes by constructing a window which includes all datapoints and initialising each prototype randomly within this window. zero initializes all prototypes by setting all values to 0 . Available schemes: mean , randomsample , randomwindow , zero . Default scheme: zero .
distscheme	The distance measure used for determining the difference between prototype and datapoint. Together with LVQscheme , relevance mode , relevance scheme and optimisationscheme this determines the complete distance measure. When using scheme custom a custom difference measure can be used by setting the customdist parameter. The distscheme variable is only used in conjunction with the LVQ1 LVQ scheme and not in conjunction with cauchyschwarz or renyi . Available schemes: manhattan , euclidean , custom . Default scheme: euclidean .
relevance mode	Determines if relevances should be used or not. normal mode does not use relevances at all. relevance mode uses a relevance vector to assign relevances to each dimension. matrix mode uses a square relevance matrix to assign relevances to dimensions and correlations between them. When using mode matrix only euclidean distance scheme is available. Relevances are not available when using cauchyschwarz - or renyi -LVQ scheme. Available values: normal , relevance , matrix . Default value: normal .
relevance scheme	Determines how many different sets of relevances should be used. When using global -relevances only 1 set of relevances is used for all prototypes. When using local -relevances, each prototype has its own set of relevances. When using classwise -relevances all prototypes of the same class share a set of relevances. Available values: global , local , classwise . Default value: global .
relevances	When mode relevance or matrix is used this parameter contains the relevance vector or matrix respectively. The relevances can be specified manually using this parameter or when no relevances are provided they will be randomly initialized. Default value: NA .
relrate	When using relevances determines the rate at which the relevance vector or matrix adapts. This can be a single value to be used throughout the whole training process or a vector of length epochs , which will use each value once in order. Default value: 0.001
customdist	When using distance measure custom , this parameter determines the distance measure used. customdist is p in $\sqrt[p]{ datapoint - prototype ^p}$

alfa	A variable used only in conjunction with the renyi LVQscheme. Determines the variant of renyi-divergence to be used.
show	When TRUE prints the prototype configuration and if applicable the relevance-vector or matrix and the costcurve to the console. Available values: TRUE , FALSE . Default value: FALSE .
graphics	When TRUE and the data is 2-dimensional the progress of the prototypes will be plotted after every epoch. Available values: TRUE , FALSE . Default value: FALSE .
plotcurve	When TRUE and costcurve is also set to TRUE the costcurve will be plotted after every training. Available values: TRUE , FALSE . Default value: FALSE .

Value

One of three object: either `nfoldoutput`, `trainoutput`, or `traintestouput` which contain the selected output.

Author(s)

Sander Kelders

References

LVQTools Bachelor project: implementing LVQ in R

See Also

[validate](#), [show](#), [showAll](#), [getCostcurve](#), [getPrototypeProgress](#), [getPrototypes](#), [getRelevanceProgress](#), [getRelevances](#), [getTestErrorProgress](#), [getTestError](#), [getTrainErrorProgress](#), [getTrainError](#), [nfoldoutput-class](#), [trainoutput-class](#), [traintestoutput-class](#)

Index

`getCostcurve`, [2](#), [2–10](#), [12–14](#), [18](#)
`getPrototypeProgress`, [2](#), [3](#), [3–10](#), [12–14](#),
[18](#)
`getPrototypes`, [2](#), [3](#), [4](#), [4–10](#), [12–14](#), [18](#)
`getRelevanceProgress`, [2–4](#), [5](#), [5–10](#), [12–14](#),
[18](#)
`getRelevances`, [2–5](#), [6](#), [6–10](#), [12–14](#), [18](#)
`getTestError`, [2–6](#), [7](#), [7–10](#), [12–14](#), [18](#)
`getTestErrorProgress`, [2–6](#), [7](#), [7–10](#), [12–14](#),
[18](#)
`getTrainError`, [2–7](#), [8](#), [8–10](#), [12–14](#), [18](#)
`getTrainErrorProgress`, [2–8](#), [9](#), [9](#), [10](#),
[12–14](#), [18](#)

`LVQTools` (`LVQTools-package`), [2](#)
`LVQTools-package`, [2](#)

`ifoldoutput-class`, [2–10](#), [12–14](#), [18](#)
`ifoldoutput-class`, [10](#)

`show`, [2–10](#), [11](#), [12–14](#), [18](#)
`showAll`, [2–10](#), [12](#), [12–14](#), [18](#)

`trainoutput-class`, [2–10](#), [12–14](#), [18](#)
`trainoutput-class`, [13](#)
`traintestoutput-class`, [2–10](#), [12–14](#), [18](#)
`traintestoutput-class`, [14](#)

`validate`, [2–10](#), [12–14](#), [15](#), [18](#)