JavaScript is disabled on your browser.

* [Overview](http://docs.google.com/overview-summary.html)
* [Package](http://docs.google.com/package-summary.html)
* Class
* [Tree](http://docs.google.com/package-tree.html)
* [Index](http://docs.google.com/index-all.html)
* [Help](http://docs.google.com/help-doc.html)
* [Prev Class](http://docs.google.com/org/opencv/ml/SVMSGD.html)
* Next Class
* [Frames](http://docs.google.com/index.html?org/opencv/ml/TrainData.html)
* [No Frames](http://docs.google.com/TrainData.html)
* [All Classes](http://docs.google.com/allclasses-noframe.html)
* Summary:
* Nested |
* Field |
* Constr |
* [Method](#3znysh7)
* Detail:
* Field |
* Constr |
* [Method](#tyjcwt)

org.opencv.ml

## Class TrainData

* java.lang.Object
  + org.opencv.ml.TrainData
* public class TrainData  
  extends java.lang.Object  
  Class encapsulating training data. Please note that the class only specifies the interface of training data, but not implementation. All the statistical model classes in \_ml\_ module accepts Ptr<TrainData> as parameter. In other words, you can create your own class derived from TrainData and pass smart pointer to the instance of this class into StatModel::train. SEE: REF: ml\_intro\_data

### Method SummaryMethods

| Modifier and Type | Method and Description |
| --- | --- |
| static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) | [**\_\_fromPtr\_\_**](http://docs.google.com/org/opencv/ml/TrainData.html#__fromPtr__(long))(long addr) |
| static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) | [**create**](http://docs.google.com/org/opencv/ml/TrainData.html#create(org.opencv.core.Mat,%20int,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, int layout, [Mat](http://docs.google.com/org/opencv/core/Mat.html) responses) Creates training data from in-memory arrays. |
| static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) | [**create**](http://docs.google.com/org/opencv/ml/TrainData.html#create(org.opencv.core.Mat,%20int,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, int layout, [Mat](http://docs.google.com/org/opencv/core/Mat.html) responses, [Mat](http://docs.google.com/org/opencv/core/Mat.html) varIdx) Creates training data from in-memory arrays. |
| static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) | [**create**](http://docs.google.com/org/opencv/ml/TrainData.html#create(org.opencv.core.Mat,%20int,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, int layout, [Mat](http://docs.google.com/org/opencv/core/Mat.html) responses, [Mat](http://docs.google.com/org/opencv/core/Mat.html) varIdx, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sampleIdx) Creates training data from in-memory arrays. |
| static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) | [**create**](http://docs.google.com/org/opencv/ml/TrainData.html#create(org.opencv.core.Mat,%20int,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, int layout, [Mat](http://docs.google.com/org/opencv/core/Mat.html) responses, [Mat](http://docs.google.com/org/opencv/core/Mat.html) varIdx, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sampleIdx, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sampleWeights) Creates training data from in-memory arrays. |
| static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) | [**create**](http://docs.google.com/org/opencv/ml/TrainData.html#create(org.opencv.core.Mat,%20int,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, int layout, [Mat](http://docs.google.com/org/opencv/core/Mat.html) responses, [Mat](http://docs.google.com/org/opencv/core/Mat.html) varIdx, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sampleIdx, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sampleWeights, [Mat](http://docs.google.com/org/opencv/core/Mat.html) varType) Creates training data from in-memory arrays. |
| int | [**getCatCount**](http://docs.google.com/org/opencv/ml/TrainData.html#getCatCount(int))(int vi) |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getCatMap**](http://docs.google.com/org/opencv/ml/TrainData.html#getCatMap())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getCatOfs**](http://docs.google.com/org/opencv/ml/TrainData.html#getCatOfs())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getClassLabels**](http://docs.google.com/org/opencv/ml/TrainData.html#getClassLabels())() Returns the vector of class labels The function returns vector of unique labels occurred in the responses. |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getDefaultSubstValues**](http://docs.google.com/org/opencv/ml/TrainData.html#getDefaultSubstValues())() |
| int | [**getLayout**](http://docs.google.com/org/opencv/ml/TrainData.html#getLayout())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getMissing**](http://docs.google.com/org/opencv/ml/TrainData.html#getMissing())() |
| int | [**getNAllVars**](http://docs.google.com/org/opencv/ml/TrainData.html#getNAllVars())() |
| void | [**getNames**](http://docs.google.com/org/opencv/ml/TrainData.html#getNames(java.util.List))(java.util.List<java.lang.String> names) Returns vector of symbolic names captured in loadFromCSV() |
| long | [**getNativeObjAddr**](http://docs.google.com/org/opencv/ml/TrainData.html#getNativeObjAddr())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getNormCatResponses**](http://docs.google.com/org/opencv/ml/TrainData.html#getNormCatResponses())() |
| int | [**getNSamples**](http://docs.google.com/org/opencv/ml/TrainData.html#getNSamples())() |
| int | [**getNTestSamples**](http://docs.google.com/org/opencv/ml/TrainData.html#getNTestSamples())() |
| int | [**getNTrainSamples**](http://docs.google.com/org/opencv/ml/TrainData.html#getNTrainSamples())() |
| int | [**getNVars**](http://docs.google.com/org/opencv/ml/TrainData.html#getNVars())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getResponses**](http://docs.google.com/org/opencv/ml/TrainData.html#getResponses())() |
| int | [**getResponseType**](http://docs.google.com/org/opencv/ml/TrainData.html#getResponseType())() |
| void | [**getSample**](http://docs.google.com/org/opencv/ml/TrainData.html#getSample(org.opencv.core.Mat,%20int,%20float))([Mat](http://docs.google.com/org/opencv/core/Mat.html) varIdx, int sidx, float buf) |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getSamples**](http://docs.google.com/org/opencv/ml/TrainData.html#getSamples())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getSampleWeights**](http://docs.google.com/org/opencv/ml/TrainData.html#getSampleWeights())() |
| static [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getSubMatrix**](http://docs.google.com/org/opencv/ml/TrainData.html#getSubMatrix(org.opencv.core.Mat,%20org.opencv.core.Mat,%20int))([Mat](http://docs.google.com/org/opencv/core/Mat.html) matrix, [Mat](http://docs.google.com/org/opencv/core/Mat.html) idx, int layout) Extract from matrix rows/cols specified by passed indexes. |
| static [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getSubVector**](http://docs.google.com/org/opencv/ml/TrainData.html#getSubVector(org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) vec, [Mat](http://docs.google.com/org/opencv/core/Mat.html) idx) Extract from 1D vector elements specified by passed indexes. |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTestNormCatResponses**](http://docs.google.com/org/opencv/ml/TrainData.html#getTestNormCatResponses())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTestResponses**](http://docs.google.com/org/opencv/ml/TrainData.html#getTestResponses())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTestSampleIdx**](http://docs.google.com/org/opencv/ml/TrainData.html#getTestSampleIdx())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTestSamples**](http://docs.google.com/org/opencv/ml/TrainData.html#getTestSamples())() Returns matrix of test samples |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTestSampleWeights**](http://docs.google.com/org/opencv/ml/TrainData.html#getTestSampleWeights())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTrainNormCatResponses**](http://docs.google.com/org/opencv/ml/TrainData.html#getTrainNormCatResponses())() Returns the vector of normalized categorical responses The function returns vector of responses. |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTrainResponses**](http://docs.google.com/org/opencv/ml/TrainData.html#getTrainResponses())() Returns the vector of responses The function returns ordered or the original categorical responses. |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTrainSampleIdx**](http://docs.google.com/org/opencv/ml/TrainData.html#getTrainSampleIdx())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTrainSamples**](http://docs.google.com/org/opencv/ml/TrainData.html#getTrainSamples())() Returns matrix of train samples transposed. |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTrainSamples**](http://docs.google.com/org/opencv/ml/TrainData.html#getTrainSamples(int))(int layout) Returns matrix of train samples |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTrainSamples**](http://docs.google.com/org/opencv/ml/TrainData.html#getTrainSamples(int,%20boolean))(int layout, boolean compressSamples) Returns matrix of train samples |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTrainSamples**](http://docs.google.com/org/opencv/ml/TrainData.html#getTrainSamples(int,%20boolean,%20boolean))(int layout, boolean compressSamples, boolean compressVars) Returns matrix of train samples |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getTrainSampleWeights**](http://docs.google.com/org/opencv/ml/TrainData.html#getTrainSampleWeights())() |
| void | [**getValues**](http://docs.google.com/org/opencv/ml/TrainData.html#getValues(int,%20org.opencv.core.Mat,%20float))(int vi, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sidx, float values) |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getVarIdx**](http://docs.google.com/org/opencv/ml/TrainData.html#getVarIdx())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getVarSymbolFlags**](http://docs.google.com/org/opencv/ml/TrainData.html#getVarSymbolFlags())() |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getVarType**](http://docs.google.com/org/opencv/ml/TrainData.html#getVarType())() |
| void | [**setTrainTestSplit**](http://docs.google.com/org/opencv/ml/TrainData.html#setTrainTestSplit(int))(int count) Splits the training data into the training and test parts SEE: TrainData::setTrainTestSplitRatio |
| void | [**setTrainTestSplit**](http://docs.google.com/org/opencv/ml/TrainData.html#setTrainTestSplit(int,%20boolean))(int count, boolean shuffle) Splits the training data into the training and test parts SEE: TrainData::setTrainTestSplitRatio |
| void | [**setTrainTestSplitRatio**](http://docs.google.com/org/opencv/ml/TrainData.html#setTrainTestSplitRatio(double))(double ratio) Splits the training data into the training and test parts The function selects a subset of specified relative size and then returns it as the training set. |
| void | [**setTrainTestSplitRatio**](http://docs.google.com/org/opencv/ml/TrainData.html#setTrainTestSplitRatio(double,%20boolean))(double ratio, boolean shuffle) Splits the training data into the training and test parts The function selects a subset of specified relative size and then returns it as the training set. |
| void | [**shuffleTrainTest**](http://docs.google.com/org/opencv/ml/TrainData.html#shuffleTrainTest())() |

### Methods inherited from class java.lang.Objectequals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

### Method Detail

#### \_\_fromPtr\_\_ public static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) \_\_fromPtr\_\_(long addr)

#### create public static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) create([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, int layout, [Mat](http://docs.google.com/org/opencv/core/Mat.html) responses) Creates training data from in-memory arrays.Parameters:samples - matrix of samples. It should have CV\_32F type.layout - see ml::SampleTypes.responses - matrix of responses. If the responses are scalar, they should be stored as a single row or as a single column. The matrix should have type CV\_32F or CV\_32S (in the former case the responses are considered as ordered by default; in the latter case - as categorical) (CV\_32S) containing 0-based variable indices or byte vector (CV\_8U) containing a mask of active variables. vector (CV\_32S) containing 0-based sample indices or byte vector (CV\_8U) containing a mask of training samples. <number\_of\_variables\_in\_responses>`, containing types of each input and output variable. See ml::VariableTypes. Returns:automatically generated

#### create public static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) create([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, int layout, [Mat](http://docs.google.com/org/opencv/core/Mat.html) responses, [Mat](http://docs.google.com/org/opencv/core/Mat.html) varIdx) Creates training data from in-memory arrays.Parameters:samples - matrix of samples. It should have CV\_32F type.layout - see ml::SampleTypes.responses - matrix of responses. If the responses are scalar, they should be stored as a single row or as a single column. The matrix should have type CV\_32F or CV\_32S (in the former case the responses are considered as ordered by default; in the latter case - as categorical)varIdx - vector specifying which variables to use for training. It can be an integer vector (CV\_32S) containing 0-based variable indices or byte vector (CV\_8U) containing a mask of active variables. vector (CV\_32S) containing 0-based sample indices or byte vector (CV\_8U) containing a mask of training samples. <number\_of\_variables\_in\_responses>`, containing types of each input and output variable. See ml::VariableTypes. Returns:automatically generated

#### create public static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) create([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, int layout, [Mat](http://docs.google.com/org/opencv/core/Mat.html) responses, [Mat](http://docs.google.com/org/opencv/core/Mat.html) varIdx, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sampleIdx) Creates training data from in-memory arrays.Parameters:samples - matrix of samples. It should have CV\_32F type.layout - see ml::SampleTypes.responses - matrix of responses. If the responses are scalar, they should be stored as a single row or as a single column. The matrix should have type CV\_32F or CV\_32S (in the former case the responses are considered as ordered by default; in the latter case - as categorical)varIdx - vector specifying which variables to use for training. It can be an integer vector (CV\_32S) containing 0-based variable indices or byte vector (CV\_8U) containing a mask of active variables.sampleIdx - vector specifying which samples to use for training. It can be an integer vector (CV\_32S) containing 0-based sample indices or byte vector (CV\_8U) containing a mask of training samples. <number\_of\_variables\_in\_responses>`, containing types of each input and output variable. See ml::VariableTypes. Returns:automatically generated

#### create public static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) create([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, int layout, [Mat](http://docs.google.com/org/opencv/core/Mat.html) responses, [Mat](http://docs.google.com/org/opencv/core/Mat.html) varIdx, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sampleIdx, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sampleWeights) Creates training data from in-memory arrays.Parameters:samples - matrix of samples. It should have CV\_32F type.layout - see ml::SampleTypes.responses - matrix of responses. If the responses are scalar, they should be stored as a single row or as a single column. The matrix should have type CV\_32F or CV\_32S (in the former case the responses are considered as ordered by default; in the latter case - as categorical)varIdx - vector specifying which variables to use for training. It can be an integer vector (CV\_32S) containing 0-based variable indices or byte vector (CV\_8U) containing a mask of active variables.sampleIdx - vector specifying which samples to use for training. It can be an integer vector (CV\_32S) containing 0-based sample indices or byte vector (CV\_8U) containing a mask of training samples.sampleWeights - optional vector with weights for each sample. It should have CV\_32F type. <number\_of\_variables\_in\_responses>`, containing types of each input and output variable. See ml::VariableTypes. Returns:automatically generated

#### create public static [TrainData](http://docs.google.com/org/opencv/ml/TrainData.html) create([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, int layout, [Mat](http://docs.google.com/org/opencv/core/Mat.html) responses, [Mat](http://docs.google.com/org/opencv/core/Mat.html) varIdx, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sampleIdx, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sampleWeights, [Mat](http://docs.google.com/org/opencv/core/Mat.html) varType) Creates training data from in-memory arrays.Parameters:samples - matrix of samples. It should have CV\_32F type.layout - see ml::SampleTypes.responses - matrix of responses. If the responses are scalar, they should be stored as a single row or as a single column. The matrix should have type CV\_32F or CV\_32S (in the former case the responses are considered as ordered by default; in the latter case - as categorical)varIdx - vector specifying which variables to use for training. It can be an integer vector (CV\_32S) containing 0-based variable indices or byte vector (CV\_8U) containing a mask of active variables.sampleIdx - vector specifying which samples to use for training. It can be an integer vector (CV\_32S) containing 0-based sample indices or byte vector (CV\_8U) containing a mask of training samples.sampleWeights - optional vector with weights for each sample. It should have CV\_32F type.varType - optional vector of type CV\_8U and size `<number\_of\_variables\_in\_samples> + <number\_of\_variables\_in\_responses>`, containing types of each input and output variable. See ml::VariableTypes. Returns:automatically generated

#### getCatCount public int getCatCount(int vi)

#### getCatMap public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getCatMap()

#### getCatOfs public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getCatOfs()

#### getClassLabels public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getClassLabels() Returns the vector of class labels The function returns vector of unique labels occurred in the responses.Returns:automatically generated

#### getDefaultSubstValues public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getDefaultSubstValues()

#### getLayout public int getLayout()

#### getMissing public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getMissing()

#### getNAllVars public int getNAllVars()

#### getNames public void getNames(java.util.List<java.lang.String> names) Returns vector of symbolic names captured in loadFromCSV()Parameters:names - automatically generated

#### getNativeObjAddr public long getNativeObjAddr()

#### getNormCatResponses public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getNormCatResponses()

#### getNSamples public int getNSamples()

#### getNTestSamples public int getNTestSamples()

#### getNTrainSamples public int getNTrainSamples()

#### getNVars public int getNVars()

#### getResponses public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getResponses()

#### getResponseType public int getResponseType()

#### getSample public void getSample([Mat](http://docs.google.com/org/opencv/core/Mat.html) varIdx, int sidx, float buf)

#### getSamples public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getSamples()

#### getSampleWeights public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getSampleWeights()

#### getSubMatrix public static [Mat](http://docs.google.com/org/opencv/core/Mat.html) getSubMatrix([Mat](http://docs.google.com/org/opencv/core/Mat.html) matrix, [Mat](http://docs.google.com/org/opencv/core/Mat.html) idx, int layout) Extract from matrix rows/cols specified by passed indexes.Parameters:matrix - input matrix (supported types: CV\_32S, CV\_32F, CV\_64F)idx - 1D index vectorlayout - specifies to extract rows (cv::ml::ROW\_SAMPLES) or to extract columns (cv::ml::COL\_SAMPLES) Returns:automatically generated

#### getSubVector public static [Mat](http://docs.google.com/org/opencv/core/Mat.html) getSubVector([Mat](http://docs.google.com/org/opencv/core/Mat.html) vec, [Mat](http://docs.google.com/org/opencv/core/Mat.html) idx) Extract from 1D vector elements specified by passed indexes.Parameters:vec - input vector (supported types: CV\_32S, CV\_32F, CV\_64F)idx - 1D index vector Returns:automatically generated

#### getTestNormCatResponses public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTestNormCatResponses()

#### getTestResponses public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTestResponses()

#### getTestSampleIdx public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTestSampleIdx()

#### getTestSamples public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTestSamples() Returns matrix of test samplesReturns:automatically generated

#### getTestSampleWeights public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTestSampleWeights()

#### getTrainNormCatResponses public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTrainNormCatResponses() Returns the vector of normalized categorical responses The function returns vector of responses. Each response is integer from 0 to `<number of classes>-1`. The actual label value can be retrieved then from the class label vector, see TrainData::getClassLabels.Returns:automatically generated

#### getTrainResponses public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTrainResponses() Returns the vector of responses The function returns ordered or the original categorical responses. Usually it's used in regression algorithms.Returns:automatically generated

#### getTrainSampleIdx public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTrainSampleIdx()

#### getTrainSamples public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTrainSamples() Returns matrix of train samples transposed. See ml::SampleTypes. sampleIdx) the active variables. In current implementation the function tries to avoid physical data copying and returns the matrix stored inside TrainData (unless the transposition or compression is needed).Returns:automatically generated

#### getTrainSamples public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTrainSamples(int layout) Returns matrix of train samplesParameters:layout - The requested layout. If it's different from the initial one, the matrix is transposed. See ml::SampleTypes. sampleIdx) the active variables. In current implementation the function tries to avoid physical data copying and returns the matrix stored inside TrainData (unless the transposition or compression is needed). Returns:automatically generated

#### getTrainSamples public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTrainSamples(int layout, boolean compressSamples) Returns matrix of train samplesParameters:layout - The requested layout. If it's different from the initial one, the matrix is transposed. See ml::SampleTypes.compressSamples - if true, the function returns only the training samples (specified by sampleIdx) the active variables. In current implementation the function tries to avoid physical data copying and returns the matrix stored inside TrainData (unless the transposition or compression is needed). Returns:automatically generated

#### getTrainSamples public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTrainSamples(int layout, boolean compressSamples, boolean compressVars) Returns matrix of train samplesParameters:layout - The requested layout. If it's different from the initial one, the matrix is transposed. See ml::SampleTypes.compressSamples - if true, the function returns only the training samples (specified by sampleIdx)compressVars - if true, the function returns the shorter training samples, containing only the active variables. In current implementation the function tries to avoid physical data copying and returns the matrix stored inside TrainData (unless the transposition or compression is needed). Returns:automatically generated

#### getTrainSampleWeights public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getTrainSampleWeights()

#### getValues public void getValues(int vi, [Mat](http://docs.google.com/org/opencv/core/Mat.html) sidx, float values)

#### getVarIdx public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getVarIdx()

#### getVarSymbolFlags public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getVarSymbolFlags()

#### getVarType public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getVarType()

#### setTrainTestSplit public void setTrainTestSplit(int count) Splits the training data into the training and test parts SEE: TrainData::setTrainTestSplitRatioParameters:count - automatically generated

#### setTrainTestSplit public void setTrainTestSplit(int count, boolean shuffle) Splits the training data into the training and test parts SEE: TrainData::setTrainTestSplitRatioParameters:count - automatically generatedshuffle - automatically generated

#### setTrainTestSplitRatio public void setTrainTestSplitRatio(double ratio) Splits the training data into the training and test parts The function selects a subset of specified relative size and then returns it as the training set. If the function is not called, all the data is used for training. Please, note that for each of TrainData::getTrain\\* there is corresponding TrainData::getTest\\*, so that the test subset can be retrieved and processed as well. SEE: TrainData::setTrainTestSplitParameters:ratio - automatically generated

#### setTrainTestSplitRatio public void setTrainTestSplitRatio(double ratio, boolean shuffle) Splits the training data into the training and test parts The function selects a subset of specified relative size and then returns it as the training set. If the function is not called, all the data is used for training. Please, note that for each of TrainData::getTrain\\* there is corresponding TrainData::getTest\\*, so that the test subset can be retrieved and processed as well. SEE: TrainData::setTrainTestSplitParameters:ratio - automatically generatedshuffle - automatically generated

#### shuffleTrainTest public void shuffleTrainTest()

* [Overview](http://docs.google.com/overview-summary.html)
* [Package](http://docs.google.com/package-summary.html)
* Class
* [Tree](http://docs.google.com/package-tree.html)
* [Index](http://docs.google.com/index-all.html)
* [Help](http://docs.google.com/help-doc.html)
* [Prev Class](http://docs.google.com/org/opencv/ml/SVMSGD.html)
* Next Class
* [Frames](http://docs.google.com/index.html?org/opencv/ml/TrainData.html)
* [No Frames](http://docs.google.com/TrainData.html)
* [All Classes](http://docs.google.com/allclasses-noframe.html)
* Summary:
* Nested |
* Field |
* Constr |
* [Method](#3znysh7)
* Detail:
* Field |
* Constr |
* [Method](#tyjcwt)

Generated on 2021-04-02 03:15:03 / OpenCV 3.4.14