TrainableStatisticalModelFactory

TrainableStatisticalModelFactory.createHomogeneousMarkovModel(alphabets, ess, order): Create a homogeneous Markov model of a given order

TrainableStatisticalModelFactory.createMixtureModel(hyperpars, models): Create a mixture model from differentiable statistical models

TrainableStatisticalModelFactory.createInhomogeneousMarkovModel(alphabets, length, ess, order): Create an inhomogeneous Markov model of a given length and order

TrainableStatisticalModelFactory.createMixtureModel(hyperpars, models): Create a mixture model from trainable statistical models

TrainableStatisticalModelFactory.createPWM(alphabets, length, ess): Create a PWM model of a given length

AbstractClassifier classify: Abstract class of a classifier

new TrainSMBasedClassifier(models): Create a classifier from TrainableStatisticalModels that is learned by ML or MAP

new MSPClassifier(params, prior, models): Create a classifier from DifferentiableStatisticalModels that is learned by MCL or MSP

new GenDisMixClassifier(params, prior, learnPrinc, models): Classifier that learns DifferentiableStatisticalModels using a unified learning principle

classif.train(dataSets): Train a classifier from training data sets

classif.classify(sequence): Classify a sequence

classif.evaluate(performanceMeasures, exc, dataSet): Evaluate performance measures for a given classifier on test data sets

AbstractPerformanceMeasure: Abstract class of all performance measures

new NumericalPerformanceMeasureParameterSet(): Create a set of scalar standard performance measures that are applicable to two-class problems (binary classification)

new PerformanceMeasureParameterSet(measures): Create a set of performance measures

PerformanceMeasureParameterSet.createFilledParameters(): Create a set of scalar standard performance measures for binary classification problems that can immediately be used

toolBox.getMaxIndex(doubles): Get the index of the maximum value in an array

toolBox.sum(doubles): Compute the sum of all elements in an array

toolBox.getMaxIndex(doubles): Get the index of the maximum value in an array

Normalisation.getLogSum(doubles): Compute the logarithm of a sum of values given as their logs

Normalisation.sumNormalisation(doubles): Normalize a given array to probabilities

XMLParser.appendObjectWithTags(buffer, storable, tag): Append storable object to StringBuilder with given tags

XMLParser.extractObjectForTags(buffer, tag): Extract storable object within tags from StringBuffer

Alignment align: Class for optimal alignments of sequences

new Alignment(type, costs): Create an object for alignments of sequences

align.getAlignment(seq1, seq2): Align two sequences

ArrayHandler.clone(array): Deep clone a multi-dimensional array

ArrayHandler.createArrayOf(template, num): Create an array containing num clones of a template

ToolBox.: Compute the sum of all elements in an array

ToolBox.getMaxIndex(doubles): Get the index of the maximum value in an array

AbstractPerformanceMeasure: Abstract class of all performance measures

new NumericalPerformanceMeasureParameterSet(): Create a set of scalar standard performance measures that are applicable to two-class problems (binary classification)

new PerformanceMeasureParameterSet(measures): Create a set of performance measures

PerformanceMeasureParameterSet.createFilledParameters(): Create a set of scalar standard performance measures for binary classification problems that can immediately be used

XMLParser.appendObjectWithTags(buffer, storable, tag): Append storable object to StringBuilder with given tags

XMLParser.extractObjectForTags(buffer, tag): Extract storable object within tags from StringBuffer

Alignment align: Class for optimal alignments of sequences

new Alignment(type, costs): Create an object for alignments of sequences

align.getAlignment(seq1, seq2): Align two sequences

ArrayHandler.clone(array): Deep clone a multi-dimensional array

ArrayHandler.createArrayOf(template, num): Create an array containing num clones of a template

ToolBox.: Compute the sum of all elements in an array

ToolBox.getMaxIndex(doubles): Get the index of the maximum value in an array

Normalisation.getLogSum(doubles): Compute the logarithm of a sum of values given as their logs

Normalisation.sumNormalisation(doubles): Normalize a given array to probabilities

Data handling

Alphabet: A set of symbols

new DiscreteAlphabet(caseInsensitive, alphabet): Create an arbitrary discrete alphabet

new ContinuousAlphabet(min, max): Create a continuous alphabet between min and max

DNAAlphabetContainer.SINGLETON: Singleton instance of a DNA-alphabet

AlphabetContainer: A set of Alphabets and their assignment to positions

new AlphabetContainer(alphabets): Create an aggregate alphabet out of Alphabets

DNAAlphabetContainer.SINGLETON: Singleton instance of aggregate DNA-alphabet

Sequence seq: Representing a biological sequence

Sequence.create(alphabets, string): Create a sequence from a string

seq.getLength(): Obtain the length of a sequence

seq.getContinuousVal(pos): Obtain the continuous value at a position (counting from 0) of a sequence

Sequence: Interface for all statistical models

Data handling

new DNADataSet(filename): Create a data set from sequences

data.getInfixDataSet(start, length): Get a data set containing all infixes of a given length starting at a given position of all sequences in the current data set

data.getNumberOfElements(): Obtain the number of sequences in a data set

data.getElementAt(index): Obtain the sequence at index from a data set

Utilities

Storable: Interface of objects that can be stored to XML

XMLParser.appendObjectWithTags(buffer, storable, tag): Append storable object to StringBuilder with given tags

XMLParser.extractObjectForTags(buffer, tag): Extract storable object within tags from StringBuffer

Alignment align: Class for optimal alignments of sequences

new Alignment(type, costs): Create an object for alignments of sequences

align.getAlignment(seq1, seq2): Align two sequences

ArrayHandler.clone(array): Deep clone a multi-dimensional array

ArrayHandler.createArrayOf(template, num): Create an array containing num clones of a template

ToolBox.: Compute the sum of all elements in an array

ToolBox.getMaxIndex(doubles): Get the index of the maximum value in an array

Normalisation.getLogSum(doubles): Compute the logarithm of a sum of values given as their logs

Normalisation.sumNormalisation(doubles): Normalize a given array to probabilities