Library for Multi-instance Multi-label learning (MIML) API Reference

MIML version 1.4
March 19, 2024

Álvaro Andrés Belmonte Pérez
Amelia Zafra Gómez
Eva Lucrecia Gibaja Galindo

Contents

Chapter 1

Package miml.core

Package Contents *Page*

1.1 Interface IConfiguration

Interface used to indicate that a class can be configured.

1.1.1 Declaration

```
public interface IConfiguration
```

1.1.2 All known subinterfaces

MIMLWel (in 5.4, page 81), MIMLSVM (in 5.3, page 73), MIMLFast (in 5.2, page 64), KiSar (in 5.1, page 57), MIMLBagging (in 6.1, page 89), EvaluatorHoldoutClus (in 8.4, page 128), EvaluatorHoldout (in 8.3, page 124), EvaluatorCV (in 8.2, page 119), MWClassifier (in 10.3, page 145), MIMLClassifier (in 10.2, page 141), MIMLClassifierToMI (in 13.2, page 188), MultiInstanceMultiLabelKNN (in 16.10, page 233), MIMLMAPkNN (in 16.9, page 229), MIMLkNN (in 16.8, page 222), MIMLIBLR (in 16.7, page 219), MIMLFuzzykNN (in 16.5, page 215), MIMLDGC (in 16.3, page 208), MIMLBRkNN (in 16.2, page 204), DMIMLkNN (in 16.1, page 201), MIMLClassifierToML (in 18.1, page 241), MIMLReport (in 23.3, page 309), BaseMIMLReport (in 23.2, page 306), MIMLRBF (in 24.3, page 329), MIMLNN (in 24.2, page 322), EnMIMLNNmetric (in 24.1, page 315)

1.1.3 All classes known to implement interface

EvaluatorHoldout (in 8.3, page 124), EvaluatorCV (in 8.2, page 119), MIMLClassifier (in 10.2, page 141), MIMLReport (in 23.3, page 309)

1.1.4 Method summary

configure(Configuration) Method to configure the class with the given configuration.

1.1.5 Methods

- **configure**

  ```
  void configure(org.apache.commons.configuration2.Configuration
      configuration)
  ```

  - **Description**
    Method to configure the class with the given configuration.
  - **Parameters**
    * `configuration` – Configuration used to configure the class.

1.2 Class ConfigLoader

Class used to read a xml file and configure an experiment.

1.2.1 Declaration

```
public class ConfigLoader
 extends java.lang.Object
```

1.2.2 Field summary

configuration Configuration object.

1.2.3 Constructor summary

ConfigLoader(String) Constructor that sets the configuration file

1.2.4 Method summary

getConfiguration() Gets the experiment's configuration.
loadClassifier() Read current configuration to load and configure the classifier.
loadEvaluator() Read current configuration to load and configure the evaluator.
loadReport() Read current configuration to load and configure the report.
setConfiguration(Configuration) Sets the configuration for the experiment.

1.2.5 Fields

- `protected org.apache.commons.configuration2.Configuration` **configuration**
  - Configuration object.

1.2.6 Constructors

- **ConfigLoader**

```
public ConfigLoader(java.lang.String path) throws org.apache.
    commons.configuration2.ex.ConfigurationException
```

  - **Description**
    Constructor that sets the configuration file
  - **Parameters**
    - `path` – The path of config file.
  - **Throws**
    - `org.apache.commons.configuration2.ex.ConfigurationException` – if occurred an error during the loading of the configuration.

1.2.7 Methods

- **getConfiguration**

```
public org.apache.commons.configuration2.Configuration
    getConfiguration()
```

  - **Description**
    Gets the experiment's configuration.
  - **Returns** – The configuration used during experimentation.

- **loadClassifier**

```
public miml.classifiers.miml.IMIMLClassifier loadClassifier()
    throws java.lang.Exception
```

  - **Description**
    Read current configuration to load and configure the classifier.
  - **Returns** – A MIMLClassifier.
  - **Throws**
    * `java.lang.Exception` – if the classifier couldn't be loaded correctly.

- **loadEvaluator**

```
public miml.evaluation.IEvaluator loadEvaluator() throws java.
    lang.Exception
```

  - **Description**
    Read current configuration to load and configure the evaluator.
  - **Returns** – A evaluator for MIML Classifiers.
  - **Throws**
    * `java.lang.Exception` – if the class loaded can't be loaded.

- **loadReport**

```
public miml.report.IReport loadReport() throws java.lang.
    Exception
```

  - **Description**
    Read current configuration to load and configure the report.
  - **Returns** – the MIML report
  - **Throws**
    * `java.lang.Exception` – if the class can't be loaded.

- **setConfiguration**

```
public void setConfiguration(org.apache.commons.configuration2.
    Configuration configuration)
```

  - **Description**
    Sets the configuration for the experiment.
  - **Parameters**
    * `configuration` – A new configuration.

1.3 Class ConfigParameters

Class used to save configuration parameters to be used in reports.

1.3.1 Declaration

```
public final class ConfigParameters
 extends java.lang.Object
```

1.3.2 Field summary

algorithmName The algorithm used in the experimentation.
classifierName The classifier used in the experimentation.
configFileName The configuration filename used in the experimentation.
dataFileName The name of data file used in the experimentation.
isTransformation If the classifier configured in the experiment uses a method transformation.
transformationMethod The name of the method used in the experiment if this is a transformation method.

1.3.3 Constructor summary

ConfigParameters()

1.3.4 Method summary

getAlgorithmName() Gets the algorithm name.
getClassifierName() Gets the classifier name.
getConfigFileName() Gets the configuration file name.
getDataFileName() Gets the name of data file.
getIsTransformation() Gets if the method used is transformation.
getTransformationMethod() Gets the transformation method used in the experiment.
setAlgorithmName(String) Sets the algorithm name.
setClassifierName(String) Sets the classifier name.
setConfigFileName(String) Sets the configuration file name.
setDataFileName(String) Sets the data file name.
setIsTransformation(Boolean) Sets if the method used is transformation.
setTransformationMethod(String) Sets the transformation method used in the experiment.

1.3.5 Fields

- `protected static java.lang.String` **algorithmName**
  - The algorithm used in the experimentation.
- `protected static java.lang.String` **configFileName**
  - The configuration filename used in the experimentation.
- `protected static java.lang.String` **dataFileName**
  - The name of data file used in the experimentation.

- `protected static java.lang.String` **classifierName**
  - The classifier used in the experimentation.
- `protected static java.lang.String` **transformationMethod**
  - The name of the method used in the experiment if this is a transformation method.
- `protected static java.lang.Boolean` **isTransformation**
  - If the classifier configured in the experiment uses a method transformation.

1.3.6 Constructors

- **ConfigParameters**

  ```
  public ConfigParameters()
  ```

1.3.7 Methods

- **getAlgorithmName**

  ```
  public static java.lang.String getAlgorithmName()
  ```

  - **Description**
    Gets the algorithm name.
  - **Returns** – The algorithm name.
- **getClassifierName**

  ```
  public static java.lang.String getClassifierName()
  ```

  - **Description**
    Gets the classifier name.
  - **Returns** – The classifier name.
- **getConfigFileName**

  ```
  public static java.lang.String getConfigFileName()
  ```

  - **Description**
    Gets the configuration file name.
  - **Returns** – The configuration file name.
- **getDataFileName**

```
public static java.lang.String getDataFileName()
```

- **Description**
  Gets the name of data file.
- **Returns** – The name of data file.

- **getIsTransformation**

```
public static java.lang.Boolean getIsTransformation()
```

  - **Description**
    Gets if the method used is transformation.
  - **Returns** – True if the method used is transformation.

- **getTransformationMethod**

```
public static java.lang.String getTransformationMethod()
```

  - **Description**
    Gets the transformation method used in the experiment.
  - **Returns** – The transformation method used in the experiment.

- **setAlgorithmName**

```
public static void setAlgorithmName(java.lang.String
    algorithmName)
```

  - **Description**
    Sets the algorithm name.
  - **Parameters**
    * `algorithmName` – The new algorithm name.

- **setClassifierName**

```
public static void setClassifierName(java.lang.String
    classifierName)
```

  - **Description**
    Sets the classifier name.
  - **Parameters**
    * `classifierName` – The classifier name.

- **setConfigFileName**

  ```
  public static void setConfigFileName(java.lang.String
      configFileName)
  ```

  - **Description**
    Sets the configuration file name.
  - **Parameters**
    * `configFileName` – The new configuration file name.

- **setDataFileName**

  ```
  public static void setDataFileName(java.lang.String dataFileName
      )
  ```

  - **Description**
    Sets the data file name.
  - **Parameters**
    * `dataFileName` – the new data file name

- **setIsTransformation**

  ```
  public static void setIsTransformation(java.lang.Boolean
      isTransformation)
  ```

  - **Description**
    Sets if the method used is transformation.
  - **Parameters**
    * `isTransformation` – If the method used is transformation.

- **setTransformationMethod**

  ```
  public static void setTransformationMethod(java.lang.String
      transformationMethod)
  ```

  - **Description**
    Sets the transformation method used in the experiment.
  - **Parameters**
    * `transformationMethod` – The transformation method used in the experiment.

1.4 Class Params

This class contains the list of classes and objects needed to create a new instance of a Multi Label classifier through a specific constructor.

1.4.1 Declaration

```
public class Params
 extends java.lang.Object
```

1.4.2 Field summary

classes List of classes needed by the Multi Label classifier's constructor.
objects List of the values for the classes array

1.4.3 Constructor summary

Params(Class[], Object[]) Generic constructor

1.4.4 Method summary

getClasses()
getObjects()
setClasses(Class[])
setObjects(Object[])

1.4.5 Fields

- `private java.lang.Class[]` **classes**
  - List of classes needed by the Multi Label classifier's constructor.
- `private java.lang.Object[]` **objects**
  - List of the values for the classes array

1.4.6 Constructors

- **Params**

  ```
  public Params(java.lang.Class[] classes ,java.lang.Object[]
      objects)
  ```

  - **Description**
    Generic constructor
  - **Parameters**
    * `classes` – The list of classes needed by the Multi Label classifier's constructor.
    * `objects` – The list of the values for the classes array.

1.4.7 Methods

- **getClasses**

  ```
  public java.lang.Class[] getClasses()
  ```

  - **Returns** – the classes

- **getObjects**

  ```
  public java.lang.Object[] getObjects()
  ```

  - **Returns** – the objects

- **setClasses**

  ```
  public void setClasses(java.lang.Class[] classes)
  ```

  - **Parameters**
    - `classes` – the classes to set

- **setObjects**

  ```
  public void setObjects(java.lang.Object[] objects)
  ```

  - **Parameters**
    - `objects` – the objects to set

1.5 Class Utils

This class has utilies that can be used anywhere in the library.

1.5.1 Declaration

```
public final class Utils
 extends java.lang.Object
```

1.5.2 Constructor summary

Utils()

1.5.3 Method summary

readMultiLabelLearnerParams(Configuration) Read the configuration parameters for a specific Multi Label classifier's constructor
resample(Instances, double, boolean, int) Obtains a sample of the original data.

1.5.4 Constructors

- **Utils**

```
public Utils()
```

1.5.5 Methods

- **readMultiLabelLearnerParams**

```
public static Params readMultiLabelLearnerParams(org.apache.
    commons.configuration2.Configuration configuration)
```

  - **Description**
    Read the configuration parameters for a specific Multi Label classifier's constructor
  - **Parameters**
    * `configuration` – Configuration used to configure the class
  - **Returns** – Params class which contains the parameters of classifier's constructor

- **resample**

```
public static weka.core.Instances resample(weka.core.Instances
    data,double percentage,boolean sampleWithReplacement,int seed
    ) throws java.lang.Exception
```

  - **Description**
    Obtains a sample of the original data.
  - **Parameters**
    * `data` – Instances with the dataset.
    * `percentage` – percentage of instances that will contain the new dataset.
    * `sampleWithReplacement` – If true the sampling will be with replacement.
    * `seed` – Seed for randomization. Necessary if instances have not been previously shuffled with randomize.
  - **Returns** – Instances.
  - **Throws**
    * `java.lang.Exception` – To be handled.

Chapter 2

Package miml.data.partitioning.random

2.1 Class RandomCrossValidation

Class to split a multi-label dataset into N multi-label random datasets for cross-validation. MIML and MVML formats are also supported. Due to this fact, applied over datasets with a high number of labels (e.g. some subsets of miml protein data), this method may generate folds with uneven number of instances and with some duplicated instances. In these cases, using a lower number of folds (eg. 3 folds) or another kind of partitioning (eg. iteratrive or powerset) is recommended. Besides, the same instance could be included twice to guarantee instances of all labels in the resulting train set.

2.1.1 Declaration

```
public class RandomCrossValidation
 extends miml.data.partitioning.CrossValidationBase
```

2.1.2 Field summary

indexes A matrix of nFoldsx2 representing the index of the first and last instance of each partition

2.1.3 Constructor summary

RandomCrossValidation(int, MultiLabelInstances) Constructor.
RandomCrossValidation(MultiLabelInstances) Default constructor.

2.1.4 Method summary

getFolds(int)

2.1.5 Fields

- `protected int[][]` **indexes**
  - A matrix of nFoldsx2 representing the index of the first and last instance of each partition

2.1.6 Constructors

- **RandomCrossValidation**

```
public RandomCrossValidation(int seed,mulan.data.
    MultiLabelInstances mlDataSet) throws mulan.data.
    InvalidDataFormatException
```

  - **Description**
    Constructor.
  - **Parameters**
    * `seed` – Seed for randomization
    * `mlDataSet` – A multi-label dataset
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – To be handled.

- **RandomCrossValidation**

```
public RandomCrossValidation(mulan.data.MultiLabelInstances
    mlDataSet) throws mulan.data.InvalidDataFormatException
```

  - **Description**
    Default constructor.
  - **Parameters**
    * `mlDataSet` – A multi-label dataset
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – To be handled.

2.1.7 Methods

- **getFolds**

```
public abstract mulan.data.MultiLabelInstances[] getFolds(int
    nFolds) throws mulan.data.InvalidDataFormatException
```

  - **Description copied from miml.data.partitioning.CrossValidationBase** (**in 21.1, page 284**)

    Splits a dataset into nfolds partitions.
  - **Parameters**
    - `nFolds` – Number of folds.
  - **Returns** – MultiLabelInstances[] a vector of nFolds. Each element represents a fold.
  - **Throws**
    - `mulan.data.InvalidDataFormatException` – To be handled.

2.1.8 Members inherited from class CrossValidationBase

`miml.data.partitioning.CrossValidationBase` (in 21.1, page 284)

- `public static MultiLabelInstances` **foldsToRounds**`(mulan.data.MultiLabelInstances[]` **Folds**`) throws java.lang.Exception`
- `public abstract MultiLabelInstances` **getFolds**`(int` **nFolds**`) throws mulan.data.InvalidDataFormatException`
- `public MultiLabelInstances` **getRounds**`(int` **nFolds**`) throws java.lang.Exception`
- `protected void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`

2.1.9 Members inherited from class PartitionerBase

`miml.data.partitioning.PartitionerBase` (in 21.2, page 287)

- `protected` **seed**
- `protected abstract void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`
- `public int` **totalExamples**`()`
- `protected` **workingSet**

2.2 Class RandomTrainTest

Class to split a multi-label dataset into two multi-label random datasets corresponding to the train and test datasets respectively. MIML and MVML formats are also supported. This class guarantees at least one instance for label in train dataset.

2.2.1 Declaration

```
public class RandomTrainTest
 extends miml.data.partitioning.TrainTestBase
```

2.2.2 Constructor summary

RandomTrainTest(int, MultiLabelInstances) Constructor.
RandomTrainTest(MultiLabelInstances) Default constructor.

2.2.3 Method summary

split(double)

2.2.4 Constructors

- **RandomTrainTest**

```
public RandomTrainTest(int seed,mulan.data.MultiLabelInstances
    mlDataSet) throws mulan.data.InvalidDataFormatException
```

  - **Description**
    Constructor.
  - **Parameters**
    * `seed` – Seed for randomization
    * `mlDataSet` – A multi-label dataset
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – To be handled

- **RandomTrainTest**

```
public RandomTrainTest(mulan.data.MultiLabelInstances mlDataSet)
     throws mulan.data.InvalidDataFormatException
```

  - **Description**
    Default constructor.
  - **Parameters**
    * `mlDataSet` – A multi-label dataset
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – To be handled

2.2.5 Methods

- **split**

```
public abstract mulan.data.MultiLabelInstances[] split(double
    percentageTrain) throws java.lang.Exception
```

- **Description copied from miml.data.partitioning.TrainTestBase (in 21.3, page 289)**

  Returns a array with two multi-label random datasets corresponding to the train and test sets respectively.
- **Parameters**
  * `percentageTrain` – Percentage of train dataset, a value in [0, 100].
- **Returns** – MultiLabelInstances[].
  MultiLabelInstances[0] is the train set.
  MultiLabelInstances[1] is the test set.
- **Throws**
  * `java.lang.Exception` – To be handled.

2.2.6 Members inherited from class TrainTestBase

`miml.data.partitioning.TrainTestBase` (in 21.3, page 289)

- `public abstract MultiLabelInstances` **split**`(double` **percentageTrain**`) throws java.lang.Exception`
- `protected void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`

2.2.7 Members inherited from class PartitionerBase

`miml.data.partitioning.PartitionerBase` (in 21.2, page 287)

- `protected` **seed**
- `protected abstract void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`
- `public int` **totalExamples**`()`
- `protected` **workingSet**

Chapter 3

Package miml.clusterers

Package Contents *Page*

3.1 Class KMedoids

Class implementing the PAM (Partitioning Around Medoids) approximation [1] to kMedoids for multi-instance data. [1] *Kaufman, L. and Rousseeuw, P.J. (1990). Partitioning Around Medoids (Program PAM). In Finding Groups in Data (eds L. Kaufman and P.J. Rousseeuw). https://doi.org/10.1002/9780470316801.ch2*

3.1.1 Declaration

```
public class KMedoids
 extends weka.clusterers.RandomizableClusterer implements weka.
    clusterers.Clusterer
```

3.1.2 Field summary

clusterAssignment The assignment of instances to medoids.
configurationCost Final cost of the clustering configuration.
distancesMatrix Distance between instances.
maxIterations The maximum number of iterations the algorithm is allowed to run.
medoidIndices The medoid indices.
medoidInstances The medoid instances.
metric Distance function.
minimize Whether the metric is maximized o minimized.
numClusters Number of clusters to generate.
numInstances Number of instances in the dataset.

numIterations Final number of iterations to perform clustering.

randomInitialization Whether the initialization of medoids is random o applying the BUILD method of PAM algorithm.

serialVersionUID For serialization.

3.1.3 Constructor summary

KMedoids() Creates a new instance of the k-medoids algorithm with default parameters.

KMedoids(IDistance) Creates a new instance of the k-medoids algorithm with the specified dist measure.

KMedoids(int) Creates a new instance of the k-medoids algorithm with with the specified parameters.

KMedoids(int, IDistance) Creates a new instance of the k-medoids algorithm with the specified parameters.

KMedoids(int, int, IDistance) Creates a new instance of the k-medoids algorithm with the specified parameters.

3.1.4 Method summary

assignInstancesToMedoids(int[]) Assign all instances from the data set to the medoids.

buildClusterer(Instances)

buildInitialization() Performs an initialization of medoids based on the BUILD step of PAM algorithm.

clusterInstance(Instance)

compare(double, double) Allows to maximize or minimize the metric according to the value of minimize property.

computeCost(int[]) Computes the cost of a configuration.

computeDistances(Instances) Computes distances between instances.

distanceToMedoids(Instance) Returns the distance of an instance to each medoid.

distanceToMedoids(int) Returns the distance of an instance in the training dataset referenced by its index to each medoid.

distributionForInstance(Instance)

getAssignment() Gets the assignment of instances to clusters.

getCapabilities()

getConfigurationCost() Gets final the cost of the configuration after applying clustering.

getDistanceFunction() Gets the distance function used by clusterer.

getDistances() Returns a matrix the distances between all instances being distances[i][j] the distance between the instances with indices i and j.

getMaxIterations() Gets the maximum number of iterations used by clusterer.

getMedoidInstances() Gets the medoids obtained after performing clustering.

getNumIterations() Gets the number of iterations performed in the clustering process.

getRandomInitialization() Gets whether a random initialization of medoids or a initialization based on the BUILD step of PAM is considered for clustering.
isMedoid(int) Determines if an instance is being considered as medoid.
medoidIndex(int) Determines if an instance is being considered as medoid.
numberOfClusters()
randomInitialization() Performs a random initialization of medoids.
setDistanceFunction(IDistance) Sets the distance function to use for clustering.
setMaxIterations(int) Sets the maximum number of iterations for clustering.
setNumClusters(int) Sets the number of clusters to perform clustering.
setRandomInitialization(boolean) Sets whether a random initialization of medoids or a initialization based on the BUILD step of PAM is considered for clustering.

3.1.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `protected miml.core.distance.IDistance` **metric**
  - Distance function. By default MaximalHausdorff distance is used
- `protected int` **numClusters**
  - Number of clusters to generate. By default 10 clusters.
- `protected int` **numInstances**
  - Number of instances in the dataset.
- `protected int` **maxIterations**
  - The maximum number of iterations the algorithm is allowed to run. By default 100 iterations
- `protected int[]` **medoidIndices**
  - The medoid indices. Element k contains the index of the instance being the k medoid, a value in (0, numInstances).
- `protected weka.core.Instance[]` **medoidInstances**
  - The medoid instances. Element k contains the instance being the k medoid
- `protected int[]` **clusterAssignment**
  - The assignment of instances to medoids. Element i contains the number of medoid assigned to instance i, a value in (0, nClusters-1).
- `protected double[][]` **distancesMatrix**
  - Distance between instances.
- `protected boolean` **minimize**
  - Whether the metric is maximized o minimized. By default the metric is minimized.

- `protected boolean` **randomInitialization**
  - Whether the initialization of medoids is random o applying the BUILD method of PAM algorithm. By default random initialization is performed.
- `protected double` **configurationCost**
  - Final cost of the clustering configuration.
- `protected double` **numIterations**
  - Final number of iterations to perform clustering.

3.1.6 Constructors

- **KMedoids**

```
public KMedoids() throws java.lang.Exception
```

  - **Description**
    Creates a new instance of the k-medoids algorithm with default parameters.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **KMedoids**

```
public KMedoids(miml.core.distance.IDistance metric) throws java
    .lang.Exception
```

  - **Description**
    Creates a new instance of the k-medoids algorithm with the specified dist measure.
  - **Parameters**
    - `metric` – The distance metric to use for measuring the distance between instances.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **KMedoids**

```
public KMedoids(int numClusters) throws java.lang.Exception
```

  - **Description**
    Creates a new instance of the k-medoids algorithm with with the specified parameters.
  - **Parameters**
    - `numClusters` – The number of clusters.

  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **KMedoids**

```
public KMedoids(int numClusters,miml.core.distance.IDistance
    metric) throws java.lang.Exception
```

  - **Description**
    Creates a new instance of the k-medoids algorithm with the specified parameters.
  - **Parameters**
    - `numClusters` – The number of clusters to generate.
    - `metric` – The distance metric to use for measuring the distance between instances.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **KMedoids**

```
public KMedoids(int numClusters,int maxIterations,miml.core.
    distance.IDistance metric) throws java.lang.Exception
```

  - **Description**
    Creates a new instance of the k-medoids algorithm with the specified parameters.
  - **Parameters**
    - `numClusters` – The number of clusters to generate.
    - `maxIterations` – The maximum number of iteration the algorithm is allowed to run.
    - `metric` – The distance metric to use for measuring the distance between instances.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

3.1.7 Methods

- **assignInstancesToMedoids**

```
protected int[] assignInstancesToMedoids(int[] medoidIndices)
```

  - **Description**
    Assign all instances from the data set to the medoids.
  - **Parameters**

    * `medoidIndices` – Candidate medoids.
  - **Returns** – An array with the best cluster number for each instance in the data set.

- **buildClusterer**

```
void buildClusterer(weka.core.Instances arg0) throws java.lang.
    Exception
```

- **buildInitialization**

```
protected void buildInitialization()
```

  - **Description**
    Performs an initialization of medoids based on the BUILD step of PAM algorithm.

- **clusterInstance**

```
int clusterInstance(weka.core.Instance arg0) throws java.lang.
    Exception
```

- **compare**

```
protected boolean compare(double metricValue1,double
    metricValue2)
```

  - **Description**
    Allows to maximize or minimize the metric according to the value of minimize property.
  - **Parameters**
    * `metricValue1` – A metric value.
    * `metricValue2` – Another metric value.
  - **Returns** – If minimize==true it returns metricValue1<=metricValue2 other case it returns metricValue1>=metricValue2.

- **computeCost**

```
protected double computeCost(int[] assignment)
```

  - **Description**
    Computes the cost of a configuration.
  - **Parameters**

    * `assignment` – Array containing in element i the index of the medoid assigned to instance i.
  - **Returns** – The sum of the distances to medoids of all instances.

- **computeDistances**

```
protected void computeDistances(weka.core.Instances data) throws
    java.lang.Exception
```

  - **Description**
    Computes distances between instances.
  - **Parameters**
    * `data` – The dataset.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **distanceToMedoids**

```
public double[] distanceToMedoids(weka.core.Instance instance)
    throws java.lang.Exception
```

  - **Description**
    Returns the distance of an instance to each medoid.
  - **Parameters**
    * `instance` – An instance. It can be either an instance of the dataset or a new instance.
  - **Returns** – The distance of the instance to each medoid.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **distanceToMedoids**

```
public double[] distanceToMedoids(int index) throws java.lang.
    Exception
```

  - **Description**
    Returns the distance of an instance in the training dataset referenced by its index to each medoid.
  - **Parameters**
    * `index` – It must be a valid instance index in the dataset used for clustering.
  - **Returns** – The distance of the instance to each medoid.

- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

- **distributionForInstance**

```
double[] distributionForInstance(weka.core.Instance arg0) throws
    java.lang.Exception
```

- **getAssignment**

```
public int[] getAssignment()
```

- **Description**
  Gets the assignment of instances to clusters. This method must be called after clustering.
- **Returns** – An array. Element i contains a value in (0, numCusters-1), the cluster number assigned to instance i.

- **getCapabilities**

```
weka.core.Capabilities getCapabilities()
```

- **getConfigurationCost**

```
public double getConfigurationCost()
```

- **Description**
  Gets final the cost of the configuration after applying clustering. This method must be called after clustering.
- **Returns** – The final cost of the clustering.

- **getDistanceFunction**

```
public miml.core.distance.IDistance getDistanceFunction()
```

- **Description**
  Gets the distance function used by clusterer.
- **Returns** – The distance function used by clusterer.

- **getDistances**

```
public double[][] getDistances()
```

  - **Description**
    Returns a matrix the distances between all instances being distances[i][j] the distance between the instances with indices i and j.
  - **Returns** – double[][]

- **getMaxIterations**

  ```
  public int getMaxIterations()
  ```

  - **Description**
    Gets the maximum number of iterations used by clusterer.
  - **Returns** – The maximum number of iterations.

- **getMedoidInstances**

  ```
  public weka.core.Instance[] getMedoidInstances()
  ```

  - **Description**
    Gets the medoids obtained after performing clustering.
  - **Returns** – An array of instances corresponding to medoids.

- **getNumIterations**

  ```
  public double getNumIterations()
  ```

  - **Description**
    Gets the number of iterations performed in the clustering process. This method must be called after clustering.
  - **Returns** – The number of iterations performed.

- **getRandomInitialization**

  ```
  public boolean getRandomInitialization()
  ```

  - **Description**
    Gets whether a random initialization of medoids or a initialization based on the BUILD step of PAM is considered for clustering.
  - **Returns** – A true value if a random initialization of medoids is performed and false if the initialization is based on the build step of PAM selecting as medoids, the instances that minimizes the sum of distances to the rest.

- **isMedoid**

```
protected boolean isMedoid(int instanceIndex)
```

- **Description**
  Determines if an instance is being considered as medoid.
- **Parameters**
  * `instanceIndex` – The index of the instance.
- **Returns** – A true value if the instance is being considered as medoid.

• **medoidIndex**

```
protected int medoidIndex(int instanceIndex)
```

- **Description**
  Determines if an instance is being considered as medoid. If true, the index of the medoid is returned, a value in (0, nClusters-1)
- **Parameters**
  * `instanceIndex` – The index of the instance.
- **Returns** – A true value if the instance is being considered as medoid.

• **numberOfClusters**

```
int numberOfClusters() throws java.lang.Exception
```

• **randomInitialization**

```
protected void randomInitialization()
```

- **Description**
  Performs a random initialization of medoids.

• **setDistanceFunction**

```
public void setDistanceFunction(miml.core.distance.IDistance
    distanceFunction)
```

- **Description**
  Sets the distance function to use for clustering. This method must be called before clustering.
- **Parameters**
  * `distanceFunction` – The distance function used for clustering.

- **setMaxIterations**

  ```
  public void setMaxIterations(int maxIterations)
  ```

  - **Description**
    Sets the maximum number of iterations for clustering. This method must be called before clustering.
  - **Parameters**
    * `maxIterations` – The maximum number of iterations for clustering.

- **setNumClusters**

  ```
  public void setNumClusters(int numClusters)
  ```

  - **Description**
    Sets the number of clusters to perform clustering. This method must be called before clustering.
  - **Parameters**
    * `numClusters` – A number of clusters.

- **setRandomInitialization**

  ```
  public void setRandomInitialization(boolean randomInitialization
    )
  ```

  - **Description**
    Sets whether a random initialization of medoids or a initialization based on the BUILD step of PAM is considered for clustering. This method must be called before clustering.
  - **Parameters**
    * `randomInitialization` – If true a random initialization of medoids is performed. Otherwise the initialization is based on the build step of PAM selecting as medoids, the instances that minimizes the sum of distances to the rest.

3.1.8 Members inherited from class RandomizableClusterer

`weka.clusterers.RandomizableClusterer`

- `public String` **getOptions**`()`
- `public int` **getSeed**`()`
- `public Enumeration` **listOptions**`()`
- `protected` **m_Seed**
- `protected` **m_SeedDefault**
- `public String` **seedTipText**`()`
- `private static final` **serialVersionUID**
- `public void` **setOptions**`(java.lang.String[]` **arg0**`) throws java.lang.Exception`
- `public void` **setSeed**`(int` **arg0**`)`

3.1.9 Members inherited from class AbstractClusterer

`weka.clusterers.AbstractClusterer`

- `public abstract void` **buildClusterer**`(weka.core.Instances` **arg0**`) throws java.lang.Exception`
- `public int` **clusterInstance**`(weka.core.Instance` **arg0**`) throws java.lang.Exception`
- `public double` **distributionForInstance**`(weka.core.Instance` **arg0**`) throws java.lang.Exception`
- `public static Clusterer` **forName**`(java.lang.String` **arg0**`, java.lang.String[]` **arg1**`) throws java.lang.Exception`
- `public Capabilities` **getCapabilities**`()`
- `public String` **getRevision**`()`
- `public static Clusterer` **makeCopies**`(Clusterer` **arg0**`, int` **arg1**`) throws java.lang.Exception`
- `public static Clusterer` **makeCopy**`(Clusterer` **arg0**`) throws java.lang.Exception`
- `public abstract int` **numberOfClusters**`() throws java.lang.Exception`
- `public static void` **runClusterer**`(Clusterer` **arg0**`, java.lang.String[]` **arg1**`)`
- `private static final` **serialVersionUID**

Chapter 4

Package miml.core.distance

Package Contents *Page*

4.1 Interface IDistance

Interface to implement the metrics used to measure the distance between `MIMLBag` (in 7.1, page 96) of a data sets.

4.1.1 Declaration

```
public interface IDistance
 extends java.io.Serializable
```

4.1.2 All known subinterfaces

MinimalHausdorff (in 4.5, page 55), MaximalHausdorff (in 4.4, page 54), HausdorffDistance (in 4.3, page 51), AverageHausdorff (in 4.2, page 49)

4.1.3 All classes known to implement interface

HausdorffDistance (in 4.3, page 51)

4.1.4 Method summary

distance(Instance, Instance) Get the distance between two bags in the form of a set of `Instance` with relational attribute.
distance(Instances, Instances) Get the distance between two bags in the form of a set of `Instances` .
distance(MIMLBag, MIMLBag) Get the distance between two `MIMLBag` (in 7.1, page 96).
setInstances(Instances) Sets the Intances in the form of a set of Instances with relational attribute.
setInstances(MIMLInstances) Sets the Intances in the form of MIMLBags.
update(Instance) Update the distance function (if necessary) for the newly added instance in the form of Instance with relational attribute.
update(MIMLBag) Update the distance function (if necessary) for the newly added instance in the form of MIMLBag.

4.1.5 Methods

- **distance**

```
double distance(weka.core.Instance first ,weka.core.Instance
    second) throws java.lang.Exception
```

  - **Description**
    Get the distance between two bags in the form of a set of `Instance` with relational attribute.
  - **Parameters**
    * `first` – First bag as Instance with relational attribute.
    * `second` – Second Bag as Instance with relational attribute.
  - **Returns** – Distance between two bags.
  - **Throws**
    * `java.lang.Exception` – if occurred an error during distance calculation.

- **distance**

```
double distance(weka.core.Instances first ,weka.core.Instances
    second) throws java.lang.Exception
```

- **Description**

  Get the distance between two bags in the form of a set of `Instances` .
- **Parameters**
  - `first` – First bag as instances.
  - `second` – Second Bag as Instances.
- **Returns** – Distance between two bags.
- **Throws**
  - `java.lang.Exception` – if occurred an error during distance calculation.

- **distance**

```
double distance(miml.data.MIMLBag first ,miml.data.MIMLBag second
    ) throws java.lang.Exception
```

- **Description**

  Get the distance between two `MIMLBag` (in 7.1, page 96).
- **Parameters**
  - `first` – First bag.
  - `second` – Second bag.
- **Returns** – Distance between two bags.
- **Throws**
  - `java.lang.Exception` – if occurred an error during distance calculation,

- **setInstances**

```
void setInstances(weka.core.Instances bags) throws java.lang.
    Exception
```

- **Description**

  Sets the Intances in the form of a set of Instances with relational attribute.
- **Parameters**
  - `bags` – The instances to be set.
- **Throws**
  - `java.lang.Exception` – to be handled in upper level.

- **setInstances**

```
void setInstances(miml.data.MIMLInstances bags) throws java.lang
    .Exception
```

  - **Description**
    Sets the Intances in the form of MIMLBags.
  - **Parameters**
    * `bags` – The instances to be set.
  - **Throws**
    * `java.lang.Exception` – to be handled in upper level.

- **update**

```
void update(weka.core.Instance bag) throws java.lang.Exception
```

  - **Description**
    Update the distance function (if necessary) for the newly added instance in the form of Instance with relational attribute.
  - **Parameters**
    * `bag` – The bag.
  - **Throws**
    * `java.lang.Exception` – to be handled in upper level.

- **update**

```
void update(miml.data.MIMLBag bag) throws java.lang.Exception
```

  - **Description**
    Update the distance function (if necessary) for the newly added instance in the form of MIMLBag.
  - **Parameters**
    * `bag` – The bag.
  - **Throws**
    * `java.lang.Exception` – to be handled in upper level.

4.2 Class AverageHausdorff

Class that implements Average Hausdorff metric to measure the distance between 2 bags of a data set.

4.2.1 Declaration

```
public class AverageHausdorff
 extends miml.core.distance.HausdorffDistance
```

4.2.2 Field summary

serialVersionUID Generated Serial version UID.

4.2.3 Constructor summary

AverageHausdorff()
AverageHausdorff(MIMLInstances)

4.2.4 Method summary

distance(Instances, Instances)

4.2.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.

4.2.6 Constructors

- **AverageHausdorff**

```
public AverageHausdorff()
```

- **AverageHausdorff**

```
public AverageHausdorff(miml.data.MIMLInstances bags) throws
    java.lang.Exception
```

4.2.7 Methods

- **distance**

```
public double distance(weka.core.Instances first ,weka.core.
    Instances second) throws java.lang.Exception
```

4.2.8 Members inherited from class HausdorffDistance

`miml.core.distance.HausdorffDistance` (in 4.3, page 51)

- **dataSet**
- **dfun**
- `public double` **distance**`(weka.core.Instance` **bag1**`, weka.core.Instance` **bag2**`) throws java.lang.Exception`
- `public double` **distance**`(miml.data.MIMLBag` **first**`, miml.data.MIMLBag` **second**`) throws java.lang.Exception`
- `public boolean` **hasInstances**`()`
- `private static final` **serialVersionUID**
- `public void` **setInstances**`(weka.core.Instances` **bags**`) throws java.lang.Exception`
- `public void` **setInstances**`(miml.data.MIMLInstances` **bags**`) throws java.lang.Exception`
- `public void` **update**`(weka.core.Instance` **bag**`) throws java.lang.Exception`
- `public void` **update**`(miml.data.MIMLBag` **bag**`) throws java.lang.Exception`

4.3 Class HausdorffDistance

4.3.1 Declaration

```
public abstract class HausdorffDistance
 extends java.lang.Object implements IDistance
```

4.3.2 All known subclasses

MinimalHausdorff (in 4.5, page 55), MaximalHausdorff (in 4.4, page 54), AverageHausdorff (in 4.2, page 49)

4.3.3 Field summary

dataSet
dfun
serialVersionUID

4.3.4 Constructor summary

HausdorffDistance()
HausdorffDistance(MIMLInstances)

4.3.5 Method summary

distance(Instance, Instance)
distance(MIMLBag, MIMLBag)
hasInstances()
setInstances(Instances)
setInstances(MIMLInstances)
update(Instance)
update(MIMLBag)

4.3.6 Fields

- `private static final long` **serialVersionUID**
- `weka.core.DistanceFunction` **dfun**
- `weka.core.Instances` **dataSet**

4.3.7 Constructors

- **HausdorffDistance**

```
public HausdorffDistance()
```

- **HausdorffDistance**

```
public HausdorffDistance(miml.data.MIMLInstances bags) throws
    java.lang.Exception
```

4.3.8 Methods

- **distance**

```
double distance(weka.core.Instance first ,weka.core.Instance
    second) throws java.lang.Exception
```

  - **Description copied from IDistance (in 4.1, page 46)**
    Get the distance between two bags in the form of a set of `Instance` with relational attribute.
  - **Parameters**
    * `first` – First bag as Instance with relational attribute.
    * `second` – Second Bag as Instance with relational attribute.
  - **Returns** – Distance between two bags.
  - **Throws**
    * `java.lang.Exception` – if occurred an error during distance calculation.

- **distance**

```
double distance(miml.data.MIMLBag first ,miml.data.MIMLBag second
    ) throws java.lang.Exception
```

  - **Description copied from IDistance (in 4.1, page 46)**
    Get the distance between two `MIMLBag` (in 7.1, page 96).
  - **Parameters**
    * `first` – First bag.
    * `second` – Second bag.
  - **Returns** – Distance between two bags.
  - **Throws**
    * `java.lang.Exception` – if occurred an error during distance calculation,

- **hasInstances**

```
public boolean hasInstances()
```

- **setInstances**

```
void setInstances(weka.core.Instances bags) throws java.lang.
    Exception
```

- **Description copied from IDistance** (**in** 4.1, **page** 46)
  Sets the Intances in the form of a set of Instances with relational attribute.
- **Parameters**
  - `bags` – The instances to be set.
- **Throws**
  - `java.lang.Exception` – to be handled in upper level.

• **setInstances**

```
void setInstances(miml.data.MIMLInstances bags) throws java.lang
    .Exception
```

- **Description copied from IDistance** (**in** 4.1, **page** 46)
  Sets the Intances in the form of MIMLBags.
- **Parameters**
  - `bags` – The instances to be set.
- **Throws**
  - `java.lang.Exception` – to be handled in upper level.

• **update**

```
void update(weka.core.Instance bag) throws java.lang.Exception
```

- **Description copied from IDistance** (**in** 4.1, **page** 46)
  Update the distance function (if necessary) for the newly added instance in the form of Instance with relational attribute.
- **Parameters**
  - `bag` – The bag.
- **Throws**
  - `java.lang.Exception` – to be handled in upper level.

• **update**

```
void update(miml.data.MIMLBag bag) throws java.lang.Exception
```

- **Description copied from IDistance** (**in** 4.1, **page** 46)
  Update the distance function (if necessary) for the newly added instance in the form of MIMLBag.
- **Parameters**
  - `bag` – The bag.
- **Throws**
  - `java.lang.Exception` – to be handled in upper level.

4.4 Class MaximalHausdorff

Class that implements Maximal Hausdorff metric to measure the distance between 2 bags of a data set.

4.4.1 Declaration

```
public class MaximalHausdorff
 extends miml.core.distance.HausdorffDistance
```

4.4.2 Field summary

serialVersionUID Generated Serial version UID.

4.4.3 Constructor summary

MaximalHausdorff()
MaximalHausdorff(MIMLInstances)

4.4.4 Method summary

distance(Instances, Instances)

4.4.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.

4.4.6 Constructors

- **MaximalHausdorff**

```
public MaximalHausdorff()
```

- **MaximalHausdorff**

```
public MaximalHausdorff(miml.data.MIMLInstances bags) throws
    java.lang.Exception
```

4.4.7 Methods

- **distance**

```
public double distance(weka.core.Instances first ,weka.core.
    Instances second) throws java.lang.Exception
```

4.4.8 Members inherited from class HausdorffDistance

`miml.core.distance.HausdorffDistance` (in 4.3, page 51)

- **dataSet**
- **dfun**
- `public double` **distance**`(weka.core.Instance` **bag1**`, weka.core.Instance` **bag2**`) throws java.lang.Exception`
- `public double` **distance**`(miml.data.MIMLBag` **first**`, miml.data.MIMLBag` **second**`) throws java.lang.Exception`
- `public boolean` **hasInstances**`()`
- `private static final` **serialVersionUID**
- `public void` **setInstances**`(weka.core.Instances` **bags**`) throws java.lang.Exception`
- `public void` **setInstances**`(miml.data.MIMLInstances` **bags**`) throws java.lang.Exception`
- `public void` **update**`(weka.core.Instance` **bag**`) throws java.lang.Exception`
- `public void` **update**`(miml.data.MIMLBag` **bag**`) throws java.lang.Exception`

4.5 Class MinimalHausdorff

Class that implements Minimal Hausdorff metric to measure the distance between 2 bags of a data set.

4.5.1 Declaration

```
public class MinimalHausdorff
 extends miml.core.distance.HausdorffDistance
```

4.5.2 Field summary

serialVersionUID Generated Serial version UID.

4.5.3 Constructor summary

MinimalHausdorff()
MinimalHausdorff(MIMLInstances)

4.5.4 Method summary

distance(Instances, Instances)

4.5.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.

4.5.6 Constructors

- **MinimalHausdorff**

```
public MinimalHausdorff()
```

- **MinimalHausdorff**

```
public MinimalHausdorff(miml.data.MIMLInstances bags) throws
    java.lang.Exception
```

4.5.7 Methods

- **distance**

```
public double distance(weka.core.Instances first ,weka.core.
    Instances second) throws java.lang.Exception
```

4.5.8 Members inherited from class HausdorffDistance

`miml.core.distance.HausdorffDistance` (in 4.3, page 51)

- **dataSet**
- **dfun**
- `public double` **distance**`(weka.core.Instance` **bag1**`, weka.core.Instance` **bag2**`) throws java.lang.Exception`
- `public double` **distance**`(miml.data.MIMLBag` **first**`, miml.data.MIMLBag` **second**`) throws java.lang.Exception`
- `public boolean` **hasInstances**`()`
- `private static final` **serialVersionUID**
- `public void` **setInstances**`(weka.core.Instances` **bags**`) throws java.lang.Exception`
- `public void` **setInstances**`(miml.data.MIMLInstances` **bags**`) throws java.lang.Exception`
- `public void` **update**`(weka.core.Instance` **bag**`) throws java.lang.Exception`
- `public void` **update**`(miml.data.MIMLBag` **bag**`) throws java.lang.Exception`

Chapter 5

Package miml.classifiers.miml.optimization

Package Contents *Page*

5.1 Class KiSar

Wrapper for Matlab **KiSar** algorithm for MIML data.
For more information see: *Y.-F. Li, J.-H. Hu, Y. Jiang, and Z.-H. Zhou. Towards discovering what patterns trigger what labels. In: Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI'12), Toronto, Canada, 2012.* It uses LIBLINEAR, compiled for Windows 64 bits see:
R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. Journal of Machine Learning Research 9(2008), 1871-1874.

5.1.1 Declaration

```
public class KiSar
 extends miml.classifiers.miml.MWClassifier
```

5.1.2 Field summary

C Parameter set for liblinear.

epsilon The epsilon parameter for the algorithm.
iteration Maximum number of optimization iterations.
K Maximum number of prototypes for k_means clustering.
kisar A Matlab object wrapping the KiSar algorithm.
relationMethod Method used to build relation matrix.
serialVersionUID For serialization.

5.1.3 Constructor summary

KiSar() No-argument constructor for xml configuration.
KiSar(double, double, double, double, double) Constuctor initializing fields of KiSar.

5.1.4 Method summary

configure(Configuration)
dispose()
getC() Gets the value of the C property.
getEpsilon() Gets the value of the epsilon property.
getIteration() Gets the value of the iteration property.
getK() Gets the value of the K property.
getRelationMethod() Gets the value of the relationMethod property.
predictMWClassifier(MWCellArray, MWNumericArray, MWNumericArray)
setC(double) Sets the value of the C property.
setEpsilon(double) Sets the value of the epsilon property.
setIteration(double) Sets the value of the iteration property.
setK(double) Sets the value of the k property.
setRelationMethod(double) Sets the value of the relationMethod property.
trainMWClassifier(MWCellArray, MWNumericArray)

5.1.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `MWAlgorithms.MWKiSar` **kisar**
  - A Matlab object wrapping the KiSar algorithm.
- `double` **C**
  - Parameter set for liblinear.
- `double` **iteration**
  - Maximum number of optimization iterations.
- `double` **epsilon**
  - The epsilon parameter for the algorithm.

- `double` **K**
  - Maximum number of prototypes for k_means clustering.
- `double` **relationMethod**
  - Method used to build relation matrix.
    * 1 =>the identity matrix is returned. No cooccurrences.
    * 2 =>all labels are related.
    * 3 =>labels i,j coocur if their coocurrence values are greater than the mean of all values in the coocurrence matrix (including main diagonal).
    * 4 =>labels i,j coocur if their coocurrence values are greater than the mean of the coocurrence values of all labels (excluding main diagonal).
    * 5 =>labels i,j coocur if prob(i, j) >min(prob(i), prob(j))*0.1 (10 percent).

5.1.6 Constructors

- **KiSar**

```
public KiSar() throws com.mathworks.toolbox.javabuilder.
    MWException
```

  - **Description**
    No-argument constructor for xml configuration.
  - **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **KiSar**

```
public KiSar(double c,double iteration ,double epsilon ,double k,
    double relationMethod) throws com.mathworks.toolbox.
    javabuilder.MWException
```

  - **Description**
    Constuctor initializing fields of KiSar.
  - **Parameters**
    * `c` – parameter for liblinear
    * `iteration` – value for iteration
    * `epsilon` – value for epsilon
    * `k` – Maximum number of prototypes
    * `relationMethod` – Method used to build the relationMatrix.
  - **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – to be handled in upper level.

5.1.7 Methods

- **configure**

  ```
  public void configure(org.apache.commons.configuration2.
      Configuration configuration)
  ```

- **dispose**

  ```
  public abstract void dispose()
  ```

  - **Description copied from miml.classifiers.miml.MWClassifier** (**in 10.3, page 145**)
    Disposes native MW classifier. This method should be called if the classifier is not been used anymore in the program in order to free the memory that the MW classifier was using.

- **getC**

  ```
  public double getC()
  ```

  - **Description**
    Gets the value of the C property.
  - **Returns** – double

- **getEpsilon**

  ```
  public double getEpsilon()
  ```

  - **Description**
    Gets the value of the epsilon property.
  - **Returns** – double

- **getIteration**

  ```
  public double getIteration()
  ```

  - **Description**
    Gets the value of the iteration property.
  - **Returns** – double

- **getK**

```
public double getK()
```

- **Description**
  Gets the value of the K property.
- **Returns** – double

• **getRelationMethod**

```
public double getRelationMethod()
```

- **Description**
  Gets the value of the relationMethod property.
- **Returns** – double

• **predictMWClassifier**

```
protected abstract java.lang.Object[] predictMWClassifier(com.
    mathworks.toolbox.javabuilder.MWCellArray train_bags,com.
    mathworks.toolbox.javabuilder.MWNumericArray train_targets,
    com.mathworks.toolbox.javabuilder.MWNumericArray test_bag)
    throws com.mathworks.toolbox.javabuilder.MWException
```

- **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**
  Performs a prediction on a test bag.
- **Parameters**
  * `train_bags` – Bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
  * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
  * `test_bag` – A test bag. It will be a MIMLBag in the format of a nInstxnAttributes MWNumericArray of double.
- **Returns** – An array of 2 Object:
  * Object[0] is a nLabelsx1 array of double containing the probability of the testing instance belonging to each label.
  * Object[1] is a nLabelsx1 array of double containing a bipartition being 1 if the label is relevant or -1 otherwise.
- **Throws**
  * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **setC**

  ```
  public void setC(double c)
  ```

  - **Description**
    Sets the value of the C property.
  - **Parameters**
    - `c` – The new value for the property.

- **setEpsilon**

  ```
  public void setEpsilon(double epsilon)
  ```

  - **Description**
    Sets the value of the epsilon property.
  - **Parameters**
    - `epsilon` – The new value for the property.

- **setIteration**

  ```
  public void setIteration(double iteration)
  ```

  - **Description**
    Sets the value of the iteration property.
  - **Parameters**
    - `iteration` – The new value for the property.

- **setK**

  ```
  public void setK(double k)
  ```

  - **Description**
    Sets the value of the k property.
  - **Parameters**
    - `k` – The new value for the property.

- **setRelationMethod**

  ```
  public void setRelationMethod(double relationMethod)
  ```

- **Description**

  Sets the value of the relationMethod property.
- **Parameters**
  * `relationMethod` – The new value for the property

- **trainMWClassifier**

```
protected abstract void trainMWClassifier(com.mathworks.toolbox.
    javabuilder.MWCellArray train_bags ,com.mathworks.toolbox.
    javabuilder.MWNumericArray train_targets) throws com.
    mathworks.toolbox.javabuilder.MWException
```

- **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**

  Trains a Matlab classifier. Returns the classifier model in an array of Object.
- **Parameters**
  * `train_bags` – bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
  * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
- **Throws**
  * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

5.1.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 10.3, page 145)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public abstract void` **dispose**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **aBag**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected abstract Object` **predictMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **test_bag**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `private static final` **serialVersionUID**
- `protected abstract void` **trainMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `protected` **wrapper**

5.1.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

5.2 Class MIMLFast

Wrapper for Matlab **MIMLFast** algorithm for MIML data.
See: *S.-J. Huang W. Gao and Z.-H. Zhou. Fast multi-instance multi-label learning. In: Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI'14), 2014.*

5.2.1 Declaration

```
public class MIMLFast
 extends miml.classifiers.miml.MWClassifier
```

5.2.2 Field summary

D Dimension of the shared space.
lambda Lambda.
maxiter Number of iterations.
mimlfast A matlab object wrapping the MIMLFast algorithm.
norm_up Norm of each vector.
num_sub Number of sub concepts.
opts_average_begin
opts_average_size
opts_norm
serialVersionUID For serialization.
step_size Step size of SGD (stochastic gradient descent).

5.2.3 Constructor summary

MIMLFast() No-argument constructor for xml configuration.
MIMLFast(int, int, int, double, double, int, int, int, int) Constructor setting several properties.
MIMLFast(int, int, int, double, int) Constructor setting several properties.

5.2.4 Method summary

configure(Configuration)
dispose()
getD() Gets the value of the D property.
getLambda() Gets the value of the lambda property.
getMaxiter() Gets the value of the maxiter property.
getNorm_up() Gets the value of the norm_up property.
getNum_sub() Gets the value of the num_sub property.
getOpts_average_begin() Gets the value of the opts_average_begin property.
getOpts_average_size() Gets the value of the opts_average_size property.
getOpts_norm() Gets the value of the opts_norm property.
getStep_size() Gets the value of the step_size property.
predictMWClassifier(MWCellArray, MWNumericArray, MWNumericArray)
setD(int) Sets the value of the D property.
setLambda(double) Sets the value of the lambda property.
setMaxiter(int) Sets the value of the maxiter property.
setNorm_up(int) Sets the value of the norm_up property.
setNum_sub(int) Sets the value of the num_sub property.
setOpts_average_begin(int) Sets the value of the opts_average_begin property.
setOpts_average_size(int) Sets the value of the opts_average_size property.
setOpts_norm(int) Sets the value of the opts_norm property.
setStep_size(double) Sets the value of the step_size property.
trainMWClassifier(MWCellArray, MWNumericArray)

5.2.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `static MWAlgorithms.MWMIMLFast` **mimlfast**
  - A matlab object wrapping the MIMLFast algorithm.
- `int` **D**
  - Dimension of the shared space.
- `int` **norm_up**
  - Norm of each vector.
- `int` **maxiter**

  - Number of iterations.
- `double` **step_size**
  - Step size of SGD (stochastic gradient descent).
- `double` **lambda**
  - Lambda.
- `int` **num_sub**
  - Number of sub concepts.
- `int` **opts_norm**
- `int` **opts_average_size**
- `int` **opts_average_begin**

5.2.6 Constructors

- **MIMLFast**

```
public MIMLFast() throws com.mathworks.toolbox.javabuilder.
    MWException
```

  - **Description**
    No-argument constructor for xml configuration.
  - **Throws**
    - `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **MIMLFast**

```
public MIMLFast(int d,int norm_up,int maxiter,double step_size,
    double lambda,int num_sub,int opts_norm,int opts_average_size
    ,int opts_average_begin) throws com.mathworks.toolbox.
    javabuilder.MWException
```

  - **Description**
    Constructor setting several properties.
  - **Parameters**
    - `d` – Value for d.
    - `norm_up` – Value for norm_up.
    - `maxiter` – Value for maxiter.
    - `step_size` – Value for step_size.
    - `num_sub` – Value for num_sub.
    - `lambda` – Value for lambda.

  * `opts_norm` – Value for opts_norm.
  * `opts_average_size` – Value for opts_average_size.
  * `opts_average_begin` – Value for opts_average_begin.
- **Throws**
  * `com.mathworks.toolbox.javabuilder.MWException` – To be handled in upper level.

• **MIMLFast**

```
public MIMLFast(int d,int norm_up,int maxiter,double step_size,
    int num_sub) throws com.mathworks.toolbox.javabuilder.
    MWException
```

- **Description**

  Constructor setting several properties.
- **Parameters**
  * `d` – Value for d.
  * `norm_up` – Value for norm_up.
  * `maxiter` – Value for maxiter.
  * `step_size` – Value for step_size.
  * `num_sub` – Value for num_sub.
- **Throws**
  * `com.mathworks.toolbox.javabuilder.MWException` – To be handled in upper level.

5.2.7 Methods

• **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

• **dispose**

```
public abstract void dispose()
```

- **Description copied from miml.classifiers.miml.MWClassifier** (**in** 10.3, **page** 145)

  Disposes native MW classifier. This method should be called if the classifier is not been used anymore in the program in order to free the memory that the MW classifier was using.

- **getD**

  ```
  public int getD()
  ```

  - **Description**
    Gets the value of the D property.
  - **Returns** – int

- **getLambda**

  ```
  public double getLambda()
  ```

  - **Description**
    Gets the value of the lambda property.
  - **Returns** – double

- **getMaxiter**

  ```
  public int getMaxiter()
  ```

  - **Description**
    Gets the value of the maxiter property.
  - **Returns** – int

- **getNorm_up**

  ```
  public int getNorm_up()
  ```

  - **Description**
    Gets the value of the norm_up property.
  - **Returns** – int

- **getNum_sub**

  ```
  public int getNum_sub()
  ```

  - **Description**
    Gets the value of the num_sub property.
  - **Returns** – int

- **getOpts_average_begin**

```
public int getOpts_average_begin()
```

- **Description**
  Gets the value of the opts_average_begin property.
- **Returns** – int

• **getOpts_average_size**

```
public int getOpts_average_size()
```

- **Description**
  Gets the value of the opts_average_size property.
- **Returns** – int

• **getOpts_norm**

```
public int getOpts_norm()
```

- **Description**
  Gets the value of the opts_norm property.
- **Returns** – int

• **getStep_size**

```
public double getStep_size()
```

- **Description**
  Gets the value of the step_size property.
- **Returns** – double

• **predictMWClassifier**

```
protected abstract java.lang.Object[] predictMWClassifier(com.
    mathworks.toolbox.javabuilder.MWCellArray train_bags,com.
    mathworks.toolbox.javabuilder.MWNumericArray train_targets,
    com.mathworks.toolbox.javabuilder.MWNumericArray test_bag)
    throws com.mathworks.toolbox.javabuilder.MWException
```

- **Description copied from miml.classifiers.miml.MWClassifier** (**in 10.3, page 145**)
  Performs a prediction on a test bag.
- **Parameters**

    * `train_bags` – Bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
    * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
    * `test_bag` – A test bag. It will be a MIMLBag in the format of a nInstxnAttributes MWNumericArray of double.
  - **Returns** – An array of 2 Object:
    * Object[0] is a nLabelsx1 array of double containing the probability of the testing instance belonging to each label.
    * Object[1] is a nLabelsx1 array of double containing a bipartition being 1 if the label is relevant or -1 otherwise.
  - **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **setD**

```
public void setD(int d)
```

  - **Description**
    Sets the value of the D property.
  - **Parameters**
    * `d` – The new value for the property.

- **setLambda**

```
public void setLambda(double lambda)
```

  - **Description**
    Sets the value of the lambda property.
  - **Parameters**
    * `lambda` – The new value for the property.

- **setMaxiter**

```
public void setMaxiter(int maxiter)
```

  - **Description**
    Sets the value of the maxiter property.
  - **Parameters**

    * `maxiter` – The new value for the property.

- **setNorm_up**

  ```
  public void setNorm_up(int norm_up)
  ```

  - **Description**
    Sets the value of the norm_up property.
  - **Parameters**
    * `norm_up` – The new value for the property.

- **setNum_sub**

  ```
  public void setNum_sub(int num_sub)
  ```

  - **Description**
    Sets the value of the num_sub property.
  - **Parameters**
    * `num_sub` – The new value for the property.

- **setOpts_average_begin**

  ```
  public void setOpts_average_begin(int opts_average_begin)
  ```

  - **Description**
    Sets the value of the opts_average_begin property.
  - **Parameters**
    * `opts_average_begin` – The new value for the property.

- **setOpts_average_size**

  ```
  public void setOpts_average_size(int opts_average_size)
  ```

  - **Description**
    Sets the value of the opts_average_size property.
  - **Parameters**
    * `opts_average_size` – The new value for the property.

- **setOpts_norm**

  ```
  public void setOpts_norm(int opts_norm)
  ```

- **Description**
  Sets the value of the opts_norm property.
- **Parameters**
  * `opts_norm` – The new value for the property.

• **setStep_size**

```
public void setStep_size(double step_size)
```

- **Description**
  Sets the value of the step_size property.
- **Parameters**
  * `step_size` – The new value for the property.

• **trainMWClassifier**

```
protected abstract void trainMWClassifier(com.mathworks.toolbox.
    javabuilder.MWCellArray train_bags ,com.mathworks.toolbox.
    javabuilder.MWNumericArray train_targets) throws com.
    mathworks.toolbox.javabuilder.MWException
```

- **Description copied from miml.classifiers.miml.MWClassifier** (**in** 10.3, **page** 145)
  Trains a Matlab classifier. Returns the classifier model in an array of Object.
- **Parameters**
  * `train_bags` – bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
  * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
- **Throws**
  * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

5.2.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 10.3, page 145)

• `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
• `protected` **classifier**
• `public abstract void` **dispose**`()`
• `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **aBag**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`

- `protected abstract Object` **predictMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **test_bag**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `private static final` **serialVersionUID**
- `protected abstract void` **trainMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `protected` **wrapper**

5.2.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

5.3 Class MIMLSVM

Wrapper for Matlab **MIMLSVM** algorithm for MIML data.
See: *Z.-H. Zhou and M.-L. Zhang. Multi-instance multi-label learning with application to scene classification. In: Advances in Neural Information Processing Systems 19 (NIPS'06) (Vancouver, Canada) Cambridge, MA: MIT Press, 2007.BIOwulf Technologies, 2001.* It employs Libsvm compiled for Windows 64 bits (available at href="https://www.csie.ntu.edu.tw/ cjlin/libsvm/) as the base learners.

5.3.1 Declaration

```
public class MIMLSVM
 extends miml.classifiers.miml.MWClassifier
```

5.3.2 Field summary

cost The cost parameter used for the base svm classifier.
h Whether to use the shrinking heuristics, 0 or 1 (default 1).
mimlsvm A matlab object wrapping the MIMLSVM algorithm.
para A string that gives the corresponding parameters used for the svm:

- If type is "RBF", para gives the value of gamma (i.e. para="1") where the kernel is exp(-Gamma*—x(i)-x(j)—∧2).

ratio Parameter k is set to be 20% of the number of training bags.
seed Seed for kmedoids clustering.
serialVersionUID For serialization.
type Gaussian kernel SVM.

5.3.3 Constructor summary

MIMLSVM() No-argument constructor for xml configuration.
MIMLSVM(String, String, double, double, double, double) Constructor initializing fields of MIMLSVM.

5.3.4 Method summary

configure(Configuration)
dispose()
getCost() Gets the value of the cost property.
getH() Gets the value of the h property.
getPara() Gets the value of the para property.
getRatio() Gets the value of the ratio property.
getSeed() Gets the value of the seed property.
getType() Gets the value of the type property.
predictMWClassifier(MWCellArray, MWNumericArray, MWNumericArray)
setCost(double) Sets the value of the cost property.
setH(double) Sets the value of the h property.
setPara(String) Sets the value of the para property.
setRatio(double) Sets the value of the ratio property.
setSeed(double) Sets the value of the seed property.
setType(String) Sets the value of the type property.
trainMWClassifier(MWCellArray, MWNumericArray)

5.3.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `MWAlgorithms.MWMIMLSVM` **mimlsvm**
  - A matlab object wrapping the MIMLSVM algorithm.

- `java.lang.String` **type**
  - Gaussian kernel SVM. The type of svm used in training, which can take the value of "RBF", "Poly" or "Linear".
- `java.lang.String` **para**
  - A string that gives the corresponding parameters used for the svm:
    * If type is "RBF", para gives the value of gamma (i.e. para="1") where the kernel is exp(-Gamma*—x(i)-x(j)—∧2).
    * If type is "Poly", then para gives the value of gamma, coefficient, and degree respectively, where the kernel is (gamma*<x(i),x(j)>+coefficient)∧degree. Values in the string are delimited by blank spaces (i.e. para="1, 0, 1").
    * If type is "Linear", then para is an empty string, where the kernel is <x(i),x(j)>(i.e. para ="").
- `double` **cost**
  - The cost parameter used for the base svm classifier.
- `double` **h**
  - Whether to use the shrinking heuristics, 0 or 1 (default 1).
- `double` **ratio**
  - Parameter k is set to be 20% of the number of training bags.
- `double` **seed**
  - Seed for kmedoids clustering.

5.3.6 Constructors

- **MIMLSVM**

```
public MIMLSVM() throws com.mathworks.toolbox.javabuilder.
    MWException
```

  - **Description**
    No-argument constructor for xml configuration.
  - **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **MIMLSVM**

```
public MIMLSVM(java.lang.String type,java.lang.String para,
    double cost,double h,double ratio,double seed) throws com.
    mathworks.toolbox.javabuilder.MWException
```

- **Description**
  Constructor initializing fields of MIMLSVM.
- **Parameters**
  * `type` – Value for type field.
  * `para` – Value for para field.
  * `cost` – Value for cost field.
  * `h` – Value for h field.
  * `ratio` – Value for ratio field.
  * `seed` – Value for seed field.
- **Throws**
  * `com.mathworks.toolbox.javabuilder.MWException` – To be handled in upper level.

5.3.7 Methods

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **dispose**

```
public abstract void dispose()
```

  - **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**
    Disposes native MW classifier. This method should be called if the classifier is not been used anymore in the program in order to free the memory that the MW classifier was using.

- **getCost**

```
public double getCost()
```

  - **Description**
    Gets the value of the cost property.
  - **Returns** – double

- **getH**

```
public double getH()
```

  - **Description**
    Gets the value of the h property.
  - **Returns** – double

- **getPara**

```
public java.lang.String getPara()
```

  - **Description**
    Gets the value of the para property.
  - **Returns** – String

- **getRatio**

```
public double getRatio()
```

  - **Description**
    Gets the value of the ratio property.
  - **Returns** – double

- **getSeed**

```
public double getSeed()
```

  - **Description**
    Gets the value of the seed property.
  - **Returns** – double

- **getType**

```
public java.lang.String getType()
```

  - **Description**
    Gets the value of the type property.
  - **Returns** – String

- **predictMWClassifier**

```
protected abstract java.lang.Object[] predictMWClassifier(com.
    mathworks.toolbox.javabuilder.MWCellArray train_bags,com.
    mathworks.toolbox.javabuilder.MWNumericArray train_targets,
    com.mathworks.toolbox.javabuilder.MWNumericArray test_bag)
    throws com.mathworks.toolbox.javabuilder.MWException
```

  - **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**
    Performs a prediction on a test bag.
  - **Parameters**
    * `train_bags` – Bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
    * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
    * `test_bag` – A test bag. It will be a MIMLBag in the format of a nInstxnAttributes MWNumericArray of double.
  - **Returns** – An array of 2 Object:
    * Object[0] is a nLabelsx1 array of double containing the probability of the testing instance belonging to each label.
    * Object[1] is a nLabelsx1 array of double containing a bipartition being 1 if the label is relevant or -1 otherwise.
  - **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **setCost**

```
public void setCost(double cost)
```

  - **Description**
    Sets the value of the cost property.
  - **Parameters**
    * `cost` – The new value for the property.

- **setH**

```
public void setH(double h)
```

  - **Description**
    Sets the value of the h property.
  - **Parameters**
    * `h` – The new value for the property.

- **setPara**

```
public void setPara(java.lang.String para)
```

- **Description**
  Sets the value of the para property.
- **Parameters**
  * `para` – The new value for the property.

- **setRatio**

```
public void setRatio(double ratio)
```

  - **Description**
    Sets the value of the ratio property.
  - **Parameters**
    * `ratio` – The new value for the property.

- **setSeed**

```
public void setSeed(double seed)
```

  - **Description**
    Sets the value of the seed property.
  - **Parameters**
    * `seed` – The new value for the property.

- **setType**

```
public void setType(java.lang.String type)
```

  - **Description**
    Sets the value of the type property.
  - **Parameters**
    * `type` – The new value for the property.

- **trainMWClassifier**

```
protected abstract void trainMWClassifier(com.mathworks.toolbox.
    javabuilder.MWCellArray train_bags ,com.mathworks.toolbox.
    javabuilder.MWNumericArray train_targets) throws com.
    mathworks.toolbox.javabuilder.MWException
```

  - **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**
    Trains a Matlab classifier. Returns the classifier model in an array of Object.

- **Parameters**
  * `train_bags` – bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
  * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
- **Throws**
  * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

5.3.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 10.3, page 145)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public abstract void` **dispose**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **aBag**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected abstract Object` **predictMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **test_bag**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `private static final` **serialVersionUID**
- `protected abstract void` **trainMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `protected` **wrapper**

5.3.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`

- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

5.4 Class MIMLWel

Wrapper for Matlab **MIMLFast** algorithm for MIML data.
See: *S.-J. Yang, Y. Jiang, and Z.-H. Zhou. Multi-instance multi-label learning with weak label.In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI'13), Beijing, China, 2013.*

5.4.1 Declaration

```
public class MIMLWel
 extends miml.classifiers.miml.MWClassifier
```

5.4.2 Field summary

mimlwel A matlab object wrapping the MIMLWel algorithm.
mu The ratio used to determine the standard deviation of the Gaussian activation function.
opts_beta Controls the similarity between training_bags and their prototypes.
opts_C Controls the empirical loss on labeled data.
opts_epsilon Value for epsilon.
opts_iteration Iteration number.
opts_m Controls the difference between the learned training targets and the original input training targets.
ratio The number of centroids of the i-th class is set to be ratio*Ti, where Ti is the number of train bags with label i.
serialVersionUID For serialization.

5.4.3 Constructor summary

MIMLWel() No-argument constructor for xml configuration.
MIMLWel(double, double, double, double, double, double, double) Constructor initializing fields of MIMLWel.

5.4.4 Method summary

configure(Configuration)
dispose()
getMu() Gets the value of the mu property.
getOpts_beta() Gets the value of the opts_beta property.
getOpts_C() Gets the value of the opts_C property.
getOpts_epsilon() Gets the value of the opts_epsilon property.

getOpts_iteration() Gets the value of the opts_iteration property.
getOpts_m() Gets the value of the opts_m property.
getRatio() Gets the value of the ratio property.
predictMWClassifier(MWCellArray, MWNumericArray, MWNumericArray)
setMu(double) Sets the value of the mu property.
setOpts_beta(double) Sets the value of the beta property.
setOpts_C(int) Sets the value of the opts_C property.
setOpts_epsilon(double) Sets the value of the opts_epsilon property.
setOpts_iteration(int) Sets the value of the opts_iteration property.
setOpts_m(double) Sets the value of the opts_m property.
setRatio(double) Sets the value of the ratio property.
trainMWClassifier(MWCellArray, MWNumericArray)

5.4.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `MWAlgorithms.MWMIMLWel` **mimlwel**
  - A matlab object wrapping the MIMLWel algorithm.
- `double` **opts_C**
  - Controls the empirical loss on labeled data.
- `double` **opts_m**
  - Controls the difference between the learned training targets and the original input training targets.
- `double` **opts_beta**
  - Controls the similarity between training_bags and their prototypes.
- `double` **opts_iteration**
  - Iteration number.
- `double` **opts_epsilon**
  - Value for epsilon.
- `double` **ratio**
  - The number of centroids of the i-th class is set to be ratio*Ti, where Ti is the number of train bags with label i.
- `double` **mu**
  - The ratio used to determine the standard deviation of the Gaussian activation function.

5.4.6 Constructors

- **MIMLWel**

```
public MIMLWel() throws com.mathworks.toolbox.javabuilder.
    MWException
```

  - **Description**
    No-argument constructor for xml configuration.
  - **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **MIMLWel**

```
public MIMLWel(double opts_C ,double opts_m ,double opts_beta ,
    double opts_iteration ,double opts_epsilon ,double ratio ,double
     mu) throws com.mathworks.toolbox.javabuilder.MWException
```

  - **Description**
    Constructor initializing fields of MIMLWel.
  - **Parameters**
    * `opts_C` – Value for the opts_C field.
    * `opts_m` – Value for the opts_m field.
    * `opts_beta` – Value for the opts_beta field.
    * `opts_iteration` – Value for the opts_iteration field.
    * `opts_epsilon` – Value for the opts_epsilon field.
    * `ratio` – Value for the ratio field.
    * `mu` – Value for the mu field.
  - **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled in upper level.

5.4.7 Methods

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **dispose**

```
public abstract void dispose()
```

  – **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**
    Disposes native MW classifier. This method should be called if the classifier is not been used anymore in the program in order to free the memory that the MW classifier was using.

- **getMu**

```
public double getMu()
```

  – **Description**
    Gets the value of the mu property.
  – **Returns** – double

- **getOpts_beta**

```
public double getOpts_beta()
```

  – **Description**
    Gets the value of the opts_beta property.
  – **Returns** – double

- **getOpts_C**

```
public double getOpts_C()
```

  – **Description**
    Gets the value of the opts_C property.
  – **Returns** – double

- **getOpts_epsilon**

```
public double getOpts_epsilon()
```

  – **Description**
    Gets the value of the opts_epsilon property.
  – **Returns** – double

- **getOpts_iteration**

```
public double getOpts_iteration()
```

  - **Description**
    Gets the value of the opts_iteration property.
  - **Returns** – double

- **getOpts_m**

```
public double getOpts_m()
```

  - **Description**
    Gets the value of the opts_m property.
  - **Returns** – double

- **getRatio**

```
public double getRatio()
```

  - **Description**
    Gets the value of the ratio property.
  - **Returns** – double

- **predictMWClassifier**

```
protected abstract java.lang.Object[] predictMWClassifier(com.
    mathworks.toolbox.javabuilder.MWCellArray train_bags,com.
    mathworks.toolbox.javabuilder.MWNumericArray train_targets,
    com.mathworks.toolbox.javabuilder.MWNumericArray test_bag)
    throws com.mathworks.toolbox.javabuilder.MWException
```

  - **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**
    Performs a prediction on a test bag.
  - **Parameters**
    * `train_bags` – Bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
    * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
    * `test_bag` – A test bag. It will be a MIMLBag in the format of a nInstxnAttributes MWNumericArray of double.
  - **Returns** – An array of 2 Object:

    * Object[0] is a nLabelsx1 array of double containing the probability of the testing instance belonging to each label.
    * Object[1] is a nLabelsx1 array of double containing a bipartition being 1 if the label is relevant or -1 otherwise.
  - **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **setMu**

```
public void setMu(double mu)
```

  - **Description**
    Sets the value of the mu property.
  - **Parameters**
    * `mu` – The new value for the property.

- **setOpts_beta**

```
public void setOpts_beta(double opts_beta)
```

  - **Description**
    Sets the value of the beta property.
  - **Parameters**
    * `opts_beta` – The new value for the property.

- **setOpts_C**

```
public void setOpts_C(int opts_C)
```

  - **Description**
    Sets the value of the opts_C property.
  - **Parameters**
    * `opts_C` – The new value for the property.

- **setOpts_epsilon**

```
public void setOpts_epsilon(double opts_epsilon)
```

  - **Description**
    Sets the value of the opts_epsilon property.
  - **Parameters**

    * `opts_epsilon` – The new value for the property.

- **setOpts_iteration**

  ```
  public void setOpts_iteration(int opts_iteration)
  ```

  - **Description**
    Sets the value of the opts_iteration property.
  - **Parameters**
    * `opts_iteration` – The new value for the property.

- **setOpts_m**

  ```
  public void setOpts_m(double opts_m)
  ```

  - **Description**
    Sets the value of the opts_m property.
  - **Parameters**
    * `opts_m` – The new value for the property.

- **setRatio**

  ```
  public void setRatio(double ratio)
  ```

  - **Description**
    Sets the value of the ratio property.
  - **Parameters**
    * `ratio` – The new value for the property.

- **trainMWClassifier**

  ```
  protected abstract void trainMWClassifier(com.mathworks.toolbox.
      javabuilder.MWCellArray train_bags ,com.mathworks.toolbox.
      javabuilder.MWNumericArray train_targets) throws com.
      mathworks.toolbox.javabuilder.MWException
  ```

  - **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**
    Trains a Matlab classifier. Returns the classifier model in an array of Object.
  - **Parameters**

    * `train_bags` – bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
    * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
  - **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

5.4.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 10.3, page 145)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public abstract void` **dispose**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **aBag**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected abstract Object` **predictMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **test_bag**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `private static final` **serialVersionUID**
- `protected abstract void` **trainMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `protected` **wrapper**

5.4.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

Chapter 6

Package miml.classifiers.miml.meta

6.1 Class MIMLBagging

MIMLBagging is the adaptation of the traditional bagging strategy of the machine learning [1] that does not need any previous transformation of the problem. *[1]Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123-140.*

6.1.1 Declaration

```
public class MIMLBagging
 extends miml.classifiers.miml.MIMLClassifier
```

6.1.2 Field summary

baseLearner Base learner.
ensemble The ensemble of MultiLabelLearners.
numClassifiers Number of classifiers in the ensemble.
samplePercentage The size of the sample to build each base classifier.
sampleWithReplacement Determines whether the classifier will consider sampling with replacement.
seed Seed for randomization.
serialVersionUID Generated Serial version UID.
threshold Threshold for predictions.
useConfidences Determines whether confidences [0,1] or relevance {0,1} is used to compute bipartition.

6.1.3 Constructor summary

MIMLBagging() No-argument constructor for xml configuration.
MIMLBagging(IMIMLClassifier, int) Constructor of the class.
MIMLBagging(IMIMLClassifier, int, double) Constructor of the class.

6.1.4 Method summary

buildInternal(MIMLInstances)
configure(Configuration)
getNumClassifiers() Returns the number of classifiers of the ensemble.
getSamplePercentage() Returns the percentage of instances used for sampling with replacement.
getThreshold() Returns the value of the threshold.
isSampleWithReplacement() Returns true if the algorithm is configured with sampling and false otherwise.
isUseConfidences() Returns whether the classifier uses confidences of bipartitions to combine classifiers in the ensemble.
makePredictionInternal(MIMLBag)
setSamplePercentage(double) Sets the percentage of instances used for sampling with replacement*.
setSampleWithReplacement(boolean) Configure the classifier to use/not use sampling with replacement.
setSeed(int) Sets the seed value.
setThreshold(double) Sets the value of the threshold.
setUseConfidences(boolean) Stablishes whether confidences or bipartitions are used to combine classifiers in the ensemble.

6.1.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.
- `protected double` **threshold**
  - Threshold for predictions.
- `protected int` **seed**
  - Seed for randomization.
- `boolean` **sampleWithReplacement**
  - Determines whether the classifier will consider sampling with replacement. By default it is false.
- `boolean` **useConfidences**
  - Determines whether confidences [0,1] or relevance {0,1} is used to compute bipartition.
- `double` **samplePercentage**

  - The size of the sample to build each base classifier.

- `protected int` **numClassifiers**
  - Number of classifiers in the ensemble.
- `protected miml.classifiers.miml.IMIMLClassifier` **baseLearner**
  - Base learner.
- `protected miml.classifiers.miml.IMIMLClassifier[]` **ensemble**
  - The ensemble of MultiLabelLearners. To be initialized by the builder method.

6.1.6 Constructors

- **MIMLBagging**

```
public MIMLBagging()
```

  - **Description**
    No-argument constructor for xml configuration.

- **MIMLBagging**

```
public MIMLBagging(miml.classifiers.miml.IMIMLClassifier
    baseLearner,int numClassifiers)
```

  - **Description**
    Constructor of the class. Its default setting is: @li sampleWithReplacement=false @li threshold=0.5.
  - **Parameters**
    * `baseLearner` – The base learner to be used.
    * `numClassifiers` – The number of base classifiers in the ensemble.

- **MIMLBagging**

```
public MIMLBagging(miml.classifiers.miml.IMIMLClassifier
    baseLearner,int numClassifiers,double samplePercentage)
```

  - **Description**
    Constructor of the class. Its default setting is: @li sampleWithReplacement=false @li threshold=0.5.
  - **Parameters**
    * `baseLearner` – The base learner to be used.
    * `numClassifiers` – The number of base classifiers in the ensemble.
    * `samplePercentage` – The size of the sample to build each base classifier.

6.1.7 Methods

- **buildInternal**

```
protected abstract void buildInternal(miml.data.MIMLInstances
    trainingSet) throws java.lang.Exception
```

  - **Description copied from miml.classifiers.miml.MIMLClassifier** (**in** 10.2, **page** 141)

    Learner specific implementation of building the model from `MultiLabelInstances` training data set. This method is called from `build(MultiLabelInstances)` method, where behavior common across all learners is applied.
  - **Parameters**
    - `trainingSet` – The training data set.
  - **Throws**
    - `java.lang.Exception` – if learner model was not created successfully.

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **getNumClassifiers**

```
public int getNumClassifiers()
```

  - **Description**

    Returns the number of classifiers of the ensemble.
  - **Returns** – Number of classifiers.

- **getSamplePercentage**

```
public double getSamplePercentage()
```

  - **Description**

    Returns the percentage of instances used for sampling with replacement.
  - **Returns** – The sample percentage.

- **getThreshold**

```
public double getThreshold()
```

  - **Description**
    Returns the value of the threshold.
  - **Returns** – double The threshold.

- **isSampleWithReplacement**

```
public boolean isSampleWithReplacement()
```

  - **Description**
    Returns true if the algorithm is configured with sampling and false otherwise.
  - **Returns** – True if the algorithm is configured with sampling and false otherwise.

- **isUseConfidences**

```
public boolean isUseConfidences()
```

  - **Description**
    Returns whether the classifier uses confidences of bipartitions to combine classifiers in the ensemble.
  - **Returns** – True, if is use confidences.

- **makePredictionInternal**

```
protected abstract mulan.classifier.MultiLabelOutput
    makePredictionInternal(miml.data.MIMLBag instance) throws
    java.lang.Exception, mulan.classifier.InvalidDataException
```

  - **Description copied from miml.classifiers.miml.MIMLClassifier** (**in** 10.2, **page** 141)
    Learner specific implementation for predicting on specified data based on trained model. This method is called from `makePrediction(Instance)` which guards for model initialization and apply common handling/behavior.
  - **Parameters**
    * `instance` – The data instance to predict on.
  - **Returns** – The output of the learner for the given instance.
  - **Throws**
    * `java.lang.Exception` – If an error occurs while making the prediction.
    * `mulan.classifier.InvalidDataException` – If specified instance data is invalid and can not be processed by the learner.

- **setSamplePercentage**

```
public void setSamplePercentage(double samplePercentage)
```

- **Description**
  Sets the percentage of instances used for sampling with replacement*.
- **Parameters**
  * `samplePercentage` – The size of the sample referring the original one.

• **setSampleWithReplacement**

```
public void setSampleWithReplacement(boolean
    sampleWithReplacement)
```

- **Description**
  Configure the classifier to use/not use sampling with replacement.
- **Parameters**
  * `sampleWithReplacement` – True if the classifier is set to use sampling with replacement.

• **setSeed**

```
public void setSeed(int seed)
```

- **Description**
  Sets the seed value.
- **Parameters**
  * `seed` – The seed value.

• **setThreshold**

```
public void setThreshold(double threshold)
```

- **Description**
  Sets the value of the threshold.
- **Parameters**
  * `threshold` – The value of the threshold.

• **setUseConfidences**

```
public void setUseConfidences(boolean useConfidences)
```

- **Description**
  Stablishes whether confidences or bipartitions are used to combine classifiers in the ensemble.
- **Parameters**
  * `useConfidences` – The value of the property.

6.1.8 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**(`miml.data.MIMLInstances` **trainingSet**) `throws java.lang.Exception`
- `public final void` **build**(`mulan.data.MultiLabelInstances` **trainingSet**) `throws java.lang.Exception`
- `protected abstract void` **buildInternal**(`miml.data.MIMLInstances` **trainingSet**) `throws java.lang.Exception`
- `protected void` **debug**(`java.lang.String` **msg**)
- `protected` **featureIndices**
- `public boolean` **getDebug**()
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**()
- `public boolean` **isUpdatable**()
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**() `throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**(`weka.core.Instance` **instance**) `throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**(`miml.data.MIMLBag` **instance**) `throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**(`boolean` **debug**)

Chapter 7

Package miml.data

Package Contents *Page*

7.1 Class MIMLBag

Class inheriting from DenseInstance to represent a MIML bag.

7.1.1 Declaration

```
public class MIMLBag
 extends weka.core.DenseInstance implements weka.core.Instance
```

7.1.2 Field summary

serialVersionUID Generated Serial version UID.

7.1.3 Constructor summary

MIMLBag(Instance) Constructor.

7.1.4 Method summary

getBagAsInstances() Gets a bag in the form of a set of instances considering just the relational information.

getInstance(int) Returns an instance of the Bag with index bagIndex.
getNumAttributesInABag() Gets the number of attributes of in the relational attribute of a Bag.
getNumAttributesWithRelational() Gets the total number of attributes of the Bag.
getNumInstances() Gets the number of instances of the Bag.
setValue(int, int, double) Sets the value of attrIndex attribute of the instanceIndex to a certain value.

7.1.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.

7.1.6 Constructors

- **MIMLBag**

```
public MIMLBag(weka.core.Instance instance) throws java.lang.
    Exception
```

  - **Description**
    Constructor.
  - **Parameters**
    * `instance` – A Weka's Instance to be transformed into a Bag.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

7.1.7 Methods

- **getBagAsInstances**

```
public weka.core.Instances getBagAsInstances() throws java.lang.
    Exception
```

  - **Description**
    Gets a bag in the form of a set of instances considering just the relational information. Neither the identifier attribute of the Bag nor label attributes are included. For instance, given the relation toy above, the output of the method is the relation bag.
    @relation toy
    @attribute id {bag1,bag2}
    @attribute bag relational
    @attribute f1 numeric

@attribute f2 numeric
@attribute f3 numeric
@end bag
@attribute label1 {0,1}
@attribute label2 {0,1}
@attribute label3 {0,1}
@attribute label4 {0,1}
@relation bag
@attribute f1 numeric
@attribute f2 numeric
@attribute f3 numeric

  - **Returns** – Instances.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **getInstance**

```
public weka.core.Instance getInstance(int bagIndex)
```

  - **Description**
    Returns an instance of the Bag with index bagIndex.
  - **Parameters**
    * `bagIndex` – The index number.
  - **Returns** – Instance.

- **getNumAttributesInABag**

```
public int getNumAttributesInABag()
```

  - **Description**
    Gets the number of attributes of in the relational attribute of a Bag. For instance, in the relation above, the output of the method is 3.
    @relation toy
    @attribute id {bag1,bag2}
    @attribute bag relational
    @attribute f1 numeric
    @attribute f2 numeric
    @attribute f3 numeric
    @end bag
    @attribute label1 {0,1}
    @attribute label2 {0,1}
    @attribute label3 {0,1}
    @attribute label4 {0,1}

- **Returns** – The number of attributes.

- **getNumAttributesWithRelational**

```
public int getNumAttributesWithRelational()
```

  - **Description**
    Gets the total number of attributes of the Bag. This number includes attributes corresponding to labels. Instead the relational attribute, the number of attributes contained in the relational attribute is considered. For instance, in the relation above, the output of the method is 8.
    @relation toy
    @attribute id {bag1,bag2}
    @attribute bag relational
    @attribute f1 numeric
    @attribute f2 numeric
    @attribute f3 numeric
    @end bag
    @attribute label1 {0,1}
    @attribute label2 {0,1}
    @attribute label3 {0,1}
    @attribute label4 {0,1}
  - **Returns** – Total number of attributes of the Bag.

- **getNumInstances**

```
public int getNumInstances()
```

  - **Description**
    Gets the number of instances of the Bag.
  - **Returns** – The number of instances of the Bag.

- **setValue**

```
public void setValue(int instanceIndex,int attrIndex,double
    value)
```

  - **Description**
    Sets the value of attrIndex attribute of the instanceIndex to a certain value.
  - **Parameters**
    * `instanceIndex` – The index of the instance.
    * `attrIndex` – The index of the attribute.
    * `value` – The value to be set.

7.1.8 Members inherited from class DenseInstance

`weka.core.DenseInstance`

- `public Object` **copy**`()`
- `protected void` **forceDeleteAttributeAt**`(int` **arg0**`)`
- `protected void` **forceInsertAttributeAt**`(int` **arg0**`)`
- `private void` **freshAttributeVector**`()`
- `public String` **getRevision**`()`
- `public int` **index**`(int` **arg0**`)`
- `public static void` **main**`(java.lang.String[]` **arg0**`)`
- `public Instance` **mergeInstance**`(Instance` **arg0**`)`
- `public int` **numAttributes**`()`
- `public int` **numValues**`()`
- `public void` **replaceMissingValues**`(double[]` **arg0**`)`
- `static final` **serialVersionUID**
- `public void` **setValue**`(int` **arg0**`, double` **arg1**`)`
- `public void` **setValueSparse**`(int` **arg0**`, double` **arg1**`)`
- `public double` **toDoubleArray**`()`
- `public String` **toStringNoWeight**`()`
- `public String` **toStringNoWeight**`(int` **arg0**`)`
- `public double` **value**`(int` **arg0**`)`

7.1.9 Members inherited from class AbstractInstance

`weka.core.AbstractInstance`

- `public Attribute` **attribute**`(int` **arg0**`)`
- `public Attribute` **attributeSparse**`(int` **arg0**`)`
- `public Attribute` **classAttribute**`()`
- `public int` **classIndex**`()`
- `public boolean` **classIsMissing**`()`
- `public double` **classValue**`()`
- `public Instances` **dataset**`()`
- `public void` **deleteAttributeAt**`(int` **arg0**`)`
- `public Enumeration` **enumerateAttributes**`()`
- `public boolean` **equalHeaders**`(Instance` **arg0**`)`
- `public String` **equalHeadersMsg**`(Instance` **arg0**`)`
- `protected abstract void` **forceDeleteAttributeAt**`(int` **arg0**`)`
- `protected abstract void` **forceInsertAttributeAt**`(int` **arg0**`)`
- `public String` **getRevision**`()`
- `public boolean` **hasMissingValue**`()`
- `public void` **insertAttributeAt**`(int` **arg0**`)`
- `public boolean` **isMissing**`(Attribute` **arg0**`)`
- `public boolean` **isMissing**`(int` **arg0**`)`
- `public boolean` **isMissingSparse**`(int` **arg0**`)`
- `protected` **m_AttValues**
- `protected` **m_Dataset**
- `protected` **m_Weight**
- `public int` **numClasses**`()`
- `public final Instances` **relationalValue**`(Attribute` **arg0**`)`
- `public final Instances` **relationalValue**`(int` **arg0**`)`
- `public static` **s_numericAfterDecimalPoint**
- `static final` **serialVersionUID**
- `public void` **setClassMissing**`()`
- `public void` **setClassValue**`(double` **arg0**`)`
- `public final void` **setClassValue**`(java.lang.String` **arg0**`)`
- `public final void` **setDataset**`(Instances` **arg0**`)`

- `public final void` **setMissing**`(Attribute` **arg0**`)`
- `public final void` **setMissing**`(int` **arg0**`)`
- `public final void` **setValue**`(Attribute` **arg0**`, double` **arg1**`)`
- `public final void` **setValue**`(Attribute` **arg0**`, java.lang.String` **arg1**`)`
- `public final void` **setValue**`(int` **arg0**`, java.lang.String` **arg1**`)`
- `public final void` **setWeight**`(double` **arg0**`)`
- `public final String` **stringValue**`(Attribute` **arg0**`)`
- `public final String` **stringValue**`(int` **arg0**`)`
- `public String` **toString**`()`
- `public final String` **toString**`(Attribute` **arg0**`)`
- `public final String` **toString**`(Attribute` **arg0**`, int` **arg1**`)`
- `public final String` **toString**`(int` **arg0**`)`
- `public final String` **toString**`(int` **arg0**`, int` **arg1**`)`
- `public final String` **toStringMaxDecimalDigits**`(int` **arg0**`)`
- `public double` **value**`(Attribute` **arg0**`)`
- `public double` **valueSparse**`(int` **arg0**`)`
- `public final double` **weight**`()`

7.2 Class MIMLInstances

Class inheriting from MultiLabelnstances to represent MIML data.

7.2.1 Declaration

```
public class MIMLInstances
 extends mulan.data.MultiLabelInstances
```

7.2.2 Field summary

serialVersionUID Generated Serial version UID.

7.2.3 Constructor summary

MIMLInstances(Instances, LabelsMetaData) Constructor.
MIMLInstances(Instances, String) Constructor.
MIMLInstances(MIMLInstances) Constructor.
MIMLInstances(MultiLabelInstances) Constructor.
MIMLInstances(String, int) Constructor.
MIMLInstances(String, String) Constructor.

7.2.4 Method summary

addBag(MIMLBag) Adds a Bag of Instances to the dataset.
addInstance(MIMLBag, int) Adds a Bag of Instances to the dataset in a certain index.
getBag(int) Gets a `MIMLBag` (in 7.1, page 96) (i.e. pattern) with a certain bagIndex.
getBagAsInstances(int) Gets a `MIMLBag` (in 7.1, page 96) with a certain bagIndex in the form of a set of `Instances` considering just the relational information.
getInstance(int, int) Gets an instance of a bag.
getMLDataSet() Returns the dataset as MultiLabelInstances.
getNumAttributes() Gets the number of attributes of the dataset considering label attributes and the relational attribute with bags as a single attribute.
getNumAttributesInABag() Gets the number of attributes per bag.
getNumAttributesWithRelational() Gets the total number of attributes of the dataset.
getNumBags() Gets the number of bags of the dataset.
getNumInstances(int) Gets the number of instances of a bag.
insertAttributesToBags(ArrayList) Adds a set of attributes to the relational attribute with values '?'
insertAttributeToBags(Attribute) Adds an attribute to the relational attribute with value '?'
roundsCV(MIMLInstances, int, int, int) Generate tran/test partitions for CV cross validation.
splitData(MIMLInstances, double, int, int) Split MIML data train and test partition given a percentage and a partitioning method.

7.2.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.

7.2.6 Constructors

- **MIMLInstances**

```
public MIMLInstances(weka.core.Instances dataSet,mulan.data.
    LabelsMetaData labelsMetaData) throws mulan.data.
    InvalidDataFormatException
```

  - **Description**
    Constructor.
  - **Parameters**
    - `dataSet` – A dataset of `Instances` with relational information.
    - `labelsMetaData` – Information about labels.
  - **Throws**

    - `mulan.data.InvalidDataFormatException` – To be handled in an upper level.

- **MIMLInstances**

```
public MIMLInstances(weka.core.Instances dataSet,java.lang.
    String xmlLabelsDefFilePath) throws mulan.data.
    InvalidDataFormatException
```

  - **Description**
    Constructor.
  - **Parameters**
    - `dataSet` – A dataset of `Instances` with relational information.
    - `xmlLabelsDefFilePath` – Path of .xml file with information about labels.
  - **Throws**
    - `mulan.data.InvalidDataFormatException` – To be handled in an upper level.

- **MIMLInstances**

```
public MIMLInstances(MIMLInstances mimlDataSet) throws mulan.
    data.InvalidDataFormatException
```

  - **Description**
    Constructor.
  - **Parameters**
    - `mimlDataSet` – A datasetof `MIMLInstances` (in 7.2, page 101).
  - **Throws**
    - `mulan.data.InvalidDataFormatException` – To be handled in an upper level.

- **MIMLInstances**

```
public MIMLInstances(mulan.data.MultiLabelInstances mlDataSet)
    throws mulan.data.InvalidDataFormatException
```

  - **Description**
    Constructor.
  - **Parameters**
    - `mlDataSet` – A multi-label datasetof .
  - **Throws**
    - `mulan.data.InvalidDataFormatException` – To be handled in an upper level.

- **MIMLInstances**

```
public MIMLInstances(java.lang.String arffFilePath ,int
    numLabelAttributes) throws mulan.data.
    InvalidDataFormatException
```

- **Description**
  Constructor.
- **Parameters**
  * `arffFilePath` – Path of .arff file with Instances.
  * `numLabelAttributes` – Number of label attributes.
- **Throws**
  * `mulan.data.InvalidDataFormatException` – To be handled in an upper level.

• **MIMLInstances**

```
public MIMLInstances(java.lang.String arffFilePath ,java.lang.
    String xmlLabelsDefFilePath) throws mulan.data.
    InvalidDataFormatException
```

- **Description**
  Constructor.
- **Parameters**
  * `arffFilePath` – Path of .arff file with Instances.
  * `xmlLabelsDefFilePath` – Path of .xml file with information about labels.
- **Throws**
  * `mulan.data.InvalidDataFormatException` – To be handled in an upper level.

7.2.7 Methods

• **addBag**

```
public void addBag(MIMLBag bag)
```

- **Description**
  Adds a Bag of Instances to the dataset.
- **Parameters**
  * `bag` – A Bag of Instances.

• **addInstance**

```
public void addInstance(MIMLBag bag ,int index)
```

- **Description**
  Adds a Bag of Instances to the dataset in a certain index.
- **Parameters**
  * `bag` – A Bag of Instances.
  * `index` – The index to insert the Bag.

• **getBag**

```
public MIMLBag getBag(int bagIndex) throws java.lang.Exception
```

- **Description**
  Gets a `MIMLBag` (in 7.1, page 96) (i.e. pattern) with a certain bagIndex.
- **Parameters**
  * `bagIndex` – Index of the bag.
- **Returns** – Bag If bagIndex exceeds the number of bags in the dataset. To be handled in an upper level.
- **Throws**
  * `java.lang.Exception` – To be handled in an upper level.

• **getBagAsInstances**

```
public weka.core.Instances getBagAsInstances(int bagIndex)
    throws java.lang.Exception
```

- **Description**
  Gets a `MIMLBag` (in 7.1, page 96) with a certain bagIndex in the form of a set of `Instances` considering just the relational information. Neither identification attribute of the Bag nor label attributes are included.
- **Parameters**
  * `bagIndex` – Index of the bag.
- **Returns** – A bag or an instance from the index of the dataset.
- **Throws**
  * `java.lang.Exception` – If bagIndex exceeds the number of bags in the dataset. To be handled in an upper level.

• **getInstance**

```
public weka.core.Instance getInstance(int bagIndex,int
    instanceIndex) throws java.lang.IndexOutOfBoundsException
```

- **Description**
  Gets an instance of a bag.

  - **Parameters**
    - `bagIndex` – The index of the bag in the data set.
    - `instanceIndex` – Is the index of the instance in the bag.
  - **Returns** – Instance.
  - **Throws**
    - `java.lang.IndexOutOfBoundsException` – To be handled in an upper level.

- **getMLDataSet**

  ```
  public mulan.data.MultiLabelInstances getMLDataSet()
  ```

  - **Description**
    Returns the dataset as MultiLabelInstances.
  - **Returns** – MultiLabelInstances.

- **getNumAttributes**

  ```
  public int getNumAttributes()
  ```

  - **Description**
    Gets the number of attributes of the dataset considering label attributes and the relational attribute with bags as a single attribute. For instance, in relation above, the returned value is 6. @relation toy
    @attribute id {bag1,bag2}
    @attribute bag relational
    @attribute f1 numeric
    @attribute f2 numeric
    @attribute f3 numeric
    @end bag
    @attribute label1 {0,1}
    @attribute label2 {0,1}
    @attribute label3 {0,1}
    @attribute label4 {0,1}
  - **Returns** – The number of attributes of the dataset.

- **getNumAttributesInABag**

  ```
  public int getNumAttributesInABag()
  ```

  - **Description**
    Gets the number of attributes per bag. In MIML all bags have the same number of attributes.* For instance, in the relation above, the output of the method is 3.

```
@relation toy
@attribute id {bag1,bag2}
@attribute bag relational
@attribute f1 numeric
@attribute f2 numeric
@attribute f3 numeric
@end bag
@attribute label1 {0,1}
@attribute label2 {0,1}
@attribute label3 {0,1}
@attribute label4 {0,1}
```

- **Returns** – The number of attributes per bag.

- **getNumAttributesWithRelational**

  ```
  public int getNumAttributesWithRelational()
  ```

  - **Description**
    Gets the total number of attributes of the dataset. This number includes attributes corresponding to labels. Instead the relational attribute, the number of attributes contained in the relational attribute is considered. For instance, in the relation above, the output of the method is 8.
    ```
    @relation toy
    @attribute id {bag1,bag2}
    @attribute bag relational
    @attribute f1 numeric
    @attribute f2 numeric
    @attribute f3 numeric
    @end bag
    @attribute label1 {0,1}
    @attribute label2 {0,1}
    @attribute label3 {0,1}
    @attribute label4 {0,1}
    ```
  - **Returns** – The total number of attributes of the dataset.

- **getNumBags**

  ```
  public int getNumBags()
  ```

  - **Description**
    Gets the number of bags of the dataset.
  - **Returns** – The number of bags of the dataset.

- **getNumInstances**

  ```
  public int getNumInstances(int bagIndex) throws java.lang.
      Exception
  ```

  - **Description**
    Gets the number of instances of a bag.
  - **Parameters**
    * `bagIndex` – A bag index.
  - **Returns** – The number of instances of a bag
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **insertAttributesToBags**

  ```
  public MIMLInstances insertAttributesToBags(java.util.ArrayList
      Attributes) throws mulan.data.InvalidDataFormatException
  ```

  - **Description**
    Adds a set of attributes to the relational attribute with values '?' at the last position of the relational attribute.
  - **Parameters**
    * `Attributes` – ArrayList of attributes to add.
  - **Returns** – new dataset.
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – if occurred an error creating new dataset.

- **insertAttributeToBags**

  ```
  public MIMLInstances insertAttributeToBags(weka.core.Attribute
      newAttr) throws mulan.data.InvalidDataFormatException
  ```

  - **Description**
    Adds an attribute to the relational attribute with value '?' at the last position.
  - **Parameters**
    * `newAttr` – The attribute to be added.
  - **Returns** – new dataset.
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – if occurred an error creating new dataset.

- **roundsCV**

```
public static MIMLInstances[][] roundsCV(MIMLInstances
    mimlDataSet, int nFolds, int seed, int partitioningMethod)
    throws java.lang.Exception
```

  - **Description**
    Generate tran/test partitions for CV cross validation.
  - **Parameters**
    * `mimlDataSet` – The MIML dataset to be splited.
    * `nFolds` – The number of folds.
    * `seed` – Seed use to randomize.
    * `partitioningMethod` – An integer with the partitioning method:
      · 1 random partitioning
      · 2 powerset partitioning
      · 3 iterative partitioning
  - **Returns** – MIMLInstances[][] a nfolds x 2 matrix. Each row represents a fold being column 0 the train set and the column 1 the test set.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **splitData**

```
public static MIMLInstances[] splitData(MIMLInstances
    mimlDataSet, double percentageTrain, int seed, int
    partitioningMethod) throws java.lang.Exception
```

  - **Description**
    Split MIML data train and test partition given a percentage and a partitioning method.
  - **Parameters**
    * `mimlDataSet` – The MIML dataset to be splited.
    * `percentageTrain` – The percentage (0-100) to be used for train.
    * `seed` – Seed use to randomize.
    * `partitioningMethod` – An integer with the partitioning method:
      · 1 random partitioning
      · 2 powerset partitioning
      · 3 iterative partitioning
  - **Returns** – A list with the dataset splited.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

7.2.8 Members inherited from class MultiLabelInstances

`mulan.data.MultiLabelInstances`

- `private boolean` **checkLabelAttributeFormat**`(weka.core.Attribute` **arg0**`)`
- `private void` **checkLabelsConsistency**`(weka.core.Instances` **arg0**`, java.util.Set` **arg1**`) throws InvalidDataFormatException`
- `private void` **checkSubtreeConsistency**`(LabelNode` **arg0**`, weka.core.Instance` **arg1**`, boolean` **arg2**`, java.util.Map` **arg3**`) throws InvalidDataFormatException`
- `public MultiLabelInstances` **clone**`()`
- `private` **dataSet**
- `public double` **getCardinality**`()`
- `public Instances` **getDataSet**`()`
- `public int` **getDepth**`(java.lang.String` **arg0**`)`
- `public Set` **getFeatureAttributes**`()`
- `public int` **getFeatureIndices**`()`
- `public Set` **getLabelAttributes**`()`
- `public HashMap` **getLabelDepth**`()`
- `public int` **getLabelDepthIndices**`()`
- `public int` **getLabelIndices**`()`
- `public String` **getLabelNames**`()`
- `public LabelsMetaData` **getLabelsMetaData**`()`
- `public Map` **getLabelsOrder**`()`
- `public Instance` **getNextInstance**`() throws java.io.IOException`
- `public int` **getNumInstances**`()`
- `public int` **getNumLabels**`()`
- `public boolean` **hasMissingLabels**`(weka.core.Instance` **arg0**`)`
- `private boolean` **isLabelSet**`(weka.core.Instance` **arg0**`, java.lang.String` **arg1**`, java.util.Map` **arg2**`)`
- `private final` **labelsMetaData**
- `private` **loader**
- `private Instances` **loadInstances**`(java.io.File` **arg0**`)`
- `private Instances` **loadInstances**`(java.io.InputStream` **arg0**`)`
- `private LabelsMetaData` **loadLabelsMeta**`(java.io.InputStream` **arg0**`)`
- `private LabelsMetaData` **loadLabelsMeta**`(java.lang.String` **arg0**`)`
- `private LabelsMetaData` **loadLabesMeta**`(weka.core.Instances` **arg0**`, int` **arg1**`, boolean` **arg2**`) throws InvalidDataFormatException`
- `public MultiLabelInstances` **reintegrateModifiedDataSet**`(weka.core.Instances` **arg0**`) throws InvalidDataFormatException`
- `private void` **validate**`(weka.core.Instances` **arg0**`, LabelsMetaData` **arg1**`) throws InvalidDataFormatException`

7.3 Class MLSave

Class with methods to write to file a multi-label dataset. MIML format is also supported.

7.3.1 Declaration

```
public final class MLSave
 extends java.lang.Object
```

7.3.2 Constructor summary

MLSave()

7.3.3 Method summary

saveArff(Instances, String) Writes an arff file with an Instances dataset.
saveArff(MIMLInstances, String) Writes an arff file with a multi-label dataset.
saveArff(MultiLabelInstances, String) Writes an arff file with a multi-label dataset.
saveXml(ArrayList, String) Writes an xml file.
saveXml(Instances, String) Writes an xml file with label definitions of an instances dataset.
saveXml(MultiLabelInstances, String) Writes an xml file with label definitions of a multi-label dataset.

7.3.4 Constructors

- **MLSave**

```
private MLSave()
```

7.3.5 Methods

- **saveArff**

```
public static void saveArff(weka.core.Instances instances,java.
    lang.String pathName) throws java.io.IOException
```

  - **Description**
    Writes an arff file with an Instances dataset.
  - **Parameters**
    * `instances` – A dataset.
    * `pathName` – Name and path for file to write.
  - **Throws**
    * `java.io.IOException` – To be handled in an upper level.

- **saveArff**

```
public static void saveArff(MIMLInstances instances,java.lang.
    String pathName) throws java.io.IOException
```

  - **Description**
    Writes an arff file with a multi-label dataset. MIML format is also supported.

- **Parameters**
  - `instances` – A multi-label dataset.
  - `pathName` – Name and path for file to write.
- **Throws**
  - `java.io.IOException` – To be handled in an upper level.

• **saveArff**

```
public static void saveArff(mulan.data.MultiLabelInstances
    instances,java.lang.String pathName) throws java.io.
    IOException
```

- **Description**
  Writes an arff file with a multi-label dataset. MIML format is also supported.
- **Parameters**
  - `instances` – A multi-label dataset.
  - `pathName` – Name and path for file to write.
- **Throws**
  - `java.io.IOException` – To be handled in an upper level.

• **saveXml**

```
public static void saveXml(java.util.ArrayList labelNames,java.
    lang.String pathName)
```

- **Description**
  Writes an xml file.
- **Parameters**
  - `labelNames` – An ArrayList<String>with label names.
  - `pathName` – Name and path for file to write.

• **saveXml**

```
public static void saveXml(weka.core.Instances instances,java.
    lang.String pathName) throws java.io.IOException, mulan.data.
    LabelsBuilderException
```

- **Description**
  Writes an xml file with label definitions of an instances dataset.
- **Parameters**
  - `instances` – A dataset.

    * `pathName` – Name and path for file to write.
  - **Throws**
    * `java.io.IOException` – To be handled in an upper level.
    * `mulan.data.LabelsBuilderException` – To be handled in an upper level.

- **saveXml**

```
public static void saveXml(mulan.data.MultiLabelInstances
    instances,java.lang.String pathName) throws java.io.
    IOException, mulan.data.LabelsBuilderException
```

  - **Description**
    Writes an xml file with label definitions of a multi-label dataset. MIML format is also supported.
  - **Parameters**
    * `instances` – A multi-label dataset.
    * `pathName` – Name and path for file to write.
  - **Throws**
    * `java.io.IOException` – To be handled in an upper level.
    * `mulan.data.LabelsBuilderException` – To be handled in an upper level.

7.4 Class MWTranslator

Class to serve as interface between MIMLInstances and Matlab data types.

7.4.1 Declaration

```
public class MWTranslator
 extends java.lang.Object
```

7.4.2 Field summary

attributesPerBag Number of attributes per bag
labelIndices Array with the attribute indices corresponding to the labels
mimlDataSet A MIML dataset.
nBags Number of bags of the dataset
nLabels Number of labels of the dataset

7.4.3 Constructor summary

MWTranslator(MIMLInstances) Constructor.

7.4.4 Method summary

getBagAsArray(int) Returns a bag in the format of a nInstxnAttributes array of double.

getBagAsArray(MIMLBag) Returns a MIMLBag in the format of a nInstxnAttributes MWNumericArray of double.

getBagAsCell(int) Returns a MIMLBag in the format of a 1x1 MWCellArray in which the bag is stored in CellArray{1,1} as an nInstxnAttributes array of double.

getBagAsCell(MIMLBag) Returns a MIMLBag in the format of a 1x1 MWCellArray in which the bag is stored in CellArray{1,1} as an nInstxnAttributes array of double.

getBags() Returns all the bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}.

getLabels() Returns label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double.

getLabels(int) Returns label associations of a MIMLbag in the format of a nLabelsx1 MWNumericArray of double.

getLabels(MIMLBag) Returns label associations of a MIMLbag in the format of a nLabelsx1 MWNumericArray of double.

7.4.5 Fields

- `MIMLInstances` **mimlDataSet**
  - A MIML dataset.
- `int` **nBags**
  - Number of bags of the dataset
- `int` **nLabels**
  - Number of labels of the dataset
- `int` **attributesPerBag**
  - Number of attributes per bag
- `int[]` **labelIndices**
  - Array with the attribute indices corresponding to the labels

7.4.6 Constructors

- **MWTranslator**

```
public MWTranslator(MIMLInstances mimlDataSet)
```

  - **Description**
    Constructor.
  - **Parameters**
    * `mimlDataSet` – A MIML dataset.

7.4.7 Methods

- **getBagAsArray**

```
public com.mathworks.toolbox.javabuilder.MWNumericArray
    getBagAsArray(int index) throws java.lang.Exception
```

  - **Description**
    Returns a bag in the format of a nInstxnAttributes array of double.
  - **Parameters**
    * `index` – The index of the bag in the MIMLInstances dataset.
  - **Returns** – A MIMLBag
  - **Throws**
    * `java.lang.Exception` – To be handled.

- **getBagAsArray**

```
public com.mathworks.toolbox.javabuilder.MWNumericArray
    getBagAsArray(MIMLBag bag) throws java.lang.Exception
```

  - **Description**
    Returns a MIMLBag in the format of a nInstxnAttributes MWNumericArray of double.
  - **Parameters**
    * `bag` – A MIMLBag
  - **Returns** – Returns a MIMLBag in the format of a nInstxnAttributes MWNumericArray of double.
  - **Throws**
    * `java.lang.Exception` – To be handled.

- **getBagAsCell**

```
public com.mathworks.toolbox.javabuilder.MWCellArray
    getBagAsCell(int index) throws java.lang.Exception
```

  - **Description**
    Returns a MIMLBag in the format of a 1x1 MWCellArray in which the bag is stored in CellArray{1,1} as an nInstxnAttributes array of double.
  - **Parameters**
    * `index` – The index of the bag in the MIMLInstances dataset.
  - **Returns** – Returns a MIMLBag in the format of a 1x1 MWCellArray in which the bag is stored in CellArray{1,1} as an nInstxnAttributes array of double.

- **Throws**
  - `java.lang.Exception` – To be handled.

- **getBagAsCell**

```
public com.mathworks.toolbox.javabuilder.MWCellArray
    getBagAsCell(MIMLBag bag) throws java.lang.Exception
```

  - **Description**
    Returns a MIMLBag in the format of a 1x1 MWCellArray in which the bag is stored in CellArray{1,1} as an nInstxnAttributes array of double.
  - **Parameters**
    - `bag` – A MIMLBag.
  - **Returns** – Returns a MIMLBag in the format of a 1x1 MWCellArray in which the bag is stored in CellArray{1,1} as an nInstxnAttributes array of double.
  - **Throws**
    - `java.lang.Exception` – To be handled.

- **getBags**

```
public com.mathworks.toolbox.javabuilder.MWCellArray getBags()
    throws java.lang.Exception
```

  - **Description**
    Returns all the bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
  - **Returns** – Returns all the bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
  - **Throws**
    - `java.lang.Exception` – To be handled.

- **getLabels**

```
public com.mathworks.toolbox.javabuilder.MWNumericArray
    getLabels() throws java.lang.Exception
```

  - **Description**
    Returns label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.

  - **Returns** – Returns label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
  - **Throws**
    - `java.lang.Exception` – To be handled.

- **getLabels**

```
public com.mathworks.toolbox.javabuilder.MWNumericArray
    getLabels(int index) throws java.lang.Exception
```

  - **Description**
    Returns label associations of a MIMLbag in the format of a nLabelsx1 MWNumericArray of double. If the bag belongs to the jth label, then aDoubleArray(j) equals +1, otherwise aDoubleArray(j,1) equals -1.
  - **Parameters**
    - `index` – The index of the bag in the MIMLInstances dataset.
  - **Returns** – label associations of a bag in the format of a nLabelsx1 MWNumericArray of double.
  - **Throws**
    - `java.lang.Exception` – To be handled.

- **getLabels**

```
public com.mathworks.toolbox.javabuilder.MWNumericArray
    getLabels(MIMLBag bag) throws java.lang.Exception
```

  - **Description**
    Returns label associations of a MIMLbag in the format of a nLabelsx1 MWNumericArray of double. If the bag belongs to the jth label, then aDoubleArray(j,1) equals +1, otherwise aDoubleArray(j,1) equals -1.
  - **Parameters**
    - `bag` – A MIMLBag.
  - **Returns** – label associations of a bag in the format of a nLabelsx1 MWNumericArray of double.
  - **Throws**
    - `java.lang.Exception` – To be handled.

Chapter 8

Package miml.evaluation

Package Contents *Page*

8.1 Interface IEvaluator

Interface for run and evaluate a experiment.

8.1.1 Declaration

```
public interface IEvaluator
```

8.1.2 All known subinterfaces

EvaluatorHoldoutClus (in 8.4, page 128), EvaluatorHoldout (in 8.3, page 124), EvaluatorCV (in 8.2, page 119)

8.1.3 All classes known to implement interface

EvaluatorHoldout (in 8.3, page 124), EvaluatorCV (in 8.2, page 119)

8.1.4 Method summary

getEvaluation() Gets the evaluation generated by the experiment.
runExperiment(IMIMLClassifier) Run an experiment.

8.1.5 Methods

- **getEvaluation**

  ```
  java.lang.Object getEvaluation()
  ```

  - **Description**
    Gets the evaluation generated by the experiment.
  - **Returns** – The evaluation.

- **runExperiment**

  ```
  void runExperiment(miml.classifiers.miml.IMIMLClassifier
      classifier) throws java.lang.Exception
  ```

  - **Description**
    Run an experiment.
  - **Parameters**
    * `classifier` – The classifier used in the experiment.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

8.2 Class EvaluatorCV

Class that allow evaluate an algorithm applying a cross-validation method with random partitioning. This class uses weka.core.Instances.trainCV and weka.core.Instances.testCV so there is not guarantee of having examples of all labels in the partitioned data.

8.2.1 Declaration

```
public class EvaluatorCV
 extends java.lang.Object implements miml.core.IConfiguration ,
     IEvaluator
```

8.2.2 Field summary

data The data used in the experiment.
multipleEvaluation The evaluation method used in cross-validation.
numFolds The number of folds.
seed The seed for the partition.
testTime Test time in milliseconds.
trainTime Train time in milliseconds.

8.2.3 Constructor summary

EvaluatorCV() No-argument constructor for xml configuration.
EvaluatorCV(MIMLInstances, int) Instantiates a new CV evaluator.

8.2.4 Method summary

configure(Configuration)
getAvgTestTime() Gets the average time of all folds in test.
getAvgTrainTime() Gets the average time of all folds in train.
getData() Gets the data used in the experiment.
getEvaluation()
getNumFolds() Gets the number of folds used in the experiment.
getSeed() Gets the seed used in the experiment.
getStdTestTime() Gets the standard deviation time of all folds in test.
getStdTrainTime() Gets the standard deviation time of all folds in train.
getTestTime() Gets the time spent in testing in each fold.
getTrainTime() Gets the time spent in training in each fold.
meanArray(long[]) Calculate the mean of given array.
runExperiment(IMIMLClassifier)
setNumFolds(int) Sets the number of folds used in the experiment.
setSeed(int) Sets the seed used in the experiment.
stdArray(long[]) Calculate the standard deviation of given array.

8.2.5 Fields

- `protected mulan.evaluation.MultipleEvaluation` **multipleEvaluation**
  - The evaluation method used in cross-validation.
- `protected miml.data.MIMLInstances` **data**
  - The data used in the experiment.
- `protected int` **numFolds**
  - The number of folds.
- `protected int` **seed**
  - The seed for the partition.
- `protected long[]` **trainTime**

  - Train time in milliseconds.

- `protected long[]` **testTime**
  - Test time in milliseconds.

8.2.6 Constructors

- **EvaluatorCV**

  ```
  public EvaluatorCV()
  ```

  - **Description**
    No-argument constructor for xml configuration.

- **EvaluatorCV**

  ```
  public EvaluatorCV(miml.data.MIMLInstances data, int numFolds)
  ```

  - **Description**
    Instantiates a new CV evaluator.
  - **Parameters**
    - `data` – The data used in the experiment.
    - `numFolds` – The number of folds used in the cross-validation.

8.2.7 Methods

- **configure**

  ```
  void configure(org.apache.commons.configuration2.Configuration
      configuration)
  ```

  - **Description copied from miml.core.IConfiguration** (**in** 1.1, **page** 18)
    Method to configure the class with the given configuration.
  - **Parameters**
    - `configuration` – Configuration used to configure the class.

- **getAvgTestTime**

  ```
  public double getAvgTestTime()
  ```

  - **Description**
    Gets the average time of all folds in test.
  - **Returns** – The average time of all folds.

- **getAvgTrainTime**

  ```
  public double getAvgTrainTime()
  ```

  - **Description**
    Gets the average time of all folds in train.
  - **Returns** – The average time of all folds.

- **getData**

  ```
  public miml.data.MIMLInstances getData()
  ```

  - **Description**
    Gets the data used in the experiment.
  - **Returns** – The data.

- **getEvaluation**

  ```
  java.lang.Object getEvaluation()
  ```

  - **Description copied from IEvaluator (in 8.1, page 118)**
    Gets the evaluation generated by the experiment.
  - **Returns** – The evaluation.

- **getNumFolds**

  ```
  public int getNumFolds()
  ```

  - **Description**
    Gets the number of folds used in the experiment.
  - **Returns** – The number of folds.

- **getSeed**

  ```
  public int getSeed()
  ```

  - **Description**
    Gets the seed used in the experiment.
  - **Returns** – The seed.

- **getStdTestTime**

```
public double getStdTestTime()
```

- **Description**
  Gets the standard deviation time of all folds in test.
- **Returns** – The standard deviation time of all folds.

- **getStdTrainTime**

```
public double getStdTrainTime()
```

- **Description**
  Gets the standard deviation time of all folds in train.
- **Returns** – The standard deviation time of all folds.

- **getTestTime**

```
public long[] getTestTime()
```

- **Description**
  Gets the time spent in testing in each fold.
- **Returns** – The test time.

- **getTrainTime**

```
public long[] getTrainTime()
```

- **Description**
  Gets the time spent in training in each fold.
- **Returns** – The train time.

- **meanArray**

```
protected double meanArray(long[] array)
```

- **Description**
  Calculate the mean of given array.
- **Parameters**
  * `array` – The array with long values.
- **Returns** – The mean of all array's values.

- **runExperiment**

```
void runExperiment(miml.classifiers.miml.IMIMLClassifier
    classifier) throws java.lang.Exception
```

- **Description copied from IEvaluator (in 8.1, page 118)**
  Run an experiment.
- **Parameters**
  * `classifier` – The classifier used in the experiment.
- **Throws**
  * `java.lang.Exception` – To be handled in an upper level.

• **setNumFolds**

```
public void setNumFolds(int numFolds)
```

- **Description**
  Sets the number of folds used in the experiment.
- **Parameters**
  * `numFolds` – The new number of folds.

• **setSeed**

```
public void setSeed(int seed)
```

- **Description**
  Sets the seed used in the experiment.
- **Parameters**
  * `seed` – The new seed

• **stdArray**

```
protected double stdArray(long[] array)
```

- **Description**
  Calculate the standard deviation of given array.
- **Parameters**
  * `array` – the array with long values.
- **Returns** – The standard deviation of all array's values.

8.3 Class EvaluatorHoldout

Class that allow evaluate an algorithm applying a holdout method.

8.3.1 Declaration

```
public class EvaluatorHoldout
 extends java.lang.Object implements miml.core.IConfiguration,
    IEvaluator
```

8.3.2 All known subclasses

EvaluatorHoldoutClus (in 8.4, page 128)

8.3.3 Field summary

evaluation The evaluation method used in holdout.
testData The test data used in the experiment.
testTime Test time in milliseconds.
trainData The data used in the experiment.
trainTime Train time in milliseconds.

8.3.4 Constructor summary

EvaluatorHoldout() No-argument constructor for xml configuration.
EvaluatorHoldout(MIMLInstances, double) Instantiates a new holdout evaluator with random partitioning method.
EvaluatorHoldout(MIMLInstances, double, int, int) Instantiates a new Holdout evaluator with a partitioning method and a seed.
EvaluatorHoldout(MIMLInstances, MIMLInstances) Instantiates a new holdout evaluator with provided train and test partitions.

8.3.5 Method summary

configure(Configuration)
getData() Gets the data used in the experiment.
getEvaluation()
getTestTime() Gets the time spent in testing.
getTrainTime() Gets the time spent in training.
runExperiment(IMIMLClassifier)

8.3.6 Fields

- `protected mulan.evaluation.Evaluation` **evaluation**
  - The evaluation method used in holdout.
- `protected miml.data.MIMLInstances` **trainData**
  - The data used in the experiment.
- `protected miml.data.MIMLInstances` **testData**
  - The test data used in the experiment.

- `protected long` **trainTime**
  - Train time in milliseconds.
- `protected long` **testTime**
  - Test time in milliseconds.

8.3.7 Constructors

- **EvaluatorHoldout**

```
public EvaluatorHoldout()
```

  - **Description**
    No-argument constructor for xml configuration.

- **EvaluatorHoldout**

```
public EvaluatorHoldout(miml.data.MIMLInstances mimlDataSet,
    double percentageTrain) throws java.lang.Exception
```

  - **Description**
    Instantiates a new holdout evaluator with random partitioning method.
  - **Parameters**
    - `mimlDataSet` – The dataset to be used.
    - `percentageTrain` – The percentage of train.
  - **Throws**
    - `java.lang.Exception` – If occur an error during holdout experiment.

- **EvaluatorHoldout**

```
public EvaluatorHoldout(miml.data.MIMLInstances mimlDataSet,
    double percentageTrain, int seed, int method) throws java.lang.
    Exception
```

  - **Description**
    Instantiates a new Holdout evaluator with a partitioning method and a seed.
  - **Parameters**
    - `mimlDataSet` – The dataset to be used.
    - `percentageTrain` – The percentage of train.
    - `seed` – Seed for randomization.
    - `method` – partitioning method.
      - 1 random partitioning
      - 2 powerset partitioning

    - · 3 iterative partitioning
  - **Throws**
    - ∗ `java.lang.Exception` – If occur an error during holdout experiment.

- **EvaluatorHoldout**

```
public EvaluatorHoldout(miml.data.MIMLInstances trainData,miml.
    data.MIMLInstances testData) throws mulan.data.
    InvalidDataFormatException
```

  - **Description**
    Instantiates a new holdout evaluator with provided train and test partitions.
  - **Parameters**
    - ∗ `trainData` – The train data used in the experiment.
    - ∗ `testData` – The test data used in the experiment.
  - **Throws**
    - ∗ `mulan.data.InvalidDataFormatException` – To be handled.

8.3.8 Methods

- **configure**

```
void configure(org.apache.commons.configuration2.Configuration
    configuration)
```

  - **Description copied from miml.core.IConfiguration** (**in** 1.1, **page** 18)
    Method to configure the class with the given configuration.
  - **Parameters**
    - ∗ `configuration` – Configuration used to configure the class.

- **getData**

```
public miml.data.MIMLInstances getData()
```

  - **Description**
    Gets the data used in the experiment.
  - **Returns** – The data.

- **getEvaluation**

```
java.lang.Object getEvaluation()
```

  - **Description copied from IEvaluator** (**in 8.1, page 118**)
    Gets the evaluation generated by the experiment.
  - **Returns** – The evaluation.

- **getTestTime**

  ```
  public long getTestTime()
  ```

  - **Description**
    Gets the time spent in testing.
  - **Returns** – The test time.

- **getTrainTime**

  ```
  public long getTrainTime()
  ```

  - **Description**
    Gets the time spent in training.
  - **Returns** – The train time.

- **runExperiment**

  ```
  void runExperiment(miml.classifiers.miml.IMIMLClassifier
      classifier) throws java.lang.Exception
  ```

  - **Description copied from IEvaluator** (**in 8.1, page 118**)
    Run an experiment.
  - **Parameters**
    * `classifier` – The classifier used in the experiment.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

8.4 Class EvaluatorHoldoutClus

Class that allow evaluate a classifier applying a holdout method with the clus System. NOTE that that RFPCT calls clus library that performs, in a single call, train and test steps. Therefore: 1. Train time got by miml library is not relevant. 2. Test time got by miml libraryr really computes the train and test time required by the call to clus library.

8.4.1 Declaration

```
public class EvaluatorHoldoutClus
 extends miml.evaluation.EvaluatorHoldout
```

8.4.2 Field summary

clusDatasetName The dataset name that will be used for training, test and settings files.
clusWorkingDir The directory where all temporary files needed or generated by CLUS library are written.

8.4.3 Constructor summary

EvaluatorHoldoutClus() No-argument constructor for xml configuration.
EvaluatorHoldoutClus(MIMLInstances, MIMLInstances, String, String) Instantiates a new holdout evaluator with provided train and test partitions.

8.4.4 Method summary

configure(Configuration)
prepareMeasuresClassification(MultiLabelInstances)
runExperiment(IMIMLClassifier)

8.4.5 Fields

- `protected java.lang.String` **clusWorkingDir**
  - The directory where all temporary files needed or generated by CLUS library are written.
- `protected java.lang.String` **clusDatasetName**
  - The dataset name that will be used for training, test and settings files.

8.4.6 Constructors

- **EvaluatorHoldoutClus**

```
public EvaluatorHoldoutClus()
```

  - **Description**
    No-argument constructor for xml configuration.

- **EvaluatorHoldoutClus**

```
public EvaluatorHoldoutClus(miml.data.MIMLInstances trainData,
    miml.data.MIMLInstances testData,java.lang.String
    clusWorkingDir,java.lang.String clusDatasetName) throws mulan
    .data.InvalidDataFormatException
```

  - **Description**
    Instantiates a new holdout evaluator with provided train and test partitions.

- **Parameters**
  - `trainData` – The train data used in the experiment.
  - `testData` – The test data used in the experiment.
  - `clusDatasetName` – The dataset name that will be used for training, test and settings files.
  - `clusWorkingDir` – The directory where all temporary files needed or generated by CLUS library are written.
- **Throws**
  - `mulan.data.InvalidDataFormatException` – To be handled.

8.4.7 Methods

- **configure**

```
void configure(org.apache.commons.configuration2.Configuration
    configuration)
```

  - **Description copied from miml.core.IConfiguration (in 1.1, page 18)**
    Method to configure the class with the given configuration.
  - **Parameters**
    - `configuration` – Configuration used to configure the class.

- **prepareMeasuresClassification**

```
protected java.util.List prepareMeasuresClassification(mulan.
    data.MultiLabelInstances mlTrainData)
```

- **runExperiment**

```
void runExperiment(miml.classifiers.miml.IMIMLClassifier
    classifier) throws java.lang.Exception
```

  - **Description copied from IEvaluator (in 8.1, page 118)**
    Run an experiment.
  - **Parameters**
    - `classifier` – The classifier used in the experiment.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

8.4.8 Members inherited from class EvaluatorHoldout

`miml.evaluation.EvaluatorHoldout` (in 8.3, page 124)

- `public void` **configure**`(org.apache.commons.configuration2.Configuration` **configuration**`)`
- `protected` **evaluation**
- `public MIMLInstances` **getData**`()`
- `public Evaluation` **getEvaluation**`()`
- `public long` **getTestTime**`()`
- `public long` **getTrainTime**`()`
- `public void` **runExperiment**`(miml.classifiers.miml.IMIMLClassifier` **classifier**`)`
- `protected` **testData**
- `protected` **testTime**
- `protected` **trainData**
- `protected` **trainTime**

Chapter 9

Package miml.transformation.mimlTOmi

Package Contents *Page*

9.1 Class BRTransformation

Class that uses Binary Relevance transformation to convert MIMLInstances to MIL Instances with relational attribute.

9.1.1 Declaration

```
public class BRTransformation
 extends java.lang.Object implements java.io.Serializable
```

9.1.2 Field summary

BRT Binary Relevance Transformation.
dataSet MIML dataSet.
serialVersionUID For serialization.

9.1.3 Constructor summary

BRTransformation(MIMLInstances) Constructor.

9.1.4 Method summary

transformBag(int, int) Removes all label attributes except labelToKeep.
transformBag(MIMLBag, int) Removes all label attributes except labelToKeep.
transformBag(MIMLBag, int[], int) Remove all label attributes except label at position indexToKeep.
transformBags(int) Remove all label attributes except labelToKeep.
transformBags(MIMLInstances, int[], int) Remove all label attributes except that at indexOfLabelToKeep.

9.1.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `protected mulan.transformations.BinaryRelevanceTransformation` **BRT**
  - Binary Relevance Transformation.
- `protected miml.data.MIMLInstances` **dataSet**
  - MIML dataSet.

9.1.6 Constructors

- **BRTransformation**

```
public BRTransformation(miml.data.MIMLInstances dataSet)
```

  - **Description**
    Constructor.
  - **Parameters**
    * `dataSet` – MIMLInstances dataset.

9.1.7 Methods

- **transformBag**

```
public weka.core.Instance transformBag(int bagIndex,int
    labelToKeep) throws java.lang.Exception
```

  - **Description**
    Removes all label attributes except labelToKeep.
  - **Parameters**
    * `bagIndex` – The bagIndex of the Bag to be transformed.
    * `labelToKeep` – The label to keep. A value in [0, numLabels-1].

  - **Returns** – Instance.
  - **Throws**
    * `java.lang.Exception` – To be handled in upper level.

- **transformBag**

```
public weka.core.Instance transformBag(miml.data.MIMLBag
    instance, int labelToKeep)
```

  - **Description**
    Removes all label attributes except labelToKeep.
  - **Parameters**
    * `instance` – The instance from which labels are to be removed.
    * `labelToKeep` – The label to keep. A value in [0, numLabels-1].
  - **Returns** – Instance

- **transformBag**

```
public static weka.core.Instance transformBag(miml.data.MIMLBag
    instance, int[] labelIndices, int indexToKeep)
```

  - **Description**
    Remove all label attributes except label at position indexToKeep.
  - **Parameters**
    * `instance` – The instance from which labels are to be removed.
    * `labelIndices` – Array storing, for each label its corresponding label. index.
    * `indexToKeep` – The label index to keep.
  - **Returns** – transformed Instance.

- **transformBags**

```
public weka.core.Instances transformBags(int labelToKeep) throws
     java.lang.Exception
```

  - **Description**
    Remove all label attributes except labelToKeep.
  - **Parameters**
    * `labelToKeep` – The label to keep. A value in [0, numLabels-1].
  - **Returns** – Instances.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **transformBags**

```
public static weka.core.Instances transformBags(miml.data.
    MIMLInstances dataSet ,int [] labelIndices ,int indexToKeep)
    throws java.lang.Exception
```

  - **Description**
    Remove all label attributes except that at indexOfLabelToKeep.
  - **Parameters**
    - `dataSet` – A MIMLInstances dataset.
    - `labelIndices` – Array storing, for each label its corresponding label index.
    - `indexToKeep` – The label index to keep.
  - **Returns** – Instances.
  - **Throws**
    - `java.lang.Exception` – when removal fails.

9.2 Class LPTransformation

Class that uses LabelPowerset transformation to convert MIMLInstances to MIL Instances with relational attribute.

9.2.1 Declaration

```
public class LPTransformation
 extends java.lang.Object implements java.io.Serializable
```

9.2.2 Field summary

LPT LabelPowerSetTransformation.
serialVersionUID For serialization.

9.2.3 Constructor summary

LPTransformation() Constructor.

9.2.4 Method summary

getLPT() Returns the format of the transformed instances.
transformBag(MIMLBag, int[])
transformBags(MIMLInstances)

9.2.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `protected MIMLLabelPowersetTransformation` **LPT**
  - LabelPowerSetTransformation.

9.2.6 Constructors

- **LPTransformation**

```
public LPTransformation()
```

  - **Description**
    Constructor.

9.2.7 Methods

- **getLPT**

```
public mulan.transformations.LabelPowersetTransformation getLPT
    ()
```

  - **Description**
    Returns the format of the transformed instances.
  - **Returns** – the format of the transformed instances.

- **transformBag**

```
public weka.core.Instance transformBag(miml.data.MIMLBag bag, int
    [] labelIndices) throws java.lang.Exception
```

  - **Parameters**
    - `bag` – The bag to be transformed.
    - `labelIndices` – The labels to remove.
  - **Returns** – Instance.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **transformBags**

```
public weka.core.Instances transformBags(miml.data.MIMLInstances
     dataSet) throws java.lang.Exception
```

- **Parameters**
  - `dataSet` – MIMLInstances dataSet.
- **Returns** – Instances.
- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

9.3 Class MIMLLabelPowersetTransformation

Class that uses LabelPowerset transformation to convert MIMLInstances to MIL Instances with relational attribute.

9.3.1 Declaration

```
class MIMLLabelPowersetTransformation
extends mulan.transformations.LabelPowersetTransformation
```

9.3.2 Field summary

serialVersionUID

9.3.3 Constructor summary

MIMLLabelPowersetTransformation()

9.3.4 Method summary

transformInstance(Instance, int[])

9.3.5 Fields

- `private static final long` **serialVersionUID**

9.3.6 Constructors

- **MIMLLabelPowersetTransformation**

```
MIMLLabelPowersetTransformation()
```

9.3.7 Methods

- **transformInstance**

```
public weka.core.Instance transformInstance(weka.core.Instance
    instance, int[] labelIndices) throws java.lang.Exception
```

- **Parameters**
  - `instance` – The instance to be transformed
  - `labelIndices` – The labels to remove.
- **Returns** – Transformed instance.
- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

9.3.8 Members inherited from class LabelPowersetTransformation

`mulan.transformations.LabelPowersetTransformation`

- `public Instances` **getTransformedFormat**`()`
- `private` **transformedFormat**
- `public Instance` **transformInstance**`(weka.core.Instance` **arg0**`, int[]` **arg1**`) throws java.lang.Exception`
- `public Instances` **transformInstances**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`

Chapter 10

Package miml.classifiers.miml

Package Contents *Page*

10.1 Interface IMIMLClassifier

Common interface for MIML classifiers.

10.1.1 Declaration

```
public interface IMIMLClassifier
 extends mulan.classifier.MultiLabelLearner, java.io.Serializable
```

10.1.2 All known subinterfaces

MIMLWel (in 5.4, page 81), MIMLSVM (in 5.3, page 73), MIMLFast (in 5.2, page 64), KiSar (in 5.1, page 57), MIMLBagging (in 6.1, page 89), MWClassifier (in 10.3, page 145), MIMLClassifier (in 10.2, page 141), MIMLClassifierToMI (in 13.2, page 188), MultiInstanceMultiLabelKNN (in 16.10, page 233), MIMLMAPkNN (in 16.9, page 229), MIMLkNN (in 16.8, page 222), MIMLIBLR (in 16.7, page 219), MIMLFuzzykNN (in 16.5, page 215), MIMLDGC (in 16.3, page 208), MIMLBRkNN (in 16.2, page 204), DMIMLkNN (in 16.1, page 201), MIMLClassifierToML (in 18.1, page 241), MIMLRBF (in 24.3, page 329), MIMLNN (in 24.2, page 322), EnMIMLNNmetric (in 24.1, page 315)

10.1.3 All classes known to implement interface

MIMLClassifier (in 10.2, page 141)

10.1.4 Method summary

build(MIMLInstances) Builds the learner model from specified `MIMLInstances` (in 7.2, page 101) data.
makeCopy()
makePrediction(Instance)
setDebug(boolean)

10.1.5 Methods

- **build**

```
void build(miml.data.MIMLInstances trainingSet) throws java.lang
    .Exception
```

  - **Description**
    Builds the learner model from specified `MIMLInstances` (in 7.2, page 101) data.
  - **Parameters**
    * `trainingSet` – Set of training data, upon which the learner model should be built.
  - **Throws**
    * `java.lang.Exception` – If learner model was not created successfully.

- **makeCopy**

```
mulan.classifier.MultiLabelLearner makeCopy() throws java.lang.
    Exception
```

- **makePrediction**

```
mulan.classifier.MultiLabelOutput makePrediction(weka.core.
    Instance arg0) throws java.lang.Exception, mulan.classifier.
    InvalidDataException, mulan.classifier.
    ModelInitializationException
```

- **setDebug**

```
void setDebug(boolean arg0)
```

10.2 Class MIMLClassifier

This java class is based on the mulan.data.Statistics.java class provided in the Mulan java framework for multi-label learning *Tsoumakas, G., Katakis, I., Vlahavas, I. (2010) "Mining Multi-label Data", Data Mining and Knowledge Discovery Handbook, O. Maimon, L. Rokach (Ed.), Springer, 2nd edition, 2010.* Our contribution is mainly related with providing a framework to work with MIML data.

10.2.1 Declaration

```
public abstract class MIMLClassifier
 extends java.lang.Object implements miml.core.IConfiguration ,
     IMIMLClassifier
```

10.2.2 All known subclasses

MIMLWel (in 5.4, page 81), MIMLSVM (in 5.3, page 73), MIMLFast (in 5.2, page 64), KiSar (in 5.1, page 57), MIMLBagging (in 6.1, page 89), MWClassifier (in 10.3, page 145), MIMLClassifierToMI (in 13.2, page 188), MultiInstanceMultiLabelKNN (in 16.10, page 233), MIMLMAPkNN (in 16.9, page 229), MIMLkNN (in 16.8, page 222), MIMLIBLR (in 16.7, page 219), MIMLFuzzykNN (in 16.5, page 215), MIMLDGC (in 16.3, page 208), MIMLBRkNN (in 16.2, page 204), DMIMLkNN (in 16.1, page 201), MIMLClassifierToML (in 18.1, page 241), MIMLRBF (in 24.3, page 329), MIMLNN (in 24.2, page 322), EnMIMLNNmetric (in 24.1, page 315)

10.2.3 Field summary

featureIndices An array containing the indexes of the feature attributes within the `Instances` object of the training data in increasing order.
isDebug Whether debugging is on/off.
isModelInitialized Boolean that indicate if the model has been initialized.
labelIndices An array containing the indexes of the label attributes within the `Instances` object of the training data in increasing order.
labelNames An array containing the names of the label attributes within the `Instances` object of the training data in increasing order.
numLabels The number of labels the learner can handle.
serialVersionUID Generated Serial version UID.

10.2.4 Constructor summary

MIMLClassifier()

10.2.5 Method summary

build(MIMLInstances)
build(MultiLabelInstances)
buildInternal(MIMLInstances) Learner specific implementation of building the model from `MultiLabelInstances` training data set.

debug(String) Writes the debug message string to the console output if debug for the learner is enabled.
getDebug() Get whether debugging is turned on.
isModelInitialized() Gets whether learner's model is initialized by `build(MultiLabelInstances)` .
isUpdatable()
makeCopy()
makePrediction(Instance)
makePredictionInternal(MIMLBag) Learner specific implementation for predicting on specified data based on trained model.
setDebug(boolean)

10.2.6 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.
- `protected boolean` **isModelInitialized**
  - Boolean that indicate if the model has been initialized.
- `protected int` **numLabels**
  - The number of labels the learner can handle. The number of labels is determined from the training data when learner is build.
- `protected int[]` **labelIndices**
  - An array containing the indexes of the label attributes within the `Instances` object of the training data in increasing order. The same order will be followed in the arrays of predictions given by each learner in the `MultiLabelOutput` object.
- `protected java.lang.String[]` **labelNames**
  - An array containing the names of the label attributes within the `Instances` object of the training data in increasing order. The same order will be followed in the arrays of predictions given by each learner in the `MultiLabelOutput` object.
- `protected int[]` **featureIndices**
  - An array containing the indexes of the feature attributes within the `Instances` object of the training data in increasing order.
- `private boolean` **isDebug**
  - Whether debugging is on/off.

10.2.7 Constructors

- **MIMLClassifier**

```
public MIMLClassifier()
```

10.2.8 Methods

- **build**

  ```
  void build(miml.data.MIMLInstances trainingSet) throws java.lang
      .Exception
  ```

  - **Description copied from IMIMLClassifier (in 10.1, page 139)**
    Builds the learner model from specified `MIMLInstances` (in 7.2, page 101) data.
  - **Parameters**
    * `trainingSet` – Set of training data, upon which the learner model should be built.
  - **Throws**
    * `java.lang.Exception` – If learner model was not created successfully.

- **build**

  ```
  public final void build(mulan.data.MultiLabelInstances
      trainingSet) throws java.lang.Exception
  ```

- **buildInternal**

  ```
  protected abstract void buildInternal(miml.data.MIMLInstances
      trainingSet) throws java.lang.Exception
  ```

  - **Description**
    Learner specific implementation of building the model from `MultiLabelInstances` training data set. This method is called from `build(MultiLabelInstances)` method, where behavior common across all learners is applied.
  - **Parameters**
    * `trainingSet` – The training data set.
  - **Throws**
    * `java.lang.Exception` – if learner model was not created successfully.

- **debug**

  ```
  protected void debug(java.lang.String msg)
  ```

  - **Description**
    Writes the debug message string to the console output if debug for the learner is enabled.
  - **Parameters**

    * `msg` – The debug message

- **getDebug**

```
public boolean getDebug()
```

  - **Description**
    Get whether debugging is turned on.
  - **Returns** – `True` if debugging output is on

- **isModelInitialized**

```
protected boolean isModelInitialized()
```

  - **Description**
    Gets whether learner's model is initialized by `build(MultiLabelInstances)` . This is used to check if `makePrediction(Instance)` can be processed.
  - **Returns** – `true` if the model has been initialized.

- **isUpdatable**

```
public boolean isUpdatable()
```

- **makeCopy**

```
mulan.classifier.MultiLabelLearner makeCopy() throws java.lang.
    Exception
```

- **makePrediction**

```
mulan.classifier.MultiLabelOutput makePrediction(weka.core.
    Instance arg0) throws java.lang.Exception, mulan.classifier.
    InvalidDataException, mulan.classifier.
    ModelInitializationException
```

- **makePredictionInternal**

```
protected abstract mulan.classifier.MultiLabelOutput
    makePredictionInternal(miml.data.MIMLBag instance) throws
    java.lang.Exception, mulan.classifier.InvalidDataException
```

- **Description**

  Learner specific implementation for predicting on specified data based on trained model. This method is called from `makePrediction(Instance)` which guards for model initialization and apply common handling/behavior.
- **Parameters**
  - `instance` – The data instance to predict on.
- **Returns** – The output of the learner for the given instance.
- **Throws**
  - `java.lang.Exception` – If an error occurs while making the prediction.
  - `mulan.classifier.InvalidDataException` – If specified instance data is invalid and can not be processed by the learner.

• **setDebug**

```
void setDebug(boolean arg0)
```

10.3 Class MWClassifier

Class to execute Matlab MIML classifiers.

10.3.1 Declaration

```
public abstract class MWClassifier
 extends miml.classifiers.miml.MIMLClassifier
```

10.3.2 All known subclasses

MIMLWel (in 5.4, page 81), MIMLSVM (in 5.3, page 73), MIMLFast (in 5.2, page 64), KiSar (in 5.1, page 57), MIMLRBF (in 24.3, page 329), MIMLNN (in 24.2, page 322), EnMIMLNNmetric (in 24.1, page 315)

10.3.3 Field summary

classifier It will store the trained classifier.
serialVersionUID For serialization.
wrapper Wrapper for Matlab data types.

10.3.4 Constructor summary

MWClassifier()

10.3.5 Method summary

buildInternal(MIMLInstances)
dispose() Disposes native MW classifier.
makePredictionInternal(MIMLBag)
predictMWClassifier(MWCellArray, MWNumericArray, MWNumericArray) Performs a prediction on a test bag.
trainMWClassifier(MWCellArray, MWNumericArray) Trains a Matlab classifier.

10.3.6 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `protected miml.data.MWTranslator` **wrapper**
  - Wrapper for Matlab data types.
- `protected java.lang.Object[]` **classifier**
  - It will store the trained classifier. The number of elements will be the same as elements returns the native MW classifier.

10.3.7 Constructors

- **MWClassifier**

```
public MWClassifier()
```

10.3.8 Methods

- **buildInternal**

```
protected abstract void buildInternal(miml.data.MIMLInstances
    trainingSet) throws java.lang.Exception
```

  - **Description copied from MIMLClassifier** (**in 10.2, page 141**)
    Learner specific implementation of building the model from `MultiLabelInstances` training data set. This method is called from `build(MultiLabelInstances)` method, where behavior common across all learners is applied.
  - **Parameters**
    - `trainingSet` – The training data set.
  - **Throws**
    - `java.lang.Exception` – if learner model was not created successfully.

- **dispose**

```
public abstract void dispose()
```

- **Description**
  Disposes native MW classifier. This method should be called if the classifier is not been used anymore in the program in order to free the memory that the MW classifier was using.

• **makePredictionInternal**

```
protected abstract mulan.classifier.MultiLabelOutput
    makePredictionInternal(miml.data.MIMLBag instance) throws
    java.lang.Exception, mulan.classifier.InvalidDataException
```

- **Description copied from MIMLClassifier (in 10.2, page 141)**
  Learner specific implementation for predicting on specified data based on trained model. This method is called from `makePrediction(Instance)` which guards for model initialization and apply common handling/behavior.
- **Parameters**
  * `instance` – The data instance to predict on.
- **Returns** – The output of the learner for the given instance.
- **Throws**
  * `java.lang.Exception` – If an error occurs while making the prediction.
  * `mulan.classifier.InvalidDataException` – If specified instance data is invalid and can not be processed by the learner.

• **predictMWClassifier**

```
protected abstract java.lang.Object[] predictMWClassifier(com.
    mathworks.toolbox.javabuilder.MWCellArray train_bags,com.
    mathworks.toolbox.javabuilder.MWNumericArray train_targets,
    com.mathworks.toolbox.javabuilder.MWNumericArray test_bag)
    throws com.mathworks.toolbox.javabuilder.MWException
```

- **Description**
  Performs a prediction on a test bag.
- **Parameters**
  * `train_bags` – Bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
  * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.

    * `test_bag` – A test bag. It will be a MIMLBag in the format of a nInstxnAttributes MWNumericArray of double.
  - **Returns** – An array of 2 Object:
    * Object[0] is a nLabelsx1 array of double containing the probability of the testing instance belonging to each label.
    * Object[1] is a nLabelsx1 array of double containing a bipartition being 1 if the label is relevant or -1 otherwise.
  - **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **trainMWClassifier**

```
protected abstract void trainMWClassifier(com.mathworks.toolbox.
    javabuilder.MWCellArray train_bags,com.mathworks.toolbox.
    javabuilder.MWNumericArray train_targets) throws com.
    mathworks.toolbox.javabuilder.MWException
```

  - **Description**
    Trains a Matlab classifier. Returns the classifier model in an array of Object.
  - **Parameters**
    * `train_bags` – bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
    * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
  - **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

10.3.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**

- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

Chapter 11

Package miml.classifiers.ml

Package Contents *Page*

11.1 Class MLDGC

Implementation of MLDGC (Multi-Label Data Gravitation Model) algorithm. For more information see: *Oscar Reyes, Carlos Morell, Sebastián Ventura (2016). Effective lazy learning algorithm based on a data gravitation model for multi-label learning. Information Sciences. Vol 340, issue C.*

11.1.1 Declaration

```
public class MLDGC
 extends mulan.classifier.lazy.MultiLabelKNN
```

11.1.2 Field summary

- **densities** Densities
- **elnn** Searching of neighborhood
- **extNeigh** Whether neighborhood is extended with all the neighbors with the same distance.
- **NGC** Neighborhood-based Gravitation Coefficient for each training example
- **serialVersionUID** For serialization
- **weight_max** Values used to normalize weights

weight_min
weights Weights

11.1.3 Constructor summary

MLDGC() The default constructor.
MLDGC(int) Constructor initializing the number of neighbors.
MLDGC(int, DistanceFunction) Constructor initializing the number of neighbors and the distance function.

11.1.4 Method summary

buildInternal(MultiLabelInstances)
computeWeightDensity(Instances, Instance, int) Given a neighborhood and an instance, computes neighborhood-weight and neighborhood-density.
getTechnicalInformation()
isExtNeigh() Gets the value of the property isExtNeigh.
labelDistance(Instance, Instance) Computes the label distance between two instances.
makePredictionInternal(Instance)
setExtNeigh(boolean) Sets the value of the property isExtNeigh.

11.1.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization
- `protected double[]` **NGC**
  - Neighborhood-based Gravitation Coefficient for each training example
- `protected double[]` **densities**
  - Densities
- `protected double[]` **weights**
  - Weights
- `protected MLDGC.LinearNNESearch` **elnn**
  - Searching of neighborhood
- `boolean` **extNeigh**
  - Whether neighborhood is extended with all the neighbors with the same distance. The default value is false.
- `protected double` **weight_max**
  - Values used to normalize weights
- `protected double` **weight_min**

11.1.6 Constructors

- **MLDGC**

  ```
  public MLDGC()
  ```

  - **Description**
    The default constructor. By default 10 neighbors and Euclidean distance.

- **MLDGC**

  ```
  public MLDGC(int numOfNeighbors)
  ```

  - **Description**
    Constructor initializing the number of neighbors. By default Euclidean Distance.
  - **Parameters**
    * `numOfNeighbors` – the number of neighbors

- **MLDGC**

  ```
  public MLDGC(int numOfNeighbors,weka.core.DistanceFunction dfunc
      )
  ```

  - **Description**
    Constructor initializing the number of neighbors and the distance function.
  - **Parameters**
    * `numOfNeighbors` – the number of neighbors
    * `dfunc` – distance function

11.1.7 Methods

- **buildInternal**

  ```
  protected abstract void buildInternal(mulan.data.
      MultiLabelInstances arg0) throws java.lang.Exception
  ```

- **computeWeightDensity**

  ```
  protected void computeWeightDensity(weka.core.Instances knn,weka
      .core.Instance instance,int index)
  ```

- **Description**
  Given a neighborhood and an instance, computes neighborhood-weight and neighborhood-density.
- **Parameters**
  * `knn` – The neighborhood of the instance.
  * `instance` – The instance for which weight and density are computed.
  * `index` – The index of the instance for which weight and density are computed.

• **getTechnicalInformation**

```
weka.core.TechnicalInformation getTechnicalInformation()
```

• **isExtNeigh**

```
public boolean isExtNeigh()
```

- **Description**
  Gets the value of the property isExtNeigh.
- **Returns** – the value of the property isExtNeigh.

• **labelDistance**

```
protected double labelDistance(weka.core.Instance instance1,weka
    .core.Instance instance2)
```

- **Description**
  Computes the label distance between two instances. The distance considered is the Hamming loss.
- **Parameters**
  * `instance1` – the first instance.
  * `instance2` – the second instance.
- **Returns** – the label distance between two instances.

• **makePredictionInternal**

```
protected abstract mulan.classifier.MultiLabelOutput
    makePredictionInternal(weka.core.Instance arg0) throws java.
    lang.Exception, mulan.classifier.InvalidDataException
```

• **setExtNeigh**

```
public void setExtNeigh(boolean extNeigh)
```

- **Description**

  Sets the value of the property isExtNeigh.
- **Parameters**
  - `extNeigh` – the value to be set.

11.1.8 Members inherited from class MultiLabelKNN

`mulan.classifier.lazy.MultiLabelKNN`

- `protected void` **buildInternal**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `protected` **dfunc**
- `protected` **distanceWeighting**
- `public boolean` **isUpdatable**`()`
- `protected` **lnn**
- `protected` **numOfNeighbors**
- `public void` **setDfunc**`(weka.core.DistanceFunction` **arg0**`)`
- `public void` **setDistanceWeighting**`(int` **arg0**`)`
- `protected` **train**
- `public static final` **WEIGHT_INVERSE**
- `public static final` **WEIGHT_NONE**
- `public static final` **WEIGHT_SIMILARITY**

11.1.9 Members inherited from class MultiLabelLearnerBase

`mulan.classifier.MultiLabelLearnerBase`

- `public final void` **build**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **arg0**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `public abstract TechnicalInformation` **getTechnicalInformation**`()`
- `private` **isDebug**
- `private` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public MultiLabelLearner` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **arg0**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(weka.core.Instance` **arg0**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `public void` **setDebug**`(boolean` **arg0**`)`

11.2 Class MLDGC.LinearNNESearch

11.2.1 Declaration

```
class MLDGC.LinearNNESearch
extends weka.core.neighboursearch.LinearNNSearch
```

11.2.2 Field summary

serialVersionUID For serialization

11.2.3 Constructor summary

LinearNNESearch(Instances)

11.2.4 Method summary

kNearestNeighboursIndices(Instance, int)

11.2.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization

11.2.6 Constructors

- **LinearNNESearch**

```
public LinearNNESearch(weka.core.Instances insts) throws java.
    lang.Exception
```

11.2.7 Methods

- **kNearestNeighboursIndices**

```
public int[] kNearestNeighboursIndices(weka.core.Instance target
    ,int kNN) throws java.lang.Exception
```

11.2.8 Members inherited from class LinearNNSearch

`weka.core.neighboursearch.LinearNNSearch`

- `public void` **addInstanceInfo**`(weka.core.Instance` **arg0**`)`
- `public double` **getDistances**`() throws java.lang.Exception`
- `public String` **getOptions**`()`
- `public String` **getRevision**`()`
- `public boolean` **getSkipIdentical**`()`
- `public String` **globalInfo**`()`

- public Instances **kNearestNeighbours**(weka.core.Instance **arg0**, int **arg1**) throws java.lang.Exception
- public Enumeration **listOptions**()
- protected **m_Distances**
- protected **m_SkipIdentical**
- public Instance **nearestNeighbour**(weka.core.Instance **arg0**) throws java.lang.Exception
- private static final **serialVersionUID**
- public void **setInstances**(weka.core.Instances **arg0**) throws java.lang.Exception
- public void **setOptions**(java.lang.String[] **arg0**) throws java.lang.Exception
- public void **setSkipIdentical**(boolean **arg0**)
- public String **skipIdenticalTipText**()
- public void **update**(weka.core.Instance **arg0**) throws java.lang.Exception

11.2.9 Members inherited from class NearestNeighbourSearch

weka.core.neighboursearch.NearestNeighbourSearch

- public void **addInstanceInfo**(weka.core.Instance **arg0**)
- public static void **combSort11**(double[] **arg0**, int[] **arg1**)
- public String **distanceFunctionTipText**()
- public Enumeration **enumerateMeasures**()
- public DistanceFunction **getDistanceFunction**()
- public abstract double **getDistances**() throws java.lang.Exception
- public Instances **getInstances**()
- public double **getMeasure**(java.lang.String **arg0**)
- public boolean **getMeasurePerformance**()
- public String **getOptions**()
- public PerformanceStats **getPerformanceStats**()
- public String **globalInfo**()
- public abstract Instances **kNearestNeighbours**(weka.core.Instance **arg0**, int **arg1**) throws java.lang.Exception
- public Enumeration **listOptions**()
- protected **m_DistanceFunction**
- protected **m_Instances**
- protected **m_kNN**
- protected **m_MeasurePerformance**
- protected **m_Stats**
- public String **measurePerformanceTipText**()
- public abstract Instance **nearestNeighbour**(weka.core.Instance **arg0**) throws java.lang.Exception
- protected static int **partition**(double[] **arg0**, double[] **arg1**, int **arg2**, int **arg3**)
- public static void **quickSort**(double[] **arg0**, double[] **arg1**, int **arg2**, int **arg3**)
- public void **setDistanceFunction**(weka.core.DistanceFunction **arg0**) throws java.lang.Exception
- public void **setInstances**(weka.core.Instances **arg0**) throws java.lang.Exception
- public void **setMeasurePerformance**(boolean **arg0**)
- public void **setOptions**(java.lang.String[] **arg0**) throws java.lang.Exception
- public abstract void **update**(weka.core.Instance **arg0**) throws java.lang.Exception

11.3 Class RFPCT

This class is a wrapper for RFPCT implemented in the clus library **CLUS** library.

11.3.1 Declaration

```
public class RFPCT
 extends mulan.classifier.clus.ClusWrapperClassification
```

11.3.2 Field summary

numTrees The number of random trees in the ensemble.
seed A seed for randomization.
serialVersionUID For serialization.

11.3.3 Constructor summary

RFPCT() No-argument constructor for xml configuration.
RFPCT(String) Constructor.
RFPCT(String, String, int, long) Constructor.

11.3.4 Method summary

buildInternal(MultiLabelInstances)
createSettingsFile() This method creates a CLUS settings file that corresponds to the MORF algorithm and writes it in clusWorkingDir.
getClusDatasetName() Gets the clus datasetName
getNumTrees() Returns the number of trees in the forest.
getSeed() Gets the seed used by the random generator.
setNumTrees(int) Sets the number of trees in the forest.
setSeed(long) Sets the seed used by the random generator.

11.3.5 Fields

- `private static final long` **serialVersionUID**
  – For serialization.
- `private int` **numTrees**
  – The number of random trees in the ensemble.
- `private long` **seed**
  – A seed for randomization.

11.3.6 Constructors

- **RFPCT**

  ```
  public RFPCT()
  ```

  – **Description**
  No-argument constructor for xml configuration.

- **RFPCT**

  ```
  public RFPCT(java.lang.String clusWorkingDir)
  ```

  - **Description**
    Constructor.
  - **Parameters**
    * `clusWorkingDir` – The directory where all temporary files needed or generated by CLUS library are written.

- **RFPCT**

  ```
  public RFPCT(java.lang.String clusWorkingDir,java.lang.String
      clusDatasetName,int numTrees,long seed)
  ```

  - **Description**
    Constructor.
  - **Parameters**
    * `clusWorkingDir` – The directory where all temporary files needed or generated by CLUS library are written.
    * `clusDatasetName` – The dataset name that will be used for training, test and settings files.
    * `numTrees` – the number of trees.
    * `seed` – The seed of random generator.

11.3.7 Methods

- **buildInternal**

  ```
  protected abstract void buildInternal(mulan.data.
      MultiLabelInstances arg0) throws java.lang.Exception
  ```

- **createSettingsFile**

  ```
  private void createSettingsFile() throws java.lang.Exception
  ```

  - **Description**
    This method creates a CLUS settings file that corresponds to the MORF algorithm and writes it in clusWorkingDir.
  - **Throws**
    * `java.lang.Exception` – Potential exception thrown. To be handled in an upper level.

- **getClusDatasetName**

  ```
  public java.lang.String getClusDatasetName()
  ```

  - **Description**
    Gets the clus datasetName
  - **Returns** – String

- **getNumTrees**

  ```
  public int getNumTrees()
  ```

  - **Description**
    Returns the number of trees in the forest.
  - **Returns** – int

- **getSeed**

  ```
  public long getSeed()
  ```

  - **Description**
    Gets the seed used by the random generator.
  - **Returns** – int

- **setNumTrees**

  ```
  public void setNumTrees(int numTrees)
  ```

  - **Description**
    Sets the number of trees in the forest.
  - **Parameters**
    * `numTrees` – The number of trees.

- **setSeed**

  ```
  public void setSeed(long seed)
  ```

  - **Description**
    Sets the seed used by the random generator.
  - **Parameters**
    * `seed` – The seed.

11.3.8 Members inherited from class ClusWrapperClassification

`mulan.classifier.clus.ClusWrapperClassification`

- `protected void` **buildInternal**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `protected` **clusWorkingDir**
- `protected` **datasetName**
- `public String` **getClusWorkingDir**`()`
- `public String` **getDatasetName**`()`
- `public TechnicalInformation` **getTechnicalInformation**`()`
- `protected` **isEnsemble**
- `public boolean` **isEnsemble**`()`
- `protected` **isRuleBased**
- `public boolean` **isRuleBased**`()`
- `public static void` **makeClusCompliant**`(mulan.data.MultiLabelInstances` **arg0**`, java.lang.String` **arg1**`) throws java.lang.Exception`
- `protected MultiLabelOutput` **makePredictionInternal**`(weka.core.Instance` **arg0**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `private static final` **serialVersionUID**
- `public void` **setEnsemble**`(boolean` **arg0**`)`
- `public void` **setRuleBased**`(boolean` **arg0**`)`
- `protected` **settingsFilePath**

11.3.9 Members inherited from class MultiLabelLearnerBase

`mulan.classifier.MultiLabelLearnerBase`

- `public final void` **build**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **arg0**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `public abstract TechnicalInformation` **getTechnicalInformation**`()`
- `private` **isDebug**
- `private` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public MultiLabelLearner` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **arg0**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(weka.core.Instance` **arg0**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `public void` **setDebug**`(boolean` **arg0**`)`

Chapter 12

Package miml.data.statistics

Package Contents *Page*

12.1 Class MIMLStatistics

Class with methods to obtain information about a MIML dataset. This java class is based on MLStatistic and MILStatistic.

12.1.1 Declaration

```
public class MIMLStatistics
 extends java.lang.Object
```

12.1.2 Field summary

dataSet A MIML data set
milstatistics Class with methods to obtain information about a MI dataset.
mlstatistics Class with methods to obtain information about a ML dataset.

12.1.3 Constructor summary

MIMLStatistics(MIMLInstances) Constructor.

12.1.4 Method summary

averageIR(double[]) Computes the average of any IR vector.
averageSkew(HashMap) Computes the average labelSkew.
calculateCooncurrence(MIMLInstances) This method calculates a matrix with the coocurrences of pairs of labels.
calculatePhiChi2(MIMLInstances) Calculates Phi and Chi-square correlation matrix.
cardinality() Computes the Cardinality as the average number of labels per pattern.
coocurrenceToCSV() Returns cooCurrenceMatrix in CSV representation.
coocurrenceToString() Returns cooCurrenceMatrix in textual representation.
correlationsToCSV(double[][]) Returns Phi correlations in CSV representation.
correlationsToString(double[][]) Returns Phi correlations in textual representation.
density() Computes the density as the cardinality/numLabels.
distributionBagsToCSV() Returns distributionBags in CSV representation.
distributionBagsToCSV(HashMap) Returns labelSkew in CSV representation.
distributionBagsToString() Returns distributionBags in textual representation.
distributionBagsToString(HashMap) Returns labelSkew in textual representation.
getChi2() Gets the Chi2 correlation matrix.
getDataSet() Returns the dataset used to calculate the statistics.
getPhi() Gets the Phi correlation matrix.
getPhiHistogram() Calculates a histogram of Phi correlations.
innerClassIR() Computes the innerClassIR for each label as negativePatterns/-positivePatterns.
interClassIR() Computes the interClassIR for each label positiveExamplesOfMajorityLabel/positivePatternsLabel.
labelCombCount() Returns the HashMap containing the distinct labelsets and their frequencies.
labelSetFrequency(LabelSet) Returns the frequency of a label set in the dataset.
labelSets() Returns a set with the distinct label sets of the dataset.
labelSkew() Computes the IR for each labelSet as (patterns of majorityLabelSet)/(patterns of the labelSet).
pMax() Returns pMax, the proportion of examples associated with the most frequently occurring labelset.
printPhiDiagram(double) This method prints data, useful for the visualization of Phi per dataset.
priors() Returns the prior probabilities of the labels.
pUnique() Returns proportion of unique label combinations (pPunique) value defined as the proportion of labelsets which are unique across the total number of examples.
setDataSet(MIMLInstances) Set the dataset used.
skewRatio() Computes the skewRatio as peak/base.
toCSV() Returns statistics in CSV representation.
topPhiCorrelatedLabels(int, int) Returns the indices of the labels that have the

strongest Phi correlation with the label which is given as a parameter.

toString() Returns statistics in textual representation.

uncorrelatedLabels(int, double) Returns the indices of the labels whose Phi coefficient values lie between -bound <= phi <= bound.

varianceIR(double[]) Computes the variance of any IR vector.

12.1.5 Fields

- `miml.data.MIMLInstances` **dataSet**
  - A MIML data set
- `protected MIStatistics` **milstatistics**
  - Class with methods to obtain information about a MI dataset.
  - See also
    - `MIStatistics` (in 12.2, page 172)
- `protected MLStatistics` **mlstatistics**
  - Class with methods to obtain information about a ML dataset.
  - See also
    - `MLStatistics` (in 12.3, page 174)

12.1.6 Constructors

- **MIMLStatistics**

```
public MIMLStatistics(miml.data.MIMLInstances dataSet)
```

  - **Description**
    Constructor.
  - **Parameters**
    - `dataSet` – A MIML data set.

12.1.7 Methods

- **averageIR**

```
public double averageIR(double[] IR)
```

  - **Description**
    Computes the average of any IR vector.
  - **Parameters**
    - `IR` – An IR vector previously computed
  - **Returns** – double

- **averageSkew**

  ```
  public double averageSkew(java.util.HashMap skew)
  ```

  - **Description**
    Computes the average labelSkew.
  - **Parameters**
    - `skew` – The IR for each labelSet previously computed.
  - **Returns** – Average labelSkew.

- **calculateCooncurrence**

  ```
  public double[][] calculateCooncurrence(miml.data.MIMLInstances
      mlDataSet)
  ```

  - **Description**
    This method calculates a matrix with the coocurrences of pairs of labels. It requires the method calculateStats to be previously called.
  - **Parameters**
    - `mlDataSet` – A multi-label dataset.
  - **Returns** – A coocurrences matrix of pairs of labels.

- **calculatePhiChi2**

  ```
  public void calculatePhiChi2(miml.data.MIMLInstances dataSet)
      throws java.lang.Exception
  ```

  - **Description**
    Calculates Phi and Chi-square correlation matrix.
  - **Parameters**
    - `dataSet` – A multi-label dataset.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **cardinality**

  ```
  public double cardinality()
  ```

  - **Description**
    Computes the Cardinality as the average number of labels per pattern. It requires the method calculateStats to be previously called.

  - **Returns** – double

- **coocurrenceToCSV**

```
public java.lang.String coocurrenceToCSV()
```

  - **Description**
    Returns cooCurrenceMatrix in CSV representation. It requires the method calculateCooncurrence to be previously called.
  - **Returns** – CooCurrenceMatrix in CSV representation.

- **coocurrenceToString**

```
public java.lang.String coocurrenceToString()
```

  - **Description**
    Returns cooCurrenceMatrix in textual representation. It requires the method calculateCoocurrence to be previously called.
  - **Returns** – CoocurrenceMatrix in textual representation.

- **correlationsToCSV**

```
public java.lang.String correlationsToCSV(double[][] matrix)
```

  - **Description**
    Returns Phi correlations in CSV representation. It requires the method calculatePhiChi2 to be previously called.
  - **Parameters**
    * `matrix` – Matrix with Phi correlations.
  - **Returns** – Phi correlations in CSV representation.

- **correlationsToString**

```
public java.lang.String correlationsToString(double[][] matrix)
```

  - **Description**
    Returns Phi correlations in textual representation. It requires the method calculatePhiChi2 to be previously called.
  - **Parameters**
    * `matrix` – Matrix with Phi correlations.
  - **Returns** – Phi correlations in textual representation.

- **density**

  ```
  public double density()
  ```

  - **Description**
    Computes the density as the cardinality/numLabels. It the method calculateStats to be previously called.
  - **Returns** – density.

- **distributionBagsToCSV**

  ```
  protected java.lang.String distributionBagsToCSV()
  ```

  - **Description**
    Returns distributionBags in CSV representation.
  - **Returns** – CSV with bags distribution.

- **distributionBagsToCSV**

  ```
  protected java.lang.String distributionBagsToCSV(java.util.
      HashMap skew)
  ```

  - **Description**
    Returns labelSkew in CSV representation.
  - **Parameters**
    * `skew` – The IR for each labelSet previously computed.
  - **Returns** – LabelSkew in CSV representation.

- **distributionBagsToString**

  ```
  protected java.lang.String distributionBagsToString()
  ```

  - **Description**
    Returns distributionBags in textual representation.
  - **Returns** – String with bags distribution.

- **distributionBagsToString**

  ```
  protected java.lang.String distributionBagsToString(java.util.
      HashMap skew)
  ```

- **Description**
  Returns labelSkew in textual representation.
- **Parameters**
  * `skew` – The IR for each labelSet previously computed.
- **Returns** – LabelSkew in textual representation.

• **getChi2**

```
public double[][] getChi2()
```

- **Description**
  Gets the Chi2 correlation matrix. It requires the method calculatePhiChi2 to be previously called.
- **Returns** – chi2.

• **getDataSet**

```
public miml.data.MIMLInstances getDataSet()
```

- **Description**
  Returns the dataset used to calculate the statistics.
- **Returns** – A MIML data set.

• **getPhi**

```
public double[][] getPhi()
```

- **Description**
  Gets the Phi correlation matrix. It requires the method calculatePhiChi2 to be previously called.
- **Returns** – phi.

• **getPhiHistogram**

```
public double[] getPhiHistogram()
```

- **Description**
  Calculates a histogram of Phi correlations. It requires the method calculatePhi to be previously called.
- **Returns** – An array with Phi correlations.

- **innerClassIR**

  ```
  public double[] innerClassIR()
  ```

  - **Description**
    Computes the innerClassIR for each label as negativePatterns/positivePatterns. It requires the method calculateStats to be previously called.
  - **Returns** – An IR for each label: negativePatterns/positivePatterns.

- **interClassIR**

  ```
  public double[] interClassIR()
  ```

  - **Description**
    Computes the interClassIR for each label positiveExamplesOfMajorityLabel/positivePatternsLabel. It requires the method calculateStats to be previously called.
  - **Returns** – An IR between binary labels: maxPositiveClassExamples/positiveExamplesLabel.

- **labelCombCount**

  ```
  public java.util.HashMap labelCombCount()
  ```

  - **Description**
    Returns the HashMap containing the distinct labelsets and their frequencies. It requires the method calculateStats to be previously called.
  - **Returns** – HashMap with distinct labelsest and their frequencies.

- **labelSetFrequency**

  ```
  public int labelSetFrequency(mulan.data.LabelSet x)
  ```

  - **Description**
    Returns the frequency of a label set in the dataset. It requires the method calculateStats to be previously called.
  - **Parameters**
    * `x` – A labelset.
  - **Returns** – The frequency of the given labelset.

- **labelSets**

  ```
  public java.util.Set labelSets()
  ```

  - **Description**
    Returns a set with the distinct label sets of the dataset. It requires the method calculateStats to be previously called.
  - **Returns** – Set of distinct label sets.

- **labelSkew**

```
public java.util.HashMap labelSkew()
```

  - **Description**
    Computes the IR for each labelSet as (patterns of majorityLabelSet)/(patterns of the labelSet). It requires the method calculateStats to be previously called.
  - **Returns** – HashMap<LabelSet, Double>

- **pMax**

```
public double pMax()
```

  - **Description**
    Returns pMax, the proportion of examples associated with the most frequently occurring labelset. It requires the method calculateStats to be previously called. More information in Jesse Read. 2010. Scalable Multi-label Classification. Ph.D. Dissertation. University of Waikato.
  - **Returns** – pMax.

- **printPhiDiagram**

```
public void printPhiDiagram(double step)
```

  - **Description**
    This method prints data, useful for the visualization of Phi per dataset. It prints int(1/step) + 1 pairs of values. The first value of each pair is the phi value and the second is the average number of labels that correlate to the rest of the labels with correlation higher than the specified Phi value. It requires the method calculatePhi to be previously called.
  - **Parameters**
    * `step` – The Ohi value increment step.

- **priors**

```
public double[] priors()
```

  - **Description**
    Returns the prior probabilities of the labels. It requires the method calculateStats to be previously called.
  - **Returns** – An array of double with prior probabilities of labels.

- **pUnique**

  ```
  public double pUnique()
  ```

  - **Description**
    Returns proportion of unique label combinations (pPunique) value defined as the proportion of labelsets which are unique across the total number of examples. It requires the method calculateStats to be previously called. More information in Jesse Read. 2010. Scalable Multi-label Classification. Ph.D. Dissertation. University of Waikato.
  - **Returns** – Proportion of unique label combinations.

- **setDataSet**

  ```
  public void setDataSet(miml.data.MIMLInstances dataSet)
  ```

  - **Description**
    Set the dataset used.
  - **Parameters**
    * `dataSet` – A MIML data set.

- **skewRatio**

  ```
  public double skewRatio()
  ```

  - **Description**
    Computes the skewRatio as peak/base. It requires the method calculateStats to be previously called.
  - **Returns** – SkewRatio as peak/base.

- **toCSV**

  ```
  public java.lang.String toCSV()
  ```

  - **Description**
    Returns statistics in CSV representation. It requires the method calculateStats to be previously called.

- **Returns** – Statistics in CSV representation.

- **topPhiCorrelatedLabels**

```
public int[] topPhiCorrelatedLabels(int labelIndex ,int k)
```

  - **Description**
    Returns the indices of the labels that have the strongest Phi correlation with the label which is given as a parameter. The second parameter is the number of labels that will be returned. It requires the method calculatePhi to be previously called.
  - **Parameters**
    * `labelIndex` – The label index.
    * `k` – The number of labels that will be returned. The number of labels that will be returned.
  - **Returns** – The indices of the k most correlated labels.

- **toString**

```
public java.lang.String toString()
```

  - **Description**
    Returns statistics in textual representation. It requires the method calculateStats to be previously called.
  - **Returns** – Statistics in textual representation.

- **uncorrelatedLabels**

```
public int[] uncorrelatedLabels(int labelIndex ,double bound)
```

  - **Description**
    Returns the indices of the labels whose Phi coefficient values lie between -bound <= phi <= bound. It requires the method calculatePhi to be previously called.
  - **Parameters**
    * `labelIndex` – The label index.
    * `bound` – The bound.
  - **Returns** – The indices of the labels whose Phi coefficient values lie between -bound <= phi <= bound.

- **varianceIR**

```
public double varianceIR(double[] IR)
```

- **Description**
  Computes the variance of any IR vector.
- **Parameters**
  - `IR` – An IR vector previously computed.
- **Returns** – Variance of any IR vector.

12.2 Class MIStatistics

Class with methods to obtain information about a MI dataset such as the number of attributes per bag, the average number of instances per bag, and the distribution of number of instances per bag...

12.2.1 Declaration

```
public class MIStatistics
 extends java.lang.Object
```

12.2.2 Field summary

attributesPerBag The number of attributes per bag.
avgInstancesPerBag The average number of instances per bag.
dataSet Instances dataset
distributionBags The distribution of number of instances per bag.
maxInstancesPerBag The maximum number of instances per bag.
minInstancesPerBag The minimum number of instances per bag.
numBags The number of bags.
totalInstances The total of instances.

12.2.3 Constructor summary

MIStatistics(Instances)

12.2.4 Method summary

calculateStats() Calculates various MIML statistics, such as instancesPerBag and attributesPerBag.
distributionBagsToCSV() Returns distributionBags in CSV representation.
distributionBagsToString() Returns distributionBags in textual representation.
toCSV() Returns statistics in CSV representation.
toString() Returns statistics in textual representation.

12.2.5 Fields

- `int` **minInstancesPerBag**
  - The minimum number of instances per bag.

- `int` **maxInstancesPerBag**
  - The maximum number of instances per bag.
- `double` **avgInstancesPerBag**
  - The average number of instances per bag.
- `int` **attributesPerBag**
  - The number of attributes per bag.
- `int` **numBags**
  - The number of bags.
- `int` **totalInstances**
  - The total of instances.
- `java.util.HashMap` **distributionBags**
  - The distribution of number of instances per bag.
- `weka.core.Instances` **dataSet**
  - Instances dataset

12.2.6 Constructors

- **MIStatistics**

```
public MIStatistics(weka.core.Instances dataSet)
```

12.2.7 Methods

- **calculateStats**

```
protected void calculateStats()
```

  - **Description**
    Calculates various MIML statistics, such as instancesPerBag and attributesPerBag.

- **distributionBagsToCSV**

```
protected java.lang.String distributionBagsToCSV()
```

  - **Description**
    Returns distributionBags in CSV representation.
  - **Returns** – DistributionBags in CSV representation.

- **distributionBagsToString**

  ```
  protected java.lang.String distributionBagsToString()
  ```

  - **Description**
    Returns distributionBags in textual representation.
  - **Returns** – DistributionBags in textual representation.

- **toCSV**

  ```
  public java.lang.String toCSV()
  ```

  - **Description**
    Returns statistics in CSV representation.
  - **Returns** – Statistics in CSV representation.

- **toString**

  ```
  public java.lang.String toString()
  ```

  - **Description**
    Returns statistics in textual representation.
  - **Returns** – Statistics in textual representation.

12.3 Class MLStatistics

Class with methods to obtain information about a ML dataset. This java class is based on the mulan.data.Statistics.java class provided in the Mulan java framework for multi-label learning Tsoumakas, G., Katakis, I., Vlahavas, I. (2010) "Mining Multi-label Data", Data Mining and Knowledge Discovery Handbook, O. Maimon, L. Rokach (Ed.), Springer, 2nd edition, 2010. Our contribution is mainly related with methods to measure the degree of imbalance and a fixed bug in the method printPhiDiagram.

12.3.1 Declaration

```
public class MLStatistics
 extends java.lang.Object
```

12.3.2 Field summary

base The lowest labelSet count.
chi2 Chi square matrix values where 0 = complete independence.
coocurrenceMatrix Coocurrence matrix.
distributionLabelsPerExample The number of examples having 0, 1, 2,... , numLabel labels.
labelCombinations LabelSets in the dataset.
maxCount Number of labelSets with the peak value.
mlDataSet Multi label dataset
numAttributes The number of attributes.
numExamples The number of examples.
numLabels The number of labels.
numNominal The number of nominal predictive attributes.
numNumeric The number of numeric attributes.
nUnique Number of labelSets with only one pattern.
peak The highest labelSet count.
phi Phi matrix values in [-1,1] where -1 = inverse relation, 0 = no relation, 1 = direct relation.
positiveExamplesPerLabel The number of positive examples per label.

12.3.3 Constructor summary

MLStatistics(MultiLabelInstances) Constructor.

12.3.4 Method summary

averageIR(double[]) Computes the average of any IR vector.
averageSkew(HashMap) Computes the average labelSkew.
calculateCoocurrence(MultiLabelInstances) This method calculates a matrix with the coocurrences of pairs of labels.
calculatePhiChi2(MultiLabelInstances) Calculates Phi and Chi-square correlation matrix.
calculateStats() Calculates various ML statistics.
cardinality() Computes the Cardinality as the average number of labels per pattern.
coocurrenceToCSV() Returns coocurrenceMatrix in CSV representation.
coocurrenceToString() Returns coocurrenceMatrix in textual representation.
correlationsToCSV(double[][]) Returns Phi correlations in CSV representation.
correlationsToString(double[][]) Returns Phi correlations in textual representation.
density() Computes the density as the cardinality/numLabels.
distributionBagsToCSV(HashMap) Returns labelSkew in CSV representation.
distributionBagsToString(HashMap) Returns labelSkew in textual representation.
getChi2() Gets the Chi2 correlation matrix.
getPhi() Gets the Phi correlation matrix.

getPhiHistogram() Calculates a histogram of Phi correlations.
innerClassIR() Computes the innerClassIR for each label as negativePatterns/-positivePatterns.
interClassIR() Computes the interClassIR for each label positiveExamplesOfMajorityLabel/positivePatternsLabel.
labelCombCount() Returns the HashMap containing the distinct labelsets and their frequencies.
labelSetFrequency(LabelSet) Returns the frequency of a label set in the dataset.
labelSets() Returns a set with the distinct label sets of the dataset.
labelSkew() Computes the IR for each labelSet as (patterns of majorityLabelSet)/(patterns of the labelSet).
pMax() Returns pMax, the proportion of examples associated with the most frequently occurring labelset.
printPhiDiagram(double) This method prints data, useful for the visualization of Phi per dataset.
priors() Returns the prior probabilities of the labels.
pUnique() Returns proportion of unique label combinations (pPunique) value defined as the proportion of labelsets which are unique across the total number of examples.
skewRatio() Computes the skewRatio as peak/base.
toCSV() Returns statistics in CSV representation.
topPhiCorrelatedLabels(int, int) Returns the indices of the labels that have the strongest Phi correlation with the label which is given as a parameter.
toString() Returns statistics in textual representation.
uncorrelatedLabels(int, double) Returns the indices of the labels whose Phi coefficient values lie between -bound <= phi <= bound.
varianceIR(double[]) Computes the variance of any IR vector.

12.3.5 Fields

- `protected int` **numLabels**
  - The number of labels.
- `protected int` **numExamples**
  - The number of examples.
- `protected int` **numAttributes**
  - The number of attributes.
- `protected int` **numNominal**
  - The number of nominal predictive attributes.
- `protected int` **numNumeric**
  - The number of numeric attributes.
- `protected int[]` **positiveExamplesPerLabel**
  - The number of positive examples per label.

- `protected int[]` **distributionLabelsPerExample**
  - The number of examples having 0, 1, 2,... , numLabel labels.
- `protected java.util.HashMap` **labelCombinations**
  - LabelSets in the dataset.
- `protected int` **peak**
  - The highest labelSet count.
- `protected int` **base**
  - The lowest labelSet count.
- `protected int` **nUnique**
  - Number of labelSets with only one pattern.
- `protected int` **maxCount**
  - Number of labelSets with the peak value.
- `double[][]` **coocurrenceMatrix**
  - Coocurrence matrix.
- `double[][]` **phi**
  - Phi matrix values in [-1,1] where -1 = inverse relation, 0 = no relation, 1 = direct relation.
- `double[][]` **chi2**
  - Chi square matrix values where 0 = complete independence. Values larger than 6.63 show label dependence at 0.01 level of significance (99%). Values larger than 3.84 show label dependence at 0.05 level of significance (95%).
- `private mulan.data.MultiLabelInstances` **mlDataSet**
  - Multi label dataset

12.3.6 Constructors

- **MLStatistics**

```
public MLStatistics(mulan.data.MultiLabelInstances mlDataSet)
```

  - **Description**
    Constructor.
  - **Parameters**
    * `mlDataSet` – MultiLabel dataset.

12.3.7 Methods

- **averageIR**

  ```
  public double averageIR(double[] IR)
  ```

  - **Description**
    Computes the average of any IR vector.
  - **Parameters**
    - `IR` – An IR vector previously computed
  - **Returns** – double

- **averageSkew**

  ```
  public double averageSkew(java.util.HashMap skew)
  ```

  - **Description**
    Computes the average labelSkew.
  - **Parameters**
    - `skew` – The IR for each labelSet previously computed.
  - **Returns** – double

- **calculateCoocurrence**

  ```
  public double[][] calculateCoocurrence(mulan.data.
      MultiLabelInstances mlDataSet)
  ```

  - **Description**
    This method calculates a matrix with the coocurrences of pairs of labels. It requires the method calculateStats to be previously called.
  - **Parameters**
    - `mlDataSet` – A multi-label dataset.
  - **Returns** – A coocurrences matrix of pairs of labels.

- **calculatePhiChi2**

  ```
  public void calculatePhiChi2(mulan.data.MultiLabelInstances
      dataSet) throws java.lang.Exception
  ```

  - **Description**
    Calculates Phi and Chi-square correlation matrix.

  - **Parameters**
    - `dataSet` – A multi-label dataset.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **calculateStats**

```
protected void calculateStats()
```

  - **Description**
    Calculates various ML statistics.

- **cardinality**

```
public double cardinality()
```

  - **Description**
    Computes the Cardinality as the average number of labels per pattern. It requires the method calculateStats to be previously called.
  - **Returns** – double

- **coocurrenceToCSV**

```
public java.lang.String coocurrenceToCSV()
```

  - **Description**
    Returns coocurrenceMatrix in CSV representation. It requires the method calculate-Coocurrence to be previously called.
  - **Returns** – string

- **coocurrenceToString**

```
public java.lang.String coocurrenceToString()
```

  - **Description**
    Returns coocurrenceMatrix in textual representation. It requires the method calculateCoocurrence to be previously called.
  - **Returns** – string

- **correlationsToCSV**

```
public java.lang.String correlationsToCSV(double[][] matrix)
```

- **Description**
  Returns Phi correlations in CSV representation. It requires the method calculatePhiChi2 to be previously called.
- **Parameters**
  - `matrix` – Matrix with Phi correlations.
- **Returns** – String

- **correlationsToString**

```
public java.lang.String correlationsToString(double[][] matrix)
```

- **Description**
  Returns Phi correlations in textual representation. It requires the method calculatePhiChi2 to be previously called.
- **Parameters**
  - `matrix` – Matrix with Phi correlations.
- **Returns** – string

- **density**

```
public double density()
```

- **Description**
  Computes the density as the cardinality/numLabels. It the method calculateStats to be previously called.
- **Returns** – double

- **distributionBagsToCSV**

```
protected java.lang.String distributionBagsToCSV(java.util.
    HashMap skew)
```

- **Description**
  Returns labelSkew in CSV representation.
- **Parameters**
  - `skew` – The IR for each labelSet previously computed.
- **Returns** – string

- **distributionBagsToString**

```
protected java.lang.String distributionBagsToString(java.util.
    HashMap skew)
```

  - **Description**
    Returns labelSkew in textual representation.
  - **Parameters**
    * `skew` – The IR for each labelSet previously computed.
  - **Returns** – string

- **getChi2**

  ```
  public double[][] getChi2()
  ```

  - **Description**
    Gets the Chi2 correlation matrix. It requires the method calculatePhiChi2 to be previously called.
  - **Returns** – chi2

- **getPhi**

  ```
  public double[][] getPhi()
  ```

  - **Description**
    Gets the Phi correlation matrix. It requires the method calculatePhiChi2 to be previously called.
  - **Returns** – phi

- **getPhiHistogram**

  ```
  public double[] getPhiHistogram()
  ```

  - **Description**
    Calculates a histogram of Phi correlations. It requires the method calculatePhi to be previously called.
  - **Returns** – An array with Phi correlations.

- **innerClassIR**

  ```
  public double[] innerClassIR()
  ```

  - **Description**
    Computes the innerClassIR for each label as negativePatterns/positivePatterns. It requires the method calculateStats to be previously called.
  - **Returns** – An IR for each label: negativePatterns/positivePatterns.

- **interClassIR**

  ```
  public double[] interClassIR()
  ```

  - **Description**
    Computes the interClassIR for each label positiveExamplesOfMajorityLabel/positivePatternsLabel. It requires the method calculateStats to be previously called.
  - **Returns** – An IR between binary labels: maxPositiveClassExamples/positiveExamplesLabel.

- **labelCombCount**

  ```
  public java.util.HashMap labelCombCount()
  ```

  - **Description**
    Returns the HashMap containing the distinct labelsets and their frequencies. It requires the method calculateStats to be previously called.
  - **Returns** – HashMap with distinct labelsest and their frequencies.

- **labelSetFrequency**

  ```
  public int labelSetFrequency(mulan.data.LabelSet x)
  ```

  - **Description**
    Returns the frequency of a label set in the dataset. It requires the method calculateStats to be previously called.
  - **Parameters**
    * `x` – A labelset.
  - **Returns** – The frequency of the given labelset.

- **labelSets**

  ```
  public java.util.Set labelSets()
  ```

  - **Description**
    Returns a set with the distinct label sets of the dataset. It requires the method calculateStats to be previously called.
  - **Returns** – Set of distinct label sets.

- **labelSkew**

  ```
  public java.util.HashMap labelSkew()
  ```

  - **Description**
    Computes the IR for each labelSet as (patterns of majorityLabelSet)/(patterns of the labelSet). It requires the method calculateStats to be previously called.
  - **Returns** – HashMap<LabelSet, Double>

- **pMax**

  ```
  public double pMax()
  ```

  - **Description**
    Returns pMax, the proportion of examples associated with the most frequently occurring labelset. It requires the method calculateStats to be previously called. More information in Jesse Read. 2010. Scalable Multi-label Classification. Ph.D. Dissertation. University of Waikato.
  - **Returns** – double

- **printPhiDiagram**

  ```
  public void printPhiDiagram(double step)
  ```

  - **Description**
    This method prints data, useful for the visualization of Phi per dataset. It prints int(1/step) + 1 pairs of values. The first value of each pair is the phi value and the second is the average number of labels that correlate to the rest of the labels with correlation higher than the specified Phi value. It requires the method calculatePhi to be previously called.
  - **Parameters**
    * `step` – The Ohi value increment step.

- **priors**

  ```
  public double[] priors()
  ```

  - **Description**
    Returns the prior probabilities of the labels. It requires the method calculateStats to be previously called.
  - **Returns** – An array of double with prior probabilities of labels.

- **pUnique**

  ```
  public double pUnique()
  ```

  - **Description**
    Returns proportion of unique label combinations (pPunique) value defined as the proportion of labelsets which are unique across the total number of examples. It requires the method calculateStats to be previously called. More information in Jesse Read. 2010. Scalable Multi-label Classification. Ph.D. Dissertation. University of Waikato.
  - **Returns** – double

- **skewRatio**

```
public double skewRatio()
```

  - **Description**
    Computes the skewRatio as peak/base. It requires the method calculateStats to be previously called.
  - **Returns** – double

- **toCSV**

```
public java.lang.String toCSV()
```

  - **Description**
    Returns statistics in CSV representation. It requires the method calculateStats to be previously called.
  - **Returns** – string

- **topPhiCorrelatedLabels**

```
public int[] topPhiCorrelatedLabels(int labelIndex ,int k)
```

  - **Description**
    Returns the indices of the labels that have the strongest Phi correlation with the label which is given as a parameter. The second parameter is the number of labels that will be returned. It requires the method calculatePhi to be previously called.
  - **Parameters**
    * `labelIndex` – The label index.
    * `k` – The number of labels that will be returned. The number of labels that will be returned.
  - **Returns** – The indices of the k most correlated labels.

- **toString**

```
public java.lang.String toString()
```

- **Description**
  Returns statistics in textual representation. It requires the method calculateStats to be previously called.
- **Returns** – string

- **uncorrelatedLabels**

```
public int[] uncorrelatedLabels(int labelIndex, double bound)
```

- **Description**
  Returns the indices of the labels whose Phi coefficient values lie between -bound <= phi <= bound. It requires the method calculatePhi to be previously called.
- **Parameters**
  - `labelIndex` – The label index.
  - `bound` – The bound.
- **Returns** – The indices of the labels whose Phi coefficient values lie between -bound <= phi <= bound.

- **varianceIR**

```
public double varianceIR(double[] IR)
```

- **Description**
  Computes the variance of any IR vector.
- **Parameters**
  - `IR` – An IR vector previously computed.
- **Returns** – double.

Chapter 13

Package miml.classifiers.miml.mimlTOmi

Package Contents *Page*

13.1 Class MIMLBinaryRelevance

Wrapper for mulan BinaryRelevance to be used in MIML to MI algorithms.

13.1.1 Declaration

```
public class MIMLBinaryRelevance
 extends mulan.classifier.transformation.BinaryRelevance
```

13.1.2 Field summary

serialVersionUID Generated Serial version UID.

13.1.3 Constructor summary

MIMLBinaryRelevance(Classifier) Creates a new instance.

13.1.4 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.

13.1.5 Constructors

- **MIMLBinaryRelevance**

```
public MIMLBinaryRelevance(weka.classifiers.Classifier
    classifier)
```

  - **Description**
    Creates a new instance.
  - **Parameters**
    * `classifier` – The base-level classification algorithm that will be used for training each of the binary models.

13.1.6 Members inherited from class BinaryRelevance

`mulan.classifier.transformation.BinaryRelevance`

- `private` **brt**
- `protected void` **buildInternal**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `private` **correspondence**
- `protected` **ensemble**
- `public Classifier` **getModel**`(java.lang.String` **arg0**`)`
- `protected MultiLabelOutput` **makePredictionInternal**`(weka.core.Instance` **arg0**`)`

13.1.7 Members inherited from class TransformationBasedMultiLabelLearner

`mulan.classifier.transformation.TransformationBasedMultiLabelLearner`

- `protected` **baseClassifier**
- `public Classifier` **getBaseClassifier**`()`
- `public TechnicalInformation` **getTechnicalInformation**`()`
- `public String` **globalInfo**`()`

13.1.8 Members inherited from class MultiLabelLearnerBase

`mulan.classifier.MultiLabelLearnerBase`

- `public final void` **build**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **arg0**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `public abstract TechnicalInformation` **getTechnicalInformation**`()`

- `private` **isDebug**
- `private` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public MultiLabelLearner` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **arg0**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(weka.core.Instance` **arg0**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `public void` **setDebug**`(boolean` **arg0**`)`

13.2 Class MIMLClassifierToMI

Class implementing the transformation algorithm for MIML data to solve it with MI learning. For more information, see *Zhou, Z. H., & Zhang, M. L. (2007). Multi-instance multi-label learning with application to scene classification. In Advances in neural information processing systems (pp. 1609-1616).*

13.2.1 Declaration

```
public class MIMLClassifierToMI
 extends miml.classifiers.miml.MIMLClassifier
```

13.2.2 Field summary

serialVersionUID Generated Serial version UID.
transformationClassifier Generic classifier used for transformation.

13.2.3 Constructor summary

MIMLClassifierToMI() No-argument constructor for xml configuration.
MIMLClassifierToMI(MultiLabelLearner) Basic constructor.

13.2.4 Method summary

buildInternal(MIMLInstances)
configure(Configuration)
makePredictionInternal(MIMLBag)

13.2.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.
- `protected mulan.classifier.MultiLabelLearner` **transformationClassifier**
  - Generic classifier used for transformation.

13.2.6 Constructors

- **MIMLClassifierToMI**

  ```
  public MIMLClassifierToMI()
  ```

  - **Description**
    No-argument constructor for xml configuration.

- **MIMLClassifierToMI**

  ```
  public MIMLClassifierToMI(mulan.classifier.MultiLabelLearner
      transformationClassifier)
  ```

  - **Description**
    Basic constructor.
  - **Parameters**
    - `transformationClassifier` – Mulan MultiLabelLearner used as transformation method from MIML to MI.

13.2.7 Methods

- **buildInternal**

  ```
  protected abstract void buildInternal(miml.data.MIMLInstances
      trainingSet) throws java.lang.Exception
  ```

  - **Description copied from miml.classifiers.miml.MIMLClassifier** (**in** 10.2, **page** 141)
    Learner specific implementation of building the model from `MultiLabelInstances` training data set. This method is called from `build(MultiLabelInstances)` method, where behavior common across all learners is applied.
  - **Parameters**
    - `trainingSet` – The training data set.
  - **Throws**
    - `java.lang.Exception` – if learner model was not created successfully.

- **configure**

  ```
  public void configure(org.apache.commons.configuration2.
      Configuration configuration)
  ```

- **makePredictionInternal**

```
protected abstract mulan.classifier.MultiLabelOutput
    makePredictionInternal(miml.data.MIMLBag instance) throws
    java.lang.Exception, mulan.classifier.InvalidDataException
```

  - **Description copied from miml.classifiers.miml.MIMLClassifier** (**in** 10.2, **page** 141)
    Learner specific implementation for predicting on specified data based on trained model. This method is called from `makePrediction(Instance)` which guards for model initialization and apply common handling/behavior.
  - **Parameters**
    * `instance` – The data instance to predict on.
  - **Returns** – The output of the learner for the given instance.
  - **Throws**
    * `java.lang.Exception` – If an error occurs while making the prediction.
    * `mulan.classifier.InvalidDataException` – If specified instance data is invalid and can not be processed by the learner.

13.2.8 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

13.3 Class MIMLLabelPowerset

Wrapper for mulan LabelPowerset to be used in MIML to MI algorithms.

13.3.1 Declaration

```
public class MIMLLabelPowerset
 extends mulan.classifier.transformation.LabelPowerset
```

13.3.2 Field summary

serialVersionUID Generated Serial version UID.

13.3.3 Constructor summary

MIMLLabelPowerset(Classifier) Constructor that initializes the learner with a base classifier.

13.3.4 Method summary

buildInternal(MultiLabelInstances)

13.3.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.

13.3.6 Constructors

- **MIMLLabelPowerset**

  ```
  public MIMLLabelPowerset(weka.classifiers.Classifier classifier)
  ```

  - **Description**
    Constructor that initializes the learner with a base classifier.
  - **Parameters**
    * `classifier` – The base single-label classification algorithm.

13.3.7 Methods

- **buildInternal**

  ```
  protected abstract void buildInternal(mulan.data.
      MultiLabelInstances arg0) throws java.lang.Exception
  ```

13.3.8 Members inherited from class LabelPowerset

`mulan.classifier.transformation.LabelPowerset`

- `protected void` **buildInternal**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `private` **confidenceCalculationMethod**
- `protected MultiLabelOutput` **makePredictionInternal**`(weka.core.Instance` **arg0**`) throws java.lang.Exception`
- `protected` **makePredictionsBasedOnConfidences**
- `protected` **Rand**
- `public void` **setConfidenceCalculationMethod**`(int` **arg0**`)`
- `public void` **setMakePredictionsBasedOnConfidences**`(boolean` **arg0**`)`
- `public void` **setSeed**`(int` **arg0**`)`
- `public void` **setThreshold**`(double` **arg0**`)`
- `protected` **threshold**
- `protected` **transformation**

13.3.9 Members inherited from class TransformationBasedMultiLabelLearner

`mulan.classifier.transformation.TransformationBasedMultiLabelLearner`

- `protected` **baseClassifier**
- `public Classifier` **getBaseClassifier**`()`
- `public TechnicalInformation` **getTechnicalInformation**`()`
- `public String` **globalInfo**`()`

13.3.10 Members inherited from class MultiLabelLearnerBase

`mulan.classifier.MultiLabelLearnerBase`

- `public final void` **build**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(mulan.data.MultiLabelInstances` **arg0**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **arg0**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `public abstract TechnicalInformation` **getTechnicalInformation**`()`
- `private` **isDebug**
- `private` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public MultiLabelLearner` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **arg0**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(weka.core.Instance` **arg0**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `public void` **setDebug**`(boolean` **arg0**`)`

Chapter 14

Package miml.data.partitioning.powerset

Package Contents *Page*

14.1 Class LabelPowersetCrossValidation

Class to split a multi-label dataset into N multi-label for cross-validation by applying a labelPowerset-based partition. MIML and MVML formats are also supported.

14.1.1 Declaration

```
public class LabelPowersetCrossValidation
 extends miml.data.partitioning.CrossValidationBase
```

14.1.2 Constructor summary

LabelPowersetCrossValidation(int, MultiLabelInstances) Constructor.
LabelPowersetCrossValidation(MultiLabelInstances) Default constructor.

14.1.3 Method summary

getFolds(int)

14.1.4 Constructors

- **LabelPowersetCrossValidation**

```
public LabelPowersetCrossValidation(int seed,mulan.data.
    MultiLabelInstances mlDataSet) throws mulan.data.
    InvalidDataFormatException
```

  - **Description**
    Constructor.
  - **Parameters**
    * `seed` – Seed for randomization
    * `mlDataSet` – A multi-label dataset
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – To be handled

- **LabelPowersetCrossValidation**

```
public LabelPowersetCrossValidation(mulan.data.
    MultiLabelInstances mlDataSet) throws mulan.data.
    InvalidDataFormatException
```

  - **Description**
    Default constructor.
  - **Parameters**
    * `mlDataSet` – A multi-label dataset
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – To be handled

14.1.5 Methods

- **getFolds**

```
public abstract mulan.data.MultiLabelInstances[] getFolds(int
    nFolds) throws mulan.data.InvalidDataFormatException
```

  - **Description copied from miml.data.partitioning.CrossValidationBase** (**in 21.1, page 284**)
    Splits a dataset into nfolds partitions.
  - **Parameters**
    * `nFolds` – Number of folds.
  - **Returns** – MultiLabelInstances[] a vector of nFolds. Each element represents a fold.
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – To be handled.

14.1.6 Members inherited from class CrossValidationBase

`miml.data.partitioning.CrossValidationBase` (in 21.1, page 284)

- `public static MultiLabelInstances` **foldsToRounds**`(mulan.data.MultiLabelInstances[]` **Folds**`) throws java.lang.Exception`
- `public abstract MultiLabelInstances` **getFolds**`(int` **nFolds**`) throws mulan.data.InvalidDataFormatException`
- `public MultiLabelInstances` **getRounds**`(int` **nFolds**`) throws java.lang.Exception`
- `protected void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`

14.1.7 Members inherited from class PartitionerBase

`miml.data.partitioning.PartitionerBase` (in 21.2, page 287)

- `protected` **seed**
- `protected abstract void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`
- `public int` **totalExamples**`()`
- `protected` **workingSet**

14.2 Class LabelPowersetTrainTest

Class to split a multi-label dataset into two multi-label datasets corresponding to the train and test datasets respectively by applying a labelPowerset-based partition. MIML and MVML formats are also supported.

14.2.1 Declaration

```
public class LabelPowersetTrainTest
 extends miml.data.partitioning.TrainTestBase
```

14.2.2 Constructor summary

LabelPowersetTrainTest(int, MultiLabelInstances) Constructor.
LabelPowersetTrainTest(MultiLabelInstances) Default constructor.

14.2.3 Method summary

split(double)

14.2.4 Constructors

- **LabelPowersetTrainTest**

```
public LabelPowersetTrainTest(int seed,mulan.data.
    MultiLabelInstances mlDataSet) throws mulan.data.
    InvalidDataFormatException
```

  - **Description**
    Constructor.

- **Parameters**
  - `seed` – Seed for randomization
  - `mlDataSet` – A multi-label dataset
- **Throws**
  - `mulan.data.InvalidDataFormatException` – To be handled

- **LabelPowersetTrainTest**

```
public LabelPowersetTrainTest(mulan.data.MultiLabelInstances
    mlDataSet) throws mulan.data.InvalidDataFormatException
```

- **Description**
  Default constructor.
- **Parameters**
  - `mlDataSet` – A multi-label dataset
- **Throws**
  - `mulan.data.InvalidDataFormatException` – To be handled

14.2.5 Methods

- **split**

```
public abstract mulan.data.MultiLabelInstances[] split(double
    percentageTrain) throws java.lang.Exception
```

- **Description copied from miml.data.partitioning.TrainTestBase** (**in 21.3, page 289**)
  Returns a array with two multi-label random datasets corresponding to the train and test sets respectively.
- **Parameters**
  - `percentageTrain` – Percentage of train dataset, a value in [0, 100].
- **Returns** – MultiLabelInstances[].
  MultiLabelInstances[0] is the train set.
  MultiLabelInstances[1] is the test set.
- **Throws**
  - `java.lang.Exception` – To be handled.

14.2.6 Members inherited from class TrainTestBase

`miml.data.partitioning.TrainTestBase` (in 21.3, page 289)

- `public abstract MultiLabelInstances` **split**`(double` **percentageTrain**`) throws java.lang.Exception`
- `protected void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`

14.2.7 Members inherited from class PartitionerBase

`miml.data.partitioning.PartitionerBase` (in 21.2, page 287)

- `protected` **seed**
- `protected abstract void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`
- `public int` **totalExamples**`()`
- `protected` **workingSet**

Chapter 15

Package miml.run

15.1 Class RunAlgorithm

Class that allow run any algorithm of the library configured by a file configuration.

15.1.1 Declaration

```
public class RunAlgorithm
 extends java.lang.Object
```

15.1.2 Constructor summary

RunAlgorithm()

15.1.3 Method summary

main(String[]) The main method to configure and run an algorithm.

15.1.4 Constructors

- **RunAlgorithm**

```
public RunAlgorithm()
```

15.1.5 Methods

- **main**

  ```
  public static void main(java.lang.String[] args)
  ```

  - **Description**
    The main method to configure and run an algorithm.
  - **Parameters**
    * `args` – The argument (route of config file with the option -c).

Chapter 16

Package miml.classifiers.miml.lazy

Package Contents *Page*

16.1 Class DMIMLkNN

DMIMLkNN is the adaptation to the MIML framework of the DMLkNN[1] multi-label algorithm. To perform this adaptation, DMIMLkNN maintains the treatment of labels of DMLkNN but uses a multi-instance measure of distance. *[1] Zoulficar Younes, Fahed Abdallah, Thierry Denceaux (2008). Multi-label classification algorithm derived from k-nearest neighbor rule with label dependencies. In Proceedings of 16th European Signal Processing Conference (EUSIPCO 2008), Lausanne, Switzerland.*

16.1.1 Declaration

```
public class DMIMLkNN
 extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

16.1.2 Field summary

serialVersionUID Generated Serial version UID.
smooth Smoothing parameter controlling the strength of uniform prior (Default value is set to 1 which yields the Laplace smoothing).

16.1.3 Constructor summary

DMIMLkNN() No-arg constructor for xml configuration
DMIMLkNN(int, double, MIMLDistanceFunction) A constructor that sets the number of neighbours and the value of smooth.
DMIMLkNN(int, MIMLDistanceFunction) A constructor that sets the number of neighbours.
DMIMLkNN(MIMLDistanceFunction) Default constructor.

16.1.4 Method summary

configure(Configuration)
getSmooth() Gets the smooth factor considered by the classifier.
setSmooth(double) Sets the smooth factor considered by the classifier.

16.1.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.
- `protected double` **smooth**
  - Smoothing parameter controlling the strength of uniform prior (Default value is set to 1 which yields the Laplace smoothing).

16.1.6 Constructors

- **DMIMLkNN**

  ```
  public DMIMLkNN()
  ```

  - **Description**
    No-arg constructor for xml configuration

- **DMIMLkNN**

  ```
  public DMIMLkNN(int numOfNeighbours,double smooth,
      MIMLDistanceFunction metric)
  ```

  - **Description**
    A constructor that sets the number of neighbours and the value of smooth.
  - **Parameters**
    * `metric` – The distance metric between bags considered by the classifier.
    * `numOfNeighbours` – The number of neighbours.
    * `smooth` – The smooth factor.

- **DMIMLkNN**

  ```
  public DMIMLkNN(int numOfNeighbours,MIMLDistanceFunction metric)
  ```

  - **Description**
    A constructor that sets the number of neighbours.
  - **Parameters**
    * `metric` – The distance metric between bags considered by the classifier.
    * `numOfNeighbours` – The number of neighbours.

- **DMIMLkNN**

  ```
  public DMIMLkNN(MIMLDistanceFunction metric)
  ```

  - **Description**
    Default constructor.
  - **Parameters**
    * `metric` – The distance metric between bags considered by the classifier.

16.1.7 Methods

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **getSmooth**

```
public double getSmooth()
```

  - **Description**
    Gets the smooth factor considered by the classifier.
  - **Returns** – the smooth factor

- **setSmooth**

```
public void setSmooth(double smooth)
```

  - **Description**
    Sets the smooth factor considered by the classifier.
  - **Parameters**
    * `smooth` – the new smooth factor

16.1.8 Members inherited from class MultiInstanceMultiLabelKNN

`miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN` (in 16.10, page 233)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public void` **configure**`(org.apache.commons.configuration2.Configuration` **configuration**`)`
- `public MultiLabelKNN` **getClassifier**`()`
- `public DistanceFunction` **getMetric**`()`
- `public int` **getNumOfNeighbours**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **metric**
- `protected` **numOfNeighbours**
- `private static final` **serialVersionUID**
- `public void` **setClassifier**`(mulan.classifier.lazy.MultiLabelKNN` **classifier**`)`
- `public void` **setMetric**`(weka.core.DistanceFunction` **metric**`)`
- `public void` **setnumOfNeighbours**`(int` **numOfNeighbours**`)`

16.1.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

16.2 Class MIMLBRkNN

MIMLBRkNN is the adaptation to the MIML framework of the BRkNN[1] multi-label algorithm. To perform this adaptation, MIMLBRkNN maintains the treatment of labels of BRkNN but uses a multi-instance measure of distance. *[1] Eleftherios Spyromitros, Grigorios Tsoumakas, Ioannis Vlahavas: An Empirical Study of Lazy Multilabel Classification Algorithms. In: Proc. 5th Hellenic Conference on Artificial Intelligence (SETN 2008), 2008.*

16.2.1 Declaration

```
public class MIMLBRkNN
 extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

16.2.2 Field summary

extension The type of extension to be used:

- NONE: Standard BR.

serialVersionUID Generated Serial version UID.

16.2.3 Constructor summary

MIMLBRkNN() No-arg constructor for xml configuration
MIMLBRkNN(MIMLDistanceFunction) Default constructor.
MIMLBRkNN(MIMLDistanceFunction, int) A constructor that sets the number of neighbours.
MIMLBRkNN(MIMLDistanceFunction, int, BRkNN.ExtensionType) Constructor giving the option to select an extension of the base version.

16.2.4 Method summary

configure(Configuration)
getExtension() Gets the type of extension to be used (see `BRkNN.ExtensionType`).
setExtension(BRkNN.ExtensionType) Sets the type of extension to be used (see `BRkNN.ExtensionType`).

16.2.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.
- `private mulan.classifier.lazy.BRkNN.ExtensionType` **extension**
  - The type of extension to be used:
    * NONE: Standard BR.
    * EXTA: Predict top ranked label in case of empty prediction set.
    * EXTB: Predict top n ranked labels based on size of labelset in neighbours.

16.2.6 Constructors

- **MIMLBRkNN**

```
public MIMLBRkNN()
```

  - **Description**
    No-arg constructor for xml configuration

- **MIMLBRkNN**

```
public MIMLBRkNN(MIMLDistanceFunction metric)
```

  - **Description**
    Default constructor.
  - **Parameters**
    * `metric` – The distance metric between bags considered by the classifier.

- **MIMLBRkNN**

  ```
  public MIMLBRkNN(MIMLDistanceFunction metric,int numOfNeighbours
      )
  ```

  - **Description**
    A constructor that sets the number of neighbours.
  - **Parameters**
    * `metric` – The distance metric between bags considered by the classifier.
    * `numOfNeighbours` – the number of neighbours.

- **MIMLBRkNN**

  ```
  public MIMLBRkNN(MIMLDistanceFunction metric,int numOfNeighbours
      ,mulan.classifier.lazy.BRkNN.ExtensionType ext)
  ```

  - **Description**
    Constructor giving the option to select an extension of the base version.
  - **Parameters**
    * `metric` – The distance metric between bags considered by the classifier.
    * `numOfNeighbours` – the number of neighbours
    * `ext` – the extension to use (see `BRkNN.ExtensionType`).

16.2.7 Methods

- **configure**

  ```
  public void configure(org.apache.commons.configuration2.
      Configuration configuration)
  ```

- **getExtension**

  ```
  public mulan.classifier.lazy.BRkNN.ExtensionType getExtension()
  ```

  - **Description**
    Gets the type of extension to be used (see `BRkNN.ExtensionType`).
  - **Returns** – extension Extension to be used

- **setExtension**

  ```
  public void setExtension(mulan.classifier.lazy.BRkNN.
      ExtensionType extension)
  ```

- **Description**

  Sets the type of extension to be used (see `BRkNN.ExtensionType`).
- **Parameters**
  * `extension` – The new value of the type of extension.

16.2.8 Members inherited from class MultiInstanceMultiLabelKNN

`miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN` (in 16.10, page 233)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public void` **configure**`(org.apache.commons.configuration2.Configuration` **configuration**`)`
- `public MultiLabelKNN` **getClassifier**`()`
- `public DistanceFunction` **getMetric**`()`
- `public int` **getNumOfNeighbours**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **metric**
- `protected` **numOfNeighbours**
- `private static final` **serialVersionUID**
- `public void` **setClassifier**`(mulan.classifier.lazy.MultiLabelKNN` **classifier**`)`
- `public void` **setMetric**`(weka.core.DistanceFunction` **metric**`)`
- `public void` **setnumOfNeighbours**`(int` **numOfNeighbours**`)`

16.2.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

16.3 Class MIMLDGC

MIMLDGC is the adaptation to the MIML framework of the MLDGC[1] multi-label algorithm. To perform this adaptation, MIMLDGC maintains the treatment of labels of MLDGC but uses a multi-instance measure of distance. *[1] Oscar Reyes, Carlos Morell, Sebastián Ventura (2016). Effective lazy learning algorithm based on a data gravitation model for multi-label learning. Information Sciences. Vol 340, issue C.*

16.3.1 Declaration

```
public class MIMLDGC
 extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

16.3.2 Field summary

serialVersionUID For serialization.

16.3.3 Constructor summary

MIMLDGC() No-arg constructor for xml configuration
MIMLDGC(MIMLDistanceFunction) Default constructor.
MIMLDGC(MIMLDistanceFunction, int) A constructor that sets the number of neighbours.

16.3.4 Method summary

configure(Configuration)

16.3.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.

16.3.6 Constructors

- **MIMLDGC**

  ```
  public MIMLDGC()
  ```

  - **Description**
    No-arg constructor for xml configuration

- **MIMLDGC**

  ```
  public MIMLDGC(MIMLDistanceFunction metric)
  ```

- **Description**
  Default constructor.
- **Parameters**
  - `metric` – The distance metric between bags considered by the classifier.

- **MIMLDGC**

```
public MIMLDGC(MIMLDistanceFunction metric, int numOfNeighbours)
```

- **Description**
  A constructor that sets the number of neighbours.
- **Parameters**
  - `metric` – The distance metric between bags considered by the classifier.
  - `numOfNeighbours` – the number of neighbours.

16.3.7 Methods

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

16.3.8 Members inherited from class MultiInstanceMultiLabelKNN

`miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN` (in 16.10, page 233)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public void` **configure**`(org.apache.commons.configuration2.Configuration` **configuration**`)`
- `public MultiLabelKNN` **getClassifier**`()`
- `public DistanceFunction` **getMetric**`()`
- `public int` **getNumOfNeighbours**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **metric**
- `protected` **numOfNeighbours**
- `private static final` **serialVersionUID**
- `public void` **setClassifier**`(mulan.classifier.lazy.MultiLabelKNN` **classifier**`)`
- `public void` **setMetric**`(weka.core.DistanceFunction` **metric**`)`
- `public void` **setnumOfNeighbours**`(int` **numOfNeighbours**`)`

16.3.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

16.4 Class MIMLDistanceFunction

Wrapper for using IDistance metrics of MIML package with Mulan Lazy algorithms.

16.4.1 Declaration

```
public class MIMLDistanceFunction
 extends weka.core.NormalizableDistance
```

16.4.2 Field summary

metric Metric to measure distance between bags.
serialVersionUID

16.4.3 Constructor summary

MIMLDistanceFunction(IDistance) Constructor that sets the metric to be used.

16.4.4 Method summary

distance(Instance, Instance)
distance(Instance, Instance, double)
distance(Instance, Instance, double, PerformanceStats)
distance(Instance, Instance, PerformanceStats)
getAttributeIndices()
getInstances()
getInvertSelection()
getMetric()
getOptions()
getRevision()
globalInfo()
listOptions()
postProcessDistances(double[])
setAttributeIndices(String)
setInstances(Instances)
setInvertSelection(boolean)
setMetric(IDistance) Sets the metric to be used.
setOptions(String[])
update(Instance)
updateDistance(double, double)

16.4.5 Fields

- `private static final long` **serialVersionUID**
- `protected miml.core.distance.IDistance` **metric**
  - Metric to measure distance between bags.

16.4.6 Constructors

- **MIMLDistanceFunction**

```
public MIMLDistanceFunction(miml.core.distance.IDistance metric)
```

  - **Description**
    Constructor that sets the metric to be used.
  - **Parameters**
    * `metric` – The metric to be used.

16.4.7 Methods

- **distance**

```
double distance(weka.core.Instance arg0 ,weka.core.Instance arg1)
```

- **distance**

```
double distance(weka.core.Instance arg0,weka.core.Instance arg1,
    double arg2)
```

- **distance**

```
double distance(weka.core.Instance arg0,weka.core.Instance arg1,
    double arg2,weka.core.neighboursearch.PerformanceStats arg3)
```

- **distance**

```
double distance(weka.core.Instance arg0,weka.core.Instance arg1,
    weka.core.neighboursearch.PerformanceStats arg2) throws java.
    lang.Exception
```

- **getAttributeIndices**

```
java.lang.String getAttributeIndices()
```

- **getInstances**

```
weka.core.Instances getInstances()
```

- **getInvertSelection**

```
boolean getInvertSelection()
```

- **getMetric**

```
public miml.core.distance.IDistance getMetric()
```

- **getOptions**

```
java.lang.String[] getOptions()
```

- **getRevision**

```
public java.lang.String getRevision()
```

- **globalInfo**

```
public abstract java.lang.String globalInfo()
```

- **listOptions**

```
java.util.Enumeration listOptions()
```

- **postProcessDistances**

```
void postProcessDistances(double[] arg0)
```

- **setAttributeIndices**

```
void setAttributeIndices(java.lang.String arg0)
```

- **setInstances**

```
void setInstances(weka.core.Instances arg0)
```

- **setInvertSelection**

```
void setInvertSelection(boolean arg0)
```

- **setMetric**

```
public void setMetric(miml.core.distance.IDistance metric)
```

  - **Description**
    Sets the metric to be used.
  - **Parameters**
    * `metric` – The metric to be used.

- **setOptions**

```
void setOptions(java.lang.String[] arg0) throws java.lang.
    Exception
```

- **update**

```
void update(weka.core.Instance arg0)
```

- **updateDistance**

```
protected abstract double updateDistance(double arg0,double arg1
    )
```

16.4.8 Members inherited from class NormalizableDistance

`weka.core.NormalizableDistance`

- `public String` **attributeIndicesTipText**`()`
- `protected double` **difference**`(int` **arg0**`, double` **arg1**`, double` **arg2**`)`
- `public double` **distance**`(Instance` **arg0**`, Instance` **arg1**`)`
- `public double` **distance**`(Instance` **arg0**`, Instance` **arg1**`, double` **arg2**`)`
- `public double` **distance**`(Instance` **arg0**`, Instance` **arg1**`, double` **arg2**`, neighboursearch.PerformanceStats` **arg3**`)`
- `public double` **distance**`(Instance` **arg0**`, Instance` **arg1**`, neighboursearch.PerformanceStats` **arg2**`)`
- `public String` **dontNormalizeTipText**`()`
- `public String` **getAttributeIndices**`()`
- `public boolean` **getDontNormalize**`()`
- `public Instances` **getInstances**`()`
- `public boolean` **getInvertSelection**`()`
- `public String` **getOptions**`()`
- `public double` **getRanges**`() throws java.lang.Exception`
- `public abstract String` **globalInfo**`()`
- `protected void` **initialize**`()`
- `protected void` **initializeAttributeIndices**`()`
- `public double` **initializeRanges**`()`
- `public double` **initializeRanges**`(int[]` **arg0**`) throws java.lang.Exception`
- `public double` **initializeRanges**`(int[]` **arg0**`, int` **arg1**`, int` **arg2**`) throws java.lang.Exception`
- `public void` **initializeRangesEmpty**`(int` **arg0**`, double[][]` **arg1**`)`
- `public boolean` **inRanges**`(Instance` **arg0**`, double[][]` **arg1**`)`
- `protected void` **invalidate**`()`
- `public String` **invertSelectionTipText**`()`
- `public Enumeration` **listOptions**`()`
- `protected` **m_ActiveIndices**
- `protected` **m_AttributeIndices**
- `protected` **m_Data**
- `protected` **m_DontNormalize**
- `protected` **m_Ranges**
- `protected` **m_Validated**
- `protected double` **norm**`(double` **arg0**`, int` **arg1**`)`
- `public void` **postProcessDistances**`(double[]` **arg0**`)`
- `public static final` **R_MAX**
- `public static final` **R_MIN**
- `public static final` **R_WIDTH**
- `public boolean` **rangesSet**`()`
- `public void` **setAttributeIndices**`(java.lang.String` **arg0**`)`
- `public void` **setDontNormalize**`(boolean` **arg0**`)`
- `public void` **setInstances**`(Instances` **arg0**`)`
- `public void` **setInvertSelection**`(boolean` **arg0**`)`
- `public void` **setOptions**`(java.lang.String[]` **arg0**`) throws java.lang.Exception`
- `public String` **toString**`()`
- `public void` **update**`(Instance` **arg0**`)`
- `protected abstract double` **updateDistance**`(double` **arg0**`, double` **arg1**`)`
- `public void` **updateRanges**`(Instance` **arg0**`)`
- `public double` **updateRanges**`(Instance` **arg0**`, double[][]` **arg1**`)`
- `public void` **updateRanges**`(Instance` **arg0**`, int` **arg1**`, double[][]` **arg2**`)`
- `public void` **updateRangesFirst**`(Instance` **arg0**`, int` **arg1**`, double[][]` **arg2**`)`
- `protected void` **validate**`()`

16.5 Class MIMLFuzzykNN

16.5.1 Declaration

```
public class MIMLFuzzykNN
 extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

16.5.2 Field summary

dataset Instances.
e Tolerance to compare float values.
elnn To perform neighborhood search.
ini Type of initialization: Crisp, fuzzy
k Neighborhood size.
kini Neighborhood size for initialization of U matrix.
m Fuzzy exponent.
serialVersionUID For serialization.
U Partition matrix of num_labels x num_bags

16.5.3 Constructor summary

MIMLFuzzykNN()

16.5.4 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `protected miml.data.MIMLInstances` **dataset**
  - Instances.
- `protected int` **k**
  - Neighborhood size.
- `protected double[][]` **U**
  - Partition matrix of num_labels x num_bags
- `protected int` **kini**
  - Neighborhood size for initialization of U matrix.
- `protected double` **m**
  - Fuzzy exponent.
- `protected int` **ini**
  - Type of initialization: Crisp, fuzzy
- `protected MIMLFuzzykNN.LinearNNESearch` **elnn**

  - To perform neighborhood search.

- `protected double` **e**
  - Tolerance to compare float values.

16.5.5 Constructors

- **MIMLFuzzykNN**

  public MIMLFuzzykNN()

16.5.6 Members inherited from class MultiInstanceMultiLabelKNN

`miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN` (in 16.10, page 233)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public void` **configure**`(org.apache.commons.configuration2.Configuration` **configuration**`)`
- `public MultiLabelKNN` **getClassifier**`()`
- `public DistanceFunction` **getMetric**`()`
- `public int` **getNumOfNeighbours**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **metric**
- `protected` **numOfNeighbours**
- `private static final` **serialVersionUID**
- `public void` **setClassifier**`(mulan.classifier.lazy.MultiLabelKNN` **classifier**`)`
- `public void` **setMetric**`(weka.core.DistanceFunction` **metric**`)`
- `public void` **setnumOfNeighbours**`(int` **numOfNeighbours**`)`

16.5.7 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`

- `public final MultiLabelOutput` **makePrediction**(`weka.core.Instance` **instance**) `throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**(`miml.data.MIMLBag` **instance**) `throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**(`boolean` **debug**)

16.6 Class MIMLFuzzykNN.LinearNNESearch

16.6.1 Declaration

```
class MIMLFuzzykNN.LinearNNESearch
extends weka.core.neighboursearch.LinearNNSearch
```

16.6.2 Field summary

serialVersionUID For serialization

16.6.3 Constructor summary

LinearNNESearch(Instances)

16.6.4 Method summary

kNearestNeighboursIndices(Instance, int)

16.6.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization

16.6.6 Constructors

- **LinearNNESearch**

```
public LinearNNESearch(weka.core.Instances insts) throws java.
    lang.Exception
```

16.6.7 Methods

- **kNearestNeighboursIndices**

```
public int[] kNearestNeighboursIndices(weka.core.Instance target
    ,int kNN) throws java.lang.Exception
```

16.6.8 Members inherited from class LinearNNSearch

`weka.core.neighboursearch.LinearNNSearch`

- public void **addInstanceInfo**(weka.core.Instance **arg0**)
- public double **getDistances**() throws java.lang.Exception
- public String **getOptions**()
- public String **getRevision**()
- public boolean **getSkipIdentical**()
- public String **globalInfo**()
- public Instances **kNearestNeighbours**(weka.core.Instance **arg0**, int **arg1**) throws java.lang.Exception
- public Enumeration **listOptions**()
- protected **m_Distances**
- protected **m_SkipIdentical**
- public Instance **nearestNeighbour**(weka.core.Instance **arg0**) throws java.lang.Exception
- private static final **serialVersionUID**
- public void **setInstances**(weka.core.Instances **arg0**) throws java.lang.Exception
- public void **setOptions**(java.lang.String[] **arg0**) throws java.lang.Exception
- public void **setSkipIdentical**(boolean **arg0**)
- public String **skipIdenticalTipText**()
- public void **update**(weka.core.Instance **arg0**) throws java.lang.Exception

16.6.9 Members inherited from class NearestNeighbourSearch

`weka.core.neighboursearch.NearestNeighbourSearch`

- public void **addInstanceInfo**(weka.core.Instance **arg0**)
- public static void **combSort11**(double[] **arg0**, int[] **arg1**)
- public String **distanceFunctionTipText**()
- public Enumeration **enumerateMeasures**()
- public DistanceFunction **getDistanceFunction**()
- public abstract double **getDistances**() throws java.lang.Exception
- public Instances **getInstances**()
- public double **getMeasure**(java.lang.String **arg0**)
- public boolean **getMeasurePerformance**()
- public String **getOptions**()
- public PerformanceStats **getPerformanceStats**()
- public String **globalInfo**()
- public abstract Instances **kNearestNeighbours**(weka.core.Instance **arg0**, int **arg1**) throws java.lang.Exception
- public Enumeration **listOptions**()
- protected **m_DistanceFunction**
- protected **m_Instances**
- protected **m_kNN**
- protected **m_MeasurePerformance**
- protected **m_Stats**
- public String **measurePerformanceTipText**()
- public abstract Instance **nearestNeighbour**(weka.core.Instance **arg0**) throws java.lang.Exception
- protected static int **partition**(double[] **arg0**, double[] **arg1**, int **arg2**, int **arg3**)
- public static void **quickSort**(double[] **arg0**, double[] **arg1**, int **arg2**, int **arg3**)
- public void **setDistanceFunction**(weka.core.DistanceFunction **arg0**) throws java.lang.Exception
- public void **setInstances**(weka.core.Instances **arg0**) throws java.lang.Exception
- public void **setMeasurePerformance**(boolean **arg0**)
- public void **setOptions**(java.lang.String[] **arg0**) throws java.lang.Exception
- public abstract void **update**(weka.core.Instance **arg0**) throws java.lang.Exception

16.7 Class MIMLIBLR

MIMLIBLR is the adaptation to the MIML framework of the IBLR_ML[1] multi-label algorithm. To perform this adaptation, MIMLIBLR maintains the treatment of labels of IBLR_ML but uses a multi-instance measure of distance. *[1] Weiwei Cheng, Eyke Hullermeier (2009). Combining instance-based learning and logistic regression for multilabel classification. Machine Learning. 76(2-3):211-225.*

16.7.1 Declaration

```
public class MIMLIBLR
 extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

16.7.2 Field summary

addFeatures By default, IBLR-ML is used (addFeatures is false).
serialVersionUID Generated Serial version UID.

16.7.3 Constructor summary

MIMLIBLR() No-arg constructor for xml configuration
MIMLIBLR(int, boolean, MIMLDistanceFunction) A constructor that sets the number of neighbours and whether IBLR-ML or IBLR-ML+ is used.
MIMLIBLR(int, MIMLDistanceFunction) A constructor that sets the number of neighbours.
MIMLIBLR(MIMLDistanceFunction) Default constructor.

16.7.4 Method summary

configure(Configuration)
getAddFeatures() Gets the value of addFeatures.
setAddFeatures(boolean) Sets the value of AddFeatures.

16.7.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.
- `private boolean` **addFeatures**
  - By default, IBLR-ML is used (addFeatures is false). One can change to IBLR-ML+ through the constructor.

16.7.6 Constructors

- **MIMLIBLR**

```
public MIMLIBLR()
```

- **Description**
  No-arg constructor for xml configuration

- **MIMLIBLR**

```
public MIMLIBLR(int numOfNeighbours,boolean addFeatures,
    MIMLDistanceFunction metric)
```

  - **Description**
    A constructor that sets the number of neighbours and whether IBLR-ML or IBLR-ML+ is used.
  - **Parameters**
    * `metric` – The distance metric between bags considered by the classifier.
    * `numOfNeighbours` – The number of neighbours.
    * `addFeatures` – If false IBLR-ML is used. If true, IBLR-ML+ is used.

- **MIMLIBLR**

```
public MIMLIBLR(int numOfNeighbours,MIMLDistanceFunction metric)
```

  - **Description**
    A constructor that sets the number of neighbours.
  - **Parameters**
    * `metric` – The distance metric between bags considered by the classifier.
    * `numOfNeighbours` – The number of neighbours.

- **MIMLIBLR**

```
public MIMLIBLR(MIMLDistanceFunction metric)
```

  - **Description**
    Default constructor.
  - **Parameters**
    * `metric` – The distance metric between bags considered by the classifier.

16.7.7 Methods

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **getAddFeatures**

  ```
  public boolean getAddFeatures()
  ```

  - **Description**
    Gets the value of addFeatures. If false IBLR-ML is used. If true, IBLR-ML+ is used.
  - **Returns** – The value of addFeatures.

- **setAddFeatures**

  ```
  public void setAddFeatures(boolean addFeatures)
  ```

  - **Description**
    Sets the value of AddFeatures. If false IBLR-ML is used. If true, IBLR-ML+ is used.
  - **Parameters**
    * `addFeatures` – The new value of addFeatures.

16.7.8 Members inherited from class MultiInstanceMultiLabelKNN

`miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN` (in 16.10, page 233)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public void` **configure**`(org.apache.commons.configuration2.Configuration` **configuration**`)`
- `public MultiLabelKNN` **getClassifier**`()`
- `public DistanceFunction` **getMetric**`()`
- `public int` **getNumOfNeighbours**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **metric**
- `protected` **numOfNeighbours**
- `private static final` **serialVersionUID**
- `public void` **setClassifier**`(mulan.classifier.lazy.MultiLabelKNN` **classifier**`)`
- `public void` **setMetric**`(weka.core.DistanceFunction` **metric**`)`
- `public void` **setnumOfNeighbours**`(int` **numOfNeighbours**`)`

16.7.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`

- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

16.8 Class MIMLkNN

Class implementing the MIMLkNN algorithm for MIML data. For more information, see *Zhang, M. L. (2010, October). A k-nearest neighbor based multi-instance multi-label learning algorithm. In 2010 22nd IEEE International Conference on Tools with Artificial Intelligence (Vol.2, pp. 207-212). IEEE.*

16.8.1 Declaration

```
public class MIMLkNN
 extends miml.classifiers.miml.MIMLClassifier
```

16.8.2 Field summary

d_size Dataset size (number of bags).
dataset MIML data.
distance_matrix Distance matrix between dataset's instances.
metric Metric for measure the distance between bags.
num_citers Number of citers.
num_references Number of references.
phi_matrix The phi matrix.
ref_matrix Instances' references matrix.
serialVersionUID Generated Serial version UID.
t_matrix The t matrix.
weights_matrix Weights matrix.

16.8.3 Constructor summary

MIMLkNN() No-argument constructor for xml configuration.
MIMLkNN(IDistance) Instantiates a new MIMLkNN with values by default except distance metric.
MIMLkNN(int, int, IDistance) Basic constructor to initialize the classifier.

16.8.4 Method summary

buildInternal(MIMLInstances)
calculateBagReferences(int) Calculate the references of a bag specified by its index.
calculateDatasetDistances() Calculate the distances matrix of current data set with the metric assigned.
calculateRecordLabel(Integer[]) Calculate the number of times each label appears in the bag's neighborhood.
calculateReferenceMatrix() Calculate the references matrix.
configure(Configuration)
getBagLabels(int) Gets the labels of specified bag.
getCiters(int) Calculate and return the citers of a bag specified by its index.
getNumCiters() Returns the number of citers considered to estimate the class prediction of tests bags.
getNumReferences() Returns the number of references considered to estimate the class prediction of tests bags.
getReferences(int) Gets the references of a specified bag.
getUnionNeighbours(int) Gets the union of references and citers (without repetitions) of the bag specified.
getWeightsMatrix() Calculate the weights matrix used for prediction.
linearClassifier(double[], double[]) Classifier that determines the labels associated with an example.
makePredictionInternal(MIMLBag)
setNumCiters(int) Sets the number of citers considered to estimate the class prediction of tests bags.
setNumReferences(int) Sets the number of references considered to estimate the class prediction of tests bags.

16.8.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.
- `protected int` **num_citers**
  - Number of citers.
- `protected int` **num_references**
  - Number of references.
- `protected miml.core.distance.IDistance` **metric**
  - Metric for measure the distance between bags.
- `protected miml.data.MIMLInstances` **dataset**
  - MIML data.
- `int` **d_size**

  - Dataset size (number of bags).

- `protected double[][]` **distance_matrix**
  - Distance matrix between dataset's instances.
- `protected int[][]` **ref_matrix**
  - Instances' references matrix.
- `protected double[][]` **weights_matrix**
  - Weights matrix.
- `protected double[][]` **t_matrix**
  - The t matrix.
- `protected double[][]` **phi_matrix**
  - The phi matrix.

16.8.6 Constructors

- **MIMLkNN**

  ```
  public MIMLkNN()
  ```

  - **Description**
    No-argument constructor for xml configuration.

- **MIMLkNN**

  ```
  public MIMLkNN(miml.core.distance.IDistance metric)
  ```

  - **Description**
    Instantiates a new MIMLkNN with values by default except distance metric.
  - **Parameters**
    * `metric` – The metric used by the algorithm to measure the distance.

- **MIMLkNN**

  ```
  public MIMLkNN(int num_references ,int num_citers ,miml.core.
      distance.IDistance metric)
  ```

  - **Description**
    Basic constructor to initialize the classifier.
  - **Parameters**
    * `num_references` – The number of references considered by the algorithm.
    * `num_citers` – The number of citers considered by the algorithm.
    * `metric` – The metric used by the algorithm to measure the distance.

16.8.7 Methods

- **buildInternal**

  ```
  protected void buildInternal(miml.data.MIMLInstances trainingSet
      ) throws java.lang.Exception
  ```

  - **See also**
    - `miml.classifiers.miml.MIMLClassifier.buildInternal(MIMLInstances)`

- **calculateBagReferences**

  ```
  protected int[] calculateBagReferences(int indexBag) throws java
      .lang.Exception
  ```

  - **Description**
    Calculate the references of a bag specified by its index. It's necessary calculate the distance matrix previously.
  - **Parameters**
    - `indexBag` – The index bag.
  - **Returns** – The references' indices of the bag.
  - **Throws**
    - `java.lang.Exception` – A exception.

- **calculateDatasetDistances**

  ```
  protected void calculateDatasetDistances() throws java.lang.
      Exception
  ```

  - **Description**
    Calculate the distances matrix of current data set with the metric assigned.
  - **Throws**
    - `java.lang.Exception` – The exception.

- **calculateRecordLabel**

  ```
  protected double[] calculateRecordLabel(java.lang.Integer[]
      indices)
  ```

  - **Description**
    Calculate the number of times each label appears in the bag's neighborhood.
  - **Parameters**

    * `indices` – The neighboor's indices.
  - **Returns** – The labels' record.

- **calculateReferenceMatrix**

```
protected void calculateReferenceMatrix() throws java.lang.
    Exception
```

  - **Description**
    Calculate the references matrix.
  - **Throws**
    * `java.lang.Exception` – the exception

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **getBagLabels**

```
protected double[] getBagLabels(int bagIndex)
```

  - **Description**
    Gets the labels of specified bag.
  - **Parameters**
    * `bagIndex` – The bag index.
  - **Returns** – The bag labels.

- **getCiters**

```
protected int[] getCiters(int indexBag)
```

  - **Description**
    Calculate and return the citers of a bag specified by its index. It's necessary calculate the distance matrix first.
  - **Parameters**
    * `indexBag` – The index bag.
  - **Returns** – The bag's citers.

- **getNumCiters**

```
public int getNumCiters()
```

- **Description**
  Returns the number of citers considered to estimate the class prediction of tests bags.
- **Returns** – The number of citers.

• **getNumReferences**

```
public int getNumReferences()
```

- **Description**
  Returns the number of references considered to estimate the class prediction of tests bags.
- **Returns** – The number of references.

• **getReferences**

```
protected int[] getReferences(int indexBag)
```

- **Description**
  Gets the references of a specified bag.
- **Parameters**
  * `indexBag` – The index bag.
- **Returns** – The bag's references.

• **getUnionNeighbours**

```
protected java.lang.Integer[] getUnionNeighbours(int indexBag)
```

- **Description**
  Gets the union of references and citers (without repetitions) of the bag specified.
- **Parameters**
  * `indexBag` – The index bag.
- **Returns** – Ihe union of references and citers.

• **getWeightsMatrix**

```
protected double[][] getWeightsMatrix()
```

- **Description**
  Calculate the weights matrix used for prediction.

– **Returns** – The weights matrix.

- **linearClassifier**

```
protected boolean linearClassifier(double[] weights,double[]
    record)
```

– **Description**
Classifier that determines the labels associated with an example. A linear classifier uses the label counting vector of the example and the weight vector corresponding to the label,

– **Parameters**
  * `weights` – The weights correspondent to the label.
  * `record` – The labels' record of bag's neighbor to be predicted.

– **Returns** – True, if belong to a determinate class, false if not.

- **makePredictionInternal**

```
protected abstract mulan.classifier.MultiLabelOutput
    makePredictionInternal(miml.data.MIMLBag instance) throws
    java.lang.Exception, mulan.classifier.InvalidDataException
```

– **Description copied from miml.classifiers.miml.MIMLClassifier** (**in** 10.2, **page** 141)
Learner specific implementation for predicting on specified data based on trained model. This method is called from `makePrediction(Instance)` which guards for model initialization and apply common handling/behavior.

– **Parameters**
  * `instance` – The data instance to predict on.

– **Returns** – The output of the learner for the given instance.

– **Throws**
  * `java.lang.Exception` – If an error occurs while making the prediction.
  * `mulan.classifier.InvalidDataException` – If specified instance data is invalid and can not be processed by the learner.

- **setNumCiters**

```
public void setNumCiters(int numCiters)
```

– **Description**
Sets the number of citers considered to estimate the class prediction of tests bags.

– **Parameters**

    * `numCiters` – The new number of citers.

- **setNumReferences**

  ```
  public void setNumReferences(int numReferences)
  ```

  – **Description**

    Sets the number of references considered to estimate the class prediction of tests bags.

  – **Parameters**

    * `numReferences` – The new number of references.

16.8.8 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

16.9 Class MIMLMAPkNN

MIMLMAPkNN is the adaptation to the MIML framework of the MLkNN[1] multi-label algorithm. To perform this adaptation, MIMLMAPkNN maintains the treatment of labels of MLkNN but uses a multi-instance measure of distance. *[1] Min-Ling Zhang, Zhi-Hua Zhou (2007). ML-KNN: A lazy learning approach to multi-label learning. Pattern Recogn.. 40(7):2038–2048.*

16.9.1 Declaration

```
public class MIMLMAPkNN
 extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

16.9.2 Field summary

serialVersionUID Generated Serial version UID.
smooth Smooth factor

16.9.3 Constructor summary

MIMLMAPkNN() No-arg constructor for xml configuration
MIMLMAPkNN(int, double, MIMLDistanceFunction) A constructor that sets the number of neighbours and the value of smooth.
MIMLMAPkNN(int, MIMLDistanceFunction) A constructor that sets the number of neighbours.
MIMLMAPkNN(MIMLDistanceFunction) Default constructor.

16.9.4 Method summary

configure(Configuration)
getSmooth() Gets the smooth factor considered by the classifier.
setSmooth(double) Sets the smooth factor considered by the classifier.

16.9.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.
- `protected double` **smooth**
  - Smooth factor

16.9.6 Constructors

- **MIMLMAPkNN**

  ```
  public MIMLMAPkNN()
  ```

  - **Description**
    No-arg constructor for xml configuration

- **MIMLMAPkNN**

  ```
  public MIMLMAPkNN(int numOfNeighbours,double smooth,
      MIMLDistanceFunction metric)
  ```

- **Description**
  A constructor that sets the number of neighbours and the value of smooth.
- **Parameters**
  - `metric` – The distance metric between bags considered by the classifier.
  - `numOfNeighbours` – The number of neighbours.
  - `smooth` – The smooth factor.

- **MIMLMAPkNN**

```
public MIMLMAPkNN(int numOfNeighbours, MIMLDistanceFunction
    metric)
```

- **Description**
  A constructor that sets the number of neighbours.
- **Parameters**
  - `metric` – The distance metric between bags considered by the classifier.
  - `numOfNeighbours` – The number of neighbours.

- **MIMLMAPkNN**

```
public MIMLMAPkNN(MIMLDistanceFunction metric)
```

- **Description**
  Default constructor.
- **Parameters**
  - `metric` – The distance metric between bags considered by the classifier.

16.9.7 Methods

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **getSmooth**

```
public double getSmooth()
```

- **Description**
  Gets the smooth factor considered by the classifier.
- **Returns** – the smooth factor

- **setSmooth**

  ```
  public void setSmooth(double smooth)
  ```

  - **Description**
    Sets the smooth factor considered by the classifier.
  - **Parameters**
    * `smooth` – the new smooth factor

16.9.8 Members inherited from class MultiInstanceMultiLabelKNN

`miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN` (in 16.10, page 233)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public void` **configure**`(org.apache.commons.configuration2.Configuration` **configuration**`)`
- `public MultiLabelKNN` **getClassifier**`()`
- `public DistanceFunction` **getMetric**`()`
- `public int` **getNumOfNeighbours**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **metric**
- `protected` **numOfNeighbours**
- `private static final` **serialVersionUID**
- `public void` **setClassifier**`(mulan.classifier.lazy.MultiLabelKNN` **classifier**`)`
- `public void` **setMetric**`(weka.core.DistanceFunction` **metric**`)`
- `public void` **setnumOfNeighbours**`(int` **numOfNeighbours**`)`

16.9.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

16.10 Class MultiInstanceMultiLabelKNN

Wrapper for class MultiLabelKNN of Mulan to work with MIML data

16.10.1 Declaration

```
public abstract class MultiInstanceMultiLabelKNN
 extends miml.classifiers.miml.MIMLClassifier
```

16.10.2 All known subclasses

MIMLMAPkNN (in 16.9, page 229), MIMLIBLR (in 16.7, page 219), MIMLFuzzykNN (in 16.5, page 215), MIMLDGC (in 16.3, page 208), MIMLBRkNN (in 16.2, page 204), DMIMLkNN (in 16.1, page 201)

16.10.3 Field summary

classifier Mulan MultiLabelKNN classifier.
metric Metric for measure the distance between bags.
numOfNeighbours Number of neighbours used in the k-nearest neighbor algorithm.
serialVersionUID For serialization.

16.10.4 Constructor summary

MultiInstanceMultiLabelKNN() No-arg constructor for xml configuration
MultiInstanceMultiLabelKNN(MIMLDistanceFunction) Constructor to initialize the classifier.
MultiInstanceMultiLabelKNN(MIMLDistanceFunction, int) Constructor to initialize the classifier.

16.10.5 Method summary

buildInternal(MIMLInstances)
configure(Configuration)
getClassifier()
getMetric() Gets the distance metric considered by the classifier.
getNumOfNeighbours() Gets the number of neigbors considered by the classifier.
makePredictionInternal(MIMLBag)
setClassifier(MultiLabelKNN)
setMetric(DistanceFunction) Sets the distance metric considered by the classifier.
setnumOfNeighbours(int) Sets the number of neigbors considered by the classifier.

16.10.6 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `protected int` **numOfNeighbours**
  - Number of neighbours used in the k-nearest neighbor algorithm.
- `protected MIMLDistanceFunction` **metric**
  - Metric for measure the distance between bags.
- `protected mulan.classifier.lazy.MultiLabelKNN` **classifier**
  - Mulan MultiLabelKNN classifier.

16.10.7 Constructors

- **MultiInstanceMultiLabelKNN**

  ```
  public MultiInstanceMultiLabelKNN()
  ```

  - **Description**
    No-arg constructor for xml configuration
- **MultiInstanceMultiLabelKNN**

  ```
  public MultiInstanceMultiLabelKNN(MIMLDistanceFunction metric)
  ```

  - **Description**
    Constructor to initialize the classifier. It sets the numberOfNeighbours to 10
  - **Parameters**
    * `metric` – The metric used by the algorithm to measure the distance between bags.
- **MultiInstanceMultiLabelKNN**

  ```
  public MultiInstanceMultiLabelKNN(MIMLDistanceFunction metric,
      int numOfNeighbours)
  ```

  - **Description**
    Constructor to initialize the classifier. It sets the numOfNeighbours to 10
  - **Parameters**
    * `metric` – The metric used by the algorithm to measure the distance between bags.
    * `numOfNeighbours` – The number of neighbours.

16.10.8 Methods

- **buildInternal**

```
protected abstract void buildInternal(miml.data.MIMLInstances
    trainingSet) throws java.lang.Exception
```

  - **Description copied from miml.classifiers.miml.MIMLClassifier** (**in** 10.2, **page** 141)
    Learner specific implementation of building the model from `MultiLabelInstances` training data set. This method is called from `build(MultiLabelInstances)` method, where behavior common across all learners is applied.
  - **Parameters**
    * `trainingSet` – The training data set.
  - **Throws**
    * `java.lang.Exception` – if learner model was not created successfully.

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **getClassifier**

```
public mulan.classifier.lazy.MultiLabelKNN getClassifier()
```

- **getMetric**

```
public weka.core.DistanceFunction getMetric()
```

  - **Description**
    Gets the distance metric considered by the classifier.
  - **Returns** – The distance metric.

- **getNumOfNeighbours**

```
public int getNumOfNeighbours()
```

  - **Description**
    Gets the number of neigbors considered by the classifier.
  - **Returns** – the number of neigbors

- **makePredictionInternal**

```
protected abstract mulan.classifier.MultiLabelOutput
    makePredictionInternal(miml.data.MIMLBag instance) throws
    java.lang.Exception, mulan.classifier.InvalidDataException
```

  - **Description copied from miml.classifiers.miml.MIMLClassifier (in 10.2, page 141)**
    Learner specific implementation for predicting on specified data based on trained model. This method is called from `makePrediction(Instance)` which guards for model initialization and apply common handling/behavior.
  - **Parameters**
    - `instance` – The data instance to predict on.
  - **Returns** – The output of the learner for the given instance.
  - **Throws**
    - `java.lang.Exception` – If an error occurs while making the prediction.
    - `mulan.classifier.InvalidDataException` – If specified instance data is invalid and can not be processed by the learner.

- **setClassifier**

```
public void setClassifier(mulan.classifier.lazy.MultiLabelKNN
    classifier)
```

- **setMetric**

```
public void setMetric(weka.core.DistanceFunction metric)
```

  - **Description**
    Sets the distance metric considered by the classifier.
  - **Parameters**
    - `metric` – The new distance metric.

- **setnumOfNeighbours**

```
public void setnumOfNeighbours(int numOfNeighbours)
```

  - **Description**
    Sets the number of neigbors considered by the classifier.
  - **Parameters**
    - `numOfNeighbours` – the new number of neigbors

16.10.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

Chapter 17

Package miml.classifiers.mi

Package Contents *Page*

17.1 Class MISMOWrapper

Wrapper for MISMO algorithm to work in MIML to MI classifiers.

17.1.1 Declaration

```
public class MISMOWrapper
 extends weka.classifiers.mi.MISMO
```

17.1.2 Field summary

serialVersionUID Generated Serial version UID.

17.1.3 Constructor summary

MISMOWrapper()

17.1.4 Method summary

distributionForInstance(Instance)

17.1.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.

17.1.6 Constructors

- **MISMOWrapper**

```
public MISMOWrapper()
```

17.1.7 Methods

- **distributionForInstance**

```
double[] distributionForInstance(weka.core.Instance arg0) throws
    java.lang.Exception
```

17.1.8 Members inherited from class MISMO

weka.classifiers.mi.MISMO

- public String **attributeNames()**
- public double **bias()**
- public void **buildClassifier**(weka.core.Instances **arg0**) throws java.lang.Exception
- public String **buildLogisticModelsTipText()**
- public String **checksTurnedOffTipText()**
- public String **classAttributeNames()**
- public String **cTipText()**
- public double **distributionForInstance**(weka.core.Instance **arg0**) throws java.lang.Exception
- public String **epsilonTipText()**
- public static final **FILTER_NONE**
- public static final **FILTER_NORMALIZE**
- public static final **FILTER_STANDARDIZE**
- public String **filterTypeTipText()**
- public boolean **getBuildLogisticModels()**
- public double **getC()**
- public Capabilities **getCapabilities()**
- public boolean **getChecksTurnedOff()**
- public double **getEpsilon()**
- public SelectedTag **getFilterType()**
- public Kernel **getKernel()**
- public boolean **getMinimax()**
- public Capabilities **getMultiInstanceCapabilities()**
- public int **getNumFolds()**
- public String **getOptions()**
- public int **getRandomSeed()**
- public String **getRevision()**
- public TechnicalInformation **getTechnicalInformation()**
- public double **getToleranceParameter()**
- public String **globalInfo()**
- public String **kernelTipText()**
- public Enumeration **listOptions()**
- protected **m_C**
- protected **m_checksTurnedOff**
- protected **m_classAttribute**
- protected **m_classifiers**

- `protected` **m_classIndex**
- `protected static` **m_Del**
- `protected` **m_eps**
- `protected` **m_Filter**
- `protected` **m_filterType**
- `protected` **m_fitLogisticModels**
- `protected` **m_kernel**
- `protected` **m_minimax**
- `protected` **m_Missing**
- `protected` **m_NominalToBinary**
- `protected` **m_numFolds**
- `protected` **m_randomSeed**
- `protected` **m_tol**
- `public static void` **main**`(java.lang.String[]` **arg0**`)`
- `public String` **minimaxTipText**`()`
- `public int` **numClassAttributeValues**`()`
- `public String` **numFoldsTipText**`()`
- `public double` **pairwiseCoupling**`(double[][]` **arg0**`, double[][]` **arg1**`)`
- `public String` **randomSeedTipText**`()`
- `static final` **serialVersionUID**
- `public void` **setBuildLogisticModels**`(boolean` **arg0**`)`
- `public void` **setC**`(double` **arg0**`)`
- `public void` **setChecksTurnedOff**`(boolean` **arg0**`)`
- `public void` **setEpsilon**`(double` **arg0**`)`
- `public void` **setFilterType**`(weka.core.SelectedTag` **arg0**`)`
- `public void` **setKernel**`(weka.classifiers.functions.supportVector.Kernel` **arg0**`)`
- `public void` **setMinimax**`(boolean` **arg0**`)`
- `public void` **setNumFolds**`(int` **arg0**`)`
- `public void` **setOptions**`(java.lang.String[]` **arg0**`) throws java.lang.Exception`
- `public void` **setRandomSeed**`(int` **arg0**`)`
- `public void` **setToleranceParameter**`(double` **arg0**`)`
- `public int` **sparseIndices**`()`
- `public double` **sparseWeights**`()`
- `public static final` **TAGS_FILTER**
- `public String` **toleranceParameterTipText**`()`
- `public String` **toString**`()`
- `public void` **turnChecksOff**`()`
- `public void` **turnChecksOn**`()`

17.1.9 Members inherited from class AbstractClassifier

`weka.classifiers.AbstractClassifier`

- `public double` **classifyInstance**`(weka.core.Instance` **arg0**`) throws java.lang.Exception`
- `public String` **debugTipText**`()`
- `public double` **distributionForInstance**`(weka.core.Instance` **arg0**`) throws java.lang.Exception`
- `public static Classifier` **forName**`(java.lang.String` **arg0**`, java.lang.String[]` **arg1**`) throws java.lang.Exception`
- `public Capabilities` **getCapabilities**`()`
- `public boolean` **getDebug**`()`
- `public String` **getOptions**`()`
- `public String` **getRevision**`()`
- `public Enumeration` **listOptions**`()`
- `protected` **m_Debug**
- `public static Classifier` **makeCopies**`(Classifier` **arg0**`, int` **arg1**`) throws java.lang.Exception`
- `public static Classifier` **makeCopy**`(Classifier` **arg0**`) throws java.lang.Exception`
- `public static void` **runClassifier**`(Classifier` **arg0**`, java.lang.String[]` **arg1**`)`
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **arg0**`)`
- `public void` **setOptions**`(java.lang.String[]` **arg0**`) throws java.lang.Exception`

Chapter 18

Package miml.classifiers.miml.mimlTOml

Package Contents *Page*

18.1 Class MIMLClassifierToML

Class implementing the transformation algorithm for MIML data to solve it with ML learning. For more information, see *Zhou, Z. H., & Zhang, M. L. (2007). Multi-instance multi-label learning with application to scene classification. In Advances in neural information processing systems (pp. 1609-1616).*

18.1.1 Declaration

```
public class MIMLClassifierToML
 extends miml.classifiers.miml.MIMLClassifier
```

18.1.2 Field summary

baseClassifier A Generic MultiLabel classifier.
mimlDataset The miml dataset.
removeFilter The filter that removes the bagId attribute
serialVersionUID Generated Serial version UID.
templateWithBagId An empty dataset used as template for prediction
transformationMethod The transform method.

18.1.3 Constructor summary

MIMLClassifierToML() No-argument constructor for xml configuration.

MIMLClassifierToML(MultiLabelLearner, MIMLtoML) Basic constructor to initialize the classifier.

18.1.4 Method summary

buildInternal(MIMLInstances)
configure(Configuration)
getBaseClassifier()
getRemoveFilter()
getTransformationMethod()
makePredictionInternal(MIMLBag)

18.1.5 Fields

- `private static final long` **serialVersionUID**
  - Generated Serial version UID.
- `protected mulan.classifier.MultiLabelLearner` **baseClassifier**
  - A Generic MultiLabel classifier.
- `protected miml.transformation.mimlTOml.MIMLtoML` **transformationMethod**
  - The transform method.
- `protected miml.data.MIMLInstances` **mimlDataset**
  - The miml dataset.
- `protected weka.filters.unsupervised.attribute.Remove` **removeFilter**
  - The filter that removes the bagId attribute
- `protected mulan.data.MultiLabelInstances` **templateWithBagId**
  - An empty dataset used as template for prediction

18.1.6 Constructors

- **MIMLClassifierToML**

```
public MIMLClassifierToML()
```

  - **Description**
    No-argument constructor for xml configuration.

- **MIMLClassifierToML**

```
public MIMLClassifierToML(mulan.classifier.MultiLabelLearner
    baseClassifier,miml.transformation.mimlTOml.MIMLtoML
    transformationMethod) throws java.lang.Exception
```

- **Description**

  Basic constructor to initialize the classifier.
- **Parameters**
  - `baseClassifier` – The base classification algorithm.
  - `transformationMethod` – Algorithm used as transformation method from MIML to ML.
- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

18.1.7 Methods

- **buildInternal**

```
protected abstract void buildInternal(miml.data.MIMLInstances
    trainingSet) throws java.lang.Exception
```

  - **Description copied from miml.classifiers.miml.MIMLClassifier** (**in 10.2, page 141**)

    Learner specific implementation of building the model from `MultiLabelInstances` training data set. This method is called from `build(MultiLabelInstances)` method, where behavior common across all learners is applied.
  - **Parameters**
    - `trainingSet` – The training data set.
  - **Throws**
    - `java.lang.Exception` – if learner model was not created successfully.

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **getBaseClassifier**

```
public mulan.classifier.MultiLabelLearner getBaseClassifier()
```

- **getRemoveFilter**

```
public weka.filters.unsupervised.attribute.Remove
    getRemoveFilter()
```

- **getTransformationMethod**

```
public miml.transformation.mimlTOml.MIMLtoML
    getTransformationMethod()
```

- **makePredictionInternal**

```
protected abstract mulan.classifier.MultiLabelOutput
    makePredictionInternal(miml.data.MIMLBag instance) throws
    java.lang.Exception, mulan.classifier.InvalidDataException
```

  - **Description copied from miml.classifiers.miml.MIMLClassifier** (**in** 10.2, **page** 141)
    Learner specific implementation for predicting on specified data based on trained model. This method is called from `makePrediction(Instance)` which guards for model initialization and apply common handling/behavior.
  - **Parameters**
    - `instance` – The data instance to predict on.
  - **Returns** – The output of the learner for the given instance.
  - **Throws**
    - `java.lang.Exception` – If an error occurs while making the prediction.
    - `mulan.classifier.InvalidDataException` – If specified instance data is invalid and can not be processed by the learner.

18.1.8 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

Chapter 19

Package miml.data.normalization

Package Contents *Page*

19.1 Class MinMaxNormalization

Class implementing min-max normalization for MIML datasets.

19.1.1 Declaration

```
public class MinMaxNormalization
 extends java.lang.Object
```

19.1.2 Field summary

Max Max, Min and Range values for features.
Min
nFeatures Number of features of the bags in the MIML dataset.
normalized Value indicating if the bag attributes of the dataset were normalized before calling normalize (e.g. the dataset does not need normalization).
Range

19.1.3 Constructor summary

MinMaxNormalization()

19.1.4 Method summary

getMax() Retuns an array with the maximum values for all bag attributes in the dataset.

getMin() Retuns an array with the minimum values for all bag attributes in the dataset.
getnFeatures() Retuns the number of bag attributes in the dataset.
getRange() Retuns an array with the range values (i.e. max-min) for all bag attributes in the dataset.
isNormalized() Returns true if the dataset does not need normalization.
normalize(MIMLInstances) Applies min-max normalization on a MIMLInstances dataset.
updateStats(MIMLInstances) Set the max and min values for all attributes in the bag.

19.1.5 Fields

- `protected double[]` **Max**
  - Max, Min and Range values for features.
- `protected double[]` **Min**
- `protected double[]` **Range**
- `int` **nFeatures**
  - Number of features of the bags in the MIML dataset.
- `boolean` **normalized**
  - Value indicating if the bag attributes of the dataset were normalized before calling normalize (e.g. the dataset does not need normalization).

19.1.6 Constructors

- **MinMaxNormalization**

```
public MinMaxNormalization()
```

19.1.7 Methods

- **getMax**

```
public double[] getMax()
```

  - **Description**
    Retuns an array with the maximum values for all bag attributes in the dataset. Requires a previous call of updateStats.
  - **Returns** – double[]

- **getMin**

```
public double[] getMin()
```

- **Description**
  Retuns an array with the minimum values for all bag attributes in the dataset. Requires a previous call of updateStats.
- **Returns** – double[]

• **getnFeatures**

```
public int getnFeatures()
```

- **Description**
  Retuns the number of bag attributes in the dataset. Requires a previous call of updateStats.
- **Returns** – int

• **getRange**

```
public double[] getRange()
```

- **Description**
  Retuns an array with the range values (i.e. max-min) for all bag attributes in the dataset. Requires a previous call of updateStats.
- **Returns** – double

• **isNormalized**

```
public boolean isNormalized()
```

- **Description**
  Returns true if the dataset does not need normalization. Requires a previous call of updateStats.
- **Returns** – boolean

• **normalize**

```
public void normalize(miml.data.MIMLInstances mimlDataSet)
    throws java.lang.Exception
```

- **Description**
  Applies min-max normalization on a MIMLInstances dataset. Given an attribute's value, x, the new x' value will be x' = (x-min(x))/(max(x)-min(x)). Thus every attribute's value is transformed into a decimal between 0 and 1.Before call this method the method update stats must be called to get the max and min values for attributes.

- **Parameters**
  - `mimlDataSet` – a dataset to normalize.
- **Throws**
  - `java.lang.Exception` – To be handled in upper level.

- **updateStats**

```
public void updateStats(miml.data.MIMLInstances mimlDataSet)
    throws java.lang.Exception
```

- **Description**

  Set the max and min values for all attributes in the bag. This method must be called before call normalized. If several datasets with the same structure are normalized at once (e.g. train and test or folds partitioned files), this method can be called for each dataset before normalization. Besides, if the method method detects that all the attributes are jet normalized, it sets the "normalized" property as true. `MinMaxNormalization norm = new MinMaxNormalization(); MIMLInstances mimlDataSet1 = new MIMLInstances("toy_train.arff", "toy.xml"); MIMLInstances mimlDataSet2 = new MIMLInstances("toy_test.arff", "toy.xml"); norm.updateStats(mimlDataSet1); norm.updateStats(mimlDataSet2); if (norm.isNormalized() == false) { norm.normalize(mimlDataSet1); norm.normalize(mimlDataSet2); }`
- **Parameters**
  - `mimlDataSet` – MIML dataset.
- **Throws**
  - `java.lang.Exception` – To be handled in upper level.

Chapter 20

Package miml.transformation.mimlTOml

Package Contents *Page*

20.1 Class ArithmeticTransformation

Class that performs an arithmetic transformation to convert a MIMLInstances class to MultiLabelInstances. This arithmetic transformation transforms each Bag into a single Instance being the value of each attribute the mean value of the instances in the bag.

20.1.1 Declaration

```
public class ArithmeticTransformation
 extends miml.transformation.mimlTOml.MIMLtoML
```

20.1.2 Field summary

serialVersionUID For serialization

20.1.3 Constructor summary

ArithmeticTransformation()
ArithmeticTransformation(MIMLInstances) Constructor.

20.1.4 Method summary

transformDataset()
transformDataset(MIMLInstances)
transformInstance(MIMLBag)
transformInstance(MIMLInstances, MIMLBag)

20.1.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization

20.1.6 Constructors

- **ArithmeticTransformation**

  ```
  public ArithmeticTransformation()
  ```

- **ArithmeticTransformation**

  ```
  public ArithmeticTransformation(miml.data.MIMLInstances dataset)
      throws java.lang.Exception
  ```

  - **Description**
    Constructor.
  - **Parameters**
    - `dataset` – MIMLInstances dataset.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

20.1.7 Methods

- **transformDataset**

```
public abstract mulan.data.MultiLabelInstances transformDataset
    () throws java.lang.Exception
```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances. To call this method is the dataset must be previously set eg. in the constructor.
  - **Returns** – MultiLabelInstances.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **transformDataset**

```
public abstract mulan.data.MultiLabelInstances transformDataset(
    miml.data.MIMLInstances dataset) throws java.lang.Exception
```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances.
  - **Parameters**
    * `dataset` – The dataset to be transformed
  - **Returns** – MultiLabelInstances.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **transformInstance**

```
public abstract weka.core.Instance transformInstance(miml.data.
    MIMLBag bag) throws java.lang.Exception
```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLBag` (in 7.1, page 96) into Instance.
  - **Parameters**
    * `bag` – The Bag to be transformed.
  - **Returns** – Instance
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **transformInstance**

```
public weka.core.Instance transformInstance(miml.data.
    MIMLInstances dataset,miml.data.MIMLBag bag) throws java.lang
    .Exception
```

20.1.8 Members inherited from class MIMLtoML

`miml.transformation.mimlTOml.MIMLtoML` (in 20.5, page 272)

- `protected` **dataset**
- `public static double` **minimax**`(weka.core.Instances` **data**`, int` **attIndex**`)`
- `protected void` **prepareTemplate**`() throws java.lang.Exception`
- `private static final` **serialVersionUID**
- `protected` **template**
- `public abstract MultiLabelInstances` **transformDataset**`() throws java.lang.Exception`
- `public abstract MultiLabelInstances` **transformDataset**`(miml.data.MIMLInstances` **dataset**`) throws java.lang.Exception`
- `public abstract Instance` **transformInstance**`(miml.data.MIMLBag` **bag**`) throws java.lang.Exception`
- `protected` **updatedLabelIndices**

20.2 Class GeometricTransformation

Class that performs a geometric transformation to convert a MIMLInstances class to MultiLabelInstances. Each Bag is transformed into a single Instance being the value of each attribute the geometric centor of its max and min values computed as (min_value+max_value)/2.

20.2.1 Declaration

```
public class GeometricTransformation
 extends miml.transformation.mimlTOml.MIMLtoML
```

20.2.2 Field summary

serialVersionUID For serialization

20.2.3 Constructor summary

GeometricTransformation()
GeometricTransformation(MIMLInstances) Constructor

20.2.4 Method summary

transformDataset()
transformDataset(MIMLInstances)
transformInstance(MIMLBag)
transformInstance(MIMLInstances, MIMLBag)

20.2.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization

20.2.6 Constructors

- **GeometricTransformation**

```
public GeometricTransformation() throws java.lang.Exception
```

- **GeometricTransformation**

```
public GeometricTransformation(miml.data.MIMLInstances dataset)
    throws java.lang.Exception
```

  - **Description**
    Constructor
  - **Parameters**
    - `dataset` – MIMLInstances dataset.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

20.2.7 Methods

- **transformDataset**

```
public abstract mulan.data.MultiLabelInstances transformDataset
    () throws java.lang.Exception
```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances. To call this method is the dataset must be previously set eg. in the constructor.
  - **Returns** – MultiLabelInstances.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **transformDataset**

```
public abstract mulan.data.MultiLabelInstances transformDataset(
    miml.data.MIMLInstances dataset) throws java.lang.Exception
```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances.
  - **Parameters**
    - `dataset` – The dataset to be transformed
  - **Returns** – MultiLabelInstances.

- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

- **transformInstance**

```
public abstract weka.core.Instance transformInstance(miml.data.
    MIMLBag bag) throws java.lang.Exception
```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLBag` (in 7.1, page 96) into Instance.
  - **Parameters**
    - `bag` – The Bag to be transformed.
  - **Returns** – Instance
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **transformInstance**

```
public weka.core.Instance transformInstance(miml.data.
    MIMLInstances dataset,miml.data.MIMLBag bag) throws java.lang
    .Exception
```

20.2.8 Members inherited from class MIMLtoML

`miml.transformation.mimlTOml.MIMLtoML` (in 20.5, page 272)

- `protected` **dataset**
- `public static double` **minimax**`(weka.core.Instances` **data**`, int` **attIndex**`)`
- `protected void` **prepareTemplate**`() throws java.lang.Exception`
- `private static final` **serialVersionUID**
- `protected` **template**
- `public abstract MultiLabelInstances` **transformDataset**`() throws java.lang.Exception`
- `public abstract MultiLabelInstances` **transformDataset**`(miml.data.MIMLInstances` **dataset**`) throws java.lang.Exception`
- `public abstract Instance` **transformInstance**`(miml.data.MIMLBag` **bag**`) throws java.lang.Exception`
- `protected` **updatedLabelIndices**

20.3 Class KMeansTransformation

Class implementing the kmeans-based transformation described in [1] to transform an MIML problem to ML. [1] *Li, Y. F., Hu, J. H., Jiang, Y., and Zhou, Z. H. (2012). Towards discovering what patterns trigger what labels. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 26, No. 1, pp. 1012-1018).* This class requires method transformDataset to have been executed before executing transformInstance method.

20.3.1 Declaration

```
public class KMeansTransformation
 extends miml.transformation.mimlTOml.MIMLtoML
```

20.3.2 Field summary

clusterer Clusterer.
clusteringDone Whether the clustering step has been executed or not.
delta The delta value for each cluster obtained as the average distance between instances in each cluster
dfunc
numClusters The number of clusters.
percentage If it is different to -1 this value represent that the number of clusters will be a percentage of the number of training bags in the dataset.
prototypes Clustering prototypes obtained each one as the nearest instance to each centroid.
seed The seed for kmeans clustering.
serialVersionUID For serialization.

20.3.3 Constructor summary

KMeansTransformation() Constructor.
KMeansTransformation(MIMLInstances) Constructor.
KMeansTransformation(MIMLInstances, SimpleKMeans) Constructor.

20.3.4 Method summary

clusterAssignment(double[][]) Computes a vector of nInstances with the index of the cluster assigned to each instance.
computeDelta(int[], Instances) Computes the delta value for each cluster that is used for similarity computation.
computeDistanceMatrix(Instances, Instances) Computes the distance matrix between centroids and single instances used for clustering.
computeIndexPrototypes(double[][]) Computes a vector of nClusters elements with the index of the prototypes obtained as the closest instance to each centroid.
configureClusterer() Determines the number of cluster depending on the values of the properties percentage and numClusters.
getNumClusters() Returns the number of clusters.
getPercentage()
getSeed() Returns the value of the seed of the clusterer.
prepareTemplate()
setNumClusters(int) Sets the number of clusters to perform clustering in both the transformer and in the clusterer.
setPercentage(double)
setSeed(int) Sets the value of the seed used for clustering in both the transformer and in the clusterer.

similarity(Instance, MIMLBag, double) Computes similarity between a centroid, represented by a single instance, and a bag.
transformDataset()
transformDataset(MIMLInstances)
transformInstance(MIMLBag)

20.3.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `protected weka.clusterers.SimpleKMeans` **clusterer**
  - Clusterer.
- `protected double` **percentage**
  - If it is different to -1 this value represent that the number of clusters will be a percentage of the number of training bags in the dataset. For instance 0.2 represents that the number of clusters is the 20% of the number of training bags, 0.45 a 45%, and so on. If this value is -1 the number of clusters to consider is represented by numClusters property. If the number of clusters is not set neither by percentage nor by the numClusters property, it will be considered by default a 50% of the number of training bags in the dataset. If both the percentage and the numClusters are set, the percentage will be applied.
- `protected int` **numClusters**
  - The number of clusters.
- `protected int` **seed**
  - The seed for kmeans clustering. By default 1.
- `protected boolean` **clusteringDone**
  - Whether the clustering step has been executed or not.
- `protected weka.core.Instances` **prototypes**
  - Clustering prototypes obtained each one as the nearest instance to each centroid.
- `protected double[]` **delta**
  - The delta value for each cluster obtained as the average distance between instances in each cluster
- `protected weka.core.EuclideanDistance` **dfunc**

20.3.6 Constructors

- **KMeansTransformation**

  ```
  public KMeansTransformation()
  ```

  - **Description**
    Constructor.

- **KMeansTransformation**

  ```
  public KMeansTransformation(miml.data.MIMLInstances dataset)
      throws java.lang.Exception
  ```

  - **Description**
    Constructor. Uses the same default number of clusters as KiSar: 50% of number of bags
  - **Parameters**
    * `dataset` – MIMLInstances dataset.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **KMeansTransformation**

  ```
  public KMeansTransformation(miml.data.MIMLInstances dataset,weka
      .clusterers.SimpleKMeans clusterer) throws java.lang.
      Exception
  ```

  - **Description**
    Constructor.
  - **Parameters**
    * `dataset` – MIMLInstances dataset.
    * `clusterer` – An instance of KMedoids.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

20.3.7 Methods

- **clusterAssignment**

  ```
  protected int[] clusterAssignment(double[][] distanceMatrix)
  ```

- **Description**

  Computes a vector of nInstances with the index of the cluster assigned to each instance.
- **Parameters**
  - `distanceMatrix` – A matrix of nInstancesxnClusters with the distance between centroids and single-instances used for clustering . Matrix[i][k] is the distance from instance i to centroid k.
- **Returns** – A vector with the index of the cluster assigned to each instance. This vector is obtained as the index of the minimum column of each row.

• **computeDelta**

```
protected double[] computeDelta(int[] clusterAssignment,weka.
    core.Instances singleInstances)
```

- **Description**

  Computes the delta value for each cluster that is used for similarity computation. This value is computed as the average distance between all pair of instances in each cluster.
- **Parameters**
  - `clusterAssignment` – A vector of nInstances elements with the indices of the clusters assigned to each one.
  - `singleInstances` – The instances used for clustering.
- **Returns** – A vector of nClusters with the delta value for each cluster.

• **computeDistanceMatrix**

```
protected double[][] computeDistanceMatrix(weka.core.Instances
    centroids,weka.core.Instances singleInstances)
```

- **Description**

  Computes the distance matrix between centroids and single instances used for clustering.
- **Parameters**
  - `centroids` – The centroids obtained by kmeans clustering.
  - `singleInstances` – The single-instance instances used for clustering.
- **Returns** – A matrix of double in which matrix[i][k] stores the distance from instance i to centroid k.

• **computeIndexPrototypes**

```
protected int[] computeIndexPrototypes(double[][] distanceMatrix
    ) throws java.lang.Exception
```

- **Description**
  Computes a vector of nClusters elements with the index of the prototypes obtained as the closest instance to each centroid.
- **Parameters**
  * `distanceMatrix` – A matrix of nInstancesxnClusters with the distance between centroids and single-instances used for clustering . Matrix[i][k] is the distance from instance i to centroid k.
- **Returns** – A vector with the index of the prototypes in the dataset of single-instances used for clustering. This vector is obtained as the index of the minimum row of each column.
- **Throws**
  * `java.lang.Exception` – To be handled in an upper level.

• **configureClusterer**

```
void configureClusterer() throws java.lang.Exception
```

- **Description**
  Determines the number of cluster depending on the values of the properties percentage and numClusters. Sets the number of clusters and the seed for clustering.
- **Throws**
  * `java.lang.Exception` – To be handled in an upper level.

• **getNumClusters**

```
public int getNumClusters() throws java.lang.Exception
```

- **Description**
  Returns the number of clusters.
- **Returns** – Returns the number of clusters to perform clustering.
- **Throws**
  * `java.lang.Exception` – To be handled in an upper level.

• **getPercentage**

```
public double getPercentage()
```

• **getSeed**

```
public int getSeed()
```

- **Description**
  Returns the value of the seed of the clusterer.
- **Returns** – int

- **prepareTemplate**

```
protected void prepareTemplate() throws java.lang.Exception
```

- **Description copied from MIMLtoML (in 20.5, page 272)**
  Prepares a template to perform the transformation from MIMLInstances to MultiLabelInstances. This template includes: the bag label attribute, all attributes in the relational attribute as independent attributes and label attributes. For instance, in the relation above, the resulting template is showed. @relation toy
  @attribute id {bag1,bag2}
  @attribute bag relational
  @attribute f1 numeric
  @attribute f2 numeric
  @attribute f3 numeric
  @end bag
  @attribute label1 {0,1}
  @attribute label2 {0,1}
  @attribute label3 {0,1}
  @attribute label4 {0,1}
  @relation template
  @attribute id {bag1,bag2}
  @attribute f1 numeric
  @attribute f2 numeric
  @attribute f3 numeric
  * @attribute label1 {0,1}
  @attribute label2 {0,1}
  @attribute label3 {0,1}
  @attribute label4 {0,1}

- **Throws**
  * `java.lang.Exception` – To be handled in an upper level.

- **setNumClusters**

```
public void setNumClusters(int numClusters) throws java.lang.
    Exception
```

- **Description**

  Sets the number of clusters to perform clustering in both the transformer and in the clusterer. If the clusterer is null the value of the property is only set in the transformer and the transformDataset method will establish this numClusters value in the clusterer after creating it.
- **Parameters**
  - `numClusters` – A number of clusters.
- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

• **setPercentage**

```
public void setPercentage(double percentage)
```

• **setSeed**

```
public void setSeed(int seed)
```

- **Description**

  Sets the value of the seed used for clustering in both the transformer and in the clusterer. If the clusterer is null the value of the property is only set in the transformer and the transformDataset method will establish this seed value in the clusterer after creating it.
- **Parameters**
  - `seed` – The seed

• **similarity**

```
protected double similarity(weka.core.Instance centroid,miml.
    data.MIMLBag bag,double delta_k) throws java.lang.Exception
```

- **Description**

  Computes similarity between a centroid, represented by a single instance, and a bag. The value is computed as Gaussian distance.
- **Parameters**
  - `centroid` – A centroid.
  - `bag` – A bag.
  - `delta_k` – A vector with a delta value for each centroid.
- **Returns** – The similarity, a value normalized to [0,1].
- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

- **transformDataset**

  ```
  public abstract mulan.data.MultiLabelInstances transformDataset
      () throws java.lang.Exception
  ```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances. To call this method is the dataset must be previously set eg. in the constructor.
  - **Returns** – MultiLabelInstances.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **transformDataset**

  ```
  public abstract mulan.data.MultiLabelInstances transformDataset(
      miml.data.MIMLInstances dataset) throws java.lang.Exception
  ```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances.
  - **Parameters**
    * `dataset` – The dataset to be transformed
  - **Returns** – MultiLabelInstances.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **transformInstance**

  ```
  public abstract weka.core.Instance transformInstance(miml.data.
      MIMLBag bag) throws java.lang.Exception
  ```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLBag` (in 7.1, page 96) into Instance.
  - **Parameters**
    * `bag` – The Bag to be transformed.
  - **Returns** – Instance
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

20.3.8 Members inherited from class MIMLtoML

`miml.transformation.mimlTOml.MIMLtoML` (in 20.5, page 272)

- `protected` **dataset**
- `public static double` **minimax**`(weka.core.Instances` **data**`, int` **attIndex**`)`
- `protected void` **prepareTemplate**`() throws java.lang.Exception`
- `private static final` **serialVersionUID**
- `protected` **template**
- `public abstract MultiLabelInstances` **transformDataset**`() throws java.lang.Exception`
- `public abstract MultiLabelInstances` **transformDataset**`(miml.data.MIMLInstances` **dataset**`) throws java.lang.Exception`
- `public abstract Instance` **transformInstance**`(miml.data.MIMLBag` **bag**`) throws java.lang.Exception`
- `protected` **updatedLabelIndices**

20.4 Class MedoidTransformation

Class implementing the medoid-based transformation described in [1] to transform an MIML problem to ML. [1] *Zhou, Z. H., Zhang, M. L., Huang, S. J., & Li, Y. F. (2012). Multi-instance multi-label learning. Artificial Intelligence, 176(1), 2291-2320.* This class requires method transformDataset to have been executed before executing transformInstance method.

20.4.1 Declaration

```
public class MedoidTransformation
 extends miml.transformation.mimlTOml.MIMLtoML
```

20.4.2 Field summary

clusterer Clusterer.
clusteringDone Whether the clustering step has been executed or not.
normalize True if the resulting transformed dataset will be normalized to (0,1) with min-max normalization.
numClusters The number of clusters for kmedoids.
percentage If it is different to -1 this value represent that the number of clusters will be a percentage of the number of bags of the dataset.
seed The seed for kmedoids clustering.
serialVersionUID For serialization

20.4.3 Constructor summary

MedoidTransformation() Constructor.
MedoidTransformation(MIMLInstances) Constructor.
MedoidTransformation(MIMLInstances, double) Constructor.
MedoidTransformation(MIMLInstances, IDistance, int) Constructor.
MedoidTransformation(MIMLInstances, int) Constructor.
MedoidTransformation(MIMLInstances, KMedoids, boolean) Constructor.

20.4.4 Method summary

clusteringStep()
configureClusterer() Determines the number of cluster depending on the values of the properties percentage and numClusters.
getDistanceFunction() Returns the distance function used for clustering.
getMaxIterations() Gets the maximum number of iterations used by the clusterer.
getNormalize() Returns the value of the property normalize.
getNumClusters() Returns the number of clusters.
getPercentage() Gets the value of the percentage property.
normalize(MultiLabelInstances) Normalizes a multi-label dataset performing min-max normalization.
prepareTemplate()
setDistanceFunction(IDistance) Sets the distance function to use for clustering.
setMaxIterations(int) Sets the maximum number of iterations for clustering.
setNormalize(Boolean) Sets the property normalized.
setNumClusters(int) Sets the number of clusters to perform clustering in both the transformer and in the clusterer.
setPercentage(double) Sets the value of the percentage property.
setSeed(int) Sets the value of the seed used for clustering in both the transformer and in the clusterer.
transformDataset()
transformDataset(MIMLInstances)
transformInstance(MIMLBag)

20.4.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization
- `protected miml.clusterers.KMedoids` **clusterer**
  - Clusterer.
- `protected java.lang.Boolean` **normalize**
  - True if the resulting transformed dataset will be normalized to (0,1) with min-max normalization. By default False. If a learning algorithm that uses a NormalizableDistance is going to be used after transformation, normalization is not needed.
- `protected double` **percentage**
  - If it is different to -1 this value represent that the number of clusters will be a percentage of the number of bags of the dataset. For instance 0.2 represents that the number of clusters is the 20% of the training bags, 0.45 a 45%, and so on. If this value is -1 the number of clusters to consider is represented by numClusters property. If the number of clusters is not set neither by percentage nor by the numClusters property, it will be considered by default a 20% of the number of training bags in the dataset. If both the percentage and the numClusters are set, the percentage will be applied.

- `protected int` **numClusters**
  - The number of clusters for kmedoids.
- `protected boolean` **clusteringDone**
  - Whether the clustering step has been executed or not.
- `protected int` **seed**
  - The seed for kmedoids clustering. By default 1.

20.4.6 Constructors

- **MedoidTransformation**

```
public MedoidTransformation()
```

  - **Description**
    Constructor.

- **MedoidTransformation**

```
public MedoidTransformation(miml.data.MIMLInstances dataset)
    throws java.lang.Exception
```

  - **Description**
    Constructor. Uses the same default number of clusters as MIMLSVM: 20% of number of bags
  - **Parameters**
    - `dataset` – MIMLInstances dataset.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **MedoidTransformation**

```
public MedoidTransformation(miml.data.MIMLInstances dataset,
    double percentage) throws java.lang.Exception
```

  - **Description**
    Constructor.
  - **Parameters**
    - `dataset` – MIMLInstances dataset.
    - `percentage` – The number of clusters for kmedoids as a percentage of the number of bags. It is a value in (0,1). For instance, 0.2 is 20%.

- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

- **MedoidTransformation**

```
public MedoidTransformation(miml.data.MIMLInstances dataset,miml
    .core.distance.IDistance metric,int numClusters) throws java.
    lang.Exception
```

  - **Description**
    Constructor.
  - **Parameters**
    - `dataset` – MIMLInstances dataset.
    - `numClusters` – The number of clusters for kmedoids.
    - `metric` – The distance function to be used by kmedoids.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **MedoidTransformation**

```
public MedoidTransformation(miml.data.MIMLInstances dataset,int
    numClusters) throws java.lang.Exception
```

  - **Description**
    Constructor.
  - **Parameters**
    - `dataset` – MIMLInstances dataset.
    - `numClusters` – number of clusters for kmedoids.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **MedoidTransformation**

```
public MedoidTransformation(miml.data.MIMLInstances dataset,miml
    .clusterers.KMedoids kmedoids,boolean normalize) throws java.
    lang.Exception
```

  - **Description**
    Constructor.
  - **Parameters**
    - `dataset` – MIMLInstances dataset.

    * `kmedoids` – An instance of kmedoids.
    * `normalize` – If true, the resulting transformed dataset will be normalized to (0,1) with min-max normalization. If a learning algorithm that uses a NormalizableDistance is going to be used, normalization is not needed.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

20.4.7 Methods

- **clusteringStep**

```
protected void clusteringStep() throws java.lang.Exception
```

- **configureClusterer**

```
void configureClusterer() throws java.lang.Exception
```

  - **Description**
    Determines the number of cluster depending on the values of the properties percentage and numClusters.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **getDistanceFunction**

```
public miml.core.distance.IDistance getDistanceFunction()
```

  - **Description**
    Returns the distance function used for clustering.
  - **Returns** – The distance function used for clustering.

- **getMaxIterations**

```
public int getMaxIterations()
```

  - **Description**
    Gets the maximum number of iterations used by the clusterer.
  - **Returns** – The maximum number of iterations.

- **getNormalize**

```
public java.lang.Boolean getNormalize()
```

  - **Description**
    Returns the value of the property normalize.
  - **Returns** – The value of the property normalize.

- **getNumClusters**

```
public int getNumClusters() throws java.lang.Exception
```

  - **Description**
    Returns the number of clusters.
  - **Returns** – Returns the number of clusters to perform clustering.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **getPercentage**

```
public double getPercentage()
```

  - **Description**
    Gets the value of the percentage property.
  - **Returns** – The percentage of the train instances used as

- **normalize**

```
protected mulan.data.MultiLabelInstances normalize(mulan.data.
    MultiLabelInstances dataset) throws java.lang.Exception
```

  - **Description**
    Normalizes a multi-label dataset performing min-max normalization.
  - **Parameters**
    - `dataset` – The dataset to be normalized.
  - **Returns** – Returns the normalized dataset as MultiLabelInstances.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **prepareTemplate**

```
protected void prepareTemplate() throws java.lang.Exception
```

- **Description copied from MIMLtoML (in 20.5, page 272)**

  Prepares a template to perform the transformation from MIMLInstances to MultiLabelInstances. This template includes: the bag label attribute, all attributes in the relational attribute as independent attributes and label attributes. For instance, in the relation above, the resulting template is showed. @relation toy
  @attribute id {bag1,bag2}
  @attribute bag relational
  @attribute f1 numeric
  @attribute f2 numeric
  @attribute f3 numeric
  @end bag
  @attribute label1 {0,1}
  @attribute label2 {0,1}
  @attribute label3 {0,1}
  @attribute label4 {0,1}
  @relation template
  @attribute id {bag1,bag2}
  @attribute f1 numeric
  @attribute f2 numeric
  @attribute f3 numeric
  * @attribute label1 {0,1}
  @attribute label2 {0,1}
  @attribute label3 {0,1}
  @attribute label4 {0,1}

- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

• **setDistanceFunction**

```
public void setDistanceFunction(miml.core.distance.IDistance
    distanceFunction)
```

- **Description**

  Sets the distance function to use for clustering. This method must be called before clustering.

- **Parameters**
  - `distanceFunction` – The distance function used for clustering.

• **setMaxIterations**

```
public void setMaxIterations(int maxIterations)
```

  - **Description**
    Sets the maximum number of iterations for clustering. This method must be called before clustering.
  - **Parameters**
    * `maxIterations` – The maximum number of iterations for clustering.

- **setNormalize**

```
public void setNormalize(java.lang.Boolean normalize)
```

  - **Description**
    Sets the property normalized. If true, the resulting transformed multi-label dataset will be normalized after transformation.
  - **Parameters**
    * `normalize` – The value of the property to be set.

- **setNumClusters**

```
public void setNumClusters(int numClusters) throws java.lang.
    Exception
```

  - **Description**
    Sets the number of clusters to perform clustering in both the transformer and in the clusterer. If the clusterer is null the value of the property is only set in the transformer and the transformDataset method will establish this numClusters value in the clusterer after creating it.
  - **Parameters**
    * `numClusters` – A number of clusters.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **setPercentage**

```
public void setPercentage(double percentage)
```

  - **Description**
    Sets the value of the percentage property.
  - **Parameters**
    * `percentage` – The percentage value in [0, 1], for instance 0.2 means that the number of clusters is 20% the number of bags.

- **setSeed**

  ```
  public void setSeed(int seed)
  ```

  - **Description**
    Sets the value of the seed used for clustering in both the transformer and in the clusterer. If the clusterer is null the value of the property is only set in the transformer and the transformDataset method will establish this seed value in the clusterer after creating it.
  - **Parameters**
    * `seed` – The seed

- **transformDataset**

  ```
  public abstract mulan.data.MultiLabelInstances transformDataset
      () throws java.lang.Exception
  ```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances. To call this method is the dataset must be previously set eg. in the constructor.
  - **Returns** – MultiLabelInstances.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **transformDataset**

  ```
  public abstract mulan.data.MultiLabelInstances transformDataset(
      miml.data.MIMLInstances dataset) throws java.lang.Exception
  ```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances.
  - **Parameters**
    * `dataset` – The dataset to be transformed
  - **Returns** – MultiLabelInstances.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **transformInstance**

  ```
  public abstract weka.core.Instance transformInstance(miml.data.
      MIMLBag bag) throws java.lang.Exception
  ```

- **Description copied from MIMLtoML (in 20.5, page 272)**

  Transforms `MIMLBag` (in 7.1, page 96) into Instance.
- **Parameters**
  - `bag` – The Bag to be transformed.
- **Returns** – Instance
- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

20.4.8 Members inherited from class MIMLtoML

`miml.transformation.mimlTOml.MIMLtoML` (in 20.5, page 272)

- `protected` **dataset**
- `public static double` **minimax**`(weka.core.Instances` **data**`, int` **attIndex**`)`
- `protected void` **prepareTemplate**`() throws java.lang.Exception`
- `private static final` **serialVersionUID**
- `protected` **template**
- `public abstract MultiLabelInstances` **transformDataset**`() throws java.lang.Exception`
- `public abstract MultiLabelInstances` **transformDataset**`(miml.data.MIMLInstances` **dataset**`) throws java.lang.Exception`
- `public abstract Instance` **transformInstance**`(miml.data.MIMLBag` **bag**`) throws java.lang.Exception`
- `protected` **updatedLabelIndices**

20.5 Class MIMLtoML

Abstract class to transform MIMLInstances into MultiLabelInstances.

20.5.1 Declaration

```
public abstract class MIMLtoML
 extends java.lang.Object implements java.io.Serializable
```

20.5.2 All known subclasses

MinMaxTransformation (in 20.6, page 276), MedoidTransformation (in 20.4, page 263), KMeansTransformation (in 20.3, page 254), GeometricTransformation (in 20.2, page 252), ArithmeticTransformation (in 20.1, page 249)

20.5.3 Field summary

dataset Original data set of MIMLInstances.
serialVersionUID For serialization.
template Template to store Instances.
updatedLabelIndices Array of updated label indices.

20.5.4 Constructor summary

MIMLtoML() Constructor that does not sets the dataset
MIMLtoML(MIMLInstances) Constructor that sets the dataset

20.5.5 Method summary

minimax(Instances, int) Get the minimal and maximal value of a certain attribute in a data set.
prepareTemplate() Prepares a template to perform the transformation from MIMLInstances to MultiLabelInstances.
transformDataset() Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances.
transformDataset(MIMLInstances) Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances.
transformInstance(MIMLBag) Transforms `MIMLBag` (in 7.1, page 96) into Instance.

20.5.6 Fields

- `private static final long` **serialVersionUID**
  – For serialization.
- `protected int[]` **updatedLabelIndices**
  – Array of updated label indices.
- `protected weka.core.Instances` **template**
  – Template to store Instances.
- `protected miml.data.MIMLInstances` **dataset**
  – Original data set of MIMLInstances.

20.5.7 Constructors

- **MIMLtoML**

```
public MIMLtoML()
```

  – **Description**
  Constructor that does not sets the dataset

- **MIMLtoML**

```
public MIMLtoML(miml.data.MIMLInstances dataset)
```

  – **Description**
  Constructor that sets the dataset

- **Parameters**
  - `dataset` – The dataset to be transformed.

20.5.8 Methods

- **minimax**

```
public static double[] minimax(weka.core.Instances data,int
    attIndex)
```

  - **Description**
    Get the minimal and maximal value of a certain attribute in a data set.
  - **Parameters**
    - `data` – The data set.
    - `attIndex` – The index of the attribute.
  - **Returns** – double[] containing in position 0 the min value and in position 1 the max value.

- **prepareTemplate**

```
protected void prepareTemplate() throws java.lang.Exception
```

  - **Description**
    Prepares a template to perform the transformation from MIMLInstances to MultiLabelInstances. This template includes: the bag label attribute, all attributes in the relational attribute as independent attributes and label attributes. For instance, in the relation above, the resulting template is showed. @relation toy
    @attribute id {bag1,bag2}
    @attribute bag relational
    @attribute f1 numeric
    @attribute f2 numeric
    @attribute f3 numeric
    @end bag
    @attribute label1 {0,1}
    @attribute label2 {0,1}
    @attribute label3 {0,1}
    @attribute label4 {0,1}
    @relation template
    @attribute id {bag1,bag2}
    @attribute f1 numeric
    @attribute f2 numeric
    @attribute f3 numeric
    * @attribute label1 {0,1}
    @attribute label2 {0,1}

@attribute label3 {0,1}
@attribute label4 {0,1}

- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

- **transformDataset**

```
public abstract mulan.data.MultiLabelInstances transformDataset
    () throws java.lang.Exception
```

  - **Description**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances. To call this method is the dataset must be previously set eg. in the constructor.
  - **Returns** – MultiLabelInstances.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **transformDataset**

```
public abstract mulan.data.MultiLabelInstances transformDataset(
    miml.data.MIMLInstances dataset) throws java.lang.Exception
```

  - **Description**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances.
  - **Parameters**
    - `dataset` – The dataset to be transformed
  - **Returns** – MultiLabelInstances.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **transformInstance**

```
public abstract weka.core.Instance transformInstance(miml.data.
    MIMLBag bag) throws java.lang.Exception
```

  - **Description**
    Transforms `MIMLBag` (in 7.1, page 96) into Instance.
  - **Parameters**
    - `bag` – The Bag to be transformed.
  - **Returns** – Instance
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

20.6 Class MinMaxTransformation

Class that performs a miniMaxc transformation to convert a MIMLInstances class to MultiLabelInstances. Each Bag is transformed into a single Instance in which, for each attribute of the bag, its min and max value are included. For instance, For instance, in the relation above, the resulting template is showed. @relation toy
@attribute id {bag1,bag2}
@attribute bag relational
@attribute f1 numeric
@attribute f2 numeric
@attribute f3 numeric
@end bag
@attribute label1 {0,1}
@attribute label2 {0,1}
@attribute label3 {0,1}
@attribute label4 {0,1}
@relation minMaxTransformation
@attribute id {bag1,bag2}
@attribute f1_min numeric
@attribute f1_max numeric
@attribute f2_min numeric
@attribute f2_max numeric
@attribute f3_min numeric
@attribute f3_max numeric
* @attribute label1 {0,1}
@attribute label2 {0,1}
@attribute label3 {0,1}
@attribute label4 {0,1}

20.6.1 Declaration

```
public class MinMaxTransformation
 extends miml.transformation.mimlTOml.MIMLtoML
```

20.6.2 Field summary

serialVersionUID For serialization

20.6.3 Constructor summary

MinMaxTransformation()
MinMaxTransformation(MIMLInstances) Constructor.

20.6.4 Method summary

prepareTemplate()

transformDataset()
transformDataset(MIMLInstances)
transformInstance(MIMLBag)
transformInstance(MIMLInstances, MIMLBag)

20.6.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization

20.6.6 Constructors

- **MinMaxTransformation**

```
public MinMaxTransformation() throws java.lang.Exception
```

- **MinMaxTransformation**

```
public MinMaxTransformation(miml.data.MIMLInstances dataset)
    throws java.lang.Exception
```

  - **Description**
    Constructor.
  - **Parameters**
    * `dataset` – MIMLInstances dataset.
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

20.6.7 Methods

- **prepareTemplate**

```
protected void prepareTemplate() throws java.lang.Exception
```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Prepares a template to perform the transformation from MIMLInstances to MultiLabelInstances. This template includes: the bag label attribute, all attributes in the relational attribute as independent attributes and label attributes. For instance, in the relation above, the resulting template is showed. @relation toy
    @attribute id {bag1,bag2}
    @attribute bag relational
    @attribute f1 numeric
    @attribute f2 numeric
    @attribute f3 numeric

@end bag
@attribute label1 {0,1}
@attribute label2 {0,1}
@attribute label3 {0,1}
@attribute label4 {0,1}
@relation template
@attribute id {bag1,bag2}
@attribute f1 numeric
@attribute f2 numeric
@attribute f3 numeric
* @attribute label1 {0,1}
@attribute label2 {0,1}
@attribute label3 {0,1}
@attribute label4 {0,1}

- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

- **transformDataset**

```
public abstract mulan.data.MultiLabelInstances transformDataset
    () throws java.lang.Exception
```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances. To call this method is the dataset must be previously set eg. in the constructor.
  - **Returns** – MultiLabelInstances.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **transformDataset**

```
public abstract mulan.data.MultiLabelInstances transformDataset(
    miml.data.MIMLInstances dataset) throws java.lang.Exception
```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLInstances` (in 7.2, page 101) into MultiLabelInstances.
  - **Parameters**
    - `dataset` – The dataset to be transformed
  - **Returns** – MultiLabelInstances.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **transformInstance**

```
public abstract weka.core.Instance transformInstance(miml.data.
    MIMLBag bag) throws java.lang.Exception
```

  - **Description copied from MIMLtoML (in 20.5, page 272)**
    Transforms `MIMLBag` (in 7.1, page 96) into Instance.
  - **Parameters**
    * `bag` – The Bag to be transformed.
  - **Returns** – Instance
  - **Throws**
    * `java.lang.Exception` – To be handled in an upper level.

- **transformInstance**

```
public weka.core.Instance transformInstance(miml.data.
    MIMLInstances dataset,miml.data.MIMLBag bag) throws java.lang
    .Exception
```

20.6.8 Members inherited from class MIMLtoML

`miml.transformation.mimlTOml.MIMLtoML` (in 20.5, page 272)

- `protected` **dataset**
- `public static double` **minimax**`(weka.core.Instances` **data**`, int` **attIndex**`)`
- `protected void` **prepareTemplate**`() throws java.lang.Exception`
- `private static final` **serialVersionUID**
- `protected` **template**
- `public abstract MultiLabelInstances` **transformDataset**`() throws java.lang.Exception`
- `public abstract MultiLabelInstances` **transformDataset**`(miml.data.MIMLInstances` **dataset**`) throws java.lang.Exception`
- `public abstract Instance` **transformInstance**`(miml.data.MIMLBag` **bag**`) throws java.lang.Exception`
- `protected` **updatedLabelIndices**

20.7 Class PropositionalTransformation

Class that performs a propositionalTransformation to convert a MIMLInstances dataset to MultiLabelInstances. This transformation transforms each Bag into a set if instances, one for each instance in the bag of the instances in the bag.

20.7.1 Declaration

```
public class PropositionalTransformation
 extends java.lang.Object implements java.io.Serializable
```

20.7.2 Field summary

dataset Original data set of MIMLInstances.
includeBagId Whether bag attribute will be included in the transformed data
removeFilter Filter
serialVersionUID For serialization.
template Template to store Instances.
updatedLabelIndices Array of updated label indices.

20.7.3 Constructor summary

PropositionalTransformation(MIMLInstances) Constructor.
PropositionalTransformation(MIMLInstances, boolean) Constructor.

20.7.4 Method summary

isIncludeBagId() Returns the value of includeBagId property.
prepareTemplate() Prepares a template to perform the transformation from MIMLInstances to MultiLabelInstances.
removeBagId(MultiLabelInstances) Removes the bagId attribute in MultiLabelInstances.
setIncludeBagId(boolean) Sets the value for includeBagId property.
transformDataset()
transformDataset(MIMLInstances)
transformInstance(MIMLBag)
transformInstance(MIMLInstances, MIMLBag)

20.7.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `protected int[]` **updatedLabelIndices**
  - Array of updated label indices.
- `protected weka.core.Instances` **template**
  - Template to store Instances.
- `protected miml.data.MIMLInstances` **dataset**
  - Original data set of MIMLInstances.
- `protected weka.filters.unsupervised.attribute.Remove` **removeFilter**
  - Filter
- `protected boolean` **includeBagId**
  - Whether bag attribute will be included in the transformed data

20.7.6 Constructors

- **PropositionalTransformation**

  ```
  public PropositionalTransformation(miml.data.MIMLInstances
      dataset) throws java.lang.Exception
  ```

  - **Description**
    Constructor.
  - **Parameters**
    - `dataset` – MIMLInstances dataset.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **PropositionalTransformation**

  ```
  public PropositionalTransformation(miml.data.MIMLInstances
      dataset ,boolean includeBagId) throws java.lang.Exception
  ```

  - **Description**
    Constructor.
  - **Parameters**
    - `dataset` – MIMLInstances dataset.
    - `includeBagId` – true if the bagId will be included in the transformed dataset
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

20.7.7 Methods

- **isIncludeBagId**

  ```
  public boolean isIncludeBagId()
  ```

  - **Description**
    Returns the value of includeBagId property.
  - **Returns** – The value of includeBagId property.

- **prepareTemplate**

  ```
  protected void prepareTemplate() throws java.lang.Exception
  ```

  - **Description**

    Prepares a template to perform the transformation from MIMLInstances to MultiLabelInstances. This template includes: the bag label attribute, all attributes in the relational attribute as independent attributes and label attributes. For instance, in the relation above, the resulting template is showed. @relation toy
    @attribute id {bag1,bag2}
    @attribute bag relational
    @attribute f1 numeric
    @attribute f2 numeric
    @attribute f3 numeric
    @end bag
    @attribute label1 {0,1}
    @attribute label2 {0,1}
    @attribute label3 {0,1}
    @attribute label4 {0,1}
    @relation template
    @attribute id {bag1,bag2}
    @attribute f1 numeric
    @attribute f2 numeric
    @attribute f3 numeric
    * @attribute label1 {0,1}
    @attribute label2 {0,1}
    @attribute label3 {0,1}
    @attribute label4 {0,1}

  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **removeBagId**

```
public static mulan.data.MultiLabelInstances removeBagId(mulan.
    data.MultiLabelInstances mlDataSetWithBagId) throws java.lang
    .Exception
```

  - **Description**

    Removes the bagId attribute in MultiLabelInstances.
  - **Parameters**
    - `mlDataSetWithBagId` – A MultiLabelInstances dataset corresponding with the propositional representation of MIML data being the first attribute the bagID.
  - **Returns** – MultiLabelInstances without first bagIdAttribute
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **setIncludeBagId**

```
public void setIncludeBagId(boolean includeBagId)
```

- **Description**
  Sets the value for includeBagId property.
- **Parameters**
  * `includeBagId` – if true the bagId will be included in the transformed data.

- **transformDataset**

```
public mulan.data.MultiLabelInstances transformDataset() throws
    java.lang.Exception
```

- **transformDataset**

```
public mulan.data.MultiLabelInstances transformDataset(miml.data
    .MIMLInstances dataset) throws java.lang.Exception
```

- **transformInstance**

```
public mulan.data.MultiLabelInstances transformInstance(miml.
    data.MIMLBag bag) throws java.lang.Exception
```

- **transformInstance**

```
public mulan.data.MultiLabelInstances transformInstance(miml.
    data.MIMLInstances dataset,miml.data.MIMLBag bag) throws java
    .lang.Exception
```

Chapter 21

Package miml.data.partitioning

Package Contents *Page*

21.1 Class CrossValidationBase

General scheme for cross validation partitioners of multi-output data. MOR, MIML and MVML formats are also supported.

21.1.1 Declaration

```
public abstract class CrossValidationBase
 extends miml.data.partitioning.PartitionerBase
```

21.1.2 All known subclasses

RandomCrossValidation (in 2.1, page 29), LabelPowersetCrossValidation (in 14.1, page 193), IterativeCrossValidation (in 25.1, page 336)

21.1.3 Constructor summary

CrossValidationBase(int, MultiLabelInstances) Constructor.
CrossValidationBase(MultiLabelInstances) Default constructor.

21.1.4 Method summary

foldsToRounds(MultiLabelInstances[]) Returns the train and test sets for each fold.
getFolds(int) Splits a dataset into nfolds partitions.
getRounds(int) Returns the train and test sets for each fold.
statsToString(MultiLabelInstances[])

21.1.5 Constructors

- **CrossValidationBase**

```
public CrossValidationBase(int seed,mulan.data.
    MultiLabelInstances mlDataSet) throws mulan.data.
    InvalidDataFormatException
```

  - **Description**
    Constructor.
  - **Parameters**
    * `seed` – Seed for randomization
    * `mlDataSet` – A multi-label dataset
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – To be handled

- **CrossValidationBase**

```
public CrossValidationBase(mulan.data.MultiLabelInstances
    mlDataSet) throws mulan.data.InvalidDataFormatException
```

  - **Description**
    Default constructor.
  - **Parameters**
    * `mlDataSet` – A multi-label dataset
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – To be handled

21.1.6 Methods

- **foldsToRounds**

```
public static mulan.data.MultiLabelInstances[][] foldsToRounds(
    mulan.data.MultiLabelInstances[] Folds) throws java.lang.
    Exception
```

- **Description**
  Returns the train and test sets for each fold. This method is static being useful if the user has partitions.
- **Parameters**
  * `Folds` – The folds.
- **Returns** – MultiLabelInstances[][] a nfolds x 2 matrix. Each row represents a fold being column 0 the train set and the column 1 the test set.
- **Throws**
  * `java.lang.Exception` – To be handled.

• **getFolds**

```
public abstract mulan.data.MultiLabelInstances[] getFolds(int
    nFolds) throws mulan.data.InvalidDataFormatException
```

- **Description**
  Splits a dataset into nfolds partitions.
- **Parameters**
  * `nFolds` – Number of folds.
- **Returns** – MultiLabelInstances[] a vector of nFolds. Each element represents a fold.
- **Throws**
  * `mulan.data.InvalidDataFormatException` – To be handled.

• **getRounds**

```
public mulan.data.MultiLabelInstances[][] getRounds(int nFolds)
    throws java.lang.Exception
```

- **Description**
  Returns the train and test sets for each fold.
- **Parameters**
  * `nFolds` – Number of folds.
- **Returns** – MultiLabelInstances[][] a nfolds x 2 matrix. Each row represents a fold being column 0 the train set and the column 1 the test set.
- **Throws**
  * `mulan.data.InvalidDataFormatException` – To be handled.

• **statsToString**

```
protected abstract void statsToString(mulan.data.
    MultiLabelInstances[] Partition)
```

- **Description copied from PartitionerBase (in 21.2, page 287)**

  Given an array with datasets corresponding to partitions, prints the number of examples of each dataset of the vector
- **Parameters**
  - `Partition` – An array with the partitions. In case of train/test, partition Partition[0] is the train set and Partition[1] is the test set. In case of CV, Partition[i] is the ith fold.

21.1.7 Members inherited from class PartitionerBase

`miml.data.partitioning.PartitionerBase` (in 21.2, page 287)

- `protected` **seed**
- `protected abstract void` **statsToString**(`mulan.data.MultiLabelInstances[]` **Partition**)
- `public int` **totalExamples**()
- `protected` **workingSet**

21.2 Class PartitionerBase

General scheme for partitioning multi-output data.

21.2.1 Declaration

```
public abstract class PartitionerBase
 extends java.lang.Object
```

21.2.2 All known subclasses

RandomTrainTest (in 2.2, page 31), RandomCrossValidation (in 2.1, page 29), LabelPowersetTrainTest (in 14.2, page 195), LabelPowersetCrossValidation (in 14.1, page 193), TrainTestBase (in 21.3, page 289), CrossValidationBase (in 21.1, page 284), IterativeTrainTest (in 25.2, page 339), IterativeCrossValidation (in 25.1, page 336)

21.2.3 Field summary

seed Seed for reproduction of results
workingSet A copy of the instances to generate partitions

21.2.4 Constructor summary

PartitionerBase(int, MultiLabelInstances) Constructor of the class
PartitionerBase(MultiLabelInstances) Constructor of the class

21.2.5 Method summary

statsToString(MultiLabelInstances[]) Given an array with datasets corresponding to partitions, prints the number of examples of each dataset of the vector
totalExamples() Returns the number of examples of the dataset to be partitioned.

21.2.6 Fields

- `protected int` **seed**
  - Seed for reproduction of results
- `protected mulan.data.MultiLabelInstances` **workingSet**
  - A copy of the instances to generate partitions

21.2.7 Constructors

- **PartitionerBase**

```
public PartitionerBase(int seed,mulan.data.MultiLabelInstances
    mlDataSet) throws mulan.data.InvalidDataFormatException
```

  - **Description**
    Constructor of the class
  - **Parameters**
    * `seed` – Seed for randomization
    * `mlDataSet` – The multi-label data set
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – To be handled.

- **PartitionerBase**

```
public PartitionerBase(mulan.data.MultiLabelInstances mlDataSet)
     throws mulan.data.InvalidDataFormatException
```

  - **Description**
    Constructor of the class
  - **Parameters**
    * `mlDataSet` – The multi-label data set
  - **Throws**
    * `mulan.data.InvalidDataFormatException` – To be handled.

21.2.8 Methods

- **statsToString**

```
protected abstract void statsToString(mulan.data.
    MultiLabelInstances[] Partition)
```

– **Description**
  Given an array with datasets corresponding to partitions, prints the number of examples of each dataset of the vector
– **Parameters**
  * `Partition` – An array with the partitions. In case of train/test, partition Partition[0] is the train set and Partition[1] is the test set. In case of CV, Partition[i] is the ith fold.

• **totalExamples**

```
public int totalExamples()
```

– **Description**
  Returns the number of examples of the dataset to be partitioned.
– **Returns** – int

21.3 Class TrainTestBase

General scheme for train test partitioning of multi-output data. MOR, MIML and MVML formats are also supported.

21.3.1 Declaration

```
public abstract class TrainTestBase
 extends miml.data.partitioning.PartitionerBase
```

21.3.2 All known subclasses

RandomTrainTest (in 2.2, page 31), LabelPowersetTrainTest (in 14.2, page 195), IterativeTrainTest (in 25.2, page 339)

21.3.3 Constructor summary

TrainTestBase(int, MultiLabelInstances) Constructor.
TrainTestBase(MultiLabelInstances) Default constructor.

21.3.4 Method summary

split(double) Returns a array with two multi-label random datasets corresponding to the train and test sets respectively.
statsToString(MultiLabelInstances[])

21.3.5 Constructors

- **TrainTestBase**

```
public TrainTestBase(int seed,mulan.data.MultiLabelInstances
    mlDataSet) throws mulan.data.InvalidDataFormatException
```

  - **Description**
    Constructor.
  - **Parameters**
    - `seed` – Seed for randomization
    - `mlDataSet` – A multi-label dataset
  - **Throws**
    - `mulan.data.InvalidDataFormatException` – To be handled

- **TrainTestBase**

```
public TrainTestBase(mulan.data.MultiLabelInstances mlDataSet)
    throws mulan.data.InvalidDataFormatException
```

  - **Description**
    Default constructor.
  - **Parameters**
    - `mlDataSet` – A multi-label dataset
  - **Throws**
    - `mulan.data.InvalidDataFormatException` – To be handled

21.3.6 Methods

- **split**

```
public abstract mulan.data.MultiLabelInstances[] split(double
    percentageTrain) throws java.lang.Exception
```

  - **Description**
    Returns a array with two multi-label random datasets corresponding to the train and test sets respectively.
  - **Parameters**
    - `percentageTrain` – Percentage of train dataset, a value in [0, 100].
  - **Returns** – MultiLabelInstances[].
    MultiLabelInstances[0] is the train set.
    MultiLabelInstances[1] is the test set.

- **Throws**
  - `java.lang.Exception` – To be handled.

- **statsToString**

```
protected abstract void statsToString(mulan.data.
    MultiLabelInstances[] Partition)
```

- **Description copied from PartitionerBase (in 21.2, page 287)**

  Given an array with datasets corresponding to partitions, prints the number of examples of each dataset of the vector
- **Parameters**
  - `Partition` – An array with the partitions. In case of train/test, partition Partition[0] is the train set and Partition[1] is the test set. In case of CV, Partition[i] is the ith fold.

21.3.7 Members inherited from class PartitionerBase

`miml.data.partitioning.PartitionerBase` (in 21.2, page 287)

- `protected` **seed**
- `protected abstract void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`
- `public int` **totalExamples**`()`
- `protected` **workingSet**

Chapter 22

Package miml.tutorial

Package Contents *Page*

22.1 Class Clustering

Class to show an example of clustering of a MIML Dataset.

22.1.1 Declaration

```
public class Clustering
 extends java.lang.Object
```

22.1.2 Constructor summary

Clustering()

22.1.3 Method summary

main(String[])

22.1.4 Constructors

- **Clustering**

```
public Clustering()
```

22.1.5 Methods

- **main**

```
public static void main(java.lang.String[] args)
```

22.2 Class CrossValidationExperiment

Class implementing an example of using cross-validation with different kinds of classifier.

22.2.1 Declaration

```
public class CrossValidationExperiment
 extends java.lang.Object
```

22.2.2 Constructor summary

CrossValidationExperiment()

22.2.3 Method summary

main(String[])
showUse() Shows the help on command line.

22.2.4 Constructors

- **CrossValidationExperiment**

```
public CrossValidationExperiment()
```

22.2.5 Methods

- **main**

```
public static void main(java.lang.String[] args) throws java.
    lang.Exception
```

- **showUse**

```
public static void showUse()
```

  - **Description**
    Shows the help on command line.

22.3 Class GeneratePartitions

Class to split a multi-output dataset into partitions for cross-validation or train-test. This class is able to work on multi-label, multi-instance multi-label, and multi-view multi-label.

22.3.1 Declaration

```
public class GeneratePartitions
 extends java.lang.Object
```

22.3.2 Constructor summary

GeneratePartitions()

22.3.3 Method summary

main(String[]) Main method.
showUse() Shows the help on command line.

22.3.4 Constructors

- **GeneratePartitions**

```
public GeneratePartitions()
```

22.3.5 Methods

- **main**

```
public static void main(java.lang.String[] args) throws java.
    lang.Exception
```

  - **Description**
    Main method.
  - **Parameters**
    * `args` – Command line arguments.
      · -f filename.arff ->name of the filename to be partitioned
      · -x file.xml
      · -[t—c] value
      · -t double_percentage ->train-test and tranin percentage
      · -c integer_nFolds ->cross-validation and number of folds
      · -s 1—2—3
      · -s 1 ->random stratification (by default)
      · -s 2 ->label powerset stratification
      · -s 3 ->iterative stratification
      *
      · -o OutputFile (without extension)
      · train-test ->OutputFile_train.arff and OutputFile_test.arff
      · cross-validation ->OutputFile_1.arff ... OutputFile_nFolds.arff
  - **Throws**
    * `java.lang.Exception` – To be handled.

- **showUse**

```
public static void showUse()
```

  - **Description**
    Shows the help on command line.

22.4 Class HoldoutExperiment

Class implementing an example of using holdout with train/test dataset and a single dataset applying percentage split.

22.4.1 Declaration

```
public class HoldoutExperiment
 extends java.lang.Object
```

22.4.2 Constructor summary

HoldoutExperiment()

22.4.3 Method summary

main(String[])
showUse() Shows the help on command line.

22.4.4 Constructors

- **HoldoutExperiment**

```
public HoldoutExperiment()
```

22.4.5 Methods

- **main**

```
public static void main(java.lang.String[] args) throws java.
    lang.Exception
```

- **showUse**

```
public static void showUse()
```

  - **Description**
    Shows the help on command line.

22.5 Class HoldoutToML_RFPCT

Class implementing an example of using holdout with train/test dataset and a toML classifier with RFPCT as base classifier.

22.5.1 Declaration

```
public class HoldoutToML_RFPCT
 extends java.lang.Object
```

22.5.2 Constructor summary

HoldoutToML_RFPCT()

22.5.3 Method summary

main(String[])

22.5.4 Constructors

- **HoldoutToML_RFPCT**

```
public HoldoutToML_RFPCT()
```

22.5.5 Methods

- **main**

```
public static void main(java.lang.String[] args) throws java.
    lang.Exception
```

22.6 Class InsertingAttributesToBags

Class implementing an example of inserting a new group of attributes to the relational attribute of the dataset with {0,1} values.

22.6.1 Declaration

```
public class InsertingAttributesToBags
 extends java.lang.Object
```

22.6.2 Constructor summary

InsertingAttributesToBags()

22.6.3 Method summary

main(String[])
showUse() Shows the help on command line.

22.6.4 Constructors

- **InsertingAttributesToBags**

```
public InsertingAttributesToBags()
```

22.6.5 Methods

- **main**

  ```
  public static void main(java.lang.String[] args) throws java.
      lang.Exception
  ```

- **showUse**

  ```
  public static void showUse()
  ```

  - **Description**

    Shows the help on command line.

22.7 Class InsertingAttributeToBag

Class implementing an example of inserting a new attribute to the relational attribute of the dataset with {0,1} values.

22.7.1 Declaration

```
public class InsertingAttributeToBag
 extends java.lang.Object
```

22.7.2 Constructor summary

InsertingAttributeToBag()

22.7.3 Method summary

main(String[])
showUse() Shows the help on command line.

22.7.4 Constructors

- **InsertingAttributeToBag**

  ```
  public InsertingAttributeToBag()
  ```

22.7.5 Methods

- **main**

  ```
  public static void main(java.lang.String[] args) throws java.
      lang.Exception
  ```

- **showUse**

  ```
  public static void showUse()
  ```

  - **Description**

    Shows the help on command line.

22.8 Class ManagingMIMLInstances

Class implementing basic handling of MIML datasets.

22.8.1 Declaration

```
public class ManagingMIMLInstances
 extends java.lang.Object
```

22.8.2 Constructor summary

ManagingMIMLInstances()

22.8.3 Method summary

main(String[])
showUse() Shows the help on command line.

22.8.4 Constructors

- **ManagingMIMLInstances**

  ```
  public ManagingMIMLInstances()
  ```

22.8.5 Methods

- **main**

  ```
  public static void main(java.lang.String[] args)
  ```

- **showUse**

  ```
  public static void showUse()
  ```

  - **Description**

    Shows the help on command line.

22.9 Class MIMLtoMITransformation

Class for basic handling of MIML to MIL LP and BR transformation.

22.9.1 Declaration

```
public class MIMLtoMITransformation
 extends java.lang.Object
```

22.9.2 Constructor summary

MIMLtoMITransformation()

22.9.3 Method summary

main(String[])
showUse() Shows the help on command line.

22.9.4 Constructors

- **MIMLtoMITransformation**

  ```
  public MIMLtoMITransformation()
  ```

22.9.5 Methods

- **main**

  ```
  public static void main(java.lang.String[] args) throws java.
      lang.Exception
  ```

- **showUse**

  ```
  public static void showUse()
  ```

  - **Description**

    Shows the help on command line.

22.10 Class MIMLtoMLTransformation

Class for basic handling of the transformation MIML to ML transformations.

22.10.1 Declaration

```
public class MIMLtoMLTransformation
 extends java.lang.Object
```

22.10.2 Constructor summary

MIMLtoMLTransformation()

22.10.3 Method summary

main(String[])
showUse() Shows the help on command line.

22.10.4 Constructors

- **MIMLtoMLTransformation**

```
public MIMLtoMLTransformation()
```

22.10.5 Methods

- **main**

```
public static void main(java.lang.String[] args) throws java.
    lang.Exception
```

- **showUse**

```
public static void showUse()
```

  - **Description**

    Shows the help on command line.

22.11 Class NormalizingDataset

Class to show an example of normalization of a MIML Dataset.

22.11.1 Declaration

```
public class NormalizingDataset
 extends java.lang.Object
```

22.11.2 Constructor summary

NormalizingDataset()

22.11.3 Method summary

main(String[])

22.11.4 Constructors

- **NormalizingDataset**

```
public NormalizingDataset()
```

22.11.5 Methods

- **main**

```
public static void main(java.lang.String[] args)
```

22.12 Class Resampling

Class to show an example of sampling with replacement.

22.12.1 Declaration

```
public class Resampling
 extends java.lang.Object
```

22.12.2 Constructor summary

Resampling()

22.12.3 Method summary

main(String[])

22.12.4 Constructors

- **Resampling**

```
public Resampling()
```

22.12.5 Methods

- **main**

```
public static void main(java.lang.String[] args) throws java.
    lang.Exception
```

Chapter 23

Package miml.report

Package Contents *Page*

23.1 Interface IReport

Interface for generate reports with the format specified.

23.1.1 Declaration

```
public interface IReport
```

23.1.2 All known subinterfaces

MIMLReport (in 23.3, page 309), BaseMIMLReport (in 23.2, page 306)

23.1.3 All classes known to implement interface

MIMLReport (in 23.3, page 309)

23.1.4 Method summary

saveReport(String) Save in a file the specified report.
toCSV(IEvaluator) Convert to CSV the evaluator results.
toString(IEvaluator) Convert to plain text the evaluator results.

23.1.5 Methods

- **saveReport**

  ```
  void saveReport(java.lang.String report) throws java.io.
      FileNotFoundException
  ```

  - **Description**

    Save in a file the specified report.
  - **Parameters**
    - `report` – The formatted string to be saved.
  - **Throws**
    - `java.io.FileNotFoundException` – To be handled in an upper level.

- **toCSV**

  ```
  java.lang.String toCSV(miml.evaluation.IEvaluator evaluator)
      throws java.lang.Exception
  ```

  - **Description**

    Convert to CSV the evaluator results.
  - **Parameters**
    - `evaluator` – The evaluator with the measures.
  - **Returns** – String with CSV content.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **toString**

  ```
  java.lang.String toString(miml.evaluation.IEvaluator evaluator)
      throws java.lang.Exception
  ```

  - **Description**

    Convert to plain text the evaluator results.
  - **Parameters**
    - `evaluator` – The evaluator with the measures.
  - **Returns** – String with the content.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

23.2 Class BaseMIMLReport

Class used to generate reports with the format specified.

23.2.1 Declaration

```
public class BaseMIMLReport
 extends miml.report.MIMLReport
```

23.2.2 Constructor summary

BaseMIMLReport() No-argument constructor for xml configuration.
BaseMIMLReport(List, String, boolean, boolean, boolean) Basic constructor to initialize the report.

23.2.3 Method summary

configure(Configuration)
crossValidationToCSV(EvaluatorCV) Read the cross-validation results and transform to CSV format.
crossValidationToString(EvaluatorCV) Read the cross-validation results and transform to plain text.
holdoutToCSV(EvaluatorHoldout) Read the holdout results and transform to CSV format.
holdoutToString(EvaluatorHoldout) Read the holdout results and transform to plain text.
toCSV(IEvaluator)
toString(IEvaluator)

23.2.4 Constructors

- **BaseMIMLReport**

  ```
  public BaseMIMLReport()
  ```

  – **Description**

    No-argument constructor for xml configuration.

- **BaseMIMLReport**

  ```
  public BaseMIMLReport(java.util.List measures,java.lang.String
      filename,boolean std,boolean labels,boolean header)
  ```

  – **Description**

    Basic constructor to initialize the report.

- **Parameters**
  - `measures` – The list of selected measures which is going to be shown in the report.
  - `filename` – The filename where the report's will be saved.
  - `std` – Whether the standard deviation of measures will be shown or not (only valid for cross-validation evaluator).
  - `labels` – Whether the measures for each label will be shown (only valid for Macro-Averaged measures).
  - `header` – Whether the header will be shown.

23.2.5 Methods

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **crossValidationToCSV**

```
protected java.lang.String crossValidationToCSV(miml.evaluation.
    EvaluatorCV evaluator) throws java.lang.Exception
```

  - **Description**

    Read the cross-validation results and transform to CSV format.

  - **Parameters**
    - `evaluator` – The evaluator.
  - **Returns** – String with CSV content.
  - **Throws**
    - `java.lang.Exception` – To be handled in an upper level.

- **crossValidationToString**

```
protected java.lang.String crossValidationToString(miml.
    evaluation.EvaluatorCV evaluator) throws java.lang.Exception
```

  - **Description**

    Read the cross-validation results and transform to plain text.

- **Parameters**
  - `evaluator` – The evaluator.
- **Returns** – String with the content.
- **Throws**
  - `java.lang.Exception` – To be handled in an upper level

- **holdoutToCSV**

```
protected java.lang.String holdoutToCSV(miml.evaluation.
    EvaluatorHoldout evaluator) throws java.lang.Exception
```

- **Description**
  Read the holdout results and transform to CSV format.
- **Parameters**
  - `evaluator` – The evaluator.
- **Returns** – String with CSV content.
- **Throws**
  - `java.lang.Exception` – To be handled in an upper level

- **holdoutToString**

```
protected java.lang.String holdoutToString(miml.evaluation.
    EvaluatorHoldout evaluator) throws java.lang.Exception
```

- **Description**
  Read the holdout results and transform to plain text.
- **Parameters**
  - `evaluator` – The evaluator.
- **Returns** – String with the content.
- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

- **toCSV**

```
public java.lang.String toCSV(miml.evaluation.IEvaluator
    evaluator) throws java.lang.Exception
```

- **toString**

```
public java.lang.String toString(miml.evaluation.IEvaluator
    evaluator) throws java.lang.Exception
```

23.2.6 Members inherited from class MIMLReport

`miml.report.MIMLReport` (in 23.3, page 309)

- `protected` **filename**
- `protected List` **filterMeasures**`(java.util.List` **allMeasures**`) throws java.lang.Exception`
- `public String` **getFilename**`()`
- `public List` **getMeasures**`()`
- `protected` **header**
- `public boolean` **isHeader**`()`
- `public boolean` **isLabels**`()`
- `public boolean` **isStd**`()`
- `protected` **labels**
- `protected` **measures**
- `public void` **saveReport**`(java.lang.String` **report**`) throws java.io.FileNotFoundException`
- `public void` **setFilename**`(java.lang.String` **filename**`)`
- `public void` **setHeader**`(boolean` **header**`)`
- `public void` **setLabels**`(boolean` **labels**`)`
- `public void` **setMeasures**`(java.util.List` **measures**`) throws java.lang.Exception`
- `public void` **setStd**`(boolean` **std**`)`
- `protected` **std**

23.3 Class MIMLReport

Abstract class for a MIMLReport.

23.3.1 Declaration

```
public abstract class MIMLReport
 extends java.lang.Object implements IReport, miml.core.
    IConfiguration
```

23.3.2 All known subclasses

BaseMIMLReport (in 23.2, page 306)

23.3.3 Field summary

filename The name of the file where report is saved.
header If the header is going to be printed.
labels If macro measures are broken down by labels.
measures The measures shown in the report.
std If measures' standard deviation are shown.

23.3.4 Constructor summary

MIMLReport() No-argument constructor for xml configuration.
MIMLReport(List, String, boolean, boolean, boolean) Basic constructor to initialize the report.

23.3.5 Method summary

filterMeasures(List) Filter measures chosen to be shown in the experiment report.
getFilename() Gets the filename.
getMeasures() Gets the measures shown in the report.
isHeader() Checks if header is shown.
isLabels() Checks if measure for each label (macro-averaged measures) is shown.
isStd() Checks if std is going to be shown (only cross-validation).
saveReport(String) Save in a file the specified report.
setFilename(String) Sets the name of the file.
setHeader(boolean) Sets if header is shown.
setLabels(boolean) Sets if measure for each label (macro-averaged measures) is shown.
setMeasures(List) Sets the measures shown in the report.
setStd(boolean) Sets if the std is going to be shown (only cross-validation).

23.3.6 Fields

- `protected java.util.List` **measures**
  - The measures shown in the report.
- `protected java.lang.String` **filename**
  - The name of the file where report is saved.
- `protected boolean` **std**
  - If measures' standard deviation are shown.
- `protected boolean` **labels**

– If macro measures are broken down by labels.

- `protected boolean` **header**
  – If the header is going to be printed.

23.3.7 Constructors

- **MIMLReport**

```
public MIMLReport()
```

  – **Description**
    No-argument constructor for xml configuration.

- **MIMLReport**

```
public MIMLReport(java.util.List measures,java.lang.String
    filename,boolean std,boolean labels,boolean header)
```

  – **Description**
    Basic constructor to initialize the report.
  – **Parameters**
    * `measures` – The list of selected measures which is going to be shown in the report.
    * `filename` – The filename where the report's will be saved.
    * `std` – Whether the standard deviation of measures will be shown or not (only valid for cross-validation evaluator).
    * `labels` – Whether the measures for each label will be shown (only valid for Macro-Averaged measures).
    * `header` – Whether the header will be shown.

23.3.8 Methods

- **filterMeasures**

```
protected java.util.List filterMeasures(java.util.List
    allMeasures) throws java.lang.Exception
```

- **Description**
  Filter measures chosen to be shown in the experiment report.
- **Parameters**
  * `allMeasures` – All the measures which the evaluation has
- **Returns** – List with the measures filtered
- **Throws**
  * `java.lang.Exception` – To be handled in an upper level.

• **getFilename**

```
public java.lang.String getFilename()
```

- **Description**
  Gets the filename.
- **Returns** – The filename.

• **getMeasures**

```
public java.util.List getMeasures()
```

- **Description**
  Gets the measures shown in the report.
- **Returns** – The measures.

• **isHeader**

```
public boolean isHeader()
```

- **Description**
  Checks if header is shown.
- **Returns** – True, if header is shown.

• **isLabels**

```
public boolean isLabels()
```

  - **Description**

    Checks if measure for each label (macro-averaged measures) is shown.

  - **Returns** – True, if measure for each label is shown.

- **isStd**

  ```
  public boolean isStd()
  ```

  - **Description**

    Checks if std is going to be shown (only cross-validation).

  - **Returns** – True, if std is going to be shown.

- **saveReport**

  ```
  public void saveReport(java.lang.String report) throws java.io.
      FileNotFoundException
  ```

  - **Description**

    Save in a file the specified report.

  - **Parameters**

    * `report` – The report.

  - **Throws**

    * `java.io.FileNotFoundException` – To be handled in an upper level.

- **setFilename**

  ```
  public void setFilename(java.lang.String filename)
  ```

  - **Description**

    Sets the name of the file.

  - **Parameters**

    * `filename` – The new filename

- **setHeader**

```
public void setHeader(boolean header)
```

- **Description**

  Sets if header is shown.
- **Parameters**
  - `header` – The new header configuration.

- **setLabels**

```
public void setLabels(boolean labels)
```

- **Description**

  Sets if measure for each label (macro-averaged measures) is shown.
- **Parameters**
  - `labels` – The new labels configuration.

- **setMeasures**

```
public void setMeasures(java.util.List measures) throws java.
    lang.Exception
```

- **Description**

  Sets the measures shown in the report.
- **Parameters**
  - `measures` – The new measures.
- **Throws**
  - `java.lang.Exception` – To be handled in an upper level.

- **setStd**

```
public void setStd(boolean std)
```

- **Description**

  Sets if the std is going to be shown (only cross-validation).
- **Parameters**
  - `std` – The new std configuration.

Chapter 24

Package miml.classifiers.miml.neural

24.1 Class EnMIMLNNmetric

Class to execute the **EnMIMLNNmetric** algorithm for MIML data. For more information, see *Wu, J. S., Huang, S. J., & Zhou, Z. H. (2014). Genome-wide protein function prediction through multi-instance multi-label learning. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 11(5), 891-902.*.

24.1.1 Declaration

```
public class EnMIMLNNmetric
 extends miml.classifiers.miml.MWClassifier
```

24.1.2 Field summary

enmimlnn A matlab object wrapping the EnMIMLNNmetric algorithm.
mu The ratio used to determine the standard deviation of the Gaussian activation function.
ratio The number of centroids of the i-th label is set to be ratio*Ti, where Ti is the number of train bags with label i.
seed Seed for kmedoids clustering.
serialVersionUID For serialization.

24.1.3 Constructor summary

EnMIMLNNmetric() No-argument constructor for xml configuration.
EnMIMLNNmetric(double, double) Basic constructor to initialize the classifier.
EnMIMLNNmetric(double, double, int) Constructor to initialize the classifier.

24.1.4 Method summary

configure(Configuration)
dispose()
getMu() Returns the scaling factor parameter considered to build the classifier.
getRatio() Returns the fraction parameter considered to build the classifier.
getSeed() Returns the seed for kmedoids clustering considered to build the classifier.
predictMWClassifier(MWCellArray, MWNumericArray, MWNumericArray)
setMu(double) Sets the scaling factor parameter to build the classifier.
setRatio(double) Sets the fraction parameter to build the classifier.
setSeed(int) Sets the seed for kmedoids clustering considered to build the classifier.
trainMWClassifier(MWCellArray, MWNumericArray)

24.1.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `static MWAlgorithms.MWEnMIMLNNmetric` **enmimlnn**
  - A matlab object wrapping the EnMIMLNNmetric algorithm.
- `double` **ratio**
  - The number of centroids of the i-th label is set to be ratio*Ti, where Ti is the number of train bags with label i.
- `double` **mu**
  - The ratio used to determine the standard deviation of the Gaussian activation function.
- `int` **seed**
  - Seed for kmedoids clustering.

24.1.6 Constructors

- **EnMIMLNNmetric**

  ```
  public EnMIMLNNmetric() throws com.mathworks.toolbox.javabuilder
      .MWException
  ```

  - **Description**

    No-argument constructor for xml configuration.

  - **Throws**

    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **EnMIMLNNmetric**

  ```
  public EnMIMLNNmetric(double ratio ,double mu) throws com.
      mathworks.toolbox.javabuilder.MWException
  ```

  - **Description**

    Basic constructor to initialize the classifier.

  - **Parameters**

    * `ratio` – The fraction parameter of EnMIMLNNmetric.
    * `mu` – The scaling factor of EnMIMLNNmetric.

  - **Throws**

    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **EnMIMLNNmetric**

  ```
  public EnMIMLNNmetric(double ratio ,double mu, int seed) throws
      com.mathworks.toolbox.javabuilder.MWException
  ```

  - **Description**

    Constructor to initialize the classifier.

  - **Parameters**

    * `ratio` – The fraction parameter of EnMIMLNNmetric.
    * `mu` – The scaling factor of EnMIMLNNmetric.
    * `seed` – Seed for kmedoids clustering.

  - **Throws**

    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

24.1.7 Methods

- **configure**

  ```
  public void configure(org.apache.commons.configuration2.
      Configuration configuration)
  ```

- **dispose**

  ```
  public abstract void dispose()
  ```

  - **Description copied from miml.classifiers.miml.MWClassifier** (**in 10.3, page 145**)

    Disposes native MW classifier. This method should be called if the classifier is not been used anymore in the program in order to free the memory that the MW classifier was using.

- **getMu**

  ```
  public double getMu()
  ```

  - **Description**

    Returns the scaling factor parameter considered to build the classifier.
  - **Returns** – The scaling factor parameter considered to build the classifier.

- **getRatio**

  ```
  public double getRatio()
  ```

  - **Description**

    Returns the fraction parameter considered to build the classifier.
  - **Returns** – The fraction parameter considered to build the classifier.

- **getSeed**

  ```
  public int getSeed()
  ```

  - **Description**

    Returns the seed for kmedoids clustering considered to build the classifier.

  - **Returns** – The seed for kmedoids clustering considered to build the classifier.

- **predictMWClassifier**

```
protected abstract java.lang.Object[] predictMWClassifier(com.
    mathworks.toolbox.javabuilder.MWCellArray train_bags,com.
    mathworks.toolbox.javabuilder.MWNumericArray train_targets,
    com.mathworks.toolbox.javabuilder.MWNumericArray test_bag)
    throws com.mathworks.toolbox.javabuilder.MWException
```

  - **Description copied from miml.classifiers.miml.MWClassifier** (**in** 10.3, **page** 145)

    Performs a prediction on a test bag.

  - **Parameters**

    * `train_bags` – Bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
    * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
    * `test_bag` – A test bag. It will be a MIMLBag in the format of a nInstxnAttributes MWNumericArray of double.

  - **Returns** – An array of 2 Object:

    * Object[0] is a nLabelsx1 array of double containing the probability of the testing instance belonging to each label.
    * Object[1] is a nLabelsx1 array of double containing a bipartition being 1 if the label is relevant or -1 otherwise.

  - **Throws**

    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **setMu**

```
public void setMu(double mu)
```

  - **Description**

    Sets the scaling factor parameter to build the classifier.

- **Parameters**
    - `mu` – The scaling factor of EnMIMLNNmetric.

- **setRatio**

```
public void setRatio(double ratio)
```

- **Description**
  Sets the fraction parameter to build the classifier.
- **Parameters**
    - `ratio` – The fraction parameter of EnMIMLNNmetric.

- **setSeed**

```
public void setSeed(int seed)
```

- **Description**
  Sets the seed for kmedoids clustering considered to build the classifier.
- **Parameters**
    - `seed` – The seed

- **trainMWClassifier**

```
protected abstract void trainMWClassifier(com.mathworks.toolbox.
    javabuilder.MWCellArray train_bags ,com.mathworks.toolbox.
    javabuilder.MWNumericArray train_targets) throws com.
    mathworks.toolbox.javabuilder.MWException
```

- **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**
  Trains a Matlab classifier. Returns the classifier model in an array of Object.
- **Parameters**
    - `train_bags` – bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.

    * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.

  – **Throws**

    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

24.1.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 10.3, page 145)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public abstract void` **dispose**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **aBag**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected abstract Object` **predictMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **test_bag**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `private static final` **serialVersionUID**
- `protected abstract void` **trainMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `protected` **wrapper**

24.1.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`

- `public final MultiLabelOutput` **makePrediction**(`weka.core.Instance` **instance**) `throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**(`miml.data.MIMLBag` **instance**) `throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**(`boolean` **debug**)

24.2 Class MIMLNN

Class to execute the **MIMLNN**algorithm for MIML data. For more information, see *Zhou, Z. H., Zhang, M. L., Huang, S. J., & Li, Y. F. (2012). Multi-instance multi-label learning. Artificial Intelligence, 176(1), 2291-2320.*.

24.2.1 Declaration

```
public class MIMLNN
 extends miml.classifiers.miml.MWClassifier
```

24.2.2 Field summary

lambda The regularization parameter used to compute matrix inverse, default=1.
mimlnn A matlab object wrapping the EnMIMLNNmetric algorithm.
ratio The number of clusters is set to ratio*numberOfTrainingBags, default=0.4.
seed The seed for kmedoids clustering
serialVersionUID For serialization.

24.2.3 Constructor summary

MIMLNN() No-argument constructor for xml configuration.
MIMLNN(double, double) Basic constructor to initialize the classifier.
MIMLNN(double, double, int) Constructor to initialize the classifier.

24.2.4 Method summary

configure(Configuration)
dispose()
getLambda() Returns the regularization parameter used to compute matrix inverse.
getRatio() Returns the fraction parameter considered to determine the number of clusters to build the classifier.
getSeed() Returns the seed for kmedoids clustering considered to build the classifier.
predictMWClassifier(MWCellArray, MWNumericArray, MWNumericArray)

setLambda(double) Sets the fraction parameter considered to determine the number of clusters to build the classifier.
setRatio(double) Sets the fraction parameter considered to determine the number of clusters to build the classifier.
setSeed(int) Sets the seed for kmedoids clustering considered to build the classifier.
trainMWClassifier(MWCellArray, MWNumericArray)

24.2.5 Fields

- `private static final long` **serialVersionUID**
  - For serialization.
- `static MWAlgorithms.MWMIMLNN` **mimlnn**
  - A matlab object wrapping the EnMIMLNNmetric algorithm.
- `double` **ratio**
  - The number of clusters is set to ratio*numberOfTrainingBags, default=0.4.
- `double` **lambda**
  - The regularization parameter used to compute matrix inverse, default=1.
- `int` **seed**
  - The seed for kmedoids clustering

24.2.6 Constructors

- **MIMLNN**

```
public MIMLNN() throws com.mathworks.toolbox.javabuilder.
    MWException
```

  - **Description**

    No-argument constructor for xml configuration.
  - **Throws**
    - `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **MIMLNN**

```
public MIMLNN(double ratio ,double lambda) throws com.mathworks.
    toolbox.javabuilder.MWException
```

  - **Description**

    Basic constructor to initialize the classifier.

  - **Parameters**

    * `ratio` – The number of clusters is set to ratio*numberOfTrainingBags.
    * `lambda` – The regularization parameter used to compute matrix inverse

  - **Throws**

    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **MIMLNN**

```
public MIMLNN(double ratio ,double lambda,int seed) throws com.
    mathworks.toolbox.javabuilder.MWException
```

  - **Description**

    Constructor to initialize the classifier.

  - **Parameters**

    * `ratio` – TThe number of clusters is set to ratio*numberOfTrainingBags.
    * `lambda` – The regularization parameter used to compute matrix inverse
    * `seed` – Seed for kmedoids clustering.

  - **Throws**

    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

24.2.7 Methods

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **dispose**

```
public abstract void dispose()
```

- **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**

  Disposes native MW classifier. This method should be called if the classifier is not been used anymore in the program in order to free the memory that the MW classifier was using.

- **getLambda**

```
public double getLambda()
```

  - **Description**

    Returns the regularization parameter used to compute matrix inverse.

  - **Returns** – The regularization parameter used to compute matrix inverse.

- **getRatio**

```
public double getRatio()
```

  - **Description**

    Returns the fraction parameter considered to determine the number of clusters to build the classifier.

  - **Returns** – The fraction parameter considered to determine the number of clusters to build the classifier.

- **getSeed**

```
public int getSeed()
```

  - **Description**

    Returns the seed for kmedoids clustering considered to build the classifier.

  - **Returns** – The seed for kmedoids clustering considered to build the classifier.

- **predictMWClassifier**

```
protected abstract java.lang.Object[] predictMWClassifier(com.
    mathworks.toolbox.javabuilder.MWCellArray train_bags,com.
    mathworks.toolbox.javabuilder.MWNumericArray train_targets,
    com.mathworks.toolbox.javabuilder.MWNumericArray test_bag)
    throws com.mathworks.toolbox.javabuilder.MWException
```

- **Description copied from miml.classifiers.miml.MWClassifier** (**in 10.3, page 145**)

  Performs a prediction on a test bag.

- **Parameters**

  * `train_bags` – Bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
  * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.
  * `test_bag` – A test bag. It will be a MIMLBag in the format of a nInstxnAttributes MWNumericArray of double.

- **Returns** – An array of 2 Object:

  * Object[0] is a nLabelsx1 array of double containing the probability of the testing instance belonging to each label.
  * Object[1] is a nLabelsx1 array of double containing a bipartition being 1 if the label is relevant or -1 otherwise.

- **Throws**

  * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

• **setLambda**

```
public void setLambda(double lambda)
```

- **Description**

  Sets the fraction parameter considered to determine the number of clusters to build the classifier.

- **Parameters**

  * `lambda` – The fraction parameter considered to determine the number of clusters to build the classifier.

• **setRatio**

```
public void setRatio(double ratio)
```

- **Description**

  Sets the fraction parameter considered to determine the number of clusters to build the classifier.

- **Parameters**

  * `ratio` – The fraction parameter considered to determine the number of clusters to build the classifier.

• **setSeed**

```
public void setSeed(int seed)
```

- **Description**

  Sets the seed for kmedoids clustering considered to build the classifier.

- **Parameters**

  * `seed` – The seed

• **trainMWClassifier**

```
protected abstract void trainMWClassifier(com.mathworks.toolbox.
    javabuilder.MWCellArray train_bags,com.mathworks.toolbox.
    javabuilder.MWNumericArray train_targets) throws com.
    mathworks.toolbox.javabuilder.MWException
```

- **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**

  Trains a Matlab classifier. Returns the classifier model in an array of Object.

- **Parameters**

  * `train_bags` – bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.

  * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.

- **Throws**

  * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

24.2.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 10.3, page 145)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public abstract void` **dispose**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **aBag**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected abstract Object` **predictMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **test_bag**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `private static final` **serialVersionUID**
- `protected abstract void` **trainMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `protected` **wrapper**

24.2.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

24.3 Class MIMLRBF

Class to execute the **MIMLRBF** algorithm for MIML data. For more information, see *Zhang, M. L., & Wang, Z. J. (2009). MIMLRBF: RBF neural networks for multi-instance multi-label learning. Neurocomputing, 72(16-18), 3951-3956.*.

24.3.1 Declaration

```
public class MIMLRBF
 extends miml.classifiers.miml.MWClassifier
```

24.3.2 Field summary

mimlrbf A matlab object wrapping the mimlrbf algorithm.
mu The ratio used to determine the standard deviation of the Gaussian activation function.
ratio The number of centroids of the i-th label is set to be ratio*Ti, where Ti is the number of train bags with label i.
seed Seed for kmedoids clustering.
serialVersionUID For serialization.

24.3.3 Constructor summary

MIMLRBF() No-argument constructor for xml configuration.
MIMLRBF(double, double) Basic constructor to initialize the classifier.
MIMLRBF(double, double, int) Constructor to initialize the classifier.

24.3.4 Method summary

configure(Configuration)
dispose()
getMu() Returns the scaling factor parameter considered to build the classifier.
getRatio() Returns the fraction parameter considered to build the classifier.
getSeed() Returns the seed for kmedoids clustering considered to build the classifier.
predictMWClassifier(MWCellArray, MWNumericArray, MWNumericArray)
setMu(double) Sets the scaling factor parameter to build the classifier.
setRatio(double) Sets the fraction parameter to build the classifier.
setSeed(int) Returns the seed for kmedoids clustering considered to build the classifier.
trainMWClassifier(MWCellArray, MWNumericArray)

24.3.5 Fields

- `private static final long` **serialVersionUID**

  – For serialization.

- `static MWAlgorithms.MWMIMLRBF` **mimlrbf**
  – A matlab object wrapping the mimlrbf algorithm.

- `double` **ratio**
  – The number of centroids of the i-th label is set to be ratio*Ti, where Ti is the number of train bags with label i.

- `double` **mu**
  – The ratio used to determine the standard deviation of the Gaussian activation function.

- `int` **seed**
  – Seed for kmedoids clustering.

24.3.6 Constructors

- **MIMLRBF**

```
public MIMLRBF() throws com.mathworks.toolbox.javabuilder.
    MWException
```

  – **Description**
    No-argument constructor for xml configuration.
  – **Throws**
    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **MIMLRBF**

```
public MIMLRBF(double ratio ,double mu) throws com.mathworks.
    toolbox.javabuilder.MWException
```

  – **Description**
    Basic constructor to initialize the classifier.
  – **Parameters**
    * `ratio` – The fraction parameter of MIMLRBF.
    * `mu` – The scaling factor of MIMLRBF.

- **Throws**
    - `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **MIMLRBF**

```
public MIMLRBF(double ratio ,double mu, int seed) throws com.
    mathworks.toolbox.javabuilder.MWException
```

  - **Description**

    Constructor to initialize the classifier.

  - **Parameters**
    - `ratio` – The fraction parameter of MIMLRBF.
    - `mu` – The scaling factor of MIMLRBF.
    - `seed` – Seed for kmedoids clustering.

  - **Throws**
    - `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

24.3.7 Methods

- **configure**

```
public void configure(org.apache.commons.configuration2.
    Configuration configuration)
```

- **dispose**

```
public abstract void dispose()
```

  - **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**

    Disposes native MW classifier. This method should be called if the classifier is not been used anymore in the program in order to free the memory that the MW classifier was using.

- **getMu**

```
public double getMu()
```

  - **Description**
    Returns the scaling factor parameter considered to build the classifier.
  - **Returns** – The scaling factor parameter considered to build the classifier.

- **getRatio**

```
public double getRatio()
```

  - **Description**
    Returns the fraction parameter considered to build the classifier.
  - **Returns** – The fraction parameter considered to build the classifier.

- **getSeed**

```
public int getSeed()
```

  - **Description**
    Returns the seed for kmedoids clustering considered to build the classifier.
  - **Returns** – The seed for kmedoids clustering considered to build the classifier.

- **predictMWClassifier**

```
protected abstract java.lang.Object[] predictMWClassifier(com.
    mathworks.toolbox.javabuilder.MWCellArray train_bags,com.
    mathworks.toolbox.javabuilder.MWNumericArray train_targets,
    com.mathworks.toolbox.javabuilder.MWNumericArray test_bag)
    throws com.mathworks.toolbox.javabuilder.MWException
```

  - **Description copied from miml.classifiers.miml.MWClassifier (in 10.3, page 145)**
    Performs a prediction on a test bag.
  - **Parameters**
    * `train_bags` – Bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.
    * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.

    * `test_bag` – A test bag. It will be a MIMLBag in the format of a nInstxnAttributes MWNumericArray of double.

  - **Returns** – An array of 2 Object:

    * Object[0] is a nLabelsx1 array of double containing the probability of the testing instance belonging to each label.
    * Object[1] is a nLabelsx1 array of double containing a bipartition being 1 if the label is relevant or -1 otherwise.

  - **Throws**

    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

- **setMu**

```
public void setMu(double mu)
```

  - **Description**

    Sets the scaling factor parameter to build the classifier.

  - **Parameters**

    * `mu` – The scaling factor of MIMLRBF.

- **setRatio**

```
public void setRatio(double ratio)
```

  - **Description**

    Sets the fraction parameter to build the classifier.

  - **Parameters**

    * `ratio` – The fraction parameter of MIMLRBF.

- **setSeed**

```
public void setSeed(int seed)
```

  - **Description**

    Returns the seed for kmedoids clustering considered to build the classifier.

  - **Parameters**

  * `seed` – Seed for kmedoids clustering.

- **trainMWClassifier**

```
protected abstract void trainMWClassifier(com.mathworks.toolbox.
    javabuilder.MWCellArray train_bags ,com.mathworks.toolbox.
    javabuilder.MWNumericArray train_targets) throws com.
    mathworks.toolbox.javabuilder.MWException
```

  - **Description copied from miml.classifiers.miml.MWClassifier** (**in** 10.3, **page** 145)

    Trains a Matlab classifier. Returns the classifier model in an array of Object.

  - **Parameters**

    * `train_bags` – bags in the MIMLInstances dataset in the format of a nBagsx1 MWCellArray in which the ith bag is stored in aCellArray{i,1}. Each bag is a nInstxnAttributes array of double values.

    * `train_targets` – Label associations of all bags in the MIMLInstances dataset in the format of a nLabelsxnBags MWNumericArray of double. If the ith bag belongs to the jth label, then aDoubleArray(j,i) equals +1, otherwise train_target(j,i) equals -1.

  - **Throws**

    * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

24.3.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 10.3, page 145)

- `protected void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected` **classifier**
- `public abstract void` **dispose**`()`
- `protected MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **aBag**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected abstract Object` **predictMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **test_bag**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `private static final` **serialVersionUID**
- `protected abstract void` **trainMWClassifier**`(com.mathworks.toolbox.javabuilder.MWCellArray` **train_bags**`, com.mathworks.toolbox.javabuilder.MWNumericArray` **train_targets**`) throws com.mathworks.toolbox.javabuilder.MWException`
- `protected` **wrapper**

24.3.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 10.2, page 141)

- `public final void` **build**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `public final void` **build**`(mulan.data.MultiLabelInstances` **trainingSet**`) throws java.lang.Exception`
- `protected abstract void` **buildInternal**`(miml.data.MIMLInstances` **trainingSet**`) throws java.lang.Exception`
- `protected void` **debug**`(java.lang.String` **msg**`)`
- `protected` **featureIndices**
- `public boolean` **getDebug**`()`
- `private` **isDebug**
- `protected` **isModelInitialized**
- `protected boolean` **isModelInitialized**`()`
- `public boolean` **isUpdatable**`()`
- `protected` **labelIndices**
- `protected` **labelNames**
- `public IMIMLClassifier` **makeCopy**`() throws java.lang.Exception`
- `public final MultiLabelOutput` **makePrediction**`(weka.core.Instance` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException, mulan.classifier.ModelInitializationException`
- `protected abstract MultiLabelOutput` **makePredictionInternal**`(miml.data.MIMLBag` **instance**`) throws java.lang.Exception, mulan.classifier.InvalidDataException`
- `protected` **numLabels**
- `private static final` **serialVersionUID**
- `public void` **setDebug**`(boolean` **debug**`)`

Chapter 25

Package miml.data.partitioning.iterative

Package Contents *Page*

25.1 Class IterativeCrossValidation

Class to carry out an stratified cross validation partition of multi-label dataset. MIML and MVML format is also supported. This java class is based on the mulan.data.IterativeStratification.java class provided in the Mulan java framework for multi-label learning Tsoumakas, G., Katakis, I., Vlahavas, I. (2010) "Mining Multi-label Data", Data Mining and Knowledge Discovery Handbook, O. Maimon, L. Rokach (Ed.), Springer, 2nd edition, 2010. The method is described in Sechidis, K.; Tsoumakas, G. and Vlahavas, I. Gunopulos, D.; Hofmann, T.; Malerba, D. and Vazirgiannis, M. (Eds.) On the Stratification of Multi-label Data Machine Learning and Knowledge Discovery in Databases, Springer Berlin Heidelberg, 2011, 6913, 145-158. Our contribution is the adaptation of method split to generate train-test partition.

25.1.1 Declaration

```
public class IterativeCrossValidation
 extends miml.data.partitioning.CrossValidationBase
```

25.1.2 Constructor summary

IterativeCrossValidation(int, MultiLabelInstances) Constructor.

IterativeCrossValidation(MultiLabelInstances) Default constructor.

25.1.3 Method summary

getFolds(int)

25.1.4 Constructors

- **IterativeCrossValidation**

```
public IterativeCrossValidation(int seed,mulan.data.
    MultiLabelInstances mlDataSet) throws mulan.data.
    InvalidDataFormatException
```

  - **Description**

    Constructor.

  - **Parameters**

    * `seed` – Seed for randomization
    * `mlDataSet` – A multi-label dataset

  - **Throws**

    * `mulan.data.InvalidDataFormatException` – To be handled

- **IterativeCrossValidation**

```
public IterativeCrossValidation(mulan.data.MultiLabelInstances
    mlDataSet) throws mulan.data.InvalidDataFormatException
```

  - **Description**

    Default constructor.

  - **Parameters**

    * `mlDataSet` – A multi-label dataset

  - **Throws**

    * `mulan.data.InvalidDataFormatException` – To be handled

25.1.5 Methods

- **getFolds**

```
public abstract mulan.data.MultiLabelInstances[] getFolds(int
    nFolds) throws mulan.data.InvalidDataFormatException
```

  - **Description copied from miml.data.partitioning.CrossValidationBase** (**in 21.1, page 284**)

    Splits a dataset into nfolds partitions.
  - **Parameters**
    - `nFolds` – Number of folds.
  - **Returns** – MultiLabelInstances[] a vector of nFolds. Each element represents a fold.
  - **Throws**
    - `mulan.data.InvalidDataFormatException` – To be handled.

25.1.6 Members inherited from class CrossValidationBase

`miml.data.partitioning.CrossValidationBase` (in 21.1, page 284)

- `public static MultiLabelInstances` **foldsToRounds**`(mulan.data.MultiLabelInstances[]` **Folds**`) throws java.lang.Exception`
- `public abstract MultiLabelInstances` **getFolds**`(int` **nFolds**`) throws mulan.data.InvalidDataFormatException`
- `public MultiLabelInstances` **getRounds**`(int` **nFolds**`) throws java.lang.Exception`
- `protected void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`

25.1.7 Members inherited from class PartitionerBase

`miml.data.partitioning.PartitionerBase` (in 21.2, page 287)

- `protected` **seed**
- `protected abstract void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`
- `public int` **totalExamples**`()`
- `protected` **workingSet**

25.2 Class IterativeTrainTest

Class to carry out an stratified iterativeTrainTest partition of multi-label dataset. MIML and MVML format is also supported. This java class is based on the mulan.data.IterativeStratification.java class provided in the Mulan java framework for multi-label learning Tsoumakas, G., Katakis, I., Vlahavas, I. (2010) "Mining Multi-label Data", Data Mining and Knowledge Discovery Handbook, O. Maimon, L. Rokach (Ed.), Springer, 2nd edition, 2010. The method is described in Sechidis, K.; Tsoumakas, G. and Vlahavas, I. Gunopulos, D.; Hofmann, T.; Malerba, D. and Vazirgiannis, M. (Eds.) On the Stratification of Multi-label Data Machine Learning and Knowledge Discovery in Databases, Springer Berlin Heidelberg, 2011, 6913, 145-158. Our contribution is the adaptation of method split to generate train-test partition.

25.2.1 Declaration

```
public class IterativeTrainTest
 extends miml.data.partitioning.TrainTestBase
```

25.2.2 Constructor summary

IterativeTrainTest(int, MultiLabelInstances) Constructor.
IterativeTrainTest(MultiLabelInstances) Default constructor.

25.2.3 Method summary

calculatingTheDesiredSplits(int[], double[], int, int) Returns the desired number of examples per label in each fold and in the last column the total desired number of examples in each fold.

calculatingTheFrequencies(Instances, int, int[]) Returns the number of examples per label in each fold.

findThePossibleSpit(double[][], int, int) Takes fold statistics and the index of the desired label (desired in the sense the label that we will apply the stratification sampling at this point) and it decides which are the folds that this instance can be inserted.

foldsCreation(Instances, Random, double[], int, int[], int)

getTrueLabels(Instance, int, int[]) Returns the relevant labels of one instance.

returnPossibleSplitsForNotAnnotated(double[][]) Returns the possible folds for the examples that are not annotated with any label.

split(double)

takeTheInstancesOfTheLabel(Instances, int, int[], int[]) Returns two sets of instances.

takingTheSmallestIndexAndNumberInVector(int[], int) Returns the rarest label and the number of examples that are annotated with that label.

updateDesiredSplitStatistics(double[], boolean[]) Updates the desired splits every time that an instance is inserted into a fold.

25.2.4 Constructors

- **IterativeTrainTest**

```
public IterativeTrainTest(int seed,mulan.data.
    MultiLabelInstances mlDataSet) throws mulan.data.
    InvalidDataFormatException
```

  - **Description**

    Constructor.

  - **Parameters**

    * `seed` – Seed for randomization
    * `mlDataSet` – A multi-label dataset

  - **Throws**

    * `mulan.data.InvalidDataFormatException` – To be handled

- **IterativeTrainTest**

```
public IterativeTrainTest(mulan.data.MultiLabelInstances
    mlDataSet) throws mulan.data.InvalidDataFormatException
```

  - **Description**

    Default constructor.

  - **Parameters**

    * `mlDataSet` – A multi-label dataset

  - **Throws**

    * `mulan.data.InvalidDataFormatException` – To be handled

25.2.5 Methods

- **calculatingTheDesiredSplits**

```
private double[][] calculatingTheDesiredSplits(int[]
    frequenciesOnDataset,double[] splitRatio,int numLabels,int
    totalNumberOfInstances)
```

- **Description**

  Returns the desired number of examples per label in each fold and in the last column the total desired number of examples in each fold.

- **Parameters**

  * `frequenciesOnDataset` –
  * `splitRatio` –
  * `numLabels` –
  * `totalNumberOfInstances` –

- **Returns** – double[][]

• **calculatingTheFrequencies**

```
private int[] calculatingTheFrequencies(weka.core.Instances
    dataSet, int numLabels, int[] labelIndices)
```

- **Description**

  Returns the number of examples per label in each fold.

- **Parameters**

  * `dataSet` – A dataset.
  * `numLabels` – Number of labels.
  * `labelIndices` – Array with label indices.

- **Returns** – int[]

• **findThePossibleSpit**

```
private int[] findThePossibleSpit(double[][] desiredSplit, int
    lab, int numFolds)
```

- **Description**

  Takes fold statistics and the index of the desired label (desired in the sense the label that we will apply the stratification sampling at this point) and it decides which are the folds that this instance can be inserted. The first priority is the fold with the smallest number of labels in the desired label. The second priority is the fold with the less number of instances.

- **Parameters**
  - `desiredSplit` –
  - `lab` –
  - `numFolds` –
- **Returns** – int[]

• **foldsCreation**

```
private weka.core.Instances[] foldsCreation(weka.core.Instances
    workingSet,java.util.Random random,double[] splitRatio ,int
    numLabels,int[] labelIndices ,int totalNumberOfInstances)
```

• **getTrueLabels**

```
private boolean[] getTrueLabels(weka.core.Instance instance ,int
    numLabels,int[] labelIndices)
```

- **Description**
  Returns the relevant labels of one instance.
- **Parameters**
  - `instance` – An instance
  - `numLabels` – The number of labels
  - `labelIndices` – The label indices
- **Returns** – boolean[]

• **returnPossibleSplitsForNotAnnotated**

```
private int[] returnPossibleSplitsForNotAnnotated(double[][]
    desiredSplit)
```

- **Description**
  Returns the possible folds for the examples that are not annotated with any label. In this special case the only criterion is the total number of examples in each fold.
- **Parameters**
  - `desiredSplit` –

- **Returns** – int[]

- **split**

```
public abstract mulan.data.MultiLabelInstances[] split(double
    percentageTrain) throws java.lang.Exception
```

  - **Description copied from miml.data.partitioning.TrainTestBase (in 21.3, page 289)**

    Returns a array with two multi-label random datasets corresponding to the train and test sets respectively.

  - **Parameters**

    * `percentageTrain` – Percentage of train dataset, a value in [0, 100].

  - **Returns** – MultiLabelInstances[].
    MultiLabelInstances[0] is the train set.
    MultiLabelInstances[1] is the test set.

  - **Throws**

    * `java.lang.Exception` – To be handled.

- **takeTheInstancesOfTheLabel**

```
private weka.core.Instances[] takeTheInstancesOfTheLabel(weka.
    core.Instances workingSet, int numLabels, int[] labelIndices,
    int[] desiredLabel)
```

  - **Description**

    Returns two sets of instances. The instances that are annotated with the label desiredLabel[0] and also returns the rest on the instances.

  - **Parameters**

    * `workingSet` –
    * `numLabels` –
    * `labelIndices` –
    * `desiredLabel` –

  - **Returns** – Instances[]

- **takingTheSmallestIndexAndNumberInVector**

```
private int[] takingTheSmallestIndexAndNumberInVector(int[]
    vectorSumOfLabels, int totalNumberOfInstances)
```

  - **Description**

    Returns the rarest label and the number of examples that are annotated with that label.

  - **Parameters**

    * `vectorSumOfLabels` –
    * `totalNumberOfInstances` –

  - **Returns** – int[]

- **updateDesiredSplitStatistics**

```
private double[] updateDesiredSplitStatistics(double[]
    desiredSplit, boolean[] trueLabels)
```

  - **Description**

    Updates the desired splits every time that an instance is inserted into a fold.

  - **Parameters**

    * `desiredSplit` –
    * `trueLabels` –

  - **Returns** – double[]

25.2.6 Members inherited from class TrainTestBase

`miml.data.partitioning.TrainTestBase` (in 21.3, page 289)

- `public abstract MultiLabelInstances` **split**`(double` **percentageTrain**`) throws java.lang.Exception`
- `protected void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`

25.2.7 Members inherited from class PartitionerBase

`miml.data.partitioning.PartitionerBase` (in 21.2, page 287)

- `protected` **seed**
- `protected abstract void` **statsToString**`(mulan.data.MultiLabelInstances[]` **Partition**`)`
- `public int` **totalExamples**`()`
- `protected` **workingSet**

Index