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

MIML version 1.3
November 21, 2022

Álvaro Andrés Belmonte Pérez
Amelia Zafra Gómez
Eva Lucrecia Gibaja Galindo

Contents

1	Package <code>miml.classifiers.miml.optimization</code>	17
1.1	Class <code>KiSar</code>	17
1.1.1	Declaration	17
1.1.2	Field summary	17
1.1.3	Constructor summary	18
1.1.4	Method summary	18
1.1.5	Fields	18
1.1.6	Constructors	19
1.1.7	Methods	20
1.1.8	Members inherited from class <code>MWClassifier</code>	23
1.1.9	Members inherited from class <code>MIMLClassifier</code>	24
1.2	Class <code>MIMLFast</code>	24
1.2.1	Declaration	24
1.2.2	Field summary	24
1.2.3	Constructor summary	25
1.2.4	Method summary	25
1.2.5	Fields	25
1.2.6	Constructors	26
1.2.7	Methods	27
1.2.8	Members inherited from class <code>MWClassifier</code>	32
1.2.9	Members inherited from class <code>MIMLClassifier</code>	33
1.3	Class <code>MIMLSVM</code>	33
1.3.1	Declaration	33
1.3.2	Field summary	34
1.3.3	Constructor summary	34
1.3.4	Method summary	34
1.3.5	Fields	34
1.3.6	Constructors	35
1.3.7	Methods	36
1.3.8	Members inherited from class <code>MWClassifier</code>	40
1.3.9	Members inherited from class <code>MIMLClassifier</code>	40
1.4	Class <code>MIMLWel</code>	41
1.4.1	Declaration	41
1.4.2	Field summary	41
1.4.3	Constructor summary	41
1.4.4	Method summary	41

1.4.5	Fields	42
1.4.6	Constructors	43
1.4.7	Methods	43
1.4.8	Members inherited from class MWClassifier	48
1.4.9	Members inherited from class MIMLClassifier	48
2	Package miml.data.partitioning	49
2.1	Class CrossValidationBase	49
2.1.1	Declaration	49
2.1.2	All known subclasses	49
2.1.3	Constructor summary	49
2.1.4	Method summary	50
2.1.5	Constructors	50
2.1.6	Methods	50
2.1.7	Members inherited from class PartitionerBase	51
2.2	Class PartitionerBase	52
2.2.1	Declaration	52
2.2.2	All known subclasses	52
2.2.3	Field summary	52
2.2.4	Constructor summary	52
2.2.5	Fields	52
2.2.6	Constructors	52
2.3	Class TrainTestBase	53
2.3.1	Declaration	53
2.3.2	All known subclasses	53
2.3.3	Constructor summary	53
2.3.4	Method summary	53
2.3.5	Constructors	54
2.3.6	Methods	54
2.3.7	Members inherited from class PartitionerBase	55
3	Package miml.classifiers.miml.mimlTOml	56
3.1	Class MIMLClassifierToML	56
3.1.1	Declaration	56
3.1.2	Field summary	56
3.1.3	Constructor summary	56
3.1.4	Method summary	57
3.1.5	Fields	57
3.1.6	Constructors	57
3.1.7	Methods	58
3.1.8	Members inherited from class MIMLClassifier	59
4	Package miml.classifiers.miml.neural	60
4.1	Class EnMIMLNNmetric	60
4.1.1	Declaration	60
4.1.2	Field summary	60
4.1.3	Constructor summary	61

4.1.4	Method summary	61
4.1.5	Fields	61
4.1.6	Constructors	61
4.1.7	Methods	62
4.1.8	Members inherited from class MWClassifier	65
4.1.9	Members inherited from class MIMLClassifier	66
4.2	Class MIMLNN	66
4.2.1	Declaration	66
4.2.2	Field summary	66
4.2.3	Constructor summary	67
4.2.4	Method summary	67
4.2.5	Fields	67
4.2.6	Constructors	67
4.2.7	Methods	68
4.2.8	Members inherited from class MWClassifier	71
4.2.9	Members inherited from class MIMLClassifier	72
4.3	Class MIMLRBF	72
4.3.1	Declaration	72
4.3.2	Field summary	72
4.3.3	Constructor summary	73
4.3.4	Method summary	73
4.3.5	Fields	73
4.3.6	Constructors	73
4.3.7	Methods	74
4.3.8	Members inherited from class MWClassifier	77
4.3.9	Members inherited from class MIMLClassifier	78
5	Package miml.data.partitioning.random	79
5.1	Class RandomCrossValidation	79
5.1.1	Declaration	79
5.1.2	Field summary	79
5.1.3	Constructor summary	80
5.1.4	Method summary	80
5.1.5	Fields	80
5.1.6	Constructors	80
5.1.7	Methods	81
5.1.8	Members inherited from class CrossValidationBase	81
5.1.9	Members inherited from class PartitionerBase	81
5.2	Class RandomTrainTest	81
5.2.1	Declaration	81
5.2.2	Constructor summary	81
5.2.3	Method summary	82
5.2.4	Constructors	82
5.2.5	Methods	82
5.2.6	Members inherited from class TrainTestBase	83
5.2.7	Members inherited from class PartitionerBase	83

6	Package <code>miml.classifiers.miml</code>	84
6.1	Interface <code>IMIMLClassifier</code>	84
6.1.1	Declaration	84
6.1.2	All known subinterfaces	84
6.1.3	All classes known to implement interface	85
6.1.4	Method summary	85
6.1.5	Methods	85
6.2	Class <code>MIMLClassifier</code>	86
6.2.1	Declaration	86
6.2.2	All known subclasses	86
6.2.3	Field summary	86
6.2.4	Constructor summary	86
6.2.5	Method summary	86
6.2.6	Fields	87
6.2.7	Constructors	87
6.2.8	Methods	88
6.3	Class <code>MWClassifier</code>	90
6.3.1	Declaration	90
6.3.2	All known subclasses	90
6.3.3	Field summary	90
6.3.4	Constructor summary	90
6.3.5	Method summary	91
6.3.6	Fields	91
6.3.7	Constructors	91
6.3.8	Methods	91
6.3.9	Members inherited from class <code>MIMLClassifier</code>	93
7	Package <code>miml.core</code>	95
7.1	Interface <code>IConfiguration</code>	95
7.1.1	Declaration	95
7.1.2	All known subinterfaces	95
7.1.3	All classes known to implement interface	96
7.1.4	Method summary	96
7.1.5	Methods	96
7.2	Class <code>ConfigLoader</code>	96
7.2.1	Declaration	96
7.2.2	Field summary	96
7.2.3	Constructor summary	96
7.2.4	Method summary	97
7.2.5	Fields	97
7.2.6	Constructors	97
7.2.7	Methods	97
7.3	Class <code>ConfigParameters</code>	98
7.3.1	Declaration	99
7.3.2	Field summary	99
7.3.3	Constructor summary	99

7.3.4	Method summary	99
7.3.5	Fields	99
7.3.6	Constructors	100
7.3.7	Methods	100
7.4	Class Params	103
7.4.1	Declaration	103
7.4.2	Field summary	103
7.4.3	Constructor summary	103
7.4.4	Method summary	103
7.4.5	Fields	103
7.4.6	Constructors	103
7.4.7	Methods	104
7.5	Class Utils	104
7.5.1	Declaration	104
7.5.2	Constructor summary	104
7.5.3	Method summary	105
7.5.4	Constructors	105
7.5.5	Methods	105
8	Package <code>miml.classifiers.ml</code>	106
8.1	Class MLDGC	106
8.1.1	Declaration	106
8.1.2	Field summary	106
8.1.3	Constructor summary	107
8.1.4	Method summary	107
8.1.5	Fields	107
8.1.6	Constructors	108
8.1.7	Methods	108
8.1.8	Members inherited from class <code>MultiLabelKNN</code>	110
8.1.9	Members inherited from class <code>MultiLabelLearnerBase</code>	110
8.2	Class <code>MLDGC.LinearNNESearch</code>	111
8.2.1	Declaration	111
8.2.2	Field summary	111
8.2.3	Constructor summary	111
8.2.4	Method summary	111
8.2.5	Fields	111
8.2.6	Constructors	111
8.2.7	Methods	111
8.2.8	Members inherited from class <code>LinearNNSearch</code>	111
8.2.9	Members inherited from class <code>NearestNeighbourSearch</code>	112
9	Package <code>miml.classifiers.miml.meta</code>	113
9.1	Class <code>MIMLBagging</code>	113
9.1.1	Declaration	113
9.1.2	Field summary	113
9.1.3	Constructor summary	114
9.1.4	Method summary	114

9.1.5	Fields	114
9.1.6	Constructors	115
9.1.7	Methods	116
9.1.8	Members inherited from class MIMLClassifier	119
10	Package miml.evaluation	120
10.1	Interface IEvaluator	120
10.1.1	Declaration	120
10.1.2	All known subinterfaces	120
10.1.3	All classes known to implement interface	120
10.1.4	Method summary	120
10.1.5	Methods	121
10.2	Class EvaluatorCV	121
10.2.1	Declaration	121
10.2.2	Field summary	121
10.2.3	Constructor summary	122
10.2.4	Method summary	122
10.2.5	Fields	122
10.2.6	Constructors	123
10.2.7	Methods	123
10.3	Class EvaluatorHoldout	126
10.3.1	Declaration	127
10.3.2	Field summary	127
10.3.3	Constructor summary	127
10.3.4	Method summary	127
10.3.5	Fields	127
10.3.6	Constructors	128
10.3.7	Methods	129
11	Package miml.transformation.mimlTOMi	131
11.1	Class BRTransformation	131
11.1.1	Declaration	131
11.1.2	Field summary	131
11.1.3	Constructor summary	131
11.1.4	Method summary	132
11.1.5	Fields	132
11.1.6	Constructors	132
11.1.7	Methods	132
11.2	Class LPTransformation	134
11.2.1	Declaration	134
11.2.2	Field summary	134
11.2.3	Constructor summary	134
11.2.4	Method summary	134
11.2.5	Fields	135
11.2.6	Constructors	135
11.2.7	Methods	135
11.3	Class MIMLLabelPowersetTransformation	136

11.3.1	Declaration	136
11.3.2	Field summary	136
11.3.3	Constructor summary	136
11.3.4	Method summary	136
11.3.5	Fields	136
11.3.6	Constructors	136
11.3.7	Methods	136
11.3.8	Members inherited from class <code>LabelPowersetTransformation</code>	137
12	Package <code>miml.data.normalization</code>	138
12.1	Class <code>MinMaxNormalization</code>	138
12.1.1	Declaration	138
12.1.2	Field summary	138
12.1.3	Constructor summary	138
12.1.4	Method summary	138
12.1.5	Fields	139
12.1.6	Constructors	139
12.1.7	Methods	139
13	Package <code>miml.data.statistics</code>	142
13.1	Class <code>MIMLStatistics</code>	142
13.1.1	Declaration	142
13.1.2	Field summary	142
13.1.3	Constructor summary	142
13.1.4	Method summary	143
13.1.5	Fields	144
13.1.6	Constructors	144
13.1.7	Methods	144
13.2	Class <code>MIStatistics</code>	153
13.2.1	Declaration	153
13.2.2	Field summary	153
13.2.3	Constructor summary	153
13.2.4	Method summary	153
13.2.5	Fields	153
13.2.6	Constructors	154
13.2.7	Methods	154
13.3	Class <code>MLStatistics</code>	155
13.3.1	Declaration	155
13.3.2	Field summary	156
13.3.3	Constructor summary	156
13.3.4	Method summary	156
13.3.5	Fields	157
13.3.6	Constructors	158
13.3.7	Methods	159

14 Package mimpl.tutorial	167
14.1 Class CrossValidationExperiment	167
14.1.1 Declaration	168
14.1.2 Constructor summary	168
14.1.3 Method summary	168
14.1.4 Constructors	168
14.1.5 Methods	168
14.2 Class GeneratePartitions	168
14.2.1 Declaration	168
14.2.2 Constructor summary	169
14.2.3 Method summary	169
14.2.4 Constructors	169
14.2.5 Methods	169
14.3 Class HoldoutExperiment	170
14.3.1 Declaration	170
14.3.2 Constructor summary	170
14.3.3 Method summary	170
14.3.4 Constructors	170
14.3.5 Methods	170
14.4 Class InsertingAttributesToBags	170
14.4.1 Declaration	171
14.4.2 Constructor summary	171
14.4.3 Method summary	171
14.4.4 Constructors	171
14.4.5 Methods	171
14.5 Class InsertingAttributeToBag	171
14.5.1 Declaration	171
14.5.2 Constructor summary	172
14.5.3 Method summary	172
14.5.4 Constructors	172
14.5.5 Methods	172
14.6 Class ManagingMIMLInstances	172
14.6.1 Declaration	172
14.6.2 Constructor summary	172
14.6.3 Method summary	172
14.6.4 Constructors	173
14.6.5 Methods	173
14.7 Class MIMLtoMITransformation	173
14.7.1 Declaration	173
14.7.2 Constructor summary	173
14.7.3 Method summary	173
14.7.4 Constructors	173
14.7.5 Methods	174
14.8 Class MIMLtoMLTransformation	174
14.8.1 Declaration	174
14.8.2 Constructor summary	174

14.8.3	Method summary	174
14.8.4	Constructors	174
14.8.5	Methods	174
14.9	Class NormalizingDataset	175
14.9.1	Declaration	175
14.9.2	Constructor summary	175
14.9.3	Method summary	175
14.9.4	Constructors	175
14.9.5	Methods	175
14.10	Class Resampling	175
14.10.1	Declaration	176
14.10.2	Constructor summary	176
14.10.3	Method summary	176
14.10.4	Constructors	176
14.10.5	Methods	176
15	Package miml.report	177
15.1	Interface IReport	177
15.1.1	Declaration	177
15.1.2	All known subinterfaces	177
15.1.3	All classes known to implement interface	177
15.1.4	Method summary	177
15.1.5	Methods	178
15.2	Class BaseMIMLReport	179
15.2.1	Declaration	179
15.2.2	Constructor summary	179
15.2.3	Method summary	179
15.2.4	Constructors	179
15.2.5	Methods	180
15.2.6	Members inherited from class MIMLReport	182
15.3	Class MIMLReport	182
15.3.1	Declaration	182
15.3.2	All known subclasses	183
15.3.3	Field summary	183
15.3.4	Constructor summary	183
15.3.5	Method summary	183
15.3.6	Fields	183
15.3.7	Constructors	184
15.3.8	Methods	184
16	Package miml.classifiers.mi	188
16.1	Class MISMOWrapper	188
16.1.1	Declaration	188
16.1.2	Field summary	188
16.1.3	Constructor summary	188
16.1.4	Method summary	188
16.1.5	Fields	188

16.1.6	Constructors	189
16.1.7	Methods	189
16.1.8	Members inherited from class MISMO	189
16.1.9	Members inherited from class AbstractClassifier	191
17	Package miml.transformation.mimlTOml	192
17.1	Class ArithmeticTransformation	192
17.1.1	Declaration	192
17.1.2	Field summary	193
17.1.3	Constructor summary	193
17.1.4	Method summary	193
17.1.5	Fields	193
17.1.6	Constructors	193
17.1.7	Methods	194
17.1.8	Members inherited from class MIMLtoML	195
17.2	Class GeometricTransformation	195
17.2.1	Declaration	195
17.2.2	Field summary	195
17.2.3	Constructor summary	195
17.2.4	Method summary	196
17.2.5	Fields	196
17.2.6	Constructors	196
17.2.7	Methods	196
17.2.8	Members inherited from class MIMLtoML	198
17.3	Class MIMLtoML	198
17.3.1	Declaration	198
17.3.2	All known subclasses	198
17.3.3	Field summary	198
17.3.4	Constructor summary	198
17.3.5	Method summary	199
17.3.6	Fields	199
17.3.7	Constructors	199
17.3.8	Methods	199
17.4	Class MinMaxTransformation	202
17.4.1	Declaration	202
17.4.2	Field summary	202
17.4.3	Constructor summary	202
17.4.4	Method summary	202
17.4.5	Fields	203
17.4.6	Constructors	203
17.4.7	Methods	203
17.4.8	Members inherited from class MIMLtoML	205
17.5	Class PropositionalTransformation	206
17.5.1	Declaration	206
17.5.2	Field summary	206
17.5.3	Constructor summary	206

17.5.4	Method summary	206
17.5.5	Fields	206
17.5.6	Constructors	207
17.5.7	Methods	208
18	Package miml.data	211
18.1	Class MIMLBag	211
18.1.1	Declaration	211
18.1.2	Field summary	211
18.1.3	Constructor summary	211
18.1.4	Method summary	211
18.1.5	Fields	212
18.1.6	Constructors	212
18.1.7	Methods	212
18.1.8	Members inherited from class DenseInstance	215
18.1.9	Members inherited from class AbstractInstance	216
18.2	Class MIMLInstances	217
18.2.1	Declaration	217
18.2.2	Field summary	217
18.2.3	Constructor summary	217
18.2.4	Method summary	217
18.2.5	Fields	218
18.2.6	Constructors	218
18.2.7	Methods	220
18.2.8	Members inherited from class MultiLabelInstances	226
18.3	Class MLSave	226
18.3.1	Declaration	227
18.3.2	Constructor summary	227
18.3.3	Method summary	227
18.3.4	Constructors	227
18.3.5	Methods	227
18.4	Class MWTranslator	230
18.4.1	Declaration	230
18.4.2	Field summary	230
18.4.3	Constructor summary	230
18.4.4	Method summary	230
18.4.5	Fields	230
18.4.6	Constructors	231
18.4.7	Methods	231
19	Package miml.run	235
19.1	Class RunAlgorithm	235
19.1.1	Declaration	235
19.1.2	Constructor summary	235
19.1.3	Method summary	235
19.1.4	Constructors	235
19.1.5	Methods	236

20 Package <code>miml.classifiers.miml.mimlTOMi</code>	237
20.1 Class <code>MIMLBinaryRelevance</code>	237
20.1.1 Declaration	237
20.1.2 Field summary	237
20.1.3 Constructor summary	237
20.1.4 Fields	238
20.1.5 Constructors	238
20.1.6 Members inherited from class <code>BinaryRelevance</code>	238
20.1.7 Members inherited from class <code>TransformationBasedMultiLabelLearner</code>	238
20.1.8 Members inherited from class <code>MultiLabelLearnerBase</code>	239
20.2 Class <code>MIMLClassifierToMI</code>	239
20.2.1 Declaration	239
20.2.2 Field summary	239
20.2.3 Constructor summary	239
20.2.4 Method summary	240
20.2.5 Fields	240
20.2.6 Constructors	240
20.2.7 Methods	240
20.2.8 Members inherited from class <code>MIMLClassifier</code>	242
20.3 Class <code>MIMLLabelPowerset</code>	242
20.3.1 Declaration	242
20.3.2 Field summary	242
20.3.3 Constructor summary	242
20.3.4 Method summary	243
20.3.5 Fields	243
20.3.6 Constructors	243
20.3.7 Methods	243
20.3.8 Members inherited from class <code>LabelPowerset</code>	243
20.3.9 Members inherited from class <code>TransformationBasedMultiLabelLearner</code>	244
20.3.10 Members inherited from class <code>MultiLabelLearnerBase</code>	244
21 Package <code>miml.data.partitioning.iterative</code>	245
21.1 Class <code>IterativeCrossValidation</code>	245
21.1.1 Declaration	245
21.1.2 Constructor summary	245
21.1.3 Method summary	246
21.1.4 Constructors	246
21.1.5 Methods	247
21.1.6 Members inherited from class <code>CrossValidationBase</code>	247
21.1.7 Members inherited from class <code>PartitionerBase</code>	247
21.2 Class <code>IterativeTrainTest</code>	247
21.2.1 Declaration	248
21.2.2 Constructor summary	248
21.2.3 Method summary	248
21.2.4 Constructors	248
21.2.5 Methods	249

21.2.6	Members inherited from class <code>TrainTestBase</code>	253
21.2.7	Members inherited from class <code>PartitionerBase</code>	253
22	Package <code>miml.core.distance</code>	254
22.1	Interface <code>IDistance</code>	254
22.1.1	Declaration	254
22.1.2	All known subinterfaces	254
22.1.3	All classes known to implement interface	255
22.1.4	Method summary	255
22.1.5	Methods	255
22.2	Class <code>AverageHausdorff</code>	256
22.2.1	Declaration	256
22.2.2	Field summary	256
22.2.3	Constructor summary	256
22.2.4	Method summary	256
22.2.5	Fields	256
22.2.6	Constructors	256
22.2.7	Methods	257
22.2.8	Members inherited from class <code>HausdorffDistance</code>	257
22.3	Class <code>HausdorffDistance</code>	257
22.3.1	Declaration	257
22.3.2	All known subclasses	257
22.3.3	Field summary	257
22.3.4	Constructor summary	257
22.3.5	Method summary	257
22.3.6	Fields	258
22.3.7	Constructors	258
22.3.8	Methods	258
22.4	Class <code>MaximalHausdorff</code>	259
22.4.1	Declaration	259
22.4.2	Field summary	259
22.4.3	Constructor summary	259
22.4.4	Method summary	259
22.4.5	Fields	259
22.4.6	Constructors	259
22.4.7	Methods	260
22.4.8	Members inherited from class <code>HausdorffDistance</code>	260
22.5	Class <code>MinimalHausdorff</code>	260
22.5.1	Declaration	260
22.5.2	Field summary	260
22.5.3	Constructor summary	260
22.5.4	Method summary	260
22.5.5	Fields	260
22.5.6	Constructors	261
22.5.7	Methods	261
22.5.8	Members inherited from class <code>HausdorffDistance</code>	261

23 Package miml.classifiers.miml.lazy	262
23.1 Class DMIMLkNN	263
23.1.1 Declaration	263
23.1.2 Field summary	263
23.1.3 Constructor summary	263
23.1.4 Method summary	263
23.1.5 Fields	263
23.1.6 Constructors	264
23.1.7 Methods	265
23.1.8 Members inherited from class MultiInstanceMultiLabelKNN	265
23.1.9 Members inherited from class MIMLClassifier	266
23.2 Class MIMLBRkNN	266
23.2.1 Declaration	266
23.2.2 Field summary	266
23.2.3 Constructor summary	267
23.2.4 Method summary	267
23.2.5 Fields	267
23.2.6 Constructors	267
23.2.7 Methods	268
23.2.8 Members inherited from class MultiInstanceMultiLabelKNN	269
23.2.9 Members inherited from class MIMLClassifier	270
23.3 Class MIMLDGC	270
23.3.1 Declaration	270
23.3.2 Field summary	270
23.3.3 Constructor summary	271
23.3.4 Method summary	271
23.3.5 Fields	271
23.3.6 Constructors	271
23.3.7 Methods	272
23.3.8 Members inherited from class MultiInstanceMultiLabelKNN	272
23.3.9 Members inherited from class MIMLClassifier	272
23.4 Class MIMLDistanceFunction	273
23.4.1 Declaration	273
23.4.2 Field summary	273
23.4.3 Constructor summary	273
23.4.4 Method summary	273
23.4.5 Fields	274
23.4.6 Constructors	274
23.4.7 Methods	274
23.4.8 Members inherited from class NormalizableDistance	276
23.5 Class MIMLFuzzykNN	278
23.5.1 Declaration	278
23.5.2 Field summary	278
23.5.3 Constructor summary	278
23.5.4 Fields	278
23.5.5 Constructors	279

23.5.6	Members inherited from class <code>MultiInstanceMultiLabelKNN</code>	279
23.5.7	Members inherited from class <code>MIMLClassifier</code>	279
23.6	Class <code>MIMLFuzzykNN.LinearNNEsearch</code>	280
23.6.1	Declaration	280
23.6.2	Field summary	280
23.6.3	Constructor summary	280
23.6.4	Method summary	280
23.6.5	Fields	280
23.6.6	Constructors	281
23.6.7	Methods	281
23.6.8	Members inherited from class <code>LinearNNEsearch</code>	281
23.6.9	Members inherited from class <code>NearestNeighbourSearch</code>	282
23.7	Class <code>MIMLIBLR</code>	282
23.7.1	Declaration	283
23.7.2	Field summary	283
23.7.3	Constructor summary	283
23.7.4	Method summary	283
23.7.5	Fields	283
23.7.6	Constructors	283
23.7.7	Methods	285
23.7.8	Members inherited from class <code>MultiInstanceMultiLabelKNN</code>	285
23.7.9	Members inherited from class <code>MIMLClassifier</code>	286
23.8	Class <code>MIMLkNN</code>	286
23.8.1	Declaration	286
23.8.2	Field summary	287
23.8.3	Constructor summary	287
23.8.4	Method summary	287
23.8.5	Fields	288
23.8.6	Constructors	289
23.8.7	Methods	289
23.8.8	Members inherited from class <code>MIMLClassifier</code>	294
23.9	Class <code>MIMLMAPkNN</code>	295
23.9.1	Declaration	295
23.9.2	Field summary	295
23.9.3	Constructor summary	295
23.9.4	Method summary	296
23.9.5	Fields	296
23.9.6	Constructors	296
23.9.7	Methods	297
23.9.8	Members inherited from class <code>MultiInstanceMultiLabelKNN</code>	298
23.9.9	Members inherited from class <code>MIMLClassifier</code>	298
23.10	Class <code>MultiInstanceMultiLabelKNN</code>	299
23.10.1	Declaration	299
23.10.2	All known subclasses	299
23.10.3	Field summary	299
23.10.4	Constructor summary	299

23.10.5 Method summary	299
23.10.6 Fields	300
23.10.7 Constructors	300
23.10.8 Methods	301
23.10.9 Members inherited from class <code>MIMLClassifier</code>	303
24 Package <code>miml.data.partitioning.powerset</code>	304
24.1 Class <code>LabelPowersetCrossValidation</code>	304
24.1.1 Declaration	304
24.1.2 Constructor summary	304
24.1.3 Method summary	304
24.1.4 Constructors	305
24.1.5 Methods	305
24.1.6 Members inherited from class <code>CrossValidationBase</code>	306
24.1.7 Members inherited from class <code>PartitionerBase</code>	306
24.2 Class <code>LabelPowersetTrainTest</code>	306
24.2.1 Declaration	306
24.2.2 Constructor summary	306
24.2.3 Method summary	306
24.2.4 Constructors	307
24.2.5 Methods	307
24.2.6 Members inherited from class <code>TrainTestBase</code>	308
24.2.7 Members inherited from class <code>PartitionerBase</code>	308

Chapter 1

Package miml.classifiers.miml.optimization

<i>Package Contents</i>	<i>Page</i>
Classes	
KiSar	17
Wrapper for Matlab KiSar algorithm for MIML data.	
MIMLFast	24
Wrapper for Matlab MIMLFast algorithm for MIML data.	
MIMLSVM	33
Wrapper for Matlab MIMLSVM algorithm for MIML data.	
MIMLWel	41
Wrapper for Matlab MIMLFast algorithm for MIML data.	

1.1 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.

1.1.1 Declaration

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

1.1.2 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.

1.1.3 Constructor summary

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

1.1.4 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)

1.1.5 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 cooccurrence values are greater than the mean of all values in the cooccurrence matrix (including main diagonal).
 - * 4 =>labels i,j coocur if their cooccurrence values are greater than the mean of the cooccurrence values of all labels (excluding main diagonal).
 - * 5 =>labels i,j coocur if $\text{prob}(i, j) > \min(\text{prob}(i), \text{prob}(j)) * 0.1$ (10 percent).

1.1.6 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**

Constructor 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.

1.1.7 Methods

- **configure**

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

- **dispose**

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

- **Description** copied from `miml.classifiers.miml.MWClassifier` (in [6.3](#), page [90](#))

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 [6.3](#), page [90](#))
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 6.3, page 90)**
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 `i`th bag is stored in a `CellArray{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 `i`th bag belongs to the `j`th label, then a `DoubleArray(j,i)` equals +1, otherwise `train_target(j,i)` equals -1.
- **Throws**
 - * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

1.1.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 6.3, page 90)

- `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`

1.1.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 6.2, page 86)

- `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)`

1.2 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.*

1.2.1 Declaration

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

1.2.2 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).

1.2.3 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.

1.2.4 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)

1.2.5 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**

1.2.6 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.

1.2.7 Methods

- **configure**

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

- **dispose**

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

– **Description copied from `miml.classifiers.miml.MWClassifier` (in [6.3](#), page [90](#))**

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 [6.3](#), page [90](#))

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 6.3, page 90)**

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 `i`th 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 `i`th bag belongs to the `j`th label, then `aDoubleArray(j,i)` equals `+1`, otherwise `train_target(j,i)` equals `-1`.

- **Throws**

- * `com.mathworks.toolbox.javabuilder.MWException` – To be handled.

1.2.8 Members inherited from class `MWClassifier`

`miml.classifiers.miml.MWClassifier` (in 6.3, page 90)

- `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**

1.2.9 Members inherited from class MIMLClassifier

miml.classifiers.miml.MIMLClassifier (in 6.2, page 86)

- 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**)

1.3 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.

1.3.1 Declaration

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

1.3.2 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 $\exp(-\text{Gamma} * \|x(i) - x(j)\|^2)$.

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.

1.3.3 Constructor summary

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

1.3.4 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)

1.3.5 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 $\exp(-\text{Gamma} * \|x(i) - x(j)\|^2)$.
 - * If type is "Poly", then para gives the value of gamma, coefficient, and degree respectively, where the kernel is $(\text{gamma} * \langle x(i), x(j) \rangle + \text{coefficient})^{\text{degree}}$. 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 $\langle x(i), x(j) \rangle$ (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.

1.3.6 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.

1.3.7 Methods

- **configure**

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

- **dispose**

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

- **Description copied from `miml.classifiers.miml.MWClassifier` (in [6.3](#), page [90](#))**

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 6.3, page 90)

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 `i`th 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 `i`th bag belongs to the `j`th 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 [6.3](#), page [90](#))
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.

1.3.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 6.3, page 90)

- 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**

1.3.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 6.2, page 86)

- 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)

1.4 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 (IJ-CAI'13), Beijing, China, 2013*.

1.4.1 Declaration

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

1.4.2 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*T_i, where T_i is the number of train bags with label i.
- serialVersionUID** For serialization.

1.4.3 Constructor summary

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

1.4.4 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)

1.4.5 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 $\text{ratio} \cdot T_i$, where T_i is the number of train bags with label i .
- **double mu**
 - The ratio used to determine the standard deviation of the Gaussian activation function.

1.4.6 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.

1.4.7 Methods

- **configure**

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

- **dispose**

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

- **Description** copied from `miml.classifiers.miml.MWClassifier` (in 6.3, page 90)

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 [6.3](#), page [90](#))
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 *i*th bag is stored in `aCellArray{i,1}`. Each bag is a `nInstxAttributes` 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 *i*th bag belongs to the *j*th 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 [6.3](#), page 90)**

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.

1.4.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 6.3, page 90)

- 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

1.4.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 6.2, page 86)

- 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 2

Package `miml.data.partitioning`

<i>Package Contents</i>	<i>Page</i>
Classes	
CrossValidationBase	49
General scheme for cross validation partitioners of multi-output data.	
PartitionerBase	52
General scheme for partitioning multi-output data.	
TrainTestBase	53
General scheme for train test partitioning of multi-output data.	

2.1 Class `CrossValidationBase`

General scheme for cross validation partitioners of multi-output data. MOR, MIML and MVML formats are also supported.

2.1.1 Declaration

```
public abstract class CrossValidationBase
    extends miml.data.partitioning.PartitionerBase
```

2.1.2 All known subclasses

`RandomCrossValidation` (in [5.1](#), page [79](#)), `IterativeCrossValidation` (in [21.1](#), page [245](#)), `LabelPower-setCrossValidation` (in [24.1](#), page [304](#))

2.1.3 Constructor summary

`CrossValidationBase(int, MultiLabelInstances)` Constructor.
`CrossValidationBase(MultiLabelInstances)` Default constructor.

2.1.4 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.

2.1.5 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

2.1.6 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.

2.1.7 Members inherited from class PartitionerBase

`miml.data.partitioning.PartitionerBase` (in [2.2](#), page [52](#))

- `protected seed`
- `protected workingSet`

2.2 Class PartitionerBase

General scheme for partitioning multi-output data.

2.2.1 Declaration

```
public abstract class PartitionerBase
  extends java.lang.Object
```

2.2.2 All known subclasses

TrainTestBase (in 2.3, page 53), CrossValidationBase (in 2.1, page 49), RandomTrainTest (in 5.2, page 81), RandomCrossValidation (in 5.1, page 79), IterativeTrainTest (in 21.2, page 247), IterativeCrossValidation (in 21.1, page 245), LabelPowersetTrainTest (in 24.2, page 306), LabelPowersetCrossValidation (in 24.1, page 304)

2.2.3 Field summary

seed Seed for reproduction of results
workingSet A copy of the instances to generate partitions

2.2.4 Constructor summary

PartitionerBase(int, MultiLabelInstances) Constructor of the class
PartitionerBase(MultiLabelInstances) Constructor of the class

2.2.5 Fields

- **protected int seed**
 - Seed for reproduction of results
- **protected mulan.data.MultiLabelInstances workingSet**
 - A copy of the instances to generate partitions

2.2.6 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.

2.3 Class TrainTestBase

General scheme for train test partitioning of multi-output data. MOR, MIML and MVML formats are also supported.

2.3.1 Declaration

```
public abstract class TrainTestBase
    extends miml.data.partitioning.PartitionerBase
```

2.3.2 All known subclasses

`RandomTrainTest` (in 5.2, page 81), `IterativeTrainTest` (in 21.2, page 247), `LabelPowersetTrainTest` (in 24.2, page 306)

2.3.3 Constructor summary

`TrainTestBase(int, MultiLabelInstances)` Constructor.
`TrainTestBase(MultiLabelInstances)` Default constructor.

2.3.4 Method summary

`split(double)` Returns a array with two multi-label random datasets corresponding to the train and test sets respectively.

2.3.5 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

2.3.6 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.

- **Returns** – MultiLabelInstances[].

MultiLabelInstances[0] is the train set.
MultiLabelInstances[1] is the test set.

- **Throws**

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

2.3.7 Members inherited from class PartitionerBase

miml.data.partitioning.PartitionerBase (in [2.2](#), page [52](#))

- protected seed
- protected workingSet

Chapter 3

Package

miml.classifiers.miml.mimlTOml

<i>Package Contents</i>	<i>Page</i>
Classes	
MIMLClassifierToML 56	
Class implementing the transformation algorithm for MIML data to solve it with ML learning.	

3.1 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).*

3.1.1 Declaration

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

3.1.2 Field summary

baseClassifier A Generic MultiLabel classifier.
mimlDataset The miml dataset.
mlDataSetWithBagId
removeFilter
serialVersionUID Generated Serial version UID.
transformationMethod The transform method.

3.1.3 Constructor summary

MIMLClassifierToML() No-argument constructor for xml configuration.

MIMLClassifierToML(MultiLabelLearner, MIMLtoML) Basic constructor to initialize the classifier.

3.1.4 Method summary

buildInternal(MIMLInstances)
configure(Configuration)
makePredictionInternal(MIMLBag)

3.1.5 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.
- **weka.filters.unsupervised.attribute.Remove removeFilter**
- **mulan.data.MultiLabelInstances mlDataSetWithBagId**

3.1.6 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.

3.1.7 Methods

- **buildInternal**

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

- **Description copied from `miml.classifiers.miml.MIMLClassifier` (in 6.2, page 86)**
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 6.2, page 86)**
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.

3.1.8 Members inherited from class MIMLClassifier

miml.classifiers.miml.MIMLClassifier (in [6.2](#), page [86](#))

- 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 4

Package `miml.classifiers.miml.neural`

<i>Package Contents</i>	<i>Page</i>
Classes	
EnMIMLNNmetric	60
Class to execute the EnMIMLNNmetric algorithm for MIML data.	
MIMLNN	66
Class to execute the MIMLNN algorithm for MIML data.	
MIMLRBF	72
Class to execute the MIMLRBF algorithm for MIML data.	

4.1 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..*

4.1.1 Declaration

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

4.1.2 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 $\text{ratio} \cdot T_i$, where T_i is the number of train bags with label i.

seed Seed for kmedoids clustering.

serialVersionUID For serialization.

4.1.3 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.

4.1.4 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)**

4.1.5 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.

4.1.6 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.

4.1.7 Methods

- **configure**

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

- **dispose**

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

- **Description** copied from `miml.classifiers.miml.MWClassifier` (in 6.3, page 90)

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 6.3, page 90)

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 6.3, page 90)**

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.

4.1.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 6.3, page 90)

- **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**

4.1.9 Members inherited from class MIMLClassifier

miml.classifiers.miml.MIMLClassifier (in 6.2, page 86)

- 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**)

4.2 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.*

4.2.1 Declaration

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

4.2.2 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.

4.2.3 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.

4.2.4 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)

4.2.5 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

4.2.6 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.

4.2.7 Methods

- **configure**

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

- **dispose**

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

- **Description copied from `miml.classifiers.miml.MWClassifier`** (in 6.3, page 90)

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 6.3, page 90)

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 6.3, page 90)**

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.

4.2.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 6.3, page 90)

- `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`

4.2.9 Members inherited from class MIMLClassifier

miml.classifiers.miml.MIMLClassifier (in 6.2, page 86)

- 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**)

4.3 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.*

4.3.1 Declaration

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

4.3.2 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*T_i, where T_i is the number of train bags with label i.
- seed** Seed for kmedoids clustering.
- serialVersionUID** For serialization.

4.3.3 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.

4.3.4 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)

4.3.5 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.

4.3.6 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.

4.3.7 Methods

- **configure**

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

- **dispose**

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

- **Description** copied from `miml.classifiers.miml.MWClassifier` (in 6.3, page 90)

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 6.3, page 90)

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 6.3, page 90)**

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.

4.3.8 Members inherited from class MWClassifier

`miml.classifiers.miml.MWClassifier` (in 6.3, page 90)

- **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**

4.3.9 Members inherited from class MIMLClassifier

miml.classifiers.miml.MIMLClassifier (in [6.2](#), page [86](#))

- 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 5

Package miml.data.partitioning.random

<i>Package Contents</i>	<i>Page</i>
Classes	
RandomCrossValidation	79
Class to split a multi-label dataset into N multi-label random datasets for cross-validation.	
RandomTrainTest	81
Class to split a multi-label dataset into two multi-label random datasets corresponding to the train and test datasets respectively.	

5.1 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 resulte train set.

5.1.1 Declaration

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

5.1.2 Field summary

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

5.1.3 Constructor summary

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

5.1.4 Method summary

getFolds(int)

5.1.5 Fields

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

5.1.6 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.

5.1.7 Methods

- `getFolds`

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

- **Description** copied from `miml.data.partitioning.CrossValidationBase` (in [2.1](#), page 49)
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.

5.1.8 Members inherited from class `CrossValidationBase`

`miml.data.partitioning.CrossValidationBase` (in [2.1](#), page 49)

- `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`

5.1.9 Members inherited from class `PartitionerBase`

`miml.data.partitioning.PartitionerBase` (in [2.2](#), page 52)

- `protected seed`
- `protected workingSet`

5.2 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.

5.2.1 Declaration

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

5.2.2 Constructor summary

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

5.2.3 Method summary

`split(double)`

5.2.4 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

5.2.5 Methods

- **split**

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

- **Description copied from `miml.data.partitioning.TrainTestBase` (in [2.3](#), page [53](#))**

Returns a array with two multi-label random datasets corresponding to the train and test sets respectively.

- **Parameters**

- * `percentageTrain` – Percentage of train dataset.
- **Returns** – `MultiLabelInstances[]`.
`MultiLabelInstances[0]` is the train set.
`MultiLabelInstances[1]` is the test set.
- **Throws**
 - * `java.lang.Exception` – To be handled.

5.2.6 Members inherited from class `TrainTestBase`

`miml.data.partitioning.TrainTestBase` (in [2.3](#), page [53](#))

- `public abstract MultiLabelInstances split(double percentageTrain)` throws `java.lang.Exception`

5.2.7 Members inherited from class `PartitionerBase`

`miml.data.partitioning.PartitionerBase` (in [2.2](#), page [52](#))

- `protected seed`
- `protected workingSet`

Chapter 6

Package `miml.classifiers.miml`

<i>Package Contents</i>	<i>Page</i>
Interfaces	
IMIMLClassifier	84
Common interface for MIML classifiers.	
Classes	
MIMLClassifier	86
This java class is based on the <code>mulan.data.Statistics.java</code> class provided in the Mulan java framework for multi-label learning <i>Tsoumakas, G., Katakis, I., Vlahavas, I. (2010) "Mining Multi-label Data", Data Mining and Knowledge Discovery Handbook, O.</i>	
MWClassifier	90
Class to execute Matlab MIML classifiers.	

6.1 Interface `IMIMLClassifier`

Common interface for MIML classifiers.

6.1.1 Declaration

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

6.1.2 All known subinterfaces

`MIMLWel` (in 1.4, page 41), `MIMLSVM` (in 1.3, page 33), `MIMLFast` (in 1.2, page 24), `KiSar` (in 1.1, page 17), `MIMLClassifierToML` (in 3.1, page 56), `MIMLRBF` (in 4.3, page 72), `MIMLNN` (in 4.2, page 66), `EnMIMLNNmetric` (in 4.1, page 60), `MWClassifier` (in 6.3, page 90), `MIMLClassifier` (in 6.2, page 86), `MIMLBagging` (in 9.1, page 113), `MIMLClassifierToMI` (in 20.2, page 239), `MultiInstanceMulti-LabelKNN` (in 23.10, page 299), `MIMLMApKNN` (in 23.9, page 295), `MIMLkNN` (in 23.8, page 286), `MIMLIBLR` (in 23.7, page 282), `MIMLFuzzykNN` (in 23.5, page 278), `MIMLDGC` (in 23.3, page 270), `MIMLBRkNN` (in 23.2, page 266), `DMIMLkNN` (in 23.1, page 263)

6.1.3 All classes known to implement interface

MIMLClassifier (in 6.2, page 86)

6.1.4 Method summary

build(MIMLInstances) Builds the learner model from specified MIMLInstances (in 18.2, page 217) data.

makeCopy()

makePrediction(Instance)

setDebug(boolean)

6.1.5 Methods

- **build**

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

- **Description**

Builds the learner model from specified MIMLInstances (in 18.2, page 217) 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)
```

6.2 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.

6.2.1 Declaration

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

6.2.2 All known subclasses

MIMLWel (in 1.4, page 41), MIMLSVM (in 1.3, page 33), MIMLFast (in 1.2, page 24), KiSar (in 1.1, page 17), MIMLClassifierToML (in 3.1, page 56), MIMLRBF (in 4.3, page 72), MIMLNN (in 4.2, page 66), EnMIMLNNmetric (in 4.1, page 60), MWClassifier (in 6.3, page 90), MIMLBagging (in 9.1, page 113), MIMLClassifierToMI (in 20.2, page 239), MultiInstanceMultiLabelKNN (in 23.10, page 299), MIMLMAPkNN (in 23.9, page 295), MIMLkNN (in 23.8, page 286), MIMLIBLR (in 23.7, page 282), MIMLFuzzykNN (in 23.5, page 278), MIMLDGC (in 23.3, page 270), MIMLBRkNN (in 23.2, page 266), DMIMLkNN (in 23.1, page 263)

6.2.3 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.

6.2.4 Constructor summary

`MIMLClassifier()`

6.2.5 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)

6.2.6 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.

6.2.7 Constructors

- **MIMLClassifier**

```
public MIMLClassifier()
```


6.2.8 Methods

- **build**

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

- **Description copied from IMIMLClassifier** (in [6.1](#), page [84](#))

Builds the learner model from specified MIMLInstances (in [18.2](#), page [217](#)) 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)
```

6.3 Class MWClassifier

Class to execute Matlab MIML classifiers.

6.3.1 Declaration

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

6.3.2 All known subclasses

MIMLWel (in 1.4, page 41), MIMLSVM (in 1.3, page 33), MIMLFast (in 1.2, page 24), KiSar (in 1.1, page 17), MIMLRBF (in 4.3, page 72), MIMLNN (in 4.2, page 66), EnMIMLNNmetric (in 4.1, page 60)

6.3.3 Field summary

classifier It will store the trained classifier.

serialVersionUID For serialization.

wrapper Wrapper for Matlab data types.

6.3.4 Constructor summary

```
MWClassifier()
```

6.3.5 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.

6.3.6 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.

6.3.7 Constructors

- **MWClassifier**

```
public MWClassifier()
```

6.3.8 Methods

- **buildInternal**

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

- **Description copied from MIMLClassifier** (in [6.2](#), page [86](#))
 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 6.2, page 86)**

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 `i`th 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 `i`th bag belongs to the `j`th 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.

6.3.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 6.2, page 86)

- `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.core

<i>Package Contents</i>	<i>Page</i>
Interfaces	
IConfiguration	95
Interface used to indicate that a class can be configured.	
Classes	
ConfigLoader	96
Class used to read a xml file and configure an experiment.	
ConfigParameters	98
Class used to save configuration parameters to be used in reports.	
Params	103
This class contains the list of classes and objects needed to create a new instance of a Multi Label classifier through a specific constructor.	
Utils	104
This class has utilities that can be used anywhere in the library.	

7.1 Interface IConfiguration

Interface used to indicate that a class can be configured.

7.1.1 Declaration

```
public interface IConfiguration
```

7.1.2 All known subinterfaces

MIMLWel (in 1.4, page 41), MIMLSVM (in 1.3, page 33), MIMLFast (in 1.2, page 24), KiSar (in 1.1, page 17), MIMLClassifierToML (in 3.1, page 56), MIMLRBF (in 4.3, page 72), MIMLNN (in 4.2, page 66), EnMIMLNNmetric (in 4.1, page 60), MWClassifier (in 6.3, page 90), MIMLClassifier (in 6.2, page 86), MIMLBagging (in 9.1, page 113), EvaluatorHoldout (in 10.3, page 126), EvaluatorCV (in 10.2, page 121), MIMLReport (in 15.3, page 182), BaseMIMLReport (in 15.2, page 179), MIMLClassifierToMI (in

20.2, page 239), MultiInstanceMultiLabelKNN (in 23.10, page 299), MIMLMAPkNN (in 23.9, page 295), MIMLkNN (in 23.8, page 286), MIMLIBLR (in 23.7, page 282), MIMLFuzzykNN (in 23.5, page 278), MIMLDGC (in 23.3, page 270), MIMLBRkNN (in 23.2, page 266), DMIMLkNN (in 23.1, page 263)

7.1.3 All classes known to implement interface

MIMLClassifier (in 6.2, page 86), EvaluatorHoldout (in 10.3, page 126), EvaluatorCV (in 10.2, page 121), MIMLReport (in 15.3, page 182)

7.1.4 Method summary

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

7.1.5 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.

7.2 Class ConfigLoader

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

7.2.1 Declaration

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

7.2.2 Field summary

configuration Configuration object.

7.2.3 Constructor summary

ConfigLoader(String) Constructor that sets the configuration file

7.2.4 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.

7.2.5 Fields

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

7.2.6 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.

7.2.7 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.

7.3 Class ConfigParameters

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

7.3.1 Declaration

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

7.3.2 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.

7.3.3 Constructor summary

ConfigParameters()

7.3.4 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.

7.3.5 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.

7.3.6 Constructors

- `ConfigParameters`

```
public ConfigParameters()
```

7.3.7 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.

7.4 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.

7.4.1 Declaration

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

7.4.2 Field summary

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

7.4.3 Constructor summary

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

7.4.4 Method summary

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

7.4.5 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

7.4.6 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.

7.4.7 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

7.5 Class Utils

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

7.5.1 Declaration

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

7.5.2 Constructor summary

```
Utils()
```

7.5.3 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.

7.5.4 Constructors

- **Utils**

```
public Utils()
```

7.5.5 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 8

Package `miml.classifiers.ml`

<i>Package Contents</i>	<i>Page</i>
Classes	
MLDGC	106
Implementation of MLDGC (Multi-Label Data Gravitation Model) algo-	
rithm.	
MLDGC.LinearNNESearch	111

8.1 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.*

8.1.1 Declaration

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

8.1.2 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

8.1.3 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.

8.1.4 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.

8.1.5 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**

8.1.6 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

8.1.7 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.

- **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.

8.1.8 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**

8.1.9 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)

8.2 Class MLDGC.LinearNNESearch

8.2.1 Declaration

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

8.2.2 Field summary

serialVersionUID For serialization

8.2.3 Constructor summary

LinearNNESearch(Instances)

8.2.4 Method summary

kNearestNeighboursIndices(Instance, int)

8.2.5 Fields

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

8.2.6 Constructors

- **LinearNNESearch**

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

8.2.7 Methods

- **kNearestNeighboursIndices**

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

8.2.8 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*

8.2.9 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*

Chapter 9

Package `miml.classifiers.miml.meta`

Package Contents

Page

Classes

MIMLBagging	113
MIMLBagging is the adaptation of the traditional bagging strategy of the machine learning [1] that does not need any previous transformation of the problem.	

9.1 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.

9.1.1 Declaration

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

9.1.2 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.

9.1.3 Constructor summary

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

9.1.4 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) Stablisthes whether confidences or bipartitions are used to combine classifiers in the ensemble.

9.1.5 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.

9.1.6 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.

9.1.7 Methods

- **buildInternal**

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

- **Description** copied from miml.classifiers.miml.MIMLClassifier (in [6.2](#), page [86](#))

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 [6.2](#), page [86](#))
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.

9.1.8 Members inherited from class MIMLClassifier

miml.classifiers.miml.MIMLClassifier (in [6.2](#), page [86](#))

- 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 10

Package `miml.evaluation`

<i>Package Contents</i>	<i>Page</i>
Interfaces	
IEvaluator	120
Interface for run and evaluate a experiment.	
Classes	
EvaluatorCV	121
Class that allow evaluate an algorithm applying a cross-validation method with random partitioning.	
EvaluatorHoldout	126
Class that allow evaluate an algorithm applying a holdout method.	

10.1 Interface `IEvaluator`

Interface for run and evaluate a experiment.

10.1.1 Declaration

```
public interface IEvaluator
```

10.1.2 All known subinterfaces

`EvaluatorHoldout` (in [10.3](#), page [126](#)), `EvaluatorCV` (in [10.2](#), page [121](#))

10.1.3 All classes known to implement interface

`EvaluatorHoldout` (in [10.3](#), page [126](#)), `EvaluatorCV` (in [10.2](#), page [121](#))

10.1.4 Method summary

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

10.1.5 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.

10.2 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.

10.2.1 Declaration

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

10.2.2 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.

10.2.3 Constructor summary

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

10.2.4 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.

10.2.5 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.

10.2.6 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.

10.2.7 Methods

- **configure**

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

- **Description copied from miml.core.IConfiguration (in 7.1, page 95)**

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 [10.1](#), page [120](#))
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 10.1, page 120)
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.

10.3 Class EvaluatorHoldout

Class that allow evaluate an algorithm applying a holdout method.

10.3.1 Declaration

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

10.3.2 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.

10.3.3 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.

10.3.4 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)

10.3.5 Fields

- protected `miml.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.

10.3.6 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.

10.3.7 Methods

- **configure**

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

– **Description copied from `miml.core.IConfiguration` (in 7.1, page 95)**

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 10.1, page 120)**

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 [10.1](#), page [120](#))
Run an experiment.
- **Parameters**
 - * **classifier** – The classifier used in the experiment.
- **Throws**
 - * **java.lang.Exception** – To be handled in an upper level.

Chapter 11

Package miml.transformation.mimlTOmi

<i>Package Contents</i>	<i>Page</i>
Classes	
BRTransformation	131
Class that uses Binary Relevance transformation to convert MIMLInstances to MIL Instances with relational attribute.	
LPTransformation	134
Class that uses LabelPowerset transformation to convert MIMLInstances to MIL Instances with relational attribute.	
MIMLLabelPowersetTransformation	136
Class that uses LabelPowerset transformation to convert MIMLInstances to MIL Instances with relational attribute.	

11.1 Class BRTransformation

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

11.1.1 Declaration

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

11.1.2 Field summary

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

11.1.3 Constructor summary

BRTransformation(MIMLInstances) Constructor.

11.1.4 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.

11.1.5 Fields

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

11.1.6 Constructors

- **BRTransformation**

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

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

11.1.7 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 $[0, \text{numLabels}-1]$.
- **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.Instanches transformBags(int labelToKeep) throws
    java.lang.Exception
```

- **Description**

Remove all label attributes except labelToKeep.
- **Parameters**
 - * `labelToKeep` – The label to keep. A value in $[0, \text{numLabels}-1]$.
- **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.

11.2 Class LPTransformation

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

11.2.1 Declaration

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

11.2.2 Field summary

LPT LabelPowerSetTransformation.
serialVersionUID For serialization.

11.2.3 Constructor summary

LPTransformation() Constructor.

11.2.4 Method summary

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

11.2.5 Fields

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

11.2.6 Constructors

- `LPTransformation`

```
public LPTransformation()
```

- **Description**
Constructor.

11.2.7 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.

11.3 Class MIMLLabelPowersetTransformation

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

11.3.1 Declaration

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

11.3.2 Field summary

```
serialVersionUID
```

11.3.3 Constructor summary

```
MIMLLabelPowersetTransformation()
```

11.3.4 Method summary

```
transformInstance(Instance, int[])
```

11.3.5 Fields

- `private static final long serialVersionUID`

11.3.6 Constructors

- `MIMLLabelPowersetTransformation`

```
MIMLLabelPowersetTransformation()
```

11.3.7 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.

11.3.8 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 12

Package `miml.data.normalization`

<i>Package Contents</i>	<i>Page</i>
Classes	
MinMaxNormalization	138
Class implementing min-max normalization for MIML datasets.	

12.1 Class `MinMaxNormalization`

Class implementing min-max normalization for MIML datasets.

12.1.1 Declaration

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

12.1.2 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

12.1.3 Constructor summary

`MinMaxNormalization()`

12.1.4 Method summary

`getMax()` Returns an array with the maximum values for all bag attributes in the dataset.

getMin() Returns an array with the minimum values for all bag attributes in the dataset.

getnFeatures() Returns the number of bag attributes in the dataset.

getRange() Returns 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.

12.1.5 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).

12.1.6 Constructors

- **MinMaxNormalization**

```
public MinMaxNormalization()
```

12.1.7 Methods

- **getMax**

```
public double [] getMax()
```

- **Description**

Returns 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 values, x , the new x' value will be $x' = (x - \min(x)) / (\max(x) - \min(x))$. 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 detects that all the attributes are not 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 13

Package `miml.data.statistics`

<i>Package Contents</i>	<i>Page</i>
Classes	
MIMLStatistics	142
Class with methods to obtain information about a MIML dataset.	
MIStatistics	153
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...	
MLStatistics	155
Class with methods to obtain information about a ML dataset.	

13.1 Class `MIMLStatistics`

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

13.1.1 Declaration

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

13.1.2 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.

13.1.3 Constructor summary

MIMLStatistics(MIMLInstances) Constructor.

13.1.4 Method summary

averageIR(double[]) Computes the average of any IR vector.

averageSkew(HashMap) Computes the average labelSkew.

calculateCooccurrence(MIMLInstances) This method calculates a matrix with the cooccurrences 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.

cooccurrenceToCSV() Returns cooCurrenceMatrix in CSV representation.

cooccurrenceToString() 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 \leq phi \leq bound.

varianceIR(double[]) Computes the variance of any IR vector.

13.1.5 Fields

- **miml.data.MIMLInstances dataSet**
 - A MIML data set
- **protected MStatistics mlstatistics**
 - Class with methods to obtain information about a MI dataset.
 - See also
 - * **MStatistics** (in [13.2](#), page [153](#))
- **protected MLStatistics mlstatistics**
 - Class with methods to obtain information about a ML dataset.
 - See also
 - * **MLStatistics** (in [13.3](#), page [155](#))

13.1.6 Constructors

- **MIMLStatistics**

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

- **Description**
Constructor.
- **Parameters**
 - * **dataSet** – A MIML data set.

13.1.7 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.

- **calculateCooccurrence**

```
public double [][] calculateCooccurrence(miml.data.MIMLInstances  
mlDataSet)
```

- **Description**

This method calculates a matrix with the cooccurrences of pairs of labels. It requires the method calculateStats to be previously called.

- **Parameters**

* **mlDataSet** – A multi-label dataset.

- **Returns** – A cooccurrences 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

- **cooccurrenceToCSV**

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

- **Description**

Returns cooCurrenceMatrix in CSV representation. It requires the method calculateCooccurrence to be previously called.

- **Returns** – CooCurrenceMatrix in CSV representation.

- **cooccurrenceToString**

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

- **Description**

Returns cooCurrenceMatrix in textual representation. It requires the method calculateCooccurrence 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 labelset 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 $\text{int}(1/\text{step}) + 1$ 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 Phi 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 (`pUnique`) 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.

13.2 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...

13.2.1 Declaration

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

13.2.2 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.

13.2.3 Constructor summary

MISTatistics(Instances)

13.2.4 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.

13.2.5 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

13.2.6 Constructors

- **MIStatistics**

```
public MIStatistics(weka.core.Instances dataSet)
```

13.2.7 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.

13.3 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`.

13.3.1 Declaration

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

13.3.2 Field summary

base The lowest labelSet count.
chi2 Chi square matrix values where 0 = complete independence.
cooccurrenceMatrix Cooccurrence 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.

13.3.3 Constructor summary

MLStatistics(MultiLabelInstances) Constructor.

13.3.4 Method summary

averageIR(double[]) Computes the average of any IR vector.
averageSkew(HashMap) Computes the average labelSkew.
calculateCooccurrence(MultiLabelInstances) This method calculates a matrix with the cooccurrences 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.
cooccurrenceToCSV() Returns cooccurrenceMatrix in CSV representation.
cooccurrenceToString() Returns cooccurrenceMatrix 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 $\text{negativePatterns} / \text{positivePatterns}$.

interClassIR() Computes the interClassIR for each label $\text{positiveExamplesOfMajorityLabel} / \text{positivePatternsLabel}$.

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 $(\text{patterns of majorityLabelSet}) / (\text{patterns of the labelSet})$.

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 $\text{peak} / \text{base}$.

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 $-\text{bound} \leq \text{phi} \leq \text{bound}$.

varianceIR(double[]) Computes the variance of any IR vector.

13.3.5 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[] [] cooccurrenceMatrix`
 - Cooccurrence 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

13.3.6 Constructors

- **MLStatistics**

```
public MLStatistics(mulan.data.MultiLabelInstances mlDataSet)
```

- **Description**
Constructor.
- **Parameters**
* `mlDataSet` – MultiLabel dataset.

13.3.7 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• **calculateCooccurrence**

```
public double[][] calculateCooccurrence(mulan.data.
    MultiLabelInstances mlDataSet)
```

– **Description**

This method calculates a matrix with the cooccurrences of pairs of labels. It requires the method calculateStats to be previously called.

– **Parameters**

* mlDataSet – A multi-label dataset.

– **Returns** – A cooccurrences 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

- **cooccurrenceToCSV**

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

- **Description**

- Returns `cooccurrenceMatrix` in CSV representation. It requires the method `calculateCooccurrence` to be previously called.

- **Returns** – string

- **cooccurrenceToString**

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

- **Description**

- Returns `cooccurrenceMatrix` in textual representation. It requires the method `calculateCooccurrence` 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 labelset 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 $\text{int}(1/\text{step}) + 1$ 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 Phi 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 14

Package miml.tutorial

<i>Package Contents</i>	<i>Page</i>
Classes	
CrossValidationExperiment	167
Class implementing an example of using cross-validation with different kinds of classifier.	
GeneratePartitions	168
Class to split a multi-output dataset into partitions for cross-validation or train-test.	
HoldoutExperiment	170
Class implementing an example of using holdout with train/test dataset and a single dataset applying percentage split.	
InsertingAttributesToBags	170
Class implementing an example of inserting a new group of attributes to the relational attribute of the dataset with {0,1} values.	
InsertingAttributeToBag	171
Class implementing an example of inserting a new attribute to the relational attribute of the dataset with {0,1} values.	
ManagingMIMLInstances	172
Class implementing basic handling of MIML datasets.	
MIMLtoMITransformation	173
Class for basic handling of MIML to MIL LP and BR transformation.	
MIMLtoMLTransformation	174
Class for basic handling of the transformation MIML to ML transformations.	
NormalizingDataset	175
Class to show an example of normalization of a MIML Dataset.	
Resampling	175
Class to show an example of sampling with replacement.	

14.1 Class CrossValidationExperiment

Class implementing an example of using cross-validation with different kinds of classifier.

14.1.1 Declaration

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

14.1.2 Constructor summary

CrossValidationExperiment()

14.1.3 Method summary

main(String[])
showUse() Shows the help on command line.

14.1.4 Constructors

- **CrossValidationExperiment**

```
public CrossValidationExperiment()
```

14.1.5 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.

14.2 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.

14.2.1 Declaration

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

14.2.2 Constructor summary

GeneratePartitions()

14.2.3 Method summary

main(String[]) Main method.

showUse() Shows the help on command line.

14.2.4 Constructors

- **GeneratePartitions**

```
public GeneratePartitions()
```

14.2.5 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.

14.3 Class HoldoutExperiment

Class implementing an example of using holdout with train/test dataset and a single dataset applying percentage split.

14.3.1 Declaration

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

14.3.2 Constructor summary

HoldoutExperiment()

14.3.3 Method summary

main(String[])
showUse() Shows the help on command line.

14.3.4 Constructors

- **HoldoutExperiment**

```
public HoldoutExperiment()
```

14.3.5 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.

14.4 Class InsertingAttributesToBags

Class implementing an example of inserting a new group of attributes to the relational attribute of the dataset with {0,1} values.

14.4.1 Declaration

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

14.4.2 Constructor summary

InsertingAttributesToBags()

14.4.3 Method summary

main(String[])
showUse() Shows the help on command line.

14.4.4 Constructors

- **InsertingAttributesToBags**

```
public InsertingAttributesToBags()
```

14.4.5 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.

14.5 Class InsertingAttributeToBag

Class implementing an example of inserting a new attribute to the relational attribute of the dataset with {0,1} values.

14.5.1 Declaration

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

14.5.2 Constructor summary

`InsertingAttributeToBag()`

14.5.3 Method summary

`main(String[])`

`showUse()` Shows the help on command line.

14.5.4 Constructors

- `InsertingAttributeToBag`

`public InsertingAttributeToBag()`

14.5.5 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.

14.6 Class ManagingMIMLInstances

Class implementing basic handling of MIML datasets.

14.6.1 Declaration

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

14.6.2 Constructor summary

`ManagingMIMLInstances()`

14.6.3 Method summary

`main(String[])`

`showUse()` Shows the help on command line.

14.6.4 Constructors

- **ManagingMIMLInstances**

```
public ManagingMIMLInstances()
```

14.6.5 Methods

- **main**

```
public static void main(java.lang.String[] args)
```

- **showUse**

```
public static void showUse()
```

- **Description**

Shows the help on command line.

14.7 Class MIMLtoMITransformation

Class for basic handling of MIML to MIL LP and BR transformation.

14.7.1 Declaration

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

14.7.2 Constructor summary

```
MIMLtoMITransformation()
```

14.7.3 Method summary

```
main(String[])
showUse() Shows the help on command line.
```

14.7.4 Constructors

- **MIMLtoMITransformation**

```
public MIMLtoMITransformation()
```

14.7.5 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.

14.8 Class MIMLtoMLTransformation

Class for basic handling of the transformation MIML to ML transformations.

14.8.1 Declaration

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

14.8.2 Constructor summary

```
MIMLtoMLTransformation()
```

14.8.3 Method summary

```
main(String[])
showUse() Shows the help on command line.
```

14.8.4 Constructors

- **MIMLtoMLTransformation**

```
public MIMLtoMLTransformation()
```

14.8.5 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.

14.9 Class NormalizingDataset

Class to show an example of normalization of a MIML Dataset.

14.9.1 Declaration

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

14.9.2 Constructor summary

```
NormalizingDataset()
```

14.9.3 Method summary

```
main(String[])
```

14.9.4 Constructors

- **NormalizingDataset**

```
public NormalizingDataset()
```

14.9.5 Methods

- **main**

```
public static void main(java.lang.String[] args)
```

14.10 Class Resampling

Class to show an example of sampling with replacement.

14.10.1 Declaration

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

14.10.2 Constructor summary

```
  Resampling()
```

14.10.3 Method summary

```
  main(String[])
```

14.10.4 Constructors

- **Resampling**

```
  public Resampling()
```

14.10.5 Methods

- **main**

```
  public static void main(java.lang.String[] args) throws java.  
    lang.Exception
```

Chapter 15

Package miml.report

<i>Package Contents</i>	<i>Page</i>
Interfaces	
IReport 177	
Interface for generate reports with the format specified.	
Classes	
BaseMIMLReport 179	
Class used to generate reports with the format specified.	
MIMLReport 182	
Abstract class for a MIMLReport.	

15.1 Interface IReport

Interface for generate reports with the format specified.

15.1.1 Declaration

```
public interface IReport
```

15.1.2 All known subinterfaces

MIMLReport (in 15.3, page 182), BaseMIMLReport (in 15.2, page 179)

15.1.3 All classes known to implement interface

MIMLReport (in 15.3, page 182)

15.1.4 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.

15.1.5 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.

15.2 Class BaseMIMLReport

Class used to generate reports with the format specified.

15.2.1 Declaration

```
public class BaseMIMLReport
    extends miml.report.MIMLReport
```

15.2.2 Constructor summary

BaseMIMLReport() No-argument constructor for xml configuration.
BaseMIMLReport(List, String, boolean, boolean, boolean) Basic constructor to initialize the report.

15.2.3 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)

15.2.4 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.

15.2.5 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
```

15.2.6 Members inherited from class MIMLReport

miml.report.MIMLReport (in [15.3](#), page [182](#))

- 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

15.3 Class MIMLReport

Abstract class for a MIMLReport.

15.3.1 Declaration

```
public abstract class MIMLReport
extends java.lang.Object implements IReport, miml.core.
    IConfiguration
```

15.3.2 All known subclasses

BaseMIMLReport (in [15.2](#), page [179](#))

15.3.3 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.

15.3.4 Constructor summary

MIMLReport() No-argument constructor for xml configuration.
MIMLReport(List, String, boolean, boolean, boolean) Basic constructor to initialize the report.

15.3.5 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).

15.3.6 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.

15.3.7 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.

15.3.8 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 16

Package `miml.classifiers.mi`

<i>Package Contents</i>	<i>Page</i>
Classes	
MISMOWrapper	188
Wrapper for MISMO algorithm to work in MIML to MI classifiers.	

16.1 Class `MISMOWrapper`

Wrapper for MISMO algorithm to work in MIML to MI classifiers.

16.1.1 Declaration

```
public class MISMOWrapper
    extends weka.classifiers.mi.MISMO
```

16.1.2 Field summary

serialVersionUID Generated Serial version UID.

16.1.3 Constructor summary

MISMOWrapper()

16.1.4 Method summary

distributionForInstance(Instance)

16.1.5 Fields

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

16.1.6 Constructors

- **MISMOWrapper**

```
public MISMOWrapper()
```

16.1.7 Methods

- **distributionForInstance**

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

16.1.8 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()`

16.1.9 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 17

Package miml.transformation.mimlTOml

<i>Package Contents</i>	<i>Page</i>
Classes	
ArithmeticTransformation	192
Class that performs an arithmetic transformation to convert a MIMLIn-	
stances class to MultiLabelInstances.	
GeometricTransformation	195
Class that performs a geometric transformation to convert a MIMLInstances	
class to MultiLabelInstances.	
MIMLtoML	198
Abstract class to transform MIMLInstances into MultiLabelInstances.	
MinMaxTransformation	202
Class that performs a miniMaxc transformation to convert a MIMLInstances	
class to MultiLabelInstances.	
PropositionalTransformation	206
Class that performs a propositionalTransformation to convert a MIMLIn-	
stances dataset to MultiLabelInstances.	

17.1 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.

17.1.1 Declaration

```
public class ArithmeticTransformation
    extends miml.transformation.mimlTOml.MIMLtoML
```

17.1.2 Field summary

`serialVersionUID` For serialization

17.1.3 Constructor summary

`ArithmeticTransformation()`

`ArithmeticTransformation(MIMLInstances)` Constructor.

17.1.4 Method summary

`transformDataset()`

`transformDataset(MIMLInstances)`

`transformInstance(MIMLBag)`

`transformInstance(MIMLInstances, MIMLBag)`

17.1.5 Fields

- `private static final long serialVersionUID`
 - For serialization

17.1.6 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.

17.1.7 Methods

- **transformDataset**

```
public abstract mulan.data.MultiLabelInstances transformDataset(
    ) throws java.lang.Exception
```

- **Description copied from MIMLtoML** (in 17.3, page 198)
Transforms MIMLInstances (in 18.2, page 217) into MultiLabelInstances.
- **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 17.3, page 198)
Transforms MIMLInstances (in 18.2, page 217) 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 17.3, page 198)
Transforms MIMLBag (in 18.1, page 211) 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
```

17.1.8 Members inherited from class MIMLtoML

`miml.transformation.mimlTOml.MIMLtoML` (in 17.3, page 198)

- 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`

17.2 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 center of its max and min values computed as $(\text{min_value} + \text{max_value})/2$.

17.2.1 Declaration

```
public class GeometricTransformation
    extends miml.transformation.mimlTOml.MIMLtoML
```

17.2.2 Field summary

`serialVersionUID` For serialization

17.2.3 Constructor summary

`GeometricTransformation()`
`GeometricTransformation(MIMLInstances)` Constructor

17.2.4 Method summary

```
transformDataset()
transformDataset(MIMLInstances)
transformInstance(MIMLBag)
transformInstance(MIMLInstances, MIMLBag)
```

17.2.5 Fields

- `private static final long serialVersionUID`
 - For serialization

17.2.6 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.

17.2.7 Methods

- **transformDataset**

```
public abstract mulan.data.MultiLabelInstances transformDataset
    () throws java.lang.Exception
```

- **Description copied from MIMLtoML** (in [17.3](#), page [198](#))

Transforms MIMLInstances (in [18.2](#), page [217](#)) into MultiLabelInstances.

- **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 [17.3](#), page [198](#))

- Transforms `MIMLInstances` (in [18.2](#), page [217](#)) 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 [17.3](#), page [198](#))

- Transforms `MIMLBag` (in [18.1](#), page [211](#)) 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
```

17.2.8 Members inherited from class MIMLtoML

`miml.transformation.mimlTOMl.MIMLtoML` (in [17.3](#), page [198](#))

- 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`

17.3 Class MIMLtoML

Abstract class to transform MIMLInstances into MultiLabelInstances.

17.3.1 Declaration

```
public abstract class MIMLtoML
    extends java.lang.Object implements java.io.Serializable
```

17.3.2 All known subclasses

`MinMaxTransformation` (in [17.4](#), page [202](#)), `GeometricTransformation` (in [17.2](#), page [195](#)), `ArithmeticTransformation` (in [17.1](#), page [192](#))

17.3.3 Field summary

dataset Original data set of MIMLInstances.
serialVersionUID For serialization.
template Template to store Instances.
updatedLabelIndices Array of updated label indices.

17.3.4 Constructor summary

`MIMLtoML()`

17.3.5 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 18.2, page 217) into `MultiLabelInstances`.
- transformDataset(MIMLInstances)** Transforms `MIMLInstances` (in 18.2, page 217) into `MultiLabelInstances`.
- transformInstance(MIMLBag)** Transforms `MIMLBag` (in 18.1, page 211) into `Instance`.

17.3.6 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`.

17.3.7 Constructors

- `MIMLtoML`

```
public MIMLtoML()
```

17.3.8 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 18.2, page 217) into `MultiLabelInstances`.

- **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 18.2, page 217) 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 18.1, page 211) into `Instance`.

- **Parameters**

- * `bag` – The Bag to be transformed.

- **Returns** – `Instance`

- **Throws**

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

17.4 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}
```

17.4.1 Declaration

```
public class MinMaxTransformation
extends miml.transformation.mimlTOMl.MIMLtoML
```

17.4.2 Field summary

```
serialVersionUID For serialization
```

17.4.3 Constructor summary

```
MinMaxTransformation()
MinMaxTransformation(MIMLInstances) Constructor.
```

17.4.4 Method summary

```
prepareTemplate()
```

```

transformDataset()
transformDataset(MIMLInstances)
transformInstance(MIMLBag)
transformInstance(MIMLInstances, MIMLBag)

```

17.4.5 Fields

- `private static final long serialVersionUID`
 - For serialization

17.4.6 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.

17.4.7 Methods

- `prepareTemplate`

```
protected void prepareTemplate() throws java.lang.Exception
```

- **Description copied from MIMLtoML (in [17.3](#), page [198](#))**

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 [17.3](#), page [198](#))

Transforms `MIMLInstances` (in [18.2](#), page [217](#)) into `MultiLabelInstances`.

– **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 [17.3](#), page [198](#))

Transforms `MIMLInstances` (in [18.2](#), page [217](#)) 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 [17.3](#), page [198](#))

Transforms MIMLBag (in [18.1](#), page [211](#)) 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
```

17.4.8 Members inherited from class MIMLtoML

`miml.transformation.mimlTOML.MIMLtoML` (in [17.3](#), page [198](#))

- `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`

17.5 Class PropositionalTransformation

Class that performs a propositionalTransformation to convert a MIMLInstances dataset to MultiLabelInstances. This transformation transforms each Bag into a set of instances, one for each instance in the bag of the instances in the bag.

17.5.1 Declaration

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

17.5.2 Field summary

dataset Original data set of MIMLInstances.
includeBagId Whether bag attribute will be included in the transformed data
removeFilter Filter
template Template to store Instances.
updatedLabelIndices Array of updated label indices.

17.5.3 Constructor summary

PropositionalTransformation(MIMLInstances) Constructor.
PropositionalTransformation(MIMLInstances, boolean) Constructor.

17.5.4 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)

17.5.5 Fields

- **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

17.5.6 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.

17.5.7 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 18

Package miml.data

<i>Package Contents</i>	<i>Page</i>
Classes	
MIMLBag	211
Class inheriting from DenseInstance to represent a MIML bag.	
MIMLInstances	217
Class inheriting from MultiLabelInstances to represent MIML data.	
MLSave	226
Class with methods to write to file a multi-label dataset.	
MWTranslator	230
Class to serve as interface between MIMLInstances and Matlab data types.	

18.1 Class MIMLBag

Class inheriting from DenseInstance to represent a MIML bag.

18.1.1 Declaration

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

18.1.2 Field summary

serialVersionUID Generated Serial version UID.

18.1.3 Constructor summary

MIMLBag(Instance) Constructor.

18.1.4 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.

18.1.5 Fields

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

18.1.6 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.

18.1.7 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.

18.1.8 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)**

18.1.9 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_Attributes
- 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()

18.2 Class MIMLInstances

Class inheriting from MultiLabelInstances to represent MIML data.

18.2.1 Declaration

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

18.2.2 Field summary

serialVersionUID Generated Serial version UID.

18.2.3 Constructor summary

MIMLInstances(Instances, LabelsMetaData) Constructor.

MIMLInstances(Instances, String) Constructor.

MIMLInstances(MIMLInstances) Constructor.

MIMLInstances(String, int) Constructor.

MIMLInstances(String, String) Constructor.

18.2.4 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 18.1, page 211) (i.e. pattern) with a certain bagIndex.

getBagAsInstances(int) Gets a MIMLBag (in 18.1, page 211) 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 '??'

splitData(MIMLInstances, double, int, int) Split MIML data train and test partition given a percentage and a partitioning method.

18.2.5 Fields

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

18.2.6 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 dataset of MIMLInstances (in [18.2](#), page [217](#)).

- **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.

18.2.7 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 [18.1](#), page [211](#)) (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 18.1, page 211) 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.

- **splitData**

```
public static java.util.List 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.

18.2.8 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`

18.3 Class MLSave

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

18.3.1 Declaration

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

18.3.2 Constructor summary

MLSave()

18.3.3 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.

18.3.4 Constructors

- MLSave

```
private MLSave()
```

18.3.5 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.

18.4 Class MWTranslator

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

18.4.1 Declaration

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

18.4.2 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

18.4.3 Constructor summary

MWTranslator(MIMLInstances) Constructor.

18.4.4 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.

18.4.5 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

18.4.6 Constructors

- **MWTranslator**

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

- **Description**

Constructor.

- **Parameters**

* `mimlDataSet` – A MIML dataset.

18.4.7 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 `j`th 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 `j`th 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 19

Package `miml.run`

<i>Package Contents</i>	<i>Page</i>
Classes	
RunAlgorithm	235
Class that allow run any algorithm of the library configured by a file configuration.	

19.1 Class `RunAlgorithm`

Class that allow run any algorithm of the library configured by a file configuration.

19.1.1 Declaration

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

19.1.2 Constructor summary

```
RunAlgorithm()
```

19.1.3 Method summary

```
main(String[]) The main method to configure and run an algorithm.
```

19.1.4 Constructors

- `RunAlgorithm`

```
public RunAlgorithm()
```

19.1.5 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 20

Package

miml.classifiers.miml.mimlTOmi

<i>Package Contents</i>	<i>Page</i>
Classes	
MIMLBinaryRelevance	237
Wrapper for mulan BinaryRelevance to be used in MIML to MI algorithms.	
MIMLClassifierToMI	239
Class implementing the transformation algorithm for MIML data to solve it with MI learning.	
MIMLLabelPowerset	242
Wrapper for mulan LabelPowerset to be used in MIML to MI algorithms.	

20.1 Class MIMLBinaryRelevance

Wrapper for mulan BinaryRelevance to be used in MIML to MI algorithms.

20.1.1 Declaration

```
public class MIMLBinaryRelevance
    extends mulan.classifier.transformation.BinaryRelevance
```

20.1.2 Field summary

serialVersionUID Generated Serial version UID.

20.1.3 Constructor summary

MIMLBinaryRelevance(Classifier) Creates a new instance.

20.1.4 Fields

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

20.1.5 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.

20.1.6 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)`

20.1.7 Members inherited from class `TransformationBasedMultiLabelLearner`

`mulan.classifier.transformation.TransformationBasedMultiLabelLearner`

- `protected baseClassifier`
- `public Classifier getBaseClassifier()`
- `public TechnicalInformation getTechnicalInformation()`
- `public String globalInfo()`

20.1.8 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)`

20.2 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).

20.2.1 Declaration

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

20.2.2 Field summary

serialVersionUID Generated Serial version UID.
transformationClassifier Generic classifier used for transformation.

20.2.3 Constructor summary

MIMLClassifierToMI() No-argument constructor for xml configuration.
MIMLClassifierToMI(MultiLabelLearner) Basic constructor.

20.2.4 Method summary

```
buildInternal(MIMLInstances)
configure(Configuration)
makePredictionInternal(MIMLBag)
```

20.2.5 Fields

- `private static final long serialVersionUID`
 - Generated Serial version UID.
- `protected mulan.classifier.MultiLabelLearner transformationClassifier`
 - Generic classifier used for transformation.

20.2.6 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.

20.2.7 Methods

- `buildInternal`

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

- **Description** copied from `miml.classifiers.miml.MIMLClassifier` (in 6.2, page 86)

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 6.2, page 86)

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.

20.2.8 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 6.2, page 86)

- `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)`

20.3 Class MIMLLabelPowerset

Wrapper for `mulan.LabelPowerset` to be used in MIML to MI algorithms.

20.3.1 Declaration

```
public class MIMLLabelPowerset
    extends mulan.classifier.transformation.LabelPowerset
```

20.3.2 Field summary

`serialVersionUID` Generated Serial version UID.

20.3.3 Constructor summary

`MIMLLabelPowerset(Classifier)` Constructor that initializes the learner with a base classifier.

20.3.4 Method summary

buildInternal(MultiLabelInstances)

20.3.5 Fields

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

20.3.6 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.

20.3.7 Methods

- **buildInternal**

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

20.3.8 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

20.3.9 Members inherited from class TransformationBasedMultiLabelLearner

`mulan.classifier.transformation.TransformationBasedMultiLabelLearner`

- `protected baseClassifier`
- `public Classifier getBaseClassifier()`
- `public TechnicalInformation getTechnicalInformation()`
- `public String globalInfo()`

20.3.10 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 21

Package miml.data.partitioning.iterative

<i>Package Contents</i>	<i>Page</i>
Classes	
IterativeCrossValidation	245
Class to carry out an stratified cross validation partition of multi-label dataset.	
IterativeTrainTest	247
Class to carry out an stratified iterativeTrainTest partition of multi-label dataset.	

21.1 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.

21.1.1 Declaration

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

21.1.2 Constructor summary

IterativeCrossValidation(int, MultiLabelInstances) Constructor.

IterativeCrossValidation(MultiLabelInstances) Default constructor.

21.1.3 Method summary

getFolds(int)

21.1.4 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

21.1.5 Methods

- **getFolds**

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

- **Description** copied from `miml.data.partitioning.CrossValidationBase` (in [2.1](#), page [49](#))

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.

21.1.6 Members inherited from class `CrossValidationBase`

`miml.data.partitioning.CrossValidationBase` (in [2.1](#), page [49](#))

- `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`

21.1.7 Members inherited from class `PartitionerBase`

`miml.data.partitioning.PartitionerBase` (in [2.2](#), page [52](#))

- `protected seed`
- `protected workingSet`

21.2 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.

21.2.1 Declaration

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

21.2.2 Constructor summary

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

21.2.3 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.

21.2.4 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

21.2.5 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 [2.3](#), page [53](#))

Returns a array with two multi-label random datasets corresponding to the train and test sets respectively.

- **Parameters**

* `percentageTrain` – Percentage of train dataset.

- **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[]`

21.2.6 Members inherited from class `TrainTestBase`

`miml.data.partitioning.TrainTestBase` (in 2.3, page 53)

- `public abstract MultiLabelInstances split(double percentageTrain)` throws `java.lang.Exception`

21.2.7 Members inherited from class `PartitionerBase`

`miml.data.partitioning.PartitionerBase` (in 2.2, page 52)

- `protected seed`
- `protected workingSet`

Chapter 22

Package `miml.core.distance`

<i>Package Contents</i>	<i>Page</i>
Interfaces	
IDistance	254
Interface to implement the metrics used to measure the distance between MIMLBag (in 18.1 , page 211) of a data sets.	
Classes	
AverageHausdorff	256
Class that implements Average Hausdorff metric to measure the distance between 2 bags of a data set.	
HausdorffDistance	257
MaximalHausdorff	259
Class that implements Maximal Hausdorff metric to measure the distance between 2 bags of a data set.	
MinimalHausdorff	260
Class that implements Minimal Hausdorff metric to measure the distance between 2 bags of a data set.	

22.1 Interface `IDistance`

Interface to implement the metrics used to measure the distance between MIMLBag (in [18.1](#), page [211](#)) of a data sets.

22.1.1 Declaration

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

22.1.2 All known subinterfaces

MinimalHausdorff (in [22.5](#), page [260](#)), MaximalHausdorff (in [22.4](#), page [259](#)), HausdorffDistance (in

[22.3](#), page [257](#)), AverageHausdorff (in [22.2](#), page [256](#))

22.1.3 All classes known to implement interface

HausdorffDistance (in [22.3](#), page [257](#))

22.1.4 Method summary

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 [18.1](#), page [211](#)).

22.1.5 Methods

- **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 [18.1](#), page [211](#)).

- **Parameters**

- * **first** – First bag.

- * `second` – Second bag.
- **Returns** – Distance between two bags.
- **Throws**
 - * `java.lang.Exception` – if occurred an error during distance calculation,

22.2 Class `AverageHausdorff`

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

22.2.1 Declaration

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

22.2.2 Field summary

`serialVersionUID` Generated Serial version UID.

22.2.3 Constructor summary

`AverageHausdorff()`
`AverageHausdorff(MIMLInstances)`

22.2.4 Method summary

`distance(Instances, Instances)`

22.2.5 Fields

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

22.2.6 Constructors

- **`AverageHausdorff`**

```
public AverageHausdorff()
```

- **`AverageHausdorff`**

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

22.2.7 Methods

- **distance**

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

22.2.8 Members inherited from class HausdorffDistance

miml.core.distance.HausdorffDistance (in [22.3](#), page [257](#))

- **dataSet**
- **dfun**
- **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**(miml.data.MIMLInstances bags) throws java.lang.Exception
- **public void update**(miml.data.MIMLBag bag) throws java.lang.Exception

22.3 Class HausdorffDistance

22.3.1 Declaration

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

22.3.2 All known subclasses

MinimalHausdorff (in [22.5](#), page [260](#)), MaximalHausdorff (in [22.4](#), page [259](#)), AverageHausdorff (in [22.2](#), page [256](#))

22.3.3 Field summary

```
dataSet  
dfun  
serialVersionUID
```

22.3.4 Constructor summary

```
HausdorffDistance()  
HausdorffDistance(MIMLInstances)
```

22.3.5 Method summary

```
distance(MIMLBag, MIMLBag)  
hasInstances()  
setInstances(MIMLInstances)  
update(MIMLBag)
```

22.3.6 Fields

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

22.3.7 Constructors

- **HausdorffDistance**

```
public HausdorffDistance()
```

- **HausdorffDistance**

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

22.3.8 Methods

- **distance**

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

- **Description copied from IDistance** (in [22.1](#), page [254](#))

Get the distance between two MIMLBag (in [18.1](#), page [211](#)).

- **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**

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

- **update**

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

22.4 Class MaximalHausdorff

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

22.4.1 Declaration

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

22.4.2 Field summary

serialVersionUID Generated Serial version UID.

22.4.3 Constructor summary

```
MaximalHausdorff()
MaximalHausdorff(MIMLInstances)
```

22.4.4 Method summary

```
distance(Instances, Instances)
```

22.4.5 Fields

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

22.4.6 Constructors

- **MaximalHausdorff**

```
public MaximalHausdorff()
```

- **MaximalHausdorff**

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

22.4.7 Methods

- **distance**

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

22.4.8 Members inherited from class HausdorffDistance

miml.core.distance.HausdorffDistance (in 22.3, page 257)

- **dataSet**
- **dfun**
- **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**(miml.data.MIMLInstances bags) throws java.lang.Exception
- **public void update**(miml.data.MIMLBag bag) throws java.lang.Exception

22.5 Class MinimalHausdorff

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

22.5.1 Declaration

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

22.5.2 Field summary

serialVersionUID Generated Serial version UID.

22.5.3 Constructor summary

```
MinimalHausdorff()
MinimalHausdorff(MIMLInstances)
```

22.5.4 Method summary

```
distance(Instances, Instances)
```

22.5.5 Fields

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

22.5.6 Constructors

- **MinimalHausdorff**

```
public MinimalHausdorff()
```

- **MinimalHausdorff**

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

22.5.7 Methods

- **distance**

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

22.5.8 Members inherited from class HausdorffDistance

miml.core.distance.HausdorffDistance (in [22.3](#), page [257](#))

- **dataSet**
- **dfun**
- **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(miml.data.MIMLInstances bags) throws java.lang.Exception**
- **public void update(miml.data.MIMLBag bag) throws java.lang.Exception**

Chapter 23

Package `miml.classifiers.miml.lazy`

<i>Package Contents</i>	<i>Page</i>
Classes	
DMIMLkNN	263
DMIMLkNN is the adaptation to the MIML framework of the DMLkNN[1] multi-label algorithm.	
MIMLBRkNN	266
MIMLBRkNN is the adaptation to the MIML framework of the BRkNN[1] multi-label algorithm.	
MIMLDGC	270
MIMLDGC is the adaptation to the MIML framework of the MLDGC[1] multi-label algorithm.	
MIMLDistanceFunction	273
Wrapper for using IDistance metrics of MIML package with Mulan Lazy algorithms.	
MIMLFuzzykNN	278
MIMLFuzzykNN.LinearNNESearch	280
MIMLIBLR	282
MIMLIBLR is the adaptation to the MIML framework of the IBLR_ML[1] multi-label algorithm.	
MIMLkNN	286
Class implementing the MIMLkNN algorithm for MIML data.	
MIMLMAPkNN	295
MIMLMAPkNN is the adaptation to the MIML framework of the MLkNN[1] multi-label algorithm.	
MultiInstanceMultiLabelKNN	299
Wrapper for class MultiLabelKNN of Mulan to work with MIML data	

23.1 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.*

23.1.1 Declaration

```
public class DMIMLkNN
    extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

23.1.2 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).

23.1.3 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.

23.1.4 Method summary

configure(Configuration)
getSmooth() Gets the smooth factor considered by the classifier.
setSmooth(double) Sets the smooth factor considered by the classifier.

23.1.5 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).

23.1.6 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.

23.1.7 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

23.1.8 Members inherited from class MultiInstanceMultiLabelKNN

miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN (in 23.10, page 299)

- 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)

23.1.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 6.2, page 86)

- `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)`

23.2 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.*

23.2.1 Declaration

```
public class MIMLBRkNN
    extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

23.2.2 Field summary

extension The type of extension to be used:

- NONE: Standard BR.

serialVersionUID Generated Serial version UID.

23.2.3 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.

23.2.4 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`).

23.2.5 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.

23.2.6 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`).

23.2.7 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.

23.2.8 Members inherited from class `MultiInstanceMultiLabelKNN`

`miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN` (in 23.10, page 299)

- `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)`

23.2.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 6.2, page 86)

- `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)`

23.3 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.*

23.3.1 Declaration

```
public class MIMLDGC
    extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

23.3.2 Field summary

`serialVersionUID` For serialization.

23.3.3 Constructor summary

MIMLDGC() No-arg constructor for xml configuration

MIMLDGC(MIMLDistanceFunction) Default constructor.

MIMLDGC(MIMLDistanceFunction, int) A constructor that sets the number of neighbours.

23.3.4 Method summary

configure(Configuration)

23.3.5 Fields

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

23.3.6 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.

23.3.7 Methods

- **configure**

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

23.3.8 Members inherited from class MultiInstanceMultiLabelKNN

miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN (in 23.10, page 299)

- **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)

23.3.9 Members inherited from class MIMLClassifier

miml.classifiers.miml.MIMLClassifier (in 6.2, page 86)

- **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)`

23.4 Class MIMLDistanceFunction

Wrapper for using IDistance metrics of MIML package with Mulan Lazy algorithms.

23.4.1 Declaration

```
public class MIMLDistanceFunction
    extends weka.core.NormalizableDistance
```

23.4.2 Field summary

metric Metric to measure distance between bags.
serialVersionUID

23.4.3 Constructor summary

MIMLDistanceFunction(IDistance) Constructor that sets the metric to be used.

23.4.4 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)

```

23.4.5 Fields

- `private static final long serialVersionUID`
- `protected miml.core.distance.IDistance metric`
 - Metric to measure distance between bags.

23.4.6 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.

23.4.7 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  
)
```

23.4.8 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()`

23.5 Class MIMLFuzzykNN

23.5.1 Declaration

```
public class MIMLFuzzykNN
    extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

23.5.2 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

23.5.3 Constructor summary

MIMLFuzzykNN()

23.5.4 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.

23.5.5 Constructors

- MIMLFuzzykNN

```
public MIMLFuzzykNN()
```

23.5.6 Members inherited from class MultiInstanceMultiLabelKNN

miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN (in [23.10](#), page [299](#))

- 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**)

23.5.7 Members inherited from class MIMLClassifier

miml.classifiers.miml.MIMLClassifier (in [6.2](#), page [86](#))

- 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**)

23.6 Class MIMLFuzzykNN.LinearNNESearch

23.6.1 Declaration

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

23.6.2 Field summary

serialVersionUID For serialization

23.6.3 Constructor summary

LinearNNESearch(Instances)

23.6.4 Method summary

kNearestNeighboursIndices(Instance, int)

23.6.5 Fields

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

23.6.6 Constructors

- **LinearNNESearch**

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

23.6.7 Methods

- **kNearestNeighboursIndices**

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

23.6.8 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`

23.6.9 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`

23.7 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.*

23.7.1 Declaration

```
public class MIMLIBLR
    extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

23.7.2 Field summary

addFeatures By default, IBLR-ML is used (addFeatures is false).
serialVersionUID Generated Serial version UID.

23.7.3 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.

23.7.4 Method summary

configure(Configuration)
getAddFeatures() Gets the value of addFeatures.
setAddFeatures(boolean) Sets the value of AddFeatures.

23.7.5 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.

23.7.6 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.

23.7.7 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.

23.7.8 Members inherited from class MultiInstanceMultiLabelKNN

miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN (in 23.10, page 299)

- 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)

23.7.9 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in 6.2, page 86)

- `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)`

23.8 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.*

23.8.1 Declaration

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

23.8.2 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.

23.8.3 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.

23.8.4 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.

23.8.5 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.

23.8.6 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.

23.8.7 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 neighbor'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** – The 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 [6.2](#), page [86](#))

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.

- **setNumCitters**

```
public void setNumCitters(int numCitters)
```

- **Description**

Sets the number of citters considered to estimate the class prediction of tests bags.

- **Parameters**

- * `numCitters` – The new number of citters.

- **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.

23.8.8 Members inherited from class MIMLClassifier

`miml.classifiers.miml.MIMLClassifier` (in [6.2](#), page [86](#))

- `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)

23.9 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.*

23.9.1 Declaration

```
public class MIMLMApKNN
    extends miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN
```

23.9.2 Field summary

serialVersionUID Generated Serial version UID.
smooth Smooth factor

23.9.3 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.

23.9.4 Method summary

configure(Configuration)

getSmooth() Gets the smooth factor considered by the classifier.

setSmooth(double) Sets the smooth factor considered by the classifier.

23.9.5 Fields

- private static final long **serialVersionUID**

– Generated Serial version UID.

- protected double **smooth**

– Smooth factor

23.9.6 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.

23.9.7 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

23.9.8 Members inherited from class MultiInstanceMultiLabelKNN

miml.classifiers.miml.lazy.MultiInstanceMultiLabelKNN (in 23.10, page 299)

- 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)

23.9.9 Members inherited from class MIMLClassifier

miml.classifiers.miml.MIMLClassifier (in 6.2, page 86)

- 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)

23.10 Class MultiInstanceMultiLabelKNN

Wrapper for class MultiLabelKNN of Mulan to work with MIML data

23.10.1 Declaration

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

23.10.2 All known subclasses

MIMLMAPkNN (in [23.9](#), page [295](#)), MIMLIBLR (in [23.7](#), page [282](#)), MIMLFuzzykNN (in [23.5](#), page [278](#)), MIMLDGC (in [23.3](#), page [270](#)), MIMLBRkNN (in [23.2](#), page [266](#)), DMIMLkNN (in [23.1](#), page [263](#))

23.10.3 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.

23.10.4 Constructor summary

MultiInstanceMultiLabelKNN() No-arg constructor for xml configuration
MultiInstanceMultiLabelKNN(MIMLDistanceFunction) Constructor to initialize the classifier.
MultiInstanceMultiLabelKNN(MIMLDistanceFunction, int) Constructor to initialize the classifier.

23.10.5 Method summary

buildInternal(MIMLInstances)
configure(Configuration)
getClassifier()
getMetric() Gets the distance metric considered by the classifier.
getNumOfNeighbours() Gets the number of neighbors considered by the classifier.
makePredictionInternal(MIMLBag)
setClassifier(MultiLabelKNN)
setMetric(DistanceFunction) Sets the distance metric considered by the classifier.
setnumOfNeighbours(int) Sets the number of neighbors considered by the classifier.

23.10.6 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.

23.10.7 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 `numOfNeighbours` 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.

23.10.8 Methods

- **buildInternal**

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

- **Description** copied from miml.classifiers.miml.MIMLClassifier (in [6.2](#), page [86](#))

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 neighbors considered by the classifier.

- **Returns** – the number of neighbors

- **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 [6.2](#), page [86](#))

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 neighbors considered by the classifier.

– **Parameters**

* **numOfNeighbours** – the new number of neighbors

23.10.9 Members inherited from class MIMLClassifier

miml.classifiers.miml.MIMLClassifier (in 6.2, page 86)

- **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 24

Package

miml.data.partitioning.powerset

<i>Package Contents</i>	<i>Page</i>
Classes	
LabelPowersetCrossValidation 304	
Class to split a multi-label dataset into N multi-label for cross-validation by applying a labelPowerset-based partition.	
LabelPowersetTrainTest 306	
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.	

24.1 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.

24.1.1 Declaration

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

24.1.2 Constructor summary

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

24.1.3 Method summary

getFolds(int)

24.1.4 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

24.1.5 Methods

- **getFolds**

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

- **Description** copied from `miml.data.partitioning.CrossValidationBase` (in [2.1](#), page [49](#))
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.

24.1.6 Members inherited from class `CrossValidationBase`

`miml.data.partitioning.CrossValidationBase` (in [2.1](#), page [49](#))

- `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`

24.1.7 Members inherited from class `PartitionerBase`

`miml.data.partitioning.PartitionerBase` (in [2.2](#), page [52](#))

- `protected seed`
- `protected workingSet`

24.2 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.

24.2.1 Declaration

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

24.2.2 Constructor summary

`LabelPowersetTrainTest(int, MultiLabelInstances)` Constructor.
`LabelPowersetTrainTest(MultiLabelInstances)` Default constructor.

24.2.3 Method summary

`split(double)`

24.2.4 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

24.2.5 Methods

- **split**

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

- **Description** copied from `miml.data.partitioning.TrainTestBase` (in 2.3, page 53)

Returns a array with two multi-label random datasets corresponding to the train and test sets respectively.

- **Parameters**

- * `percentageTrain` – Percentage of train dataset.

- **Returns** – `MultiLabelInstances[]`.
`MultiLabelInstances[0]` is the train set.
`MultiLabelInstances[1]` is the test set.

- **Throws**

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

24.2.6 Members inherited from class `TrainTestBase`

`miml.data.partitioning.TrainTestBase` (in 2.3, page 53)

- `public abstract MultiLabelInstances split(double percentageTrain)` throws `java.lang.Exception`

24.2.7 Members inherited from class `PartitionerBase`

`miml.data.partitioning.PartitionerBase` (in 2.2, page 52)

- protected `seed`
- protected `workingSet`

Index

addBag(MIMLBag), 220
addFeatures, 283
addInstance(MIMLBag, int), 220
algorithmName, 99
ArithmeticTransformation, 192
ArithmeticTransformation(), 193
ArithmeticTransformation(MIMLInstances), 193
attributesPerBag, 154, 231
AverageHausdorff, 256
AverageHausdorff(), 256
AverageHausdorff(MIMLInstances), 256
averageIR(double[]), 144, 159
averageSkew(HashMap), 145, 159
avgInstancesPerBag, 154

base, 158
baseClassifier, 57
baseLearner, 115
BaseMIMLReport, 179
BaseMIMLReport(), 179
BaseMIMLReport(List, String, boolean, boolean, boolean), 179
BRT, 132
BRTransformation, 131
BRTransformation(MIMLInstances), 132
build(MIMLInstances), 85, 88
build(MultiLabelInstances), 88
buildInternal(MIMLInstances), 58, 88, 91, 116, 240, 289, 301
buildInternal(MultiLabelInstances), 108, 243

C, 18
calculateBagReferences(int), 290
calculateCooccurrence(MultiLabelInstances), 159
calculateCooccurrence(MIMLInstances), 145
calculateDatasetDistances(), 290
calculatePhiChi2(MIMLInstances), 145
calculatePhiChi2(MultiLabelInstances), 159
calculateRecordLabel(Integer[]), 290
calculateReferenceMatrix(), 291
calculateStats(), 154, 160
calculatingTheDesiredSplits(int[], double[], int, int), 249
calculatingTheFrequencies(Instances, int, int[]), 250
cardinality(), 145, 160
chi2, 158
classes, 103
classifier, 91, 300
classifierName, 100
computeWeightDensity(Instances, Instance, int), 108
configFileName, 99
ConfigLoader, 96
ConfigLoader(String), 97
ConfigParameters, 98
ConfigParameters(), 100
configuration, 97
configure(Configuration), 20, 27, 36, 43, 58, 62, 68, 74, 96, 116, 123, 129, 180, 241, 265, 268, 272, 285, 291, 297, 301
cooccurrenceMatrix, 158
cooccurrenceToCSV(), 160
cooccurrenceToString(), 160
cooccurrenceToCSV(), 146
cooccurrenceToString(), 146
correlationsToCSV(double[][]), 146, 160
correlationsToString(double[][]), 146, 161
cost, 35
CrossValidationBase, 49
CrossValidationBase(int, MultiLabelInstances), 50
CrossValidationBase(MultiLabelInstances), 50
CrossValidationExperiment, 167
CrossValidationExperiment(), 168

- crossValidationToCSV(EvaluatorCV), 180
- crossValidationToString(EvaluatorCV), 180
- D, 25
- d_size, 288
- data, 122
- dataFileName, 99
- dataSet, 132, 144, 154, 258
- dataset, 199, 206, 278, 288
- debug(String), 88
- densities, 107
- density(), 147, 161
- dfun, 258
- dispose(), 20, 27, 36, 43, 62, 68, 74, 91
- distance(Instance, Instance), 274
- distance(Instance, Instance, double), 274
- distance(Instance, Instance, double, PerformanceStats), 274
- distance(Instance, Instance, PerformanceStats), 274
- distance(Instances, Instances), 255, 257, 260, 261
- distance(MIMLBag, MIMLBag), 255, 258
- distance_matrix, 288
- distributionBags, 154
- distributionBagsToCSV(), 147, 154
- distributionBagsToCSV(HashMap), 147, 161
- distributionBagsToString(), 147, 155
- distributionBagsToString(HashMap), 147, 161
- distributionForInstance(Instance), 189
- distributionLabelsPerExample, 158
- DMIMLkNN, 263
- DMIMLkNN(), 264
- DMIMLkNN(int, double, MIMLDistanceFunction), 264
- DMIMLkNN(int, MIMLDistanceFunction), 264
- DMIMLkNN(MIMLDistanceFunction), 264
- e, 279
- elnn, 107, 279
- enmimlnn, 61
- EnMIMLNNmetric, 60
- EnMIMLNNmetric(), 61
- EnMIMLNNmetric(double, double), 62
- EnMIMLNNmetric(double, double, int), 62
- ensemble, 115
- epsilon, 18
- evaluation, 127
- EvaluatorCV, 121
- EvaluatorCV(), 123
- EvaluatorCV(MIMLInstances, int), 123
- EvaluatorHoldout, 126
- EvaluatorHoldout(), 128
- EvaluatorHoldout(MIMLInstances, double), 128
- EvaluatorHoldout(MIMLInstances, double, int, int), 128
- EvaluatorHoldout(MIMLInstances, MIMLInstances), 128
- extension, 267
- extNeigh, 107
- featureIndices, 87
- filename, 183
- filterMeasures(List), 184
- findThePossibleSpit(double[][], int, int), 250
- foldsCreation(Instances, Random, double[], int, int[], int), 250
- foldsToRounds(MultiLabelInstances[]), 50
- GeneratePartitions, 168
- GeneratePartitions(), 169
- GeometricTransformation, 195
- GeometricTransformation(), 196
- GeometricTransformation(MIMLInstances), 196
- getAddFeatures(), 285
- getAlgorithmName(), 100
- getAttributeIndices(), 275
- getAvgTestTime(), 123
- getAvgTrainTime(), 123
- getBag(int), 220
- getBagAsArray(int), 231
- getBagAsArray(MIMLBag), 232
- getBagAsCell(int), 232
- getBagAsCell(MIMLBag), 232
- getBagAsInstances(), 212
- getBagAsInstances(int), 221
- getBagLabels(int), 291
- getBags(), 233
- getC(), 20
- getChi2(), 148, 162
- getCiters(int), 291

- getClasses(), 104
- getClassifier(), 301
- getClassifierName(), 100
- getConfigFileName(), 100
- getConfiguration(), 97
- getCost(), 36
- getD(), 28
- getData(), 124, 129
- getDataFileName(), 100
- getDataSet(), 148
- getDebug(), 89
- getEpsilon(), 20
- getEvaluation(), 121, 124, 129
- getExtension(), 268
- getFilename(), 185
- getFolds(int), 51, 81, 247, 305
- getH(), 36
- getInstance(int), 213
- getInstance(int, int), 221
- getInstances(), 275
- getInvertSelection(), 275
- getIsTransformation(), 101
- getIteration(), 20
- getK(), 20
- getLabels(), 233
- getLabels(int), 234
- getLabels(MIMLBag), 234
- getLambda(), 28, 69
- getLPT(), 135
- getMax(), 139
- getMaxiter(), 28
- getMeasures(), 185
- getMetric(), 275, 301
- getMin(), 139
- getMLDataSet(), 222
- getMu(), 44, 63, 75
- getnFeatures(), 140
- getNorm_up(), 28
- getNum_sub(), 28
- getNumAttributes(), 222
- getNumAttributesInABag(), 213, 222
- getNumAttributesWithRelational(), 214, 223
- getNumBags(), 223
- getNumCitters(), 292
- getNumClassifiers(), 116
- getNumFolds(), 124
- getNumInstances(), 214
- getNumInstances(int), 224
- getNumOfNeighbours(), 301
- getNumReferences(), 292
- getObjects(), 104
- getOptions(), 275
- getOpts_average_begin(), 28
- getOpts_average_size(), 29
- getOpts_beta(), 44
- getOpts_C(), 44
- getOpts_epsilon(), 44
- getOpts_iteration(), 44
- getOpts_m(), 45
- getOpts_norm(), 29
- getPara(), 37
- getPhi(), 148, 162
- getPhiHistogram(), 148, 162
- getRange(), 140
- getRatio(), 37, 45, 63, 69, 75
- getReferences(int), 292
- getRelationMethod(), 21
- getRevision(), 275
- getRounds(int), 51
- getSamplePercentage(), 116
- getSeed(), 37, 63, 69, 75, 124
- getSmooth(), 265, 297
- getStdTestTime(), 124
- getStdTrainTime(), 125
- getStep_size(), 29
- getTechnicalInformation(), 109
- getTestTime(), 125, 129
- getThreshold(), 116
- getTrainTime(), 125, 130
- getTransformationMethod(), 101
- getTrueLabels(Instance, int, int[]), 251
- getType(), 37
- getUnionNeighbours(int), 292
- getWeightsMatrix(), 293
- globalInfo(), 275
- h, 35
- hasInstances(), 258
- HausdorffDistance, 257
- HausdorffDistance(), 258
- HausdorffDistance(MIMLInstances), 258
- header, 184
- HoldoutExperiment, 170

- HoldoutExperiment(), 170
- holdoutToCSV(EvaluatorHoldout), 181
- holdoutToString(EvaluatorHoldout), 181
- IConfiguration*, 95
- IDistance*, 254
- IEvaluator*, 120
- IMIMLClassifier*, 84
- includeBagId, 207
- indexes, 80
- ini, 279
- innerClassIR(), 149, 162
- insertAttributesToBags(ArrayList), 224
- insertAttributeToBags(Attribute), 224
- InsertingAttributesToBags, 170
- InsertingAttributesToBags(), 171
- InsertingAttributeToBag, 171
- InsertingAttributeToBag(), 172
- interClassIR(), 149, 163
- IRReport*, 177
- isDebug, 87
- isExtNeigh(), 109
- isHeader(), 185
- isIncludeBagId(), 208
- isLabels(), 185
- isModelInitialized, 87
- isModelInitialized(), 89
- isNormalized(), 140
- isSampleWithReplacement(), 117
- isStd(), 186
- isTransformation, 100
- isUpdatable(), 89
- isUseConfidences(), 117
- iteration, 18
- IterativeCrossValidation, 245
- IterativeCrossValidation(int, MultiLabelInstances), 246
- IterativeCrossValidation(MultiLabelInstances), 246
- IterativeTrainTest, 247
- IterativeTrainTest(int, MultiLabelInstances), 248
- IterativeTrainTest(MultiLabelInstances), 249
- K, 19
- k, 278
- kini, 278
- KiSar, 17
- kisar, 18
- KiSar(), 19
- KiSar(double, double, double, double, double), 19
- kNearestNeighboursIndices(Instance, int), 111, 281
- labelCombCount(), 149, 163
- labelCombinations, 158
- labelDistance(Instance, Instance), 109
- labelIndices, 87, 231
- labelNames, 87
- LabelPowersetCrossValidation, 304
- LabelPowersetCrossValidation(int, MultiLabelInstances), 305
- LabelPowersetCrossValidation(MultiLabelInstances), 305
- LabelPowersetTrainTest, 306
- LabelPowersetTrainTest(int, MultiLabelInstances), 307
- LabelPowersetTrainTest(MultiLabelInstances), 307
- labels, 183
- labelSetFrequency(LabelSet), 149, 163
- labelSets(), 149, 163
- labelSkew(), 150, 163
- lambda, 26, 67
- linearClassifier(double[], double[]), 293
- LinearNNESearch(Instances), 111, 281
- listOptions(), 275
- loadClassifier(), 97
- loadEvaluator(), 98
- loadReport(), 98
- LPT, 135
- LPTransformation, 134
- LPTransformation(), 135
- m, 278
- main(String[]), 168–176, 236
- makeCopy(), 85, 89
- makePrediction(Instance), 85, 89
- makePredictionInternal(Instance), 109
- makePredictionInternal(MIMLBag), 58, 89, 92, 117, 241, 293, 302
- ManagingMIMLInstances, 172
- ManagingMIMLInstances(), 173

- Max, 139
- maxCount, 158
- MaximalHausdorff, 259
- MaximalHausdorff(), 259
- MaximalHausdorff(MIMLInstances), 259
- maxInstancesPerBag, 154
- maxiter, 25
- meanArray(long[]), 125
- measures, 183
- metric, 274, 288, 300
- milstatistics, 144
- MIMLBag, 211
- MIMLBag(Instance), 212
- MIMLBagging, 113
- MIMLBagging(), 115
- MIMLBagging(IMIMLClassifier, int), 115
- MIMLBagging(IMIMLClassifier, int, double), 115
- MIMLBinaryRelevance, 237
- MIMLBinaryRelevance(Classifier), 238
- MIMLBRkNN, 266
- MIMLBRkNN(), 267
- MIMLBRkNN(MIMLDistanceFunction), 267
- MIMLBRkNN(MIMLDistanceFunction, int), 268
- MIMLBRkNN(MIMLDistanceFunction, int, BrkNN.ExtensionType), 268
- MIMLClassifier, 86
- MIMLClassifier(), 87
- MIMLClassifierToMI, 239
- MIMLClassifierToMI(), 240
- MIMLClassifierToMI(MultiLabelLearner), 240
- MIMLClassifierToML, 56
- MIMLClassifierToML(), 57
- MIMLClassifierToML(MultiLabelLearner, MIMLtoML), 57
- mimlDataSet, 230
- mimlDataset, 57
- MIMLDGC, 270
- MIMLDGC(), 271
- MIMLDGC(MIMLDistanceFunction), 271
- MIMLDGC(MIMLDistanceFunction, int), 271
- MIMLDistanceFunction, 273
- MIMLDistanceFunction(IDistance), 274
- MIMLFast, 24
- mimlfast, 25
- MIMLFast(), 26
- MIMLFast(int, int, int, double, double, int, int, int, int), 26
- MIMLFast(int, int, int, double, int), 27
- MIMLFuzzykNN, 278
- MIMLFuzzykNN(), 279
- MIMLFuzzykNN.LinearNNEsearch, 280
- MIMLIBLR, 282
- MIMLIBLR(), 283
- MIMLIBLR(int, boolean, MIMLDistanceFunction), 283
- MIMLIBLR(int, MIMLDistanceFunction), 284
- MIMLIBLR(MIMLDistanceFunction), 284
- MIMLInstances, 217
- MIMLInstances(Instances, LabelsMetaData), 218
- MIMLInstances(Instances, String), 218
- MIMLInstances(MIMLInstances), 219
- MIMLInstances(String, int), 219
- MIMLInstances(String, String), 219
- MIMLkNN, 286
- MIMLkNN(), 289
- MIMLkNN(IDistance), 289
- MIMLkNN(int, int, IDistance), 289
- MIMLLabelPowerset, 242
- MIMLLabelPowerset(Classifier), 243
- MIMLLabelPowersetTransformation, 136
- MIMLLabelPowersetTransformation(), 136
- MIMLMApkNN, 295
- MIMLMApkNN(), 296
- MIMLMApkNN(int, double, MIMLDistanceFunction), 296
- MIMLMApkNN(int, MIMLDistanceFunction), 296
- MIMLMApkNN(MIMLDistanceFunction), 297
- MIMLNN, 66
- mimlnn, 67
- MIMLNN(), 67
- MIMLNN(double, double), 68
- MIMLNN(double, double, int), 68
- MIMLRBF, 72
- mimlrbf, 73
- MIMLRBF(), 73
- MIMLRBF(double, double), 74
- MIMLRBF(double, double, int), 74

- MIMLReport, 182
- MIMLReport(), 184
- MIMLReport(List, String, boolean, boolean, boolean), 184
- MIMLStatistics, 142
- MIMLStatistics(MIMLInstances), 144
- MIMLSVM, 33
- mimlsvm, 34
- MIMLSVM(), 35
- MIMLSVM(String, String, double, double, double, double, double), 35
- MIMLtoMITransformation, 173
- MIMLtoMITransformation(), 173
- MIMLtoML, 198
- MIMLtoML(), 199
- MIMLtoMLTransformation, 174
- MIMLtoMLTransformation(), 174
- MIMLWel, 41
- mimlwel, 42
- MIMLWel(), 43
- MIMLWel(double, double, double, double, double, double, double), 43
- Min, 139
- MinimalHausdorff, 260
- MinimalHausdorff(), 261
- MinimalHausdorff(MIMLInstances), 261
- minimax(Instances, int), 199
- minInstancesPerBag, 153
- MinMaxNormalization, 138
- MinMaxNormalization(), 139
- MinMaxTransformation, 202
- MinMaxTransformation(), 203
- MinMaxTransformation(MIMLInstances), 203
- MISMOWrapper, 188
- MISMOWrapper(), 189
- MISStatistics, 153
- MISStatistics(Instances), 154
- mlDataSet, 158
- mlDataSetWithBagId, 57
- MLDGC, 106
- MLDGC(), 108
- MLDGC(int), 108
- MLDGC(int, DistanceFunction), 108
- MLDGC.LinearNNESearch, 111
- MLSave, 226
- MLSave(), 227
- MLStatistics, 155
- mlstatistics, 144
- MLStatistics(MultiLabelInstances), 158
- mu, 42, 61, 73
- MultiInstanceMultiLabelKNN, 299
- MultiInstanceMultiLabelKNN(), 300
- MultiInstanceMultiLabelKNN(MIMLDistanceFunction), 300
- MultiInstanceMultiLabelKNN(MIMLDistanceFunction, int), 300
- multipleEvaluation, 122
- MWClassifier, 90
- MWClassifier(), 91
- MWTranslator, 230
- MWTranslator(MIMLInstances), 231
- nBags, 231
- nFeatures, 139
- NGC, 107
- nLabels, 231
- norm_up, 25
- normalize(MIMLInstances), 140
- normalized, 139
- NormalizingDataset, 175
- NormalizingDataset(), 175
- num_citers, 288
- num_references, 288
- num_sub, 26
- numAttributes, 157
- numBags, 154
- numClassifiers, 115
- numExamples, 157
- numFolds, 122
- numLabels, 87, 157
- numNominal, 157
- numNumeric, 157
- numOfNeighbours, 300
- nUnique, 158
- objects, 103
- opts_average_begin, 26
- opts_average_size, 26
- opts_beta, 42
- opts_C, 42
- opts_epsilon, 42
- opts_iteration, 42
- opts_m, 42

- opts_norm, 26
- para, 35
- Params, 103
- Params(Class[], Object[]), 103
- PartitionerBase, 52
- PartitionerBase(int, MultiLabelInstances), 52
- PartitionerBase(MultiLabelInstances), 53
- peak, 158
- phi, 158
- phi_matrix, 288
- pMax(), 150, 164
- positiveExamplesPerLabel, 157
- postProcessDistances(double[]), 275
- predictMWClassifier(MWCellArray, MWNumericArray, MWNumericArray), 21, 29, 37, 45, 63, 69, 75, 92
- prepareTemplate(), 200, 203, 208
- printPhiDiagram(double), 150, 164
- priors(), 150, 164
- PropositionalTransformation, 206
- PropositionalTransformation(MIMLInstances), 207
- PropositionalTransformation(MIMLInstances, boolean), 207
- pUnique(), 151, 164
- RandomCrossValidation, 79
- RandomCrossValidation(int, MultiLabelInstances), 80
- RandomCrossValidation(MultiLabelInstances), 80
- RandomTrainTest, 81
- RandomTrainTest(int, MultiLabelInstances), 82
- RandomTrainTest(MultiLabelInstances), 82
- Range, 139
- ratio, 35, 42, 61, 67, 73
- readMultiLabelLearnerParams(Configuration), 105
- ref_matrix, 288
- relationMethod, 19
- removeBagId(MultiLabelInstances), 209
- removeFilter, 57, 207
- resample(Instances, double, boolean, int), 105
- Resampling, 175
- Resampling(), 176
- returnPossibleSplitsForNotAnnotated(double[][]), 251
- RunAlgorithm, 235
- RunAlgorithm(), 235
- runExperiment(IMIMLClassifier), 121, 125, 130
- samplePercentage, 114
- sampleWithReplacement, 114
- saveArff(Instances, String), 227
- saveArff(MIMLInstances, String), 228
- saveArff(MultiLabelInstances, String), 228
- saveReport(String), 178, 186
- saveXml(ArrayList, String), 228
- saveXml(Instances, String), 229
- saveXml(MultiLabelInstances, String), 229
- seed, 35, 52, 61, 67, 73, 114, 122
- serialVersionUID, 18, 25, 34, 42, 57, 61, 67, 73, 87, 91, 107, 111, 114, 132, 135, 136, 188, 193, 196, 199, 203, 212, 218, 238, 240, 243, 256, 258–260, 263, 267, 271, 274, 278, 280, 283, 288, 296, 300
- setAddFeatures(boolean), 285
- setAlgorithmName(String), 101
- setAttributeIndices(String), 275
- setC(double), 22
- setClasses(Class[]), 104
- setClassifier(MultiLabelKNN), 302
- setClassifierName(String), 101
- setConfigFileName(String), 102
- setConfiguration(Configuration), 98
- setCost(double), 38
- setD(int), 30
- setDataFileName(String), 102
- setDataSet(MIMLInstances), 151
- setDebug(boolean), 85, 90
- setEpsilon(double), 22
- setExtension(BRkNN.ExtensionType), 269
- setExtNeigh(boolean), 109
- setFilename(String), 186
- setH(double), 38
- setHeader(boolean), 186
- setIncludeBagId(boolean), 209
- setInstances(Instances), 276
- setInstances(MIMLInstances), 258
- setInvertSelection(boolean), 276
- setIsTransformation(Boolean), 102

- setIteration(double), 22
- setK(double), 22
- setLabels(boolean), 187
- setLambda(double), 30, 70
- setMaxiter(int), 30
- setMeasures(List), 187
- setMetric(DistanceFunction), 302
- setMetric(IDistance), 276
- setMu(double), 46, 64, 76
- setNorm_up(int), 31
- setNum_sub(int), 31
- setNumCitters(int), 294
- setNumFolds(int), 126
- setnumOfNeighbours(int), 303
- setNumReferences(int), 294
- setObjects(Object[]), 104
- setOptions(String[]), 276
- setOpts_average.begin(int), 31
- setOpts_average.size(int), 31
- setOpts_beta(double), 46
- setOpts_C(int), 46
- setOpts_epsilon(double), 46
- setOpts_iteration(int), 47
- setOpts_m(double), 47
- setOpts_norm(int), 31
- setPara(String), 38
- setRatio(double), 39, 47, 64, 70, 76
- setRelationMethod(double), 22
- setSamplePercentage(double), 117
- setSampleWithReplacement(boolean), 118
- setSeed(double), 39
- setSeed(int), 64, 70, 76, 118, 126
- setSmooth(double), 265, 297
- setStd(boolean), 187
- setStep_size(double), 32
- setThreshold(double), 118
- setTransformationMethod(String), 102
- setType(String), 39
- setUseConfidences(boolean), 118
- setValue(int, int, double), 215
- showUse(), 168–175
- skewRatio(), 151, 165
- smooth, 263, 296
- split(double), 54, 82, 251, 307
- splitData(MIMLInstances, double, int, int), 225
- std, 183
- stdArray(long[]), 126
- step_size, 26
- t_matrix, 288
- takeTheInstancesOfTheLabel(Instances, int, int[], int[]), 252
- takingTheSmallestIndexAndNumberInVector(int[], int), 252
- template, 199, 206
- testData, 127
- testTime, 122, 127
- threshold, 114
- toCSV(), 151, 155, 165
- toCSV(IEvaluator), 178, 182
- topPhiCorrelatedLabels(int, int), 152, 165
- toString(), 152, 155, 165
- toString(IEvaluator), 178, 182
- totalInstances, 154
- trainData, 127
- trainMWClassifier(MWCellArray, MWNumericArray), 23, 32, 39, 47, 65, 71, 77, 93
- TrainTestBase, 53
- TrainTestBase(int, MultiLabelInstances), 54
- TrainTestBase(MultiLabelInstances), 54
- trainTime, 122, 127
- transformationClassifier, 240
- transformationMethod, 57, 100
- transformBag(int, int), 132
- transformBag(MIMLBag, int), 133
- transformBag(MIMLBag, int[]), 135
- transformBag(MIMLBag, int[], int), 133
- transformBags(int), 133
- transformBags(MIMLInstances), 135
- transformBags(MIMLInstances, int[], int), 134
- transformDataset(), 194, 196, 200, 204, 209
- transformDataset(MIMLInstances), 194, 197, 201, 204, 209
- transformInstance(Instance, int[]), 136
- transformInstance(MIMLBag), 194, 197, 201, 205, 209
- transformInstance(MIMLInstances, MIMLBag), 195, 197, 205, 210
- type, 35
- U, 278

uncorrelatedLabels(int, double), [152](#), [166](#)
update(Instance), [276](#)
update(MIMLBag), [259](#)
updateDesiredSplitStatistics(double[],
 boolean[]), [253](#)
updateDistance(double, double), [276](#)
updatedLabelIndices, [199](#), [206](#)
updateStats(MIMLInstances), [141](#)
useConfidences, [114](#)
Utils, [104](#)
Utils(), [105](#)

varianceIR(double[]), [152](#), [166](#)

weight_max, [107](#)
weight_min, [107](#)
weights, [107](#)
weights_matrix, [288](#)
workingSet, [52](#)
wrapper, [91](#)