Multi-Label Dataset Analyzer
User Guide

Author:
Jose María Moyano Murillo

Directors:
Dra. Eva Lucrecia Gibaja Galindo
Dr. Sebastián Ventura Soto

Contents

Chapter 1

Multi-label dataset analyzer tool

1.1 Downloading and executing

Before downloading the tool, it is necessary to have Java version 1.8 or higher installed. In order to download and install Java, you have to access to `https://www.java.com/es/download/`, download the version which corresponds to your operative system and install it.

The tool is available for download at `http://www.uco.es/grupos/kdis/kdiswiki/MLLResources/Software/MLDA_v1_1.zip`. Once downloaded, the file have to be decompressed, and it contains the executable *.jar* file. To execute the tool, you have to open a command-line interface, access to the path where the *.jar* file is located and type the command `java -jar MLDA.jar`.

1.2 Datasets format

1.2.1 Multi-label datasets

The tool is able to read two multi-label dataset formats, which are Mulan [1] and Meka [2] formats. Both of them are based on the *.arff* format from Weka [3]. Weka's *.arff* format has the following characteristics:

- The relation name goes in the first line, following the statement "*@relation*". If the relation name has spaces, it must be between quotes.
- Each attribute must be in a different line, and its syntax is: "*@attribute name type*" .The attribute name must be between quotes if it has spaces. The attribute type could be:
  - numeric: integer and real are treated as numeric.
  - <nominal-values>: between braces and separated by commas all the possible values.
  - string.
  - date[<format>].

[1] G. Tsoumakas et al. "Mulan: A Java Library for Multi-Label Learning". In: Journal of Machine Learning Research 12 (2011), pp. 2411–2414.

[2] MEKA: A Multi-label Extension to WEKA. `http://meka.sourceforge.net/`. Last accessed: 21-04-2016.

[3] Mark Hall et al. "The WEKA Data Mining Software: An Update". In: SIGKDD Explor. Newsl. 11.1 (2009), pp. 10–18.

- Instances start with the statement "*@data*", and each one will be in a different line. Each attribute value is separated by commas, and they will be in the same order they were declared. Also there are a shortly way to write the instances, where attributes with zero value are not included. In this case, instances will be in braces, and separated by commas each pair composed by attribute and value.
- Line comments are inserted with the character "%".

Mulan and Meka uses this format but they are different in how they define the labels. Mulan uses two different files: an *.arff* file and a *.xml* file. In the *.arff* file are exposed the full set of attributes and labels, without distinguishing among them. The *.xml* file indicates which are the attributes that act like labels in the dataset. This *.xml* file is necessary because nowhere in the *.arff* file is indicated which the labels are. Figures 1.1 and 1.2 shows the *.arff* and *.xml* files respectively from a dataset in Mulan format.

```
@relation emotions_test

@attribute att1 numeric
@attribute att2 numeric
@attribute att3 numeric
   ...
@attribute att72 numeric
@attribute amazed-suprised {0,1}
@attribute happy-pleased {0,1}
@attribute relaxing-calm {0,1}
@attribute quiet-still {0,1}
@attribute sad-lonely {0,1}
@attribute angry-aggresive {0,1}

%Starting with the data
@data
0.094829,0.204498,0.082824, ..., 0.335371,1,0,0,0,1,1
0.065248,0.117975,0.08597, ..., 0.442898,0,0,0,1,0,0
0.101287,0.23254,0.078028, ..., 1.183461,1,1,0,0,0,0
   ...
0.172427,0.378696,0.081777, ..., 1.294949,1,1,1,0,0,0
```

FIGURE 1.1: Mulan *.arff* file

```
<?xml version="1.0" encoding="utf-8"?>
  <labels xmlns="http://mulan.sourceforge.net/labels">
   <label name="amazed-suprised"></label>
   <label name="happy-pleased"></label>
   <label name="relaxing-calm"></label>
   <label name="quiet-still"></label>
   <label name="sad-lonely"></label>
   <label name="angry-aggresive"></label>
</labels>
```

FIGURE 1.2: Mulan *.xml* file

On the other hand, Meka format only needs the *.arff* file. In this case, the separation between features and labels is done in the *.arff* file. Meka's *.arff* format is similar to Mulan's *.arff*, excepting the relation name line, where Meka indicates which attributes are the labels. Following the relation name, and separated by a colon, it is indicated with parameter "-C" and a integer

positive number if the first q attributes are labels, or with an integer negative number if the labels are the last q attributes. Figure 1.3 shows the *.arff* file from a Meka dataset.

```
@relation "emotions: -C -6"

@attribute att1 numeric
@attribute att2 numeric
@attribute att3 numeric
   ...
@attribute att72 numeric
@attribute amazed-suprised {0,1}
@attribute happy-pleased {0,1}
@attribute relaxing-calm {0,1}
@attribute quiet-still {0,1}
@attribute sad-lonely {0,1}
@attribute angry-aggresive {0,1}

%Starting with the data
@data
0.094829,0.204498,0.082824, ..., 0.335371,1,0,0,0,1,1
0.065248,0.117975,0.08597, ..., 0.442898,0,0,0,1,0,0
0.101287,0.23254,0.078028, ..., 1.183461,1,1,0,0,0,0
   ...
0.172427,0.378696,0.081777, ..., 1.294949,1,1,1,0,0,0
```

FIGURE 1.3: Meka *.arff* file

As seen, both formats are similar, and while Meka's format is simpler, Mulan's format is more powerful because it does not need to define all the labels at the beginning or the end of the attributes set. Because of this diversity in multi-label datasets format, the tool is able to read both.

1.2.2 Multi-view multi-label datasets

The format to define the multiple views consists only in adding this information to the header of the *.arff* file. To define the views, inside the relation name the parameter "*-V*" is included. It is followed by a colon "*:*" and the set of views. Each view is defined with an interval with lower and higher attribute indices of the view, both included, and separated by a hyphen "-". Different views are separated by an exclamation mark "*!*". Figure 1.4 shows an example of a multi-view multi-label dataset, where two views are defined, one from attribute 0 to 63 and other from attribute 64 to 71. This declaration of views is applicable to both Mulan and Meka formats.

1.3 Summary

Once executed, the first tab is the summary, where main metrics for dataset characterization can be calculated. For loading a dataset, you have to click on the button in the upper right corner (Figure 1.5), and select an *.arff* file in Mulan or Meka formats. As it can be seen, this tab shows in its top a set of basic metrics for characterizing datasets, while in the bottom it shows a larger set of metrics, from which you can select a subset of them, and calculate with the *Calculate* button (Figure 1.6).

In order to select the metrics, there also exist four extra buttons: *All*, to select all the metrics, *None*, to unselect all of them, *Invert*, to invert the

```
@relation "emotions -V:0-63!64-71"

@attribute att1 numeric
@attribute att2 numeric
@attribute att3 numeric
   ...
@attribute att72 numeric
@attribute amazed-suprised {0,1}
@attribute happy-pleased {0,1}
@attribute relaxing-calm {0,1}
@attribute quiet-still {0,1}
@attribute sad-lonely {0,1}
@attribute angry-aggresive {0,1}

%Starting with the data
@data
0.094829,0.204498,0.082824, ..., 0.335371,1,0,0,0,1,1
0.065248,0.117975,0.08597, ..., 0.442898,0,0,0,1,0,0
0.101287,0.23254,0.078028, ..., 1.183461,1,1,0,0,0,0
   ...
0.172427,0.378696,0.081777, ..., 1.294949,1,1,1,0,0,0
```

FIGURE 1.4: MVML dataset *.arff* file

selection, and *Clear*, which unselect and clear all the calculated values. The *Save* button allows to save all the metric values in four formats: *.txt*, *.csv*, *.arff* y *.tex*.

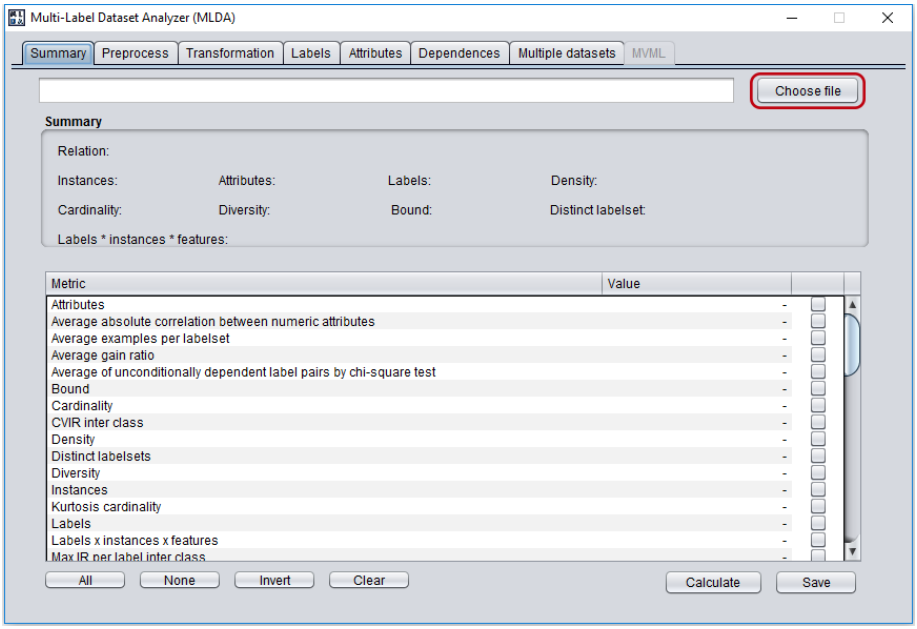

FIGURE 1.5: Select a dataset in summary tab

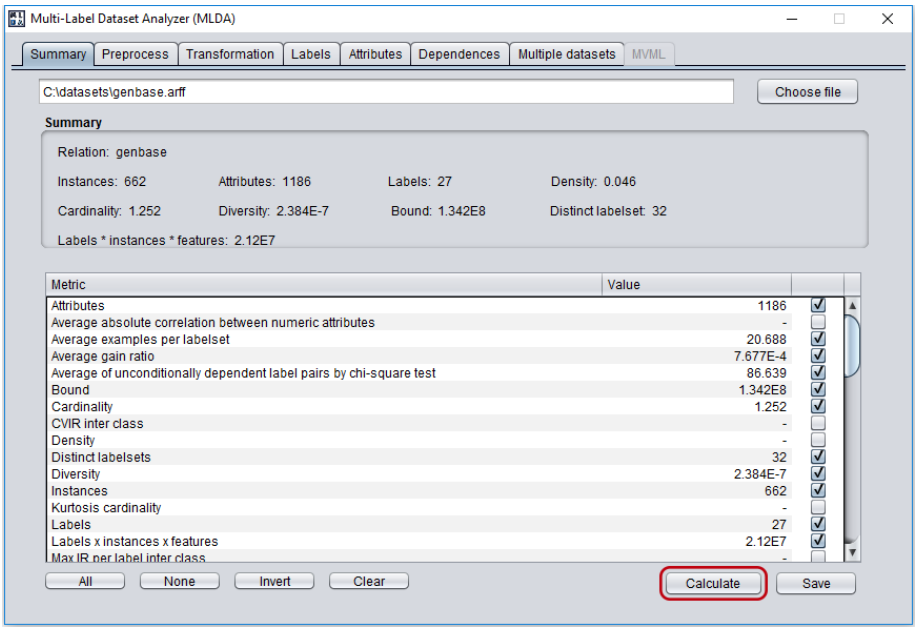

FIGURE 1.6: Calculate metrics in summary tab

1.4 Preprocess

The preprocess tab allows to perform preprocess tasks over the multi-label dataset. It includes instance selection, feature selection, data partitioning and format conversion. All the preprocess types could be done separately or together, where the process follows the mentioned order. That is to say, in case of selecting the three preprocessing types, first it will be made the instance selection, then, over the modified dataset, it will be made the feature selection, and last the splitting. Finally, the dataset could be saved in Mulan or Meka format.

1.4.1 Instance selection

To perform the random instance selection, only the number of instances to select have to be indicated, which must be less than the number of instances in the original dataset (Figure 1.7).

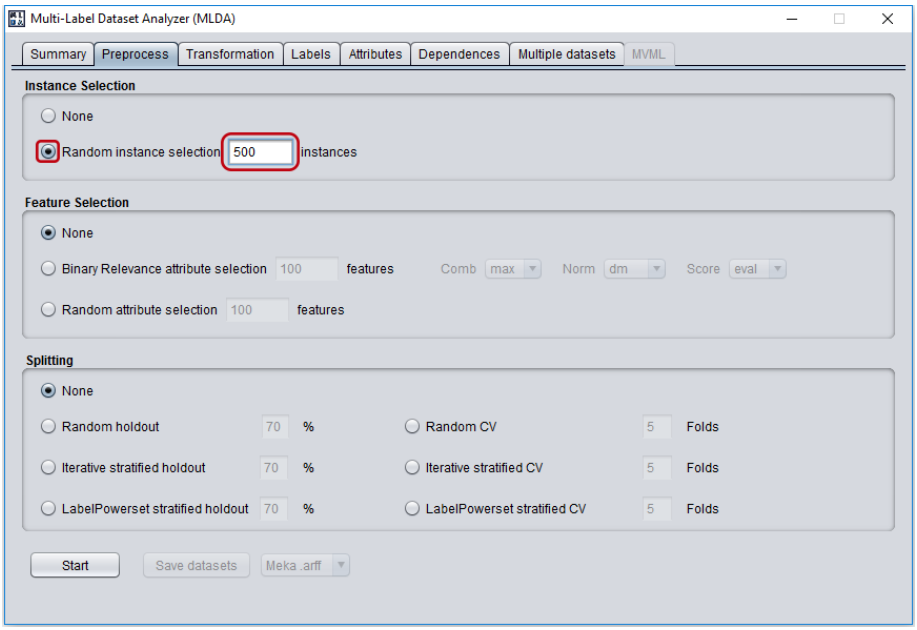

FIGURE 1.7: Random instance selection

1.4.2 Feature selection

For feature selection, the tool includes two methods: BR based and random. For both types, the number of features must be indicated, always less than the number of features in the original dataset. The BR based method, as shown in Figure 1.8, needs three parameters: the combination approach method (which may be by maximum *max*, average *avg* or minimum *min*), normalization mode (which may be dividing by length *dl*, dividing by maximum *dm* or no normalization *none*), and score mode (which may be by evaluation score *eval* or by ranking score *rank*).

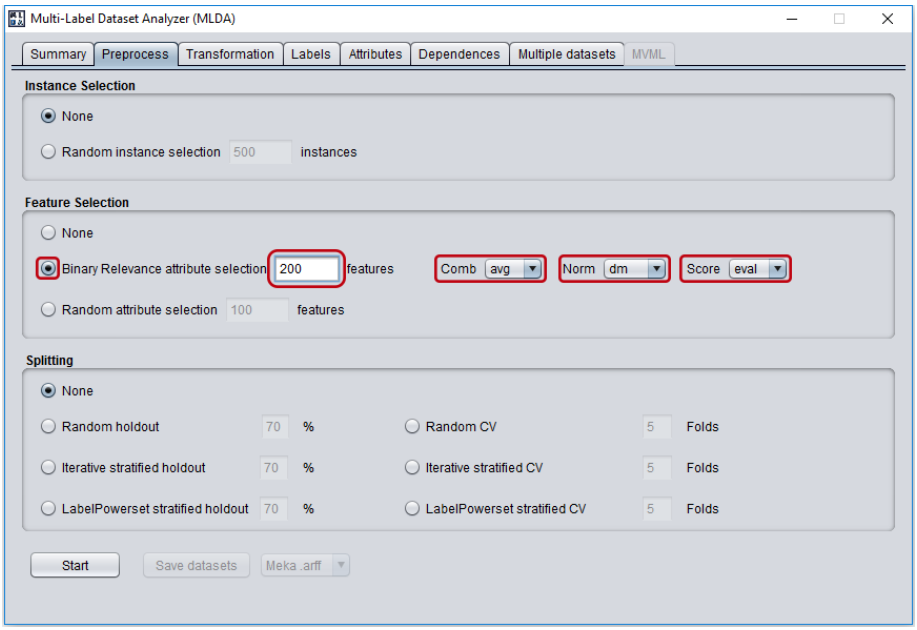

FIGURE 1.8: BR based feature selection

Data partitioning The dataset splitting could be made of two different types: holdout and k-folds cross-validation (cv) (Figure 1.9). For holdout partitioning, the percentage of instances for the train file must be specified, while for k-folds cv, it has to be indicated the number of folds to split the original dataset. Holdout partitioning saves two new dataset files, a train file and a test file, while the k-folds cv saves k train files and k test files (Figure 1.10). For both types, there exists three methods for splitting: random, iterative stratified and LP stratified.

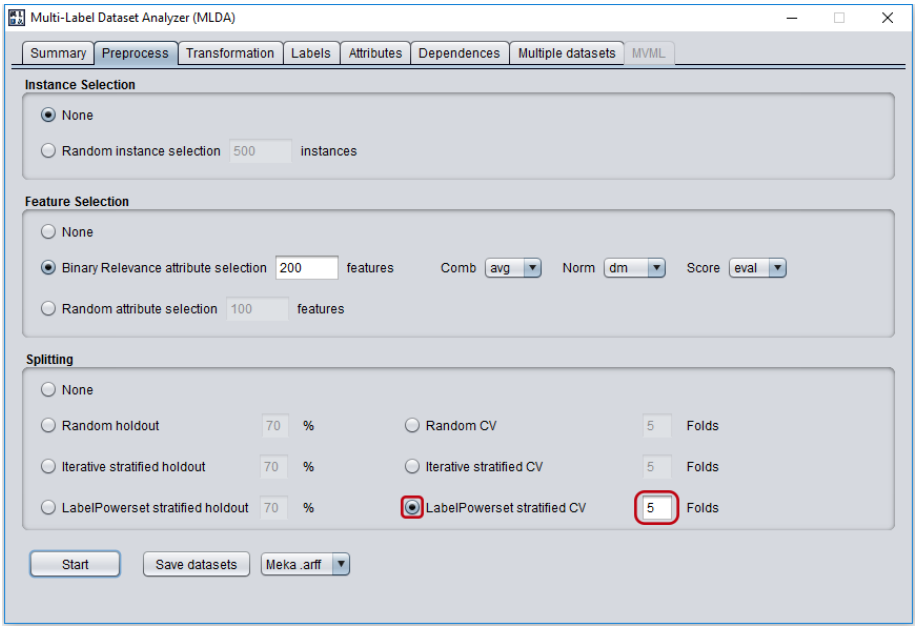

FIGURE 1.9: LP k-folds cv dataset splitting

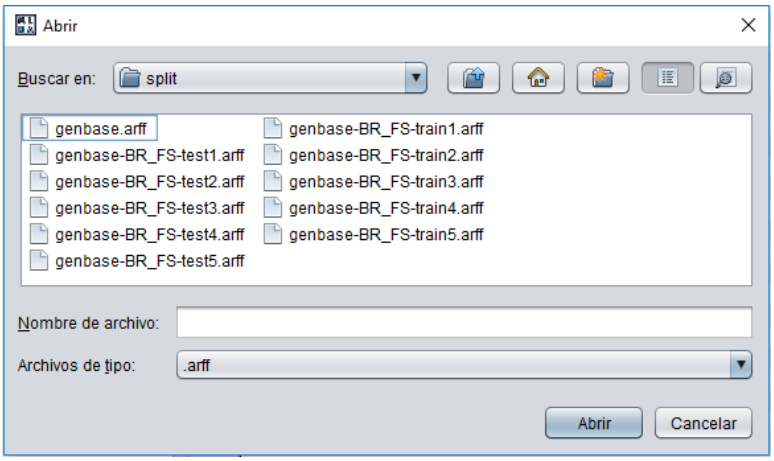

FIGURE 1.10: Result of a 5-folds cv splitting

1.4.3 Dataset format conversion

Since the tool is able to load both Mulan and Meka dataset formats and also allows to save the preprocessed datasets in both formats, if no preprocessing type is selected (Figure 1.11), the dataset format could be converted without modifying data. In all cases, before saving the preprocessed data with *Save datasets* button, it is needed to press the *Start* button in order to execute preprocessing.

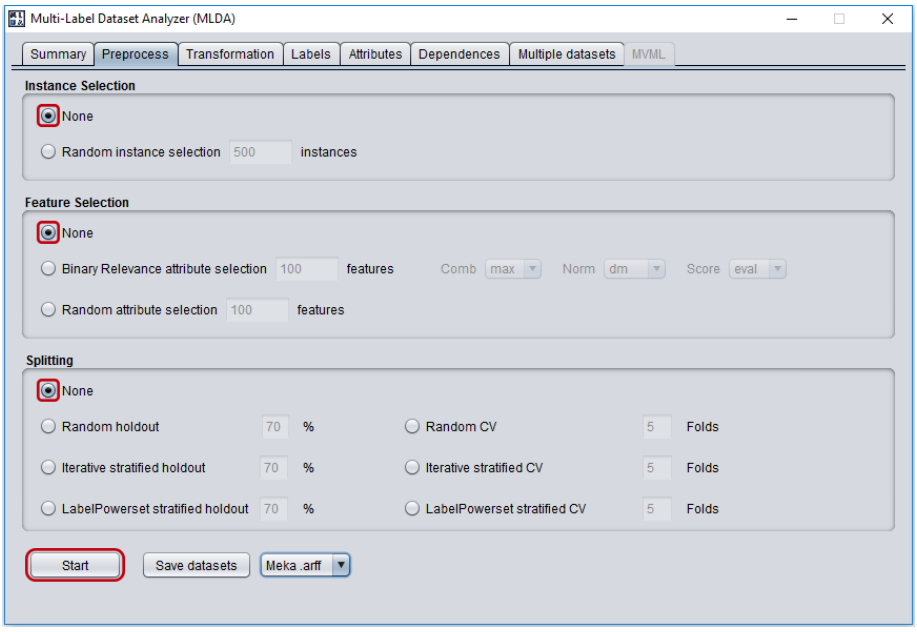

FIGURE 1.11: Dataset format conversion

1.5 Transformation

Transformation tab includes four transformation methods for multi-label datasets into single-label datasets. It includes: Binary Relevance transformation, Label Powerset transformation, include labels transformation

and remove all labels transformation. Binary Relevance method generates q binary datasets, one for each existing label. Label Powerset generates one multi-class dataset, where the classes are the labelsets of the original dataset. Include labels transformation generates a binary dataset. For that, each instance is replicated q times, where each new instance is extended with the label name and the class is the label value of this instance. Finally, remove all labels transformation generates a new dataset removing all the label attributes.

To transform the current dataset, we only have to select the desired method, click on the *Transform* button, and then save it with the *Save* button, as shown in Figure 1.12.

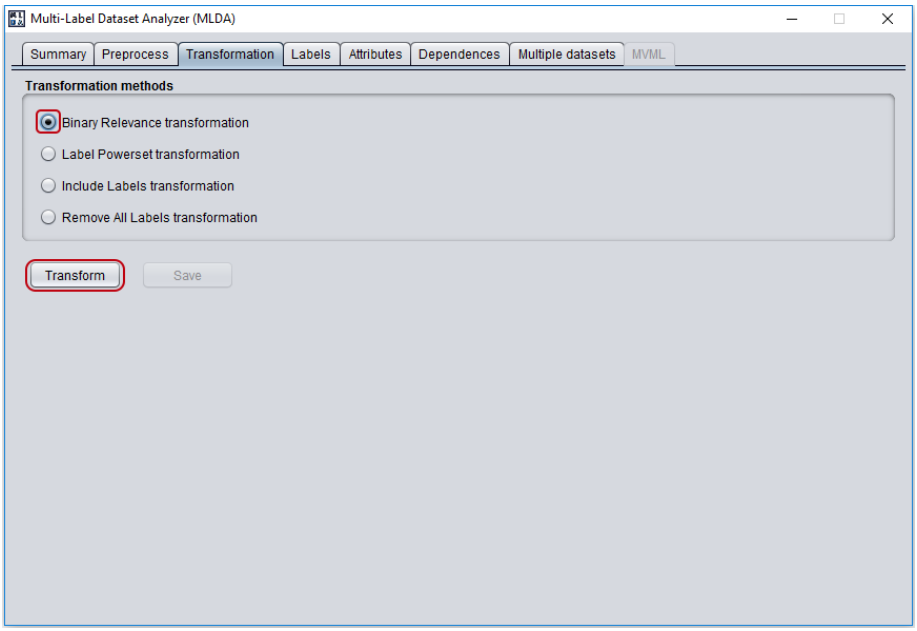

FIGURE 1.12: Transformation tab

1.6 Labels

Labels tab allows to obtain information and graphics about labels distribution and labels imbalance. On the top appears a dropdown menu (Figure 1.13) where it can be selected the graphic type. In all cases, on the left side appears a table with the data, which also can be saved in *.csv* format by clicking on the *Save* button, and on the right side appears the graphic. By clicking in the chart with the right mouse button (Figure 1.14), it includes some options as zoom in or zoom out, auto range, print, properties, or saving the chart as a *.png* file in the *Save as* option (Figure 1.15) .

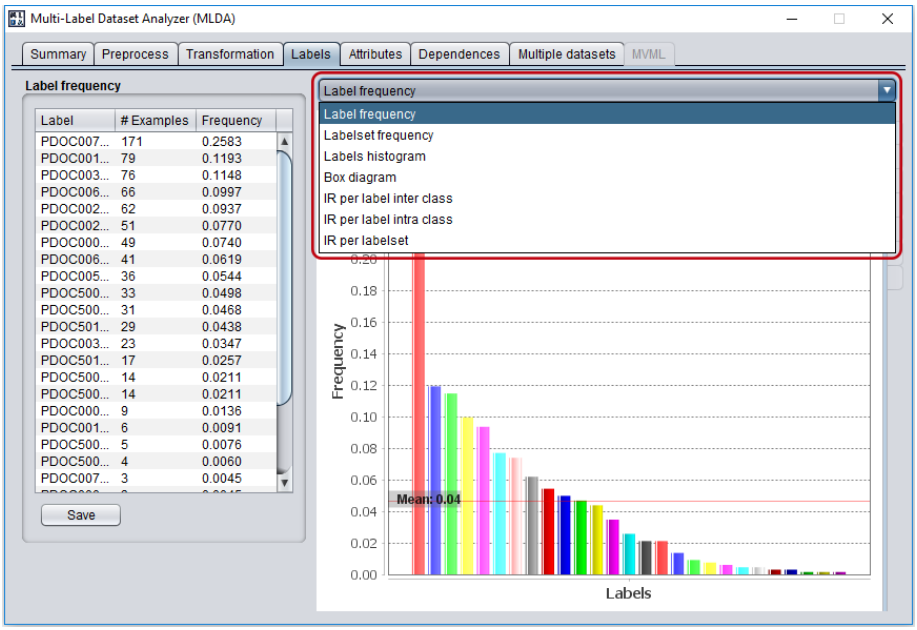

FIGURE 1.13: Dropdown menu on labels tab

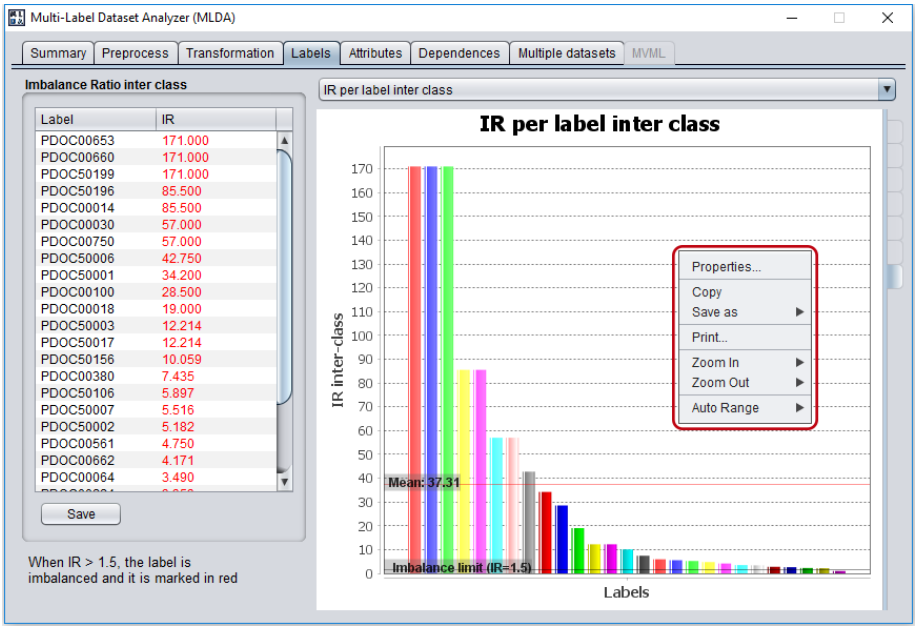

FIGURE 1.14: Chart options

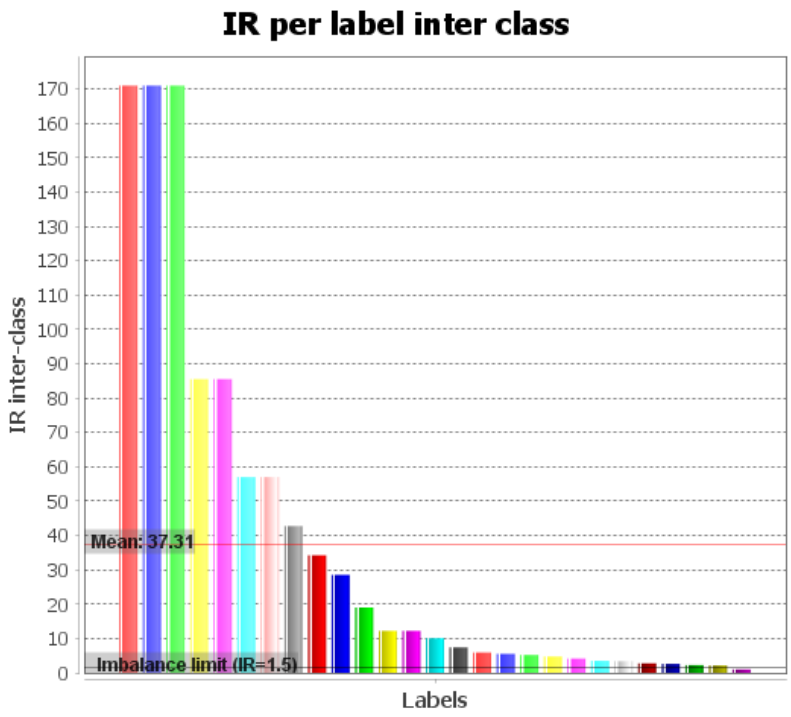

FIGURE 1.15: Example of saved chart

In the dropdown menu, there are the following options:

- **Label frequency:** on the left side it shows a table with the labels, the number of examples to which it is associated and the frequency of the label. On the right side it shows a bar chart where each bar corresponds to a label and its height indicates the frequency. Besides, it shows the average frequency value as a horizontal line. If the mouse is over a bar, it shows a tooltip message with the label name and frequency (Figure 1.16).

- **Labelset frequency:** similar to the previous case, this tab shows the frequency of each labelset. So, the left table shows, ordered by frequency, the number of instances associated to each labelset and its frequency. When clicking on each labelset in the table, it appears information about the labelset, showing the number of labels that compose it, the binary coded labelset (1s and 0s shows if each label is associated or not with the labelset respectively), the labelset frequency and the name and frequency of each label included in the labelset (Figure 1.17). In the bar chart of the right side, each bar corresponds to a labelset and the height shows its frequency.

- **Labels histogram:** in this case, the table shows how many examples are associated with a certain number of labels and its frequency. That is to say, it shows how many examples have only one label associated, how many have two labels, how many have three, and so on. The bar chart indicates the frequency of the previous table. If the mouse is over a bar, it shows a tooltip message with the number of associated labels and their frequency. (Figure 1.18).

- **Box diagram:** it shows two possible boxplot charts, showing the distribution of the number of examples by label (Figure 1.19) or of number of examples by labelset (Figure 1.20).

- **IR inter class:** imbalance ratio or IR indicates how the labels are imbalanced in the dataset. In this case, the IR inter class shows the imbalance ratio between labels. The table on the left side shows each label name and its IR inter class value, which are sorted by descending IR value. The bar chart shows the IR inter class value for each label. Also it shows a line for the mean IR inter class value and other line for the imbalance limit (Figure 1.21). This limit is fixed to 1.5, and each label which IR is over this limit, is considered to be imbalanced. The IR values in the table are shown in red if they are greater than 1.5. As in previous charts, if the mouse is over a bar, it shows a tooltip message with the label name and its IR inter class value.

- **IR intra class:** IR intra class shows the imbalance ratio inside a label. Left side table shows each label name and its IR intra class value. The labels in the table are sorted by descending IR value. The bar chart shows the IR intra class value for each label. Also it shows a line for the mean IR intra class value and other line for the imbalance limit (Figure 1.22). This limit is also fixed to 1.5. IR values in the table are shown in red if they are greater than 1.5. As in previous charts, if the mouse is over a bar, it shows a tooltip message with the label name and its IR intra class value.

- **IR per labelset:** IR per labelset is similar to IR inter class, but it shows the imbalance ratio between labelsets. Left side table shows each labelset and its IR value. The labelset in the table are sorted by descending IR value. The bar chart shows the IR value for each labelset, and also it shows a line for the mean IR value (Figure 1.23). IR values in the table are shown in red if they are greater than the imbalance limit fixed to 1.5. As in previous charts, if the mouse is over a bar, it shows a tooltip message with its IR value.

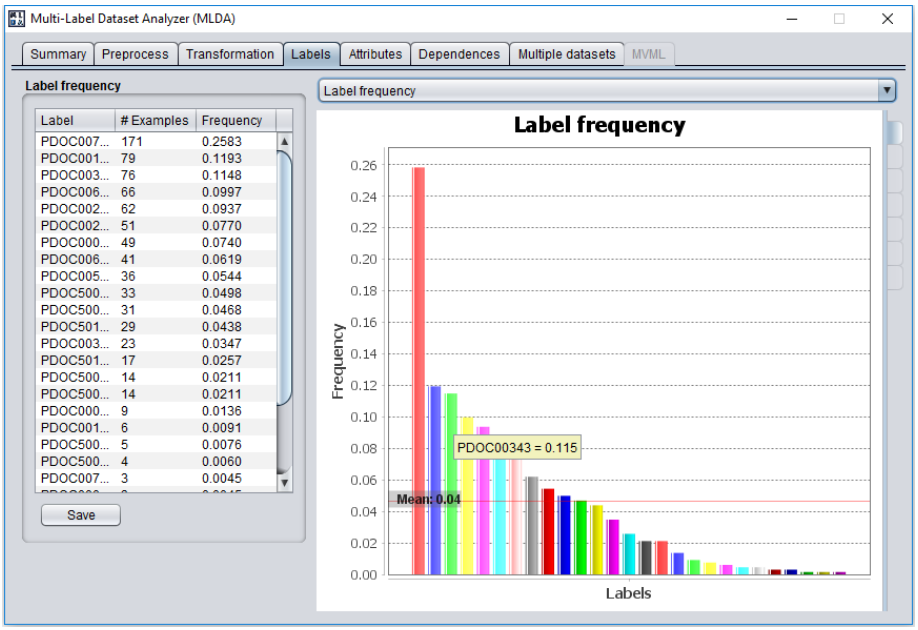

FIGURE 1.16: Example of label frequency chart

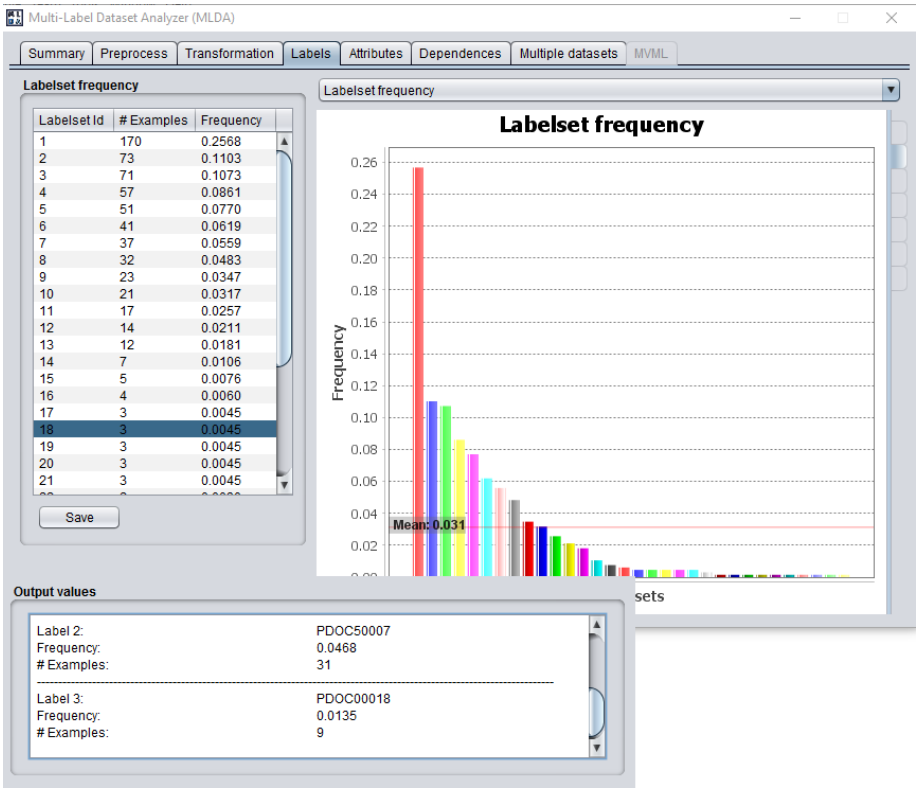

FIGURE 1.17: Example of labelset frequency chart

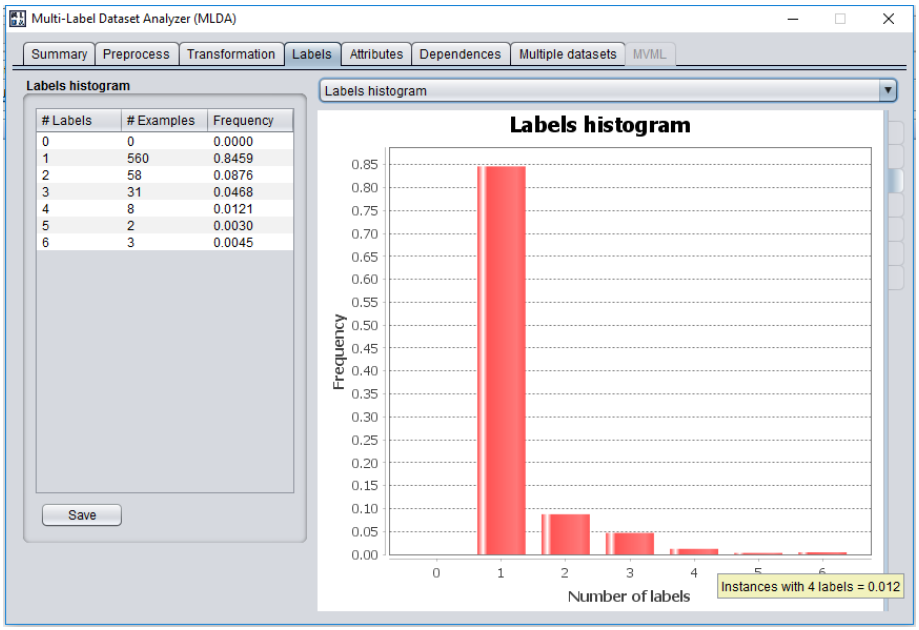

FIGURE 1.18: Example of labels histogram chart

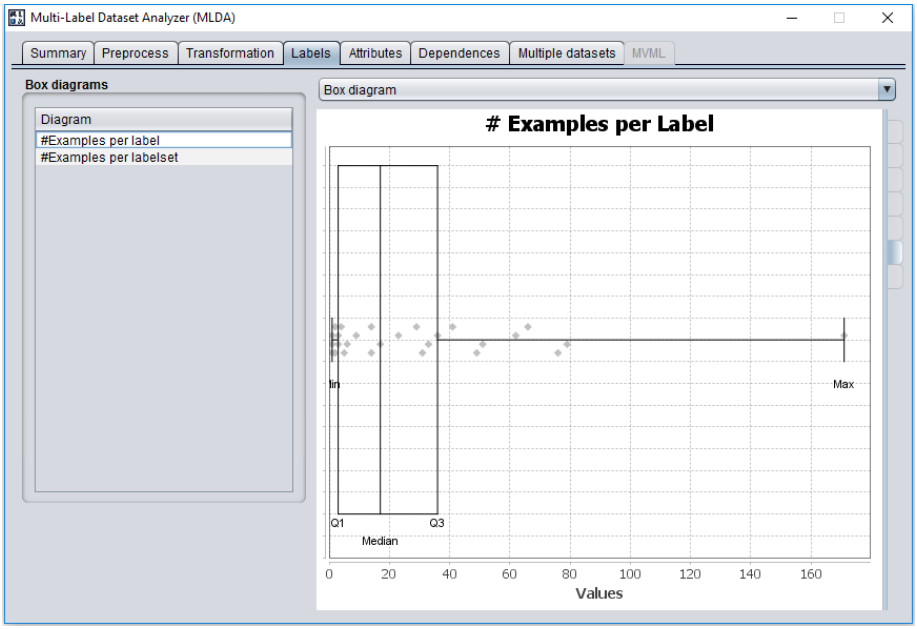

FIGURE 1.19: Example of boxplot for number of examples by label

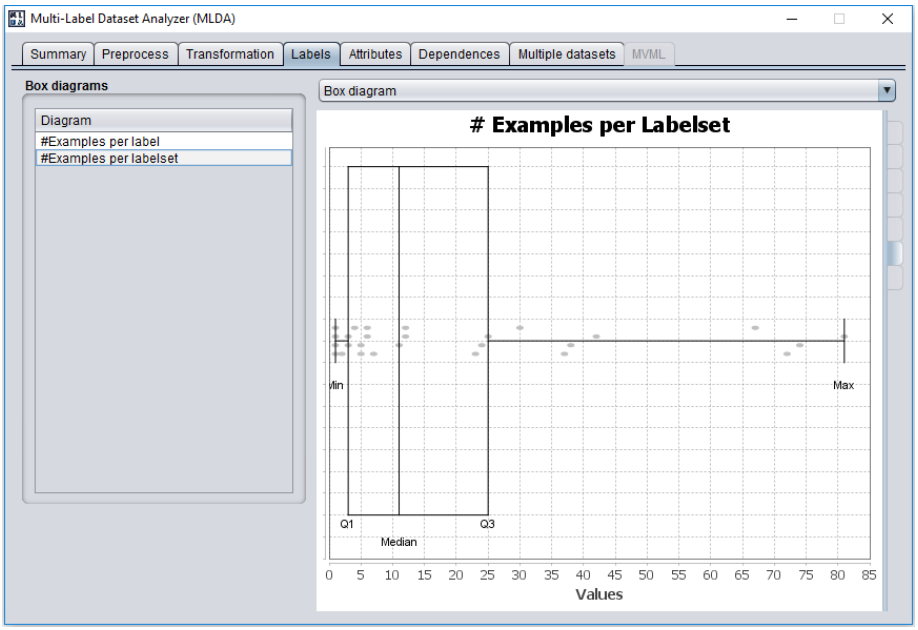

FIGURE 1.20: Example of boxplot for number of examples by labelset

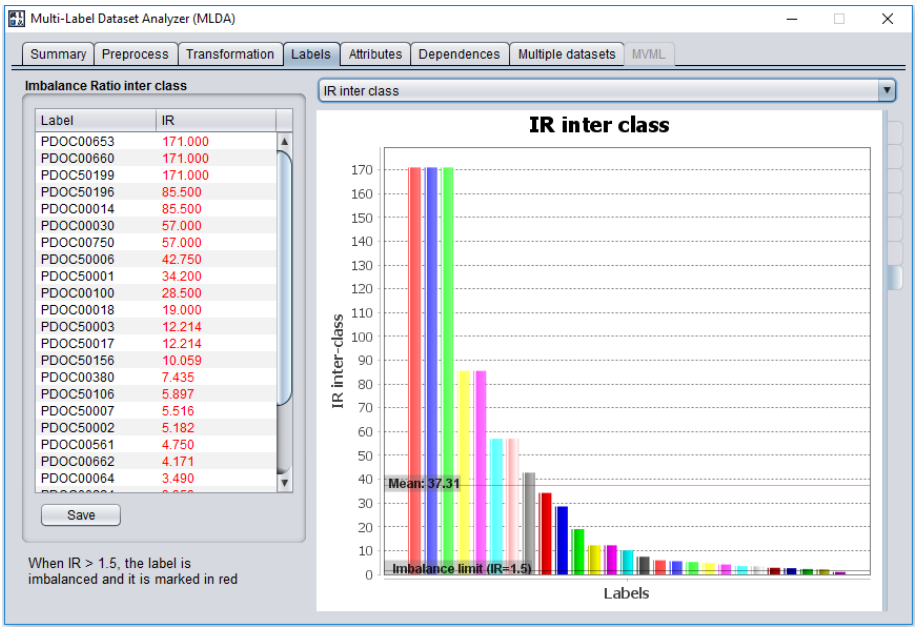

FIGURE 1.21: Example of IR inter class chart

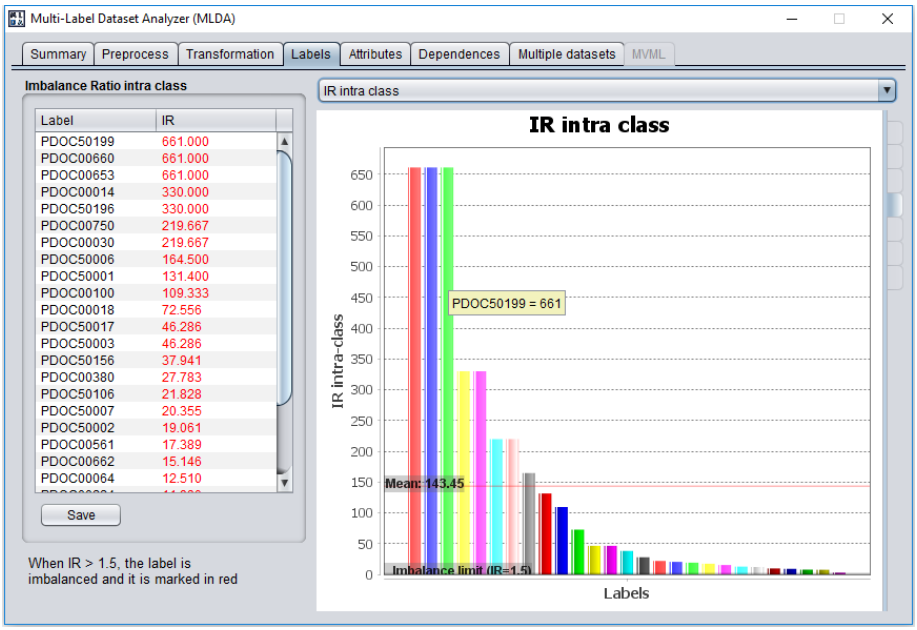

FIGURE 1.22: Example of IR intra class chart

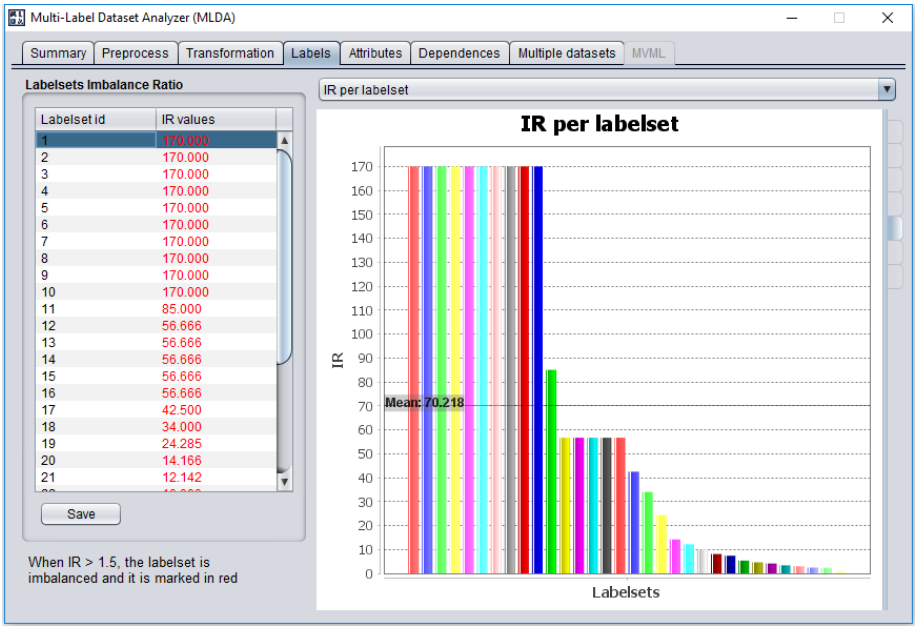

FIGURE 1.23: Example of IR per labelset chart

1.7 Attributes

The attributes tab show boxblot charts similar as in labels tab, but in this case, for each numeric attribute. As shown in Figure 1.24, on the left side there is a table listing all the numeric attributes, and on the right, there is a boxplot chart showing the values of the selected numeric attribute. Only selecting an attribute in the list, the boxplot is shown.

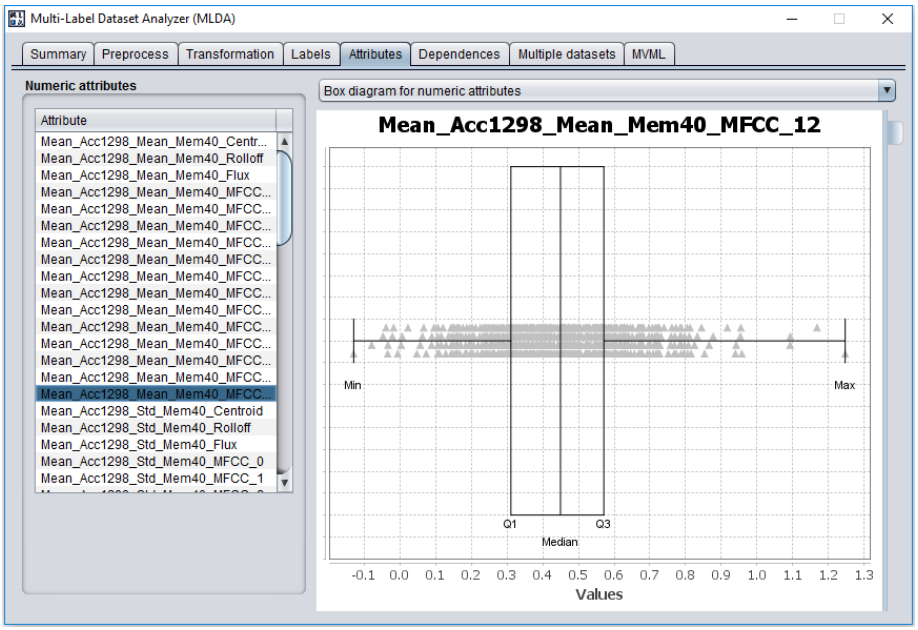

FIGURE 1.24: Example of numeric attributes boxplot

1.8 Dependences

The dependences tab shows information about the dependences or relationship between labels. This tab is divided in five sub-tabs, including three dependence types: chi and phi coefficients, co-occurrence and heatmap.

1.8.1 Chi and Phi coefficients

Chi coefficient (χ^2) identifies the unconditional relationship between label pairs, using Chi-square tests between each label pair. If Chi value is greater than 6.635, labels will be considered dependent at 99% confidence. Phi (ϕ) coefficient can be calculated from Chi as $\chi^2 = n \cdot \phi^2$, being n the number of instances. The *Chi & Phi coefficient* sub-tab shows a table with the Chi and Phi values between each pair of labels. As shown in Figure 1.25, Chi values are in white cells and Phi values in gray cells. Also, Chi values which are greater than 6.635 are marked in red, in order to show the dependence between these labels. If the mouse is over a cell, it shows a tooltip message with the label names to which correspond the value. Finally, the table can be saved in *.csv* format by clicking on the *Save* button. It will store a file with two tables, one for each coefficient.

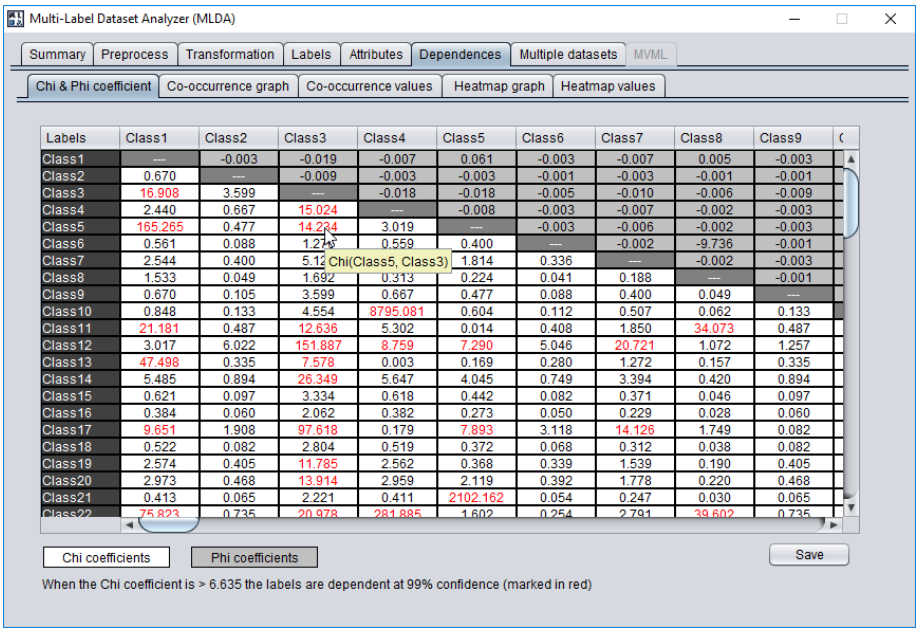

FIGURE 1.25: Chi and Phi coefficients table in Dependences tab

1.8.2 Co-occurrence

The co-occurrence measures how many times a label appears with another. The sub-tab *Co-occurrence values* show the values of co-occurrence between each pair of labels (Figure 1.26). This table is triangular, because co-occurrence between i and j is the same as between j and i. As before, stopping the mouse over a cell will show a tooltip message with the labels that correspond to this value. White cells with no value means no co-occurrence, the same as a 0 value. This table could be saved by the *Save* button.

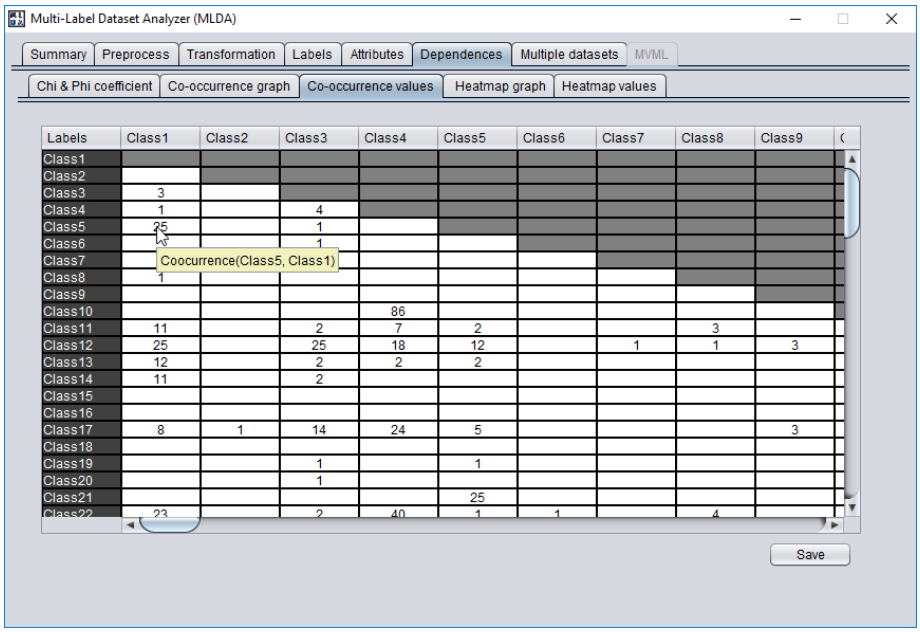

FIGURE 1.26: Co-occurrence values table in Dependences tab

The *Co-occurrence graph* sub-tab shows a graph representing the previous table. In this graph, the label thickness represents the *priori* probability of the current label $P(\lambda_i)$ and the link thickness represents the co-occurrence probability between two labels $P(\lambda_i \wedge \lambda_j)$. On the left side, it is shown a table with the label names and frequencies, which is ordered by descending frequency. Under the table, there are four different options to show the graph:

- Show selected: creates a graph with the selected labels in the table.
- Show most frequent: creates a graph with the n most frequent labels. The value of n must be less than the number of labels.
- Show most related: creates a graph with the n most related labels. For that, it selects the labels with higher value of co-occurrence between them. The value of n must be less than the number of labels. This graph is shown by default with the 10 most related labels.
- Show most frequent $\cup$ most related: creates a graph with the union of the n most frequent labels and the n most related labels. The number of labels in the graph may vary between n and $2n$, because some labels could be in both most frequent and most related sets. The value of n must be less than the number of labels.

Figure 1.27 shows an example of co-occurrence graph, selecting the 10 most related labels. As it can be seen, after clicking on the *Show most related* button, the table automatically selects the rows of the labels in the graph. The graph may be modified, moving the nodes or resizing them. Then, the *Save* button allows us to save the graph in *.png* format.

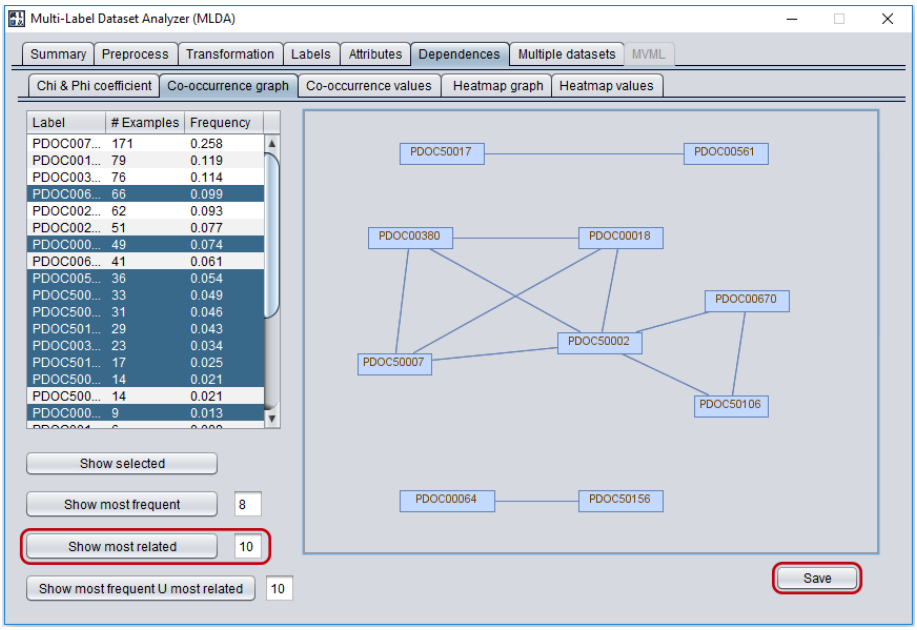

FIGURE 1.27: Co-occurrence graph for 10 most related labels in Dependences tab

1.8.3 Heatmap

Heatmap table in *Heatmap values* sub-tab also represents the relationship among labels. Each cell in the table or matrix $\mathbf{M}$ represents the conditional probability, being $\mathbf{M}_{ij} = P(\lambda_i|\lambda_j) = P(\lambda_i \wedge \lambda_j)/P(\lambda_j)$, where i is a row and j a column. The diagonal (gray background) represents the *priori* probabilities $\mathbf{M}_{jj} = P(\lambda_j)$. White cells with no value represents a probability of 0.0, and as in previous cases, if the mouse is over a cell, it shows a tooltip message with the labels that correspond to this value. This table could be saved by the *Save* button (Figure 1.28).

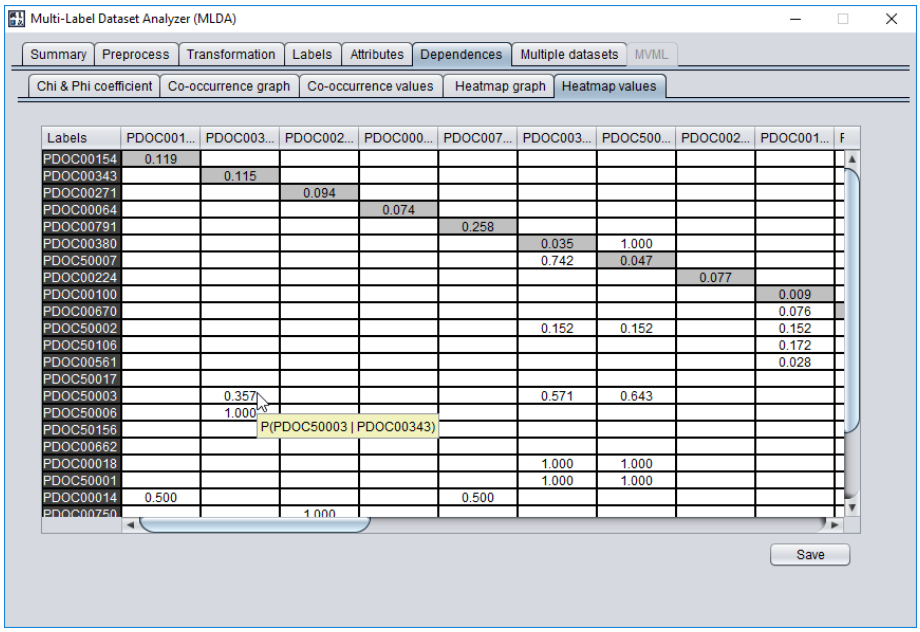

FIGURE 1.28: Heatmap values table in Dependences tab

The *Heatmap graph* sub-tab shows a graph representing the previous heatmap matrix. In this graph, probabilities are represented in grayscale, where black is a conditional probability of 0.0 and white is a conditional probability of 1.0, being the gray colors values between these. The diagonal running from the bottom left to the upper right corner represents the *prior* probabilities. On the left side, it is shown a table with the label names and frequencies, which is ordered by descending frequency. Under the table, there are four different options to show the graph:

- Show selected: creates a graph with the selected labels in the table.
- Show most frequent: creates a graph with the n most frequent labels. The value of n must be less than the number of labels.
- Show most related: creates a graph with the n most related labels. For that, it select the labels with higher value of co-occurrence between them. The value of n must be less than the number of labels.
- Show most frequent $\cup$ most related: creates a graph with the union of the n most frequent labels and the n most related labels. The number of labels in the graph may vary between n and $2n$, because some labels could be in both most frequent and most related sets. The value of n must be less than the number of labels.

Figure 1.29 shows an example of heatmap. It has been created with all the labels (default), in order to prove that this type of graph is better when the number of labels is higher. Finally, the *Save* button allows to save the graph in *.png* format.

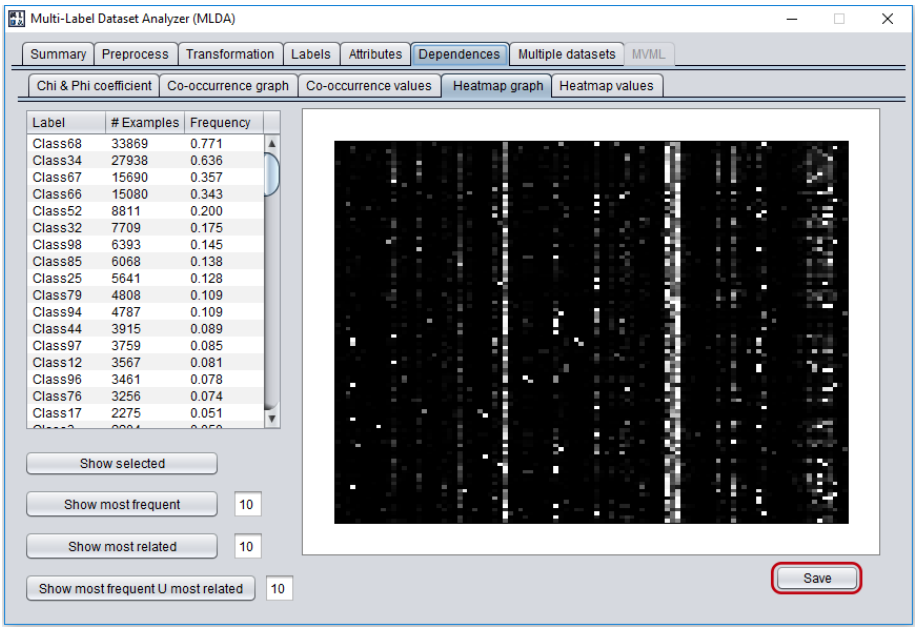

FIGURE 1.29: Heatmap graph in Dependences tab

1.9 Multiple datasets

The multiple datasets tab allows to load several datasets simultaneously, in order to calculate a set of metrics for all of them. As shown on Figure 1.30, on the left side there is an empty list and two buttons: *Add* and *Remove*. To add datasets to the list, we have to click on the *Add* button and select one or more datasets (Figure 1.31). We can add as many datasets as we want. Also, with the *Remove* button, the current selected dataset could be removed from the list.

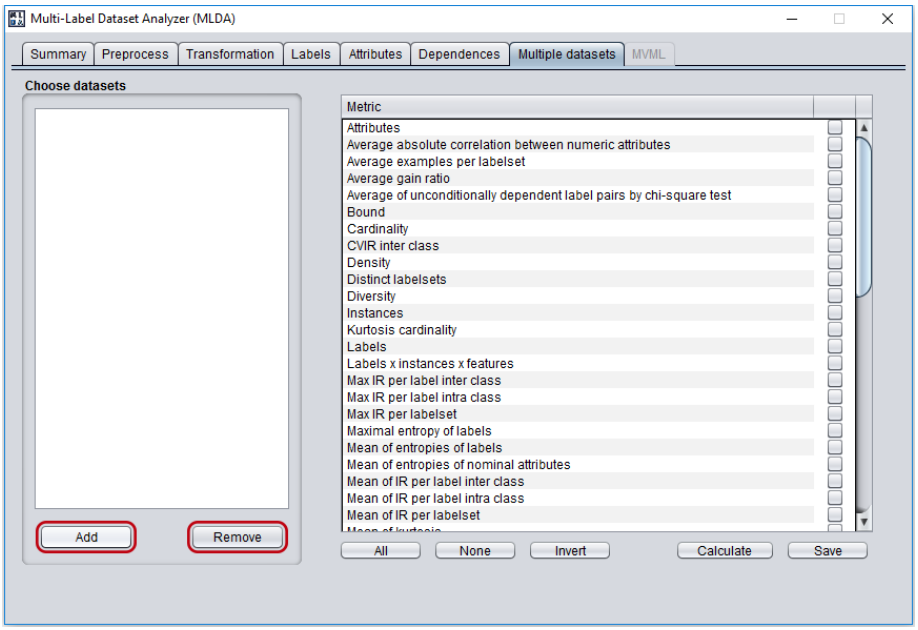

FIGURE 1.30: Add/remove datasets in multiple datasets tab

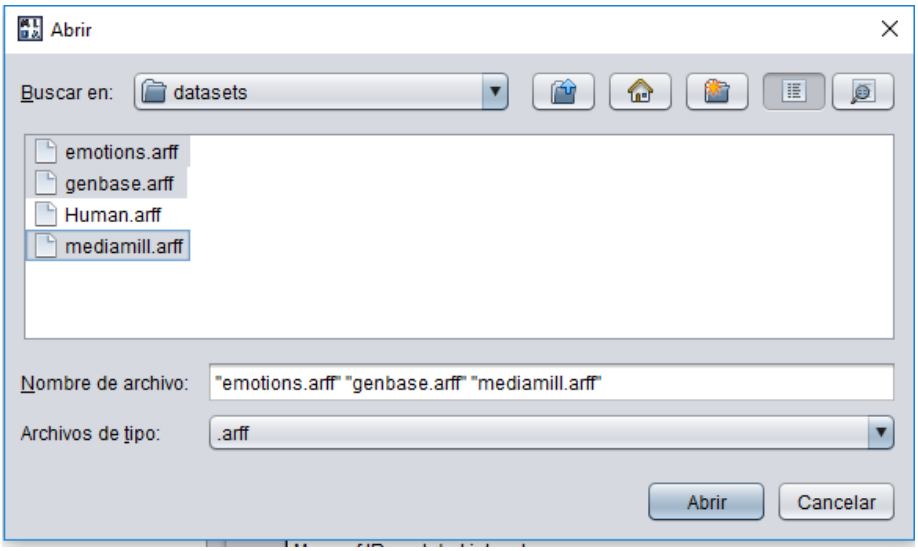

FIGURE 1.31: Adding datasets in multiple datasets tab

Once the datasets have been added, on the right side there is a table with metrics as in the summary tab. It can be selected a set of metrics, and be calculated clicking on the *Calculate* button. When they are calculated, they can be saved with the *Save* button in *.csv*, *.arff*, *.txt* and *.tex* formats (Figure 1.32).

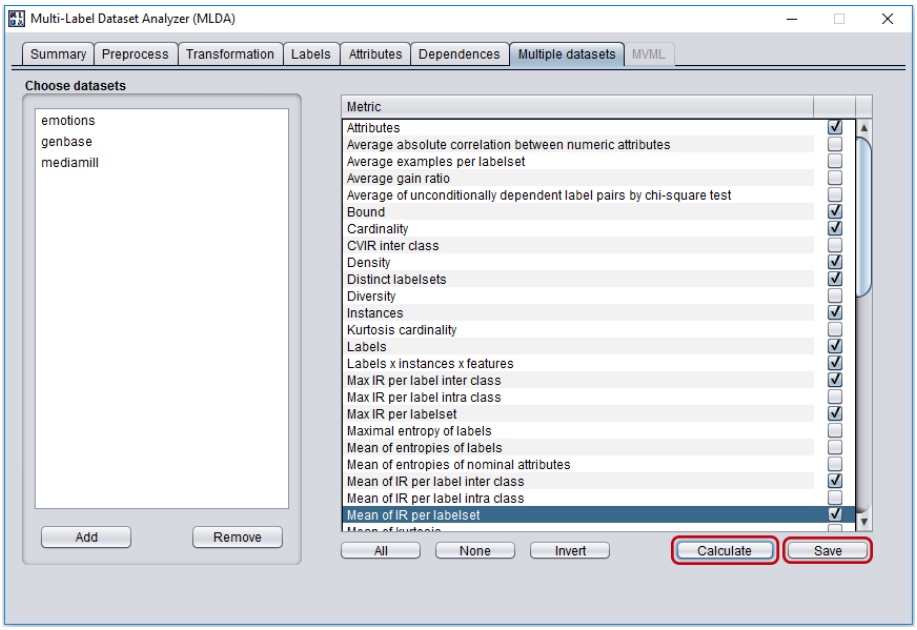

FIGURE 1.32: Calculating metrics for multiple datasets

1.10 MVML

The MVML tab gives an extra utility to datasets which are multi-view multi-label (MVML). Multi-view datasets are those which the input space is partitioned in some views. This tab is disabled if the dataset is not MVML.

The difference in format between a multi-label dataset and a MVML dataset is in its header. The *@relation* clause in the *.arff* file includes a `-V` parameter indicating the views. For example, this could be the header of a MVML *.arff* file: `@relation 'relationName -V:0-31!32-63!64-71'`. As it can be seen, the different views are separated by a `!` character and intervals of attribute indices are indicated as `a-b`, separating the interval ends with a hyphen, and being oth interval ends included in the view.

As shown on Figure 1.33, this tab at the top shows some basic metrics for multi-view datasets, as the number of views and the minimum, maximum and average number of attributes per view. On the bottom, the tool shows a table with the different views, giving for each one the number of attributes and some metrics like LxIxF, ratio of number of instances to the number of attributes and average gain ratio. Selecting views on the table, it can be seen at the right a list with the attribute names of the selected views. This tab allows to save both the table with the metrics for each view and the list of attributes for each selected view by clicking on the *Save table* button. Finally, the tool also allows to save a new multi-view multi-label dataset only with the selected views in the table. For that, we only have to select the format to save (Mulan or Meka) and click on *Save views* (Figure 1.33).

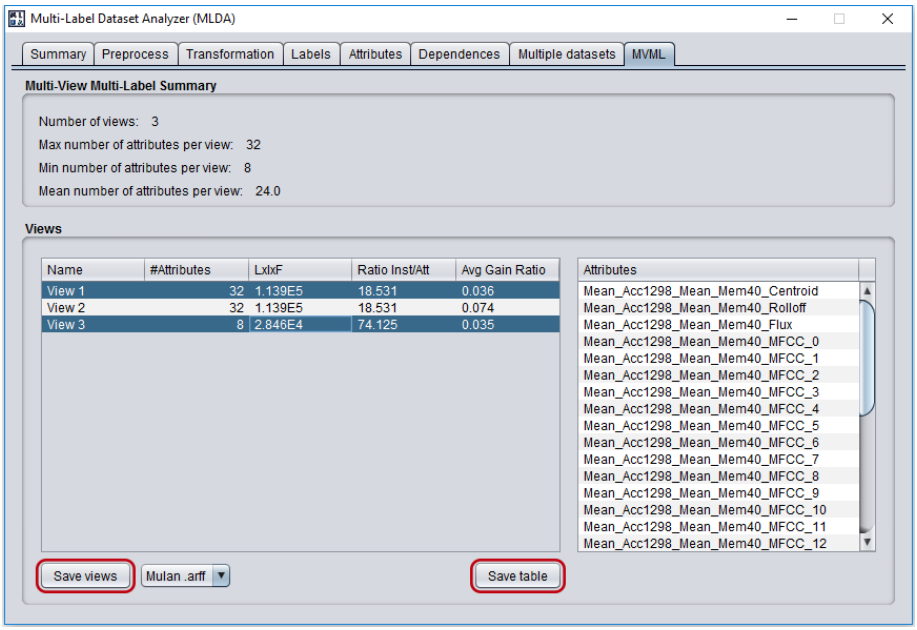

FIGURE 1.33: Saving views and table in MVML tab